

Summer2026

Cybersecurity Research Publication Bundle

COMPILED DATE:
2026-07-14 23:53:37

TOTAL PUBLICATIONS:
10 papers

COMPILED BY:
Gabriel Gonzalez Garcia
www.gabrielcybersecurity.com

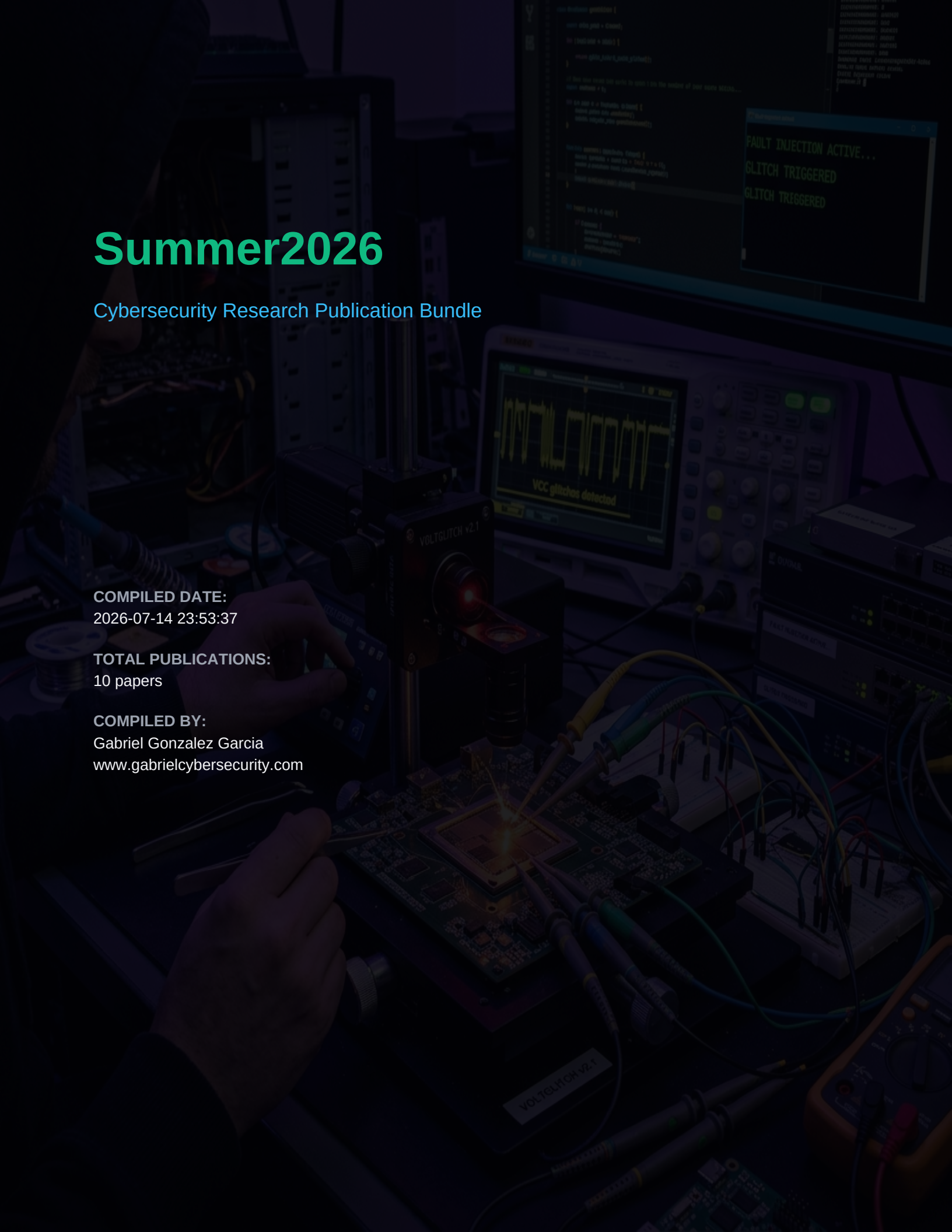


Table of Contents (Page 1 of 1)

Publication / Title	Conference Venue	Page Offset
Protocol Prying: Systematic Vulnerability Research in the Apple AirDrop and Android Quick Share Proximity Transfer Protocols <i>Authors: Arash Ale Ebrahim, Nils Ole Tippenhauer</i>	arXiv	3
Breaking 5G on The Lower Layer <i>Authors: Subangkar Karmaker Shanto, Imtiaz Karim, Elisa Bertino</i>	NDSS	17
SoK: The Constant Time Model <i>Authors: Billy Bob Brumley</i>	USENIX WOOT	26
Bit2Watt: A Cyber-Physical Vulnerability Exploiting GPU Workloads Across Power and Computing Infrastructures <i>Authors: Zhouhao Ji, Kaikai Pan, Wenyuan Xu</i>	CHES	42
Banshee: Target Switch Attacks on Gimbal-Stabilized Visual Tracking Systems via Acoustic Injection <i>Authors: Jiarui Li, Joseph Brewington, Qingzhao Zhang, Z. Morley Mao</i>	IEEE S&P;	70
Active Electromagnetic Side-Channel Analysis: Crossing Physical Security Boundaries through Impedance Variations <i>Authors: Taiki Kitazawa, Lennert Wouters, Benedikt Gierlichs, Daisuke Fujimoto, Ingrid Verbauwhede, Yuichi Hayashi</i>	CHES	90
Improving the Selection Rule of Correlation Attacks for Remote Power Analysis <i>Authors: Oriol Farràs, Vincent Grosso, Miquel Guiot, Carlos Andres Lara-Nino</i>	CHES	116
Turn Your Face Into An Attack Surface: Screen Attack Using Facial Reflections in Video Conferencing <i>Authors: Yong Huang, Yanzhao Lu, Mingyang Chen, En Zhang, Jiazi Li, Wanqing Tu</i>	USENIX Security	139
Racing against the clock: side-channel leakage from crossing asynchronous clock domains <i>Authors: Martin Thompson, David Oswald</i>	CHES	159
Secret Key Recovery of FALCON using Simple Power Analysis in Conditional Calculator <i>Authors: GyuSang Kim, JeongHwan Lee, Myeonghoon Lee, Seokhie Hong, HeeSeok Kim</i>	CHES	182



Protocol Prying: Systematic Vulnerability Research in the Apple AirDrop and Android Quick Share Proximity Transfer Protocols

Arash Ale Ebrahim

Nils Ole Tippenhauer

CISPA Helmholtz Center for Information Security

Abstract

Apple AirDrop and Google/Samsung Quick Share are proximity file-transfer protocols used by over five billion devices, yet their application-layer security properties remain largely unstudied because both stacks are proprietary and undocumented. Both protocols are reachable from wireless proximity without any prior pairing and process complex serialized content (binary plists, CPIO archives, Protocol Buffers, UKEY2 handshakes) inside privileged daemons, making them attractive zero-click targets across multiple operating systems. We perform the first cross-platform reverse engineering and protocol-aware fuzzing study of both stacks. We reconstruct AirDrop’s seven-layer state machine and DVZip adaptive compression from binary analysis, build AIRFUZZ, a protocol-aware fuzzer that mutates pre-compression representations, and complement it with targeted hand-written analyses of Samsung’s Quick Share service and Google’s Quick Share for Windows. We discover six vulnerabilities (V1-V6): three pre-authentication issues in macOS/iOS AirDrop (V1: Swift `fatalError` DoS in the HTTP path router; V2: unbounded XML plist recursion in `Foundation`, V3: NULL deref in `Network.framework`’s HTTP/1.1 parser), two protocol-layer flaws in Samsung Quick Share (V4: pre-authentication `OfflineFrame` dispatch, V5: D2D encryption bypass for three frame types), and a heap use-after-free in Google Quick Share for Windows (V6) for which Google awarded a bounty. We responsibly disclosed all findings, Apple, Samsung, and Google have acknowledged the reports. For transparency and reproducibility, the AirFuzz artifacts are publicly available on Zenodo: <https://zenodo.org/records/20442029>.

1 Introduction

Proximity-based file transfer is a fundamental feature of modern mobile and desktop operating systems. Apple’s AirDrop, introduced in 2011, and Google’s Nearby Share (now Quick Share), launched in 2020, collectively serve over five billion

active devices [2, 8]. These protocols enable ad hoc peer-to-peer file transfer between nearby devices over a combination of Bluetooth Low Energy (BLE) for discovery and Wi-Fi for data transfer, without requiring shared network infrastructure.

From a security perspective, proximity transfer protocols present a compelling attack surface [3, 10, 12, 16, 21]. The attacker need only be within wireless range (typically 10–30 m for AWDL/Wi-Fi Direct), and the initial protocol phases (service discovery and connection establishment) require *no authentication*. On Apple devices configured with “Everyone” visibility, the entire Discover and Ask phases are reachable without any user interaction or prior pairing. This creates a zero-click, pre-authentication attack surface that extends from the privileged `sharingd` daemon (which also manages AirPlay, Handoff, Universal Clipboard, and Continuity Camera) all the way down into the kernel: Beer’s iOS zero-click radio-proximity exploit [3] demonstrated remote code execution against the AWDL kernel driver itself, showing that the attack surface is not limited to user-space sharing code.

Despite the ubiquity and security relevance of these protocols, prior academic work has focused primarily on privacy aspects of AirDrop’s contact discovery mechanism [11, 15, 17, 18] and the underlying AWDL link layer. The foundational work of Stute et al. [18–20] reverse engineered AWDL itself, released the open-source OWL stack, and demonstrated link-layer denial-of-service and man-in-the-middle attacks; their results provide the wireless substrate on which our application-layer analysis builds. In parallel, Project Zero [3] weaponised the AWDL kernel attack surface against iOS. A comprehensive security analysis of the *application-layer* AirDrop protocol (binary property lists, DVZip compression, CPIO archive extraction, and the HTTP-layer state machine in `sharingd`) has not been published. Similarly, while Google’s Quick Share has received attention following the SafeBreach RCE chain in its Windows client [21], the Samsung Android implementation’s security properties remain largely unexamined.

In this paper, we bridge this gap with a systematic security analysis of both Apple AirDrop and Android Quick Share.

Our work makes the following contributions:

1. **Application-layer reverse engineering of AirDrop.** Building on the AWDL link-layer work of Stute et al. [19, 20], we reverse engineer the complete *application-layer* AirDrop stack on current macOS and iOS, documenting seven protocol layers from AWDL up through DVZip/CPIO archive extraction. We reconstruct the receiver-side state machine of `sharingd`, document the previously unpublished DVZip adaptive chunked compression format, and catalogue 40+ protocol commands. Detailed offsets, disassembly, and symbol lists for specific binary versions are released as artifacts rather than reproduced inline.
2. **Protocol-Aware Fuzzer.** We develop AIRFUZZ, a protocol-aware fuzzer with nine mutation layers (HTTP, binary plist, DVZip, CPIO, DER, havoc, memcorrupt, and field-level) and Frida-based coverage collection. We discuss why we did not use Frida Stalker, FPICKER [12, 14], or TINYINST/JACKALOPE [5–7] as our coverage engine; in particular, we reproduce the Stalker breakage on arm64e macOS/iOS (Section 4).
3. **AirDrop Vulnerabilities.** We discover two zero-click, and one post-accept pre-authentication vulnerabilities: (V1) a remote DoS via Swift `fatalError` in the HTTP path router, (V2) a stack overflow in Foundation’s XML property list parser and (V3) a NULL pointer dereference in `Network.framework`.
4. **Quick Share Vulnerabilities.** We discover three vulnerabilities in Quick Share: (V4) a pre-authentication frame processing bypass that allows unauthenticated protocol interaction before UKEY2 handshake completion, (V5) a device-to-device (D2D) encryption bypass where 3 out of 7 post-handshake frame types are processed without the mandatory `SecureMessage` encryption wrapper in Samsung’s Android implementation, and (V6) a critical use-after-free via an endpoint collision race condition in Google Quick Share for Windows which leads to remote-code-execution (version 1.2.2472.1).
5. **Cross-Platform Comparison.** We provide the first cross-platform security comparison of Apple and Google/Samsung proximity transfer implementations, identifying shared vulnerability patterns and divergent design decisions across six total vulnerabilities spanning macOS, iOS, Android, and Windows.

We responsibly disclosed all vulnerabilities to Apple Product Security, Samsung Mobile Security, and the Google Vulnerability Reward Program. Apple acknowledged V1–V3 and remediation is in progress; Samsung transferred V4–V5 to Google; Google acknowledged V6 and awarded a bounty for the Windows Quick Share use-after-free, with a CVE pending assignment. Section 7.5 provides the full disclosure timeline and vendor responses.

Vulnerability overview. Table 1 summarises the six vulnerabilities, their target component, the precondition required to reach them, and the observed impact. We use this table as a roadmap throughout the paper.

2 Background

2.1 Apple AirDrop

AirDrop, introduced in OS X Lion (2011) and iOS 7 (2013), enables peer-to-peer file transfer between Apple devices. It relies on two wireless technologies: *Bluetooth Low Energy* (BLE) for initial device discovery and wake-up, and *Apple Wireless Direct Link* (AWDL) for the actual data transfer. AWDL is a proprietary Wi-Fi-based protocol that creates an ad-hoc mesh network on the `awdl0` virtual interface using IPv6 link-local addresses [19].

The AirDrop application logic is implemented in `sharingd`, a privileged daemon that manages multiple Apple continuity services. On the receiver side, `sharingd` listens on TCP port 8770 over AWDL with TLS, serving an HTTP/1.1 API. The daemon is managed by `launchd` and automatically restarts on crash, but with exponentially increasing throttle delays after consecutive failures.

`sharingd` is a large, monolithic system daemon. On iOS 18.1/26.3 (ARM64e), The binary has *stripped* only one exported symbols (`_main` and `__mh_execute_header`) are visible via standard tools. It dynamically loads the private `Sharing.framework` (3.8 MB, 5,032 exported symbols), which implements the core AirDrop protocol logic, and links against approximately 30 system frameworks including Foundation, Security, Network, CoreFoundation, and `libswift_Concurrency`. The daemon manages not only AirDrop but also AirPlay, Handoff, Universal Clipboard, Continuity Camera, NameDrop, and SharePlay, meaning a crash in any subsystem disrupts all Continuity services simultaneously.

AirDrop supports three visibility modes: *Receiving Off* (no AWDL listener), *Contacts Only* (requires Apple ID certificate chain verification), and *Everyone for 10 Minutes* (accepts connections from any device). In “Everyone for 10 Minutes” mode, the complete HTTP API is reachable without any form of authentication. On iOS, Apple added a 10-minute auto-timeout that reverts “Everyone” to “Contacts Only”, but this mitigation does not protect against attacks during the window of exposure, and does not apply to macOS.

2.2 Android Quick Share

Quick Share (formerly Nearby Share) is Google’s cross-platform file-sharing feature for Android, Chrome OS, and Windows. On Samsung devices, it is deeply integrated as the default sharing mechanism, replacing Samsung’s earlier

Table 1: Overview of the six vulnerabilities (V1–V6). **Precond.:** “Everyone” = AirDrop in “Everyone for 10 minutes” visibility; “Visible” = Quick Share visible to nearby devices; “On-path” = same-LAN attacker (V5 only); “Race” = concurrent connect/disconnect (V6). **Impact:** DoS = denial of service; PoP = protocol-state manipulation; UAF = potentially exploitable use-after-free.

ID	Target	Component	Class	Precondition	Impact
V1	AirDrop	Sharing.framework path router	Fatal assert	Everyone	DoS
V2	AirDrop	Foundation XML plist	Stack overflow	Everyone	DoS
V3	AirDrop	Network.framework HTTP/1.1	NULL pointer dereference	Everyone	DoS
V4	QuickShare	Samsung GMS dispatcher	Authentication bypass	Visible	PoP
V5	QuickShare	Samsung D2D layer	Crypto bypass	On-path	PoP
V6	QuickShare	Win. endpoint mgmt	UAF race	Race	RCE

“Nearby Share” implementation. Quick Share uses a combination of BLE, Wi-Fi Direct, Wi-Fi LAN, and may also use WebRTC depending on the scenario.

Quick Share’s session traffic is largely structured as Protobuf-serialized messages (e.g., the `OfflineFrame` family), and is typically protected by an authenticated key-exchange followed by encrypted transport. In particular, reverse-engineering of the Quick Share for Windows implementation reports an initial UKEY2 handshake after which subsequent `OfflineFrame` packets are encrypted. [22]

At a high level, the connection lifecycle can be described in four phases: (1) *Discovery / advertising*, where nearby devices become visible to each other using short-range radios (commonly Bluetooth/BLE) and exchange ephemeral endpoint identifiers; (2) *Bandwidth upgrade negotiation*, where the peers negotiate a higher-throughput out-of-band channel (e.g., Wi-Fi Direct or a temporary Wi-Fi hotspot / access point) for bulk transfer; (3) *Introduction & consent*, where the sender transmits file-introduction metadata and the receiver is prompted to accept the transfer; and (4) *Payload transfer*, where the file bytes are transmitted over the negotiated high-bandwidth channel. [9, 21]

Samsung’s implementation extends the base Quick Share with additional features including link sharing with server-side URL preview generation, a feature not present in Google’s reference implementation. On the Samsung Galaxy S23 Ultra (our test device), Quick Share runs as a system-level service (`com.samsung.android.nearby.discovery`) with access to network I/O, file system operations, and WebView rendering for link previews.

Google’s Quick Share for Windows implementation handles endpoint management, the lifecycle of connected peers through a state machine that allocates and deallocates endpoint objects as devices connect and disconnect. During concurrent transfers involving multiple devices, these endpoint objects can be subject to race conditions if connection and disconnection events arrive simultaneously, a property we exploit in v6 (Section 6.4).

2.3 Threat Model

We consider an attacker in wireless proximity to the target device (within AWDL/Wi-Fi Direct range, typically 10-30 meters). The attacker has a commodity laptop with Wi-Fi capability and requires no prior relationship with the target (no pairing, no contact exchange, no shared network). The target device has AirDrop or Quick Share enabled in a mode that accepts connections (“Everyone” for AirDrop; default visibility for Quick Share). We focus on *zero-click* attacks that require no user interaction on the target device. For AirDrop, this means exploiting the pre-authentication phases (Discover, Ask routing) before the user acceptance prompt is displayed.

3 AirDrop Protocol Reverse Engineering

We reverse engineered the AirDrop protocol by static and dynamic analysis of `sharingd` and the private `Sharing.framework` on macOS 15.7.4 and on the latest macOS 26.3 and iOS 18.1/26.3 releases. Throughout the paper we describe findings at the protocol level; exact symbol offsets and per-build disassembly listings are released as artifacts.

3.1 Protocol Stack Overview

Table 2 summarizes the seven-layer AirDrop protocol stack we identified. Each layer introduces distinct serialization formats and potential vulnerability surfaces.

3.2 Discovery and Connection Establishment

AirDrop device discovery uses DNS-SD (mDNS) over AWDL, advertising service type `_airdrop._tcp.local.` with ~25 TXT keys; the `flags` field is a bitmask (e.g., `0x3FB`) encoding DVZip, pipelined uploads, mixed content types, URL sharing, contact resolution, and BLE support. Upon discovery, the sender opens a TLS connection to port 8770 on the receiver’s IPv6 link-local address over AWDL, using self-signed certificates with no hostname verification and no client certificate. *Any device within AWDL range can establish a*

Table 2: AirDrop protocol stack. Each layer adds serialization complexity and attack surface.

#	Layer	Technology	Key Attack Surface
7	Archive	CPIO newc / BOM (Bill of Materials)	Path traversal, symbolic-link attacks
6	Compression	DVZip / gzip	Chunk-length validation, decompression bombs
5	Encoding	Binary property list (bplist)	Type confusion, unbounded recursion
4	Transport	HTTP/1.1	Path routing, request smuggling
3	Security	TLS 1.2 / 1.3	Self-signed certificates, no client authentication
2	Network	IPv6 link-local	fe80:: addresses on the awdl0 interface
1	Link	AWDL / BLE	Wireless-proximity discovery

TLS connection and interact with the HTTP API regardless of the receiver’s visibility mode.

3.3 HTTP API and State Machine

The AirDrop HTTP API exposes seven POST endpoints:

```
/Discover, /Hello, /Ask, /Upload,
/Exchange, /SharedIdentity, /Error
```

The receiver-side state machine is implemented in `SDAirDropConnection`. Each TLS connection has three *request slots*—one each for *Discover*, *Ask*, and *Upload*—plus control flags and a `dispatch_semaphore_t` that blocks the request handler while the user-consent prompt is on screen. A slot is `NULL` until the corresponding request arrives. *Discover* is accepted only when all three slots are `NULL` and returns a plist `StatusCode` of 100 (Everyone), 200 (contacts matched), or 401 (Contacts-only, no match). *Ask* also requires all slots `NULL` (a prior *Discover* slot is cleared first), stores the payload, displays the consent prompt, and blocks on the semaphore until accept/decline or a 15 s timeout. *Upload* requires the Upload slot `NULL` and normally follows a successful *Ask* on the *same* TLS connection; two TXT-record flags from discovery are consulted—`SupportsUPP` (bit 0xb) permits Upload *without* a prior *Ask*, and `SupportsDVZip` enables adaptive compression. Upload data *streams into memory while the Ask prompt is displayed*; the semaphore gates processing, not reception.

A normal AirDrop transfer is therefore: *Discover* (optional) → *Ask* (same TLS connection) → *Upload* (same TLS connection). The protocol requires *Ask* and *Upload* to share the same TLS session; *Upload* on a separate connection is rejected. Figure 1 shows this exchange end-to-end.

3.3.1 Path Routing Vulnerability Surface

The HTTP path router lives in `Sharing.framework` (file `Network+SFAirDropMessage.swift` in Apple’s source layout) and dispatches incoming request paths via a Swift switch statement over an `SFAirDropMessagePath` enum. The default case calls `fatalError("Unhandled Path \ (path)")`, which the Swift compiler lowers to a trapping

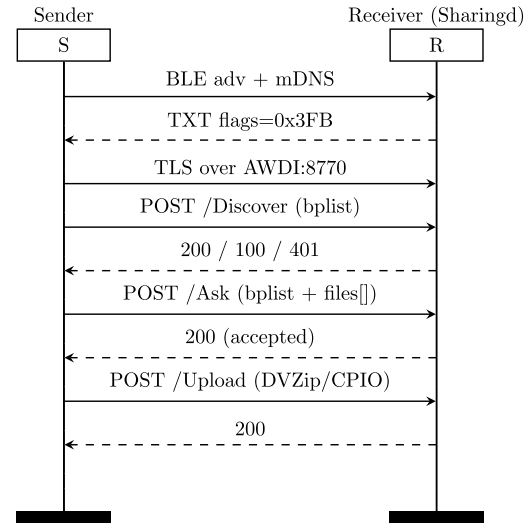


Figure 1: End-to-end AirDrop exchange. *Discover* and *Ask* are reachable pre-consent on the same TLS connection. The receiver’s flags TXT record (e.g., 0x3FB, with bit 0xb = `SupportsUPP`) is consulted on *Upload* to decide whether pipelined uploads are allowed.

instruction that aborts the process on any architecture. Any unauthenticated client sending an HTTP request to an unrecognised path triggers this trap; we exploit it as V1 (Section 5.1).

3.4 Serialization Formats

Binary Property Lists. *Discover* and *Ask* request/response bodies are serialized as Apple binary property lists (bplist00), parsed by `CFPropertyListCreateWithData` and type-checked as `CFDictionary`.

DVZip Adaptive Compression. Upload bodies use DVZip (Content-Type: `application/x-dvzip`), an Apple-proprietary adaptive chunked format we identified through binary analysis: 4-byte big-endian chunk-length headers wrap individual zlib (deflate) blocks, with adaptive

switching between gzip and DVZip mid-stream depending on compressibility, and graceful fallback to plain gzip when the receiver does not advertise SupportsDVZip.

CPIO Archives. The compressed payload contains a CPIO “newc” archive (magic 070701) terminated by TRAILER!!!. After decompression, sharingd extracts files via Apple’s BOM (Bill of Materials) library using BOMCopierCopyWithOptions, with CPIO filenames passed to BOMFSObjectNewFromPath—the surface for path-traversal, symlink, and filename-based attacks.

3.5 Authentication and Identity

3.5.1 Protocol Keys (summary).

Through string extraction and cross-referencing with disassembly we identified 40+ protocol commands in AirDrop’s binary-property-list messages, grouped into Discover keys (e.g., SenderComputerName, SenderModelName, SenderRecordData—the DER-encoded Apple ID certificate chain, SenderID, TransferVersion), Ask keys (Files, FileType, FileName, FileBomPath, ConvertMediaFormats), and capability flags (SupportsDVZip, SupportsUPP, SupportsMixedTypes, SupportsURLSharing, SupportsStreamZip). The complete key list with descriptions is released as an artifact. The three security-sensitive keys are SenderRecordData (parses untrusted DER), FileBomPath (path-traversal surface, mitigated by BOMCopierCopyWithOptions sandboxing), and TransferVersion (gates DVZip and pipelined-upload behaviour).

3.5.2 AirDrop Authentication Model

AirDrop’s authentication is layered. *TLS* uses self-signed certificates and provides encryption only, not peer verification. *Contact resolution* parses the DER-encoded Apple ID certificate chain in SenderRecordData via SFAAppleIDVerifyCertificateChain to decide whether the sender is a known contact. *Auto-accept* is gated by the canAutoAccept predicate, whose senderIsMe flag is set only when SFAAppleIDVerifyCertificateChain succeeds against the local user’s iCloud-bound certificate and the resulting Apple ID matches the receiver’s; senderIsMe is *not* a plist field controlled by the sender. *User consent* is enforced by the Ask-phase UI prompt gated on a dispatch_semaphore; all file delivery requires either user acceptance or same-Apple-ID auto-accept. We tested 15+ bypass variants that manipulate plist-level fields purporting to set senderIsMe or replace SenderRecordData; all were rejected (Section 3.5.3).

3.5.3 Zero-Click Bypass Analysis

We systematically tested ten strategies to bypass the user-acceptance requirement—direct Upload skipping Ask, HTTP-pipelined atomic Ask+Upload, plist-field manipulation of SenderIsMe, forged SenderRecordData, fifteen further Ask-field manipulations, notification flooding, repurposing the Exchange/NameDrop TransferType, Upload on a separate TLS session, chunked-Upload interleaving, and forged TransferID. *All ten* were rejected or timed out on iOS 18.1/26.3 and macOS 15.7.4/26.3: the auto-accept decision is gated behind SDAppleIDAuthenticateCertificateChainSync, which verifies the sender’s Apple ID against a hardware-bound chain that cannot be forged without access to the sender’s iCloud credentials. V1 (Section 5.1) instead abuses the unrecognised-path trap before any state transition occurs.

4 Fuzzing Methodology

4.1 Challenges

Fuzzing AirDrop presents four unique challenges compared to traditional network protocol fuzzing:

1. **AWDL transport:** The target is only reachable over the awdl0 interface via IPv6 link-local addresses, requiring active AWDL participation (BLE advertising + Wi-Fi channel synchronization).
2. **TLS requirement:** All HTTP communication is over TLS with self-signed certificates, requiring a proper TLS handshake for each connection.
3. **Multi-layer encoding:** A valid Upload requires DVZip-compressed CPIO archives containing binary plist metadata-mutations at any single layer are likely rejected by parsers at other layers.
4. **State dependency:** Upload requires a prior Ask on the *same* TLS connection, and the Ask phase blocks on user interaction (15-second timeout without our auto-accept hook).

4.2 AIRFUZZ Architecture

We developed AIRFUZZ, a 12,300-line Python protocol-aware fuzzer. Figure 2 shows its architecture.

Frida Auto-Accept Daemon. To remove the user-interaction dependency, we deploy a Frida-based instrumentation agent on the target machine. The agent hooks five functions in sharingd three ObjC auto-accept predicates, the native canAutoAccept at offset +0x041f10, and the UNUserNotificationCenter notification suppressor, forcing acceptance while leaving all other code paths unmodified.

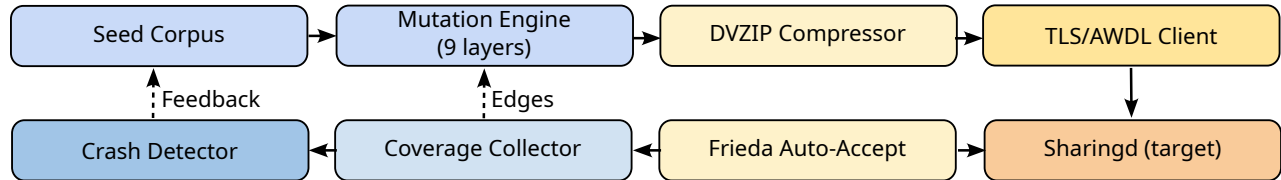


Figure 2: AIRFUZZ architecture. Mutations are applied to seed inputs *before* DVZip compression; Frida hooks bypass the user-acceptance prompt; coverage feedback guides mutation selection.

This enables fully automated fuzzing without user interaction. The daemon monitors the target process and automatically reattaches after crashes with configurable recovery using `launchctl bootout/bootstrap` to reset `launchd` throttling.

Mutation Engine. AIRFUZZ implements nine mutation layers spanning the protocol stack: HTTP (24 strategies: path corruption, header smuggling, content-length mismatch, chunked-encoding abuse, method fuzzing), binary plist (12 strategies: type confusion, key deletion, oversized strings, nested-container injection, cross-type substitution), DVZip (chunk-length corruption, truncated/oversized chunks, adaptive-mode confusion), CPIO (header corruption, path traversal, symlink/device-node injection, deep path nesting), DER (ASN.1 length overflows, zero-key injection), havoc (random bit flips and byte insert/delete), memcorrupt (25 strategies: boundary values, arithmetic, dictionary ops, block shuffling), and field-level mutations that target specific protocol keys while preserving overall structure.

Pre-Compression Mutation (Key Insight). Our initial fuzzer (AF1–AF3) mutated the *compressed* DVZip output, resulting in near-zero (<1%) server acceptance because even single-byte mutations in the compressed stream corrupt the zlib checksum. The critical design insight in v4 was to mutate the raw CPIO archive *before* DVZip compression. This ensures that the compressed output has valid zlib framing while carrying mutated content, increasing the acceptance rate to over 90% and enabling meaningful exploration of the archive extraction code paths.

4.3 Fuzzer Evolution

AIRFUZZ evolved through four versions (AF1–AF4) totalling 1.8K→12.3K LoC. AF1 (HTTP + plist only) and AF2 (added DVZip/CPIO mutations) had <5% server acceptance but AF2 still surfaced V1 within the first 200 test cases. AF3 added DER/havoc mutations, Frida auto-accept, and coverage-guided feedback, discovering V2 and V3. AF4’s key change was *pre-compression* mutation plus memcorrupt and field-level strategies, raising acceptance above 90% and expanding edge coverage from 120 to 950 edges into `Foundation` and `Network.framework`.

4.4 Seed Corpus and Coverage

We constructed the seed corpus by capturing legitimate AirDrop transfers between two Apple devices, covering six scenarios: single image, multi-file, large file (500 MB), URL sharing, contact card (VCF), and directory transfer. Using Wireshark with TLS key logging, we extracted 162 seed templates in total: 70 for Discover, 45 Ask, plus 42 complete Upload payloads. We additionally generated synthetic seeds with boundary values and maximum-length strings for 40+ protocol commands. Coverage is collected via Frida Interceptor hooks on ~100 manually selected functions across `sharingd`, `Sharing.framework`, and system frameworks, achieving approximately 950 unique edges across 250K+ executions.

Choice of coverage instrumentation. Frida’s `Stalker` engine in principle yields basic-block and edge coverage without manual hook selection, and `FPICKER` [14] (an extension of `TOOTHPICKER` [12]) wraps it for AFL++-style feedback. We initially attempted this route and found that on our test hosts (**macOS 26.3 / Apple Silicon ARM64e** and **iOS 26.3 / arm64e**) `Stalker` reliably crashes the *target* once it begins instrumenting an arm64e thread that executes inside system frameworks: the target is killed with `SIGKILL` by the kernel, consistent with arm64e codesigning / pointer-authentication enforcement against `Stalker`’s in-process code rewriting. The Frida agent itself loads and runs fine, baseline Interceptor hooks are stable, and `Stalker.follow()` on Frida’s own injected thread also works; the failure is specific to following arm64e threads that re-enter signed system code, which is exactly where `sharingd` spends most of its time. We reproduced this in isolation with `Stalker.follow()` on the main thread of `/usr/bin/osascript` on macOS 26.3, which kills the target on script load. For the same reason, `FPICKER`’s LibAFL/Rust mode hits identical mitigations and `TINYINST/JACKALOPE` [5, 6] (which performs out-of-process binary rewriting and was used by Project Zero against `coreaudiod` [7]) would require porting to AWDL-bound `sharingd` and re-implementing the seven-layer AirDrop encoding stack as a `TinyInst` harness, which we leave to future work. We therefore fall back to a smaller, hand-picked set of Interceptor hooks placed at the input boundary of each protocol layer (HTTP path router, plist scanner, DVZip header, CPIO header, DER/SecureMessage parser). This loses

fine-grained edge coverage but is stable on arm64e, survives launchd respawns, and is sufficient to discover V1–V3 and to guide AF4’s pre-compression mutation strategy.

4.5 Crash Detection and Recovery

Crash detection is performed via SSH-based health monitoring of the target machine. When `sharingd` crashes, `launchd` restarts it automatically. Our fuzzer detects the restart (new PID) and automatically:

1. Reattaches the Frida auto-accept daemon to the new process.
2. Copies the triggering input to a crash corpus.
3. If `launchd` throttles restarts (consecutive crash count ≥ 4 , resulting in exponentially increasing delays), uses `launchctl bootout/bootstrap` to reset the throttle state.

Crash reports (`.ips` files) are automatically collected from the target’s `DiagnosticReports` directory for post-mortem analysis.

5 AirDrop Vulnerabilities

We discovered two zero-click and one post-accept pre-authentication vulnerabilities in AirDrop, summarised alongside the Quick Share findings in Table 1 (Section 1). We detail each AirDrop vulnerability below.

5.1 V1: Remote DoS via Unhandled HTTP Path

Root Cause. The HTTP path router in `Sharing.framework` (see `Network+SFAirDropMessage.swift` in Apple’s source layout) maps incoming request paths to `SFAirDropMessagePath` enum values via a Swift `switch` statement. Any path that does not match a defined enum value falls into the default case, which calls `fatalError("Unhandled Path \(path)")`. Unlike Swift’s `assertionFailure` (compiled out in release builds), `fatalError` is unconditionally fatal, so the process traps and is terminated. The vulnerability is therefore a simple, architecture-independent crash: any unauthenticated client that POSTs to an unrecognised URI path forces `sharingd` to abort. The impact is bounded to a denial of service (no attacker-controlled register or memory state at the crash point) and does not require deep protocol knowledge to reach.

Trigger. A single HTTP POST request with any unrecognized URI path and a non-empty body, sent over TLS to port 8770 on the AWDL interface (e.g., `POST /X HTTP/1.1\r\nContent-Length: 1\r\n\r\nA`).

Impact. The crash kills `sharingd`, immediately disabling AirDrop, AirPlay, Handoff, Universal Clipboard, and Continuity Camera. While `launchd` restarts the daemon, an attacker can send crash packets in a loop (approximately one every two seconds) to create a *persistent* denial of service. During active loop attack, 0 out of 6 legitimate AirDrop connection attempts succeeded, all 10 attempts succeeded after stopping the attack.

Affected Versions. Confirmed on macOS 15.7.3, iOS 18.x, macOS 26.3, and iOS 26.3. The vulnerability is *not present* on iOS 16.7.11, which returns HTTP 400 for unknown paths, suggesting the `fatalError` was introduced during a Swift port of the path router.

5.2 V2: Foundation XML Plist Stack Overflow

Root Cause. The `XMLPlistScanner.scanDict()` of `Foundation.framework` uses recursive descent parsing with *no depth limit*. Each recursion level consumes approximately 2,960 bytes of stack (0x60 bytes for register saves plus 0xb30 bytes for locals). On cooperative thread pools (stack size ~ 512 KB), approximately 180–200 levels of nesting exhaust the stack. The next recursive call attempts to write its prologue (saved registers and locals) into the unmapped *stack guard page*, raising a memory *write* fault that the kernel reports as `SIGBUS / KERN_PROTECTION_FAILURE`. The fault address is fully determined by the recursion (the bottom of the cooperative-pool stack) and is not attacker-controlled; no useful register or memory write primitive is exposed.

Trigger. An HTTP POST to `/Discover` with a ~ 6 KB XML property list containing ~ 200 nested `<dict>` elements (a `<dict><key>a</key>... cascade` terminated by matching close tags).

Impact. This vulnerability affects not just `sharingd`, but `Foundation.framework` itself. Any Apple application that deserializes untrusted XML property lists via `PropertyListDecoder` or `PropertyListSerialization` is vulnerable. The attack surface spans macOS, iOS, watchOS, tvOS, and visionOS. We reproduced the crash on all three architectures Apple currently ships AirDrop on: **Apple Silicon Mac** (macOS 26.3, ARM64e), **Intel Mac** (macOS 15.7.4, x86_64), and **iPhone with PAC** (iOS 26.3, arm64e). The crash threshold varies slightly with cooperative-thread-pool stack allocation (approximately 200 levels on macOS and 180 on iOS); PAC on arm64e does not change the outcome because the fault originates from the recursive prologue itself rather than a corrupted return pointer. Symbolicated crash logs and reproduction artifacts for each platform are included in the artifact under `crash-notes/report-3-sigbus-xml.plist-recursion/`.

5.3 V3: Network.framework NULL Pointer Dereference

Root Cause. Within `Network.framework`, the `HTTP/1.1` connection-setup path `nw_protocol_http1_connect()` contains a NULL-pointer dereference reachable when the framer is forced into an inconsistent state. The crash happens at a fixed offset into a connection structure whose backing object is `nil`, so the faulting address sits in the unmapped low page; the precise struct-member offset shifts between builds but the bug is identical across them.

Trigger. Crafted HTTP requests with one of: (a) Transfer-Encoding: chunked with negative chunk sizes (-1), (b) duplicate Content-Length headers with conflicting values, or (c) Content-Length vastly exceeding the actual body size.

Impact. SIGSEGV (`KERN_INVALID_ADDRESS`) on standard systems where the zero page is unmapped. This vulnerability is in `Network.framework`'s HTTP/1.1 parser, potentially affecting other Apple applications that process untrusted HTTP traffic through the system networking stack, though the specific trigger conditions (conflicting framing under concurrent load) may limit practical reachability beyond `sharingd`.

6 Quick Share Vulnerabilities

In addition to AirDrop, we analyzed Quick Share implementations across platforms. On a Samsung Galaxy S23 Ultra running Android 16 (firmware S918BXXS8EZA1, Quick Share v13.8.01.11, GMS 26.05.34), we discovered two vulnerabilities (V4, V5) in the Nearby Connections protocol layer underlying Samsung's Quick Share service. We additionally analyzed Google Quick Share for Windows (version 1.0.2472.1), discovering a critical use-after-free (v6) in the endpoint management subsystem.

6.1 Quick Share Protocol Analysis

To understand the attack surface, we reverse engineered the Quick Share on Samsung Android by combining protobuf definition extraction from the GMS APK, traffic captures between two Samsung devices, and systematic probing of the TCP listener.

Wire Format. Quick Share communicates over TCP using a simple framing protocol: each message is preceded by a 4-byte big-endian length prefix, followed by a Protocol Buffer payload. Application-level messages use the `OfflineFrame` protobuf, which contains a `V1Frame` with a `type` enum field that determines the message category. We identified

seven frame types in active use: `CONNECTION_REQUEST` (1), `CONNECTION_RESPONSE` (2), `PAYLOAD_TRANSFER` (3), `BANDWIDTH_UPGRADE` (4), `KEEP_ALIVE` (5), `DISCONNECTION` (6), and `PAIRED_KEY_ENCRYPTION` (7). A `PAIRED_KEY_RESULT` (8) type is also defined in the binary.

UKEY2 Handshake. After the initial `CONNECTION_REQUEST/CONNECTION_RESPONSE` exchange registers the endpoint, UKEY2 [21] performs mutual authentication via an Elliptic-Curve Diffie-Hellman (ECDH) key agreement on NIST P-256. The handshake proceeds in three messages using the same 4-byte length-prefix framing but with raw `Ukey2Message` protobufs (not wrapped in `OfflineFrame`): *ClientInit* (client → server, containing a SHA-512 commitment hash of the *ClientFinished* message, a random nonce, and the selected cipher suite), *ServerInit* (server → client, containing the server's ephemeral P-256 public key, a random nonce, and the selected cipher suite), and *ClientFinished* (client → server, revealing the client's ephemeral P-256 public key). This commit-then-reveal scheme prevents an attacker from choosing a public key that is a function of the peer's key. Both parties then derive a shared secret via ECDH and expand it using HKDF-SHA256 with a protocol-specific salt (`D2D_SALT`, 32 bytes, hardcoded in GMS) to produce per-direction session keys for AES-256-CBC encryption and HMAC-SHA256 authentication.

D2D Encryption Layer. After UKEY2 completes, all subsequent frames are expected to use the D2D `SecureMessage` wire format: the plaintext `OfflineFrame` protobuf is encrypted with AES-256-CBC (IV prepended), then an HMAC-SHA256 tag is computed over the ciphertext. The encrypted payload and tag are wrapped in a `SecureMessage` protobuf containing `header_and_body` (the encrypted frame) and `signature` (the HMAC tag). On the receiver side, the `D2DConnectionContext` class is responsible for unwrapping, verifying, and decrypting `SecureMessage` envelopes before passing the plaintext frame to the dispatcher. As we show in V4 and V5, this decryption step is not enforced uniformly. We detail the three discovered vulnerabilities below. Figure 3 shows the intended Quick Share connection flow that V4/V5 violate.

6.2 V4: Pre-Authentication Frame Processing Bypass

Root Cause. The Nearby Connections protocol used by Samsung Quick Share is designed to enforce a strict handshake sequence: a `ConnectionRequest` registers the endpoint, then UKEY2 (*ClientInit* → *ServerInit* → *ClientFinished*) establishes mutual authentication and derives D2D encryption keys. Only after this four-step handshake should

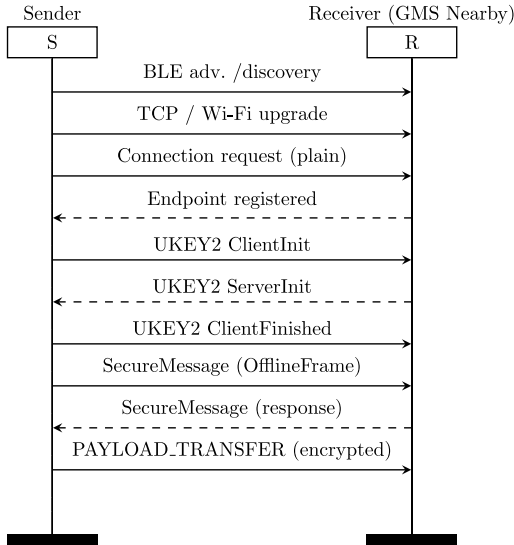


Figure 3: Intended Quick Share connection flow. V4 sends *OfflineFrames* above the dashed encryption boundary (before UKEY2 finishes), and V5 sends three *OfflineFrame* types in plaintext below the boundary (after UKEY2 finishes), in both cases bypassing the *SecureMessage* envelope.

the server process application-level *OfflineFrame* messages. However, the *PcpManager* in Google Mobile Services (GMS) begins dispatching *OfflineFrame* messages to handlers *immediately after Step 1* (*ConnectionRequest*), without waiting for UKEY2 to complete. An unauthenticated attacker needs only to send a single, fixed-format *ConnectionRequest* protobuf (requiring no secrets), after which the server parses, dispatches, and responds to *KEEP_ALIVE*, *BANDWIDTH_UPGRADE*, and *CONNECTION_RESPONSE* frames, despite never having established an authenticated or encrypted session.

Trigger. The attacker opens a TCP connection to the Quick Share port (53601 in our tests), sends a valid *ConnectionRequest*, and immediately follows it with an *OfflineFrame* of any of the three affected types. No UKEY2 handshake messages are required. The server responds within milliseconds: for *KEEP_ALIVE*, a 58-byte UKEY2 error message ("Expected, but did not find *ClientInit* message type") confirming the frame was parsed and dispatched to the UKEY2 state machine; for *BANDWIDTH_UPGRADE* and *CONNECTION_RESPONSE*, similar error responses. We confirmed this at every stage of the connection lifecycle across 5 independent trials each (100% reproduction rate).

Systematic Verification. To characterize the pre-authentication attack surface, we tested frame processing at each stage of the connection lifecycle (5 trials/stage). Without any prior *ConnectionRequest* (CR) all frames are silently

dropped (negative control, 5/5); immediately after a single CR—with *no* UKEY2 message exchanged—*KEEP_ALIVE*, *BANDWIDTH_UPGRADE*, and *CONNECTION_RESPONSE* are all dispatched and produce server responses (5/5 each); after CR+*ClientInit*, *KEEP_ALIVE* is still processed (5/5); after CR+*ClientInit*+*ServerInit*, the server transitions to a state that expects only *ClientFinished* and drops non-handshake frames (5/5); after the full handshake all frames are accepted (positive control, 5/5). Stages 1–1c are the key finding—three frame types are dispatched to handlers before any UKEY2 authentication.

Impact. This vulnerability (CWE-287, CWE-306) breaks the fundamental assumption that no application-level interaction occurs before cryptographic authentication: the attacker drives the Quick Share protocol state machine without any authentication; *KEEP_ALIVE* injection prevents connection timeouts; and the dispatcher processes attacker-controlled protobuf content before any integrity check, expanding the pre-authentication surface to frame types designed to be post-authentication only. Combined with V5, this lets a fully unauthenticated attacker inject and have processed all three affected frame types without ever establishing encryption.

6.3 V5: Device-to-Device Encryption Bypass

Root Cause. After the UKEY2 handshake completes and derives AES-256-CBC + HMAC-SHA256 session keys, all subsequent *OfflineFrame* messages *must* be wrapped in a D2D *SecureMessage* envelope that provides both confidentiality and integrity. However, the frame dispatcher in the *EndpointChannelManager* routes incoming frames based on the *V1Frame.type* field *before* checking whether the frame was received through the D2D decryption pathway. As a result, 3 out of 7 post-handshake *OfflineFrame* types are processed as raw (unencrypted) protobuf: *CONNECTION_RESPONSE* (type 2), *BANDWIDTH_UPGRADE* (type 4), and *KEEP_ALIVE* (type 5). The remaining 4 types (*PAYLOAD_TRANSFER*, *DISCONNECTION*, *PAIRED_KEY_ENCRYPTION*, *PAIRED_KEY_RESULT*) are correctly rejected when sent without encryption, indicating the encryption check is performed per-handler rather than at the dispatcher level and three handlers simply omit it.

Trigger. The attacker completes a full UKEY2 handshake with the target device over TCP (*ConnectionRequest* → *ClientInit* → *ServerInit* → *ClientFinished* → *ConnectionResponse*), deriving D2D encryption keys. The attacker then sends a *raw* *OfflineFrame* protobuf not wrapped in the mandatory *SecureMessage* for any of the three affected types. The server processes the frame and responds:

- *CONNECTION_RESPONSE*: 104-byte response; the server advances its connection state machine to “accepted,” as con-

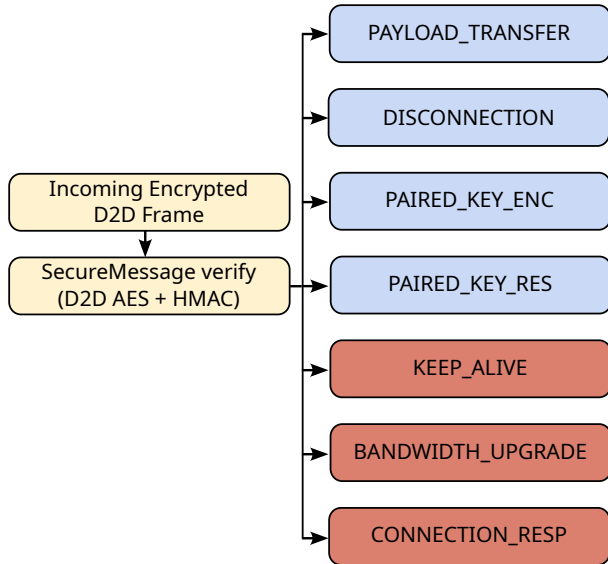


Figure 4: V5 per-handler encryption enforcement. Four of seven post-handshake `OfflineFrame` types are correctly rejected when unencrypted; the three red types are dispatched and processed even when delivered as a raw, unwrapped protobuf, despite the active D2D session.

firmed by logcat: "Endpoint [id] has accepted the connection".

- `BANDWIDTH_UPGRADE`: 29-byte response containing a `SAFE_TO_CLOSE` event that *echoes back* the attacker-supplied IP address and port number, proving the server parsed attacker-controlled data from the unencrypted frame.
- `KEEP_ALIVE`: 16-byte response with `ack=true`, demonstrating the KeepAlive handler has no encryption check whatsoever.

We confirmed all three bypasses across 5 independent trials with 100% reproduction rate on Samsung Galaxy S23 Ultra running Quick Share v13.8.01.11 with GMS 26.05.34. Figure 4 highlights which post-handshake frame types cross the `SecureMessage` boundary and which do not.

Impact. An on-path attacker on the same Wi-Fi network can inject unencrypted control frames into an active Quick Share session, defeating the D2D encryption layer: unencrypted `CONNECTION_RESPONSE` forces a connection into “accepted” state, potentially bypassing user consent; `KEEP_ALIVE` injection prevents timeouts and keeps sessions alive indefinitely; and unencrypted `BANDWIDTH_UPGRADE` frames cause the server to process attacker-supplied IP/port values, leaking endpoint state.

6.4 v6: Use-After-Free via Endpoint Collision Race Condition

Root Cause. Google Quick Share for Windows (version 1.0.2472.1) manages connected peers through dynamically allocated `EndpointChannel` objects in the Nearby Connections library. We discovered a critical use-after-free (UAF) in the `OnEncryptionFailed` callback, triggered when `EncryptionRunner::StartServer()` in `encryption_runner.cc` fails the UKEY2 handshake. The vulnerability manifests when two connections arrive with *identical endpoint identifiers and nonce values*: a collision handler in `base_pcp_handler.cc` tears down the endpoint (closing both channels and freeing the `EndpointChannel` object), while a concurrent thread pool worker executes the encryption failure callback that still holds a raw pointer to the freed object. The stale pointer is subsequently dereferenced for a vtable virtual function call, causing an `ACCESS_VIOLATION`.

Notably, the source code at `base_pcp_handler.cc` (line 814) contains a developer comment acknowledging a *previous bug* in this exact code path: “We had a bug here, caused by a race with `EncryptionRunner`.” The applied fix performs a channel identity comparison (`*endpoint_channel != *pending_connection_info.channel`), but this comparison *itself dereferences the freed pointer* before comparing, creating a new UAF.

Trigger. We developed a multi-phase Python reproducer that rapidly cycles real ECDH handshakes against the Quick Share TCP listener using a *fixed* endpoint identifier (“FZ”), causing register/teardown task queue races. The reproducer proceeds through six phases: sequential handshake cycling (30 rounds), concurrent handshake storm (4 threads × 15), interleaved collision during handshake (30 rounds), heavy sustained cycling (100 handshakes), final storm (6 threads × 20), and a combined attack phase. The crash typically occurs when the collision handler logs “cleaned up the collision with endpoint FZ by closing both channels. Our nonces were identical” while a concurrent worker in `encryption_runner.cc` attempts to invoke a virtual method on the freed `EndpointChannel`. The race window is wide (~11+ seconds of task queue stall), making reproduction reliable.

Crash Analysis. Under WinDbg with Page Heap enabled, the crash manifests as a vtable dereference on a freed `EndpointChannel`: `mov rcx, [rdi+20h]; mov rax, [rcx] (read freed vtable); call [rax+60h]`. The exception occurs at `nearby_share!GetStringFeatureFlagDart+0xaa61e` (file offset 0x0116fb0e), with `rcx` pointing into a Page-Heap guard page (`MEM_RESERVE` state), confirming the object has been freed. The call stack originates from

ntdll!TppWorkerThread through the Nearby Connections thread pool.

Impact. The crash primitive is a classic vtable-hijack pattern and exploitation is plausible: (i) Control Flow Guard is *disabled* in `nearby_share.exe`, so no indirect-call target validation is performed; (ii) without Page Heap, the freed slot falls into the normal process heap and can be reclaimed by a subsequent same-size-class allocation; (iii) the attacker controls incoming heap-allocated connection data that could reclaim the freed `EndpointChannel`; (iv) the wide ~ 11 s race window provides ample time for heap spraying between free and use. A standard exploitation path is fake-vtable $\rightarrow$ ROP $\rightarrow$ `VirtualProtect` $\rightarrow$ shellcode (the usual DEP bypass for CFG-disabled binaries). We confirmed the DoS impact but did not develop a full exploit.

Disclosure. We reported V4–V6 through Samsung Mobile Security and the Google Vulnerability Reward Program (VRP). Samsung transferred V4 and V5 to Google after determining that the affected code paths originate in Google’s Nearby/Quick Share components shipped to Samsung; those reports are under investigation at the time of writing. Google’s security team acknowledged the validity of V6 against Quick Share for Windows version 1.0.2472.1 and awarded a bounty; a CVE identifier is pending assignment. Section 7.5 summarises the full disclosure status.

7 Discussion

7.1 Cross-Platform Comparison

Both protocols expose pre-authentication attack surfaces from wireless proximity (AirDrop’s TLS listener accepts any connection; Quick Share’s BLE/Wi-Fi-Direct discovery is similarly open) and exhibit insufficient input validation at the application layer—Apple in HTTP path handling and plist parsing, Samsung in pre-authentication frame dispatching and encryption enforcement, Google in endpoint lifecycle management. The designs diverge in encoding (AirDrop’s seven-layer HTTP $\rightarrow$ bplist $\rightarrow$ DVZip $\rightarrow$ CPIO stack vs. Quick Share’s flatter Protocol Buffers) and in concurrency model (AirDrop serializes via a dispatch semaphore, while Quick Share’s multi-threaded endpoint manager admits the V6 use-after-free). Each of V1–V6 instantiates a distinct root-cause class—reachable assertion (V1), unbounded recursion (V2), parser state confusion + NULL deref (V3), missing pre-authentication frame check (V4), missing encryption enforcement (V5), and concurrent endpoint UAF (V6)—suggesting that proximity transfer protocols expose a broad spectrum of bug classes rather than a single dominant failure mode. The recurrence of assertion- and parser-class bugs in Apple’s code indicates that defensive patterns suited to application

reliability (`fatalError`) become liabilities in network-facing daemons; in Quick Share, V4 and V5 share a single root cause—authentication and encryption checks delegated to individual frame handlers rather than enforced at the dispatcher.

7.2 Recommended Mitigations

For Apple AirDrop (V1–V3). Replace every `fatalError()` and force-unwrap reachable from network input with guard let and a graceful error return (V1); enforce a maximum nesting depth in Foundation’s `XMLPlistScanner` (e.g., 64 levels; legitimate AirDrop traffic uses depth ≤ 4) together with a maximum HTTP body size (V2); and validate HTTP framing strictly per RFC 9110 §8.6, rejecting conflicting `Content-Length` headers, negative chunk sizes, and oversized header values (V3).

For Quick Share (V4–V6). Reject all `OfflineFrame` types except UKEY2 handshake messages until authentication completes (V4); decrypt and integrity-verify all post-handshake frames at the dispatcher rather than per-handler (V5); and protect both the endpoint map and endpoint objects with a single lock and defer deallocation via reference counting or epoch-based reclamation until all in-flight operations complete (V6).

7.3 Broader Impact

The vulnerabilities affect a substantial device population: Apple reports over 2.2 billion active devices [2] running `sharingd`, and Google reports over 3 billion active Android devices [8] with Quick Share as the default sharing mechanism on Samsung devices and available system-wide on Android. Since V4 and V5 reside in Samsung’s Quick Share service layer, further investigation is needed to determine whether other Android OEMs that integrate Quick Share share the same vulnerable code paths. While proximity limits the attack radius to ~ 10 –30 m, high-density environments (airports, transit, conferences) enable a single attacker to reach hundreds of devices simultaneously.

7.4 Future Work

Promising directions include escalating V3’s NULL-pointer dereference to code execution; verifying V4 and V5 on other Android OEMs that integrate Quick Share, exploring LLM-assisted decompilation to reduce the manual binary analysis effort our AirDrop reverse engineering required, fuzzing XNU’s `I080211AWDLPeer` AWDL driver at the link layer, and formally verifying the state machine in Figure ?? with ProVerif or Tamarin.

7.5 Ethical Considerations and Responsible Disclosure

All testing was performed exclusively on devices we own in an isolated network environment with no bystander devices in wireless range. All six vulnerabilities were reported to the respective vendors through their coordinated disclosure channels:

- **Apple (V1–V3).** Reported to Apple Product Security. All three issues were *acknowledged* and are currently *under fix*; no CVE identifiers have been assigned at the time of writing.
- **Samsung (V4–V5).** Reported to Samsung Mobile Security, who *transferred* the cases to Google after determining that the affected code paths originate in Google’s Near-by/Quick Share components shipped to Samsung.
- **Google (V4–V6).** Reported via Google’s Vulnerability Reward Program. V6 (the Windows Quick Share use-after-free) was acknowledged and Google awarded a bounty for this report; a CVE identifier for V6 is pending assignment. V4 and V5 remain under investigation.

We released AIRFUZZ, crash reproduction scripts, the Frida auto-accept daemon, protocol documentation, and the seed corpus for WOOT artifact evaluation on Zenodo: <https://zenodo.org/records/20442029>

7.6 Limitations

Our analysis has several limitations: `sharingd` is stripped, so our symbolication relies on export tables, string references, and function boundary heuristics; Frida-based coverage instrumentation may mask or trigger additional race conditions; our Quick Share analysis focused on Samsung Android and Google Windows, other OEMs may differ, and we achieved code execution only once for V6, though V3 to V5 warrant further investigation.

8 Related Work

AWDL and AirDrop Protocol Analysis. Stute et al. [18–20] reverse engineered the AWDL link layer, released the open-source OWL implementation, and demonstrated denial-of-service and man-in-the-middle attacks at the wireless layer. Their work forms the substrate on which the present paper’s application-layer analysis builds: where OWL exposed AWDL itself, we focus on the HTTP/binaryplist/DVZip/CPIO stack served over AWDL by `sharingd`. Project Zero’s Beer [3] subsequently demonstrated a full iOS zero-click radio-proximity exploit against the AWDL kernel driver, showing that the AWDL surface extends below user space; our V1–V3 add three new pre-authentication bugs on the same wireless reach, but in the user-space sharing daemon and its supporting frameworks.

AirDrop Privacy. Heinrich et al. [17] developed OpenDrop, an open-source AirDrop implementation, and analyzed privacy issues in AirDrop’s contact discovery and authentication design, showing that exchanged hash-based contact identifiers can enable leakage of phone numbers and email addresses to nearby adversaries. Public reporting has also discussed how such weaknesses may be abused in practice; for example, Bloomberg news [4] reported claims by Chinese authorities about identifying AirDrop message sources. Martin et al. [15] analyzed privacy properties of Apple’s broader Continuity protocol family over BLE, demonstrating identifying and behavioral leakage in related always-on proximity services.

PrivateDrop. Heinrich et al. [11] proposed PrivateDrop, a privacy-preserving mutual authentication design based on Private Set Intersection (PSI) to replace AirDrop’s hash-based contact verification. Apple later introduced an “Everyone” receive-mode time limit (first deployed in China with iOS 16.1.1 and then rolled out more broadly) which mitigates some abuse scenarios but does not adopt PSI-style mutual verification [11].

Nearby Connections, Quick Share, and Nearby Share. Antonioli et al. [1] reverse engineered and analyzed Google’s *Nearby Connections* framework on Android, uncovering design and implementation weaknesses in proximity-based device-to-device communication. More recently, Yair and Cohen [21] analyzed Google’s Quick Share for Windows and reported a remote code execution attack chain (including CVE-2024-38271 and CVE-2024-38272) exploiting file-handling and protocol-frame processing. Complementing these vulnerability-centric analyses, Kleidermacher [13] describes Google’s security-driven design considerations for enabling Android Quick Share interoperability with Apple’s AirDrop. Our work differs in focus by examining Samsung’s extended Android Quick Share implementation and by developing protocol-aware fuzzing tailored to multilayer proximity-transfer stacks.

Mobile Daemon Fuzzing and Zero-Click Surfaces. Protocol-aware fuzzing of mobile system components and always-on daemons has been explored in several contexts. Groß [10] demonstrated remote exploitation of the iPhone via iMessage, highlighting the risk of zero-click attack surfaces in background services. Silvanovich [16] systematically analysed the fully remote attack surface of the iPhone, cataloguing zero-click entry points across system services. Closest to our fuzzing methodology, TOOTHPICKER [12] fuzzed the iOS Bluetooth daemon over the BLE link layer using Frida-based in-process feedback, and FPICKER [14] extends that approach with AFL++-style mutation, Frida *Stalker* coverage, and richer mode support; both target the same class of always-on Apple daemons we analyse, and

we discuss why their Stalker-based coverage fails on current arm64e macOS/iOS (Section 4). Project Zero’s binary-instrumentation toolchain, TINYINST and JACKALOPE [5, 6], was used to fuzz `coreaudiod` on macOS 15 [7]; porting that out-of-process rewriting model to AWDL-attached `sharingd` is left to future work. Our fuzzer therefore borrows the protocol-aware, Frida-based posture of `ToothPicker/fpicker` while replacing Stalker coverage with hand-selected `Interceptor` hooks at each layer boundary of the AirDrop stack.

9 Conclusion

We presented the first cross-platform security analysis of Apple AirDrop and Android Quick Share, the two dominant proximity file-transfer protocols serving over five billion devices. By reverse engineering AirDrop’s seven-layer protocol stack—including the undocumented DVZip compression format and developing the AIRFUZZ protocol-aware fuzzer, we discovered three pre-authentication vulnerabilities in Apple AirDrop (V1–V3): two zero-click and one post-accept. Through complementary manual security assessment of Android Quick Share, we found three additional vulnerabilities: a pre-authentication frame-processing bypass (V4) and a D2D encryption bypass (V5) in Samsung Quick Share, and a use-after-free in Google Quick Share for Windows (V6). Our findings show that proximity protocols are a *structural* vulnerability class: fatal assertions in network-facing code, missing dispatcher-level authentication and encryption enforcement, and unsynchronized concurrent endpoint management recur across independently developed implementations. All six vulnerabilities have been responsibly disclosed; AIRFUZZ and all reproduction artifacts released as open source.

References

- [1] Daniele Antonioli, Nils Ole Tippenhauer, and Kasper Rasmussen. Nearby threats: Reversing, analyzing, and attacking Google’s ‘nearby connections’ on Android. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2019.
- [2] Apple Inc. Apple reports first quarter results. <https://www.apple.com/newsroom/2024/02/apple-reports-first-quarter-results/>, February 2024. Includes statement that the installed base of active devices surpassed 2.2 billion. Accessed: 2026-02-25.
- [3] Ian Beer. An iOS zero-click radio proximity exploit odyssey. Google Project Zero Blog, December 2020. <https://googleprojectzero.blogspot.com/2020/12/an-ios-zero-click-radio-proximity.html>. Accessed: 2026-05-13.
- [4] Bloomberg. China says cracked Apple’s AirDrop to identify message sources. <https://www.bloomberg.com/news/articles/2024-01-09/china-says-cracked-apple-s-airdrop-to-identify-message-sources>, January 2024. Accessed: 2026-02-25.
- [5] Ivan Fratric. TinyInst: A lightweight dynamic instrumentation library. <https://github.com/googleprojectzero/TinyInst>, 2020.
- [6] Ivan Fratric. Jackalope: A binary, coverage-guided fuzzer built on top of TinyInst. <https://github.com/googleprojectzero/Jackalope>, 2021.
- [7] Ivan Fratric. The GASMAN approach: fuzzing coreaudiod on macOS. Google Project Zero blog, 2024. <https://googleprojectzero.blogspot.com/>.
- [8] Google. There are over 3 billion active Android devices. <https://www.theverge.com/2021/5/18/22440813/android-devices-active-number-smartphones-google-2021>. Accessed: 2026-02-25, May 2021.
- [9] Google. Use Quick Share on your Android device. <https://support.google.com/android/answer/9286773>, 2024. Official documentation describing Quick Share discovery, transfer, and visibility settings. Accessed: 2026-02-26.
- [10] Samuel Groß. Remote iPhone exploitation part 1: Poking memory via iMessage and CVE-2019-8641. Google Project Zero Blog, January 2020. <https://projectzero.google/2020/01/remote-iphone-exploitation-part-1.html>. Accessed: 2026-02-25.
- [11] Alexander Heinrich, Matthias Hollick, Thomas Schneider, Milan Stute, and Christian Weinert. PrivateDrop: Practical privacy-preserving authentication for Apple AirDrop. In *Proceedings of the 30th USENIX Security Symposium*. USENIX Association, 2021.
- [12] Dennis Heinze, Jiska Classen, and Felix Rohrbach. ToothPicker: Apple picking in the iOS Bluetooth stack. In *Proceedings of the 14th USENIX Workshop on Offensive Technologies (WOOT)*, 2020.
- [13] Dave Kleidermacher. Android quick share support for airdrop: A secure approach to cross-platform file sharing. <https://security.googleblog.com/2025/11/android-quick-share-support-for-airdrop-security.html>, November 2025.
- [14] Dennis Maier. fpicker: Frida-based fuzzing suite supporting various modes including AFL++ in-process fuzzing. <https://github.com/ttdennis/fpicker>, 2021.

- [15] Jeremy Martin, Douglas Alpuche, Kristina Bodeman, Lamont Brown, Ellis Fenske, Lucas Foppe, Travis Mayberry, Erik Rye, Brandon Sipes, and Sam Teplov. Hand-off all your privacy – a review of Apple’s Bluetooth Low Energy continuity protocol. *Proceedings on Privacy Enhancing Technologies*, 2019(4):34–53, 2019.
- [16] Natalie Silvanovich. The fully remote attack surface of the iPhone. Google Project Zero Blog (Black Hat USA 2019 talk announcement/material), 2019. <https://projectzero.google/2019/08/the-fully-remote-attack-surface-of.html>. Accessed: 2026-02-25.
- [17] Milan Stute and Alexander Heinrich. OpenDrop: An open-source AirDrop implementation. <https://github.com/seemoo-lab/opendrop>, 2021. Secure Mobile Networking Lab (SEEMOO), TU Darmstadt. Accessed: 2026-02-25.
- [18] Milan Stute, Alexander Heinrich, Jannik Lorenz, and Matthias Hollick. Disrupting continuity of Apple’s wireless ecosystem security: New tracking, DoS, and MitM attacks on iOS and macOS through Bluetooth Low Energy, AWDL, and Wi-Fi. In *Proceedings of the 30th USENIX Security Symposium*. USENIX Association, 2021.
- [19] Milan Stute, David Kreitschmann, and Matthias Hollick. One billion apples’ secret sauce: Recipe for the Apple Wireless Direct Link ad hoc protocol. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom)*. ACM, 2018.
- [20] Milan Stute, Sashank Narain, Alex Mariotto, Alexander Heinrich, David Kreitschmann, Guevara Noubir, and Matthias Hollick. A billion open interfaces for Eve and Mallory: MitM, DoS, and tracking attacks on iOS and macOS through Apple Wireless Direct Link. In *Proceedings of the 28th USENIX Security Symposium*. USENIX Association, 2019.
- [21] Or Yair and Shmuel Cohen. QuickShell: Sharing is caring about an RCE attack chain on quick share. SafeBreach Labs (research write-up and conference materials), 2024. <https://www.safebreach.com/blog/rce-attack-chain-on-quick-share/>, Accessed: 2026-02-25.
- [22] Or Yair and Shmuel Cohen. QuickShell: Sharing is caring about an RCE attack chain on quick share. SafeBreach Labs (research write-up and conference materials), 2024. <https://www.safebreach.com/blog/rce-attack-chain-on-quick-share/>, Accessed: 2026-02-25.

Breaking 5G on The Lower Layer

Subangkar Karmaker Shanto
Purdue University
sshanto@purdue.edu

Imtiaz Karim
The University of Texas at Dallas
imtiaz.karim@utdallas.edu

Elisa Bertino
Purdue University
bertino@purdue.edu

Abstract—As 3GPP systems have strengthened security at the upper layers of the cellular stack, plaintext PHY and MAC layers have remained relatively understudied, though interest in them is growing. In this work, we explore lower-layer exploitation in modern 5G, where recent releases have increased the number of lower-layer control messages and procedures, creating new opportunities for practical attacks. We present two practical attacks and evaluate them in a controlled lab testbed. First, we reproduce a SIB1 spoofing attack to study manipulations of unprotected broadcast fields. By repeatedly changing a key parameter, the UE is forced to refresh and reacquire system information, keeping the radio interface active longer than necessary and increasing battery consumption. Second, we demonstrate a new Timing Advance (TA) manipulation attack during the random access procedure. By injecting an attacker-chosen TA offset in the random access response, the victim applies incorrect uplink timing, which leads to uplink desynchronization, radio link failures, and repeated reconnection loops that effectively cause denial of service. Our experiments use commercial smartphones and open-source 5G network software. Experimental results in our testbed demonstrate that TA offsets exceeding a small tolerance reliably trigger radio link failures in our testbed and can keep devices stuck in repeated re-establishment attempts as long as the rogue base station remains present. Overall, our findings highlight that compact lower-layer control messages can have a significant impact on availability and power, and they motivate placing defenses for initial access and broadcast procedures.

I. INTRODUCTION

Security analysis of upper layers of the cellular stack of 4G and 5G have been a field of research for a long time. A lot of research works focus on the upper layers of the cellular stack, including RRC and NAS [7], [8], [9], [10], [11], [40], [41], [30]. Accordingly, the defense has been studied thoroughly for those layers. Consequently, 3GPP has strengthened the security of 5G increasing subscriber-identity privacy via SUPI concealment (SUCI) and applies NAS security after the security mode procedure [1]. However, the introduction of encryption mechanisms at control layers comes at a cost of higher latency [31]. Hence, to facilitate faster reconfiguration of the network many PHY/MAC procedures in current 4G/5G systems are neither encrypted nor integrity protected [6], [32]. This creates scope for adversaries to exploit

the users. However, prior works have predominantly focused on vulnerabilities in 4G/5G upper-layer protocols and services, such as core network functions, RRC/NAS signaling, and network slicing, while comparatively fewer studies examine low-layer (PHY/MAC) control procedures [33], [34]. A recent work has explored this direction of exploiting the lower layers of 4G/5G [32] motivating further exploration of this direction.

In this work, we present a systematic evaluation for unprotected control-plane parameters in 5G NR, identifying critical vulnerable fields across MAC and RRC layers that lack integrity protection during initial access and broadcast procedures. Building upon prior work [32] that speculated potential vulnerabilities in Timing Advance Command manipulation and performed SIB1 spoofing, we implement and empirically validate these attacks through rogue gNodeB deployment combining passive reconnaissance via downlink sniffing and active manipulation using software-defined radio platforms. Our systematic experimental evaluation demonstrates that Timing Advance Command manipulation in Random Access Response messages causes connected-mode denial-of-service through uplink timing desynchronization and Radio Link Failure, while SIB1 spoofing induces idle-mode battery drain through forced system information re-acquisition cycles. Additionally, we systematically explore Tracking Area Code modification and SI window length toggling attacks to assess their practical feasibility.

II. RELATED WORKS

Though security analyses of 3GPP networks have largely centered on RRC and NAS layers where cryptographic protections exist, recent works have emphasized that the PHY and MAC layers remain *unprotected*, enabling energy-efficient spoofing, resource abuse, and passive privacy leakage. *Ludant et al.* [32] demonstrate systematic exploitation of beamforming, DCI, and MAC CE messages for sub-20 m localization, induced jamming, HARQ desynchronization, and carrier aggregation misuse in both 4G and 5G, motivating further lower layer security analysis.

Multiple recent studies characterize and exploit similar low-layer weaknesses. *Shaik et al.* and subsequent PHY/MAC injection works on 4G LTE highlight the practicality of control channel spoofing and adaptive overshadowing to degrade link availability [10], [16]. In 5G, NR-specific telemetry and decoder tools further show the feasibility of capturing temporary identifiers and unprotected scheduling procedures, expanding the attack surface for MAC-layer abuse [35].

Recent 4G/5G research also targets physical and MAC-layer vulnerabilities in real deployments. *Cao et al.* specifically examine 5G MAC Control Elements, revealing that integrity-less scheduling and mobility signaling can be tampered to force misallocation or sustain energy-draining states [36]. Complementary surveys highlight jamming, spoofing, beam manipulation, and ML-based detection challenges at the PHY/MAC layers in 5G, LTE and wireless networks [37], [38], [39]. Finally, while prior work demonstrated generic PHY/MAC spoofing including resource abuse and beam leakage, the *TA MAC CE injection* threat was only discussed as a potential vector for uplink timing desynchronization. In addition, there is no work that experimentally evaluated additional unprotected broadcast or scheduling fields. Our work moves beyond speculation by providing the first empirical evidence that *over the air TA manipulation in 5G NR* can break PHY timing alignment, trigger RLF, and sustain RLC/RRC reconnection loops, exposing a concrete and previously unvalidated low layer vulnerability [32]. Building on their insights, we also systematically analyze additional integrity-less broadcast and scheduling fields through SIB1 spoofing to expand the known 4G/5G unprotected control-plane attack surface.

III. BACKGROUND

A. 5G NR

5G New Radio (NR) organizes the radio interface into three main layers: L1 (PHY), L2 (which includes MAC, RLC and PDCP), and L3 (RRC) [3, Sec. 4.4]. At a high level, PHY (L1) handles the actual radio waveform (for example, synchronization and transmission/reception on physical channels), while MAC (part of L2) coordinates shared access to the air interface by scheduling and multiplexing control/data over time–frequency resources [2, Sec. 4.4]. RRC (L3) is the main access-stratum control protocol between the UE and gNB; it configures radio resources and bearers, supports mobility and measurements, and manages key connection-level control functions [6, Sec. 4.4]. Above these, the Non-Access Stratum (NAS) is the UE-to-core control protocol (for example, with the AMF), and its messages are carried through the RAN using RRC signaling [3, Sec. 7.6].

A typical control-plane entry flow starts when the UE finds a cell, synchronizes, reads broadcast system information, and then runs Random Access to obtain uplink timing alignment and initial uplink resources [3, Sec. 5.3.4, Sec. 9.2.6]. In contention-based Random Access, the MAC layer coordinates the key steps (preamble transmission, random access response reception, scheduled uplink transmission, and contention resolution) before declaring the procedure complete [2, Sec. 5.1]. Once access succeeds, the UE establishes an RRC connection to enable dedicated signaling (and subsequent security activation and configuration) [6, Sec. 5.3.3]. With the RRC connection in place, the UE can execute NAS procedures with the core network, such as 5G registration [4, Sec. 5.5.1].

B. System Information Block

Following the broader role of system information in 5G NR, we now focus on the primary broadcast block that governs UE initial access behavior. The System Information Block Type 1 (SIB1) serves as the fundamental broadcast information container in 5G NR networks, transmitted periodically on the Physical Downlink Shared Channel to provide essential cell access parameters to all UEs regardless of their connection state. According to 3GPP TS 38.331 [6], SIB1 carries critical configuration including PLMN identities, Tracking Area Code, cell barring information, RACH configuration, and common channel parameters required for initial cell selection and access. UEs in *RRC_IDLE* and *RRC_INACTIVE* states continuously monitor SIB1 transmissions by decoding PDCCH with SI-RNTI (System Information RNTI) to detect scheduling grants for SIB1 on PDSCH, typically transmitted with a periodicity of 160 milliseconds. Central to the change notification mechanism, SIB1 contains the *valueTag* field (a 5-bit parameter ranging from 0 to 31) that acts as a version indicator for system information content. When the network modifies any system information parameters, it increments *valueTag* (modulo 32) to signal UEs that cached system information has become stale. The detection of a changed *valueTag* can be used by the UE to judge whether stored SI remains valid to ensure they operate with current network configuration. This mechanism enables efficient broadcast updates without requiring dedicated signaling to individual UEs, but critically assumes integrity of the value tag parameter to prevent unnecessary acquisition cycles.

The Tracking Area Code (TAC) field: *trackingAreaCode*, within SIB1's *cellAccessRelatedInfo* specifies the 24-bit tracking area identifier for mobility management. UEs compare the received TAC against their registered tracking area; mismatches trigger Tracking Area Update procedures to maintain location registration with the 5G Core AMF, enabling efficient paging and mobility tracking across network coverage areas. Additionally, the *si-WindowLength* parameter in SIB1's *si-SchedulingInfo* defines the time window duration during which on-demand System Information messages (SIB2-SIB9) are transmitted on PDSCH. UEs use this value to calculate precise PDCCH monitoring occasions for SI-RNTI-based scheduling grants, establishing deterministic timing for acquiring cell-specific configuration beyond minimum SIB1 content.

C. Timing Advance Value

In 5G New Radio (NR) networks, the Timing Advance (TA) mechanism maintains its critical role as a foundational synchronization primitive at the Medium Access Control (MAC) layer, ensuring temporal alignment of uplink transmissions within the Orthogonal Frequency Division Multiple Access (OFDMA) framework. The necessity of TA in 5G NR becomes even more pronounced compared to LTE due to the introduction of flexible numerologies, higher frequency bands, and stringent latency requirements.

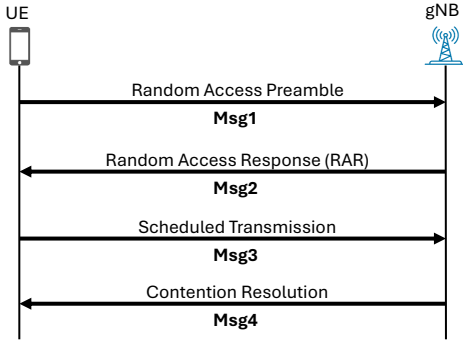


Fig. 1. RACH procedure for initial cell access

In the Random Access Channel (RACH) procedure specified in 3GPP TS 38.321 [2] as shown in Figure 1, the UE transmits Msg1 containing a randomly selected preamble sequence from a configured set on the Physical Random Access Channel (PRACH), similar to LTE but with enhanced preamble formats supporting different cell sizes and subcarrier spacings. The gNodeB (gNB) performs preamble detection through correlation processing, identifying both the preamble index and its timing offset relative to the expected PRACH reception window. This timing offset directly reflects propagation delay between UE and gNB, ranging from near-zero for small cells to several hundred microseconds for macro deployments. The gNB quantizes this measured offset into a Timing Advance Command, where each TA unit in 5G NR corresponds to a numerology-dependent time quantum to maintain consistent physical timing resolution across numerologies.

The Random Access Response (RAR), transmitted as Msg2 in RACH, carries the computed TA command from gNB to UE via a MAC Protocol Data Unit (PDU) on the Physical Downlink Shared Channel (PDSCH). The MAC RAR payload in 5G NR contains a 12-bit Timing Advance Command field (expanded from LTE’s 11 bits) capable of representing values from 0 to 3846, providing finer granularity and extended range to accommodate diverse deployment scenarios from dense urban small cells to rural macro cells spanning tens of kilometers. Upon successful decoding of the RAR, identified by monitoring the Physical Downlink Control Channel (PDCCH) for Downlink Control Information (DCI) format 1_0 scrambled with the Random Access Radio Network Temporary Identifier (RA-RNTI) computed from the PRACH transmission occasion parameters including time-frequency resource index, slot number, etc. The UE extracts the TA command and adjusts its uplink timing parameter NTA. This adjustment shifts the UE’s entire uplink transmission timing forward by the specified amount, ensuring that subsequent uplink transmissions, beginning with Msg3 (containing RRC Setup Request in the procedure) and continuing through all Physical Uplink Shared Channel (PUSCH) and Physical Uplink Control Channel (PUCCH) transmissions, arrive at the gNB aligned to slot boundaries within the tolerance provided by the cyclic prefix.

According to 3GPP TS 33.501 [1], the 5G security architecture implements integrity protection and ciphering of user-plane and control-plane data only after successful completion of the primary authentication procedure in the 5G Core Network, which necessarily occurs after RACH completion since radio resource establishment and RRC connection setup are prerequisites for authentication messaging. Consequently, the Random Access procedure encompassing Msg1 through Msg4 is executed without integrity protection, creating an exploitable attack surface at the physical and MAC layers. This design decision, inherited from previous cellular generations and maintained due to bootstrapping constraints where UEs lack established security context during initial network access, allows adversaries to manipulate critical synchronization parameters without cryptographic verification.

IV. SYSTEMATIC ANALYSIS OF LOWER LAYER PARAMETERS

We systematically selected parameters on PDSCH to explore based on their lack of integrity protection, direct control over resource-intensive UE operations, and broadcast impact scope, with part of the selection rationale motivated by [32] on lower-layer desynchronization and mobility signaling attacks. Table I lists the explored fields. We prioritize parameters that are (i) pre-integrity, (ii) critical to broadcast/initial access, and (iii) induce extra UE control procedures (e.g., SI reacquisition/mobility) or impact uplink timing. The *valueTag* of SIB1 triggers computationally expensive system information reacquisition affecting battery life, Tracking Area Code controls mobility signaling to core network, SI window length governs scheduling calculations for on-demand broadcasts, and Timing Advance Command manages uplink synchronization at the PHY/MAC layer where misalignment causes radio link failures. Together, these fields provide systematic coverage of the unprotected control-plane attack surface.

A. Threat Model

We consider an adversary located within the radio coverage region of a 4G/5G base station (BS). The adversary is assumed to possess a Software Defined Radio (SDR) platform and is capable of passively capturing RF I&Q sample streams from nearby cellular cells. Furthermore, consistent with prior work [12], [13], [14], [15], [32], we assume that the adversary can decode unencrypted BS downlink channels, including but not limited to synchronization and broadcast signals, system information blocks, scheduling grants, and low layer control headers. We name this decoder as sniffer. Finally, building on capabilities demonstrated in the literature [16], [18], [17], [32], the adversary is assumed to be able to synthesize and inject valid 4G/5G waveforms over the air with fine-grained temporal control, enabling targeted signal transmission at attacker specified time instants. Our testbed evaluation uses a rogue gNB/fake base station (FBS) to realize the required over-the-air downlink injection capability in a controlled setting. We do not evaluate “overshadowing” like [18] against a live serving gNB in our experiment.

TABLE I
SELECTED UNPROTECTED CONTROL-PLANE PARAMETERS

Parameter	Layer	Message	Role
Timing Advance (TA) Command	MAC	RAR	Controls uplink timing synchronization by specifying transmission advance to compensate propagation delay
<i>valueTag</i>	RRC	SIB1	Version indicator for system information content; increment triggers complete SI re-acquisition
Tracking Area Code (TAC)	RRC/NAS	SIB1 cellAccessRelatedInfo	Identifies tracking area for mobility management; mismatch triggers Tracking Area Update to AMF
<i>si-WindowLength</i>	RRC	SIB1 si-SchedulingInfo	Defines time window duration (5/10/15/20 ms) for scheduling on-demand SI messages (SIB2-SIB9)

B. Root Causes of Vulnerability.

We leverage the absence of integrity protection as per 3GPP TS 33.501 [1] in 5G broadcast and initial access procedures. The UE continuously monitors SIB1 on PDSCH, as required by 3GPP compliance, even if the actual configuration is unchanged. Since SIB1 must be decoded before security context establishment, it lacks cryptographic integrity protection, enabling a rogue base station to modify parameters undetected and trigger false change notifications. The same vulnerability applies to Timing Advance (TA) extraction in the RAR, which occurs prior to RRC security activation; the UE applies the malicious TA to its physical layer uplink timing reference and sends Msg3 on the granted PUSCH resource using the injected offset, shifting all subsequent uplink transmissions, including the RRC Setup Request. Persistent uplink desynchronization can trigger RLF and reconnection loops.

C. SIB1 Spoofing Attack

Attack Overview. We systematically evaluate manipulation of three unprotected SIB1 broadcast parameters—*valueTag*, *trackingAreaCode*, and *si-WindowLength*. Our experiments use a rogue gNodeB (fake base station) to perform controlled over-the-air injection and to assess denial-of-service and resource-exhaustion effects on UEs in *RRC_IDLE* and *RRC_INACTIVE* states that monitor system information. UEs in *RRC_CONNECTED* typically do not reacquire SIB1 unless explicitly triggered by the network (e.g., paging), so we focus on idle/inactive states.

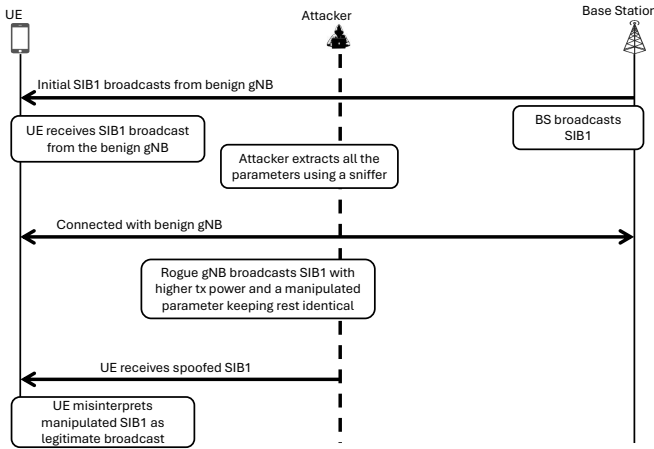


Fig. 2. SIB1 Spoofing

Attack Steps. We next describe the procedure for modifying each selected field in the SIB1 message for analysis using the SIB1 spoofing attack from [32] as shown in Figure 2.

valueTag Manipulation: After capturing legitimate SIB1 parameters via downlink sniffing from Primary and Secondary Synchronization Signals and Physical Broadcast Channel, the rogue gNB periodically increments the *valueTag* field (5-bit value ranging 0-31) every 10 seconds while maintaining all other SIB1 content unchanged. Upon detecting the modified value tag through periodic SI-RNTI monitoring on Physical Downlink Control Channel, the victim UE interprets this as system information change notification and discard cached SI. The UE then initiates complete system information acquisition by continuously monitoring PDCCH Control Resource Set zero for Downlink Control Information Format 1_0 with CRC scrambled by SI-RNTI, performing blind decoding to locate scheduling grants. Once detected, the UE decodes the scheduled Physical Downlink Shared Channel transmission containing SIB1, applies LDPC decoding for transport block extraction, only to discover no meaningful parameter changes beyond the incremented tag forcing repeated unnecessary acquisition cycles.

Tracking Area Code Modification: The rogue gNB periodically modifies the 24-bit Tracking Area Code field within SIB1's cellAccessRelatedInfo information element every 30 seconds after initial Parameter harvesting via PDSCH decoding of legitimate broadcasts, cycling through different TAC values to simulate UE movement across tracking area boundaries. According to 3GPP TS 24.501, a UE upon detection of TAC mismatches between cached registration area and received SIB1 while camping in *RRC_IDLE* mode may trigger a registration update when the UE determines it is camping on a TA not in its stored TAI list involving NAS Registration Request messages encapsulated in RRC UplinkInformationTransfer and transmitted via Physical Uplink Shared Channel to initiate N1 signaling with the Access and Mobility Management Function in 5G Core Network, potentially generating cascading signaling storms as multiple victim UEs simultaneously respond to broadcast TAC modifications with individual mobility management transactions consuming core network processing resources.

SI Window Length Toggling: The rogue gNB alternates the *si-WindowLength* parameter in SIB1's si-SchedulingInfo information element between enumerated values ms5, ms10,

and ms20 across periodic SIB1 transmissions on Physical Downlink Shared Channel with 160 ms periodicity, disrupting the deterministic scheduling framework defined in TS 38.331 Section 5.2.2.3 for on-demand System Information Blocks. UEs calculate exact Physical Downlink Control Channel monitoring occasions for additional SI messages (SIB2-SIB9) based on the standard formula. When the window length toggles between monitoring cycles for example from ms10 to ms20, the UEs applying previously cached values miscalculate SI-RNTI-scrambled PDCCH monitoring occasions within Control Resource Sets, causing blind decoding attempts in incorrect time-frequency resources while missing actual SI transmission slots, forcing extended PDCCH monitoring across multiple SI windows and potential system information acquisition failures for cell-edge UEs with marginal Signal-to-Interference-plus-Noise Ratio.

Expected Impact. Frequent modification of *valueTag* may damage UE's battery life because it systematically disrupts the Discontinuous Reception power-saving mechanism that enables energy-efficient idle-mode operation. Under normal conditions, idle-mode UEs enter deep sleep states between paging occasions, waking briefly to monitor paging channels and periodically verify system information validity, achieving duty cycles below 1 percent with receiver circuitry powered down more than 99 percent of the time. The continuous *valueTag* modifications force UEs to remain in active receiver mode for extended periods performing computationally intensive operations. Alongside individual UE battery drain, the attack may generate sustained Physical Downlink Control Channel congestion as all victim UEs simultaneously attempt SI acquisition following each value tag modification, competing for limited PDCCH blind decoding capacity and potentially blocking detection of legitimate paging messages or other control signaling. Tracking Area Code modification aims to trigger mobility management signaling storms by forcing victim UEs to initiate Tracking Area Update procedures upon detecting TAC mismatches, causing each UE to transmit NAS Registration Request messages via Physical Uplink Shared Channel to the Access and Mobility Management Function. When multiple victim UEs simultaneously respond to broadcast TAC changes, this might generate localized signaling load on the serving AMF, though large-scale overload would require coordinated attacks across multiple base stations given typical AMF dimensioning for large subscriber populations. In an overshadowing setting against a legitimate network, broadcast TAC changes can increase NAS update attempts toward the core; in our rogue gNB (standalone) evaluation, the effect is primarily UE-side activity toward the rogue cell. SI window length toggling may disrupt deterministic scheduling for on-demand system information acquisition by causing UEs to miscalculate Physical Downlink Control Channel monitoring occasions when cached *si-WindowLength* values mismatch current broadcasts. This might force extended PDCCH monitoring across multiple windows, delay acquisition of critical system information (SIB2 cell reselection parameters, SIB4 neighboring cell lists), and cause service disruption for cell-

edge UEs with marginal SINR where scheduling misalignment compounds into complete SI acquisition failure.

D. TA Manipulation Attack

Attack Overview. In this attack we manipulate the Timing Advance (TA) value during the RACH procedure by an attacker-chosen offset with the intention to cause denial of service on the victim UE. Compared to a fake base station (FBS) or rogue gNB that simply attracts UEs and provides no service, TA manipulation breaks availability by desynchronizing uplink timing via the (integrity-less) RAR TA field. In contrast to inactive rogue gNBs, TA manipulation maintains a partially functional connection: the UE completes RACH, believes it is connected, and repeatedly attempts recovery. This can cause failures after access appears to progress, leading to repeated RLF/retry loops, rather than an immediate "no service" failure.

Attack Steps. Figure 3 demonstrates our attack. Our attack methodology operates in two phases: passive reconnaissance through downlink sniffing to harvest system parameters followed by active manipulation via rogue gNB deployment replicating legitimate cell configurations to attract victim UEs. In the reconnaissance phase, the sniffer synchronizes to the target cell by detecting and decoding the Primary Synchronization Signal (PSS) and Secondary Synchronization Signal (SSS) transmitted in every Synchronization Signal Block (SSB), which occurs periodically with configurable periodicity. From the detected Physical Cell Identity (PCI) encoded in PSS/SSS and the Master Information Block (MIB) carried on the Physical Broadcast Channel (PBCH) within each SSB, the sniffer extracts critical system parameters including System Frame Number (SFN), subcarrier spacing for common control channels, PDCCH configuration via ControlResourceSet (CORESET) #0, and frequency location of initial downlink bandwidth part. With successful SSB detection and MIB decoding, the sniffer proceeds to decode System Information Block 1 (SIB1) scheduled on PDSCH according to timing and frequency allocation rules defined in 3GPP TS 38.213 [5]. SIB1 contains essential cell configuration parameters including Public Land Mobile Network (PLMN) identities, cell identity, Tracking Area Code, cell barring status, RACH configuration parameters, common channel configurations, and serving cell configuration for initial access. Of particular importance for the attack, SIB1's RACH-ConfigCommon information element specifies critical Random Access procedure parameters: the set of available PRACH preamble formats supporting different cell sizes and subcarrier spacings, PRACH frequency and time resource allocation defining which resource blocks and slots carry PRACH occasions, preamble transmission power parameters specifying target received power at gNB, power ramping parameters controlling power increase across consecutive RACH attempts, and RA response window configuration defining the duration during which UEs monitor PDCCH for RAR following preamble transmission. Additionally, the sniffer monitors PDCCH transmissions to observe Downlink Control Information scheduling RAR mes-

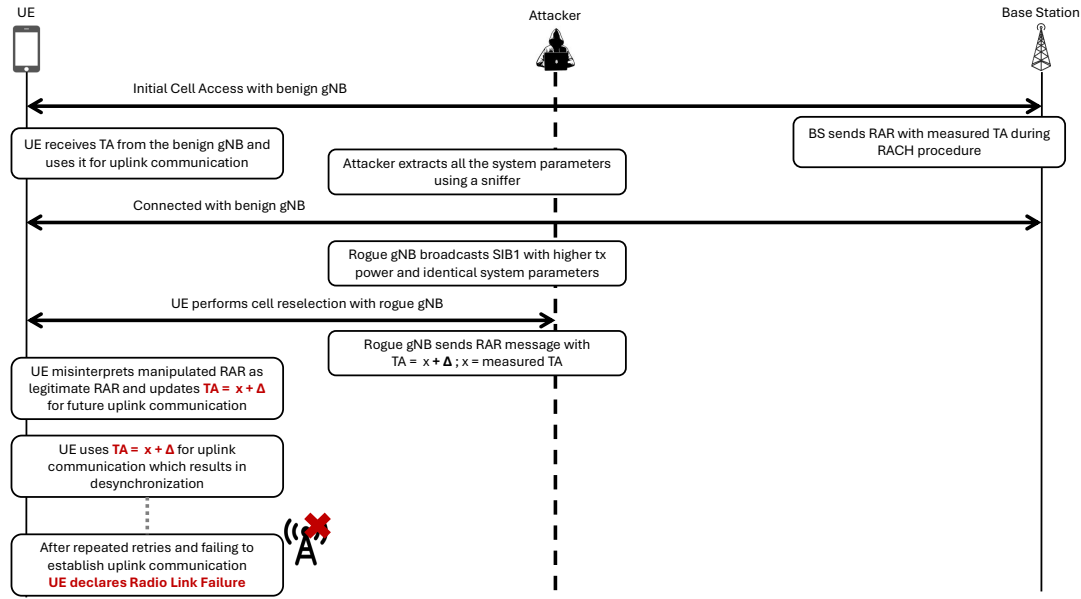


Fig. 3. TA Manipulation causes uplink de-synchronization and leads to Radio Link Failure (RLF)

sages. By decoding DCI format transmissions with CRC scrambled by RA-RNTI values corresponding to observed PRACH occasions, the sniffer can capture subsequent PDSCH transmissions carrying RAR MAC PDUs. Analysis of these legitimate RAR messages reveals typical Timing Advance Command values employed by the target gNB for UEs at various distances.

Armed with comprehensive system information and operational parameters extracted during reconnaissance, the adversary transitions to the active attack phase by deploying a rogue gNB. The adversary configures the rogue gNB with identical PLMN identities extracted from legitimate SIB1 during reconnaissance, ensuring victims perceive it as belonging to their home network or authorized roaming partners, avoiding automatic rejection due to PLMN mismatch. Crucially, the rogue gNB replicates all essential system parameters and transmits with higher signal power with a shorter distance from the UE than the benign gNB. This replication ensures that victim UEs, upon selecting the rogue cell based on favorable RSRP measurements, find expected system information enabling successful initial access procedures without triggering protocol inconsistencies.

When a victim UE selects the rogue gNB and initiates the Random Access procedure by transmitting a PRACH preamble on Msg1, the rogue base station performs standard preamble detection to identify the transmitted preamble index and measure its timing offset. This measured timing offset represents the legitimate propagation delay requiring compensation through timing advance. At this point, the attack diverges from legitimate protocol operation through injection of a manipulated timing offset into the Random Access Response. The rogue gNB computes the manipulated timing advance command by adding an offset to the legitimate measured value. The magnitude and sign of this

offset determine attack severity and impact characteristics: positive values cause premature UE transmission with uplink signals arriving earlier than intended subframe boundaries, while negative values induce delayed transmission with late-arriving signals. The adversary selects offset values based on attack objectives, balancing several considerations. Finally, the manipulated timing advance value is encoded into the Random Access Response MAC PDU structure defined in 3GPP TS 38.321 [2], which contains MAC subheader identifying the RAR for the detected preamble index and MAC RAR payload comprising the Timing Advance Command field, frequency domain resource assignment for Msg3 uplink grant, time domain resource assignment, Modulation and Coding Scheme for Msg3, and Temporary C-RNTI assignment. The rogue gNB assembles a properly formatted RAR with the manipulated timing advance value while maintaining correct structure for all other fields, ensuring the victim UE successfully decodes and processes the message. The RAR is transmitted on PDSCH in the time window specified by the RACH response window parameter, scheduled by DCI format on PDCCH with CRC scrambled by the appropriate RA-RNTI value computed from the PRACH occasion parameters matching the victim's Msg1 transmission.

From the victim UE's perspective, the received RAR appears entirely legitimate and indistinguishable from authentic network responses. The RAR arrives within the expected response window following PRACH transmission, carries appropriate RA-RNTI scrambling matching the transmitted preamble occasion, and contains correctly formatted MAC subheader and payload fields passing all protocol validity checks.

Expected Impact. The physical layer consequences of timing advance manipulation in 5G NR might be substantially severe. A manipulated TA value may disrupt uplink transmission

alignment, potentially driving the UE into a persistent loss of uplink synchronization. When the induced timing offset exceeds the gNB's tolerance for symbol boundary alignment, repeated decoding failures on scheduled uplink channels (e.g., PUSCH or PUCCH) prevent successful reception of control feedback and data. As a result, the gNB ceases uplink scheduling, marking the UE as out of sync. The UE, observing consecutive HARQ feedback failures and the absence of valid uplink grants, subsequently declare a Radio Link Failure (RLF). This transitions the UE to *RRC_IDLE* and initiates cell reselection followed by a new RACH procedure. Under sustained TA manipulation, the UE can reenter *RRC_CONNECTED* only to encounter recurring uplink timing misalignment, forming a reconnection loop characterized by repeated RLF declarations and RACH retries. Such loops amplify signaling overhead, increase random access congestion, and degrade service continuity, effectively creating a **denial of service** (DoS) condition through forced desynchronization and repeated connection reestablishment.

V. EXPERIMENT AND EVALUATION

A. Evaluation Setup

Hardware: Our experimental testbed comprises software-defined radio (SDR) devices utilizing USRP B210 hardware platforms for both legitimate and rogue gNB implementations. Each USRP B210 connects via USB 3.0 to dedicated laptop systems with low-latency kernel configurations, equipped with Intel Core Ultra 7 processors and 32GB RAM. For UE, we used commercial off-the-shelf (COTS) smartphones including OnePlus Nord 5G (Qualcomm Snapdragon 765G with X52 5G modem supporting FR1 bands n1/n3/n7/n28/n78/n78) and Google Pixel 5 (Snapdragon 765G with identical modem capabilities), both running Android Operating System without custom radio interface modifications, representing realistic victim devices operating according to 3GPP standard UE behavior. The testbed operates in controlled indoor laboratory environment with RF shielding to prevent interference with commercial networks.

Software: The legitimate base station deployment consists of integrated BS and Core network software components executing on a single host machine running Ubuntu 22.04 LTS, implementing complete end-to-end network functionality including Access and Mobility Management Function, Session Management Function, and User Plane Function elements. In contrast, the rogue base station operates in standalone mode executing only Radio Access Network protocol stack without core network connectivity, running on identical hardware specifications. Consequently, any NAS/control signaling impact on a real core network is not exercised in our evaluation. The base station software implementations utilize the open-source srsRAN [28] codebase for physical layer and protocol stack functionality, with the legitimate deployment using unmodified release version while the rogue implementation incorporates custom modifications to MAC layer Random Access Response generation and System Information Block broadcasting procedures as detailed in previous sections. The testbed evaluation

was conducted using one concurrently active UE. The setup models simultaneous uplink timing behavior and generalizes to multi-UE scenarios without altering the validity of the experiment conditions. Real-time protocol monitoring and packet inspection at the UE side utilizes Network Signal Guru (NSG) application [42, ver. 4.7.10] with root privileges enabled through Magisk framework, providing access to Qualcomm diagnostic interface for capturing physical layer parameters, RRC messages, MAC PDUs, and timing advance values during Random Access procedures and connected mode operation.

B. Experimental Results

1) *SIB1 Spoofing Attack Result:* Our experimental validation of the SIB1 spoofing attack demonstrates increased system-information reacquisition activity across both test UE models under sustained *valueTag* manipulation. With the rogue base station configured to increment *valueTag* every 10 seconds while maintaining constant transmission power 5 dB higher than the legitimate network to ensure victim cell selection, both OnePlus Nord 5G and Google Pixel 5 devices exhibited increased SI reacquisition activity, which is expected to reduce standby time compared to baseline. Network Signal Guru logs confirmed the attack mechanism, showing SIB1 acquisition events occurring at precise 10-second intervals with *valueTag* values incrementing sequentially, while actual SIB1 content parameters including PLMN-IdentityInfo, cellAccessRelatedInfo, and connEstFailureControl remained constant across all acquisitions. This confirms the successful demonstration of the attack in our testbed. In contrast, Tracking Area Code modification and SI window length toggling attacks demonstrate limited practical impact in our testbed. Despite successful TAC modification in broadcast SIB1 and confirmed UE detection through Network Signal Guru logging showing updated TAC values in decoded cellAccessRelatedInfo fields, commercial UEs (OnePlus Nord 5G and Pixel 5) did not trigger expected Tracking Area Update procedures or generate NAS signaling to the Core. Implementation-specific optimizations may defer mobility updates until user-plane traffic demand. Similarly, SI window length toggling between ms5, ms10, and ms20 values produced no observable system information acquisition failures or extended monitoring delays in our testbed. These negative results suggest that while these broadcast parameters lack integrity protection and remain theoretically manipulable, their exploitation requires either specific network topology conditions, edge-case propagation scenarios (low SINR for SI window), or alternative attack methodologies beyond simple parameter toggling to manifest observable impact.

2) *TA Manipulation Attack Result:* Our experimental evaluation of the Timing Advance (TA) manipulation attack demonstrates severe disruption to UE connectivity under controlled manipulation parameters. Upon activating the rogue BS with manipulated Random Access Response transmission, both the OnePlus Nord 5G and the Google Pixel 5 devices exhibited Radio Link Failure within 30 to 60 seconds of initial RACH procedure completion, with timing variations attributable to

UE-specific RLF detection thresholds and timer configurations per 3GPP TS 38.331 [6]. We systematically evaluated TA_delta injection values ranging from 5 to 60 timing advance units to characterize the empirically observed TA_delta tolerance range (for the tested UEs). At $TA_delta \leq 10$, both tested UE models maintained stable connectivity without observable service degradation, indicating that this manipulation magnitude remains within the combined tolerance of cyclic prefix protection and UE receiver robustness margins. However, increasing TA_delta to values exceeding 20 units consistently triggered Radio Link Failure for both device models, with TA_delta values of 30-60 units producing reliable RLF within the observed window. The RLF triggering mechanism follows the standard 3GPP procedure where persistent uplink timing misalignment causes repeated PUSCH and PUCCH decoding failures at the BS, resulting in accumulated out-of-sync indications at the UE physical layer, subsequently starting timer T310, and declaring Radio Link Failure upon T310 expiration without synchronization recovery.

Following RLF declaration, both UE models initiated RRC re-establishment procedures as specified in TS 38.331 [6, Sec. 5.3.7], attempting to recover the radio connection by performing cell selection and transmitting *RRCReestablishmentRequest* messages. However, due to the rogue BS maintaining higher transmission power for Synchronization Signal Blocks compared to the legitimate network, the UEs consistently reselected the rogue cell during the re-establishment procedure rather than reverting to the legitimate BS as shown in Figure 3. Consequently, subsequent RACH attempts during re-establishment again received manipulated Random Access Response messages with identical TA_delta injection, perpetuating the timing misalignment condition. This failure loop persisted for the entire duration that the rogue gNB was active in our testbed. We did not observe an automatic “escape” (e.g., cell blacklisting) within our observation window (more than an hour) for both the UEs we experimented with; such vendor-specific mitigation, if present, may bound persistence and is left to future work. Network Signal Guru packet captures confirmed this cyclic behavior, showing repeated *RRCReestablishmentRequest* transmissions at intervals corresponding to RLF detection periods, with each re-establishment attempt receiving manipulated Timing Advance Commands visible in MAC RAR subheader decoding. The UEs remained trapped in this connection failure loop for the entire duration of rogue BS operation, effectively denying stable network connectivity despite successful completion of individual RACH procedures, demonstrating the persistent denial-of-service capability of the attack as long as the rogue base station maintains transmission.

C. Limitations

Our evaluation uses two commercial UEs that share the same baseband family; thus, the observed TA -offset tolerance ranges and time-to-RLF should be interpreted as

implementation-dependent and may vary across other modem vendors and firmware versions. For SIB1 spoofing, we observed increased system-information reacquisition activity compared to baseline; however, we do not report power/current measurements and leave quantifying the resulting battery-life reduction (hours or % under controlled power methodology) to future work.

VI. CONCLUSION

Our TA manipulation attack empirically demonstrates that small control-plane tweaks at the MAC/PHY boundary can have an outsized impact on uplink alignment, pushing the UE into desynchronization and recovery loops, while our replication of the System Information Block spoofing attack validates that broadcast parameter manipulation creates increased control plane activity through forced system information reacquisition cycles. These results highlight that lower-layer control messages, although compact and frequent, are security-critical because they directly influence radio timing, access stability, and power management. The absence of integrity protection for both Random Access Response messages at the MAC layer and broadcast System Information Blocks at the RRC layer creates exploitable vulnerabilities affecting idle-mode UEs, demonstrating that security gaps exist at the lower layers.

Practical defense against these attacks requires combining cryptographic protection with intelligent monitoring at both UE and network sides. Integrity protection for Random Access Response messages and broadcast SIB1, using keys derived from UE credentials and access parameters, can allow UEs to reject manipulated TA commands and spoofed system information before application. Complementary UE and network-side monitoring—such as detecting inconsistent TA -signal strength relationships, abnormal *valueTag* update patterns, anomalous TA distributions, or synchronized SIB1 re-acquisition patterns, and RF fingerprinting that analyzes hardware-specific transmission characteristics—can reduce attack success rates while preserving normal 5G operation. Future work will quantify detection/false-positive rates across vendor-diverse UEs and live networks, and prototype lightweight authenticated RAR/SIB1 signaling with measured overhead.

ETHICS CONSIDERATIONS

All experiments conducted in this study adhered to applicable ethical guidelines. The evaluations were performed in a controlled lab environment using only our own attacker and victim devices. We further ensured physical and RF isolation of the testbed, with no adjacent third-party systems present or impacted during attack execution.

ACKNOWLEDGMENT

The work reported in this paper has been supported by NSF under grant 2112471, the University of Texas System Rising STARs Award (No. 40071109), and the startup funding from the University of Texas at Dallas.

REFERENCES

- [1] “5G; Security architecture and procedures for 5G System (3GPP TS 33.501 version 16.9.0 Release 16),” 2022.
- [2] “5G; NR; Medium Access Control (MAC) protocol specification (3GPP TS 38.321 version 17.3.0 Release 17),” 2023.
- [3] 3GPP TS 38.300, “NR; NR and NG-RAN Overall description; Stage-2,” Rel-18, v18.4.0, Dec. 2024.
- [4] 3GPP TS 24.501, “Non-Access-Stratum (NAS) protocol for 5G System (5GS); Stage 3,” Rel-18.
- [5] 3GPP, “NR; Physical layer procedures for control,” 3rd Generation Partnership Project (3GPP), TS 38.213, v18.5.0 (Release 18), Jan. 2025.
- [6] 3GPP, “NR; Radio Resource Control (RRC); Protocol specification,” 3rd Generation Partnership Project (3GPP), TS 38.331, v17.2.0, Sep. 2023.
- [7] K. Tu, A. Al Ishtiaq, S. M. M. Rashid, Y. Dong, W. Wang, T. Wu, and S. R. Hussain, “Logic Gone Astray: A Security Analysis Framework for the Control Plane Protocols of 5G Basebands,” in *Proc. USENIX Security Symp.*, 2024.
- [8] H. Kim, J. Lee, E. Lee, and Y. Kim, “Touching the untouchables: Dynamic security analysis of the LTE control plane,” in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019.
- [9] C. Yu, S. Chen, Z. Cai, and J. Diaz-Verdejo, “LTE Phone Number Catcher: A Practical Attack against Mobile Privacy,” *Security and Comm. Networks*, 2019.
- [10] A. Shaik, R. Borgaonkar, N. Asokan, V. Niemi, and J.-P. Seifert, “Practical Attacks Against Privacy and Availability in 4G/LTE Mobile Communication Systems,” 2016.
- [11] D. Rupperecht, K. Kohls, T. Holz, and C. Pöpper, “Breaking LTE on layer two,” in *IEEE Symposium on Security & Privacy (SP)*, 2019.
- [12] N. Bui and J. Widmer, “OWL: A Reliable Online Watcher for LTE Control Channel Measurements,” in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, 2016.
- [13] R. Falkenberg and C. Wietfeld, “FALCON: An Accurate Real-Time Monitor for Client-Based Mobile Network Data Analytics,” in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019.
- [14] S. Kumar, E. Hamed, D. Katabi, and L. Erran, “LTE Radio Analytics Made Easy and Accessible,” in *Proceedings of the ACM Conference on SIGCOMM*, 2014.
- [15] N. Ludant, P. Robyns, and G. Noubir, “From 5G Sniffing to Harvesting Leakages of Privacy-Preserving Messengers,” in *2023 IEEE Symposium on Security and Privacy (SP)*, 2023.
- [16] S. Erni, M. Kotuliak, P. Leu, M. Roeschlin, and S. Capkun, “Adaptover: Adaptive overshadowing attacks in cellular networks,” in *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, 2022.
- [17] M. Kotuliak, S. Erni, P. Leu, M. Roeschlin, and S. Capkun, “LTrack: Stealthy tracking of mobile phones in LTE,” in *31st USENIX Security Symposium*, 2022.
- [18] H. Yang, S. Bae, M. Son, H. Kim, S. M. Kim, and Y. Kim, “Hiding in plain signal: Physical signal overshadowing attack on LTE,” in *28th USENIX Security Symposium (USENIX Security 19)*, 2019.
- [19] T. D. Hoang, C. Park, M. Son, T. Oh, S. Bae, J. Ahn, B. Oh, and Y. Kim, “LTESniffer: An Open-Source LTE Downlink/Uplink Eavesdropper,” in *Proceedings of the 16th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2023.
- [20] N. Ludant and G. Noubir, “SigUnder: A Stealthy 5G Low Power Attack and Defenses,” in *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2021.
- [21] M. Lichtman, R. P. Jover, M. Labib, R. Rao, V. Marojevic, and J. H. Reed, “LTE/LTE-A jamming, spoofing, and sniffing: threat assessment and mitigation,” *IEEE Communications Magazine*, 2016.
- [22] Z. Tan, B. Ding, J. Zhao, Y. Guo, and S. Lu, “Data-Plane Signaling in Cellular IoT: Attacks and Defense,” in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, 2021.
- [23] R. Sanchez-Mejias, Y. Guo, M. Lauridsen, P. Mogensen, and L. A. Maestro Ruiz de Temino, “Current consumption measurements with a carrier aggregation smartphone,” in *2014 IEEE 80th Vehicular Technology Conference*, 2014.
- [24] A. Blanco, N. Ludant, P. J. Mateo, Z. Shi, Y. Wang, and J. Widmer, “Performance Evaluation of Single Base Station ToA-AoA Localization in an LTE Testbed,” in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, 2019.
- [25] D. Forsberg, H. Leping, K. Tsuyoshi, and S. Alanara, “Enhancing Security and Privacy in 3GPP E-UTRAN Radio Interface,” in *IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications*, 2007.
- [26] R. P. Jover, “LTE security, protocol exploits and location tracking experimentation with low-cost software radio,” 2016.
- [27] Qualcomm, “QxDM Professional Qualcomm eXtensible Diagnostic Monitor,” 2022.
- [28] SRS, “Software Radio Systems. Open source SDR 4G/5G software suite,” <https://github.com/srsran/srsRAN>, 2020.
- [29] D. Rupperecht, K. Kohls, T. Holz, and C. Pöpper, “IMP4GT: IMPersonation Attacks in 4G NeTworks,” in *ISOC Network and Distributed System Security Symposium (NDSS)*. ISOC, Feb. 2020.
- [30] S. R. Hussain, O. Chowdhury, S. Mehnaz, and E. Bertino, “LTEInspector: A Systematic Approach for Adversarial Testing of 4G LTE,” in *Proceedings 2018 Network and Distributed System Security Symposium*. San Diego, CA: Internet Society, 2018.
- [31] S. Michaelides, J. Mucke, and M. Henze, “Assessing the Latency of Network Layer Security in 5G Networks,” in *Proceedings of the 18th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '25)*, New York, NY, USA: Association for Computing Machinery, 2025, pp. 262–267, doi: 10.1145/3734477.3734722.
- [32] N. Ludant, M. Vomvas, and G. Noubir, “Low-Layer Attacks Against 4G/5G Networks,” in *Proceedings of the 18th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '25)*, ACM, 2025.
- [33] X. Zhang, M. Liyanage, M. Ylianttila, and A. Gurtov, “Improving 4G/5G air interface security: A survey of existing attacks and defenses,” *Computer Networks*, vol. 212, p. 109032, 2022.
- [34] M. T. Alshurideh *et al.*, “A Systematic Literature Review of Security in 5G-based Systems,” in *Proc. 2019 International Electrical Engineering Congress (IEECON)*, IEEE, 2022, doi: 10.1109/IEECON56254.2022.9996068.
- [35] H. Wan, X. Cao, A. Marder, and K. Jamieson, “NR-Scope: A Practical 5G Standalone Telemetry Tool,” CoNEXT Companion, 2024.
- [36] J. Cao, Y. Yang, R. Ma, S. Li, and H. Li, “The Security Overview and Analysis of 3GPP 5G MAC CE,” *arXiv preprint arXiv:2506.09502*, 2025.
- [37] M. Vu, A. Kalla, and V. Kozma, “Survey on 5G Physical Layer Security Threats and Countermeasures,” *Sensors*, vol. 24, no. 17, p. 5574, Aug. 2024.
- [38] H. Rahman, M. H. M. Selim, Y. E. Baykas, and M. J. Hossain, “Jamming Attacks and Anti-Jamming Strategies in Wireless Networks,” *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2841–2887, 2022.
- [39] S. Zhang, J. Liu, Y. Wu, Z. Lin, and D. W. K. Ng, “A systematic survey on physical layer security oriented to reconfigurable intelligent surface empowered 6G,” *Comput. Secur.*, vol. 148, p. 104100, Jan. 2025.
- [40] S. R. Hussain, M. Echeverria, I. Karim, O. Chowdhury, and E. Bertino, “5GReasoner: A Property-Directed Security and Privacy Analysis Framework for 5G Cellular Network Protocol,” in *Proc. ACM Conf. Computer and Communications Security (CCS)*, 2019.
- [41] M. Akon, M. T. Toufikuzzaman, and S. R. Hussain, “From Control to Chaos: A Comprehensive Formal Analysis of 5G’s Access Control,” in *Proc. IEEE Symp. Security and Privacy (S&P)*, 2025.
- [42] Network Signal Guru, “NSG: Multi-functional Android OS tool for voice/data service QoS troubleshooting and RF optimization,” QTRUN Technologies, Android app v4.7.10, Mar. 2025.



SoK: The Constant Time Model

Billy Bob Brumley
Rochester Institute of Technology

Abstract

Constant time programming patterns is the primary defense against timing attacks on cryptographic implementations, yet what “constant time” means varies across academia and industry. This work systematizes constant time models and their evolution, identifies a recurring gap between what models protect and what specifications assume, and distills an offensive methodology for discovering timing vulnerabilities that originate outside the cryptographic primitive boundary. Applying this methodology, we locate a specification-level vulnerability related to private key loading, and confirm the leak in both OpenSSL and BoringSSL. Counterintuitively, BoringSSL’s per-observation signal is several orders of magnitude stronger than OpenSSL’s, despite an explicitly stricter threat model.

1 Introduction

Timing attacks on cryptographic implementations date to the seminal work of Kocher [33], who demonstrated that secret-dependent execution time in modular exponentiation leaks private key material. The three decades since have produced a rich body of work exploiting timing leaks across increasingly sophisticated microarchitectures, from data cache attacks on AES [11, 42] and RSA [44] to speculation attacks [32, 36] and beyond, broadening timing attacks from strictly a cryptography concept to a system security one.

The primary defense is constant time programming paradigms, ensuring that execution flow and memory access patterns are independent of secret data. In practice, what “constant time” means varies considerably. Multiple models have emerged over the past two decades, each making different trade-offs between security guarantees and practical constraints such as performance, deployability, intellectual property, and risk. The security implications of those trade-offs are not always obvious.

An orthogonal problem is that constant time models typically focus on cryptographic primitives in isolation, or “crypto in a vacuum.” Specifications that invoke these primitives (e.g.,

TLS, SSH, PKCS, X.509, etc.) routinely mandate operations on secret data that are difficult or impossible to implement in constant time. Historical examples include padding validation with error handling [17, 54], variable-length encoding of secrets [40], and key persistence formats that leak through encoding [51]. These specification-level vulnerabilities affect multiple implementations independently, not due to shared code but because they all must satisfy the same requirements to achieve interoperability.

Exacerbating the situation, compilers and hardware can silently undermine constant time programming patterns in source code. Compiler optimizations have introduced branches on secret data in Curve25519 [29] and ML-KEM [9] implementations, while variable-time instructions such as integer division affected a wide range of Kyber implementations [15]. These failures exist only in compiled binaries, invisible to source-level analysis.

Contributions. This paper makes three contributions. First, we systematize the evolution of constant time models (Section 2) and compare their features, guarantees, and documented failures (Section 3), identifying gaps between what models protect and what specifications assume. Second, we distill an offensive methodology (Section 4) from a broad analysis of specification-level vulnerability classes that provides a systematic framework for discovering constant time violations that originate outside the cryptographic primitive boundary. Third, we apply this methodology to EC private key loading (Section 5), demonstrating that both OpenSSL and BoringSSL accept non-conformant variable-length keys and that BoringSSL’s per-observation signal is several orders of magnitude stronger than OpenSSL’s, despite an explicitly stricter threat model.

Outline. Section 2 introduces the constant time models and traces their evolution. Section 3 compares these models and identifies gaps in empirical validation, compiler optimization, and at the protocol level. Section 4 develops our offensive methodology through case studies of specification-level vulnerabilities. Section 5 applies the methodology empirically to

EC private key loading. We conclude in [Section 6](#).

2 Constant Time Models: A Brief History

The evolution of constant time models in cryptography reflects the symbiotic relationship between implementers focused on efficiency and attackers discovering increasingly subtle leakage sources. What began as simple considerations for embedded systems has evolved into a complex landscape of threat models, each attempting to balance security guarantees with practical requirements. Understanding this evolution helps show why the traditional constant time model, despite numerous attempts to relax its constraints over the years, remains the gold standard for secure implementation.

2.1 The Program Counter Security Model

The earliest formal constant time considerations emerged in the context of embedded systems and smart cards, where microarchitectures were relatively simple and lacked complexity such as cache hierarchies and memory management units. At this point, seminal work by Kocher [33] in 1996 had already demonstrated the risk that timing attacks pose to implementations of various public key cryptosystems, and Quisquater and Samyde [48] in 2001 on the physical channel side for symmetric key cryptosystems on embedded devices.

The Program Counter Security Model proposed by Molnar et al. [41] formalized the approach that was being deployed in the embedded space at the time (2004–2005). The model allows an attacker access to all values of the program counter as the cryptography application executes. If this sequence is fixed and constant, the implementation should resist timing attacks, because constant program counter values leads to constant execution time in their model where all instructions have fixed latency. The authors admit this assumption is unlikely to hold outside the embedded space: “This is not true on some architectures, due to, among other things, cache effects, data-dependent instruction timing, and speculation” [41, Sec. 3].

The rationale for allowing table lookups in this model was device driven. In processors without caches, accessing any memory location required the same number of cycles. A lookup table implementation of AES S-boxes, for instance, appeared to offer constant-time execution since each table access took identical time regardless of the index. A square-and-multiply-always algorithm for modular exponentiation (RSA, Diffie-Hellman, DSA, etc.) that calls a single multiplication function for both the square step and the multiply step would not, in this model, allow attackers to distinguish between calls with distinct operands (multiply) and calls with identical operands (square). This model assumes that the primary observable characteristic was branching and execution time at the instruction level, not the pattern of memory addresses (containing data, not code) being accessed.

However, this simplistic model failed to account for several crucial factors, including, for example, the observation by Kelsey et al. [30, Sec. 5]: “We believe attacks based on cache hit ratio in large S-box ciphers ... are possible.” This is not a failure in the model design per se, as a major goal of the model was to reflect industry practice at the time. As these implementations were ported to more complex processors, the security assumptions of the Program Counter Security Model began to break down.

2.2 The Traditional Constant Time Model

The traditional constant time model emerged gradually through the work of several researchers, rather than being defined in a single seminal paper. The model took form through a series of contributions that each highlighted different threats not accounted for in the program counter model.

The 2002–2003 work by Page [43] on the theoretical side and Tsunoo et al. [53] on the empirical side showed that secret-dependent lookups from memory-resident tables in DES implementations leads to key recovery attacks through cache timings. This eventually fed into the 2004–2005 work by Bernstein [11], a contribution on the offensive side with a data cache-timing attack on AES, but also on the defensive side with guidelines for implementers of cryptosystems. Bernstein warned from the data perspective: “Using secret data as an array index is a recipe for disaster” [11, Sec. 1]. Even if less relevant for AES with its largely straight-line structure, Bernstein furthermore warned from the code perspective and input-dependent branches: “Skipping an operation is faster than doing it” [11, Sec. 9]. These two principles eventually became the core pillars of the traditional constant time model.

In parallel work, Osvik et al. [42] introduced their Prime+Probe and Evict+Time microarchitecture timing attack techniques, applied to AES implementations and targeting data caches. The authors give similar warnings as Bernstein regarding table lookups and conditional branches, an example of the latter being with RSA “when exponentiation is performed using a precomputed table of small powers” [42, Sec. 6]. Indeed in more parallel work, Percival [44] carried out such an attack on RSA, an L1 data cache-timing attack on Intel HyperThreading architectures. Percival concludes with similar mitigation strategies for cryptography applications: “the code path and sequence of memory accesses are oblivious to the data and key being used” [44, Sec. 7].

In summary, the combined results and recommendations of Bernstein [11], Osvik et al. [42], and Percival [44] led to the traditional constant time model as we know it today. In this light, these recommendations culminate in the 2006 design of Curve25519, which Bernstein states “avoids all input-dependent branches, all input-dependent array indices, and other instructions with input-dependent timings” [12, Sec. 1]. Subsequent work supports the model and continues to make it explicit. For example, the NaCl library has the design

goal of “No data flow from secrets to load addresses” and “No data flow from secrets to branch conditions” [14, Sec. 3].

The traditional model thus operates under a simple but powerful principle: the sequence of memory addresses accessed during cryptographic operations must be identical regardless of secret values. This includes both data addresses (no secret-dependent array indexing) and code addresses (no secret-dependent conditional branches). From the offensive perspective, model-wise the attacker sees all addresses accessed while the cryptography application executes, both data and code. As we later discuss in Section 3, modern compilers have become quite skilled in recognizing related constant-time programming patterns and instead emitting branches (see [29] from 2016 and [47, Sec. 2] for a wider 2025 survey), and the model does not consider the latency of instructions being potentially operand dependent, both of which can violate the guarantees of the traditional constant time model.

2.3 The Intel Scatter-Gather Model

Percival’s work resulted in CVE-2005-0109, where Intel contributed a patch¹ to OpenSSL (0.9.7h, October 2005) for hardening against timing attacks in the case of modular exponentiation. The case is non-trivial, since high-speed implementations of Diffie-Hellman, RSA, and DSA typically use a square-and-multiply variant of modular exponentiation with memory-resident table lookups, costly to accomplish in constant time w.r.t. the traditional model. Intel introduced a new software implementation strategy, leveraging “scatter-gather” to transpose data across cache lines (scatter) then assemble the data with a series of secret-dependent table lookups (gather) that have lower than cache line granularity. Data cache attacks at the time were only exploiting cache set access, with no published method to determine access within a single cache line (intra-cache-line access).

However, this approach flouts the traditional constant time model and disregards previous advice explicitly disallowing intra-cache-line lookups. All three of the seminal works that led to the traditional constant time model warn against such behavior. Percival calls it “dangerous at best” [44, Sec. 7]. Bernstein documents the related cache bank problem [11, Sec. 15], and Osvik et al. worry it “leaks information on memory accesses with resolution better than [a cache line]” [42, Sec. 3.7].

Undeterred by academic advice, Brickell [20] (Intel) explicitly articulated the security model in an invited talk at CHES 2011, stating that Intel recommends “No secret key or data dependent: memory access (at coarser than cache line granularity), code branching.” Yet Intel continued to develop and push the scatter-gather technique, for example, 2012 work by Gueron [23] (Intel) on efficient modular exponentiation.

To summarize the Intel model, the fundamental assumption is that if all accesses fall within the same sequence of cache

lines, the cache behavior, and thus the timing, will be identical regardless of the specific location accessed within those lines. The scatter-gather technique organizes data so that all possible accesses for a given operation map to the same sequence of cache lines, ensuring uniform cache behavior. As techniques to glean intra-cache-line timings via cache bank conflicts later emerged [13, 59], these days the scatter-gather technique is no longer en vogue and the Intel model is understood to be significantly weaker than the traditional constant time model.

2.4 The OpenSSL Security Policy

The OpenSSL project has developed a pragmatic threat model that focuses on the capabilities of attackers rather than properties of the source code. Prior to the Heartbleed vulnerability in 2014 (CVE-2014-0160), OpenSSL had no explicit release strategy or security policy. The catastrophic impact of Heartbleed, a simple out-of-bounds read that exposed sensitive data from millions of servers, forced a fundamental reassessment of the project’s approach to security [55].

Until 2019, OpenSSL would selectively issue CVEs for various types of timing side channels (excluding physical attacks such as power analysis [34], electromagnetic analysis [48], fault injection [16], hardware faults [31], data remanence [24], etc.), including those requiring colocation, such as the CacheBleed attack [59] exploiting cache bank conflicts targeting “constant time” RSA in OpenSSL (CVE-2016-0702), a Flush+Reload attack [58] on “constant time” P-256 arithmetic [45] in OpenSSL (CVE-2016-7056), or Controlled Channel attacks [57] on DSA and ECDSA [56] in OpenSSL (CVE-2018-0734, CVE-2018-0735). However, the disclosure process was subjective and based on undocumented aspects such as perceived risk. This evolved further with the advent of speculative execution attacks like Spectre [32] and Melt-down [36] in 2018, leading to the current security policy² that explicitly defines which attack classes warrant CVEs and security bulletins.

The current policy attempts to strike a balance between security and practicality. OpenSSL recognizes that all side channel threats are important, but issuing CVEs requires significant project resources and creates downstream burdens, particularly for FIPS-certified versions that cannot be easily modified. Their model therefore prioritizes attacks that can be executed by less capable attackers, i.e., those that can be mounted remotely or with limited access and resources. The change in fact represented a major shift in the security policy of the project, since attacks requiring colocation are not purely remote, therefore no longer captured by the model. This excludes the vast majority of microarchitectural attacks that require local spy processes, starting from the seminal work by Percival [44].

¹<https://github.com/openssl/openssl/commit/ecb1445ce2df>

²<https://openssl-library.org/policies/general/security-policy/>

To summarize, OpenSSL explicitly excludes certain attack classes from warranting security bulletins and CVEs, including same-physical-system side channels requiring colocation, and physical side channels such as those mentioned above. The model assumes the offensive role and focuses on how attackers gather data and potentially exploit vulnerabilities.

3 Analysis: Constant Time Models Compared

The evolution of constant time models highlights the struggle between theoretical security guarantees and practical implementation constraints. Each model makes different trade-offs, and history has shown that deviations from the most conservative approach consistently introduce vulnerabilities. [Table 1](#) systematizes these features, guarantees, constraints, trade-offs, deviations, and risks. This section consists of a discussion centered around the contents of [Table 1](#).

3.1 Model-Specific Analysis

Program Counter. This model assumes that instruction timing is predictable and data-independent. While largely suitable for simple embedded systems without caches, the model immediately fails when applied to large classes of modern processors. Three seminal attacks invalidated this model: (i) the AES data cache attack by Osvik et al. [42], demonstrating that data cache attacks could reveal the complete granular memory access pattern to S-boxes through timing analysis; (ii) the RSA data cache attack by Percival [44], a practical attack on HyperThreading architectures leading to complete private key recovery via highly granular L1 data cache timing data; (iii) the AES data cache attack by Bernstein [11], a purely statistical attack focusing on threat models involving remote attackers with only coarse timing data. The fundamental flaw in the model was defining table lookups with secret indices as safe, an assumption that fails when applied to the vast majority of COTS processors today that typically feature several layers of cache memory. The model’s validation approach of analyzing PC sequences indeed allows verification that no secret-dependent branches occur, yet fails to capture the threat of secret-dependent data access.

Traditional. The main focus of the traditional constant time model is on the source code itself. The model considers only a single seemingly simple requirement: all addresses are public, the consequence being that all code must ensure no branching on secrets and no data access using secret indices. These features are typically accomplished with a combination of side channel friendly algorithm selection and (now) well-understood constant time programming paradigms. Regardless of its theoretical soundness, the traditional model faces many practical challenges.

First, compiler optimizations can silently counteract constant time programming patterns. Work by Kaufmann et al.

[29] demonstrated how constant time source code for Curve25519 became variable time after compilation. These implementations contain several multiplications with public curve-related constants, and Microsoft’s MSVC 2015 toolchain expanded one such multiplication by a 32-bit constant inside a subroutine using a branch on the sign of the second operand, since the upper 32 bits of said constant are known to be zero a priori. The authors outline a method for complete private key recovery leveraging this leak, using modest computation power.

Second, certain instructions have input-dependent timing. An excellent example is early-termination multipliers on ARM processors [22]. Multiplication unit designs in older 32-bit ARM chips leveraged smaller parallel multipliers that would exit early when encountering leading zero bytes in one of the operands. This meant the latency of a multiplication instruction would vary between 2–5 clock cycles, depending on the number of leading zero bytes. A more recent example is the KyberSlash attack [15]. Many Kyber implementations feature an integer division by a small public constant, and expect the compiler to either replace that with a multiplication by a constant or a small number of integer shifts and integer additions. Yet some compilers with appropriate optimization flags would faithfully emit an integer division instruction, which is notoriously variable time on Intel architectures. KyberSlash was both timely and impactful, discovered around the time of NIST standardization and applying to a large portion of software implementations, including the reference implementation by the Kyber authors themselves. It continues to resurface, e.g., [CVE-2026-22705](#) in RustCrypto’s ML-DSA implementation.

In summary, despite its age, the traditional constant time model has stood the test of time and proves to be the de facto standard for secure implementations.

Intel Scatter-Gather. Intel’s constant time model attempts to balance security and performance. It is conceptually similar to the traditional constant time model, yet allows data flow from secrets to addresses at cache line granularity [20]. Put another way, in the traditional model where all addresses are public, in the Intel model all addresses sans the least significant $\log_2(\delta)$ bits are public (where δ is the number of bytes in a cache line). This relaxation was based on the assumption that accesses within the same cache line would have fixed timing. At least three works demonstrate concrete failure of the Intel model.

First, Bernstein and Schwabe [13] presented empirical data that showed data-dependent timing variations by artificially inducing cache bank conflicts. While the experiment in question targeted strawman code and not actual cryptography code, the fundamental microarchitectural effect was clear from the empirical data.

The CacheBleed attack built on that proof-of-concept soon thereafter [59]. The code targeted by CacheBleed was in fact a contribution from Intel as a countermeasure in OpenSSL to the attack by Percival [44]. This scatter-gather code used

Table 1: Constant time models compared across rules, scope, validation, and documented failures. Every relaxation of the traditional model has eventually been exploited.

Model	Year	Memory Access Rules	Control Flow Rules	Protocol-Layer Scope	Validation Approach	Why it Failed
Program Counter	2004–2005	Allows data flow from secrets to load addresses	No data flow from secrets to branch conditions	Crypto primitives only	Differential analysis of PC sequences	Data cache timing attacks
Traditional CT	2004–2006	No data flow from secrets to load addresses	No data flow from secrets to branch conditions	Crypto primitives only	Static code analysis	Compiler optimizations, variable-time instructions
Intel Scatter-Gather	2005	Allows data flow from secrets to load addresses (within cache lines)	No data flow from secrets to branch conditions	Crypto primitives only	Microarchitectural assumptions	Cache bank conflicts (CacheBleed)
OpenSSL	2019	Context-dependent restrictions	Context-dependent restrictions	Protocol aware	Threat modeling, empirical data	Limited threat model scope
BoringSSL	2025	Allows data flow from secrets to load addresses (variable-length inputs)	Allows data flow from secrets to branch conditions (variable-length inputs)	Crypto primitives only	Static code analysis, formal methods	This work

secret-dependent table lookups within cache lines during modular exponentiation, which the CacheBleed attack exploited due to timing variations caused by cache bank conflicts. The authors were able to recover an RSA private key using a modest number of signature queries.

Lastly, even secret-dependent control flow within a cache line is not a secure strategy. The PortSmash attack (CVE-2018-5407) exploited port contention in Simultaneous Multithreading (SMT) architectures (such as Intel’s Hyper-Threading) to leak control flow [5]. One of the experiments the authors conducted targets code with multiple branches within a single cache line, and PortSmash is still able to distinguish between the branch targets [5, Sec. 4].

In summary, cache bank conflicts prophetically invalidated Intel’s constant time model for data, and port contention accomplished the same for code. The Intel model nevertheless serves as an exemplary case study of why deviating from the traditional constant time model carries significant risk.

OpenSSL. OpenSSL’s pragmatic, data-driven approach focuses on attacker capabilities and empirical data rather than source code analysis. Hardware-based side channels and any attack requiring co-location (e.g., most microarchitecture attacks) are out of scope. With these restrictions, OpenSSL is hoping to exclude local attackers from their model, yet retain remote attackers. This model fails in predictable ways (e.g., the vast majority of cache attacks), yet also fails in surprising ways, discussed below.

Weiser et al. [56] discovered several vulnerabilities in multiprecision integer arithmetic libraries used by cryptography libraries. Their “V8” vulnerability applied to a division function as part of the extended Euclidean algorithm, their main application being to DSA signing. The leak was due to data structure resizing, and was limited to a single top-most bit of the secret. Their “V9” vulnerability is in the same algorithm, a conditional negation that leaks the top-most bit of a secret inverse. These vulnerabilities are outside the scope of the OpenSSL security policy because (i) they are unlikely to be observable remotely; and (ii) it is unclear from the theory perspective how a remote attacker can effectively utilize such a small leak. Hence, while the vulnerabilities are certainly valid from the academic perspective, OpenSSL’s security policy discards them.

A second example is the obscure key formats studied by Pereida García et al. [46]. The code loading these keys neglected to configure the private keys to ensure constant time code paths within the library, which led to a single trace private key recovery attack with local L1 data cache timings. While that is clearly outside OpenSSL’s model, the same leak revealed information about the top set bit of the private key, potentially measurable in remote scenarios. Yet it is again unclear from the theory perspective how a remote attacker can effectively utilize such a small leak on fixed bits. Hence, while this vulnerability is certainly valid from the academic perspective, it is out of scope for OpenSSL’s security policy.

A third example is the LadderLeak attack by Aranha et al. [8], capable of recovering ECDSA private keys with less than one bit of nonce leakage per signature, but requiring local cache timings to detect branch targets. Coupled with the large data requirement of the attack (needing approximately 2^{33} signatures for key recovery), LadderLeak falls outside OpenSSL’s security policy, requiring local access and significant data, time, and resources. Yet again, this attack is certainly valid from the academic perspective.

A final and recent example is the SLasH-DSA attack by Boy et al. [19], a fault attack on hash-based PQC as implemented in OpenSSL 3.5. The authors utilize the RowHammer technique [31] to induce memory read faults while querying digital signatures. Hardware faults are indeed explicitly excluded in OpenSSL’s security policy, and furthermore RowHammer requires local access. Yet again, this attack is certainly valid from the academic perspective.

An interesting model-related case study is the OpenSSL vulnerability linked to CVE-2025-27587, which led to conflict between OpenSSL and the wider security community. The vulnerability leaks part of the ECDSA nonce in OpenSSL’s implementation on PowerPC architectures, yet the timing distributions are highly overlapping and it remains unclear how to separate them. Furthermore, the experiment data is with local timings, while remote timings exacerbate the situation from the attacker perspective, increasing the statistical overlap. OpenSSL typically manages its own CVEs, but in this case MITRE chose to issue its own against OpenSSL, who disputed the vulnerability because it fell outside their written security policy, for similar reasons as the LadderLeak

attack.

In summary, OpenSSL’s model has the best and worst of both worlds. On one hand, it flippantly excludes large classes of attacks that the security community historically recognizes as valid, i.e., requiring co-location and local execution. On the other hand, their data-driven approach potentially scopes in threats that are typically not captured in other models, where “crypto in a vacuum” is the norm, source code analysis stops at the low-level crypto, and vulnerabilities in callers to the crypto are another layer’s problem, even if attackers care little about such semantics. We elaborate in the next subsection.

BoringSSL. We defer this discussion to the end of the paper.

3.2 Constant Time Models: Gaps

Following the previous analysis surrounding [Table 1](#), in this subsection we expand the discussion to consider areas where the models may not perform as expected. These issues are pervasive, and represent a security concern for all of the models.

Protocol-layer leakage. A critical limitation of most constant time models is their focus on cryptographic primitives and low-level arithmetic in isolation, ignoring how these primitives behave when integrated into protocols. This gap has led to numerous attacks that bypass even high-assurance and high-security cryptography implementations by exploiting protocol-level leaks. While we expand later in [Section 4](#), here are a few seminal works that highlight this gap.

The Million Message Attack (MMA) by Bleichenbacher [17] demonstrated that implementing RSA PKCS#1 v1.5 (RFC 2313 [27], 1998) carries significant security risk. Implementations tend to leak whether decryptions produce valid padding or not, either directly through error messages or indirectly through timing. These leaks enable adaptive chosen ciphertext attacks to decrypt without the private key. Manger [39] subsequently showed that RSA-OAEP and PKCS#1 v2.0 (RFC 2437 [28], 1998), designed to resist MMA, remained vulnerable to similar oracle attacks through timing leaks in their more complex padding validation.

On the symmetric side, Vaudenay [54] targeted implementations of CBC mode for block ciphers with PKCS#7 padding, where timing differences in padding validation expose the plaintext. While Vaudenay described several different oracles, the general area of padding oracles emerged only later, encompassing both the attacks by Bleichenbacher and Vaudenay under the general umbrella that validating padding applied to plaintext is a dubious security practice. The work extends naturally to hardware implementations [10].

These seminal results reveal a fundamental class of vulnerabilities: protocols requiring conditional processing of padding or error conditions create timing channels that propagate beyond the cryptographic primitive layer.

Empirical validation. Consulting the rightmost column of

[Table 1](#), the failures across all models highlight the critical importance of empirical validation, both through direct measurement of execution latency on target hardware and through analysis of compiled binaries. Even when source code follows constant-time programming practices, the underlying hardware may still exhibit input-dependent timing variations, and mainstream compilers actively undermine constant time guarantees.

The tool *dudect* [49] takes a purely empirical approach, which utilizes classical leakage detection methods. It executes the target function with two classes of input data, collects cycle-accurate timings, and applies statistical tests to determine whether the resulting distributions are distinguishable. Because this approach requires no hardware model and no source code annotations, any timing variation observable on the target platform can in principle be detected, provided the effect is large enough relative to measurement noise. However, the tool cannot pinpoint the source of a detected leak, and subtle violations may require millions of measurements to reach statistical significance.

Constant time violations can originate in the toolchain rather than the hardware. The Clangover vulnerability (CVE-2024-37880, Antoon Purnal) in the ML-KEM reference implementation demonstrates a case where the source code used a constant-time conditional assignment, but modern Clang compiled it into a conditional branch on secret data [9]. The vulnerability is particularly stealthy because source-level analysis tools correctly report the code as constant-time, since the violation exists only in the compiled binary. LLVM recently implemented support for source code annotations that preserve select constant-time properties through optimization passes³, demonstrating that compiler designers are aware of the issue, yet the narrow scope and opt-in nature make this approach inherently fragile. Arranz-Olmos et al. [9] further showed that popular decompilers compound the problem by performing optimizations that eliminate constant time violations during analysis, causing decompile-then-analyze approaches to miss vulnerabilities that exist in deployed binaries. In summary, this is a challenging compiler security gap since developers write correct constant-time code, but the toolchain silently transforms it into vulnerable binaries, and standard analysis techniques fail to detect the resulting violations.

To approach this challenge, TIMECOP uses Valgrind’s Memcheck infrastructure to detect potential constant-time violations through dynamic binary analysis. It marks secret data as uninitialized via Valgrind client requests, causing Memcheck to flag any secret-dependent branch conditions or memory accesses encountered during execution. Because this analysis operates on compiled binaries, it can detect violations introduced by compiler optimizations that are not visible at the source level. TIMECOP originated as prototype patches

³<https://blog.trailofbits.com/2025/12/02/introducing-constant-time-support-for-llvm-to-protect-cryptographic-code/>

to the SUPERCOP benchmarking framework, with a leakage model limited to branching and memory accesses. From the other direction, patches to Valgrind added support for detecting potential variable-time instructions operating on secret data. Finally, the KyberSlash work [15] combines both approaches to allow the analysis to scale by scoping in more ISAs, extending the list of potentially vulnerable instructions, and enabling tests across a wide spectrum of implementations and architectures.

In summary, the two approaches complement one another, with duedect analyzing end-to-end timing behavior on real hardware, while binary analysis tools like TIMECOP pinpoint the location of the leak.

4 An Offensive Security Methodology

Most of the constant time models in Section 3 stop at the crypto layer, yet the protocols using these primitives often operate on secret data well beyond that boundary. Specifications, standards, and protocols are typically written with correctness and interoperability as primary goals, not constant time execution. They often require operations that create natural timing channels, such as padding schemes with conditional processing, error handling with different code paths, and variable-length encodings. No implementation can satisfy both the specification and constant time requirements, because the specification itself mandates contradicting behavior.

This section explores those gaps through various case studies, culminating in an offensive methodology which identifies standards whose flexibility or complexity creates said conflict, then tests implementations for any leaks predicted by the analysis. We organize the case studies by the specification at fault, not by the models from Table 1. Which constant time model a given attack violates depends on the threat model, while the specification does not. Each case below is an instance of protocol-layer leakage.

4.1 Case Studies

Specifications have been a steady source of timing leaks since (at least) the seminal attacks discussed in Section 3, with new instances appearing across protocols and implementations in the decades since. Standards and specifications play a significant role in proliferating these vulnerabilities, and in this section we investigate that relationship through several recurring cases (e.g., error handling, padding validation, key persistence formats, encoding of secrets) where compliant implementations undermine primitive-level constant time guarantees with system-level violations.

Public key padding oracles. As previously discussed, the way that PKCS#1 v1.5 (RFC 2313 [27]) specifies the handling of malformed ciphertexts lends itself to vulnerability, first exploited by Bleichenbacher [17] in 1998. Each fix has

produced a new variant, several of those which we now discuss.

Jager et al. [26] observed that the XML encryption standard specified its own error handling requirements for RSA PKCS#1 v1.5, independently recreating the same class of oracle in a completely different protocol (CVE-2011-2487). This is despite the W3C having published the specification in 2002, four years after Bleichenbacher’s original disclosure.

Kupser et al. [35] subsequently automated the attack, finding that several implementations (CVE-2015-0226, etc.) remained vulnerable even after deploying mitigations. The authors circumvent fixes via wrapping techniques that relocate encrypted elements within the XML document tree to bypass integrity checks.

The ROBOT attack [18] (CVE-2017-6168, etc.) demonstrated that even the TLS 1.2 specification’s own countermeasure against Bleichenbacher was itself a source of new oracles. The mitigation requires servers to generate a random premaster secret on padding failure and continue the handshake. The work highlights that ensuring the success and failure codepaths are indistinguishable is indeed difficult to achieve in constant time.

Ronen et al. [50] identified nine distinct attack variations across the TLS ecosystem, each exploiting different leakage sources in implementations with mitigations (CVE-2018-12404, etc.). Each fix to a specific Bleichenbacher variant introduced new, subtly different oracles exploited by the authors. This shows that when a specification is fundamentally incompatible with constant time goals, patches merely shift the leak.

Symmetric key padding oracles. The 2002 padding oracle attack by Vaudenay [54], mentioned in Section 3, is the symmetric key counterpart. The specification-level root cause (protocols that require the receiver to act on plaintext properties before integrity can be verified) has proven equally persistent, with instances across SSH, XML encryption, and TLS.

Albrecht et al. [3] analyzed the SSH Binary Packet Protocol (BPP, RFC 4253 [38]), finding that it encrypts the packet length field within the first CBC ciphertext block (CVE-2008-5161, etc.). The receiver must decrypt this block to see how many additional bytes to read. The specification makes it impossible to authenticate the MAC before parsing this field, a parallel to the (legacy) TLS MAC-then-encrypt ordering. Observing how many bytes the server reads before responding allows an attacker to infer the decrypted length field, yielding a practical chosen-ciphertext attack enabled through the standard.

Jager and Somorovsky [25] showed that the XML encryption specification’s use of CBC mode created a similar oracle, exploiting the server’s error handling when decrypted plaintext contained malformed character encodings. This vulnerability (CVE-2011-1096) originated from the XML parsing layer rather than cryptographic padding validation.

Previously mentioned in the public key context, Jager et al. [26] further showed that XML encryption combines the two oracles on the symmetric key and public key sides. CBC malleability allowed an attacker to test candidate RSA session keys by observing whether the resulting symmetric decryption produced valid plaintext, enabling Bleichenbacher attacks without relying on error messages or timing from the RSA layer itself.

The Lucky Thirteen attack [6] (CVE-2013-0169) extended the CBC work to (pre 1.3) TLS with a timing attack on the MAC computation itself. The number of hash iterations depends on the data length after stripping padding (mandated by the specification), leaking whether a modified ciphertext contained valid padding through timing. The attack affected virtually every TLS implementation because the leak originated in the specification’s ordering of operations, not in any particular cryptographic primitive. Apecechea et al. [7] subsequently showed that cache-based timing attacks could amplify these differences with local access.

Encoding leakage. Beyond padding validation and error handling, specifications that define key persistence formats are another source of timing leaks. Encodings such as hex and base64 (required by any library that loads cryptographic material from PEM files or similar formats) typically use a lookup table indexed by the ASCII value of each input character, easily leading to secret-dependent memory access, and furthermore entirely outside the constant time cryptographic primitive boundary.

Sieck et al. [51] demonstrated this concretely, showing that base64 decoding in OpenSSL, mbedTLS (CVE-2021-24119), Botan (CVE-2021-24115), Mozilla NSS, and GNU Nettle used secret-dependent memory accesses when loading RSA private keys from PEM files. Observing a single PEM key loading event allowed partial recovery of RSA private key material, leakage that no amount of constant time arithmetic in the cryptographic primitive layer can mitigate. Subsequent work increased the leakage with finer spatial resolution through advanced microarchitectural techniques [52].

Notably, BoringSSL had proactively addressed this leak in 2017 before either attack was published, implementing constant time base64 decoding by replacing table lookups with arithmetic range comparisons. Quoting the commit message:⁴ “PEM files sometimes carry private keys so, in principle, we’d probably prefer not to leak the contents when we encode or decode them.” BoringSSL chose to extend constant time techniques to key decoding well before any published attack, acknowledging that decoding can leak secret material and applying meaningful mitigations beyond the cryptographic primitive boundary.

Canonicalization leakage. Beyond character-level encoding, some specifications also mandate canonical representations for secret values. This might include stripping leading ze-

ros, conditionally prepending sign bytes (e.g., for DER), or enforcing a particular integer representation. Compliant implementations would then perform variable-length processing on secret data, potentially leaking the length of the secret (confidential in many cases).

A good example is the SSH protocol (RFC 4253 [38], discussed earlier in the separate context of BPP), where the standard mandates that the Diffie-Hellman shared secret be represented as an MPINT. This data type is defined in RFC 4251 [37] with the explicit requirement that “unnecessary leading bytes with the value 0 . . . MUST NOT be included.” RFC 8731 [2] (which defines Curve25519 and Curve448 key exchange for SSH) explicitly acknowledges this weakness in its Security Considerations section: “The way the derived MPINT binary secret string is encoded before it is hashed (i.e., adding or removing zero bytes for encoding) raises the potential for a side-channel attack.”

The Raccoon attack by Merget et al. [40] exploited similar canonicalization in (pre 1.3) TLS, where DH(E) cipher suites require stripping leading zeros from the pre-master secret before key derivation. The attack affected several security-critical libraries (CVE-2020-1968 in OpenSSL, CVE-2020-12413 in Mozilla NSS, etc.), not due to shared code, but because they implemented the same specification. Subsequent work [4] demonstrated that microarchitectural timings from local attackers could amplify the leakage, extending it to CMS and S/MIME contexts. The Raccoon authors confirmed that SSH suffers from the same specification-level flaw (discussed previously), but could not exploit it due to SSH’s strict use of fresh keys per session [40, Sec. 8]. Exploitability depends on the protocol context, even when the underlying specification-level vulnerability is identical.

4.2 The Offensive Methodology

In each of the previous case studies, the specification requires an operation on secrets that is inherently difficult to realize in constant time. Implementations that comply for interoperability leak in ways that no amount of constant time arithmetic or programming paradigms in the cryptographic primitive layer can prevent. We now distill this into a methodology.

Step 1: Identify specs with problematic operations. Locate operations a specification requires on secret data that conflict with constant time goals. For example, padding schemes with conditional processing (PKCS#1 v1.5), error conditions that mandate distinguishable responses (CBC), variable-length encodings (ASN.1, DER, PEM), canonicalization of keys (byte stripping), etc.

Step 2: Analyze implementation strategies. Investigate how systems handle these problematic operations. For example, they might knowingly tolerate the leak, deploy countermeasures that might expose corner cases, or compromise on compatibility to achieve better timing attack resistance. Implementation strategies may also vary across architectures and

⁴<https://boringssl.googlesource.com/boringssl/+536036ab>

compiler toolchains, so the same code can exhibit different leakage characteristics depending on the target platform.

Step 3: Confirm the analysis. Run experiments against these implementations to confirm or refute the predicted leak. The systematization in [Section 3](#) suggests that multiple analysis techniques apply in tandem. The scope, validation, and failure considerations in [Table 1](#) highlight static methods, dynamic analysis, and targeted timing measurements as examples of analytical methods. Casting a wide net allows leveraging a fuller spectrum of evidence, from more formal techniques all the way through empirical timings.

In summary, different implementations of the same specification tend to leak in similar ways, not because they share code, but because they share the specification. Diverse analysis methods complement one another, and our methodology is sound largely due to inheriting more of the strengths of these methods and less of the inherent limitations. We validate our methodology in the remainder of this paper.

5 Analysis: Private Key Loading

Applying said methodology, this case study examines how a seemingly minor specification detail, the encoding of elliptic curve private keys, creates systematic timing leaks across implementations. By tracing the evolution from early standards through modern implementations, we demonstrate how specification-level decisions made decades ago continue to create meaningful timing leaks today.

5.1 Elliptic Curve Cryptography Standards

The encoding of elliptic curve private keys dates to the Standards for Efficient Cryptography Group’s specification SEC 1 [1], published in 2000, which defined the `ECPrivateKey` syntax with the private key encoded as a fixed-length octet string derived from the Integer-to-Octet-String-Primitive (I2OSP). [RFC 5915](#) [21] later echoed this requirement, specifying that the `privateKey` field must be of length $\lceil \log_2(n)/8 \rceil$ bytes, where n is a public constant (the curve order). For example, with the NIST standard curves P-256, P-384, and P-521, this mandates exactly 32 bytes, 48 bytes, and 66 bytes, respectively, regardless of the numeric value of the scalar, and any private key with a small value must be zero-padded to the full length. This fixed-length requirement happens to enable constant-time processing, as decoders can unconditionally read a publicly known number of bytes and convert them to an internal BIGNUM representation using fixed-width operations.

In practice, implementations are not always so rigid. The `privateKey` field is a DER-encoded OCTET STRING, but the value it carries is semantically an integer, and internally many implementations will convert it to a BIGNUM. DER encoding of INTEGER values requires minimal-length representation, and per X.690 rules, the output must use the smallest

number of octets needed, canonicalization that strips leading zero bytes. While OCTET STRING and INTEGER are distinct ASN.1 types with different encoding rules, the boundary might blur in practice. BIGNUM libraries often strip leading zeros on import regardless of the source type, in which case subsequent serialization routines naturally produce minimal-length output. Ecosystem-wise, the result would be a population of existing keys with private key fields shorter than the specification mandates, encoding the effective (bit, byte) length of the private key (secret integer scalar) directly in the serialized representation. Implementations could choose to reject these non-conformant keys, yet lessons from padding oracle attacks tell us this is more difficult to do securely than it might seem.

Applying the [Section 4](#) methodology, the first step is to confirm that the target implementations accept non-conformant keys at all. We generated P-384 private keys with effective bit lengths ranging from 128 to 384 in multiples of 8, encoding each as a truncated (leading-zeros-stripped) octet string within an otherwise valid `ECPrivateKey` structure. Both OpenSSL and BoringSSL accept every such key without error, confirming that the threat surface exists and justifying deeper analysis of the loading paths.

In summary, this suggests canonicalization leakage (previously discussed in [Section 4](#)), where a specification defines a fixed-length format, but implementations that tolerate variable-length inputs potentially introduce vulnerabilities outside the cryptographic primitive boundary.

5.2 Implementation Analysis

Having confirmed that both implementations accept variable-length keys, we now examine their respective loading paths to understand where secret-dependent processing occurs. We also discuss adversarial scenarios and the high level impact to security.

OpenSSL. EC private key loading follows the path `d2i_ECPrivateKey` $\rightarrow$ `EC_KEY_oct2priv` $\rightarrow$ `ossl_ec_key_simple_oct2priv`. Two operations process the variable-length private key material. First, OpenSSL’s ASN.1 runtime (`d2i_EC_PRIVATEKEY`) decodes the DER structure, populating an `ASN1_OCTET_STRING` whose length field reflects the encoded size of the private key. For a non-conformant key this is shorter than the curve order, and the parser performs correspondingly less work. Second, `ossl_ec_key_simple_oct2priv` passes that variable-length buffer and its length directly to `BN_bin2bn` to construct a BIGNUM holding the private key, which is not a constant-time function. It processes exactly the number of bytes present, so the amount of work varies with the encoded length of the private key.

BoringSSL. BoringSSL’s loading path is architecturally similar. The function `ec_key_parse_private_key` handles both

ASN.1 parsing and key construction in a single routine, using BoringSSL’s CBS (“crypto byte string”) interface for DER decoding. As with OpenSSL, two operations process the variable-length private key material. First, `CBS_get_asn1` extracts the OCTET STRING from the DER structure, parsing a TLV whose length field encodes the size of the private key material. For a non-conformant key, this length is shorter than the curve order, and the parser performs correspondingly less work. Second, the extracted bytes are passed to `BN_bin2bn` to construct a BIGNUM holding the private key, which as in OpenSSL processes exactly the number of bytes present with no padding to a fixed width. Both leakage points (the ASN.1 length parsing and the BIGNUM conversion) operate on the actual encoded length rather than the expected (public) curve order length.

Security implications. Some services amortize the cost of key loading using daemons that load keys once at startup and serve from memory. Others load per connection, a good example being the classical Unix service dispatcher `inetd` (“internet service daemon”) where (typically) each incoming connection spawns a fresh process that parses its keys before serving the request. In the latter case, an adversary can force an arbitrary number of key load observations simply by opening connections.

For this leak, increasing samples does not provide new information in an unbounded way. The leak reveals fixed information, the byte length of the encoded scalar, which is a property of the long-term private key, not the session. Additional queries only increase confidence. To give a worked example, probabilistically $1/256$ keys will be a single byte shorter than a full-length key, and in that case it leaks 8 bits (“the top byte is zero”). So while this leak indeed leads to an information-theoretic reduction in confidentiality, it cannot be used to extract a growing fraction of the fixed secret.

The utility of the leak from the adversarial perspective is also dubious. Continuing the worked example, any cryptanalytic attack that benefits from a short fixed scalar can simply assume one and proceed, without any leak or side channel. Such an attack succeeds $1/256$ of the time. While the leak is real and any model violations are real, we are not aware of any exploit, and see little to no practical relevance in the leak.

5.3 Results

We constructed a harness that loads each PEM-encoded key file into memory once, then repeatedly parses it from an in-memory buffer. The timed region brackets a single call to `PEM_read_bio_PrivateKey` (PEM base64 decoding, DER parsing, and key object construction), with no disk I/O in the measurement path. The harness runs on a quad-core Intel i5-7500T at 2.70GHz with frequency scaling disabled, equipped with 16GB of DDR and running Debian 12.13 “bookworm.” This CPU does not feature HyperThreading. It loads each key 1,000 times, each load timed in CPU clock cycles. We linked

Table 2: Statistical comparison of key load timing leakage for OpenSSL, BoringSSL (unpatched and hedged), NIST curve P-384. MI and permutation z -scores characterize the channel; Cohen’s d and Bayesian sample counts characterize the adversary’s decision problem under a 255/256 prior for full-length keys.

Metric	OpenSSL (stock)	BoringSSL (stock)	BoringSSL (patched)
MI, 33-class (bits)	0.0031	0.6104	0.0055
MI permutation z -score	249	57,861	434
Cohen’s d , binary	0.098	1.11	0.008
Binary MI (bits)	0.0003	0.037	0.0006
Binary MI z -score	151	19,021	250
Samples to $P > 0.5$	$\approx 1,000$	≈ 10	$\approx 100,000$
Samples to $P > 0.999$	$\approx 10,000$	≈ 50	$\approx 1,000,000$

the same harness against both OpenSSL and BoringSSL, using a corpus of P-384 keys at 33 distinct bit lengths (128 to 384, multiples of 8) with equal sample counts per class. This uniform distribution is artificial (a randomly generated P-384 scalar has probability roughly 255/256 of occupying the full 48 bytes), but allows characterizing the timing distribution by class.

Figure 1 shows the empirical CDFs of loading time grouped by encoded byte length. The OpenSSL distributions (left) overlap almost entirely, with no visually discernible separation. The BoringSSL distributions (center) separate visibly, with a staircase structure suggesting leakage at byte granularity. Table 2 summarizes the statistical comparison across three complementary metrics. The mutual information $I(\text{bitlength}; \text{timing})$ across all 33 classes measures how much key-length information a single timing observation reveals. OpenSSL leaks 0.0031 bits of key-length information per timing observation; BoringSSL leaks 0.6104 bits, approximately $200\times$ more. To test the null hypothesis and establish a noise baseline, we shuffled class labels and recomputed MI over 100 rounds; the observed MI for all configurations exceeds this baseline, confirming the signal is genuine, though the magnitude differs significantly.

The 33-class MI characterizes the channel, but the adversary’s task is rather to decide if a key is full length or not. Partitioning into 384-bit keys versus all shorter encodings, Cohen’s d (where values near zero indicate overlap and values above one indicate separation) is 0.098 for OpenSSL, meaning the distributions are nearly indistinguishable, and 1.11 for BoringSSL, meaning the distributions barely overlap. This interpretation is consistent with the binary MI permutation z -score (standard deviations above the null hypothesis), which are 151 for OpenSSL and 19,021 for BoringSSL. All reject the null hypothesis, but by this metric the BoringSSL leak is two orders of magnitude stronger than that of OpenSSL.

For a randomly generated P-384 key, the probability of the scalar having full byte length is 255/256, a somewhat low

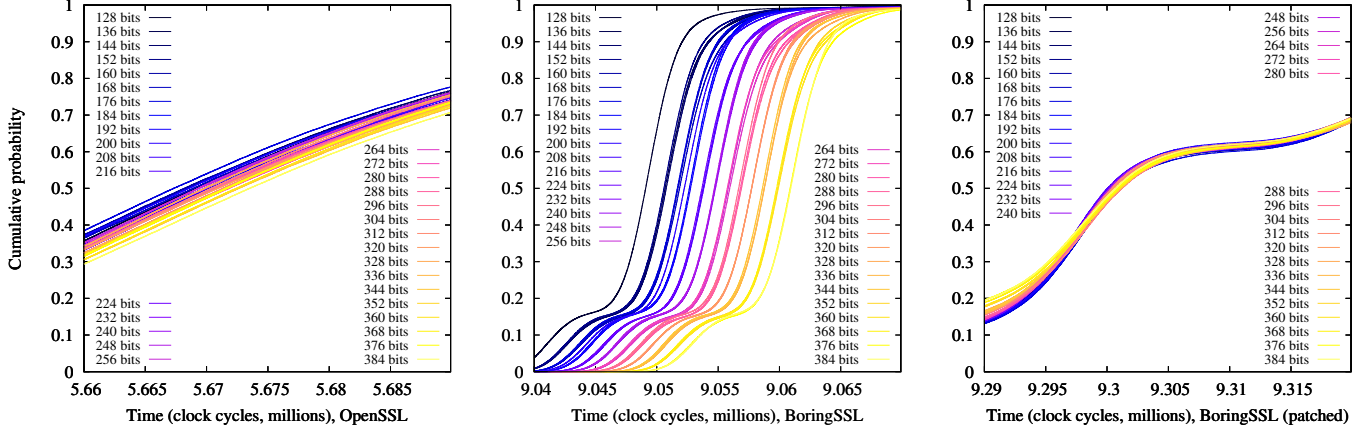


Figure 1: Empirical CDFs of EC private key loading time (P-384) grouped by effective byte length. OpenSSL (left): distributions overlap almost entirely. BoringSSL unpatched (center): distributions separate visibly, with a staircase structure at byte granularity. BoringSSL patched (right): with the hedge, distributions overlap, similar to OpenSSL.

probability for observing a truncated key. Nevertheless, this true a priori probability reflects the adversary’s actual decision problem. For OpenSSL, the adversary requires approximately 1,000 observations to reach a posterior probability $P > 0.5$, and 10,000 for $P > 0.999$. For BoringSSL, averaging roughly 10 observations yields $P > 0.5$, and 50 for $P > 0.999$, two orders of magnitude less effort than OpenSSL.

5.4 Disclosure, Defense, and Discussion

We shared our findings with the BoringSSL security team⁵. The remainder of this section discusses their response, a simple hedge that we prototyped, and the corresponding empirical results.

What BoringSSL could’ve done. The leakage in BoringSSL spans the entire decode stack, where PEM base64 decoding, DER parsing, and BIGNUM conversion all process variable-length input proportional to the encoded key size. Achieving constant-time behavior across these layers would require invasive changes to the serialization infrastructure, which the implementation currently does not pursue.

As an alternative, we implement a timing hedge internal to `PEM_read_bio_PrivateKey`. On each call, it spins up a second thread and runs the real parse concurrently with that of a fixed, full-length private key, and returns only after both complete (post-join). The observed time is therefore dominated by the slower parse, which (in practice) will approach the time of a full-length parse. We acknowledge this strategy violates most of the models in Table 1, yet the goal of this experiment is to raise the cost for adversaries, not to eliminate all potential leakage.

We then gathered a third dataset from the methodology in Section 5.3 against this patched BoringSSL version. The

hedged BoringSSL distributions (Figure 1, right) overlap, visually similar to OpenSSL. Regarding information-theoretic metrics (Table 2), 33-class MI drops from 0.61 to 0.005 bits (a $110\times$ reduction, versus 0.003 for OpenSSL), and binary MI from 0.037 to 0.0006 (versus 0.0003 for OpenSSL). On the adversary’s decision problem (short versus full) the hedge improves further. Cohen’s d drops from 1.11 to 0.008, below OpenSSL’s 0.098, and the adversary now requires roughly 100,000 observations to reach $P > 0.5$ and 1,000,000 for $P > 0.999$, notably four orders of magnitude more effort than against unpatched BoringSSL and two orders of magnitude more than even OpenSSL. Permutation testing for the null hypothesis still detects a signal across all datasets, yet our patched BoringSSL is harder to target than OpenSSL across every Bayesian metric.

What BoringSSL did. During disclosure, BoringSSL acknowledged the leak but declined to react with any code changes. On one hand, rejecting these malformed keys has unknown consequences since the key landscape is not well understood, which is a valid concern from an ecosystem and business perspective. Compounding this, in their view non-conformant keys fall outside their constant time boundary, which they subsequently clarified means constant time only for fixed-width inputs (see the bottom row of Table 1). The leak therefore falls outside their model by construction, and for variable-width inputs it follows that any mitigation that continues to accept malformed keys would also fail to satisfy the traditional constant time model. On the other hand, our data in Table 2 offers an adversary-driven perspective on the vulnerability. Even our simple hedge that clearly still violates the traditional constant time model increases the adversary’s cost by four orders of magnitude, and it does so while targeting only the OpenSSL model, the weakest of all in Table 1. Counterintuitively, BoringSSL aims at considerably higher security in their model, yet the leak is measurably stronger

⁵<https://marc.info/?l=oss-security&m=176046647932425>

than in OpenSSL, which applies a significantly weaker model. In 1772, Voltaire wrote⁶ “le mieux est l’ennemi du bien” or *the best is the enemy of the good*. Four orders of magnitude is a quantitative, objective improvement, and we claim that practical hedges have a useful place in security hardening, over arguments about model semantics.

6 Conclusion

This work systematizes the evolution of cryptographic constant time models in security-critical libraries, from early embedded-era assumptions through modern project-specific policies. The historical pattern is consistent, showing every relaxation of the traditional model has eventually been exploited. The traditional model’s strength is its simplicity, where all addresses are public, making fewer assumptions about hardware behavior than any of the alternatives.

The gap between what constant time models protect and what specifications assume is a recurring source of vulnerabilities. Padding oracles, encoding leakage, and canonicalization issues all originate in specifications that mandate operations on secrets that are difficult or impossible to realize in constant time. The methodology we distill from these case studies provides a systematic framework for exploring this gap.

Applying the methodology to EC private key loading illustrates the limits of threat model strength as a proxy for implementation security. BoringSSL’s constant time model is explicitly stronger than OpenSSL’s, in that it aims to protect against local and microarchitectural attackers, while OpenSSL’s security policy excludes colocation attacks entirely. Yet on this specific leak, our data from the adversarial perspective demonstrates that OpenSSL is empirically more resistant.

We close with the following joke posted to `rec.humor.funny` by Wolfgang Sohrt in 1989.

An engineer, a physicist and a mathematician have to build a fence around a flock of sheep, using as little material as possible.

The engineer forms the flock into a circular shape and constructs a fence around it.

The physicist builds a fence with an infinite diameter and pulls it together until it fits around the flock.

The mathematician thinks for a while, then builds a fence around himself and defines himself as being outside.

A strong threat model on paper means little if the boundaries are drawn too narrowly. Specifications exist in the wild, non-conformant inputs exist in the wild, and attackers do not respect scope declarations. Empirical measurement, not

model coverage, is the final arbiter of whether an implementation is vulnerable to timing attacks.

Availability. In support of Open Science and to ensure reproducibility, we released⁷ our tooling from the data procurement methodology in [Section 5.3](#) as a freely available research artifact for the benefit of the community. The artifact can be used to reproduce at least [Figure 1](#) (left, center).

References

- [1] SEC 1: Elliptic Curve Cryptography. SEC 1, Standards for Efficient Cryptography Group, Sep 2000. URL <https://www.secg.org/SEC1-Ver-1.0.pdf>.
- [2] Aris Adamantiadis, Simon Josefsson, and Mark D. Baushke. Secure Shell (SSH) Key Exchange Method Using Curve25519 and Curve448. RFC 8731, RFC Editor, February 2020. URL <https://datatracker.ietf.org/doc/rfc8731/>.
- [3] Martin R. Albrecht, Kenneth G. Paterson, and Gaven J. Watson. Plaintext recovery attacks against SSH. In *30th IEEE Symposium on Security and Privacy (SP 2009)*, 17-20 May 2009, Oakland, California, USA, pages 16–26. IEEE Computer Society, 2009. URL <https://doi.org/10.1109/SP.2009.5>.
- [4] Alejandro Cabrera Aldaya and Billy Bob Brumley. HyperDegrade: From GHz to MHz effective CPU frequencies. In Kevin R. B. Butler and Kurt Thomas, editors, *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, pages 2801–2818. USENIX Association, 2022. URL <https://www.usenix.org/conference/usenixsecurity22/presentation/aldaya>.
- [5] Alejandro Cabrera Aldaya, Billy Bob Brumley, Sohaib ul Hassan, Cesar Pereida García, and Nicola Tuveri. Port contention for fun and profit. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 870–887. IEEE, 2019. URL <https://doi.org/10.1109/SP.2019.00066>.
- [6] Nadhem J. AlFardan and Kenneth G. Paterson. Lucky Thirteen: Breaking the TLS and DTLS record protocols. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 526–540. IEEE Computer Society, 2013. URL <https://doi.org/10.1109/SP.2013.42>.
- [7] Gorka Irazoqui Apecechea, Mehmet Sinan Inci, Thomas Eisenbarth, and Berk Sunar. Lucky 13 strikes back. In Feng Bao, Steven Miller, Jianying Zhou, and Gail-Joon Ahn, editors, *Proceedings of the 10th ACM Symposium on Information, Computer and Communications*

⁶Voltaire, “La Bégueule” (1772)

⁷<https://gitlab.com/platsec/boringssl-keyload-vuln>

Security, AsiaCCS 2015, Singapore, April 14-17, 2015, pages 85–96. ACM, 2015. URL <https://doi.org/10.1145/2714576.2714625>.

- [8] Diego F. Aranha, Felipe Rodrigues Novaes, Akira Takahashi, Mehdi Tibouchi, and Yuval Yarom. LadderLeak: Breaking ECDSA with less than one bit of nonce leakage. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 225–242. ACM, 2020. URL <https://doi.org/10.1145/3372297.3417268>.
- [9] Santiago Arranz-Olmos, Gilles Barthe, Lionel Blatter, Youcef Bouzid, Sören van der Wall, and Zhiyuan Zhang. Decompiling for constant-time analysis. *Proceedings of the ACM on Programming Languages*, 10(OOPSLA1): 29–58, April 2026. URL <https://doi.org/10.1145/3798201>.
- [10] Romain Bardou, Riccardo Focardi, Yusuke Kawamoto, Lorenzo Simionato, Graham Steel, and Joe-Kai Tsay. Efficient padding oracle attacks on cryptographic hardware. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 608–625. Springer, 2012. URL https://doi.org/10.1007/978-3-642-32009-5_36.
- [11] Daniel J. Bernstein. Cache-timing attacks on AES, 2005. URL <http://cr.yp.to/papers.html#cachetiming>.
- [12] Daniel J. Bernstein. Curve25519: New Diffie-Hellman speed records. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, New York, NY, USA, April 24-26, 2006, Proceedings*, volume 3958 of *Lecture Notes in Computer Science*, pages 207–228. Springer, 2006. URL https://doi.org/10.1007/11745853_14.
- [13] Daniel J. Bernstein and Peter Schwabe. A word of warning. CHES 2013 Rump Session, August 2013. URL <https://cryptojedi.org/peter/data/chesrump-20130822.pdf>.
- [14] Daniel J. Bernstein, Tanja Lange, and Peter Schwabe. The security impact of a new cryptographic library. In Alejandro Hevia and Gregory Neven, editors, *Progress in Cryptology - LATINCRYPT 2012 - 2nd International Conference on Cryptology and Information Security in Latin America, Santiago, Chile, October 7-10, 2012. Proceedings*, volume 7533 of *Lecture Notes in Computer Science*, pages 159–176. Springer, 2012. URL https://doi.org/10.1007/978-3-642-33481-8_9.
- [15] Daniel J. Bernstein, Karthikeyan Bhargavan, Shivam Bhasin, Anupam Chattopadhyay, Tee Kiah Chia, Matthias J. Kannwischer, Franziskus Kiefer, Thales B. Paiva, Prasanna Ravi, and Goutam Tamvada. KyberSlash: Exploiting secret-dependent division timings in Kyber implementations. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2025(2):209–234, 2025. URL <https://doi.org/10.46586/tches.v2025.i2.209-234>.
- [16] Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer, 1997. URL <https://doi.org/10.1007/BFb0052259>.
- [17] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 1998. URL <https://doi.org/10.1007/BFb0055716>.
- [18] Hanno Böck, Juraj Somorovsky, and Craig Young. Return of Bleichenbacher’s oracle threat (ROBOT). In William Enck and Adrienne Porter Felt, editors, *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, pages 817–849. USENIX Association, 2018. URL <https://www.usenix.org/conference/usenixsecurity18/presentation/bock>.
- [19] Jeremy Boy, Antoon Purnal, Anna Pätschke, Luca Wilke, and Thomas Eisenbarth. SLasH-DNA: Breaking SLH-DNA using an extensible end-to-end Rowhammer framework. *Proc. Microarchitecture Secur. Conf.*, 2026, 2026. URL <https://doi.org/10.46586/uasc.2026.009>.
- [20] Ernie Brickell. Technologies to improve platform security. CHES 2011 Invited Talk, September 2011. URL https://iacr.org/workshops/ches/ches2011/presentations/Invited%201/CHES2011_Invited_1.pdf.
- [21] Daniel R. L. Brown and Sean Turner. Elliptic Curve Private Key Structure. RFC 5915, RFC Editor, June 2010. URL <https://datatracker.ietf.org/doc/rfc5915/>.

- [22] Johann Großschädl, Elisabeth Oswald, Dan Page, and Michael Tunstall. Side-channel analysis of cryptographic software via early-terminating multiplications. In Dong Hoon Lee and Seokhie Hong, editors, *Information, Security and Cryptology - ICISC 2009, 12th International Conference, Seoul, Korea, December 2-4, 2009, Revised Selected Papers*, volume 5984 of *Lecture Notes in Computer Science*, pages 176–192. Springer, 2009. URL https://doi.org/10.1007/978-3-642-14423-3_13.
- [23] Shay Gueron. Efficient software implementations of modular exponentiation. *J. Cryptographic Engineering*, 2(1):31–43, 2012. URL <https://doi.org/10.1007/s13389-012-0031-5>.
- [24] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: Cold boot attacks on encryption keys. In Paul C. van Oorschot, editor, *Proceedings of the 17th USENIX Security Symposium, July 28-August 1, 2008, San Jose, CA, USA*, pages 45–60. USENIX Association, 2008. URL http://www.usenix.org/events/sec08/tech/full_papers/halderman/halderman.pdf.
- [25] Tibor Jager and Juraj Somorovsky. How to break XML encryption. In Yan Chen, George Danezis, and Vitaly Shmatikov, editors, *Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, Illinois, USA, October 17-21, 2011*, pages 413–422. ACM, 2011. URL <https://doi.org/10.1145/2046707.2046756>.
- [26] Tibor Jager, Sebastian Schinzel, and Juraj Somorovsky. Bleichenbacher’s attack strikes again: Breaking PKCS#1 v1.5 in XML encryption. In Sara Foresti, Moti Yung, and Fabio Martinelli, editors, *Computer Security - ESORICS 2012 - 17th European Symposium on Research in Computer Security, Pisa, Italy, September 10-12, 2012. Proceedings*, volume 7459 of *Lecture Notes in Computer Science*, pages 752–769. Springer, 2012. URL https://doi.org/10.1007/978-3-642-33167-1_43.
- [27] Burt Kaliski. PKCS #1: RSA Encryption Version 1.5. RFC 2313, March 1998. URL <https://www.rfc-editor.org/info/rfc2313>.
- [28] Burt Kaliski and Jessica Staddon. PKCS #1: RSA Cryptography Specifications Version 2.0. RFC 2437, October 1998. URL <https://www.rfc-editor.org/info/rfc2437>.
- [29] Thierry Kaufmann, Hervé Pelletier, Serge Vaudenay, and Karine Villegas. When constant-time source yields variable-time binary: Exploiting Curve25519-donna built with MSVC 2015. In Sara Foresti and Giuseppe Persiano, editors, *Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings*, volume 10052 of *Lecture Notes in Computer Science*, pages 573–582, 2016. URL https://doi.org/10.1007/978-3-319-48965-0_36.
- [30] John Kelsey, Bruce Schneier, David A. Wagner, and Chris Hall. Side channel cryptanalysis of product ciphers. In Jean-Jacques Quisquater, Yves Deswarte, Catherine Meadows, and Dieter Gollmann, editors, *Computer Security - ESORICS 98, 5th European Symposium on Research in Computer Security, Louvain-la-Neuve, Belgium, September 16-18, 1998, Proceedings*, volume 1485 of *Lecture Notes in Computer Science*, pages 97–110. Springer, 1998. URL <https://doi.org/10.1007/BFb0055858>.
- [31] Yoongu Kim, Ross Daly, Jeremie S. Kim, Chris Fallin, Ji-Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ACM/IEEE 41st International Symposium on Computer Architecture, ISCA 2014, Minneapolis, MN, USA, June 14-18, 2014*, pages 361–372. IEEE Computer Society, 2014. URL <https://doi.org/10.1109/ISCA.2014.6853210>.
- [32] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. Spectre attacks: Exploiting speculative execution. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 1–19. IEEE, 2019. URL <https://doi.org/10.1109/SP.2019.00002>.
- [33] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996. URL https://doi.org/10.1007/3-540-68697-5_9.
- [34] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 1999. URL https://doi.org/10.1007/3-540-48405-1_25.

- [35] Dennis Kupser, Christian Mainka, Jörg Schwenk, and Juraj Somorovsky. How to break XML encryption—automatically. In Aurélien Francillon and Thomas Ptacek, editors, *9th USENIX Workshop on Offensive Technologies, WOOT '15, Washington, DC, USA, August 10-11, 2015*. USENIX Association, 2015. URL <https://www.usenix.org/conference/woot15/workshop-program/presentation/kupser>.
- [36] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown: Reading kernel memory from user space. In William Enck and Adrienne Porter Felt, editors, *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, pages 973–990. USENIX Association, 2018. URL <https://www.usenix.org/conference/usenixsecurity18/presentation/lipp>.
- [37] Chris M. Lonvick and Tatu Ylonen. The Secure Shell (SSH) Protocol Architecture. RFC 4251, RFC Editor, January 2006. URL <https://datatracker.ietf.org/doc/rfc4251/>.
- [38] Chris M. Lonvick and Tatu Ylonen. The Secure Shell (SSH) Transport Layer Protocol. RFC 4253, RFC Editor, January 2006. URL <https://datatracker.ietf.org/doc/rfc4253/>.
- [39] James Manger. A chosen ciphertext attack on RSA optimal asymmetric encryption padding (OAEP) as standardized in PKCS #1 v2.0. In Joe Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 230–238. Springer, 2001. URL https://doi.org/10.1007/3-540-44647-8_14.
- [40] Robert Merget, Marcus Brinkmann, Nimrod Aviram, Juraj Somorovsky, Johannes Mittmann, and Jörg Schwenk. Raccoon attack: Finding and exploiting most-significant-bit-oracles in TLS-DH(E). In Michael Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 213–230. USENIX Association, 2021. URL <https://www.usenix.org/conference/usenixsecurity21/presentation/merget>.
- [41] David Molnar, Matt Piotrowski, David Schultz, and David A. Wagner. The program counter security model: Automatic detection and removal of control-flow side channel attacks. In Dongho Won and Seungjoo Kim, editors, *Information Security and Cryptology - ICISC 2005, 8th International Conference, Seoul, Korea, December 1-2, 2005, Revised Selected Papers*, volume 3935 of *Lecture Notes in Computer Science*, pages 156–168. Springer, 2005. URL https://doi.org/10.1007/11734727_14.
- [42] Dag Arne Osvik, Adi Shamir, and Eran Tromer. Cache attacks and countermeasures: The case of AES. In David Pointcheval, editor, *Topics in Cryptology - CT-RSA 2006, The Cryptographers' Track at the RSA Conference 2006, San Jose, CA, USA, February 13-17, 2006, Proceedings*, volume 3860 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2006. URL https://doi.org/10.1007/11605805_1.
- [43] Dan Page. Theoretical use of cache memory as a cryptanalytic side-channel. *IACR Cryptology ePrint Archive*, 2002(169), 2002. URL <http://eprint.iacr.org/2002/169>.
- [44] Colin Percival. Cache missing for fun and profit. In *BSD-Can 2005, Ottawa, Canada, May 13-14, 2005, Proceedings*, 2005. URL <http://www.daemonology.net/papers/cachemissing.pdf>.
- [45] Cesar Pereida García and Billy Bob Brumley. Constant-time callees with variable-time callers. In Engin Kirda and Thomas Ristenpart, editors, *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017*, pages 83–98. USENIX Association, 2017. ISBN 978-1-931971-40-9. URL <https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/garcia>.
- [46] Cesar Pereida García, Sohaib ul Hassan, Nicola Taveri, Iaroslav Gridin, Alejandro Cabrera Aldaya, and Billy Bob Brumley. Certified side channels. In Srdjan Capkun and Franziska Roesner, editors, *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pages 2021–2038. USENIX Association, 2020. URL <https://www.usenix.org/conference/usenixsecurity20/presentation/garcia>.
- [47] Thomas Pornin. Constant-time code: The pessimist case. *IACR Cryptology ePrint Archive*, 2025(435), 2025. URL <https://eprint.iacr.org/2025/435>.
- [48] Jean-Jacques Quisquater and David Samyde. Electromagnetic analysis (EMA): measures and countermeasures for smart cards. In Isabelle Attali and Thomas P. Jensen, editors, *Smart Card Programming and Security, International Conference on Research in Smart Cards, E-smart 2001, Cannes, France, September 19-21, 2001, Proceedings*, volume 2140 of *Lecture Notes in Computer Science*, pages 200–210. Springer, 2001. URL https://doi.org/10.1007/3-540-45418-7_17.

- [49] Oscar Reparaz, Josep Balasch, and Ingrid Verbauwhede. Dude, is my code constant time? In David Atienza and Giorgio Di Natale, editors, *Design, Automation & Test in Europe Conference & Exhibition, DATE 2017, Lausanne, Switzerland, March 27-31, 2017*, pages 1697–1702. IEEE, 2017. URL <https://doi.org/10.23919/DATE.2017.7927267>.
- [50] Eyal Ronen, Robert Gillham, Daniel Genkin, Adi Shamir, David Wong, and Yuval Yarom. The 9 lives of Bleichenbacher’s CAT: new cache ATtacks on TLS implementations. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*, pages 435–452. IEEE, 2019. URL <https://doi.org/10.1109/SP.2019.00062>.
- [51] Florian Sieck, Sebastian Berndt, Jan Wichelmann, and Thomas Eisenbarth. Util: : Lookup: Exploiting key decoding in cryptographic libraries. In Yongdae Kim, Jong Kim, Giovanni Vigna, and Elaine Shi, editors, *CCS ’21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*, pages 2456–2473. ACM, 2021. URL <https://doi.org/10.1145/3460120.3484783>.
- [52] Florian Sieck, Zhiyuan Zhang, Sebastian Berndt, Chitchanok Chuengsatiansup, Thomas Eisenbarth, and Yuval Yarom. TeeJam: Sub-cache-line leakages strike back. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2024(1):457–500, 2024. URL <https://doi.org/10.46586/tches.v2024.i1.457-500>.
- [53] Yukiyasu Tsunoo, Teruo Saito, Tomoyasu Suzaki, Maki Shigeri, and Hiroshi Miyauchi. Cryptanalysis of DES implemented on computers with cache. In Colin D. Walter, Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems - CHES 2003, 5th International Workshop, Cologne, Germany, September 8-10, 2003, Proceedings*, volume 2779 of *Lecture Notes in Computer Science*, pages 62–76. Springer, 2003. URL https://doi.org/10.1007/978-3-540-45238-6_6.
- [54] Serge Vaudenay. Security flaws induced by CBC padding. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *Lecture Notes in Computer Science*, pages 534–546. Springer, 2002. URL https://doi.org/10.1007/3-540-46035-7_35.
- [55] James Walden. The impact of a major security event on an open source project: The case of OpenSSL. In Sunghun Kim, Georgios Gousios, Sarah Nadi, and Joseph Hejderup, editors, *MSR ’20: 17th International Conference on Mining Software Repositories, Seoul, Republic of Korea, 29-30 June, 2020*, pages 409–419. ACM, 2020. URL <https://doi.org/10.1145/3379597.3387465>.
- [56] Samuel Weiser, David Schrammel, Lukas Bodner, and Raphael Spreitzer. Big numbers - big troubles: Systematically analyzing nonce leakage in (EC)DSA implementations. In Srdjan Capkun and Franziska Roesner, editors, *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pages 1767–1784. USENIX Association, 2020. URL <https://www.usenix.org/conference/usenixsecurity20/presentation/weiser>.
- [57] Yuanzhong Xu, Weidong Cui, and Marcus Peinado. Controlled-channel attacks: Deterministic side channels for untrusted operating systems. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 640–656. IEEE Computer Society, 2015. URL <https://doi.org/10.1109/SP.2015.45>.
- [58] Yuval Yarom and Katrina Falkner. FLUSH+RELOAD: A high resolution, low noise, L3 cache side-channel attack. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014*, pages 719–732. USENIX Association, 2014. ISBN 978-1-931971-15-7. URL <https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/yarom>.
- [59] Yuval Yarom, Daniel Genkin, and Nadia Heninger. CacheBleed: A timing attack on OpenSSL constant time RSA. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings*, volume 9813 of *Lecture Notes in Computer Science*, pages 346–367. Springer, 2016. URL https://doi.org/10.1007/978-3-662-53140-2_17.

Bit2Watt: A CyberPhysical Vulnerability Exploiting GPU Workloads Across Power and Computing Infrastructures

Zhouhao Ji, Kaikai Pan* and Wenyuan Xu

Zhejiang University, Hangzhou, China

zhouhao0779@zju.edu.cn, pankaikai@zju.edu.cn, wyxu@zju.edu.cn

Abstract. Modern data centers increasingly rely on large-scale GPU clusters and on-site renewable energy resources, resulting in a tightly coupled cyberphysical system between computing workloads and power-electronic-dominated grids. In this paper, we reveal **Bit2Watt**, a previously unexplored vulnerability in which an adversary manipulates GPU workloads to induce controlled, high-frequency power modulations that destabilize local power infrastructure and propagate back to disrupt computing services. Unlike traditional attacks that compromise grid-side devices or communication channels, **Bit2Watt** operates entirely within the cyber layer as a legal tenant, which could amplify fluctuations, harmonic distortion, and damping degradation, particularly in high-DER-penetration scenarios. This risk is difficult to detect under routine cloud- and facility-side monitoring because **Bit2Watt** exploits legitimate workload execution paths and concentrates much of its distinctive behavior in high-frequency components that are weakly captured by common telemetry. We validate **Bit2Watt** through impedance-based analysis, power system simulations, and real-world experiments on GPUs and grid-connected PV inverters. Under the synchronized worst-case aggregation model studied in the paper, manipulating 1,000 GPUs in a 1-MW local power system with 90% DERs raises current THD to 46.8% and results in a damping ratio of -0.27. We further show that the resulting power-quality degradation can stress data-center power-delivery equipment, trigger protection mechanisms, and, in extreme simulated cases, induce cascading failures in transmission-scale systems. In addition, we analyze a plausible **Watt2Bit** feedback path, including denial-of-service risks and covert information exfiltration via EMI side channels. This work exposes a vulnerability at the intersection of modern computing and renewable-integrated power systems, highlighting the urgent need for cross-layer defenses that jointly consider workload scheduling and power electronics.

Keywords: Cyber-physical security · Systemic vulnerability · Electric power networks · Computing infrastructure · GPU power modulation

1 Introduction

Modern digital infrastructure is increasingly built upon the deep integration of large-scale computing systems and electrical power networks. The rapid proliferation of GPU-accelerated workloads driven by artificial intelligence and high-performance computing (HPC) has dramatically increased both the magnitude and volatility of the power demand in data centers, which varies at sub-millisecond timescales. In parallel, power networks worldwide are transforming toward high penetration of inverter-control-based distributed

*Corresponding author.

energy resources (DERs), such as photovoltaics (PVs). As a result, modern data centers and renewable-integrated grids together constitute a networked cyberphysical system, in which cyber-level workload scheduling directly drives physical-layer actuation through power-electronic interfaces and fast control loops.

While prior research [SCB22, SMP18, DUW17, NNQAG18, JFK17, ZL25, SZ21] has independently highlighted the security issues in power systems and computing systems, existing cyberphysical threat models largely overlook adversaries who do not compromise sensing, control or communication infrastructure, but instead exploit legitimate cyber-level control inputs to induce cyberphysical security issues. Modern cloud platforms powered by data centers allow untrusted but fully legitimate tenants to control thousands of GPUs through official software, raising a critical question: ***can purely computational actions, executed as legitimate workloads, be weaponized to destabilize power infrastructure?*** This question motivates our study, as such risks would bypass conventional perimeter defenses by operating entirely within legitimate access privileges and exploiting the intrinsic cyberphysical coupling of next-generation infrastructures.

Addressing this question is promising yet challenging. First, revealing the principle of how to induce power fluctuation through GPU tasks is a challenge. Second, it remains challenging to rigorously analyze how computation-induced power modulations propagate and interact with the dynamics of power infrastructure. To overcome the challenges, we first propose two kinds of task-based power modulation methods in the cyber domain by leveraging the phase-driven power behavior of GPUs and using only legitimate user-level workloads. We then establish an impedance-based analytical model that captures the interaction between GPU clusters, utility grid and the frequency-shaped dynamics of inverter-based DERs in the physical domain.

In this paper, we conduct a systematic risk analysis of these two tightly coupled infrastructures, aiming to uncover the risk of how adversaries could induce disruptions to power infrastructure through the computing infrastructure referred to as the **Bit2Watt** risk. We further highlight that the **Bit2Watt** risk may potentially give rise to a self-reinforcing feedback loop, thereby amplifying the overall impact and triggering what we term the **Watt2Bit** risk, including denial-of-service (DoS) risks and exfiltration risks. The **Bit2Watt** risk exhibits a high degree of covertness, as it operates strictly within authorized workload execution paths, thereby circumventing contemporary cloud-side monitoring frameworks, creating a critical security blind spot regarding the cyber-physical ramifications induced by malicious workload orchestration.

Our results indicate that GPU loads can reach modulation frequencies exceeding 6,000 Hz, compared with only a few hertz observed in conventional household loads such as air conditioners. Such high-frequency modulations can substantially induce voltage excursions, harmonic distortion, and damping degradation. In particular, within a 1-MW local power grid comprising 90% DERs, manipulating just 1,000 GPUs results in a total harmonic distortion (THD) of 46.8%, implying that nearly half of the current is expended on non-productive work and dissipated as heat approximately 20% more than under attack-free conditions. This not only threatens the availability of the computing equipment but also produces a negative damping ratio of -0.27, introducing an unstable mode into the system. Once the protections are triggered and computing loads are shed, it can trigger cascading failures, potentially leading to blackouts exceeding 80% in large-scale power systems.

1.1 Contributions

First, this paper formalizes **Bit2Watt** as a new class of workload-driven cyberphysical threat, where computational processes are leveraged as *controllable excitation sources* that can interact with power system dynamics. Unlike prior observations of workload-induced power variability, the proposed framework characterizes how such fluctuations can be

systematically structured in frequency and temporal patterns, enabling persistent and targeted perturbations to power infrastructures. This establishes a previously unexplored mechanism that links computing workload orchestration with grid-level dynamic responses, providing a principled foundation for analyzing cyberphysical risks at the interface of digital computation and physical energy systems.

Second, this paper develops two distinct power modulation techniques: synthetic workload modulation attack (SWMA) and LLM training modulation attack (LTMA), leveraging user-level GPU programming capabilities without requiring elevated privileges. SWMA achieves high-frequency power modulations (up to 6,000 Hz) via persistent CUDA kernels, while LTMA embeds modulation logic within standard deep learning pipelines, effectively camouflaging malicious power signatures as legitimate computational noise.

Third, this paper provides a rigorous impedance-based analysis of how GPU power modulation interacts with inverter-dominated grids. The study quantifies the impact of high-frequency GPU power modulations on power quality and system damping, offering a principled approach to understanding system vulnerabilities.

Fourth, this paper extends the impact beyond grid-side disruption by analyzing the Watt2Bit risk. We show that attack-induced electrical stress can create DoS conditions through protection trips and workload interruption, and further demonstrate that the same mechanism may also enable covert information exfiltration via EMI side channels.

Fifth, this paper conducts simulation studies and real-world experiments across both the cyber computing and physical power layers. Quantitative validations are provided to illustrate the intrinsic vulnerabilities of GPU workloads and their external impact on power system stability.

1.2 Related Work

Power characteristics of data centers. Empirical measurements of the instantaneous power draw of an 8-GPU NVIDIA H100 HGX node were provided during the training of models, and it was shown that the power draw of the GPU fluctuated sharply during different phases [LNC⁺24]. To meet the growing power demand of modern data centers, efforts have been made to incorporate more renewable energy sources (RESs) in the power supply architecture of the data center. To cope with the intermittent nature of the RES and improve the efficiency of RES utilization, dynamic request dispatching and task scheduling of the GPUs are proposed for searching for suitable computing nodes and determining the start and finish time [ZWW11, LWZ⁺16, Kwo20], while [VGGQ22] used the reinforcement learning based method. Techno-economic analysis and optimal sizing methodology were proposed to quantify the trade-off between power dependency and the financial cost [GBO23].

Interaction between grid operations and data centers. It is found that optimally placed dispatchable computing loads, for instance, next to wind farms, can facilitate consumption of stranded power and thus enable better scaling to high renewable portfolio standard (RPS) [KYZC16]. Microsoft, OpenAI and NVIDIA jointly proposed the challenge arising from the frequency spectrum of the power swings during different GPU tasks [CWH⁺25]. They also developed innovative solutions from the perspective of the software, GPU power smoothing and rack-level energy storage. Another recent study explored the interplay between AI workload transients and data center power electronics, revealing that conventional CPU-oriented power architectures may fail to support the rapid load variations of AI accelerators [LL25]. Conversely, the power quality of the electrical grid can also affect the normal operation of GPUs in data centers. It was proved that the fluctuating power supply could influence the performance of the GPUs [NG23].

Cyber and physical attacks on modern power and computing infrastructure. Cyberphysical attacks on power infrastructure have traditionally targeted central control systems and edge devices. Prior studies show that power system operations can be dis-

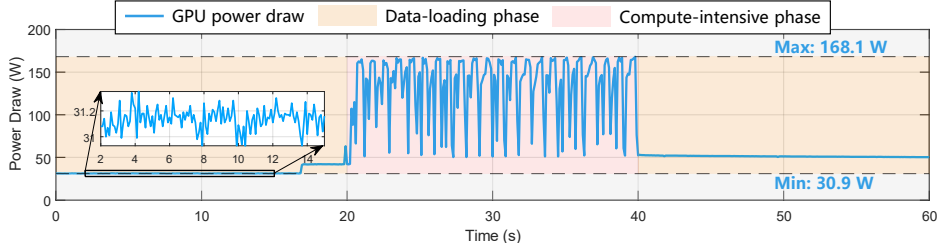


Figure 1: Instantaneous power draw of a single NVIDIA Volta V100 GPU [SWB⁺21].

rupted through false data injection and GPS spoofing targeting control and monitoring layers [MBA⁺23]. Moreover, frequency disturbances and cascading failures can be induced via large-scale botnets, including computer botnets [DUW17] and appliance-based botnets [SCB22, SMP18]. The rapid proliferation of electric vehicles (EVs) further expands the attack surface, enabling distributed load manipulation attacks that threaten grid stability [WL23, GKMA22]. In computing systems, attacks primarily target the cyber domain, with side-channel attacks being a prominent example. In particular, GPUs and heterogeneous computing platforms have been shown to leak sensitive information through shared resources and execution-dependent power signatures [NNQAG18, JFK17, ZL25, SZ21].

In conclusion, existing studies have extensively analyzed GPU power characteristics and cyber-domain attacks, but largely overlook how workload-induced power variations can be exploited to impact power grid operation. This missing cross-domain perspective motivates our work, which reveals a new cyberphysical attack surface arising from the tight coupling between computing infrastructures and power systems.

2 Background

2.1 Rise of Large Data Centers

Data centers host large-scale CPU and GPU clusters for AI and cloud computing, requiring stable power supply. Their aggregated demand is both substantial and highly dynamic, making them not only critical loads but also potential sources of stress to the power grid.

1. Huge electricity demand. Data center deployment has rapidly expanded in recent years, with major hubs in regions such as Virginia, California, and Texas. Other spatial distribution in the U.S. is presented in the appendix. Global data center electricity consumption exceeded 400 TWh in 2024 (about 2% of total demand) and is projected to double by 2030 [LC25]. In data-center-dense regions like Virginia, this share already exceeds 25% [KZ24]. At the device level, as shown in Figure 1, during compute-intensive phases, a single V100 GPU can exhibit a power draw of nearly 170 W, approaching its hardware thermal design power (TDP). Large facilities may host over 10^5 GPUs [DPO25], resulting in total power demand on the order of 20-50 MW, accounting for about one-tenth of Berlin’s total peak electrical load in 2024 [Ber25].

2. Drastic power ramps. In contrast to traditional CPU workloads that typically display smooth and predictable power consumption with lower peak-to-average ratios, GPUs demonstrate extreme power volatility. During data-loading phases, GPUs may consume only a fraction of the peak power, whereas during compute-intensive phases, the power demand can surge dramatically. The transitions between GPU workload stages often occur within milliseconds, leading to sudden spikes or drops in power demand. As illustrated in Figure 1, the rate of change of power (RoCoP), $\Delta P/\Delta t$, is measured at nearly 1 kW/s for an individual V100 GPU and extrapolates to more than 1000 MW/s

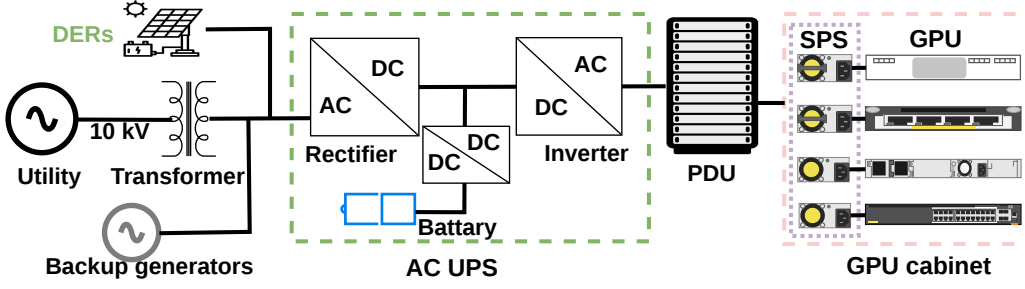


Figure 2: Typical power supply architecture in modern data centers.

for the aggregate data center load.

2.2 Power Supply Architecture in Data Centers

In recent years, the power supply architecture of data centers has undergone a profound transformation driven by the increasing integration of DERs. This shift is reflected not only in design standards such as TIA-942-C, but also in the widespread deployment of renewable-integrated power architectures in real-world data center projects. For instance, Meta Platforms has entered into multiple agreements with renewable energy developer Invenergy to procure approximately 791 MW of solar and wind power for data center operations. Similar renewable energy integration initiatives have also been undertaken by NVIDIA and other major technology companies [Reu25, Bee25].

To guarantee high availability and fault tolerance, modern hyperscale data centers universally employ a redundant power distribution architecture (typically a 2N topology), where critical IT workloads are sustained by two independent and symmetrical power feeds. Without loss of generality, Figure 2 presents the single-line diagram of one such power path since the redundant path shares an identical topological structure. As illustrated, energy from the upstream grid with high DER penetration enters the AC uninterruptible power supply (UPS) system through voltage transformation and power conversion. Subsequently, the power is routed through power distribution units (PDUs), which facilitate circuit protection and granular monitoring, before reaching the GPU cabinets. Finally, dedicated switching power supplies (SPSs) execute the terminal conversion stage to provide regulated DC power to the GPUs.

Within this power architecture, the extensive integration of the DERs, UPS, PDUs and SPSs marks a transition toward a converter-dominated system governed by sophisticated control strategies. However, this high degree of power electronization and control complexity inherently introduces significant vulnerabilities, which will be discussed later.

2.3 Modern Power Grid

Modern power grids are increasingly dominated by inverter-based resources (IBRs) such as PV and energy storage systems. Unlike synchronous generators, their dynamics are governed by multi-loop cascaded power-electronic control rather than physical inertia, resulting in more complex system responses. A typical PV inverter adopts a hierarchical control architecture consisting of three cascaded loops: Phase-Locked Loop (PLL) control, voltage control, and current control, as illustrated in Figure 5.

The PLL loop provides synchronization by estimating the grid phase angle ($\theta = \theta_0 + \omega t$) and aligning the inverter with the PCC (point of common coupling) voltage V_{pcc} . By transforming three-phase voltages from the abc frame to the rotating dq frame, AC tracking is converted into a DC regulation problem. The resulting v_d and v_q correspond

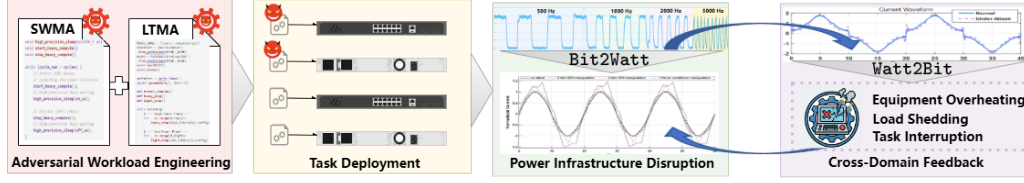


Figure 3: Overview of the **Bit2Watt** risk. It has a four-stage cycle. It begins with the malicious workload design, i.e., malicious **Bit**. After the malicious **Bit** is deployed on the cloud platform, it induces high RoCoP and disrupts the local power grid. Power distortions in turn interrupt the computing devices via overheating and triggering the shedding protection, forming **Watt2Bit**. Finally, a vicious circle is established.

to active and reactive voltage components. Due to the trigonometric transformations, the PLL exhibits inherent nonlinearity.

The outer DC-bus voltage control loop maintains the energy balance between the input power from the PV source and the output AC power. The controller operates as an outer control loop that regulates the DC bus voltage by adjusting the active power exchanged with the grid. Based on the measured DC-bus voltage deviation from its reference value, the controller generates an active current reference for the inner current control loop.

The current control loop is the innermost and fastest layer of a grid-connected DER inverter, directly governing how the inverter injects current into the grid. It ensures that the inverter output current follows the reference commands generated by the outer control loops, thereby stabilizing the inverter-grid interaction.

3 Threat Model

We consider a legitimate but malicious tenant operating GPU workloads in a shared computing environment (e.g., a data center connected to a DER-rich grid). The attacker requires no privileged access and instead exploits standard workload-level controls (e.g., scheduling and GPU utilization) to modulate computational intensity over time.

By coordinating high-power workloads and their startstop timing, the attacker can induce fast and repetitive power variations at the facility level. This capability can be amplified with access to a large number of GPUs (e.g., enterprise-scale accounts).

The attackers objective is to inject high-frequency power fluctuations into the data centers electrical interface. These disturbances propagate through power-electronic converters and interact with DER control loops, potentially degrading voltage/current regulations and introducing harmonics. Such effects may further feed back to the computing infrastructure (**Watt2Bit**), forming a closed-loop cyberphysical risk.

4 Understanding the Bit2Watt Risk

4.1 Overview

Figure 3 illustrates the four-stage execution cycle of the **Bit2Watt** vulnerability, revealing a closed-loop cyber-physical risk between computing workloads and power infrastructure. To understand the **Bit2Watt** risk, we first elaborate on: 1) how to induce power fluctuation through GPU tasks and 2) how the power fluctuation affects the power infrastructure, and we validate these principles with simulations and real-world experiments.

4.2 How to induce power fluctuation through GPU tasks?

4.2.1 Principle and Design

The instantaneous GPU power draw is tightly coupled with the computational workload running on streaming multiprocessors (SMs), tensor cores, memory controllers, and the associated on-chip clock generators. When computational intensity increases, the GPU automatically raises frequency and current to deliver higher performance, resulting in a rapid increase in dynamic power consumption. Conversely, when workload demand drops, the GPU transitions toward energy-saving states.

This behavior provides a software-accessible mechanism for power modulation without requiring privileged access. By controlling GPU workloads, an attacker can induce oscillations between high- and low-power phases. In this subsection, we design two attack methods based solely on user-level GPU programming capabilities.

1. Synthetic Workload Modulation Attack (SWMA). The attacker uploads a purpose-built CUDA program consisting of two internal execution modes: 1) active mode, which executes high-intensity tensor-core or FP32 FMA loops with large global-memory access, saturating SM occupancy and maximizing power consumption; 2) passive mode, in which the kernel performs negligible computation and quickly enters an idle state.

The implementation of SWMA relies on a GPU kernel and a CPU controller. The GPU kernel features two internal execution modes, while the CPU controller communicates with the kernel via a unified-memory control flag allocated using `cudaMallocManaged`, enabling real-time modulation of the kernel’s execution mode. Once the kernel is launched, it enters a continuous polling loop in which each thread block monitors the control flag and instantly transitions to the corresponding execution mode upon detecting a change. Meanwhile, a CPU-side controller periodically updates the flag according to a predefined switching schedule. The controller itself runs entirely in user space and therefore requires no privileged access to GPU hardware or system firmware.

2. LLM Training Modulation Attack (LTMA). This approach exploits the intrinsic compute variability within Large Language Model (LLM) training to induce power modulations without designing an additional synthetic workload. Instead of launching a GPU kernel, the adversary embeds modulation logic directly into the training pipeline of an LLM. Modern distributed training frameworks (e.g., PyTorch, DeepSpeed, Megatron-LM) execute complex sequences of compute-heavy primitives such as tensor-core GEMM kernels, fused attention kernels, and backpropagation primitives whose instantaneous intensity and invocation patterns strongly influence GPU power draw.

In this design, the training script functions as the modulation controller. The control is achieved by dynamically manipulating hyperparameters or inserting auxiliary operations at specific time intervals. During the high-load phases, the training script intentionally increases instantaneous compute demand through legitimate mechanisms. The lowload phase can be produced through modifications that temporarily lighten the computations without compromising training correctness.

This attack requires no elevated privileges: cloud tenants ordinarily possess full control over their training scripts, batch schedules, and dataflow definitions. Therefore, the modulation is implemented entirely in user-space within the training framework, blending seamlessly into normal training behavior and leaving no externally visible synthetic kernels.

Comparison. The SWMA could offer precise control over frequency and duty cycle through the host-side modulation logic, and support higher-frequency switching. However, its reliance on a custom CUDA kernel and device-level polling might reduce the concealment. LTMA operates entirely within standard Python-level training pipelines, making it far easier to implement and far more difficult for the cloud provider to distinguish from legitimate training noise. Its control accuracy is inherently constrained by the iteration

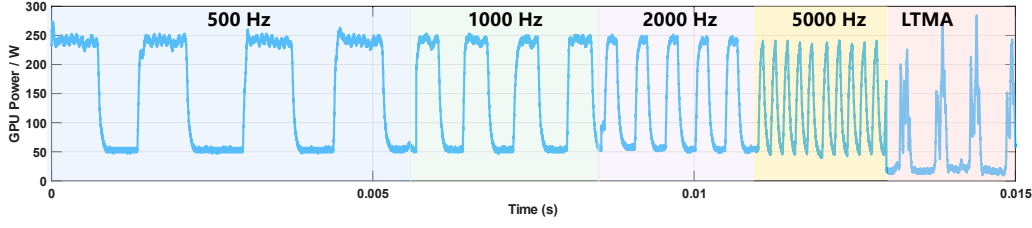


Figure 4: Concatenated results of different power modulation methods on RTX 3090. The first four attacks are SWMA and the last is a 2000 Hz LTMA.

rate of the training loop.

4.2.2 Feasibility Analysis

Availability. Modern hyperscale cloud platforms, including AWS, Google Cloud, Alibaba Cloud, Huawei Cloud, and Azure, provide tenants with on-demand, elastic access to powerful HPC infrastructure. This pervasive availability of rentable GPU-accelerated compute has dramatically lowered the barrier for running large-scale AI and data-intensive workloads. These corporations expose unified SDKs, REST APIs, and CLI tools that enable tenants to automate job submissions, upload containerized training tasks, and precisely control start/stop timing.

Spatial aggregation. Existing schedulers use topology-aware placement, which strategically maps distributed tasks—particularly large-scale AI training—to physically adjacent GPU nodes within the same rack to exploit high-bandwidth interconnects like NVLink and InfiniBand. This feature inadvertently facilitates the physical aggregation of electrical demand, thereby avoiding the geographical dispersion of power fluctuations from different GPUs, which would otherwise dilute their aggregate impact.

Temporal synchronization. Adversaries could utilize user-space busy-waiting loops on high-resolution timestamps, thereby reducing the non-deterministic scheduling jitter of the operating system. In HPC or dedicated GPU clusters, this synchronization can be achieved via Precision Time Protocol (PTP), whereas in public clouds it is typically provided implicitly by provider-managed clock services. Nevertheless, achieving perfectly aligned power transitions between high-power and low-power phases across all GPUs is still a challenge.

Cost. An adversary can readily acquire legitimate access to these resources by obtaining valid tenant accounts or credentials, operating under the guise of a normal user. The competitive pricing of high-performance GPU rental—ranging from \$0.39/hour to \$2.99/hour [GPU25c, GPU25b]—further lowers the financial threshold for mounting an attack. Although these costs are typically quoted per hour, a synchronized attack often requires an execution time of only a few minutes to degrade the power quality and stability, significantly reducing the actual cost and time commitment for the adversary.

4.2.3 Validation

Experiment setup. The data were captured on modified in-lab workstations running Docker, equipped with INTEL(R) XEON(R) SILVER 4510 processor and 128 GB of SK Hynix HMC78AE BRA107N DDR5 RDIMM memory, configured as eight 16 GB DIMMs operating at 4400 MT/s. The LTMA is realized through the training process of GPT-2. Power data are measured using an NI USB-6421 data acquisition system (DAQ) in conjunction with a LabVIEW-based data acquisition application. The DAQ senses the current and voltage signals from the GPU-side auxiliary +12 V supply lines.

Table 1: Power modulation results of representative GPUs

GPU	Arch.	Year	TDP (W)	SWMA			LTMA		
				Freq. (Hz)	Amp. (W)	Ctrl.	Freq. (Hz)	Amp. (W)	Ctrl.
RTX 2080	Turing	2018	185	1,500	103	✓	1,200	134	×
RTX 3080	Ampere	2020	320	2,000	157	✓	1,800	225	×
RTX 3090	Ampere	2020	350	5,000	204	✓	2,000	265	×
RTX 4090	Ada Lovelace	2022	450	6,000	248	✓	3,000	355	×
Tesla V100	Volta	2017	250	3,000	124	✓	2,000	210	×
A100	Ampere	2020	250	5,000	135	✓	3,000	215	×

† Arch.: architecture; Freq.: frequency; Amp.: amplitude; Ctrl.: controllability.

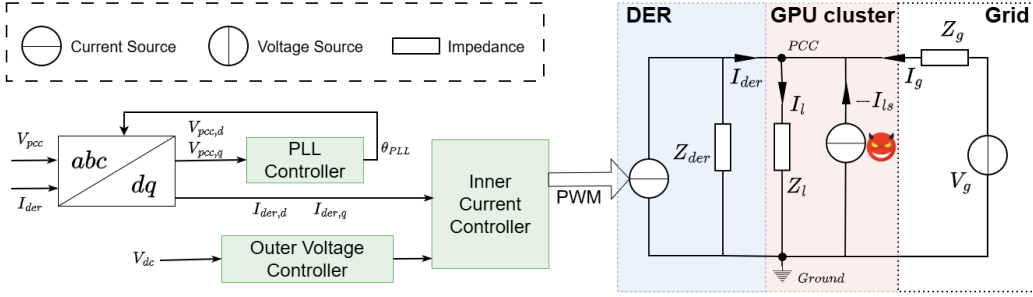


Figure 5: Typical power supply system of the local data center with integration of DERs.

Result analysis. Table 1 summarizes the power modulation capability of representative NVIDIA GPUs across both consumer-grade and data-center architectures. Across all tested GPUs, SWMA achieves higher modulation frequencies than LTMA, with spectral components ranging from 1.5 kHz to 6 kHz. This is enabled by fine-grained workload scheduling and kernel-level execution dynamics. In contrast, LTMA exhibits lower modulation frequencies (1.23 kHz) but with larger modulation amplitudes. The time-series power profiles under different modulation strategies are concatenated and illustrated in Figure 4. Taking the 5,000 Hz SWMA as an example, when the modulation frequency exceeds this level, the fluctuation amplitude can no longer reach 204 W. This limitation is attributed to the finite dynamic response of GPU power management and voltage regulation, which cannot fully track rapid workload transitions.

From a controllability perspective, SWMA allows explicit and precise tuning of the modulation frequency via workload scheduling, whereas LTMA lacks such direct control. In LTMA, both frequency and amplitude are implicitly determined by training dynamics (e.g., batch processing, optimizer updates, and synchronization), making them task-dependent and less predictable.

The observed extreme modulation frequency of up to 6,000 Hz suggests that such loads have the potential to excite high-frequency resonance modes in DER systems, which are usually not considered in conventional stability analyses.

4.3 How the power fluctuation affects the power infrastructure?

4.3.1 Fundamental Model

To capture the interaction mechanism, we adopt an impedance-based model using Thévenin and Norton equivalents for the cascaded power supply system [Sun11]. The system consists of three components connected at the PCC: the main grid, the converter-based DERs, and the GPU cluster, as shown in Fig. 5.

The main grid is represented by a Thévenin equivalent, i.e., an ideal voltage source V_g with an equivalent impedance Z_g . The DER is modeled as a controlled current source with an output impedance Z_{der} shaped by its internal control loops. The GPU cluster behaves as a near constant-power load (CPL) at the PCC due to multi-stage power-electronic regulation [GPU25a]. CPL represents an intrinsic short-term electrical behavior, distinct from longer-term power fluctuation. Therefore, the GPU cluster is modeled as a workload-controlled current sink $-I_{ls}$ in parallel with a CPL impedance Z_l , where the negative sign of the controlled current indicates power absorption from the PCC rather than current injection into the grid.

In the subsequent subsections, we first clarify the mechanism by which GPUs respond to workload-dependent power demand, where workload-induced power fluctuations are manifested as current fluctuations and interact with the DER and grid impedance. We then analyze the internal vulnerabilities that make the system sensitive to such fluctuations. Finally, we explain the potential impacts caused by these fluctuations.

4.3.2 Power Adjustment Mechanism of GPU Clusters

At the board level, the single GPU's internal voltage regulator module (VRM) maintains a constant core voltage (V_{core}) within its control bandwidth (typically tens of kHz), forcing the power demand to be supplied primarily through rapid current adjustments. At the system level, upstream power conditioning stages regulate the intermediate DC bus voltage by modulating the AC-side current. This energy-balancing mechanism propagates the GPU's power variations to the grid interface in the form of current fluctuations.

4.3.3 Internal Vulnerability Analysis

1. The negative incremental resistance of the GPU clusters: The defining characteristic of the CPL is its negative incremental resistance (NIR), as small changes in voltage (Δv) result in inverse changes in current (Δi) to maintain constant power ($P = V \cdot I$). The impedance of the load Z_l in Figure 5 is approximated by a negative real part: $Z_l(s) \approx -V_l^2/P_l$. NIR could amplify disturbances through a positive feedback mechanism. With grid impedance Z_g , the PCC voltage is $V_{pcc} = V_g - I_g \cdot Z_g$. If V_{pcc} drops due to the increase of $I_{gpu} = I_l + I_{ls}$, the CPL further increases I to maintain P , which further decreases the voltage across Z_g , worsening the fluctuation, and vice versa. Meanwhile, the negative resistance could reduce the system damping, potentially degrading stability margins.

2. Control-induced frequency response of the DER output impedance: Unlike synchronous generators, DERs have negligible physical inertia, so their voltage and frequency responses rely mainly on control dynamics. It results in a complex characteristic frequency response $Z_{der}(s)$ and exhibits resonant behavior, where disturbances at certain frequencies can be amplified, leading to voltage and current distortions and oscillations.

Although modern systems employ control and protection mechanisms like virtual inertia emulation and over-current/voltage protection schemes, they are often limited by practical constraints, such as communication delays, measurement errors, parameter uncertainty, and insufficient coordination under high DER penetration. High-frequency fluctuations induced by GPU clusters can exceed their response bandwidth, making these mechanisms ineffective and exposing system vulnerabilities.

4.3.4 Validation

In this subsection, we validate the above mentioned internal vulnerabilities, respectively.

1. Validation of the CPL and NIR of the GPU clusters: We conduct real-world voltage and current measurement experiments on a GPU cluster testbed shown

in Fig. 6(a). Abrupt voltage perturbations are intentionally introduced by adjusting the transformer, and the corresponding root-mean-square (RMS) voltage and current are simultaneously measured at the electrical interface of the platform.

Experiment setup. The tested GPU cluster is hosted in a dedicated server room with standard industrial cooling infrastructure and is powered by a three-phase AC UPS (Envicool ER010L). The cluster comprises three GPU racks with approximately 100 GPUs, including NVIDIA A100, V100, and RTX-series. Power for the entire room is supplied through the distribution box in Figure 6(a). Therefore, measurements taken at the distribution box capture the collective electrical behavior of the GPU cluster and serve as a representative interface for this small-scale deployment. The measuring tools include the current probe ETA5301M, the voltage probe TR1005B and the oscilloscope RIGOL MSO8064. Signals are sampled at a rate of 10 MSa/s.

Result analysis. To mitigate incidental effects and improve statistical robustness, we applied abrupt voltage perturbations in ten independent trials, including five voltage sag events (approximately 225 V $\rightarrow$ 195 V) and five voltage swell events (approximately 195 V $\rightarrow$ 225 V). The normalized statistical I - V and P - V trajectory results are depicted in Figure 6(b).

As illustrated, the RMS current exhibits an inverse dependence on the RMS voltage, with measured samples closely distributed around the theoretical CPL reference curve ($I \propto 1/V$). The power consumption data (green scatter points) remain tightly concentrated around the nominal baseline (green reference line). The mean error of the power measurements is only 0.0151, with a standard deviation of $\sigma = 0.0123$. In other words, voltage steps exceeding 10% induce power variations of merely 1%, providing empirical evidence of short-term constant-power behavior and NIR characteristics.

The slight dispersion and occasional outliers observed in the data points can be attributed to the composite nature of the data center. In addition to the primary GPU computational loads, the system inevitably incorporates non-CPL auxiliary components, such as resistive lighting elements and frequency-controlled cooling fans. These loads introduce a minor constant-impedance behavior, accounting for the measured variances. Specifically, points in the shaded dispersion band ($\pm 1\sigma$) reflect the inherent power variability.

2. Validation of the frequency response of the DER output impedance: We scan the impedance frequency response using the perturbation injection method, by injecting small-amplitude sinusoidal current perturbations at selected frequencies and extracting the corresponding output voltage components via FFT-based spectral analysis.

Experiment setup. As shown in Figure 7(a), the tested devices include the inverter (TI C2000 inverter development kit) [Ins14], and a programmable solar panel emulator (TEWERDTPV1000) to emulate its upstream solar panels. We use the SIGLENT SDG6032X-E signal generator to generate signals from 1 Hz to 10 kHz, use amplifier HPA-25W-272+ to amplify it to 25 W, and couple the signal into the circuit with the electromagnetic coupler EM5011. The data are acquired with the current probe ETA5301M, voltage probe TR1005B and the oscilloscope RIGOL MSO8064.

Result analysis. As shown in Fig. 7(b), the DER output impedance exhibits a frequency-dependent profile with resonant peaks at specific high frequencies, indicating increased sensitivity to high-frequency perturbations, i.e., “out-of-band” vulnerabilities. As a result, GPU-induced current fluctuations may amplify harmonic components, leading to power quality degradation and oscillations.

Fig. 7(b) also illustrates the interaction between the DER and grid impedance. According to the Middlebrook criterion, system stability depends on the ratio between grid impedance Z_g and the equivalent load impedance $Z_{dl} = Z_l \parallel Z_{der}$, with stability maintained when

$$|T_m(j\omega)| = \left| \frac{Z_g(j\omega)}{Z_{dl}(j\omega)} \right| < 1. \quad (1)$$

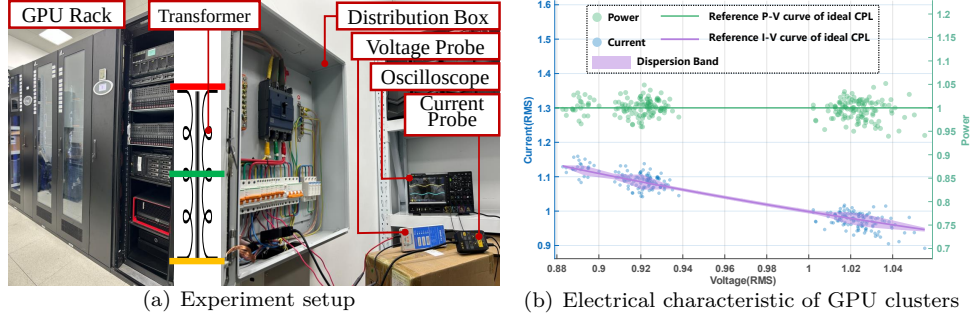


Figure 6: Validation of the CPL and NIR of the GPU clusters.

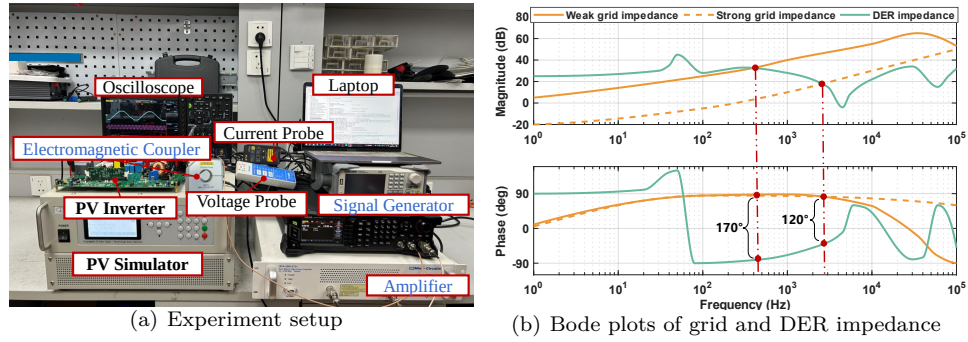


Figure 7: Validation of the frequency response of the DER output impedance.

The phase margin is defined as $PM = 180^\circ - |\angle T_m(j\omega_c)|$, where ω_c is the crossover frequency that satisfies $|T_m| = 1$, i.e., $|Z_g(j\omega_c)| = |Z_{dl}(j\omega_c)|$. As DER penetration increases, the system transitions toward a weak grid, significantly reducing the stability margin (e.g., from $\sim 60^\circ$ to $\sim 10^\circ$). The negative incremental resistance of GPU clusters further reduces the remaining margin, exacerbating instability.

4.3.5 Impact Analysis of the External Fluctuation

1. Power quality. The impact on power quality includes voltage excursions and harmonics. Due to the extremely fast power ramps of GPU clusters, the current at the PCC must change rapidly to maintain power balance. In regulated systems where voltage variations are tightly constrained, these fast current transients propagate through the network impedance Z_g and result in noticeable voltage disturbances: $\Delta V(t) = Z_g \cdot \Delta I(t)$.

Beyond voltage excursions, the high-frequency current variations dI/dt induced by GPU power ramps generate harmonic components that propagate through the network and degrade power quality. When these components align with resonant frequencies of DER control loops or grid elements (e.g., LCL filters), they can be significantly amplified.

2. System instability. In practical conditions, there are plenty of nonlinear elements that cannot be ignored when the system suffers from attacks, such as amplitude limiting, time delay, dead zone, saturation and complex power electronic switching and control (e.g., PLL) in DERs. The high-frequency power modulations from GPU clusters fundamentally alter the systems nonlinear operating regime. Let the nonlinear system be represented as $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mu)$, where $\mathbf{x}$ denotes the system state vector, and μ is the disturbance (in this case it is the power fluctuation of the GPU clusters).

Typically, it operates around an equilibrium $\mathbf{x}_0$, whose stability is determined by the

Table 2: Electrical parameters of the Simulink model.

Block / stage	Value / description
PV array	$N_{\text{series}} = 100$, $N_{\text{parallel}} = 47$; per-module $P_m = 214.97$ W.
PV-side boost filter	Input shunt capacitor $C = 4.7$ mF, ESR = 1 m Ω ; boost inductor $L = 20$ mH, $R = 1$ m Ω ; output RC branch $C = 4.7$ mF, ESR = 1 m Ω .
Grid source	400 V, 50 Hz; source resistance = 10^{-5} Ω ; short-circuit level = 100 MVA.
Feeder line	Length = 5 km; $r = [0.01273, 0.3864]$ Ω , $l = [0.9337, 4.1264]$ mH, $c = [12, 12]$ nF.
Bridges	$R_{\text{on}} = 1$ m Ω ; snubber resistance = 100 k Ω ; snubber capacitance = ∞ .
UPS output filter	Per-phase RL filter with $R = 0.05$ Ω and $L = 2$ mH.

Table 3: DER, UPS and SPS control settings in the Simulink model.

Loop / block	Value / description
PV DC/DC duty control	PI Controller with $K_p = 0.001$, $K_i = 0.01$; $f_s = 5$ kHz.
PLL PI regulator	$K_p = 10$, $K_i = 5 \times 10^4$; q -axis voltage error as the locking signal
PV inverter dq outer loop	Voltage PI Controller: $K_p = 0.1$, $K_i = 100$; $V_{PV,d}^* = 285$ V, $V_{PV,q}^* = 0$ V.
PV inverter dq inner loop	Current PI Controller with $K_p = 30$, $K_i = 200$.
UPS rectifier outer loop	DC-bus voltage PI Controller with $K_p = 5$, $K_i = 0.001$; $V_{ref}^* = 500$ V.
UPS rectifier inner dq loop	Current PI Controller with $K_p = 1$, $K_i = 0.1$; reference $i_q^* = 0$.
UPS DC/DC loop	DC-bus voltage PI Controller with $K_p = 0.002$, $K_i = 0.05$; reference $V_{\text{rectiDC}}^* = 850$ V; $f_s = 20$ kHz.
UPS inverter outer loop	Voltage PI Controller with $K_p = 0.01$, $K_i = 12$; reference $V_Q^* = 0$.
UPS inverter inner loop	Current PI Controller with $K_p = 1.0$, $K_i = 120$.
SPS rectifier outer loop	DC-bus voltage PI Controller with $K_p = 0.18$, $K_i = 6.0$; $V_{ref}^* = 500$ V.
SPS rectifier inner loop	Current PI Controller with $K_p = 1.2$, $K_i = 20.0$; reference $i_q^* = 0$.

Jacobian matrix $\mathbf{J} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}|_{\mathbf{x}_0, \mu_0}$. Under the GPU-induced power modulations, the effective operating point of the system shifts, modifying the Jacobian matrix $\mathbf{J}(\mu)$ and dominant eigenvalues $\lambda_{1,2}(\mu) = \alpha \pm j\beta(\mu)$. A critical transition is triggered when μ reaches a threshold μ_c satisfying the Hopf bifurcation conditions: $\alpha(\mu_c) = 0$, $\beta(\mu_c) = \omega_{osc} \neq 0$ and $\frac{d\alpha}{d\mu}|_{\mu_c} > 0$. Physically, this reflects the reduction of effective damping due to the interaction between GPU-induced load dynamics and DER control loops.

4.4 Watt2Bit Risk

The above analysis focuses on the impact of GPU power manipulation on power systems. A critical yet overlooked question arises: *can the induced power disturbances, in turn, affect the computing domain?* The answer is yes and we call it the **Watt2Bit** risk.

DoS. The cyberphysical interaction does not terminate at the power system boundary, but forms a closed-loop cyberphysical vulnerability. Due to the nonlinear switching mode of the power supply in data centers, the mechanisms behind the **Watt2Bit** can be complex and various; here we illustrate it via harmonic distortions. High-frequency harmonics can lead to elevated resistive losses and increased thermal stress within power conversion stages due to increased RMS current and the frequency-dependent rise in equivalent series resistance (ESR). This may trigger protection mechanisms (e.g., over-temperature or over-current), interrupting GPU operation and causing denial of service (DoS). Consequently, watt-level electrical disturbances propagate into bit-level computational disruptions.

Exfiltration. The attack may also act as an side-channel exfiltration vector. In particular, an adversary could encode bits by modulating either the attack frequency or the attack amplitude, thereby inducing distinguishable patterns in the measured power/EMI traces. A receiver observing these side-channel emissions could then demodulate the corresponding spectral or amplitude features to recover the transmitted symbols.

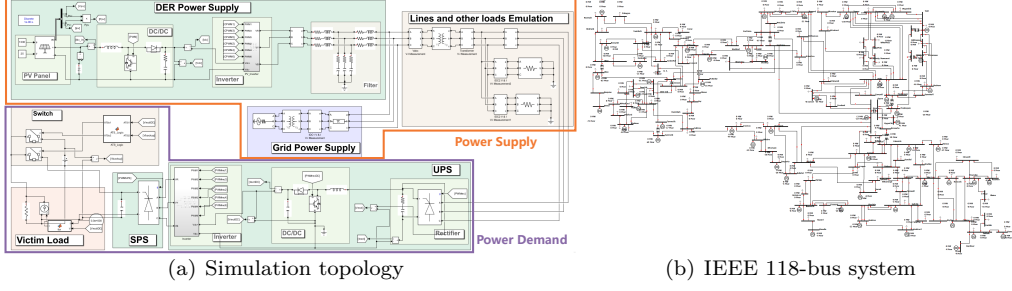


Figure 8: System topologies used in the simulation.

5 Experiments

We assess the impact of **Bit2Watt** on local power infrastructure and investigate the scale at which the GPU power manipulation leads to tangible physical effects. We begin with simulation-based analysis to characterize the system-level response across different attack configurations. Guided by the results, we conduct real-world experiments on a locally deployed power system to corroborate simulation findings and confirm the physical realizability.

5.1 Simulation Evaluation

GPU load aggregation setup. We adopt the NVIDIA RTX 3090 as the reference GPU model, and simulate a local power system with an aggregate load of 1 MW, of which approximately one quarter consists of the victim GPU loads and the remaining three quarters comprise background loads such as household appliances. This composition is motivated by electricity consumption patterns in Virginia, which hosts the worlds highest concentration of data centers [KZ24]. The simulation information can be found in Figure 8, Table 2 and Table 3.

To evaluate the worst-case impact on the grid, the GPU units are aggregated under a synchronized assumption, where all units operate with identical modulation frequency and phase. In Simulink, this aggregated behavior is emulated by a controlled current source driven by a **MATLAB Function** block, defined as

$$I_{\text{GPU}}(t) = -\min\left(\frac{P_{\text{GPU}}(t)}{\max(|V_{\text{SPS}}|, 80)}, 10\right) \text{ A},$$

where $P_{\text{GPU}}(t) = 50 + 200 \sin(2\pi f_{\text{att}} t)$ kW, and V_{SPS} denotes the local DC voltage used for scaling the load current. Device-level heterogeneity, phase dispersion, and communication delays across individual GPU units are not explicitly modeled and thus the reported results provide a conservative bound on realistic system behavior.

Result analysis. The simulation results in Figures 9(a), 9(b), and 9(c) collectively demonstrate that coordinated GPU power manipulation can simultaneously induce the voltage excursion, harmonics, and instability of a local power system, with these effects being significantly exacerbated by high levels of DER penetration. Taking a 50% DER penetration level as an example, manipulating only 500, 400, and 900 GPUs is sufficient to degrade voltage quality, increase current harmonic distortion, and compromise system stability beyond the thresholds specified by relevant standards [Ass18, Int11].

Figures 9(d) and 9(e) show the current waveforms and harmonic spectra under 50% DER penetration with 1,000 GPUs manipulated. GPU power modulation induces severe current distortion, producing a sharp, peaky waveform with a high crest factor that increasingly deviates from the nominal sinusoid as the attack frequency rises. In contrast,

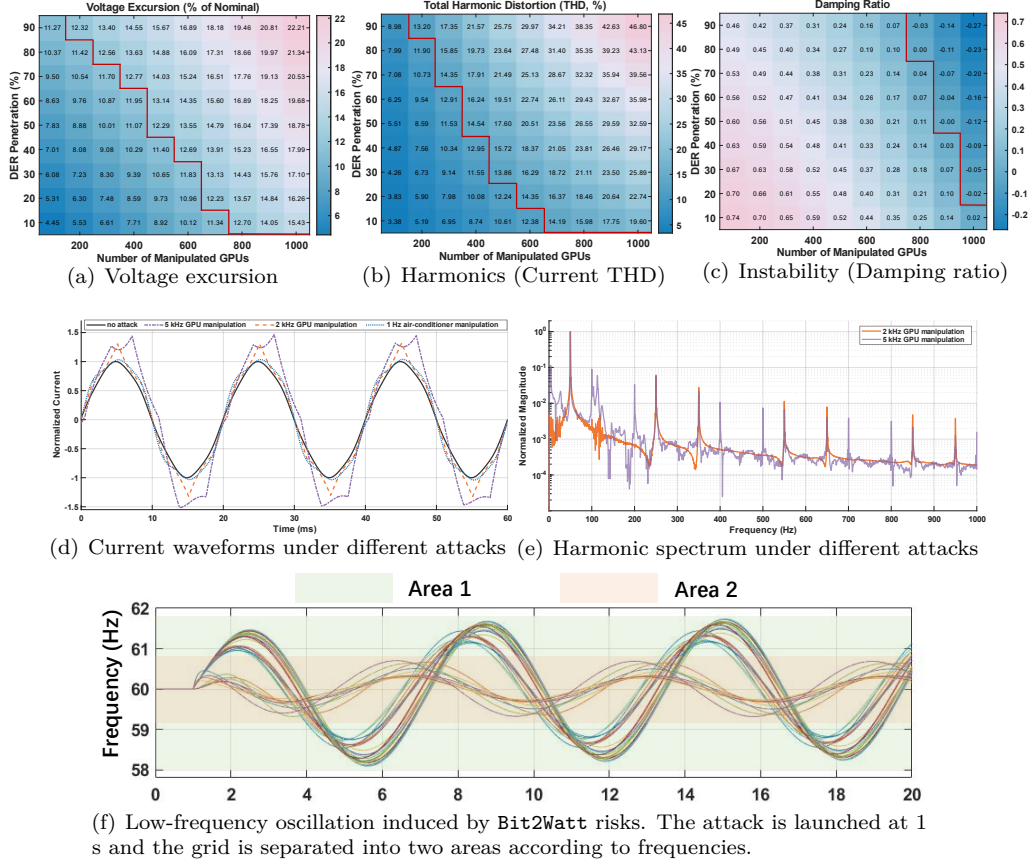


Figure 9: Impact of Bit2Watt risks on a 1-MW local power system. The red lines in (a) (b) (c) mark the standard-specified thresholds, separating values above and below them.

low-frequency disturbances analogous to air-conditioner loads (blue dotted lines) produce negligible effects. Physically, the observed pulsating current reflects the GPU clusters rapid power ramps, which draw short, pulse-like currents during fast load transitions.

In Figure 9(e), a distribution of high-order harmonics is observed and a persistent cluster of high-frequency components extends across the spectrum, contributing to the substantial rise in total harmonic distortion (THD). Under a 90% DER penetration level, the calculated THD reaches an extreme value of 46.80% when 1000 GPUs are manipulated. Even at a lower penetration of 10%, the THD remains as high as 19.60%, both of which substantially exceed the 13% guideline recommended by IEC 61000-3-12 [Int11].

More critically, at 90% DER penetration, the damping ratio drops from 0.46 with 200 manipulated GPUs to 0.27 with 1,000 GPUs, marking a transition from a well-damped to an unstable regime. This instability emerges as the NIR induced by coordinated GPU power modulation progressively overwhelms the inherent damping of the grid and inverter controls. A critical threshold appears at approximately 800 manipulated GPUs under 80% DER penetration, beyond which the damping ratio becomes negative and the system enters an oscillatory regime.

Based on the Simulink results, we further examine area oscillations using the IEEE 118-bus system (Figure 8(b)). Under attacks, the system exhibits oscillatory behavior driven by GPU-induced power variations. As the damping ratio crosses zero, the response transitions from stable to oscillatory. As shown in Fig. 9(f), this leads to frequency bifurcation, where the network splits into two coherent frequency clusters. The resulting

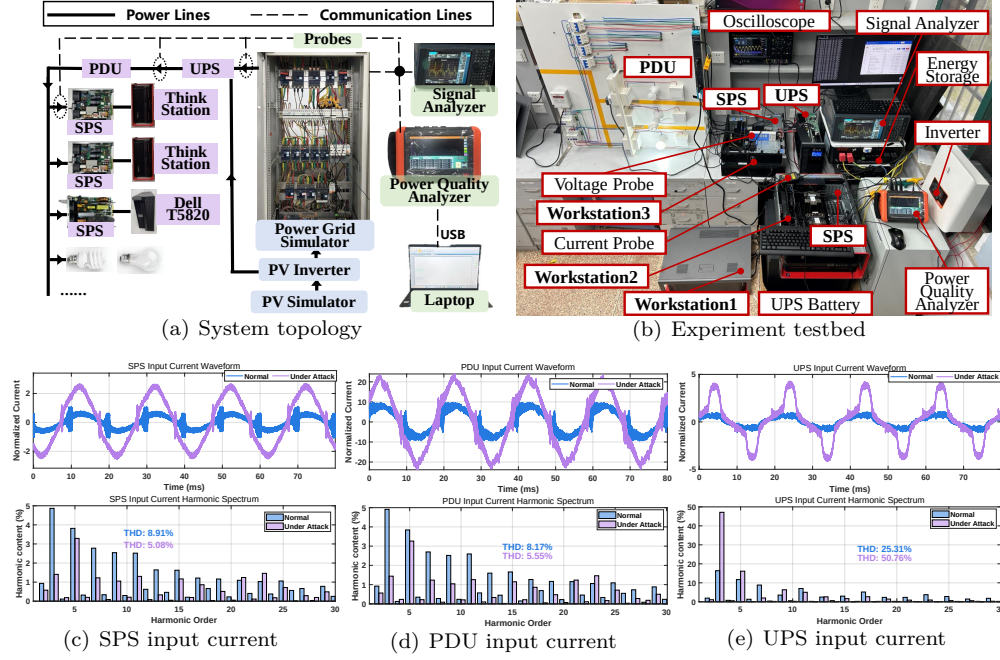


Figure 10: Results of the hardware experiments.

loss of synchronism indicates transient grid partitioning, which may trigger widespread protection actions and potentially lead to a black-start condition.

5.2 Real-world Experiments

Experiment setup. We build a local power grid testbed that emulates a small-scale, inverter-dominated distribution system, as illustrated in Figure 10(a). The system consists of a power grid simulator that provides a 220 V AC supply, a PV simulator, a grid-tied PV inverter, an energy storage battery, and a diverse set of electrical loads. These loads include typical household appliances (e.g., water heaters, bulbs) as well as three workstations with six GPUs. The total power demand of the testbed is approximately 8 kW, with roughly 40% supplied by the PV units, resulting in a mixed gridDER operating condition. Electrical measurements are collected and analyzed using a handheld power quality analyzer (VICTOR 5000M) and KEYSIGHT EXA Signal Analyzer N9010B.

The prototype power supply chain of the workstations consists of GPUSPSPDUUPS PCC, as shown in Figure 10(b). The setup includes two ThinkStation PX units powered by dedicated SPS modules (Delta DPS-1850CB A, 1850 W), and one Dell T5820 powered by an SPS module (Dell H950EF-00, 950 W). These loads are aggregated at the PDU, where additional equipment is connected to emulate background disturbance loads. The PDU is implemented using an energy monitoring device (DELIXI DDS606), circuit breakers (DELIXI HDBE-63), and power strips. An AC UPS (SHANKE C3KS), equipped with eight 12 V battery cells connected in series, interfaces the system with the PCC.

Ethical Considerations. All experiments were conducted in controlled environments. No attacks were performed on production systems, and no vulnerabilities in specific commercial products are disclosed.

Result analysis. Figures 10(c), 10(d), and 10(e) compare current waveforms and harmonic spectra across the cascaded power delivery chain under normal and attack conditions. At all stages, the attack significantly increases the current magnitude. The THD

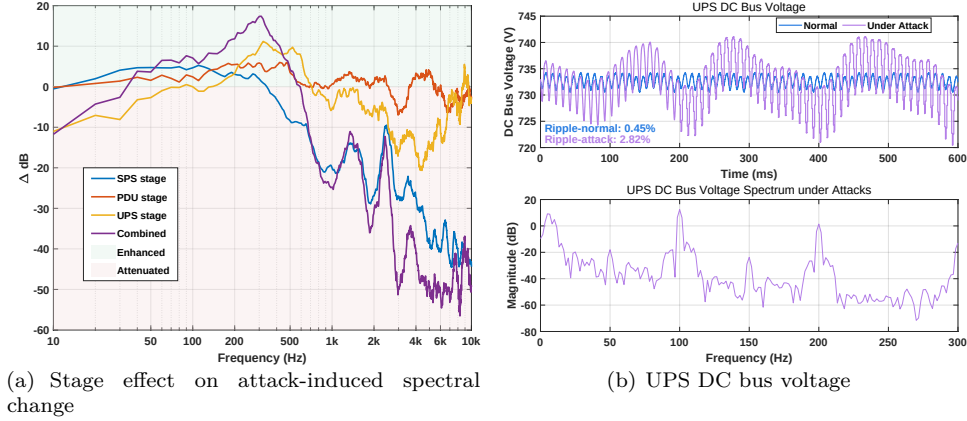


Figure 11: Spectral attenuation and reshaping across the power delivery chain.

at the SPS and PDU stages decreases, mainly due to the increased fundamental current and the low-pass filtering effect of the SPS. However, low-order harmonics (e.g., 3rd and 5th) sharply increase at the UPS stage. This indicates that intermediate stages not only attenuate high-frequency components but also reshape the disturbance through control dynamics, ultimately projecting the distortion onto the upstream grid.

5.2.1 Signal Propagation through Intermediate Power Electronics

In this section, we quantify how the attack-induced spectral increment propagates across intermediate stages. For each stage s and frequency f , we define $\Delta_s(f) := M_s^{\text{atk}}(f) - M_s^{\text{base}}(f)$, where $M_s^{\text{atk}}(f)$ and $M_s^{\text{base}}(f)$ denote the spectral magnitudes (in dB) at stage s under attack and attack-free conditions, respectively. Figure 11(a) shows the stage-induced variation of this increment, i.e., $\Gamma_s(f) := \Delta_s(f) - \Delta_{s-1}(f)$, where $s-1$ denotes the preceding stage. Hence, $\Gamma_s(f) > 0$ indicates that stage s enhances the attack-induced spectral increment at frequency f , whereas $\Gamma_s(f) < 0$ indicates attenuation.

Figure 11(a) reveals a clear frequency-dependent behavior. The SPS stage introduces the strongest high-frequency attenuation, indicating that a large portion of the high-frequency modulation is absorbed locally. This is consistent with the SPS internal structure, where DC-bus capacitors and local power-conditioning circuits provide strong buffering and smoothing effects against fast fluctuations. In contrast, low-frequency components below approximately 300 Hz are mildly enhanced. The PDU stage has a relatively weak impact, consistent with its largely passive role.

The UPS stage exhibits mixed behavior, with attenuation at high frequencies but enhancement in the low-frequency range (200–700 Hz), indicating a reshaping effect. The complex UPS control system plays a decisive role here. Unlike the unidirectional regulation and straightforward control logic of an SPS, the UPS utilizes tightly coupled, cascaded feedback loops to synchronize the input rectifier with the output inverter. As shown in Figure 11(b), the attack induces pronounced low-frequency oscillations on the UPS DC bus voltage, increasing the ripple ratio from 0.45% to 2.82%, which exceeds the suggested limit [ABB15]. This is because rapid GPU power transients disrupt the nominal constant-power mode of the UPS and force it into current-limiting mode. It indicates that, although high-frequency components are suppressed upstream, the energy is converted into UPS DC-bus-level power fluctuations, which in turn distort the input current waveform.

Overall, the results support an attenuation-and-reshaping mechanism: the attack-induced spectral increment is selectively suppressed at high frequencies and reshaped at some

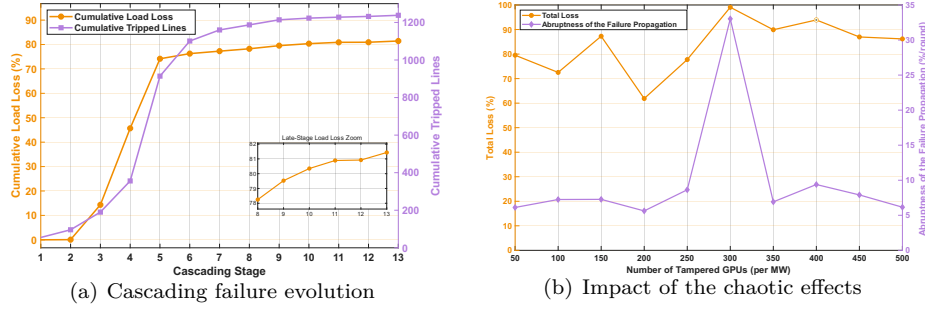


Figure 12: Cascading results in the 9241-bus European transmission network and associated chaotic effects.

lower-frequency components along the intermediate power electronics.

5.3 Discussions

5.3.1 From Local to Wide-Area Power Infrastructure

We extend the analysis from individual data-center interfaces to a large-scale transmission network with simulations on a modified 9241-bus European transmission network representative of modern DER-rich grids [FPCW13]. As shown in Figure 12(a), a localized disturbance (accounting for only 2% of the total system load) triggers a cascade over 13 stages. Both the number of tripped lines and load loss increase sharply in the early stages, indicating a fast propagation phase, followed by a slower saturation regime. In total, 1,238 transmission lines are disconnected, leading to severe network fragmentation and a final load loss of approximately 81%. These results demonstrate that disturbances originating at localized interfaces, when coupled with protection actions, can propagate through interconnected networks and result in large-scale cascading failures.

5.3.2 Influence of Noisy Environments and Chaotic Effects

The Bit2Watt attack may be degraded by environmental noise, which affects the synchronization of distributed GPUs, as timing jitter directly determines the efficacy of constructive power-wave superposition. We model the asynchronization $\Delta\tau_i$ of each victim GPU as a Gaussian random variable, i.e., $\Delta\tau_i \sim \mathcal{N}(0, \sigma_\tau^2)$, capturing the combined effects of network latency and OS scheduling jitter. Such perturbations reduce the aggregated attack amplitude due to imperfect phase alignment. For a 2 kHz modulation, when $\sigma_\tau < 50 \mu\text{s}$, the reduction in attack amplitude is limited, although the waveform gradually transitions from a square-like shape to a more sinusoidal profile. When $\sigma_\tau = 100 \mu\text{s}$, the attack amplitude decreases by approximately 20%. The corresponding waveforms are shown in the appendix. The robustness of the attack decreases as the frequency increases. However, this loss in aggregation efficiency caused by asynchronization can be compensated by increasing the number of victim devices, owing to the scaling effect of large populations.

Beyond external noise, the power-electronic-dominated system exhibits strong nonlinear dynamics that may lead to bifurcations and chaotic effects. In such regimes, the system can appear non-deterministic in practice and become sensitive under certain settings. To quantify the abruptness of failure propagation, we introduce a cascading progression rate, defined as the ratio between total load loss and the number of cascading rounds, which correlates with the temporal evolution of the cascade. Lower abruptness implies a slower progression, providing operators with more time for mitigation. As shown in Figure

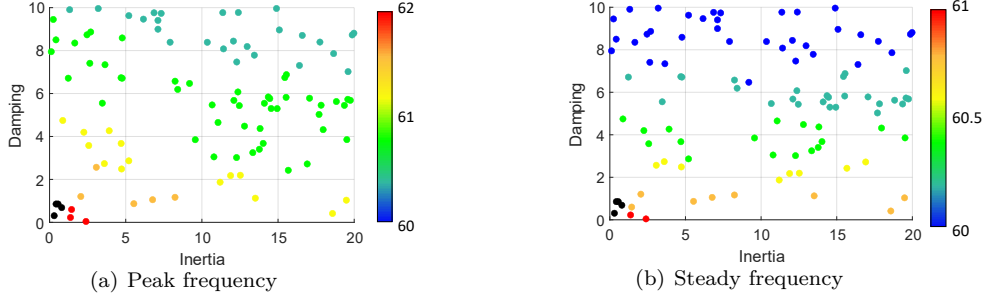


Figure 13: Frequency as a function of inertia and damping. The frequency values are color-coded, with cool colors representing low values and hot colors indicating high values. Black indicates that the grid becomes destabilized under this combination.

12(b), both the total load loss and the propagation abruptness exhibit sharp extrema when the number of tampered GPUs reaches 300/MW, indicating strong sensitivity to the initial attack configuration. The cascading process also shows irregular and unpredictable line disconnections, reflecting chaos-like behavior near the stability boundary. A similar sensitivity is observed with respect to attack timing, which alters the initial system states (e.g., voltage magnitudes and phase angles). Under such conditions, the system may exhibit pseudo-stable behavior, such as bounded oscillations (Fig. 9(f)) that appear stable but do not converge to an equilibrium. This variability highlights a double-edged effect of the underlying nonlinear dynamics: near transition boundaries, small perturbations can trigger disproportionately large instability, while the same complexity limits the attackers ability to optimize attack strategies based on partial system knowledge.

5.3.3 Influence of System Parameters: Inertia and Damping

Inertia fundamentally governs the systems immediate frequency response by determining how rapidly frequency deviates following a disturbance, a role that becomes increasingly critical as high DER penetration displaces synchronous generation. Damping dictates the systems ability to attenuate post-disturbance oscillations and restore stable operation.

To examine their relative influence, we conduct simulations on the IEEE 118-bus system using 100 different inertiadamping combinations. Figure 13 shows the peak and steady-state frequency following the attack. Increasing either inertia or damping improves frequency stability. Notably, damping exhibits a stronger stabilizing effect than inertia. As illustrated in Figure 13(a), the peak frequency deviation decreases more rapidly along the damping axis than along the inertia axis. This indicates that, under GPU power manipulations, system stability is more sensitive to damping degradation than to inertia reduction, highlighting the critical role of damping in inverter-dominated power systems.

5.3.4 Detectability and Stealth Analysis

To evaluate the detectability and stealth under realistic monitoring, we conduct the quantitative study summarized in Table 4 and Figure 14. Table 4 reviews representative cloud- and facility-side monitoring interfaces, including NVML, RAPL, Intel Power Gadget, BMC/IPMI, and rack-level PDU telemetry. These interfaces are primarily designed for operational monitoring and logging rather than adversarial detection, and they typically expose coarse-grained, temporally averaged statistics instead of high-bandwidth instantaneous waveforms. Figure 14(a) corroborates this observation empirically: the Bit2Watt signature is progressively smoothed from GPU-side telemetry to CPU-, server-,

Table 4: Typical power monitoring tools related to Bit2Watt.

Tool	Scope	Achieved Sampling Frequency	Related Data	Access Method
AIDA64	GPU, CPU	10 Hz	Power, voltage, current	User-level software
NVML	GPU	450 Hz	Power, utilization, temperature	<code>nvidia-smi</code> command
RAPL	CPU	1 kHz	Cumulative energy	<code>powercap/intel-rapl</code>
Intel Power Gadget 3.6	CPU	10 Hz	Package power, frequency, temperature	User-level GUI
BMC	Server level	1 kHz	Power, voltage, current	<code>ipmitool</code> command
PDU	Rack level	1 Hz	Power, energy, voltage, current	User-level GUI

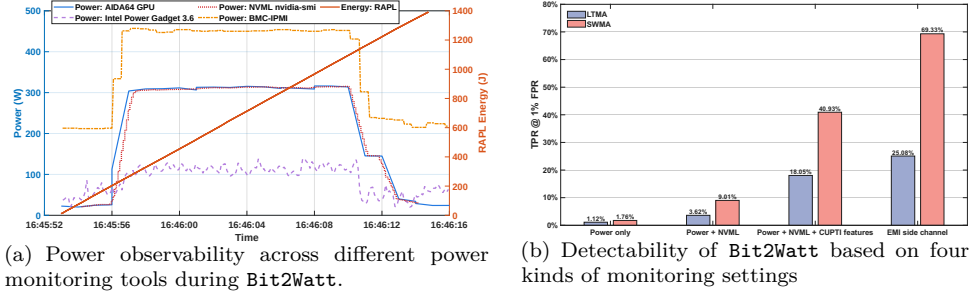


Figure 14: Detectability and stealth evaluation.

and rack-level monitoring. Since no established detector specifically targets this attack class, we implement a lightweight logistic-regression baseline and evaluate four settings: power-only monitoring, power combined with NVML runtime features, power combined with NVML and CUPTI features, and EMI side channel sensing from the power line and VRM. As shown in Figure 14(b), coarse cloud/facility-side telemetry provides only limited detectability, and LTMA is consistently harder to detect than SWMA under the same detector family. In contrast, dedicated physical-side sensing substantially improves detection. These results suggest that the mere availability of monitoring data does not directly translate into reliable detection of Bit2Watt-style attacks; rather, detectability depends critically on the monitoring layer, temporal granularity, and whether the defender deploys profiling-aware or physical-layer instrumentation beyond routine operational telemetry.

5.3.5 Watt2Bit Risk

Figure 15(a) presents the normalized cumulative heating under nominal and attack conditions. Under normal operation, the cumulative heating remains close to unity, indicating that the contribution of higher-order harmonics is limited. In contrast, the attack-induced spectrum produces a clear and progressively widening deviation, resulting in an increase of approximately 17.8% when harmonics up to the 50th order are considered, as highlighted in Figure 15(a). This amplification is not merely a consequence of increased current magnitude, but rather stems from the frequency-dependent growth of the equivalent series resistance (ESR). At elevated harmonic frequencies, skin and proximity effects significantly increase the effective resistance, thereby leading to disproportionate thermal stress and causing localized overheating, accelerated aging, and degradation of power-electronic components, including UPSs and PDUs.

Figure 15(b) illustrates the trip-time characteristics of IEC 60255 IDMT protection relays. The results show that the elevated RMS current induced by GPU power manipulation can readily trigger these protections. For example, under the Standard Inverse curve,

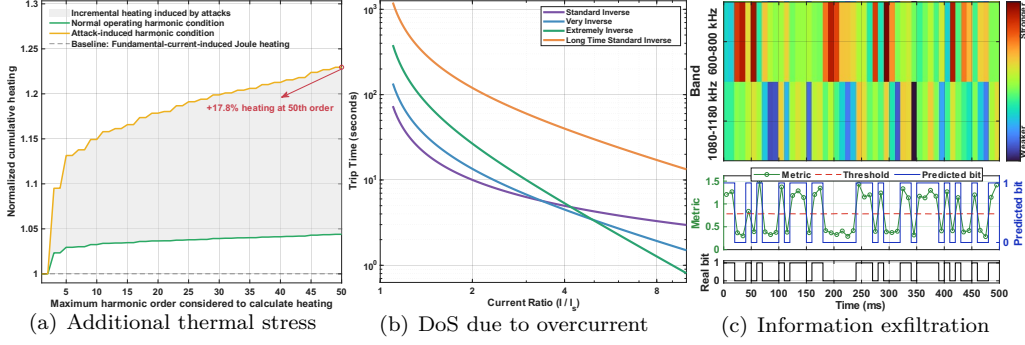


Figure 15: Watt2Bit risk analysis.

a current doubling leads to a trip within approximately two seconds. It manifests as DoS events, abruptly shutting down GPU servers and terminating active workloads. In large-scale training or inference, even brief interruptions can cause task failures and cascading restarts, allowing watt-level disturbances to propagate into bit-level service disruptions.

In our exfiltration experiment, binary bits are frequency-shift keying (FSK)-encoded on the GPU load profile using 2 kHz (“1”) and 200 Hz (“0”) modulation frequencies, with a 10 ms bit duration. EMI traces captured via a near-field antenna-coupled USRP B210 reveal that while the absolute spectral power lacks discernible regularity, as shown in the heatmap in Figure 15(c), the relative energy ratio $M = E_{\text{upper}}/E_{\text{lower}}$ (E_{lower} : 600–800 kHz; E_{upper} : 1080–1180 kHz) remains discriminative. By applying a threshold to a 10 ms sliding window, our decoder achieves an accuracy exceeding 99%. Figure 15(c) demonstrates the successful recovery of a 50-bit test sequence, where every transmitted bit is correctly identified with zero bit errors. It suggests that the power modulation could potentially serve as a feasible covert channel for clandestine information exfiltration.

5.4 Countermeasures and Mitigation Strategies

Mitigating the proposed Bit2Watt attacks requires coordinated defenses across both the cyber and physical layers, as the vulnerability arises from their tight coupling. From the information layer, proactive detection of malicious computational behaviors is essential. This includes identifying abnormal GPU utilization patterns, synchronized workload execution, or anomalous training schedules. Runtime monitoring of power-aware performance counters, combined with anomaly detection, can provide early warning of coordinated power modulation attempts. In addition, integrity verification of training pipelines, model update processes, and job schedulers can reduce the risk of malicious code or adversarial training logic being embedded into large-scale computing workloads.

At the physical layer, enhancing the robustness of data center power infrastructure is equally critical. Local energy buffering mechanisms, such as battery energy storage systems (BESS) and supercapacitors, can effectively absorb fast power fluctuations and high-frequency harmonic energy induced by GPU power modulations [Gen25]. Properly sized filtering components and active power conditioning devices can further attenuate harmonic currents and limit their propagation into upstream networks.

Importantly, these countermeasures should be jointly designed and coordinated across cyber and physical domains. Isolated defenses at either layer are insufficient to address the closed-loop nature of the Bit2Watt risk. A cross-layer security paradigm that integrates computing workload monitoring with power system protection and control is therefore essential to ensure the resilient operation of future data centers and high-renewable power grids, which constitutes future work.

6 Conclusion

This paper exposes Bit2Watt, a cyberphysical vulnerability arising from the tight coupling between GPU workloads and renewable-integrated power systems. Coordinated manipulation of legitimate GPU tasks can inject structured power fluctuations, exploiting the negative incremental resistance of constant-power loads and the frequency-shaped impedance of inverter-based DERs. The attack can ultimately degrade power quality, reduce system damping, and potentially trigger cascading failures. The attack requires no compromise of grid or computing components, operating entirely as a legitimate tenant, and can propagate disturbances back to data centers, causing forced workload interruptions, termed Watt2Bit. These findings underscore a fundamental shift: as power and computing infrastructures converge, security must be addressed across domains, requiring coordinated defenses that consider workload behavior, power electronics, and grid dynamics.

Acknowledgments

The authors are grateful to Runmin Ou and Yiqi Chen for their assistance with the experiments, and to Dr. Chen Yan of the Ubiquitous System Security Laboratory (US-SLab) for his guidance throughout this work. This work was supported by the National Natural Science Foundation of China under Grant 62571474, Grant 62201501 and Grant U25B20188.

References

- [ABB15] ABB. *Guideform Specification for ABB Conceptpower DPA 500 480V, UL UPS System*, 2015.
- [Ass18] IEEE Standards Association. IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems. *IEEE Std 1547-2018 (Revision of IEEE Std 1547-2003)*, 2018.
- [Bee25] Marie Beerens. NVIDIA Targets AI Data Centers And All-Renewable Electricity. <https://www.investors.com/news/nvidia-targets-ai-data-centers-and-all-renewable-electricity/>, 2025.
- [Ber25] Publication under the Energy Industry Act (Energiewirtschaftsgesetz - EnWG). <https://www.stromnetz.berlin/en/about-us/disclosure-requirements/enwg/>, 2025.
- [CWH⁺25] Esha Choukse, Brijesh Warriar, Scot Heath, Luz Belmont, April Zhao, Hassan Ali Khan, Brian Harry, Matthew Kappel, Russell J Hewett, Kushal Datta, et al. Power stabilization for AI training datacenters. *arXiv preprint arXiv:2508.14318*, 2025.
- [DC25] Hannah Beckler Dakin Campbell. See where data centers are across the US on our interactive map. <https://www.businessinsider.com/data-center-locations-us-map-ai-boom-2025-9>, 2025.
- [DPO25] Daniel Nishball Dylan Patel and Jeremie Eliahou Ontiveros. Gigawatt Clusters, Telecom Networking, Long Haul Fiber, Hierarchical & Asynchronous SGD, Distributed Infrastructure Winners. <https://semianalysis.com/2024/09/04/multi-datacenter-training-openais/>, 2025.

- [DUW17] Adrian Dabrowski, Johanna Ullrich, and Edgar R. Weippl. Grid Shock: Coordinated Load-Changing Attacks on Power Grids: The Non-Smart Power Grid is Vulnerable to Cyber Attacks as Well. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, pages 303–314, Orlando FL USA, December 2017. ACM.
- [FPCW13] Stéphane Fliscounakis, Patrick Panciatici, Florin Capitanescu, and Louis Wehenkel. Contingency ranking with respect to overloads in very large power systems taking into account uncertainty, preventive, and corrective actions. *IEEE Transactions on Power Systems*, 28(4):4909–4917, 2013.
- [GBO23] Wedan Emmanuel Gnibga, Anne Blavette, and Anne-Cécile Orgerie. Renewable energy in data centers: the dilemma of electrical grid dependency and autonomy costs. *IEEE Transactions on Sustainable Computing*, 9(3):315–328, 2023.
- [Gen25] Dina Genkina. Will Supercapacitors Come to AIs Rescue? <https://spectrum.ieee.org/supercapacitor-2671883490/>, 2025.
- [GKMA22] Mohsen Ghafouri, Ekram Kabir, Bassam Moussa, and Chadi Assi. Coordinated charging and discharging of electric vehicles: A new class of switching attacks. *ACM Transactions on Cyber-Physical Systems (TCPS)*, 6(3):1–26, 2022.
- [GPU25a] Constant Power Loads. https://www.cat.com/en_US/by-industry/electric-power/Articles/White-papers/constant-power-loads.html, 2025.
- [GPU25b] NVIDIA H100 Price Guide 2025: Detailed Costs, Comparisons & Expert Insights. <https://docs.jarvislabs.ai/blog/h100-price#2-cloud-gpu-pricing-per-hour>, 2025.
- [GPU25c] Rent GPUs for AI Development. <https://jarvislabs.ai/>, 2025.
- [Ins14] Texas Instruments. C2000 Solar Micro Inverter, Quick Start Guide. <https://www.ti.com/lit/ug/tidu406/tidu406.pdf?ts=1766971345992>, 2014.
- [Int11] International Electrotechnical Commission. Electromagnetic compatibility (EMC) - Part 3-12: Limits - Limits for harmonic currents produced by equipment connected to public low-voltage systems with input current > 16 A and ≤ 75 A per phase. Standard IEC 61000-3-12:2011, IEC, Geneva, Switzerland, 2011.
- [JFK17] Zhen Hang Jiang, Yungsi Fei, and David Kaeli. A novel side-channel timing attack on GPUs. In *Proceedings of the Great Lakes Symposium on VLSI 2017*, pages 167–172, 2017.
- [Kwo20] Soongeol Kwon. Ensuring renewable energy utilization with quality of service guarantee for energy-efficient data center operations. *Applied Energy*, 276:115424, 2020.
- [KYZC16] Kibaek Kim, Fan Yang, Victor M Zavala, and Andrew A Chien. Data centers as dispatchable loads to harness stranded power. *IEEE Transactions on Sustainable Energy*, 8(1):208–218, 2016.
- [KZ24] Sabrina Lam Kayla Zhu, Niccolo Conte. Mapped: Data Center Electricity Consumption By State. <https://www.visualcapitalist.com/mapped-data-center-electricity-consumption-by-state/>, 2024.

- [LC25] Siddharth Singh Laura Cozzi, Thomas Spencer. *Energy and AI*. International Energy Agency, 2025.
- [LL25] Yuzhuo Li and Yunwei Li. AI Load Dynamics—A Power Electronics Perspective. *arXiv preprint arXiv:2502.01647*, 2025.
- [LNC⁺24] Imran Latif, Alex C Newkirk, Matthew R Carbone, Arslan Munir, Yuewei Lin, Jonathan Koomey, Xi Yu, and Zhiuha Dong. Empirical Measurements of AI Training Power Demand on a GPU-Accelerated Node. *arXiv preprint arXiv:2412.08602*, 2024.
- [LWZ⁺16] Hongtao Lei, Rui Wang, Tao Zhang, Yajie Liu, and Yabing Zha. A multi-objective co-evolutionary algorithm for energy-efficient scheduling on a green data center. *Computers & Operations Research*, 75:103–117, 2016.
- [MBA⁺23] Suman Maiti, Anjana Balabhaskara, Sunandan Adhikary, Ipsita Koley, and Soumyajit Dey. Targeted attack synthesis for smart grid vulnerability analysis. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 2576–2590, 2023.
- [NG23] Seyed Morteza Nabavinejad and Tian Guo. Opportunities of renewable energy powered DNN inference. *arXiv preprint arXiv:2306.12247*, 2023.
- [NNQAG18] Hoda Naghibijouybari, Ajaya Neupane, Zhiyun Qian, and Nael Abu-Ghazaleh. Rendered insecure: GPU side channel attacks are practical. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 2139–2153, 2018.
- [Reu25] Reuters. Meta signs deals to source more solar, wind power for data centers. <https://www.reuters.com/sustainability/invenenergy-supply-meta-with-additional-791-mw-data-center-operations-2025-06-26/>, 2025.
- [SCB22] Tohid Shekari, Alvaro A Cardenas, and Raheem Beyah. MaDIoT 2.0: Modern High-Wattage IoT botnet attacks and defenses. In *31st USENIX security symposium (USENIX Security 22)*, pages 3539–3556, 2022.
- [SMP18] Saleh Soltan, Prateek Mittal, and H Vincent Poor. BlackIoT:IoT botnet of high wattage devices can disrupt the power grid. In *27th USENIX security symposium (USENIX Security 18)*, pages 15–32, 2018.
- [Sun11] Jian Sun. Impedance-based stability criterion for grid-connected inverters. *IEEE Transactions on Power Electronics*, 26(11):3075–3078, 2011.
- [SWB⁺21] Siddharth Samsi, Matthew L Weiss, David Bestor, Baolin Li, Michael Jones, Albert Reuther, Daniel Edelman, William Arcand, Chansup Byun, John Holodnack, et al. The MIT Supercloud Dataset. In *2021 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–8. IEEE, 2021.
- [SZ21] Chao Su and Qingkai Zeng. Survey of CPU Cache-Based Side-Channel Attacks: Systematic Analysis, Security Models, and Countermeasures. *Security and Communication Networks*, 2021(1):5559552, 2021.
- [VGGQ22] Vanamala Venkataswamy, Jake Grigsby, Andrew Grimshaw, and Yanjun Qi. Rare: renewable energy aware resource management in datacenters. In *Workshop on Job Scheduling Strategies for Parallel Processing*, pages 108–130. Springer, 2022.

- [WE25] Connor Waldoch and Tim Ennis. Byte Blackouts: How Large Data Center Loads Are Surfacing New Issues. <https://blog.gridstatus.io/byte-blackouts-large-data-center-loads-new-issues-pjm/>, April 2025. Accessed: 2026-07-04.
- [WL23] Fanrong Wei and Xiangning Lin. Cyber-physical attack launched from EVSE botnet. *IEEE Transactions on Power Systems*, 39(2):3603–3614, 2023.
- [ZL25] Junfan Zhu and Jiqiang Lu. Leading degree: A metric for model performance evaluation and hyperparameter tuning in deep learning-based side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2025(2):333–361, 2025.
- [ZWW11] Yanwei Zhang, Yefu Wang, and Xiaorui Wang. Greenware: Greening cloud-scale data centers to maximize the use of renewable energy. In *ACM/IFIP/USENIX international conference on distributed systems platforms and open distributed processing*, pages 143–164. Springer, 2011.

A Landscape of U.S. Data Centers and Power Grid Stress

This appendix provides supplementary evidence for the scale and distribution of data-center loads discussed in Section 2.1. As shown in Figure 16, data centers in the U.S. are widely deployed and form dense clusters in several major computing hubs. Table 5 further shows that data centers already consume a substantial share of state-level electricity in many regions, reaching 25.6% in Virginia and exceeding 10% in several other states.

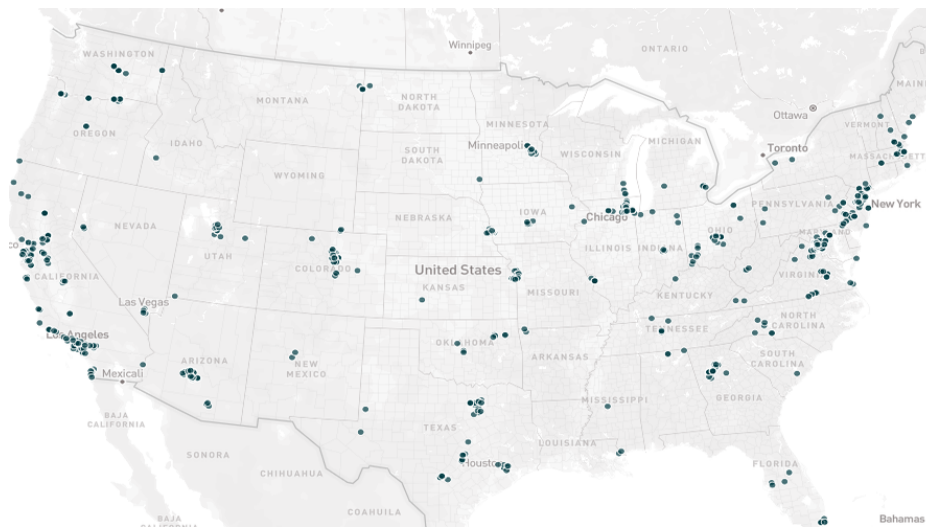


Figure 16: Spatial distribution of the data centers in the U.S. [DC25].

These observations indicate that data-center loads are geographically concentrated and large enough to impose significant demand on local power infrastructure. Meanwhile, GPU clusters exhibit highly dynamic power consumption with abrupt transitions occurring on millisecond timescales. When aggregated at scale, these rapid power fluctuations impose concentrated and non-cyclical stress on local grids. Data centers are no longer passive consumers; their reliance on high-precision power delivery renders them highly sensitive to even minor voltage disturbances. As demonstrated by the July 2024 “Byte Blackout” event in the PJM Dominion zone [WE25], the spatial concentration of data centers often

Table 5: Share of U.S. State Electricity Used by Data Centers [KZ24]

Block 1		Block 2		Block 3	
State	Share	State	Share	State	Share
Virginia	25.6%	California	3.7%	Connecticut	1.0%
Nebraska	11.7%	Pennsylvania	3.2%	Florida	0.6%
Iowa	11.4%	New York	2.8%	Idaho	0.6%
Oregon	11.4%	Colorado	2.7%	Michigan	0.5%
Wyoming	11.3%	South Carolina	2.5%	South Dakota	0.5%
Nevada	8.7%	Kentucky	2.2%	New Hampshire	0.2%
Utah	7.7%	Massachusetts	2.1%	Maryland	0.2%
Arizona	7.4%	North Carolina	1.9%	Rhode Island	0.2%
Washington	5.7%	Oklahoma	1.8%	Maine	0.2%
Illinois	5.5%	Alabama	1.7%	Wisconsin	0.2%
New Jersey	5.4%	Ohio	1.6%	Indiana	0.2%
Texas	4.6%	New Mexico	1.5%	Louisiana	0.1%
North Dakota	4.4%	Tennessee	1.3%	Hawaii	0.1%
Georgia	4.3%	Minnesota	1.2%	Kansas	0.0%
Montana	3.7%	Missouri	1.2%		

referred to as “Data Center Alleys” can transform localized transmission faults into regional reliability incidents. Transient events can trigger the near-instantaneous transfer of gigawatt-scale loads to internal UPS systems, producing abrupt load shedding that challenges conventional grid balancing mechanisms.

B Waveform Evidence of Power Adjustment Mechanism of GPU Clusters

This appendix provides supporting evidence for the power adjustment mechanism of GPU clusters discussed in Section 4.3.2, using measurements collected under normal operation. Figure 17 shows the board-level DC voltage and current measurements of a single GPU, while Figure 18 shows the corresponding system-level AC-side waveforms. As can be seen, the voltage remains relatively regulated at both levels. However, the board-level DC current shows more pronounced fluctuations and the system-level AC current exhibits pronounced non-sinusoidal behavior, with a heavily distorted waveform and multiple local maxima within each half-cycle. These measurements qualitatively illustrate that the workload-dependent power variations are primarily realized through changes in the current.

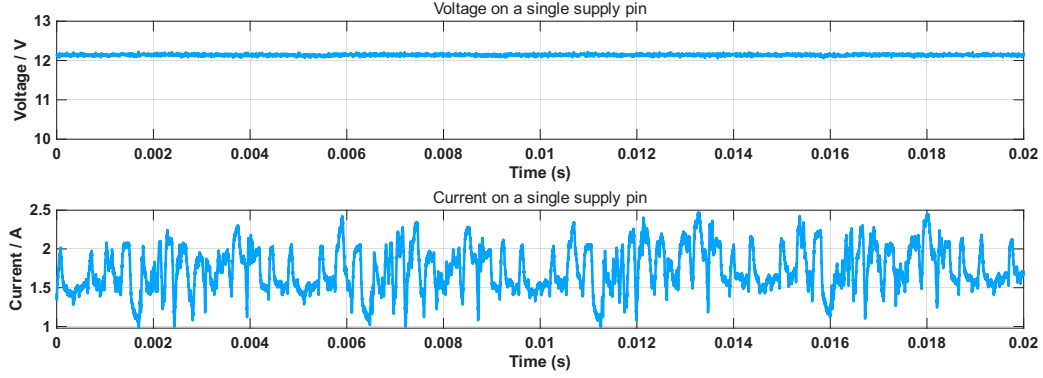


Figure 17: Board-level voltage and single-line current waveforms of an RTX 3090 GPU during GPT-2 training.

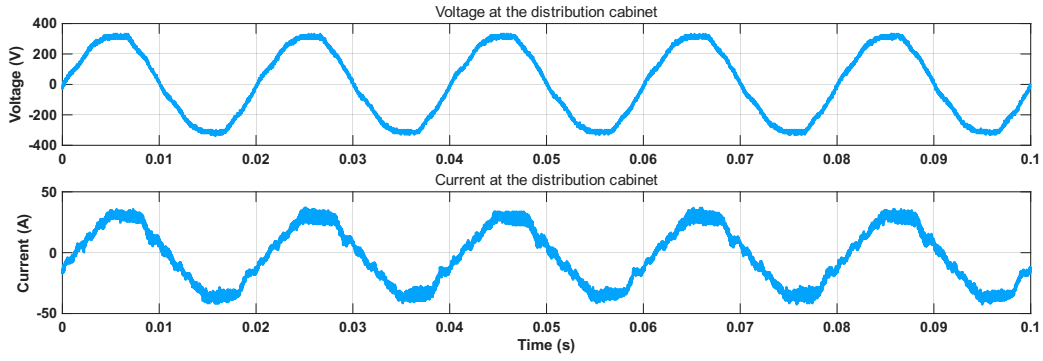


Figure 18: System-level voltage and current at the distribution cabinet of a GPU cluster.

C Effect of Timing Jitter on Attack Aggregation

Figure 19 provides a visual illustration of the timing-jitter effect discussed in Section 5.3.2. It shows the aggregated `Bit2Watt` power waveform of 1,000 GPUs under a 2 kHz modulation. As σ_τ increases, the imperfect factor progressively smooths the originally square-like waveform and reduces the aggregate modulation amplitude.

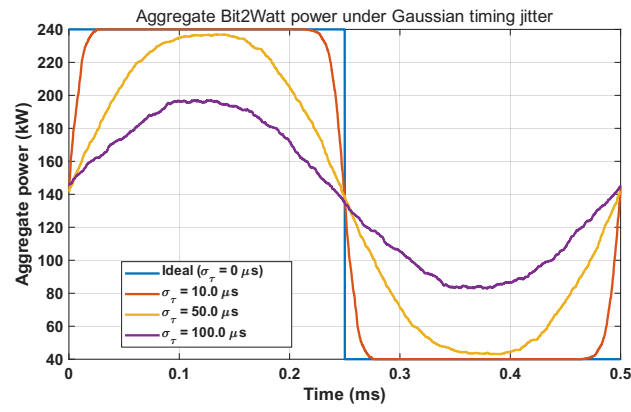
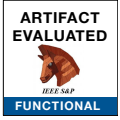


Figure 19: Aggregated Bit2Watt-induced power modulation of 1,000 GPUs under different levels of timing jitter.



Banshee: Target Switch Attacks on Gimbal-Stabilized Visual Tracking Systems via Acoustic Injection

Jiarui Li
University of Michigan
jiairuili@umich.edu

Joseph Brewington
University of Michigan
brewing@umich.edu

Qingzhao Zhang*
The University of Arizona
qzzhang@arizona.edu

Z. Morley Mao*
University of Michigan
zmao@umich.edu

Abstract—Gimbal-stabilized visual tracking is critical for modern autonomous systems such as Unmanned Aerial Vehicles (UAVs). While prior work shows acoustic signals can disturb gimbal internals, the impact of such attacks on real-world applications like UAV tracking and following remains under-explored. Existing demonstrations largely overlook practical challenges for real-world attacks, such as object-motion uncertainty and runtime latency. To bridge this gap, we present **Banshee**¹, the first physically realizable attack that induces target switching in UAV visual tracking systems by exploiting acoustic vulnerabilities in gimbal-camera systems. **Banshee** generates carefully crafted acoustic waveforms that induce optimized adversarial gimbal oscillations, causing directionally biased camera-view drifts that break inter-frame target associations. Consequently, the onboard tracker is driven to switch from the original target to an attacker-selected object with high probability, with occasional target loss. **Banshee** achieves a 93.6% success rate in simulation across two commercial gimbal systems and five trackers. Real-world benchtop and in-flight black-box attacks against a commercial drone across varied scenarios show an overall 95.5% attack success rate. Our results reveal a practical cross-domain vulnerability between acoustics and vision, highlighting the need for robust designs of gimbal systems and applications. Our code is available at: <https://github.com/Ultra/Banshee>.

1. Introduction

Gimbal-stabilized visual tracking is a core capability of modern camera systems, enabling persistent, high-precision object following in dynamic scenes. As a prominent example, commercial Unmanned Aerial Vehicles (UAVs) widely deploy target following, which typically pairs a multi-axis gimbal with an onboard camera, combined with object tracking algorithms running in software, to enable active

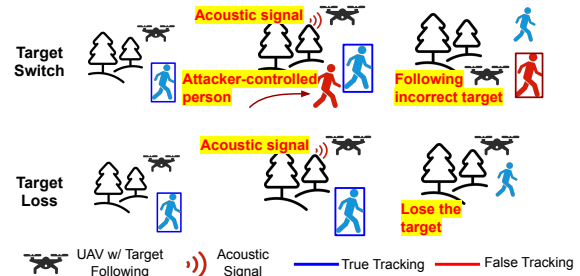


Figure 1: Illustration of Banshee in UAV target following. Crafted acoustic signals induce the UAV’s visual tracker to switch to an incorrect target or lose track.

tracking and following on a selected mobile target [1], [2], [3], [4], [5], [6]. Gimbal-stabilized visual tracking enables applications such as autonomous filming, surveillance, and infrastructure inspection, but also creates a single point of failure: compromising this pipeline can lead to severe consequences, including flight hazards, loss of vehicle control, and tracking of false targets [7], [8], [9], [10], [11].

Gimbal systems commonly rely on real-time inertial measurement unit (IMU) feedback to mechanically stabilize onboard cameras during rapid motion [12], [13]. Prior research has shown that carefully crafted acoustic patterns, delivered via speakers, ultrasonic transducers, or even laser systems, can manipulate IMU readings and, in turn, disrupt gimbal stabilization [14], [15], [16], [17], [18]. While these studies establish the feasibility of influencing gimbal motion at the sensor level, they remain disconnected from real-world applications such as UAV target tracking and following, leaving their practical impact uncertain.

To address this gap, we propose **Banshee**, the first physical target switch attack against UAV visual tracking systems via acoustic injection, linking a hardware acoustic vulnerability with application-level tracking weaknesses to enable end-to-end system compromise. By injecting acoustic signals to the UAV, the attack induces gimbal oscillations with directional bias that accumulates drift along a vulnerable axis. This abnormal motion disrupts motion smoothness and corrupts the tracker’s association with the true target, increasing the likelihood of an incorrect tracking output. Figure 1 illustrates two scenarios. (1) In the *target switch*

*. These authors are corresponding authors.

1. Our attack is named after the banshee, a mythical spirit whose scream causes or signals harm, reflecting the attack’s acoustic nature.

© 2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses. This is the accepted version of the paper published in 2026 IEEE Symposium on Security and Privacy. The final published version is available at IEEE Xplore. DOI: 10.1109/SP63933.2026.00129.

attack, a UAV performing target following switches from its original target to an attacker-selected object, enabling the attacker to potentially steal the UAV from its owner or transfer tracking onto an unintended target. (2) In the *target loss* attack, the UAV loses its tracking target, allowing a suspect under surveillance to escape.

The attack has two stages. In *offline gimbal profiling*, the attacker uses an identical gimbal to learn a black-box mapping from acoustic signals to induced gimbal motion. In the *online attack*, the attacker runs two loops simultaneously. A *surrogate tracking* loop that runs a surrogate of the UAV onboard tracking mimicking the actual UAV tracking behavior using black-box knowledge. A *planning-execution* loop then optimizes a sequence of acoustic signals under physical and algorithmic constraints, leveraging both the gimbal acoustic response model and the tracking surrogate. These signals are then injected through a speaker or piezo-electric transducers to induce the desired gimbal motion, probabilistically biasing the tracker away from the true target toward incorrect associations.

Designing this application-aware acoustic attack raises several key challenges. First, the attack must achieve an empirically sufficient alignment between physical acoustic signals and their induced camera motion, in order to produce desired adversarial motion that disrupts tracking. Second, the attack must operate at runtime without prior knowledge of the UAV behavior, which motivates an adaptive online strategy that updates the surrogate tracker and signal injection plan as the scene evolves. Third, the attack must remain effective under real-world uncertainties, including unknown future object motion, which we address with optimization algorithms that tolerate uncertainties. *Banshee* overcomes these challenges and achieves high-probability empirical success in inducing target switch or target loss.

Extensive experiments prove the practicality of the proposed attack. First, offline profiling on built-in gimbals of two commercial UAV models shows that consistent runtime directional bias in gimbal motion is feasible. Second, we run large-scale simulation in Gazebo simulator, which deploys PX4-Autopilot flight stack, uses the profiled gimbal parameters, and tests the attack on five representative trackers and diverse scenarios. The results show that *Banshee* overall corrupts the tracking in 93.6% of trials (including 75.0% target switch and 18.6% target loss), proving attack effectiveness and robustness. Finally, real-world experiments further validate successful black-box target switch on commercial drones, including a realistic exploit on the built-in object tracking during flight.² We summarize our contributions:

- We design *Banshee*, the first end-to-end attack that uses acoustic injection to compromise UAV visual tracking, bridging gimbal-level vulnerabilities to application-level impacts, and adapts to diverse real-world conditions.

2. We have completed responsible disclosure to the vendor and, at their request, anonymized the commercial product models. This anonymization does not affect reproducibility; the vulnerability is not specific to the vendor’s product and may apply broadly to gimbal mechanisms, and we provide proof-of-concept simulation environments at GitHub.

Acoustic Attack	Consistent Biasing	Affected Module	Signal Source	Evaluation Platform
Rocking [16]	○	Firmware	Speaker	Drone controller
WALNUT [14]	●	Sensor	Speaker	Toy car
Injected [15]	●	Firmware	Speaker	Smart phone
Poltergeist [19]	○	Detection	Speaker	Smart phone
KITE [17]	○	Localization	Speaker&Piezo	Drone controller
Laser [18]	○	Detection	Laser	Drone gimbal
Ours	●	Tracking	Speaker&Piezo	Drone gimbal
Tracking Attack	Online Optimize	Real-world Robustness	Attack Vector	Impact/Algo.
Fooling [20]	○	○	Patch	Loss/MOT
AttackZone [21]	●	○	Projector	Loss/SOT
ControlLoc [22]	○	●*	Patch	Loss/MOT
AdvTraj [23]	●	○	Person	Switch/MOT
FlyTrap [24]	○	●*	Patch	Shrink/SOT
Ours	●	●	Acoustic	Switch, Loss/SOT, MOT

* Robust patch generation by applying image transformations.

TABLE 1: Comparison with prior acoustic/tracking attacks.

- We propose the first systematic method to empirically model gimbal motion under acoustic injection. Using offline profiling and online phase modulation, the attacker can induce consistent, directionally biased gimbal angular offsets in real time.
- We extensively evaluate *Banshee* in high-fidelity simulation (Gazebo + PX4-Autopilot), with realistic simulation of gimbal vulnerability. We also demonstrate real-world online attacks on a commercial HighEndDrone.

2. Related Works

UAV visual tracking systems. Visual tracking enables core UAV functions such as target following, obstacle avoidance, and aerial monitoring [3], [4], [25], [26], [27]. Tracking algorithms generally fall into single-object tracking (SOT) and multi-object tracking (MOT). SOT methods, widely used for target following, include correlation-filter-based [28], [29] and siamese-network-based approaches [30], [31], both relying on appearance similarity. More advanced methods (e.g., transformer-based [32] and online learning-based [33], [34]) are often too resource-intensive for onboard deployment. MOT methods, used in aerial surveillance and obstacle avoidance [4], [27], [35], maintain associations across multiple dynamic objects and largely rely on motion consistency. Across these systems, a shared assumption is smooth inter-frame object displacement under benign conditions—an assumption our attack exploits.

Acoustic vulnerabilities of gimbal systems. Gimbal-stabilized cameras are standard in UAVs for stable video capture [4], [5], [6], relying on IMU-based feedback control [12], [13]. Prior work (Table 1) has shown that MEMS IMUs are vulnerable to acoustic injection [14], [15], [16], [17], [18], [36]. By exploiting resonant frequencies, attackers can inject false IMU signals, causing the gimbal to compensate for nonexistent motion and shift camera orientation. While these attacks demonstrate destabilization or sensor spoofing, they have not been systematically leveraged to induce target switch on visual tracking systems.

Prior physical attacks on visual tracking. Physical attacks on visual tracking have been explored but face unique challenges in UAV settings (Table 1). Adversarial patch-based attacks [20], [22], [37] can degrade tracking, yet lack online

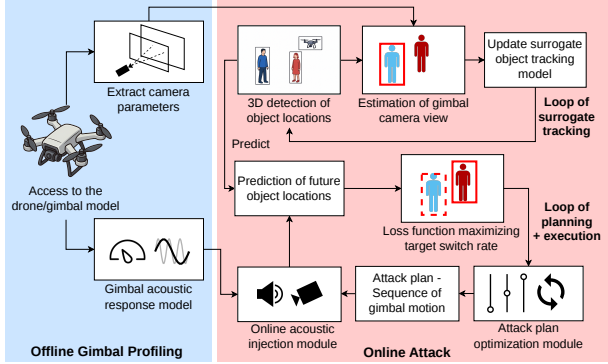


Figure 2: Overview of Banshee attack.

adaptation [38], [39] Real-time attacks using projectors or adversarial trajectories [21], [23] struggle with real-world uncertainty and latency. Acoustic attacks on object detection [19], [40] do not extend to the more complex tracking pipeline. FlyTrap [24] disrupts tracking via physical patches but under a different threat model. To date, no prior work demonstrates a physically realizable acoustic target switch attack on UAV visual tracking.

3. Threat Model

System and environment assumptions. In the attack scenario, a UAV performs target following using a gimbal-stabilized camera, whose video stream is processed by an onboard visual tracker. The flight controller then commands the UAV to follow the tracked target. Besides the legitimate target, the scene contains an attacker-controlled object that can be placed near the target during the attack. For example, if the UAV is following a pedestrian, an attacker-guided pedestrian can casually walk nearby (e.g., within a few meters), creating a normal and inconspicuous scenario. We consider a setting with no other nearby objects, as commonly seen in open-space tracking, and leave attacks in crowded scenes to future work. We denote the originally tracked object as **the true target** and the object after switching as **the false target**.

Attack goal. The adversary aims to launch an online **target switch** attack that achieves persistent target switching from the true target to the false target. The attack succeeds if the tracker consistently associates the false target, within the same semantic category (e.g., pedestrian or vehicle), as the true target. In different application contexts, the attacker can achieve two representative outcomes: (1) *Takeover*: the UAV, following its legitimate target (e.g., its owner), switches to an attacker-controlled object, resulting in effective capture; (2) *Escape*: the UAV, surveilling an attacker-controlled target, switches to another object, allowing the attacker to evade tracking. Both are instances of target switch, and we therefore do not distinguish them in evaluation.

A secondary goal is the **target loss**, where the tracker loses the target entirely, resulting in denial of service. While

target loss often arises when target switch fails, it still has significant adversarial impact.

Due to real-world uncertainties and limited attacker knowledge (discussed below), the attack is inherently probabilistic rather than deterministic; nevertheless, it should achieve a high probability of tracking corruption.

Attacker knowledge and capability. First, we assume the attacker knows the target gimbal-camera model and can obtain an identical device, which is feasible for commercially available UAVs. Using this device, the attacker performs offline gimbal profiling (Section 4.1) to characterize its acoustic vulnerability and learn an empirical mapping from acoustic signals to induced gimbal motion, which supports the online attack (Section 4.2). This mapping guides not only signal optimization but also attacker object placement, e.g., aligning with the most vulnerable gimbal axis and maintaining a moderate distance from the true target. The attacker also obtains key system parameters, including gimbal constraints (e.g., rotation limits and degrees of freedom) and the camera intrinsic matrix.

Second, we assume the attacker has only black-box knowledge of the tracking algorithm deployed on the UAV system. The attacker does not know any implementation details including the algorithm and parameters, but the attacker can exploit surrogate models to mount transfer attacks.

Third, the attacker needs a mechanism to deliver acoustic impact during flight. The most reliable option is attaching a **malicious payload** (such as a piezoelectric transducer [17]) to ensure stable acoustic signal injection. Installing a malicious payload requires physical access, for example, an attacker posing as a maintenance technician could covertly attach the device during routine UAV servicing.

Finally, the attacker can observe the UAV system, the true target, and the false target throughout the attack, allowing a sensor system (such as LiDAR, radar, or camera) to perform **real-time 3D object detection** on the UAV system and key objects to be tracked, obtaining location information necessary for runtime attack optimization. The attacker also needs to know the **real-time gimbal orientation** to inject biased gimbal motion (Section 4.1) and achieve an accurate online surrogate of UAV tracking (Section 4.2.3). To this end, the attacker can install an IMU on the gimbal (as part of the malicious payload). Note that we do not assume any access to the UAV system’s camera stream or internal software data. Our attack predicts the behavior of the tracking algorithm relying only on the external sensing and the profiled gimbal system parameters.

Relaxed requirements. Beyond the **primary** threat model, we consider a **relaxed** setting that reduces attacker requirements and validate the effectiveness of this threat model through physical experiments (Section 5.3). To avoid physical access, the attacker may perform a **remote attack** via long-range ultrasonic emitters [19], [41] or laser-induced vibrations [18] (see Section 6). The attacker may also estimate gimbal orientation using vision-based methods [42], [43] instead of installing a malicious IMU.

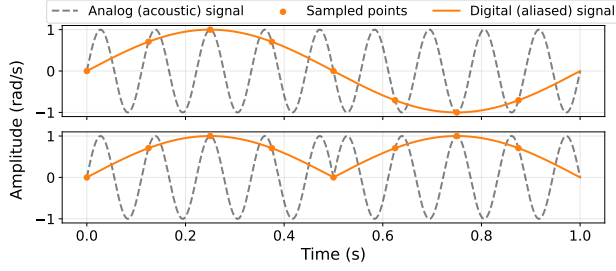


Figure 3: Illustration of phase-based directional bias in gimbal motion. Injected and aliased signals with (top) and without (bottom) phase shifting.

4. Banshee Design

Figure 2 shows the two stages of Banshee: offline gimbal profiling and online attack.

Offline gimbal profiling requires access to a gimbal identical to the target. Through controlled acoustic injection experiments, the attacker infers camera parameters, identifies the most vulnerable gimbal axis, and learns a black-box mapping from acoustic signals to induced motion (Gimbal Acoustic Response Model). This stage guides the selection of attack signals that induce abnormal gimbal motion. Details are in Section 4.1.

Online attack injects crafted acoustic signals into the target gimbal to induce abnormal motion along a vulnerable axis and cause tracking errors such as target switch. It consists of two iterative loops: *surrogate tracking* and *planning-execution*. The surrogate tracking loop maintains a surrogate tracker that mimics the UAV tracker, updated using the attacker’s external sensor data and the estimated gimbal view. The planning-execution loop optimizes a sequence of acoustic signals under physical and algorithmic constraints, using the offline profiling results, the surrogate tracker, and the current 3D scene as inputs. These signals are then transmitted through the attack vector (for example, speakers and piezoelectric transducers) to gradually redirect the tracking from the legitimate target to the attacker-selected object. Details are in Section 4.2.

4.1. Adversarial Acoustic Gimbal Biasing

We design a physical attack that injects selected acoustic waveforms *at runtime* to induce biased, abnormal gimbal motion. Specifically, the acoustic signal excites resonance and corrupts gyroscope readings, causing the stabilization controller to apply incorrect compensation. This results in *directionally biased oscillations* that accumulate into abnormal gimbal motion.

Formally, by offline measurements on the the gimbal device, we empirically approximate a mapping from acoustic waveform with frequency f_{in} and amplitude A_{in} to the gimbal motion response as a three-axis angular velocity $\omega[t]$.

Let the acoustic drive be $x(t) = A_{\text{in}} \sin(2\pi f_{\text{in}} t)$. We learn a **Gimbal Acoustic Response Model** $\hat{\mathcal{M}}$ satisfying

$$\omega[t] \approx \hat{\mathcal{M}}(A_{\text{in}}, f_{\text{in}}, t) + \varepsilon[t], \quad (1)$$

where $\varepsilon[t]$ is the residual error. This section presents the theoretical background and the offline profiling procedure used to obtain $\hat{\mathcal{M}}$, detailed in Sections 4.1.2-4.1.3. In addition, runtime directional biasing must handle uncertainty in the gyroscope sampling phase, which affects the induced motion direction; we therefore apply a phase-modulation routine that estimates and aligns the sampling phase at runtime, as detailed in Section 4.1.4.

4.1.1. Gimbal Fundamentals: Analog-Digital-Control.

MEMS gyroscopes measure angular velocity via the Coriolis force on a vibrating proof mass. Suspended by springs, the mass resonates at its natural frequency f_n . In previous work, it has been established that an external acoustic signal with frequency $f \approx f_n$ can drive large oscillations even without actual motion [14], [15], [16]. We use the definitions standard among works on acoustic injection attacks and model the response of the proof mass as a driven harmonic oscillator:

$$\omega(t) = A \sin(2\pi f t + \phi), \quad (2)$$

where A is the vibration amplitude and ϕ is the phase of the signal.

These oscillations are then sampled by an analog-to-digital converter (ADC). Since the sampling rate f_s is much lower than the resonant frequency, the digitized signal appears as a lower-frequency alias rather than at f (Figure 3). For an ideal, constant-rate ADC, the sampled signal is

$$\omega[t] = A \sin(2\pi f_d \frac{t}{f_s} + \phi), \quad \{t \in \mathbb{N}\}. \quad (3)$$

The digitized frequency of the aliased signal is subject to the Nyquist theorem

$$f_d = f - n \cdot f_s \quad \{n \in \mathbb{N}, \quad f_d \leq \frac{1}{2} f_s\}. \quad (4)$$

The aliased frequency f_d is always less than half of f_s and approaches zero as the frequency of the injected signal nears an integer multiple of this rate.

The digitized acoustic signal propagates into the system as perceived motion, leading to compensatory control algorithm behaviors [44], [45]. Because commercial IMUs operate at modest sampling frequencies typically in the tens to hundreds of hertz and rarely exceeding 1 kHz [46], [47], [48], [49], the resulting aliased oscillations appear at low, easily observable frequencies, enabling our proposed profiling and modeling methodology described below.

4.1.2. The Model of Adversarial Gimbal Biasing.

In theory, an adversary can influence and bias gimbal motion through modulated acoustic signals. To approximate this goal, we design a simplified gimbal motion abstraction leveraging structural and algorithmic insights as follows.

Scaling vibration amplitude. Amplitude scaling leverages the linear relationship between the injected signal amplitude and the resulting vibration amplitude. Following the amplitude equation for a driven harmonic oscillator we have

$$A = \frac{1}{m Z_m 2\pi f} \cdot A_{in}, \quad (5)$$

where the specific IMU model determines mass m and mechanical impedance Z_m , while the attacker controls signal frequency f and the acoustic amplitude A_{in} . We capture this linear relationship by fitting a linear regression for each of the identified resonant frequencies, simplifying the denominator to a single constant a

$$\omega[t] = (a \cdot A_{in}) \sin(2\pi f_d \frac{t}{f_s} + \phi). \quad (6)$$

Correcting direction of gimbal motion. The direction of gimbal motion is subject to real-world uncertainty since the phase of $\omega[t]$ depends on the relative timing between the injected signal and the sampling of the gimbal gyroscope. We extend the phase modulation technique in [14] to address limitations when the timing of samples is subjected to drift. We modulate phase in response to the direction of detected gimbal motion to improve robustness in the presence of sampling rate drifts. Specifically, when the direction of the observed gimbal motion begins to differ from the intended motion, we shift the phase of the injected signal by π so as to invert ω , also shown in Figure 3

$$-\omega[t] = A \sin(2\pi f_d \frac{t}{f_s} + \phi + \pi) \quad (7)$$

Adversarial biasing is independent of the gimbal orientation at the time of signal injection. UAV gimbal systems exhibit significant variability in servo actuation as a response to injected acoustic signals depending on gimbal orientation. The cause of this phenomenon lies in the gimbal stabilization mechanism. A 3-axis gimbal stabilizes a mounted camera using servos fixed in the drone body reference frame, while the camera body houses a gyroscope that is used in a feedback control loop to maintain stability in the world reference frame. To achieve this, gyroscope readings must be rotated by the orientation of the camera body relative to the drone body in order to actuate the correct servos. The closed-form matrix derived in [50] describes the rotation from the drone reference frame to the camera body reference frame that is used to transform gyroscope readings into correct servo actuation. Given angular values of pitch, roll, and yaw θ , the rotation matrix is

$$R = R_p R_r R_y = \begin{bmatrix} C_y C_p - S_y S_r S_p & -C_r S_y & C_y S_p + C_p S_y S_r \\ C_p S_y + C_y S_r S_p & C_y C_r & S_y S_p - C_p C_p S_r \\ -C_r S_p & S_r & C_r C_p \end{bmatrix} \quad (8)$$

where $C_i = \cos \theta_i$, $S_i = \sin \theta_i$, and current gimbal orientation θ . In short, when motion is perceived in the gimbal gyroscope the stabilization system will always actuate the correct servos in order to produce an opposite motion along the same axis in the camera-body frame, resulting in a stable motion response independent of the gimbal orientation. Section 5.1 also shows supporting results.

Overall, our Gimbal Acoustic Response Model $\hat{\mathcal{M}}$ has the following form

$$\omega[t] = a \cdot A_{in} \begin{cases} \sin(\theta_t), & \text{if } \text{sgn}(\sin \theta_t) = s, \\ \sin(\theta_t + \pi), & \text{if } \text{sgn}(\sin \theta_t) \neq s, \end{cases} \quad (9)$$

where $\theta_t = 2\pi f_d \frac{t}{f_s} + \phi$ and intended direction $s \in \{-1, 1\}$. In the following section, we present the profiling steps to obtain f_d and a . The additional residual error term $\varepsilon[t]$ is modeled as Gaussian noise, parameterized using the real-world experimental traces.

4.1.3. Gimbal Profiling for Obtaining the Model. The profiling procedure consists of two major steps that sweep over a single selected parameter of interest – **frequency** and **amplitude**, which produces f_d and a , respectively. In each sweep we first collect raw profiling data, then proceed to extract key information using spectral analysis and construct the necessary model.

Black-box gyroscope modeling via observation. Without loss of generality, we consider a three-axis gimbal camera system with an embedded three-axis gyroscope. By applying acoustic signals, the attacker induces oscillating motion readings in each axis, $\omega[t] = [\omega_p[t], \omega_r[t], \omega_y[t]]^T$, resulting in observable gimbal motion. The attacker installs an external malicious gyroscope on the camera body and collects the external gimbal motions $\omega_{obs}[t]$.

Frequency sweep. The first profiling step returns the set of resonant frequencies Q of the gyroscope that are most effective in driving proof-mass oscillations. (1) Following [15], sweep a sine wave across a wide range of acoustic signal frequencies (e.g., 1 Hz to 40 kHz) and record the motion response of the gimbal. (2) The acoustic amplitude A_{in} should be set to a fixed maximum value. For each injected test frequency f , extract aliased frequency f_d and amplitude A using spectral analysis. (3) Examining amplitude as a function of injected frequency and identifying local maxima of $A(f)$; these peaks reveal ideal injected signal frequencies for attacks and we collect them into a set Q .

Amplitude sweep. The second profiling step returns the linear relationships $\mathcal{A}$ between the injected acoustic amplitude and the amplitude of the induced gimbal motion. (1) For a selected frequency in Q , sweep the injected acoustic signal amplitude A_{in} from low to high (e.g., 1% to 100% signal power) and record the motion response of the gimbal. (2) For each injected signal, extract the observed motion amplitude A using spectral analysis. (3) Fit a linear regression of A as a function of A_{in} to obtain a constant a such that $A \approx a \cdot A_{in}$. (4) Repeat the procedure for remaining frequencies in Q .

4.1.4. Directional Biasing During Online Attacks. Since the phase of the sampled signal cannot be determined during offline profiling, we bias the direction of gimbal motion with real-time feedback using the theoretical model in Section 4.1.2. For enhanced robustness against real-world uncertainties, we can use a history-based method and only trigger a switch when the past N samples show motion in the

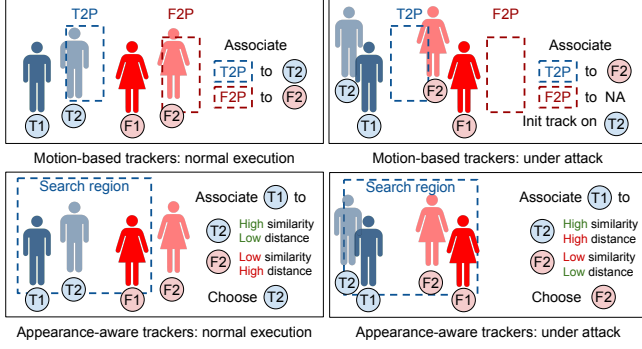


Figure 4: Illustration of how visual tracking is compromised by malicious gimbal motion. The subfigures show the gimbal camera view; T/F stand for the true target and the false target; numbers 1/2 are frame IDs; P is the predicted object location in motion-based trackers. The attack triggers a yaw-axis gimbal rotation and all objects are shifted to the left.

incorrect direction. We also incorporate a time restriction, limiting the number of phase switches to a period of once per elapsed time T . The attacker can tune N and T to allow the proper level of feedback sensitivity. As a general rule of thumb, $N < f_{\text{feedback}}$ and $T < \frac{1}{f_d}$, where f_{feedback} is the frequency of the motion direction detection.

In summary, the attacker can inject consistent and directionally biased gimbal angular offsets in real-time. The accuracy of each component and more detailed settings of our profiling approach are available in Section 5.1. The online attack (Section 4.2) takes advantage of this constructed gimbal biasing pipeline to maximize the attack success rate.

4.2. Online Attack

As summarized above and illustrated in Figure 2, the online attack runs in two iterative loops: *surrogate tracking* and *planning-execution*. In this section, we first explain the underlying visual tracking vulnerabilities (Section 4.2.1) and key attack parameter choices (Section 4.2.2), then detail the surrogate tracking loop and the planning-execution loop (Sections 4.2.3–4.2.4).

4.2.1. Vulnerability of Visual Object Tracking. Visual object tracking processes a 2D image stream to localize a designated object across frames. While tolerant of occasional detection errors, trackers become unstable when gimbal motion is perturbed. Below, we explain the reasons, also illustrated in Figure 4.

Motion-based trackers. Such trackers (e.g., SORT [51]) rely on spatio-temporal consistency, such as bounding-box overlap and motion-model prediction, to associate objects across frames. These algorithms assume that target displacement between consecutive frames is small and predictable: they predict the object’s location using a motion model and then match the predicted box to actual detections based on distance or overlap. For instance, SORT uses a Kalman filter

for motion prediction and Hungarian matching with IoU cost for object association. Acoustic injection disrupts this assumption by inducing abrupt shifts in the gimbal’s field of view, creating large apparent jumps in target position. As a result, the tracker may fail to maintain association with the legitimate target and instead link the track to a nearby object. In Figure 4, the predicted location of the true target (T2P) under attack falls closer to the false target’s detection (F2), causing the tracker to switch its association.

Appearance-aware trackers. Appearance-aware trackers (e.g., Siamese trackers) employ deep similarity networks to match object patches across frames. However, they still depend on spatial proximity constraints to narrow the candidate search space. For instance, KCF [52] considers object candidates only within a search region predicted from the last occurrence of the object; SiamRPN [53] further penalizes candidates that are far from the search region center and whose scale deviates significantly from the previous frame. Abrupt displacements caused by the adversarial gimbal motion could render the true target falling outside or further from the search region, reducing the reliability of matching.

Therefore, in both categories, acoustic-induced gimbal perturbations could be optimized to undermine the fundamental assumptions of inter-frame correlation that tracking algorithms depend upon.

4.2.2. Online Attack Parameters. As introduced in the threat model (Section 3), the attack seeks to mislead the tracker from following **the true target to the false target**. The attacker should determine the following parameters.

Geometric relationship of the two objects. Depending on the attack goal – *takeover* or *escape*, the adversary can manipulate either the false target or the true target. By controlling the trajectory of one object, the attacker determines the relative positioning between the two, tailoring it to the vulnerabilities of the target gimbal system. Specifically, in the offline profiling, we access which axis of the target gimbal is the most vulnerable to acoustic injection. For example, if profiling reveals that the gimbal’s yaw axis is most susceptible to acoustic injection, the attacker should arrange the true target and the false target horizontally within the camera’s field of view. In this case, perturbations along the yaw axis can shift the tracker’s focus from the true target to the false target. During the attack window, the two objects are typically placed in close proximity (e.g., 1–3 meters apart), as shorter separation reduces the required attack duration given the limited induced angular velocity.

Cycle times. The two in-loop processes should run as frequently as possible to enable fine-grained online optimization of the attack. However, several constraints govern the choice of cycle times. (1) Both loops must respect computational limits, which depend on the attacker’s hardware. In our case, the attack algorithm is lightweight, and its efficiency is evaluated in Section 5.2.5. (2) Stable gimbal motion requires that the planning-execution loop operate with a sufficiently long cycle time to allow acoustic signals to take effect. In practice, we align the planning-

execution cycle time to an integer multiple of the attack signal frequency (Section 4.1). (3) The two loops must remain synchronized as the planning-execution takes the 3D object detection and surrogate model status as input. In practice, because the planning-execution loop is usually slower than surrogate tracking, its cycle time is set to be an integer multiple of the surrogate tracker’s cycle time, ensuring coordination.

4.2.3. Loop of Surrogate Tracking. The attacker uses its own sensors to track objects in the scene, estimate the viewpoint of the target UAV gimbal camera, and run a black-box surrogate of the UAV’s tracking algorithm that mimics its runtime behavior, preparing for later optimization.

Real-time gimbal view estimation. A key challenge of the black-box attack is that the video stream from the target gimbal is unavailable, limiting direct use of a surrogate tracker. To overcome this, *Banshee* estimates the gimbal’s camera view in real time using external observations of the scene. Specifically, the attacker first applies 3D object detection (via an external sensor suite) to obtain the world coordinates of the true target, the false target, and the UAV. The gimbal’s orientation is then derived either by (1) a malicious IMU attached to the gimbal (requiring physical access), or (2) a high-resolution external camera with orientation detection algorithms (also see Section 3). Additionally, camera parameters are already known from the offline profiling stage. With the above information, *Banshee* projects the 3D world coordinates of the true target and the false target into the estimated target UAV’s 2D camera coordinate system. These projected 2D bounding boxes are then used as inputs to the surrogate tracking model, updated at each cycle of the loop.

Surrogate tracking models. Without access to the UAV system’s image stream and internal data, we design surrogate tracking models to enable the optimization.

Motion-based trackers. Modern motion-based trackers typically operate in two steps: object detection followed by ID association. In our surrogate model, object detection is replaced by the real-time gimbal view estimation. As such, at each frame, instead of running an image-based detector (which the attacker cannot access), we estimate the 2D bounding boxes of the true target and the false target and treat them as detection results. These estimated boxes are then the input of remaining surrogate tracking steps, with details depending on the specific surrogate model implementation. If using SORT [51] as the surrogate model, for instance, the two boxes are then matched with Kalman Filter (KF) [54] predictions using the similarity metric of intersection over union (IoU), after which tracks are updated and KF states are refined. To enable efficient optimization, we formulate this pipeline in a differentiable manner and compatible with gradient descent.

Appearance-aware trackers. Appearance-aware trackers localize targets by searching within a region around the last known position, leveraging both appearance cues and spatial consistency. In our black-box setting, the surrogate model cannot replicate the deep learning component that

Algorithm 1 Planning of angular velocity to inject.

Input: S_i : list of 3D bounding boxes for the true target, the false target, and UAV system at cycle i , θ_i : gimbal orientation, Surr: surrogate tracking algorithm, ΔP_i : position offset to be injected, t^c : planning-execution cycle time.

Output: ω_{i+1} : the planned angular velocity for cycle $i + 1$.

```

1: function FINDOPTIMALANGULARVELOCITY
2:   for  $iter = 0$  to  $N$  do
3:      $T_{i:i+2} \leftarrow \text{GENERATETRAJECTORIES}(S_i, S_{i-1}, t^c)$ 
4:      $\Theta_{i:i+1} \leftarrow \text{GENERATEROTATIONS}(\theta_i, \Delta P_i, t^c)$ 
5:      $\text{UPDATE}(\text{Surr}, T_{i:i+1}, \Theta_{i:i+1})$ 
6:     for  $tr$  in  $T_{i+1:i+2}$  do
7:        $X_{i+1} \leftarrow \text{3DTo2DPROJECT}(tr, \omega_{i+1})$ 
8:        $\text{COMPUTELOSS}(X_{i+1}, \text{Surr})$ 
9:     end for
10:     $\text{GRADIENTDESCENT}(\omega_{i+1})$ 
11:     $\text{APPLYCONSTRAINT}(\omega_{i+1}, \Theta_i)$ 
12:   end for
13:   return  $\omega_{i+1}$ 
14: end function

```

processes raw images. Instead, it directly estimates the deep learning component’s outputs and uses them as inputs to the subsequent object association stage. Taking a SiamRPN-based surrogate model [53] as an example, it crops a search region centered on the previous tracking box, typically four times its size. In the original tracker, this region is processed by a neural network to produce appearance-correlated proposals of boxes to be tracked. In our surrogate model, we mock this step by generating box proposals directly around the estimated positions of the true target and the false target in the gimbal camera view. Specifically, we sample multiple box proposals (e.g., 10) around each of the two objects, with slight random perturbations in their sizes and locations, and assign them high confidence scores (e.g., 1.0), reflecting the assumption that the component can reliably recognize foreground objects. These proposals are then passed to the association module, which selects the final tracked box. The entire process can also be implemented in a differentiable manner. Note that while the mocked proposals and confidence scores differ from those produced by the actual tracking with image feeds, they capture the key association behavior of appearance-aware trackers that they favor proposals closer to the search region center, which is sufficient for the optimization to exploit the tracker vulnerability.

4.2.4. Loop of Planning and Execution. The attacker starts an iterative process with planning and execution run simultaneously in parallel when the surrogate tracking model is initialized and the targets are in position. The iterative process stops after the attack success or timeout.

Planning. In the i -th planning-execution cycle, the algorithm computes the next attack plan ΔP_{i+1} , i.e., the gimbal orientation offset to be executed in cycle $i + 1$ (Algorithm 1). The algorithm first loads information from the surrogate tracking loop, including 3D detections of the true target, the false target, the UAV, and the latest surrogate model.

To plan for the future, the algorithm predicts the states of objects, the gimbal, and the tracker. The attacker uses motion/trajectory prediction models to estimate the trajectories of the true target, the false target, and the UAV across cycles i and $i + 1$. Future gimbal motion is predicted using the current attack plan ΔP_i and offline profiling results. To increase robustness against prediction errors, we apply expectation-over-transformation (EoT): each predicted trajectory or motion is augmented with sampled noise, forming sets of trajectories for optimization. For each combination of sampled predicted states, the surrogate tracking model is executed on them and produces an estimated future tracker state, i.e., the location of the tracking box if the scene evolves according to the predicted trajectories. The optimization objective is then defined as the below:

$$\arg \min_{\omega < \omega_{max}} \mathbb{E}_{P^t, P^f, X, G} \left[\sum_{i=1}^I \text{dist}(Box_i^{track}, Box_i^{false}) \right], \quad (10)$$

where P^t , P^f , and X denote the trajectory sets of the true target, the false target, and the UAV (from EoT), G is the gimbal orientation distribution, I is the number of frames in a cycle, Box^{track} is the surrogate tracker output tracking box, and Box^{false} is the false target box. In short, the optimization searches for a gimbal angular velocity ω that maximizes the chance of misplacing the tracking box onto the false target by minimizing the distance between the two boxes. The solution is constrained by the maximum achievable velocity ω_{max} from profiling. Because the surrogate tracking model is differentiable (Section 4.2.3), the optimization can be efficiently solved with gradient descent.

The final step of planning is to translate the angular velocity into the acoustic signal to inject. The translation is straightforward by applying the gimbal acoustic response model (Section 4.1).

Execution. The execution step runs in parallel with the planning step. The optimized attack plans (i.e., the acoustic signals to inject) are transmitted to the signal generator immediately after completion. When the i -th planning-execution cycle starts, the attack plan ΔP_i is immediately executed as it is generated in the last cycle (the first cycle has no execution step). This parallel scheduling minimizes the effect of optimization latency on attack performance, with efficiency shown in Section 5.2.5.

5. Evaluation

We extensively evaluate *Banshee* to demonstrate its effectiveness and impact. First, we validate offline gimbal profiling, which underpins our runtime attack, by showing it consistently injects gimbal responses with directional bias (Section 5.1). Next, we evaluate attack effectiveness and robustness in high-fidelity simulation across diverse scenarios, including ablations (Section 5.2). Finally, we demonstrate practicality via physical experiments on a commercial drone, achieving results consistent with simulation (Section 5.3).

5.1. Gimbal Biasing Experiments

In this section, we evaluate the feasibility and consistency of offline gimbal profiling and biasing.

Tested drone and gimbal systems. We profile two state-of-the-art commercial UAV gimbal systems: a high-end industrial drone with a payload-mounted gimbal and a mid-end consumer drone with an integrated gimbal. For HighEndDrone, we perform full profiling and physical experiments, while for MidEndDrone, we conduct frequency sweeps and directly adjust amplitudes in simulation-based evaluations.

Experimental setup. We attach a piezoelectric transducer directly to the gimbal housing and drive it with a function generator (AD9033) and amplifier (TDA8932) to inject crafted acoustic signals into the gimbal through vibration. An IMU (BMI160) is also attached to the gimbal housing to obtain the gimbal orientation.

Frequency sweep. For both HighEndDrone and MidEndDrone, we sweep 5 kHz to 30 kHz. After applying Least-Squares Spectral Analysis and local maxima filtering, we obtain resonant frequency sets $\{7744, 23232\}$ and $\{5526, 9214\}$ for HighEndDrone and MidEndDrone respectively. Figure 5 shows a reconstructed signal using frequency, amplitude, and phase of oscillation in each axis obtained through spectral analysis.

Amplitude regression. Figure 6 shows one regression produced at an injected signal frequency of 23232 Hz. We use a digitally controlled potentiometer to control the signal amplitude in terms of percentage power. We profiled 3 values $a_p = 0.27, a_r = 0.27, a_y = 2.63$ for pitch, roll, and yaw, respectively. These three regressions achieve an average R^2 of 0.997, showcasing the accuracy of our model. Our measurements also show that, at this frequency, the motion response amplitude of the yaw axis is significantly more sensitive to acoustic signal strength than pitch and roll.

Gimbal orientation independence. To validate that acoustic gimbal biasing is independent of gimbal orientation, we test combinations of pitch at -90° to 45° , roll at -15° to 15° , and yaw at -90° to 90° , which covers the full range of possible orientations of HighEndDrone’s gimbal. In Figure 9 we plot the average error across all three axes at each orientation, compared to the motion observed at the neutral orientation. Our experiments show that across all orientations, the movement perceived by our external IMU is nearly identical with only slight variation. These minor errors can be most likely attributed to factors such as measurement error and any mechanical differences such as wear in the servos which rotate the different axes.

Biasing results. To evaluate the consistency and accuracy of our online biasing methodology, we conduct 100 trials at a frequency of 23232 Hz over 100% and 50% signal power in clockwise and counterclockwise yaw rotation. In Figure 7 we plot all 100 traces integrated over a 1 second period. We successfully demonstrate that using our online biasing methodology, we can achieve consistent gimbal rotation in the desired direction with only a small degree of error.

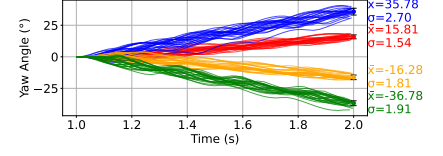
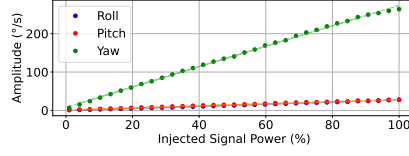
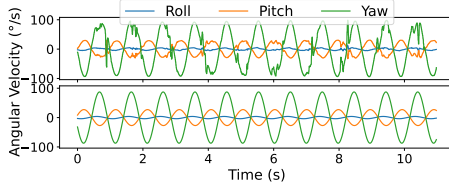


Figure 5: Accuracy of spectral analysis for 7744 Hz. Reconstructed signal (bottom) is regression at 23232 Hz. close to raw gyroscope readings (top).

Figure 7: Online gimbal biasing: integrated angle over time for different amplitudes and directions.

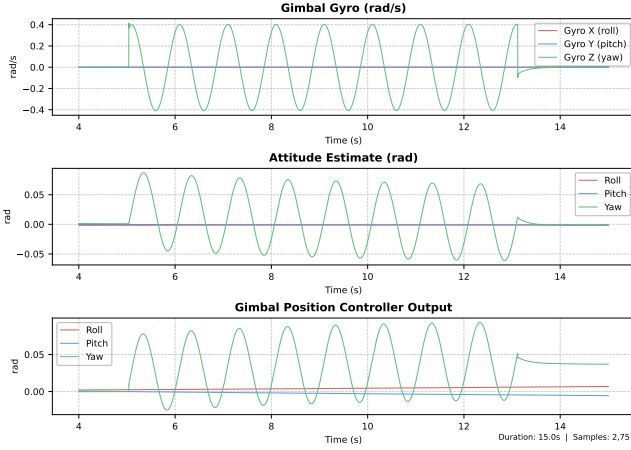


Figure 8: Time domain recording of intermediate gimbal states during acoustic injection.

Execution latency. A potential concern is latency between acoustic injection and gimbal response. In our experiments, latency was negligible, consistent with expectations: the proof mass responds instantaneously, and gimbals must react in real time to support image stabilization.

Time domain demonstration. To expose intermediate gimbal states and better understand acoustic injection, we simulate the gimbal stabilization pipeline in Gazebo [55]. Our implementation follows the STorM32-BGC design: IMU (gyroscope/accelerometer) measurements are fused via a complementary filter for attitude estimation, followed by a cascaded PID controller with an outer attitude loop and a high-gain inner rate loop. Acoustic perturbations are injected directly into gyroscope measurements (prior to Gazebo noise) as sinusoidal signals, with rectified variants for phase modulation along a selected axis. Figure 8 shows gyroscope readings, attitude estimates, and controller outputs under a 1 Hz yaw-axis injection. The resulting sinusoidal motion is consistent with our empirical model despite pipeline nonlinearity.

Overall, these results demonstrate both the feasibility of the profiling and the consistency of gimbal biasing through acoustic injection.

Algorithm	Motion Appear.	Robust Design	FPS
KCF [52]	○	●	-
SiamRPN [53]	○	●	-
DaSiamRPN [57]	○	●	Distractor aware
SORT [51]	●	○	-
UCMCTrack [60]	●	○	Motion compensate

TABLE 2: Details of evaluated tracking algorithms. Motion/appearance—tracker types; C/G—CPU or GPU.

	Normal	Noisy	HighEndDrone	MidEndDrone
SORT	11.1%/24%	0%/75%	59.3%/94.4%	55.6%/96.3%
UCMC.	0%/0%	6%/16%	68.5%/92.6%	61.1%/87.0%
Siam.	0%/0%	0%/0%	83.3%/88.9%	90.2%/93.1%
DaSiam.	0%/0%	0%/0%	92.6%/98.1%	97.2%/100%
KCF	0%/0%	17.6%/17.6%	70.3%/94.4%	72.2%/87.5%

TABLE 3: Switch/disable attack success rates in simulation.

5.2. Simulation Experiments

Our simulation experiments comprehensively evaluate the attack’s effectiveness, robustness, and ablation results.

5.2.1. Experiment Setup. We begin by describing how the simulation is configured to approximate real-world attacks.

Tracking algorithms. We select five representative visual tracking algorithms, as shown in Table 2. The algorithms span both major categories: motion-based trackers and appearance-aware trackers (details in Section 4.2.1). In each category of tracker, we include the lightweight method (e.g., KCF and SORT), deep-learning-based methods (e.g., SORT and SiamRPN), as well as the state-of-the-art variants with robustness enhancement (e.g., DaSiamRPN and UCMCTrack). For SORT and UCMCTrack which require an object detection component, we use the pretrained model YOLOv8x [59]. For learning-based methods SiamRPN and DaSiamRPN, we use pretrained models trained on a range of datasets: VID [61], YoutubeBB [62], COCO [63], and ImageNetDet [61]. Our evaluated algorithms are lightweight enough to run on a typical UAV onboard computer. Other algorithms such as transformer-based methods are not suitable for real-time tracking on a mobile system (i.e. STARK [32] runs at only a max 30-40 FPS even using a Tesla V100 GPU).

Evaluation metrics. We evaluate the following metrics:

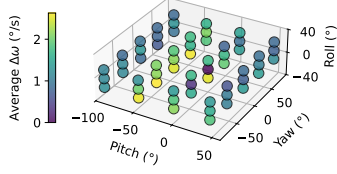


Figure 9: Orientation independence: difference from neutral orientation in amplitude, averaged across pitch/roll/yaw over various orientations.

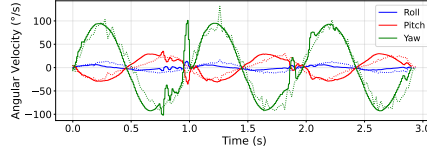


Figure 10: Acoustic gimbal motion injection trace in the physical world (solid line) and simulation (dotted line).

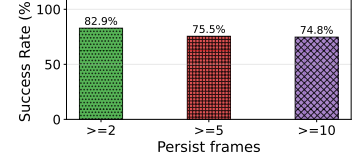


Figure 11: Distribution of number of consecutive frames of target switch, showing that target switch may become target loss in attacks.

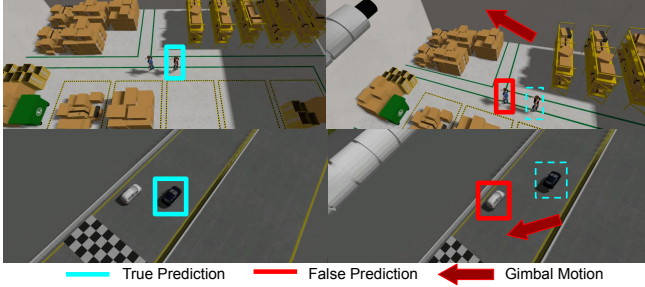


Figure 12: Demonstration of successful simulated target switch attacks.

- *Target switch* is successful if the tracking box contains the center of the false target for at least ten consecutive frames and persists to the end of the simulation.
- *Target loss* happens when the tracking box has no overlap with either the true target or the false target (often a side effect of failed target switch).
- *Disable*: The union of target switch and target loss attacks.

Simulation scenario setup. We simulate the attack with the high-fidelity physical simulator Gazebo [55] and the complete UAV software stack PX4-Autopilot [64]. The simulation experiments are carried out on a desktop with Intel i9-14900K CPU and RTX 4080 GPU running Ubuntu 24.04. Figure 12 demonstrates some of the attack scenes.

To demonstrate the generality of Banshee, the test cases are distributed as follows:

- *Worlds*: We include four distinct environments: factory, urban, field, and raceway. The factory world is an indoor setting, while the others are outdoor scenarios.
- *Object categories, motion, and appearances*: Both the true target and the false target can be either pedestrians or vehicles. Pedestrians move at speeds of 0.5-1.5 m/s in random directions and vehicles move at speeds of 5-15 mph straight. Both pedestrian models (blue, green, red outfits) and vehicle models (silver, blue, and red hatchbacks) have pairwise distinct appearances.

In total, we conduct 108 trials for each evaluation case. 54 are pedestrian cases, evenly distributed across three worlds (factory, urban, field) and six appearance pairs, with randomized initial motions. The remaining 54 trials involve vehicles, conducted in three worlds (raceway, urban, field),

also covering six appearance pairs.

Simulation of the system and attacks. For realistic simulation, we follow the threat model (Section 3) and configure the simulation as follows:

- *Adversarial gimbal biasing*: We implement adversarial acoustic gimbal biasing in simulation using the gimbal acoustic response model (Section 4.1). We inject Gaussian noise parameterized by the residual error from physical profiling via spectrum analysis (Section 5.1), modeling realistic motion perturbations (Figure 10). Specifically, we configure the gimbal maximum angular speed for the roll, pitch, and yaw gimbal axes as (0.04, 0.27, 2.35) and (0.76, 2.92, 0.21) rad/s, with realistic motion injection noise of (0.03, 0.04, 0.34) and (0.17, 0.51, 0.05) for HighEndDrone and MidEndDrone respectively. Other simulated gimbal characteristics, including rotation limits, degrees of freedom, and frame rate, are aligned with HighEndDrone or MidEndDrone.
- *UAV altitudes*: UAVs are above the ground by 10 meters and 25 meters for the pedestrian and vehicle scenarios, respectively, which aligns with real-world drone setup.
- *Distance of two objects*: In the pedestrian scenario, the true target and the false target are placed 1–2 meters apart, while in the vehicle scenario they occupy adjacent lanes. Such distances are common in real-world settings and therefore do not appear suspicious as malicious behavior.
- *Uncertain object motion*: To reflect the real-world uncertainty, the true target and the false target move with injected Gaussian noise. Uncertainties in UAV flight are simulated with Gazebo’s physical engine.
- *Motion blur*: Abrupt camera shifts naturally introduce motion blur. We model this effect using linear and rotational blur kernels in OpenCV [65]. However, since the gimbal’s maximum angular speed is constrained, the blur remains minor, and our attack does not depend on it for success.
- *Baselines*: In Normal, we simulate UAV object tracking without the presence of attack. In Noisy, we simulate UAV object tracking under strong and gusty wind supported by the realistic Gazebo WindEffects plugin.
- The attack uses a surrogate model aligned with the evaluated tracker, with a 4 Hz planning–execution cycle, one gradient descent step per cycle, and three trajectory samples for expectation-over-transformation.

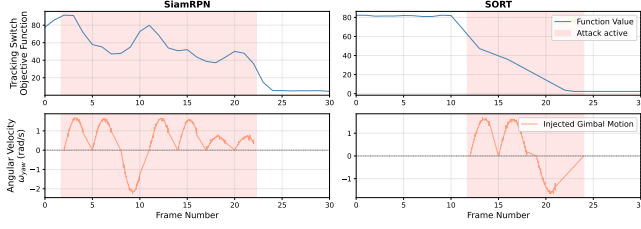


Figure 13: Time domain demonstration of injected gimbal motion induces target switch in simulation.

5.2.2. Attack Effectiveness. Table 3 presents the attack results in the simulation setup as described in Section 5.2.1. Our attack demonstrates overall success on different object tracking algorithms with **75.0%/93.6%** averaged success rate for target switch and disable (target switch & target loss) respectively. Target loss arises as a side effect of target switch. Although the attack is optimized for target switch, its probabilistic nature can still disrupt tracking and cause all objects to be lost. Figure 11 shows that 8.1% achieve target switch for at least two frames before transitioning to target loss eventually.

Time domain demonstration. Figure 13 shows a target switch attack example for SiamRPN and SORT in simulation. The target switch objective function (Equation 10) captures key attack success criteria. Minimizing the function value (blue curve) maximizes attack success. Orange curve shows simulated gimbal motion under acoustic injection attack, configured using the acoustic parameters obtained from the physical profiling experiment. It shows that injecting the optimized gimbal motion minimizes the objective function leading to target switch attack success.

Baselines. Switching is negligible under both settings, confirming that target switching does not arise in benign or naturally perturbed conditions. While noise can degrade tracking (e.g., causing target loss for SORT), it does not induce sustained target switching.

Across tracking algorithms. Interestingly, more advanced tracking algorithms, i.e., with embedding of object appearance or compensation to camera instability, do not exhibit stronger resilience to our attacks. Instead, appearance-aware trackers (SiamRPN, DaSiamRPN) achieve up to 29% higher target switch success rates than motion-based trackers (SORT, UCMCTrack). This indicates that the attack perturbation is strong enough for corrupted spatio-temporal consistency to outweigh appearance features and the discriminative appearance awareness assists the attack by identifying the false object instead of losing in the background. The increasing target switch success rate among KCF, SiamRPN, and DaSiamRPN with deeper appearance awareness provides further evidence.

The advanced motion-based tracker UCMCTrack includes mechanisms to compensate for camera instability, yet it still fails under our target switch attack, with a target switch success rate higher than the vanilla SORT. The attack success DaSiamRPN and UCMCTrack highlights that

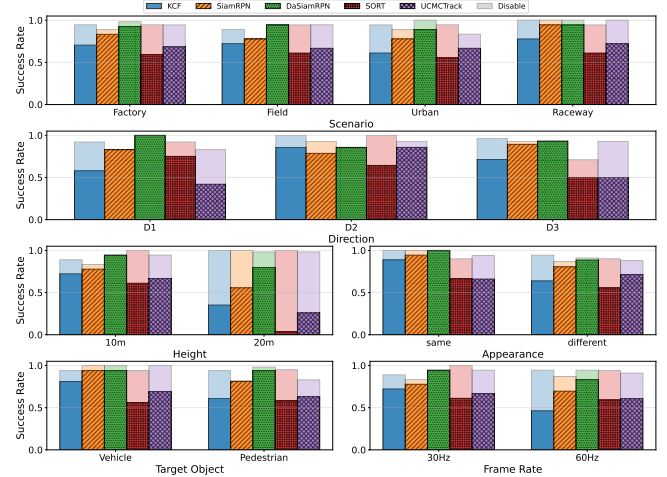


Figure 14: Simulation results analysis. Filled bars: target switch success rate. Translucent bars: disable success rate.

existing robustness designs cannot withstand the malicious perturbations but only assists the attack by reducing target loss. Across three runs, the standard deviations of target switch success (SiamRPN: $\pm 9.1\%$, DaSiamRPN: $\pm 3.5\%$, SORT: $\pm 4.5\%$, UCMCTrack: $\pm 8.2\%$, KCF: $\pm 10.3\%$) show the consistency of this trend. See Finding 1.

Finding 1: Robustness oriented tracking designs, including appearance aware and motion compensated trackers, do not withstand our target switch attack; instead, they often increase target switch success rates by preserving corrupted tracks rather than allowing target loss.

Across different gimbals.. The attack demonstrates stable success across two commercial gimbal camera systems with distinct acoustic vulnerability shown in Section 5.1. This suggests that the proposed online attack automatically adapts to specific gimbal parameters, regardless of the realistic random noise we added to the simulated gimbal motion.

5.2.3. Analysis of Impacting Factors. Beyond the default simulation setup, we vary several factors reflecting real-world conditions to evaluate the robustness of the attack. Results are shown in Figure 14.

Different environmental scenarios. Different environmental scenarios have only limited impact on target switch success rates, showing a minimum of 70.2% in the factory scenario and a maximum of 80.0% in the raceway scenario. This is likely because the raceway vehicles are easier to track than the factory pedestrians because of object sizes.

Different true target directions. This analysis focuses on the pedestrian scenario. We categorize the randomized pedestrian motion into 3 directions in the gimbal camera field of view: D1 (moving away), D2 (moving horizontally), and D3 (moving closer). According to our results, different motion directions have limited impact on target switch attack

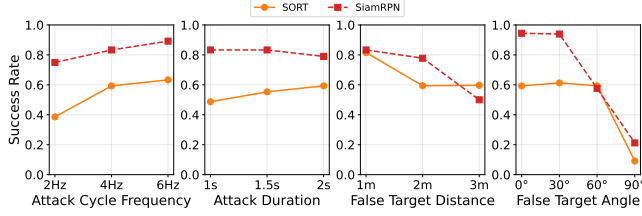


Figure 15: Target switch success rate w/ various conditions.

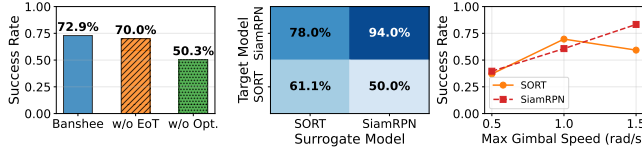


Figure 16: Left: attack components ablation. Middle: surrogate models transferability. Right: gimbal speed ablation.

success, confirming that our optimization algorithm can handle variation in the the true target’s motion.

Different UAV heights. The result shows attack success rates at varying UAV heights in the pedestrian scenario. The attack maintains high target switch success at low altitude (10 m), where richer visual cues and reliable detections benefit both category of trackers. However, target switch success drops markedly at higher altitudes (20 m) as smaller object size degrades tracking performance, leading to increased target loss. The vehicle scenario results further support this hypothesis with a 78.9% target switch success rate at 30 m.

Target scenarios. The attack consistently achieves comparable or higher success rates in vehicle scenarios across all tracking algorithms. This indicates that targets which are inherently easier to track, such as vehicles with regular motion patterns and stronger visual features, are also more vulnerable to target switch.

Target appearance. Appearance-aware trackers show markedly higher target switch success when the two objects appear identical, as expected from their reliance on appearance matching. These results indicate that an attacker can enhance success by intentionally making objects visually similar.

Camera frame rate. The 30 Hz setting, used by default in simulation, reflects common real-world configurations (Table 2). Results show that 60 Hz frame rate mitigates the attack, as the same gimbal motion causes smaller inter-frame misalignment. Appearance-based trackers are most affected, suggesting that increasing frame rate can enhance robustness against malicious gimbal motion.

We conclude to Finding 2 summarizing the above:

Finding 2: The attack remains highly effective across real-world variations, and while factors such as higher UAV heights and higher tracking frame rates can reduce success, these challenging cases are limited.

5.2.4. Ablation Study. We perform a set of experiments to understand the effect of different attack parameters and design components. Experiments are performed against SORT and SiamRPN with the same configuration as in Section 5.2.1 with HighEndDrone acoustic parameters.

The usefulness of design components. We perform two ablation studies on the optimization component: removing expectation over transformation (EoT) and removing the entire optimization. Without EoT, optimization samples a single trajectory from the object motion models (4.2.4); without optimization, the attack applies a constant angular speed to shift the camera toward the false target at the maximum controllable rate. As shown in Figure 16, both components substantially improve target switch success.

Distance between two objects. In the pedestrian scenario, we evaluate attack performance as the distance between the true target and the false target varies (either may be the attacker controlled object; see Section 3 and Figure 4). For SiamRPN, target switch success declines sharply with increasing distance, as its fixed search region limits association, making appearance-based trackers harder to compromise when objects are far apart. In contrast, motion-based trackers assign new IDs once geometric consistency breaks, enabling target switch even at larger separations.

Angle between two objects. We vary the relative angle between the true target and the false target. At 0° and 90°, the controlled object aligns with or is orthogonal to the gimbal axis most susceptible to acoustic injection (Section 4.2.2). Deviations from this axis markedly reduce attack effectiveness (Figure 15). The motion-based SORT is more sensitive to angle changes, as its association depends on whether the attacker lies along the injected motion direction, while the appearance-based SiamRPN is less affected due to its broader search region.

These results highlight that both distance and angle of the attacker-controlled object are critical. To maximize success, the attacker should align objects with the gimbal’s most vulnerable axis in the camera view—achievable through scene understanding and viewpoint prediction. The distance can then be tuned to balance success and stealth: greater separation slightly lowers target switch probability but benefits inconspicuity. In summary:

Finding 3: Attack success strongly depends on how the attacker places the controlled object, with favorable distance and alignment along the most vulnerable gimbal axis significantly boosting target switch effectiveness.

Planning-execution cycle time. A shorter cycle time yields more frequent optimization updates, making the attack more responsive to subtle scene changes. As shown in Figure 15, shorter cycle time increases success rates, particularly for SORT. Ideally, the attacker would operate at the fastest cycle allowed by available computational power.

Attack duration. The impact of attack duration differs markedly between motion based and appearance aware trackers. For SiamRPN, high target switch success is

achieved with short (1 s) perturbations, whereas the motion-based SORT tracker requires longer durations and additional online optimization to reach similar performance. Empirically, excessively long attacks reduce stealth by causing noticeable camera shifts or leading to target loss instead of target switch. Hence, the attacker should determine an appropriate duration through offline testing on the target gimbal system.

Maximum angular speed injected. Our attack remains stealthy, as the attacker-controlled object follows the victim at an unsuspecting distance (1–2 m or in an adjacent lane) and the acoustic signal is invisible. On average, successful trials induce no more than a 30.2° gimbal movement. Moreover, tracker confidence levels remain consistent before and after the attack, suggesting that the perturbation is difficult to detect. Stealthiness can also be tuned by parameters such as maximum gimbal speed, enabling a trade-off between attack effectiveness and detectability. Figure 16 shows attack success rates with respect to gimbal angular speed, and they are indeed strongly correlated.

Until now, we have identified multiple parameters that influence stealthiness, as summarized in Finding 4.

Finding 4: An attacker can tune attack parameters to trade off stealthiness against effectiveness: placing objects too close makes their motion suspicious, while long attack durations or high gimbal angular speeds cause noticeable camera viewpoint changes.

Surrogate model choices. In Figure 16, we evaluate attacks using different combinations of surrogate and target trackers, such as applying SORT to attack SiamRPN and vice versa. Transferability is limited between motion-based and appearance-aware trackers, consistent with their distinct design principles. This suggests that attackers can perform offline attack before the online attack to select the surrogate model, or infer the type of the UAV tracker. See Finding 5

Finding 5: Beyond gimbal profiling, the attacker can further improve success by inferring whether the UAV uses a motion-based or appearance-aware tracker and identifying effective attack durations or angular speeds.

Attacker-controlled object motion. Beyond noisy pedestrian motion, we evaluate more realistic motion models [66], [67], where the attacker-controlled object follows physically plausible, smooth trajectories. Under the HighEndDrone main settings, the attack achieves 78.2% (SiamRPN) and 67.2% (SORT) target switch success rates, demonstrating robustness to constrained and imprecise object motion.

Gimbal motion pattern variations. The injected motion pattern depends on resonant frequency and amplitude, which may drift due to environmental and hardware factors. To evaluate robustness, we execute an attack plan optimized for 4 Hz under perturbed conditions (3 Hz with profiled amplitudes and noise), resulting in a biased oscillation that

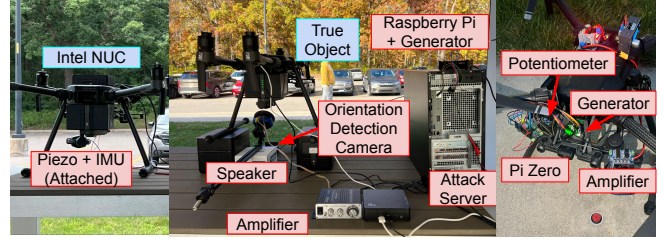


Figure 17: Benchtop direct-contact (left), benchtop contact-free (middle), in-flight direct-contact (right) attack testbeds.

differs from the planned pattern. The attack remains effective, achieving 77.8% (SiamRPN) and 55.6% (SORT) target switch success.

5.2.5. Attack Efficiency. Without dedicated optimization, our planning step runs at ~ 7.6 Hz on our machine, which is fast enough to tolerate the selected cycle time (4 Hz). The main bottleneck for a real-time attack is the gradient optimization process. The attacker can parallelize the gradient optimization process (e.g., gradient step and EoT) using dedicated GPU devices.

5.3. Physical Experiments

In this section, we demonstrate real-world attack performance through physical experiments on HighEndDrone.

5.3.1. Experiment Setup. As shown in Figure 17, we deploy three types of physical experiments.

- *Benchtop direct-contact:* The drone is placed on an elevated platform about 1.5 m above ground, representing typical consumer tracking scenarios at person height. The gimbal camera is driven by an Intel NUC 13 Pro (Core i5), a common companion computer for onboard tracking algorithms. For acoustic injection, we attach a piezoelectric transducer as described in Section 5.1.
- *Benchtop contact-free:* A Fostex FT17H speaker radiates the crafted acoustic signal through air. An EMEET C960 webcam observes the gimbal from below. The camera feed is processed with an optical-flow algorithm to estimate real-time gimbal orientation. Other hardware and configurations match the direct-contact benchtop setup.
- *In-flight direct-contact:* We build a compact malicious payload with multiple low-power integrated-circuits (which could be reduced to a single embedded unit with further engineering), containing a Raspberry Pi Zero 2W, function generator (AD9033), amplifier (TDA8932), digital potentiometer (DS3502), and IMU (BMI160). The Pi Zero controls the generator and potentiometer to produce the crafted waveform, delivered by a piezoelectric transducer mounted on the gimbal housing. During tests, the drone hovers at about 5 meters.

The online attack algorithm runs on a desktop with Intel i9-14900K CPU and RTX 4080 GPU. An EMEET C960 Webcam is used to obtain the input for the surrogate tracking

	FPS	Direct-contact	Contact-free	In-flight
SORT	25.0	60%/100%	60%/100%	-
UCMCTrack	24.9	80%/100%	100%/100%	-
SiamRPN	23.4	90%/100%	90%/90%	-
DaSiamRPN	7.5	90%/90%	100%/100%	-
KCF	30.0	70%/100%	70%/90%	-
HighEndDrone	-	-	-	60%/80%

TABLE 4: Target switch/disable attack success rate of real-world physical experiments.

algorithm. We use an appearance-aware surrogate tracker as it outperforms motion-based alternatives on HighEndDrone, indicating the tested drone may utilize appearance-aware visual tracking.

We evaluate five representative tracking algorithms (Table 2) in benchtop attacks, and the HighEndDrone built-in HighEndDrone’s tracking in the in-flight setting. All trackers achieve real-time performance (~ 30 FPS or higher) on an Intel NUC 13 Pro (Core i5), except DaSiamRPN. For each tracker, we conduct 10 trials with two pedestrians of distinct appearance and a false-target separation of approximately 3 m. The attack uses a 4 Hz planning–execution cycle, with one gradient descent step per cycle and three trajectory samples for expectation-over-transformation.

5.3.2. Physical Experiment Results. The attack achieves an average target switch success rate of **79.1%/95.5%** across all benchtop and in-flight trials (Table 4). The overall performance is consistent with the simulation results (Table 3), validating the attack’s practicality in the real world.

Across the three experiment settings, as the scenario becomes more challenging in terms of reduced attacker capability and increasing real-world uncertainty and latency, the proposed attack demonstrates stable success. The comparable results between benchtop direct-contact and benchtop contact-free attacks demonstrate that physical access to the victim UAV is not necessary to launch the attack.

More importantly, the attack achieves 60% target switch success rate against the built-in tracking of the industrial-grade HighEndDrone during flight, with complete black-box knowledge (Figure 18). This not only demonstrates that the attack is physically achievable against an operating UAV system but also confirms this security vulnerability in commercial closed-source UAVs.

Failure cases of target switch (target loss and tracking maintained) typically include one or more transient target switch successes across non-consecutive frames. This suggests that the failures arise from acoustic control noise in real-world environments rather than inherent robustness in the target system. The results lead to Finding 6.

Finding 6: Physical experiments confirm that both direct-contact and contact-free attack vectors can corrupt UAV tracking, including successful exploits on a fully closed-source drone, although real-world environmental noise introduces additional challenges for the acoustic attack.



Figure 18: Before/after target switch HighEndDrone’s tracking. First row: daytime. Second row: at dusk.

6. Discussion

Potential defenses (details in Appendix A). Several mitigation strategies can be applied against Banshee, but each has notable limitations. Hardware approaches aim to eliminate the attack vector via acoustic isolation of the gyroscope [14], [19] or improved signal conditioning and ADC design [14]. However, these solutions are costly, require modifications to off-the-shelf components, and often entail substantial gimbal redesign, making them primarily viable for future UAV systems. Software defenses are easier to deploy, but remain challenging: Banshee induces large, rapid motions, while image stabilization demands high-rate processing. Existing detection methods based on simple motion models [68], [69], [70], sensor fusion [71], or specialized hardware [72] also do not align well with typical UAV gimbal architectures.

There are promising defense directions. One option, inspired by GPS spoofing defenses, is to use image-based signals such as visual odometry to detect inconsistencies between visual and inertial measurements [73], [74]. This approach is lightweight and software-based, but can be bypassed by sophisticated adversaries, as visual pipelines themselves are also vulnerable [11], [75], [76].

Attack controllability. Our attack achieves empirical success using a learning-based gimbal response model under a simplified control abstraction. A more principled control theory analysis of modern gimbals with multi-loop control and sensor fusion could further improve the fidelity of this gimbal control approximation. In addition, reliably selecting a specific target in the presence of multiple nearby objects remains an open challenge for practical deployment. We leave these directions to future work.

Remote attack feasibility. While we demonstrate attack success in physical experiments under the benchtop contact-free setup, we observe that motion injection is sensitive to sound pressure level (SPL). In our setup, placing the speaker approximately 4 inches below the gimbal achieves ~ 110 dB, inducing sufficient motion. Prior work [14], [15], [16], [17], [19], [40] demonstrates acoustic attacks over distances ranging from 10 cm to 7.6 m, while long-range ultrasonic emitters [19], [41] and laser-based approaches [18] can potentially extend attack distances beyond 100 m.

Threats to validity. While simulation enables diverse scenarios, it cannot fully capture real-world complexity. Our physical experiments validate the attack’s practicality in physical world, though under limited scenario diversity. Future work will expand real-world evaluations to additional gimbal and system models.

7. Conclusion

This paper introduces *Banshee*, the first physically realizable target switch attack on UAV visual tracking via acoustic injection on gimbal-camera systems. By empirically profiling gimbal acoustic responses, *Banshee* generates perturbations that induce directionally biased motion and probabilistically redirect tracking to another object. Our pipeline achieves $> 90\%$ success in simulation and successful black-box attacks on a commercial drone. These results show that acoustic vulnerabilities extend beyond sensor disruption to application-level compromise.

Acknowledgments. We thank the anonymous reviewers and our shepherd for their valuable feedback and constructive suggestions throughout the revision process. This work was partly supported by NSF under the Grant CNS-2321532 and the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks, Grant # 2112562.

Ethical Considerations

This work reveals an end-to-end vulnerability in UAV visual tracking systems, linking acoustic injection at the sensor level to application-level target-following failures. Given the potential safety risks, we take several measures to ensure responsible conduct.

Responsible disclosure. We have contacted the vendor of the UAV and gimbal systems used and shared our findings prior to publication. Product identifiers are anonymized to reduce misuse risk. We will follow coordinated disclosure practices and release full details only after mitigations are available.

Controlled experiments. All physical experiments were conducted in controlled environments using a single UAV at low altitude within confined areas. Tests were performed under supervision, including benchtop setups and carefully managed outdoor trials, ensuring no risk to bystanders, property, or other aircraft.

Limited artifact release. To support reproducibility, we release simulation code, profiling tools, and non-sensitive datasets, but exclude any software or hardware details that enable direct acoustic exploitation. The artifacts allow validation in simulation and study the implications for defense without providing end-to-end attack capabilities.

Purpose of research. Our goal is to expose a critical class of vulnerabilities in widely deployed UAV systems, not to enable misuse. By demonstrating the feasibility of acoustic-induced target switching, we highlight the need for more robust gimbal design, sensor fusion, and tracking algorithms.

We believe responsible disclosure of these findings will support timely mitigation by manufacturers, researchers, and regulators.

In summary, this work follows principles of responsible disclosure, controlled experimentation, limited release, and a focus on improving UAV system safety and robustness.

LLM Usage Considerations

Originality. LLMs were used only for editorial refinement (e.g., clarity, grammar, and style). All scientific content—including problem formulation, methodology, experiments, and analysis—was developed and validated by the authors. The literature review and citation decisions were performed manually. In accordance with conference guidelines: “*LLMs were used for editorial purposes in this manuscript, and all outputs were inspected by the authors to ensure accuracy and originality.*”

Transparency. LLMs are not part of our methodology or evaluation. All experiments and analyses are fully reproducible without any LLM service, and no technical contribution or conclusion depends on LLM outputs.

Responsibility. We did not train or fine-tune any LLMs or collect data for that purpose, nor did we provide proprietary or sensitive information to LLM services. As such, the environmental and ethical impact is minimal, and all contributions remain the authors’ own.

References

- [1] Z. Han, R. Zhang, N. Pan, C. Xu, and F. Gao, “Fast-tracker: A robust aerial system for tracking agile target in cluttered environments,” in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 328–334.
- [2] H. Cheng, L. Lin, Z. Zheng, Y. Guan, and Z. Liu, “An autonomous vision-based target tracking system for rotorcraft unmanned aerial vehicles,” in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 1732–1738.
- [3] A. Maalouf, N. Jadhav, K. M. Jatavallabhula, M. Chahine, D. M. Vogt, R. J. Wood, A. Torralba, and D. Rus, “Follow anything: Open-set detection, tracking, and following in real-time,” *IEEE Robotics and Automation Letters*, vol. 9, no. 4, pp. 3283–3290, 2024.
- [4] DJI, “Best drones that follow you automatically (2024),” <https://store.dji.com/content/camera-drone-that-follows-you>, 2024, accessed: 2025-08-08.
- [5] Skydio, “The best follow me drone in 2022,” <https://www.skydio.com/blog/10-reasons-skydio-makes-the-best-follow-me-drone>, 2022, accessed: 2025-02-25.
- [6] Autel, “Autel evo ii drone dynamic track mode full review,” <https://www.autelpilot.com/blogs/buying-guides/autel-evo-ii-drone-dynamic-tracking-mode>, 2020, accessed: 2025-02-25.
- [7] H. B. Salameh, M. Alhafnawi, A. Masadeh, and Y. Jararweh, “Federated reinforcement learning approach for detecting uncertain deceptive target using autonomous dual uav system,” *Information Processing & Management*, vol. 60, no. 2, p. 103149, 2023.
- [8] J. Li, J. Brewington, Q. Zhang, and Z. M. Mao, “Wip: Hijacking attacks on uav follow-me systems in realistic scenarios.”
- [9] J. Hibberd, “Using drones for peeping, burglaries on rise: “it’s gotten dramatically worse”,” <https://www.hollywoodreporter.com/lifestyle/lifestyle-news/drones-spying-robberies-solutions-hollywood-1236166714/>, 2025, accessed: 2025-08-13.

- [10] P. Dolan, "Like moths to a false flame: Lethality and protection through deception operations," https://www.army.mil/article/286861/like_moths_to_a_false_flame_lethality_and_protection_through_deception_operations, 2025, accessed: 2025-08-13.
- [11] D. Davidson, H. Wu, R. Jellinek, V. Singh, and T. Ristenpart, "Controlling {UAVs} with sensor input spoofing attacks," in *10th USENIX workshop on offensive technologies (WOOT 16)*, 2016.
- [12] D. Bereska, K. Daniec, S. Fraś, K. Jedrasiak, M. Malinowski, and A. Nawrat, "System for multi-axial mechanical stabilization of digital camera," in *Vision Based Systems for UAV Applications*. Springer, 2013, pp. 177–189.
- [13] A. Altan and R. Hacıoğlu, "Model predictive control of three-axis gimbal system mounted on uav for real-time target tracking under external disturbances," *Mechanical Systems and Signal Processing*, vol. 138, p. 106548, 2020.
- [14] T. Trippel, O. Weisse, W. Xu, P. Honeyman, and K. Fu, "Walnut: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks," in *2017 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2017, pp. 3–18.
- [15] Y. Tu, Z. Lin, I. Lee, and X. Hei, "Injected and delivered: Fabricating implicit control over actuation systems by spoofing inertial sensors," in *27th USENIX Security Symposium (USENIX Security 18)*. Baltimore, MD: USENIX Association, Aug. 2018, pp. 1545–1562. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity18/presentation/tu>
- [16] Y. Son, H. Shin, D. Kim, Y. Park, J. Noh, K. Choi, J. Choi, and Y. Kim, "Rocking drones with intentional sound noise on gyroscopic sensors," in *24th USENIX security symposium (USENIX Security 15)*, 2015, pp. 881–896.
- [17] M. Gao, L. Zhang, L. Shen, X. Zou, J. Han, F. Lin, and K. Ren, "Kite: Exploring the practical threat from acoustic transduction attacks on inertial sensors," in *Proceedings of the 20th ACM conference on embedded networked sensor systems*, 2022, pp. 696–709.
- [18] N. Shamsi, K. Chandrasekar, Y. Long, C. Limbach, K. Rebello, and K. Fu, "Wip: Threat modeling laser-induced acoustic interference in computer vision-assisted vehicles."
- [19] X. Ji, Y. Cheng, Y. Zhang, K. Wang, C. Yan, W. Xu, and K. Fu, "Poltergeist: Acoustic adversarial machine learning against cameras and computer vision," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021.
- [20] Y. J. Jia, Y. Lu, J. Shen, Q. A. Chen, H. Chen, Z. Zhong, and T. W. Wei, "Fooling detection alone is not enough: Adversarial attack against multiple object tracking," in *International Conference on Learning Representations (ICLR'20)*, 2020.
- [21] R. Muller, Y. Man, Z. B. Celik, M. Li, and R. Gerdes, "Physical hijacking attacks against object trackers," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2309–2322.
- [22] C. Ma, N. Wang, Z. Zhao, Q. Wang, Q. A. Chen, and C. Shen, "Controlloc: Physical-world hijacking attack on visual perception in autonomous driving," *arXiv preprint arXiv:2406.05810*, 2024.
- [23] C. Wang, Y. Man, R. Muller, M. Li, Z. B. Celik, R. Gerdes, and J. Petit, "Physical id-transfer attacks against multi-object tracking via adversarial trajectory."
- [24] S. Xie, M. H. Fakihi, J. Lu, F. Alshammari, N. Wang, T. Sato, H. Bouzidi, M. A. A. Faruque, and Q. A. Chen, "Flytrap: Physical distance-pulling attack towards camera-based autonomous target tracking systems," 2025. [Online]. Available: <https://arxiv.org/abs/2509.20362>
- [25] J. Ji, N. Pan, C. Xu, and F. Gao, "Elastic tracker: A spatio-temporal trajectory planner for flexible aerial tracking," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 47–53.
- [26] Y. Gao, J. Ji, Q. Wang, R. Jin, Y. Lin, Z. Shang, Y. Cao, S. Shen, C. Xu, and F. Gao, "Adaptive tracking and perching for quadrotor in dynamic scenarios," *IEEE Transactions on Robotics*, vol. 40, pp. 499–519, 2023.
- [27] S. Liu, X. Li, H. Lu, and Y. He, "Multi-object tracking meets moving uav," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8876–8885.
- [28] Y. Li, C. Fu, F. Ding, Z. Huang, and G. Lu, "Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 923–11 932.
- [29] J. Ye, C. Fu, F. Lin, F. Ding, S. An, and G. Lu, "Multi-regularized correlation filter for uav tracking and self-localization," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 6, pp. 6004–6014, 2021.
- [30] D. Xing, N. Evangeliou, A. Tsoukalas, and A. Tzes, "Siamese transformer pyramid networks for real-time uav tracking," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2022, pp. 2139–2148.
- [31] Z. Cao, Z. Huang, L. Pan, S. Zhang, Z. Liu, and C. Fu, "Tctrack: Temporal contexts for aerial tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 14 798–14 808.
- [32] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu, "Learning spatio-temporal transformer for visual tracking," *CoRR*, vol. abs/2103.17154, 2021. [Online]. Available: <https://arxiv.org/abs/2103.17154>
- [33] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ATOM: accurate tracking by overlap maximization," *CoRR*, vol. abs/1811.07628, 2018. [Online]. Available: <http://arxiv.org/abs/1811.07628>
- [34] G. Bhat, M. Danelljan, L. V. Gool, and R. Timofte, "Learning discriminative model prediction for tracking," *CoRR*, vol. abs/1904.07220, 2019. [Online]. Available: <http://arxiv.org/abs/1904.07220>
- [35] Y. Du, J. Wan, Y. Zhao, B. Zhang, Z. Tong, and J. Dong, "Giaotracker: A comprehensive framework for mcmot with global information and optimizing strategies in visdrone 2021," in *Proceedings of the IEEE/CVF International conference on computer vision*, 2021, pp. 2809–2819.
- [36] J. Jeong, D. Kim, J.-H. Jang, J. Noh, C. Song, and Y. Kim, "Un-rocking drones: Foundations of acoustic injection attacks and recovery thereof," in *NDSS*, 2023.
- [37] C. Ma, N. Wang, Q. A. Chen, and C. Shen, "Wip: Towards the practicality of the adversarial attack on object tracking in autonomous driving," in *ISOC Symposium on Vehicle Security and Privacy (VehicleSec)*, 2023.
- [38] T. Liu, C. Yang, X. Liu, R. Han, and J. Ma, "Rpau: Fooling the eyes of uavs via physical adversarial patches," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 3, pp. 2586–2598, 2023.
- [39] R. R. Wiyatno and A. Xu, "Physical adversarial textures that fool visual object tracking," in *Proceedings of the IEEE/CVF International Conference on computer vision*, 2019, pp. 4822–4831.
- [40] W. Zhu, X. Ji, Y. Cheng, S. Zhang, and W. Xu, "TPatch: A triggered physical adversarial patch," in *32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA: USENIX Association, Aug. 2023, pp. 661–678. [Online]. Available: <https://www.usenix.org/conference/usenixsecurity23/presentation/zhu>
- [41] wikipedia, "Long-range acoustic device," https://en.wikipedia.org/wiki/Long-range_acoustic_device, 2025, accessed: 2025-08-24.
- [42] X. Yang, J. Yang, J. Yan, Y. Zhang, T. Zhang, Z. Guo, X. Sun, and K. Fu, "Scrdet: Towards more robust detection for small, cluttered and rotated objects," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 8232–8241.
- [43] C. Xu, J. Ding, J. Wang, W. Yang, H. Yu, L. Yu, and G.-S. Xia, "Dynamic coarse-to-fine learning for oriented tiny object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 7318–7328.

- [44] OlliW, "Storm32bgc gimbal controller," https://www.olliw.eu/storm32bgc-v1-wiki/Tuning_Guide, 2015, accessed: 2025-11-07.
- [45] BaseCam, "Simplebgc gimbal controller," <https://www.basecamelectronics.com/>, 2025, accessed: 2025-11-07.
- [46] InvenSense, "Mpu-6000 and mpu-6050 product specification," <https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf>, 2025, accessed: 2025-11-10.
- [47] STMicroelectronic, "L3g4200d product specification," <http://wikitronica.labc.usb.ve/images/f/fc/L3G4200D.pdf>, 2025, accessed: 2025-11-10.
- [48] BOSCH, "Bmi160 data sheet," <https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmi160-ds000.pdf>, 2025, accessed: 2025-11-10.
- [49] STMicroelectronic, "Lsm6ds3tr-c data sheet," <https://www.st.com/resource/en/datasheet/lsm6ds3tr-c.pdf>, 2025, accessed: 2025-11-10.
- [50] R. J. Rajesh and P. Kavitha, "Camera gimbal stabilization using conventional pid controller and evolutionary algorithms," in *2015 International Conference on Computer, Communication and Control (IC4)*, 2015, pp. 1–6.
- [51] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*. Ieee, 2016, pp. 3464–3468.
- [52] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 583–596, 2014.
- [53] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8971–8980.
- [54] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.
- [55] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sep 2004, pp. 2149–2154.
- [56] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4282–4291.
- [57] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 101–117.
- [58] C. Fu, K. Lu, G. Zheng, J. Ye, Z. Cao, B. Li, and G. Lu, "Siamese object tracking for unmanned aerial vehicle: A review and comprehensive analysis," *Artificial Intelligence Review*, vol. 56, no. Suppl 1, pp. 1417–1477, 2023.
- [59] Ultralytics, "Quick start guide: Nvidia jetson with ultralytics yolo11," <https://docs.ultralytics.com/guides/nvidia-jetson/#nvidia-jetson-agx-orin-developer-kit-64gb>, 2025, accessed: 2025-07-04.
- [60] K. Yi, K. Luo, X. Luo, J. Huang, H. Wu, R. Hu, and W. Hao, "Ucm-track: Multi-object tracking with uniform camera motion compensation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 38, no. 7, 2024, pp. 6702–6710.
- [61] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [62] E. Real, J. Shlens, S. Mazzocchi, X. Pan, and V. Vanhoucke, "Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5296–5305.
- [63] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [64] P. D. Team, "Px4 autopilot," <https://github.com/PX4/PX4-Autopilot>, 2025, accessed: 2025-08-26.
- [65] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.
- [66] M. Moussaïd, N. Perozo, S. Garnier, D. Helbing, and G. Theraulaz, "The walking behaviour of pedestrian social groups and its impact on crowd dynamics," *PloS one*, vol. 5, no. 4, p. e10047, 2010.
- [67] D. Yang, Ü. Özgüner, and K. Redmill, "A social force based pedestrian motion model considering multi-pedestrian interaction with a vehicle," *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, vol. 6, no. 2, pp. 1–27, 2020.
- [68] R. Quinonez, J. Giraldo, L. Salazar, E. Bauman, A. Cardenas, and Z. Lin, "Savior: securing autonomous vehicles with robust physical invariants," in *Proceedings of the 29th USENIX Conference on Security Symposium*, ser. SEC'20. USA: USENIX Association, 2020.
- [69] Y. Wang, C. Sun, Q. Liu, B. Su, Z. Zhang, M. Norris, G. Tan, and J. Ma, "Vimu: Effective physics-based realtime detection and recovery against stealthy attacks on uavs," 2025. [Online]. Available: <https://arxiv.org/abs/2504.20569>
- [70] H. Meng, S. Luo, Z. Liang, Q. Huang, A. Khazraei, and M. Pajic, "Mars: Defending unmanned aerial vehicles from attacks on inertial sensors with model-based anomaly detection and recovery," 2025. [Online]. Available: <https://arxiv.org/abs/2505.00924>
- [71] S. Lee, "Gyro-mag: Attack-resilient system based on sensor estimation," *Sensors*, vol. 25, no. 7, 2025. [Online]. Available: <https://www.mdpi.com/1424-8220/25/7/2208>
- [72] Y. Wang, Y. Tu, S. Rampazzi, Z. Lin, I. Lee, and X. Hei, *ADC-Bank: Detecting Acoustic Out-of-Band Signal Injection on Inertial Sensors*, 02 2024, pp. 53–72.
- [73] N. Gu, F. Xing, and Z. You, "Gnss spoofing detection based on coupled visual/inertial/gnss navigation system," *Sensors*, vol. 21, no. 20, 2021. [Online]. Available: <https://www.mdpi.com/1424-8220/21/20/6769>
- [74] M. Varshosaz, A. Afary, B. Mojaradi, M. Saadatseresht, and E. Ghanbari Parmehr, "Spoofing detection of civilian uavs using visual odometry," *ISPRS International Journal of Geo-Information*, vol. 9, no. 1, 2020. [Online]. Available: <https://www.mdpi.com/2220-9964/9/1/6>
- [75] A. Ranjan, J. Janai, A. Geiger, and M. J. Black, "Attacking optical flow," 2019. [Online]. Available: <https://arxiv.org/abs/1910.10053>
- [76] J. Schmalfuss, P. Scholze, and A. Bruhn, "A perturbation-constrained adversarial attack for evaluating the robustness of optical flow," 2022. [Online]. Available: <https://arxiv.org/abs/2203.13214>
- [77] J. Jeong, D. Kim, J. Jang, J. Noh, C. Song, and Y. Kim, "Un-rocking drones: Foundations of acoustic injection attacks and recovery thereof," 01 2023.
- [78] H. Sathaye, M. Strohmeier, V. Lenders, and A. Ranganathan, "An experimental study of {GPS} spoofing and takeover attacks on {UAVs}," in *31st USENIX security symposium (USENIX security 22)*, 2022, pp. 3503–3520.
- [79] D. F. Kune, J. Backes, S. S. Clark, D. Kramer, M. Reynolds, K. Fu, Y. Kim, and W. Xu, "Ghost talk: Mitigating emi signal injection attacks against analog sensors," in *2013 IEEE Symposium on Security and Privacy*, 2013, pp. 145–159.

Appendix A.

Detailed Discussion of Defenses

Physical layer. A direct mitigation is to eliminate the attack vector via improved physical housing of the gimbal

gyroscope. Acoustic isolation using microfibrus metallic fabric, MEMS acoustic metamaterials, or acoustic foam can attenuate injected signals [14], [19]. However, these approaches increase weight, cost, and design complexity, and may still be bypassed by strong acoustic signals.

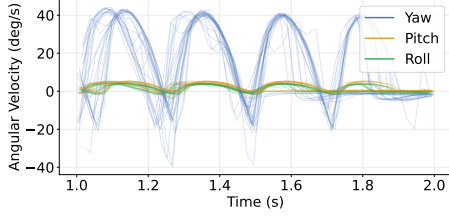


Figure 19: 3-axis angular velocity readings over 25 consecutive injections with directional biasing.

Hardware layer. Secure sensor design can mitigate acoustic injection by strengthening signal conditioning in MEMS IMUs. Gyroscopes and accelerometers infer motion from small proof-mass voltages that are amplified, filtered, and sampled. Acoustic attacks exploit weaknesses in this pipeline, such as amplifier clipping (introducing DC bias) or inadequate low-pass filtering. Secure sampling techniques, e.g., randomized or out-of-phase sampling aligned with sensor resonance, can further reduce vulnerability [14]. However, these defenses require IMU or ADC modifications and are typically infeasible for commodity UAV systems.

Software layer. Software-based defenses include digital image stabilization and learning-based denoising [19], [77]. However, *Banshee* induces much larger motion than typical blur, limiting the effectiveness of stabilization. Learning-based filtering can suppress injected noise but requires large models and extensive training, introducing latency incompatible with real-time gimbal stabilization.

Detection. Detection methods include model-based approaches (e.g., SAVIOR [68], VIMU [69], MARS [70]) that predict IMU readings, and sensor-fusion approaches (e.g., Gyro-Mag [71], ADC-Bank [72]) that detect inconsistencies across sensors. However, model-based methods assume tractable system dynamics (e.g., rigid-body or aerodynamic models), which do not apply to gimbals with nonlinear controllers and friction. Sensor-fusion methods require additional sensors or hardware redundancy, which are often unavailable in gimbal cameras and may themselves be vulnerable to attacks [78], [79].

Camera motion compensation. Incorporating camera motion compensation (CMC) into tracking can reduce attack success rates, especially when trading target switch for target loss. However, the reduction is insufficient to fully mitigate *Banshee*. Moreover, CMC incurs significant computational overhead, limiting feasibility on resource-constrained UAVs, particularly when combined with heavy tracking models [60].

Visual odometry. Visual odometry (VO), which estimates camera egomotion from image sequences, can be used to detect acoustic injection by comparing VO estimates with

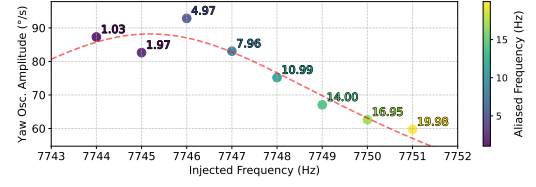


Figure 20: Injected signal frequency vs. yaw oscillation amplitude. Aliased signal frequency shown as changing color. Curve fitted by regression shown as dotted line.

gyroscope readings. Under normal operation, the two are aligned; under attack, they diverge. VO can run in parallel with stabilization and tracking, requiring only the onboard camera, making it lightweight and deployable via software updates. However, VO is also vulnerable to spoofing, and a sophisticated attacker may evade detection by aligning both modalities [11], [75], [76].

Recovery. Upon detection, the system can enter a recovery mode, e.g., limiting gimbal angular velocity to reduce attack impact, which we observe lowers attack success rates. Alternatively, the UAV can hover or land to preserve safety. These strategies improve robustness but may degrade normal tracking performance.

Appendix B. Gimbal Oscillation with Directional Biasing

Figure 19 shows 25 consecutive trials of injection at 23232 Hz, 50% signal power, and clockwise directional biasing in the yaw axis. Pitch, roll, and yaw axis angular velocities are overlaid into a single plot to show we achieve a consistent induced motion with little variation. While some traces show a slight variation in frequency, the amplitude remains consistent, ensuring that the accumulated angular displacement of the gimbal between different trials remains nearly identical as previously shown in Figure 7.

Appendix C. Effects of Detuning Injected Frequency

As the injected frequency f deviates from the natural resonant frequency f_n , the aliased frequency increases while the oscillation amplitude decreases. Equation (4) shows a linear relationship between f and f_d . The amplitude follows from Equation (5), extended with mechanical impedance:

$$Z_m = \sqrt{(4\pi f_n \zeta)^2 + \frac{1}{(2\pi f)^2} \left((2\pi f_n)^2 - (2\pi f)^2 \right)^2}, \quad (11)$$

where ζ is the damping ratio. As $|f - f_n|$ increases, Z_m grows, reducing amplitude.

In our evaluated systems, this reduction is moderate, but it becomes important when signal strength is limited or damping is high. This creates a trade-off: larger amplitudes increase attack impact, while higher frequencies yield more

stable responses and faster optimization. We model this trade-off by fitting ζ , F_0 , m , and f_n via regression to measured data. The aliased frequency follows a near-linear trend ($R^2 = 0.99$), enabling accurate prediction. Figure 20 confirms that as f moves away from f_n , aliased frequency increases while amplitude decreases, and that the fitted model closely matches observations.

Appendix D. Meta-Review

The following meta-review was prepared by the program committee for the 2026 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

D.1. Summary

This paper presents *Banshee*, an acoustic injection attack against UAV gimbal-stabilized visual tracking systems. The attack exploits MEMS gyroscope resonance to induce abnormal gimbal motion and camera viewpoint drift, which can cause visual trackers to lose the true target or switch to a nearby object. The paper develops an attack pipeline combining offline gimbal profiling with an online adaptive signal injection strategy. The evaluation includes both simulation and real-world experiments on a commercial drone platform, demonstrating how sensor-level disturbances can propagate through stabilization and affect application-level tracking behavior.

D.2. Scientific Contributions

- 3) Creates a New Tool to Enable Future Science.
- 4) Provides a Valuable Step Forward in an Established Field.
- 5) Establishes a New Research Direction.

D.3. Reasons for Acceptance

- 1) The paper demonstrates a practically relevant cross-domain attack path in autonomous systems, showing that acoustic injection at the sensing layer can propagate through gimbal stabilization and induce failures in visual tracking pipelines used by UAVs.
- 2) The work presents a complete attack workflow that combines offline profiling with an online adaptive attack loop, and evaluates the approach across multiple tracking algorithms and experimental settings.
- 3) The paper provides a useful experimental framework and methodology for studying perception-layer vulnerabilities in UAV systems, which can support follow-on research in robustness, detection, and mitigation.

D.4. Noteworthy Concerns

- 1) **Interpretation of axis-dependent behavior.** The paper identifies axis-dependent responses (e.g., yaw-dominant behavior) through black-box profiling. While the empirical procedure is clearly described, it remains unclear whether such axis-selective resonance can be consistently identified across different gyroscope designs and implementations. For example, it is unclear whether similar yaw-dominant behavior can always be found for arbitrary commercial gimbal systems, or whether the observed behavior depends on specific hardware characteristics.

2) **Limited characterization of intermediate mechanisms.**

The paper demonstrates that acoustic injection induces gimbal motion and leads to tracking degradation. However, the relationship between the induced oscillatory motion and the resulting tracking behavior is not fully characterized. In particular, it remains unclear how different motion patterns (e.g., oscillation versus accumulated drift) contribute to the observed tracking outcomes.

3) **Limitations of the in-flight evaluation.** The in-flight experiments demonstrate that acoustic injection can affect the built-in tracking system of a commercial drone. However, the evaluation is conducted under specific conditions (e.g., limited number of targets, controlled scenarios, and close-range acoustic injection), and does not explore how the attack behaves under more diverse real-world settings. As a result, the results should be interpreted within the scope of the evaluated scenarios.

Active Electromagnetic Side-Channel Analysis: Crossing Physical Security Boundaries through Impedance Variations

Taiki Kitazawa¹, Lennert Wouters², Benedikt Gierlichs², Daisuke Fujimoto¹,
Ingrid Verbauwhede² and Yuichi Hayashi¹

¹ Nara Institute of Science and Technology (NAIST), Japan
{[kitazawa.taiki.kq8](mailto:kitazawa.taiki.kq8@is.naist.jp), [fujimoto.yu-ichi](mailto:fujimoto.yu-ichi@is.naist.jp)}@is.naist.jp

² COSIC, KU Leuven, Belgium
{[lennert.wouters](mailto:lennert.wouters@esat.kuleuven.be), [benedikt.gierlichs](mailto:benedikt.gierlichs@esat.kuleuven.be), [ingrid.verbauwhede](mailto:ingrid.verbauwhede@esat.kuleuven.be)}@esat.kuleuven.be

Abstract. Embedded devices that are physically unreachable or contained within tamperproof enclosures are often considered naturally resilient to physical side-channel attacks. We present an active EM side-channel attack technique that enables side-channel attacks across the security boundaries introduced by these physical security measures. Our technique actively induces side-channel leakage based on a relay mechanism, which serves as a means of leakage propagation from the cryptographic IC to the attacker via intermediate Relay Points located within the boundary. In the first leakage path, the instantaneous current consumption caused by transistor switching inside the cryptographic IC affects the impedance of nearby nonlinear elements (such as regulators) that act as Relay Points. This phenomenon is inevitable due to the necessity of the regulator’s operations to ensure the provision of a stable voltage to the cryptographic IC. In the second path, EM waves irradiated by the attacker from outside the security boundary create a leakage channel where reflected waves are modulated by the impedance variation of Relay Points, and leak outside as side-channel information. We experimentally demonstrate that side-channel attacks can be performed from the power cable on the primary side of an AC/DC adapter, even in environments protected by physical security measures, including a shielded box and ferrite cores. This attack can be executed non-invasively using EM waves in the hundreds of MHz band, and has the advantage of being able to actively control the presence and intensity of leakage. The proposed attack technique may be applicable in a wide range of applications as it exploits the behavior of nonlinear elements present in all embedded systems. As countermeasures against the proposed attack, we discuss the effectiveness of EM fault sensors for detecting continuous wave irradiation, induced EM interference detectors with broadband monitoring capabilities, and tamper detection systems utilizing radio frequencies. This research highlights the importance of considering active EM measurements as a new threat model in physical security evaluations.

Keywords: Side-channel Analysis · Power/Electromagnetic/Backscatter/Impedance Analysis

1 Introduction

Side-Channel Analysis (SCA) is the art of recovering secret information through the analysis of a physical observable such as the instantaneous current consumption [KJJ99]. Power analysis primarily exploits the dynamic power consumption associated with transistors switching inside Integrated Circuits (ICs) as a leakage source. Electromagnetic

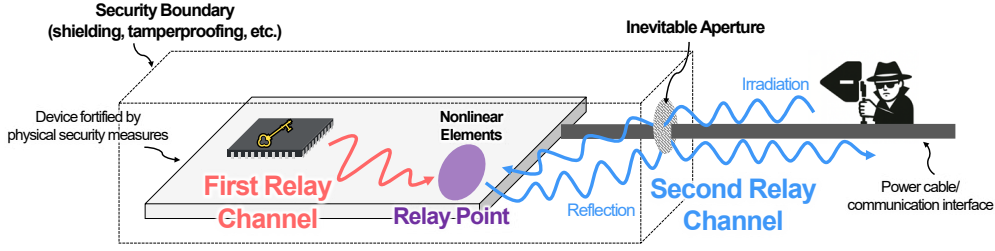


Figure 1: Overview of Active EMA

Analysis (EMA), which exploits unintentional electromagnetic (EM) emanations during cryptographic operations, is a natural extension of power analysis [QS01, GMO01]. Unlike power analysis, EMA enables attacks using EM probes or antennas, without requiring physical contact or modification.

SCA fundamentally consists of two stages: “measurement” and “analysis” of side-channel information. This two-stage nature of SCA leads to a basic defense principle: if measurement can be prevented, subsequent analysis cannot succeed. Based on this principle, various countermeasures have been proposed at the board design [HHM⁺13], circuit level [HB25], architecture, and algorithm levels [TG07]. Among these, forming security boundaries using physical security measures is one of the most robust approaches.

Physical security measures create distinct security boundaries between potential attackers and target devices [Com22, Nat01, Nat19]. For example, tamper detection [GS22, OI18, ION⁺18] protects by detecting and responding to unauthorized physical access, while shielding [Int15] prevents emanations from leaking outside the security boundary while also suppressing external interference and disturbance. These technologies are complemented by other physical security measures such as tamper evidence, tamper resistance, and zoning [Wei00]. The strength of physical layer countermeasures lies in their ability to enforce physical spatial separation between attackers and devices, providing algorithm-independent security that continues protection even when new vulnerabilities are discovered in cryptographic devices.

Even if such security boundaries are established through physical security measures, inevitable apertures exist to accommodate power delivery and communication interfaces [OI18]. Information leakage through such apertures on security boundaries has been demonstrated as leakage from conducted EM emissions¹ through power and communication cables [HSK⁺10, SHH⁺09, GPT15], and leakage from clock jitter and phase noise in input/output (I/O) signals of communication interfaces [SMTG23, O’F24, AHC⁺24].

However, such attacks are performed through passive observation of side-channel information from outside the security boundary. Therefore, it is possible to try to confine this leakage within the boundary by applying techniques such as filtering and isolation. In such cases, leakage generated from cryptographic devices is reduced below background noise levels outside the security boundary, making passive SCA infeasible for attackers without physical access to the inside of the boundary.

Physical security measures that impede physical access while confining leakage within the security boundary can be considered exceptionally robust. In this work, we propose the use of active EM measurement methods that allow us to pierce the security boundaries put in place by such physical security measures. The proposed Active EMA technique exploits electrical characteristics of nonlinear components in embedded devices and is based on the

¹Conducted EM emissions are unintended RF currents and voltages that travel along power or signal conductors (e.g., power or communication lines). Radiated EM emissions, by contrast, are unintended EM fields that propagate through free space from structures acting as unintentional antennas (e.g., cables, PCB traces, or enclosures) [PSS22].

following relay mechanism (see [Figure 1](#)).

First Relay Channel: This channel relays side-channel leakage to nonlinear elements in the device. These nonlinear components are electrically and electromagnetically coupled with the target IC. Specifically, the transient current generated by Complementary Metal-Oxide-Semiconductor (CMOS) transistors switching inside cryptographic ICs causes voltage fluctuations (so-called ground bounce) through parasitic EM coupling on the IC and board. These voltage fluctuations affect the impedance of nonlinear elements such as regulators, creating Relay Points. In particular, since cryptographic ICs and regulators are electrically coupled for stable operation of the device, this relay path inevitably exists.

Second Relay Channel: This channel enables us to cross the physical security boundary by actively irradiating EM waves from outside the boundary. Specifically, when irradiated EM waves reach nonlinear elements serving as Relay Points, modulated reflected waves emerge due to variations in their impedance. These reflected waves thus serve as a relay channel that pierces the physical security boundary from within.

Through the relay mechanisms, we demonstrate attacks against cryptographic devices with physical security measures that are considered challenging or even impossible to compromise using conventional passive measurement techniques. Specifically, we recover full Advanced Encryption Standard (AES) secret keys from the primary side of the Alternating Current (AC)/Direct Current (DC) adapter, even in environments protected by a shielded box and ferrite cores. The technique introduced here is related to other side-channel approaches such as impedance- and backscatter-based analysis, but differs from them in that it actively induces leakage (see [Section 2.3](#)). Furthermore, Active EMA has the following advantageous characteristics: (1) Controllability: Through active measurement, attackers can actively control the presence and intensity of leakage. (2) Non-invasive, non-contact execution: EM wave attacks in the hundreds of MHz band enable attack execution without physical contact. (3) Applicability: Exploits nonlinear elements commonly found in typical electronic devices, making it applicable to various embedded systems under realistic scenarios.

The contributions of this paper are as follows:

- We propose a novel active EM side-channel measurement method based on the relay mechanism.
- We experimentally demonstrate that secret keys can be recovered from cryptographic devices with physical security measures, such as shielding and ferrite cores, applied to the primary side power cable of AC/DC adapters.
- We demonstrate that attackers can arbitrarily adjust the intensity and range of leakage by controlling the intensity of irradiated signals.
- We summarize defense guidelines against the proposed attack method for strengthening physical security boundaries.

In the remaining paper, we will summarize related works in [Section 2](#). In [Section 3](#), we will explain the threat model of Active EMA. In [Section 4](#), we will present the details of the relay mechanism of our proposed method: Active EMA. Our experimental setups and results are described in [Section 5](#). In [Section 6](#), we summarize countermeasures and future challenges. Finally, [Section 7](#) concludes the paper.

2 Related Work and Background

2.1 Electromagnetic Analysis (EMA)

EMA is a kind of SCA that recovers secret keys and data using EM waves unintentionally radiated from cryptographic devices [QS01, GMO01, AARR03]. Attackers capture EM waves using antennas or probes and analyze them with receivers. EMA includes “Local EMA” [PSQ07, SSS⁺13, SBO⁺15, LDMPT15], which observes localized EM waves from ICs, and “Far-Field EMA” [AARR03, SHH⁺09, GPT15], which leak far away as unintended emissions in the MHz to GHz bands. Recently, threats have been reported where side-channel information is superimposed on wireless communications due to coupling between digital and analog circuits in System-on-a-Chip (SoC), the so-called Screaming Channels [CPM⁺18, CFS20, GPNS25].

The advantage of EMA lies in non-invasive and non-physical contact measurement [KOP09, HMF07]. Still, since it is based on passive measurement, it is possible to reduce radiation intensity and make attacks difficult through design considerations of Electromagnetic Compatibility (EMC), Power Integrity (PI), and Signal Integrity (SI), as well as shielding and filtering. Screaming Channels can also prevent observation of side-channel information on communication channels by suppressing internal EM coupling.

2.2 SCA from the Primary Side

As SCA from the primary side, some attacks utilize pulse width changes in switching power supplies [SLT16, KFH24]. However, the target of conventional research is DC/DC regulators that directly supply power to cryptographic cores, requiring attackers to have proximity access to cryptographic devices. Furthermore, in real systems, as signals pass through multiple stages of regulators, side-channel information on primary side power lines is expected to be filtered down to background noise levels. Additionally, since these attacks are executed through passive measurement, leakage can be suppressed by countermeasures in the same way as passive EMA.

2.3 Impedance-/backscatter-based Analysis

Research has been reported on measuring internal IC impedance variations using THz-band EM waves [AJCZ19, WPZ22], but secret key recovery has not yet been achieved. Furthermore, the THz band has high attenuation, making measurements from a distance difficult and limiting them to those directly above the chip. In contrast, Monfared et al. proposed LeakyOhm using the several GHz band [MMT23, AGR24]. While it utilizes Power Delivery Network (PDN) impedance changes associated with channel formation changes in CMOS inside ICs, it requires the suspension of operation of cryptographic devices and a physical connection to the power supply. Additionally, the GHz band also has high attenuation, and non-contact attacks have not been realized.

2.4 Active/Passive TEMPEST

TEMPEST targets EM leakage of unencrypted I/O signals from computer peripherals, assuming attacks under Non-Line-of-Sight (NLOS) conditions from distances of tens of meters or more [KAK⁺22, vE85, Kuh05]. The recently reported Echo TEMPEST is an active attack that irradiates MHz-band EM waves and recovers I/O information from reflected waves [KFKH23, PLHD25]. While attack distance and resolution are controllable, the target is low-speed digital signals, making the measurement of side-channel information inside ICs difficult. This is because internal CMOS gates are small and can only be observed with EM waves above the GHz band, making propagation from a distance difficult.

3 Threat Model

Adversary's Goal: The attacker's objective is to obtain secret keys and data from cryptographic devices installed within security boundaries, without physical contact. Specifically, the attacker aims to actively relay side-channel information confined within the boundary outward, and recover secret keys by collecting and analyzing the leaked signals.

Scenario: The targeted cryptographic devices are assumed to be mission-critical systems such as Automated Teller Machines (ATMs), infrastructure control systems, servers equipped with Hardware Security Modules (HSMs), and smart meters. In these devices, clear security boundaries are formed by physical security measures such as shielding [Int15] and tamperproofing [GS22, OI18, ION⁺18], which thwart physical access within the boundary and SCA through passive measurement. However, practically inevitable apertures for the connection of power supply lines and communication interfaces still remain and serve as attack surfaces. In this scenario, we assume a situation where attackers irradiate high-frequency EM waves in the hundreds of MHz band into the boundary through these inevitable apertures. The EM waves propagate to nonlinear elements (Relay Points) whose impedance varies due to ground bounce generated during operation of cryptographic ICs (First Relay Channel). Subsequently, the modulated reflected waves generated from these nonlinear elements are emitted beyond the boundary (Second Relay Channel).

Attack Setup: The attacker utilizes the following equipment: a signal generator, a receiver, transmitting/receiving antennas/probes, and analysis equipment (e.g., Personal Computer (PC)). By optionally using signal amplifiers, it is possible to enhance irradiation intensity and reception sensitivity, allowing for attacks from greater distances and improving the quality of acquired traces.

4 Flow of Active EMA

This section explains the attack flow of Active EMA based on the relay mechanism (see Figure 2).

First Relay Channel (Section 4.1): A relay channel where side-channel information generated inside cryptographic ICs propagates to surrounding nonlinear elements through EM and electrical coupling. We refer to the nonlinear elements relaying side-channel information between the First Relay Channel and the Second Relay Channel as "Relay Points".

Second Relay Channel (Section 4.2): A relay channel where side-channel information that has reached Relay Points propagates beyond the security boundary by actively irradiated EM waves.

4.1 First Relay Channel Induced Through Ground Bounce

4.1.1 Ground Bounce Generated by Cryptographic Operations

The source of side-channel information leakage is the switching operation of CMOS gates within cryptographic ICs [MOP07]. When logic values of each logic circuit transition, dynamic power consumption occurs due to charging current and short-circuit current associated with transistor on/off operations. During cryptographic operations, many such switching events occur simultaneously, and their collective power consumption depends on data, enabling secret key recovery [KJJ99].

In power analysis, voltage fluctuations on the PDN of the cryptographic IC are measured². Such voltage fluctuations are known in the fields of EMC and PI as ground

²Similar information can be obtained from current fluctuations measured via shunt resistors

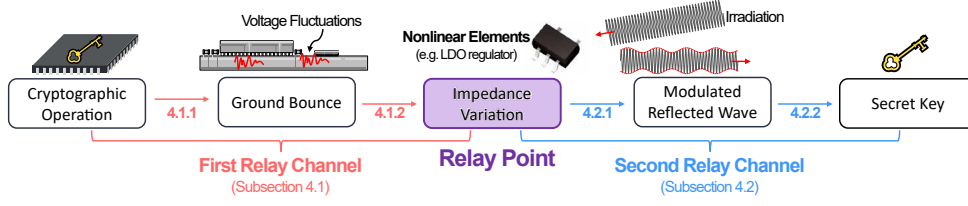


Figure 2: Flow of Active EMA

bounce caused by the simultaneous switching of CMOS gates [SE07]. Ground bounce occurs when transient charging current and short-circuit current induce voltage through parasitic components on the IC core, package, and PDN. Therefore, such phenomena inevitably occur with IC operation, i.e., side-channel information is widely distributed on the PDN as voltage fluctuations.

4.1.2 Formation of Relay Points via Ground Bounce

Ground bounce containing side-channel information induces impedance variations in nonlinear elements existing on the same PDN of cryptographic ICs. These nonlinear elements function as Relay Points, effectively enabling us to cross security boundaries. The requirements for nonlinear elements to act as Relay Points are: (1) the impedance of nonlinear elements varies due to the influence of ground bounce, and (2) EM waves efficiently propagate from outside to inside the security boundary, reaching the relevant elements to generate reflected waves.

Impedance Variation: Nonlinear elements exist in typical components on Printed Circuit Boards (PCBs), including digital circuits within ICs, diodes (Transient Voltage Suppressors (TVSs), Light Emitting Diodes (LEDs), etc.), Multi-Layer Ceramic Capacitors (MLCCs), and regulators. The PDN of a cryptographic IC is no exception; nonlinear elements are used in peripheral circuits both inside and outside cryptographic ICs to ensure normal operation. These nonlinear elements show significant impedance changes compared to linear elements, even with weak voltage changes from ground bounce, potentially enabling the exploitation of side-channel information.

As a specific example, we explain impedance variations in Low-Dropout (LDO) regulators (see Figure 3). LDO regulators are widely used for both on-chip and off-chip power supply circuits, and its operation utilizes the characteristics of internal nonlinear elements. The purpose of the LDO regulator is to convert the voltage supplied from the input side (primary side) to the output side (secondary side) while dynamically adjusting the output current to maintain a constant voltage value. In detail, the operation works as follows: (1) the load consumes current, (2) the LDO's output voltage drops, (3) a difference from the reference voltage is detected, (4) the equivalent resistance of the power Field Effect Transistor (FET) changes, causing impedance variation, (5) output is compensated to maintain constant voltage. Assuming (1) and (2) are the result of current consumption and ground bounce due to cryptographic operations, we find that regulator impedance variation occurs at step (4). Furthermore, this behavior is essential for the stable operation of output-side loads, making impedance variations of peripheral nonlinear elements unavoidable [SJL⁺21].

Beyond regulators, other ubiquitous nonlinear components on the PDN can act as Relay Points. (1) P-N junction devices, including TVSs, ESD protection circuits, and indicator LEDs, are known to have capacitance and conductance that change depending on the applied bias. Even small ground-bounce fluctuations can therefore alter their electrical behavior [SB15, ch. 5]. (2) Decoupling MLCCs, particularly those using Class II dielectrics such as X5R or X7R, show a strong dependence of their capacitance and

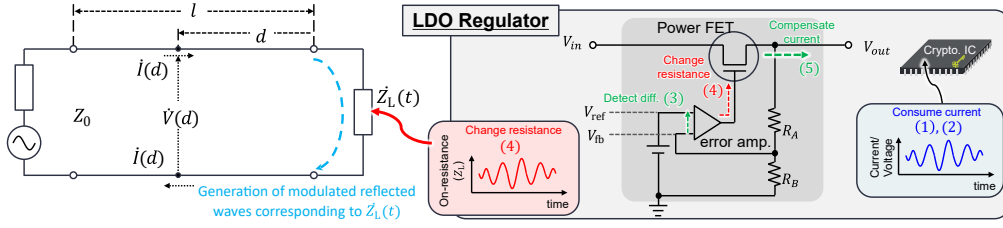


Figure 3: Flow from the occurrence of ground bounce to the generation of modulated reflected waves. (Left) Simplified transmission line model using a distribution multiplier circuit. The reflected waves are modulated according to the time variation of $Z_L(t)$. (Right) Example of impedance variation accompanying cryptographic operations of nonlinear elements. LDO regulator stabilizes the load operation, so the on-resistance variation is unavoidable.

losses on both DC and AC bias [NWM⁺11]³. This makes them especially susceptible to changes in operating conditions around cryptographic devices. (3) I/O interfaces and power-management circuits inside integrated-circuit packages also include non-linear transistors and protection structures.

Transfer Characteristics from Outside to Relay Points: Impedance variations at Relay Points are actively sensed via EM waves, as explained in Section 4.2.1 below. Therefore, Relay Points require sensitivity in frequency bands where EM waves—irradiated by the attacker and passing through apertures in the security boundary—can efficiently reach the nonlinear elements serving as Relay Points without attenuating below the background-noise level⁴.

First, we explain why impedance changes associated with channel formation states of CMOS gates inside cryptographic ICs, which were focused on in conventional impedance-based analysis [MMT23, AGR24, AJCZ19, WPZ22], are not suitable as Relay Points. Such impedance change points are located within the internal digital circuits of ICs, and their physical structures are extremely small at the nanometer scale. To propagate EM waves from there and sense them, it is necessary to inject EM waves in the GHz/THz bands with short wavelengths. However, as EM waves have increasing attenuation rates at higher frequencies, they are unsuitable for sensing from distant locations. Indeed, conventional research has required physical access to cryptographic devices (connection to the PDN via Sub Miniature Type-A (SMA) connectors) and temporary suspension of cryptographic processing to extract secret keys.

In contrast, the nonlinear elements focused on in this research are surface-mounted on PCBs or larger I/O circuits within the IC. Particularly, regulators are placed between power planes, making them easily reachable by EM waves from outside. Furthermore, as the physical structures of these elements are on the micrometer scale, they show sensitivity to the hundreds of MHz band with longer wavelengths than the GHz band [KFKH23]. As a result, they have smaller propagation attenuation when compared to the GHz/THz bands and offer good transfer characteristics, which have been confirmed in actual EM information leakage [KFKH23, dMSV20, IMK⁺25].

³In actual components, the voltage dependence of capacitance can be confirmed from datasheets and tools published by manufacturers (e.g., Murata Manufacturing Co., Ltd., SimSurfing, <https://ds.murata.com/simsurfing/mlcc.html>, accessed 28 September 2025)

⁴Due to EM reciprocity, propagation characteristics from nonlinear elements to the point where attackers observe EM waves are similar, so we consider only one direction here.

4.2 Second Relay Channel by Active EM Measurement

4.2.1 Modulated Reflected Waves Generated at Relay Points

We describe how to generate a Second Relay Channel that leverages the impedance variations at Relay Points. The Second Relay Channel uses active EM measurements to pierce the security boundaries. Specifically, attackers inject continuous waves into cryptographic devices, resulting in amplitude and phase modulated reflected waves, because of impedance changes at Relay Points. First, we explain the physical mechanism of modulated reflected wave generation, and then describe the method for constructing a relay channel by optimizing parameters such as irradiation frequency, amplitude, and transmission mode.

Mechanism of Modulated Reflected Wave Generation: The derivations in this Subsection follow the textbook [Poz21, ch. 2]. To explain how EM waves irradiate and propagate to cryptographic devices, we use the model of Figure 3, where a transmission line of length l is excited at an excitation source and terminated with load impedance $\dot{Z}_L = R + jX$. When the line length is sufficiently long relative to the excitation source wavelength, the line is treated as a distributed-element model, and voltage $\dot{V}$ and current $\dot{I}$ at distance $d = 0$ from the receiving end are expressed as follows⁵

$$\dot{V} = \dot{V}_i + \dot{V}_r, \quad \dot{I} = \dot{I}_i + \dot{I}_r = \frac{\dot{V}_i}{Z_0} - \frac{\dot{V}_r}{Z_0} \quad (1)$$

Here, Z_0 is the characteristic impedance of the line. The first terms ($\dot{V}_i, \dot{I}_i$) are incident wave components. In contrast, the second terms ($\dot{V}_r, \dot{I}_r$) are reflected wave components, which change depending on the impedance of the load. Specifically, they are expressed by the following relational expressions using the reflection coefficient $\dot{\Gamma}$.

$$\dot{V}_r = \dot{\Gamma} \dot{V}_i = |\dot{\Gamma}| \dot{V}_i e^{-j\phi}, \quad \dot{I}_r = -\dot{\Gamma} \dot{I}_i = -|\dot{\Gamma}| \dot{I}_i e^{-j\phi} \quad (2)$$

$$\therefore \dot{\Gamma} = (\dot{Z}_L - Z_0) / (\dot{Z}_L + Z_0) \quad (3)$$

$|\dot{\Gamma}|$ is the absolute value of the reflection coefficient, where $0 \leq |\dot{\Gamma}| \leq 1$. ϕ shows the phase component of the reflection coefficient. As seen from Equation 2 and Equation 3, reflected waves relative to incident waves change according to load impedance. That is, assuming the load impedance $\dot{Z}_L$ is the time-varying impedance of Relay Points containing side-channel information $\dot{Z}_L(t)$, voltage $\dot{V}(t)$ and reflected current $\dot{I}(t)$ on the transmission line become:

$$\dot{V}(t) = \dot{V}_i + \dot{V}_r(t) = \dot{V}_i \left(1 + |\dot{\Gamma}(t)| e^{-j\phi(t)} \right), \quad \dot{I}(t) = \dot{I}_i + \dot{I}_r(t) = \dot{I}_i \left(1 - |\dot{\Gamma}(t)| e^{-j\phi(t)} \right) \quad (4)$$

From this, we see that when the load (Relay Point) changes impedance over time, modulated signals with amplitude and phase modulation relative to the incident waves are generated. Thus, attackers can generate reflected waves modulated with side-channel information from Relay Points by propagating continuous EM waves with constant amplitude and phase as sensing signals.

Generation Method of Reflected Waves Modulated with Side-channel Information: From the above mechanism, attackers must first ensure that continuous waves reach the Relay Points. At that time, measurements are taken while optimizing the amplitude and frequency of EM waves used for sensing. This leads to problems where increasing amplitude reduces modulation depth due to self-interference, and challenges in searching for optimal irradiation frequency bands.

⁵At this time, input impedance $\dot{Z}_{in}$ seen from $d = 0$ is $\dot{Z}_L = \dot{Z}_{in}$. For simplicity, the attenuation constant $\alpha = 0$.

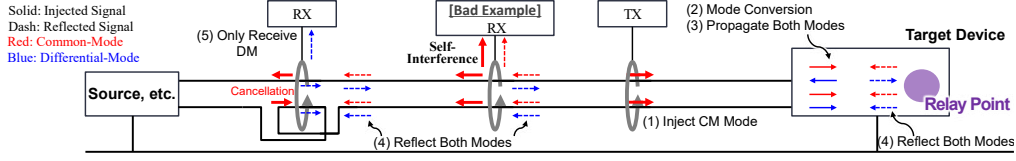


Figure 4: Active EMA measurement methods executed through transmission lines. When using the same mode, signals output from the transmitter directly superimpose on the receiver, degrading modulation depth due to self-interference effects, consequently leading to a decline in the accuracy of information acquisition. When using different modes, signals output from the transmitter cancel out on the receiver side, enabling the reduction of self-interference effects.

Algorithm 1: Frequency Sweep TVLA for Relay Channel Search

Input: $f_{\text{start}}, f_{\text{end}}, f_{\text{step}}, P_{\text{TX}}, N$
Output: T-test values $T[f_c]$ for each frequency f_c
for $f_c = f_{\text{start}}$ **to** f_{end} **step** f_{step} **do**
 $\text{injection_on}(P_{\text{TX}}, f_c);$
 $\text{adjust_gain}();$
 $(\text{traces}_{\text{fixed}}, \text{traces}_{\text{random}}) \leftarrow \text{collect_traces}(2N);$
 $(\text{traces}_{\text{fixed}}^{\text{pp}}, \text{traces}_{\text{random}}^{\text{pp}}) \leftarrow \text{apply_preprocessing}(\text{traces}_{\text{fixed}}, \text{traces}_{\text{random}});$
 $T[f_c] \leftarrow \text{TVLA}(\text{traces}_{\text{fixed}}^{\text{pp}}, \text{traces}_{\text{random}}^{\text{pp}});$
 $\text{injection_off}();$

The former challenge regarding self-interference can be addressed by the measurement method that focuses on different transmission modes, as shown in Figure 4. Self-interference is a problem where irradiated EM waves from the transmission side directly interfere with the reception side, reducing modulation depth and degrading information recovery accuracy. Particularly for attacks from distant locations, the power of reflected waves $\dot{P}_r$ becomes very small relative to incident waves $\dot{P}_i$ ($\dot{P}_i > \dot{P}_r$) due to increased attenuation effects with long propagation paths, making demodulation difficult. This method enables reception of reflected waves while suppressing direct interference of transmitted waves by using different transmission modes on the transmission and reception sides [PSS22, ch. 6]. For example, in transmission lines composed of two wires, EM waves propagate in either differential mode (DM)⁶ or common mode (CM)⁷. First, we inject CM or DM signals into the transmission lines. (2) Then, due to changes in geometric structure or the existence of parasitic coupling, some EM energy converts their mode. (3) The EM waves of both modes reach the relay points. (4) Both modes are reflected from the relay points. (5) Although both modes reach the receiver, we only observe one mode on the transmission side. This is because, if we set the same mode as the transmitter, direct interference occurs.

Next, the frequency to create the relay channel is determined by algorithm 1. (1) Sweep frequencies in the range of hundreds of MHz to several GHz that tend to be well distributed on electronic devices. (2) Receive reflected waves at each frequency and adjust receiver gain based on their intensity ($\text{adjust_gain}()$). (3) Acquire multiple traces and optionally apply preprocessing techniques ($\text{collect_traces}()$, $\text{apply_preprocessing}()$). (4) Perform Test Vector Leakage Assessment (TVLA) on acquired traces ($\text{TVLA}()$). (5) Based on TVLA results, we determine the most suitable frequency bands for relaying

⁶In differential mode, one wire serves as the forward path and the other as the return path for EM wave propagation.

⁷In common mode, EM waves propagate in the same direction on both wires, using the reference ground plane as the return path.

side-channel information.

Note that propagation characteristics to the Relay Points depend on board size and shape, cable length, and chassis structure. If these elements do not change significantly then the same carrier frequency bands can be used for similar target scenarios [HHM⁺13]⁸.

4.2.2 Analysis of Received EM Waves

Preprocessing: Reception and signal processing of EM waves consists of demodulation, outlier removal, and filtering. First, we apply demodulation to the received signals. Reflected signals generated by EM irradiation are expressed as amplitude and phase modulated signals $s_r(t) = A(t) \sin(2\pi f_{\text{inj}}t + \phi(t))$. Here, f_{inj} is the irradiation frequency, and $A(t)$ and $\phi(t)$ are the modulated amplitude and phase signals. After the reflected waves are down-converted to baseband signals at the receiver, they are represented as orthogonal components as follows:

$$I(t) = A(t) \cos \phi(t), \quad Q(t) = A(t) \sin \phi(t), \quad (5)$$

Here, variations in Relay Point impedance are obtained by applying amplitude and phase demodulation.

$$A(t) = \sqrt{I(t)^2 + Q(t)^2}, \quad \phi(t) = \text{atan2}(Q(t), I(t)). \quad (6)$$

Time-series data obtained through this demodulation processing directly reflects impedance variations at Relay Points, and this data is used for analysis. In this research, we utilized amplitude demodulation traces, which allowed for clearer observation of impedance changes.

Next, in Active EMA measurements under real environments, surge-like noise may be present on traces due to external EM noise or unintended modulation from other device operations, so outlier removal is optionally applied. Abnormal traces are removed by anomaly detection using the Interquartile Range (IQR) method. A detailed explanation is shown in Appendix A.

Finally, a High-Pass Filter (HPF) and a Low-Pass Filter (LPF) are applied. The HPF is used to suppress carrier intensity drift caused by temperature characteristics of transmission lines, peripheral equipment, and cryptographic devices themselves [MMT23]. If needed the LPF is applied to reduce high-frequency noise within the reception frequency band.

Leakage assessment. We use TVLA [GJJR11]: two trace sets (fixed plaintext vs. random plaintexts) are compared using Welch's t-test. Here, if $|t| > 4.5$, leakage is determined to exist with a confidence level of > 0.99999 . For key recovery on AES, we employ the widely used Hamming weight of the S-box output leakage model to perform CPA [MOP07], since impedance variations at Relay Points are caused by ground bounce accompanying transistor switching, they are amplitude/phase modulated in reflected waves.

5 Experimental Results

This section experimentally verifies the feasibility of AES secret key recovery through Active EMA based on the relay mechanism. First, Section 5.1 presents a preliminary setup for evaluating the basic feasibility of the proposed method and for profiling the relay channel, as well as a setup with security boundaries, assuming a real environment with physical security measures such as shielding and filtering applied. Next, Section 5.2 demonstrates leakage induction through the proposed method using the preliminary setup and conducts a frequency profiling of the available relay channel. Subsequently, Section 5.3 shows that secret key recovery is possible from reflected waves by injecting EM waves

⁸While execution is possible with online search alone, measurement time becomes enormous in proportion to sweep range, so combined use with advance profiling is practical.

from the external DC power line of the PCB. Then, Section 5.4 demonstrates that in cryptographic devices installed within security boundaries, side-channel information can be leaked outside the security boundaries through filters, adapters, and a shielded box from the primary side power cable of AC/DC adapters, enabling secret key recovery. Finally, Section 5.5 demonstrates that by adjusting the EM wave irradiation intensity, it is possible to control leakage from cryptographic devices, thereby expanding the leakage range and enhancing attack performance.

5.1 Experimental Setup

Setup I (Figure 5) is a preliminary setup for verifying the feasibility of Active EMA and for profiling the relay channel. **Setup II** (Figure 6) is reminiscent of a real environment with physical security measures applied through shielding and filtering. First, we describe the common elements of each setup.

We use a CW308T-XMEGA (NewAE Technology) target board with an Atmel 8-bit microcontroller mounted on a CW308 UFO board (NewAE Technology). The power switches on the CW308 UFO were set to **S1:Auto** and **S2:LDO**, converting the 5 V supply voltage to 3.3 V through the off-chip LDO regulator for the cryptographic core power supply. The cryptographic device's clock frequency was set to 12 MHz. The CW308 base and target boards include decoupling MLCCs placed according to the chip vendor's reference design—at the LDO input and output, as well as on the high side of the shunt resistor on the target board. We did not modify this decoupling network in our experiments. A CW1173 ChipWhisperer-Lite (NewAE Technology) [OC14] was used for AES plaintext/ciphertext transmission and reception, with only the communication cable connected to CW308 UFO to ensure power isolation. A signal generator (Rohde & Schwarz, SMA100B) was used as the transmitter, and a software-defined radio (SDR) (Ettus Research, USRP N210, UBX Board) as the receiver. Transmission and reception were synchronized by supplying a reference clock from the transmitter to the receiver, and trigger signals were input to the Pulse-Per-Second (PPS) port to control measurement timing. The ChipWhisperer-Lite, signal generator, and SDR were all connected to a PC that controls the execution of cryptographic operations, EM wave irradiation control, and waveform acquisition/analysis.

We used an unprotected AES-128⁹ implementation, and measurements were conducted targeting the 1st round. All experiments used a single fixed 128-bit AES key `2b7e151628aed2a6abf7158809cf4f3c` as per NIST SP 800-38A. The SDR sampling rate was set to 25 MSa/s, with a measurement duration of 125 μ s (3125 samples). The SDR gain was set to the maximum gain within the range where the received signal would not saturate.

Setup I: This setup verifies the basic feasibility of Active EMA and profiles the relay channel on the same device model as the target. While it does not emulate a protected environment, an attacker can use such a setup to identify favorable parameters prior to attacking a device within a security boundary. To inject EM waves from power lines connected to the cryptographic device and observe reflected waves, a coaxial cable was connected to the DC adapter. This cable was connected to a directional coupler (Mini-Circuits: ZFDC-20-5-S+) to separate incident and reflected waves (Figure 5(b)). Specifically, the transmitter was connected to the coupler's **IN Port**, and the **OUT Port** was connected to the cryptographic device's power line. EM waves propagating from **IN Port** to **OUT Port** were observed by connecting the **CPL Port** to the receiver. Power was supplied from different pins¹⁰.

⁹<https://github.com/newaetech/chipwhisperer/tree/develop/firmware/mcu/simpleserial-aes>

¹⁰Similar results were obtained when power was supplied from the DC adapter using a bias tee.

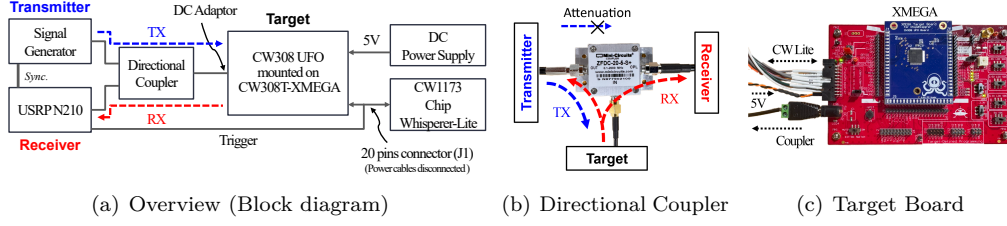


Figure 5: Setup I. (a) Block diagram of the preliminary setup for verifying the feasibility of Active EMA. (b) Reflected waves are observed separately using a coupler. (c) EM waves are irradiated from the DC adaptor of the target cryptographic device to generate reflected waves.

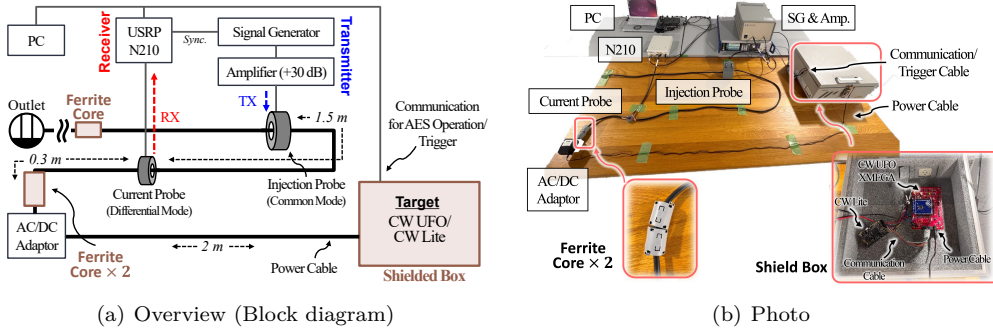


Figure 6: Setup II. Block diagram and photo of setup for verifying whether Active EMA is feasible against cryptographic devices installed within security boundaries. The cryptographic device was installed inside a shielded box. Power cables were connected to the device, and assuming an attacker can only access the primary side of the AC/DC adapter, injection probes and current probes were installed there. Additionally, the power cables were clamped with two ferrite cores.

Setup II: This setup instantiates the threat model of Section 3. The cryptographic device was installed inside a shielded box (Micronix, MY1510), with only the power and communication cables exposed, mimicking the inevitable apertures of a security boundary. To suppress leakage through these apertures, two high-frequency ferrite cores (TDK: ZCAT3035-1330) were clamped on the power cable, and a 5 V AC/DC adapter was used to further attenuate coupling toward the primary side. Due to enclosure size limits, the AC/DC adapter and ferrites were placed outside; to avoid unintentionally simplifying the scenario, the injection probe was positioned ≈ 1 m away on the AC/DC adapter. Under this configuration, the attacker cannot physically access the inside of the boundary, and passive EMA from outside the boundary fails (Section 5.4). Therefore, it was assumed that the attacker could only access the power cable exposed outside the boundary and execute Active EMA from there.

An Injection Probe (Fischer Custom Communications: F-140) was used to inject EM waves into the AC power cable, clamping the power lines to apply CM current. A Current Probe (Fischer Custom Communications: F-2000) was used to measure the reflected waves, clamping the power lines to observe DM current, thereby suppressing self-interference of EM waves injected from the transmitter. The probes were positioned 3.8 meters and 2.3 meters away from the cryptographic device, respectively. A signal amplifier (R&K: A030k102-3040R) with a 30 dB gain was used to increase irradiation intensity.

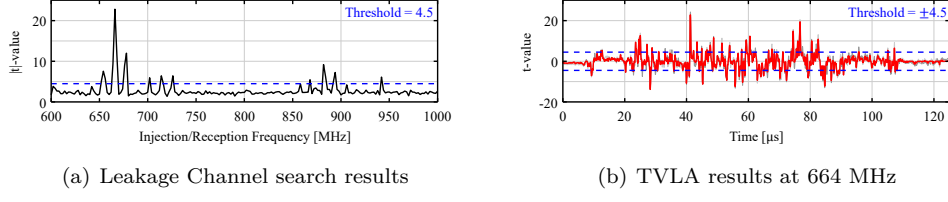


Figure 7: (a) Results of relay channel search conducted on CW308T-XMEGA mounted on CW308 UFO using **Setup I**. TVLA was performed with $2N = 100,000$ traces at each frequency. Several frequency bands exceeding the t-value threshold of 4.5 were observed, (b) with the highest value of 22.8 observed at 664 MHz.

5.2 Relay Channel Search

Assuming a scenario where attackers can obtain the same device as the target device to perform profiling, we performed a frequency characterization of the relay channel, as shown in [algorithm 1](#), using **Setup I**. Search parameters were: the intensity of signal generator $P_{TX} = 5$ dBm, search frequency range $(f_{start}, f_{end}, f_{step}) = (600 \text{ MHz}, 1000 \text{ MHz}, 2 \text{ MHz})$. Measurements were conducted with $2N = 100,000$ traces for TVLA, using fixed vs. random plaintext. Amplitude demodulation was applied as preprocessing, with HPF and LPF applied with cutoff frequencies $f_{cutoff,HPF} = 30 \text{ kHz}$, $f_{cutoff,LPF} = 3 \text{ MHz}$. Outlier removal was not performed due to the minimal influence of external noise in close-range measurements via a coaxial cable.

Figure 7 shows the relay channel leakage characterization results. In Figure 7(a), the vertical axis t-value is shown in absolute value notation to confirm the presence or absence of leakage. Frequency bands exceeding the t-value of $|t| > 4.5$ were observed, confirming the existence of frequency bands where side-channel information can be observed through Active EMA. Specifically, leakage was confirmed in the ranges of 650 MHz to 670 MHz and 850 MHz to 900 MHz, with the highest t-value (22.8) obtained at 664 MHz. Figure 7(b) shows the TVLA results with a 664 MHz irradiation and reception frequency. The horizontal axis represents the time index, the vertical axis represents the t-value, and multiple points where $|t| > 4.5$ were observed. Furthermore, to assess reproducibility, we repeated the profiling experiment (**Setup I**) on two CW308T-XMEGA target boards and two CW308 UFO base boards from the same manufacturer (four boards in total). The frequency bands with strong relay-channel characteristics, including the peak near 664 MHz, were consistent across these boards, demonstrating reproducibility and feasibility of the profiling procedure. The security-boundary experiment (**Setup II**) was performed on one board combination, yielding consistent results under shielded and filtered conditions.

5.3 Key Recovery from Outside PCB

We examined the possibility of AES key recovery through Active EMA using **Setup I**. Based on the relay channel search results from [Section 5.2](#), the irradiation and reception frequency were set to $f_c = 664 \text{ MHz}$. The intensity of the signal generator was set to $P_{TX} = 5$ dBm. As in the previous experiment, amplitude demodulation was applied as preprocessing to the acquired traces, without applying outlier removal, and HPF and LPF were applied with cutoff frequencies $f_{cutoff,HPF} = 30 \text{ kHz}$, $f_{cutoff,LPF} = 3 \text{ MHz}$. Acquired traces are shown in Figure 8. Time variations in impedance accompanying AES computational processing were observed.

Next, we collected $N = 500,000$ traces and performed CPA; all 16 bytes of the AES secret keys were recovered. CPA results for each byte are summarized in [Appendix B](#). Figure 9 shows CPA results targeting the 5th byte. Figure 9(a) shows correlation values

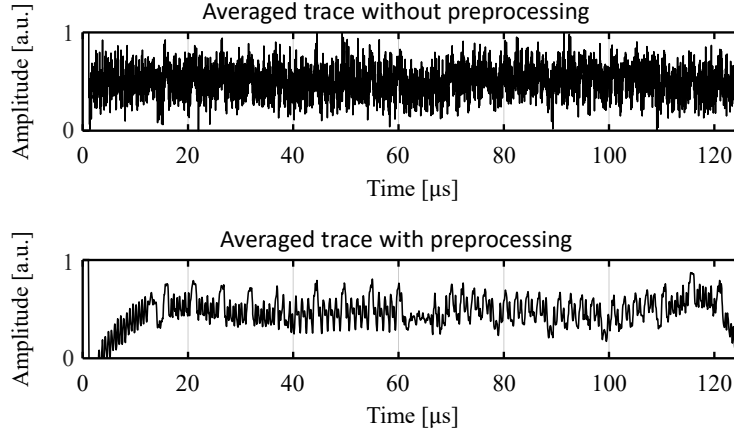


Figure 8: Traces measured with irradiation and reception frequency $f_c = 664$ MHz with **Setup I**. Averaged over 2,000 traces to clarify the waveform outline. Averaged waveforms were not used in CPA. (top) Waveform without preprocessing applied, (bottom) Waveform with HPF and LPF applied.

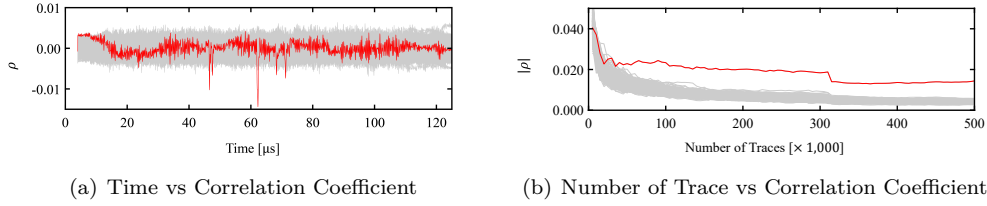


Figure 9: CPA attack results when executing Active EMA with **Setup I** (5th byte). All 16 bytes of AES secret keys were recovered. Irradiation and reception frequency is $f_c = 664$ MHz, number of traces is $N = 500,000$ traces. The red line shows the correlation value of the correct key.

for 256 key candidates against the time index, and **Figure 9(b)** shows correlation values for 256 key candidates against the number of traces used in the analysis. These results demonstrate that leakage can be induced through Active EMA and that secret keys can be recovered.

5.4 CPA Results from the Primary Side

This section demonstrates the possibility of AES key recovery through Active EMA from the primary side of AC/DC adapters under the measurement environment of **Setup II**. This demonstrates that side-channel information can be leaked outside security boundaries through the relay mechanism, crossing boundaries formed by shielded boxes. This setup presents the following challenges arising from physical security boundaries: (1) Because injection probes and current probes are used to inject and observe EM waves non-invasively and without physical contact, EM waves do not propagate efficiently. (2) EM wave propagation to cryptographic devices inside security boundaries formed by shielded boxes. (3) Signal attenuation and multipath reflections introduced by ferrite cores, AC/DC adapters, and long transmission lines. (4) Increased self-interference caused by high-intensity EM-wave irradiation. Due to these factors, the acquisition of side-channel information confined within boundaries becomes more difficult compared to **Setup I**.

Based on the profiling results from **Section 5.2**, the irradiation and reception frequencies

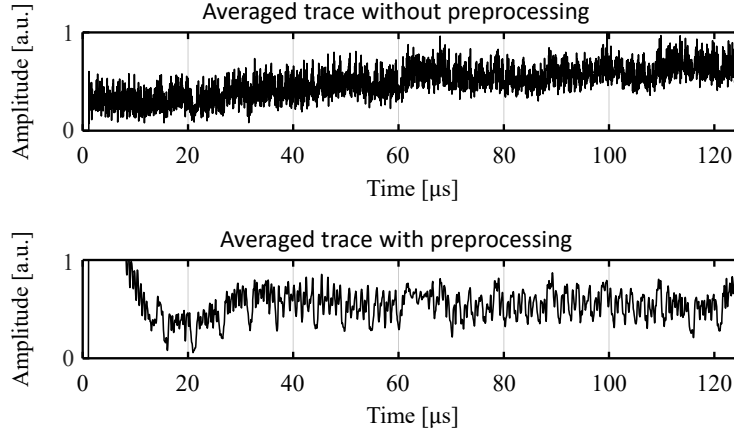


Figure 10: Traces measured from primary side with irradiation and reception frequency $f_c = 664$ MHz with **Setup II**. Averaged over 10,000 traces to clarify the waveform outline. Averaged waveforms were not used in CPA. (top) Waveform without preprocessing applied, (bottom) Waveform with outlier removal, HPF, and LPF applied.

were set to $f_c = 664$ MHz. The intensity of the signal generator output was set to $P_{TX} = 17$ dBm and amplified to approximately 47 dBm by the amplifier. The acquired traces were preprocessed using amplitude demodulation, and traces containing surge noise were removed by outlier removal with threshold $k = 1.2$. Afterwards, HPF and LPF with cutoff frequencies $f_{\text{cutoff,HPF}} = 30$ kHz, $f_{\text{cutoff,LPF}} = 3$ MHz were applied. A total of $N = 5,000,000$ traces were collected.

To establish the effectiveness of our active attack, we start by performing a passive attack under the same measurement conditions. In this scenario, we do not irradiate EM waves, but only passively capture EM emanations (with the receiver gain maximized) from the device. In this scenario, we were unable to recover any AES key bytes. For example, Figure 11 shows attack results targeting the 5th byte. Notably, even when the number of traces used for the attack is increased to 5,000,000, the correlation value of the correct key indicates that there is no prospect of extracting the key (Figure 11(b)). This shows that the side-channel information is confined within the physical security boundaries achieved by measures such as ferrite cores and a shielded enclosure. Detailed results are shown in Appendix C.

Next, we conducted our proposed active attack. 5.42% (270,873 traces) of all traces were detected as abnormal traces by outlier removal. In subsequent analysis, the remaining 94.58% (4,729,127) of traces were used. Abnormal traces showed surge-like noise with rapid amplitude changes at specific timings, likely caused by unintended modulation from surrounding electronic device operations or external noise superimposition on power lines. Figure 10 shows traces before and after the preprocessing. Time variations in impedance accompanying AES computational processing were observed outside security boundaries through the relay channel.

We performed CPA using $N = 5,000,000$ traces¹¹, and were able to recover all 16 bytes of the AES secret key. CPA attack results for each byte are shown in Appendix C. Figure 12 shows CPA attack results targeting the 5th byte. Figure 12(a) shows correlation values of 256 key candidates against the time index, and Figure 12(b) shows correlation values of 256 key candidates against the number of traces. In contrast to the passive EMA

¹¹Here, $N = 5,000,000$ traces is the total number of observations, including abnormal traces removed by outlier removal. For fair evaluation of attack performance, we use “number of traces observed” by the attacker as the evaluation criterion. Actually, $N = 4,729,127$ traces were used for CPA.

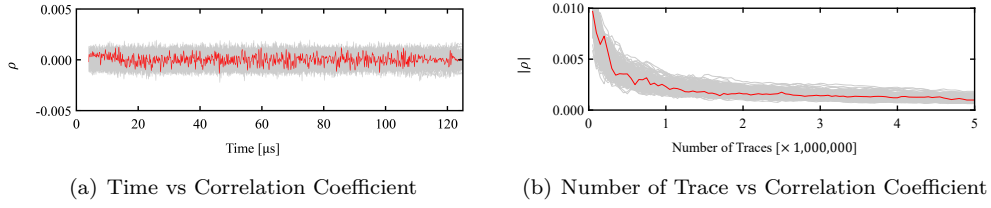


Figure 11: CPA attack results when executing conventional passive EMA in **Setup II** (5th byte). We cannot extract even one subkey using 5,000,000 traces. Therefore, physical security measures could suppress leakage across the boundary.

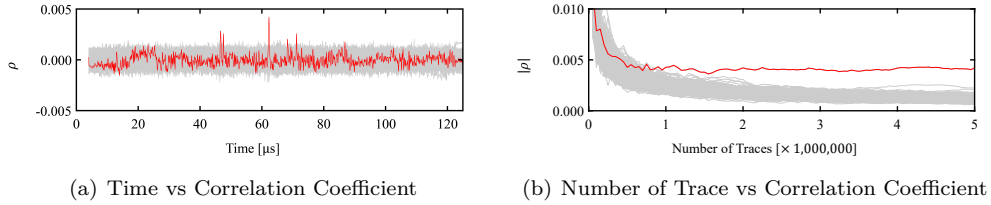


Figure 12: CPA attack results when executing Active EMA in **Setup II** (5th byte). We recovered all 16 bytes of the AES secret key. Irradiation and reception frequency were $f_c = 664$ MHz, the number of traces was 5,000,000 traces. The red line shows the correlation value of the correct key.

results, the correlation value of the correct subkey is highest when using approximately 1,000,000 traces or more for CPA. Although the measurement took several days, key recovery was achieved without being affected by temperature changes or environmental changes, even in long-term measurements.

5.5 Controllability of Leakage Intensity

The experiments up to the previous section demonstrated that Active EMA can leak side-channel information beyond security boundaries. This section verifies another important characteristic of Active EMA: controllability of leakage intensity. In conventional passive measurement, the intensity of leaking EM waves depends on the implementation and design of cryptographic devices and cannot be controlled by attackers. In contrast, Active EMA enables attackers to actively control leakage by adjusting the intensity and frequency of irradiated EM waves.

In this experiment, we used **Setup I**, which uses coaxial cables and couplers to accurately separate incident and reflected waves, allowing pure evaluation of reflected wave effects with less external noise influence for high reproducibility and more quantitative evaluation of Active EMA attack accuracy. In the experiment, the signal generator irradiation intensity was varied as $P_{TX} = 0$ dBm, 5 dBm, 10 dBm, 12 dBm, and CPA attacks were performed under each condition. The receiver gain was adjusted to avoid saturation at each irradiation intensity, with gains set to 17.0, 12.5, 7.0, and 5.0, respectively, for each irradiation intensity within the USRP N210 gain setting range (0~32). This systematic gain adjustment ensures optimal measurement conditions at each irradiation intensity for fair comparison.

Figure 13 shows key recovery rates when varying irradiation intensity. The horizontal axis represents the number of traces, and the vertical axis represents the number of correctly recovered key bytes (maximum 16 bytes). Results confirmed that key estimation accuracy significantly improved with increased irradiation intensity. Specifically, comparing the

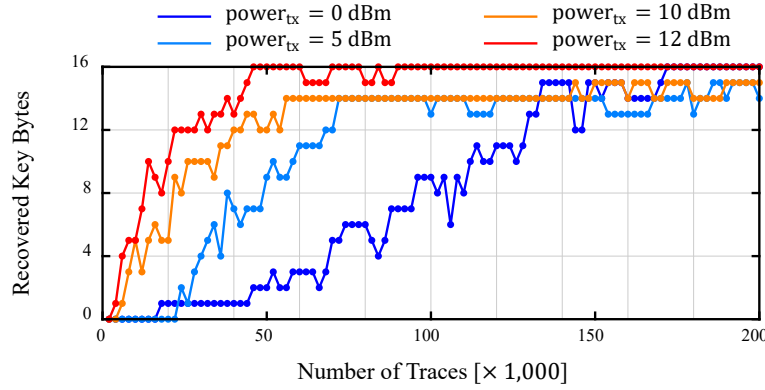


Figure 13: Number of traces required to recover the secret key for multiple irradiation intensities. An increasing irradiation intensity enables key recovery using fewer traces. This is because higher irradiation intensity improves reflected wave intensity, enabling acquisition of higher signal-to-noise ratio traces.

number of traces required to recover full 16 bytes of keys, approximately 50,000 traces were needed at 12 dBm irradiation intensity, while approximately 175,000 traces were required at 0 dBm, a 3.5-fold difference.

In contrast to traditional passive measurements, Active EMA allows control of the leaked signal intensity. Attackers can obtain higher signal-to-noise ratio traces by increasing the radiation intensity. This lets them extend the distance over which the attack can be performed [KFKH23, PLHD25, KFH19]. We use this control feature when we discuss extending attack distance by increasing the irradiation intensity in Section 6.2. On the other hand, practical constraints also exist on irradiation intensity. Excessive irradiation may cause interference or disturbance to cryptographic devices, so attackers need to set the irradiation intensity within the range where the target devices maintain regular operation. In this experimental environment, no communication errors or cryptographic operation errors were observed even when increasing irradiation intensity to 15 dBm in **Setup I**. Still, this upper limit is expected to vary depending on device implementation and environmental conditions.

In Section 5, the experimental results demonstrated that even against cryptographic devices where security boundaries are formed by physical security measures such as shielded enclosures and filters, making attacks difficult with conventional passive measurement, Active EMA based on the relay mechanism can actively leak side-channel information beyond security boundaries, enabling secret key recovery. Furthermore, as shown in Section 5.5, Active EMA enables attackers to intentionally control the leakage intensity by adjusting the irradiation intensity, allowing for the optimization of attack efficiency. This controllability and reproducibility demonstrate that Active EMA is not merely a theoretical threat, but an attack method that can be systematically executed in real-world environments, emphasizing its importance as a new threat model to consider in physical security evaluations.

6 Countermeasures and Future Challenges

In this section, we discuss countermeasures against Active EMA (Section 6.1) and then outline future research directions, including spatial radiative attacks, extended attack distance, and improvements in measurement and analysis performance (Section 6.2).

6.1 Countermeasures against Active EMA

This section discusses physical security measures against Active EMA. Active EMA is established through a relay mechanism, and since impedance variations of nonlinear elements in the First Relay Channel are difficult to suppress, countermeasures against active EM measurement in the Second Relay Channel seem more promising. As shown in [Section 5.4](#), physical security measures designed to suppress leakage can be bypassed. On the other hand, monitoring systems to detect irradiation of EM waves might be effective.

EM fault sensors implement voltage and EM field sensors within cryptographic ICs to detect external EM waves [[MNH⁺16](#), [FHB⁺18](#), [ZDT⁺14](#)]. They can detect continuous waves in the hundreds of MHz band and high-intensity EM waves of 47 dBm used in Active EMA, preventing information leakage by stopping cryptographic processing or switching to dummy operations. However, technical extensions from conventional EM pulse wave assumptions to continuous wave support are necessary.

Intentional Electromagnetic Interference (IEMI) detectors have broadband monitoring and direction-of-arrival estimation capabilities [[GT04](#), [BRS22](#), [KRS25](#)], and can detect Active EMA-specific patterns such as frequency searches from 600 MHz to 1 GHz and long-duration continuous irradiation. They can learn environmental noise levels and identify low-power continuous waves that do not damage equipment.

Tamper detection using radio frequencies continuously monitors the EM environment within protected areas [[STZP22](#), [TKS⁺25](#)], detecting environmental changes and changes in reflection characteristics caused by injected EM waves. Combination with on-chip impedance measurement [[MST22](#)] can also be effective. By combining these countermeasures, a multi-layered defense that effectively detects and prevents channel creation by active EM measurement can be established.

6.2 Future Research Directions

While our experiments demonstrate Active EMA through conductive paths, extending to spatial radiative paths is a natural next step. In through-wall or through-enclosure scenarios, barrier attenuation and multipath interference are key challenges; they may be mitigated by appropriate frequency selection for penetration and by using directional antennas to suppress unintentional reflections. Prior studies also demonstrated that power and interface cables outside the boundary can act as unintentional antennas that receive and re-radiate injected EM waves, enabling radiative leakage channels [[KFKH23](#), [PLHD25](#)].

Extending attack distance is also a significant challenge. Based on the relationship between irradiation intensity and key recovery rate shown in [Section 5.5](#), attacks from distances of tens of meters or from adjacent rooms may be possible. However, solving self-interference problems due to increased distance requires complete separation of transmission and reception systems.

Performance improvement is also a challenge. Our attack required 10^3 to 10^5 times more traces than conventional power analysis. Adoption of multi-band measurements [[GPNS25](#)] and high-performance receivers is expected to improve the overall measurement quality. For the analysis aspects, using both the amplitude and phase information [[AHC⁺24](#)] and applying deep learning [[MPP16](#)] techniques are also expected to improve the results. Similar to the development process of Screaming Channels [[CPM⁺18](#), [CFS20](#), [GPNS25](#)], systematic development of measurement and analysis systems for Active EMA will improve the overall results and will further increase our understanding of an attacker's capabilities.

The decoupling network affects Active EMA in two opposing ways. First, lowering the PDN impedance reduces ground bounce, which disfavors both passive and active measurements. Second, Class II MLCCs can form new Relay Points because their effective capacitance depends on DC/AC bias, which can change the modulation depth and the most favorable carrier band. Disentangling these effects requires a board redesign and

DC/AC-bias characterization as discussed in [NWM⁺11], which we leave for future work. Our results in Section 5 indicate that, despite decoupling, Active EMA remains effective under the default configuration.

7 Conclusion

This research demonstrates that secret key recovery is possible through active EM side-channel measurements (Active EMA). Our technique uses a relay mechanism that is effective against cryptographic devices with physical security boundaries. The proposed method is based on the principle that ground bounce in cryptographic ICs affects the impedance of peripheral nonlinear elements (First Relay Channel), and EM waves irradiated by attackers are modulated by these impedance variations and leak outside the boundary as reflected waves (Second Relay Channel).

To demonstrate the effectiveness of our technique, we performed an active EM side-channel attack from the primary side of an AC/DC adapter. This was achieved while the target device was placed in a shielded enclosure and protected by ferrite core filters. We showed that attacks can be realized through active EM measurement in environments where SCA was infeasible with conventional passive measurements. Furthermore, we demonstrated that attackers can control the leakage intensity by adjusting the irradiation intensity, demonstrating that Active EMA is a practical attack method that can be systematically executed.

Future challenges include realizing attacks through spatial emission paths, extending to attacks from long distances, and improving performance through multi-band measurement and deep learning-based analysis.

As countermeasures, a multi-layered defense is effective, including the implementation of EM fault sensors and IEMI detectors with continuous wave detection capabilities, and prevention of relay channel generation by monitoring impedance variations.

This research suggests the necessity of considering active EM measurement in addition to conventional passive threat models in physical security evaluation. Particularly in mission-critical systems such as ATMs, infrastructure control systems, and HSMs, countermeasures against new attack paths through inevitable apertures in security boundaries are needed. Active EMA is not merely a theoretical threat, but an attack method that can be executed in real-world environments, calling for the establishment of new security evaluation standards.

Acknowledgments

This work was supported by JSPS KAKENHI (Grant Numbers JP23H00472 and JP23K28087), the JST FOREST Program (Grant Number JPMJFR206L), the JSPS Grant-in-Aid for JSPS Fellows (Grant Number JP23KJ1593), the Flemish Government through the Cybersecurity Research Program (Grant Number VOEWICS02), and the European Commission through the Horizon 2020 research and innovation program under grant agreement Belfort ERC Advanced Grant 101020005 695305. This work was also conducted as part of the JSPS Overseas Challenge Program for Young Researchers, and the authors are grateful for its support.

References

- [AARR03] Dakshi Agrawal, Bruce Archambeault, Josyula R Rao, and Pankaj Rohatgi. The EM side—channel(s). In *Cryptographic Hardware and Embedded Systems*

- *CHES 2002*, Lecture notes in computer science, pages 29–45. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
- [AGR24] Md Sadik Awal, Buddhipriya Gayanath, and Md Tauhidur Rahman. Impedance vs. power side-channel vulnerabilities: A comparative study. *arXiv preprint arXiv:2405.06242*, 2024.
- [AHC⁺24] Pierre Ayoub, Aurélien Hernandez, Romain Cayre, Aurélien Francillon, and Clémentine Maurice. PhaseSCA: Exploiting phase-modulated emanations in side channels. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2025(1):392–419, dec 2024.
- [AJCZ19] Sinan Adibelli, Prateek Juyal, Chia-Lin Cheng, and Alenka Zajic. Terahertz near-field focusing using a 3-D printed cassegrain configuration for backscattered side-channel detection. *IEEE Trans. Antennas Propag.*, 67(10):6627–6638, October 2019.
- [BRS22] Ajay Kumar Bellamkonda, Patnam Hanumantha Rao, and Saurabh Saxena. Intentional electromagnetic interference reception in 0.5–2.0 GHz. *IEEE Trans. Electromagn. Compat.*, 64(6):2163–2169, December 2022.
- [CFS20] Giovanni Camurati, Aurélien Francillon, and François-Xavier Standaert. Understanding screaming channels: From a detailed analysis to improved attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 358–401, jun 2020.
- [Com22] Common Criteria Recognition Arrangement. Common criteria for information technology security evaluation – part 2: Security functional components (CC:2022 revision 1). Technical report, Common Criteria Recognition Arrangement (CCRA), November 2022.
- [CPM⁺18] Giovanni Camurati, Sebastian Poeplau, Marius Muench, Tom Hayes, and Aurélien Francillon. Screaming channels: When electromagnetic side channels meet radio transceivers. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS ’18*, pages 163–177, New York, NY, USA, 2018. ACM.
- [dMSV20] Pieterjan de Meulemeester, Bart Scheers, and Guy A E Vandenbosch. Eavesdropping a (ultra-)high-definition video display from an 80 meter distance under realistic circumstances. In *2020 IEEE International Symposium on Electromagnetic Compatibility & Signal/Power Integrity (EMCSI)*, pages 517–522. IEEE, July 2020.
- [FHB⁺18] Daisuke Fujimoto, Yu-Ichi Hayashi, Arthur Beckers, Josep Balasch, Benedikt Gierlichs, and Ingrid Verbauwhede. Detection of IEMI fault injection using voltage monitor constructed with fully digital circuit. In *2018 IEEE International Symposium on Electromagnetic Compatibility and 2018 IEEE Asia-Pacific Symposium on Electromagnetic Compatibility (EMC/APEMC)*. IEEE, May 2018.
- [GJJR11] Gilbert Goodwill, Benjamin Jun, Josh Jaffe, and Pankaj Rohatgi. A testing methodology for side-channel resistance validation. In *NIST Non-Invasive Attack Testing (NIAT) Workshop*. csrc.nist.gov, 2011.
- [GMO01] Karine Gandolfi, Christophe Mourtél, and Francis Olivier. Electromagnetic analysis: Concrete results. In *Cryptographic Hardware and Embedded Systems – CHES 2001*, volume 2162 of *Lecture notes in computer science*, pages 251–261. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

- [GPNS25] Jeremy Guillaume, Maxime Pelcat, Amor Nafkha, and Rubén Salvador. Multi-screaming-channel attacks: Frequency diversity for enhanced attacks. *arXiv [cs.CR]*, apr 2025.
- [GPT15] Daniel Genkin, Itamar Pipman, and Eran Tromer. Get your hands off my laptop: physical side-channel key-extraction attacks on PCs: Extended version. *J. Cryptogr. Eng.*, 5(2):95–112, June 2015.
- [GS22] Jan Sebastian Götte and Björn Scheuermann. Can’t touch this: Inertial hsm’s thwart advanced physical attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 69–93, 2022.
- [GT04] D V Giri and F M Tesche. Classification of intentional electromagnetic environments (IEME). *IEEE Trans. Electromagn. Compat.*, 46(3):322–328, August 2004.
- [HB25] Vedad Hadžic and Roderick Bloem. Efficient and composable masked aes s-box designs using optimized inverters. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2025(1):656–683, 2025.
- [HHM⁺13] Yu-Ichi Hayashi, Naofumi Homma, Takaaki Mizuki, Takafumi Aoki, Hideaki Sone, Laurent Sauvage, and Jean-Luc Danger. Analysis of electromagnetic information leakage from cryptographic devices with different physical structures. *IEEE Trans. Electromagn. Compat.*, 55(3):571–580, June 2013.
- [HMF07] Michael Hutter, Stefan Mangard, and Martin Feldhofer. Power and EM attacks on passive 13.56 MHz RFID devices. In *Cryptographic Hardware and Embedded Systems - CHES 2007*, Lecture notes in computer science, pages 320–333. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
- [HSK⁺10] Yu-Ichi Hayashi, Takeshi Sugawara, Yoshiki Kayano, Naofumi Homma, Takaaki Mizuki, Akashi Satoh, Takafumi Aoki, Shigeki Minegishi, Hideaki Sone, and Hiroshi Inoue. Information leakage from cryptographic hardware via common-mode current. In *2010 IEEE International Symposium on Electromagnetic Compatibility*, pages 109–114. IEEE, July 2010.
- [IMK⁺25] Hayato Ide, Shohei Matsumoto, Taiki Kitazawa, Shugo Kaji, Daisuke Fujimoto, Takashi Kasuga, and Yuichi Hayashi. Amplifying electromagnetic leakage by hardware trojans through cable geometry manipulation. In *2025 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 261–267. IEEE, may 2025.
- [Int15] International Telecommunication Union-Telecommunication Standardization Sector (ITU-T). K.115: Mitigation methods against electromagnetic security threats, 2015.
- [ION⁺18] Vincent Immler, Johannes Obermaier, Kuan Kuan Ng, Fei Xiang Ke, Jinyu Lee, Yak Peng Lim, Wei Koon Oh, Keng Hoong Wee, and Georg Sigl. Secure physical enclosures from covers with tamper-resistance. *TCHES*, pages 51–96, 2018.
- [KAK⁺22] Taiki Kitazawa, Kimihiro Arai, Youngwoo Kim, Daisuke Fujimoto, and Yuichi Hayashi. A novel remote visualization of screen images against high-resolution display with divided screens focusing on the difference of transfer function of multiple emanations. *IEEE Trans. Electromagn. Compat.*, 64(6):1941–1948, December 2022.

- [KFH19] Masahiro Kinugawa, Daisuke Fujimoto, and Yuichi Hayashi. Electromagnetic information extortion from electronic devices using interceptor and its countermeasure. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 62–90, aug 2019.
- [KFH24] Taiki Kitazawa, Daisuke Fujimoto, and Yuichi Hayashi. Fundamental study on simple power analysis using backscattering from switching regulators. In *2024 International Symposium on Electromagnetic Compatibility – EMC Europe*, pages 22–26. IEEE, sep 2024.
- [KFKH23] Shugo Kaji, Daisuke Fujimoto, Masahiro Kinugawa, and Yuichi Hayashi. Echo TEMPEST: EM information leakage induced by IEMI for electronic devices. *IEEE Trans. Electromagn. Compat.*, 65(3):655–666, June 2023.
- [KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology — CRYPTO’ 99*, Lecture notes in computer science, pages 388–397. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.
- [KOP09] Timo Kasper, David Oswald, and Christof Paar. EM side-channel attacks on commercial contactless smartcards using low-cost equipment. In *Information Security Applications*, Lecture notes in computer science, pages 79–93. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
- [KRS25] Pallepogu Prasanna Kumar, Patnam Hanumantha Rao, and Saurabh Saxena. Design and analysis of 0.5–2.0 GHz RF frontend subsystems for IEMI applications. *IEEE Trans. Electromagn. Compat.*, 67(1):237–246, February 2025.
- [Kuh05] Markus G Kuhn. Security limits for compromising emanations. In *Cryptographic Hardware and Embedded Systems – CHES 2005*, Lecture notes in computer science, pages 265–279. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.
- [LDMPT15] J Longo, E De Mulder, D Page, and M Tunstall. SoC it to EM: ElectroMagnetic side-channel attacks on a complex system-on-chip. In *Lecture Notes in Computer Science*, Lecture notes in computer science, pages 620–640. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.
- [MMT23] Saleh Khalaj Monfared, Tahoura Mosavirik, and Shahin Tajik. Leakyohm: Secret bits extraction using impedance analysis. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 1675–1689, 2023.
- [MNH⁺16] Noriyuki Miura, Zakaria Najm, Wei He, Shivam Bhasin, Xuan Thuy Ngo, Makoto Nagata, and Jean-Luc Danger. PLL to the rescue: a novel EM fault countermeasure. In *Proceedings of the 53rd Annual Design Automation Conference*, New York, NY, USA, jun 2016. ACM.
- [MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks: Revealing the secrets of smart cards*. Springer, New York, NY, 2007 edition, mar 2007.
- [MPP16] Housseem Maghrebi, Thibault Portigliatti, and Emmanuel Prouff. Breaking cryptographic implementations using deep learning techniques. In *Security, Privacy, and Applied Cryptography Engineering*, Lecture notes in computer science, pages 3–26. Springer International Publishing, Cham, 2016.

- [MST22] Tahoura Mosavirik, Patrick Schaumont, and Shahin Tajik. ImpedanceVerif: On-chip impedance sensing for system-level tampering detection. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 301–325, nov 2022.
- [Nat01] National Institute of Standards and Technology. Security requirements for cryptographic modules (FIPS PUB 140-2). Technical report, U.S. Department of Commerce, National Institute of Standards and Technology, Gaithersburg, MD 20899-8900, May 2001.
- [Nat19] National Institute of Standards and Technology. Security Requirements for Cryptographic Modules (FIPS PUB 140-3). Technical Report FIPS PUB 140-3, U.S. Department of Commerce, National Institute of Standards and Technology, Gaithersburg, MD 20899-8900, March 2019.
- [NWM⁺11] Istvan Novak, Kendrick Barry Williams, Jason R. Miller, Gustavo Blando, and Nathaniel Shannon. Dc and ac bias dependence of capacitors including temperature dependence. In *Proceedings of DesignCon East 2011*, Santa Clara, CA, 2011. DesignCon.
- [OC14] Colin O’Flynn and Zhizhang Chen. ChipWhisperer: An open-source platform for hardware embedded security research. In *Constructive Side-Channel Analysis and Secure Design*, Lecture notes in computer science, pages 243–260. Springer International Publishing, Cham, 2014.
- [O’F24] Colin O’Flynn. Phase modulation side channels: Jittery JTAG for on-chip voltage measurements. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2024(4):382–424, sep 2024.
- [OI18] Johannes Obermaier and Vincent Immler. The past, present, and future of physical security enclosures: From battery-backed monitoring to PUF-based inherent security and beyond. *J. Hardw. Syst. Secur.*, 2(4):289–296, December 2018.
- [PLHD25] Lina Pu, Yu Luo, Song Han, and Junming Diao. Your cable, my antenna: Eavesdropping serial communication via backscatter signals. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 3710–3726, 2025.
- [Poz21] David M Pozar. *Microwave engineering: theory and techniques*. John Wiley & Sons, New York, NY, 2021.
- [PSQ07] Eric Peeters, François-Xavier Standaert, and Jean-Jacques Quisquater. Power and electromagnetic analysis: Improved model, consequences and comparisons. *Integration (Amst.)*, 40(1):52–60, 1 2007.
- [PSS22] Clayton R Paul, Robert C Scully, and Mark A Steffka. *Introduction to electromagnetic compatibility*. John Wiley & Sons, Nashville, TN, 3 edition, 7 November 2022.
- [QS01] Jean-Jacques Quisquater and David Samyde. ElectroMagnetic analysis (EMA): Measures and counter-measures for smart cards. In *Smart Card Programming and Security*, Lecture notes in computer science, pages 200–210. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.
- [SB15] B. G. Streetman and Sanjay Banerjee. *Solid State Electronic Devices*. Pearson, Upper Saddle River, NJ, 7 edition, 2015.

- [SBO⁺15] Daehyun Strobel, Florian Bache, David Oswald, Falk Schellenberg, and Christof Paar. Scandalee: a side-channel-based disassembler using local electromagnetic emanations. pages 139–144, 9 March 2015.
- [SE07] M Swaminathan and E Engin. Power integrity modeling and design for semiconductors and systems. 2007.
- [SHH⁺09] Takeshi Sugawara, Yu-Ichi Hayashi, Naofumi Homma, Takaaki Mizuki, Takafumi Aoki, Hideaki Sone, and Akashi Satoh. Mechanism behind information leakage in electromagnetic analysis of cryptographic modules. In *Information Security Applications*, Lecture notes in computer science, pages 66–78. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
- [SJL⁺21] Yin Sun, Junho Joo, Jongjoo Lee, Keeyoung Kwon, and Chulsoon Hwang. Analysis of power supply induced jitter of high speed output buffer with on-die low-dropout voltage regulator. In *2021 IEEE International Joint EMC/SI/PI and EMC Europe Symposium*, pages 318–322. IEEE, 26 July 2021.
- [SLT16] Sami Saab, Andrew Leiserson, and Michael Tunstall. Key extraction from the primary side of a switched-mode power supply. In *2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST)*, pages 1–7. IEEE, December 2016.
- [SMTG23] Kai Schoos, Sergej Meschkov, Mehdi B Tahoori, and Dennis R E Gnad. JitSCA: Jitter-based side-channel analysis in picoscale resolution. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 294–320, jun 2023.
- [SSS⁺13] Takeshi Sugawara, Daisuke Suzuki, Minoru Saeki, Mitsuru Shiozaki, and Takeshi Fujino. On measurable side-channel leaks inside ASIC design primitives. In *Cryptographic Hardware and Embedded Systems - CHES 2013*, Lecture notes in computer science, pages 159–178. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
- [STZP22] Paul Staat, Johannes Tobisch, Christian Zenger, and Christof Paar. Anti-tamper radio: System-level tamper detection for computing systems. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1722–1736. IEEE, May 2022.
- [TG07] Stefan Tillich and Johann Großschädl. Power analysis resistant aes implementation with instruction set extensions. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 303–319. Springer, 2007.
- [TKS⁺25] Maryam Shaygan Tabar, Johannes Kortz, Paul Staat, Harald Elders-Boll, Christof Paar, and Christian Zenger. Anti-tamper radio meets reconfigurable intelligent surface for system-level tamper detection. *arXiv [cs.CR]*, mar 2025.
- [vE85] Wim van Eck. Electromagnetic radiation from video display units: An eavesdropping risk? *Comput. Secur.*, 4(4):269–286, December 1985.
- [Wei00] Steve H Weingart. Physical security devices for computer subsystems: A survey of attacks and defenses. In *Cryptographic Hardware and Embedded Systems — CHES 2000*, Lecture notes in computer science, pages 302–317. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

- [WPZ22] Frank T Werner, Milos Prvulovic, and Alenka Zajić. Detection of recycled ics using backscattering side-channel analysis. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 30(9):1244–1255, 2022.
- [ZDT⁺14] Loic Zussa, Amine Dehbaoui, Karim Tobich, Jean-Max Dutertre, Philippe Maurine, Ludovic Guillaume-Sage, Jessy Clediere, and Assia Tria. Efficiency of a glitch detector against electromagnetic fault injection. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*, New Jersey, 2014. IEEE Conference Publications.

Appendix

A Outlier Removal

Using the variance value of each trace as an indicator, anomaly detection and removal of traces are performed using the IQR. Let traces be $L_i = (l_{i1}, l_{i2}, \dots, l_{iN})$, $i = 1, 2, \dots, M$, and the variance value of each trace be σ_i^2 . The quartiles and their IQR are calculated as follows, and outlier thresholds are set.

$$Q_1 = \text{Percentile}_{25}(\{\sigma_i^2\}_{i=1}^M), \quad Q_3 = \text{Percentile}_{75}(\{\sigma_i^2\}_{i=1}^M), \quad \text{IQR} = Q_3 - Q_1. \quad (7)$$

$$\text{Lower}_{\text{th}} = Q_1 - k \cdot \text{IQR}, \quad \text{Upper}_{\text{th}} = Q_3 + k \cdot \text{IQR}, \quad (8)$$

where k is an arbitrary constant. Using these thresholds, only normal traces L_{normal} are extracted.

$$L_{\text{normal}} = \{X_i \mid i \in \mathcal{N}\}, \quad \mathcal{N} = \{i \mid \text{Lower}_{\text{th}} \leq \sigma_i^2 \leq \text{Upper}_{\text{th}}\} \quad (9)$$

B CPA from outside PCB (Setup I)

Figure 14 shows the analysis results for all subbytes in the CPA attack through Active EMA in Section 5.3. The display area of results is set to 300 indices (12 μs) before and after the Point of Interest (POI) calculated by performing Known Answer Test (KAT) using Active EMA traces executed with $P_{\text{TX}} = 12 \text{ dBm}$ in Section 5.5.

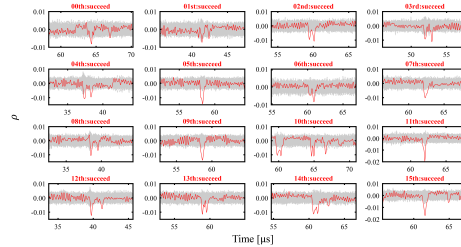


Figure 14: CPA results when executing Active EMA in **Setup I**. Irradiation and reception frequency is $f_c = 664 \text{ MHz}$, number of traces is $N = 500 \text{ k}$. The red line shows the correlation value of the correct key.

C CPA from Primary Side (Setup II)

Figure 15(a) shows the analysis results for all subbytes in the CPA attack through Active EMA in Section 5.4. The display area of results is set to 300 indices (12 μ s) before and after the PoI calculated by performing KAT using Active EMA traces executed with $P_{TX} = 12$ dBm in Section 5.5.

Additionally, to demonstrate that the above CPA attack results were achieved through active EM measurement, EM wave measurement was conducted with the signal generator irradiation stopped and SDR gain maximized. After measuring 5,000,000 traces, the same signal processing as in Section 5.4 was applied, and a CPA attack was performed. As a result, passive measurement could not extract even one byte of the secret key. Figure 15(b) shows the analysis results when the CPA attack is performed.

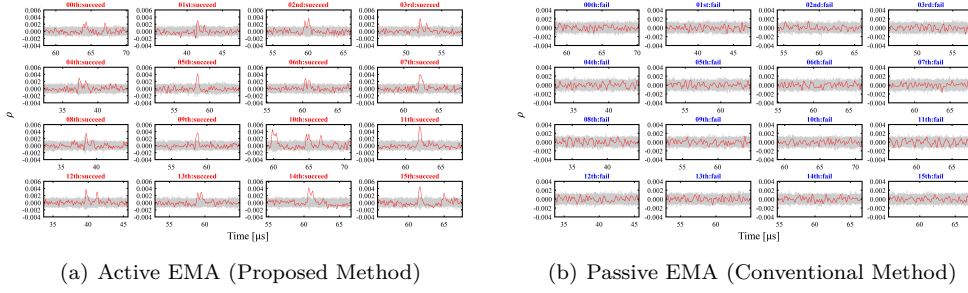


Figure 15: CPA attack results when executing EMA in **Setup II**. Frequency is $f_c = 664$ MHz, number of traces is $N = 5,000,000$ traces. The red line shows the correlation value of the correct key.

Improving the Selection Rule of Correlation Attacks for Remote Power Analysis

Oriol Farràs¹, Vincent Grosso², Miquel Guiot¹ and Carlos Andres Lara-Nino¹

¹ Universitat Rovira i Virgili, Tarragona, Spain

² CNRS/Laboratoire Hubert Curien, Saint-Étienne, France

Abstract. Remote power analysis is a novel threat to information systems. Under this attack model, the adversary does not require direct physical access to the platform or specialized sensing equipment. Most of the literature in this field deals with advanced acquisition methods and adversarial models. In contrast, side-channel analysis techniques for remote attacks have not been sufficiently explored. We bridge this gap by taking a look at the characteristics of the data recovered from remote power analysis. We use these insights to propose a novel selection rule for correlation-based attacks that boosts success confidence. This improvement comes from the observation that the samples in a power trace are not independent. We show that adjacent samples can also provide useful information by proposing a post-processing step that capitalizes on these additional leakages. In contrast to previous work, the proposed technique does not rely on the selection of points of interest within the power traces. We further investigate the characteristics of “remote” power traces and their effect on the proposed selection rule through experiments with real (TDC, ChipWhisperer) and synthetic data sets. To assess the advantage of the proposed improvement, we also introduce novel performance metrics that divert from known-key evaluation techniques.

Keywords: Attack performance evaluation · Correlation power analysis · Hypothesis discrimination · Remote power analysis · Side channel attacks · Statistical attacks

1 Introduction

Side channel analysis attacks (SCA) exploit the dependencies between the logical and physical states of a computing system. The “logical state” of a circuit is constituted by program data, constants, inputs, outputs, register values, etc. The physical state consists of characteristics that can be monitored through an acquisition process. These include phenomena such as sound, voltage, current, power, temperature, and electromagnetism. By studying fluctuations in the physical state, an adversary might gain insight into its logical state. And potentially into its sensitive information [CKN01, MOP08].

The usual adversarial models in SCAs confer the adversary with several advantages. We assume that the attacker can query a “physical” oracle an unrestricted number of times and that they have the capability to register, store and operate over these responses. This physical oracle $\mathcal{E}[k]$ (depicted in Fig. 1) contains some secret information (k) whose recovery is the ultimate goal of the adversary. The rule of the game is that the adversary cannot directly request k , but can provide y queries (m) and the oracle will respond with a cryptographic output (c) plus additional information (ℓ). If the algorithm $\mathcal{E}$ is cryptographically sound, we can assume that it is computationally difficult to infer k from $\{m, c\}$. However, due to the physical nature of the oracle, some information “is leaking” from k through ℓ . In practical terms, the physical oracle is the device that performs sensitive operations over m and produces c . This process involves internal secrets k , which

can be some data or the operation itself. The additional information ℓ revealed by the oracle is embedded in the physical magnitudes associated with the device: current draw, power dissipation, heat dissipation, electromagnetic emanation, etc. Therefore, the adversary must have some control over the device and the sensing capabilities to retrieve the second oracle output (ℓ).

However, the physical transients produced by computing platforms are challenging to observe. Their fluctuation rate is, in most cases, proportional to the operating frequency of the device. Additionally, their magnitude is quite small and easily obfuscated by noise. Historically, this has been understood to pose two main challenges for adversaries. First, they would require sophisticated sensing equipment (high monetary costs). And second, they would need physical access to the device in order to prepare a proper experimental setup [MOP08]. However, advances in the development of sensing technologies now allow us to instantiate internal sensors in ordinary computing systems [ZS18, GDTL21]. In fact, it is now possible to leverage hardware components within the device to retrieve measurements that correlate with the electrical activity of the circuit. These sensors can be introduced during the manufacturing step or by software means, enabling an attacker to mount SCAs remotely. This paradigm change can lead to the introduction of more practical adversarial scenarios and the definition of *remote power analysis* (RPA).

Internal sensors such as time-to-digital converters (TDC) and ring oscillators are not capable of capturing physical phenomena *per se*. Instead, they are designed to register logical values which may fluctuate as a function of the propagation delay within the circuit. That is, when the system is in repose, their output is more-or-less constant. However, when some computing operation is performed, the charge/discharge of components within the circuit affects its current draw. In turn, this produces an instantaneous power drop, which leaves the system in a transitory state. During this phase, anything connected to the same power source is bound to be affected. The most notable outcome of this process is that the speed of the stimuli propagating throughout the platform is reduced. Thus, the logical values registered by the sensor circuit are expected to divert from their steady states. This phenomenon is demonstrated in [FTBLN24]. Multiple quantification and decoding strategies can then be applied to obtain estimations of the fluctuation in the propagation delay. Ideally, there would be some linearity in the sensor output. In such a case, the output of the internal sensor would be similar to the observations registered by a conventional capture device, such as an oscilloscope or a spectrum analyzer. However, using internal sensors as capture systems poses new challenges.

They are meant to be small to improve their chances of remaining undetected. This impacts their resource availability and thus their quantification step, which yields low resolution traces. A common solution has been to capture multiple traces under the same $\{m, k\}$ tuple and then to “compose” these traces to improve their resolution. This increases the number of observations that must be captured, however. Another important issue is that trigger mechanisms are not ideal. On one hand, some works assume that the adversary can probe the control signals of the victim circuit to start and stop the acquisition. This limits the scope of RPA attacks. To solve this issue, some works have proposed to detect large fluctuations in the sensor output [SGMT21] or to use covert channels [FTBLN23] to synchronize the capture setup. These methods come with the disadvantage that some traces are not properly captured. These observations not only do not contribute to the quality of the dataset, but effectively act as noise which impacts it. Additional steps are needed to re-synchronize the traces.

In this work, we introduce a post-processing step for correlation-based attacks that improves the divergence of the correct hypothesis. Instead of simply selecting the maximum absolute value from the correlation matrix, we combine multiple Pearson coefficients for consecutive time points into a single coefficient. Then, we select the maximum value among these combined coefficients. This approach takes into account multiple time points

near the instant of interest, which is maximized for the correct hypothesis. Some works in the literature [LCSL07] have proposed to first create a composition of samples and then apply the statistical distinguisher. Our approach first computes the distinguisher and then composes a selection function. We base the soundness of this method on the analysis of Pearson’s correlation coefficient and its composability through Fisher’s transform. The proposed selection rule can help to discern the correct hypothesis with a smaller number of traces. In some cases, this is evidenced by an increased divergence (as a function of the number of observations) between the correct hypothesis and the corpus of other hypotheses. In other cases, the proposed improvement can discriminate the correct hypothesis with fewer observations. To assess its performance, we introduce three indicators based on the divergence of the correct hypothesis (separation from the second-highest hypothesis, the ratio between the top hypothesis and the second-best, the ratio between the top hypothesis and the cluster mean) as the number of observations increases.

We investigate the characteristics of “remote” power traces and their effect on the proposed method. We study the effects of noise and misalignment in the traces through experiments with real (TDC, ChipWhisperer) and synthetic data sets. These data capture (TDC, ChipWhisperer) or model (synthetic) the behavior of the encryption function of AES-128, allowing us to adjust the noise (synthetic) and misalignment (ChipWhisperer) rates of the dataset to present a comprehensive evaluation of the method. As shown, our findings can be useful for attacking any type of dataset. Indeed, a similar method has concurrently been proposed to mount template attacks [DJJW25]. In our case, we focus on RPA to address the aforementioned open challenges.

The remainder of the paper is structured as follows. In Section 2, we describe some mathematical methods employed in our work. Section 3 provides an overview of the previous work and the positioning of our study with regard to the literature. In Section 4, we describe the experimental conditions that led to the proposed method, as well as the improvements introduced in this paper. Section 5 features an experiment with synthetic data that aims to demonstrate the reproducibility of our findings. In Section 6, we show the applicability of the proposed improvement to conventional data sets. In Section 7 we show how the proposed method contributes to improve the literature results. Lastly, Section 8 expands our discussion and concludes the manuscript.

2 Preliminaries

We denote datasets X using capital letters, their i -th observation as X_i , and their mean as $\bar{X}$. We denote vectors $\mathbf{x}$ using bold symbols, and their i -th coordinates as x_i . We denote matrices M using capital letters, their i -th column as M_i , their j -th row as M^j , and their entry in the i -th row and j -th column as M_i^j .

2.1 Pearson’s correlation coefficient

Pearson’s correlation coefficient, in this work denoted as ρ , is a statistical measure used to quantify the linear relationship between two continuous variables [PG95]. It serves as a foundational tool in many scientific disciplines to assess how closely two variables are linearly related.

Definition 1. Let $X = \{X_1, X_2, \dots, X_n\}$ and $Y = \{Y_1, Y_2, \dots, Y_n\}$ represent two datasets, each of size n . The Pearson correlation coefficient, ρ , is given by

$$\rho(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2 \sum_{i=1}^n (Y_i - \bar{Y})^2}}. \quad (1)$$

The formula of Pearson's correlation coefficient is equivalent to the covariance of X and Y divided by the product of their standard deviations. In this way, we ensure that ρ is dimensionless and invariant to the scale and units of the variables being studied. Several key properties of Pearson's correlation coefficient make it a valuable measure. Firstly, it is symmetric, meaning that the correlation between X and Y is the same as that between Y and X . Furthermore, the coefficient ranges from -1 to 1 , and its interpretation depends on both its magnitude and direction, with values closer to ± 1 indicating a stronger linear relationship. More specifically, a value of $\rho \approx 1$ indicates a positive linear relationship, $\rho \approx -1$ signifies a negative linear relationship, and $\rho \approx 0$ implies no linear relationship between variables. However, it is important to note that ρ captures only linear relationships and alternative methods are required to evaluate non-linear relationships. Furthermore, Pearson's correlation coefficient is sensitive to outliers, which can disproportionately affect its value.

2.2 Fisher's transform

Fisher's transform, in this work denoted as z , is a mathematical function used to stabilize the variance of the Pearson's correlation coefficient and make it suitable for parametric statistical methods [Fis15]. This transform is particularly useful when combining or comparing correlations, as the distribution of ρ is not normally distributed, especially for values close to -1 or 1 . By applying the Fisher transform, the correlation coefficients are converted into a form that more closely approximates a normal distribution.

Definition 2. Let $X = \{X_1, X_2, \dots, X_n\}$ and $Y = \{Y_1, Y_2, \dots, Y_n\}$ represent two datasets of size n , and let ρ represent their Pearson correlation coefficient. The Fisher transform is given by

$$z(X, Y) = \frac{1}{2} \ln \left(\frac{1 + \rho}{1 - \rho} \right). \quad (2)$$

Fisher's transform z is approximately normally distributed. It is symmetric and ensures that small sample variances and skewness in the distribution of r are reduced, allowing for more reliable inferential procedures. When analyzing data from multiple studies or groups, it is often necessary to compute the mean of several Pearson correlation coefficients. Direct averaging of them is inappropriate because ρ does not have additive properties. Instead, the Fisher transform is used to compute the mean more accurately [DMCB98].

3 Positioning in the literature

Let an observation of the physical state of a characteristic of the device (ℓ) be composed of an ordered sequence of x samples ℓ^j , uniformly spaced at intervals t . Let the system $\mathcal{E}$ instantiated by $k \in \mathcal{K}$ (its sensitive information) be noted as $\mathcal{E}[k]$ and receive an input $m \in \mathcal{M}$. The system produces the output $c \in \mathcal{C}$ as $c = f(m, k)$ and $\ell = \mu(m, k, t)$. We assume that the adversary can choose m or c , query $\mathcal{E}$ a number of times y , and select the start and end of ℓ (and thus implicitly define x). In general terms, the function f is the procedure under attack and may represent a cipher, a hash function, a digital signature algorithm, a neural network, etc. Whereas μ is known as the leakage model and can be understood as the linear combination of f and a stochastic process:

$$\mu(m, k, t) = f(m, k) + \{\mathcal{R}_t\}. \quad (3)$$

In the literature, $\{\mathcal{R}_t\}$ is generally assumed to be a signal with normal distribution (Gaussian noise). In contrast, much research has been done on the nature of $f(m, k)$. Different authors have proposed to model this component as the Hamming weight or the Hamming distance, for example [BCO04].

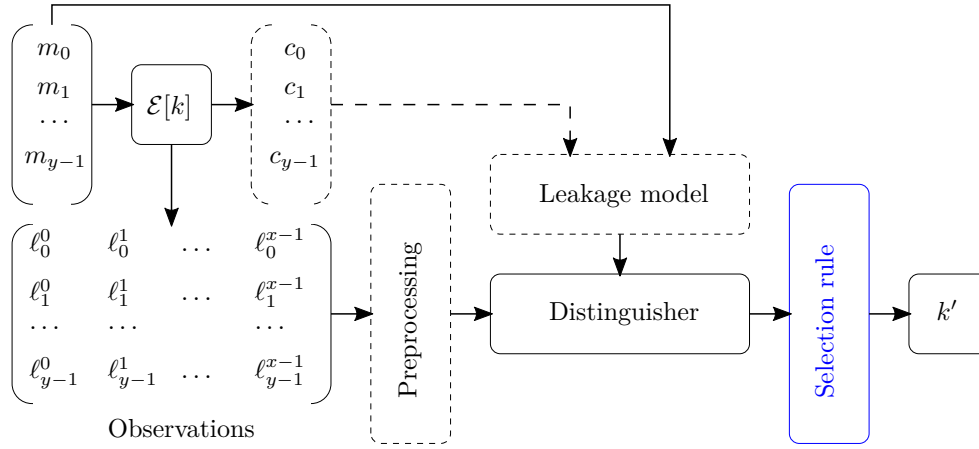


Figure 1: The general composition of a vertical SCA. A full description of this graphic is available in [FGGLN25].

SCAs were first introduced by Kocher et al. observing that the logical state influences the power consumption of the device [KJJ99]. The authors first described the *simple power analysis* attack (SPA), which studies the direct relationship between power traces and runtime [Koc96]. Subsequently, they proposed the *differential power analysis* attack (DPA), which studies the statistical dependency of the power traces on the secret data on the platform. Their seminal work has been adopted as a reference for the design of attacks and countermeasures. However, it also established a framework for the classification of SCAs into two broad families: horizontal and vertical. The *correlation power analysis* (CPA) proposed by [BCO04] belongs to the latter group.

As illustrated in Fig. 1, vertical SCAs follow a well-defined sequence of steps, an idea partially developed by [DPRS11]. A full description of these steps may be found in [FGGLN25]. The final step of attacks like CPA is the selection rule used to discriminate between the different hypotheses. This process is often overlooked, as—historically—most works dictate the selection of the hypothesis that maximizes the absolute magnitude of the distinguisher [KJJ99, BCO04, GHT05, MOP08, PRB09, MOS09, DPRS11, OP13, LPR⁺14]. To our knowledge, only one work in the literature has elaborated on the selection rule that best adapts to a particular kind of attack [LPR13]. In their work, Lomné et al. proposed an improvement in hypotheses discrimination by capitalizing on the characteristics of linear regression attacks [DPRS11]. Similarly, we observe the characteristics of the Pearson correlation test to propose an improvement to the selection rule of correlation attacks, particularly CPA. Instead of simply selecting the maximum absolute value from the correlation matrix, we first calculate a weighted mean of the absolute values over a defined interval of x . Then, we identify the maximum value among these weighted means. This approach takes into account all correlation coefficients near the instant of interest, which is maximized for the correct hypothesis.

Our proposed improvement comes from the observation that the samples in a power trace are not independent. Indeed, CPA was designed to select the sample with the highest correlation coefficient. However, this ignores adjacent samples, which can also provide useful information. Some works in the literature [LCSL07] have proposed to first create a composition of samples and then apply the statistical distinguisher. Our approach first computes the distinguisher and then composes a selection function. We base the soundness of this method on the analysis of Pearson’s correlation coefficient and its composability through Fisher’s transform.

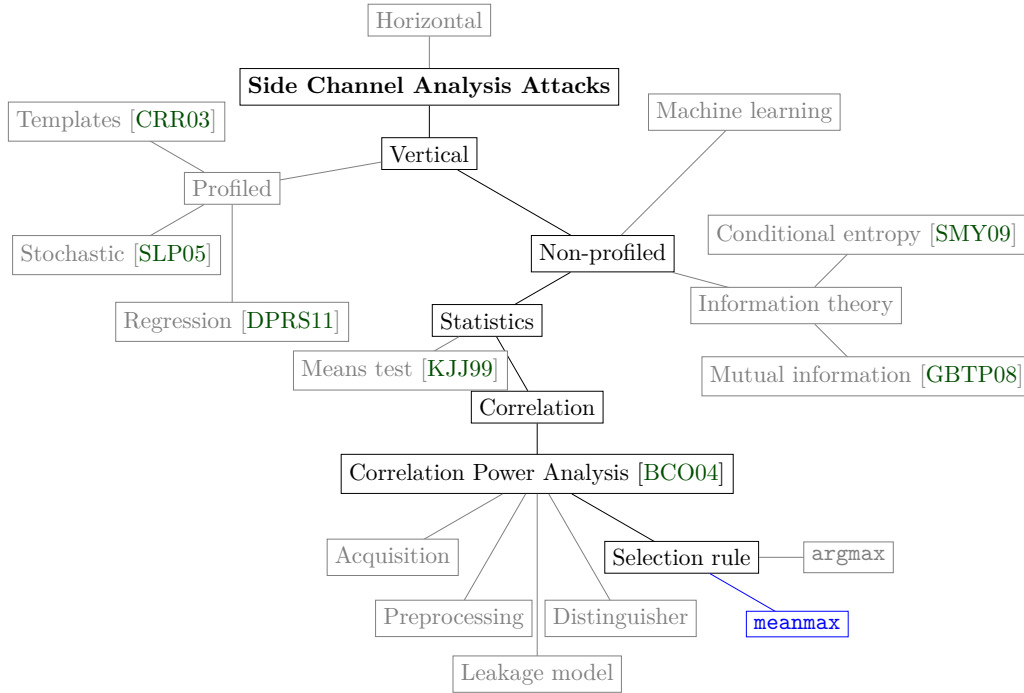


Figure 2: Taxonomy of side channel analysis attacks.

In Fig. 2, we show the positioning of our work in relation to the state of the art. We focus on the selection rule for correlation attacks, using as a case study the correlation power analysis attack proposed by Brier et al. [BCO04]. Instead of using the conventional **argmax** function, we propose a post-processing + selection step called **meanmax**. This has implications for attacks that use the Pearson correlation coefficient as a distinguisher.

4 Correlation attacks in practice

Remote power analysis is an emerging threat to computing systems. It allows us to reduce the adversarial assumptions required to mount an attack. Indeed, the versatility and proficiency of these attacks cannot be underestimated. However, as a recent field of study, it has many open questions and offers areas for improvement. In Fig. 3, we illustrate a basic RPA attack scenario. The adversary implements a sensor circuit in a heterogeneous SoC and uses side-channel leakage to carry out an attack. This allows them to retrieve the secret key of an encryption algorithm, in this case AES-128.

4.1 Correlation power analysis

Correlation power analysis improved the original DPA attack in multiple ways. It introduced the use of leakage models, extended the number of bits to be “guessed” in a single pass, and used Pearson’s correlation coefficient as a distinguisher. A detailed overview of this method can be found in [FGGLN25]. If we look inside the algorithm, the result of CPA is not a single value for the suggested hypothesis. Rather, the algorithm constructs a two-dimensional array of correlation coefficients with dimensions 2^n -by- x . Recall that x is the number of samples in the power observations and let n represent the binary length of the secret k . In DPA, we have $n = 1$ since each bit is evaluated independently. But CPA allows us to test multiple bits simultaneously. When attacking AES-128 it is common to

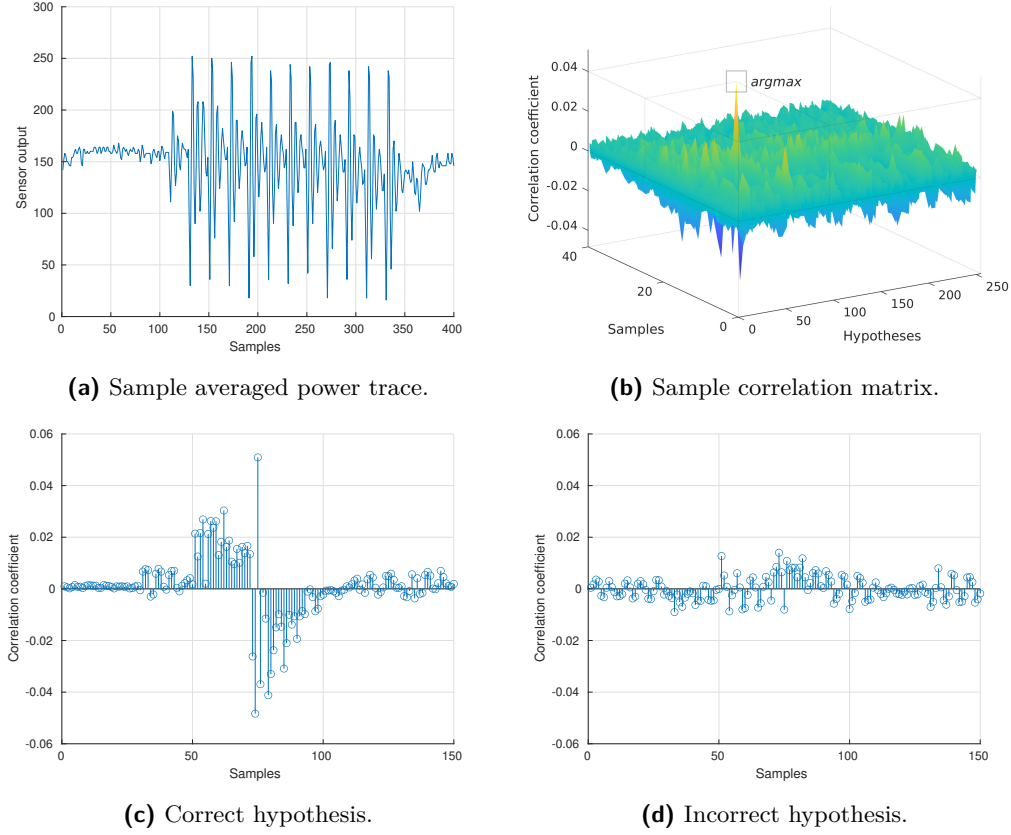


Figure 3: Remote power analysis results using an AMD-Xilinx Zynq-7000 SoC-FPGA (28nm). The sensor is an array of eight time-to-digital converters with 8-bit resolution and a sampling rate of 200MHz. The victim is a textbook, iterative implementation of AES-128 clocked at 10MHz. Both, the sensor and the target are implemented in the reconfigurable fabric of the device.

target its SBOX output, thus $n = 8$ -bits are tested and $2^n = 256$. As all the hypotheses are tested for each sample in the power observations, this yields a correlation matrix with the dimensions aforementioned. An example of this correlation matrix is illustrated in Fig. 3b. Conventionally, the **argmax** selection rule is applied to identify the global maximum of this array. However, when y is not large enough, the correlation values for the correct hypothesis can be obfuscated by other local maxima. It is assumed that with increasing y the correct hypothesis will eventually “emerge.” Nonetheless, a larger y means capturing and processing more power observations.

The main limitation with the use of **argmax** as a selection rule is that it only considers one dimension of the correlation matrix. Indeed, from the proposed definition of vertical attacks, CPA is most efficient with $x = 1$. That is, if we can sample the precise instant in which the information leakage takes place. But in practice, the adversary needs to capture multiple samples and test them. The attacker is supposed to know the instants in which the victim begins and ends their processing. This is a very generous assumption and yet produces a large number of samples, as can be observed in Fig. 3a. Based on this, we will operate on the understanding that $x \gg 1$ and therefore there is information that is not fully used by **argmax**. In fact, when we look at the correlation values for the correct hypothesis, it is possible to see that multiple samples contain useful information. In

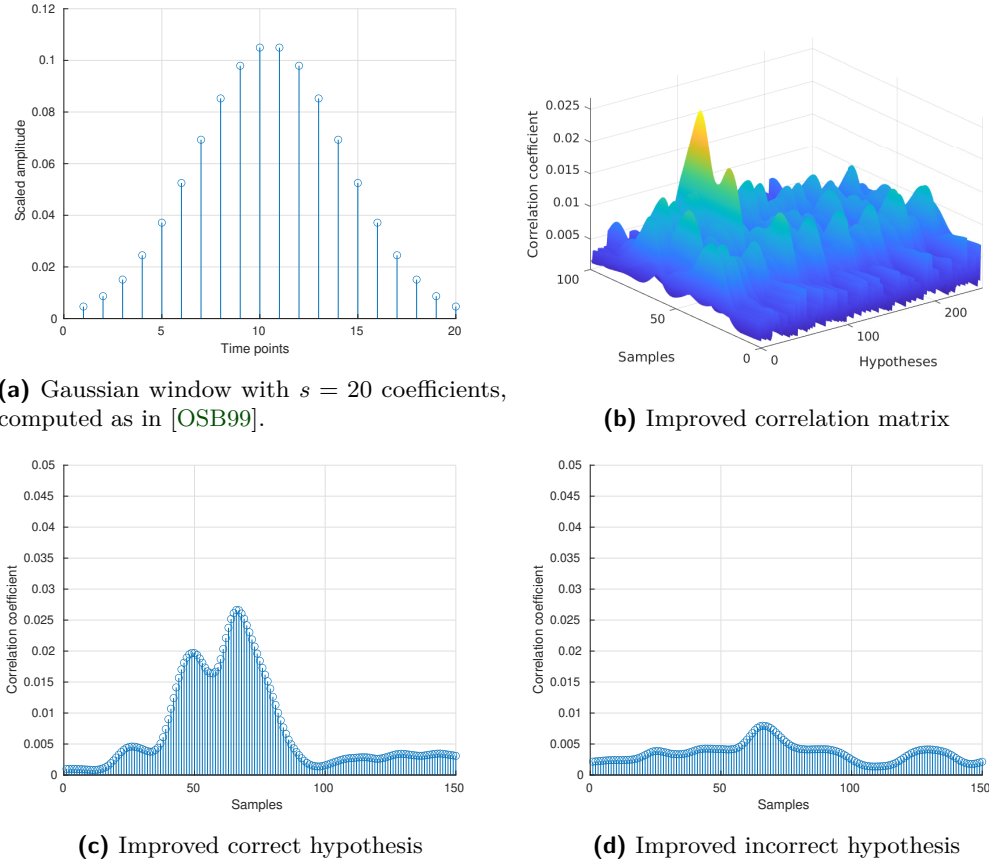


Figure 4: Application of a sliding-window post-processing step to improve the selection rule of CPA. These data correspond with the experiment in Fig. 3.

contrast, the behavior of the correlation coefficients for multiple samples over an incorrect is mostly uniform. This is shown in Fig. 3c and Fig. 3d, respectively.

It is likely that there are several factors behind the behavior of the correlation test on the samples of a correct hypothesis. First, positive and negative correlation values are interesting since they imply some dependence on the secret information. That is why we take the absolute value on the correlation matrix as noted in [BCO04]. We must then recall that the time samples retrieved from the system are not independent events. The behavior of the system at instant t will depend on its state at instant $t - 1$, and both will influence its state at instant $t + 1$. Thus, we expect information leakage to be present in several samples. This phenomenon can be amplified if the sampling frequency surpasses the frequency of the target. In fact, logical circuits are expected to complete a given operation within a cycle. However, the actual time response of the circuit depends on several implementation factors. This leads to a scenario where we cannot assume that we obtain a single sample of interest. Lastly, physical defects, such as parasitic currents and synchronization issues during acquisition, can “shift” the point of interest over time. The latter is of special interest for this work.

4.2 Our improvement

Trace desynchronization or misalignment can lead to observations that yield correlation maxima at different moments t . This affects the conventional argmax selection rule, which

Algorithm 1 The proposed **meanmax** selection rule for CPA.

Require: a vector $\mathbf{m}$ of known y inputs processed by $\mathcal{E}[k]$
Require: a matrix $\mathbf{L}$ of y power traces (of x samples each)
Require: n , the binary length of k
Require: a scalar s , the number of samples per cycle
Ensure: k' , a hypothesis for the sensitive information k

```

1:  $\mathbf{w} \leftarrow \omega(s)$  {Compute the window weights, as per [OSB99].}
2: for  $i = 0$  to  $2^n - 1$  do
3:    $\mathbf{u} = f(\mathbf{m}, i)$  {Compute the leakage model, as per (3).}
4:   for  $j = 0$  to  $x - 1$  do
5:      $q_j \leftarrow z(|\rho(\mathbf{u}, \mathbf{L}^j)|)$  {Fisher transform ( $z$ ) is applied over the  $\rho$ , as per (1), (2).}
6:   end for
7:   for  $j = 0$  to  $x - s - 1$  do
8:      $R_i^j \leftarrow z^{-1}(\mathbf{w} \cdot (q_j, \dots, q_{j+s}))$  {Sliding window, inverse Fisher transform.}
9:   end for
10: end for
11:  $k' \leftarrow \operatorname{argmax}(\mathbf{R})$ 
12: return  $k'$ 

```

relies on a single instant to determine the best candidate hypothesis. In this regard, pre-processing techniques are normally suggested to deal with this issue. However, we can also leverage the characteristics of the correlation matrix to improve CPA.

To account for the information leakage that appears along multiple samples, we can compute a weighted mean over a finite interval of time points, where the size of this interval can be defined as the number of samples obtained from a cycle of the target (the number of samples per cycle). That is, the ratio between the sampling frequency and the target's frequency. The weights represent the probability of the maximum correlation spike to be found in any given t . Thus, we can employ a window function with a given distribution for the weights and perform a convolution over the correlation matrix along the x samples for all the $|\mathbf{h}|$ hypotheses. However, since Pearson's correlation coefficients do not have additive properties, we first apply the Fisher's transform over the correlation matrix. Then, we compute the mean of the resulting values and convert it back using the inverse Fisher's transform. Finally, we can test the maximum among all these means to choose a hypothesis. By applying this sliding-window technique, we mitigate the dependency on any techniques to detect points of interest. The effects of the proposed selection rule, called **meanmax** are shown in Fig. 4 and its application to CPA is detailed in Alg. 1.

The proposed selection rule can help to discern the correct hypothesis with a smaller number of traces. This result is illustrated in Fig. 5. In some cases, this is evidenced by an increased divergence, as a function of the number of observations, between the correct hypothesis and the corpus of other hypotheses. In other cases, the proposed improvement can separate the correct hypothesis with fewer observations than the **argmax** selection rule. However, questions remain that must be answered. How can we determine that the correct hypothesis becomes clearer? Can this improvement be quantified? What is the success rate of the proposed method over conventional CPA? Can this be used in other contexts? In the following section, we try to address these points.

5 Correlation attacks *in vitro*

In this section, we intend to reproduce the behavior observed in remote power analysis with synthetic power traces. This allowed us to control the noise and misalignment rates.

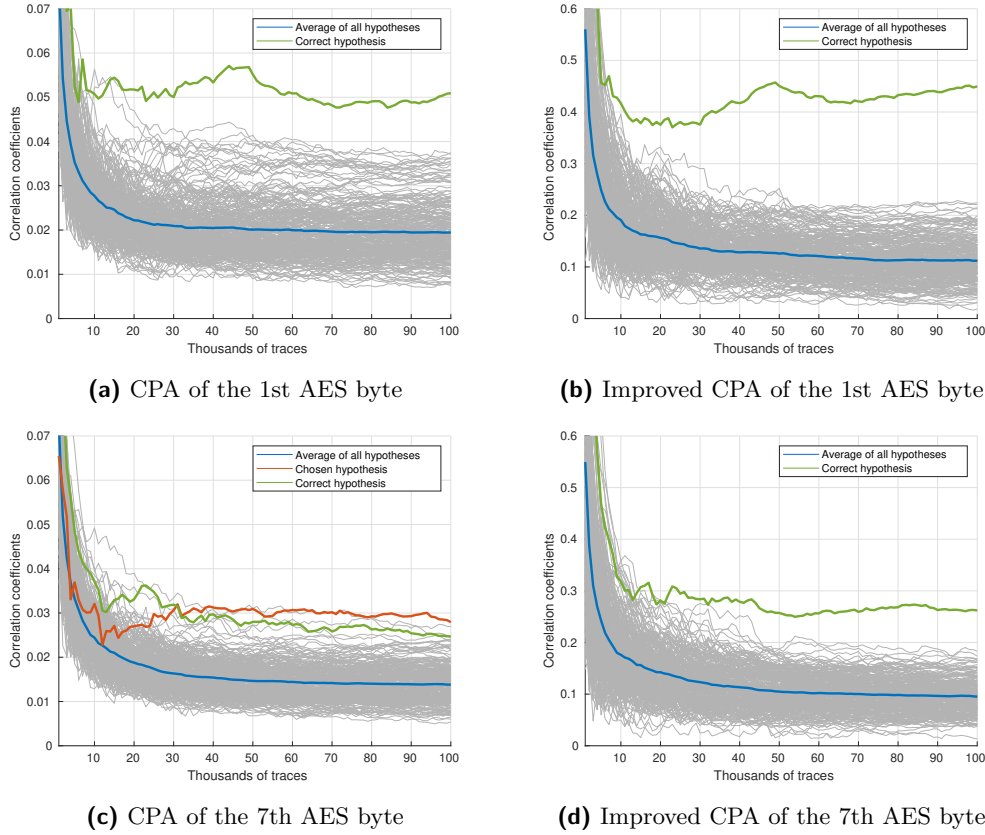


Figure 5: Comparison of two CPA attacks using the conventional **argmax** selection rule (left column) and the introduced **meanmax** selection rule. The proposed approach increases the divergence of the correct hypothesis (top row) and in some cases can reveal the correct hypotheses using less power traces (bottom row). In this instance, we use a Gaussian window function with $s = 20$ coefficients and standard deviation $\sigma = 0.346$. These data correspond with the experiment in Fig. 3.

5.1 Divergence of the correct hypothesis

Quantifying the performance of SCAs has been a topic of much discussion. Usually, we can register the progression of the greatest correlation point for each hypothesis as a function of the number of observations. Nevertheless, by looking at these results we can draw different conclusions. The adversary will be inclined to report that the correct key was identified with a lower number of observations. So, they might stop the algorithm as soon as the correct hypothesis “comes on top.” We refer to this point as the first divergence from the other hypotheses (the corpus). This notion has been complemented with the development of the key-rank metric commonly used in the literature [VGS13]. However, in some circumstances, an incorrect hypothesis may “remount” when more observations are added to the dataset. The adversary is only led to believe that the correct key was found because they know its value *a priori*. On the other hand, the victim may claim that the actual value of the key is not known until the correct hypothesis shows a significant divergence from the corpus. A similar challenge arises when trying to compare SCA methods, as in our case.

We propose that the rate of divergence for an attack can be estimated based on the Euclidean distances between the top hypothesis and the second-best hypothesis (δ_1) and

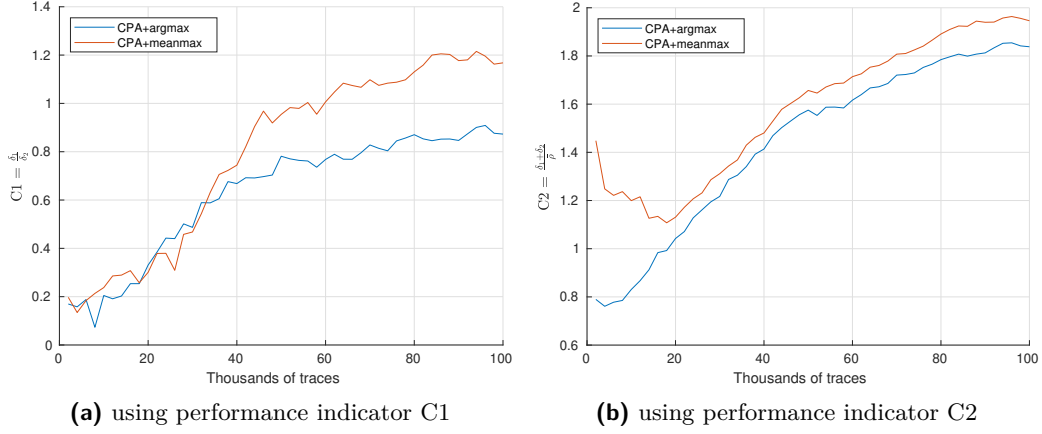


Figure 6: Comparison between CPA+argmax and CPA+meanmax. The computed performance indicators are averaged over 16 bytes. These data correspond with those in Fig. 3.

between the latter and the mean of the corpus (δ_2), for a fixed number of observations. Please refer to Fig. 6 of [FGGLN25] for a graphic description of these distances. Let $\bar{\rho}$ represent the mean correlation score of all the hypotheses for a given number of observations (y) and define the following *performance indicators*:

$$C1 = \frac{\delta_1}{\delta_2}, \quad C2 = \frac{\delta_1 + \delta_2}{\bar{\rho}}. \quad (4)$$

As the magnitudes of these indicators are normalized, we can use them to assess the performance of the proposed selection rule compared to the **argmax** function. This is illustrated in Fig. 6. In this case, when looking at C1 we can see how the proposed selection rule improves the separation of the first hypothesis as the number of observation increases. As for C2, our improvement consistently increases the distance between the top hypothesis and the mean of the cluster. This indicates that the proposed technique has the effect of smothering the correlation scores of the incorrect hypotheses and thus reducing the number of traces reduced to conduct a successful attack.

There is room for discussion regarding the selection of the proposed performance indicators. As observed by [Man04], “the number of samples that is needed to see a [correlation] peak in practice, is mainly determined by the distance between the sampling distributions with $\rho = 0$ and $\rho = \rho_{max}$.” These sampling distributions represent the corpus of all incorrect hypotheses which converges towards $\rho = 0$ and the correct hypothesis which converges towards $\rho = \rho_{max}$. It follows that we want to observe two indicators: C1 verifies that a single hypothesis is left in the sampling distribution with $\rho = \rho_{max}$ and C2 verifies that it indeed converges towards $\rho = \rho_{max}$ (as the number of observations increases). Note that we do not simply rely on the Euclidean distances δ_1, δ_2 as we also want to normalize the magnitude of the indicators as to allow for straightforward comparison between independent pairs of sampling distributions.

5.2 Synthetic power traces

Following the leakage model in (3), we generated a dataset of synthetic power traces. We used the AES-128 SubBytes function to define the procedure under attack

$$f(m, k) = \text{HW}(\text{SubBytes}(m \mathbf{xor} k)), \quad (5)$$

where $\text{HW}()$ represents the calculation of Hamming weight normalized to the interval $[-0.5, 0.5]$. The values of k and m were chosen at random every time, note that $\log_2 m =$

$\log_2 k = 128$. The output of f was collected in a single sample of interest i . The precedent and subsequent samples in the trace were initialized with the Hamming weight of the output of the AddRoundKey and MixColumns functions of AES-128, respectively. We used this strategy to approach the behavior of a hardware AES-128 architecture. To model the influence of the stochastic process $\{\mathcal{R}_t\}$ we generated a sequence of pseudo-random bits $R(t)$ in the interval $[-0.5, 0.5]$ with a normal distribution. We used a scaling constant α to adjust the amplitude of this noise. It can be said that the noise amplitude is alpha times greater than the data. Each synthetic trace consisted of multiple samples t defined as

$$\mu(m, k, t) = \begin{cases} \text{HW}(m \mathbf{xor} k) + \alpha R(t) & t < i \\ \text{HW}(\text{SubBytes}(m \mathbf{xor} k)) + \alpha R(t) & t = i \\ \text{HW}(\text{MixColumns}(\text{SubBytes}(m \mathbf{xor} k))) + \alpha R(t) & t > i. \end{cases} \quad (6)$$

This strategy allowed us to produce a two-dimensional array of y -by- x synthetic samples with y as the number of observations (messages, synthetic traces) and $x \geq 3$ as the number of samples. Evidently, only the samples at $t = i$ contain leakage information. However, since we intend to study the characteristics of RPA (namely misalignment), additional containers are needed.

5.3 Effects of trace misalignment

The leakage model provided in (6) accounts for most of the noise generated by the acquisition process, if we assume that it is uniformly distributed. But it disregards noise sources with low periodicity (i.e. thermal noise and jitter), as their influence over small values of x can be considered negligible. This model is accurate for the study of power analysis attacks in conventional setups. However, remote power analysis attacks are susceptible to other perturbations, notably trace misalignment. This is especially noticeable in capture schemes that do not rely on logical trigger signals [FTBLN23]. Although it is true that conventional capture setups are also prone to trace misalignment, in such cases, the common recommendation is to perform a pre-processing step which considers resynchronization algorithms [MOP08]. But this process is complex in remote scenarios given the lower resolution of the capture setup and the limited processing capabilities of the sensor itself. Nonetheless, we show that the **meanmax** selection rule can contribute to improving the performance indicators of CPA (C1, C2) in the presence of moderate noise and trace misalignment rates.

We designed an experimental methodology inspired by the evaluation methods described in [DPRS11] (Fig. 10 in their paper). This statistical test was originally intended to observe the evolution of the success rate according to the number of messages (y) and the amplitude of the noise component (α). However, we added an additional dimension to this analysis by using datasets with different misalignment rates (β). We provide experimental results for $\alpha \leq 2.5$ and $\beta \leq 40$. These upper bounds were determined by the computing time. We observe that the proposed method is most effective for large α and β , if enough observations are provided. As we also perform multiple CPA runs per data point, the chosen values show this effect with reasonable computing time. We considered that each synthetic trace could be misaligned by ± 1 sample, with $\beta\%$ of the traces being randomly misaligned. This means that $\beta/2$ of every 100 traces will have the AddRoundKey data at instant i and $\beta/2$ of every 100 traces will have the MixColumns data at instant i . The reason behind choosing a single sample misalignment is that the temporal leakage dispersion can also be captured with the proposed sliding window. Thus, greater dispersion distributions can be addressed by adjusting the window function.

Since we wanted to evaluate the prowess of the proposed selection rule, each test reports only the percentage of experiments where CPA+**meanmax** is correct (i.e. the key is

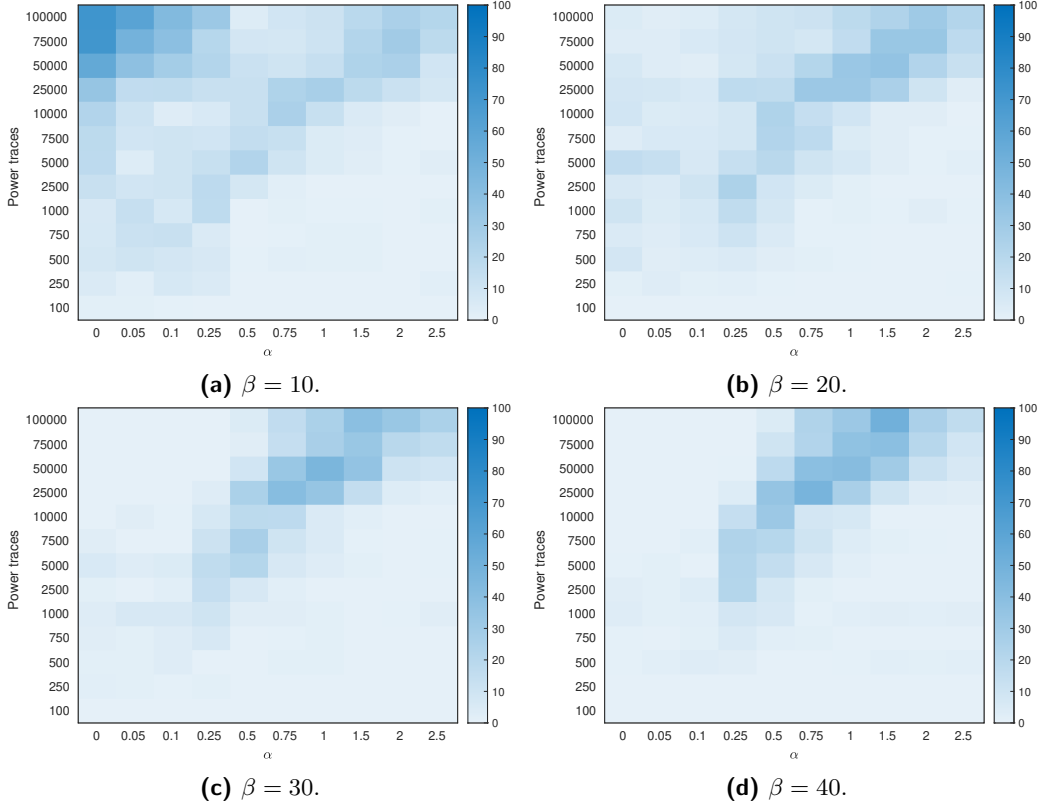


Figure 7: Success rate (W1) of CPA+meanmax in function to the number of messages (y), the noise gain (α), and the misalignment rate (β).

successfully recovered) and a) improves indicator C1 (the results are shown in Fig. 7) or b) improves indicator C2 (the results are shown in Fig. 8). These indicators are compared against the same indicators calculated for CPA+argmax. Given that the synthetic traces produced from the model in (6) contain information about 16 different AES-128 key bytes, we repeated the experiments 10 times to estimate the success rate of attacking 160 different key bytes. To derive our comparison results we performed a byte-to-byte analysis of both methods. We first computed the distances δ_1, δ_2 and asserted their correctness. We then calculated the performance indicators C1 and C2 as in (4). Lastly, we computed the “success rate” (W) for CPA+meanmax as

$$W1 = \frac{\sum_{byte=1}^{160} ((C1' > C1) \& \text{byte_found})}{160} \times 100. \quad (7)$$

In this case, C1 is computed for argmax and C1' is computed for meanmax. Note that we denote the “success rate according to the indicator C1” as W1, but an analogue formula can be derived for indicator C2.

To clarify, these experiments show that in the given percentage of experiments (how many bytes are recovered out of the 160) the proposed method (CPA+meanmax) contributes either to increase the divergence between the correct hypothesis and the second best (Fig. 7) or to segregate the correct hypothesis while a substantial portion of the corpus is led to converge towards low correlation values at a faster rate (Fig. 8). This implies that in many cases CPA+meanmax is correct, but it does not manage to improve either indicator. Indeed, as shown in Appendix A, the proposed method effectively improves the success rate of the baseline.

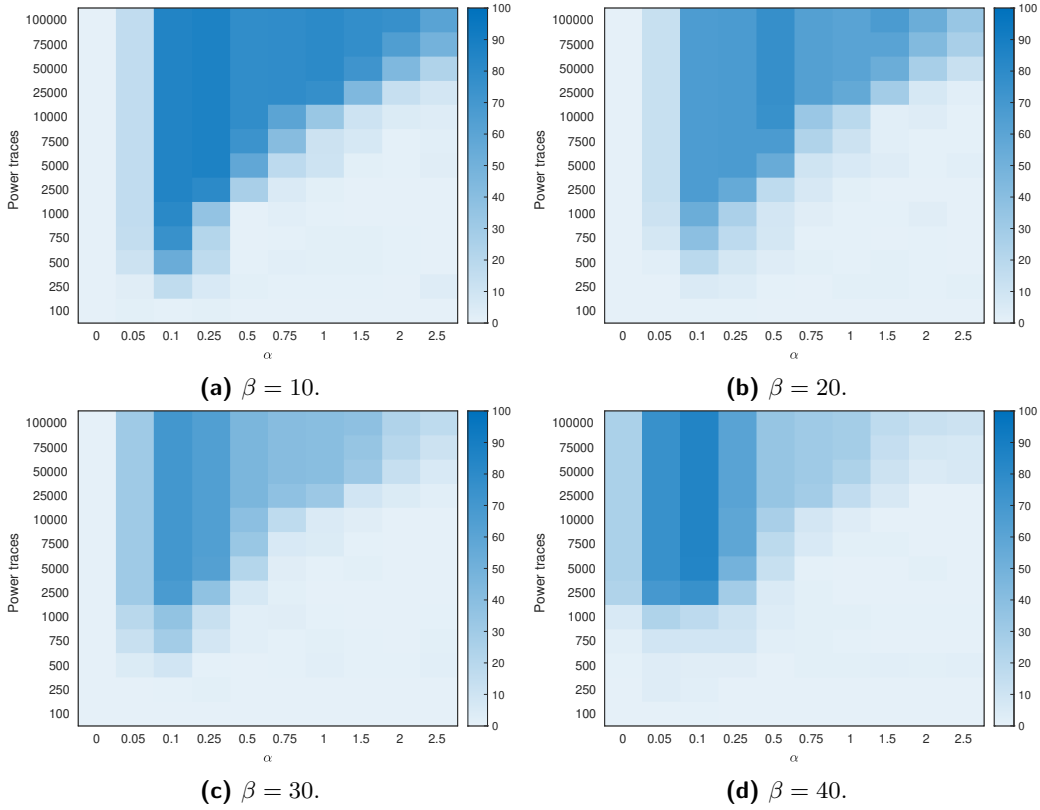


Figure 8: Success rate (W2) of CPA+meanmax in function to the number of messages (y), the noise gain (α), and the misalignment rate (β).

In Fig. 7 we observe a moderate advantage for CPA+meanmax when looking at the indicator C2. The success rate appears to be somewhat affected by the misalignment rate (β), in particular for a large number of observations (y) and lower noise content (α). This is expected since misaligned traces introduce additional noise into the dataset. It is likely that by increasing the number of power traces we would observe a similar behavior for all values of β . More research is needed to understand the non-continuity of the success rate W1 in relation to the noise gain α and the number of observations. However, we remark that this phenomenon only affects the capacity of CPA+meanmax for improving the performance indicator C1. Indeed, from Appendix A it can be corroborated that with enough observations both methods are consistently correct.

In Fig. 8 we instead look at the advantage obtained for the indicator C2. In this case, we observe a greater advantage for CPA+meanmax which appears to be maintained even as the misalignment rate increases. This analysis highlights the effectiveness of the proposed selection rule for smothering the incorrect hypothesis. An interesting effect that can be observed in Fig.8 is that when the noise gain (α) is zero, the argmax selection rule has a significant advantage. In fact, the proposed meanmax is a post-processing technique that can mitigate spurious correlation spikes found in the correlation matrix. And these spikes are generated, in the synthetic model, from the Gaussian noise ($R(t)$) introduced in the traces. In that sense, we can consider the proposed method as a low-pass filter applied over the correlation matrix. The cut frequency is implicitly defined by the number of samples in the sliding window (s). As described above, a convenient way to define s is to take the relationship between the sampling frequency and the operating frequency of the target. Nevertheless, this simply represents an upper bound, and in some cases a

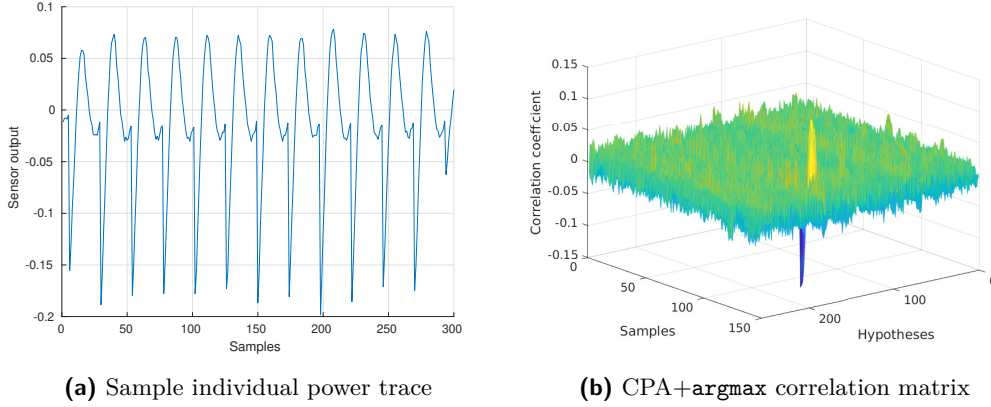


Figure 9: Power analysis results using the ChipWhisperer-Lite and ChipWhisperer-CW305 setup. In the sample power trace it is possible to appreciate eleven segments separated by power spikes, these correspond to the internal iterations of the AES-128 architecture.

smaller window could yield better results.

In the experiments reported in this section, we consider that the percentage of misaligned samples is known. Therefore, we can define a perfect mask instead of employing a window function as in the previous section. This represents the optimal window where the proposed selection function is immune to misalignment. However, as shown in Section 4, it is possible to employ window functions that approximate the probability density function of the misalignment in the correlation coefficients for the correct hypothesis. The width and shape of this window should approach the distribution of the dispersion of the misaligned traces (how many samples) and the misalignment rate (how many traces). Evidently, applying a sliding-window filter over the correlation matrix is equivalent to performing a convolution with a filter window. This suggests that applying a Fourier transform to the correlation matrix can yield useful information for discriminating the correct hypothesis. This idea is further explored in [FGGLN25].

6 Correlation attacks *in vivo*

In this section, we study the performance of the proposed selection rule when small misalignments are introduced into the traces. That is, we used a dataset of generic side-channel measurements and induced some controlled perturbations. This experiment differs from the results provided in the previous sections in the following aspects:

- The sensor is a ChipWhisperer-Lite device with an acquisition rate of 96 MSps.
- The victim is a textbook iterative implementation of AES-128 (a round is computed each cycle) clocked at 4 MHz. Implemented in a Chipwhisperer CW305, which embeds an AMD-Xilinx Artix 7 FPGA (28nm).
- This capture setup showed greater challenges to obtain positive CPA results; therefore, we used the last round differential leakage model (hence, a lower number of traces is required w.r.t. Section 4).

The traces used in this experiment included a full encryption of AES-128 with random plaintexts. We obtained ten sets of traces with random different keys. Therefore, we could attack a total of 160 different key bytes as in the previous Section. For the capture, we used the internal triggers of the victim to synchronize the acquisition, and thus it

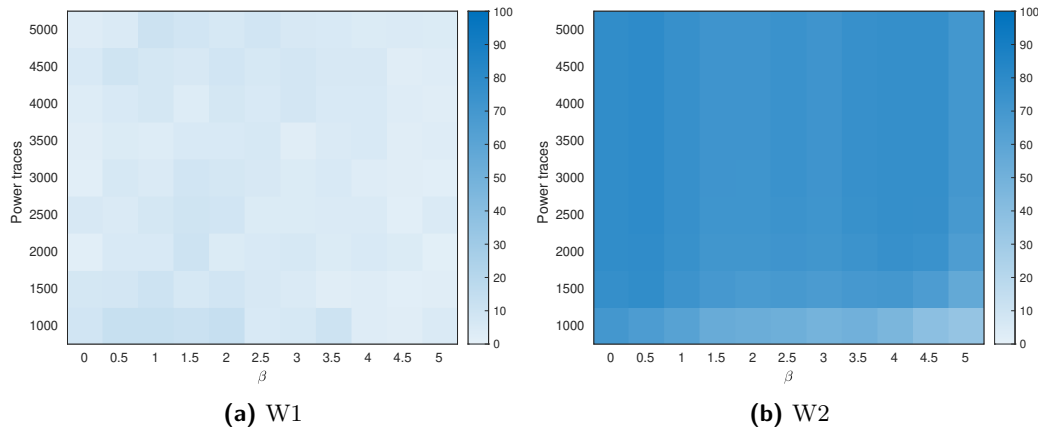


Figure 10: Success rate (7) of CPA+meanmax according to the performance indicators from (4), in function to the number of messages (y) and the misalignment rate (β).

represents a scenario where $\beta \approx 0$. However, with this dataset we can introduce a controlled misalignment rate to study the behavior of the proposed selection rule when the noise (α) comes from a physical system. Fig. 9 shows a sample trace and classic CPA results.

Next, we applied the proposed selection rule to perform an evaluation of its improvement in conventional datasets. The attack attempted to retrieve the 160 secret bytes in the dataset for misalignment rates up to 5%. As in the previous section, we were interested in the cases where CPA+meanmax outperforms CPA+argmax according to indicators C1 (greater divergence between the second hypothesis and the mean of the cluster) and C2 (overall divergence from the mean of the cluster). The results are shown in Fig. 10.

This experiment shows that CPA+meanmax contributes moderately to improve the separation between the correct hypothesis and the second choice for small misalignment rates. In contrast, it significantly increases the separation between the correct hypothesis and the cluster’s mean, even for small misalignment rates. This suggests that the proposed method overall accelerates the convergence of incorrect hypothesis towards a low correlation value, while in some cases not necessarily reducing the correlation score of some spurious hypotheses. Indeed, this can be observed in Fig. 5. This suggests that in cases where the segregation of the correct hypothesis is not conclusive, it may still reduce the number of potential candidates for an enumeration attack. These results can appear less favorable than the results given in the previous Section 5, where significant misalignment rates were used. However, we must note that, in this case the artificial misalignment is applied over a strong noise component derived from a real measurement system. Another relevant factor is the change in the acquisition setup. In particular, a closer look reveals that the dispersion first observed in Fig. 3 is less pronounced when using the ChipWhisperer capture setup, as illustrated in Fig. 11.

This poses the question of why the leakage is more dispersed over the RPA power traces. Initially, we formulated the hypothesis that trace misalignment could contribute to increasing this dispersion. But as shown in this section, a strong desynchronization is in fact detrimental for the proposed method. Nevertheless, the solution is evidently to adjust the number of samples used to define the sliding window in Alg. 1. For example, for the ChipWhisperer scenario, the results are improved if we set a window width of four samples. These results are provided in Fig. 12. With this adjustment, the improvement rates for the divergence between the second hypothesis and the cluster mean are less sensitive to large misalignment rates. This is consistent with the results observed in Section 5.

In Section 4.2 we noted that the window width could be computed as the ratio between the sampling and the victim’s frequencies. This works well for RPA scenarios. However, as

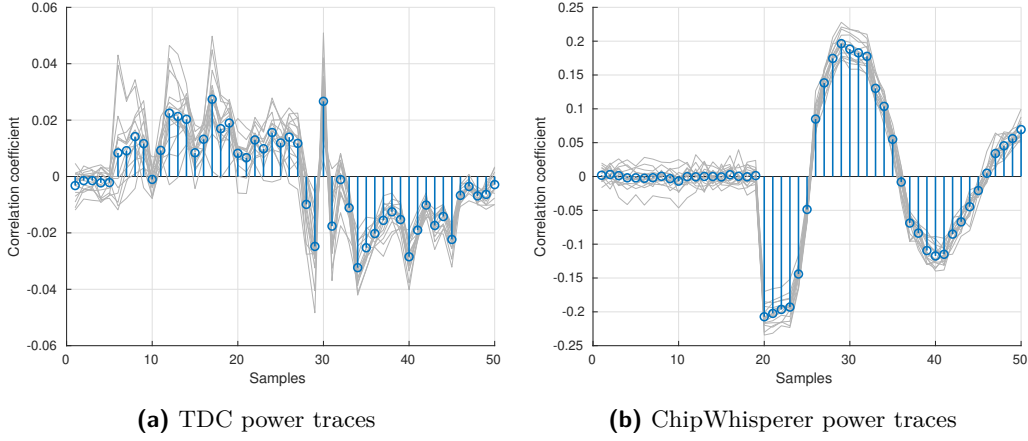


Figure 11: Detail of the correlation coefficients for the correct hypotheses over multiple samples. The correlation coefficients are computed following the conventional CPA approach (CPA+ argmax). In gray we show individual (correct) hypotheses from the correlation matrix and in blue their average values.

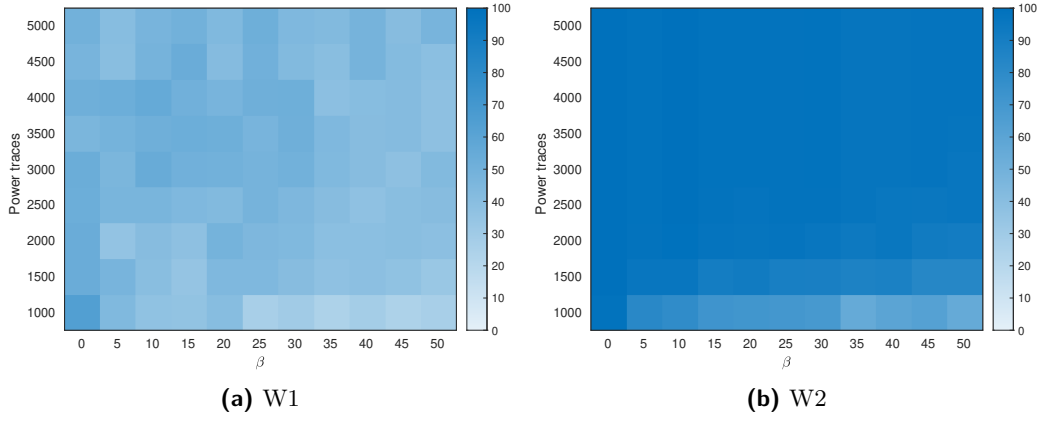


Figure 12: Evolution of the success rate (7) in function to the number of messages (y) and the misalignment rate (β). In this case the sliding window considers a leakage dispersion over four samples.

shown in this Section, this likely represents an upper bound in terms of finding the target computation within a given time frame (concretely a clock cycle of the victim). Therefore, for some cases it might be more interesting to focus on narrower intervals. This can be addressed either by reducing the window width or by adjusting the window model so that its coefficients better capture the behavior of the system.

7 Comparison with the literature

Works in the literature have proposed pre-processing techniques oriented to combine multiple samples of a trace into a single time instant [LCC⁺06, LCSL07]. This allows statistical attacks to capitalize on the amplified leakages to derive a result through argmax , which in this way can benefit from multiple time points. One of these approaches [LCC⁺06] relies on selecting multiple samples of interest to formulate a multiple-bit DPA, ultimately

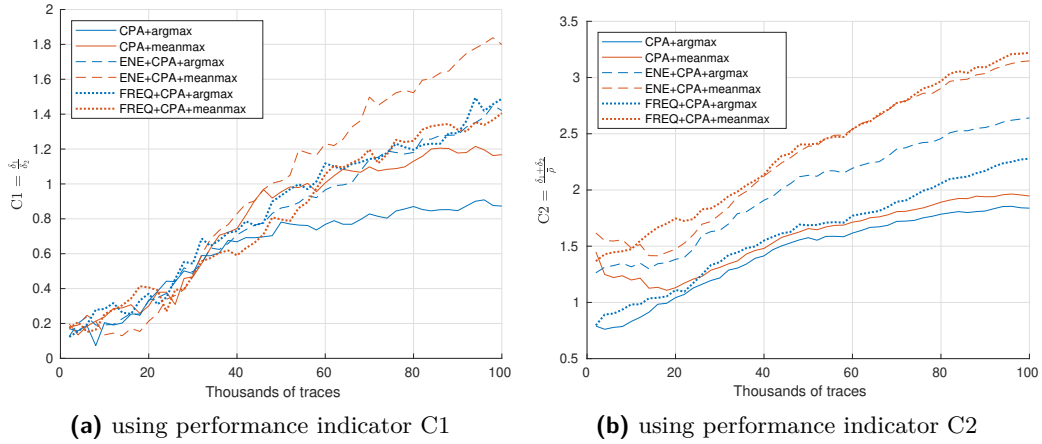


Figure 13: Comparison of different pre-processing techniques and their interaction with the proposed selection rule. For each pre-processing method (None, ENE [LCSL07], FREQ [GHT05]) we perform CPA using both selection rules (argmax , meanmax) and compute the performance indicators from (4). For ENE+CPA+ meanmax we used a pre-processing interval of 20 samples and a Gaussian window with five samples for meanmax . In the other instances we used a Gaussian window of 20 samples for meanmax .

concluding that CPA is an instance of their method. [LCSL07] proposed to divide the traces in small windows and to integrate over these intervals to create energy scores. This has the additional benefit of reducing the size of the input. Other works have instead propose to study the frequency contents of the input traces, which can solve the issue of misalignment [GHT05, JZ23]. We note that since these are pre-processing techniques they are compatible with our approach. Indeed, we can easily apply the proposed selection rule to the CPA output regardless of the nature of the input. This is demonstrated in Fig. 13.

From this experiment we can corroborate that the proposed selection rule can improve the performance indicators of CPA independently of the type of input. Nonetheless, further research is needed to understand the effect of these methods on the correlation matrix.

8 Conclusions

In this paper, we have introduced a new selection rule for correlation attacks that exhibits favorable characteristics in the presence of trace misalignment. The proposed improvement relies on the observation that the correlation matrix used as a distinguisher in certain vertical attacks is not made up of totally independent observations. Instead, the independence exists only between hypotheses, but the correlation values for consecutive samples are related. Indeed, for the vector associated with the correct hypothesis we can identify—under ideal conditions—not only the global maximum, but also a subset of samples with increased correlation values.

This observation was first made in the analysis of the behavior of remote power analysis attacks, where the proposed technique can be used in practice to improve the effectiveness of CPA. From this scenario, we set out to investigate the causes of the observed phenomenon. It is likely that multiple factors are at play. But in this paper, we focus on the idea that moderate desynchronization of the traces is one of the causes. This is a particular characteristic of remote power analysis, where the acquisition mechanism is sometimes decoupled from the logical triggers of the algorithm to simplify the adversarial assumptions. This produces, consequently, measurements that have to be resynchronized to be useful for standard analysis methodologies.

To test this hypothesis, we constructed a set of power traces according to well-understood power models and modified the noise and misalignment rates. This allowed us to study the performance of CPA under the classical and novel selection rules. We introduced two attack performance indicators based on the divergence of the correct hypothesis (C1: separation from the second highest hypothesis, C2: the separation between the top hypothesis and the cluster mean) as the number of observations increases. This allowed us to demonstrate that the proposed method improves CPA under certain scenarios.

We also evaluated our approach with a conventional power traces dataset. We used a ChipWhisperer capture setup and created a corpus of well-aligned power traces. We then introduced some artificial misalignment to these data and evaluated the performance of the proposed selection rule. In this case, we identified that the method is sensitive to the width of the window, which can be upper-bounded by the ratio between the sampling frequency and the frequency of the target.

The proposed selection rule requires applying a sliding window over the correlation matrix along the time dimension for each hypothesis. This can be perceived as an application of a low-pass filter on the correlation matrix. Indeed, smoothening high frequency components contributes to discern the correct hypothesis with greater certainty. As a side effect, the amplitude of the correlation coefficients decreases. But this is true for all hypotheses. Yet, this is not a drawback since the proposed evaluation metrics rely on the normalized Euclidean distances within the cluster and thus are agnostic to this effect. The window function used with the proposed method simply approaches the probability density function of the correlation coefficients. We have defined its width as the number of samples captured from a cycle of the target and the windows are normalized so that the sum of their coefficients is 1. Therefore, it should be noted that as the width of the window decreases, the proposed selection rule approaches the behavior of CPA+argmax.

Identifying temporal dependencies within the correlation matrix has interesting implications. For example, it is possible to compute its Fourier transform and perform the hypothesis selection over a defined frequency band. However, this requires additional assumptions which are not necessary for the proposed method. The proposed approach is of special interest when dimension reduction/projection is not an option. In fact, the proposed technique can be seen as a projection of the post-processing results.

In our work, we used a dataset of power traces from an actual RPA setup. We have also created a synthetic dataset of traces that approach the characteristics of RPA scenarios where misalignment is found. We have also used a dataset of ChipWhisperer traces. These data may be found in the associated artifact.

Acknowledgments

The first, third and fourth authors are supported by grant 2021 SGR 00115 from the Government of Catalonia, by the projects ACITHEC PID2021-124928NB-I00 and MATSE PID2024-156636NB-C22 funded by MCIN/AEI/10.13039/501100011033/FEDER, EU, and by the project HERMES, funded by the European Union NextGenerationEU/PRTR via INCIBE. The second author is supported by the French Agence Nationale de la Recherche through the grant ANR-22-CE39-0008 (Project PROPHY).

References

- [BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors, *Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of

- Lecture Notes in Computer Science*, pages 16–29, Cambridge, Massachusetts, USA, August 11–13, 2004. Springer, Heidelberg, Germany.
- [CKN01] Jean-Sébastien Coron, Paul C. Kocher, and David Naccache. Statistics and secret leakage. In Yair Frankel, editor, *FC 2000: 4th International Conference on Financial Cryptography*, volume 1962 of *Lecture Notes in Computer Science*, pages 157–173, Anguilla, British West Indies, February 20–24, 2001. Springer, Heidelberg, Germany.
- [CRR03] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 13–28, Redwood Shores, CA, USA, August 13–15, 2003. Springer, Heidelberg, Germany.
- [DJJW25] Elena Dubrova, Sönke Jendral, Yanning Ji, and Ruize Wang. The Weighted Sum Correlation Analysis. Cryptology ePrint Archive, Paper 2025/1250, 2025.
- [DMCB98] William P. Dunlap David M. Corey and Michael J. Burke. Averaging correlations: Expected values and bias in combined pearson rs and fisher’s z transformations. *The Journal of General Psychology*, 125(3):245–261, 1998.
- [DPRS11] Julien Doget, Emmanuel Prouff, Matthieu Rivain, and François-Xavier Standaert. Univariate side channel attacks and leakage modeling. *Journal of Cryptographic Engineering*, 1(2):123–144, August 2011.
- [FGGLN25] Oriol Farràs, Vincent Grosso, Miquel Guiot, and Carlos Andres Lara-Nino. Improving the selection rule of correlation attacks for remote power analysis. Cryptology ePrint Archive, Paper 2025/1290, 2025. <https://eprint.iacr.org/2025/1290>.
- [Fis15] R. A. Fisher. Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. *Biometrika*, 10(4):507–521, 1915.
- [FTBLN23] Anis Fellah-Touta, Lilian Bossuet, and Carlos Andres Lara-Nino. Combined internal attacks on SoC-FPGAs: Breaking AES with remote power analysis and frequency-based covert channels. In *2023 IEEE European Symposium on Security and Privacy Workshops*, pages 281–286, Delft, Netherlands, July 03–07, 2023. IEEE Computer Society Press.
- [FTBLN24] Anis Fellah-Touta, Lilian Bossuet, and Carlos Andres Lara-Nino. A lightweight non-oscillatory delay-sensor for remote power analysis. In *2024 IEEE International Symposium on Hardware Oriented Security and Trust*, pages 343–348, Tysons Corner, VA, USA, May 06–09, 2024. IEEE.
- [GBTP08] Benedikt Gierlichs, Lejla Batina, Pim Tuyls, and Bart Preneel. Mutual information analysis. In Elisabeth Oswald and Pankaj Rohatgi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2008*, volume 5154 of *Lecture Notes in Computer Science*, pages 426–442, Washington, DC, USA, August 10–13, 2008. Springer, Heidelberg, Germany.
- [GDTL21] Joseph Gravellier, Jean-Max Dutertre, Yannick Tégia, and Philippe Loubet-Moundi. SideLine: How delay-lines (may) leak secrets from your SoC. In Shivam Bhasin and Fabrizio De Santis, editors, *COSADE 2021: 12th International Workshop on Constructive Side-Channel Analysis and Secure Design*,

- volume 12910 of *Lecture Notes in Computer Science*, pages 3–30, Lugano, Switzerland, October 25–27, 2021. Springer, Heidelberg, Germany.
- [GHT05] Catherine H. Gebotys, Simon Ho, and C. C. Tiu. EM analysis of Rijndael and ECC on a wireless Java-based PDA. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 250–264, Edinburgh, UK, August 29 – September 1, 2005. Springer, Heidelberg, Germany.
 - [JZ23] Chengbin Jin and Yongbin Zhou. Enhancing non-profiled side-channel attacks by time-frequency analysis. *Cybersecurity*, 6(15), August 2023.
 - [KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO’99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Heidelberg, Germany.
 - [Koc96] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO’96*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Heidelberg, Germany.
 - [LCC⁺06] Thanh-Ha Le, Jessy Clédière, Cécile Canovas, Bruno Robisson, Christine Servière, and Jean-Louis Lacoume. A proposition for correlation power analysis enhancement. In Louis Goubin and Mitsuru Matsui, editors, *Cryptographic Hardware and Embedded Systems – CHES 2006*, volume 4249 of *Lecture Notes in Computer Science*, pages 174–186, Yokohama, Japan, October 10–13, 2006. Springer, Heidelberg, Germany.
 - [LCSL07] Thanh-Ha Le, Jessy Clédière, Christine Servière, and Jean-Louis Lacoume. Efficient solution for misalignment of signal in side channel analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP 2017*, pages II–257–II–260, Honolulu, HI, USA, April 15–20, 2007. IEEE.
 - [LPR13] Victor Lomné, Emmanuel Prouff, and Thomas Roche. Behind the scene of side channel attacks. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 506–525, Bangalore, India, December 1–5, 2013. Springer, Heidelberg, Germany.
 - [LPR⁺14] Victor Lomné, Emmanuel Prouff, Matthieu Rivain, Thomas Roche, and Adrian Thillard. How to estimate the success rate of higher-order side-channel attacks. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems – CHES 2014*, volume 8731 of *Lecture Notes in Computer Science*, pages 35–54, Busan, South Korea, September 23–26, 2014. Springer, Heidelberg, Germany.
 - [Man04] Stefan Mangard. Hardware Countermeasures against DPA – A Statistical Analysis of Their Effectiveness. In Tatsuaki Okamoto, editor, *Cryptographers’ Track at the RSA Conference*, volume 2964 of *Lecture Notes in Computer Science*, pages 222–235, Heidelberg, Germany, February 2004. Springer.
 - [MOP08] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks: Revealing the secrets of smart cards*. Springer, Heidelberg, Germany, Berlin-Heidelberg, Germany, first edition, 2008.

- [MOS09] Stefan Mangard, Elisabeth Oswald, and Francois-Xavier Standaert. One for all - all for one: Unifying standard DPA attacks. Cryptology ePrint Archive, Report 2009/449, 2009. <https://eprint.iacr.org/2009/449>.
- [OP13] David Oswald and Christof Paar. Improving side-channel analysis with optimal linear transforms. In Stefan Mangard, editor, *International Conference on Smart Card Research and Advanced Applications - CARDIS 2013*, pages 219—233, Graz, Austria, November 28–30, 2013. Springer Cham.
- [OSB99] Alan V. Oppenheim, Ronald W. Schaffer, and John R. Buck. *Discrete-Time Signal Processing*. Prentice Hall, Upper Saddle River, NJ, 1999.
- [PG95] Karl Pearson and Francis Galton. Vii. note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58(347-352):240–242, 1895.
- [PRB09] Emmanuel Prouff, Matthieu Rivain, and Regis Bevan. Statistical analysis of second order differential power analysis. *IEEE Transactions on Computers*, 58(6):799–811, January 2009.
- [SGMT21] Falk Schellenberg, Dennis R. E. Gnad, Amir Moradi, and Mehdi B. Tahoori. An inside job: Remote power analysis attacks on FPGAs. *IEEE Design & Test*, 38(3):58–66, 2021.
- [SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. A stochastic model for differential side channel cryptanalysis. In Josyula R. Rao and Berk Sunar, editors, *Cryptographic Hardware and Embedded Systems – CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 30–46, Edinburgh, UK, August 29 – September 1, 2005. Springer, Heidelberg, Germany.
- [SMY09] François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany.
- [VGS13] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Security evaluations beyond computing power. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 126–141, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany.
- [ZS18] Mark Zhao and G. Edward Suh. FPGA-based remote power side-channel attacks. In *2018 IEEE Symposium on Security and Privacy*, pages 229–244, San Francisco, CA, USA, May 21–23, 2018. IEEE Computer Society Press.

A Attacks’ correctness

In Figs. 14 and 15 we provide results for the success rates of CPA on the synthetic traces introduced in Section 5. In these experiments we do not use any performance indicators. This complicates conducting any comparison, beyond intuitive visual analyses. However, we provide these data to illustrate the actual success rates of the attacks.

From these results, some readers might appreciate a slight advantage of the `meanmax` selection rule. We have quantified this improvement in Section 5 by measuring the divergence of the correct hypothesis. We note that counterintuitively in some cases CPA+`meanmax` is correct when CPA+`argmax` is not.

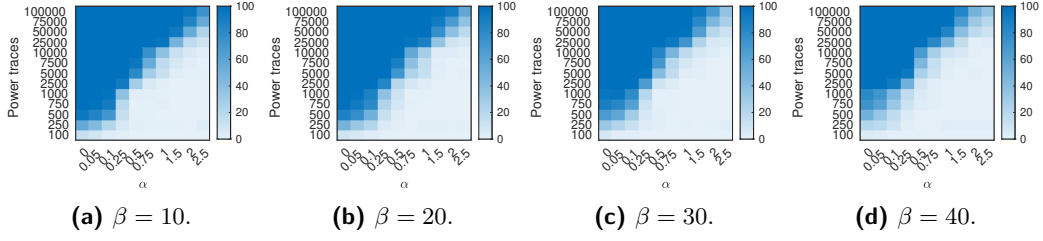


Figure 14: Percentage of bytes retrieved correctly by CPA+argmax in function to the number of traces (y), the noise gain (α), and the misalignment rate (β).

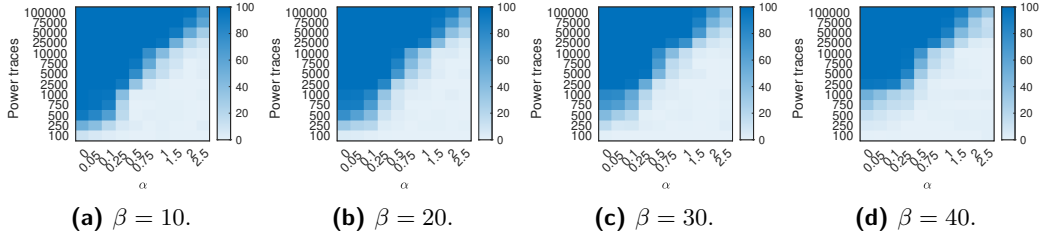


Figure 15: Percentage of bytes retrieved correctly by CPA+meanmax in function to the number of traces (y), the noise gain (α), and the misalignment rate (β).

Similarly, we provide attack-correctness results for the experiments in Section 6.

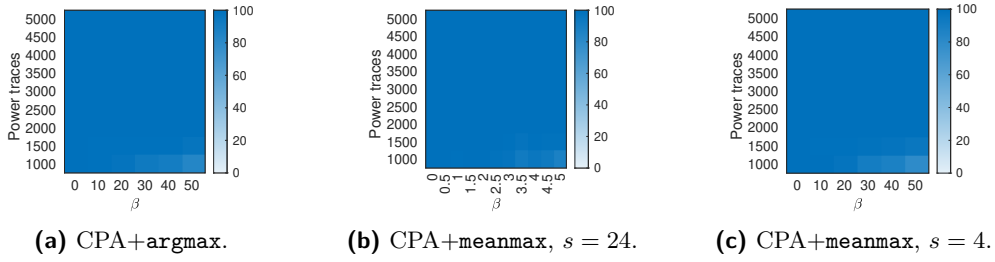


Figure 16: Percentage of bytes retrieved correctly in function to the number of traces (y) and the misalignment (β). For CPA+meanmax we provide results using windows of different widths (s).

Turn Your Face Into An Attack Surface: Screen Attack Using Facial Reflections in Video Conferencing

Yong Huang¹, Yanzhao Lu¹, Mingyang Chen¹, En Zhang¹, Jiazi Li¹, and Wanqing Tu^{2*}

¹Zhengzhou University, ²Durham University

yonghuang@zzu.edu.cn, yanzhaolu@gs.zzu.edu.cn, chenmingyangzzu@gmail.com,
{zzu202278030528, ljz202278040309}@stu.zzu.edu.cn, wanqing.tu@durham.ac.uk

Abstract

In video conferencing, human faces serve as the primary visual focal points, playing multifaceted roles that enhance visual communication and emotional connection. However, we argue that a human face is also a side channel, which can unwittingly leak on-screen information through online video feeds. To demonstrate this, we conduct feasibility studies, which reveal that, illuminated by both ambient light and light emitted from displays, the human face can reflect optical variations of different on-screen content. The paper then proposes FaceTell, a novel side-channel attack system that eavesdrops on fine-grained application activities from pervasive yet subtle facial reflections during video conferencing. We implement FaceTell in a real-world testbed with three different brands of laptops and four mainstream video conferencing platforms. FaceTell is then evaluated with 24 human subjects across 13 unique indoor environments. With more than 12 hours of video data, FaceTell achieves a high accuracy of 99.32% for eavesdropping on 28 popular applications and is resilient to many practical impact factors. Finally, potential countermeasures are proposed to mitigate this new attack.

1 Introduction

Since the recent COVID-19 pandemic, video conferencing has become an essential tool for remote work, education, social connections, etc. It is reported that the count of daily active Zoom users has reached 300 million and yielded over 3.3 trillion meeting minutes in 2024 [16]. However, with the unprecedented growth in video calls, participants often multitask, such as reading news, checking email, and using chatting applications, during conferencing to combat Zoom fatigue or boost productivity [24]. The frequent engagement in these secondary activities introduces a new vulnerability, allowing untrusted participants to launch screen attacks on victims' devices remotely. Under such attacks, a malicious party can infer private and confidential information on victims' screens

based on *compromising reflections*, i.e., unintended reflections of light from screens on nearby objects.

While compromising reflections have a long history in the literature, the impact of such threats on emerging video conferencing remains understudied. Previous attacks show that reflections on nearby objects, such as teapots, spoons, eyeglasses, and even eyes, are exploitable using either high-end telescopic lenses [3, 4] or inexpensive commodity cameras [23, 33] in close proximity. The variation in the overall luminance (color or brightness) of the background area in the webcam's Field of View (FoV) is also leveraged to detect on-screen images [31]. Recent work [18, 30] breaks the distance limitation by exploiting eyeglass reflections to infer screen content via a remote webcam in video conferencing. However, these existing attacks are faced with at least one of the following problems. First, the popular virtual background feature in mainstream video conferencing platforms (e.g., Zoom, Teams) can blur both the background and any reflective items. Second, the low image resolution of webcams and the variable network bandwidth render it difficult to infer screen content from participants' eyes. Third, specular reflections (e.g., from eyeglasses) require the observer to be positioned precisely on the propagation path of the reflected light that has the same angle as the incident light [6]. Since online meeting participants do not remain stationary all the time, as we will analyze in § 6.5, eyeglasses do not constantly reflect light from the screen. Moreover, not everyone wears eyeglasses in video conferencing, making eyeglasses an unreliable source for reflection attacks. The above reasons limit the applicability of the existing screen attacks in real-world video conferencing settings.

This paper pushes the boundary of screen attacks during video conferencing by exploiting the more pervasive yet unremarkable facial reflections of screen content. We observe that the human face, the key part of video conferencing images, is illuminated by not only ambient light but also the screen's light, making it a promising stable source of compromising reflections. Although the face is not a polished surface, it generally exhibits a mixture of diffuse and specular reflec-

*Corresponding author.

tions [9]. Due to different angular and distance characteristics, different facial regions, such as the right and left cheeks, the forehead, and the nose, are sensitive to light from different parts of the entire screen (analyzed in § 3). In this way, facial reflections included in online video streams can be employed to characterize applications or software with unique user interface (UI) layouts on the screen. This creates a potential attack surface for eavesdropping on users’ multitasking activities during video conferencing.

This paper proposes FaceTell, a novel eavesdropping system, to leverage subtle facial reflections in online video streams to infer the secondary activities of video conferencing participants. More specifically, a malicious online meeting participant may launch such attacks by sending the video streams of other users to FaceTell, which will then extract facial images to support screen content prediction. In designing this system, we tackle the following challenges.

1) *How to obtain high-quality face images from dynamic online video streams?* Due to limitations in camera capability and variable network bandwidth, the video resolution of online meetings is variable, which typically ranges from 360p to 720p, significantly lower than that of high-end telescopic lens photos. Moreover, human faces occupy only a small portion of the camera’s field of view (FoV) (analyzed in § 4.2), making it challenging to characterize the subtle facial reflection of screen content. To handle this challenge, FaceTell extracts video frames from video conferencing streams and crops raw face images (i.e., eliminates background items) by employing Mask-R-CNN [14] and Haar feature classifier [28]. Then, using the CAMixerSR network [29], FaceTell implements super-resolution reconstructions of the cropped face images. Finally, reconstructed images will be resized and normalized into higher resolutions with fixed sizes.

2) *How to accurately infer screen content from subtle facial reflections?* As mentioned, it is still non-trivial to extract information from facial reflections as they are a mixture of specular and diffuse reflections [9]. To address this difficulty, we formulate screen content inference as a classification task. This classification task enhances the high-level features related to facial reflections by a residual convolutional block (ResBlock) [15] and a convolutional block attention module (CBAM) [32], and devises an effective two-tier classification model for inferring on-screen content categories and applications. The classification results are further improved by a new heuristic algorithm that conducts label correction based on the temporal continuity of human-computer interactions.

Experimental Results. We implement FaceTell in a real-world testbed with three different brands of laptops (i.e., ASUS, Lenovo, HP) and four mainstream video conferencing platforms, including Zoom, Teams, Skype, and WeChat. More than 12 hours of video data from 24 online meeting participants are collected. These 24 participants interact with 28 common computer applications in 13 different environments. The evaluation results demonstrate that FaceTell achieves an

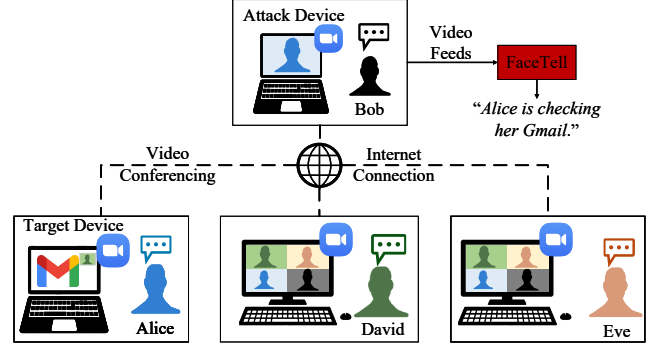


Figure 1: An example of screen-based attacks.

accuracy of 99.32% in application prediction. The accuracy is achieved with a runtime of approximately 124 ms per video frame. Moreover, the experimental results show that FaceTell is robust to impact factors, including subject gender, facial occlusions, distances, angles, and ambient light.

Contributions. Our contributions are summarized below:

- **A New Side-Channel Attack.** This paper is among the first to develop theoretical modeling and real-world experimental analyses to prove the feasibility of a new side-channel attack. This attack exploits subtle facial reflections in online video streams to eavesdrop on the secondary activities of video conference participants.
- **The Novel FaceTell System.** This paper proposes a novel eavesdropping system, FaceTell, that provides an efficient tool to launch this new side-channel attack. The FaceTell system constitutes a set of novel schemes, including a two-tier classification model and a heuristic label correction algorithm, to accurately reveal victims’ private information regarding their secondary activities.
- **Comprehensive Evaluation.** FaceTell is evaluated with 28 popular computer applications, four laptops, four mainstream video conferencing platforms, and 24 victims in 13 different environments. The experimental results demonstrate that FaceTell is highly effective and robust to various potential confounding factors.

2 Threat Model

A typical screen-based attack in video conferencing is illustrated in Figure 1. Suppose that all conference participants use the same video conferencing platform (e.g., Zoom, Teams). As the mutual trust among video conferencing participants is not always guaranteed [7], we assume Bob is a screen-based attacker who wants to eavesdrop on the screen activities of Alice, i.e., the victim. Hereinafter, Alice’s computer is referred to as the target device, and Bob’s computer is called the attack device. Bob’s goal is to identify on-screen activities

by analyzing the light patterns reflected on Alice's face in available online video streams from the target device. Bob can achieve this goal by intercepting the video feeds from the video conferencing platform or recording these video feeds using screen recording software on the attack device.

3 Feasibility Study

3.1 Theoretical Modeling

We first introduce the notations that we employ for developing the theoretical model. Let us denote the location of a digital screen as $\mathbf{E}$ as depicted in Figure 2. An individual light-emitting point on this screen, such as a liquid crystal (LC) unit that acts as a light source, is located at a point $e \in \mathbf{E}$. The orientation of the screen at the location e is defined by its normal vector $\vec{n}_e$. The surface of Alice's face is represented by the coordinate set $\mathbf{F}$. For a single point on the face, we use f to represent its coordinate, where $f \in \mathbf{F}$. Similarly, the normal vector for any point f on the face is denoted as $\vec{n}_f$. The camera is located at the location c . Moreover, we define several key direction vectors to describe the path of light propagation. The emitting direction, representing the light ray traveling from the screen unit at e to the face point at f , is denoted as $\vec{e} = \frac{\vec{f}e}{|\vec{f}e|}$. The viewing direction at the camera location c from the face point at f is $\vec{v} = \frac{\vec{f}c}{|\vec{f}c|}$. The receiving direction, from the face point at f toward the screen unit at e , is defined as $\vec{r} = \frac{\vec{f}e}{|\vec{f}e|}$. In addition, $\vec{m}$ is the mirror direction in which a light ray from the screen unit at e is perfectly reflected off f .

Based on the above notations, we first deduce the light intensity from the screen unit to the face point. The light intensity emitted by the screen unit along $\vec{n}_e$ is denoted as I_e , which is determined by the screen content. Due to the spatial directivity of each LC unit, the light intensity emitted to each angle is different [25, 26]. Concretely, the light intensity from the screen unit to the face point can be calculated as [1]

$$I_f = \frac{I_e \cdot W(\theta_e)}{(d_{ef})^2}, \quad (1)$$

where θ_e is the angle between the emitting direction $\vec{e}$ and the normal vector $\vec{n}_e$, and d_{ef} is the distance between e and f . In addition, $W(\cdot)$ represents the angular distribution of light intensity from the screen unit at e .

Then, we proceed to compute the optical reflection intensity of the face point at f from the entire screen. To do this, we must sum up all the LC units' light emanations. According to the Phong reflection model [21], the reflected light intensity pointing to the viewing direction $\vec{v}$ can be expressed as

$$I_{\vec{v}} = \sum_{e \in \mathbf{E}} I_f \cdot k_d \cdot \cos \theta_r + \sum_{e \in \mathbf{E}} I_f \cdot k_s \cdot \cos^{n_s} \theta_m + k_a \cdot I_a, \quad (2)$$

where k_d , k_s , and k_a are the diffuse, specular, and ambient reflection coefficients, respectively. As illustrated in Figure 2,

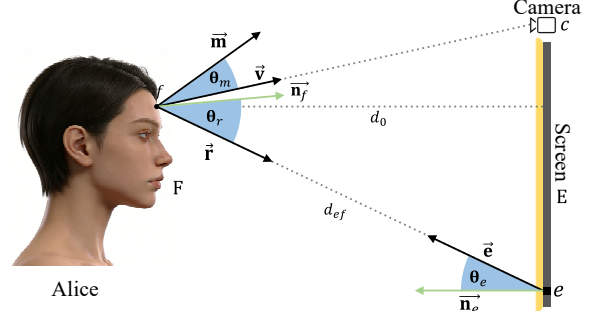


Figure 2: An example to demonstrate the theoretical model of facial reflections in video conferencing.

θ_r is the angle between the receiving direction $\vec{r}$ and the normal vector $\vec{n}_f$, and θ_m is the angle between the mirror direction $\vec{m}$ and the viewing direction $\vec{v}$. The parameter n_s is the shininess exponent, which controls the size and intensity of the specular highlight; a higher n_s value results in a smaller highlight, simulating a glossier surface. The term $\cos^{n_s} \theta_m$ thus models how the specular reflection intensity falls off as the viewing direction diverges from the perfect mirror direction. Additionally, I_a stands for the intensity of ambient light sources. Therefore, in Eq. (2), the first part $\sum_{e \in \mathbf{E}} I_f \cdot k_d \cdot \cos \theta_r$ corresponds to the diffuse reflection, and the second part $\sum_{e \in \mathbf{E}} I_f \cdot k_s \cdot \cos^{n_s} \theta_m$ is for the specular reflection.

Next, we integrate Eq. (1) into Eq. (2). Let us assume that the screen is flat and each LC unit has the same normal vector. Given any screen unit at e , its distance to a face point at f can be expressed as $d_{ef} = \frac{d_0}{\sin(\pi/2 - \theta_e)} = \frac{d_0}{\cos \theta_e}$, where d_0 is the perpendicular line between f and $\mathbf{E}$. In this way, we obtain a new form of the reflected light intensity $I_{\vec{v}}$ as

$$I_{\vec{v}} = \frac{1}{d_0^2} \sum_{e \in \mathbf{E}} I_e \cdot (k_d \cdot G_d(\theta_e, \theta_r) + k_s \cdot G_s(\theta_e, \theta_m)) + k_a \cdot I_a. \quad (3)$$

Therein, $G_d(\cdot, \cdot)$ is the importance weight of the screen unit for diffuse reflections, which is given by

$$G_d(\theta_e, \theta_r) = W(\theta_e) \cdot \cos^2 \theta_e \cdot \cos \theta_r. \quad (4)$$

Moreover, $G_s(\cdot, \cdot)$ is the importance weight of the screen unit for specular reflections, which is expressed as

$$G_s(\theta_e, \theta_m) = W(\theta_e) \cdot \cos^2 \theta_e \cdot \cos^{n_s} \theta_m. \quad (5)$$

The above equations tell us: 1) The intensity of the screen light reflected at each face point is a weighted sum of the light from all screen units. 2) The importance weight of each screen unit on the illuminance of each face point is different and mainly depends on the angular variables in the reflection of light.

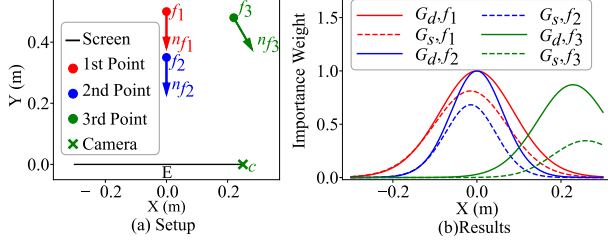


Figure 3: Simulation setup and results.

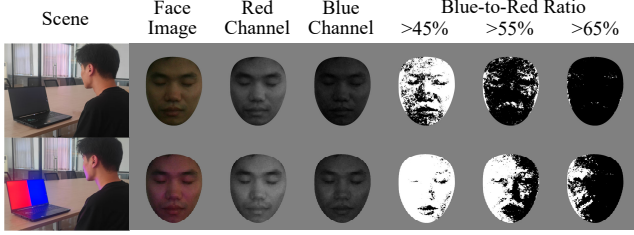


Figure 4: Experimental setup and results.

3.2 Simulation Validation

With the above theoretical model, a Python based simulation is conducted to analyze $G_d(\cdot, \cdot)$ and $G_s(\cdot, \cdot)$ in Eq. (4) and Eq. (5). Specifically, as depicted in Figure 3 (a), the simulation sets up a 2-dimensional plane and places a screen and a camera on the x-axis. Then, three face points with different positions and normal vectors are configured. In $G_d(\cdot, \cdot)$ and $G_s(\cdot, \cdot)$, the simulation adopts $W(\theta_e) = \cos^g \theta_e$ with $g = 30$ [1] and sets the shininess exponent n_s to be 2 [2]. As shown in Figure 3 (b), each curve has a peak value in the diffuse reflection, and the peak value is always located around the screen region directly opposite the face point because the angles θ_e and θ_r are relatively small in this condition. For example, the first face point has relatively high weights on the screen units in the middle area, while the right side of the screen mainly influences the reflection of the third point. In addition, compared with the first point, the second one has a sharper curve, because it is closer to the screen. Similar observations can be found in specular reflections.

The above results indicate that: 1) *The light reflection off any given face point is not uniformly influenced by the entire screen. Instead, it is highly sensitive to a localized "zone" on the screen.* 2) *The shape and intensity of this sensitivity zone are unique to the spatial location of one face point and the type of reflection.*

3.3 Experimental Validation

As depicted in Figure 4, one subject sits in front of one laptop and views a black image and a colorful image on the laptop's screen. The black image is used to simulate the situation where no light is emitted from the screen. The left half of the

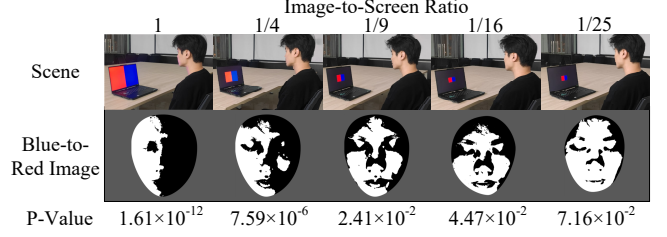


Figure 5: Investigation of minimally differentiable content.

colorful image (1440ps*2560ps) is red, and the right half is blue. In each case, we leverage the laptop's built-in camera to take a photo of the subject. After that, we crop the human face out and obtain a clear face image. As Figure 4 shows, when the subject is faced with the colorful image, his face seems more vivid. To prove this, we extract the red-channel and blue-channel images and observe that they have higher intensity values and become brighter when their faces are in front of the colorful image.

To investigate the impact of different parts of the screen on facial reflections, we show the change in the color composition of each pixel in face images. For this purpose, we compute a blue-to-red ratio for each pixel by dividing the blue intensity by the red intensity. Then, we check whether the blue-to-red ratio is bigger than a certain threshold. If true, the pixel is marked with white; otherwise, it is marked with black. Three thresholds, i.e., 45%, 55%, and 65%, are selected in each case. As Figure 4 depicts, when the subjects view the black image, blue and red colors are more symmetrically distributed on the entire face. The reason is that human faces are mainly illuminated by ambient light in this setting, and the right and left sides of the faces have the same illumination condition. However, when viewing the colorful image, there is blue light distributed on the right-hand side of the subjects' faces and more red light shown on the left-hand side. This is because the right-hand side of their faces is closer to the right half of the screen, thus reflecting more blue light emitted from that half.

We further investigate the minimally differentiable content (MDC) on the screen (as shown in Figure 5), by conducting the Kolmogorov-Smirnov test with a shrunk version of the colorful image. We observe whether the left and right halves of a blue-to-red image follow the same distribution. In each shrunk version, we vary the blue-to-red ratio between 1% and 99% and select a blue-to-red image with the lowest p-value. When the shrunk image accounts for 1/16 of the entire screen (360px*640px), the p-value is 0.0416, smaller than a common significance level of 0.05. When the image-to-screen ratio is 1/25, the p-value is 0.069, larger than 0.05. Thus, the MDC of facial reflections is a rectangle occupying 1/16 of the screen area under the experimental settings.

The above experimental results verify that: 1) *Different facial regions are sensitive to different screen areas.* 2) *Facial*

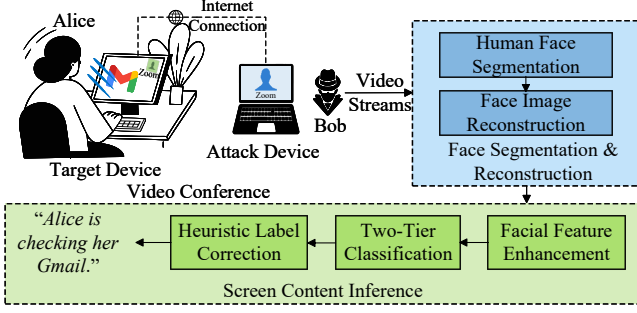


Figure 6: An overview of our FaceTell system.

reflections captured by the webcam can be used to characterize the light patterns of different on-screen contents. Thus, facial reflections of an online meeting participant hold the potential to launch screen attacks during video conferencing.

4 Design of FaceTell

4.1 System Overview

In this section, we propose FaceTell, a novel screen attack system via pervasive yet unremarkable facial reflections during video conferencing. At a malicious participant, FaceTell can eavesdrop on the victim’s multitasking activities without requiring specular objects in the surrounding environment. This is achieved by two phases: a training phase and an attacking phase. In the training phase, FaceTell collects a set of labeled video frames and then trains a dedicated application classifier for screen content inference. In the attacking phase, FaceTell takes the victim’s video frames as input and infers the secondary applications accessed by the victim. As shown in Figure 6, these two phases are implemented by two core components of the FaceTell system: the face segmentation and reconstruction, and the screen content inference. We will elaborate on the two components in the following subsections.

4.2 Face Segmentation and Reconstruction

With the victim’s video feed, FaceTell first segments human face images from raw video frames and enhances their quality using super-resolution reconstruction, as shown in Figure 7.

Human Face Segmentation. In video conferencing, the background within the camera’s FoV always changes as meeting environments vary. More importantly, the virtual background feature is usually enabled when people engage in a video call. The above reasons render background objects less useful for screen attacks in video conferencing. In contrast, human faces are generally the foreground objects within the camera’s FoV, making them reliable reflectors of light from displays. Thus, given a video frame, the first step of FaceTell is to isolate a human face.

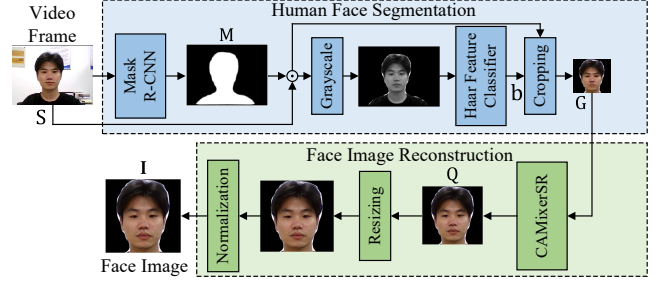


Figure 7: Workflow of face segmentation and reconstruction.

To do this, FaceTell samples a video frame from the received video stream every δ seconds using the OpenCV library. The reason behind this is that adjacent video frames are very similar due to the temporal continuity of screen content, and sampling video frames can reduce computational complexity without significantly impacting screen content inference. In our experiment, δ is set to 0.5 s. Let us denote $\mathbf{V} \in [0, 255]^{3 \times W \times H}$ as one extracted RGB frame, where W and H are the width and height. FaceTell utilizes the mask branch of Mask R-CNN [14], denoted as $\mathcal{F}_{mask}(\cdot)$, to segment all instances of objects shown in $\mathbf{V}$ and generate a set of masks. Therein, $\mathbf{M} \in \{0, 1\}^{W \times H}$ is denoted as a mask matrix for each instance, where $\mathbf{M}(w, h) = 1$ indicates that the pixel (w, h) is part of the instance, and $\mathbf{M}(w, h) = 0$ indicates the background. Because surrounding items, such as desks, closets, and appliances, are also recognized by Mask R-CNN, FaceTell proceeds to pick a human face out of all instances. It achieves this by exploiting the efficient Haar feature classifier [28], which uses a cascade function to identify human faces in images. Specifically, the Hadamard product is performed between $\mathbf{M}$ and $\mathbf{V}$ to obtain a clear image with an instance only. Then, the grayscale version of the product result is fed into the Haar feature classifier $\mathcal{F}_{haar}(\cdot)$ for face detection. The above process can be expressed as

$$\begin{aligned} \mathbf{M} &= \mathcal{F}_{mask}(\mathbf{V}), \\ b &= \mathcal{F}_{haar}(\text{Gray}(\mathbf{V} \odot \mathbf{M})), \end{aligned} \quad (6)$$

where $\text{Gray}(\cdot)$ is the grayscale function and $\odot$ stands for the Hadamard product. b is a Boolean value. If b is true, an instance is detected as a human face. After that, a clear human face image $\mathbf{G} \in [0, 255]^{3 \times M \times N}$, having a width of M and a height of N , without background objects can be extracted as

$$\mathbf{G} = \text{Crop}(b, \mathbf{V} \odot \mathbf{M}), \quad (7)$$

where $\text{Crop}(\cdot, \cdot)$ is the cropping operation that generates the bounding box and crops a face image out when b is true.

Face Image Reconstruction. Due to limitations in camera capability and variable network bandwidth, the video resolution during an online meeting is low and variable. To verify this, we collect an online video stream lasting 25 minutes from Zoom, extract one frame per second, and record the

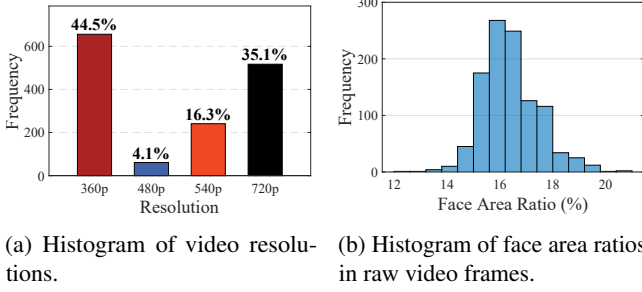


Figure 8: Characteristics of online video streams on Zoom.

frame resolution. As depicted in Figure 8 (a), the video resolution of online meetings is dynamic, ranging from 360p to 720p, and more than 40% of the extracted frames have a resolution of 360p, which is very low compared to high-end telescopic lens photos [3, 4]. What makes things worse is that a human face is only a small part of a video frame. We crop 1,000 face images using the above face segmentation method and compute their area ratios in the raw video frames. As reported in Figure 8 (b), human faces occupy about 16% of pixels in a video frame on average. The above two reasons render it challenging to extract effective information directly using the cropped face images.

To alleviate this problem, FaceTell transforms the low-quality face images into high-resolution versions using the CAMixerSR network [29], an image super-resolution technique that is based on a content-aware mixer that assigns convolution for simple contexts and additional deformable window attention for sparse textures. Taking the face image $\mathbf{G}$ as input, the CAMixerSR network $\mathcal{F}_{sr}(\cdot)$ generates a higher-quality image $\mathbf{Q} \in [0, 255]^{3 \times 2M \times 2N}$ by upscaling the height and the width of $\mathbf{G}$ at two times as $\mathbf{Q} = \mathcal{F}_{sr}(\mathbf{G})$. Because the size of the upscaled image $\mathbf{Q}$ is changing among different frames, FaceTell resizes $\mathbf{Q}$ into a square image with a fixed size of $L \times L$. In addition, the resized image is further normalized using a z-score normalization within each channel for effective hidden feature extraction. The final face image $\mathbf{I} \in [0, 255]^{3 \times L \times L}$ can be obtained by $\mathbf{I} = \text{Norm}(\text{Resize}(L, \mathbf{Q}))$, where $\text{Resize}(\cdot)$ and $\text{Norm}(\cdot)$ represent the resize and normalization operations, respectively.

4.3 Screen Content Inference

As depicted in Figure 9, the second step of FaceTell is to perform screen content inference based on face images.

Facial Feature Enhancement. In this component, FaceTell takes a face image $\mathbf{I}$ as input and generates a high-level feature map. It first leverages a residual convolutional block, which includes a main path and a shortcut connection path, to extract content-related features for effective image feature abstraction. Three convolutional kernels with a size of 3×3 are used in the main path. Their stride and padding size are both set to 1. A

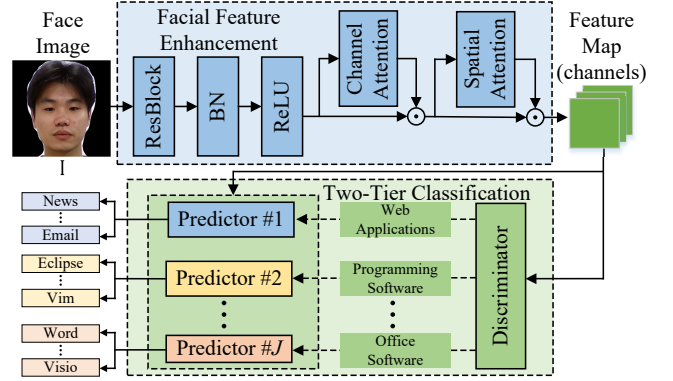


Figure 9: The framework of FaceTell's screen content inference component.

2-dimensional batch normalization (BN) layer and a rectified linear unit (ReLU) are followed for feature representation and non-linearity introduction. In this way, an intermediate feature map $\mathbf{C} \in \mathbb{R}^{3 \times L \times L}$ is obtained.

Next, FaceTell exploits a convolutional block attention module [32] (CBAM) to further enhance its feature representation capability. The CBAM comprises a channel attention submodule and a spatial attention submodule, which are used to capture significant features in the channel and spatial dimensions, respectively. Therein, the channel attention submodule has a max-pooling (MaxPool) branch and an average-pooling (AvgPool) branch. Two branches are followed by a shared multi-layer perceptron (MLP). Their intermediate features are aggregated and fed into a sigmoid function, consequently yielding a channel attention map. When it comes to the spatial attention submodule, it applies average-pooling and max-pooling operations along the channel axis and concatenates their outputs, which are further processed by a convolution layer (Conv) with a kernel size of 7×7 and a sigmoid function to generate a spatial attention map. In the CBAM, the intermediate feature map $\mathbf{C}$ is first fed into the channel attention submodule and multiplied with the channel attention map using the Hadamard product. Then, the product result $\mathbf{C}' \in \mathbb{R}^{3 \times L \times L}$ is processed by the spatial attention submodule in the same way to obtain the enhanced feature map. Mathematically, the final feature map $\mathbf{S} \in \mathbb{R}^{3 \times L \times L}$ after the CBAM can be expressed as

$$\begin{aligned} \mathbf{C}' &= \phi(\text{MLP}(\text{MaxPool}(\mathbf{C})) + \text{MLP}(\text{AvgPool}(\mathbf{C}))) \odot \mathbf{C}, \\ \mathbf{S} &= \phi(\text{Conv}(\text{MaxPool}(\mathbf{C}') \oplus \text{AvgPool}(\mathbf{C}')) \odot \mathbf{C}', \end{aligned} \quad (8)$$

where $\phi(\cdot)$ denotes the sigmoid function and $\oplus$ is the concatenation operation.

Two-Tier Classification. Despite the enhanced feature map, it is still challenging to accurately recognize fine-grained screen content due to the large number of online applications and software. FaceTell addresses this challenge by devising

Algorithm 1 Heuristic Label Correction

Input: Predicted label sequence $\mathcal{Y}$ and its length T

Output: Corrected label sequence $\mathcal{Z}$

```
1:  $t \leftarrow 1$ 
2: if  $t \leq T$  and  $\text{Start}(t) = \text{True}$  then      // a step starts
3:    $\mathbf{z}^t \leftarrow \mathbf{y}^t$       // known content
4:    $t \leftarrow t + 1$ 
5:   if  $\text{End}(t) = \text{False}$  then      // a step continues
6:      $\mathbf{z}^t \leftarrow \mathbf{z}^{t-1}$ 
7:     goto line 4
8:   else      // a step ends
9:     goto line 2
10:  end if
11: else
12:   $\mathbf{z}^t \leftarrow (0, \dots, 0)$       // unknown content
13:   $t \leftarrow t + 1$ 
14:  goto line 2
15: end if
```

a two-tier classification model to first classify screen content into several main categories, i.e., web applications, programming software, office software, etc., and then predict the finer-grained applications in this category. For example, the web application category includes reading news, online shopping, and checking emails. The two-tier classification model achieves this by employing a category discriminator and a set of application predictors.

Specifically, the category discriminator transforms each feature map $\mathbf{S}$ into a category label $\mathbf{g} \in \{0, 1\}^J$, which is the one-hot encoding vector that indicates the category of applications shown on the screen. Therein, J is the category number. The category discriminator consists of three fully connected (FC) layers with 512, 256, and J neurons, respectively, followed by a softmax function to yield a probability vector $\mathbf{p} \in [0, 1]^J$. Then, FaceTell encodes the probability vector $\mathbf{p}$ into the one-hot vector $\mathbf{g}$ corresponding to the j -th category.

After determining the category of screen content, the application predictor belonging to the j -th category is selected and facilitates fine-grained application classification. The selected predictor transforms each feature map $\mathbf{S}$ into a probability vector $\mathbf{q} \in [0, 1]^{K_j}$. K_j is the number of computer applications in the j -th category. For simplicity, each application predictor has the same network architecture as the category discriminator. Similarly, the probability vector $\mathbf{q}$ is converted into an application label $\mathbf{r} \in \{0, 1\}^{K_j}$, which is the one-hot encoding vector that indicates which application the victim is using.

For each face image, we combine its category-application label pair of $\mathbf{g}$ and $\mathbf{r}$ into a unified label $\mathbf{y} \in \{0, 1\}^K$, where $\sum_{j=1}^J K_j = K$, to represent a specific application. Given a conferencing video stream, FaceTell can generate a content label sequence as $\mathcal{Y} = \{\mathbf{y}^1, \dots, \mathbf{y}^t, \dots, \mathbf{y}^T\}$, where $\mathbf{y}^t$ is the estimated label at time t and T is the sequence length.

Heuristic Label Correction. In this component, FaceTell

takes a further step to correct face images that are mistakenly classified. Misclassifications could stem from either inadequate features extracted from a single face image or screen content unseen in the training phase. To deal with this issue, FaceTell proposes a novel heuristic label correction (HLC) algorithm. The correction algorithm is based on the observation that it generally takes a while for users to interact with a computer application, and users are not likely to switch them too frequently. For example, in [5], it reports that web sessions typically last 2 to 3 minutes on average. Thus, we have prior knowledge that, in most cases, the current prediction label estimated by our predictor is likely to be the same as the previous one. Additionally, face images whose labels appear frequently within a period are likely to be correctly classified.

Based on the above observation, a corrected content label sequence can be considered as a step function, which is a piecewise constant function having finite steps. Let us denote the corrected label sequence as $\mathcal{Z} = \{\mathbf{z}^1, \dots, \mathbf{z}^t, \dots, \mathbf{z}^T\}$, where $\mathbf{z} \in \{0, 1\}^K$ could be one of K one-hot vectors for predicted applications or an all-zero vector for other unknown content. Using the above notations, we define two functions regarding the two label sequences $\mathcal{Y}$ and $\mathcal{Z}$ as follows.

Definition 1 (Start of A Step): At time t , let us denote $\mathbf{W}_t = \{\mathbf{y}^t, \mathbf{y}^{t+1}, \dots, \mathbf{y}^{t+T_s-1}\}$ as a segment of the label sequence $\mathcal{Y}$ with a length of T_s . We can determine that a step of $\mathcal{Z}$ starts at the time t or not as

$$\text{Start}(t) = \begin{cases} \text{True}, & \text{if } \frac{\text{Count}(\mathbf{y}^t, \mathbf{W}_t)}{T_s} \geq \sigma_s; \\ \text{False}, & \text{else.} \end{cases} \quad (9)$$

Therein, $0.5 \leq \sigma_s < 1$ is a determining threshold and $\text{Count}(\mathbf{y}^t, \mathbf{W}_t)$ represents the count number of the label $\mathbf{y}^t$ present in the segment $\mathbf{W}_t$. This definition means that when the current label $\mathbf{y}^t$ accounts for the majority of the next T_s labels in the sequence $\mathcal{Y}$, the start of a step in $\mathcal{Z}$ begins.

Definition 2 (End of A Step): At time t , we denote $V_t(\tau) = \{\mathbf{y}^t, \dots, \mathbf{y}^{t+\tau}\}$, where $\tau \geq 0$. It represents a segment of the label sequence $\mathcal{Y}$ with a varying length of $\tau + 1$. An end of a step of $\mathcal{Z}$ is detected or not as

$$\text{End}(t) = \begin{cases} \text{False}, & \text{if } \exists 0 \leq \tau \leq T_e, \text{ s. t., } \frac{\text{Count}(\mathbf{z}^{t-1}, V_t(\tau))}{\tau+1} \geq \sigma_e; \\ \text{True}, & \text{else,} \end{cases} \quad (10)$$

where T_e stands for the maximum window size of $V_t(\tau)$ and $0 < \sigma_e \leq 0.5$ is another determining threshold. $\text{Count}(\mathbf{z}^{t-1}, V_t(\tau))$ represents the count number of the previous step label $\mathbf{z}^{t-1}$ present in the segment $V_t(\tau)$. This definition indicates that when the previous label $\mathbf{z}^{t-1}$ never has a high proportion in any segment of the next T_e labels in $\mathcal{Y}$, the end of a step in $\mathcal{Z}$ is recognized.

Based on the defined functions $\text{Start}(\cdot)$ and $\text{End}(\cdot)$, a heuristic label correction algorithm is proposed in Algorithm 1. Its core idea is to alternately use $\text{Start}(\cdot)$ and $\text{End}(\cdot)$ on the raw

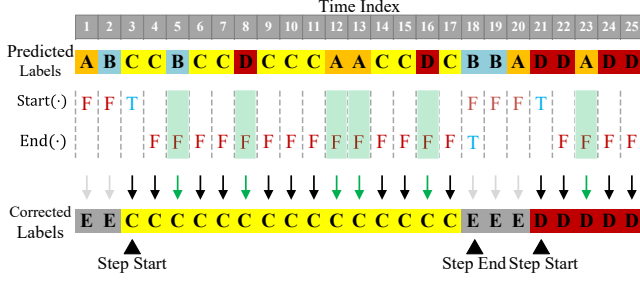


Figure 10: Illustration of heuristic label correction. A, B, C, and D stand for four application labels, and E is for unknown ones. T and F denote True and False, respectively. In this setting, $\sigma_s = 0.6$, $T_s = 8$, $\sigma_e = 0.4$, and $T_e = 6$. A green rectangle means the occurrence of a label correction.

Table 1: Screen content of interest.

Category	Application
Web Applications	Searching, News, Shopping, Email, Social, Sports
Office Software	Word, Excel, Visio, OneNote, PowerPoint, Adobe Reader,
Programming Software	MATLAB, Visual Studio, Eclipse, PyCharm, Vim, Notepad
Multimedia Software	Tik Tok, Disney, Dts Sound, Pandora, Hulu, Netflix, iHeartRadio, Crunchyroll
OS Applications	Files Searching, OS Setting
Video Conferencing	\

label sequence $\mathcal{Y}$ to estimate the time interval of a label step and convert all involving labels in $\mathcal{Z}$ into the same. An illustration of our algorithm is depicted in Figure 10. At the beginning, the function $\text{Start}(\cdot)$ is used to detect the start of a step. If a step starts, we continuously leverage the function $\text{End}(\cdot)$ to identify the end of a step. If the end is identified, the function $\text{Start}(\cdot)$ is called again to find the start of the next step. Moreover, if a predicted label does not belong to any step, we set it to an all-zero vector, indicating unknown content. As demonstrated in Figure 10, the 5th, 8th, 12th, 13th, 16th, and 23rd predicted labels are corrected by our algorithm. In our experiments, we empirically set $\sigma_s = 0.90$, $T_s = 10$, $\sigma_e = 0.10$, and $T_e = 10$. Towards this end, our heuristic label correction algorithm $\mathcal{F}_{\text{correction}}(\cdot)$ outputs a corrected label sequence as $\mathcal{Z} = \mathcal{F}_{\text{correction}}(\mathcal{Y})$.

5 Implementation and Setup

Target and Attack Devices. We use four consumer-grade laptops as target and attack devices in our experiments. Specifically, three of them serve as target devices, responsible for capturing videos of victims. The specifications of the three target devices are as follows: the ASUS TUF A16 is equipped with a USB 2.0 HD UVC WebCam; the Lenovo Y7000P has

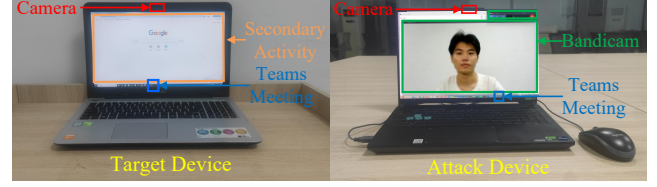


Figure 11: Device setups during data collection.

a 720p HD camera; the HP ProBook 440 is equipped with a 720p HD camera. In addition, a Lenovo Y9000P with a 720p HD camera is used as an attack device to record video streams from the other devices. Note that all devices have an auto-brightness-adjustment feature, which is activated by Windows automatically.

Video Conferencing Platforms. We select four mainstream applications that enable video calls: Skype, Zoom, Teams, and WeChat, and install them on the target and attack devices. These applications have a diverse range of underlying video encoding/decoding protocols and network transmission optimization strategies.

Screen Content of Interest. As shown in Table 1, we select 28 representative computer applications when people multi-task in online meetings. They are divided into five categories, including web applications, office software, programming software, multimedia software, and OS applications. Moreover, we add another category of video conferencing, which indicates that an online meeting participant does not conduct secondary activities.

Data Collection. We recruit 24 subjects, including 14 males and 10 females, to collect online video streams in 13 different environments, containing 3 bedrooms, 7 offices, and 3 dormitories. All subjects are college students, aged from 21 to 25 years old. Some of them wear eyeglasses or face masks. With the 24 subjects, we collect video streams in 24 rounds, resulting in more than 12 hours of video data. In each round, the attack device and one target device are placed in two different rooms and set to join the same online meeting with their cameras activated. One subject is invited to sit within the FoV of the target device to act as a victim. Initially, the subject does not engage in any secondary activity for two minutes. The subject then minimizes the video conference window and sequentially uses each application listed in Table 1 for one minute. We record the usage time of each application for sample annotation. At the same time, the attack device runs Bandicam to capture video streams from the target device, as illustrated in Figure 11. In this way, a 30-minute video is collected in each round.

Datasets. We convert each collected video into a group of image frames with a sampling rate of 2 Hz. Using the proposed face segmentation and reconstruction approach in § 4.2, a clear face image is obtained with a size of 800×800 for each frame. We sort images within each group based on time

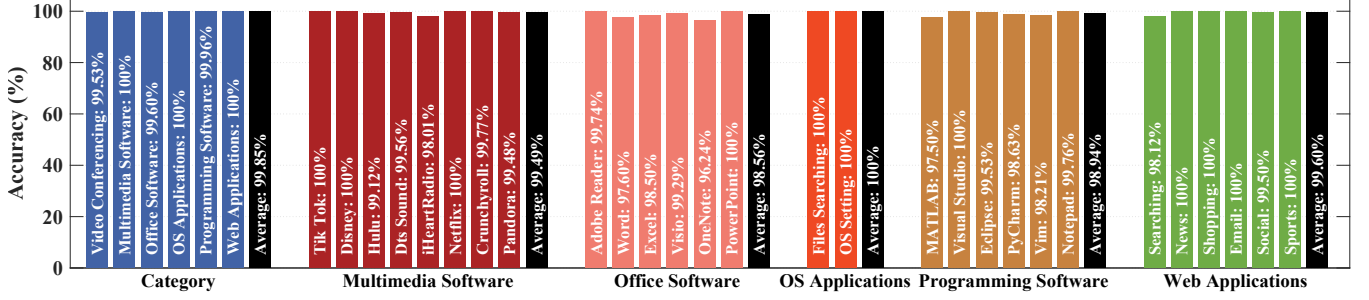


Figure 12: Overall performance of FaceTell.

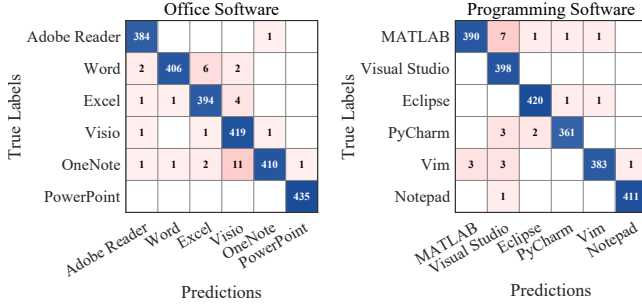


Figure 13: Confusion matrices for office software and programming software.

stamps and annotate them with corresponding category and application labels. In this way, we obtain two disjoint datasets, i.e., a training dataset and a testing dataset. Specifically, the training dataset contains 49,407 face images, and the testing dataset has 12,372 face images.

Training Details. We implement FaceTell on an OMNISK AIX7550-G3 workstation running on Ubuntu 20.04.6TLS with 32GB RAM and 4 NVIDIA GeForce RTX3090 graphics cards. The category discriminator and application predictors are built using PyTorch and optimized using the cross-entropy loss on the training dataset. The Adam optimizer with a learning rate of 1×10^{-4} and a batch size of 16 is adopted. The model is optimized over five epochs.

Evaluation Metric. We adopt accuracy as the main evaluation metric. It is defined as the ratio of correctly classified samples to the total number of samples as

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Samples}}.$$

6 Evaluation Results

6.1 Overall Performance

We first present the overall performance of FaceTell. To do this, we evaluate our system on the testing dataset and report system performance for coarse-grained content category prediction and fine-grained application prediction in Figure 12.

As the figure shows, FaceTell achieves an average accuracy of 99.85% for category prediction. In particular, it obtains a prediction accuracy of 100% in multimedia software, OS applications, and web applications. This is because multimedia software and web applications generally have a more colorful layout, which is significantly different from other applications. As for OS applications, the screen would display a large amount of blank area in the file searching, and mostly shows the system’s default theme color in the OS setting, making them more distinguishable. When it comes to fine-grained application prediction, FaceTell still presents high performance in each category with an average accuracy of 99.32%. It has a relatively low accuracy of 98.56% in office software and an accuracy of 98.94% in programming software. To understand the underlying reasons, we plot their confusion matrices in Figure 13. We find that there are some misclassifications among Word, Excel, Visio, and OneNote in the office software. This is because these applications belong to the Microsoft Office Suite and share many similarities in UI layouts, such as the ribbon interface and quick access toolbar, to provide users with a consistent experience. As for programming software, FaceTell yields some wrong predictions on MATLAB, Visual Studio, Eclipse, PyCharm, and Vim. The reason may be that these applications generally follow a conventional integrated development environment (IDE) layout with a menu bar at the top, a main editor window, and sidebar panels. Despite that, FaceTell still achieves a prediction accuracy of more than 98% in the two categories.

6.2 Ablation Study

We proceed to conduct an ablation study to validate the effectiveness of each core component within FaceTell. For this purpose, we create five variants by systematically ablating specific components. The variants are described below. 1) The first variant removes the CAMixerSR network and the HLC algorithm. 2) The second variant ablates the channel attention submodule in the CBAM block, and the HLC algorithm is also omitted. 3) The third variant discards the spatial attention submodule in the CBAM block and also omits the HLC algorithm. 4) The fourth variant exploits an application

Table 2: Ablation study on FaceTell.

Variants	Category	Multimedia Software	Office Software	OS Applications	Programming Software	Web Applications
#1: w/o CAMixerSR and HLC	89.88%	82.40%	82.81%	90.06%	83.56%	82.72%
#2: w/o Channel Attention and HLC	90.78%	81.49%	82.65%	89.24%	82.37%	81.23%
#3: w/o Spatial Attention and HLC	91.38%	81.99%	83.49%	90.76%	82.91%	82.43%
#4: w/o Two-Tier Classification and HLC	\	74.14%	73.09%	72.66%	72.87%	77.38%
#5: w/o HLC	92.56%	84.44%	84.98%	91.70%	85.09%	84.22%
FaceTell (Complete Model)	99.85%	99.49%	98.56%	100%	98.94%	99.60%

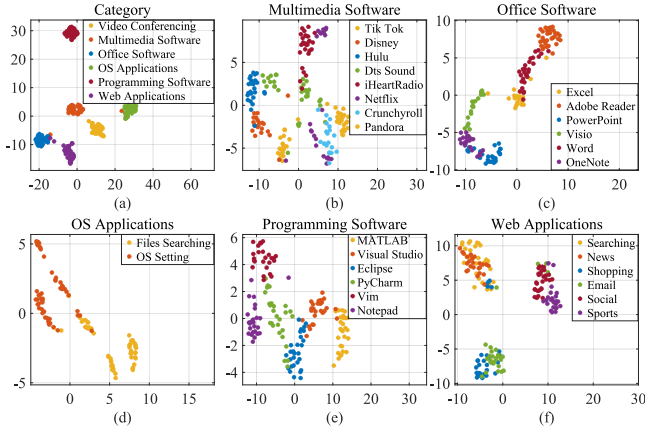


Figure 14: Feature visualization of the category discriminator and application predictors using t-SNE.

classifier to directly recognize the 28 applications without two-tier classification and the HLC algorithm. 5) The fifth variant removes only the HLC algorithm. As Table 2 shows, the accuracy of each variant is at least 7% lower than that of FaceTell in each task, suggesting the effectiveness of each component proposed in FaceTell. Specifically, the first three variants perform lower than the fifth, which indicates that the lack of face image reconstruction or facial feature enhancement will lead to insufficient feature representation of facial reflections and a decrease in accuracy. Moreover, the proposed two-tier classification model is also important to FaceTell. Without it, the fourth variant suffers an accuracy decrease of at least 6.84% when compared to the fifth variant. We can also observe that the label correction algorithm plays an important role in FaceTell and leads to an accuracy increase from 7.29% to 27.34%. The above observation proves that the temporal continuity of human-computer interactions is useful in correcting mistakenly predicted labels.

6.3 Feature Visualization

We exploit the t-distributed stochastic neighbor embedding (t-SNE) technique to visualize the output of the second FC layer of the category discriminator and five application predic-

tors, respectively. In this way, six scatter plots are present in Figure 14, where each point corresponds to a labeled sample. As Figure 14 (a) shows, the category discriminator presents a good separability of different kinds of samples. Six groups of points can be clearly observed. In each group, feature points are highly compact. Moreover, each group is distinct from the others, except for a few outliers. In addition, the points in the video conferencing group are the farthest from those of the others, which tells us that the light variations of video conferencing are quite different from those of the secondary activities. The results indicate that each category has a unique pattern of facial reflections. As for the application predictors, the feature points within different groups are relatively scattered. Some points belonging to different groups are even intermixed, but the overall boundaries can still be roughly observed. This is because the applications within the same category probably have similar UI layouts and generate indistinguishable facial reflections in some cases.

6.4 Time Consumption and Scalability

We report FaceTell’s runtime on the OMNISK AIX7550-G3 workstation. We extract 1,000 video frames from the collected video, feed them into FaceTell, and record the average runtime of the five core components, i.e., human face segmentation, face image reconstruction, facial feature enhancement, two-tier classification, and heuristic label correction, per sample. Compared to the first three components, the two-tier classification and heuristic label correction have a negligible runtime of less than 6 μ s due to their low computational complexity. Moreover, the face image reconstruction has a runtime of 102.50 ms, because the super-resolution operation is relatively time-consuming. Nevertheless, FaceTell has an average runtime of approximately 124 ms per frame. To further demonstrate its scalability with multiple victims, we build a real-time version of FaceTell on a computer with a 62.4 GB memory CPU and a 24 GB memory GPU. With only Zoom activated, the CPU (GPU) usage is 10.4% (0%). If eavesdropping on one Zoom user, FaceTell increases the CPU (GPU) usage to 20.9% (62%). When eavesdropping on two victims simultaneously, the CPU (GPU) usage is 26.8% (98%). With these hardware configurations, FaceTell can at-

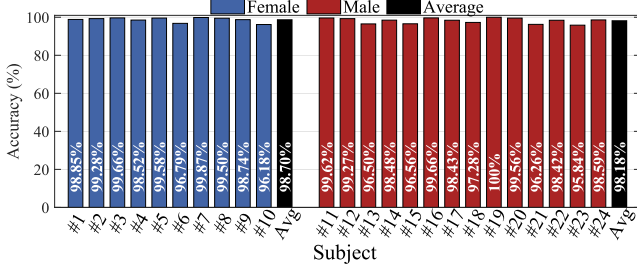


Figure 15: Performance on different subjects.

Table 3: Performance under different occlusion conditions.

Occlusion	Accuracy w/o HLC	Accuracy w/ HLC
No Occlusion	93.31%	99.43%
Eyeglasses	92.62%	99.93%
Face Masks	90.95%	99.64%

tack two victims simultaneously.

6.5 Impact Factors

In this subsection, we investigate FaceTell’s performance under different impact factors, including subject gender, facial occlusions, distances, angles, and ambient light.

Impact of Subject Gender. There are 24 subjects, including 10 women and 14 men, involved in our experiments. The subject’s gender may have an impact on FaceTell, as women and men exhibit different facial features in terms of skin texture, hair characteristics, etc. To understand this impact, we report the average accuracy of the application prediction for all subjects in Figure 15. FaceTell achieves an accuracy of at least 96.18% in the female group and an accuracy of more than 95.84% in the male group. When it comes to overall performance, the average accuracy for female subjects is 98.70%, and that for male subjects is 98.18%, indicating a minimal performance difference between women and men. The above results indicate that the subject’s gender has a negligible impact on FaceTell.

Impact of Facial Occlusions. In practice, facial occlusions may appear during video conferencing and change the facial reflections of participants. To explore this impact, we compute the average accuracy under various facial appearance conditions, including no occlusion, wearing eyeglasses, and wearing face masks, with and without the heuristic label correction. The results are reported in Table 3. Before the label correction, FaceTell achieves the best performance with an accuracy of 93.31% when no occlusion is present, and has the lowest accuracy of 90.95% in the scenario of wearing face masks. To understand the underlying reasons, we showcase face images under the three types of facial occlusions in Figure 16. When no occlusion is shown on the subjects’ faces,



Figure 16: Face images under different occlusion conditions.

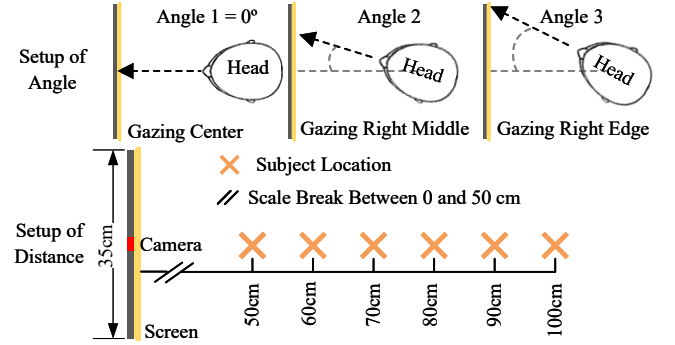


Figure 17: Setups of different distances and angles.

there are many highlighted areas, such as foreheads, cheeks, and noses, that can generate obvious facial reflections. However, when subjects wear eyeglasses, the lenses do not always show recognizable screen content. Since specular reflections have strict requirements on the angles between the screen, the camera, and the lenses, the images on subjects’ eyeglasses sometimes come from the items outside of the screen and weaken the light illumination from the screen, resulting in lower performance. As for face masks, they cover most of the area around the mouth and reduce the area of facial reflections. As a result, the performance of FaceTell is the worst in this case. Despite that, the prediction accuracy of FaceTell exceeds 99% in each scenario after the label correction. We can conclude that FaceTell is robust to different types of facial occlusions in real-world video conferencing scenarios.

Impact of Distance and Angle. As indicated in Eq. (3), the distance and angle between a face point and the screen determine the intensity of the screen light and would impact FaceTell’s performance. Because it is non-trivial to investigate the impact of distance and angle on a face point basis in practice, we choose to study the impact of overall distance and angle between a subject’s face and the screen. To do this, we collect online video streams under different distance and angle settings. As illustrated in Figure 17, one subject is recruited to sit at a fixed location and gaze at the center of the screen, the middle of the right half of the screen, and the right edge of the screen to simulate three angles, i.e., Angle 1, Angle 2, and Angle 3, between the whole face and the screen.

Table 4: Performance under different distances and angles.

Distance (cm)	Angle 1		Angle 2		Angle 3	
	w/o HLC	w/ HLC	w/o HLC	w/ HLC	w/o HLC	w/ HLC
50	91.49	99.76	91.33	99.76	90.31	97.39
60	92.74	98.70	91.40	99.63	90.86	97.79
70	92.75	99.42	92.41	98.64	91.18	98.43
80	91.87	97.24	90.92	97.79	89.08	99.35
90	89.54	98.99	88.89	97.08	86.97	97.56
100	86.01	96.81	85.15	97.23	83.36	97.76



Figure 18: Testbed for investigation of ambient light.

In addition, we vary the distance between the screen and the subject location from 50 cm to 100 cm. The setting ranges of angles and distances conform to people’s daily computer usage habits in practice. In each setting, a 30-minute video is collected and fed into FaceTell for screen content inference. The application prediction performance with and without the HLC algorithm is illustrated in Table 4. In each angle setting, the accuracy stays stable at first and then gradually declines after 70 cm without the label correction. This is because the increase in distance will weaken the intensity of facial reflections and reduce the proportion of a face in the camera’s FoV, rendering FaceTell harder to capture light emanations from the screen. Moreover, in each distance setting, as the angle increases, the accuracy of FaceTell declines. This is because the increase in angle will reduce the face area that can reflect screen light. Despite these, the label correction can significantly improve the accuracy to more than 96.81% in all cases. The above results show the robustness of FaceTell to practical variations in the face-screen distances and angles.

Impact of Ambient Light. As depicted in Figure 18, we build a testbed with a SMART SENSOR AS803 illuminance meter and an adjustable Mi Desk Lamp Pro to collect video data under different intensities of ambient light. First, we use the illuminance meter to measure light illuminance in 5 rooms, including a meeting room, a corridor, a laboratory, a dormitory, and a classroom, under normal lighting conditions. In each room, the illuminance meter is held vertically to simulate the light reception of human faces and record 20 measurements in lux. Then, we plot the histogram of the collected illuminance measurements in Figure 19 and find that facial illuminance ranges from 36 to 131 lux in indoor environments. Then, a subject is asked to sit in front of the

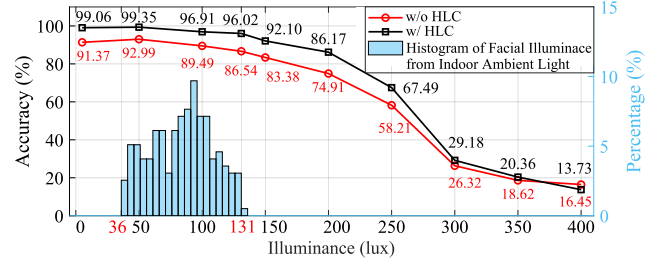


Figure 19: Performance under different intensities of ambient light.

ASUS laptop in an office with closed curtains and turned-off lights. The adjustable lamp is placed 1 meter away from the laptop to serve as the source of ambient light, and the illuminance meter is vertically mounted near the subject’s face and towards the screen. Using this testbed, we simulate facial illuminance of 5, 50, 100, 131, 150, 200, 250, 300, 350, and 400 lux from ambient light. In each illuminance setting, the laptop’s screen is first turned off, and the desk lamp is adjusted to generate a specified illuminance. Figure 19 shows that, as the illuminance of ambient light increases, FaceTell’s performance degrades. This is because facial reflections are a composition of ambient light and screen light as indicated in Eq. (3). Thus, the increase in ambient light will reduce the proportion of screen light in facial reflections, making FaceTell more difficult to extract light variations on the subject’s face. However, in the indoor facial illuminance range, FaceTell still achieves a high accuracy from 96% to 99%. The above results demonstrate that FaceTell has resilience to ambient light in indoor environments.

6.6 Practicality Analysis

According to the experimental results in § 6.5, the performance of FaceTell is significantly influenced by face-screen distance and ambient light intensity. More specifically, as shown in Table 4 and Figure 19, the results indicate that FaceTell can achieve $\geq 96\%$ accuracy when a victim is 50 cm to 100 cm away from the screen and exposed to an ambient facial illuminance < 130 lux. We believe that these settings conform to practical computer usage and indoor illumination. First, the concluded face-screen distance range is in line with the user-screen distance (i.e., 25 inches/63.5 cm) recommended by the American Academy of Ophthalmology to minimize eye strain and prevent visual fatigue [22]. Second, considering that the vertical-to-horizontal illuminance ratio in typical office environments ranges from 0.3 to 0.5 [8], the ambient facial illuminance of 130 lux corresponds to approximately 260 to 430 lux on indoor work planes. This illuminance range includes the recommended light levels of many typical indoor environments, such as homes, coffee break rooms, and classrooms [27].

6.7 Failure Cases

Although FaceTell can obtain high performance in most cases, we report the conditions where our system is limited.

Excessive Face-Screen Distances. We increase the face-screen distances to 120 cm and 150 cm and test the accuracy of FaceTell. The accuracy drops to 75% and 25.65%, respectively. In practice, such excessive face-screen distances may not be realistic as they cause visual discomfort (e.g., eye strain, blurry vision).

High Ambient Light Intensities. As shown in Figure 18, when the facial illuminance from ambient light exceeds 300 lux, FaceTell’s accuracy drops to below 30%. This is because the high ambient light overwhelms the facial optical variations caused by the screen. However, such high-intensity frontal lighting is practically undesirable, as excessive vertical illuminance causes discomfort glare and visual fatigue [34], leading users to naturally avoid these conditions.

New Applications. For the new applications that are not listed in Table 1, FaceTell may experience poor performance. However, if a new application belongs to a defined category in Table 1, FaceTell can recognize its category, and hence guarantee the basic prediction performance for monitoring user activities. To verify this, samples from a new application, YouTube, are collected for testing. The discriminator classifies these samples as multimedia software with an accuracy of 82.88%. For those applications that do not belong to any categories in Table 1, the adaptation only requires fine-tuning the discriminator and predictors, without necessitating a complete retraining of the entire model from scratch.

Dark Mode Settings. Dark mode is a user interface display setting that employs a color scheme with light-colored text and icons on a dark background. By design, this mode significantly reduces the overall light intensity emitted from the screen, thus hampering FaceTell’s performance. To verify this, we collect new data from OS applications under the dark mode setting and test the category discriminator on them. The accuracy of category prediction decreases to 64.13%, which, however, is still much higher than the random guess baseline of 16.67% (1/6). One possible solution to combat this issue is to retrain FaceTell using video data from different color settings of applications.

7 Related Work

Screen Attacks. The foundational work in this domain is established by Backes et al. [4]. By exploiting optical reflections off objects such as teapots, eyeglasses, and even eyeballs, they recover content from an LC screen from a distance of over 30 meters using a telescope. In their subsequent work [3], they improve the quality of images recovered from eyeball reflections to infer on-screen information. Using consumer-grade cameras, Raguram et al. [23] achieve automatic reconstruction of typed input by capturing the key pop-out effect of

virtual keyboards of smartphones in reflections off sunglasses. The work [33] tracks the motion trajectories of a user’s fingers from eyeball reflections to predict human-device interactions.

With the proliferation of video conferencing, the attack vector has shifted from close-proximity observations to online video feeds. Zachary et al. [31] exploit the background area of a webcam’s FoV to detect uniform-color images on the screen. However, the background area can be easily blurred by the virtual background feature. Moreover, this approach is susceptible to human movements. Instead, we choose to exploit indispensable human faces, the foreground area of the FoV. However, the foreground part is dynamic and much smaller than the background part. To address this challenge, FaceTell employs Mask R-CNN and a Haar feature classifier for human face detection and segmentation. Then, the super-resolution CAMixerSR network is applied to enhance the quality of a segmented face image. Both the prior works [18, 30] leverage the foreground eyeglasses as attack surfaces. Private Eye [18] investigates the physical limits of reconstructing textual information from eyeglass reflections in video conferencing. Hassan et al. [30] leverage the power of image processing and computer vision techniques to predict the kind of content displayed on the victim’s screen. However, eyeglasses cannot serve as a reliable source of compromising reflections, because not everyone wears eyeglasses, and eyeglass reflections do not always come from screens. To transcend this limitation, FaceTell exploits non-glossy human faces as attack surfaces. Using human faces is generally more challenging because facial reflections are a mixture of specular and diffuse reflections, making it harder to recognize the screen contents. To tackle this challenge, we devise a dedicated neural network, comprising a CBAM block and an effective two-tier classification model, for content label estimation and propose an HLC algorithm to boost our prediction performance.

Other Side-Channel Attacks. Synesthesia [11] exploits acoustic noise emitted by an LC screen to eavesdrop on screen content in a voice call remotely. Similarly, Ceconello et al. [7] recover device input by analyzing keyboard acoustic emanations transmitted during a Skype call. Homma et al. [13] leverage electromagnetic (EM) leakage from a tablet and reconstruct the screen image from 2 meters away. Ni et al. [20] recover fingerprint images from in-display fingerprint sensors in smart devices via EM signals. WaveSpy [17] probes the state of LC molecules on a screen using the mmWave radar, allowing the eavesdropping of screen content through the wall. AppListener [19] infers mobile app activities by analyzing RF energy harvested from Wi-Fi signals. Fang et al. [10] exploit disturbances in Wi-Fi signals to infer keyboard input. This paper exploits a new side channel, i.e., facial reflections, to eavesdrop on the screen activities of online meeting participants.

Table 5: FaceTell’s performance under challenging scenarios.

Challenging Scenario	Accuracy
Additional Screens	15.09%
Outdoor Environments	13.99%
Skin Smoothing Filters	17.91%

8 Limitations

We conduct further experiments to evaluate FaceTell in three challenging scenarios: 1) using additional screens, 2) operating in outdoor sunny environments, and 3) applying skin smoothing filtering. The results are reported in Table 5.

Using Additional Screens. The attack model of FaceTell has an implicit assumption: the primary dynamic illumination on the victim’s face originates from a single screen. However, a video conferencing participant may have multiple screens, e.g., a programmer may have a multi-monitor setup to improve work efficiency. To investigate FaceTell performance under such interference, we test it using data collected from the victim who faces two screens. The accuracy of the FaceTell attack is 15.09%. The reason is that the additional light source produces overlapping illumination on the victim’s face. Consequently, changes in facial reflections cannot be accurately attributed to a specific screen’s activity, resulting in some misclassifications or unknown categories. We intend to explore this direction in our future work.

Operating in Outdoor Sunny Environments. We extend our evaluation to outdoor environments. We collect facial videos outdoors, where a subject sits in a shady spot on a sunny day (4:20 pm to 5:00 pm). The environmental illuminance is approximately 1300 lux. In this condition, an accuracy of 13.99% is achieved, suggesting that meeting outdoors can effectively withstand “FaceTell” attacks.

Applying Skin Smoothing Filters. During a video conference, a participant may use skin smoothing filters. These filters apply a smoothing effect to the facial regions in video frames. This results in difficulty for FaceTell to obtain application-related optical features. To verify this, FaceTell is tested when the skin smoothing filter is enabled on Zoom. It has a low accuracy of 17.91%.

We hope that by presenting the above limitations of FaceTell, we can help other researchers understand future research directions on screen attacks. Meanwhile, to maximize this possibility, we have open-sourced the code to facilitate subsequent researchers to reproduce our system.

9 Countermeasures

Introducing Additional Dynamic Light Sources. As discussed in § 8, additional dynamic light sources will degrade

FaceTell’s performance. This strategy constitutes a form of active interference at the physical level. A user can place a secondary dynamic light source, e.g., another turned-on display, in front of his face but outside the camera’s FoV. This source generates a task-irrelevant facial illumination and prevents the accurate activity inference of FaceTell.

Enabling Skin Smoothing Filters. As suggested in § 8, enabling skin smoothing filters can significantly worsen FaceTell’s performance and thus is an effective countermeasure to human face-based screen attacks. However, these filters will make a participant’s face look less realistic, causing some participants to be reluctant to enable them to preserve an authentic communication experience.

Adding Adversarial Perturbations. To generate effective and imperceptible software-level filters, adversarial perturbations are a promising solution. They bring minimal changes to input images and can steer a classifier away from true predictions [12]. Thus, dedicated adversarial perturbations can be generated based on our discriminator and predictors and added to facial regions before sending video feeds to other online meeting participants. In this way, the perturbed video frames are received by FaceTell and would cause wrong predictions regarding the secondary activities of other participants.

10 Conclusion

In this paper, we present a new side channel to launch screen attacks using facial reflections in video conferencing. To validate the feasibility, we propose FaceTell, the first automated system that exploits pervasive yet subtle facial reflections to eavesdrop on the secondary activities of online meeting participants. Experimental results show that FaceTell has a high application prediction accuracy of 99.32% for 28 computer applications. Also, FaceTell has an average runtime of about 124 ms for each inference. We also evaluate the resilience of FaceTell to different practical impact factors, including subject gender, facial occlusions, distances, angles, and ambient light. Potential countermeasures are discussed to mitigate the FaceTell attack. We expect that our research results can increase public awareness and stimulate deeper research on this side-channel attack, as well as new defense strategies.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China with Grant 62301499 and the Henan Association for Science and Technology with Grant 2025HYTP037. Wanqing Tu’s work is supported by the EP-SRC and DSIT through the Communications Hub CHEDAR (grant numbers EP/X040518/1 and EP/Y037421/1), and the European Union’s Horizon Europe research and innovation programme grant “SENSORBEES” (agreement no. 101130325).

Ethical Considerations

Our research explores a novel side-channel attack vector in video conferencing. We acknowledge the dual-use nature of this work and the ethical sensitivities inherent in any study involving human subjects and potential vulnerabilities. From its inception, this project was designed with a proactive ethical framework to ensure our methods were sound, our participants were protected, and our findings would contribute positively to the security community. All procedures were conducted in strict accordance with the ethical principles of the Belmont Report and our institution’s policies.

.1 Study Justification and Societal Benefit

The primary motivation for this research is to proactively identify and understand a previously unknown privacy risk in ubiquitous video conferencing systems. By demonstrating the feasibility of the “FaceTell” attack, our goal is to alert developers, platform providers, and the public to this threat. The principal societal benefit of this work is to enable the timely development of countermeasures, thereby protecting millions of users from potential eavesdropping attacks. We argue that the benefit of raising awareness and fostering defensive innovation significantly outweighs the risks associated with the disclosure of this attack method.

.2 Institutional Review Board (IRB) Approval

We have prioritized ethical considerations at every stage of our research. The complete research protocol—encompassing participant recruitment, the informed consent process, data management, and our privacy protection plan—was formally reviewed and approved by our institution’s Institutional Review Board (IRB).

.3 Data Minimization and Protection

Our data handling practices were guided by the principles of data minimization and robust security.

- **Minimization:** We collected only the data essential for validating our attack model—specifically, facial video feeds during scripted application use. No audio was recorded, and no personal or personally identifiable information beyond the facial data itself was collected.
- **Anonymity and Protection:** To safeguard participant privacy, all collected video data was immediately de-identified by assigning a non-identifiable participant ID. The raw and processed data were stored on encrypted, access-controlled servers, with access strictly limited to the core research team. Furthermore, to rigorously protect the privacy of our volunteer participants, all facial images illustrated in this publication are from the authors.

.4 Risk/Benefit Analysis and Responsible Disclosure

We detail the risks and benefits associated with this study by answering the following questions:

1) Who may benefit from the FaceTell attack and how they will be harmed by the disclosure of the proposed countermeasures?

Malicious actors, such as hackers, would have benefited from the undefended state of the FaceTell attack. Hackers could use this attack to gain unauthorized access to sensitive information, such as financial data, personal identities, and business secrets.

With the disclosure of the countermeasures proposed in § 9, these malicious actors will find it much more difficult to execute such attacks. They will have to invest significant time and resources to find new ways to bypass these defenses.

2) Who may be harmed by the FaceTell attack and what are their benefits from the countermeasures?

Millions of individual users who participate in video conferences are at risk of having their private information eavesdropped on. This includes personal interests, health information, and political inclination.

The development of countermeasures (described in § 9) will protect their privacy and prevent potential identity theft and financial fraud. They can participate in video conferences with greater peace of mind, knowing that their multitasking behaviors are unknown to others.

3) What may need to change for the countermeasures to be implemented and how will that affect people?

Three countermeasures are proposed in § 9. We analyze each countermeasure separately.

- **Introducing Additional Dynamic Light Sources.** Users need to place an extra dynamic light source, e.g., a second screen, outside the camera’s FoV. This setup should be easy and low-cost. However, more screens may accelerate visual fatigue during online meetings.
- **Enabling Skin Smoothing Filters.** Users just have to turn on filtering features during video conferencing. While these features are easy to use, some video conferencing participants may be reluctant to use them as the smoothed faces may cause issues relating to authenticity.
- **Adding Adversarial Perturbations.** The owners of video conferencing platforms need to conduct significant technical development to develop and integrate this feature. Once developed, it can be an automatic defense. But the development takes time and may consume extra computational resources, impacting device performance and power consumption.

In conclusion, the publication of our research on the FaceTell attack will encourage the development of countermeasures that protect millions of users from potential eaves-

dropping attacks. While there are risks associated with the disclosure of the attack method, the long-term benefits of raising awareness and fostering defensive innovation clearly outweigh these risks.

.5 Informed Consent Documentation

We recruited 24 adult volunteers who provided their written informed consent. To ensure transparency and full understanding of the risks, participants were required to review and affirm a detailed pre-experimental questionnaire. The specific items affirmed by participants were as follows:

1. Do you confirm that your participation is entirely voluntary and that you are an adult providing your own consent?
2. Do you consent to the recording of your facial video stream while using specific applications on the computer during video conferencing?
3. Do you confirm that no private audio, keyboard input, or personally identifiable information (PII) other than your visual presence was recorded?
4. Do you understand that all raw video data will be immediately de-identified (assigned a non-identifiable ID) and stored on an encrypted, access-controlled server?
5. Do you understand that the full, sensitive video dataset collected from the subjects will not be made public?
6. Do you authorize the use of cropped, processed, and low-resolution images of your face in the publication figures for scientific illustration?
7. Do you understand that the primary purpose of using these images is to demonstrate the visual patterns of light reflection and not to reveal your personal identity?
8. Do you understand that consenting to the use of your images will contribute to the validation of a crucial defensive security finding (in accordance with the Beneficence principle)?
9. Do you understand that we cannot use your images for other purposes without your consent?
10. Do you understand that it is our responsibility to maintain your data integrity and protect it from security threats?
11. Do you understand that we will inform you in a timely manner if any security issues affect your data?
12. Do you understand that you have the right to withdraw from the study at any time, for any reason, without suffering penalty or loss of benefits?

Open Science

We are committed to the principles of open and reproducible research. To support the verification of FaceTell and allow other researchers to build upon our work, we provide a comprehensive artifact package that includes our source code, pre-trained models, and a sample dataset.

.1 Artifact Availability

Our artifacts are available for review via a Zenodo link. This package allows for the replication of FaceTell as described in the paper. The link is: <https://zenodo.org/records/16925324>.

.2 Artifact Description

The provided artifact package is organized into the following components, with detailed instructions available in the `README.md` file:

- **Source Code:** We provide all Python scripts for the two main components of our pipeline:
 1. *Face Segmentation and Reconstruction:* It includes scripts for video processing, face detection, and super-resolution.
 2. *Screen Content Inference:* It includes the complete code for training our two-tier classification model, for testing new data using the provided pre-trained models, and for the heuristic label correction algorithm.
- **Pre-trained Models:** To facilitate immediate testing and verification of FaceTell, we provide the full set of pre-trained models described in the paper. This includes the Tier-1 category discriminator and all specialized Tier-2 application predictors.
- **Dataset:** The full dataset, collected from 24 human subjects, contains sensitive facial video information. In accordance with our IRB-approved protocol and our ethical commitment to protect participant privacy, the full dataset cannot be made publicly available. To address this limitation while still supporting reproducibility, we provide a sample dataset. This dataset contains data from a single participant who fully understands the study's implications and has explicitly consented to their data being used for this specific purpose. This sample data is structured identically to our full dataset and is sufficient to run the entire code pipeline, from data processing to final classification, allowing reviewers to verify the functionality and methodology of FaceTell.

.3 Reproducibility Statement

We believe that the combination of our detailed paper, the complete source code, the pre-trained models, and the sample dataset provides sufficient materials for the research community to verify our core claims and reproduce our methodology.

References

- [1] The radiometry of light emitting diodes. Technical guide, Labsphere Inc., 2001. <https://www.labsphere.com/wp-content/uploads/2021/09/Radiometry-of-Light-Emitting-Diodes.pdf>.
- [2] Faisal R Al-Osaimi, Mohammed Bennamoun, and Ajmal Mian. Illumination normalization of facial images by reversing the process of image formation. *Machine Vision and Applications*, 22(6):899–911, 2011. <https://link.springer.com/article/10.1007/s00138-010-0309-5>.
- [3] Michael Backes, Tongbo Chen, Markus Dürmuth, Hendrik PA Lensch, and Martin Welk. Tempest in a teapot: Compromising reflections revisited. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 315–327, 2009. <https://ieeexplore.ieee.org/document/5207653>.
- [4] Michael Backes, Markus Dürmuth, and Dominique Unruh. Compromising reflections-or-how to read LCD monitors around the corner. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 158–169, 2008. <https://ieeexplore.ieee.org/document/4531151>.
- [5] Evan Bailyn. Average Session Duration By Industry, 2022. <https://firstpagesage.com/reports/average-session-duration-by-industry/>.
- [6] Jonathan Balzer and Stefan Werling. Principles of shape from specular reflection. *Measurement*, 43(10):1305–1317, 2010. <https://www.sciencedirect.com/science/article/abs/pii/S0263224110001570>.
- [7] Stefano Cecconello, Alberto Compagno, Mauro Conti, Daniele Lain, and Gene Tsudik. Skype & Type: Keyboard eavesdropping in Voice-over-IP. *ACM Transactions on Privacy and Security*, 22(4):1–34, 2019. <https://dl.acm.org/doi/10.1145/3365366>.
- [8] Maíra Vieira Dias, Ali Motamed, Paulo Sergio Scarazato, and Jean-Louis Scartezzini. Toward proper evaluation of light dose in indoor office environment by frontal lux meter. *Energy Procedia*, 122:835–840, 2017. https://www.sciencedirect.com/science/article/pii/S187661021733148X?ref=pdf_download&fr=RR-2&rr=9ac48880bc91a9b5.
- [9] Bernhard Egger, William AP Smith, Ayush Tewari, Stefanie Wührer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, et al. 3D morphable face models—past, present, and future. *ACM Transactions on Graphics*, 39(5):1–38, 2020. <https://dl.acm.org/doi/10.1145/3395208>.
- [10] Song Fang, Ian Markwood, Yao Liu, Shangqing Zhao, Zhuo Lu, and Haojin Zhu. No training hurdles: Fast training-agnostic attacks to infer your typing. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, page 1747–1760, 2018. <https://dl.acm.org/doi/10.1145/3243734.3243755>.
- [11] Daniel Genkin, Mihir Pattani, Roei Schuster, and Eran Tromer. Synesthesia: Detecting screen content via remote acoustic side channels. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 853–869, 2019. <https://www.computer.org/csdl/proceedings-s-article/sp/2019/666000a853/1dlwjC6dIT6>.
- [12] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv:1412.6572*, 2014. <https://arxiv.org/pdf/1412.6572>.
- [13] Yuichi Hayashi, Naofumi Homma, Mamoru Miura, Takafumi Aoki, and Hideaki Sone. A threat for tablet PCs in public space: Remote visualization of screen images using em emanation. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 954–965, 2014. <https://dl.acm.org/doi/10.1145/2660267.2660292>.
- [14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017. https://openaccess.thecvf.com/content_ICCV_2017/papers/He_Mask_R-CNN_ICCV_2017_paper.pdf.
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. <https://ieeexplore.ieee.org/document/7780459>.
- [16] Naveen Kumar. Zoom user statistics 2024 — market share & revenue, 2024. <http://pages.cs.wisc.edu/~remzi/OSTEP/>.
- [17] Zhengxiong Li, Fenglong Ma, Aditya Singh Rathore, Zhuolin Yang, Baicheng Chen, Lu Su, and Wen Yao Xu. Wavespy: Remote and through-wall screen attack via mmWave sensing. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 217–232, 2020. <https://ieeexplore.ieee.org/document/9152804>.
- [18] Yan Long, Chen Yan, Shilin Xiao, Shivan Prasad, Wenyan Xu, and Kevin Fu. Private Eye: On the limits of textual screen peeking via eyeglass reflections in video conferencing. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pages 3432–3449, 2023. <https://www.computer.org/csdl/proceedings-article/sp/2023/933600a870/10XGUMtuJLa>.

- [19] Tao Ni, Guohao Lan, Jia Wang, Qingchuan Zhao, and Weitao Xu. Eavesdropping mobile app activity via radio-frequency energy harvesting. In *Proceedings of the USENIX Security Symposium (USENIX Security)*, pages 3511–3528, 2023. <https://www.usenix.org/system/files/usenixsecurity23-ni.pdf>.
- [20] Tao Ni, Xiaokuan Zhang, and Qingchuan Zhao. Recovering fingerprints from in-display fingerprint sensors via electromagnetic side channel. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 253–267, 2023. <https://dl.acm.org/doi/10.1145/3576915.3623153>.
- [21] Bui Tuong Phong. Illumination for computer generated pictures. In *Seminal graphics: pioneering efforts that shaped the field*, pages 95–101. 1998. http://www.cs.northwestern.edu/~ago820/cs395/Papers/Phong_1975.pdf.
- [22] Daniel Porter. Digital Devices and Your Eyes, 2025. <https://www.aao.org/eye-health/tips-prevention/digital-devices-your-eyes>.
- [23] Rahul Raguram, Andrew M White, Dibyendusekhar Goswami, Fabian Monrose, and Jan-Michael Frahm. iSpy: automatic reconstruction of typed input from compromising reflections. In *Proceedings of the ACM conference on Computer and Communications Security (CCS)*, pages 527–536, 2011. <https://dl.acm.org/doi/10.1145/2046707.2046769>.
- [24] René Riedl. On the stress potential of videoconferencing: definition and root causes of Zoom fatigue. *Electronic Markets*, 32(1):153–177, 2022. <https://link.springer.com/article/10.1007/s12525-021-00501-3>.
- [25] Linas Svilainis. LED directivity measurement in situ. *Measurement*, 41(6):647–654, 2008. <https://www.sciencedirect.com/science/article/abs/pii/S0263224107000917>.
- [26] Linas Svilainis and Vytautas Dumbrava. Numerical comparison of LED directivity approximation functions for video displays. *Displays*, 31(4-5):196–204, 2010. <https://www.sciencedirect.com/science/article/abs/pii/S0141938210000661>.
- [27] The Engineering ToolBox. Illuminance - Recommended Light Levels, 2004. https://www.engineeringtoolbox.com/light-level-rooms-d_708.html.
- [28] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages I–I, 2001. <https://ieeexplore.ieee.org/document/990517>.
- [29] Yan Wang, Yi Liu, Shijie Zhao, Junlin Li, and Li Zhang. CAMixerSR: Only details need more "attention". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 25837–25846, 2024. https://openaccess.thecvf.com/content/CVPR2024/papers/Wang_CAMixerSR_Only_Details_Need_More_Attention_CVPR_2024_paper.pdf.
- [30] Hassan Wasswa and Abdul Serwadda. The proof is in the glare: On the privacy risk posed by eyeglasses in video calls. In *Proceedings of the 2022 ACM on International Workshop on Security and Privacy Analytics*, pages 46–54, 2022. <https://dl.acm.org/doi/abs/10.1145/3510548.3519378>.
- [31] Zachary Weinberg, Eric Y Chen, Pavithra Ramesh Jayaraman, and Collin Jackson. I still know what you visited last summer: Leaking browsing history via user interaction and side channel attacks. In *2011 IEEE Symposium on Security and Privacy*, pages 147–161, 2011. <https://ieeexplore.ieee.org/document/5958027>.
- [32] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018. https://openaccess.thecvf.com/content_ECCV_2018/papers/Sanghyun_Woo_Convolutional_Block_Attention_EC_CV_2018_paper.pdf.
- [33] Yi Xu, Jared Heinly, Andrew M White, Fabian Monrose, and Jan-Michael Frahm. Seeing double: Reconstructing obscured typed input from repeated compromising reflections. In *Proceedings of the ACM conference on Computer and Communications Security (CCS)*, pages 1063–1074, 2013. <https://dl.acm.org/doi/10.1145/2508859.2516709>.
- [34] Jiahong Zhang, Kunjie Lv, Xiaoqian Zhang, Mingxiao Ma, and Jiahui Zhang. Study of human visual comfort based on sudden vertical illuminance changes. *Buildings*, 12(8):1127, 2022. <https://www.mdpi.com/2075-5309/12/8/1127>.

A Details of Implementation

A.1 Screenshots of Selected Applications

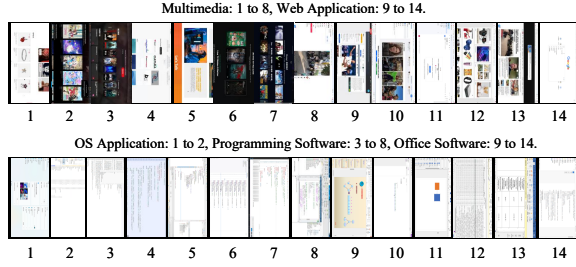


Figure 20: Screenshots of 28 selected applications. Each image is rotated by 90 degrees and concatenated horizontally.

A.2 Subject Information

Table 6: Subject information.

Subject	Gender	Facial Occlusion	Platform	Computer	Location
1	Female	None	WeChat	ASUS	Bedroom 1
2	Female	None	WeChat	Lenovo	Bedroom 2
3	Female	None	Skype	HP	Office 1
4	Female	Eyeglasses	Zoom	ASUS	Office 2
5	Female	Eyeglasses	Zoom	ASUS	Office 3
6	Female	Eyeglasses	Skype	Lenovo	Bedroom 3
7	Female	Eyeglasses	Skype	Lenovo	Office 4
8	Female	Eyeglasses	Teams	HP	Office 5
9	Female	Mask	Teams	HP	Dormitory 1
10	Female	Mask	WeChat	ASUS	Office 2
11	Male	None	Teams	Lenovo	Office 2
12	Male	None	Zoom	ASUS	Office 1
13	Male	Eyeglasses	WeChat	HP	Office 1
14	Male	Eyeglasses	WeChat	Lenovo	Office 1
15	Male	Eyeglasses	WeChat	HP	Office 6
16	Male	Eyeglasses	Skype	Lenovo	Dormitory 2
17	Male	Eyeglasses	Skype	HP	Office 6
18	Male	Eyeglasses	Zoom	Lenovo	Office 6
19	Male	Eyeglasses	Zoom	ASUS	Office 1
20	Male	Eyeglasses	Teams	ASUS	Dormitory 3
21	Male	Eyeglasses	Teams	HP	Office 2
22	Male	Eyeglasses	Teams	ASUS	Office 7
23	Male	Mask	Skype	Lenovo	Office 2
24	Male	Mask	Zoom	HP	Office 1

B Details of Evaluations

B.1 Parameter Determination

We justify the choice of the four parameters in the proposed heuristic label correction (HLC) algorithm. There are four key hyperparameters, i.e., σ_s , T_s , σ_e , and T_e . T_s and σ_s are the window length and decision threshold of $\text{Start}(\cdot)$, and T_e and σ_e are those of $\text{End}(\cdot)$. We evaluate our multimedia

predictor, which has the most prediction labels, on the testing dataset. As depicted in Figure 21, the HLC algorithm can significantly improve the predictor’s performance in each setting. Moreover, the larger σ_s and T_s are, the better the performance is generally achieved. While a small σ_e and a large T_e can also boost classification performance in most cases. After dozens of rounds of tests, we find that the highest accuracy of 99.11% is obtained in our experiments, when $\sigma_s = 0.90$, $T_s = 10$, $\sigma_e = 0.10$, and $T_e = 10$.

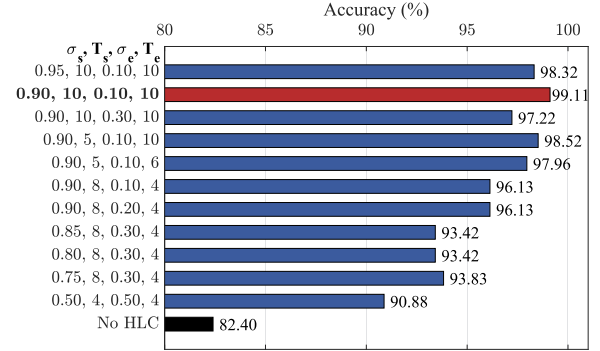


Figure 21: The multimedia predictor’s performance under different HLC parameter settings.

B.2 Impact of Video Conferencing Platform and Computer Hardware

We investigate the impact of two key external factors: variations in video conferencing platforms and computer hardware. In our experiments, four mainstream video conferencing platforms, i.e., Skype, Zoom, Teams, and WeChat, run on three laptops with different brands, including ASUS, Lenovo, and HP. As shown in Figure 22 (a), FaceTell does not show significant performance differences on Zoom, Skype, Teams, and WeChat, and the accuracy on each platform reaches over 99%. The same observation can also be found in Figure 22 (b), where the prediction accuracy on different computer hardware is around 99%. The above results indicate that FaceTell has good stability and generalization capability on different software and hardware platforms.

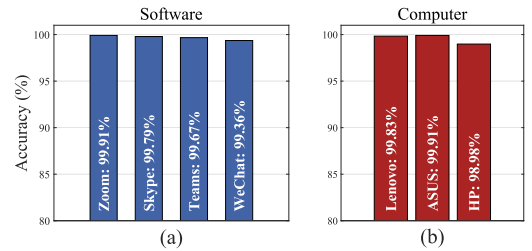


Figure 22: Performance under different video conferencing platforms and computer hardware.

Racing against the clock: side-channel leakage from crossing asynchronous clock domains

Martin Thompson^{1,2} and David Oswald¹

¹ Durham University, Durham, UK, martin.j.thompson@durham.ac.uk,
david.f.oswald@durham.ac.uk

² ZF Automotive UK Ltd, Solihull, UK, martin.j.thompson@zf.com

Abstract. Current physical side-channel attacks are based on direct power or electro-magnetic measurements, or indirect quantities such as precise measurements of clock jitter. On the other hand, timing side-channel attacks exfiltrate secrets by observing secret-dependent variations in runtime caused by branches or micro-architectural effects, *e.g.* due to caching. However, little attention has been paid to the fact that power leakage might also manifest architecturally, for example through variations in cycle counts of clock domain crossings, due to the supply-voltage-dependence of the Voltage-Controlled Oscillators within the on-chip Phase-Locked Loop blocks. In this paper, we show that memory and peripherals that operate in a different clock domain to a target processor can be used to indirectly measure power consumption. This allows power side-channel attacks to be performed from software, without privileged access to direct power measurements and using only a cycle counter.

We first demonstrate that in a (noisy) x86 environment, it is possible to differentiate between instructions (and instruction operands) based on the variation of cycle counts relative to wall-clock time. For an embedded environment (using an NXP i.MX RT1160-family System-on-Chip (SoC)), we similarly show surprisingly large variations in cycle count correlated with the Hamming weight of data. Based on these observations, we hypothesise a leakage mechanism and present evidence from simulation and a Field Programmable Gate Array-based experiment. We then construct a timing-independent covert channel based on this direct proxy for power consumption on the SoC. Going further, we show that it is possible to recover the private exponent from an RSA modular exponentiation as well as an AES key across a Real-time Operating System thread boundary.

Keywords: side-channel analysis · clocks · timing leakage · software-based side channels

1 Introduction

Side-channel analysis based on physical measurements of *e.g.* power consumption and electro-magnetic (EM) emanation has been widely explored in the literature [GMO01; Goo+11; VMC19; MOP10; Lon+15; Kim+19; SM22; Udv+21]. Building on this, in recent years, software-based methods to capture power side-channel leakage have emerged. For example, Lipp et al. demonstrated that low-resolution on-chip power measurements can be exploited to leak secrets across the Intel Software Guard Extensions (SGX) security barrier [Lip+21]. Gnad, Krautter, and Tahoori and O’Flynn and Dewar showed that on-chip peripherals such as analogue-to-digital converters leak significant information about power consumption, which can be exploited across security boundaries such as Trustzone [GKT19; OD19]. Other peripherals that provide indirect measurement of power consumption include delay-lines [Gra+21].

Further, Schoos et al., O’Flynn and Ayoub et al. demonstrated that precise measurement of clock variation (visible as phase-modulation or ‘jitter’) can also be used to infer power consumption [Sch+23; OF124; Ayo+24]. These works use additional peripherals or lab equipment, connected to the target device, in order to make precise measurements of jitter. Considering workload-dependent clocking, or Dynamic Voltage and Frequency Scaling (DVFS), Wang et al. developed an attack that leaks data over longer timescales due to the heating effect on the processor [Wan+22].

While variations in execution time—*e.g.* due to branches, memory access patterns and cache activity—has been extensively studied as a source of leakage [Ber05; BB03], according attacks can be mitigated through “constant-time” software development patterns.

More precisely, we focus on the effect of *power consumption on execution time* in the absence of any dynamic clocking (*i.e.* DVFS) and assuming a target implementation free of (cache)timing leakage. We show that it is possible to measure leakage-in-time related to *short-term power consumption*, allowing us to apply simple clock-cycle timing measurements through the entire literature of power-based side-channels. To our knowledge, we are the first to explore and demonstrate this effect on short time scales.

1.1 Voltage-Dependent Clock Variations

In modern processors and system-on-chips, a Phase-Locked Loop (PLL) is often used to provide the core frequency (at a non-integer multiple of the source rate). PLLs consist of a control-loop, based on a Phase-Frequency Detector (PFD) (and a frequency divider, if a multiplication of the reference frequency is required). The voltage output of these circuits is filtered and fed to the control input of a Voltage-Controlled Oscillator (VCO) (see Figure 1). As the supply voltage V_{supply} varies, each block that is connected to it will undergo changes in electrical response [Abr02]. The VCO is the most sensitive item, resulting in direct, small changes in the output frequency. Every PLL has a parameter known as ‘supply voltage pushing’ which describes the transfer function from changes in supply voltage to changes in VCO frequency [Ban17].

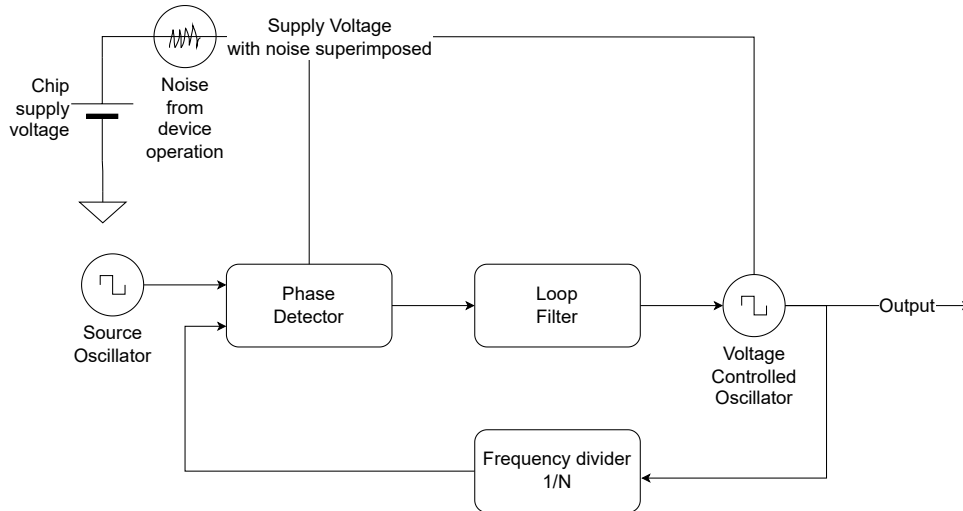


Figure 1: The basic structure of a phase-locked-loop, showing how the supply voltage can influence behaviour

1.2 Related Work

The minute changes in timing between different operations, based on the information provided to a system by an attacker, have been exploited many times. Private keys can be extracted from an undefended asymmetric cryptosystem based solely on the time it takes to perform an operation [Koc96]. Symmetric cryptosystems, particularly AES, are equally vulnerable to differences in timing for different attacker-provided inputs [Kel+98; Per05; Ber05; BM06]. This is due to software implementations of AES depending on one or more look-up tables (either the SBox or T-table) which are stored in memory which the processor accesses via its cache. More recent work goes beyond caches into deeper micro-architectural features [Lip+18; Koc+19], exploiting the fact that modern processor architectures speculatively execute instructions, which leaks information allowing an attacker to infer secrets.

Power analysis attacks use variations of the power consumption over time while a target device performs operations involving a secret. Current flow can be measured directly (by measuring the voltage drop across a series resistor [Ohm27]) or indirectly (by measuring the changes in electro-magnetic field [Max73]). Modern processors provide the ability to measure the short-term power consumption on-chip and make this measurement available to software [Lip+21]. Other mechanisms which can be used to infer power consumption include measurements of jitter on Serialiser/Deserialiser (SerDes) blocks [Sch+23], phase-modulation of the electro-magnetic emissions of a device [Ayo+24] or on the debugger pins [OF124]. Logic in Field Programmable Gate Arrays (FPGAs) can provide measurements of delay (through frequency variations) that are proportional to power consumption [Udu+21; FBL24].

Once measurements of power are available, Simple Power Analysis (SPA) [KJJ99] directly interprets the power trace of the device under attack to deduce information about the secrets being used. Differential Power Analysis (DPA) [KJJ99] separates the power traces according to some prediction of the expected power consumption of the device using a single bit of the postulated secret conditions, and compares the average power consumptions of the two sets. When the guess of the secret is correct, there will be clear differences between the two sets. Absolute Sum Differential Power Analysis (AS-DPA) [Dog+11] develops the DPA concept further by combining the results of DPA on each bit through the sum of the absolute differences of the means. This combines the information available in a form which is robust to inaccurate modelling (as there is no explicit model invoked). Correlation Power Analysis (CPA) [BCO04] makes predictions of hypothetical power consumption, usually according to a Hamming Weight (HW) or Hamming-distance (HD) model, based on the attacker's guesses as to the secret information. A statistical correlation is performed between these predictions and the measured power traces. Those with the highest correlation correspond to the most likely guesses of the secret. Linear Regression Analysis (LRA) [SLP05] does not assume a specific leakage model for the power, but builds a model during a training phase, and uses this model as a distinguisher during the attack phase. The use of Mutual Information Analysis (MIA) to combine leakage across multiple bits has been evaluated [Gie+08]. This technique does not require a training phase, nor make any assumptions about the leakage-model to be used or the distribution of the noise, and has been shown to be effective, especially in the case of targets designed to resist side-channel analysis through physical implementation countermeasures (in this case the use of dual-rail precharge logic).

1.3 Contributions

Continuing in the line of work that measures power leakage through indirect and software-exposed side channels, in this paper we demonstrate that:

- variations in power consumption produce measurable variations in short-term clock frequency of a processor, and that these variations differ between clock domains;
- these small variations can be measured through processor cycle counters, and amplified in the presence of an asynchronously-clocked Random-Access Memory (RAM) block or other asynchronously-clocked peripherals;
- it is possible to exploit these timing changes to construct a covert channel across process boundaries;
- indirect cycle-counter measurements can be used to extract AES and RSA keys using standard power analysis techniques from software, even in the absence of data caching.

Responsible Disclosure We disclosed our findings regarding the leakage of the i.MX RT1166 to the vendor NXP on 11 November 2024. NXP acknowledged and reproduced the leakage, and our findings were discussed in several follow-up calls. We further provided NXP with a copy of the submitted paper. NXP have provided the following comment: *“To ensure authenticated code execution and protect sensitive data, NXP recommends enabling Secure Boot on the i.MX RT1160. Secure Boot establishes a hardware root of trust that validates firmware signatures at startup, preventing unauthorized or tampered code from executing. Additionally, leverage the Cryptographic Acceleration and Assurance Module (CAAM) for supported cryptographic operations, including AES, to keep key material inside secure hardware, reducing exposure and improving resistance to side-channel attacks. Together, Secure Boot and CAAM deliver a robust security foundation by guaranteeing trusted code and safeguarding cryptographic processes throughout the system lifecycle.”* For x86 systems, we did not find directly exploitable leakage that would warrant responsible disclosure at this point.

Data Availability The code and data underlying this paper is available at: <https://doi.org/10.5281/zenodo.18146389>

Outline The remainder of this paper is structured as follows: we begin in Section 2 with our initial investigations into this topic (using an x86 Linux environment and an NXP i.MX RT1166¹ Arm-based embedded System-on-Chip (SoC)), and the results which motivated our further activity. In Section 3 we explain the conjectured mechanism behind the leakage supported by evidence from a VHSIC (Very High Speed Integrated Circuit) Hardware Description Language (VHDL)-based simulation environment. Following this, in Section 4, we show an FPGA-based target to confirm the simulation results in real hardware while retaining strong control over the structure of the experimental platform. Returning to the i.MX RT1166 SoC, in Section 5 we demonstrate the use of a DMA channel (reading a timer) to provide information on the variation of power over time, akin to traditional power traces. From all these results, in Section 6 we demonstrate proof-of-concept attacks in real-world representative circumstances using the i.MX RT1166 SoC. We conclude in Section 7 with a discussion of our results and the implications for system design, including potential countermeasures.

¹<https://www.nxp.com/products/i.MX-RT1160>

2 Initial Exploration

We sought to establish evidence in support of the hypothesis that the clocks available in a processing system will vary differently with power consumption. To this end, we captured clock measurements and the known power measurements available on Intel x86 processors using the Running Average Power Limit (RAPL) interface under various conditions of varying power. We performed these measurements on a PC-based platform (Section 2.1), and later moved to an embedded i.MX RT1166 SoC (Section 2.2).

2.1 Comparison of Clock Variations to Software Measurements on x86

Our first experiments used simple Python code to randomly execute one of two workloads: a high power-consumption workload, `sha`, performed continual SHA-256 operations, and a low power-consumption workload, `spin`, merely waited while performing no additional work. We executed each workload for approximately the same amount of time, as determined by the system clock. We performed this first investigation on an Intel Core i7-7820HQ CPU running at 2.90 GHz running Linux. We disabled dynamic power/frequency scaling to avoid this affecting our results. We captured power readings from the RAPL interface for comparison purposes. For timing measurements, we captured the number of processor ticks from the performance counter (using the RDTSC instruction) and the time difference using the `CLOCK_MONOTONIC` timers. The `CLOCK_MONOTONIC` counter is calibrated by the operating system, using the network Time Protocol (NTP), and therefore provides a reference timesbase against which any variation in the processor tick count can be compared. Using the null-hypothesis that “the mean value of a measurement relating to the power consuming task is no different to that of a random selection of measures”, we perform *t*-tests on each measure [Goo+11] and for each task (*cf.* Table 1).

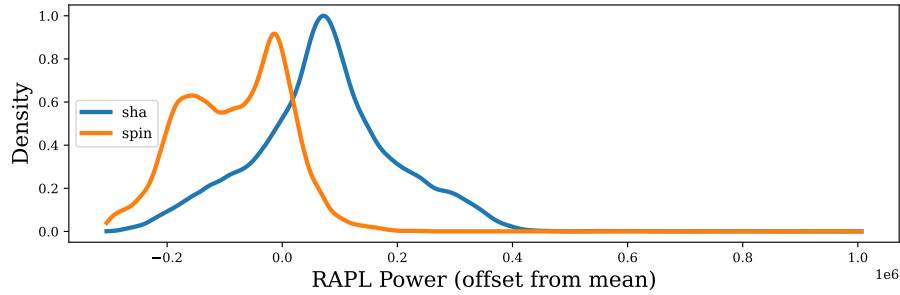
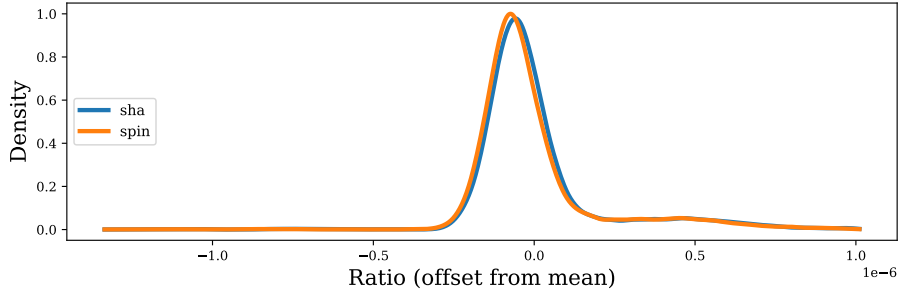


Figure 2: Comparison of activity for RAPL

We see that (unsurprisingly), the RAPL power readings show a very marked difference between the two tasks (*cf.* Figure 2). The leakage of the individual clocks is, as expected, insignificant, but taking the ratio of the two clocks produces $t > 6$, a strong indicator that we should reject the null hypothesis and there is indeed evidence of a correlated leakage (with $p \lesssim 3 \times 10^{-8}$). Plotting a kernel density estimate with a relatively narrow bandwidth (0.1) clearly shows the differences in distribution of the RAPL measurement (Figure 2) and the ratio between RDTSC and `CLOCK_MONOTONIC` (Figure 3). The distributions of the single timing measurements are both similar to each other and between the two processing tasks (Figure 18 in Appendix A), confirming that other power-related timing issues (*e.g.* due to DVFS, *cf.* [Wan+22]) are not present.

Following this, we wrote a program that executes different instructions in a tight loop, each executed for a fixed number of times. We again recorded the `CLOCK_MONOTONIC` clocks and the RDTSC performance counter and executed the test with successive runs of NOP, ADD and IMUL instructions. We then carried out a *t*-test between each pair of instructions to

**Figure 3:** Comparison of activity for RDTSC/CLOCK_MONOTONIC**Table 1:** Comparison of `spin` and `sha`

Measure	<i>t</i> -value
RDTSC/CLOCK_MONOTONIC	-6.44
RAPL power	-163.51
RDTSC	-0.19
CLOCK_MONOTONIC	-0.15

determine if there was a statistically-significant difference between the means (Table 2). The result demonstrates that individual instructions are distinguishable using the ratio of RDTSC and the wall-clock time.

Table 2: Comparison of pairwise *t*-values for different instructions

Measure	IMUL vs. ADD	NOP vs. ADD	NOP vs. IMUL
CLOCK_MONOTONIC/RDTSC	17.87	-22.80	-40.09

Next, we performed an experiment to establish if operands could be distinguished. In this experiment, we executed a fixed length loop of continuous multiplication instructions, with the second operand of the multiplier constant for each run, but selectable for each set of iterations. The initial value of the cumulative multiplication was 3. We gathered data from two runs, with the multiplier constant set to either `0x00000000` or `0xFFFFFFFF` (Table 3). We captured this data using relatively short timescales (each data point was captured over ~ 0.1 sec, approximately 290×10^6 processor ticks).

Although the ratio of the clocks provided the most significant *t*-values, it was surprising to note that the actual execution time of a series of constant multiplies depends on the operand, which should not be the case according to Intel’s documentation². We leave further investigation of this (possible) effect for future work, as it does not directly seem to relate to leakage between clock domains.

From this initial investigation, we concluded that the ratio between different clocks indeed is a source of (minute) power leakage on x86 processors. We attempted to exploit this effect for a CPA but were unsuccessful, although this was not surprising, given the large amount of background noise which is inherent in a system of this sort. However, despite being small, the leakage was statistically very unlikely to have been by chance ($|t| > 18$), and therefore, we opted to investigate the effect on a simpler, embedded processor.

²<https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/best-practices/data-operand-independent-timing-isa-guidance.html>

Table 3: Comparison of multiplication by all-0 vs. all-1

Measure	<i>t</i> -value
CLOCK_MONOTONIC/RDTSC	-18.69
RDTSC	11.29
CLOCK_MONOTONIC	-4.28

2.2 Exploration on i.MX RT1166

An embedded platform provides a more controlled environment for further experiments, in particular, reducing the amount of parallel/background activity in the processor. As we desired a number of different clocking options, we selected a device from the NXP i.MX RT1160 family (specifically, a IMXRT1166DVM5A on an IMXRT1160-EVK evaluation board). This family of devices allows up to six independent timers to be setup, each running from different clock sources. The clock sources range from a high-quality crystal to on-chip PLL and resistor-capacitor (RC) oscillators—in order of quality, from least-to-most-variable. This specific device contains two processor cores, one Cortex-M7 and one Cortex-M4. In all experiments and demonstrations, the Cortex-M7 was used to execute code, and the Cortex-M4 was idle.

Through various experiments on this target, we found a particularly interesting result. We timed a simple table lookup over 1 000 000 loop iterations. Each loop performed a lookup into a customised SBox using random plaintexts, and an implicit key addition of an all zeros key, *cf.* Listing 1. The customised SBox was designed for an even distribution of HWs (*cf.* Listing 2 in Appendix B) in order to reduce the data capture times whilst allowing the effect to be seen clearly.

```

1 for (uint32_t n = 0; n < iterations; n++) {
2     for (int i = 0; i < 4; i++) {
3         ct[i] = sbox_even[pt[i]];
4     }
5 }

```

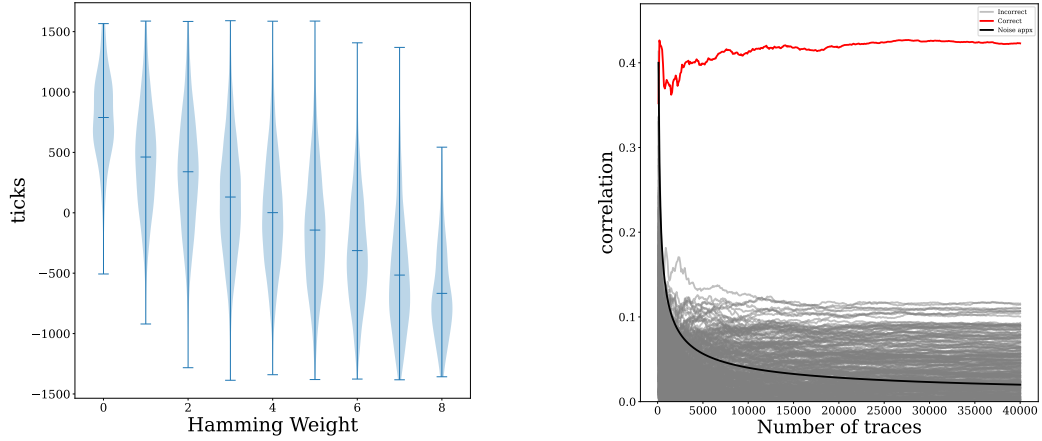
Listing 1: Toy cipher with SBox lookup and (implicit) XOR of all-zero key

We measured the total loop time with a timer clocked directly from a high-quality external crystal oscillator. Such crystals have a maximum deviation over all conditions and lifetime on the order of 50 ppm. We also captured the processor-synchronous `systick` clock. The cycle count of the loop lookup—as measured by `systick`—was expected to be constant, while the variation of the processor PLL clock in relation to the crystal oscillator was expected to vary slightly.

What we actually found was a variation in *both* of the timing measures of approximately 70 ppm, which is *linear* in relation to the HW of the value looked up, *cf.* Figure 5a. Only around 4000 samples are required to produce a correlation value more than $3\times$ that of the incorrect values. To remove the possibility that the memory *location* was the source of leakage, we randomised the SBox content. The leakage remained at a similar level, being clearly visible in a regression analysis (the probability of the line actually having zero slope is $p_{zero} < 1 \times 10^{-27}$ with only 230 samples).

This indicates a potential mechanism for detecting this leakage (which could be employed by either attacker or defender). Timing a loop of a large (but constant) number of `ADD` operations, with the increment being either 0 or non-zero, and the accumulator variable in the asynchronous memory area, should generate an observable difference depending on the increment value.

Experiments with the state of the processor instruction and data caches showed that the data cache being disabled was significant—the data memory was operating asynchronously to the main processor, and therefore reducing the number of hits to the asynchronous bus clearly impacted on the results.



(a) Violin plot showing the linear execution time (with mean subtracted) compared to Hamming Weight (HW) of data

(b) Progression of correlation on toy cipher. Correct key candidate in red, others in gray, noise level in black.

Figure 4: Leakage behaviour on i.MX RT1166 Cortex-M7 processor

Through a further series of experiments, we identified a single instruction which could be changed (from an unnecessary sign-extension to a `NOP`) to reduce the amount of leakage dramatically. We established that disabling the Branch Target Address Cache (BTAC) in the processor reduced the leakage almost completely. We also observed that the silicon temperature had a small effect on the amount of leakage, with the maximum being at normal lab temperatures. Brief experimentation with the core supply voltage (which can be varied under application-software control) indicated that lowering the core voltage reduced the leakage slightly. Raising the core voltage was not attempted to avoid potential damage.

A further intriguing observation is that the “noise floor” of the CPA progression does not follow the theoretical pattern (of $4/\sqrt{n_{\text{samples}}}$ —shown by the black line in Figure 5b [Osw13]), indicating some non-ideal effects in the leakage. Unfortunately, even tiny changes to the code or location of the code in memory reduced the leakage (as measured by a single timer) by several orders of magnitude, or indeed to zero. Further investigation of these interesting (and potentially significant) effects is left to future work.

Given our demonstration of a correlation between power consumption and measurements of timing, in support of our basic hypothesis that variations in timing can be used to infer power consumption, in the following we describe a number of demonstrative experiments to establish a source of the timing changes. We then revisit the observed leakage on the i.MX RT1166 in that light, and construct software-based leakage primitives and corresponding proof-of-concept attacks (*cf.* Section 5 and Section 6).

3 Simulation-based Experiments

Because the timing variations observed were visible even when measured using the clock synchronous to the processor, we hypothesised that the variation was due to accesses performed to another clock-domain.

We constructed a VHDL simulation to investigate the potential parameters around this leakage (Figure 6). The simulation is a very simple model of two parallel processes

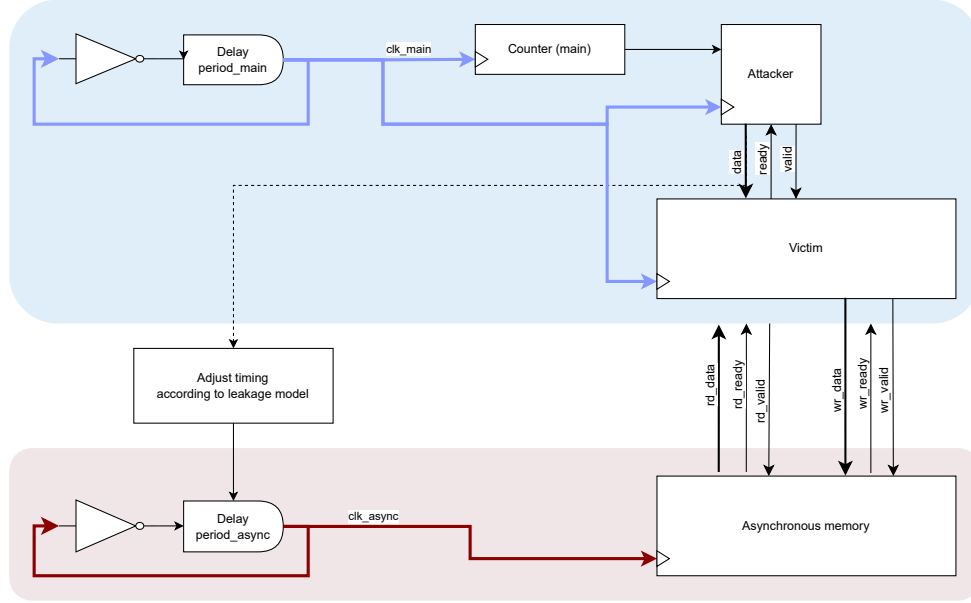


Figure 6: Block diagram of VHDL simulation of leakage due to asynchronous clocks—the two clock domains are highlighted in different colours

(‘attacker’ and ‘victim’) that communicate using an Advanced eXtensible Interface (AXI)³-like pair of “ready/valid” signals. Both these processes are synchronous, driven from the main clock. The simulation was executed with a femtosecond-precision timebase. The attacker process also communicates with an asynchronously-clocked RAM model, using it to read/write an incrementing counter. No clock-domain crossing logic is required, as the simulation model has perfect flip-flops (without any metastable behaviour). This provides an idealised representation of a real-world system. The model allows the asynchronous clock speed of the RAM communication to be manipulated on a cycle-by-cycle basis. This is made dependent on the data being ‘processed’ in the victim process, provided by the attacker. The victim process runs for a configurable number of clock cycles, before indicating it has ‘finished’ to the attacker. The attacker measures the ‘main’ and ‘async’ clock count differences across this time.

Results In this experiment, we used the clock parameters that were active in the original experiments with the i.MX RT1166. Therefore, the main clock frequency is 600 MHz and the asynchronous clock frequency 198 MHz. The attacker process provides single bits of 0 or 1 to the victim. The victim executes its “processing loop” 3 M times, representing about 5 ms of activity. Based on the data communicated to the victim, the simulation introduces a very small change to the period of the asynchronous clock (of 10 fs). This small change (~ 2 ppm) results in small changes to the clock count that the victim is measured as being active for. Performing a *t*-test on the data for 0 and 1 results in a very significant *t*-value of >200 with only 30 samples. The simulated time of this experiment was just 0.16 sec. In this (noise-free) environment, we are able to use the simple cycle-count to confidently distinguish between very small changes in clock frequency with relatively limited data.

³<https://developer.arm.com/documentation/102202/0300/Channel-transfers-and-transactions>

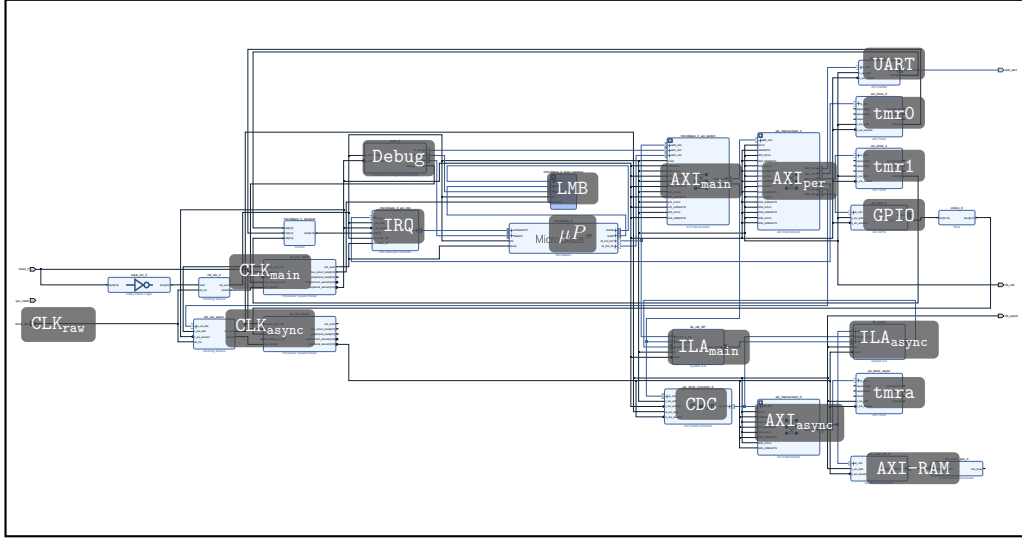


Figure 7: Block diagram of FPGA to explore leakage due to asynchronous clocks

4 Experiments on FPGA

To obtain real-world insights, we next evaluated a target built around a Xilinx Artix-7 FPGA (hosted on a Digilent Arty-7 board). This allows the construction of a controlled lab environment within which we can adjust experimental variables. Again, we only modeled the key parts of the system we are exploring.

In Figure 7, we show the entire system: the main clock is sourced from an external 100 MHz crystal CLK_{raw} . This clock is conditioned and divided-down for the majority of the elements using a clock PLL/reset subsystem CLK_{main} —this main clock runs at 80 MHz. A second PLL and reset provide a second clock domain CLK_{async} . The frequency of this clock is under control of the main microprocessor μP . The microprocessor is tightly coupled (with single-cycle access) to Local Memory Bus (LMB) RAM LMB, which is synchronous to the processor clock. The rest of the system is accessed through a synchronous AXI-interconnect AXI_{main} , which provides the dividing line between the synchronous peripherals (which form the ‘laboratory’) and those operating in the asynchronous domain (which are providing the ‘experimental’ aspects). Two timer peripherals are included in the main microprocessor clock domain $tmr0$ and $tmr1$. A General Purpose I/O (GPIO) $GPIO$ is provided, which allows the microprocessor to reset the asynchronous clock domain after a change of frequency. In order to make a useful experimental platform, a debug interface **Debug** and Universal Asynchronous Receiver Transmitter (UART) **UART** are present to provide the ability to download/execute code and retrieve results. An interrupt system is also present **IRQ**. These peripherals are connected via an AXI interconnect AXI_{per} .

To interface between the main and experimental asynchronous domain, clock-domain crossing (CDC) logic is provided **CDC**. In the asynchronous clock domain, an AXI interconnect AXI_{async} connects a further timer $tmra$, and another RAM block, termed the **AXI-RAM**. To monitor the AXI interconnect signals on either side of the **CDC**, two ILA are present: ILA_{main} and ILA_{async} .

We perform an experiment—designed to be similar to the code described in Section 2—such that the core loop (performing the SBox lookups) and the SBox being accessed are both stored in the non-processor-synchronous AXI-RAM. The remainder of the code and data are stored in the fast, processor-synchronous, LMB RAM, allowing the experiment to generate data as quickly as possible.

The frequencies which are available from the on-chip PLL are determined by the driver software, as not all ratios are available. Therefore random frequencies are requested, but the actual frequency of operation is read back from the PLL and recorded. By varying the frequency of the asynchronous RAM, we confirm that the level of leakage is very dependent on the AXI-RAM clock speed (*cf.* Figure 9b).

This dependence happens because—when signals are resampled into the main clock domain from the asynchronous domain—there is a small window around the main clock edge (the so-called ‘metastable window’) where the capture is unpredictable. During this time, any coincident changes in the signal being sampled will result in stochastic delays at the captured output. This metastable window is dependent on many aspects, including the voltage at the instant of the clock cycle. Therefore a stochastic delay only occurs on the cycles where the two clocks are closely aligned, and is more likely to occur under certain (voltage-related) conditions than others, providing an estimate of power consumption.

If the quotient of the nominal frequencies of the two clocks is rational ($\frac{n}{m}$), the clocks will either align precisely, or be very out of alignment. In either case, the timing of the signals captured from the asynchronous domain is always the same. We expect there to be little dependence on the frequency variation (as the variation is expected to be smaller than the window of sensitivity). There may be a large spread of results if the number of opportunities for this “collision” to occur is large, but the deviations will all be randomly distributed. When the two clocks have no rational relationship, they will move “in and out of phase” with each other and there will be times where the two clocks are very closely, but not exactly, aligned. It is in these instants that unpredictable behaviour related to the power consumption will be captured—Figure 8 shows this behaviour for the non-rational clocks `CLK ref` and `CLK async`.

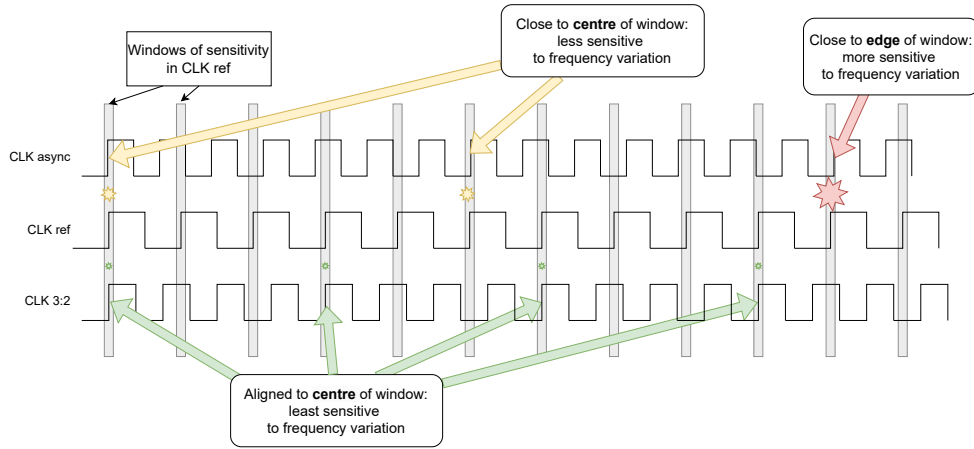


Figure 8: Rational and non-rational divided clocks with respect to a reference clock

We investigated the effect of the clock-frequency of the asynchronous part on the leakage of a series of memory reads. For this experiment, the memory content read was always either `0x00000000` or `0xffffffff`. We measured the time taken to read 100 000 times from RAM using the tick counters `tmr0` and `tmra` and compared t -values for two groups based on the memory read value.

Results Figure 9a shows the variability of the experimental tick count with varying frequency: there is a large distribution of values for the range displayed during the operation, from single cycles, through hundreds, thousands and tens of thousands. We observe that multiples of 10 MHz produce very large (or very small) variations. This fits

our hypothesis, as, depending on the relative phase of the two clocks, these exact multiples of the reference clock would be expected to remain with the same phase relationship, either resulting in “close timing” (causing variation in the tick count) or in “far timing” (causing no clock collisions and hence much smaller variations).

There are numerous frequencies at which there is no variation at all in our (relatively small) experiment, from which we infer that the two clocks do not closely align often enough for there to be a different cycle count. Graphically inspecting the distributions

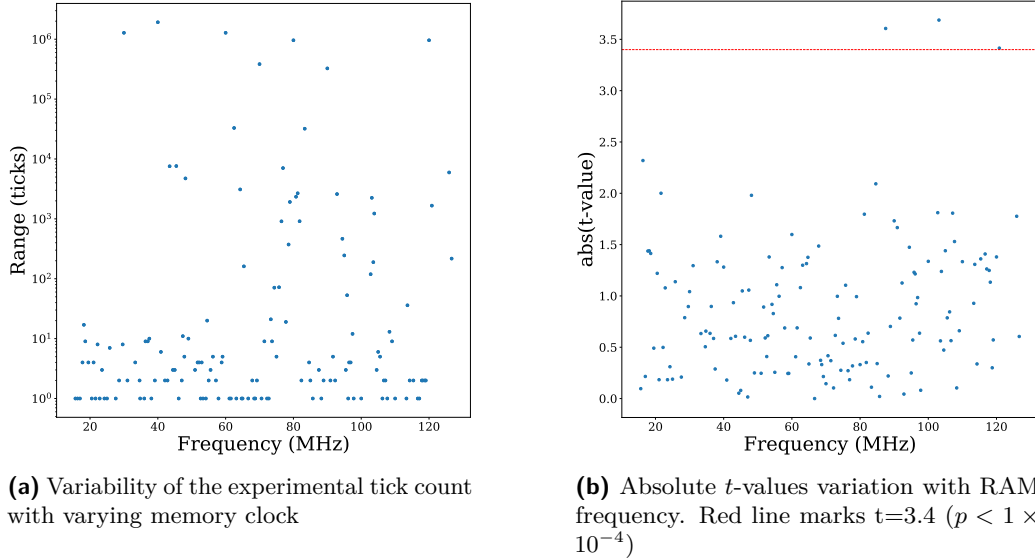


Figure 9: Experiments on FPGA over different memory clock frequencies

of the sample differences from the mean is interesting: for integer multiples of the main clock frequency, we see a bimodal distribution. The magnitude of the difference across the x-axis is very large at different frequencies. At 100 MHz there is only 1 tick (Figure 10a), whereas at 120 MHz (Figure 10b), the difference is almost 100 000 ticks (or 1 tick per loop iteration). While the total range of values is large, in these two cases, the actual measurement is strongly quantised to the extreme values. The difference in distributions between the large and small HW is still visible on the graphs. In comparison, looking at an asynchronous clock frequency 103.125 MHz, the distribution is less regular (Figure 10c), but the difference between the two distributions for different HWs is visually obvious. Considering the magnitude of the t -values in Figure 9b, we find that for most frequencies, the t -value is below 3.4, indicating no evidence of leakage.

Table 4 shows the three frequencies where the magnitude of t -value for the difference between the two datasets is > 3.4 ($p \ll 0.001$). From this table, we also note that the absolute range of the values is not correlated with the amount of leakage.

Table 4: Comparison of different frequencies on FPGA with significant leakage

Frequency (Hz)	Range (ticks)	t -value (ones vs. zeros)
87500000	3	-3.61
103125000	2247	-3.69
120833336	1664	-3.41

Extending this experiment to perform a toy-cipher SBox lookup (*cf.* Listing 1) and running the asynchronous clock as fast as we are able to, produces results that show a

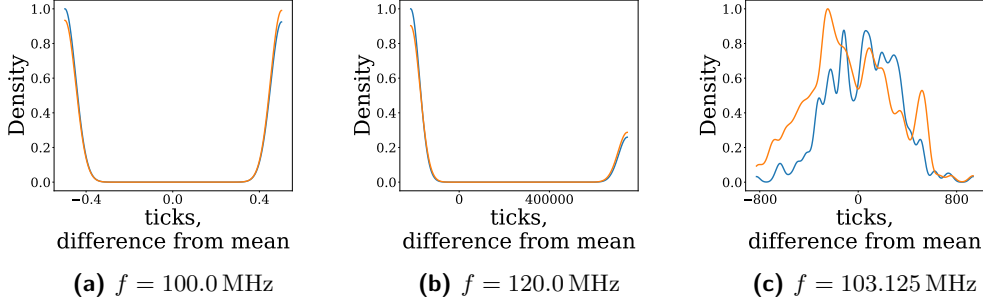


Figure 10: Distribution of measurements for HW0 (blue) and HW8 (orange) at different frequencies, highlighting substantial difference at asynchronous clock

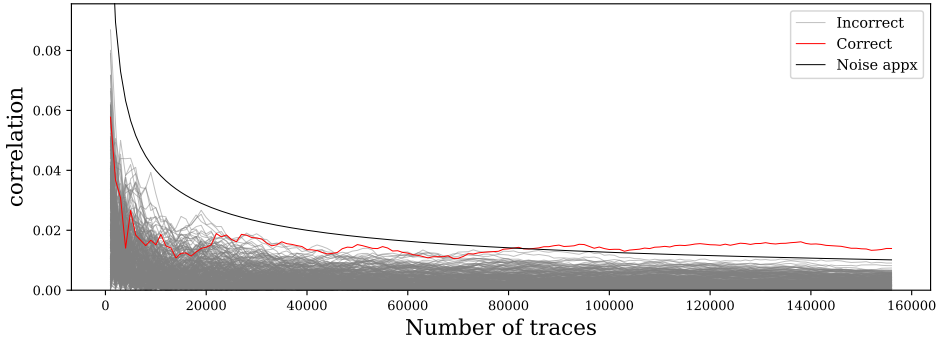


Figure 11: CPA convergence traces for toy-cipher on FPGA platform

leakage related to HW. The linear regression p_{zero} is very low ($p < 4 \times 10^{-8}$), although the r^2 value is also low ($\sim 2 \times 10^{-4}$) due to the level of noise in the system. The slope from the synchronous timer is ~ 4 . This means the variation due to change in HW is 4 ticks per HW bin (out of a total range of ~ 4700). Still, a CPA correlation is above the theoretical noise value at 1.38 for the correct key candidate, as evident in Figure 11.

5 Clock Variation Measurement over Time

Having investigated the mechanism of the observed leakage due to asynchronous clocks in simulation and on FPGA, we now revisit the i.MX RT1166 behaviour. So far, we have demonstrated a mechanism by which a single sample can be used to measure cumulative energy over some time and infer secrets. In this section, we show how an attacker can gain a trace over time of the i.MX RT1166 power consumption by taking regular samples of a timer during the execution of some process operating on secret data.

More precisely, we perform this capture on the attacker-side of the clock domain boundary by using the Direct Memory Access (DMA) engine within the i.MX SoC. The DMA engine is configured to continually capture the timer value into a RAM buffer. The DMA controller operates in the same clock domain as the RAM block, but in a different clock domain to the timer (and incidentally the processor). Therefore, it is expected to collect the same behaviour, of occasional differences in tick count.

The exact details of the capture are as follows: The DMA engine captures the timer approximately every 26 ticks of the 240 MHz clock (around 108 ns). There is sufficient RAM in the SoC to capture 16 000 samples, over a time period of ~ 1.73 ms in total.

We initially demonstrate this with a victim process executing two loops of either NOP

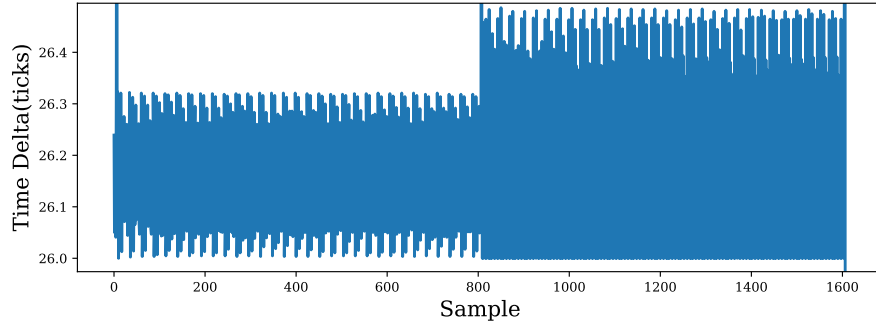


Figure 12: Timer delta capture through DMA over time for NOP and ADD

or ADD instructions. Each loop was tuned to execute in the same amount of time in order to remove an obvious timing side channel. This code executes from the instruction cache and uses the tightly-coupled memory for local variables, meaning it is very well-controlled in time, and operates from a single clock-domain. We take the difference between adjacent time-samples as our data point. A single trace is not useful, as the values are very strongly quantised. Therefore, we capture 1000 such traces and take the pointwise average, with no correction for sample timing. Figure 12 shows a clear transition from the low-power (NOP) to the higher-power condition (ADD) at (approximately) sample 800.

In order to target a victim process that lasts longer than the available data capture time, we set up the DMA capture to provide a random delay between the start of the victim processing and starting the DMA. This is achieved by starting the DMA from an interrupt, triggered by the same timer as is being used for the sampling. This results in a set of partial traces, each covering a section of the target process execution. The samples read back from the timer directly indicate the (random) DMA delay. Hence, they can be used to reconstruct the full trace based on several sub-traces (recorded at different, random offsets). From the measurements of time, we generate a proxy for power from the difference in consecutive time measurements—corresponding to how long each (notionally constant time) DMA activity took. In our specific example, each DMA took either 26 or 27 cycles.

Due to the random DMA delay, none of these partial time-series are aligned or synchronised; therefore we resample them to a regular grid using a zero-order hold (to extend each sample as a constant until the next sample time). These resampled traces are then averaged to produce a raw power trace. This generates a noisy trace, which benefits from filtering along the time axis. We experimentally evaluated a number of digital filter structures and parameters, which led to the use of a 31-tap digital Bessel-filter⁴, with a cutoff frequency (relative to $f_s/2$) of 0.02 and 40 dB of attenuation.

6 Proof-of-Concept Attacks

From this HW-dependent leakage through time measurements, we now build proof-of-concept attacks in a realistic execution environment. Unless otherwise mentioned, for our demonstrations, we extended the i.MX RT1166 software platform to execute the Eclipse ThreadX⁵ real-time operating system (on the Cortex-M7 processor as with all previous experiments), showcasing the ability of an attacker thread to exfiltrate secret information from a separate victim thread. We maintain the disabling of the data cache in order to

⁴<https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.bessel.html>

⁵<https://threadx.io/>

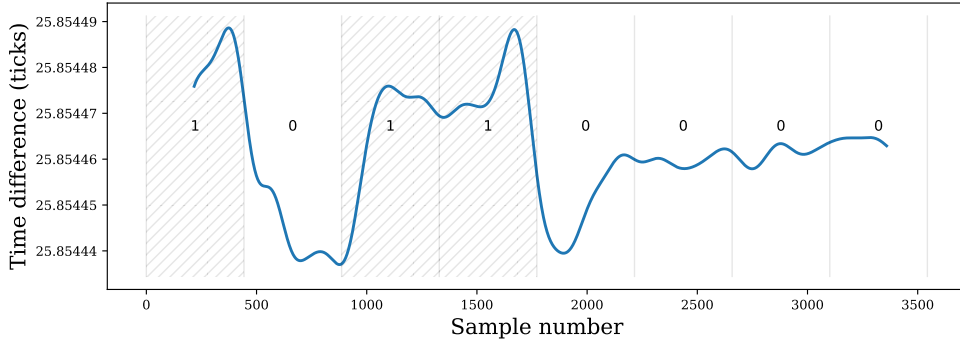


Figure 13: Average of 10 000 interpolated traces after filtering, showing the ability to transmit individual bits through a covert channel.

exclude cache-timing as a leakage path and constrain the variability to only RAM-access timing. Again, no DVFS was active.

6.1 Timing-independent Covert Channel

First, we show how unprivileged access to timer counts can be used to create a (timing-independent) covert channel. The sender process performs an activity which consumes more power to send a 1 bit and idles for the same amount of time (as measured by the processor tick count) to send a 0. Again, in this example, a series of `ADD` instructions are used for the higher and `NOP` for the lower power conditions, respectively. Using the mechanism described in Section 5, the receiver can capture the data over multiple traces and form a time-series, allowing data to be exfiltrated. In this experiment, an entire example byte (`0b10110000`) can be captured without using any offset at the start of the DMA. The attacker can exfiltrate one byte per set of traces captured. We found that 10 000 traces are necessary for straightforward visual inspection.

This allows one byte every 17.3 s or approximately 58 mBps. Many options likely exist to increase this (low) data rate, but we leave the investigation of the true Signal to Noise Ratio (SNR) and bandwidth of this channel (which would enable calculation of the channel capacity according to [Sha49]) to future work.

6.2 RSA Decryption/Signature

In order to demonstrate the ability to use our side channel to attack longer cryptographic functions, we constructed a simple modular exponentiation following [Knu69, Section 4.6.3]. This uses the mbedTLS v3.6.4⁶ bignum functions to perform the multiplications. The victim processes a constant, attacker-provided ‘ciphertext’—modelling an RSA decryption/signature operation using the private key as the exponent. This attack was not performed in the context of a ThreadX environment, as the setup only has a single thread of execution at any one time (the attacker starts the DMA and blocks while the victim executes). We do not expect additional noise to be introduced by the Operating System (OS).

For evaluation, we used two 512-bit private keys: Key 0 was randomly generated (`1ebc1cffcb0a683 ... 39e0fa7a6f3ea8b`), while key 1 (`fc3f0fc3f0 ... fc3f0fc3f0`) was selected to be easy to identify in binary form, consisting of a repeating pattern of six 1s and four 0s.

⁶<https://github.com/Mbed-TLS/mbedtls/releases/tag/v3.6.4>

The attacker makes multiple requests to the victim to this operation using one of two keys (unknown to the attacker). To interleave measurements, each time, a key is selected at random by the attacker. Each iteration is also captured with a different random attacker-selected DMA start delay, allowing coverage of the entire execution. The attacker then performs the interpolation and lowpass-filtering described in Section 5.

In this experiment, we set the maximum DMA delay to cover half of the activity, in order to reduce the data-capture time for demonstration. In this case, we captured $\sim 10\,000$ traces over approximately 15 h. The entire activity could therefore be captured in 30 h and a realistic 2048 bit key in 5 d (we note that we did not particularly optimise our measurement primitive, and speed-ups could likely be achieved). The measurement primitive induces zero overhead on the victim as the DMA, timer and sample RAM are all operating in a separate clock domain, and on a separate set of bus infrastructure, to the processor on which the victim executes.

The results of this attack are presented in Figure 14. A Savitzky-Golay filter [SG64], with a window size of 500 samples and using a second-order fit, has been additionally applied in order to make the plots more visually accessible. We present the first fifth of the data (corresponding to 56 and 58 bits of the keys, key 0 is the top row and key 1 is the lower row) in the left column. Note that the first 6 bits are obscured by the transient at the start. This transient is caused by the uniform distribution of the DMA delay sampling - this results in this area of the activity being sampled less frequently, resulting in increased noise. Similarly, on the right column, the final fifth of the data (covering 54 bits of each key) are plotted—in this case, the final 8 bits are obscured by a similar transient. It would be straightforward to arrange the DMA delay distribution to provide even coverage of these areas in future work, by starting the DMA some time before the victim activity, and allowing it to continue for some time afterwards, which would result in a small increase in size of the data captured (approximately 2.5%).

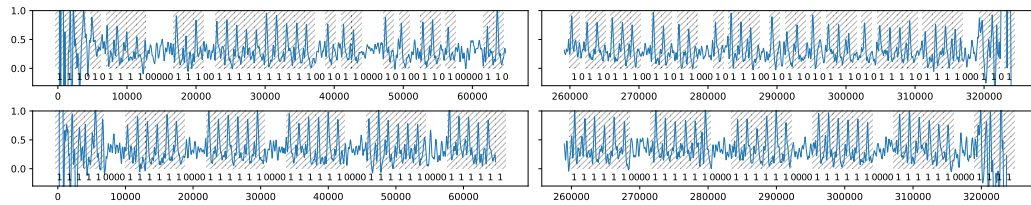


Figure 14: Attack on sections of 512-bit key. The correct key bits are annotated at the bottom of each plot. Top: key 0 (random). Bottom: key 1 (structured). Left: first 55-bits. Right: later 55 bits.

Both the structured and random key bits can be seen to align with the power traces, as would be expected in a traditional SPA-attack, allowing extraction of all but the transiently-obscured 14 bits of the key, leading to a successful attack.

6.3 CPA on TinyAES

In this demonstration, the victim executes TinyAES⁷ encryption operations with attacker-controlled plaintext, returning the ciphertext. The attacker loops, incrementing a counter stored in asynchronous RAM, waiting for the victim to finish. In this experiment, the attacker-controlled plaintext, ciphertext, and the background loop counter, as well as the victim’s SBox and AES context (which includes the key schedule) are located in asynchronous memory. All other data is located in tightly-coupled RAM, which is clocked by the processor clock. Again, the program code and constants (including the secret key)

⁷<https://github.com/kokke/tiny-AES-c>

are stored in cached external flash (Figure 15). The two threads are scheduled by the ThreadX operating-system. The measurement does not impact on the victim execution, as it consists only of a timer read before execution and another timer read afterwards.

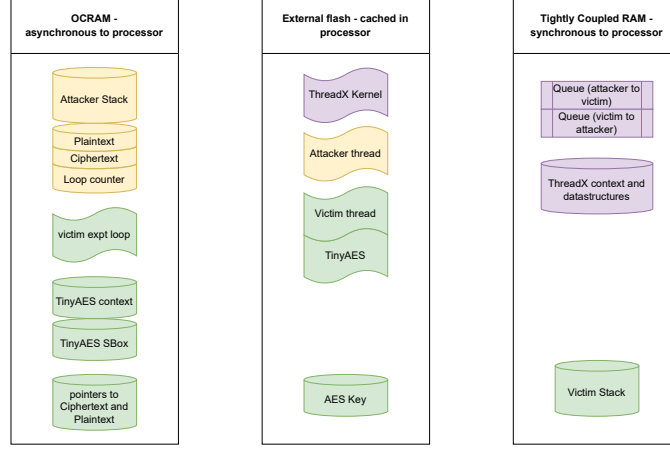


Figure 15: Memory arrangement of TinyAES proof-of-concept on i.MX RT1166

Targeting the first **SubBytes** operation using a HW model, key byte 12 exhibits a clear correlation (exceeding the expected noise level of $4/\sqrt{N}$ by a factor of ~ 2.0). We observe that the other key bytes do not leak under this model (Figure 16): we leave the investigation of this single-byte leakage phenomenon for future work.

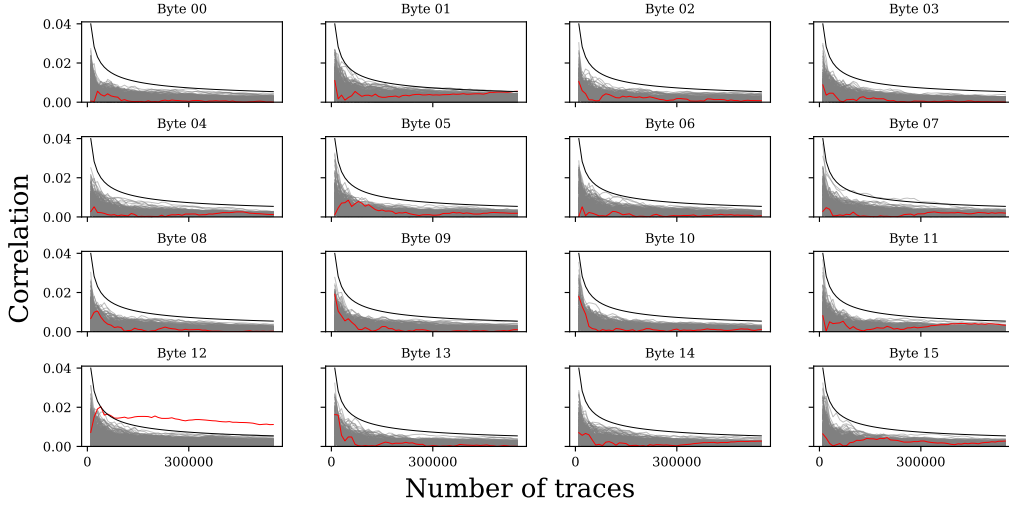


Figure 16: CPA convergence on first SubBytes over 545 000 traces on i.MX RT1166

7 Summary

We demonstrated that even carefully-designed constant-time operations can exhibit measurable low-level timing variations, which are dependent on the power being consumed. In addition, continuous fine-grained low-level measurements of time across clock-domains can be used as a proxy for power measurements, revealing details of activity over time which allow the exfiltration of secret information. The details of operation times of non-constant time activities can also be inferred from time-measurements made outside of the privileged domain. Although the effect is very small, it is sufficient to be used to perform traditional power-based side-channel attacks such as CPA and SPA.

7.1 Discussion

The fundamental issue enabling these attacks is the measurement of time from two different sources. These do not have to be formal timers: as we have demonstrated, even the access time of an asynchronous peripheral or RAM block will have timing variability relative to a constant execution time in terms of processor cycles. This timing variability is very small—orders of magnitude smaller than a single clock cycle—but can be effectively measured by exploiting the fact that it will occasionally result in a time measurement which is a whole clock cycle larger. Platforms which are entirely synchronous in nature will not expose such issues. However, these are rare, as interfacing with external devices often requires specific clock frequencies, necessitating the generation of asynchronous clocks.

Based on our understanding of the phenomenon, and the early results presented in Section 2, we believe that this effect is also weakly present on complex x86 processors due to the provision of a real-time clock, which is calibrated with respect to real time, allowing the observation of variations in the processor cycle time. In addition, the external RAM interfaces of such systems are asynchronous. Forcing memory access to external DRAM while secrets are being processed (even from cache) may create a similar side channel.

7.2 Countermeasures

A key mitigation is to locate critical data structures in memory which is synchronous to the processor accessing it. Critical data structures include not just key material, but also lookup tables such as SBoxes and T-tables. Organising these structures in this way removes the power-dependent variability that comes from the asynchronous accesses. We performed such an experiment and demonstrated that no leakage was then detectable from the single timer measurement. However, as demonstrated by the RSA attack, even the behaviour of entirely synchronous code and data can be inferred from the variations in timing of a separate peripheral which is sampled numerous times.

A further approach would be to disable access to all timers. This would be an intolerable imposition on performance-oriented software developers (which is very common in embedded systems work), but perhaps can be considered in deployed products outside of the development process. Quantising the reported time measured by timer peripherals would extend the period over which attacks would have to take place (as the attacker would have to run commensurately more executions of each data point), but does not fix the fundamental problem. In systems which use an external time source to calibrate their idea of time, unprivileged processes could be granted access to the corrected time-base only. Even without access to timers, in a multi-tasking context, an attacker thread which counts loop iterations while the victim is executing can serve as a time base [Lip+16].

All usual mitigations against power side-channels remain applicable, *e.g.* masking for symmetric cryptography [RP10] and blinding [Koc96] in the case of asymmetric cryptography. Interestingly, mitigations which introduce jitter are unlikely to be successful, as our sampling is very slow in relation to the process being attacked and therefore the jitter

would have to be greater than the time between samples. In the case of the DMA-based attacks, this is on the order of 100 ns. When a single data point is used to perform the attack (as in our AES demonstration), any jitter will still result in a leakage captured within that single sample time.

Acknowledgements

Martin gratefully acknowledges the support of his employer, ZF Automotive UK Ltd, and the scholarships provided by the Universities of Birmingham and Durham. This research was partially funded by the Engineering and Physical Sciences Research Council (EPSRC) under grants EP/X03738X/1 and EP/X03738X/2.

References

- [Abr02] D. Abramovitch. “Phase-locked loops: a control centric tutorial”. In: *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*. 2002. DOI: [10.1109/ACC.2002.1024769](https://doi.org/10.1109/ACC.2002.1024769).
- [Ayo+24] Pierre Ayoub et al. “PhaseSCA: Exploiting Phase-Modulated Emanations in Side Channels”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2025.1 (Dec. 2024), pp. 392–419. DOI: [10.46586/tches.v2025.i1.392-419](https://doi.org/10.46586/tches.v2025.i1.392-419). URL: <https://tches.iacr.org/index.php/TCHES/article/view/11934>.
- [Ban17] Dean Banerjee. *PLL Performance, Simulation, and Design, 5th Edition (SNAA106c)*. 2017. URL: <https://www.ti.com/lit/ml/snaa106c/snaa106c.pdf>.
- [BB03] David Brumley and Dan Boneh. “Remote Timing Attacks Are Practical”. In: *USENIX Security 2003*. USENIX Association, Aug. 2003.
- [BCO04] Éric Brier, Christophe Clavier, and Francis Olivier. “Correlation Power Analysis with a Leakage Model”. In: *CHES*. 2004.
- [Ber05] Daniel J Bernstein. *Cache-timing attacks on AES*. 2005. URL: <https://cr.yp.to/antiforgery/cachetiming-20050414.pdf>.
- [BM06] Joseph Bonneau and Ilya Mironov. “Cache-Collision Timing Attacks Against AES”. In: *Cryptographic Hardware and Embedded Systems - CHES 2006*. Ed. by Louis Goubin and Mitsuru Matsui. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 201–215.
- [Dog+11] Julien Doget et al. “Univariate side channel attacks and leakage modeling”. In: *J. Cryptogr. Eng.* 1.2 (2011), pp. 123–144. DOI: [10.1007/s13389-011-0010-2](https://doi.org/10.1007/s13389-011-0010-2). URL: <https://doi.org/10.1007/s13389-011-0010-2>.
- [FBL24] Anis Fellah-Touta, Lilian Bossuet, and Carlos Andres Lara-Nino. “A Lightweight Non-Oscillatory Delay-Sensor for Remote Power Analysis”. In: *2024 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*. 2024, pp. 343–348. DOI: [10.1109/HOST55342.2024.10545353](https://doi.org/10.1109/HOST55342.2024.10545353).
- [Gie+08] Benedikt Gierlichs et al. “Mutual Information Analysis A Generic Side-Channel Distinguisher”. In: *Lecture Notes in Computer Science* 5154 (2008), pp. 426–442. URL: <https://api.semanticscholar.org/CorpusID:3754674>.

- [GKT19] Dennis R. E. Gnad, Jonas Krautter, and Mehdi B. Tahoori. “Leaky Noise: New Side-Channel Attack Vectors in Mixed-Signal IoT Devices”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2019.3 (2019), pp. 305–339. DOI: [10.13154/tches.v2019.i3.305-339](https://doi.org/10.13154/tches.v2019.i3.305-339). URL: <https://tches.iacr.org/index.php/TCHES/article/view/8297>.
- [GMO01] Karine Gandolfi, Christophe Mourtél, and Francis Olivier. “Electromagnetic Analysis: Concrete Results”. In: *Cryptographic Hardware and Embedded Systems — CHES 2001*. Ed. by Çetin K. Koç, David Naccache, and Christof Paar. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 251–261. ISBN: 978-3-540-44709-2.
- [Goo+11] Gilbert Goodwill et al. *A testing methodology for side-channel resistance validation*. 2011. URL: https://csrc.nist.gov/csrc/media/events/non-invasive-attack-testing-workshop/documents/08_goodwill.pdf (visited on 03/2022).
- [Gra+21] Joseph Gravelier et al. “SideLine: How Delay-Lines (May) Leak Secrets from Your SoC”. In: *COSADE 2021*. Ed. by Shivam Bhasin and Fabrizio De Santis. Vol. 12910. LNCS. Springer, Cham, Oct. 2021, pp. 3–30. DOI: [10.1007/978-3-030-89915-8_1](https://doi.org/10.1007/978-3-030-89915-8_1).
- [Kel+98] John Kelsey et al. “Side channel cryptanalysis of product ciphers”. In: *Computer Security — ESORICS 98*. Ed. by Jean-Jacques Quisquater et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 97–110.
- [Kim+19] Jaehun Kim et al. “Make Some Noise. Unleashing the Power of Convolutional Neural Networks for Profiled Side-channel Analysis”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 3 (2019). DOI: [10.13154/tches.v2019.i3.148-179](https://doi.org/10.13154/tches.v2019.i3.148-179). URL: <https://tches.iacr.org/index.php/TCHES/article/view/8292>.
- [KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. “Differential Power Analysis”. In: *Advances in Cryptology — CRYPTO’ 99*. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 388–397. ISBN: 978-3-540-48405-9.
- [Knu69] Donald E. Knuth. *The Art of Computer Programming: Vol. 2 Seminumerical Algorithms*. Addison-Wesley Series in Computer Science and Information Processing. Addison-Wesley, 1969.
- [Koc+19] Paul Kocher et al. “Spectre Attacks: Exploiting Speculative Execution”. In: *Proceedings of the 40th IEEE Symposium on Security and Privacy*. 2019, pp. 1–19. DOI: [10.1109/SP.2019.00001](https://doi.org/10.1109/SP.2019.00001).
- [Koc96] Paul C. Kocher. “Timing Attacks on Implementations of Diffie-Hellman RSA, DSS, and Other Systems”. In: *Advances in Cryptology — CRYPTO ’96*. Ed. by Neal Koblitz. 1996. URL: https://link.springer.com/chapter/10.1007%2F3-540-68697-5_9#aboutcontent.
- [Lip+16] Moritz Lipp et al. “ARMageddon: Cache Attacks on Mobile Devices”. In: *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 549–564. ISBN: 978-1-931971-32-4. URL: <https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/lipp>.
- [Lip+18] Moritz Lipp et al. “Meltdown: Reading Kernel Memory from User Space”. In: *Proceedings of the 27th USENIX Security Symposium*. 2018, pp. 973–990. DOI: [10.1177/1555343419850132](https://doi.org/10.1177/1555343419850132).

- [Lip+21] Moritz Lipp et al. “PLATYPUS: Software-based Power Side-Channel Attacks on x86”. In: *2021 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2021, pp. 355–371. DOI: [10.1109/SP40001.2021.00063](https://doi.org/10.1109/SP40001.2021.00063).
- [Lon+15] Jake Longo et al. “SoC It to EM: ElectroMagnetic Side-Channel Attacks on a Complex System-on-Chip”. In: *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*. Ed. by Tim Güneysu and Helena Handschuh. Vol. 9293. Lecture Notes in Computer Science. Springer, 2015, pp. 620–640. DOI: [10.1007/978-3-662-48324-4_31](https://doi.org/10.1007/978-3-662-48324-4_31). URL: https://doi.org/10.1007/978-3-662-48324-4_31.
- [MA0-] Michael McKerns and Michael Aivazis. *pathos: a framework for heterogeneous computing*. 2010-. URL: <https://uqfoundation.github.io/project/pathos>.
- [Max73] James Clerk Maxwell. *A Treatise on Electricity and Magnetism*. Oxford University Press, 1873.
- [McK+11] M.M. McKerns et al. “Building a framework for predictive science”. In: *Proceedings of the 10th Python in Science Conference*. 2011. URL: <https://eprint.iacr.org/1202.1056>.
- [MOP10] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. 1st. Springer Publishing Company, Incorporated, 2010. ISBN: 1441940391.
- [OD19] Colin O’Flynn and Alex Dewar. “On-Device Power Analysis Across Hardware Security Domains.: Stop Hitting Yourself.” In: 2019 (2019). DOI: [10.13154/tches.v2019.i4.126-153](https://doi.org/10.13154/tches.v2019.i4.126-153). URL: <https://tches.iacr.org/index.php/TCHES/article/view/8347>.
- [OF124] Colin O’Flynn. *Phase Modulation Side Channels: Jittery JTAG for On-Chip Voltage Measurements*. Cryptology ePrint Archive, Paper 2024/1107. 2024. URL: <https://eprint.iacr.org/2024/1107>.
- [Ohm27] G.S. Ohm. *Die galvanische Kette, mathematisch bearbeitet*. Reimann, 1827.
- [Osw13] David Oswald. “Implementation Attacks: From Theory to Practice”. PhD thesis. Ruhr-University Bochum, 2013. URL: <http://www.emsec.rub.de/media/crypto/veroeffentlichungen/2010/09/07/paper.pdf>.
- [Per05] Colin Percival. *CACHE MISSING FOR FUN AND PROFIT*. 2005. URL: <https://www.daemonology.net/papers/htt.pdf>.
- [RP10] Matthieu Rivain and Emmanuel Prouff. “Provably Secure Higher-Order Masking of AES”. In: *Cryptographic Hardware and Embedded Systems, CHES 2010*. Ed. by Stefan Mangard and François-Xavier Standaert. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 413–427. ISBN: 978-3-642-15031-9.
- [Sch+23] Kai Schoos et al. “JitSCA: Jitter-based Side-Channel Analysis in Picoscale Resolution”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2023.3 (June 2023), pp. 294–320.
- [SG64] Abraham. Savitzky and M. J. E. Golay. “Smoothing and Differentiation of Data by Simplified Least Squares Procedures.” In: *Analytical Chemistry* 36.8 (1964), pp. 1627–1639. DOI: [10.1021/ac60214a047](https://doi.org/10.1021/ac60214a047). eprint: <https://doi.org/10.1021/ac60214a047>. URL: <https://doi.org/10.1021/ac60214a047>.
- [Sha49] C.E. Shannon. “Communication in the Presence of Noise”. In: *Proceedings of the IRE* 37.1 (1949), pp. 10–21. DOI: [10.1109/JRPROC.1949.232969](https://doi.org/10.1109/JRPROC.1949.232969).

- [SLP05] Werner Schindler, Kerstin Lemke, and Christof Paar. “A Stochastic Model for Differential Side Channel Cryptanalysis”. In: *CHES 2005*. Ed. by Josyula R. Rao and Berk Sunar. Vol. 3659. LNCS. Springer, Berlin, Heidelberg, Aug. 2005, pp. 30–46. DOI: [10.1007/11545262_3](https://doi.org/10.1007/11545262_3).
- [SM22] Tobias Schneider and Amir Moradi. “Leakage Assessment Methodology”. In: (2022), pp. 495–513. DOI: [10.1007/978-3-662-48324-4_25](https://doi.org/10.1007/978-3-662-48324-4_25). URL: https://doi.org/10.1007/978-3-662-48324-4_25.
- [Udu+21] Brian Udugama et al. “VITI: A Tiny Self-Calibrating Sensor for Power-Variation Measurement in FPGAs”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2022.1 (Nov. 2021), pp. 657–678. DOI: [10.46586/tches.v2022.i1.657-678](https://tches.iacr.org/index.php/TCHES/article/view/9311). URL: <https://tches.iacr.org/index.php/TCHES/article/view/9311>.
- [Udv+21] Balazs Udvarhelyi et al. “On the Security of Off-the-Shelf Microcontrollers: Hardware Is Not Enough”. In: *Smart Card Research and Advanced Applications*. Ed. by Pierre-Yvan Liardet and Nele Mentens. Cham: Springer International Publishing, 2021, pp. 103–118. ISBN: 978-3-030-68487-7.
- [VMC19] Aurélien Vasselle, Philippe Maurine, and Maxime Cozzi. “Breaking Mobile Firmware Encryption through Near-Field Side-Channel Analysis”. In: *Proceedings of the 3rd ACM Workshop on Attacks and Solutions in Hardware Security Workshop* (2019).
- [Wan+22] Yingchen Wang et al. “Hertzbleed: Turning Power Side-Channel Attacks Into Remote Timing Attacks on x86”. In: *Proceedings of the USENIX Security Symposium (USENIX)*. 2022.

A Distribution for Different Measurements on x86

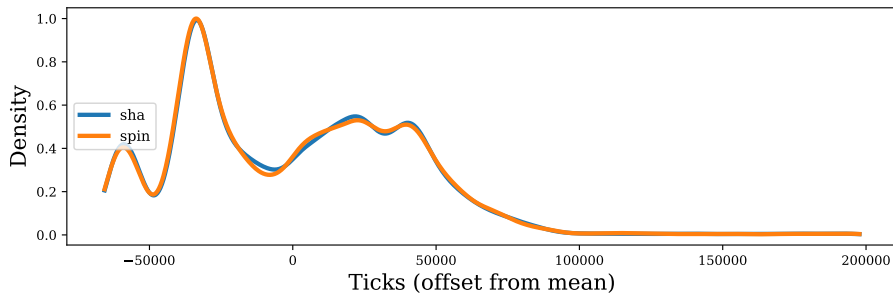


Figure 17: Comparison of activity for CLOCK_MONOTONIC

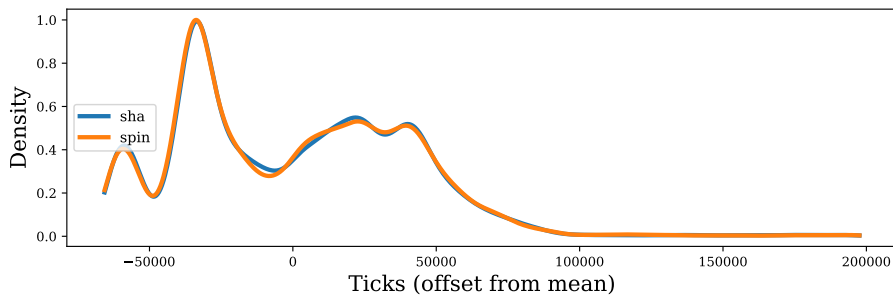


Figure 18: Comparison of activity for RDTSC

B Appendix - “sbox_even”

```

1 #define SBOX_EVEN_LENGTH (252u)
2 const uint8_t sbox_even[SBOX_EVEN_LENGTH] =
3 {0xff, 0x44, 0x37, 0x0, 0xf0, 0x2, 0x6f, 0x0, 0xa0, 0x10, 0xfc, 0x8d, 0xff, 0x9f, 0x0,
4 0xda, 0xff, 0x8e, 0xba, 0x0, 0x41, 0x0, 0xef, 0x8, 0xd2, 0xff, 0x0, 0xd1, 0xbf, 0x8,
5 0xcf, 0x8f, 0x4f, 0xac, 0x94, 0xfd, 0xff, 0x0, 0x0, 0xdb, 0x0, 0x1, 0xff, 0xfa, 0xd,
6 0x12, 0xff, 0x8f, 0x2d, 0x0, 0x0, 0xff, 0xef, 0x80, 0x0, 0x64, 0xdf, 0x41, 0x90, 0x4,
7 0xaf, 0xf3, 0x60, 0x80, 0xed, 0x40, 0x57, 0x52, 0x3b, 0xc3, 0xd8, 0x39, 0x2, 0x22,
8 0xfb, 0x0, 0x0, 0x54, 0xbb, 0x40, 0xd9, 0x5f, 0xc3, 0x0, 0x60, 0xfb, 0x40, 0x6e,
9 0x6b, 0x20, 0x99, 0x0, 0xff, 0xf8, 0x6c, 0xb0, 0x80, 0x12, 0x9f, 0xbf, 0xff, 0xef, 0x7,
10 0x0, 0xff, 0xf1, 0x2, 0x39, 0x9f, 0xff, 0x1, 0x0, 0x60, 0x88, 0x43, 0x4e, 0x0, 0x4c,
11 0x4b, 0x6b, 0xf0, 0x7f, 0x8, 0x0, 0xeb, 0xb4, 0xde, 0xff, 0xff, 0x26, 0xdb, 0x8a, 0x83,
12 0x0, 0xde, 0x13, 0x10, 0x1, 0xff, 0x64, 0xfb, 0xd5, 0x45, 0x0, 0x6a, 0x80, 0xfe, 0x90,
13 0xbf, 0x20, 0x33, 0xff, 0xff, 0x0, 0xff, 0x9e, 0x7f, 0xf7, 0x2, 0xba, 0x7f, 0x5e, 0xb,
14 0x90, 0xcf, 0x9f, 0x34, 0x98, 0x38, 0xff, 0xff, 0xfe, 0xee, 0xa9, 0xd4, 0x92, 0xef,
15 0x4, 0x0, 0x44, 0x79, 0x0, 0x9, 0xaf, 0x63, 0xcf, 0x5b, 0x0, 0xa, 0xd3, 0x3f, 0xd5,
16 0xef, 0x18, 0xff, 0x6f, 0x48, 0xce, 0xff, 0xc, 0xfe, 0xb8, 0xfe, 0x32, 0x12, 0x20,
17 0xff, 0x77, 0x21, 0xf7, 0x8, 0x2, 0x48, 0x0, 0xff, 0xff, 0xef, 0x0, 0x5, 0xa4, 0x79,
18 0x30, 0xff, 0xfe, 0xb0, 0x73, 0x33, 0x24, 0x20, 0xf3, 0x2b, 0x31, 0x13, 0xdc, 0x89,
19 0xff, 0x99, 0x29, 0x77, 0xb2, 0xc, 0xb7, 0x98, 0x3b, 0xdf, 0xff, 0xfd, 0x81, 0xbf,
20 0xf7, 0x6e, 0x2};

```

Listing 2: Custom SBox with even distribution of HW

Secret Key Recovery of FALCON using Simple Power Analysis in Conditional Calculator

GyuSang Kim¹, JeongHwan Lee¹, Myeonghoon Lee¹, Seokhie Hong¹ and
HeeSeok Kim^{2*}

¹ Korea University, Seoul, South Korea, {ks9509,hwani0814,ope2527,shhong}@korea.ac.kr

² Korea University, Sejong, South Korea, 80khs@korea.ac.kr

Abstract. Among the NIST-standardized algorithms, FALCON is a lattice-based digital signature scheme that offers strong security and compactness. However, FALCON’s reliance on floating-point arithmetic makes it vulnerable to side-channel attacks. In particular, certain operations in FALCON, such as floating-point conversion and floating-point addition within the FFT transform, may result in data-dependent power consumption patterns. These patterns can be exploited by Simple Power Analysis to extract secret key information, even from a single trace.

We present a Simple Power Analysis against the FALCON digital signature scheme, focusing on the *Conditional calculator* involved in floating-point conversion and floating-point addition. We also analyze the effectiveness of two countermeasures for the *Conditional calculator*. We propose a post-processing procedure that computes all possible candidates and applies a pruning strategy to eliminate impossible ones. The secret key can be recovered within 0.12 seconds for secret keys generated from a discrete Gaussian distribution and 21.02 seconds for those generated from a uniform distribution. We further propose an advanced post-processing procedure that ranks candidate keys based on their consistency with observed side-channel information, enabling full secret key recovery even when partial information is incorrect. The proposed attack is evaluated with up to 5,000 intentionally corrupted side-channel information entries ($\approx 9.3\%$ of the leakage bits); within this range, the correct key was consistently recovered with a 100% success rate. Additionally, we successfully recovered the correct secret key across 1,000 distinct FALCON secret keys.

Keywords: FALCON · Simple Power Analysis · Key Recovery · Conditional calculator

1 Introduction

NIST started the standardization of Post-Quantum Cryptography (PQC) in 2017 and selected four schemes as standards in 2022: Kyber [SAB⁺22] for Public Key Encryption, and Dilithium [LDK⁺22], FALCON [PFH⁺22], and SPHINCS+ [HBD⁺22] for Digital Signatures. Later, a code-based scheme called HQC [AAB⁺22] was also added to the standards. FALCON is a lattice-based digital signature scheme characterized by its small signature and public key sizes, which improve efficiency in storage and transmission. FALCON performs polynomial operations in the Fast Fourier Transform (FFT) domain, a method that enables efficient convolution and multiplication. Operations in FFT domain use floating-point arithmetic, which improves computational efficiency but increases implementation complexity. Floating-point numbers are typically represented according to the IEEE 754 standard [DBL85], consisting of three components: sign, exponent, and

*Corresponding author.

mantissa. Each of these components is handled differently during arithmetic operations. When floating-point addition is implemented in software, the process becomes particularly complex, involving multiple stages such as normalization, alignment, and rounding. These characteristics improve numerical efficiency but also increase implementation complexity. In fact, FALCON includes software implementations of floating-point operations to support environments that lack floating-point units, such as embedded systems. Although FALCON is designed with strong cryptanalytic security based on short integer solution problem [GPV08, BDF⁺11], it may still be vulnerable to side-channel analysis in practical implementations.

Side-channel analysis targets unintended physical leakages that occur during computation, such as power consumption, electromagnetic emissions, and execution time. Simple Power Analysis [KJJ99] is a type of side-channel attack that uses a single power trace to extract secret information by analyzing power consumption during cryptographic operations. This makes Simple Power Analysis a particularly practical and low-effort threat, emphasizing the need for effective countermeasures at the implementation level.

Related Works. Lattice-based signature schemes have been widely investigated from a side-channel perspective. In particular, FALCON has attracted considerable attention due to its use of floating-point operations and discrete Gaussian sampling, both of which can lead to exploitable leakage. Several studies have demonstrated key recovery by analyzing timing, power, or electromagnetic variations in the discrete Gaussian sampler.

BEARZ Attack [MHS⁺19] is a fault attack technique targeting the FFT algorithm, which proposes a method of recovering the secret key by changing the internal values of the FFT algorithm through aborting, zeroing, and skipping.

Timing leakage during Gaussian sampling has been shown to reveal Gram–Schmidt orthogonalized basis vector norms, and in the case of DLP, it has been experimentally demonstrated that this enables full secret key recovery [FKT⁺20]. In contrast, for FALCON, key recovery remains an open problem due to its structural complexity.

An attack was proposed that uses differential EM attack to recover the sign, mantissa, and exponent values from electromagnetic leakages during FFT-domain multiplications [KA21]. This approach improved upon earlier attack by reducing the number of required traces [GMRR22].

Another attack leveraged Simple Power Analysis on the Gaussian sampler to recover partial information, which was then used to filter signatures and apply a Hidden Parallelepiped Problem (HPP) solver to reconstruct the direction of the Gram–Schmidt orthogonalized basis [GMRR22].

Subsequently, a more efficient method was introduced that further improved these attacks by incorporating spectral decomposition of the covariance matrix and exploiting sign leakage, significantly reducing the number of traces and computational cost [ZLYW23].

In addition, a single-trace attack [QA25] was proposed that targets the key generation of FALCON. The attack targets Gaussian sampler and uses Hamming weight differences to directly recover the secret key from a single power trace.

In addition to leakage from FALCON, recent studies have also highlighted the impact of *conditional leakage*—information leakage that arises from control-dependent operations such as conditional negation.

The first work to investigate conditional leakage in post-quantum cryptography [KAA21] demonstrated that secret-dependent conditional operations can be successfully recovered by targeting operations where two different values are selected depending on its value, specifically the sort operation in NTRU and the polynomial-generation operation in Dilithium.

Along the same line, side-channel attacks on CRYSTALS-Kyber demonstrate that bitmasking-based message encoding still leaks due to large Hamming weight swings of the

mask, enabling practical message recovery even under first-order masking [WCCL22].

The study on the HAWK signature scheme [GR24] analyzed leakage from conditional operations in its sampler, recovering secret-dependent predicates (e.g., sign, parity, and norm information) from power traces and using them to reconstruct each coefficient of the secret key polynomial.

Our Contributions. In this paper, we propose an attack method that targets the floating-point conversion and addition operations in FALCON, performs a Simple Power Analysis to recover secret information, and reconstructs the full secret key through a post-processing procedure. Our contributions are as follows:

- We propose a Simple Power Analysis similar in spirit to prior studies on conditional leakage, but specifically applied to the *Conditional calculator* used in FALCON’s floating-point conversion operations and addition operations in the FFT transformation. The recovered side-channel information includes the sign, exponent, and zero status of secret key coefficients during floating-point conversion, and the comparison result of the absolute values of the operands, sign matching, and exponent increment about floating-point addition’s operands. We verify experimentally on the ChipWhisperer Husky that the control bit can be recovered with a 99.89% success rate. Also, from the power traces collected over 1,000 independent experiments with different secret keys, the control bits for `fprscale` and `fpradd` were recovered with 100% and 99.48% success rates in FFT transformation operations, respectively, yielding an overall recovery rate of 99.52% across both operations.
- We present two countermeasures against Simple Power Analysis that prevent information leakage from the *Conditional calculator*. We evaluate each countermeasure on the same platform, achieving recovery rates of 56.86% and 57.41%, respectively.
- We propose a post-processing procedure that computes all possible candidates and applies a pruning strategy to eliminate impossible ones. The secret key can be recovered within 0.12 seconds for secret keys generated from a discrete Gaussian distribution and 21.02 seconds for those generated from a uniform distribution.
- We propose an advanced post-processing procedure that allows for secret key reconstruction even when incorrect side-channel information is included, by ranking candidates according to their consistency with the available information, so that the correct key is recovered even under noisy conditions. We further tested the advanced post-processing procedure in scenarios where up to 5,000 incorrect leakage bits, demonstrating that it could still recover the correct key with a 100% success rate. We also propose a likelihood-based variant that converts measured power samples into soft information and applies it to the key-ranking procedure. We further confirm a sharp phase transition: in our toy FFT experiment ($n = 32$) under the chosen computational budget, once more than 28.2% of the leakage bits are corrupted, the advanced post-processing fails in all trials, indicating that the proposed countermeasures ($\approx 43\%$ error rate) raise the error level beyond what our attack can tolerate. In addition, we show that the secret key was successfully recovered in all 1,000 tests, each performed on a different FALCON secret key.

The attacks presented in this paper target the PQClean [KSSW22] implementation of FALCON and are also applicable to FALCON reference implementations. It should be emphasized that neither the reference implementation nor PQClean is designed to resist side-channel attacks such as Simple Power Analysis. Nevertheless, showing that even these baseline and widely used implementations exhibit exploitable leakages underlines the need for dedicated countermeasures in future optimized or standardized versions.

Code. The source code of post-processing procedure is accessible through this git repository¹. This code allows you to check the estimated side-channel information for the secret key and recover the secret key from some incorrect side-channel information.

Organization. In this paper, Section 2 provides the preliminaries for reading the paper, including background on FALCON, FFT, floating-point arithmetic, and Simple Power Analysis. Section 3 describes the Simple Power Analysis on the *Conditional calculator*, the proposed two countermeasures to mitigate this leakage, and the experimental validation of both the attack and two countermeasures. Section 4 presents the post-processing procedure and the advanced post-processing procedure, along with the experimental evaluation of both approaches. Section 5 discusses additional considerations, including applicability of pqm4 and HAWK. Section 6 concludes the paper.

2 Preliminaries

2.1 FALCON

FALCON Signature Generation. FALCON is based on the GPV hash-and-sign framework and operates over the NTRU lattice structure. As a digital signature scheme, FALCON consists of three stages: key pair generation, signature generation, and signature verification.

The degree n of the NTRU lattice is chosen as a power of two, with $n = 512$ and $n = 1024$ adopted in the NIST standard. The modulus q used for modular arithmetic must satisfy $q \equiv 1 \pmod{n}$ to enable Number Theoretic Transform (NTT) operations, which is required due to n being a power of two. In FALCON, q is set to 12289 to meet this requirement. Polynomial arithmetic over the NTRU lattice is performed modulo $\phi(x) = x^n + 1$.

Algorithm 1 NTRUGen(ϕ, q) [PFH⁺22]

Require: A monic polynomial $\phi \in \mathbb{Z}[x]$ of degree n , a modulus q

Ensure: Secret key f, g, F, G

```

1:  $\sigma_{f,g} \leftarrow 1.17\sqrt{q/2n}$ 
2: for  $i \leftarrow 0$  to  $n - 1$  do
3:    $f_i \leftarrow D_{\mathbb{Z}, \sigma_{f,g}, 0}$ 
4:    $g_i \leftarrow D_{\mathbb{Z}, \sigma_{f,g}, 0}$ 
5: end for
6:  $f \leftarrow \sum_i f_i x^i$ 
7:  $g \leftarrow \sum_i g_i x^i$ 
8: if  $NTT(f)$  contains 0 as a coefficient then
9:   restart
10: end if
11:  $\gamma \leftarrow \max\{\|(g, -f)\|, (\frac{qf^*}{ff^*+gg^*}, \frac{qg^*}{ff^*+gg^*})\}$ 
12:  $(F, G) \leftarrow \text{NTRUSolve}_{n,q}(f, g)$ 
13: if  $(F, G) = \perp$  then
14:   restart
15: end if
16: return  $f, g, F, G$ 
```

The secret key in FALCON consists of four polynomials f, g, F, G , which are short integer polynomials that satisfy the NTRU equation $fG - gF = q \pmod{\phi}$. These polynomials are generated using the procedure described in Algorithm 1. In particular, f and g are

¹https://github.com/ks950909/FALCON_CC_SPA

Algorithm 2 FALCON Sign [PFH+22]**Require:** A message m , a secret key sk , a bound $\lfloor \beta^2 \rfloor$ **Ensure:** A signature sig of m

```

1:  $r \leftarrow \{0, 1\}^{320}$  uniformly
2:  $c \leftarrow \text{HashToPoint}(r \parallel m, q, n)$ 
3:  $t \leftarrow (-\frac{1}{q} \text{FFT}(c) \odot \text{FFT}(F), \frac{1}{q} \text{FFT}(c) \odot \text{FFT}(f))$ 
4: repeat
5:   repeat
6:      $z \leftarrow \text{ffSampling}_n(t, T)$ 
7:      $s = (t - z)\hat{B}$ 
8:   until  $\|s\|^2 > \lfloor \beta^2 \rfloor$ 
9:    $(s_1, s_2) \leftarrow \text{invFFT}(s)$ 
10:   $s \leftarrow \text{Compress}(s_2, 8 \cdot \text{bytelen} - 328)$ 
11: until  $(s = \perp)$ 
12: return  $sig = (r, s)$ 

```

sampled from a discrete Gaussian distribution, as shown in lines 3 and 4 of the algorithm. Due to encoding constraints, their coefficients must lie within the range $-31 \leq f_i, g_i \leq 31$. In addition, line 11 imposes a constraint on the sum of the norms of f and g , which must not exceed $1.17 \times \sqrt{q} = \sqrt{16822.4121}$. Hence, $\sum_i f_i^2 + \sum_i g_i^2 \leq 16822.4121$ must hold. To compute the orthogonalized vector norm in line 11, the calculation is carried out in the FFT domain. The public key h in FALCON is derived from f and g , satisfying $h = gf^{-1} \bmod (\phi, q)$. Therefore, recovering either f or g is sufficient to reconstruct the entire secret key tuple f, g, F, G .

The signing process in FALCON begins by lifting the hashed message onto the lattice and performing **ffSampling** to sample a point from the lattice that follows a discrete Gaussian distribution. The difference between the sampled point and the hashed message is then compressed and used as the signature.

As shown in **Algorithm 2**, lines 3, 6 and 7 involve the FFT transformation of the secret key, indicating the use of the FFT domain during signature generation.

FFT. FALCON involves computations in the FFT domain, which necessitate FFT transformations. The operation $\text{FFT}(f)$ consists of evaluating the polynomial f at every root ζ of $\phi(x)$. Since these roots ζ are complex numbers, each $f(\zeta)$ is also a complex value, requiring an appropriate method to handle complex arithmetic in implementation. Although FFT domain operations and FFT transform involve complex arithmetic, FALCON avoids direct complex-number computation by storing the real and imaginary parts separately and performing all calculations using floating-point arithmetic. Since each root ζ of ϕ satisfies $\zeta^{-1} = \bar{\zeta}$, its complex conjugate $\bar{\zeta}$ is also a root of ϕ . Moreover, when all coefficients of f are real, it holds that $f(\bar{\zeta}) = \overline{f(\zeta)}$, allowing storage of only n real values instead of $2n$.

A non-recursive FFT implementation requires the use of a bit-reversal strategy. To minimize data movement, FALCON performs degree-1 FFT computations between index pairs spaced $n/2$ apart before beginning the main butterfly operations. These degree-1 FFTs involve real coefficients f_j and $f_{j+n/2}$, and since they reduce to the form $f_j + i \cdot f_{j+n/2}$, they can be computed without any actual floating-point arithmetic.

Numerous optimized FFT implementations have been studied, with the most widely used being the Cooley–Tukey FFT algorithm [Coo66]. This algorithm applies a divide-and-conquer strategy by recursively decomposing the FFT into smaller subproblems. The recombination of these subproblems is performed using butterfly operations, making the overall FFT computation effectively a sequence of such structures. As illustrated

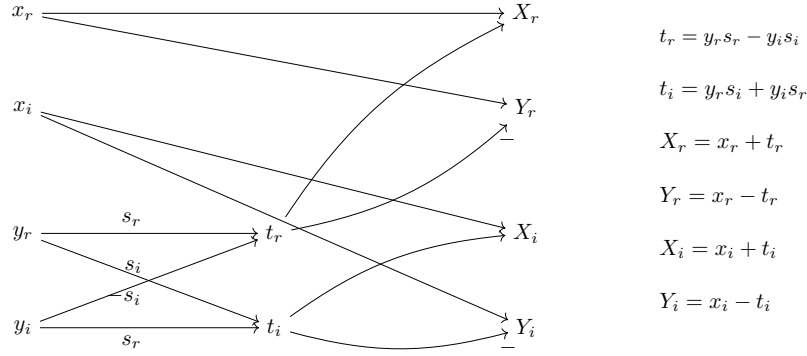


Figure 1: Butterfly operation in FALCON's reference FFT. FALCON uses a radix-2 Cooley-Tukey FFT in bit-reversed order with a half-complex polynomial representation. This figure is a schematic derived from the reference implementation: one butterfly consists of a complex twiddle multiplication (four floating-point multiplications and one floating-point additions/subtractions) followed by two complex addition and two complex subtraction for recombination. Since subtractions are implemented as additions with negated operands, the butterfly can be viewed as involving six floating-point additions (three additions and three subtractions) and four floating-point multiplications.

in Figure 1, each butterfly operation involves three floating-point additions and three floating-point subtractions. However, since subtractions are implemented as additions with negated operands, the butterfly can be viewed as comprising six floating-point addition operations and four floating-point multiplication.

The most common representation of floating-point numbers is the IEEE 754 double precision format. Each real number is represented by 64 bits: 1 bit for the sign s , 11 bits for the exponent e , and 52 bits for the mantissa m . Typically, a number is expressed in the form $(-1)^s \cdot 2^{e-1023} \cdot (1 + m \cdot 2^{-52})$.

In FALCON, for environments without floating-point unit(FPU) support, floating-point arithmetic is implemented in the software. This includes defining explicit functions for floating-point operation without relying on hardware FPU capabilities.

2.2 Simple Power Analysis

The Hamming-weight Leakage model is widely used to explain the power consumption of software implementation. This model assumes that power consumption is proportional to the Hamming weight of the intermediate value. This is expressed as a formula as follows:

$$W = a \cdot Hw(x) + b + \mathcal{N}(0, \sigma^2) \quad (1)$$

where W is the power consumption, $Hw(x)$ is the Hamming weight of the intermediate value x , a is the scalar gain between the $Hw(x)$, b is the offset, $\mathcal{N}(0, \sigma^2)$ is Gaussian random noise. Therefore, a larger difference in the Hamming weight of intermediate value leads to more distinguishable power consumption. An analysis that exploits such differences to infer intermediate values from a single power trace is known as Simple Power Analysis(SPA). To perform a Simple Power Analysis, it is necessary to correctly identify the point at which the intermediate value is calculated within the power trace. For this purpose, the Signal-to-Noise Ratio(SNR) analysis technique is widely used. SNR is typically defined as $SNR = \frac{Var(Signal)}{Var(Noise)}$. A high SNR value at a point indicates that the power consumption varies depending on the intermediate value, suggesting the corresponding operation is being performed at that point. Such analysis is effective in identifying the timing of data-dependent leakages during the preprocessing phase of Simple Power Analysis.

Statistical Modeling of Leakage and Likelihood Computation. Based on the Hamming-weight Leakage model in Equation 1, the leakage distribution for each intermediate value can be approximated as Gaussian. We therefore model the leakage l under each class as $l \mid (x = k) \sim \mathcal{N}(\mu_k, \sigma_k^2)$, where μ_k and σ_k denote the mean and standard deviation of the measured power for class k . The class-conditional likelihood is defined as $f_k(l) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left(-\frac{(l-\mu_k)^2}{2\sigma_k^2}\right)$. Assuming two classes with equal priors, the normalized posterior probabilities are given by

$$p(x = 1 \mid l) = \frac{f_1(l)}{f_0(l) + f_1(l)}, \quad p(x = 0 \mid l) = \frac{f_0(l)}{f_0(l) + f_1(l)}. \quad (2)$$

These posterior probabilities represent the confidence of each hypothesis.

Leakage target. In several constant-time routines of FALCON’s floating-point FFT, a branchless selection chooses between two values b and c depending on a bit $a \in \{0, 1\}$, which we refer to as a *Conditional calculator*. We call a the control bit. In practice, this selection is often implemented via a control mask $m = -a \in \{0, -1\}$ (all-zero vs all-one), and the large Hamming weight gap of m can be a strong leakage source. In our attack pipeline, Simple Power Analysis aims to recover these control bits (equivalently, the corresponding masks), which are then used as side-channel information for the subsequent post-processing stage.

3 Single-trace attack on FFT

In this section, we introduce a Simple Power Analysis method targeting the *Conditional calculator*, a component commonly used in both floating-point conversion and FFT transformation. We identify the locations in FALCON where the *Conditional calculator* is employed. Furthermore, countermeasures against Simple Power Analysis are presented, along with experimental results evaluating their effectiveness.

3.1 Simple Power Analysis on *Conditional calculator*

Conditional branches are essential for implementing data-dependent behavior, but they can be vulnerable to timing attacks. Therefore, cryptographic implementations avoid secret-dependent branches and instead use branchless constructs that select between two values depending on a binary condition. We call this construct a *Conditional calculator*.

For example, consider a control bit $a \in \{0, 1\}$ that selects between two values: the output equals x if $a = 0$ and y if $a = 1$. A common branchless implementation uses a control mask $m = -a \in \{0, -1\}$ and computes the selection as $x \oplus ((x \oplus y) \wedge m)$, where m becomes either all-zero or all-one.

Since m takes two extreme Hamming weights (all zeros vs all ones), the Hamming weight difference between the two cases is maximized. Under the Hamming-weight Leakage model, this maximizes the difference in the leakage variable W (see Equation 1). Consequently, operations involving the control mask (or values combined with it) can produce highly distinguishable power consumption patterns, making these *Conditional calculator* instances particularly vulnerable to Simple Power Analysis.

3.2 *Conditional calculator* in FALCON

The FFT domain operation of the secret key is used for key generation and signature generation of FALCON, so the floating-point conversion and FFT transformation of the secret key are used for key generation and signature generation of FALCON. During

Algorithm 3 Floating-point conversion from [KSSW22].

```

1 fpr fpr_scaled(int64_t i, int sc) {
2     int s, e;
3     uint32_t t;
4     uint64_t m;
5     s = (int)((uint64_t)i >> 63);
6     i ^= -(int64_t)s;
7     i += s;
8     m = (uint64_t)i;
9     e = 9 + sc;
10    FPR_NORM64(m, e);
11    m |= ((uint32_t)m & 0x1FF) + 0x1FF;
12    m >>= 9;
13    t = (uint32_t)((uint64_t)(i | -i) >> 63);
14    m &= -(uint64_t)t;
15    e &= -(int)t;
16    return FPR(s, e, m);
17 }

```

the key generation, immediately after generating the secret key, the polynomials f and g are transformed into the FFT domain to compute the orthogonalized vector norm. In the signature generation process, f and g are similarly transformed into the FFT domain to compute the Gram matrix. To convert the integer-coefficient polynomials f and g into the FFT domain, each coefficient must first be converted into a floating-point representation, followed by an FFT transformation. Therefore, floating-point conversion and FFT transformation are performed during both the key generation and signature generation processes.

Floating-point conversion. The conversion from integer to floating-point is performed using Algorithm 3 with the scaling parameter $sc = 0$. In this algorithm, lines 5–7 use s as a control bit, and line 6 derives the corresponding control mask $-s \in \{0, -1\}$. Line 10 calls Algorithm 5. Similarly, in lines 13–15, t is used as a control bit, and lines 14 and 15 derive the corresponding control masks, whose values are restricted to 0 or -1 .

Floating-point addition. The FFT transformation process includes calls to both floating-point addition and multiplication. Among these, the floating-point addition operation is defined in Algorithm 4. In this function, lines 7–11 use cs as a control bit, and line 9 derives the control mask $-cs \in \{0, -1\}$. Line 13 uses $sx \oplus sy$ as a control bit, and the computed value $-(\text{uint64_t})(sx \oplus sy)$ is the corresponding control mask, again limited to 0 or -1 . Line 14 calls Algorithm 5.

FPRNORM. Both Algorithm 3 and Algorithm 4 call Algorithm 5. Within Algorithm 5, the variable nt serves as a control bit in lines 6, 10, 14, 18, 22, and 25. In each case, the computation derives the corresponding control mask $((\text{uint64_t})nt - 1)$, whose value is restricted to either 0 or -1 .

Hence, floating-point conversion and floating-point addition use eight *Conditional calculators*.

Location of the *Conditional calculator* in FALCON. The signature generation of FALCON follows the process shown in Algorithm 6. As can be seen, both the floating-point conversion and the FFT transformation are performed not only for the four secret-key components (f, g, F, G) but also for the hashed message. This indicates that identical operations are executed across both secret and public data paths, implying that the

Algorithm 4 Floating-point addition from [KSSW22].

```

1 fpr fpr_add(fpr x, fpr y) {
2     uint64_t m, xu, yu, za;
3     uint32_t cs;
4     int ex, ey, sx, sy, cc;
5     m = ((uint64_t)1 << 63) - 1;
6     za = (x & m) - (y & m);
7     cs = (uint32_t)(za >> 63)
8         | ((1U - (uint32_t)(-za >> 63)) & (uint32_t)(x >> 63));
9     m = (x ^ y) & -(uint64_t)cs;
10    x ^= m;
11    y ^= m;
12    ...
13    xu += yu - ((yu << 1) & -(uint64_t)(sx ^ sy));
14    FPR_NORM64(xu, ex);
15    ...
16 }

```

same *Conditional calculator* function is repeatedly invoked during this process. Therefore, the leakage observed in our measurements originates from these common floating-point conversion and FFT transformation.

3.3 Countermeasure

The Simple Power Analysis introduced in Subsection 3.1 exploits the fact that the control mask stores values with significantly different Hamming weights, leading to observable power variations. Therefore, the attack can be mitigated by reducing the Hamming weight difference between the possible values of the control mask.

In scenarios where a variable takes the value x when $a = 0$ and y when $a = 1$, we propose two countermeasures that minimize the Hamming weight difference of the control mask derived from a . We propose the first countermeasure described as follows:

$$2x - y + ((y - x) \ll a). \quad (3)$$

Equation 3 is *Conditional calculator* because it satisfies the following: $2x - y + ((y - x) \ll 0) = 2x - y + (y - x) = x$ when $a = 0$, $2x - y + ((y - x) \ll 1) = 2x - y + 2(y - x) = y$ when $a = 1$. In Equation 3, the intermediate value that changes depending on the value of a is computed as a and $(y - x) \ll a$. When $a = 0$, the resulting value is 0 and $y - x$. When $a = 1$, the resulting value is 1 and $2(y - x)$. Since the Hamming weight difference between these two cases is at most 1, this countermeasure makes it difficult to recover the control bit using Simple Power Analysis.

Before introducing the second countermeasure, we define the variables b and c , such that b is randomly generated for each execution. These variables satisfy $b \oplus c = -1$, $b \wedge c = 0$, $Hw(b) = Hw(c)$, $b < c$. We propose a second countermeasure described as follows:

$$(x \oplus ((x \oplus y) \wedge b)) \oplus ((x \oplus y) \wedge (2b - c + ((c - b) \ll a))). \quad (4)$$

The expression $2b - c + (c - b) \ll a$ takes b when $a = 0$, and c when $a = 1$. Equation 4 is *Conditional calculator* because it satisfies the following: $x \oplus (x \oplus y) \wedge b \oplus (((x \oplus y) \wedge b) = x$ when $a = 0$, $x \oplus ((x \oplus y) \wedge b) \oplus ((x \oplus y) \wedge c) = x \oplus ((x \oplus y) \wedge (b \oplus c)) = x \oplus (x \oplus y) = y$. In Equation 4, the intermediate value that changes depending on the value of a is computed as a , $(c - b) \ll a$, $2b - c + ((c - b) \ll a)$, $(x \oplus y) \wedge (2b - c + (c - b) \ll a)$. When $a = 0$, the resulting value is 0, $c - b$, b and $(x \oplus y) \wedge b$. When $a = 1$, the resulting value is 0, $2(c - b)$, c and $(x \oplus y) \wedge c$. The Hamming weight difference between the two cases is at most 1 for the first three cases. Moreover, since b and c are randomly chosen and Hamming weight

Algorithm 5 FPRNORM64 from [KSSW22].

```

1  #define FPR_NORM64(m, e)    do { \
2      uint32_t nt; \
3      (e) -= 63; \
4      nt = (uint32_t)((m) >> 32); \
5      nt = (nt | -nt) >> 31; \
6      (m) ^= ((m) ^ ((m) << 32)) & ((uint64_t)nt - 1); \
7      (e) += (int)(nt << 5); \
8      nt = (uint32_t)((m) >> 48); \
9      nt = (nt | -nt) >> 31; \
10     (m) ^= ((m) ^ ((m) << 16)) & ((uint64_t)nt - 1); \
11     (e) += (int)(nt << 4); \
12     nt = (uint32_t)((m) >> 56); \
13     nt = (nt | -nt) >> 31; \
14     (m) ^= ((m) ^ ((m) << 8)) & ((uint64_t)nt - 1); \
15     (e) += (int)(nt << 3); \
16     nt = (uint32_t)((m) >> 60); \
17     nt = (nt | -nt) >> 31; \
18     (m) ^= ((m) ^ ((m) << 4)) & ((uint64_t)nt - 1); \
19     (e) += (int)(nt << 2); \
20     nt = (uint32_t)((m) >> 62); \
21     nt = (nt | -nt) >> 31; \
22     (m) ^= ((m) ^ ((m) << 2)) & ((uint64_t)nt - 1); \
23     (e) += (int)(nt << 1); \
24     nt = (uint32_t)((m) >> 63); \
25     (m) ^= ((m) ^ ((m) << 1)) & ((uint64_t)nt - 1); \
26     (e) += (int)(nt); \
27 } while (0)

```

of final value is determined by b and c , final value does not correlate with the value of a . Hence, this countermeasure makes it difficult to recover the control bit using Simple Power Analysis.

The second countermeasure is designed to ensure that no intermediate variable directly propagates the values of x or y to the output a . All data-dependent operations involving x and y are represented only as $x \oplus y$ or through randomized pairs (b, c) satisfying $b \oplus c = -1$ and $b \wedge c = 0$, which decorrelates the instantaneous power consumption from the individual operands. This design prevents an attacker from aggregating leakage across multiple *Conditional calculators* in the signing routine, where several instances are executed in sequence. When x and y are independent of any secret material, the first countermeasure provides sufficient protection with lower overhead. However, if x or y depend on the secret, the second countermeasure is preferable because it eliminates direct leakage paths related to secret-dependent intermediate values.

Performance Overhead. To evaluate the performance impact of the proposed countermeasures, we measured the number of clock cycles on an ARM Cortex-M4 microcontroller. The baseline *Conditional calculator* requires 13 cycles, while Countermeasure 1 and Countermeasure 2 take 17 and 35 cycles, respectively. When measured over the full FALCON signing process, these translate to overall overheads of approximately 2.15% and 11.83%. Although Countermeasure 2 incurs a higher local overhead due to additional bitwise operations, its overall impact on the full signing runtime remains modest. These results are summarized in Table 1.

3.4 Experiment

Experimental Setting. We acquire the power traces using the ChipWhisperer [OC14] platform. The device under test is equipped with an STM32F405RGT6 microcontroller

Algorithm 6 Sign Generation from [KSSW22]

```

1  ...
2  smallints_to_fpr(b01, f, logn); // Floating-point conversion of Secret
   Key f
3  smallints_to_fpr(b00, g, logn); // Floating-point conversion of Secret
   Key g
4  smallints_to_fpr(b11, F, logn); // Floating-point conversion of Secret
   Key F
5  smallints_to_fpr(b10, G, logn); // Floating-point conversion of Secret
   Key G
6  PQCLEAN_FALCON512_CLEAN_FFT(b01, logn); // FFT of Secret Key f
7  PQCLEAN_FALCON512_CLEAN_FFT(b00, logn); // FFT of Secret Key g
8  PQCLEAN_FALCON512_CLEAN_FFT(b11, logn); // FFT of Secret Key F
9  PQCLEAN_FALCON512_CLEAN_FFT(b10, logn); // FFT of Secret Key G
10 PQCLEAN_FALCON512_CLEAN_poly_neg(b01, logn);
11 PQCLEAN_FALCON512_CLEAN_poly_neg(b11, logn);
12 ...
13 for (u = 0; u < n; u ++) {
14     t0[u] = fpr_of(hm[u]); // Floating-point conversion of Hashed
   Message (same as smallints_to_fpr)
15 }
16 PQCLEAN_FALCON512_CLEAN_FFT(t0, logn); // FFT of Hashed Message
17 ...

```

Table 1: Cycle count and overall overhead of countermeasures.

Method	Cycles	Overhead (local)	Overhead (full)
No Countermeasure	13	–	–
Countermeasure 1	17	+30.8%	+2.15%
Countermeasure 2	35	+169.2%	+11.83%

unit with a 32-bit ARM Cortex-M4 processor and operating at clock frequency of 7.37MHz. Power traces were captured using the Chipwhisperer-Husky with a sampling rate of 29.48 MHz. To filter high-frequency noise, we applied a hardware low-pass filter with a cutoff frequency of 20MHz. The target device was programmed with a binary compiled using `arm-none-eabi-gcc 9.2.1` with the `-O0` optimization flag.

Experiment Method. Points of Interest (POI) were determined automatically by scanning for the highest SNR peaks in the traces, and manually verified to ensure consistent alignment across measurements. Each POI corresponds to the sampling point where the conditional operation exhibits the largest variation between classes.

For the Bit recovery evaluation, we compared the observed trace values at each POI with the pre-characterized mean traces for the two conditional bit values. A bit was regarded as correctly recovered if its observed value was closer to the correct class mean than to the opposite one. The same evaluation procedure was applied uniformly to all implementations discussed in the sections.

Conditional calculator. We perform a Simple Power Analysis on three versions of a *Conditional calculator*: one without any countermeasures, and two others with different countermeasures applied. For each implementation, we collected 1,000,000 power traces.

Figure 2a shows the mean traces for control mask values of 0 and -1 , and Figure 2b presents the corresponding SNR distribution. The POI used for bit classification was selected as the time sample with the highest SNR. At this POI, the distributions of the two classes are visualized in Figure 2c.

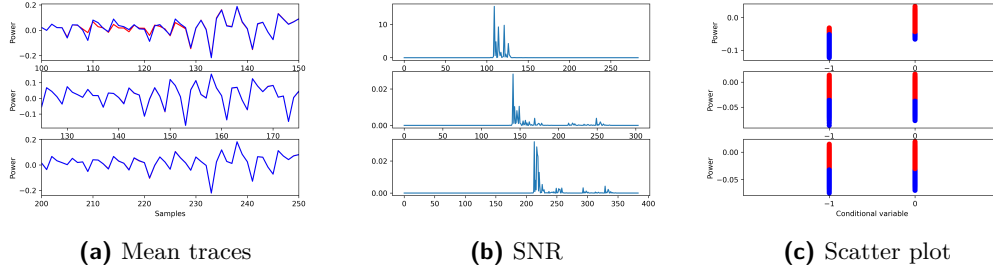


Figure 2: Experimental results of the *Conditional calculator*. Rows correspond to implementations with (top) no countermeasure, (middle) Countermeasure 1, and (bottom) Countermeasure 2. Columns correspond to (a) mean traces for $a = 0$ (red) and $a = -1$ (blue), (b) SNR distribution, and (c) scatter plot at the Point of Interest.

The success rates of bit recovery for each implementation were as follows in Table 2: 99.89% for the *Conditional calculator* without countermeasures, 56.86% for the *Conditional calculator* with the first countermeasure, and 57.41% for the *Conditional calculator* with the second countermeasure. The unprotected version exhibits a clear separation between the mean traces of the two classes, while both countermeasure implementations show strong overlap, indicating that the proposed countermeasures effectively suppress the data-dependent leakage.

In FALCON. Based on the observation that `fprscale` and `fpradd` utilize the *Conditional calculator* in Subsection 3.2, we conducted a Simple Power Analysis on these components. Power traces for `fprscale` and `fpradd` were collected over 1,000 FFT transformation stages. The SNR analysis successfully identified the Points of Interests (POIs) corresponding to the *Conditional calculator* invocations in both `fprscale` and `fpradd`, as illustrated in Figure 3a. For `fpradd`, 49,152,000 input values were analyzed across 8 instances, out of which 48,897,240 were successfully classified. For `fprscale`, 4,096,000 input values were analyzed across 8 instances, and all were successfully classified. This is a 99.52% recovery rate, as 52,993,240 out of 53,248,000 were recovered. In the scatter plot of Figure 3b, only the first two instances of `fpradd` failed to recover the control bits perfectly. Notably, indices 2, 3, and 4 of control masks are always fixed to -1 , making classification unnecessary. This occurred because the corresponding secret key coefficients f_i had absolute values less than or equal to 31, resulting in non-negative intermediate values.

Likelihood-based leakage. In the SPA experiments, each measured power sample at the Point of Interest was converted into a likelihood value representing the probability of the control mask taking the value 0 or -1 . The likelihood-based leakage was computed as defined in Equation 2. These likelihood values were then used as inputs to the subsequent key-ranking process described in Subsection 4.2.

Compiler Flags. To evaluate the robustness of the proposed attack with respect to compiler optimizations, we recompiled the target implementations with various optimization levels: `-O0`, `-O1`, `-O2`, `-O3`, and `-Os`. As shown in Table 2, recovery is reliable for binaries compiled with `-O0`, `-O1`, and `-Os`, but it degrades under `-O2` and `-O3` and is not sufficient for reliable key recovery. At higher optimization levels, the compiler removes or replaces the LDR instructions that previously caused data-dependent leakage. Consequently, the conditional operations are transformed into register-based computations, and the corresponding intermediate values no longer produce observable side-channel variations.

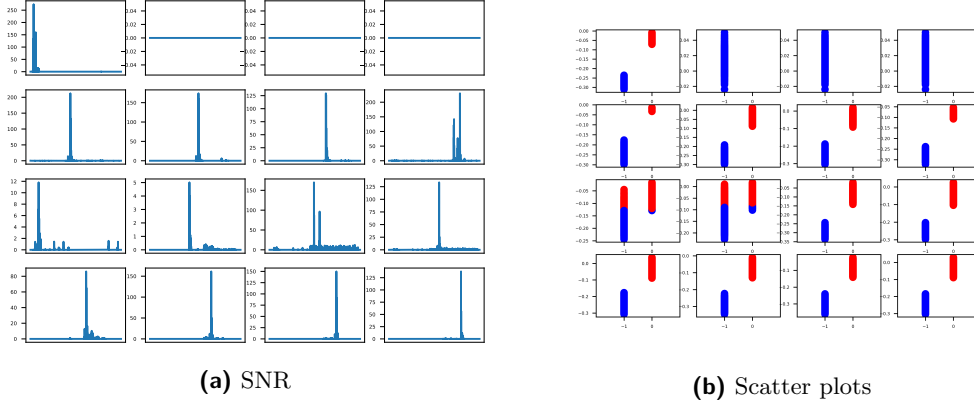


Figure 3: Experimental results of the FALCON. The upper two rows correspond to the `fprscale` operation, and the lower two rows correspond to the `fpradd` operation. Columns correspond to (a) SNR distributions and (b) scatter plots at the Point of Interest.

Table 2: Bit recovery results of the *Conditional calculator* and under different compiler optimization levels.

Method	Bit recovery (Total 1,000,000)
No Countermeasure	998,899
Countermeasure 1	568,624
Countermeasure 2	574,088
Optimization Level	Bit recovery (Total 1,000,000)
-O0	998,899
-O1	1,000,000
-O2	599,523
-O3	601,479
-Os	983,811

4 Post-Processing procedure

In Section 3, we demonstrated that control bits can be recovered using Simple Power Analysis. In this section, we propose a procedure to find secret key based on the recovered side-channel information.

4.1 Post-Processing procedure

Each coefficient of the secret key polynomials f and g satisfies $-31 \leq f_i, g_i \leq 31$ for $n = 512$, and we aim to find the full secret key such that the result is the same as the side-channel information obtained in the previous stage.

Check information. In Algorithm 5, the variable nt extracted from lines 6, 10, 14, 18, 22, and 25 indicates the increment in the exponent e through lines 7, 11, 15, 19, 23, and 26. This increment can be interpreted as the exponent growth during normalization of mantissa.

In Algorithm 3, the sign bit s is recovered from lines 5–7 to determine the sign of the integer. The exponent increment is recovered via line 10, which calls `FPRNORM`. Since the input to `FPRNORM` is fixed to 9 in this case due to $sc = 0$, recovering the exponent increment

directly reveals the exponent value. Whether a coefficient is zero or not is determined through recovery of t from lines 13–15. In [Algorithm 4](#), the comparison result of the absolute values $|x|$ and $|y|$ is recovered from lines 7–9 (encoded in the variable cs), and the sign matching information (represented as $sx \oplus sy$) is recovered from line 13. The exponent increment for the addition result is obtained from the call to `FPRNORM` at line 14. The secret key contains n coefficients, and from each invocation of `fprscale`, a total of 8 pieces of side-channel information can be extracted, amounting to $8n$ points of leakage. The FFT process consists of $(\log n - 1) \cdot n/4$ butterfly operations. Each butterfly involves 6 floating-point additions, and each addition contributes 8 pieces of side-channel information. Therefore, the FFT contributes $8 \cdot 6 \cdot (\log n - 1) \cdot n/4 = 12n \cdot (\log n - 1)$ additional pieces of leakage. Summing both sources, we obtain a total of $8n + 12n \cdot (\log n - 1)$ side-channel information. For $n = 512$, `fprscale` contributes 4,096 information, and `fpradd` contributes 49,152, yielding a total of 53,248 information.

Post-processing procedure. Since the control bits in `fprscale` and `fpradd` are uniquely determined by their operands, they are deterministic functions of the inputs. Therefore, we can define $\text{fprscale}_{\text{leak}}$ and $\text{fpradd}_{\text{leak}}$ as augmented functions that return both the computation result and the corresponding expectation leakage.

$$\text{fprscale}_{\text{leak}}(x) = (x', \text{leak}_s) \quad (5)$$

where leak_s is an 8-bit value representing the expected leakage from [Algorithm 3](#).

$$\text{fpradd}_{\text{leak}}(x, y) = (x', \text{leak}_a) \quad (6)$$

where leak_a is an 8-bit value representing the expected leakage from [Algorithm 4](#).

Since one butterfly operation consists of six calls to `fpradd`, the expected leakage from a butterfly can be expressed as a 48-bit value:

$$\text{butterfly}_{\text{leak}}(x_r, x_i, y_r, y_i, s_r, s_i) = (x'_r, x'_i, y'_r, y'_i, \text{leak}_b) \quad (7)$$

where leak_b is a 48-bit concatenation of the six leak_a from six `fpradd` invocations.

To efficiently handle the 53,248 pieces of side-channel information, we organize the data layer by layer. We denote the side-channel information obtained from `fprscale` at index j as $\text{info}_{0,j}$, and the side-channel information from the j -th butterfly in layer i as $\text{info}_{i,j}$.

The post-processing procedure proceeds as follows:

1. For each coefficient f_j , simulate $\text{fprscale}_{\text{leak}}$ with all integers in the range $[-31, 31]$, and retain only those candidates for which the resulting expected leakage matches $\text{info}_{0,j}$.
2. For each j , the remaining candidates connected to the j -th butterfly operation in Layer 1 simulate $\text{butterfly}_{\text{leak}}$, and retain only those whose expected leakage matches $\text{info}_{1,j}$.
3. For each j , candidates connected to the j -th butterfly operation in Layer 2 simulate $\text{butterfly}_{\text{leak}}$ and are retained only if their expected leakage matches $\text{info}_{2,j}$.
4. Repeat this procedure layer by layer up to Layer 8, where candidates involved in each butterfly operation are simulated and pruned based on whether their expected leakage matches the corresponding $\text{info}_{i,j}$. The remaining candidates in Layer 8 represent possible values of $FFT(f)$.

After the first step, only candidates consistent with the sign and exponent inferred from floating-point conversion leakage remain. Let $A_{0,i}$ denote the set of remaining candidates at index i :

$$A_{0,i} = \{x' \mid \text{fprscale}_{\text{leak}}(x) = (x', \text{leak}_s), -31 \leq x \leq 31, x \in \mathbb{Z}, \text{leak}_s = \text{info}_{0,i}\}$$

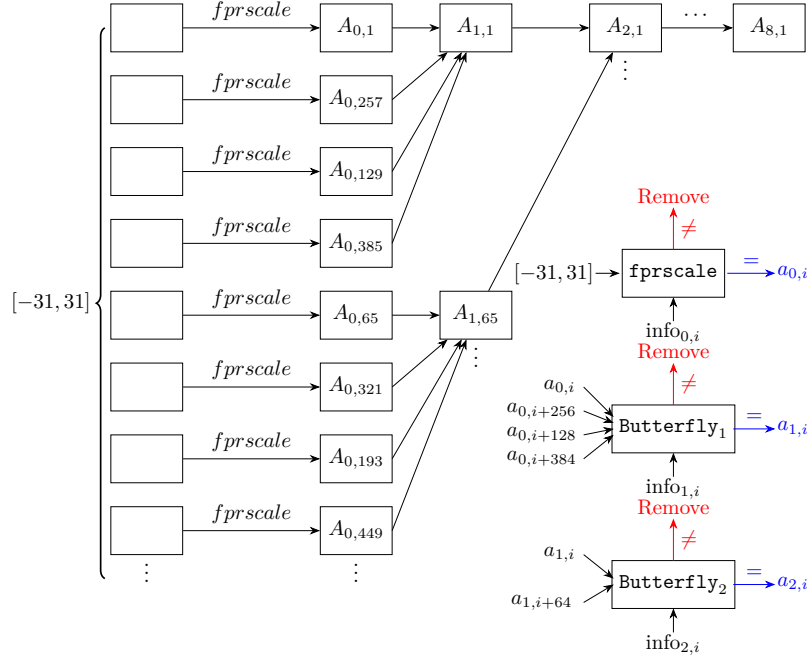


Figure 4: Post-processing procedure for recovering FFT output states. A value $a_{i,j} \in A_{i,j}$ is reconstructed using `fprscale` and `Butterfly` operations if all information up to layer i are consistent with the recovered information. Red annotations indicate candidates eliminated due to inconsistencies, while blue arrows represent transitions to the next state when all conditions match.

The total number of remaining full-key candidates is then $\prod_{i=1}^{512} |A_{0,i}|$.

After the second step, only candidates consistent with the floating-point addition leakage remain. Let $A_{1,i}$ denote the candidate set for index i after Layer 1:

$$A_{1,i} = \left\{ (x'_1, x'_2, x'_3, x'_4) \left| \begin{array}{l} \text{butterfly}_{\text{leak}}(x_1, x_2, x_3, x_4, s_{r1}, s_{i1}) = (x'_1, x'_2, x'_3, x'_4, \text{leak}_b), \\ \text{leak}_b = \text{info}_{1,i}, \\ x_1 \in A_{0,i}, x_2 \in A_{0,i+256}, x_3 \in A_{0,i+128}, x_4 \in A_{0,i+384} \end{array} \right. \right\}$$

The total number of remaining full-key candidates is then $\prod_{i=1}^{128} |A_{1,i}|$.

After the third step, candidate sets are refined using Layer 2 leakage. Let $A_{2,i}$ denote the updated candidate set:

$$A_{2,i} = \left\{ (x'_1, \dots, x'_8) \left| \begin{array}{l} \text{butterfly}_{\text{leak}}(x_1, x_2, x_5, x_6, s_{r2}, s_{i2}) = (x'_1, x'_2, x'_5, x'_6, \text{leak}_{b_1}), \\ \text{butterfly}_{\text{leak}}(x_3, x_4, x_7, x_8, s_{r3}, s_{i3}) = (x'_3, x'_4, x'_7, x'_8, \text{leak}_{b_2}), \\ \text{leak}_{b_1} = \text{info}_{2,i}, \text{leak}_{b_2} = \text{info}_{2,i+64}, \\ (x_1, x_2, x_3, x_4) \in A_{1,i}, (x_5, x_6, x_7, x_8) \in A_{1,i+64} \end{array} \right. \right\}$$

The total number of remaining full-key candidates is then $\prod_{i=1}^{64} |A_{2,i}|$.

This iterative filtering continues up to Layer 8. At the final step, index 1 holds the set $A_{8,1}$, which contains all candidate solutions in the FFT domain. These can be inverse FFT transformed and exhaustively searched to recover the secret key.

Candidate Space Estimation. To execute the post-processing procedure efficiently, the candidate set must be reduced to a computationally manageable size. The initial number

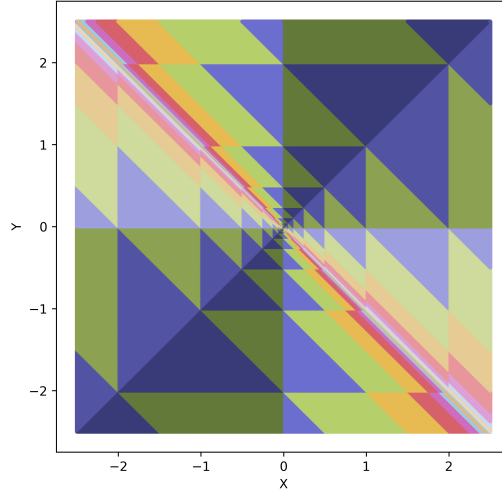


Figure 5: Floating-point addition over the range $-2.5 \leq x \leq 2.5$, $-2.5 \leq y \leq 2.5$. Different colors represent different leakage of **fpradd**.

of candidates is 63^{512} , but the post-processing procedure significantly reduces this space by eliminating candidates inconsistent with the side-channel information at each layer. Although the exact size of the pruned set after **fprscale** is input-dependent and thus difficult to compute precisely, the number of remaining candidates is bounded by $\prod_{i=1}^{512} |A_{0,i}|$. Since both the sign and exponent are fixed, we can conservatively estimate $|A_{0,i}| \leq 16$, leading to an upper bound of the number of remaining candidates is 16^{512} .

In contrast, **fpradd** reveals only the exponent increment, making it harder to identify exact values of x and y . Therefore, to estimate the reduction in candidate space, we must examine how the leakage varies with the inputs of **fpradd**. **fpradd** distinguishes between different cases based on the magnitude comparison $|x| \geq |y|$ and sign equality $sx \oplus sy$. For example, when $x \geq y > 0$, the conditions $|x| \geq |y|$ and $sx \oplus sy = 0$ holds, and the exponent increment depends on whether $x + y$ lying in the range $[x, 2x]$, which means that the exponent increment is either 0 or 1. For a given exponent k of x , if $x + y < 2^{k+1}$, the exponent increment is 0; if $x + y \geq 2^{k+1}$, the exponent increment is 1. In another case, for $x \geq -y$ with $x > 0$ and $y < 0$, we have $|x| > |y|$ and $sx \oplus sy = 1$. The sum $x + y$ lies in $[0, x]$, and the exponent increment is less than or equal to 0. Specifically, if $x + y \geq 2^k$, the exponent increment is 0; if $x + y \in [2^{k-1}, 2^k)$, the exponent increment is -1, with smaller sums resulting in even more negative increments. Note that **fpradd**(x, y) produces the same leakage as **fpradd**($-x, -y$), since they satisfy the same sign difference $sx \oplus sy$, the same absolute input values $|x|$ and $|y|$, and the same absolute sum $|x + y|$. These properties ensure that the exponent increment is identical, resulting in equivalent expectation leakage. **fpradd**(x, y) and **fpradd**(y, x) produce identical results in terms of the exponent increment and sign leakage, since $|x + y|$ and $\max(|x|, |y|)$ remain the same, and the sign difference $sx \oplus sy$ is also unchanged. However, the comparison between $|x|$ and $|y|$ is reversed, so the magnitude comparison information differs. Figure 5 visualizes the variation of expectation leakage with x, y in the range $[-2.5, 2.5]$. We observe boundary discontinuities along $x + y = \pm 2^k$, $\max(x, y) = 2^k$, $y = x$, $x = 0$, and $y = 0$. These lines correspond to transitions in the leakage behavior.

Importantly, **fpradd** leakage is invariant under scaling: if (x, y) and $(2x, 2y)$ are inputs, they yield identical leakage since comparisons on magnitude, sign, and exponent increment remain unchanged. Thus, exploring input ranges where $2^{k-1} < \max(x, y) < 2^k$ suffices to infer the behavior of **fpradd** on other inputs.

The total area of the region where $2^{k-1} < \max(x, y) < 2^k$ is $2^{2k+2} - 2^{2k}$. Among these, the largest region producing identical leakage—that is, the largest same-colored area—has an area of 2^{2k-1} . According to a Monte Carlo estimation, the probability that a randomly chosen input falls into the same leakage region—i.e., producing identical side-channel information—is at most $1/6$.

Given that **fpradd** is invoked 6,144 times, the candidate reduction factor is approximately $(\frac{1}{6})^{6144}$. Since $16^{512} \cdot (\frac{1}{6})^{6144} \approx 0$, the expected number of remaining candidates is much less than one, strongly suggesting that the post-processed set includes only the true FFT-transformed secret key.

4.2 Advanced post-processing procedure

If the side-channel information is not correctly recovered, the post-processing procedure may fail, as the correct secret key could be discarded due to mismatches with the observed leakage. The simulated leakage generated by the correct key exhibits a clear distinction from that of incorrect keys, indicating strong separability based on side-channel leakage. To improve robustness against mismatches, we rank candidates based on their cumulative agreement with the side-channel leakage and retain only those within a fixed top-rank threshold. This approach significantly enhances the success rate of key recovery, even when the side-channel leakage is partially wrong.

Post-Processing procedure with Ranking System. Instead of immediately discarding mismatched candidates, the proposed method in [Subsection 4.1](#) accumulates the number of mismatches (errors) and ranks all candidates accordingly. Only those within a predetermined rank are retained at each layer, which necessitates changes to how candidate sets are defined.

The candidate set $A_{0,i}$ is modified to record the number of differing bits between the recovered and expected leakage:

$$A_{0,i} = \{(x', k) \mid \mathbf{fprscale}(x) = (x', \mathbf{leak}_s), -31 \leq x \leq 31, x \in \mathbb{Z}, k = Hw(\mathbf{leak}_s \oplus \mathbf{info}_{0,i})\}$$

The set $A_{1,i}$ is redefined as:

$$A_{1,i} = \left\{ (x'_1, x'_2, x'_3, x'_4, k) \left| \begin{array}{l} \mathbf{butterfly}_{\text{leak}}(x_1, x_2, x_3, x_4, s_{r1}, s_{i1}) = (x'_1, x'_2, x'_3, x'_4, \mathbf{leak}_b), \\ k = k_1 + k_2 + k_3 + k_4 + Hw(\mathbf{leak}_b \oplus \mathbf{info}_{1,i}), \\ (x_1, k_1) \in A_{0,i}, (x_2, k_2) \in A_{0,i+256}, \\ (x_3, k_3) \in A_{0,i+128}, (x_4, k_4) \in A_{0,i+384} \end{array} \right. \right\}$$

Likewise, $A_{2,i}$ becomes:

$$A_{2,i} = \left\{ (x'_1, \dots, x'_8, k) \left| \begin{array}{l} \mathbf{butterfly}_{\text{leak}}(x_1, x_2, x_5, x_6, s_{r2}, s_{i2}) = (x'_1, x'_2, x'_5, x'_6, \mathbf{leak}_{b_1}), \\ \mathbf{butterfly}_{\text{leak}}(x_3, x_4, x_7, x_8, s_{r3}, s_{i3}) = (x'_3, x'_4, x'_7, x'_8, \mathbf{leak}_{b_2}), \\ k = k_1 + k_2 + Hw(\mathbf{leak}_{b_1} \oplus \mathbf{info}_{2,i}) + Hw(\mathbf{leak}_{b_2} \oplus \mathbf{info}_{2,i+64}), \\ (x_1, x_2, x_3, x_4, k_1) \in A_{1,i}, (x_5, x_6, x_7, x_8, k_2) \in A_{1,i+64} \end{array} \right. \right\}$$

The modified post-processing procedure with **rank** is summarized as follows:

1. For each coefficient f_j , simulate **fprscale**_{leak} with all integers in the range $[-31, 31]$. Rather than discarding mismatched candidates, record the number of differing bits compared to $\mathbf{info}_{0,j}$, which becomes the initial mismatch score for each candidate. If the number of candidate groups exceeds **rank**, remove the candidate groups in order of the largest mismatch score, leaving group size is **rank**.
2. For each j , simulate **butterfly**_{leak} for all candidate pairs connected to the j -th butterfly operation in Layer 1, and update the cumulative mismatch score by counting the number of differing bits against $\mathbf{info}_{1,j}$. As with step 1, only **rank** candidates with the highest score are kept, and the remaining candidates are removed.

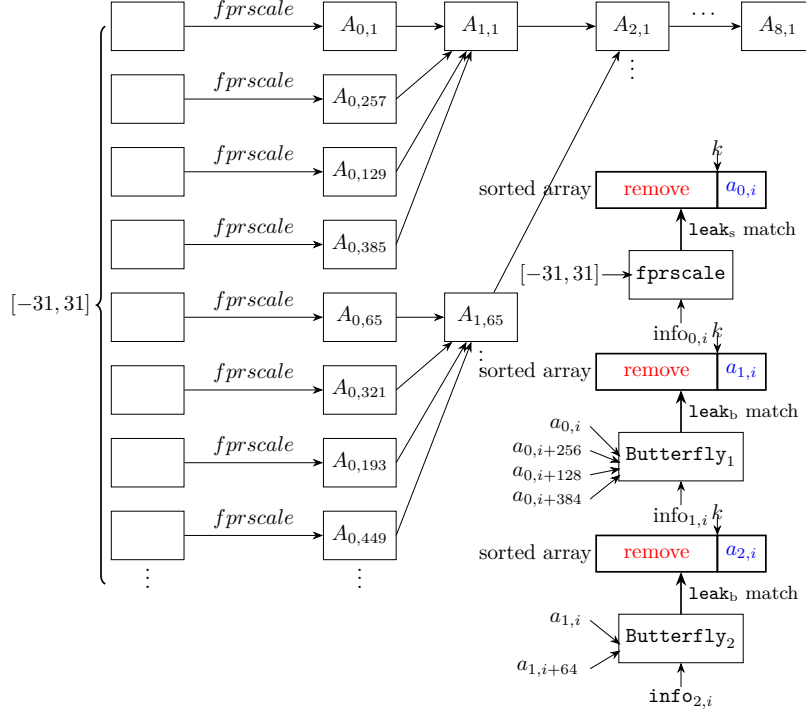


Figure 6: Advanced post-processing procedure for recovering FFT output states. Intermediate states $a_{i,j}$ are reconstructed using **fprscale** and **Butterfly** operations. Unlike the standard mode, full consistency with all control bits is not required. Instead, at each layer i , candidates are ranked based on their agreement with the recovered condition information, and only the top- k are propagated to the next layer. Red annotations indicate pruned candidates, while blue arrows represent surviving transitions.

3. For each j , simulate **butterfly_{leak}** for candidates involved in the j -th butterfly operation in Layer 2, and similarly update the cumulative mismatch score using **info_{2,j}**, keeping candidates with high **rank** scores.
4. Continue this process layer by layer up to Layer 8. At each layer, for each butterfly operation, candidates are simulated and scored based on mismatch, and only those within a fixed top-**rank** ranking based on the total accumulated mismatch are retained.

If the number of mismatches created by the correct secret key does not exceed **rank**, the secret key can be recovered, so the max rank created by the correct secret key becomes the performance of this procedure. The **rank** parameter can be tuned based on whether the computation remains possible in practice. At Layer 1, each index processes 63^4 candidates, which is computationally manageable. From Layer 2 onward, the candidate count becomes $63^8 \approx 2^{48}$, necessitating pruning.

Because butterfly operations from Layer 2 onward combine two sets, complexity grows as **rank**². The full computational cost is on the order of **rank**² · 128 · 7 · $O(\text{butterfly}_{\text{leak}})$, and the **rank** should be set such that the total computation remains possible under the available system resources.

Likelihood-Based Advanced Post-Processing. Since the individual leakages are assumed to be conditionally independent, the joint posterior probability can be expressed as $p(x_1, x_2, \dots, x_n \mid l_1, l_2, \dots, l_n) = \prod_{i=1}^n p(x_i \mid l_i)$. Accordingly, instead of counting the number of mismatches as in the original advanced post-processing method, we accumulate the posterior probabilities of each leakage. By aggregating the log-likelihoods across all samples, the final score corresponds to the posterior probability of the correct key hypothesis. This probabilistic formulation provides a

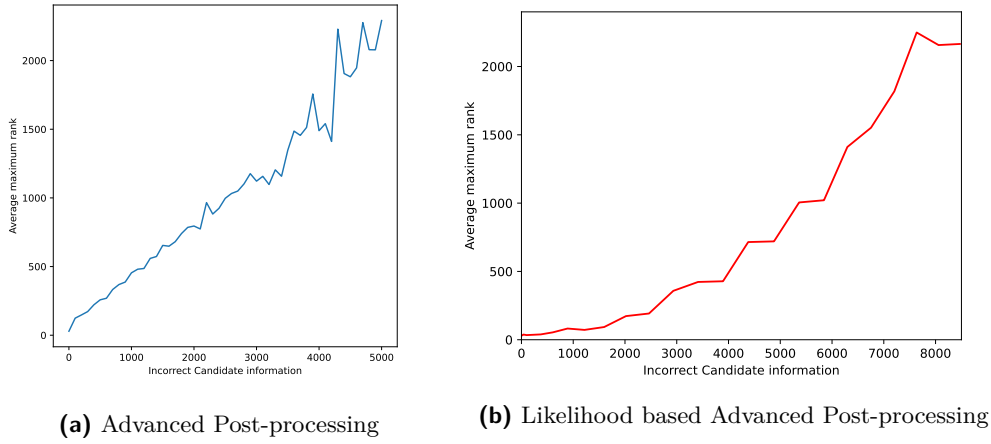


Figure 7: Effect of intentionally corrupted candidate information (i.e., recovered side-channel information entries used as inputs to the advanced post-processing) on the ranking results. We report the average of the maximum rank observed across trials.

more rigorous mathematical interpretation and improves robustness against noise compared to the deterministic approach.

4.3 Experimental Result

Key Recovery Rate. It is defined as the proportion of trials in which the correct secret key is contained in the final candidate set after performing the post-processing procedure using the already inferred side-channel information.

Post-processing. We experimentally verify the effectiveness of the post-processing procedure for key recovery.

As summarized in Table 3, the secret key was successfully recovered in all 100,000 Gaussian-sampled and 1,000 uniformly sampled tests. Uniform sampling, which produces larger key coefficients, increases the computational cost (21.0 s vs. 0.12 s on average) while maintaining the same 100% recovery success. All experiments were performed on a 12th Gen Intel® Core™i7-12700K (3.60 GHz) CPU with 64 GB RAM.

Table 3: Key recovery via post-processing procedure under different sampling distributions.

Sampling	Gaussian distribution	Uniform distribution
Number of tests	100,000	1,000
Key recovery rate	100%	100%
Key rank	1	1
Time (s)	0.12	21.02

Advanced post-processing procedure. We also investigate how robust our advanced post-processing procedure remains when some of the recovered information is incorrect. Secret key coefficients were sampled from a Gaussian distribution, and recovery was tested by varying the number of side-channel information that was intentionally corrupted. The number of incorrect entries was increased from 0 to 5,000 in increments of 100. Each corruption level was tested over 100 independent trials. For each level of corruption, we recorded the maximum rank observed in each trial and report its average across trials. Figure 7a shows that even when 5,000 out of 53,248

Table 4: Sensitivity of layer-wise post-processing to erroneous leakage bits (toy FFT, $n = 32$). The leakage extractor outputs a binary vector of length $N = 1792$. We flip w randomly chosen positions (uniformly, without replacement) and run 30 trials per setting. We use beam width $K = 10,000$, tie truncation `eq_keep` = 100, and pair-evaluation budget `max_pairs_eval` = 10^6 per block. A trial is counted as *Fail* if either (i) at least one block is skipped (infeasible due to pair explosion) or (ii) the correct path is pruned (its rank upper bound exceeds K in some layer). Note that *Fail* is the union of *Skip* and *Pruned*.

w/N	Error (%)	Fail (%)	Skip (%)	Pruned (%)
0/1792	0.0	0.0	0.0	0.0
179/1792	10.0	0.0	0.0	0.0
358/1792	20.0	53.3	36.7	30.0
448/1792	25.0	83.3	73.3	66.7
505/1792	28.2	100.0	93.3	90.0
538/1792	30.0	100.0	100.0	100.0
717/1792	40.0	100.0	100.0	100.0
896/1792	50.0	100.0	100.0	96.7

side-channel information entries were corrupted, the advanced post-processing procedure remained effective, with the average maximum rank staying around 2,291. In all test cases, the correct candidate consistently ranked first, indicating that the correct key was successfully recovered regardless of the level of corruption. This means that information can be recovered even if the side-channel information is wrong by $\frac{5000}{53248} = 9.3\%$.

To evaluate the proposed likelihood-based post-processing method, we conducted experiments using simulated traces generated from the leakage model in Equation 1. The parameters were fixed to $a = 1$ and $b = 0$, while the noise level σ was varied. For each setting, we computed the average rank of the correct candidate and the average of the maximum rank observed. For comparison with the conventional method, we assumed that the bit information was selected based on the class whose posterior probability exceeded 50%, and we derived the number of incorrect candidate information accordingly. This allows the average rank and average maximum rank to be directly compared on the same scale as those obtained by the binary decision method. The experiment was conducted 100 times, increasing σ by 1 from 1 to 32. As shown in Figure 7b, the likelihood-based approach demonstrates improved performance compared with the conventional method.

To emulate imperfect SPA outcomes (e.g., when countermeasures partially randomize the leakage of control masks), we inject errors into the leakage-bit vector and measure the impact on our layer-wise post-processing. For a toy FFT instance with $n = 32$, the leakage extractor produces a binary vector of length $N = 1792$. We flip w uniformly random positions (without replacement) and run 30 trials for each w . We set the beam width to $K = 10,000$, use `eq_keep` = 100 to limit the number of tie candidates, and impose a per-block budget `max_pairs_eval` = 10^6 ; blocks exceeding this budget are marked as skipped. A trial is considered failed if at least one block is skipped (pair explosion) or if the correct path is pruned (its rank upper bound exceeds K in some layer). As shown in Table 4, the post-processing exhibits a sharp phase transition: it succeeds in all trials up to 10% errors, but already fails in 53.3% of trials at 20% errors, and in 83.3% of trials at 25% errors. From 28.2% errors onward, all trials fail, predominantly due to infeasibility (skipped blocks) and/or pruning of the correct path. These results indicate that moderate corruption of leakage bits is sufficient to render the post-processing ineffective under the chosen computational budget.

As shown in Section 3.4, it was observed that the number of incorrect leakage entries was 254,760 over 1,000 tests. Using a rank threshold of 300, the advanced post-processing procedure achieved a 100% success rate in secret key recovery.

5 Discussion

The pqm4 implementation [KRSS19] replaces Algorithm 5 with the ARM CLZ instruction to determine the size of the mantissa. However, since the CLZ instruction computes the number of leading zeros in a single instruction, it becomes significantly more difficult to recover the exponent increment via Simple Power Analysis.

Although HAWK shares structural similarities with FALCON, their implementations differ substantially. In particular, HAWK employs fixed-point rather than floating-point arithmetic, making direct SPA on the FFT layers impractical. However, we observed that HAWK still relies on several conditional operations such as `tbmask` and `lzcnt`², which exhibit the same control-dependent leakage behavior as the *Conditional calculator* analyzed in this work. These routines return bitwise masks depending on the sign or magnitude of their inputs, and thus may leak information through power or electromagnetic traces.

As future work, we plan to extend our analysis to the pqm4 implementation to investigate potential SPA leakage caused by the CLZ instruction, and to perform a detailed study of the new conditional leakages introduced by HAWK’s `tbmask` and `lzcnt` functions.

6 Conclusion and Future Work

In this paper, we present Simple Power Analysis on FALCON and recover the secret key through a post-processing procedure using the side-channel information obtained from it. Simple Power Analysis can be performed by targeting the *Conditional calculator* used in FALCON to recover the control bit. We also present two countermeasures to prevent the recovery of control bits via Simple Power Analysis. Simple Power Analysis for the three types of *Conditional calculators* presented was performed on the STM32F405 target, and the control bits were recovered with recovery rates of 99.89%, 56.86%, and 57.41%, respectively. In addition, the *Conditional calculator* used for the FFT transform of the secret key in FALCON was targeted, and the control bits were successfully recovered with a recovery rate of 99.52% overall across `fprscale` and `fpradd`.

In addition, we present a post-processing procedure to recover the secret key of FALCON using control bits that exist as side-channel information. To tolerate incorrect recovery of side-channel information, an advanced post-processing procedure is additionally proposed to recover secret key more practically. We also performed advanced post-processing procedures according to the number of incorrect information to verify that the attack works practically, and the correct key consistently ranked first even when 5,000 pieces of information were incorrect. We also propose a likelihood-based variant that converts measured power samples into soft information for the key-ranking procedure. On the other hand, in our toy FFT experiment ($n = 32$) under the chosen computational budget, we confirm that when more than 28.2% of the leakage information is corrupted, the advanced post-processing can no longer recover the correct inputs, indicating that the proposed countermeasures ultimately increase the error beyond the attack’s tolerance. Finally, we achieved a 100% success rate in the above experiment over 1,000 FALCON secret-key pairs.

Future work includes identifying use cases of *Conditional calculators* in other post-quantum signature schemes, evaluating Simple Power Analysis on the pqm4 implementation, and developing a formal proof for the uniqueness and input-distinguishability of side-channel leakage in FFT-based computations.

Acknowledgements

This work was supported by Samsung Electronics Co., Ltd (IO251216-14690-01), in part by the National Research Foundation of Korea (NRF) funded by the Korean government (MSIT) (RS-2024-00348138), in part by the Basic Science Research Program through the NRF funded by the Ministry of Education (RS-2025-25411243), and in part by the IITP under the ITRC program funded by the Ministry of Science and ICT (IITP-2026-RS-2022-00164800).

²https://github.com/hawk-sign/dev/blob/main/src/ng_inner.h

References

- [AAB⁺22] Carlos Aguilar-Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, Jurjen Bos, Arnaud Dion, Jerome Lacan, Jean-Marc Robert, and Pascal Veron. HQC. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions>.
- [BDF⁺11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Berlin, Heidelberg, December 2011.
- [Coo66] SA Cook. On the minimum computation time for multiplication. *Doctoral diss., Harvard U., Cambridge, Mass*, 1, 1966.
- [DBL85] *IEEE standard for binary floating-point arithmetic - IEEE standard 754-1985*. Beuth, 1985.
- [FKT⁺20] Pierre-Alain Fouque, Paul Kirchner, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu. Key recovery from Gram-Schmidt norm leakage in hash-and-sign signatures over NTRU lattices. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 34–63. Springer, Cham, May 2020.
- [GMRR22] Morgane Guereau, Ange Martinelli, Thomas Ricosset, and Mélissa Rossi. The hidden parallelepiped is back again: Power analysis attacks on falcon. *IACR TCHES*, 2022(3):141–164, 2022.
- [GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.
- [GR24] Morgane Guereau and Mélissa Rossi. A not so discrete sampler: Power analysis attacks on HAWK signature scheme. *IACR TCHES*, 2024(4):156–178, 2024.
- [HBD⁺22] Andreas Hülsing, Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, Jean-Philippe Aumasson, Bas Westerbaan, and Ward Beulens. SPHINCS⁺. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [KA21] Emre Karabulut and Aydin Aysu. FALCON down: Breaking FALCON post-quantum signature scheme through side-channel attacks. In *58th ACM/IEEE Design Automation Conference, DAC 2021, San Francisco, CA, USA, December 5-9, 2021*, pages 691–696. IEEE, 2021.
- [KAA21] Emre Karabulut, Erdem Alkim, and Aydin Aysu. Single-trace side-channel attacks on ω -small polynomial sampling: With applications to ntru, NTRU prime, and CRYSTALS-DILITHIUM. In *IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2021, Tysons Corner, VA, USA, December 12-15, 2021*, pages 35–45. IEEE, 2021.
- [KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 388–397. Springer, Berlin, Heidelberg, August 1999.
- [KRSS19] Matthias J. Kannwischer, Joost Rijneveld, Peter Schwabe, and Ko Stoffelen. pqm4: Testing and benchmarking NIST PQC on ARM Cortex-M4. *Cryptology ePrint Archive*, Report 2019/844, 2019.
- [KSSW22] Matthias J. Kannwischer, Peter Schwabe, Douglas Stebila, and Thom Wiggers. Improving software quality in cryptography standardization projects. In *IEEE European Symposium on Security and Privacy, EuroS&P 2022 - Workshops, Genoa, Italy, June 6-10, 2022*, pages 19–30, Los Alamitos, CA, USA, 2022. IEEE Computer Society.

- [LDK⁺22] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [MHS⁺19] Sarah McCarthy, James Howe, Neil Smyth, Séamus Brannigan, and Máire O'Neill. BEARZ attack FALCON: implementation attacks with countermeasures on the FALCON signature scheme. In Mohammad S. Obaidat and Pierangela Samarati, editors, *Proceedings of the 16th International Joint Conference on e-Business and Telecommunications, ICETE 2019 - Volume 2: SECRIPT, Prague, Czech Republic, July 26-28, 2019*, pages 61–71. SciTePress, 2019.
- [OC14] Colin O'Flynn and Zhizhang (David) Chen. ChipWhisperer: An open-source platform for hardware embedded security research. In Emmanuel Prouff, editor, *COSADE 2014*, volume 8622 of *LNCS*, pages 243–260. Springer, Cham, April 2014.
- [PFH⁺22] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [QA25] Jinyi Qiu and Aydin Aysu. SHIFT SNARE: Uncovering secret keys in FALCON via single-trace analysis. Cryptology ePrint Archive, Report 2025/146, 2025.
- [SAB⁺22] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, Damien Stehlé, and Jintai Ding. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [WCCL22] Jian Wang, Weiqiong Cao, Hua Chen, and Haoyuan Li. Practical side-channel attack on masked message encoding in latticed-based KEM. Cryptology ePrint Archive, Report 2022/859, 2022.
- [ZLYW23] Shiduo Zhang, Xiuhua Lin, Yang Yu, and Weijia Wang. Improved power analysis attacks on falcon. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 565–595. Springer, Cham, April 2023.