

Practice Makes (Im)Perfect: A Look Back at Benchmarking Practices for Microarchitectural Side-Channel Attacks

Iliana Fayolle
iliana.fayolle@inria.fr
Univ. Lille, CNRS, Inria
Lille, France

Antoine Geimer
antoine.geimer@inria.fr
Univ. Lille, CNRS, Inria
Lille, France

Daniel De Almeida Braga
daniel.de-almeida-braga@irisa.fr
Univ. Rennes, CNRS, IRISA
Rennes, France

Clémentine Maurice
clementine.maurice@inria.fr
Univ. Lille, CNRS, Inria
Lille, France

Abstract

Microarchitectural side-channel research has grown at an exceptional pace in recent years, increasing the need for rigorous and meaningful benchmarking. Early attack papers typically relied on indirect proxies, such as covert-channel bandwidth or key-recovery on naive AES and RSA implementations, setting de facto standards that many subsequent works continued to replicate, sometimes by directly comparing against raw numbers from prior work. While these practices offer convenient points of comparison, current benchmarks may not be the most relevant to assess specific properties of new primitives. Even more problematic, microarchitectural attacks are notoriously sensitive to experimental conditions: minimal changes in the target system can significantly alter outcomes and performance. As a result, inadequate evaluation practices undermine reproducibility and cast doubt on the relevance of comparisons, even in top-tier venues where such issues should be identified.

This paper[1] tackles the core problem of proper benchmarking for microarchitectural side-channel attacks and examines its broader impact on research quality in the field. We survey 83 attack papers published in top-ranked security and architecture conferences from 2014 to 2024. From this corpus, we identify and define 19 recurrent benchmarking flaws that affect evaluation completeness, relevance, soundness, and reproducibility. These flaws include unfair or absent comparisons, missing code or materials, and the failure to evaluate the key attack properties. On average, each paper exhibits 5.5 such flaws, highlighting how widespread the issue is, even in highly selective venues. Based on our findings, we identify and suggest key properties that are relevant to properly evaluate new attacks. We also highlight trends over time and different practices between security and architecture conferences.

CCS Concepts

• **Security and privacy → Security in hardware**; **Systems security**.

Keywords

Microarchitectural; side-channel attacks; benchmarking

ACM Reference Format:
Iliana Fayolle, Antoine Geimer, Daniel De Almeida Braga, and Clémentine Maurice. 2026. Practice Makes (Im)Perfect: A Look Back at Benchmarking Practices for Microarchitectural Side-Channel Attacks . In *Proceedings of the 2026 ACM SIGSAC Conference on Computer and Communications Security (CCS '26), November 15–19, 2026, The Hague, Netherlands.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3830454.3846555

[1]Extended version: https://arxiv.org/abs/2609.03893



This work is licensed under a Creative Commons Attribution 4.0 International License.
CCS '26, The Hague, Netherlands
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2871-6/2026/11
https://doi.org/10.1145/3830454.3846555

1 Introduction

Within the last 20 years, we have witnessed a growing interest in microarchitectural side channels. Broadly speaking, contributions in this area fall into two categories: those that focus on the *software* side by uncovering and exploiting constant-time violations in real implementations [5, 7–9, 22, 23, 51], and those that focus on the *hardware* side by introducing new microarchitectural primitives or mechanisms that enable information leakage. In this work, we concentrate on the latter category, where contributions aim to characterize or exploit microarchitectural behavior directly. The novelty of an attack may stem from leveraging a previously unexplored component [1, 4], improving an existing technique [20, 65], or porting a known attack to a new architecture [33]. Regardless of the specific contribution, any work must rigorously evaluate its methodology and the performance of its various features through a well-designed benchmark. Despite the central role of evaluation in these contributions, a fundamental question remains unanswered: **How should we meaningfully evaluate and compare microarchitectural side-channel attacks?**

A first issue is that *state-of-the-art benchmarks are based on historical practices that may no longer be suitable.* They often boil down to comparing against "classical" attacks (*e.g.*, attacking an AES T-table implementation or "square-and-multiply" modular exponentiation) and trying to maximize the bit rate of a covert channel [29, 42, 63]. However, benchmarks exploiting a long-known and patched vulnerability are of limited relevance in modern implementations[2]. In practice, the vulnerability being exploited is not the point of the evaluation; it is merely a proxy to demonstrate that the side channel has sufficient resolution to attack non-constant-time cryptographic implementations. Nevertheless, this practice has become almost

[2]https://www.redhat.com/en/blog/its-all-question-time-aes-timing-attacks-openssl

universal in the community, with little reflection on whether such benchmark examples remain valid or meaningful.

A second issue is that *it is not possible to compare "apple-to-apple" different side channels based on publications alone*. Indeed, implementations vary from one paper to another, and therefore the outcome of a specific primitive may vary dramatically across implementations. For example, a covert channel that integrates an error-correction mechanism and a full protocol stack [36] cannot be meaningfully compared to one that simply transmits raw bits with a 20% error rate. Moreover, experimental conditions vary across works, and, because microarchitectural side channels are notoriously sensitive to noise, results obtained in a controlled "lab" setup cannot be meaningfully compared to results obtained in more realistic conditions, *e.g.*, in a cloud environment.

A third issue is *reproducibility*. To mitigate the second issue, researchers sometimes attempt to run state-of-the-art side channels on their experimental setup to enable meaningful comparison. However, code is not always available or working as expected. Artifact evaluations (AE) and the increasing requirement to release artifacts are gradually improving this situation, but dozens of previously published side channels still need to be reimplemented by authors—an important engineering effort that is not always well recognized. Beyond the engineering workload, academic publications frequently omit basic engineering "details", which may be considered scientifically minor but are nevertheless crucial for practical reproduction. Last, microarchitectural side channels themselves are intrinsically difficult to reproduce. For example, any change in a new processor generation may introduce undocumented modifications in components, causing an attack to stop working or requiring significant adaptation.

In this paper, we provide an overview of the evolution and current state of benchmarking practices in microarchitectural side-channel research from 2014 to 2024. We studied 83 papers published in eight leading conferences in this field. Specifically, we selected papers from top security venues (CCS, NDSS, Security & Privacy, and USENIX Security Symposium) and top architecture conferences (ASPLOS, HPCA, ISCA, and MICRO). Based on this corpus, we identify and classify the features that are commonly evaluated and emphasized, both explicitly and implicitly. We also introduce a taxonomy that captures the key properties that should be exposed and assessed when evaluating microarchitectural side channels. Finally, we provide a checklist to help researchers benchmark their attacks in a way that yields meaningful insights into the actual capabilities of their primitives. In the future, this checklist could help establish a unified benchmark for the rigorous evaluation of microarchitectural side-channel research.

In a nutshell, our contributions are the following:

(i) We systematically study 83 papers presenting microarchitectural side channels, published in top conferences from 2014 to 2024. We highlight common practices in the evaluation of microarchitectural side channels.
(ii) We highlight multiple flaws that hinder completeness, relevance, soundness, or reproducibility of microarchitectural side-channel attacks, and complicate their reproducibility.
(iii) We analyze trends over time, and differences between security and architecture communities.
(iv) We propose a set of recommendations to reduce improper benchmarking and mitigate its negative impact on research quality in the field.

We share all data and scripts in Appendix A.

2 Motivation

Our aim throughout this article is not to name and shame individual works, but to highlight recurring practices that meaningfully reduce the value of empirical evidence. The examples below are representative and motivated our work: they are in no way a judgment on the merit or quality of the cited papers.

Even within a single primitive such as Prime+Probe, initially introduced in 2005 [40], reported numbers span a wide range because platforms, protocols, and threat models differ. For instance, in [36], a cloud-oriented resilient implementation of the covert channel reporting tens of kilobytes per second with strong error correction and no observable error in a realistic setting sits alongside more controlled evaluations that may optimize the speed at the expense of error rate (see [36], Table 1). These inaccuracies may be exacerbated when comparing new results against raw numbers of older papers that lack most recent optimizations [14]. Likewise, some works [31] choose to compare to a single other work reporting a similar bandwidth, measured in a substantially different setting, to justify the relevance of their channel. Without normalizing the evaluation parameters and accounting for most recent progress, comparing these raw bandwidths is at best meaningless, and may mislead readers.

Another central source of confusion, obscuring trade-offs with other properties, is the notion of "error rate", which is defined differently across papers. Several works measure errors using the Edit distance (Levenshtein distance, or Wagner-Fisher algorithm [37]) to capture bit flips, insertions, and deletions [14, 18, 60, 68]. Others argue that using Hamming distance (which only captures bit flips) better reflects classical error-correcting capabilities [59]. Other works define error via accuracy or report an error rate without specifying the computation [19, 57]. Since Edit and Hamming distances capture different transmission failures, "x% error" conveys different meanings depending on the definition. Unless the metric and any ECC are explicitly defined, fair comparison is impossible.

3 Methodology

In this section, we present our detailed methodology, the definition of our criteria, and the design of a systematic review pipeline for selected papers. We describe our process for selecting and discarding papers, coding qualitative data while accounting for claim-specific context, and conducting statistical analyses of quantitative data. We also discuss the limitations and potential biases of our work.

3.1 Survey methodology

To conduct a consistent meta-analysis of benchmarking practices, we first designed a structured survey and a generic codebook to guide paper annotation. The survey captures recurring elements of a paper's evaluation, such as the type of primitive introduced, the stated evaluation goals, the targets and platforms considered, the reported metrics, the comparisons performed, and the availability of artifacts. The purpose of this template is to provide a common

vocabulary for analysis across papers, *not to impose a rigid checklist irrespective of the contribution.*

Our methodology follows a "paper claims first, survey criteria second" approach. For each paper, we first identified the contribution as framed by the authors and the scope of the claims supported by the evaluation. We then applied the flaw criteria conditionally, assessing whether the absence of a benchmark, comparison, or other relevant information weakened the evidence supporting the paper's stated claims. This distinction is important because microarchitectural papers do not all aim to demonstrate the same kind of result. Some papers introduce a broadly applicable primitive and therefore require evidence about portability, robustness across settings, or comparison to related work. Others intentionally focus on a mechanism tied to a specific platform, component, or microarchitecture family. In the latter case, a broader evaluation may strengthen the paper, but it is not necessarily required to support the contribution as claimed. For instance, if a paper explicitly studies a feature specific to the Apple M1, evaluating the attack on a single target is not, by itself, grounds for marking the paper as flawed under the "single platform" criterion. In such cases, the criterion is treated as not flawed, unless the paper makes broader portability or generality claims.

With this approach in mind, we developed the survey and codebook iteratively. Our group of evaluators, composed of four academic researchers with prior experience in microarchitectural side-channel research (2-10 years of experience, with multiple publications in recognized venues), first discussed the paper selection criteria and an initial list of benchmarking issues expected in the corpus. All evaluators, also designated as experts, are authors of this article. We then performed a pilot annotation on a set of papers and refined both the survey and the codebook to better match the diversity of evaluation styles encountered in practice. During the full review process, one researcher performed the initial extraction for each paper, recording both structured answers and supporting evidence from the paper, and, when relevant, from its published artifact. We did not treat information as missing when the main paper explicitly delegated implementation details, parameters, or scripts to an accompanying artifact.

To improve robustness, a second researcher then reviewed the full annotation against the paper and checked whether they reached the same conclusion. Disagreements were resolved by re-reading the relevant passages and discussing the case among the coauthors until reaching a common interpretation. Throughout this process, we adopted a conservative resolution strategy: when a criterion did not clearly apply to the paper's claims, or when the evidence was ambiguous, we preferred leniency over strictness. The codebook, used as a template for annotation, is available in the artifact (see Appendix A).

3.2 Selection of papers to review

Selecting representative papers is essential for a meaningful study of benchmarking trends. Microarchitectural side-channel attacks have become a prolific research field, especially over the last two decades. We studied papers discussing new side channels or improvements to existing methods, focusing on the microarchitectural aspect, and not the constant-time violations in specific software implementations. Specifically, we discarded any article whose core contribution is to attack a specific cryptographic implementation. Similarly, we excluded articles focusing exclusively on speculative execution attacks, such as Spectre, Meltdown, and MDS-based attacks, as they reuse existing side channels for secret exfiltration, but their core contribution is not about the side channel itself, but rather traces left by transient execution.

Microarchitectural security is mainly studied by two closely related research communities: security-focused and architecture-focused. Similar to the methodology from [53], we restricted our selection to the main conferences in each field based on their CORE2026 rankings. Although CORE ranking may not be the best metric, it is well-established as a metric influencing a paper's visibility and impact. Similarly, these venues have selective reviewing and *should* filter out contributions with obviously flawed evaluations. Ultimately, we focused on the following venues: ASPLOS, HPCA, ISCA, and MICRO for architecture conferences, and CCS, NDSS, Security & Privacy, and USENIX Security Symposium for security conferences.

We focused on articles published within the last decade, from 2014 to 2024. We chose 2014 as the starting date because it coincided with the publication of a keystone in the field: the Flush+Reload attack [65], which arguably renewed interest in the field. We ended our search in 2024 to cover the ten-year period.

We systematically scraped the proceedings of all the selected conferences, manually excluding any paper that did not evaluate a side or covert channel, yielding a corpus of 114 papers further refined to 83 to match our scope. Four experts performed this task independently and discussed any uncertainties about the scope of the papers until they reached an agreement. An extensive list of the papers selected for this study is available in the artifact (see Appendix A), sorted by year, venue, and title.

3.3 Limitations

We consider multiple limitations to our methodology. First, we ensured that experts did not evaluate their own papers. However, as experts in the field, we may still be biased due to our knowledge and experience. For example, we may be subject to opt-in or self-reporting biases. To limit these effects, we cross-checked all our annotations, with a different expert proofreading and checking for mistakes or inaccuracies. If they could not reach a consensus after discussing it, all four experts gathered to decide on the final answer.

Despite these precautions, we do not claim our list of flaws to be complete or sound, as compiling such a list is not the main goal of our paper. Our selection of top conferences might introduce some bias in our results, as it excludes papers from other well-known venues. However, this selection was necessary to keep our corpus at a manageable size, and still covers dozens of papers from the security and architecture communities. Moreover, we note that our corpus is composed of more papers from security conferences than architecture conferences, and that more recent papers are represented. While this imbalance may be the result of differing interests in side-channel attacks, it may introduce some bias in our results. We hope our sample represents the most impactful and visible papers in both fields. Similarly, our scope for included attacks may lead to some flaws being omitted, as we exclude transient execution

Table 1: Flaws and their impact on evaluation properties.

Flaw	Com.	Rel.	S.	Rep.
A1 No error rate	●	●		
A2 No spatial resolution	●			
A3 No temporal resolution	●			
A4 No evaluation of noise resilience	●	●		
A5 No evaluation against mitigations the paper claims to bypass	●	●		
A6 Evaluated on a single platform	●			●
A7 Evaluated on a single configuration	●			
B1 Unfair comparisons		●	●	
B2 No comparison		●		
B3 Evaluation designed to highlight performances		●	●	
B4 Benchmark only on highly controlled/simplified environments		●	●	
C1 Missing target/target version				●
C2 Missing CPU model	●		●	●
C3 No code / material / documentation to reproduce				●
C4 No number of samples for side-channel attacks		●		
C5 Missing details on classification of side-channel traces				●
C6 Missing protocols for covert channels	●			●
C7 Missing information on error rate computation / error correction			●	●
C8 Missing prerequisites	●			●

Com.: Completeness **Rel.**: Relevance **S.**: Soundness
Rep.: Reproducibility ● denotes an impact on the corresponding property.

attacks and papers whose novelty lies in exploiting constant-time violations in implementations. Benchmarking practices in these papers, particularly for speculative attacks, tend to be different from side-channel attacks and evaluating them deserves its own paper.

Lastly, we merged attacks with similar characteristics and targets introduced within the same paper in a single occurrence. This was necessary as some papers did not clearly distinguish the number of attacks they used for evaluation. This has a minimal impact on the total number of attacks evaluated.

4 Taxonomy of flaws

To build our taxonomy, we first define criteria expected from a good evaluation. From there, we can identify the aspects in which the various benchmarks differ from a "perfect" evaluation. Following the reasoning of van der Kouwe *et al.* [53], we expect a good evaluation to meet the following properties.

First, it should be **complete**, *i.e.*, it should properly assess all key properties of the novel attack primitive, and highlight their limitations. We emphasize that we do not consider completeness as measuring everything, but *stating and measuring the right things*. Second, it should be **relevant**, *i.e.*, conveying meaningful information without requiring inference from the reader. Third, it should be **sound**, meaning that the results must accurately measure the intended properties rather than reflecting unintended side effects or very controlled settings only. Last, the evaluation should be as **reproducible** as possible. Readers should have enough elements to reproduce the experiment and the benchmarks on their own, at least on the same device.

We adapted and complemented a list of flaws from previous work [53] to fit the specificity of our field. Based on our experience and this large-scale and systematic survey (Section 5), we define 19 flaws that may impact properties. We summarize the most impacted properties for each flaw in Table 1.

Nonetheless, it is important to note that not all papers are affected to the same extent. We used a range of labels to describe the impact of a flaw on an evaluation. We started with a generic definition of these labels and refined it based on each flaw when necessary (see the codebook in the artifact: Appendix A). Specifically, we used the following labels:

- *Ok*: the requirement is met, and the evaluation is not affected by the flaw.
- *Underspecified* - *Unclear*: the idea is discussed, but it is scattered or implicit. An informed reader could infer it, but it is not stated plainly.
- *Underspecified* - *Incomplete*: the information is partially provided, but elements are missing, or only provided for some evaluations.
- *Flawed*: clearly wrong, misleading, or does not meet the requirement.
- *NA*: Not Applicable to this paper (*e.g.*, covert-channel protocol details in a pure side-channel paper).

In the following subsection, we describe the considered flaws and their impact, grouped into three categories corresponding to the following subsections.

A. Benchmark omissions

The first category of flaws denotes any evaluation practices that may hide or skip (either intentionally or unintentionally) any experimental dimensions that affect a relevant property of the attack primitive. Microarchitectural channels live at the intersection of hardware, firmware/microcode, OS and application behavior. Any parameter across layers may interact with multiple aspects of the channel: sampling rate, thresholds, affinity, etc. However, it is impractical to exhaustively cover this space. The goal is to provide a principled evaluation that *(i)* states what is being measured, *(ii)* quantifies key trade-offs, and *(iii)* enables future normalization.

Omitting elements affecting properties such as reachable sampling frequency, noise resilience, or error rate impacts the evaluation's quality and does not allow for a proper comparison, making primitives appear universally strong. We refer to these as *benchmarking omissions*. Ultimately, benchmarking omissions obscure trade-offs that are essential for scientific comparisons.

When assessing channel reliability, reporting the error rate is key. The unit (bit/byte) and an appropriate number of trials must be explicitly stated. Vague statements such as "zero-error" are insufficient in that regard. Flaw **A1** covers such issues. Failing to report error rates allows high throughput to mask errors. This flaw mostly impacts *completeness*, but *relevance* may also be weakened since the reader cannot properly assess the attack's practicality.

Spatial and temporal resolutions are two keystones for evaluating attack potency. The former describes the smallest distinguishable unit and the mapping from observation to unit (may it be cache line, set, bank, page, port, etc.). The latter covers the time-per-sample and maximal sustainable sampling rate. Knowing the theoretical and practical limitations of such dimensions improves the reader's understanding of the attack and its potency, offering insights on how it may compare to other attacks. A lack of proper characterization of these properties is described by flaws **A2** and **A3**, respectively. In both cases, the completeness of the evaluation is impacted: key properties that should be explicit are missing, hindering comparison with other works.

While theoretical resolution is important, it is not sufficient. Quantifying a primitive's *noise resilience* under realistic events that may impact its performance provides an insight on the resilience of the attack. Indeed, a primitive working flawlessly on a target, with skyrocketing error rates as soon as another process is running may not be comparable to a more resilient attack. A lack of noise resilience evaluation (flaw **A4**) impacts both evaluation *completeness* and, to a greater degree, *relevance*. Without a performance degradation curve under system stress, we may not understand that the attack works only in quiescent setups.

Flaw **A5** is an instance of an unsubstantiated claim. Any claim unsupported by evidence should not be included in a scientific article. It *affects* both the completeness of the evaluation and its *relevance*, as the claim should be disregarded.

Portability is especially challenging in microarchitectural attacks. While seminal papers were often content with an attack working on a specific platform, it has been widely acknowledged that a generic attack should work on multiple platforms. Even when full portability is not the ultimate goal, studying which platform may be affected by a new primitive is key to conceptualizing the attack surface. In this regard, we expect key behaviors causing the attack to be replicated on different architectures, and a description of necessary adaptations. Restricting the entire evaluation to a single platform (**A6**) impacts the evaluation's *reproducibility* significantly, as well as its *completeness*.

Finally, studying various primitive configurations is required to test the attack's sensitivity to such parameters. We consider a change of configuration as a sweep of at least one impactful (*i.e.*, affecting observable outcomes) factor during the evaluation. This includes changes in the noise level, different thresholds, or window sizes, etc. This is only required to find the best settings for the attack, and we are not expecting all further evaluation based on the same primitive to be done in multiple configurations. However, if a paper introduces different attack primitives (based on different behaviors, or different components), each should be evaluated in multiple configurations. An evaluation performed on a single configuration (**A7**) would focus on a single aspect, lacking *completeness*.

B. Improper comparisons

An evaluation compares *properties*, under specific *conditions*, using some *metrics*. A proper comparison should be both qualitative and quantitative. Improper comparisons may arise when results are compared across incompatible protocols, hardware, threat model, or any measurement semantics without normalization. This issue is especially relevant in microarchitectural security: any small change in the evaluation target or setup may drastically alter observations. Combined with setup heterogeneity, it becomes particularly difficult to compare without *(i)* reproducing the attack we are comparing to, or *(ii)* explicitly normalizing and qualifying the differences.

In a sense, an unfair comparison (**B1**) may be worse than none at all (**B1**), as it creates a false sense of superiority, when it is in fact comparing different elements. On the one hand (**B1**), comparing raw numbers without accounting for a different setup, or generic improvement brought by recent works could mislead a reader into thinking a contribution is superior, while the comparison is not meaningful. This would therefore impact the *relevance* of the evaluation. In some cases, the *soundness* may be impacted, as we are no longer comparing the primitives' properties, but rather the incidental platforms (*e.g.*, different cache size, higher clock, etc.). On the other hand (**B2**), absence of comparison prevents readers from positioning the contribution in the literature, impacting the *relevance* of the evaluation. We acknowledge that first-of-its-kind attacks may not be comparable, and were labeled *NA*.

Evaluations that use "easy" payloads (*e.g.*, strong repeating patterns) or extrapolate from partial attack stages to non-trivial end-to-end scenarios are designed to highlight performance (**B3**). In the former case, repetitive payloads may trigger additional microarchitectural effects; in the latter, they may lead to incorrect conclusions due to oversimplified reasoning. This significantly impacts *soundness* and may also undermine practical *relevance*.

Similarly, benchmarking only in highly controlled or simplified setups (**B4**) also impacts the evaluation's *soundness* and *relevance*. Evaluating an attack only in idealized systems, or with prerequisites inconsistent with the threat model, may measure properties related to the setup itself, not the attack.

C. Missing information

This category of flaws prevents readers from reconstructing the experiments or understanding precisely what is measured. As stated before, minor setup differences can have cascading effects, breaking the chain from phenomenon to measurement. Hence, this category has a significant impact on *reproducibility* and, by extension, on future comparability.

Omitting OS, kernel, or targeted software version (**C1**) directly impacts *reproducibility*. Indeed, any library update could introduce subtle changes that impact the attack. Two systems with the "same browser" may implement different timer restrictions depending on the version [47].

Similarly, two machines with the "same CPU/GPU family" may behave widely differently due to microarchitectural changes. While **C1** is about the versions in the evaluation, neglecting to specify the CPU/GPU model(s) (**C2**) directly relates to the main properties defining the attack. In addition to the impact on *soundness* and *reproducibility*, it has a major impact on *completeness*.

Making attack code available (**C3**) could be considered the backbone of *reproducibility*. However, merely making an artifact available is not sufficient: attack primitives rely on implementation details, and numerous elements may need to be modified and adapted to make it work. A proper documentation should help an external reader to not only reproduce the experiment under the same conditions, but also to get an idea of what needs to be adapted to make it work on *similar* targets. Beyond reproducibility, good artifacts enable apple-to-apple comparisons by letting future works re-run baselines under the same conditions, leading to an improvement regarding **B1**.

When evaluating a side channel to demonstrate an attack's potency, it is important to document the number of samples required. This number could be reported for recovering an element (*e.g.*, how many samples are required to recover a bit of the RSA key) or for the end-to-end attack. Most attacks do not run reliably with a single measurement, so the number of samples gives an indication

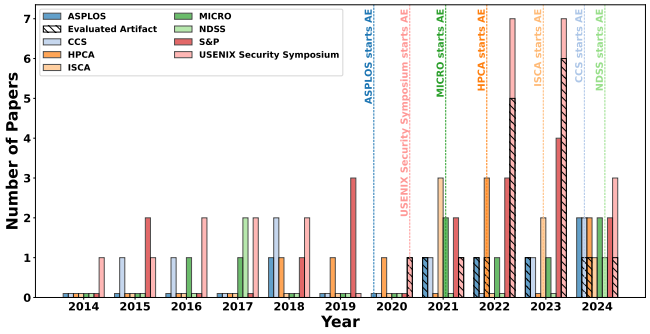


Figure 1: Number of papers and evaluated artifacts per conference per year, with the date that AE began, if applicable.

of the attack's signal-to-noise ratio. A side-channel evaluation not reporting sample numbers (**C4**) lacks some *relevance* in this regard.

Many evaluations aim at classifying instances, such as fingerprinting (website, application, etc.) or side-channel trace denoising. Classification details (model used, features, training set, etc.) are crucial to reproduce the experiments. While the primitive would still perform in the same way, omitting these elements (**C5**) has an impact on *reproducibility*.

For covert channels, the communication protocol must be well-defined and explicitly described. Missing initialization/synchronization (**C6**), whether the channel is unilateral or bilateral, framing, clocking, and initial shared knowledge have an impact on the *completeness* and *reproducibility* of the evaluation. In addition, a proper description of the protocol helps to characterize the attack model, helping comparison to other works.

Reporting the error rate (see flaw **A1**) is important, but a plain error rate is not enough as various metrics can be used to compute it. Error rates can also be computed from channel accuracy, or computed after applying various error corrections. It is thus important to describe how the error rate was obtained. Missing this (**C7**) would impact both the *soundness* and *reproducibility* of the evaluation.

Finally, some papers may introduce multiple primitives, or variants with different prerequisites (privileges, co-location, etc.). If these changes are not stated carefully (**C8**), the threat model and evaluation prerequisites may become unclear, impacting both *completeness* and *reproducibility*.

5 Survey results

In this section, we present the results of our survey on 83 papers published between 2014 and 2024 in top security and architecture conferences, as shown in Fig. 1. We analyze the prevalence of the 19 flaws in our taxonomy across the surveyed papers, as illustrated in Fig. 2. To simplify the reporting of results, we use *partial flaws* to refer to *Underspecified - Incomplete*, *Underspecified - Unclear*, or *Underspecified - Unclear & Incomplete* flaws, indicating that the information was either partially provided, scattered, or implicit.

In our corpus, we found that 77 papers (92.8%) suffer from at least one full flaw and 76 papers (91.6%) suffer from at least one partial flaw, with a median of 3 full flaws and 2 partial flaws per paper. The distribution of full and partial flaws per paper is shown in Fig. 3. On

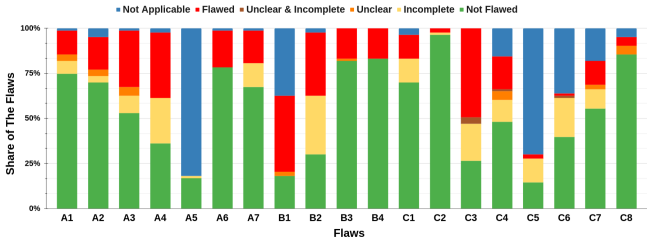


Figure 2: Prevalence of the 19 flaws in the 83 surveyed papers.

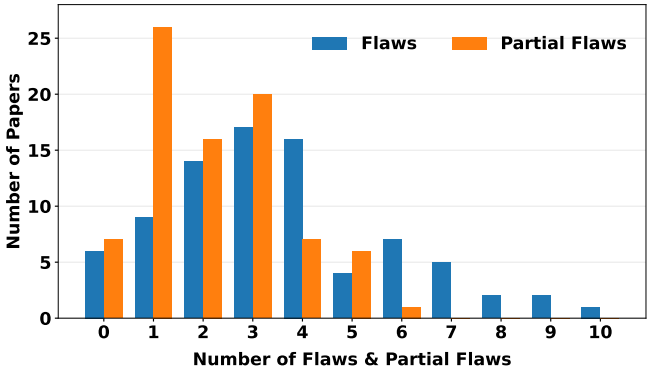


Figure 3: Distribution of the number of flaws per paper.

average, a paper suffers from 5.5 full or partial flaws. This highlights the pervasiveness of benchmarking issues in microarchitectural research. We distinguish the flaws based on their impact within each category (benchmark omissions, improper comparisons, and missing information), as described in Table 1.

After summarizing trends from Table 2, we present low-prevalence flaws, then focus on the most common ones, organized by the three categories of Section 4.

5.1 General observationCCSs

Covert-channels vs. side-channels distribution. As shown in Fig. 1, our corpus contains a diverse range of papers from various conferences and years. These papers cover different types of microarchitectural side-channel and covert-channel attacks, as well as other related contributions. Indeed, the 83 papers describe a total of 218 microarchitectural side-channel and covert-channel attacks. Among those, 147 (67.4%) are side-channel attacks, while 71 (32.6%) are covert-channel attacks. Additionally, we identified a few other types of contributions, such as improving building blocks of widespread attacks (minimal eviction set finding, performance degradation improvement), reverse engineering efforts, Rowhammer attacks, or specific microbenchmarks.

Assessment of covert-channels. For covert-channel attacks, we analyzed the most commonly reported properties of the channel, as depicted in Fig. 4. We found that 59 papers (71.1%) evaluate the raw speed of their covert channels, and 56 (67.5%) assess the error rate. Notably, 18 papers (21.7%) evaluate only speed and error rate.

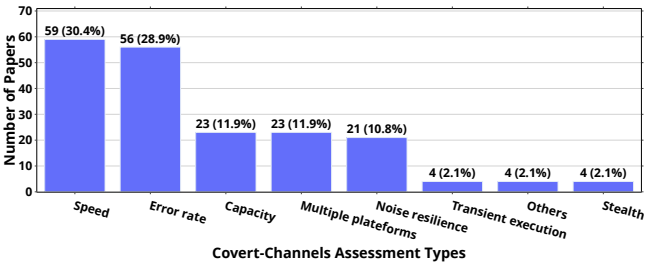


Figure 4: Distribution of reported properties in covert-channel evaluations. Percentages are normalized over all instances (*e.g.*, "Speed" represents 30.4% of all properties).

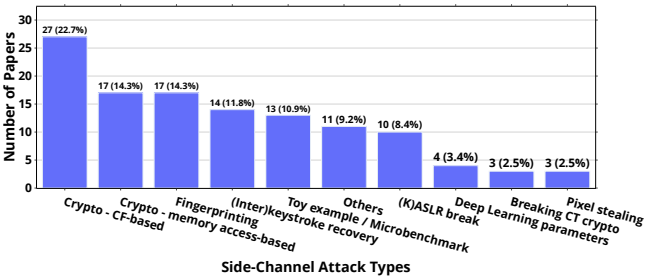


Figure 5: Distribution of side-channel attack types. Percentages show proportions across all attacks and duplicates per paper counted once (*e.g.*, "CF" represents 22.7% of all attacks).

30 papers (36.1%) evaluate either speed or error rate in conjunction with other metrics, such as noise resilience or multi-platform performance.

Types of side-channel attacks. Regarding side-channel attacks, we examined the distribution of attack types in our corpus, as illustrated in Fig. 5. In general, we found that 27 papers (32.5%) include at least one control-flow (CF) cryptographic attack, while 17 papers (20.5%) feature at least one fingerprinting attack, and 17 papers (20.5%) present at least one memory access-based cryptographic attack (*e.g.*, against AES T-table).

Most commonly targeted victims. We further analyzed the most commonly targeted victims in each of these three categories of side-channel attacks in our corpus. The results are similar to those of [24] and Lou *et al.* [35] and can be found in the artifact (see Appendix A).

Types of attacked platforms. We also analyzed the hardware platforms used in the evaluations. Unsurprisingly, most of the evaluated attacks were on CPUs (73 papers, representing 88% of the corpus). In particular, a large majority (65 papers, 78.3%) test at least one Intel CPU. In contrast, we observed that 9 papers (10.8%) evaluate their attacks on discrete GPUs, and 3 papers (3.6%) focus on integrated GPUs. Lastly, 9 papers (10.8%) include evaluations on Trusted Execution Environments (TEEs).

Table 2: Prevalence of benchmarking flaws across papers.

Flaw	Applicability	Flawed		Underspecified	
	count	count	%	count	%
A1	82	11	13.4	9	11.0
A2	79	15	19.0	6	7.6
A3	82	26	31.7	12	14.6
A4	81	30	37.0	21	25.9
A5	15	0	0	1	6.7
A6	82	17	20.7	0	0.0
A7	82	15	18.3	11	13.4
B1	52	35	67.3	2	3.8
B2	81	29	35.8	27	33.3
B3	83	14	16.9	1	1.2
B4	83	14	16.9	0	0.0
C1	80	11	13.8	11	13.8
C2	83	2	2.4	1	1.2
C3	83	41	49.4	20	24.1
C4	70	15	21.4	15	21.4
C5	25	2	8.0	11	44.0
C6	53	1	1.9	19	35.8
C7	68	11	16.2	11	16.2
C8	79	4	5.1	4	5.1

Prevalence : **rare** (< 10%), **occasional** (10%-30%), **moderate** (30%-50%), **frequent** (> 50%).

5.2 Low-prevalence flaws

While these flaws were hardly present in our corpus, they still undermine the overall quality of the evaluation. Full flaws are rare, but partial flaws are more frequent and collectively affect the trustworthiness and clarity of reported results.

A5 No evaluation against claimed mitigations they bypass. Among the 15 papers addressing this flaw, only 1 (6.7%) claims to bypass existing mitigations, yet evaluates only a subset of them, leaving the remaining claims unverified.

C2 Missing CPU model. In the entire corpus, three papers (3.6%) omit or partially report the CPU model, complicating reproducibility and generalization across platforms.

C5 Missing classification details. Among the 25 papers using classification, only 2 (8%) provide no details at all about their classification's model. Although 1 of these works describes its training and validation sets, and specifies what it evaluates, it omits essential information about the classifier itself, such as the model type, selected features, windowing strategy, or hyperparameters—making the experiments difficult to reproduce. 11 other papers (44%) report this information only partially. Overall, machine-learning-based attacks frequently lack crucial classifier details, hindering full reproducibility.

C6 Missing covert-channel protocol. Among the 53 papers addressing this flaw, only 1 (1.9%) provides no protocol details at all. However, 19 papers (35.8%) report these details only partially, typically because they describe some aspects of the protocol while omitting others, or document the protocol for one covert channel but not for another. In several cases, the description is incomplete, for instance lacking information on synchronization, calibration, or transmission steps, which further hinders reproducibility.

C8 Missing prerequisites. Among the 79 concerned papers, 8 (10.1%) fully or partially fail to indicate changes in prerequisites between attacks, typically shifts in shared-memory assumptions or platform requirements. In these cases, missing hardware information or unclear assumptions make it difficult to determine the exact conditions under which each attack operates.

5.3 A. Benchmark omissions

A1 No error rate. 13.4% of papers fully miss the error rate evaluation, meaning they did not provide any information on the error rate itself, the accuracy, the true/false positive rates, or any related metric that would give insight into the reliability of their results.

Among the 11% of papers that partially report this information, most acknowledge the presence of errors without providing any clear quantitative evaluation. Information might be hidden in other global metrics like the F1-score or the capacity. Sometimes, only a few of the benchmarked attacks of the paper report their error rate.

Other works took care to describe the evaluation of the bit error rate in detail [54] or use various kinds of error rate metrics (*e.g.*, accuracy, precision, and F-score) for each of their attacks [25].

A2 No spatial resolution. 19% of papers do not attempt to determine true spatial resolution. Some of them claim to distinguish between states of instructions, such as in a loop of various instructions or subtle variations in frequency due to power consumption, but their measurements are not granular or reproducible enough to be considered spatial resolutions. 1 paper clearly applies a prior attack with a defined spatial resolution to other cryptosystems and attacks, but the spatial resolution is not recalled, making it flawed.

Only 7.6% of papers report incomplete or unclear resolution information. Most provide descriptions (*e.g.*, uop cache usage, distinguishing seq0/seq1, "zebra vs. tiger"-type functions) suggesting spatial resolution *exists*, but without sufficient detail for proper evaluation. Some also discuss improving spatial resolution without providing numerical values or a concrete evaluation.

Papers like [10, 26, 27, 42] indicate clear spatial resolution granularity or metrics such as the cache set size, a whole cache line or the evolution of frequency of cores.

A3 No temporal resolution. 31.7% of papers omit temporal resolution or limit themselves to discussing cycles or bandwidth. However, this cannot be considered temporal resolution due to its dependence on the setup of attacks. Additionally, 1 paper applies a prior attack to other cryptosystems or attacks with the same properties without recalling its resolution.

14.6% of papers give incomplete or unclear temporal resolution, in most cases because they do not give any concrete measurements or thresholds. For example, although they refer to estimating thresholds (*e.g.*, distinguishing "1" vs. "0"), no numerical values or figures are provided. Plus, some papers summon magic numbers (*e.g.*, transmission intervals values) without evaluation or explanation.

Good examples of temporal resolution reporting appear in different forms. In [68], the authors search for the delay that maximizes error-free exfiltration speed. In [11], they dedicate a full subsection to the microbenchmark of their Inter-Processor Interrupt (IPI) latency, used as a temporal resolution. In [30], they define the theoretical transmission rate and use it for the evaluation of their attack, giving both the best theoretical bound and practical values. Another approach consists in assessing the evolution of the error rate considering various transmission rates, giving a good idea of the temporal resolution [60].

Altogether, among 79 papers concerned by flaws A2 and A3 we found that 9 papers (11.4%) miss both spatial and temporal resolutions fully or partially. These issues affect the completeness of the paper and tend to have cascading effects, rippling into other evaluation weaknesses.

A4 No evaluation of noise resilience. This flaw fully affects 37% of papers, and partially affects 25.9% of papers. Among the partially affected papers, several discuss noise conceptually or describe how it could be mitigated, but do not actually measure it using clear metrics (*e.g.*, error rates, accuracy). Some evaluate noise resilience only for a subset of their attacks, leaving others unevaluated. Others state that their attack works in noisy environments, or that higher noise requires more measurements, but do not provide any quantitative analysis to support these claims.

A good example of noise-resilience evaluation is [56] which studies four noise sources on the memory-dependent prefetcher (DMP). Likewise, [39] dedicates a full subsection to the sources, effects, and mitigations of noise on their attack.

A6 Evaluated on a single platform. 20.7% of papers evaluate their attacks on a single platform, usually a single CPU model. This lack of diversity raises concerns about the generalizability of the results to other hardware configurations, and deters reproduction efforts.

However, it is noteworthy that some papers check if their attack could be applied to platforms other than the one tested, even without a full evaluation. Others claim they did reproduce their attack on other platforms, but without giving any details about the results. Lastly, some papers evaluate their attacks on multiple CPU models from the same vendor and the same microarchitecture generation, resulting in a limited diversity and relevance of the evaluation.

[57, 69] demonstrate their attack by evaluating on multiple platforms from different vendors, showcasing the generalizability of their results. Peculiar cases were treated conservatively, as described in the codebook (see Appendix A).

A7 Evaluated on a single configuration. 31.7% of papers do not evaluate all of their attack primitives on multiple configurations. Some papers only test their attacks under a single set of system settings, such as specific CPU frequencies, memory configurations, or network conditions. This limits the relevance of the benchmark, as it does not show how the attack performs under different settings, nor how robust it is against noise, for instance.

As an example of good practice, [66] has 2 primitives, both tested under different workloads. Likewise, [41] evaluates the DRAM row-buffer-based leakage on various environments with both a covert channel and a side channel.

5.4 B. Improper comparisons

B1 Unfair comparisons to other work. 67.3% of papers provide unfair comparisons. Most flawed papers compare their results to raw numbers of prior work obtained in different settings, hardware, or threat models. In some cases, they cite other works but limit their comparison to either quantitative or qualitative data.

For 2 papers (3.8%), we are unable to conclude on the fairness of their comparisons, due to their lack of clarity or completeness. First, comparisons are made through unclear statements (*e.g.*, "our results are similar to those in this paper"). Second, they only compare parts of the properties (usually the ones at which the current work excels) but not all.

We would like to highlight that 2 papers [2, 19] explicitly acknowledge unfair comparisons (*e.g.*, due to differing setups). We therefore do not label them as flawed.

A better practice would be to reproduce prior works under the same conditions to have a fair ground for comparison, as done in [50, 67].

B2 No comparison to other work. 35.8% of papers do not provide any comparison to prior work. Often, this is due to papers putting their focus on demonstrating the mere *feasibility* of an attack through toy examples, neglecting a full comparison. For 1 paper, the evaluation's focus is on a novel mitigation to a side channel, leaving the evaluation of the attack itself with no comparison with prior work.

Moreover, 33.3% of papers only partially succeed in giving a comparison (let alone a *fair* comparison). This is mostly due to comparing only part of their evaluation, even when there is plenty of related work to compare to (*e.g.*, website fingerprinting).

For example, [28] directly compare their approach against re-implementations of Prime+Probe, Prime+Scope, and Flush+Reload in all evaluations. Similarly, [67] implement Flush+Reload and Prime+Probe as baselines for comparison.

B3 Evaluation designed to highlight performances. This flaw affects 16.9% of papers. Usually, this results from other flaws, such as A1 (no error rate); A6 (evaluation on a single platform); A7 (evaluated on a single configuration); B1 (unfair comparison); B2 (no comparison); B4 (benchmark only on highly controlled/simplified environments); C1 (missing target/target version) or C3 (no code/materials/artifact).

Still, other factors can lead to this flaw, like a toy example attack (an attack that is presented as a demonstration rather than a proper benchmark) mainly for showcasing performance. In some cases, papers claim their primitive enables various side-channel attacks but do not actually fully demonstrate any attack. We encountered one case where the attack was not carried out to completion: the paper recovered only a single byte and extrapolated the remaining ones. Likewise, we found 1 paper that reports manual error correction, which is impractical and raises concerns about bias or correctness. Some papers generalize the attack's applicability without experiments, or did covert-channel validation only on trivial message patterns (*e.g.*, sending only 1), or with the claim they can recover most of the key without providing any evidence on consistency, robustness, or reliability.

For 1 paper (1.2%), it is unclear whether the evaluation is designed to highlight performances or not, mostly because they claim to recover most of the key, with indication of where the missing bits are, but did not give any clue whether those results are consistent or even reliable.

Also, some works could initially be considered flawed, because of initial naive benchmarks or incomplete attack, but later provide strong benchmarks in a variety of contexts [50] or can justify the incomplete attack [58]. We did not consider these papers as flawed as they were transparent about the limitations of the evaluation.

B4 Benchmarks only on highly controlled/simplified systems. 16.9% of papers only benchmark their attacks in highly controlled or simplified environments. Indeed, many experiments depend on fine-tuned parameters, specific platform settings, or non-default system configurations. Plus, some attacks assume privileges or access unlikely in practice (*e.g.*, MSR, RAPL, disabling prefetchers). Others sometimes need patched or specially instrumented victim implementations, which the paper does not clearly acknowledge. We also encountered the case where the setup disables OS services, fixes CPU frequency, isolates cores, or otherwise removes noise, and sometimes justifies ignoring noise altogether.

On the attacks themselves, they might be limited or partially demonstrated (*e.g.*, key-recovery results cover only a few bits; covert-channel experiments use only simple, repeating messages; side-channel evaluations remain toy examples) to only show feasibility and not full exploitation. Due to all those aspects, it is unclear how the attacks would perform in realistic environments.

Attacks that explicitly require a specific threat model, such as an adversarial OS to target a TEE, were not considered flawed. Other examples of realistic environments can be found in [13] which attack real-word targets such as the network traffic patterns of users. Another example is [3] which shows many different benchmarks on real data (with a Raccoon attack on TLS-DH key exchanges) on different Intel architectures.

5.5 C. Missing information

C1 Missing target/target version. 27.6% of papers do not clearly identify targets and versions. Most provide only partial information on the target, or omit the driver/OS/kernel versions when they are relevant to the attack. Some refer to targets imprecisely (*e.g.*, MbedTLS "last version" or OpenSSL's "AES T-table") without giving exact version numbers. This makes reproduction difficult, as different versions may exhibit different vulnerabilities or behaviors.

Targets and their version should be clearly specified for all attacks in the paper, such as in [39, 43, 62].

C3 No code/materials/artifact. 49.4% of papers do not provide any code or material to reproduce their results. This issue manifests itself in three ways. First, there is no link to the code in the paper, nor any code on the web. Second, there is no link in the paper, but a repository exists. Third, there is a link in the paper pointing to a now inaccessible resource, and no other accessible repository exists.

For 24.1% of papers, only part of the artifact is available. Others might be missing, containing only a subset of what the paper describes (*e.g.*, demos or tools instead of full attacks) or "TBA." for many years so far. The code might also be poorly documented, making it unclear how to reproduce the experiments. As a consequence, because key artifacts are absent or incomplete, the experiments cannot be reproduced in full.

We consider that this flaw did not apply to papers that did not put a link to their repositories in their paper, but did went through AE [12, 34], or are waiting for stake-holders' approval [49].

One of the clearest examples of good reproducibility practices can be found in [32], where the authors provide a comprehensive repository containing all the necessary code and documentation (including duration and warning) to reproduce their experiments.

C4 No number of samples (side channel). 42.8% of papers do not fully provide the number of samples used in their side-channel attacks. Most of the time, there is no clear number of samples provided for all the attacks, or they only give execution times but not the sample count needed for success. For some papers with toy-examples, because those are minimal demonstrations, they appear to work with a single trace if no samples are given, but this might not be the case. We also encountered a paper where even when sample counts are mentioned, they relate only to observing an effect, not necessarily to completing a full attack. With no information about the number of samples, it becomes difficult to assess the practicality and efficiency of a side-channel attack.

Papers such as [15, 45, 49, 61] provide clear information about the number of samples used in their side-channel attacks, enhancing the reproducibility and credibility of their results.

C7 Missing error-rate computation / error correction. 32.4% of papers fully or partially miss major details about error-rate computation or error correction. Indeed, they do not explicitly explain how they compute error rate or the error correction technique they may use. 1 flawed paper justifies it by indicating that it wanted to show the conservative result (with the lower bounds). None of the other flawed papers give any justification for this omission. For those which partially miss information on error-rate computation or correction, it is mostly because they mention error rates but never specify how they are calculated. In some cases, we see unusual correction techniques like "blocks shorter than 4 samples are errors" or an 8-bit framing rule given with no justification or methodology. In others, we get no standard, systematic error-correction mechanism, or framing described in a reproducible way. Other papers state what might cause errors and how to avoid them, but provide no quantitative measurement.

Good examples of error-rate computation or correction can be found in works like [45, 46, 70] which dedicate at least a paragraph or a footnote to their error computation or correction mechanism.

6 Discussion

Beyond identifying benchmarking flaws, we also highlight several tendencies observed in our corpus. We examine various aspects of benchmarking practices, including: *(1)* the evolution in the prevalence of flaws *(2)* code and artifacts availability *(3)* trends in the number of benchmarks, platforms, and configurations evaluated per paper. Our goal is to understand how these practices have evolved over time, and how they differ between the security and architecture communities.

However, these results must be interpreted with caution, as our corpus is imbalanced, with more security papers and a greater number of recent publications. Due to the small number of architecture papers before 2020, our analysis primarily focuses on the period from 2020 to 2024, while keeping in mind the potential biases this may introduce. Due to space constraints, most of the figures supporting our claims are provided in the Artifact Appendix, in the artifact itself (see Appendix A), and in the extended version of this paper [21]. For consistency, these figures are referred to by the same figure numbers across all three resources and this paper.

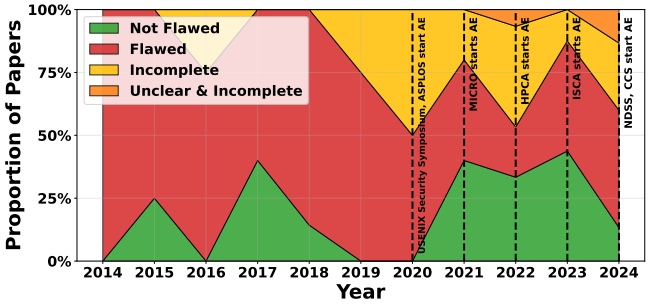


Figure 6: Availability of code, materials, and documentation over time for all conferences (with AE start).

6.1 Benchmarking practices evolution over time

Artifact evaluation. The adoption of AE by more conferences appears to have contributed to the improvement in availability of code and documentation. These improvements are clearly visible in Fig. 6, as the proportion of flawed papers decreased. However, this problem is far from fully resolved, and many papers also remain partially flawed, with only parts of the evaluation code made available. AE practices and expectations also vary from one venue to another, which would be interesting to study in future work.

General trends. We observe a slight increase in the number of platforms evaluated per paper (Fig. 10). More generally, we note that including a wide array of platforms (sometimes 10+) in the evaluation has become a well-anchored practice in recent publications. The total number of benchmarks used for evaluating primitives increased after 2020 (Fig. 12), primarily driven by a rise in benchmarks based on side-channel attacks. This may reflect reviewing practices becoming stricter for this type of paper. Conversely, interest in covert channels has declined slightly at the same time. However, we see no such trends in the number of benchmarked configurations (Fig. 11), with values varying considerably.

6.2 Communities' practices

General trends. The number of flaws per paper reveals clear differences across communities. Papers published in security venues contain fewer flaws overall (Fig. 8). They seem to show an improvement since 2020, coinciding with the introduction of AE at the USENIX Security Symposium. In contrast, architecture papers appear to exhibit more flaws (Fig. 9), possibly reflecting differing expectations from reviewers and the community on evaluation. For example, we noted that several papers provided limited details on the attack they implemented, or relied on toy examples, primarily to demonstrate feasibility. We do not see the same trend of improvement for architecture conferences, though this may be due to the lower number of papers included overall.

Evaluation. Trends across communities are particularly visible in *how* evaluations are conducted. As shown in Fig. 7, security papers tend to evaluate their side-channel primitives mostly using

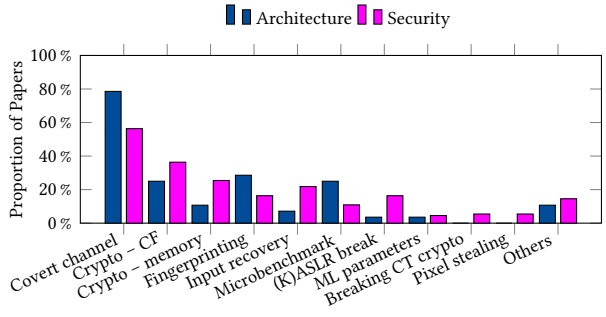


Figure 7: Distribution of papers with covert channels and side-channel attack types (Architecture vs. Security conferences).

a broad diversity of side-channel attacks. Architecture papers instead primarily use covert channels for evaluation: 78.6% evaluate at least one, compared to 56.4% for security papers. Security papers however tend to evaluate more platforms compared to architecture papers, as a way to demonstrate the robustness and applicability of their attacks (Fig. 10). Both communities rely on similar metrics for covert-channel assessment (Fig. 15): speed and error rate remain the dominant choices. However, architecture papers more readily introduce specialized or original metrics (*e.g.*, cache miss rate, stability). The increase in the number of benchmarks used for evaluation is mostly due to security papers (Fig. 13) and architecture papers to a lesser extent (Fig. 14), although this trend must be nuanced given our small corpus.

Code availability. Papers in security venues tend to share code substantially more often, although sometimes incompletely (Fig. 16). By contrast, even after the introduction of AE, more than half of architecture papers do not publish their code (Fig. 17).

7 Recommendations

Based on the previous sections, we provide recommendations on what to test when benchmarking a primitive. A good benchmark depends not only on authors but also on reviewers, program committees, and the community itself. We give recommendations for each group below.

7.1 To authors

Share the Code, Strengthen the Science. Sharing artifacts such as code is crucial for advancing science and establishing the credibility of research. It can clarify or supplement information in papers, facilitating a general understanding of experiments and results.

When authors provide access to the underlying code and materials used in their research, they enable other researchers to reproduce their results, validate their findings, and build upon their work. This transparency fosters collaboration and accelerates the pace of scientific discovery, dissemination of knowledge, and best practices. Furthermore, sharing code and materials enables the identification and correction of errors, resulting in more robust and reliable research outcomes. However, if sharing the code would be unethical (*e.g.*, if mitigations are unavailable or impossible), the author should explicitly state this, as in [49].

Resolution is the Real Baseline. This recommendation stems from flaws *A2* and *A3*. So far, there is no consensus on how a new covert-channel or side-channel attack should be benchmarked. Many studies focus on demonstrating attacks against specific cryptographic algorithms or applications (*e.g.*, AES T-table, modular exponentiation, ECDSA, (K)ASLR). However, these attacks are merely proxy metrics for the real capabilities of their primitive and can be misleading. They do not indicate how the primitive would perform in other scenarios or applications.

Generally, the greater the spatial resolution and the lower the temporal resolution are, the more impactful the primitive will be. These metrics provide a clear understanding of the primitive's capabilities and limitations, allowing for meaningful comparisons with existing work and in accordance with the recommendations from [6]. Authors should also discuss the applicability of their primitive based on its resolution, highlighting effective and ineffective scenarios. In the case where spatial or temporal resolution is not applicable, authors should clearly explain why and provide alternative metrics that better capture the primitive's performance. In any case, replacing proxy metrics with resolution metrics improves the benchmark's completeness.

Fair Comparisons Build Trust. This recommendation stems from flaws *B1*, *B2*, *B3*, and *B4*. When benchmarking a new primitive, it is crucial to provide fair and honest comparisons to prior work whenever possible. This involves re-implementing existing methods under the same or similar conditions as the new primitive to ensure a level playing field for comparison. The introduction of a standardized common benchmark, or benchmark protocol, would help provide fair evaluations. This could take the form of a framework helping authors to implement primitives, similarly to Mastik [64]. If re-implementation is not feasible, authors should clearly acknowledge the limitations of their comparisons and avoid cherry-picking favorable results.

Additionally, authors should strive to evaluate their primitive in realistic environments that reflect practical scenarios. This includes considering various system configurations, noise levels, and access privileges that an attacker might realistically have. Providing fair comparisons and realistic evaluations builds trust in the results and contributes to the advancement of the field by demonstrating the relevance and soundness of the work.

Errors Should Never Pass Silently. This recommendation stems from flaws *A1*, *A4*, *B3*, and *C7*. When benchmarking a new primitive, it is essential to thoroughly evaluate and report error rates and noise resilience. Authors should provide clear metrics on the reliability of their results, including error rates, accuracy, and related metrics.

Additionally, authors should assess how their primitive performs in the presence of noise, and quantify its impact on the results. This includes evaluating the primitive under various noise conditions and discussing any mitigation strategies employed.

Furthermore, authors should detail their error-rate computation methods and any error correction techniques employed. Transparent reporting of errors and noise resilience enhances the credibility of the benchmarks and provides valuable insights into the robustness of the primitive. This transparency improves the completeness, relevance, soundness, and reproducibility of the benchmarks.

Generalization is not a Given. This recommendation stems from flaws *A6*, *A7*, and *B4*. When benchmarking a new primitive, it is crucial to evaluate its performance across multiple platforms and configurations. This includes testing the primitive on different hardware models, OS, and system settings to assess its generalization.

Authors should also consider evaluating their primitive in various environmental conditions, such as different noise levels and access privileges. Demonstrating the robustness of their primitive across diverse scenarios provides a more comprehensive understanding of its capabilities and limitations.

Additionally, authors should clearly discuss the applicability of their primitive, highlighting scenarios in which it performs well and those in which it may struggle. This transparency helps readers assess the completeness, the practical relevance, and the soundness of the primitive.

Explicit is Better Than Implicit. This recommendation stems from flaws *C1*, *C4*, *C7*, *A2*, and *A3*. Flaws *C2*, *C5*, *C6*, and *C8* are also addressed, even though they were less prevalent in our corpus. When benchmarking a new primitive, it is essential to provide explicit and detailed information about all relevant aspects of the evaluation. This includes clearly specifying the target system and its version, the number of samples used in side-channel attacks, and the methods for error-rate computation and correction.

When possible, authors should provide comprehensive details on spatial and temporal resolution metrics. Authors should include any other pertinent information, such as classification details, covert-channel protocol specifics, and prerequisites for the attack. Being explicit about these details enhances the completeness, relevance, soundness, reproducibility, and general credibility of benchmarks, enabling others to accurately assess and build upon the work.

Every Claim Must Be Evaluated Properly. This recommendation stems from flaw *A5*, particularly applied to properties of stealthiness and detectability. Although none of the papers in our corpus exhibited this flaw, we believe it is an important consideration. If authors claim that their primitive is stealthy or difficult to detect, they must provide empirical evidence to support their assertion. This may involve conducting experiments to assess the primitive's detectability under various conditions and comparing the results to those of existing methods. Rigorously evaluating all claims strengthens the completeness and relevance of benchmarks and provides a more comprehensive understanding of a primitive's capabilities.

7.2 To reviewers and program committees

Beyond authors, reviewers in program committees play a crucial role in ensuring the quality of benchmarks and their evolution over time. They critically evaluate the benchmarks presented in submitted papers and provide constructive feedback to authors. We provide the following recommendation:

Encourage Relevant Benchmarks. This recommendation stems from flaws *A2*, *A3*, *A6*, *A7*, and *B3*. Without micro-managing authors' work, reviewers and program committees should encourage authors to focus on relevant benchmarks that provide meaningful insights into the capabilities of their primitives. Although end-to-end attacks are always appreciated, architecture papers should not prioritize them over a complete and thorough evaluation. Instead, reviewers should advocate for benchmarks that assess the fundamental properties of primitives themselves, including spatial and temporal resolution metrics, and should discourage the use of proxy metrics that may not accurately reflect the primitive's performance. Reviewers should also encourage the use of benchmarks that evaluate primitives in realistic environments and across multiple platforms and configurations. Promoting relevant benchmarks helps improve the completeness, relevance, soundness, reproducibility, and general quality of research in the field.

7.3 To the community

Last but not least, the community has a significant role in shaping benchmarking practices. The more the community emphasizes the importance of proper benchmarking, the more likely authors, reviewers, and program committees will prioritize it. We provide the following recommendations:

Establish Standardized AE. This recommendation stems from flaw *C3*. The community should work towards establishing standardized AE processes across conferences and journals, with clear guidelines and criteria to evaluate the reproducibility of research artifacts, such as code, materials, and documentation. As of 2025, all major conferences in our corpus have adopted AE, though many smaller conferences and journals have not, and even where AE exists, it is not applied to all papers. Therefore, venues should consider making AE mandatory—or provide stronger incentives—to ensure that published work meets a baseline level of transparency and reproducibility. Recently, [17, 38, 52, 55] evaluated and compared AEs for different top-ranked security and system venues and proposed best practices for their standardization.

Promote Comprehensive Benchmarking Practices. This recommendation stems from the overall findings of our study. The community should promote comprehensive benchmarking practices that encompass all relevant aspects of evaluating new primitives. This includes encouraging authors to assess spatial and temporal resolution, error rates, noise resilience, generalizability, and other relevant metrics. It should also advocate for transparent reporting of all evaluation details, including target systems, sample counts, and error correction methods. Fostering a culture of comprehensive benchmarking will help ensure that research in the field is robust, reliable, and meaningful.

8 Related Work

Recent surveys on microarchitectural side-channel attacks by Ge *et al.* [24] and Lou *et al.* [35] provide broad overviews of the field. Ge *et al.* focus on shared environments, particularly cloud settings with co-resident VMs, while Lou *et al.* adopt a more environment-independent perspective. Both categorize attack techniques, outline common mitigation strategies, and evaluate cryptographic libraries such as OpenSSL and GNU Crypto, with findings consistent with ours. However, their emphasis lies on attack classes, targets, and countermeasures rather than on benchmarking practices.

Related meta-analyses exist in other fields. Schloegel *et al.* [48] evaluated the reproducibility of fuzzing papers and their adherence to established guidelines. In our work, such pre-existing criteria do

not exist yet. Rauscher *et al.* [44] identified often overlooked evaluation properties in side-channel research (*e.g.*, hit-miss margins, topological scope, attack time, blind-spot, etc.), focusing on cache attacks similar to Flush+Reload and Prime+Probe. In contrast, we adopt a broader meta-level perspective not tied to any specific attack types. They consistently benchmark inter-keystroke timing, KASLR, and AES T-table attacks, while we advocate for consistent resolution metrics rather than reusing specific attack scenarios.

Van der Kouwe *et al.* [53] and De Meulemeester *et al.* [16] are the closest to our work methodologically, although in different fields. The former focused on benchmarking flaws in systems defense papers, deriving recommendations to improve benchmark quality and scientific rigor. The latter focused on hardware-cost evaluation strategies, highlighted reporting deficiencies, and proposed a uniform evaluation methodology. Our recommendations draw inspiration from the SIGPLAN empirical evaluation checklist [6], adapted to microarchitectural side-channel attacks and their specific benchmarking flaws.

9 Conclusion

In this work, we conducted a comprehensive survey of 83 papers and identified 19 benchmarking flaws that undermine the validity of microarchitectural side-channel and covert-channel evaluations. No paper in our corpus is free of flaws, with an average of 5.5 issues per paper. Based on these findings, we provide recommendations for authors, reviewers, program committees, and the broader community to improve benchmarking practices.

Our analysis highlights the need to rethink how evaluations are designed, conducted, and assessed. In particular, we advocate wider adoption of resolution metrics, which provide an attack-agnostic way to characterize leakage. When used rigorously, these metrics enable meaningful evaluation even for weaker or non–end-to-end attacks and can fully support a paper's empirical contribution. Improving benchmarking is not solely the responsibility of authors: reviewers and program committees play a crucial role in enforcing sound methodology and clear reporting standards. Reproducibility is equally important—sharing code, artifacts, and experimental material reduces the likelihood of flawed evaluations and increases the reliability of published results. We hope that our work supports the community in moving toward more robust and transparent benchmarking practices for microarchitectural side-channel and covert-channel attacks and ultimately encourages progress toward a unified, widely accepted benchmark suite for this domain.

Acknowledgments

This work benefited from the support of the AID, ANR-19-CE39-0007 MIAOUS, and ANR-21-CE39-0019/Deutsche Forschungsgemeinschaft (DFG) 491039149 FACADES.

References

[1] Onur Aciiçmez, Çetin Kaya Koç, and Jean-Pierre Seifert. 2007. On the power of simple branch prediction analysis. In *Proceedings of the 2nd ACM symposium on Information, computer and communications security*. 312–320.

[2] Jaeguk Ahn, Jiho Kim, Hans Kasan, Zhixian Jin, Leila Delshadtehrani, WonJun Song, Ajay Joshi, and John Kim. 2021. Network-on-Chip Microarchitecture-based Covert Channel in GPUs. In *MICRO*. ACM, 565–577.

[3] Alejandro Cabrera Aldaya and Billy Bob Brumley. 2022. HyperDegrade: From GHz to MHz Effective CPU Frequencies. In *USENIX Security Symposium*. USENIX Association, 2801–2818.

[4] Alejandro Cabrera Aldaya, Billy Bob Brumley, Sohaib ul Hassan, Cesar Pereida García, and Nicola Tuveri. 2019. Port Contention for Fun and Profit. In *IEEE Symposium on Security and Privacy*. IEEE, 870–887.

[5] Naomi Benger, Joop van de Pol, Nigel P. Smart, and Yuval Yarom. 2014. "Ooh Aah... Just a Little Bit" : A Small Amount of Side Channel Can Go a Long Way. In *CHES (Lecture Notes in Computer Science, Vol. 8731)*. Springer, 75–92.

[6] Steve Blackburn, Matthias Hauswirth, Emery Berger, Michael Hicks, and Shriram Krishnamurthi. 2017. SIGPLAN Empirical Evaluation Checklist. https://raw.githubusercontent.com/SIGPLAN/empirical-evaluation/master/checklist/checklist.pdf.

[7] Daniel De Almeida Braga, Pierre-Alain Fouque, and Mohamed Sabt. 2020. Dragonblood is Still Leaking: Practical Cache-based Side-Channel in the Wild. In *ACSAC*. ACM, 291–303.

[8] Daniel De Almeida Braga, Pierre-Alain Fouque, and Mohamed Sabt. 2021. PARASITE: PAssword Recovery Attack against Srp Implementations in ThE wild. In *CCS*. ACM, 2497–2512.

[9] Daniel De Almeida Braga, Natalia Kulatova, Mohamed Sabt, Pierre-Alain Fouque, and Karthikeyan Bhargavan. 2023. From Dragondoom to Dragonstar: Side-channel Attacks and Formally Verified Implementation of WPA3 Dragonfly Handshake. In *EuroS&P*. IEEE, 707–723.

[10] Samira Briongos, Pedro Malagón, José Manuel Moya, and Thomas Eisenbarth. 2020. RELOAD+REFRESH: Abusing Cache Replacement Policies to Perform Stealthy Cache Attacks. In *USENIX Security Symposium*. USENIX Association, 1967–1984.

[11] Jo Van Bulck, Nico Weichbrodt, Rüdiger Kapitza, Frank Piessens, and Raoul Strackx. 2017. Telling Your Secrets without Page Faults: Stealthy Page Table-Based Attacks on Enclaved Execution. In *USENIX Security Symposium*. USENIX Association, 1041–1056.

[12] Boru Chen, Yingchen Wang, Pradyumna Shome, Christopher W. Fletcher, David Kohlbrenner, Riccardo Paccagnella, and Daniel Genkin. 2024. GoFetch: Breaking Constant-Time Cryptographic Implementations Using Data Memory-Dependent Prefetchers. In *USENIX Security Symposium*. USENIX Association.

[13] Yun Chen, Ali Hajiabadi, Lingfeng Pei, and Trevor E. Carlson. 2024. PREFETCHX: Cross-Core Cache-Agnostic Prefetcher-based Side-Channel Attacks. In *HPCA*. IEEE, 395–408.

[14] Yujie Cui, Chun Yang, and Xu Cheng. 2022. Abusing Cache Line Dirty States to Leak Information in Commercial Processors. In *HPCA*. IEEE, 82–97.

[15] Miles Dai, Riccardo Paccagnella, Miguel Gomez-Garcia, John D. McCalpin, and Mengjia Yan. 2022. Don't Mesh Around: Side-Channel Attacks and Mitigations on Mesh Interconnects. In *USENIX Security Symposium*. USENIX Association, 2857–2874.

[16] Jesse De Meulemeester, Quinten Norga, Frank Piessens, Ingrid Verbauwhede, and Marton Bognar. 2025. Hardware Cost Evaluation in Systems Security. In *Proceedings of the 3rd ACM Conference on Reproducibility and Replicability*. ACM, 229–233.

[17] Daniele Cono D'Elia, Thaleia Dimitra Doudali, Cristiano Giuffrida, Miguel Matos, Mathias Payer, Solal Pirelli, Georgios Portokalidis, Valerio Schiavoni, Salvatore Signorello, and Anjo Vahldiek-Oberwagner. 2025. Lessons Learned from Five Years of Artifact Evaluations at EuroSys. In *ACM REP*. ACM, 108–120.

[18] Shuwen Deng, Bowen Huang, and Jakub Szefer. 2022. Leaky Frontends: Security Vulnerabilities in Processor Frontends. In *HPCA*. IEEE, 53–66.

[19] Sankha Baran Dutta, Hoda Naghibijouybari, Arjun Gupta, Nael B. Abu-Ghazaleh, Andres Marquez, and Kevin J. Barker. 2023. Spy in the GPU-box: Covert and Side Channel Attacks on Multi-GPU Systems. In *ISCA*. ACM, 45:1–45:13.

[20] Dmitry Evtyushkin, Ryan Riley, Nael CSE Abu-Ghazaleh, ECE, and Dmitry Ponomarev. 2018. Branchscope: A new side-channel attack on directional branch predictor. *ACM SIGPLAN Notices* 53, 2 (2018), 693–707.

[21] Iliana Fayolle, Antoine Geimer, Daniel De Almeida Braga, and Clémentine Maurice. 2026. Practice Makes (Im) Perfect: A Look Back at Benchmarking Practices for Microarchitectural Side-Channel Attacks. *arXiv preprint arXiv:2609.03893* (2026).

[22] Cesar Pereida García and Billy Bob Brumley. 2017. Constant-Time Callees with Variable-Time Callers. In *USENIX Security Symposium*. USENIX Association, 83–98.

[23] Cesar Pereida García, Billy Bob Brumley, and Yuval Yarom. 2016. "Make Sure DSA Signing Exponentiations Really are Constant-Time". In *CCS*. ACM, 1639–1650.

[24] Qian Ge, Yuval Yarom, David A. Cock, and Gernot Heiser. 2016. A Survey of Microarchitectural Timing Attacks and Countermeasures on Contemporary Hardware. *IACR Cryptol. ePrint Arch.* (2016), 613.

[25] Lukas Gerlach, Daniel Weber, Ruiyi Zhang, and Michael Schwarz. 2023. A Security RISC: Microarchitectural Attacks on Hardware RISC-V CPUs. In *IEEE Symposium on Security and Privacy*. IEEE, 2321–2338.

[26] Daniel Gruss, Raphael Spreitzer, and Stefan Mangard. 2015. Cache Template Attacks: Automating Attacks on Inclusive Last-Level Caches. In *USENIX Security Symposium*. USENIX Association, 897–912.

[27] Yanan Guo, Dingyuan Cao, Xin Xin, Youtao Zhang, and Jun Yang. 2023. Uncore Encore: Covert Channels Exploiting Uncore Frequency Scaling. In *MICRO*. ACM,

843–855.
[28] Yanan Guo, Xin Xin, Youtao Zhang, and Jun Yang. 2022. Leaky Way: A Conflict-Based Cache Covert Channel Bypassing Set Associativity. In *MICRO*. IEEE, 646–661.
[29] Daniel Katzman, William Kosasih, Chitchanok Chuengsatiansup, Eyal Ronen, and Yuval Yarom. 2023. The Gates of Time: Improving Cache Attacks with Transient Execution. In *USENIX Security Symposium*. USENIX Association, 1955–1972.
[30] S. Karen Khatamifard, Longfei Wang, Amitabh Das, Selçuk Köse, and Ulya R. Karpuzcu. 2019. POWERT Channels: A Novel Class of Covert Communication Exploiting Power Management Vulnerabilities. In *HPCA*. IEEE, 291–303.
[31] Sowoong Kim, Myeonggyun Han, and Woongki Baek. 2022. DPrime+DAbort: A High-Precision and Timer-Free Directory-Based Side-Channel Attack in Non-Inclusive Cache Hierarchies Using Intel TSX. In *HPCA*. IEEE, 67–81.
[32] Andreas Kogler, Jonas Juffinger, Lukas Giner, Lukas Gerlach, Martin Schwarzl, Michael Schwarz, Daniel Gruss, and Stefan Mangard. 2023. Collide+Power: Leaking Inaccessible Data with Software-based Power Side Channels. In *USENIX Security Symposium*. USENIX Association, 7285–7302.
[33] Moritz Lipp, Daniel Gruss, Raphael Spreitzer, Clémentine Maurice, and Stefan Mangard. 2016. ARMageddon: Cache Attacks on Mobile Devices. In *USENIX Security Symposium*. USENIX Association, 549–564.
[34] Sihang Liu, Suraaj Kanniwadi, Martin Schwarzl, Andreas Kogler, Daniel Gruss, and Samira Manabi Khan. 2023. Side-Channel Attacks on Optane Persistent Memory. In *USENIX Security Symposium*. USENIX Association, 6807–6824.
[35] Xiaoxuan Lou, Tianwei Zhang, Jun Jiang, and Yinqian Zhang. 2022. A Survey of Microarchitectural Side-channel Vulnerabilities, Attacks, and Defenses in Cryptography. *ACM Comput. Surv.* 54, 6 (2022), 122:1–122:37.
[36] Clémentine Maurice, Manuel Weber, Michael Schwarz, Lukas Giner, Daniel Gruss, Carlo Alberto Boano, Stefan Mangard, and Kay Römer. 2017. Hello from the Other Side: SSH over Robust Cache Covert Channels in the Cloud. In *NDSS*. The Internet Society.
[37] Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM computing surveys (CSUR)* 33, 1 (2001), 31–88.
[38] Daniel Olszewski, Allison Lu, Anna Crowder, Nathaniel Bennett, Seth Layton, Sri Hrushikesh Varma Bhupathiraju, Tyler Tucker, Siddhant Kalgutkar, Hunter Ver Helst, Carson Stillman, Kevin R. B. Butler, Sara Rampazzi, and Patrick Traynor. 2025. Reproducibility in Applied Security Conferences: An 11-Year Review on Artifacts and Evaluation Committees. In *ACM REP*. ACM, 96–107.
[39] Yossef Oren, Vasileios P. Kemerlis, Simha Sethumadhavan, and Angelos D. Keromytis. 2015. The Spy in the Sandbox: Practical Cache Attacks in JavaScript and Their Implications. In *CCS*. ACM, 1406–1418.
[40] Dag Arne Osvik, Adi Shamir, and Eran Tromer. 2006. Cache Attacks and Countermeasures: The Case of AES. In *CT-RSA (Lecture Notes in Computer Science, Vol. 3860)*. Springer, 1–20.
[41] Peter Pessl, Daniel Gruss, Clémentine Maurice, Michael Schwarz, and Stefan Mangard. 2016. DRAMA: Exploiting DRAM Addressing for Cross-CPU Attacks. In *USENIX Security Symposium*. USENIX Association, 565–581.
[42] Antoon Purnal, Furkan Turan, and Ingrid Verbauwhede. 2021. Prime+Scope: Overcoming the Observer Effect for High-Precision Cache Contention Attacks. In *CCS*. ACM, 2906–2920.
[43] Antoon Purnal, Furkan Turan, and Ingrid Verbauwhede. 2022. Double Trouble: Combined Heterogeneous Attacks on Non-Inclusive Cache Hierarchies. In *USENIX Security Symposium*. USENIX Association, 3647–3664.
[44] Fabian Rauscher, Carina Fiedler, Andreas Kogler, and Daniel Gruss. 2025. A Systematic Evaluation of Novel and Existing Cache Side Channels. In *NDSS*. The Internet Society.
[45] Fabian Rauscher, Andreas Kogler, Jonas Juffinger, and Daniel Gruss. 2024. IdleLeak: Exploiting Idle State Side Effects for Information Leakage. In *NDSS*. Internet Society, San Diego, CA, USA. doi:10.14722/ndss.2024.24078
[46] Xida Ren, Logan Moody, Mohammadkazem Taram, Matthew Jordan, Dean M. Tullsen, and Ashish Venkat. 2021. I See Dead *M*ops: Leaking Secrets via Intel/AMD Micro-Op Caches. In *ISCA*. IEEE, 361–374.
[47] Thomas Rokicki, Clémentine Maurice, and Pierre Laperdrix. 2021. Sok: In search of lost time: A review of javascript timers in browsers. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 472–486.
[48] Moritz Schloegel, Nils Bars, Nico Schiller, Lukas Bernhard, Tobias Scharnowski, Addison Crump, Arash Ale Ebrahim, Nicolai Bissantz, Marius Muench, and Thorsten Holz. 2024. SoK: Prudent Evaluation Practices for Fuzzing. In *SP*. IEEE, 1974–1993.
[49] Mingtian Tan, Junpeng Wan, Zhe Zhou, and Zhou Li. 2021. Invisible Probe: Timing Attacks with PCIe Congestion Side-channel. In *IEEE Symposium on Security and Privacy*. IEEE, 322–338.
[50] Andrei Tatar, Daniël Trujillo, Cristiano Giuffrida, and Herbert Bos. 2022. TLB;DR: Enhancing TLB-based Attacks with TLB Desynchronized Reverse Engineering. In *USENIX Security Symposium*. USENIX Association, 989–1007.
[51] Sohaib ul Hassan, Iaroslav Gridin, Ignacio M. Delgado-Lozano, Cesar Pereida García, Jesús-Javier Chi-Domínguez, Alejandro Cabrera Aldaya, and Billy Bob Brumley. 2020. Déjà Vu: Side-Channel Analysis of Mozilla's NSS. In *CCS*. ACM, 1887–1902.
[52] Anjo Vahldiek-Oberwagner, Marton Bognar, and Salvatore Signorello. 2026. ReproDB: An Open Platform for Discovering Research Artifacts and Analyzing their Evaluation in Security and Systems. (2026).
[53] Erik van der Kouwe, Gernot Heiser, Dennis Andriesse, Herbert Bos, and Cristiano Giuffrida. 2019. SoK: Benchmarking Flaws in Systems Security. In *EuroS&P*. IEEE, 310–325.
[54] Stephan van Schaik, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. 2018. Malicious Management Unit: Why Stopping Cache Attacks in Software Is Harder Than You Think. In *USENIX Security Symposium*. USENIX Association, 937–954.
[55] Daan Vansteenhuyse, Arthur Bols, Lieven Desmet, Victor Le Pochat, Jo Van Bulck, and Marton Bognar. 2026. Not All Those Who Share Are Lost: Analyzing 25 Years of Cybersecurity Artifact Sharing Practices Through Automated Discovery. In *Proceedings of the 35th USENIX Security Symposium*.
[56] Jose Rodrigo Sanchez Vicarte, Michael Flanders, Riccardo Paccagnella, Grant Garrett-Grossman, Adam Morrison, Christopher W. Fletcher, and David Kohlbrenner. 2022. Augury: Using Data Memory-Dependent Prefetchers to Leak Data at Rest. In *IEEE Symposium on Security and Privacy*. IEEE, 1491–1505.
[57] Yingchen Wang, Riccardo Paccagnella, Elizabeth Tang He, Hovav Shacham, Christopher W. Fletcher, and David Kohlbrenner. 2022. Hertzbleed: Turning Power Side-Channel Attacks Into Remote Timing Attacks on X86. In *USENIX Security Symposium*. USENIX Association, 679–697.
[58] Yingchen Wang, Riccardo Paccagnella, Alan Wandke, Zhao Gang, Grant Garrett-Grossman, Christopher W. Fletcher, David Kohlbrenner, and Hovav Shacham. 2023. DVFS Frequently Leaks Secrets: Hertzbleed Attacks Beyond SIKE, Cryptography, and CPU-Only Data. In *IEEE Symposium on Security and Privacy*. IEEE, 2306–2320.
[59] Wenjie Xiong, Stefan Katzenbeisser, and Jakub Szefer. 2021. Leaking information through cache LRU states in commercial processors and secure caches. In *Transactions on Computers*. IEEE, 511–523.
[60] Wenjie Xiong and Jakub Szefer. 2020. Leaking Information Through Cache LRU States. In *HPCA*. IEEE, 139–152.
[61] Ke Xu, Ming Tang, Quancheng Wang, and Han Wang. 2024. Exploitation of Security Vulnerability on Retirement. In *HPCA*. IEEE, 1–14.
[62] Mengjia Yan, Read Sprabery, Bhargava Gopireddy, Christopher W. Fletcher, Roy H. Campbell, and Josep Torrellas. 2019. Attack Directories, Not Caches: Side Channel Attacks in a Non-Inclusive World. In *IEEE Symposium on Security and Privacy*. IEEE, 888–904.
[63] Fan Yao, Milos Doroslovacki, and Guru Venkataramani. 2018. Are Coherence Protocol States Vulnerable to Information Leakage?. In *HPCA*. IEEE Computer Society, 168–179.
[64] Yuval Yarom. 2016. Mastik: A Micro-Architectural Side-Channel Toolkit. https://cs.adelaide.edu.au/~yval/Mastik/.
[65] Yuval Yarom and Katrina Falkner. 2014. FLUSH+RELOAD: A High Resolution, Low Noise, L3 Cache Side-Channel Attack. In *USENIX Security Symposium*. USENIX Association, 719–732.
[66] Jiyong Yu, Aishani Dutta, Trent Jaeger, David Kohlbrenner, and Christopher W. Fletcher. 2023. Synchronization Storage Channels (S2C): Timer-less Cache Side-Channel Attacks on the Apple M1 via Hardware Synchronization Instructions. In *USENIX Security Symposium*. USENIX Association, 1973–1990.
[67] Ruiyi Zhang, Taehyun Kim, Daniel Weber, and Michael Schwarz. 2023. (M)WAIT for It: Bridging the Gap between Microarchitectural and Architectural Side Channels. In *USENIX Security Symposium*. USENIX Association, 7267–7284.
[68] Zhenkai Zhang, Tyler N. Allen, Fan Yao, Xing Gao, and Rong Ge. 2023. TunneLs for Bootlegging: Fully Reverse-Engineering GPU TLBs for Challenging Isolation Guarantees of NVIDIA MIG. In *CCS*. ACM, 960–974.
[69] Zhiyuan Zhang, Mingtian Tao, Sioli O'Connell, Chitchanok Chuengsatiansup, Daniel Genkin, and Yuval Yarom. 2023. BunnyHop: Exploiting the Instruction Prefetcher. In *USENIX Security Symposium*. USENIX Association, 7321–7337.
[70] Zirui Neil Zhao, Adam Morrison, Christopher W. Fletcher, and Josep Torrellas. 2022. Binoculars: Contention-Based Side-Channel Attacks Exploiting the Page Walker. In *USENIX Security Symposium*. USENIX Association, 699–716.

A Open science

All the material used for this study is available for consultation and reuse. This includes the form used to fill in information for each article, and all the data we collected on the papers we selected for our study. The code we used to analyze the data and produce the figures is also available: https://archive.softwareheritage.org/swh:1:dir:ec9c4e46e14ba48a7bda9f9ae4e220809e538622;origin=https://github.com/Naeele/Practice_Makes_-Im-Perfect---Artifacts;visit=swh:1:snp:37bcd9c16f024719aab030fea686ddfbbbe301b1;anchor=swh:1:rev:08b8a2eebbda4c8d3faa1016abb37ceb367c4d2e

B Ethical considerations

Our work raises no ethical concerns: it involves no human subjects, user data, or real-world vulnerability testing, but only a meta-analysis of existing research. The paper evaluation was conducted over an extended period to avoid overburdening the authors.