Burning Fetch Execution: A Framework for Zero-Trust Multi-Party Confidential Computing

Shahin Roozkhosh*, Bassel El Mabsout*, Cristiano Rodrigues[‡§], Patrick Carpanedo*, Denis Hoornaert[†], Su Min Tan*, Benjamin Lubin*, Marco Caccamo[†], Sandro Pinto[‡§], Renato Mancuso*,
*Boston University, email: {shahin, bmabsout, pfcarp21, tansumin, blubin, rmancuso}@bu.edu
[†]Technical University of Munich, email: {denis.hoornaert,mcaccamo}@tum.de
[‡]Zero-Day Labs & [§]UMinho, email: cristiano.rodrigues@algoritmi.uminho.pt, sandro.pinto@dei.uminho.pt

***Abstract*—How can one tamper with data that does not exist? Motivated by this question, we present the Burning Fetch eXecution (BFX) paradigm. Data *in-use* is vulnerable, and the current focus on encrypting and/or isolating *in-use* data has fallen short. Frequently reported breaches of "secure" hardware and indispensable overhead with encryption schemes confirm that trust is the modern bottleneck. This work tackles the gap in existing safeguarding technology by avoiding byte-level decryption until it is immediately fetched by the processor, only to burn it right after. We perform on-the-fetch data decryption, immediately followed by *burning*, i.e., erasing right after processing cycles. Thus, BFX minimizes the existence of sensitive data *in-use*. BFX does not demand new processing hardware units nor requires restructuring application software. Three pillars set the BFX paradigm apart: (1) *zero-trust* multi-party confidentiality with (2) security rooted in transparency, and (3) high performance.**

By tackling the root of the issue, BFX enables a zero-trust multi-partied *sharing without showing* scenarios that were previously unthinkable. We showcase the impact of the BFX in a scenario with a highly privileged cloud insider attacker present. We exercise a sensitive mission whereby a third-party cloud processes fourth-party confidential real-time data streamed by a drone swarm. To further highlight the *zero-trust* nature of BFX, we assume the inference model (code) stream-processing on swarm data to be top-secret and owned by yet another party. The unknown threat, however, is the compromised processing system (cloud) where sensitive code and data are about to be deployed by all other parties—thanks to misplaced trust.

I. The Why: A *Burning* Need

The computing landscape has evolved from a single-party processing model to a multi-party environment. Market demands for higher performance and lower costs have led to adopting economies of scale in computing, resulting in the distributed processing paradigm. In this model, computationally intensive tasks are outsourced to third-party providers like Amazon AWS or Microsoft Azure. The rise of Artificial Intelligence (AI) has further accelerated this trend. However, security technologies have not kept pace with these changes.

A. The Unmet Need: Securely Confidential Processing

The 2019 Capital One breach[1] exposed the personal data of over 100 million customers, an act committed by a former cloud employee who leaked data from Capital One's server hosted on AWS. This incident highlights the ongoing challenges in securing data processed by third-party facilities [1]. Generally, data exists in three forms: *at-rest* (stored), *in-transit*, and *in-use* (actively processed). Cryptographic methods offer strong protection for data *at-rest* and *in-transit*; however, they fall short of protecting data *in-use*. This hinders multi-party collaboration where sensitive data is involved. Not data *in-use* can be secured, nor can parties be trusted.

While hardware-assisted security approaches have emerged as a practical solution for Confidential Computing, numerous vulnerabilities have been unveiled affecting Intel SGX [2]–[7], Arm TrustZone [8]–[10], and AMD SEV [4], [11], [12], exposing the weaknesses of current leading hardware (HW) technologies. Security is arguably an overloaded multifaceted term, and the inherent complexity of HW security is not well understood even by computing system professionals [13]. With only increasing threat actors' activity for financial/political gain, the message is loud and clear: existing methods remain insufficient [14]. Unsurprisingly, entities with sensitive data are hesitant about data sharing or cloud migration. Once deployed for cloud (or edge) processing, data is essentially in strangers' hands. Even with *ideal* external protection, the "trust" issue is only exacerbated as insiders account for 60% of all cyber-attacks and 43% of data breaches [15], [16] in 2023 alone.

Delays, threats, and breaches have significant costs beyond mere dollars and cents. They affect organizational integrity, collective efficiency, and, worst of all, fatal catastrophes. The missed collaboration and data-sharing opportunities remain a pressing challenge to overcome.

B. The Shortcomings of Current Solutions

Two dominant classes of 1) Cryptographic and 2) HW-assisted methods govern Confidential computing technology. The former includes Homomorphic Encryption and secure Multi-Party Computation (MPC), imposing a considerable computational overhead. On the other hand, HW solutions, such as enclaves, offer a more efficient and practical approach, e.g., Arm's TrustZone and Intel's SGX. Combined crypto-HW methods also come with challenges as secure enclaves are primarily available for CPUs, and support for such specialized HW remains limited[2]. This restriction poses challenges, particularly in feeding data to train neural networks (NNs)

[1]Capital One breach statement

[2]handful of academic research has explored their potential for usage with accelerators such as GPUs and other PCI devices

[17]. Recent studies [18] show that, for instance, on Intel's SGX, this can result in an approximately 400-fold slowdown when running image classifiers.

Implementing secure hardware enclaves demands substantial application (code/data) modifications, requiring organizations with sensitive workloads to redesign their systems or hire cryptography and kernel development experts. Additionally, exclusive enclave solutions from providers like AWS and Google Cloud lead to vendor lock-ins, posing restrictions. Existing solutions are either complexly inefficient or lack robust security for broad adoption.

C. The Path Forward

To enable secure processing and provide the ultimate security for all data forms with added emphasis on data *in-use* hosted in untrusted environments, we introduce Burning Fetch eXecution (BFX). BFX provides a framework where sensitive data only materializes just upon the processor's fetch and vanishes immediately after processing without a trace. BFX has the potential to redefine how industries (e.g., healthcare providers, financial institutions, and governmental entities) engage with data. It facilitates seamless *sharing without showing* among parties via automatic compliance with data management strict regulations such as the EU's GDPR, the CCPA [19], and alike [20]–[23]. BFX provides a flexible, software-agnostic framework that can be deployed from the edge—e.g., for securing data captured by drones—to the cloud, where it can safeguard security-critical computations involving sensitive data.

II. The What: A *Burning* Strategy

What sets Burning Fetch eXecution apart? Three pillars set BFX paradigm apart: (1) literal *zero-rustiness*, (2) security in transparency, and (3) high performance, all without demanding new processing hardware and thus software re-usability. To effectively grasp them, it is crucial to understand why the need for high performance *undeniably* contradicts the *zero-trust* philosophy by imposing a degree of trust on multi-party confidentiality. The answer lies around transparency, the *root of the issue*.

A. Understanding the Root of the Issue

As detailed in Sec. I, the primary challenge arises from the divide among parties. By formalizing the multi-party model next, we highlight the widened divide by modern AI/ML applications' reliance on HW accelerators.

B. A Truly Multi-Party Model

At a high level, we classify "parties" in the following multi-party/distributed secure processing model: 1) **Data Owners** are typically concerned with sharing but keen to collect data; 2) **Data Users** are solely interested in specific results attained from processing the data, (and not the data); 3) **Data Processors**, e.g., clouds, are needed but cannot be trusted by other parties. They are neither interested in the data nor the results. HW add-on accelerators also fall under this category. 4) **Data Subjects**, e.g., civilians, are reluctant with their data collected, and 5) **Data Regulators** are entities that govern, validate, and enforce situation-specific regulations.

The complexity of system-wide *trustworthiness* only increases with the **number of parties**. For example, the Data Processor party may adopt specific HW accelerators that appear insiders to the Processor party **but** manufactured by a (potentially breached) third-party HW provider.

The overarching problem we aim to address is the compromised security during its active processing (on sensitive data *in-use*) in the system, whether edge or cloud. Within the intricately interconnected nature of modern systems, laden with numerous complex on-chip components, plain-text data is considered vulnerable solely by existing. As a response, numerous cloud/edge providers and HW manufacturers have introduced hardened on-chip security. However, these solutions 1) heavily rely on placing trust in chip/service providers and 2) limit the integrability of new HW, such as GPUs, limiting any HW-assisted acceleration. Unfortunately, even with current solutions, a reported 43% of all breaches originate from "insiders", which highlights the **truth: trust is often misplaced** (see *a note on trust* in appendix A).

III. The How: Burning Fetch eXecution Paradigm

How is it possible to tackle the issue at its root? To minimize trust to the maximum extent, we need to think out of the (limiting) box of "secure walls" [24]–[26]. We propose a paradigm shift introducing a *burning* element and an *on-demand* data fetcher to our BFX paradigm.

Burning Fetch eXecution. We offer a novel BFX model that redefines data *in-use* security by removing the need for a trusting party, a model whose strength is rooted in its transparency. At the core of our technology lies the ability to decouple the physical address seen by the processor, e.g., CPU, from its corresponding memory location. This fundamental shift in approach allows for the disassociation between the contents in memory and what's fetched to the processing HW. Our ongoing research [27]–[30] has unlocked the potential to logically interpose a programmable module between the processors and main memory **without** any HW system on chip (SoC) modification. Leveraged by memory-mapped semantics, transactions are re-routed through the BFX module instead of taking the conventional data path. Once fetch-originated memory requests pass through the BFX logic, they are redirected to the internal memory controller. This entire mechanism effectively crafts a secondary route to memory. It is possible to use BFX routing for detailed inspection [31], profiling [32], take action based on the traffic's characteristics [28], [33]–[35], or to enforce Confidentiality, Integrity, and Availability even on the presence of advanced software **and** hardware attackers (Trojans) [30]. Our recent advancements opened a new class of actions including (but not limited to) *just-on-fetch*, *fetch-n-burn*, and *fetch-n-monitor*. As such, BFX is enabled with a guaranteed *on-demand burning* scheme over the fetched data.

just-on-fetch. With current processing architectures, data must be decrypted in memory **before** its fed to the CPU. This conventional flow, involving data transfer between the CPU and memory, exposes the decrypted data while it is *in-use*, significantly increasing the potential for security vulnerabilities [30]. In contrast, *just-on-fetch* keeps data encrypted in memory at all times, decrypting it on-the-fly **just** when the CPU **fetches** it. This decryption happens immediately before the data reaches the CPU, with a cache line granularity, minimizing the exposure of raw data and substantially reducing the attack surface.

fetch-n-burn. To ensure that fetched data remains confined to the processor, our model mandates immediate destruction within the processing unit, leaving no residual traces. Specifically, once processed, the data bytes are instantly (and irreversibly) "**burned**". This approach guarantees unencrypted data only lives **transiently**, preventing any possibility of it residing elsewhere in the system. By enforcing data nullification at the hardware data path level, our model enables *zero-trust* secure computation on high-end COTS processing elements, even in adversarial environments.

fetch-n-monitor. The BFX's temporal monitoring capabilities ensure that decrypted data fed by *just-on-fetch* technique is continuously **monitored** and protected through live authentication, with unencrypted bytes available only to a specific processor within a strictly controlled time frame. Unauthorized access and anomalies [30], [31], [36] can be detected in real-time, even when the entire software stack is compromised. This method enables *zero-trust* and fail-safe operations.

IV. Demonstration

The real-world scenario adopted in our demo is a complex inference mission involving sensitive data being processed on a cloud device, a confidential model, and civilians (see Fig. 2 in Appendix). Involved parties are unaware of an insider attacker in the target (cloud) processor. We demo how the BFX paradigm transparently 1) prevents sensitive data leakage while 2) being able to run high-performance resource-intensive tasks. In the demo, we run a real-time inference model on a live camera feed, all within a compromised cloud device!

Fig. 1 illustrates how the demo follows the formalized model in Sec. II-B. **Data Owners** represent A) confidential images about the **Data Subject(s)** to be detected (target) and B) a proprietary (and expensive) NN model owner. Both parties must maintain exclusive ownership while collaborating on sharing their data and code. **Data Processors** emulate a (cloud) system capable of processing the real-time stream from the drone swarm (with data *in-transit* protection). The swarm transmits a video stream of the **Data Subjects**, i.e., people present in the flight zone. The drones must comply with data protection protocols, stating "the captured video should not be stored, redirected, or used for any other purposes". This task is extremely challenging in untrusted multi-party scenarios. Finally, the **Data User** is the party interested in receiving live results (yes/no) on the target's presence in the surveillance footage. This party would have coordinated the mission by hiring all previous parties. Hence, they would have agreed to comply with all mission-specific and data-privacy-related protocols passed by the **Data Regulators**. Regulations encompass the combined interests and concerns of all the parties above. We show that even with compromised parties, previously unthinkable models can be governed and validated by enforcing on-demand regulations. For demonstration purposes, we also introduce an extra party, an **Attacker**, i.e., a privileged insider with access to the Data Processor (cloud), who attempts to maliciously extract the confidential code/data *in-use*.

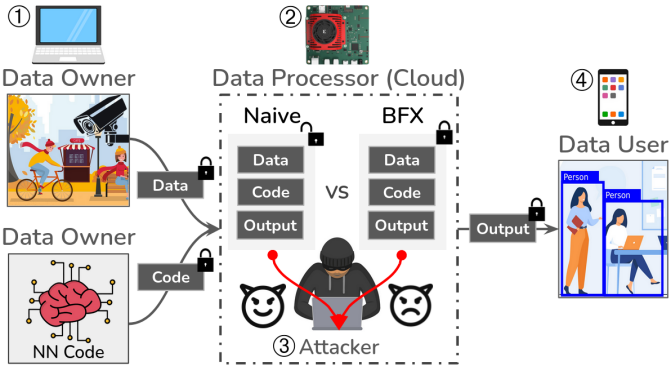

Fig. 1. The data flow between untrusted parties in the theory and demo.

Demo Setup. We consider a breached cloud provider (Data Processor) with a full-stack implementation of BFX. As depicted in Fig 1, the cloud device ② is a scaled-down server on a Kria board. The Kria is breached by an Attacker ③, a *sudoer* who knows the exact physical addresses of sensitive code/data. The drone (**Data Owner**) captures sensitive images of people in the room (**Data Subjects**) ①. This will be depicted as a laptop with a camera transmitting an encrypted stream (i.e., data *in-transit*) to the Kria ②. For simplicity, we assume that the same Data Owner ① also securely transmits a proprietary trained NN to the Kria. The NN is written with `Tensorflow-Lite` and trained for real-time inference on the camera stream; this serves as a computationally intensive task. Finally, a mobile device (**Data User**) ④, which is solely interested in knowing if the target(s) appear(s) in the room, receiving real-time results.

We highlight the need for a *zero-trust* paradigm by allowing the attacker ③ to attempt displaying the sensitive images (demoed using a separate laptop). With our BFX implementation, the hacker only sees randomized encrypted bytes, while in the naive case, they can display the full camera stream. BFX uses authentication and phone-home procedures to perform various monitoring tasks, ensuring mission-wide system integrity.

In conclusion BFX achieves transparent security not by trust but by design. Adhering to the *zero-trust* philosophy, we welcome adopters to understand our design and not to trust it. With extendable support for compliant HW accelerators and adaptability to high-end COTS SoCs, BFX goes well beyond the proposed demo.

References

[1] S. Khan, I. Kabanov, Y. Hua, and S. Madnick, "A systematic analysis of the capital one data breach: Critical lessons learned," *ACM Trans. Priv. Secur.*, vol. 26, no. 1, nov 2022. [Online]. Available: https://doi.org/10.1145/3546068

[2] S. van Schaik, A. Seto, T. Yurek, A. Batori, B. AlBassam, D. Genkin, A. Miller, E. Ronen, Y. Yarom, and C. Garman, "Sok: Sgx.fail: How stuff gets exposed," in *Proc. of S&P*, 2024.

[3] A. Nilsson, P. N. Bideh, and J. Brorsson, "A survey of published attacks on intel SGX," *CoRR*, vol. abs/2006.13598, 2020. [Online]. Available: https://arxiv.org/abs/2006.13598

[4] C. Canella, J. Van Bulck, M. Schwarz, M. Lipp, B. Von Berg, P. Ortner, F. Piessens, D. Evtyushkin, and D. Gruss, "A systematic evaluation of transient execution attacks and defenses," in *Proc. of USENIX Security*, 2019.

[5] D. Moghimi, J. Van Bulck, N. Heninger, F. Piessens, and B. Sunar, "Copycat: Controlled instruction-level attacks on enclaves," in *Proc. of USENIX Security*, 2020.

[6] S. van Schaik, M. Minkin, A. Kwong, D. Genkin, and Y. Yarom, "Cacheout: Leaking data on intel cpus via cache evictions," in *Proc. of S&P*, 2021.

[7] J. Van Bulck, D. Moghimi, M. Schwarz, M. Lippi, M. Minkin, D. Genkin, Y. Yarom, B. Sunar, D. Gruss, and F. Piessens, "Lvi: Hijacking transient execution through microarchitectural load value injection," in *Proc. of S&P*, 2020.

[8] D. Cerdeira, N. Santos, P. Fonseca, and S. Pinto, "Sok: Understanding the prevailing security vulnerabilities in trustzone-assisted tee systems," in *2020 IEEE Symposium on Security and Privacy (SP)*, 2020.

[9] C. Rodrigues, D. Oliveira, and S. Pinto, "Busted!!! microarchitectural side-channel attacks on the mcu bus interconnect," in *Proc. of S&P*, 2024.

[10] K. Ryan, "Hardware-backed heist: Extracting ecdsa keys from qualcomm's trustzone," in *Proc. of ACM CCS*, 2019.

[11] M. Li, Y. Zhang, H. Wang, K. Li, and Y. Cheng, "CIPHERLEAKS: Breaking constant-time cryptography on AMD SEV via the ciphertext side channel," in *Proc. of USENIX Security*, 2021.

[12] R. Zhang, L. Gerlach, D. Weber, L. Hetterich, Y. Lü, A. Kogler, and M. Schwarz, "CacheWarp: Software-based fault injection using selective state reset," in *Proc. of USENIX Security*, 2024.

[13] N. Potlapally, "Hardware security in practice: Challenges and opportunities," in *2011 IEEE International Symposium on Hardware-Oriented Security and Trust*, 2011. [Online]. Available: https://doi.org/10.1109/HST.2011.5955003

[14] Gartner, "Cloud adoption failures and successes 2023," link, 2023.

[15] Verizon, "Verizon data breach investigations report," https://enterprise.verizon.com/resources/reports/dbir/.

[16] Securonix, "2024 insider threat report," link, 2024.

[17] Opaque, "Confidential data for secure ai," opaque, 2023.

[18] M. El-Hindi, T. Ziegler, M. Heinrich, A. Lutsch, Z. Zhao, and C. Binnig, "Benchmarking the second generation of intel sgx hardware," in *DaMoN '22*. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3533737.3535098

[19] S. of California, "California consumer privacy act," CCPA, 2018.

[20] B. Government, "Brazilian general data protection law," BGDPL.

[21] T. Government, "Thailand personal data protection act," TPDPA, 2019.

[22] G. of India, "India personal data protection bill," IPDPB, 2019.

[23] C. Government, "Personal information protection law of the people's republic of china," PIPLPRC.

[24] S. Pinto and N. Santos, "Demystifying Arm TrustZone: A Comprehensive Survey," in *Journ. ACM Computing Surveys*, 2019.

[25] V. Costan and S. Devadas, "Intel sgx explained," 2016.

[26] AMD, "Protecting VM Register State With SEV-ES," AMD, Tech. Rep., Feb 2017.

[27] S. Roozkhosh and R. Mancuso, "The potential of programmable logic in the middle: Cache bleaching," in *26th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2020)*, Sydney, Australia, April 2020, conference.

[28] D. Hoornaert, S. Roozkhosh, and R. Mancuso, "A Memory Scheduling Infrastructure for Multi-Core Systems with Re-Programmable Logic," in *33rd Euromicro Conference on Real-Time Systems (ECRTS 2021)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), B. B. Brandenburg, Ed., vol. 196. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, pp. 2:1–2:22. [Online]. Available: https://drops.dagstuhl.de/opus/volltexte/2021/13933

[29] S. Roozkhosh, D. Hoornaert, J. H. Mun, T. I. Papon, U. Drepper, R. Mancuso, and M. Athanassoulis, "Relational Memory: Native In-Memory Accesses on Rows and Columns," *CoRR*, vol. abs/2109.1, 2022. [Online]. Available: https://arxiv.org/abs/2109.14349

[30] C. Rodrigues, S. Roozkhosh, S. Pinto, and R. Mancuso, "Coherent trojans: Achilles' heel of heterogeneous computing," in *Submitted IEEE S&P*, 2025.

[31] D. Hoornaert, S. Roozkhosh, R. Mancuso, and M. Caccamo, "Work in progress: Identifying unexpected inter-core interference induced by shared cache," in *2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, 2021, pp. 517–520.

[32] S. Roozkhosh, D. Hoornaert, and R. Mancuso, "CAESAR: Coherence-Aided Elective and Seamless Alternative Routing via on-chip FPGA," in *43rd IEEE Real-Time Systems Symposium (RTSS)*, 2022.

[33] S. Roozkhosh, D. Hoornaert, J. H. Mun, T. I. Papon, U. Drepper, R. Mancuso, and M. Athanassoulis, "Relational memory: Native in-memory accesses on rows and columns," *CoRR*, vol. abs/2109.14349, 2021. [Online]. Available: https://arxiv.org/abs/2109.14349

[34] T. I. Papon, J. H. Mun, K. Karatsenidis, S. Roozkhosh, D. Hoornaert, A. Sanaullah, U. Drepper, R. Mancuso, and M. Athanassoulis, "Effortless locality on data systems using relational fabric," *Special issue for ICDE 2023 (Best and Innovation Papers)*, 2023.

[35] T. I. Papon, J. H. Mun, S. Roozkhosh, D. Hoornaert, A. Sanaullah, U. Drepper, R. Mancuso, and M. Athanassoulis, "Relational Fabric: Transparent Data Transformation," in *ICDE*, 2023.

[36] S. Roozkhosh, D. Hoornaert, R. Mancuso, and M. Athanassoulis, "Hardware data re-organization engine for real-time systems," *WiP Session @ 43rd IEEE Real-Time Systems Symposium (RTSS@Work 2022)*. [Online]. Available: https://par.nsf.gov/biblio/10482041

Appendix

Fig. 2. A drone swarm, (1), sends data to a **Data Processor** server (2), which, in tandem, utilizes databases managed by **Data Owners** (3) to output the result to the **Data User** (4)

A. A note on trust

Schneuier's principle states that anyone can design a system they cannot break. This emphasizes the importance of independent review and transparency in system security. We aim to minimize the amount of **trust** by welcoming scrutiny. However, some form of "trust" is always involved, whether in the technology or teams. Even trusted teams may face an insider/outsider attack while not fully at fault. Lacking any trusted breach (or leak) detection technology, leakages are almost always discovered (up to months) later when the damage is permanent. The multi-faceted challenge around trust hinders actions we are already capable of performing, many of which rely on trusted data-sharing windows. We believe that trust is becoming the modern bottleneck.