Research Statement

Peterson Yuhala
University of Neuchâtel, Switzerland

January 15, 2026

My primary research interests are in systems security and data privacy, particularly on securing data in use. The growing popularity of cloud-based computing and data-driven technologies like deep learning, genome analysis, graph analytics, etc. requires collecting and processing potentially sensitive data. This creates an urgent need for robust data security and privacy mechanisms.

State-of-the-art solutions such as hardware-based *trusted execution environments* (TEEs) and software-based cryptographic primitives like *fully homomorphic encryption* (FHE) provide ways to mitigate these security and privacy issues. Nevertheless, these techniques often involve trade-offs between security, usability, and performance. The central theme driving my research is finding practical middle grounds: designing privacy-preserving systems that provide strong security guarantees while remaining efficient and user-friendly.

Dissertation Research

The focus of my PhD thesis [22] was on hardware-based security with trusted execution environments (TEEs), with an emphasis on designing and building tools to 1) enhance security via *trusted computing base* (TCB) reduction, and 2) improve performance in TEE-based applications.

TCB reduction. The complexity of state-of-the-art TEE implementations, *e.g.,* Intel SGX [3] for process-level isolation, has led to the adoption of tools like library operating systems [14, 13], which allow unmodified legacy applications to be run inside TEEs. While this approach enhances the usability of TEEs (especially by non-experts), it increases the attack surface and bloats the trusted computing base. Moreover, prior research [8] has shown that larger code bases are more prone to bugs and security vulnerabilities.

As a result, reducing the TCB in TEE-based applications is crucial for achieving strong security: this can be done through *privilege separation* [1], a security principle which divides a program into parts with different levels of privilege or access rights. In this context, my first PhD research work [24] proposed an approach to partition Java programs for Intel SGX enclaves. The proposed approach relies on code annotations and bytecode transformations to partition Java classes into trusted and untrusted components. These components are then ahead-of-

time compiled into binaries that run in and out of an enclave, while maintaining sufficient interaction to preserve the functional goals of the original program.
The second research [23] work extended the concept of code partitioning to a multi-language context. It leverages GraalVM's Truffle framework [25] to provide abstract syntax tree (AST) nodes that encapsulate sensitive data in polyglot programs. The resulting AST is then analyzed via dynamic taint analysis [5], and the secure nodes are used to deduce sensitive portions of the program to be isolated inside an enclave.
These tools also tackle TEE usability challenges for high-level languages (*e.g.,* Java, JavaScript, Python) for which popular TEE technologies like Intel SGX provide little to no support. These research works were done in collaboration with Oracle Labs Zürich.

TEE performance improvement. While TEEs offer strong isolation guarantees, they often impose significant overhead, *e.g.,* through costly context switches. The subsequent research conducted focused on tackling these performance overheads in TEEs. First, I explored how emerging hardware technologies such as persistent memory (PM) can be used to mitigate I/O overhead for applications executing in secure enclaves [20]. Persistent memory has both storage and memory characteristics, and provides memory access semantics that can be leveraged by TEEs to eschew costly CPU context switches required to manipulate data in the underlying storage device. I leveraged these properties of PM to provide efficient fault tolerance guarantees for in-enclave data structures. I further tackled the TEE performance problem by leveraging multi-threading in enclaves [21]: using worker threads in and out of enclaves that use shared memory to prevent costly context switches, *i.e.,* switchless calls. This approach involves a (more) dynamic technique for tuning worker threads in Intel SGX's switchless routine library so as to obviate the performance penalty associated with poor static configurations.

Ongoing Research: secure processing-in-memory

Several recent studies [9, 10] have shown that traditional processor-centric computing systems are inadequate for today's data-intensive applications. Workloads such as machine learning, genome analytics, graph processing, etc. are often memory-bound: the limited CPU-memory bandwidth introduces a significant performance bottleneck (commonly known as the *von Neumann* bottleneck) in these applications. This has led to the emergence of *processing-in-memory* (PIM), a memory-centric computing paradigm which augments memory with compute capabilities. For example, a recently commercialized PIM hardware architecture, UPMEM PIM [15], associates a low-power processor called a DRAM processing unit (DPU) with every 64 MB of DRAM. This design allows extensive parallelism and enables computing resources to scale with memory size.
Despite these advantages, current PIM hardware lacks mechanisms to protect data processed in memory from unauthorized access and modification. As current TEE solutions are processor-centric, they cannot be used as a drop-in replacement for PIM hardware.
My current research aims to address this security problem in memory-centric computing designs. In that light, we have been investigating the adoption of purely cryptographic solutions

like fully homomorphic encryption (FHE), which allow arbitrary computation over encrypted data. The advantage of such cryptographic approaches is that they are entirely software-based and thus can be readily deployed on current PIM hardware. This creates a symbiotic relationship between PIM and FHE, where FHE allows privacy-preserving computation in PIM-based applications, while PIM accelerates the computationally heavy FHE primitives. Our recent work [11] explored the feasibility of current PIM hardware to accelerate several underlying algorithms in FHE in popular FHE libraries.

Still in this direction, we have equally been investigating PIM-based acceleration in privacy-preserving contexts that do not rely on complex cryptography like FHE. A notable example is information-theoretic private information retrieval (PIR), which allows a client to query a database on untrusted servers obliviously, *i.e.,* without exposing the exact query. PIR is inherently memory-bound since the entire database always needs to be accessed to prevent query-specific data access patterns. We have designed a PIM-based PIR solution which allows in-place processing of large PIR databases, enabling extensive acceleration of the primary bottleneck in PIR, *i.e.,* bit-wise XOR operations on database items.

Future Work

My future research prospects build upon my prior work in systems security and privacy, and span several key areas: first, hardware-enforced security for PIM architectures, and extending previous work on privilege separation to confidential virtual machines (CVMs), which are becoming the de facto confidential computing solution.

Hardware-enforced security in PIM. Notwithstanding the advantages of purely software cryptographic solutions like FHE, they are prohibitively expensive, especially when compared to hardware-based security primitives like TEEs. In that light, I plan to investigate the feasibility of extending existing (processor-centric) hardware security mechanisms like TEEs to improve the security posture of PIM-based applications. Leveraging these existing solutions promotes technology reuse and support for workloads running on existing hardware.

Nevertheless, memory-centric computing introduces threat models that do not necessarily align with current processor-centric TEEs. As such, the ultimate aim in this direction is to design novel TEE architectures tailored to PIM computing models, but inspired upon the large body of prior work done on processor-centric hardware security designs [16, 17]. Further, the design of some PIM architectures helps mitigate security issues common in processor-centric architectures. For example, in UPMEM's PIM architecture, DPUs are not shared among processes. This enhances spatial isolation (*e.g.,* in multi-tenant scenarios) and limits the efficacy of side-channel vulnerabilities [7, 18] and micro-architectural attacks [4, 2] common in current processor-centric TEE architectures. These security-enabling features of PIM can serve as strong foundations for building specialized, secure-by-design TEE architectures.

TCB reduction in confidential VMs. As previously discussed, the dilemma surrounding security, usability, and performance has remained a major research challenge in the context of

data security and privacy. This problem is equally present in confidential VMs, which aim to ease deployment of TEE-based applications, but in doing so, result in a significantly larger TCB as the entire guest operating system is included in the TEE. To address this issue, I envision adopting a low-TCB architecture for confidential VMs by leveraging robust security-first OS designs like microkernels. Unlike monolithic kernel designs that bundle all kernel functionality, microkernels split the kernel into loosely coupled modules which perform specific tasks. This approach promotes better isolation, security, modularity and minimalism. The latter facilitates techniques like formal verification which provide mathematical proof of the trustworthiness of software, e.g., in seL4, making such microkernels central in ensuring OS security. Compartmentalization techniques [12] can then be used to further ensure isolation of sensitive information from non-core modules and co-located applications. Recent works like [6] have explored a similar direction by adopting unikernel-based CVMs.
Still in the direction of TCB reduction in CVMs, service-oriented partitioning can be introduced to convert critical functions into microservices that run in a CVM, while the non-critical functions remain in an unprotected VM. This idea can be used alongside solutions like [19] to achieve fine-grained security/isolation for data processing pipelines in confidential VMs.

Open source repositories

In line with open science principles, I have open-sourced the code for most of my research projects. This promotes reproducibility and provides a foundation for future research. The following GitHub repositories correspond to some of my key publications:

1. Partitioning Java applications for TEEs
2. Multi-language program partitioning for TEEs using GraalVM Truffle
3. Improving I/O performance in TEEs with persistent memory (applied to ML)
4. Reducing enclave context switches with multi-threading (dynamic switchless calls)

References

[1] David Brumley and Dawn Song. Privtrans: automatically partitioning programs for privilege separation. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, page 5, USA, 2004. USENIX Association.

[2] Guoxing Chen, Sanchuan Chen, Yuan Xiao, Yinqian Zhang, Zhiqiang Lin, and Ten H. Lai. Sgxpectre: Stealing intel secrets from sgx enclaves via speculative execution. In *2019 IEEE European Symposium on Security and Privacy (EuroSP)*, pages 142–157, 2019.

[3] Victor Costan and Srinivas Devadas. Intel SGX explained. *IACR Cryptol. ePrint Arch.*, 2016:86, 2016.

[4] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, et al. Spectre attacks: Exploiting speculative execution. *Communications of the ACM*, 63(7):93–101, 2020.

[5] Jacob Kreindl, Daniele Bonetta, and Hanspeter Mössenböck. Towards efficient, multi-language dynamic taint analysis. MPLR 2019, pages 85–94, New York, NY, USA, 2019. Association for Computing Machinery.

[6] Dmitrii Kuvaiskii, Dimitrios Stavrakakis, Kailun Qin, Cedric Xing, Pramod Bhatotia, and Mona Vij. Gramine-tdx: A lightweight os kernel for confidential vms. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, page 45984612, New York, NY, USA, 2024. Association for Computing Machinery.

[7] Fangfei Liu, Yuval Yarom, Qian Ge, Gernot Heiser, and Ruby B. Lee. Last-level cache side-channel attacks are practical. In *2015 IEEE Symposium on Security and Privacy*, pages 605–622, 2015.

[8] Subhas C Misra and Virendra C Bhavsar. Relationships between selected software measures and latent bug-density: Guidelines for improving quality. In *Computational Science and Its ApplicationsICCSA 2003: International Conference Montreal, Canada, May 18–21, 2003 Proceedings, Part I 3*, pages 724–732. Springer, 2003.

[9] Onur Mutlu, Saugata Ghose, Juan Gómez-Luna, and Rachata Ausavarungnirun. Processing data where it makes sense: Enabling in-memory computation. *Microprocessors and Microsystems*, 67:28–41, 2019.

[10] Onur Mutlu, Saugata Ghose, Juan Gómez-Luna, and Rachata Ausavarungnirun. A modern primer on processing in memory. In *Emerging computing: from devices to systems: looking beyond Moore and Von Neumann*, pages 171–243. Springer, 2022.

[11] Mpoki Mwaisela, Joel Hari, Peterson Yuhala, Jämes Ménétrey, Pascal Felber, and Valerio Schiavoni. Evaluating the potential of in-memory processing to accelerate homomorphic encryption: Practical experience report. In *2024 43rd International Symposium on Reliable Distributed Systems (SRDS)*, pages 92–103, 2024.

[12] Vasily A. Sartakov, Lluís Vilanova, and Peter Pietzuch. Cubicleos: a library os with software componentisation for practical isolation. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '21, page 546558, New York, NY, USA, 2021. Association for Computing Machinery.

[13] Youren Shen, Hongliang Tian, Yu Chen, Kang Chen, Runji Wang, Yi Xu, Yubin Xia, and Shoumeng Yan. Occlum: Secure and efficient multitasking inside a single enclave of intel SGX. In James R. Larus, Luis Ceze, and Karin Strauss, editors, *ASPLOS '20: Architectural Support for Programming Languages and Operating Systems, Lausanne, Switzerland, March 16-20, 2020*, pages 955–970. ACM, 2020.

[14] Chia-Che Tsai, Donald E. Porter, and Mona Vij. Graphene-SGX: A practical library OS for unmodified applications on SGX. In Dilma Da Silva and Bryan Ford, editors, *2017 USENIX Annual Technical Conference (USENIX ATC 2017)*, pages 645–658, Santa Clara, CA, USA, 2017.

[15] UPMEM. UPMEM processing in-memory (PIM): ultra-efficient acceleration for data-intensive applications. White paper, August 2022.

[16] Stavros Volos, Kapil Vaswani, and Rodrigo Bruno. Graviton: Trusted execution environments on GPUs. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 681–696, Carlsbad, CA, October 2018. USENIX Association.

[17] Xiaolong Wu, Dave Jing Tian, and Chung Hwan Kim. Building gpu tees using cpu secure enclaves with gevisor. In *Proceedings of the 2023 ACM Symposium on Cloud Computing*, SoCC '23, page 249264, New York, NY, USA, 2023. Association for Computing Machinery.

[18] Yuanzhong Xu, Weidong Cui, and Marcus Peinado. Controlled-channel attacks: Deterministic side channels for untrusted operating systems. In *2015 IEEE Symposium on Security and Privacy*, pages 640–656, 2015.

[19] Yuqin Yan, Pritish Mishra, Wei Huang, Aastha Mehta, Oana Balmau, and David Lie. Stream processing with adaptive edge-enhanced confidential computing. In *Proceedings of the 7th International Workshop on Edge Systems, Analytics and Networking*, EdgeSys '24, page 3742, New York, NY, USA, 2024. Association for Computing Machinery.

[20] P. Yuhala, P. Felber, V. Schiavoni, and A. Tchana. Plinius: Secure and persistent machine learning model training. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 52–62, Los Alamitos, CA, USA, jun 2021. IEEE Computer Society.

[21] P. Yuhala, M. Paper, T. Zerbib, P. Felber, V. Schiavoni, and A. Tchana. Sgx switchless calls made configless. In *2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 229–238, Los Alamitos, CA, USA, jun 2023. IEEE Computer Society.

[22] Peterson Yuhala. *Enhancing Security and Performance in Trusted Execution Environments*. PhD thesis, University of Neuchâtel, Switzerland, 2024.

[23] Peterson Yuhala, Pascal Felber, Hugo Guiroux, Jean-Pierre Lozi, Alain Tchana, Valerio Schiavoni, and Gaël Thomas. Secv: Secure code partitioning via multi-language secure values. In *Proceedings of the 24th International Middleware Conference on ZZZ*, Middleware '23, pages 207–219, New York, NY, USA, 2023. Association for Computing Machinery.

[24] Peterson Yuhala, Jämes Ménétrey, Pascal Felber, Valerio Schiavoni, Alain Tchana, Gaël Thomas, Hugo Guiroux, and Jean-Pierre Lozi. Montsalvat: Intel SGX Shielding for

GraalVM Native Images. In *Proceedings of the 22nd International Middleware Conference*, Middleware '21, pages 352–364, New York, NY, USA, 2021. Association for Computing Machinery.

[25] M. ipek, B. Mihaljevi, and A. Radovan. Exploring aspects of polyglot high-performance virtual machine graalvm. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1671–1676, 2019.