Benchmarking the Second Generation of Intel SGX for Machine Learning Workloads

Adrian Lutsch[1], Gagandeep Singh[1], Martin Mundt[1,2], Ragnar Mogk[1], Carsten Binnig[1]

Abstract: For domains with high data privacy and protection demands, such as health care and finance, outsourcing machine learning tasks often requires additional security measures. Trusted Execution Environments like Intel SGX are a powerful tool to achieve this additional security. Until recently, Intel SGX incurred high performance costs, mainly because it was severely limited in terms of available memory and CPUs. With the second generation of SGX, Intel alleviates these problems. Therefore, we revisit previous use cases for ML secured by SGX and show initial results of a performance study for ML workloads on SGXv2.

Keywords: Trusted Execution Environments; Intel SGX; Machine Learning; Benchmarking

1 Introduction

The Importance of Trusted Computing. Trusted Execution Environments (TEE) are a powerful tool for privacy-preserving, trusted, and secure data processing in cloud environments. TEEs have been used to build secure systems like databases [PVC18; VGG19], storage engines [Su21], and data processing systems [Sc15]. Furthermore, they have also been used for secure machine learning (ML) systems. This includes work for secure neural network training [Hu18; Le20; Oh16; Qu20] or secure inference [Hu18; Le19; Qu20], secure federated learning [KCZ21; Mo21a; QF21; Qu20], and many more.

Intel SGX for Trusted Computing. The most widely used TEE implementation of the aforementioned systems is Intel Software Guard Extensions (SGX) [An13; CD16; In22b; Mc13]. Intel SGX assures the integrity of processes and the confidentiality of their data by running them inside of protected memory regions called enclaves. Data inside an enclave can only be accessed by the process running inside the same enclave, not by other processes, the operating system, or a hypervisor. Additionally, SGX supports so-called attestation. With attestation, a process can prove that it is running the expected code inside an enclave [An13; CD16; Mc13]. Thereby, SGX protects against strong adversaries with full control over operating system and hardware.

SGXv2 relieves previous Limitations. Although Intel SGX is a useful security technology, its first version (which we call SGXv1 in this paper) has severe limitations for the shielded applications, especially: (1) The encrypted and integrity-protected memory (called Enclave Page Cache, EPC) is limited to 128 MB, of which only ~90 MB are usable for user enclaves.

[1] Technical University of Darmstadt, Contact: adrian.lutsch@cs.tu-darmstadt.de
[2] Hessian Center for AI (hessian.AI)

(2) Context switches between normal unprotected execution and secure protected execution of the enclave are costly. (3) Memory decryption and integrity protection cause overhead on cache misses, and (4) server-grade and multi-socket CPUs are not supported. With the most recent generation of SGX-enabled processors (which we call Second Generation of SGX or SGXv2 in this paper), Intel introduced several enhancements to the SGX technology which address some of the previous limitations [Jo21]. Primarily, new CPUs support up to 512 GB of EPC per CPU socket, which alleviates the need for expensive EPC paging. Additionally, SGX is now supported for multi-socket server systems. Enclaves on these systems can use the combined cores and EPC of all sockets, enabling higher degrees of parallelization.

Revisit Secure ML on SGXv2. These developments raise the question whether previous workarounds and optimizations for ML use cases in the restricted SGXv1 are still necessary. Therefore, we study the performance of ML use cases secured using SGXv2. Towards this goal, we measure the overhead of securely running ML tasks in SGXv2 enclaves and compare the results to SGXv1. The impact of SGXv2 was already analyzed for database workloads [El22]. However, we think that a closer look at machine learning workloads is justified because they have very different characteristics in terms of data access, compute intensity, and communication patterns. Furthermore, while the existing benchmarks are rather low-level, we investigate the performance of more complex algorithms and systems. In the rest of this paper we will present two of the most important use cases for Intel SGX in machine learning, Outsourced ML and Federated Learning, and report on our first evaluation results for outsourced neural network inference.

2 Using SGX for Secure ML

Since SGX is a versatile security technology with strong guarantees, it has been used to secure different applications in various settings. Next, we discuss two of the main use cases for SGX in secure ML and how SGXv1 limited them in terms of performance.

Outsourced ML is a setting in which an untrusted cloud provider offers infrastructure or services used for machine learning applications. In this setting, a malicious cloud provider is able to access the cloud customer's model and data while it is decrypted in memory. Furthermore, the cloud provider can also use its privileges to manipulate model and data causing false or low quality model predictions. TEEs have been shown to prevent such attacks [Gr19; Hu18; Le19; Le20; Oh16; Qu20; TB19]. For example, due to the confidentiality guarantees, cloud customers can be sure that the model can not be accessed by the cloud provider. Moreover, attestation and integrity guarantees ensure that the cloud provider does not tamper with the model without the customer being able to detect it.

The main limiting factor, however, of SGXv1 for this use case is the enclave memory size [Gr19; Le19; Qu20] which causes enclave paging. Enclave paging happens when the CPU-supported EPC is exceeded. Since today's deep neural network architectures commonly require gigabytes of memory and SGXv1 only supports 90 MB of active memory, previous approaches try to reduce memory consumption to prevent enclave paging [Le19; Qu20]. Furthermore, previous works suggest that parallelizing training and inference in

SGXv1 did not yield expected speedups [Qu20]. In Sect. 3 we report our results of using SGXv2 which shows that these limitations do not hold anymore.

Federated Learning is an approach for machine learning over data of multiple data owners. Instead of centralizing the data, the data owners (called clients) train their model together. In the centralized setup, each client trains the model on its own data and a central parameter server regularly collects model updates, averages them, and sends the updated model to all clients. This process is repeated until convergence. At the end of this process, all participants have a model trained on their joint data without exchanging the data itself [Li20]. Although Federated Learning can mitigate some privacy risks in machine learning by not centralizing the data, there exist attacks against it. For example, it has been shown that a curious parameter server can reconstruct training data from model updates of the clients [Ge20; Ph18]. Furthermore, a malicious parameter server can manipulate the model and the training process [Ge20]. Running the server inside an SGX enclave mitigates these kinds of attacks [KCZ21; Mo21a; Mo21b; QF21; Xu21].

The main limitation of SGXv1 for this use case is again the EPC size [KCZ21; Mo21a]. Additionally, the communication of clients and server via the network requires enclave transitions. Frequent enclave transitions are known as a bottleneck of Intel SGX. Each transition causes a constant overhead for flushing caches and TLBs as well as a linear overhead with the parameters and gradients copied to and from the encrypted memory region. Therefore, we will investigate how parameter servers for federated learning behave when secured by SGX enclaves and whether SGXv2 improves compared to SGXv1.

3 Benchmarking SGX Neural Network Inference

In this section, we report on our initial results depicted in Fig. 1 and 2. We analyze the overhead SGXv1 and SGXv2 cause when executing inference on neural networks of different sizes (Fig. 1) and investigate the potential speedup through parallelism enabled by more CPU cores in SGXv2 (Fig. 2). To show the influence of increasing network sizes, we compare a small multi-layer perceptron (MLP), a simplified version of AlexNet [KSH17], and the VGG19 convolutional neural network architecture [SZ15]. The MLP has three layers with sizes 784, 100, and 10. The AlexNet was simplified by replacing the three final layers with one layer of size 500. These networks thus cover a wide spectrum of model sizes: 2 MB (MLP), 80 MB (AlexNet) and 1.4 GB (VGG19). As such, VGG19 (1.4 GB) cannot be stored in the EPC of SGXv1 whereas the other models fit in the EPC of SGXv1. For our experiments, we use the Intel DNNL library available as part of the SGX SDK [In22a]. To show differences between SGXv1 and SGXv2, we executed the inference on an Intel Xeon E-2288G (SGXv1) and a server with two Intel Xeon Gold 6326 CPUs (SGXv2).

Fig. 1 shows the relative overhead ; i.e., the time required for inference inside an enclave divided by the time required without SGX on the same hardware. We can see that inference inside SGX has very low overheads if the enclave fits into the EPC and context switches are amortized. The overhead for the small MLP can be explained largely by the necessary

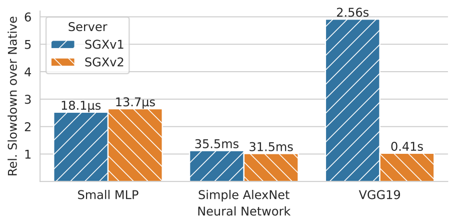

Fig. 1: Relative slowdown of SGX over native execution (bars) and absolute SGX times (labels).

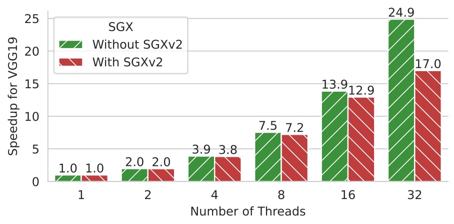

Fig. 2: Speedup by using multi-threading to parallelize ML inference with and without SGXv2.

context switches. Context switches are needed to copy the input data to the model in the enclave and inference results out of the enclave. Since the inference of the MLP takes only 5 - 7 microseconds, the relative cost of two context switches (that take ~4 microseconds) is high. For the two larger neural networks, the relative context switch cost is negligible. For the AlexNet we measured around 10% slower inference for SGXv1 and only 3% slower inference for SGXv2. The large VGG19, which does not fit into the EPC of SGXv1, has a nearly 5 times longer inference time in SGXv1 due to EPC paging. On SGXv2, paging is not necessary, which leads to negligible overheads.

Additionally, in a second experiment we analyzed if the increased number of CPU cores of the SGXv2 hardware can be used to speed up inference of large networks that fit into the enlarged EPC. For the smaller models that would also fit into the EPC of SGXv1, the parallelization speed up with higher core counts is outweighed by the overhead of thread synchronization. Hence, we do not show these results. Fig. 2 shows the speedup gained through parallelization for the large VGG19 network on the SGXv2 hardware. When using only CPU cores on one socket, speedups with and without SGX are very similar. For example, with 16 threads we observe a 12.9 times speedup in an SGXv2 enclave and a 13.9 times speedup without. However, when the work is distributed over both sockets of the server, SGX seems to reduce parallelization gains. Using the 32 cores of both CPUs, we observe a 24.9 times speedup without SGX and only a 17 times speedup with SGXv2. We hypothesize that this is due to the non-uniform memory access and the encryption of communication between both CPUs in SGX mode.

4 Conclusion

We benchmark SGXv2 for ML workloads. Our first experiments show, among other insights, that the increased EPC capacity enables inference of today's deep neural networks with negligible overhead. We will continue our work with more in-depth analysis of neural network inference, other secure ML use cases, such as training and federated learning, an investigation into library operating systems like Gramine [Th22; TPV17], and an investigation into application optimizations for the new hardware.

References

[An13] Anati, I.; Gueron, S.; Johnson, S. P.; Scarlata, V. R.: Innovative Technology for CPU Based Attestation and Sealing, 2013, URL: https://www.intel.com/content/dam/develop/external/us/en/documents/hasp-2013-innovative-technology-for-attestation-and-sealing.pdf, visited on: 11/23/2022.

[CD16] Costan, V.; Devadas, S.: Intel SGX Explained, 2016, URL: https://eprint.iacr.org/2016/086.pdf, visited on: 11/23/2022.

[El22] El-Hindi, M.; Ziegler, T.; Heinrich, M.; Lutsch, A.; Zhao, Z.; Binnig, C.: Benchmarking the Second Generation of Intel SGX Hardware. In: Data Management on New Hardware. DaMoN'22, Association for Computing Machinery, New York, NY, USA, pp. 1–8, June 12, 2022, ISBN: 978-1-4503-9378-2.

[Ge20] Geiping, J.; Bauermeister, H.; Dröge, H.; Moeller, M.: Inverting Gradients - How Easy Is It to Break Privacy in Federated Learning? In: Advances in Neural Information Processing Systems. Vol. 33, Curran Associates, Inc., pp. 16937–16947, 2020, URL: https://proceedings.neurips.cc/paper/2020/hash/c4ede56bbd98819ae6112b20ac6bf145-Abstract.html, visited on: 11/28/2022.

[Gr19] Grover, K.; Tople, S.; Shinde, S.; Bhagwan, R.; Ramjee, R.: Privado: Practical and Secure DNN Inference with Enclaves, Sept. 5, 2019, arXiv: 1810.00602 [cs], URL: http://arxiv.org/abs/1810.00602, visited on: 04/08/2022.

[Hu18] Hunt, T.; Song, C.; Shokri, R.; Shmatikov, V.; Witchel, E.: Chiron: Privacy-preserving Machine Learning as a Service, Mar. 15, 2018, arXiv: 1803.05961 [cs], URL: http://arxiv.org/abs/1803.05961, visited on: 04/07/2022.

[In22a] Intel Corporation: Intel(R) Software Guard Extensions for Linux* OS, Nov. 26, 2022, URL: https://github.com/intel/linux-sgx/tree/master/external/dnnl, visited on: 11/29/2022.

[In22b] Intel Corporation: Intel® Software Guard Extensions, Intel, 2022, URL: https://www.intel.com/content/www/us/en/developer/tools/software-guard-extensions/overview.html, visited on: 11/23/2022.

[Jo21] Johnson, S.; Makaram, R.; Santoni, A.; Scarlata, V.: Supporting Intel SGX on Multi-Socket Platforms, 2021, URL: https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/supporting-intel-sgx-on-mulit-socket-platforms.pdf, visited on: 11/23/2022.

[KCZ21] Kuznetsov, E.; Chen, Y.; Zhao, M.: SecureFL: Privacy Preserving Federated Learning with SGX and TrustZone. In: 2021 IEEE/ACM Symposium on Edge Computing (SEC). Pp. 55–67, Dec. 2021.

[KSH17] Krizhevsky, A.; Sutskever, I.; Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks. Communications of the ACM 60/6, pp. 84–90, May 24, 2017, ISSN: 0001-0782.

[Le19] Lee, T.; Lin, Z.; Pushp, S.; Li, C.; Liu, Y.; Lee, Y.; Xu, F.; Xu, C.; Zhang, L.; Song, J.: Occlumency: Privacy-preserving Remote Deep-learning Inference Using SGX. In: The 25th Annual International Conference on Mobile Computing and Networking. MobiCom '19, Association for Computing Machinery, New York, NY, USA, pp. 1–17, Oct. 11, 2019, ISBN: 978-1-4503-6169-9.

[Le20] Lee, D.; Kuvaiskii, D.; Vahldiek-Oberwagner, A.; Vij, M.: Privacy-Preserving Machine Learning in Untrusted Clouds Made Simple, Sept. 9, 2020, arXiv: 2009.04390 [cs].

[Li20] Li, T.; Sahu, A. K.; Talwalkar, A.; Smith, V.: Federated Learning: Challenges, Methods, and Future Directions. IEEE Signal Processing Magazine 37/3, pp. 50–60, May 2020, ISSN: 1558-0792.

[Mc13] McKeen, F.; Alexandrovich, I.; Berenzon, A.; Rozas, C. V.; Shafi, H.; Shanbhogue, V.; Savagaonkar, U. R.: Innovative Instructions and Software Model for Isolated Execution. In: Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy. HASP '13, Association for Computing Machinery, New York, NY, USA, p. 1, June 23, 2013, ISBN: 978-1-4503-2118-1.

[Mo21a] Mo, F.; Haddadi, H.; Katevas, K.; Marin, E.; Perino, D.; Kourtellis, N.: PPFL: Privacy-Preserving Federated Learning with Trusted Execution Environments. In: Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services. MobiSys '21, Association for Computing Machinery, New York, NY, USA, pp. 94–108, June 24, 2021, ISBN: 978-1-4503-8443-8.

[Mo21b] Mondal, A.; More, Y.; Rooparaghunath, R. H.; Gupta, D.: Poster: FLATEE: Federated Learning Across Trusted Execution Environments. In: 2021 IEEE European Symposium on Security and Privacy (EuroS&P). Pp. 707–709, Sept. 2021.

[Oh16] Ohrimenko, O.; Schuster, F.; Fournet, C.; Mehta, A.; Nowozin, S.; Vaswani, K.; Costa, M.: Oblivious Multi-Party Machine Learning on Trusted Processors. In. 25th USENIX Security Symposium (USENIX Security 16). Pp. 619–636, 2016, ISBN: 978-1-931971-32-4, URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/ohrimenko, visited on: 12/01/2022.

[Ph18] Phong, L. T.; Aono, Y.; Hayashi, T.; Wang, L.; Moriai, S.: Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. IEEE Transactions on Information Forensics and Security 13/5, pp. 1333–1345, May 2018, ISSN: 1556-6021.

[PVC18] Priebe, C.; Vaswani, K.; Costa, M.: EnclaveDB: A Secure Database Using SGX. In: 2018 IEEE Symposium on Security and Privacy (SP). 2018 IEEE Symposium on Security and Privacy (SP). Pp. 264–278, May 2018.

[QF21] Quoc, D. L.; Fetzer, C.: SecFL: Confidential Federated Learning Using TEEs, Oct. 7, 2021, arXiv: 2110.00981 [cs].

[Qu20] Quoc, D. L.; Gregor, F.; Arnautov, S.; Kunkel, R.; Bhatotia, P.; Fetzer, C.: secureTF: A Secure TensorFlow Framework. In: Proceedings of the 21st International Middleware Conference. Middleware '20, Association for Computing Machinery, New York, NY, USA, pp. 44–59, Dec. 7, 2020, ISBN: 978-1-4503-8153-6.

[Sc15] Schuster, F.; Costa, M.; Fournet, C.; Gkantsidis, C.; Peinado, M.; Mainar-Ruiz, G.; Russinovich, M.: VC3: Trustworthy Data Analytics in the Cloud Using SGX. In: 2015 IEEE Symposium on Security and Privacy. 2015 IEEE Symposium on Security and Privacy. Pp. 38–54, May 2015.

[Su21] Sun, Y.; Wang, S.; Li, H.; Li, F.: Building Enclave-Native Storage Engines for Practical Encrypted Databases. Proceedings of the VLDB Endowment 14/6, pp. 1019–1032, Apr. 12, 2021, ISSN: 2150-8097.

[SZ15] Simonyan, K.; Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition, Apr. 10, 2015, arXiv: 1409.1556 [cs].

[TB19] Tramèr, F.; Boneh, D.: Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware. In. 7th International Conference on Learning Representations, ICLR. New Orleans, LA, USA, May 6–9, 2019, arXiv: 1806.03287 [cs, stat].

[Th22] The Gramine Project: Gramine, 2022, URL: https://gramineproject.io/, visited on: 11/24/2022.

[TPV17] Tsai, C.-C.; Porter, D. E.; Vij, M.: Graphene-SGX: A Practical Library OS for Unmodified Applications on SGX. In. 2017 USENIX Annual Technical Conference (USENIX ATC 17). Pp. 645–658, 2017, ISBN: 978-1-931971-38-6, URL: https://www.usenix.org/conference/atc17/technical-sessions/presentation/tsai, visited on: 11/24/2022.

[VGG19] Vinayagamurthy, D.; Gribov, A.; Gorbunov, S.: StealthDB: A Scalable Encrypted Database with Full SQL Query Support. Proceedings on Privacy Enhancing Technologies 2019/3, pp. 370–388, July 1, 2019, ISSN: 2299-0984.

[Xu21] Xu, T.; Zhu, K.; Andrzejak, A.; Zhang, L.: Distributed Learning in Trusted Execution Environment: A Case Study of Federated Learning in SGX. In: 2021 7th IEEE International Conference on Network Intelligence and Digital Content (IC-NIDC). Pp. 450–454, Nov. 2021.