Machine Learning IP Protection

(Invited Paper)

Rosario Cammarota, Indranil Banerjee and Ofer Rosenberg
Qualcomm Technologies, Inc.
San Diego, USA
{rosarioc, ibanerje, oferr}@qti.qualcomm.com

ABSTRACT

Machine learning, specifically deep learning is becoming a key technology component in application domains such as identity management, finance, automotive, and healthcare, to name a few. Proprietary machine learning models – Machine Learning IP – are developed and deployed at the network edge, end devices and in the cloud, to maximize user experience.

With the proliferation of applications embedding Machine Learning IPs, machine learning models and hyper-parameters become attractive to attackers, and require protection. Major players in the semiconductor industry provide mechanisms on device to protect the IP at rest and during execution from being copied, altered, reverse engineered, and abused by attackers. In this work we explore system security architecture mechanisms and their applications to Machine Learning IP protection.

CCS CONCEPTS

• **Computer systems organizations → Embedded systems**; • **Security and Privacy → Security in Hardware**; **Security in Hardware → Hardware Attacks and Countermeasures**; • **Hardware Attacks and Countermeasures → Side Channel Analysis and Countermeasures**;

KEYWORDS

Machine Learning, Intellectual Properties (IP), Trusted Computing; Computer Security.

ACM Reference format:

Rosario Cammarota, Indranil Banerjee and Ofer Rosenberg. 2018. Machine Learning IP Protection. In *Proceedings of International Conference on Computer Aided Design (ICCAD'18). ACM, New York, NY, USA, 3 pages.* https://doi.org/10.1145/3240765.3270589

ICCAD '18, November 5–8, 2018, San Diego, CA, USA

ACM ISBN 978-1-4503-5950-4/18/11…$15.00
https://doi.org/10.1145/3240765.3270589

1 INTRODUCTION

Machine learning, specifically deep learning models have shown great success for decision making, vision and audio tasks, to a point that major technology companies embed proprietary machine learning models – Machine Learning IPs – in their Artificial Intelligence (AI) products and services. [1] Application domains that see the deployment of Machine Learning IP include, but are not limited to identity management, finance, automotive, home automation and healthcare.

For example, face recognition [2] offers great advantages over other biometric solutions for identity management. The technology is contactless, making it convenient for a wide range of applications. In healthcare, face recognition can facilitate easy check-in at medical facilities. In finance, face recognition promises to ease point of sale payments, where the demand is for more customer-centric solutions that are usable and secure. Machine Learning IP are being investigated for their use in dynamic orchestration of heterogeneous system resources for computing and communication. [3] Managing cybersecurity on end devices and core network benefit from using Machine Learning IP for security analytics. [4]

Machine Learning IP are valuable, and require protection on the end devices against attackers. To develop – design, train and test, a proprietary production-level machine learning model requires a large amount of training data, powerful computing resources and long CPU/GPU cycles, and expert data scientists. Once developed, the Machine Learning IP is packaged for its deployment and execution within commercial products, to monetize AI-based products and services. By compromising Machine Learning IPs, attackers can favor the rise of plagiarized AI services; reverse engineer the IP, that enable an attacker to manipulate system input/ output to control the intended behavior of software/ hardware on the end device to her advantage.

Major players in the semiconductor industry provide security mechanisms on the end devices for Machine Learning IP protection. Such mechanisms prevent the IP to be copied, altered, reverse engineered, or abused by attackers. This article provides an overview of Machine Learning IP protection mechanisms on

the end devices, and the corresponding implementation challenges.

2 LIFECYCLE

Machine Learning IPs are developed by third party providers, e.g., Original Equipment Manufacturers (OEM,) or Original Design Manufacturers (ODM,) and packaged for their development and use on end devices and service providers within commercial AI-based products and services. Once an AI-service is invoked for execution at the end device, the model is loaded within an AI run-time framework, then executed – it executes an inference on given inputs, and the inference result is presented to the next stage in the pipeline of the AI-service.

3 ATTACKER MODEL

We assume an attacker that controls high-level system software, e.g., High-Level Operating System (HLOS) and Applications; that there exists a subset of trusted low-level software (firmware,) which, with the hardware, forms a Trusted Computing Base (TCB.) The attacker can tamper with the layers of software under her control, and install malicious software. This includes the possibility of the attacker attempting to replace the Machine Learning IP with one under her control. The attacker controls the communication channels to the end device and can perform communication protocol level attacks.

4 SECURITY PROPERTIES

Protecting Machine Learning IPs (its application code and data) can be achieved by a system security architecture that supports sealed execution – isolation and encryption, and attestation – offline or online. Isolation protects the IP from other distrusted applications. For Machine Learning IP, isolation implies enveloping the AI run-time framework, the model and its hyper-parameters in a protected module. This includes enveloping weights and biases when deep learning models, e.g., convolutional neural networks, are part of the IP. Software outside the protected module cannot read or write in the protected module run-time state or modify the code. In addition, execution of code inside the protected module starts from a predefined location.

Attestation allows a third party to get proof that the loaded software is genuine. Attestation can be local or remote. Local attestation refers to one software module attesting its state to another software module before running on the same end device. Remote attestation is to a remote party residing outside the end device. To provide strong security guarantees, and architecture that supports attestation should guarantee integrity of the attested state.

Sealing wraps confidential code or data of the Machine Learning IP. Unwrapping of code or data occurs when specific device conditions are met, e.g., a specific device configuration, software state, or a combination thereof. Usually, wrapping is based on encryption, and rely on similar mechanisms as software attestation, i.e., the key for encrypting code or data is typically derived from the software module measurements during initialization.

5 ENABLING TECHNOLOGIES

IP protection on the end device can leverage implementations of (a) Trusted Platform Module (TPM), to measure software modules during their initialization; and (b) Trusted Execution Environment (TEE), to prevent later modifications by means of isolation (or sealing.)

A TPM is a co-processor that can serve as a cryptographic key storage and can perform software module attestation. To attest software modules locally, e.g., boot loader(s), TEE, such software needs to be measured successively by the TPM. This establishes a chain of trust, which is anchored in a hardware component that is inherently trusted, the Root of Trust (RoT.)

Global Platform TEE [5] specifies a secure area of the main processor meant to provide isolated execution, integrity of trusted applications, as well as confidentiality of trusted application resources.

A widely used implementation of TEE is included in the ARM TrustZone Architecture. [6] TrustZone is a hardware-based security architecture for a System-on-Chip (SoC) in which the TEE, also called secure world, provides protection for trusted hardware and software resources. The SoC bus enforces in hardware that the secure world resources are not accessed from normal word resources (HLOS and Applications.) For this, a control signal known as the Non-Secure (NS) bit is added to the bus specification to assert legitimate access to protected resources. A TrustZone enabled core uses time-slicing to execute code in either the secure or normal world. To perform context switches to the other world, the processor invokes a new execution mode called Monitor mode. The Operating Systems in each security domain performs its own memory management. Inter-Process Communication (IPC) is possible via shared memory for large messages.

During the boot process (secure boot,) a chain of trust is formed by verifying the integrity of the trusted second stage boot loader and TrustZone Operating Systems before execution. The TrustZone processor starts in the secure world when it is powered on. Specifically, the firmware of the first stage boot loader is implicitly trusted, and is typically located in immutable storage (read-only memory, ROM.) After successful execution of the secure boot process, the HLOS is started.

The TrustZone Operating System can authenticate and verify the integrity of trusted applications by leveraging the TPM for executing a digital signature scheme. Third party IP vendors, e.g., OEM/ODM use their private key to sign trusted applications and provision TPM with the verification key.

6 PROTECTION MECHANISMS

Global Platform provides foundation to define a security architecture for Machine Learning IP protection. However, the nature of machine learning models require care in the definition of the TCB, and the resources to be managed in the TEE. Machine

Learning IP vendors provide trusted applications including or interfacing with an AI run-time framework to the secure operating system. The AI run-time framework becomes part of the TCB, and can increase the code size of the TEE significantly. The IP, the model and its hyper-parameters, can be part of the trusted application or it is provided as an authenticated encrypted blob. In the former case, the Machine Learning IP is deployed and stored in the TEE. This imposes additional storage requirements to the TEE. In the latter case, the model can be stored outside the TEE. Before invoking the IP for its execution, the IP is loaded in a portion of the main memory assigned to the TEE, authenticated and decrypted leveraging the TPM. This protects the IP against software attackers when the IP is at rest, and after the trusted application and the IP is loaded.

Machine learning IP, specifically the IP including deep learning models have significant requirements in terms of storage and computation (number of layers, activation sizes etc.) to meet accuracy requirements. [7, 8] Neural Processing Units (NPUs) [7, 8, 9] are used to provide hardware acceleration for deep learning workloads to meet inference execution time constraints while delivering the needed inference accuracy. Hence, to executed and protect a Machine Learning IP, the NPU needs to be available or securely configured to be assigned to the TEE. Furthermore, shared memory between the TrustZone core and the accelerator needs to be managed securely.

When factoring in Machine Learning IP protection, TEE requirements increase significantly, posing additional implementation, validation and testing costs.

7 A NOTE ON PHYSICAL ATTACKS

Admittedly, the attacker can perform side channel attacks to leak information regarding the Machine Learning IP. An attacker can probe off chip memory, read hardware counters, monitor cache access patterns etc. [10, 11, 12, 13]

Side channel resistance with respect to software attacks is a security property in which software modules outside the protected modules, including privileged software under the attacker's control, cannot deduce information about the Machine Learning IP (model and hyper-parameters) from observing input, output, and proxies to data dependent information. A side channel resistant architecture does not leak information through untrusted channels, such as caches, network on chip shared buffers etc. This can be achieved, for example, via flush caches during context switches, constant time implementations, and other techniques to decorrelate the side channel information from data dependent computation.

While side channel resistance is customarily deployed on end devices for implementations of cryptography (e.g., for content protection in mobile phones,) and there is academic research showing practical side channel attacks to reverse engineer machine learning model from NPUs, [14, 15] the path to adoption of side channel mitigation on end device for Machine Learning IP is still long – attacks exist, but, beyond basic recommendations, sound mitigations have yet to be proposed.

8 CONCLUSIONS

Machine Learning IPs are developed and deployed at the network edge, end devices and in the cloud to maximize user experience. Machine Learning IPs require protection at the end device against attackers in control of the software outside the Trusted Computing Base.

This article survey technologies and challenges for Machine Learning IP protection. While Global Platform guidelines provide the foundation for Machine Learning IP protection, storage and computation requirements of machine learning, specifically deep learning workloads, poses additional requirements to the TCB and TEE implementation, validation and testing.

Finally, we point out that side channel resistance needs to be factored in the mechanisms for Machine Learning IP protection.

ACKNOWLEDGMENTS

The authors would like to thank their colleagues at Qualcomm Technologies Inc. for their valuable feedback on this work. The ideas presented in this work are of the authors only.

REFERENCES

[1] 2017. We are making device AI ubiquitous. https://www.qualcomm.com/news/onq/2017/08/16/we-are-making-device-ai-ubiquitous

[2] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. 2017. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In AAAI. 4278–4284

[3] C. Zhang, P. Patras, and H. Haddadi. 2018. Deep Learning in Mobile and Wireless Networking: A Survey. CoRR abs/1803.04311 (2018)

[4] 2015. Qualcomm Smart Protect. https://www.qualcomm.com/news/onq/2015/08/31/snapdragon-820-countdown-snapdragon-smart-protect-detects-more-mobile-malware

[5] 2018. Global Platform Technical Overview. https://globalplatform.org/specifications/technical-overview/

[6] B. Ngabonziza, D. Martin, A. Bailey, H. Cho, and S. Martin. 2016. TrustZone Explained: Architectural Features and Use Cases. CIC 2016

[7] V. Sze, Y. Chen, T. Yang, and J. S. Emer. 2017. Efficient Processing of Deep Neural Networks: A Tutorial and Survey. Proc. IEEE 105, 12 (2017), 2295–2329

[8] Norman P. Jouppi, Cliff Young, Nishant Patil, and David Patterson. 2018. A domain-specific architecture for deep neural networks. Commun. ACM 61, 9 (August 2018), 50-59

[9] 2013. Introducing Qualcomm Zeroth Processors: Brain-Inspired Computing. https://www.qualcomm.com/news/onq/2013/10/10/introducing-qualcomm-zeroth-processors-brain-inspired-computing

[10] J. Longo, E. De Mulder, D. Page, M. Tunstall. SoC It to EM: ElectroMagnetic Side-Channel Attacks on a Complex System-on-Chip. CHES 2015

[11] Paul Kocher: Complexity and the challenges of securing SoCs. DAC 2011

[12] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, Y. Yarom. Spectre Attacks: Exploiting Speculative Execution. CoRR abs/1801.01203 (2018)

[13] H. Fang, S. Dayapule, F. Yao, M. Doroslovacki, G. Venkataramani. Prefetch-guard: Leveraging hardware prefetchers to defend against cache timing channels, IEEE International Symposium on Hardware Oriented Security and Trust (HOST), May, 2018

[14] C. Luo, Y. Fei, P. Luo, S. Mukherjee, and D. Kaeli. Side-channel power analysis of a GPU AES implementation. In ICCD 2015

[15] W. Hua, Z. Zhang, and E. Suh. Reverse Engineering Convolutional Neural Networks Through Side-channel Information Leaks. DAC 2018