Hardware Trojans in eNVM Neuromorphic Devices

Lingxi Wu[*,1], Rahul Sreekumar[†,1], Rasool Sharifi[*], Kevin Skadron[*], Mircea R. Stan[†], and Ashish Venkat[*]
[1](equal contributors to this work)
[*]*Department of Computer Science, University of Virginia*
[†]*Department of Electrical and Computer Engineering, University of Virginia*
Email: {lw2ef, rs2xd, as3mx, skadron, mircea, venkat}@virginia.edu

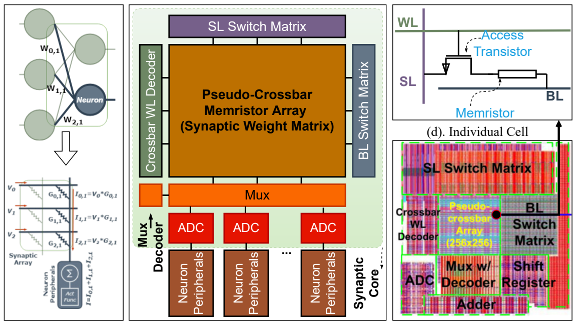
***Abstract*—Fast and energy-efficient execution of a DNN on traditional CPU- and GPU-based architectures is challenging due to excessive data movement and inefficient computation. Emerging non-volatile memory (eNVM)-based accelerators that mimic biological neuron computations in the analog domain have shown significant performance improvements. However, the potential security threats in the supply chain of such systems have been largely understudied. This work describes a hardware supply chain attack against analog eNVM neural accelerators by identifying potential Trojan insertion points and proposes a hardware Trojan design that stealthily leaks model parameters while evading detection. Our evaluation shows that such a hardware Trojan can recover over 90% of the synaptic weights.**

I. Introduction

Deep neural networks have been extensively employed in several applications. However, their execution on traditional architectures has shown to be inefficient due to their inherently memory-bounded nature, a problem that has been exacerbated by rapidly growing model and input data sizes. For example, data movement in GoogLeNet accounts for roughly 70% of the overall energy consumption [1]. In recent years, emerging non-volatile memory (eNVM)-based neuromorphic computing systems that emulate biological computing with artificial neurons in the analog domain, have gained traction, due to their high energy efficiency and throughput advantages [2]. Their security implications, however, remain largely unexplored.

The process of setting up a trusted end-to-end production line for eNVM-based neuromorphic accelerators can be prohibitively expensive, especially considering frequent algorithmic updates and tuning of models. This has paved the way for a decentralized design-manufacturing approach that involves a coordinated effort among multiple stakeholders. However, it also inevitably invites exploitation from bad actors to stealthily inject malicious hardware Trojans into the product [3]. Semiconductor supply chain attacks are critical threats that can disrupt the operations of high-value mission-critical systems such as military, financial, and medical infrastructure.

This work is the first to demonstrate the feasibility of carrying out a hardware supply chain attack against analog eNVM neural accelerators to leak IP-sensitive synaptic weights. We discuss potential Trojan insertion points within the supply chain and due to the lack of openly available commercial implementations, we dissect a generic eNVM accelerator derived from recent works to identify vulnerable probe points. The crux of this work is the design and stealthy placement of a neuron-suppressing hardware Trojan that can be reliably triggered by a colluding adversary. The findings from this research are expected to foster the design of such eNVM neural accelerators with a security focus.



(d). Individual Cell

(a). NN Layer to Synaptic Array (b). Synaptic Core Block Diagram (c). Sample Synaptic Core Layout

Fig. 1: **Synaptic Core layout [6] and Neuron Architecture.**

There are two major motivations for such a model extraction attack that aims at cloning a victim model of similar performance without going through the expensive training process. First, the synaptic weights of a neural network are considered core IP as they separate a properly trained network with high accuracy from a poorly trained one. Second, stealing weights is increasingly more economical than training. To obtain a model with competitive performance, typically, a large set of high-quality labeled data and proprietary training algorithms are required [4]. Further, even with access to proprietary training data, the process of training can take weeks and is likely to worsen with model sizes, affecting the time-to-market [5].

The key to our attack is the fact that synaptic weights are encoded as conductances of eNVM devices in the analog eNVM device array, and the total current representing a dot-product result depends on the synaptic weights. This allows for the isolation of the switching activity of a single neuron, enabling the sequential extraction of all the synaptic weights using power side-channel analysis while evading detection.

II. Background and Related Work

Analog eNVM Neuromorphic Devices. Fig. 1a illustrates the mapping of an MLP layer onto an eNVM device. All incoming synaptic weights of the highlighted output neuron $(W_{0,1}, W_{1,1}, W_{2,1})$ are stored as distinct conductances $(G_{0,1}, G_{1,1}, G_{2,1})$ of the eNVM cells along the same column. Suppose the inputs to this neuron are encoded as voltage levels applied to the wordlines (V_0, V_1, V_2), then each cell contributes a small current of $V_{\mathrm{i}} \times G_{\mathrm{i},1}$ Ampere to the bitline. By Kirchoff's Law, the total current passed to the neuron circuit at the end of the bit line is the sum of the three partial currents generated at each cell, representing the dot product of the input vector and the weight vector of a single neuron.

Synaptic cores (SC) are the fundamental building blocks of a neuromorphic architecture (Fig. 1b). They consist of a 2D synaptic device array that stores a weight matrix in the

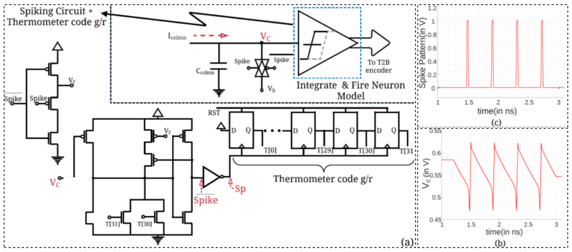


Fig. 2: **Schematics and waveforms that depict (a). Generic Integrate and Fire Neuron model; transient signals that exhibit (b). current integration, and (c). spiking pattern.**

form of conductance levels and supporting peripheral circuits. An additional access transistor is inserted per eNVM cell to mitigate the current sneak-path problem (Fig. 1d). During the weighted sum operation, wordlines (WLs) are switched on in parallel, thereby selecting multiple rows of eNVM cells. Inputs are provided as serial bit vectors to the memory cell and the generated currents on the selectline (SLs) represent weighted sums that propagate toward the neuron peripherals. The results of the matrix-vector multiplication are first converted by a series of ADCs (described next) to digital values, then sent to the neuron peripherals for activation, in turn producing the results bit vector to be sent to the next SC to select a set of BLs, subsequently activating neurons of the next layer. To save area, both ADC and neuron peripherals are shared among artificial neurons through a multiplexer [2], [7].

Neuron ADC. Fig. 2a shows a generic Integrate-and-Fire ADC, which consists of a thermometer code generator and a Thermometer-to-Binary encoder. Such a design is popular as it provides good energy efficiency [2], [6], [8]. Since the resulting cumulative current is to be *integrated* as a potential that represents the weighted sum, the integration is carried out as a potential buildup on a capacitor (C_{colmn}), and as the potential crosses a threshold value (V_{thr}), the neuron *fires* in the form of a spike, causing an instantaneous discharge of the potential buildup to a predetermined base potential (V_b). The circuit designed for generating spikes is highlighted within the dashed lines and involves an inverter with a reset element that allows for instantaneous discharge and regeneration of a spike potential. Fig. 2b depicts the transient waveform showing the potential buildup on the capacitor up to V_{thr} and discharge to V_b. Fig. 2c shows the resulting train of spikes.

Hardware Trojans. A hardware Trojan consists of a trigger circuit that activates it on a specific condition and a payload circuit that causes functional perturbations, carries out failures, or covertly leaks private information. They typically ought to be stealthy, meaning the underlying malicious circuitry occupies a small area, consumes negligible power, and remains dormant until triggered by rare events. Every stage in the distributed IC supply chain is susceptible to the insertion of hardware Trojans by any of the entities involved. The design team might unintentionally use tainted third-party IP blocks or tools, resulting in Trojan-infected netlist or layout files [9]. A rogue engineer in the design team can insert a Trojan directly at the RTL level [10]. Even if the chip specifications are correctly implemented by the design house and verified by the backend design house, the malicious foundry can tamper with the mask layout during fabrication [3], [9].

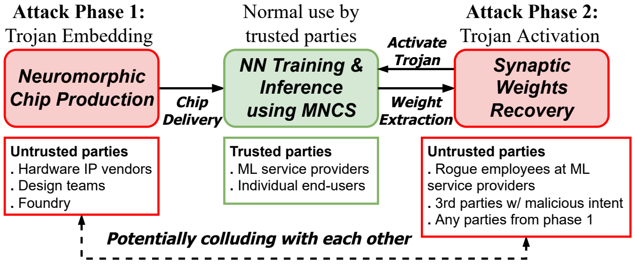


Fig. 3: **Trusted and untrusted parties in the supply chain.**

Related Work. An adversary with physical access to an eNVM device can probe it to extract the weights directly [11]–[13]. However, such an attack is destructive as probing one cell could damage adjacent ones [11]. Moreover, encrypting data before the system powers down is an effective countermeasure [11], [13]. On the other hand, stealing weights online is superior as it is non-destructive and cannot be mitigated by encryption because, at any time, there is at least one layer of synaptic weights remaining in plaintext [13]. Rajamanikkam et al. [12] outline two attacks to compromise the *availability* of neuromorphic devices. The first makes use of current sneak paths to mount a fault injection attack by sending malicious inputs and leveraging leakage currents to alter synaptic weights, resulting in incorrect inference outputs. This can be mitigated by inserting gating transistors. The other attack embeds hardware Trojans to degrade classification accuracy, as opposed to our attack which steals IP-sensitive model weights, which is of greater interest because properly trained weights are usually hard to obtain due to lack of high-quality and proprietary training data and algorithms [4].

III. Threat Model

Attacker Intent. The attacker intends to extract the synaptic weights of a neural network from an analog neuromorphic system in two phases. First, a Trojan is inserted at the hardware design or fabrication stage. Second, the synaptic weights are extracted at the NN inference stage by activating the Trojan such that the resulting power trace can be attributed to the requested synaptic weight. The attacker at each of these phases might not necessarily be a single entity, but could involve two separate colluding malicious parties. A detailed overview that depicts a product development life-cycle and the potential parties involved in the two phases of the threat model is shown in Fig. 3. The malicious entities in the supply chain do not possess the intricate details of the NN models (including weights) but have the ability to embed a Trojan, given the distributed nature of modern IC supply chains. A colluding entity could then trigger the Trojan post-deployment using a known activation code and then steal sensitive IP information. Alternatively, a rogue engineer in the supply chain can cause damage by simply publishing the Trojan activation code without explicitly colluding with another player. Either way, even if a trusted entity is tasked with securely programming the synaptic weights into the device, it would still remain vulnerable to a Trojan placed in the supply chain.

Trojan Insertion Points. We consider three possible insertion points. First, the Trojan could be injected at a very early stage (e.g., in the HDL code). However, this might stand out under scrutiny during post-design verification. Second, the Trojan could be placed in open spaces in the GDSII layout file after the circuits have been placed and routed following the model described in [3]. Third, an attacker from an untrusted fab house could inject the Trojan, which entails reverse engineering the victim wires to tap into, leveraging the knowledge of algorithms used in floor-planning, placement, and layout tools, which has been shown to be feasible [14].

Grey-box Model. As is common with conventional grey-box models [15], we assume that the attacker is aware of the neural network structure, such as the number of layers, but not the IP-sensitive model parameters, i.e., synaptic weights. Many production ML services leverage well-known neural networks whose structures are publicly available (e.g., ResNet, VGG-16/19, etc.) [4], due to which, all entities along the supply chain would have access to the model structures. However, the weights learned during the training process is often proprietary. We also assume that a malicious entity with access to the Trojan trigger code has the ability to buy such a device from the market and activate the Trojan by sending arbitrary inputs to covertly extract the synaptic weights through a power side-channel attack [3], [15]. We note that leaking synaptic weights in the absence of a Trojan is likely more challenging as it entails attributing signal leakage to particular weights.

IV. Attack Overview

A. Feasibility of Exploitation

The key insight to this attack is that the dot-product results are represented as analog currents, and the strengths of those currents are directly correlated to the synaptic weights, i.e., larger weights (conductances) produce larger currents. As this current is converted into a train of spikes by the Neuron ADC (described in Sec. II), it results in dynamic switching transients within the power trace, allowing the attacker to approximate weights by intercepting the power trace. Even if an alternative ADC is chosen, the generation of spikes would lead to a certain amount of switching activity, exposing it as a possible target for exploitation. Furthermore, ADCs consume over 80% of the total system power, allowing the attacker to estimate the power of ADCs using the global power trace [15]. Finally, since the ADCs are time-shared due to their large area, the attacker can target each ADC individually using our novel neuron suppression scheme that allows a malicious Trojan to isolate a particular neuron ADC, ultimately correlating the switching activity of the architecture to a single eNVM cell.

B. Attack Procedure

Online Weight Recovery. Fig. 4 illustrates the proposed Trojan-assisted power side-channel attack. The key idea is to attribute the observed power activity to a single ADC by sending specially-crafted input images containing Trojan codes (certain pixel patterns) to the device, which triggers the Trojan to iteratively select only one ADC to be functional and forces it to process a current generated by a weighted sum operation of one eNVM cell. An attacker can then collect power traces from a Trojan-infected chip using off-the-shelf instruments such as oscilloscopes [3], to deduce the weights by comparing it to a library of reference power traces obtained offline (described below). It is preferred that the attacker use a high sampling rate ($\geq$ 10GS/s) to capture sufficient sampling points within a read cycle. Once the conductance values associated with the currently functioning ADC are recovered, the attacker sends a reset signal using the same malicious image that activates the Trojan (toggle trigger), putting all ADCs back in working order. A different activation code is needed to target a different ADC. The attacker applies the above procedure repeatedly to cycle through all ADCs to recover all synaptic weights. The Trojan design and activation process are detailed in Sec. V.

Offline Characterization. The offline characterization step (highlighted in Fig. 4 dashed line) allows the attacker to build a library of reference traces used for comparison during online weight recovery. This is possible since: (1) the operational states of an ADC are finite, and (2) each conductance value generates a unique ADC output spiking pattern, allowing the attacker to thoroughly sweep through an ADC's current resolution steps and collect a library of distinct power traces. The development of such a characterization portfolio involves the generation of a step response chart (ADC step resolution graph in Fig. 4) that allows the attacker to map a unique spike pattern to a deterministic range of input current values during the Trojan embedding phase. Next, for each distinct ADC input current, its frequency domain signature (FFT) is extracted, which is used to approximate the synaptic weights.

C. Establishing Power-to-Weight Correlation

Signal Processing (FFT). The attacker can infer the ADC input current based on the decomposed frequency components of the power trace, as the frequency domain allows for a significantly higher fidelity comparison than the time domain. As a result, a Fast Fourier transform (FFT) is performed on both the victim and reference traces to compare and identify key frequency components, increasing the visibility of individual sub-components within the trace, thereby isolating unique signatures. The strongest signals within the spectrum can be attributed to the static (DC) energy costs, clock tree consumption, and spiking signature. The FFT analysis reveals that, (1) larger the input current, higher the frequency signature, and (2) each current step resolution emits a unique frequency signature. This allows the attacker to examine the frequencies extracted from the victim traces and match them against the signature frequencies within the library.

Synaptic Weights Recovery. There are two factors that determine the precision of recovered weights. First, the ADC can only respond to a set of discrete current ranges (i.e., ADC resolution steps) rather than continuous current values. This means two conductances (i.e., weights) with small differences could produce similar currents that lead to a similar switching activity (ADC power traces). The larger the step count, the more "sensitive" the ADC is to differentiate between input currents, and thus increase the resolution of the stolen weights.

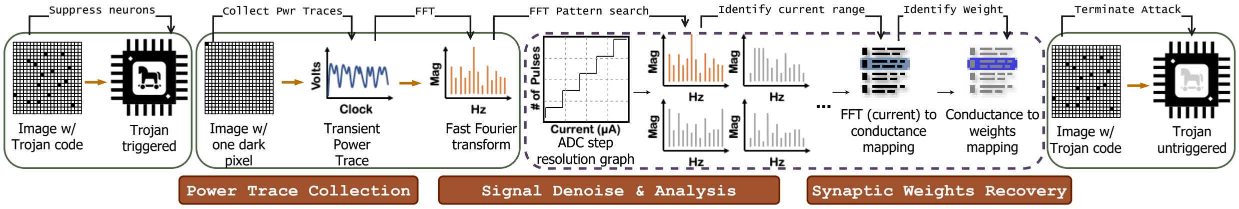


Fig. 4: **Synaptic weights recovery through a Trojan-created power side-channel.**

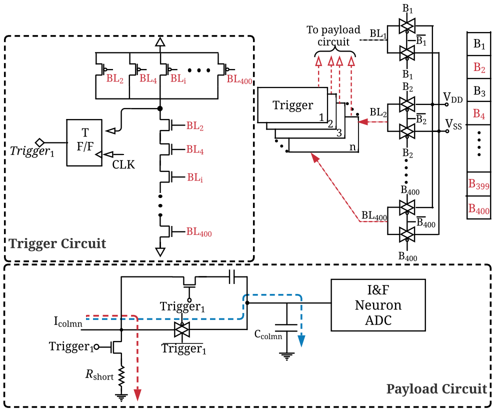


Fig. 5: **Trojan trigger module and payload circuit**

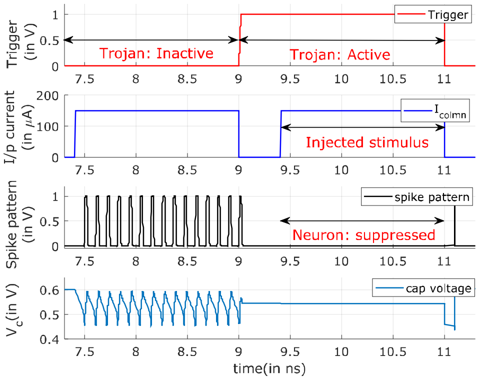


Fig. 6: **Transient waveforms of payload circuit.**

Seconds, ADCs are typically calibrated to work with the cumulative current produced by multiple eNVM cells. The current generated from one eNVM cell might be too small to excite the ADCs thermometer circuits, as a result of which the power trace might not yield meaningful leakage information for the attacker to extract weights. We assign minimal conductance values to those that are not recoverable through the power side channel. While this results in a small loss in the overall model inference accuracy, in some cases, it results in the cloned network outperforming the original (Sec. VII-B).

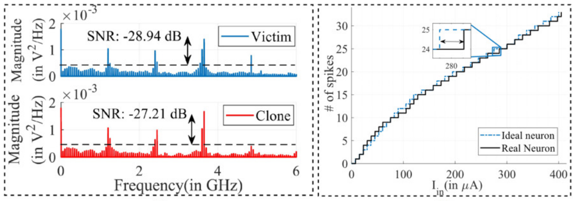
V. Trojan Design

The suppression of the neuron is achieved through the design of an analog Trojan that consists of a trigger and a payload module. The trigger circuit determines the operating condition of the payload, i.e., if the trigger state is high, the payload is active. If the payload is activated, the neuron circuit is suppressed through bypassing the current generated by the synaptic array away from its signal path, thereby depriving it of a valid input. Fig. 5, shows the circuit for a switched leakage short-circuit path (highlighted in red), that deviates the current flow from its normal path (highlighted in blue). As long as the trigger state is high, the cumulative current leaks through this path and prevents any switching activity. The possibility of current leakage creeping into the neuron ADC is prevented using a DC blocking capacitor C_{DC} and the average sizing of the transistors ensures minimal charge leakage.

The functioning of the payload can be observed by analyzing the transient signal characteristics of the neuron ADC between two trigger states (see Fig. 6). When the Trojan is inactive (shown as *Trojan:Inactive*), the cumulative input current triggers a train of spikes. However, when the Trojan is active (shown as *Trojan:Active*) and an input current stimulus is provided, it can be seen that the neuron is *"suppressed"*. The deviation of current by the payload element can be confirmed by analyzing the build-up of potential on the capacitor, V_c. Hardware Trojans typically use combinational or sequential elements to monitor internal states within a system and trigger a payload based on a predefined condition [16]. Since the attacker is capable of sending specific input image vectors, combinational logic can be used, where the input vector contains a unique combination of pixels to activate a trigger circuit. We prefer complementary pull-up and pull-down network (PUN/PDN)-based combinational circuit over a standard cell-based logic tree circuit, to enable microscopic design corruption. Fig. 5 depicts the schematic diagram of the trigger circuit that implements a NAND function that directly taps the inputs from the BL switch matrix and the inclusion of a Toggle flip-flop allows for the state of the trigger circuit to switch between the two operational conditions.

VI. Experimental Setup

Due to the lack of openly available commercial eNVM neuromorphic implementations, we customize a high-fidelity simulation environment representative of several recently published designs (Fig. 1), using Neurosim [6]. A 3-layer MLP (400-neuron input layer, followed by a 100-neuron hidden layer, and a 10-neuron output layer corresponding to 10 digits) is mapped to such architecture for training using 60,000 black-and-white images in 125 epochs until its inference accuracy stabilizes (~93%). We then collect a set of output currents making up the dot-product operations by opening different rows. Neurosim faithfully models an analog synaptic device with many non-ideal device properties such as variations within Long-Term Potentiation/Depression (LTP/LTD), cycle-to-cycle conductance variation, and, spatial variations across a memory array. This ensures that a realistic weight-to-conductance mapping is implemented, and the generated current traces encompasses both temporal and spatial variations.



(a) (b)

Fig. 7: **(a) SNR comparison, (b) Offline characterization**

Note that, since this is an initial foray into this field, we limit the scope of this work to target eNVM accelerators with limited hyperparameter reconfigurability (e.g., number of layers and dimension of each layer). We leave the secret extraction of more complex models for future work. However, the key insight of this work, namely that the spiking activity of the neuron ADC can leak sensitive model parameters, is expected to hold for other architectures.

The overall neuron microarchitecture is designed and evaluated in Cadence Virtuoso and Calibre tool using the TSMC 65nm Low Power(LP) flavor PDK. Transistor-level simulations are carried out to generate power traces and other relevant transient signals that allow for generating the offline characterization power traces, as well as mimicking conditions for triggering the payload. Process, Voltage, and Temperature (PVT) variations within the neuron ADC design are considered during the *offline characterization* phase of the attack. These variations result in a deviation in the step response mapping, which is visualized as a mismatch in step width and height in comparison to ideal characteristics (Fig.7b). We inject stochastic noise sources that replicate the average switching activities that potentially occur in a co-processor [17].

The switching transients are monitored over the power rail trace, and by denoising the baseline power trace from the monitored signals, a higher SNR trace can be deduced upon which the FFT transform is applied. The trace is sampled at a 100ps sampling interval and a 4096 FFT bin trace is generated, which translates to approximately a 250 kHz resolution. To build the characterization portfolio, the dynamic input current range is generated based on the parameters of selected eNVM characteristics. The most prominent large signal frequency bins from each step of the sweep are collected and assigned to an input current value. The relevance of their magnitude can be deduced from their unique frequency signatures, as they share a similar spectral magnitude characteristic.

VII. Evaluation

A. Trojan Stealth

Design and Verification Time Detection. To remain stealthy, it is imperative that the Power Spectral Density (PSD) of the Trojan-infected malicious unit under normal operating conditions must not significantly deviate from the average PSD of unaffected hardware. The PSD of unaffected hardware (*victim*) is visualized by extracting the switching current trace from the power rail across 400 read cycles under normal operating conditions. The resulting spectrogram is then compared with that of a Trojan-infected (*clone*) malicious unit, where the payload circuits are deactivated to mimic normal operating conditions. As seen in Fig. 7 the Signal-to-Noise Ratio (SNR) deviation is under 1.75 dB, with both spectrograms exhibiting a similar fingerprint.

To ensure the undetectability of the trigger and payload circuits through test pattern generation techniques, we stipulate that the area overhead is under a margin of 0.5% [18]. The area overhead of the trigger element is a function of the synaptic core area. The number of input bits that map to the payload element controls the length of the Trojan code and hence the size of the PUN/PDN network. Every neuron ADC must be embedded with a payload element, resulting in a fixed overhead. A similar trend can be observed with the average leakage power dissipated by the Trojan (Fig. 9a and Fig. 9b).

While our ability to evaluate against verification-time detection frameworks is limited, as they are not open source, we offer a qualitative discussion. Frameworks such as FANCI [19] that operate at the RTL level would not be able to detect our Trojan, owing to its form factor and its ability to embed the Trojan at the GDS-II levels and polygon pattern etching foundry stages. Methods such as UCI [20] mainly apply for digital Trojans, as they rely on analyzing switching activity.

Run-time Detection. By exploiting the input vector to encode the necessary bits, it is possible to generate an extremely large set of combinations for the trigger code (there are 2^{400} possible codes that can be uniquely assigned to each ADC). Our analysis shows that a 50-75 bit long trigger code results in a false activation of a single Trojan, only once in 1000 random input test patterns. Furthermore, a trigger code that is at least 35 bits long can ensure the prevention of two simultaneous false activations of Trojans across 1000 input test patterns, significantly enhancing our ability to evade run-time detection.

B. Sensitivity Study

We sweep the ADC resolution from 8 to 256 steps and vary the device conductance levels, thereby simulating a wide array of neuromorphic design choices. Fig. 8 shows the inference accuracy of the original (victim) model, the cloned model using the stolen weights, and the percentage of weights recovered by the attacker for accelerators implemented using multiple eNVM technologies (EpiRAM(Ag:SiGe), HZO FeFET, TaOx/HfOx , and GST PCM). We draw four major conclusions. **First**, regardless of the underlying eNVM technology, we are able to recover more than 90% of the weights. The remainder of the weights do not build a sufficient input impulse to generate a spike train. **Second**, as the ADC resolution improves, more weights can be recovered, because the ADC resolution becomes more sensitive to the small current generated by a single eNVM cell. The overall weight extraction of our attack improves from $94.4\% \rightarrow 97.8\%$ and $64.1\% \rightarrow 97.1\%$ for a 2-bit improvement in ADC resolution, when the hardware is simulated in EpiRAM and HZO FeFET, respectively. The greater improvement in the case of HZO FeFET is attributed to the larger conductance density characteristics offered by the device. **Third**, in many cases, even an ADC with a lower resolution, poses a serious threat, as the attacker can reliably clone a model with comparable performance. For instance, when evaluating the attack strategy for a 5-bit resolution ADC, the worst case performance delta

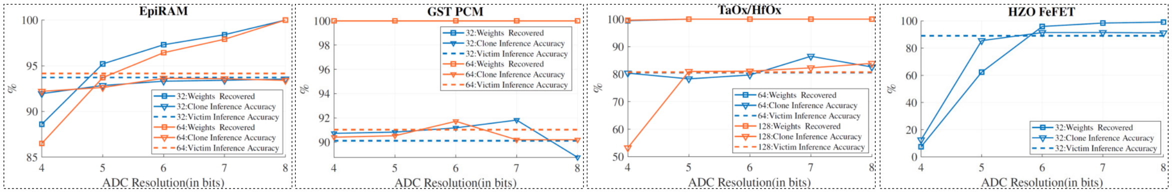
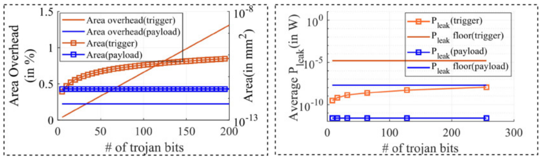


Fig. 8: **Sensitivity to Conductance Levels and ADC Resolutions for add references for devices**.



(a)

(b)

Fig. 9: **(a). Area overhead and (b).** P_{leak} **in comparison to the noise floor, as a function of the input Trojan vector.**

between the original and recovered inference accuracy across the four memory flavors is under 2.65%. **Fourth**, a higher percentage of weights recovered by the attacker does not always translate to higher inference accuracy of the cloned model. This is because ADCs are calibrated to segment continuous current ranges to resolution steps and the current generated from a single cell may be mapped to a current value that is slightly off compared to the true current. In some cases, we're able to obtain a cloned network with higher accuracy than the victim model. We suspect that this is because the weights of the victim models are sometimes stuck at local minimums.

VIII. Discussion on Mitigations and Challenges

Trojan detection. Several techniques have been proposed to prevent the insertion of a hardware Trojan into ICs. Waksman et al. [19] propose FANCI, a framework for profiling activities of wires inside a chip, and flagging nearly unused ones as possible Trojan paths. Their insight stems from the fact that Trojan functionalities are mostly dormant until triggered by external malicious inputs. This can potentially catch the Trojan logic embedded in an eNVM device. However, FANCI, as described in the paper, examines the hardware implementation at the RTL level, such as a netlist file, while the Trojan we describe can also be placed inside a layout GDSII file, thereby circumventing it. Extending detection frameworks to enable more comprehensive detection is interesting future work.

Trojan insertion prevention. To prevent insertion at the layout level, a potential countermeasure that can be used is layout masking [21]. However, this is expected to prohibitively increase the power and area overhead (by $\sim$ 10%). If an eNVM device is deployed as an IoT or a wearable device that is power/area-constrained, layout masking may not be ideal. The weight recovery attack may also be defeated by integrating an ultra-low resolution ADC with four or eight resolution steps, preventing the successful extraction of most of the weights. However, this would severely limit the capabilities of such devices to scale to larger workloads.

Side-channel prevention. Masking the signal (EM, power, thermal, etc.) signature, therefore, preventing the side-channel attacks (SCA), usually requires dedicated SCA countermeasure hardware which can be impractical (power and area overhead) to integrate for neuromorphic devices.

IX. Conclusion

We explore the feasibility of a novel supply-chain threat against eNVM neuromorphic devices. We identify ADC as the key element that exposes a vulnerability within the neuron core and further deduce the weights by analyzing and isolating the switching activities of the ADC. We also design a stealthy hardware Trojan that allows the attacker to correlate the transient system power consumption to the synaptic weight and subsequently reconstruct a cloned model with high fidelity.

Acknowledgment: This work was supported by NSF CCRI Grant CNS-2213700, NSF PPoSS Grant CCF-2217071, NSF/Intel Foundational Microarchitecture Research (FoMR) Grant CCF-1912608, Semiconductor Research Corporation (SRC) contract 2019-NM-2875, and CRISP, one of six centers in JUMP, an SRC program sponsored by DARPA.

References

[1] T.-J. Yang *et al.*, "A method to estimate the energy consumption of deep neural networks," ACSCC '17.

[2] P.-Y. Chen *et al.*, "Neurosim+: An integrated device-to-algorithm framework for benchmarking synaptic devices and array architectures," IEDM '17.

[3] K. Yang *et al.*, "A2: Analog malicious hardware," in *SP '16*.

[4] F. Tramèr *et al.*, "Stealing machine learning models via prediction apis," USENIX Security '16.

[5] IBM, "Ibm advances research in analog ai computing."

[6] P.-Y. Chen *et al.*, "Neurosim: A circuit-level macro model for benchmarking neuro-inspired architectures in online learning," ITCSDI '18.

[7] A. Shafiee *et al.*, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," ISCA '16.

[8] D. Kadetotad *et al.*, "Parallel architecture with resistive crosspoint array for dictionary learning acceleration," JETCAS '15.

[9] S. Bhasin *et al.*, "A survey on hardware trojan detection techniques," in *ISCAS '15*.

[10] Y. Jin *et al.*, "Experiences in hardware trojan design and implementation," in *HOST '09*.

[11] C. Yang *et al.*, "Security of neuromorphic computing: Thwarting learning attacks using memristor's obsolescence effect," ICCAD '16.

[12] C. Rajamanikkam *et al.*, "Understanding security threats in emerging neuromorphic computing architecture," JHSS '21.

[13] Y. Cai *et al.*, "Enabling secure in-memory neural network computing by sparse fast gradient encryption.," ICCAD '19.

[14] J. Rajendran *et al.*, "Is split manufacturing secure?," DATE '13.

[15] L. Wei, B. Luo, *et al.*, "I know what you see: Power side-channel attack on convolutional neural network accelerators," ACSAC '18.

[16] M. Tehranipoor *et al.*, "Power supply signal calibration techniques for improving detection resolution to hardware trojans," in *ICCAD '08*.

[17] V. Tenentes *et al.*, "Run-time protection of multi-core processors from power-noise denial-of-service attacks," TDMR '20.

[18] R. S. Chakraborty *et al.*, "Mero: A statistical approach for hardware trojan detection," CHES '09.

[19] A. Waksman *et al.*, "Fanci: Identification of stealthy malicious logic using boolean functional analysis," CCS '13.

[20] M. Hicks *et al.*, "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically," in *2010 IEEE Symposium on Security and Privacy*, pp. 159–172, 2010.

[21] M. Montoya *et al.*, "Adaptive masking: a dynamic trade-off between energy consumption and hardware security," ICCD '19.