MagTracer: Detecting GPU Cryptojacking Attacks via Magnetic Leakage Signals

Rui Xiao[1], Tianyu Li[1], Soundarya Ramesh[2], Jun Han[3]*, Jinsong Han[1]*
[1]Zhejiang University, Zhejiang, China
[2]National University of Singapore, Singapore
[3]Yonsei University, Seoul, Korea
ruixiao24@zju.edu.cn, tianyl@zju.edu.cn, sramesh@comp.nus.edu.sg,
jun.han@yonsei.ac.kr, hanjinsong@zju.edu.cn

ABSTRACT

GPU cryptojacking is an attack that hijacks GPU resources of victims for cryptocurrency mining. Such attack is becoming an emerging threat to both local hosts and cloud platforms. These attacks result in huge economic losses for the victims due to significant power consumption by cryptomining applications. Unfortunately, there are no adequate solutions to detect such attacks. In this paper, we propose *MagTracer*, a novel GPU cryptojacking detection system that leverages magnetic leakage signals emanating from GPUs. We make a key observation that GPUs emanate a *distinct* magnetic signal while mining, which can be attributed to the core feature of all cryptomining algorithms (as they are compute-intensive as well as memory-bounded). We design and implement a proof-of-concept detection system to demonstrate *MagTracer*'s feasibility. We evaluate *MagTracer* on 14 heterogeneous GPU models and achieve a high average true positive rate of over 98% and a low false positive rate below 0.7% in all cases. Furthermore, our comprehensive evaluation confirms that *MagTracer* is *scalable* across different mining applications and *robust* against several targeted attacks.

CCS CONCEPTS

• **Security and privacy → Side-channel analysis and countermeasures**.

*Jun Han and Jinsong Han are co-corresponding authors.

ACM MobiCom '23, October 2–6, 2023, Madrid, Spain

ACM ISBN 978-1-4503-9990-6/23/10…$15.00
https://doi.org/10.1145/3570361.3613283

KEYWORDS

Cryptojacking Detection, Magnetic Side Channel, GPU Mining, Side-channel based Defense

ACM Reference Format:
Rui Xiao[1], Tianyu Li[1], Soundarya Ramesh[2], Jun Han[3]*, Jinsong Han[1]*. 2023. *MagTracer*: Detecting GPU Cryptojacking Attacks via Magnetic Leakage Signals. In *The 29th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '23), October 2–6, 2023, Madrid, Spain.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3570361.3613283

1 INTRODUCTION

Reported cases of cryptojacking – where remote attackers access victims' computing resources to stealthily mine cryptocurrencies – are increasing significantly. According to a 2021 Linux Threat Report, cryptojacking is the largest malware family affecting Linux servers, contributing to over 24% of all threats [46]. Furthermore, in 2022, cryptojacking cases increased by 269% in the financial sector alone [16].

Large costs involved in hardware maintenance for mining is a primary reason for cryptojacking [19, 31]. As a result, attackers target victims' machines, most notably, graphics processing units (or GPUs) due to their high performance in terms of *hash rates*, which is crucial for cryptomining[1] [49, 50]. Furthermore, such cryptojacking malware has been shown to be easily disseminated to GPUs through modified software such as NVIDIA drivers and Kubernetes clusters [8, 75].

The consequences of GPU cryptojacking are detrimental for the victims as they result in significant costs due to their immense power consumption [77]. In addition, cryptojacking attacks caused loss of revenue for businesses due to degraded GPU performance, in addition to reduced GPU lifespan due to overheating [51, 69]. Meanwhile, the thriving AI industry has contributed to the increased accessibility of GPUs for cryptojacking attacks [28, 66]. Hence, developing cost-effective measures to defend against strong attackers is crucial in mitigating these risks.

[1]We use cryptojacking, cryptomining, and mining interchangeably.

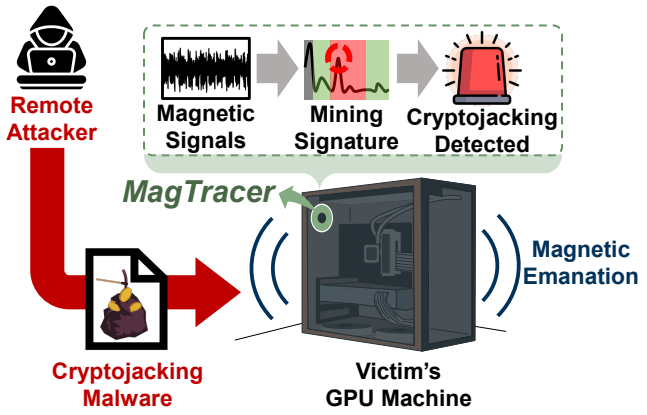

Figure 1: Figure depicts our cryptojacking attack detection scenario, where *MagTracer*'s hardware is placed in contact with the victim's GPU machine. When a remote attacker executes cryptojacking malware on the victim's GPU, *MagTracer* detects its unique *mining signature* from the captured magnetic signals and alerts the user.

There have been several efforts from academia and industry to detect cryptojacking. Some prior works suggest maintaining a predefined blacklist of malicious sites, whose requests can be automatically blocked [1, 35, 79, 83]. However, such an approach is *not scalable* due to the increasing number of rapidly evolving threats [44]. Researchers also explore detecting cryptojacking on CPUs by monitoring features such as CPU cache utilization and memory events [34, 41, 43, 53, 67, 78]. However, these approaches are *not applicable to GPUs* due to differences in both the processor architecture and the types of cryptojacking programs. Furthermore, even these attempts are prone to common evasion attacks such as setting up proxies, dynamic domain names, or encrypted communication [53].

The aforementioned shortcomings lead us to the following research question: *Can we design a novel GPU cryptojacking detection system that is - (1) scalable to unseen cryptojacking malware, (2) scalable across different GPU models, as well as (3) resilient to attacks from powerful remote adversaries?* To this end, we propose *MagTracer*, which utilizes the phenomenon that GPUs, when performing mining, ***emanate distinct magnetic leakage signals***. The leakage signal stems from the varying current transmitted through the GPU power lines. Specifically, we identify that it is the *compute-intensive* and *memory-bounded* properties of cryptomining algorithms that result in their distinct mining leakage patterns. Figure 1 illustrates our cryptojacking detection scenario, in which a remote attacker executes cryptojacking malware on a victim's GPU machine without the user's knowledge. *MagTracer* utilizes a small magnetic sensor hardware placed in contact with the machine's exterior housing to detect the presence of any distinct *mining signature* in the captured signal and ultimately alerts the victim about potential cryptojacking.

Designing *MagTracer* comes with two main challenges. The first challenge is *achieving robustness to noise* arising from external sources such as adjacent electronic components (CPUs), and internal sources such as programs running alongside cryptomining, that significantly affect the magnetic leakage signals. Furthermore, the second challenge is that the *mining signature* is highly GPU-dependent. In particular, we observe that the mining leakage *frequency* changes with the GPU model under test. Hence, *developing a detection technique that works across different GPU models* with varying architectures and specifications is challenging.

We solve the aforementioned challenges in the following manner. We incorporate different signal processing techniques including *outlier removal* and *signal detrending* to remove the effects of external and internal noise sources. Subsequently, we extract *GPU-aware features* by, first, systematically identifying a correspondence between the GPU specifications and the magnetic leakage signals. In particular, we observe that given the GPU model, we can estimate the magnetic leakage frequency obtained during mining, which we refer to as the *mining frequency*. We then extract *GPU-aware features* such as the *peak prominence*, based on the identified *mining frequency*. Finally, we further improve the robustness of internal noise sources by aggregating features across multiple magnetic signals, which we leverage to train our support vector machine (SVM) classifier.

MagTracer has several advantages. First, as *MagTracer* is designed and implemented as standalone hardware that is physically separated from the host device and lacks network access, it is unaffected by powerful remote attackers with unconstrained access to the victim's host. Second, *MagTracer* is easy to deploy as it does not require any bulky hardware. Specifically, we implement our proof-of-concept *MagTracer* hardware utilizing a lost-cost magnetic sensor (costing only about 3 USD) for magnetic acquisition and a Raspberry Pi for post-processing. If deployed, *MagTracer* hardware would have a smaller form-factor at a cheaper cost.

We evaluate *MagTracer* on 14 GPU models by capturing magnetic signals of total duration over 23 hours across several cryptomining and benign (i.e., non-mining) tasks. We comprehensively evaluate *MagTracer*'s performance over different mining applications, sensor locations, and background applications. Additionally, we perform a detailed security analysis by subjecting *MagTracer*'s detection to strong anti-detection mechanisms such as throttling that attenuate the signal-to-noise ratio (SNR) of the mining leakage signals. Overall, *MagTracer* achieves high average true positive rates above 98%, and low false positive rates below 0.7% in all cases,

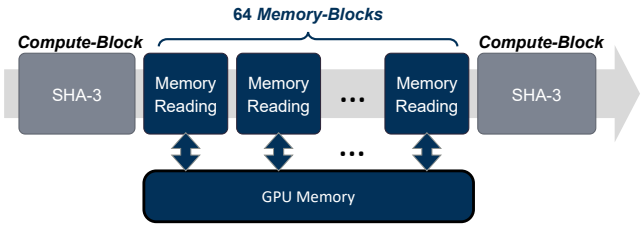

Figure 2: Figure depicts Ethash, a typical GPU mining algorithm, consisting of two compute-blocks and 64 memory-blocks.

Algorithm	Ethash	Etchash	KawPow	ProgPow	FiroPow
Hash Rate	32 Mh/s	32 Mh/s	21.5 Mh/s	21.5 Mh/s	21.5 Mh/s

Table 1: Table depicts the *hash rates* (measured in hashes per second or h/s) for popular cryptomining algorithms on an NVIDIA GTX 1080Ti GPU.

demonstrating its effectiveness across five GPU architectures from popular vendors such as NVIDIA and AMD.

2 BACKGROUND

This section presents the background of cryptomining, working of GPUs, and the causes of GPUs' magnetic leakage.

2.1 Cryptomining on GPUs

We provide an overview of cryptomining and its unique features that lead to distinct magnetic leakage in GPUs.

Existing cryptocurrencies (e.g., EthereumPoW), are based on blockchain technology, which rely on a *consensus algorithm* to verify and add new transactions. One popular consensus algorithm is Proof-of-Work (PoW), where distributed nodes called *miners* solve cryptographic puzzles, specifically *hash functions*, to add valid blocks to the blockchain. Hence, cryptomining algorithms are ***computationally intensive*** by design, and are typically executed on GPUs due to their good performance. GPUs have high ***hash rates***, i.e., the number of hashes that can be computed per second, making them well-suited for cryptomining. Hash rate is dependent both on the GPU model as well as the cryptomining algorithm, and Table 1 enumerates the hash rates for common cryptomining algorithms on NVIDIA GTX 1080Ti GPU.

In addition to being compute-intensive, GPU mining algorithms are also ***memory-bounded*** by design, i.e., the hash functions involve significant amounts of memory readings, making memory the major bottleneck in execution time [32]. This memory-bounded feature is intended to make mining resistant to Application-Specific Integrated Circuits (ASICs), hence preventing domination of large players [4, 80].

Figure 2 depicts the high-level working of a popular cryptomining algorithm, *ETHash*, used for mining EthereumPoW [20]. ETHash is a modified version of SHA-3 hash function, specifically consisting of two SHA-3 hash function modules (*compute-block*), and 64 random table lookups (*memory-block*) for each run. In Section 3.3, we further explore how these compute and memory blocks impact mining signature.

2.2 GPU Computational Model

GPUs are high-performance processors that follow Single Instruction Multiple Data (SIMD) architecture, thereby enabling efficient computing through data-parallelism, making them well-suited for cryptomining.

The mining algorithm is implemented as a GPU program called a ***kernel***. Upon execution, the kernel launches a large number of parallel computing ***threads*** on the GPU. Threads will be assigned to the hardware units of computation, known as ***Streaming Multiprocessors (SMs)*** for execution. A GPU has a scalable number of SMs. For example, the NVIDIA GTX 1080 Ti [54] has 28 SMs. The mining kernel will be executed on all these 28 SMs. Each SM has its own set of resources, such as ***registers*** and shared memory. As each thread requires its own set of registers to execute, the total number of registers determines how many threads (or hashes) can be computed simultaneously on an SM. During mining, the GPU scheduler will periodically assign threads to the SMs for execution, which we refer to as ***thread assignment***. In Section 5.3, we elaborate on how the periodicity of these *thread assignments* during cryptomining leads to its unique mining signature.

2.3 Magnetic Emanations from GPU

As a typical electronic component, GPUs emit – high frequency electromagnetic signals due to digital circuits (e.g., VRAM clocks), as well as low frequency magnetic signals due to components such as power lines. The signals due to the former are typically in the order of MHz or higher and can be attenuated through shielding. Hence, in this work, we explore low frequency magnetic signals, which are produced as a side-effect of the electric current transmitted through power lines connecting the GPU and the power supply unit. Specifically, according to the Biot-Savart Law, the magnitude of the magnetic field due to a long current carrying wire with current I can be described as – $\mathbb{B} = \frac{\mu_0 I}{2\pi r}$, where μ_0 is the magnetic constant, and r is the distance from the wire [38].

During cryptomining, high magnitude currents are produced in GPUs (reaching up to 30A), which in turn result in strong magnetic fields, on the order of $100\mu T$, in close proximity to the GPU [13]. In this paper, we analyze these leaked magnetic signals from GPUs for cryptojacking detection.

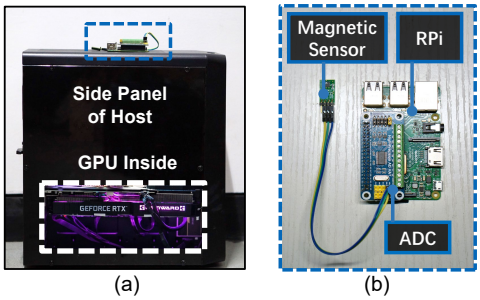

Figure 3: Figure (a) depicts the side panel of the host device equipped with a GPU. *MagTracer*'s setup is affixed to its top panel; (b) depicts our setup's zoomed-in view which consists of a low-cost magnetic sensor (3 USD), an ADC, and a Raspberry Pi.

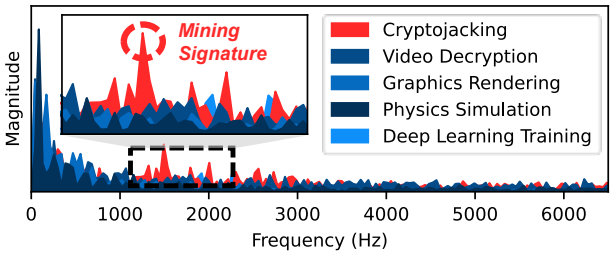

Figure 4: Figure depicts that cryptojacking program depicts a distinct mining signature in the frequency domain, in comparison to representative non-mining tasks, such as graphics and deep learning training.

3 FEASIBILITY STUDY

In this section, we verify the feasibility of GPU cryptojacking detection using magnetic leakage signals.

3.1 Feasibility Setup

Our setup (Figure 3) consists of a test device (i.e., device with a GPU), a DRV425 fluxgate magnetic sensor, which outputs voltage proportional to magnetic field strength (sensitivity = 5mv/μT), an ADS1263 analog-to-digital converter that digitizes the magnetic signals at a sampling rate of 38.4 KHz, and a Raspberry Pi 4B for post-processing [36, 37]. For our feasibility study, we leverage a test device equipped with a GTX 1080 Ti GPU.

3.2 Evidence of Mining Signature

We now verify if cryptomining applications are distinguishable from other applications executed on a GPU. For this experiment, we capture magnetic signals when the GPU performs cryptojacking, as well as four other tasks typically performed on a GPU, namely, deep learning training, 3D graphics rendering, video decryption, and physics simulation. From Figure 4, we observe that cryptojacking indeed produces a *unique magnetic signature*, which we refer to as *mining signature*, compared to all other tasks. In particular, cryptojacking produces a distinct peak at around 1.5 kHz. This experiment thus confirms our hypothesis that magnetic signals can serve as a proxy for cryptojacking detection.

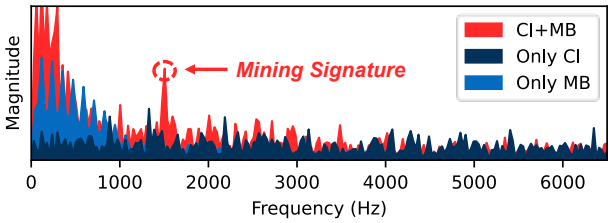

Figure 5: Figure depicts the significance of both the compute-intensive (CI) and the memory-bounded (MB) components of the cryptomining algorithm, Ethash, towards the occurrence of the mining signature.

3.3 Cause for Distinct Mining Signature

Recall from Section 2.1 that cryptomining consists of both a *compute-intensive* as well as a *memory-bounded* components. We now perform experiments to verify their contributions to the detected *mining signature*. In particular, we conduct two preliminary experiments by separately removing the compute-intensive and memory-bounded components of the Ethash cryptomining algorithm. As depicted in Figure 5, we observe that the mining signature disappears in the absence of either, indicating the importance of both components in generating the unique signature. Furthermore, this indicates that non-mining applications with repetitive computations are still unlikely to produce magnetic patterns similar to mining due to the lack of a memory-bounded component that is unique to cryptomining applications.

4 SYSTEM AND THREAT MODEL

We present the system and threat models of *MagTracer*.

System Model. The *goal* of *MagTracer* is to leverage magnetic leakage signals to detect *GPU cryptojacking* in victim-owned GPU hosts, such as personal computers and GPU servers. To provide such detection, we *require MagTracer* to be – (1) scalable to unseen mining programs, (2) applicable to heterogeneous GPU models, and (3) resilient to powerful remote attacks. To achieve this, we *assume* that the magnetic sensor is placed in a standalone manner in close proximity (within a few cm) of the target device's exterior.

Threat Model. The attacker's *goal* is to conduct *profitable* cryptojacking activities by mining cryptocurrencies using

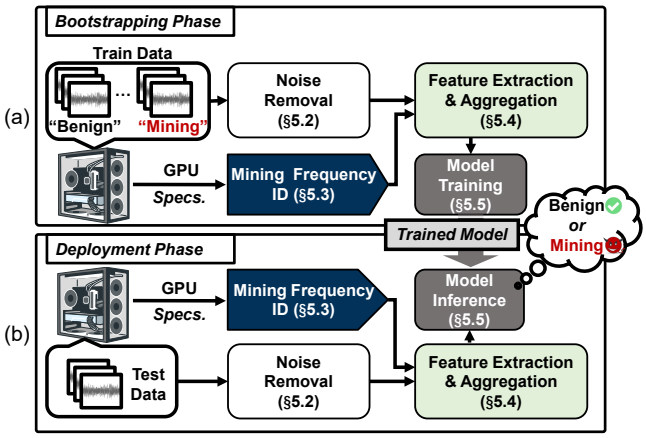

Figure 6: Figure depicts *MagTracer*'s design overview. (a) depicts *Bootstrapping Phase*, where we capture magnetic signals emanated by GPUs while executing mining and benign programs to ultimately train a binary classifier. (b) depicts *Deployment Phase* where we leverage the trained model to detect cryptojacking based on magnetic samples from the GPU under test.

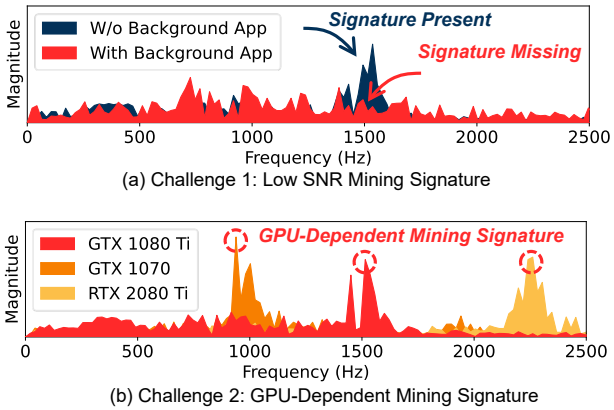

Figure 7: Figure depicts the two design challenges: (a) the mining signature has low SNR due to noise sources (e.g., background applications), and (b) the *mining frequency* is highly dependent on the GPU model.

the victim's GPU resources. The attacker's *capabilities* include launching *remote attacks*, such as injecting cryptojacking malware into victim GPUs through modified software [8, 75]. Advanced attackers are capable of bypassing software-based defenses, e.g., in the past, they have bypassed binary analysis-based defenses using obfuscation [53, 56]. Furthermore, in certain cases, attackers can exploit system vulnerabilities to gain root privileges and disable all host detection mechanisms [17, 18]. This underscores the significance of *MagTracer* as a robust defense mechanism, as state-of-the-art software-based defenses would be insufficient against such sophisticated attacks [29, 58, 61].

5 *MAGTRACER* DESIGN AND IMPLEMENTATION

We now present *MagTracer*'s design and implementation.

5.1 Design Overview

MagTracer's design enables the detection of cryptojacking through the magnetic side-channel of GPUs, which consists of two phases, namely the *Bootstrapping phase* and the *Deployment phase*, as shown in Figure 6. During the *Bootstrapping Phase*, magnetic traces are collected from a sensor placed on the external of a GPU to train a binary SVM classifier that can differentiate between mining applications and benign applications, such as gaming or video encoding. In the *Deployment Phase*, the trained classifier is used to detect cryptojacking events based on the magnetic leakage signals.

The collected magnetic traces in both phases are first preprocessed in the Noise Removal module (§5.2) to filter out ambient and internal noises. Concurrently, the expected leakage frequency for mining, defined as *mining frequency*, is estimated using GPU specifications in the Mining Frequency Identification module (§5.3). The denoised signals and *mining frequency* are then input to the Feature Extraction and Aggregation module (§5.4) where features related to GPU mining activity are extracted and aggregated across multiple trials. Finally, the computed features are used for mining detection in the Model Training and Inference module (§5.5).

Designing *MagTracer* involves two main challenges –

Challenge 1: Low SNR Magnetic Side-Channel. The magnetic side-channel is inevitably affected by both the *external* and *internal* noise sources. The *external* noise sources refer to the electronic components surrounding the protected GPU, such as CPUs and power supply units, which emit strong magnetic signals, thereby interfering with the mining signature. The *internal* noise sources refer to the execution of non-mining programs, such as graphics processing, in parallel with mining applications. The magnetic signal due to mining is altered because of GPU *time-slicing*, where each application gets a time-shared access to the GPU. As a result, the signal-to-noise ratio (SNR) of the GPU magnetic side-channel is low. Figure 7(a) illustrates how the *mining signature* disappears to the noise floor when it is executed together with a graphics application. We address this challenge in the Noise Removal (§5.2) as well as the Feature Extraction and Aggregation modules (§5.4) modules.

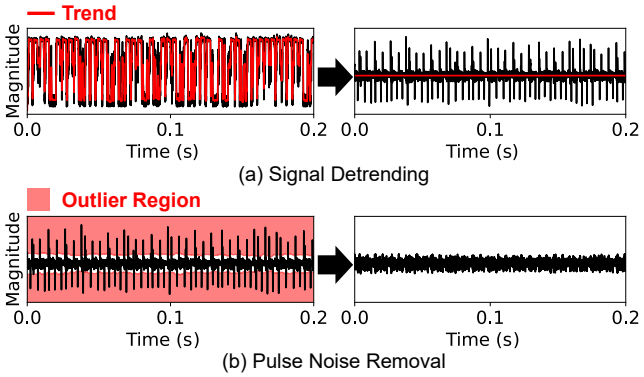

Figure 8: Figure depicts the effects of (a) Signal Detrending and (b) Pulse Noise Removal steps, in the Noise Removal module.

Challenge 2: GPU-Dependent Mining Signature. In the Feasibility section (§3.2), we demonstrate that mining programs result in a unique *mining signature* compared to non-mining applications. However, we encounter a technical challenge in determining the *optimal frequency band* to monitor as the *mining frequency* changes with the GPU model. We illustrate this changing *mining frequency* on three GPU models in Figure 7(b). Hence, we address this challenge with the Mining Frequency Identification module (§5.3), where-in we systematically estimate the *mining frequency* given the GPU specifications, and leverage this frequency to subsequently learn *GPU-aware* features.

5.2 Noise Removal

This module takes as input the raw time-series data captured from the magnetic sensor and performs several preprocessing steps which are crucial to address external and internal noise sources (i.e., Challenge 1), to finally output a denoised signal. We list each of the steps, in order, below.

Signal Detrending. The collected magnetic signals are subject to transient level shifts, caused by the dynamic transfer between different GPU processes (i.e., mining and non-mining), as well as long-term level shifts caused by external noises. To address this issue, as depicted in Figure 8(a), we remove these level shifts by *detrending* the signal through the subtraction of the moving mean (i.e., the red line) computed over a window of 50 samples (or 1.3 ms).

Pulse Noise Removal. Magnetic signals also consist of impulsive noise generated by surrounding electronic components (e.g., CPUs). We resolve this type of noise by identifying and eliminating outliers of the time series using the *Hampel filter* [57]. Specifically, the Hampel filter computes the median, m, and standard deviation, σ, of the samples in each sliding window of 200 samples. If the value of a time sample deviates beyond the value, 3σ, from the median, m, such a sample is identified as an outlier and is replaced with the median's value, m (see Figure 8(b)).

Normalization. We perform *z-normalization* on the magnetic signal, $x(t)$, and obtain the denoised signal, $\tilde{x}(t)$, to normalize the signals collected across varying distances and locations. Specifically, we compute the denoised signal, $\tilde{x}(t)$, as $\frac{(x(t)-\mu)}{\sigma}$, where μ and σ are the mean and standard deviation of the overall magnetic signal [60].

5.3 Mining Frequency Identification

This module takes as input the specifications of the GPU model under test to output the estimated *mining frequency* range. Recall that the *mining signature* changes with the GPU model (i.e., Challenge 2). Hence, in this module, we account for the various GPU specific parameters (e.g., number of streaming multiprocessors (SMs), number of registers per SM) to systematically identify the minimum and maximum possible *mining frequency* per GPU model, namely f_{mine}^{min} and f_{mine}^{max}, respectively. The resulting *mining frequency* range is then used to extract features of mining signature in the Feature Extraction and Aggregation module (§5.4).

Recall from our background on GPU computational model (§2.2) that each kernel invocation involves multiple *thread assignments*. Within each assignment, the GPU scheduler assigns as many threads as possible while adhering to resource constraints. From our empirical analysis, we observe that during cryptomining, the magnetic side channel produces a *periodic peak* for every *thread assignment*, as depicted in Figure 9. We propose that the periodic leakage patterns observed may be attributed to the repeated execution of hash computations in cryptomining. Such computations are performed across multiple thread assignments and kernel invocations, leading to periodic leakage patterns throughout mining program execution. Hence, the leakage frequency during cryptomining, or the *mining frequency*, can be computed as the number of thread assignments every second.

Mining frequency, or the *rate of thread assignments*, depends on the *GPU specifications*, specifically the number of SMs (n_{sm}), and the number of registers per SM (n_{reg}). Furthermore, *mining frequency* is also dependent on characteristics of the *cryptomining algorithm*, specifically the number of registers required per thread for mining ($\tilde{n}_{reg}$), as well as its *hash rate* ($\mathcal{H}$) for the given GPU model. However, as we have access only to the GPU model specifications and not the mining algorithm, we compute a range of possible values for the *mining frequency* (f_{mine}^{min}, f_{mine}^{max}) as explained below.

An Example. Let us consider the GPU GTX 1080 Ti as an example, with the number of SMs (n_{sm}) = 28, and the number of registers per SM (n_{reg}) = 65535. If each thread execution for cryptomining takes up $\tilde{n}_{reg}$ registers = 80, then the number

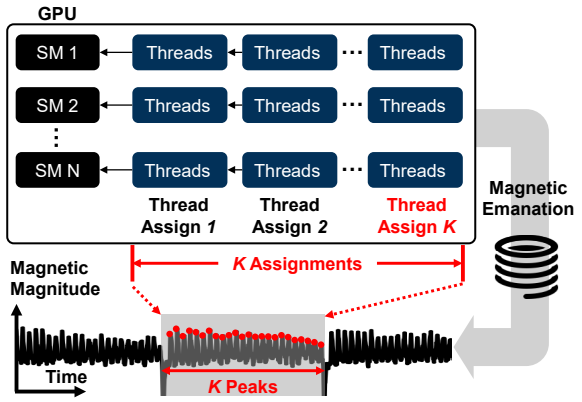

Figure 9: Figure depicts our observation that each *thread assignment* leads to a corresponding peak in the magnetic signals for cryptomining. We further leverage this observation to compute the *mining frequency*.

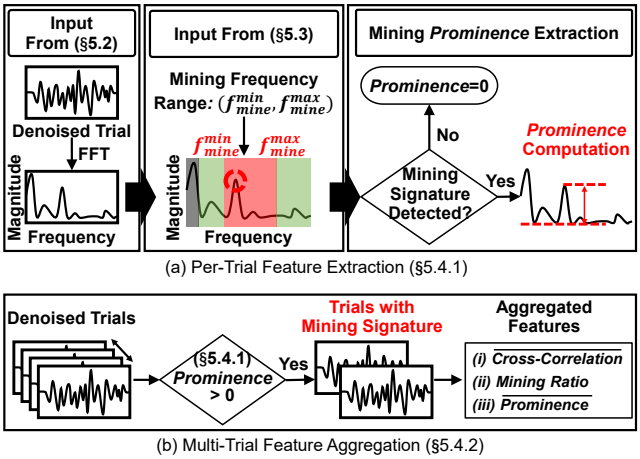

Figure 10: Figure (a) depicts the Per-Trial Feature Extraction step which takes as input the denoised signal as well as the range of *mining frequency*, to output the *peak prominence*; (b) depicts the Multi-Trial Feature Aggregation step which computes statistical features across several trials with non-zero prominence values.

of threads per thread assignment in GPU, $n_{threads} \approx n_{sm} * (n_{reg}/\tilde{n}_{reg})$ threads[2]. Subsequently, the *mining frequency*, f_{mine}, is computed as the ratio, $(\mathcal{H}/n_{threads})$, which equals 1,473 Hz[3], when the hash rate, $\mathcal{H}$ = 33 Mh/s. However, note that, in general, we only know approximate values for the number of registers for mining, i.e., $\tilde{n}_{reg} \in [75, 85]$ registers, as well as the hash rate, i.e., $\mathcal{H} \in [20, 35]$ Mh/s. Consequently, we estimate the minimum and maximum *mining frequency* as – f^{min}_{mine} = 827 Hz and f^{max}_{mine} = 1,628 Hz, respectively.

5.4 Feature Extraction and Aggregation.

This module takes as input the *pre-processed signal* (§5.2) as well as the GPU *mining frequency* range, i.e., $(f^{min}_{mine}, f^{max}_{mine})$ Hz (§5.3) to output *GPU-aware* mining features. Specifically, we segment the magnetic signal into smaller segments (512 samples each), which we refer to as *trials*, to perform per-trial feature extraction. These per-trial features are then aggregated to obtain statistical features that are subsequently leveraged for model training/inference. The steps involved in this process are elaborated below.

5.4.1 Per-Trial Feature Extraction. This step takes the magnetic signal of each trial to output their mining *prominence* feature, as depicted in Figure 10(a). *Prominence* measures the height of a peak relative to its neighboring peaks. Specifically, for each of the trials, we first compute the magnitude spectrum based on the Fast Fourier Transform (FFT) algorithm [7]. Subsequently, we identify the frequency peak, f_{peak}, with the maximum *prominence*. If f_{peak} falls within the *mining frequency* range, i.e., between f^{min}_{mine} and f^{max}_{mine}, we output its corresponding *prominence* value. However, if f_{peak} is out of range, we output zero, as it indicates the absence of significant *mining frequency* in the magnetic signal, suggesting that the corresponding trial includes potential execution of non-mining applications.

5.4.2 Multi-Trial Feature Aggregation. We aggregate features across multiple trials to generate statistical features that we utilize for model training or inference.

Although *mining prominence per trial* obtained from the feature extraction step (§5.4.1) is an indicator of cryptomining activity, it may still be error prone in the presence of background applications due to noisy magnetic signals caused by sharing GPU time (Challenge 1). Hence, we address this challenge by computing aggregated features over multiple trials, as depicted in Figure 10(b). In particular, we learn three features: ***(i) Mining Ratio*** is computed as the fraction of trials with a positive *prominence* value output by the Feature Extraction Module (recall that feature extraction step returns zero if there is no f_{peak} within the *mining frequency* range). ***(ii) Mean Prominence*** is computed as the average *prominence* among all trials with positive *prominence* values. To compute ***(iii) Mean Correlation***, we first compute the maximum correlation between all pairwise trials with positive prominence scores. Subsequently, we compute the average of the maximum correlation scores to obtain the *mean correlation*. We leverage this feature due to the temporal similarity of magnetic traces during cryptomining.

[2]In general, the exact number of threads will be affected by warp execution, execution of background applications among others.

[3]After correcting for warp execution, i.e., threads execute in sets of 32.

Applications	# Tasks	# Traces
Deep Learning Training	13	9204
Cryptography	5	3540
Games & Graphics	11	8496
Video & Image En/Decoding	7	4956
Image & Signal Processing	13	9204
Algebra & Optimization	11	7788
Physics Simulation	4	2832
Statistics & Other	16	11800
Total	**80**	**57820**

Table 2: Table enumerates the eight benign (non-mining) applications, along with the number of tasks and magnetic traces per application leveraged for evaluating *MagTracer*.

5.5 Model Training and Inference.

In this module, we input the aggregated features for each magnetic trace along with their binary labels (i.e., *mining* or *benign*), to train a binary classifier (during *Bootstrapping*) and detect cryptojacking (during *Deployment*). Specifically, in *Bootstrapping* phase, we train a light-weight Support Vector Machine (SVM) binary classifier with Radial Basis Function (RBF) as the kernel [2]. Subsequently, during *Deployment*, we input the aggregated features into the SVM model to obtain a prediction on potential cryptomining events. We choose SVMs over neural networks due to their ability to achieve high classification accuracy in the presence of small training dataset as well as low-dimensional features [40].

6 EVALUATION

We present the evaluation of *MagTracer* through comprehensive real-world experiments, demonstrating its feasibility.

6.1 Experimental Setup

Platform. Our setup (Figure 3) consists of a computer with a GPU, a DRV425 magnetic-field sensor, an ADS1263 ADC with 38.4 kHz sampling rate, and a Raspberry Pi 4B [36, 37, 62]. The computer consists of an AMD Ryzen 5 2600X CPU and runs Windows 10 OS. We place the magnetic sensor attached to the side panel of the computer, unless mentioned otherwise. We collect a total of 512 samples from the magnetic sensor per trial, and aggregate a total of 75 trials (with a total duration of about 1 second) per SVM model training or inference. Henceforth, we refer to the aggregated signal of 75 trials a *magnetic trace*.

Data Collection. We evaluate *MagTracer* on a total of 14 GPU models released in the past seven years from popular GPU vendors, NVIDIA and AMD. These models vary in their architecture and number of streaming multiprocessors as depicted in Table 3. We collect magnetic traces for different types of mining and non-mining applications. For mining, we leverage *ethminer*, a widely-used open-source implementation GPU mining program, as our baseline application [21, 25], and we capture a total of 18,998 traces (i.e., 1,357 traces per GPU). For non-mining setting, we collect a large dataset of 57, 820 magnetic traces for 80 tasks from 8 representative applications including deep learning training, gaming, and image processing (see Table 2). While we collect magnetic traces for all 80 tasks from NVIDIA GPUs, we collect traces from only 10 of those tasks for AMD due to CUDA dependency for all other tasks, making them inapplicable to AMD GPUs. Overall, we collect magnetic traces of a total duration exceeding 23 hours.

We evaluate *MagTracer*'s performance over unseen mining software and cryptocurrencies (§6.4.1 & §6.4.2), its robustness to varying sensing distances and locations (§6.4.3 & §6.4.4), different background applications (§6.4.6) as well as ambient noise (§6.4.7). We also test its performance in the presence of unseen GPUs (§6.4.8). In addition, we perform detailed security analysis where we evaluate *MagTracer*'s resilience to advanced attacks, namely binary obfuscation (§6.5.1), GPU throttling (§6.5.2) as well as several targeted attacks on *MagTracer*'s detection (§6.5.3).

Performance Metrics. We define *True Positive Rate (TPR)* and *False Positive Rate (FPR)* to evaluate *MagTracer*'s performance on cryptojacking detection. We consider a magnetic trace to be a *positive* example if *MagTracer* recognizes it as a mining magnetic trace (and a negative example otherwise). Hence, we define TPR as the fraction of all traces that are identified to be positive examples, when the GPU is cryptomining, and FPR as the fraction of all traces that are identified as positive examples when the GPU is not cryptomining.

6.2 *MagTracer* Overall Performance

We evaluate *MagTracer*'s performance across 14 GPU models, by collecting a total dataset of 18, 998 positive (i.e., cryptomining) traces and 57, 820 negative traces from a wide range of benign applications (Table 2). Among the 80 non-mining tasks, it is noteworthy that 36 of them demonstrate GPU usage exceeding 99%, comparable to that of cryptojacking.

In Table 3, we present the 14 different GPU models tested, their architecture, number of SMs in them, as well as their computed minimum and maximum *mining frequency*, f_{mine}^{min} and f_{mine}^{max}, based on the GPU specs. We now evaluate *MagTracer* by utilizing 70% of collected traces from each GPU for training, and the rest for testing. Figure 11 illustrates that *MagTracer* achieves a high average TPR of 98.6%, while maintaining an FPR below 0.69% across all GPUs. This highlights the effectiveness of *MagTracer* in distinguishing between cryptojacking and compute-intensive applications. These results demonstrate the efficacy of *MagTracer*'s design in

No.	GPU Models	Architecture	# SMs	f^{min}_{mine} (Hz)	f^{max}_{mine} (Hz)
1	GTX 1060	[N] Pascal	10	1042	2604
2	GTX 1070	[N] Pascal	15	1119	2040
3	GTX 1070Ti	[N] Pascal	19	914	1919
4	GTX 1080Ti	[N] Pascal	28	827	1628
5	GTX 1660Ti	[N] Turing	24	641	1519
6	RTX 2060	[N] Turing	30	694	1345
7	RTX 2070	[N] Turing	36	675	1483
8	RTX 2070S	[N] Turing	40	608	1335
9	RTX 2080Ti	[N] Turing	68	536	1057
10	RTX 3060	[N] Ampere	28	889	1907
11	RTX 3060Ti	[N] Ampere	38	822	2056
12	RTX 3070	[N] Ampere	46	694	1698
13	RX 5500XT	[A] RDNA	22	1182	2364
14	RX 580	[A] Polaris	36	552	1694

Table 3: Table enumerates the 14 tested GPU models, along with their architecture, number of streaming multiprocessors (SMs), as well as the computed minimum and maximum *mining frequency*, f^{min}_{mine} and f^{max}_{mine} (as explained in §5.3). Here [N] indicates an NVIDIA GPU, and [A] indicates an AMD GPU.

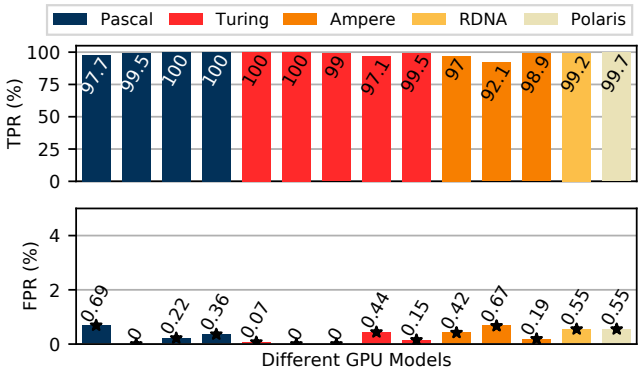

Figure 11: Figure depicts the individual TPRs and FPRs for the 14 GPU models of five different architectures.

estimating the *mining frequency* for detecting cryptomining and its ability to scale across different GPU models. Furthermore, note that our considered setting to collect training data from the same GPU models that we test is valid given our system model (§4) where-in the victim installs *MagTracer* to monitor their own device.

6.3 Performance of System Modules

We now evaluate the different design modules of *MagTracer*, and justify their selection.

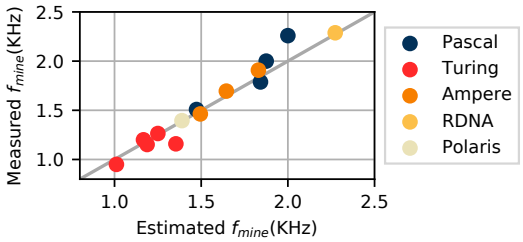

Figure 12: Figure depicts that the estimated *mining frequency* of the Mining Frequency ID Module closely matches the measured value from the magnetic signals.

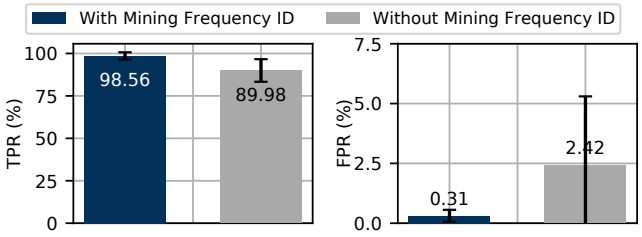

Figure 13: Figure depicts the TPR and FPR values of *MagTracer* with and without the Mining Frequency Identification module.

6.3.1 Mining Frequency ID Performance. We evaluate this module – (1) for its accuracy in estimating *mining frequency*, and (2) its impact on overall cryptomining detection.

Recall from Mining Frequency ID module (§5.3) that *mining frequency* depends both on GPU specifications as well as the mining algorithm. In this evaluation, we compare our estimated *mining frequency* (computed assuming access to GPU and algorithm information) and measured *mining frequency* (obtained from magnetic signals). Specifically, we evaluate on all 14 GPUs with Ethash mining algorithm. As depicted in Figure 12, our estimated values closely match the measured *mining frequency*, with a mean error of 4.5%.

We perform another experiment where we compare the overall performance of cryptojacking detection, with and without the *mining frequency* estimation module. In the absence of this module, we selected the frequency band 500 Hz to 3 KHz, which is a wideband range that covers *mining frequency* across all 14 GPUs. As depicted in Figure 13, our estimation module improves the average TPR by 8.58% and drastically reduces the average FPR from 2.42% to 0.31%, thereby demonstrating its overall significance.

6.3.2 Performance of Different Classifiers. We evaluate six different classifiers including SVM, Random Forest, Multilayer Perceptron, K-Nearest-Neighbors, Decision Tree, and AdaBoost, for performing binary classification (§5.5). The average TPRs and FPRs for all classifiers are above 98.1%,

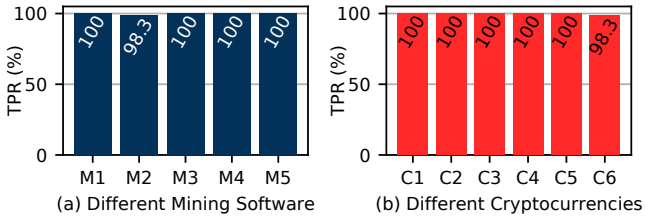

Figure 14: Figure depicts the TPR achieved by *MagTracer* for (a) different mining software and (b) different cryptocurrencies.

and below 0.53% respectively. As the different classifiers indicate only marginal differences, we chose SVM for its overall generalization performance [40].

6.4 Differing Experimental Conditions

We evaluate *MagTracer*'s performance across several factors. For this purpose, we perform our experiments on a representative GPU – NVIDIA GTX 1080 Ti. Unless mentioned otherwise, in all these experiments we leverage the same trained model based on data from the GTX 1080 Ti model.

6.4.1 Impact of Unseen Mining Software. To test the effectiveness of *MagTracer* in detecting cryptojacking in the presence of unseen GPU mining software, we report TPR by performing cryptomining using Bminer, GMiner, lolMiner, Nanominer, and PhoenixMiner mining software [6, 21, 30, 47, 52, 59]. Note that these mining programs differ from the mining program utilized to train our model (Ethminer). As observed in Figure 14(a), *MagTracer* achieves TPRs above 98% in all cases, depicting the effectiveness of *MagTracer* in identifying unseen mining software.

6.4.2 Impact of Unseen Cryptocurrencies. In the previous evaluation (§6.4.1), we test *MagTracer* for unseen mining software for EthereumPoW cryptocurrency alone. Hence, we now evaluate *MagTracer* for six mainstream cryptocurrencies in GPU mining, namely – ETHF, ETC, RVN, SERO, ZANO, FIRO [22, 24, 26, 65, 68, 82]. These cryptocurrencies apply various mining algorithms (§5.4) [14, 20, 23, 27, 64, 81], which have different hash rates on a given GPU, affecting the *mining frequency* (Section 5.4). Their hash rates on GTX 1080 Ti vary from 21.5 Mh/s to 35 Mh/s, resulting in *mining frequency* varying from 827 Hz to 1,628 Hz. As depicted in Figure 14(b), *MagTracer* achieves high TPRs above 98% in all cases. We attribute this high performance to the fact that all GPU mining algorithms adhere to the GPU PoW principle, wherein they execute simple and repetitive hash functions that are both compute-intensive and memory-bounded. Hence, algorithms from different cryptocurrencies produce similar leakage signals even though they exhibit different *mining frequencies*.

6.4.3 Impact of Sensing Distance. In order to evaluate the effect of sensing distance, we test *MagTracer*'s performance for varying distances up to 40 cm from the device's exterior side panel. In particular, as depicted in Figure 15, we observe that *MagTracer* achieves a high TPR above 91.5% and 98.3% at a distance of 24 cm and 28 cm for the computer's side panel built of metal and plastic respectively. While the metal casing attenuates the magnetic signal, it is not as effective against low-frequency GPU leakage signals, hence reducing detection distance merely by 4 cm. These findings, particularly regarding metal casing, underscore *MagTracer*'s robustness against conductive shielding and ferromagnetic objects.

6.4.4 Impact of Sensor Location. We evaluate our system by placing the sensor on four exterior panels of the host device. At each location, we also vary the orientation of magnetic sensor, which affects its sensitivity axis. We depict the average TPR for the three different orientations at each location in Figure 16. The results indicate that performing detection from the left panel of host achieves the highest TPR compared to other locations. This is because the left panel is made of plastic while the other three panels are metal. Besides, the left panel is closest to the GPU power line, which emanates strong magnetic signals. However, the average TPR across all the locations is still higher than 97.7% (σ = 2.59%). This evaluation illustrates the negligible impact of sensor placement, depicting *MagTracer*'s *usability*.

6.4.5 Impact of Sensing Duration. Recall that *MagTracer* combines 75 magnetic signal trials, which account to a sensing duration of one second to detect cryptojacking (§6.1). In this experiment, we vary the number of trials such that the sensing duration varies from 0.1 s to 2 s. As depicted in Figure 17, *MagTracer* achieves a TPR higher than 99% even for a low sensing duration of 0.5 seconds, depicting its potential for fast detection of cryptojacking.

6.4.6 Impact of Background Applications. We evaluate the impact of background software on *MagTracer*'s detection by running cryptojacking software together with other 16 benign GPU tasks, including seven games and graphics programs, three deep learning training tasks, three image processing programs, and three algebra programs. Executing background applications will throttle the *hash rate* for cryptomining applications. We define *throttling ratio* as the fractional reduction in *hash rate* in the presence of background programs, where a higher throttling ratio indicates a more compute-intensive background application. From Figure 18, we observe that the average TPR is above 96.5% when the throttling ratio is lower than 0.79. This indicates for background programs that are relatively less GPU-intensive (e.g.,

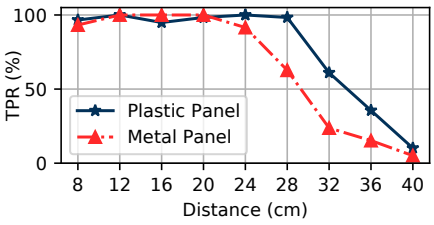

Figure 15: Figure depicts TPR under different sensing distances.

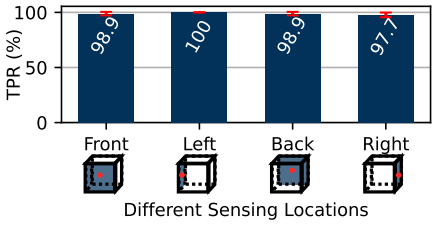

Figure 16: Figure depicts TPR at different sensor placement locations.

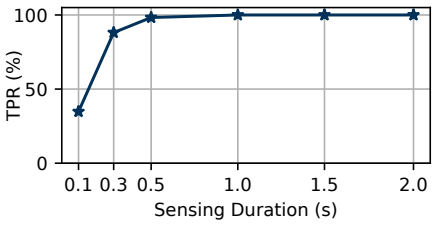

Figure 17: Figure depicts TPR under different sensing duration.

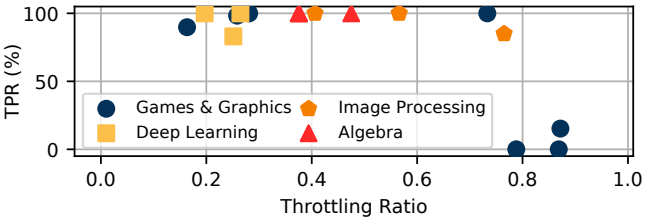

Figure 18: Figure depicts the effect on TPR due to execution of background applications together with mining.

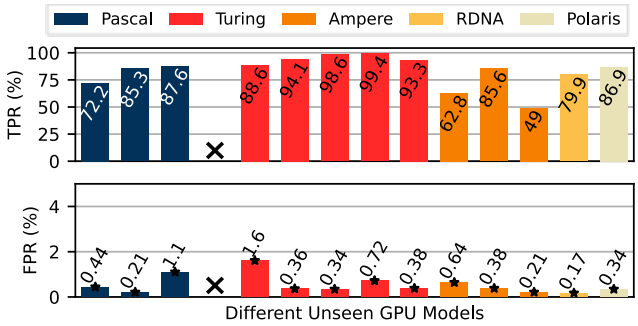

Figure 19: Figure depicts *MagTracer*'s cross-GPU detection performance when trained on GTX 1080 Ti and tested on the other 13 GPUs. We do not evaluate the performance on the trained GPU (denoted by a '✗').

algebra tasks), cryptomining can be successfully detected with a high TPR. However, when the mining software is running in parallel with highly compute-intensive games and graphics benchmarks, i.e., with a throttling ratio over 0.79, the TPR drops below 50%.

6.4.7 Impact of Ambient Magnetic Noise. We now evaluate the effect of external noise on performance of *MagTracer*. So far, all our evaluations are done in the presence of other electronic components on computers, such as power supply units, memory, and CPUs, all emanating EM noises. To further evaluate the impact of ambient noises, we conduct experiments in a $4m \times 4.5m$ server room housing more than 30 CPU/GPU servers. In this setting, *MagTracer* achieves a high TPR and low FPR of 97.4% and 0% respectively. This result depicts the robustness of *MagTracer* to such ambient magnetic noises, which is primarily attributed to our Noise Removal module (§5.2) that significantly filters all non-mining patterns.

6.4.8 Impact of Unseen GPU Models. To evaluate the transferability of *MagTracer*, we conduct cross-GPU experiments where the SVM model is trained on data from GTX 1080 Ti and tested on the other 13 GPUs, as shown in Figure 19 (the '✗' in the figure corresponds to the GPU model utilized for training). Our results indicate that *MagTracer* can detect cryptomining events with an average TPR of 83.3% and FPR of 0.53%. We attribute the degradation in the performance to the varying SNRs in the magnetic emanation pattern across GPUs. In particular, we observe that for the RTX 3070 GPU which achieves the lowest TPR of 49%, the noise floor of the magnetic signal is significantly higher, making the mining signature less prominent in comparison to other GPUs. Despite this case, our results suggest that *MagTracer* achieves over 85% accuracy in 9/13 GPU models in the cross-GPU setting, thereby demonstrating its transferability. We believe that fine tuning the SVM model with fewer training samples from each GPU can further improve the performance.

6.5 Security Analysis

In this subsection, we evaluate the robustness of *MagTracer* in the presence of several advanced techniques that could be employed by a remote adversary to evade our detection.

6.5.1 Robustness against Obfuscation. Attackers may use binary obfuscation to evade detection by altering the structure of the mining executable while maintaining its original functionality [53, 56]. To evaluate its impact on *MagTracer*, we use two well-known packers, UPX and Enigma, to obfuscate mining executables [74, 76]. In particular, we utilize UPX, a popular executable compressor used for binary obfuscation, to perform five levels of file compression from 36.08% to 52.03%. Similarly, we leverage Enigma to perform API obfuscation at three different levels, where a higher level activates additional features. As depicted in Figure 20, our results show that *MagTracer* maintains a TPR above 98% in all cases,

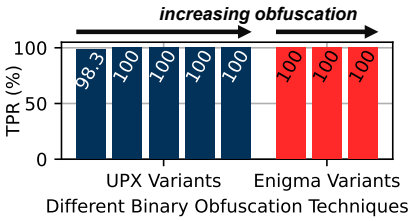

Figure 20: Figure depicts the effect on TPR due to binary obfuscation.

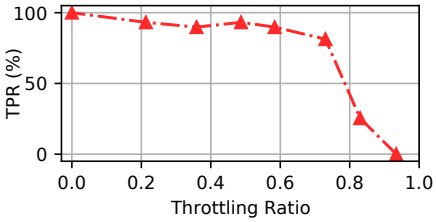

Figure 21: Figure depicts the effect on TPR due to throttling.

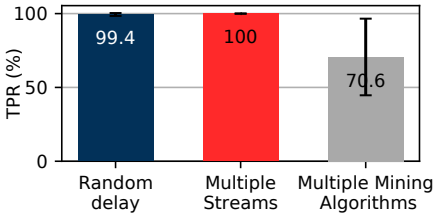

Figure 22: Figure depicts the effect on TPR due to targeted attacks.

demonstrating its resilience to binary obfuscation techniques. The reason for this is that although binary obfuscation transforms the mining binary, its physical characteristics are still retained, which can be captured by *MagTracer*.

6.5.2 Robustness against Throttling. In this experiment, we throttle the mining program's active time by adding inactive time slots (or pauses) periodically during the execution of the mining program. Here, throttling ratio is defined as the fraction of inactive time, i.e., the fraction of total time when the mining program is *not* executed. As depicted in Figure 21, we achieve TPRs above 81% for throttling ratios below 0.77. However, we regard this acceptable as high throttling ratios above 0.77 indicate low hash rates, which are not only unprofitable for the attacker but also consume fewer resources from the server, hence making them less concerning.

6.5.3 Robustness against Targeted Attacks. Finally, strong attackers may adopt targeted tactics to eliminate the correlation between the magnetic trace and the executed operations, thereby weakening the mining signature. Specifically, as depicted in Figure 22, we explore three anti-detection approaches that remote adversaries may leverage to invalidate *MagTracer*'s detection as elaborated below.

First, we **insert random delay** to make threads go out of synchronization, which is a common countermeasure against side-channels [42]. We evaluate this by performing 60 tests, where we sample a random delay from a uniform distribution between 0-1 milliseconds. However, in this case, *MagTracer* still achieves TPR over 99%. This is because the mining signature is caused by the assignment of large sets of threads, thereby unaffected by random delays of individual threads.

Second, we **issue multiple streams** to change the GPU control flow, and thereby diminish the mining signature. Here, a stream refers to a sequence of operations that execute in issue-order on GPUs [15]. Our results depict that *MagTracer* can detect mining events with high TPRs when the number of streams is varied from two to four. This good performance is because the asynchronous execution across streams does not affect the cyclic thread assignment within individual streams, thus preserving the mining signature.

Third, an attacker may **execute multiple mining algorithms simultaneously** to reduce the mining signature's SNR while still earning reasonable profits. To evaluate this, we run Ethash together with different mining algorithms (FiroPow, ProgPow, and KawPow). In such a scenario, the hash rates of the two concurrent mining algorithms reduce by half, and consequently, the TPR drops to 70.6%. While running more mining algorithms simultaneously may further mitigate the mining signature, it is increasingly unrealistic to carry out such attacks. This is because each mining algorithm needs to allocate a significant amount of GPU memory (around 4GB), making it impractical to run multiple cryptomining algorithms at the same time [20].

In summary, *MagTracer* maintains its effectiveness against obfuscation, throttling, and potential targeted evasion methods employed by strong attackers. This demonstrates the resilience and efficacy of *MagTracer* in the face of sophisticated evasion attempts.

7 DISCUSSION

We discuss future directions and alternatives to *MagTracer*.

Hardware Integration. Recall that *MagTracer*'s current setup (Figure 3) involves a variety of components including a magnetic sensor, ADC as well as a Raspberry Pi 4B. The total cost of the current prototype is around 40 USD. By integrating the 3 USD magnetometer with a low-cost microcontroller like the 1 USD RP2040 [63] onto a single printed circuit board, we can significantly reduce the overall cost and achieve a smaller form-factor.

Deployment Considerations. Although we currently evaluate *MagTracer* on a standalone host system with a single GPU, we envision *MagTracer* to be deployed in server rooms and data centers with machines equipped with several GPUs. We believe *MagTracer*'s approach can be extended to detecting several GPUs within the same host due to their physical proximity to each other. Standard server racks typically have a width of approximately 48.2 cm [5, 71]. Considering that the working range of *MagTracer* is approximately 24 cm, we believe that a single *MagTracer* unit strategically positioned at the center of the server can effectively monitor all GPUs

within it. Furthermore, with a miniaturized form-factor discussed earlier, we may be able to attach one magnetic sensor per machine and report potential cryptojacking events to the server administrator in real time.

Comparison with Other Side Channels. In addition to magnetic side channels, our preliminary experiments reveal that direct access to power traces from GPU power lines can provide signals with higher SNR than magnetic signals for cryptojacking detection. However, obtaining such traces involves cutting GPU power lines, making the solution less usable. While power traces can also be obtained from GPU's built-in power analyzer, this approach poses two concerns – 1) their sampling rates are typically low (about 100 Hz), hence making cryptojacking detection challenging [55, 72], and 2) since the readings would be obtained through the host's interface, attackers with strong software capabilities may alter them. In contrast, *MagTracer*'s detection is unaffected by attackers with strong software capabilities.

8 RELATED WORKS

We now present closely related work with *MagTracer*.

Cryptojacking Detection. Most prior works on cryptojacking detection focus on CPUs, and are primarily software-based defenses [3, 34, 41, 43, 53, 67, 73, 78]. The closest work to ours is by Gangwal *et al.*, where they leverage magnetic side-channel to detect CPU cryptojacking [29]. *MagTracer* 's unique contribution lies in its GPU hardware execution analysis, specifically in establishing the connection between cryptomining and GPU magnetic emanation by accurately identifying the *mining frequency*. This distinguishes our work from previous studies, including [29]. Furthermore, *MagTracer*'s technique ensures scalability across diverse GPU architectures and robust detection in the presence of interferences.

Electromagnetic Side-Channels. Prior works have leveraged electromagnetic (EM) leakage signals for extracting cryptographic keys, passwords, screen content, audio, and payment tokens, among others [9, 11, 12, 39, 45, 48, 70]. Several other works have also utilized EM signals for *defense*, in particular for malware and eavesdropping detection [10, 33, 58, 61]. Of these works, one particular work leverages GPU magnetic signals for inferring neural network architecture by leveraging the GPU synchronization points to identify network layers and activation function types [48]. Unlike their approach, *MagTracer* systematically analyses the magnetic signals during mining, in particular, the correspondence between the *rate of thread assignments* by GPU scheduler and the *mining frequency*, to ultimately perform cryptojacking detection.

9 CONCLUSION

We propose *MagTracer*, a novel GPU cryptojacking detection system, based on the distinct magnetic leakage signals emanated from GPUs while mining. We design and implement *MagTracer*, as well as perform a real-world evaluation on 14 heterogeneous GPU models, and achieve high true positive and low false positive rates. Our solution is both low-cost and a practical solution for detecting cryptojacking in servers. Through this work, we hope to inspire more research in the direction of leveraging non-invasive, proximate sensing for securing, as well as monitoring the health of highly valuable computing resources such as GPUs.

ACKNOWLEDGMENTS

We sincerely thank our anonymous reviewers and shepherd for their valuable feedback. This paper is partially supported by the National Key R&D Program of China (2021QY0703), National Natural Science Foundation of China (Grant No. U21A20462), Research Institute of Cyberspace Governance in Zhejiang University, Leading Innovative and Entrepreneur Team Introduction Program of Zhejiang (Grant No. 2018R010 05), "Pioneer"and "Leading Goose" R&D Program of Zhejiang (Grant No. 2023C01033), Basic Research Project of Shenzhen Science and Technology Innovation Commission (Project No. JCYJ20190812155213250), Shenzhen Longhua District Science and Technology innovation special fund project (Project No. JCYJ201903), the Institute of Information and Communications Technology Planning and Evaluation (IITP-2022-0-00420) grant funded by Ministry of Science and ICT (MSIT) in Korea, and Google PhD Fellowship 2021.

REFERENCES

[1] 1lastBr3ath. 2022. Dr. Mine. https://github.com/1lastBr3ath/drmine, accessed: 2022-05-02.

[2] Kristin P Bennett and Colin Campbell. 2000. Support vector machines: hype or hallelujah? *ACM SIGKDD explorations newsletter* 2, 2 (2000), 1–13.

[3] Weikang Bian, Wei Meng, and Mingxue Zhang. 2020. Minethrottle: Defending against wasm in-browser cryptojacking. In *The World Wide Web Conference (WWW)*. 3112–3118.

[4] Jeremiah Blocki, Ling Ren, and Samson Zhou. [n. d.]. Bandwidth-Hard Functions: Reductions and Lower Bounds. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS*, David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang (Eds.).

[5] Rack Blog. 2020. Server Rack Sizes: Understanding the Differences. https://www.racksolutions.com/news/blog/server-rack-sizes/.

[6] Bminer. 2022. Bminer - A faster Cryptocurrency miner runs on GPUs. https://www.bminer.me/, accessed: 2022-05-02.

[7] E Oran Brigham and RE Morrow. 1967. The fast Fourier transform. *IEEE spectrum* 4, 12 (1967), 63–70.

[8] Virus Bulletin. 2022. Cryptojacking on the Fly: TeamTNT Using NVIDIA Drivers to Mine Cryptocurrency. https://www.virusbulletin.com/virusbulletin/2022/04/cryptojacking-fly-teamtnt-using-nvidia-drivers-mine-cryptocurrency/.

[9] Giovanni Camurati, Sebastian Poeplau, Marius Muench, Tom Hayes, and Aurélien Francillon. 2018. Screaming Channels: When Electromagnetic Side Channels Meet Radio Transceivers. In *Proceedings of*

the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS. ACM, 163–177.

[10] Anadi Chaman, Jiaming Wang, Jiachen Sun, Haitham Hassanieh, and Romit Roy Choudhury. 2018. Ghostbuster: Detecting the Presence of Hidden Eavesdroppers. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking, MobiCom*. ACM, 337–351.

[11] Jieun Choi, Hae-Yong Yang, and Dong-Ho Cho. 2020. TEMPEST Comeback: A Realistic Audio Eavesdropping Threat on Mixed-signal SoCs. In *2020 ACM SIGSAC Conference on Computer and Communications Security, CCS*. ACM, 1085–1101.

[12] Myeongwon Choi, Sangeun Oh, Insu Kim, and Hyosu Kim. 2022. MagSnoop: listening to sounds induced by magnetic field fluctuations to infer mobile payment tokens. In *MobiSys '22: The 20th Annual International Conference on Mobile Systems, Applications and Services*. ACM, 409–421.

[13] CNET. 2023. Bitcoin Mining: How Much Electricity It Takes and Why People Are Worried. https://www.cnet.com/personal-finance/crypto/bitcoin-mining-how-much-electricity-it-takes-and-why-people-are-worried/, accessed: 2023-03-17.

[14] Greg Colvin, Andrea Lanfranchi, Michael Carter, and IfDefElse. 2022. ProgPoW, a Programmatic Proof-of-Work. https://eips.ethereum.org/EIPS/eip-1057, accessed: 2022-11-11.

[15] CUDA Toolkit Documentaion. 2022. Stream Management. https://docs.nvidia.com/cuda/cuda-runtime-api/group__CUDART__STREAM.html, accessed: 2022-05-02.

[16] Neelanjit Das. 2022. Cryptojacking Cases Are Rising Globally, Why So And Should This Worry You? https://www.outlookindia.com/business/cryptojacking-cases-are-rising-globally-why-so-and-should-this-worry-you--news-212990.

[17] National Vulnerability Database. 2022. CVE-2022-29799. https://nvd.nist.gov/vuln/detail/CVE-2022-29799.

[18] Debian. 2023. Security Infomation – DSA-5402-1 linux. https://www.debian.org/security/2023/dsa-5402.

[19] Digiconomist. 2022. Bitcoin Energy Consumption Index. https://digiconomist.net/bitcoin-energy-consumption.

[20] Ethereum. 2022. Ethash. https://eth.wiki/en/concepts/ethash/ethash/, accessed: 2022-05-02.

[21] Ethereum-mining. 2022. ethminer. https://github.com/ethereum-mining/ethminer/, accessed: 2022-05-02.

[22] EthereumClassic. 2022. Ethereum Classic. https://ethereumclassic.org/, accessed: 2022-11-11.

[23] EthereumClassic. 2022. Ethereum Classic Improvement Proposals. https://ecips.ethereumclassic.org/ECIPs/ecip-1049, accessed: 2022-11-11.

[24] EthereumFair. 2022. EthereumFair. https://etherfair.org/, accessed: 2022-11-11.

[25] EthereumPoW. 2022. EthereumPoW. https://ethereumpow.org/, accessed: 2022-11-11.

[26] Firo. 2022. Firo. https://firo.org/, accessed: 2022-11-11.

[27] Firo. 2022. How to ming Firo (FIRO) with FiroPow. https://firo.org/guide/how-to-mine-firo.html, accessed: 2022-11-11.

[28] Forbes. 2023. AI Startups Boom In San Francisco Amid $100 Billion Google Mistake. https://www.forbes.com/sites/martineparis/2023/02/23/ai-startups-boom-in-san-francisco-amid-100-billion-google-mistake/.

[29] Ankit Gangwal and Mauro Conti. 2020. Cryptomining Cannot Change Its Spots: Detecting Covert Cryptomining Using Magnetic Side-Channel. *IEEE Transactions on Information Forensics and Security* 15 (2020), 1630–1639.

[30] GMiner. 2022. GMiner. https://gminer.info/, accessed: 2022-05-02.

[31] Oscar Gonzalez. 2022. Bitcoin Mining: How Much Electricity It Takes and Why People Are Worried. https://www.cnet.com/personal-finance/crypto/bitcoin-mining-how-much-electricity-it-takes-and-why-people-are-worried/.

[32] Runchao Han, Nikos Foutris, and Christos Kotselidis. 2019. Demystifying Crypto-Mining: Analysis and Optimizations of Memory-Hard PoW Algorithms. In *IEEE International Symposium on Performance Analysis of Systems and Software, ISPASS*. IEEE, 22–33.

[33] Jiaji He, Xiaolong Guo, Haocheng Ma, Yanjiang Liu, Yiqiang Zhao, and Yier Jin. 2020. Runtime trust evaluation and hardware trojan detection using on-chip em sensors. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 1–6.

[34] Geng Hong, Zhemin Yang, Sen Yang, Lei Zhang, Yuhong Nan, Zhibo Zhang, Min Yang, Yuan Zhang, Zhiyun Qian, and Haixin Duan. 2018. How you get shot in the back: A systematical study about cryptojacking in the real world. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 1701–1713.

[35] hoshsadiq. 2022. NoCoin. https://github.com/hoshsadiq/adblock-nocoin-list.

[36] Texas Instrument. 2023. DRV425 - Fully-integrated fluxgate magnetic sensor for open-loop applications. https://www.ti.com/product/DRV425, accessed: 2023-02-12.

[37] Texas Instruments. 2022. ADS1263. https://www.ti.com/lit/ds/symlink/ads1263.pdf, accessed: 2022-05-02.

[38] John David Jackson. 1975. *Classical electrodynamics; 2nd ed.* Wiley, New York, NY.

[39] Wenqiang Jin, Srinivasan Murali, Huadi Zhu, and Ming Li. 2021. Periscope: A Keystroke Inference Attack Using Human Coupled Electromagnetic Emanations. In *2021 ACM SIGSAC Conference on Computer and Communications Security, CCS*. ACM, 700–714.

[40] Thorsten Joachims. 2001. *Estimating the generalization performance of a SVM efficiently*. Technical Report. Technical Report.

[41] Amin Kharraz, Zane Ma, Paul Murley, Charles Lever, Joshua Mason, Andrew Miller, Nikita Borisov, Manos Antonakakis, and Michael Bailey. 2019. Outguard: Detecting in-browser covert cryptocurrency mining in the wild. In *The World Wide Web Conference (WWW)*. 840–852.

[42] Paul Kocher, Joshua Jaffe, and Benjamin Jun. 1999. Differential power analysis. In *Annual international cryptology conference (CRYPTO)*. Springer, 388–397.

[43] Radhesh Krishnan Konoth, Emanuele Vineti, Veelasha Moonsamy, Martina Lindorfer, Christopher Kruegel, Herbert Bos, and Giovanni Vigna. 2018. Minesweeper: An in-depth look into drive-by cryptocurrency mining and its defense. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 1714–1730.

[44] Marc Kührer, Christian Rossow, and Thorsten Holz. 2014. Paint it black: Evaluating the effectiveness of malware blacklists. In *International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)*. Springer, 1–21.

[45] Zhuoran Liu, Niels Samwel, Leo Weissbart, Zhengyu Zhao, Dirk Lauret, Lejla Batina, and Martha A. Larson. 2020. Screen Gleaning: A Screen Reading TEMPEST Attack on Mobile Devices Exploiting an Electromagnetic Side Channel. *CoRR* abs/2011.09877 (2020).

[46] Magno Logan and Pawan Kinger. 2021. Linux Threat Report 2021 1H. https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/linux-threat-report-2021-1h-linux-threats-in-the-cloud-and-security-recommendations.

[47] lolMiner. 2022. lolMiner. https://lolminer.site/, accessed: 2022-05-02.

[48] Henrique Teles Maia, Chang Xiao, Dingzeyu Li, Eitan Grinspun, and Changxi Zheng. 2022. Can one hear the shape of a neural network?: Snooping the GPU via Magnetic Side Channel. In *31st USENIX Security Symposium, USENIX Security 2022*. USENIX Association, 4383–4400.

[49] Microsoft Security. 2020. Misconfigured Kubeflow workloads are a security risk. https://www.microsoft.com/security/blog/2020/06/10/misconfigured-kubeflow-workloads-are-a-security-risk/.

[50] Microsoft Security. 2021. New large-scale campaign targets Kubeflow. https://techcommunity.microsoft.com/t5/microsoft-defender-for-cloud/new-large-scale-campaign-targets-kubeflow/ba-p/2425750/.

[51] MUO. 2021. Why Cryptojacking Is Better Than Ransomware for Cybercriminals. https://www.makeuseof.com/why-cryptojacking-is-better-than-ransomware-for-cybercriminals/.

[52] Nanominer. 2022. Nanominer - best cryptocurrency miner! https://nanominer.org/, accessed: 2022-05-02.

[53] Faraz Naseem, Ahmet Aris, Leonardo Babun, Ege Tekiner, and A. Selcuk Uluagac. 2021. MINOS: A lightweight real-time cryptojacking detection system. In *28th Annual Network and Distributed System Security Symposium (NDSS)*.

[54] NVIDIA. 2016. GeForce GTX 1080 Ti. https://www.nvidia.com/en-gb/geforce/graphics-cards/geforce-gtx-1080-ti/specifications/, accessed: 2022-07-27.

[55] NVIDIA. 2022. NVIDIA Power Capture Analysis Tool. https://developer.nvidia.com/nvidia-power-capture-analysis-tool, accessed: 2022-11-07.

[56] Sergio Pastrana and Guillermo Suarez-Tangil. 2019. A First Look at the Crypto-Mining Malware Ecosystem: A Decade of Unrestricted Wealth. In *Proceedings of the Internet Measurement Conference (IMC)*. ACM, 73–86.

[57] Ronald K Pearson, Yrjö Neuvo, Jaakko Astola, and Moncef Gabbouj. 2016. Generalized hampel filters. *EURASIP Journal on Advances in Signal Processing* 2016, 1 (2016), 1–18.

[58] Duy-Phuc Pham, Damien Marion, Matthieu Mastio, and Annelie Heuser. 2021. Obfuscation Revealed: Leveraging Electromagnetic Signals for Obfuscated Malware Classification. In *ACSAC '21: Annual Computer Security Applications Conference*. ACM, 706–719.

[59] PhoenixMiner. 2022. PhoenixMiner - AMD+NVIDIA GPU Miner. https://phoenixminer.org/, accessed: 2022-05-02.

[60] Thanawin Rakthanmanon, Bilson Campana, Abdullah Mueen, Gustavo Batista, Brandon Westover, Qiang Zhu, Jesin Zakaria, and Eamonn Keogh. 2012. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 262–270.

[61] Soundarya Ramesh, Ghozali Suhariyanto Hadi, Sihun Yang, Mun Choon Chan, and Jun Han. 2022. TickTock: Detecting Microphone Status in Laptops Leveraging Electromagnetic Leakage of Clock Signals. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS*. ACM, 2475–2489.

[62] Raspberry Pi. 2022. Raspberry Pi 4. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/, accessed: 2022-05-02.

[63] Raspberry Pi. 2023. Raspberry Pi Pico. https://www.raspberrypi.com/products/raspberry-pi-pico/, accessed: 2023-03-08.

[64] Ravencoin. 2022. KAWPOW algorithm. https://ravencoin.org/about/, accessed: 2022-05-02.

[65] Ravencoin. 2022. Ravencoin. https://ravencoin.org/, accessed: 2022-05-02.

[66] Reuters. 2023. Chip giant Nvidia nears trillion-dollar status on AI bet. https://www.reuters.com/technology/nvidia-close-becoming-first-trillion-dollar-chip-firm-after-stellar-forecast-2023-05-25/.

[67] Juan D Parra Rodriguez and Joachim Posegga. 2018. Rapid: Resource and api-based detection against in-browser miners. In *Proceedings of the 34th Annual Computer Security Applications Conference (ACSAC)*. ACM, 313–326.

[68] Sero. 2022. Sero. https://sero.cash/en/, accessed: 2022-11-11.

[69] Slashgear. 2022. Why You Probably Shouldn't Buy A Used Graphics Card. https://www.slashgear.com/907810/why-you-probably-shouldnt-buy-a-used-graphics-card/.

[70] Yang Su, Daniel Genkin, Damith Chinthana Ranasinghe, and Yuval Yarom. 2017. USB Snooping Made Easy: Crosstalk Leakage Attacks on USB Hubs. In *26th USENIX Security Symposium, USENIX Security 2017*. USENIX Association, 1145–1161.

[71] Dell Technologies. 2022. Dell PowerEdge R750xa. https://i.dell.com/sites/csdocuments/Product_Docs/en/poweredge-R750xa-spec-sheet.pdf.

[72] TechPowerUp. 2022. GPU-Z. https://www.techpowerup.com/gpuz/, accessed: 2022-05-02.

[73] Ege Tekiner, Abbas Acar, and A. Selcuk Uluagac. 2022. A Lightweight IoT Cryptojacking Detection Mechanism in Heterogeneous Smart Home Networks. In *29th Annual Network and Distributed System Security Symposium (NDSS)*.

[74] The Enigma Protector. 2022. Enigma Protector: A professional system for executable files licensing and protection. https://enigmaprotector.com/, accessed: 2022-05-02.

[75] TREND. 2021. TeamTNT Upgrades Arsenal, Refines Focus on Kubernetes and GPU Environments. https://www.trendmicro.com/en_us/research/21/k/teamtnt-upgrades-arsenal-refines-focus-on-kubernetes-and-gpu-env.html.

[76] UPX. 2022. UPX packer. https://upx.github.io/, accessed: 2022-05-02.

[77] USAO. 2021. Two Iranian Nationals Indicted in Local Cryptojacking Case. https://www.justice.gov/usao-edmo/pr/two-iranian-nationals-indicted-local-cryptojacking-case/.

[78] Wenhao Wang, Benjamin Ferrell, Xiaoyang Xu, Kevin W Hamlen, and Shuang Hao. 2018. Seismic: Secure in-lined script monitors for interrupting cryptojacks. In *European Symposium on Research in Computer Security (ESORICS)*. Springer, 122–142.

[79] xd4rker. 2022. MinerBlock. https://github.com/xd4rker/MinerBlock, accessed:2022-05-02.

[80] Xilinx. 2023. Ethash Kernel. https://xilinx.github.io/blockchainacceleration/kernel_design.html, accessed: 2023-03-17.

[81] Zano. 2022. ProgPowZ. https://zano.org/downloads/zano_wp.pdf, accessed: 2022-11-11.

[82] Zano. 2022. Zano. https://zano.org/, accessed: 2022-11-11.

[83] ZeroDot1. 2022. CoinBlockerLists. https://zerodot1.gitlab.io/CoinBlockerListsWeb/, accessed: 2022-05-02.