Cryptojacking Attacks: Mechanisms, Detection, and Prevention

A Comprehensive Analytical Study in the Context of Modern Cybersecurity Threats

Piyush Vats
BCA , Sem-VI
Institute. of Information Technology & Management,
New Delhi, India
Enrol: 03913702023
Piyushvats469@gmail.com

Dr. Ramandeep Kaur
Dept. of Computer Applications
Inst. of Information Technology & Management,
New Delhi, India
ramandeep.kaur@iitmipu.ac.in

Abstract **-** Cryptojacking, the unauthorized hijacking of a victim's computing resources for the purpose of mining cryptocurrencies, has become one of the most prevalent and financially destructive categories of cyber threats. Unlike conventional cyber threats characterized by destructive intent, cryptojacking is a covert operation whose primary goal is the unauthorized exploitation of computing resources for an indefinite period of time. The attack surface spans web browsers, operating systems, containerized environments, and serverless computing. This paper presents a comprehensive analytical study covering the cryptocurrency mining mechanism, a structured taxonomy of attack vectors, a comparative analysis of ML-based and heuristic detection techniques, and a layered prevention framework. Notable incidents including Coinhive, the Smominru botnet, and the Tesla AWS hijacking are analyzed. Research gaps identified include the absence of ASIC-resistant IoT detection, adversarially robust ML for fileless attacks, and a coherent legal deterrence framework.

Index Terms *— Cryptojacking, Cryptocurrency Mining, Malware Detection, Browser Security, Monero, XMRig, Machine Learning, Endpoint Security, Threat Taxonomy, Cloud Security, Fileless Malware.*

I. INTRODUCTION

The economic model that cryptocurrencies follow is based on a mining paradigm — a distributed proof-of-work mechanism that requires participants to contribute computing power in order to validate transactions and be rewarded with new currency. For Bitcoin, Ethereum (pre-2022), Monero, and dozens of other cryptocurrencies, mining is the key that drives the consensus mechanism. It is a costly endeavor; mining rigs are capital-intensive devices that require a lot of electrical power to run.

This cost structure makes for a very enticing offer for bad actors. The cost of mining, including hardware and electricity, is shifted onto unsuspecting third parties, causing the entire revenue of the mining operation to be profit. This is exactly what cryptojacking is: unauthorized execution of mining code on unsuspecting third parties' computer systems. The Smominru botnet infected over half a million Windows servers, the Coinhive JavaScript cryptojacker infected tens of thousands of websites, and cloud systems of Tesla Motors and various government agencies have been compromised. Yet despite its prevalence, the academic literature comprehensively covering all facets of cryptojacking is still incomplete.

This paper fills that gap. The specific contributions are:

- A taxonomy of cryptojacking attack mechanisms based on a three-axis model of delivery, execution contexts, and persistence methods.
- A comparative analysis of detection methods including signature-based, behavioral, network-level, and machine-learning approaches.
- A prevention framework based on a multi-level architecture spanning browser, endpoint, network, and cloud layers.
- A critical identification of research gaps and open challenges that define the frontiers of cryptojacking defense.

The rest of this paper is organized as follows. Section II reviews literature. Section III presents the problem statement. Section IV identifies research gaps. Section V states research objectives. Section VI covers background. Section VII describes the attack taxonomy. Section VIII covers detection. Section IX covers prevention. Section X presents case studies. Section XI discusses open challenges. Section XII concludes.

II. LITERATURE REVIEW

A. Early Work on Browser-Based Cryptojacking

The study of browser-based cryptojacking started to gain traction when Coinhive came into the picture in September 2017. Eskandari et al. [1] performed the pioneering study on browser-based cryptomining, analyzing more than 2,700 websites via the Coinhive API, developing a taxonomy of script injection techniques, and proposing a static analysis technique using JavaScript abstract syntax trees (ASTs) that achieved a detection rate of 97.3%. Musch et al. [2] expanded the scope to the public Internet, crawling the top one million sites and detecting over 1,200

domains with active mining scripts, and reporting on the rise of WebAssembly (WASM) offering 2–3× the performance of JavaScript-based code.

B. Host-Based and Fileless Cryptojacking

Zimba et al. [3] provided a first taxonomy of host-based cryptojacking attacks, distinguishing binary-based from fileless approaches and emphasizing XMRig's modularity. Caprolu et al. [4] focused on fileless cryptojacking via PowerShell, WMI, and process injection, revealing near-zero AV detection rates for fileless variants and establishing that kernel-level behavioral monitoring is a necessity.

C. Machine Learning-Based Detection

Hong et al. [5] were first to use an SVM classifier on CPU utilization, memory access, and network flow metrics, achieving 94.1% accuracy but with high false positives on HPC workloads. Sayadi et al. [6] used CNN with hardware performance counters (HPCs), achieving 96.8% accuracy but requiring OS-level PMU access. Kharraz et al. [7] proposed Outguard, a decision tree using JS engine traces achieving 93.5% accuracy. Liu et al. [8] used GNN on execution graphs achieving 98.9%, the best reported accuracy, at the cost of high compute overhead for real-time inference.

D. Network-Level Detection and Cloud

Bijmans et al. [9] measured cryptojacking at internet scale, finding over 4,000 unique mining pool domains and demonstrating that DNS blocking is ineffective when attackers use direct IP connections or rapidly register new domains. Tahir et al. [10] found browser-level defenses like CPU throttling are easily evaded across iframes or web workers. Naseem et al. [12] proposed MINOS, a lightweight random-forest solution combining HPCs and network flows, achieving 97.3% accuracy with under 200 ms detection latency in cloud/edge environments.

E. Summary and Identified Gaps

The existing literature has established strong foundations in browser-based detection, ML classifier design, and network-level analysis. However, no study has provided a unified analytical framework spanning all attack categories. ML detection work has largely been conducted on static, non-adversarial test sets. IoT cryptojacking has received minimal attention, and legal dimensions are almost entirely absent from the technical literature.

Table I: Summary of Key Literature — Methods, Contributions, and Limitations

Reference	Year	Approach	Key Contribution	Limitation
Eskandari et al. [1]	2018	JS AST + Random Forest	First systematic browser CJ study; 97.3%	Static; evaded by obfuscation
Musch et al. [2]	2019	Web crawl + WASM analysis	Scale measurement; WASM characterization	Detection not proposed
Zimba et al. [3]	2019	Taxonomy + binary analysis	Host-based CJ taxonomy; XMRig analysis	No ML classifier proposed
Caprolu et al. [4]	2020	Behavioral analysis	Fileless CJ; AV evasion analysis	No detection system built
Hong et al. [5]	2018	SVM on telemetry	94.1% accuracy; system telemetry ML	High FP on HPC workloads
Sayadi et al. [6]	2019	CNN + HPC features	96.8% accuracy; hardware-based	Requires HW access; fails in VMs
Kharraz et al. [7]	2019	JS engine trace + DT	Outguard; 93.5% accuracy	Browser-only scope
Liu et al. [8]	2022	GNN on process graphs	98.9% accuracy; fileless detection	High computational overhead
Bijmans et al. [9]	2019	DNS + network analysis	Internet-scale measurement	DNS blocking easily evaded
Naseem et al. [12]	2021	RF ensemble (MINOS)	97.3%; real-time cloud detection	Limited adversarial evaluation

III. PROBLEM STATEMENT

In essence, cryptojacking is a very unbalanced attack, where the attacker spends very little in terms of resources, yet the victim loses a lot, especially in terms of system performance, power consumption, and wear and tear. In cloud environments, victims are also directly charged for the resources used in mining. Despite these obvious disadvantages, cryptojacking has managed to evade detection with impunity in most organizations.

The main technical challenge is behavioral ambiguity. Continuous heavy-duty floating-point operations for mining look identical to legitimate high-compute tasks such as video processing or ML inference. Sophisticated attackers deliberately throttle CPU usage to stay under detection thresholds. Browser-based variants leave no disk evidence; fileless variants produce no file artifacts; cloud-based attacks are indistinguishable from legitimate administrator activity. Each environment requires its own detection software, yet they must be integrated into a single coherent system. Detection latency is also critical — cloud mining accumulates charges of tens of dollars per hour, requiring sub-second detection. Finally, the institutional and legal dimensions are poorly addressed: endpoint telemetry is often insufficient, and attribution is hampered by Monero's privacy properties and the absence of harmonized legal frameworks.

IV. RESEARCH GAP

A. Lack of a Unified Cross-Environment Taxonomy

Currently, the taxonomies of cryptojacking attacks are limited in scope, covering either browser-based or host-based attacks. No prior work has attempted to uniformly classify browser-based, host-based, fileless, container, cloud, and IoT-based attacks under a single framework with a consistent set of dimensions.

B. Adversarial Robustness of ML Detection

Machine learning classifiers for cryptojacking detection have been tested only on static test sets. No study has assessed adversarial robustness against adaptive adversaries who change their mining behavior in response to the detection system.

C. IoT Coverage, Detection Latency, and Legal Gaps

Nearly no attention has been given to cryptojacking detection in IoT environments, where traditional EDR tooling is impractical due to resource constraints. The relationship between detection latency and economic damage in cloud environments has not been formally established. The legal aspects of cryptojacking, including its relationship with computer fraud laws and the requirements for prosecution, have received little to no attention, which contributes to the economic viability of the threat.

V. RESEARCH OBJECTIVES

Based on the problem statement and identified shortcomings, this paper aims to:

1. Develop a unified taxonomy of cryptojacking attack modalities across browser, host, fileless, container, cloud, and IoT environments using a three-dimensional framework: delivery vector, execution context, and persistence strategy.
2. Carry out a comparative analytical study of detection techniques — signature-based, heuristic, network-based, and ML-based — with respect to accuracy, false positives, detection latency, and evasion resistance.
3. Develop a multi-layered defense-in-depth prevention framework with specific controls for browser, endpoint, network, and cloud environments.
4. Identify the most important open research challenges in adversarial robustness, IoT applicability, detection latency, and legal aspects.
5. Derive generalizable lessons from the Coinhive, Smominru, and Tesla AWS case studies to inform defensive architecture and incident response playbooks.

VI. BACKGROUND: CRYPTOCURRENCY MINING AND CRYPTOJACKING ECONOMICS

A. Proof-of-Work Mining Mechanics

Mining cryptocurrencies can be viewed as a distributed game of validation, wherein miners use their genuine computing resources to generate valid block hashes. The hash is valid only if its corresponding numeric value is lower than a pre-defined network-wide threshold known as the difficulty. The odds of successfully generating a valid hash are extremely low for any single attempt, making the whole system resemble a lottery, with more computing power translating to more lottery tickets within a given time frame. The Bitcoin network uses SHA-256, which has been so optimized that only ASICs can mine it economically, making it unworthy as a cryptojacking target. Monero uses the RandomX hash, which is memory-hard and ASIC-resistant, making hijacked CPUs a viable and profitable mining resource.

Table II: Cryptocurrency Hashing Algorithms and Cryptojacking Suitability

Currency	Algorithm	Optimal HW	ASIC Resistant	CJ Target Priority
Monero (XMR)	RandomX	CPU	Yes	Very High — primary CJ target
Bitcoin (BTC)	SHA-256	ASIC	No	Negligible — CPU uneconomical
Litecoin (LTC)	Scrypt	ASIC/GPU	Partial	Low — ASIC-dominated
Ethereum (ETH)	Ethash → PoS	GPU (historical)	Partial	Low — transitioned to PoS 2022
Zcash (ZEC)	Equihash	GPU	Partial	Moderate — GPU campaigns viable
Ravencoin (RVN)	KawPow	GPU	Yes (GPU)	Moderate — GPU attacks documented

B. Economics of Cryptojacking

The economics of cryptojacking are quite simple: Monero mining on a modern CPU makes a relatively small profit, certainly less than a dollar a day given current exchange rates. However, after establishing their infection pipeline, adding a machine to their botnet is nearly costless for the attacker. Consider a botnet of 100,000 infected devices, each making about 0.05 USD a day. This translates into a profit of 5,000 USD a day, or about 1.8 million USD a year, with minimal ongoing expenses. The Smominru botnet's documented profit of approximately 3.6 million USD in Monero over nine months is consistent with these figures. Cryptojacking is a completely hands-off, completely silent profit generator for its authors, which makes it a relatively safe and scalable way for criminals to generate revenue.

VII. TAXONOMY OF CRYPTOJACKING ATTACK MECHANISMS

A. Three-Axis Classification Framework

The different cryptojacking attacks can be described using a three-dimensional (3D) approach, which we can use to classify the attacks. The first is how the mining program enters the victim's system - the delivery method, for example, phishing, download or vulnerability.The second axis is where the mining program is executed - the execution context. This is where the mining program runs, such as on the host, in a browser or in a virtual machine.

The third axis is how the cryptojacking program continues to run even if it is terminated, for example, due to a system reboot - the persistence mechanism. This could be tasks, registry entries or services. In conclusion, this three-dimensional model offers a way of describing cryptojacking attacks, and covers all the major categories.

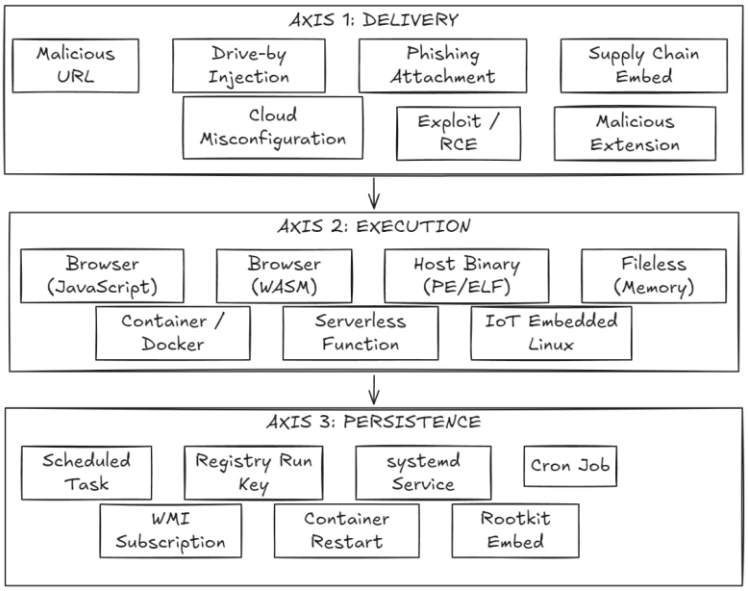

Fig. 1. Three-axis taxonomy of cryptojacking attack delivery, execution, and persistence.

B. Browser-Based Cryptojacking

This type of cryptojacking utilizes mining code that runs directly inside the victim's web browser, implemented as JavaScript or WebAssembly code that is executed simply by loading a compromised or malicious webpage. No code is written to disk, no installation is required — just loading the webpage initiates and terminates the process. The latest versions employ evasion methods including CPU throttling to 40–60 percent, mining only when the page is in the foreground, delayed start to evade short-visit scanners, and use of WebAssembly for its resistance to text-based scanning and higher execution performance.

C. Host-Based Cryptojacking

This type of cryptojacking drops a native mining binary directly onto the victim's operating system, where it is executed as a customized version of an open-source mining tool named XMRig. Attack vectors include phishing, zero-day exploits, brute-force SSH/RDP attacks, and supply chain attacks. The XMRig mining binary is open-source, making it very easy for adversaries to customize its code, add their own pool address, wallet ID, remove debug information, and even add competition-killer functionality that terminates other miners on the system.

D. Fileless Cryptojacking

Fileless cryptojacking is arguably the most technically advanced threat in this category. The mining software integrates directly into the memory space of a legitimate running process, with no file on disk for traditional antivirus software to detect. In Windows environments, common techniques include reflective DLL injection, process hollowing, and exploitation of PowerShell, WMI, and Windows Script Host. In the absence of disk-based evidence, incident responders are forced to rely exclusively on memory forensics. A simple system reboot makes the infection appear to have resolved, with reinfection via a persistent dropper occurring on the very next system start.

E. Cloud and Container Cryptojacking

In cloud environments, cryptojacking attacks target the platform's programmatic management interfaces to schedule mining tasks on a victim's own cloud compute resources. Misconfigured cloud environments are the most common entry vector, with exposed Kubernetes dashboards without authentication, IAM roles with overly privileged permissions, publicly accessible Docker daemon sockets, and S3 buckets containing credential files being documented as initial vectors. Once inside, attackers schedule containerized mining tasks via legitimate container orchestration APIs, giving these attacks a superficial appearance of legitimate system administrator activity.

Table III: Cryptojacking Attack Categories — Comparative Technical Profile

Category	Delivery	Persistence	Disk Footprint	Detection Difficulty	Hash Rate
Browser (JS)	Webpage visit	None (session)	None	Moderate	Low
Browser (WASM)	Webpage visit	None (session)	None	High	Low–Med
Host Binary	Phishing/exploit	High	Yes (.exe/.elf)	Moderate	High
Fileless	Memory injection	Moderate	None	Very High	Medium
Container/Cloud	API misconfiguration	High	Image only	High	Very High
IoT/Embedded	Default creds/exploit	Moderate	Yes (small)	Very High	Low–Med
Supply Chain	Trojanized package	High	Yes (package)	Very High	Variable

VIII. DETECTION METHODOLOGIES

A. Signature-Based Detection

Signature-based detection relies on matching known patterns — binary hash values, byte sequences, script strings, or domain names — against inspected content. Its main advantages are computational efficiency and a near-zero false positive rate for known threats. The main drawback is its reactive architecture: it cannot detect novel variants, polymorphic miners, or scripts created after the last database update. The open-source availability of mining tools makes variant generation trivially easy — a recompilation with modified strings or an obfuscation pass produces a binary that evades all existing hash signatures.

B. Behavioral and Heuristic Detection

Behavioral detection analyzes the way programs behave in real-time to identify mining activities. The strongest indication is a consistent surge in CPU utilization. However, modern cryptojacking tools have rate-limiting mechanisms that keep CPU utilization below detection thresholds. More reliable indicators include RandomX memory allocation patterns, unusual thread creation rates, thermal anomalies, and deviant process trees. On Windows, WMI subscription creations, unusual scheduled tasks, and PowerShell script executions are relevant. On Linux, unusual cron entries, new systemd unit files, and shared library injection attempts are key signals.

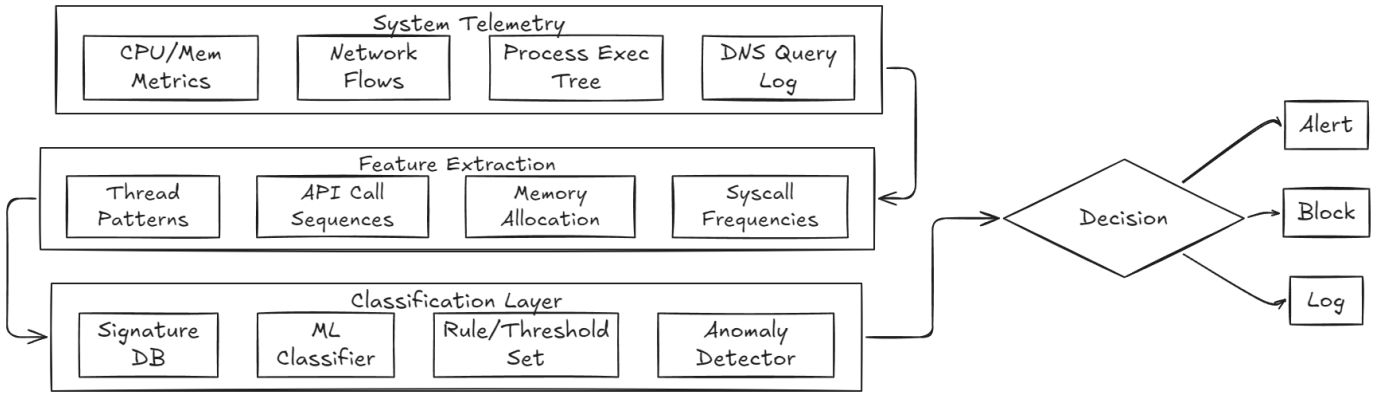

Fig. 2. Behavioral detection pipeline architecture for cryptojacking identification.

C. Network Traffic Analysis

The Stratum protocol is the de facto protocol that miners use to communicate with pools, built on TCP with a simple JSON-RPC message exchange model. Deep packet inspection can identify Stratum traffic by message patterns even over non-standard ports or TLS. DNS-based detection provides an additional layer: configuring enterprise DNS resolvers to block mining pool domains disables the communication channel for job assignment and share submission. However, as demonstrated by Bijmans et al., miners can evade DNS blocking by rapidly switching pool domains or using direct IP connections.

D. Machine Learning Detection — Comparative Analysis

Table IV: Comparative Analysis of Machine Learning Cryptojacking Detection Approaches

Study	Algorithm	Feature Set	Accuracy	FP Rate	Scope	Key Limitation
Eskandari et al. [1]	Random Forest	JS AST features	97.3%	1.2%	Browser (JS)	Static; obfuscation evades
Hong et al. [5]	SVM	CPU/net telemetry	94.1%	5.8%	Host	High FP on HPC workloads
Sayadi et al. [6]	CNN	HW perf. counters	96.8%	2.1%	Host	Requires HW PMU access
Kharraz et al. [7]	Decision Tree	JS engine trace	93.5%	1.8%	Browser	Browser scope only
Tanana [13]	LSTM	Network flow seqs	95.6%	3.2%	Network	High latency detection
Naseem et al. [12]	Random Forest	HPC + network flows	97.3%	2.1%	Cloud/Edge	Limited adversarial eval
Liu et al. [8]	GNN	Process exec. graph	98.9%	0.9%	Host + fileless	High compute overhead

E. Hybrid Detection Architecture

In today's fast and efficient detection systems, multiple detection methods are used in a layered approach to maintain accuracy. Typically, a quick and efficient filter like signature scanning to detect known attacks with low overhead is applied first. When an item is overlooked or is suspicious, heuristics provide a second line of defense by monitoring behaviour and anomalies. Lastly, machine learning inference is used to further assess the outcomes of the first two stages for more sophisticated or novel attacks.This approach helps keep the aggregate system costs low, but also allows for comprehensive detection of a variety of attacks, which makes the system scalable and able to operate reliably in production environments.

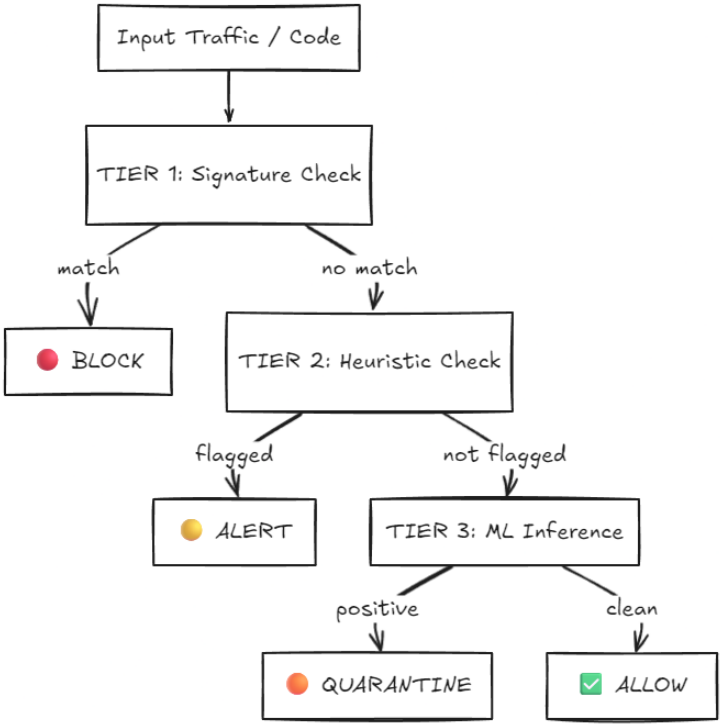

Fig. 3. Tiered hybrid detection pipeline minimizing overhead while maximizing coverage

.

IX. PREVENTION AND MITIGATION STRATEGIES

A. Layered Defense-in-Depth Framework

There is no one measure that can completely eliminate cryptojacking risks. Rather, a layered security strategy is required to address cryptojacking attacks. All layers where a cryptojacking attack can play out - the browser, endpoint, network and cloud - must employ relevant and effective control measures. For instance, browser protections can prevent the execution of malicious scripts, endpoint security can prevent the execution of processes, network security can detect irregular network traffic, and cloud environment security can prevent the abuse of computer resources. Users and timely software patching and updates also contribute to enhancing control. The risk is mitigated if one layer is not covered or is breached, as the other layers can provide the needed protection.

This approach enhances defense capabilities, reduces the impact and protects against a wide range of attacks including advanced methods.

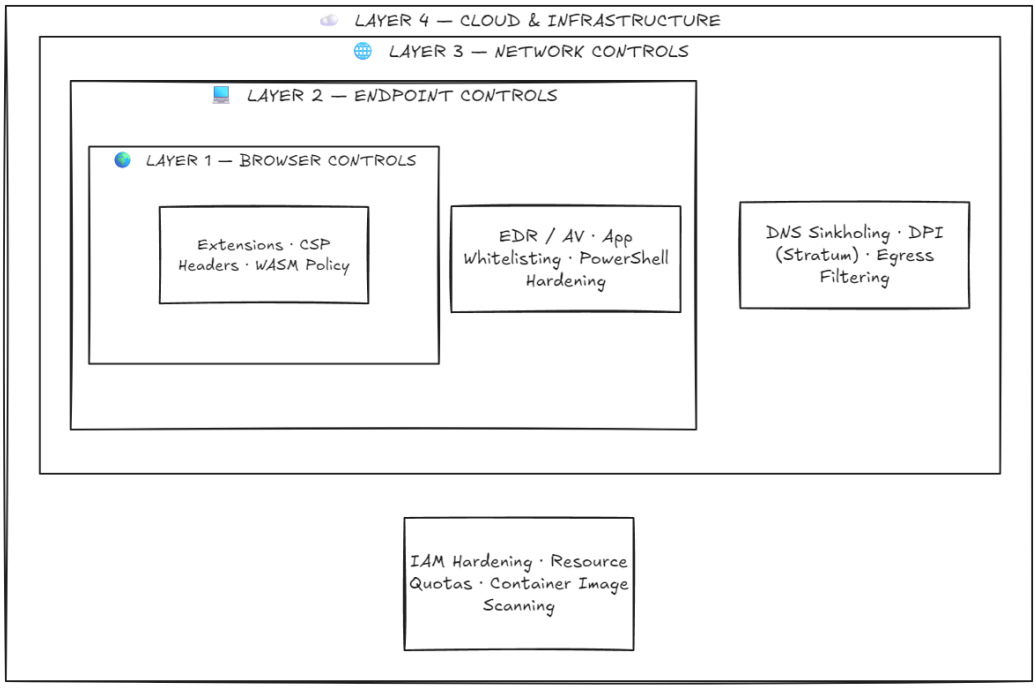

Fig. 4. Layered defense-in-depth architecture for cryptojacking prevention.

B. Browser-Level Controls

The browser-based cryptojacking exploits the transient or session-based nature of our interactions on the internet. To prevent this, users rely on browser extensions like NoCoin and MinerBlock, along with ad blockers like uBlock Origin with mining filter lists. Content Security Policy headers define which scripts are allowed from external sources, providing a protective barrier against mining scripts. Browser vendors are also working on solutions like Firefox 67's cryptomining blocker and Chrome's Site Isolation feature that helps control cross-tab resource usage.

C. Endpoint and Network-Level Controls

Endpoint Detection and Response (EDR) solutions continuously monitor activity across processes, files, networks, and registry items. Application whitelisting will stop any unauthorized binaries from executing, making it one of the most effective measures for preventing host-based cryptojacking, although it requires significant administration effort. PowerShell hardening, AMSI, and script block logging address fileless attacks. DNS sinkholing and filtering provide organization-wide protection against pool-communicating miners. Egress firewall rules blocking Stratum ports 3333, 4444, 5555, and 14444 add a network-layer enforcement point that complements DNS filtering.

Table V: Prevention Controls — Effectiveness Assessment by Deployment Layer

Control	Layer	Threat Coverage	Deploy Complexity	Evasion Resistance
Browser mining extensions	Browser	Browser-based only	Very Low	Moderate
Content Security Policy	Browser	Browser-based only	Low	High
Application Whitelisting	Endpoint	Host-based, fileless	High	Very High
EDR Behavioral Monitoring	Endpoint	Host-based, fileless	Medium	High
PowerShell Hardening	Endpoint	Fileless (Windows)	Low	High
DNS Sinkholing	Network	All pool-communicating	Low	Moderate

Egress Firewall (ports)	Network	All (known ports)	Medium	Moderate
IDS/IPS (Stratum sigs)	Network	Pool-communicating	Medium	Moderate
Container Image Scanning	Cloud/Infra	Container-based	Low	High
IAM Policy Hardening	Cloud/Infra	Cloud resource abuse	Medium	Very High
Resource Quotas (Cloud)	Cloud/Infra	Cloud resource abuse	Low	High

X. CASE STUDIES

A. The Coinhive Ecosystem (2017–2019)

Coinhive ushered in the age of browser-based cryptojacking in September 2017 by allowing attackers to add a JavaScript API to their sites that would mine Monero in the visitor's browser instead of relying on advertising revenue. Within weeks, researchers detected thousands of sites using Coinhive against visitors. High-profile targets included Showtime's site and in February 2018, a supply chain attack against the Browsealoud accessibility plugin injected cryptojacking code onto hundreds of government sites in the UK, US, and Australia, including the UK Information Commissioner's Office. Coinhive ceased operations in March 2019 due to the falling value of Monero, demonstrating the direct relationship between cryptocurrency valuations and cryptojacking campaign intensity.

B. Smominru Botnet (2017–2019)

The Smominru botnet was a huge case of host-based cryptojacking, leveraging the EternalBlue exploit to infect around 526,000 Windows servers worldwide. The malware mined Monero using XMRig variants, garnering around 9,000 XMR (approximately $3.6 million) over nine months. While law enforcement sought to take it down, the malware kept coming back through its domain generation algorithm, which helped it recreate its command structure even when individual pool domains were sinkholed. The malware also carried credential-stealing malware, indicating that cryptojacking is just one part of a larger criminal model.

C. Tesla AWS Cryptojacking Incident (2018)

In February 2018, RedLock reported that Tesla's AWS infrastructure had been compromised for cryptojacking. The attack vector was a misconfigured Kubernetes admin dashboard left accessible without authentication on the Internet — a configuration error, not a software vulnerability. Once in, they launched mining containers into Tesla's cloud infrastructure. The attackers employed smart evasion techniques: mining traffic was sent via a custom proxy instead of a mining pool, further obfuscated with Cloudflare, and CPU usage was kept just under Tesla's detection threshold. This attack became a model for cloud-based cryptojacking and highlights how critical configuration hygiene is for cloud security.

D. Implications for Defense Architecture

From these three case studies, some common factors can be identified. First and foremost, the importance of supply chain integrity cannot be overstated — the Browsealoud exploit and Smominru's EternalBlue delivery both highlight the importance of third-party dependencies as high-leverage attack vectors. Second, misconfiguration in the cloud is just as effective as exploiting a software vulnerability. Finally, sophisticated cryptojacking attacks seek to evade all expected detection mechanisms — domain-based detection, resource utilization thresholds, and signature-based tools are all specifically addressed in the Tesla attack's evasion profile.

XI. OPEN CHALLENGES AND FUTURE RESEARCH DIRECTIONS

A. Adversarial Robustness of Detection Systems

Studies have proven that malware classifiers designed using machine learning are prone to evasion attacks. In the case of cryptojacking, detectors based on CPU behavior, network flow, or program execution graphs have not been extensively validated against adversaries who design their mining behavior to evade the detector. Developing robust detectors against adaptive adversaries is an important open problem.

B. IoT, Legal, and Federated Detection

Detection tools created for enterprise endpoints do not fit well in the world of IoT. Lightweight detection techniques leveraging hardware performance counters, network edge analysis, and cloud-based behavioral profiling need to be developed for ARM and MIPS-based devices. The negligible number of cryptojacking operators prosecuted reflects the difficulty of tracking those responsible when Monero's privacy-enhancing properties are involved, as well as the lack of a unified international legal framework. Finally, the federated learning paradigm offers a promising solution for keeping detection models updated across organizations without sharing raw data, though robust aggregation mechanisms resistant to model poisoning remain an open problem.

XII. CONCLUSION

From its rough and tumble origins as a nascent trick for web browsers, cryptojacking has evolved into a multifaceted and lucrative threat that crosses over multiple

environments and vectors. The key characteristic that differentiates cryptojacking from other forms of malware is its focus on persistence and the generation of passive revenue streams. This paper has presented a holistic model for analyzing the cryptojacking threat from all angles: a three-axis attack taxonomy, a comparative analysis of detection mechanisms demonstrating that no single approach is viable, a layered defense-in-depth prevention model, and broad lessons from real-world attacks. The near-term focus should be to advance adversarially robust detection, lightweight IoT mechanisms, federated detection, and the integration of legal and technical perspectives — areas critical in shifting the balance in favor of organizations and against cryptojacking operators.

REFERENCES

[1] S. Eskandari, A. Leoutsarakos, T. Mursch, and J. Clark, "A first look at browser-based cryptojacking," in Proc. IEEE European Symp. Security and Privacy Workshops, London, UK, 2018, pp. 58–66.

[2] M. Musch, C. Wressnegger, M. Johns, and K. Rieck, "New kid on the web: A study on the prevalence of WebAssembly in the wild," in Proc. DIMVA, 2019, pp. 23–42.

[3] A. Zimba, Z. Wang, and H. Chen, "Multi-stage crypto ransomware attacks: A new emerging cyber threat to critical infrastructure and industrial control systems," ICT Express, vol. 4, no. 1, pp. 14–18, 2018.

[4] M. Caprolu, R. Di Pietro, F. Lombardi, and S. Raponi, "Cryptomining makes noise: Detecting cryptojacking via machine learning," Computer Communications, vol. 171, pp. 126–139, 2021.

[5] G. Hong et al., "How you get shot in the back: A systematical study about cryptojacking in the real world," in Proc. ACM CCS, Toronto, ON, Canada, 2018, pp. 1701–1713.

[6] H. Sayadi, N. Nguyen, A. Sasan, S. Rafatirad, and H. Homayoun, "Comprehensive assessment of run-time hardware-supported malware detection using general and ensemble learning," in Proc. ACM GLSVLSI, 2019, pp. 149–154.

[7] A. Kharraz et al., "Outguard: Detecting in-browser covert cryptocurrency mining in the wild," in Proc. World Wide Web Conf., San Francisco, CA, USA, 2019, pp. 840–852.

[8] Z. Liu, N. Zhang, B. Li, and L. Liu, "MineThrottle: Defending against wasm in-browser cryptojacking," in Proc. Web Conf., Ljubljana, Slovenia, 2020, pp. 3112–3118.

[9] H. Bijmans, T. Booij, and C. Doerr, "Inadvertently making cyber criminals rich: A comprehensive study of cryptojacking campaigns at internet scale," in Proc. 28th USENIX Security Symp., Santa Clara, CA, USA, 2019, pp. 1627–1644.

[10] R. Tahir, M. Durrani, F. Ahmed, H. Saeed, F. Zaffar, and S. Ilyas, "The browsers strike back: Countering cryptojacking and parasitic miners on the web," in Proc. IEEE INFOCOM, Paris, France, 2019, pp. 703–711.

[11] Y. Xin et al., "Machine learning and deep learning methods for cybersecurity," IEEE Access, vol. 6, pp. 35 365–35 381, 2018.

[12] M. Naseem, A. Aris, L. Babun, E. Tekiner, and A. S. Uluagac, "MINOS: A lightweight real-time cryptojacking detection system," in Proc. NDSS, 2021.

[13] M. Tanana, "Behavior-based detection of cryptojacking malware," in Proc. USBEREIT, 2020, pp. 0543–0545.

[14] P. Papadopoulos et al., "Master of web puppets: Abusing web browsers for persistent and stealthy computation," in Proc. NDSS, San Diego, CA, USA, 2019.

[15] E. Tekiner, A. Acar, A. S. Uluagac, E. Kirda, and A. A. Selcuk, "In-browser cryptomining for good: An analysis of cryptocurrencies viability," in Proc. BRAINS, 2021, pp. 149–156.

[16] F. Liu, Y. Li, J. Xu, and X. Liu, "Cryptomining attack detection in cloud environment using machine learning," IEEE Access, vol. 9, pp. 158 460–158 474, 2021.

[17] RedLock CSI Team, "Tesla cloud resources are hacked to run cryptocurrency-mining malware," RedLock Inc., Security Report, Feb. 2018.

[18] Proofpoint Threat Research, "Smominru monero mining botnet making millions for operators," Proofpoint Inc., Tech. Rep., Jan. 2018.

[19] CrowdStrike, "2023 Global Threat Report," CrowdStrike Holdings Inc., Austin, TX, USA, 2023.

[20] T. Lanet, R. Cayre, G. Auriol, M. Mushtaq, and H. Lanet, "Obfuscation techniques for cryptojacking malware," in Proc. ARES, 2020.