SPECTRUM OF ENGINEERING SCIENCES
ISSN (E) 3007-3138 (P) 3007-312X VOLUME 4, ISSUE 5, 2026

DETECTION OF CRYPTOJACKING THREATS BY USING MACHINE LEARNING WITHIN CLOUD SERVER ENVIRONMENTS

Imtiaz Ali Khoso[*1], Mumtaz Ali Khoso[2], Asif Ali[3], Muhammad Mureed[4], Muhammad Hamza Subhpoto[5], Abdul Samad Khaskheli[6]

[*1,3,4]*Department Information Technology Centre, Faculty of Agricultural Social Sciences, Sindh Agriculture University, Tandojam, Pakistan*
[2,5,6]*Department of Agricultural Economics, Faculty of Agricultural Social Sciences, Sindh Agriculture University, Tandojam, Pakistan*

[*1]imtiazali78699@gmail.com, [2]mzkmumtaz@gmail.com, [3]asifaliabbasi118@gmail.com, [4]muhammadmureed02@gmail.com, [5]muhammadhamzasp2@gmail.com, [6]asamad3454255@gmail.com

DOI: https://doi.org/10.5281/zenodo.20483878

Keywords
Machine Learning, Cryptojacking, Cyber Threat Intelligence, Cloud Security, Intrusion Detection, Random Forest.

Article History
Received: 03 April 2026
Accepted: 15 May 2026
Published: 30 May 2026

Corresponding Author: *
Imtiaz Ali Khoso

Abstract
Cryptojacking is a form of unauthorized attack where a hacker covertly uses the computational capabilities of a target system to mine cryptocurrencies. These attacks are usually targeted at cloud servers where downloaded malicious code uses system resources to solve cryptographic computations on their behalf. After being compromised, the infrastructure of the victim is used without their knowledge to mine on behalf of the attacker. This research paper proposes a solution that can identify the existence and magnitude of cryptojacking behaviour that takes place across cloud server networks. A machine learning classification environment is framed to address the problem of detecting such attacks. Multiple classifiers were experimented with, including SGD, Decision Tree, Random Forest, K Nearest Neighbors, MLP, Naive Bayes, Logistic Regression, Bagging variants and ADA Boost. The performance of all classification models was tested using Cryptojacking dataset. The outcomes of the experiments demonstrate that the maximum of the detection rate of experiment in the proposed methodology (Random Forest classifier) was 99.87 which confirms the utility of the provided methodology.

1. INTRODUCTION

1.1 Block Chain Technology

A blockchain technology is an open and distributed registry system that allows recording transactions, which are secure and transparent in a network of computers connected to each other. The specified technology can be regarded as a form of distributed ledger technology (DLT), since it implies utilizing cryptographic algorithms to guarantee data integrity and safety. Decentralization, consensus mechanism and smart contract are several key elements of the blockchain technology that are unique compared to the traditional centralized system and enable to perform peer to peer transactions without the middlemen and do not compromise security (Zheng et al., 2022). These qualities are: Decentralization It is the uniqueness of blockchain technology that it is not a conventional centralized system where one of the authorities,

e.g., a bank or a governmental entity, assumes the responsibility of maintaining and controlling the registry. On the other hand, blockchain is launched on the basis of a system of interconnected computers, or nodes that authenticate and document the transactions. Risks arising from centralized control points can be addressed through decentralization, thereby strengthening the reliability and validity of the entire system.

A distributed ledger is a database that is managed and maintained by various members of a network. All these participants are referred to as nodes and possess a complete copy of the entire ledger. A process employed to update these replicas at the same time is called a consensus mechanism. Such system is decentralized in nature and all the participants have equal access to the information hence transparency and removal of data manipulation is realized. The blockchain technology has cryptography, which guarantees security of transacting and controlling the release of new units of digital resources, including cryptocurrency. Each block has a distinct digital signature of its predecessor, forming an unbroken and insecure serial chain. Consensus Mechanism: The members of the network are expected to arrive at a consensus with the aim of having a common opinion as to what is contained in the ledger. Various agreement protocols exist, including Bitcoin's energy intensive mining-based model and stake-based alternatives, which are expected to ensure that a significant majority of the parties have verified and accepted transactions before being included in the blockchain Smart contracts are agreements between parties, encoding the terms directly as executable computer code and thus self executing (Hong et al., 2018). The expressions are automatically used in the fulfillment of pre set requirements and this saves the use of middlemen and increases efficiency in diverse applications.

Bitcoin (BTC) was launched in 2009 and blockchain technology was observed but today its application has outgrown the cryptocurrency world. The current application and research of this technology spanned numerous sectors such as financial services, logistics and medical industries among others. This is largely driven by its perceived capability to trigger improved data governance, transactional integrity and operational efficiency.

1.2 Crypto mining

As block chain technology advances and digital currency value rises, malicious actors are increasingly deploying cryptojacking malware across browsers, cloud servers and IoT devices to secretly mine cryptocurrencies without user consent (Tekiner et al., 2021). The majority of cryptojackers, as detailed in research by Tencent Security on the topic, get access to a network by taking advantage of software flaws or (weak login passwords). The assailants would proceed to install the crypto mining malware on the hacked system. In case of a web server as the target system, the crypto mining script might be included in the web page. The crypto mining script will silently run in the background of the browser of an average user whenever he/she opens this page. It is crucial to determine if the targeted website has crypto mining activity in order to prevent the misuse of users' computing resources by attackers. There have been two main categories of previous solutions to the cryptojacking detection challenge, the static method and the dynamic way. The first of them examines the JavaScript in the webpage through static analysis. The static approach can recognize the crypto mining website since it is able to extract the unique characteristics of the script that make it stand out. However, the static method can only identify websites with known cryptojacking scripts by analyzing JavaScript classifiers, which limits its detection scope (Aponte Novoa et al., 2022). In contrast to static methods, dynamic ones must first dynamically run the page in order to obtain its runtime behaviour. These techniques have the potential to effectively identify masked and encrypted cryptojacking through analysis of its dynamic properties. However, the current dynamic approaches still have three major drawbacks: 1) a significant performance penalty is introduced by some approaches; 2) several approaches require hooking the web browser, which may limit their widespread adoption; and 3) some approaches based on the

Hardware Performance Counter (HPC) are not so reliable in certain cases.

Cryptojacking malware has been discovered not only on websites but also on highly secure government and military servers. During a bug bounty competition, the United States Department of Defence's servers were found to be infected with cryptojacking malware. Coin hive, a well-known service provider, developed the cryptojacking virus that was discovered on servers and used it to generate 35.4% of a Monero coin. Similarly, the Russian Nuclear Weapons Research Centre brought up another government case. Several researchers at this university were taken into custody after they were found to have installed cryptocurrency miners on the institution's servers. In addition to utilizing the service providers' scripts, attackers have been known to modify non malicious, lawful, open-source crypto miners. The corporate network of an Italian bank, for instance, was discovered to have sent anomalous data to a well known European based botnet by a cybersecurity firm. After more research, the Bitcoin mining capability of this virus was confirmed.

1.3 Machine Learning

Machine learning (ML) is a branch of artificial intelligence that tries to design systems that learn through self improvement in response to data, without specific programming. The primary objective of ML is to design models capable of making predictions or facilitating decision making processes using input data.

Machine learning is a significant component of cybersecurity because it provides intelligent solutions, which are automated in detecting and mitigating cyber threat. Machine learning offers different algorithms that enhances the performance of a system by using data. Machine learning, particularly deep learning approaches, offers powerful solutions for host-based cryptojacking malware detection by learning behavioural patterns from system data (Sanda et al., 2023); Firstly, the model is trained using available labelled training samples. Secondly, by using that trained data start classifying that unseen data. With training data, Machine learning generates an algorithm to improve classification output.

Standard evaluation metrics including accuracy, precision, recall and F1 score are employed to assess classifier effectiveness. The paper also examines the challenges and drawbacks of the ML based malware detection like the adversarial evasion measures employed by the advanced malware to avoid detection systems.

The machine learning as a tool to detect malware needs a model to be trained on features, capable of distinguishing between malicious and benign software. The development process of an ML driven malware detection system is summarized below. The overall effectiveness of such a system is largely determined by the quality and diversity of the data used during training and the chosen features and the strength of the chosen model. Besides, none of the systems can be regarded as foolproof and it is important to be on alert and update the model periodically in order to react to the new and emerging threats.

1.4 Problem Statement

As a form of resource hijacking, cryptojacking involves the covert execution of mining software on a victim's machine, generating financial gains for the attacker without any awareness from the target. Emerging prominently around 2017, it rapidly escalated into one of the most significant threats facing network security globally. Similarly, most existing ML based botnet detection systems are constrained by the specific datasets used during training, making them less effective when applied to different data distributions due to the evolving diversity of attack patterns (Sanda et al., 2023). Because cryptojacking operates silently in the background, a sudden and unexplained slowdown during web browsing may indicate an active attack. Its highly concealed nature contributes to millions of incidents being reported globally every year. As of 2022, the world has experienced 139.3 million more attacks in cryptojacking 43% more than in 2021 but as of 2023, the figure has risen to 1.06 billion attacks (SonicWall, 2023). Cryptojacking will soon become the highest priority cyber threat in the world, replacing ransomware.

1.5 Objectives

1. To accurately identify and classify the features of Crytojacking attacks on cloud servers.

2. To propose a method for the detection and mitigation Crytojacking attacks on cloud servers using machine learning techniques.

2. MATERIALS AND METHODS

The materials and methods utilized in the study. The first part focuses on describing environments used. The second part discusses how and where the data was collected, as well as the approach taken for data preparation. The final part elaborates on the procedure followed for the machine learning techniques used.

2.1 Programming Environment

Machine learning Python is very popular as a programming language in this area and it has a rich library ecosystem. The programming language used in this research is Python, which was chosen because it has a rich library support and it is generally easy to use in data driven applications. Scikit learn is one of the best Python machine learning libraries. The variety of supervised machine learning algorithms supported by this library is great and the number of methods of classification offered is enormous: Support Vector Machines (SVM), Naive Bayes and numerous others. In addition, Scikit learn possess capabilities of extracting features and it can be employed to augment the general extensibility of the library.

2.2 Methodology

To achieve the goal of this study to predict the cryptojacking attacks on the cloud environment Figure 2.1 illustrates the sequential procedure employed.

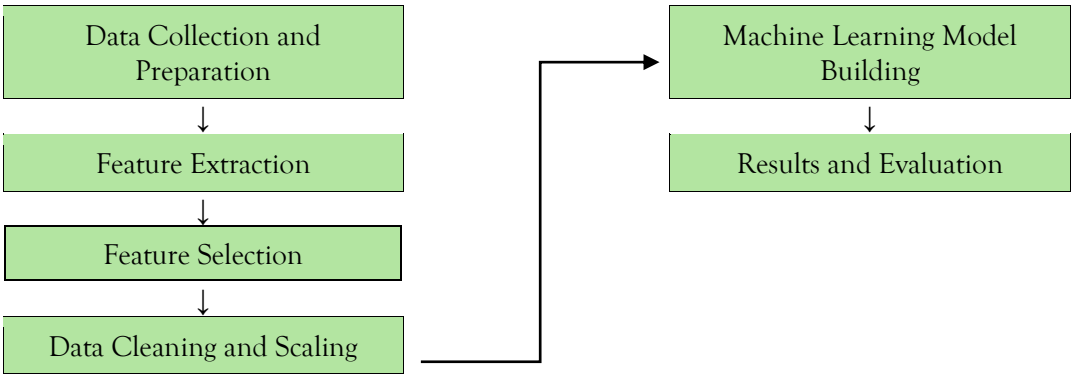

Figure 2.1 Methodology employed

2.2.1 Data Collection and Preparation

The Cryptojacking attacks dataset was downloaded on Kaggle in the First step. The downloaded file is in Comma Separated Values (CSV) which can be manipulated more easily with Python. The dataset includes 3 CSV files, as described below. (i) A normal dataset (ii) Normal dataset and (iii) Complete dataset. Normal and normal datasets include the time series performance data during a cryptojacking attack and during no cryptojacking attacks respectively. A combination of those two datasets is contained in the complete dataset.

2.2.2 Feature Extraction

For static analysis purposes, the Plato tool was utilized to measure JavaScript code complexity. Running this tool produced several complexity related features: SLOC, Cyclomatic Complexity, its density variant, Halstead Complexity Metrics and a Maintainability Score.

2.2.3 Feature Selection

In a bid to find out the most statistically significant features, the study used the Two Tailed Z test accompanied by multicollinearity evaluation. Variables failing to satisfy the significance

threshold were dropped and finally, the dataset was narrowed down to a final set of 29 meaningful variables.

2.2.4 Data Cleaning and Scanning

All feature values underwent normalization to fall within a 0 to 1 range, achieved through min max scaling to maintain uniform distribution across all variables. Although the resulting dataset exhibited significant class imbalance, no resampling techniques were applied, as the core objective of this study is to accurately distinguish abnormal system behaviour from normal usage patterns a scenario that inherently reflects real world conditions.

2.2.5 Machine Learning Model Building

Dynamic analysis was carried out using nine supervised classifiers SGD, Decision Tree, Random Forest, K Nearest Neighbors (KNN), MLP, Bernoulli Naive Bayes, Gaussian Naive Bayes, Logistic Regression and ADA Boost. The dataset was divided into a 75% training portion and a 25% testing portion, enabling each model to learn to distinguish abnormal system activity from normal operation.

2.2.6 Results and Evaluation

The results for both processes indicate that some of the models used in this study were able to identify covert crypto mining behaviour on the system easily with a high accuracy rate. The confusion matrix showed the number of false negative and false positive cases the models made.

3. RESULTS

To perform the experiments, all the classifier algorithms will have to be trained and then the tests must be done. To enable this, the dataset was partitioned into two separate subsets one for training and one for testing purposes. In this division we are able to train each classifier was fitted on the training subset and then evaluated against the test data, with the training portion accounting for 75% of the full dataset. The other 25% of the data is assigned to the test data.

3.1 Model Selection

Initially, choose the top performing model through the utilization of cross validation. Evaluate various classification algorithms and proceed with the model selection procedure. A variety of sentiment analysis classification models were utilized by us: Decision Tree, Support Vector Classifier (SVC), K Nearest Neighbours (KNN), Naive Bayes and Logistic Regression were selected and each implemented through its corresponding classifier within the experimental framework.
Each model's performance was assessed via 10-fold cross validation, with mean accuracy scores computed through the cross_val_score function and averaged over all folds. The accuracy scores were obtained by averaging the accuracy across the 10 folds.

3.2 Applying Machine Learning Algorithms

Experimental results on the combined dataset showed that the Decision Tree classifier recorded the highest F1 Score of 99.7% before any dimensionality reduction technique was applied.

Table 3. 1 Performance of the machine learning classification models using PCA

Classifier Name	Dimensionality Reduction Method	TP Rate	FP Rate	Precision	Recall	Accuracy	F1 Score
SGD	None	0.9924	0.0016	0.9968	0.9919	0.9960	0.9965
	PCA	0.9839	0.0032	0.9958	0.9881	0.9970	0.9921
	LDA	0.9013	0.0357	0.9162	0.9113	0.9522	0.9188
Decision Tree	None	0.9130	0.0025	0.9851	0.9262	0.9986	0.9975
	PCA	0.9799	0.0031	0.9789	0.9829	0.9953	0.9854

	LDA	0.9122	0.0322	0.9335	0.9222	0.9465	0.9528
Random Forest	None	0.9193	0.0324	0.9768	0.9123	0.9667	0.9221
	PCA	0.9207	0.0377	0.9778	0.9027	0.9746	0.9177
	LDA	0.9045	0.0324	0.9462	0.9336	0.9543	0.9245
KNN	None	0.9912	0.0024	0.9947	0.9349	0.9463	0.9893
	PCA	0.9764	0.0040	0.9848	0.9564	0.9950	0.9867
	LDA	0.8760	0.0373	0.9638	0.8540	0.9674	0.9343
MLP	None	0.9899	0.0056	0.9923	0.9739	0.9962	0.9886
	PCA	0.9958	0.0022	0.9920	0.9373	0.9932	0.9872
	LDA	0.9166	0.0333	0.9762	0.9066	0.9652	0.9321
Bernoulli Naive Bayes	None	0.9884	0.0051	0.5724	0.9864	0.7213	0.6574
	PCA	0.9679	0.0360	0.3876	0.9679	0.5132	0.5257
	LDA	0.9224	0.0321	0.9738	0.9224	0.9678	0.9210
Naive Bayes	None	0.9165	0.0343	0.9129	0.9365	0.9533	0.9166
	PCA	0.7829	0.0678	0.8202	0.7232	0.9296	0.8258
	LDA	0.9127	0.0285	0.8765	0.9257	0.9220	0.8721
Logistic Regression	None	0.9260	0.0403	0.9773	0.9364	0.9262	0.9365
	PCA	0.9152	0.0535	0.9465	0.9134	0.9626	0.9265
	LDA	0.9169	0.0240	0.9565	0.9113	0.9613	0.9254

When the dataset was reduced using Principal Component Analysis (PCA) with 20 components along with Linear Discriminant Analysis (LDA) and the classifiers were subsequently applied to this reduced dataset, it was observed that the Random Forest classifier produced an F1-Score of 99.2%. It was further noted that incorporating dimensionality reduction into the pipeline resulted in a decline in the F1-Score of the Random Forest Classifier when compared to the F1-Score achieved by the same classifier operating without any dimensionality reduction method being applied. The Table 3.1 reveals that Bernoulli Naive Bayes consistently produces the worst outcomes.

Table 3. 2 Performance of the classification models after hyper parameter tuning

Classifier Name	TP Rate	FP Rate	Precision	Recall	F1 Score	Accuracy
Decision Tree	0.998893529	0.002103063	0.997756321	0.997893529	0.99782492	0.998296681
Regular Random Forest	0.947380437	0.002610808	0.998973983	0.997380437	0.998176574	0.998779629

Gaussian Naive Bayes	0.977821669	0.061588611	0.945210148	0.937821669	0.941501413	0.943381234
Bernoulli Naive Bayes	0.845862619	0.153785263	0.949903646	0.825862619	0.883550868	0.881249605
Logistic Regression	0.957854582	0.061567778	0.945059513	0.937854582	0.941443263	0.953717717
Bagging Decision	0.996643295	0.002349977	0.998746545	0.997643295	0.998194615	0.994197197
Bagging Naive Bayes	0.998976775	0.023157935	0.58462557	0.992976775	0.735951762	0.654048071
Bagging Logistic	0.978956937	0.023235059	0.681561286	0.944956937	0.744895377	0.634451895
ADA Boost	0.96313224	0.046133244	0.981355614	0.96313224	0.972158534	0.972541266

From the Table 3.2, we can see that Bagging Logistic performs comparatively worse than other classifiers with accuracy of 63.4451% and f1 score of 88.3550%, recall score of 82.58% and lastly the precision of 94.99%. Also, we can state that, the performance of Bagging Logistic with accuracy of 63.44% and f1 score of 74.49%, recall score of 99.29% and lastly the precision of 58.46% is not satisfactory.

On the other hand, we can see that Regular Random Forest worked better than all other classifiers we experimented with the accuracy of 99.87% and f1 score of 99.81%, recall score of 99.71% and lastly the precision of 99.89%. It is also noticeable that Decision Tree also performed well with accuracy of 99.82% and f1 score of 99.78%, recall score of 99.78% and lastly the precision of 99.77%.

3.3 Applying Confusion Matrix for Machine Learning Algorithms

In this section, confusion matrices were utilized to analyze and observe the classification behaviour of the models that demonstrated the highest and lowest accuracy levels when measured against all other classifiers included in this study.

49.88 %	**0.06 %**
0.11 %	**49.93 %**

Figure 3.1 Confusion Matrix of Regular Random Forest

The confusion matrix produced by the Regular Random Forest as shown in Figure 3.1 indicates that the classifier has been able to detect both True positive and True negative with great accuracy. Moreover, one can note that the amount of false classified cases in the False positive and False negative categories was small throughout the dataset. Using these figures, we can note that we

have come up with sufficient machine learning models.

11.79 %	35.25 %
0.55 %	49.60 %

Figure 3.2 Bagging logistic Confusion Matrix.

According to Figure 3.2, the confusion matrix of Bagging Logistic, it is clear that this classifier had very poor performance in predicting True Positive class, whereas it showed almost acceptable performance in predicting the True Negative class. Moreover, both the False Positive and False Negative groups showed major misclassifications in the entire dataset. According to these results, it is possible to conclude that Bagging Logistic showed the lowest overall classification performance of all the models considered.

4. DISCUSSION

The classifiers tested throughout this study yielded reliable and adequate outcomes for identifying cryptojacking activity within cloud server environments. There were a lot of challenges occur for detecting the cryptojacking attacks on a cloud server using machine learning. From our point of view, dataset plays a major role for getting the better accuracy. The dataset utilized for the static analysis is constrained, which is inhibiting a comprehensive comprehension of how the research could operate with a more extensive dataset. It is difficult to distinguish the prediction of attack label in the imbalanced dataset, particularly as cryptojacking attack patterns continue to evolve across IoT and cloud environments (Reddy et al., 2024). Machine learning models tend to perform better on the majority class, leading to suboptimal performance on the minority class. Because the minority class has fewer observations, it can be difficult to find meaningful patterns in the data, leading to poor performance (Almajed et al., 2022). Imbalanced datasets remain a major challenge in cryptojacking detection and a systematic review of machine learning approaches confirms that selecting the right classification strategy is critical for improving detection accuracy (Singh & Murugan, 2024). Also, before starting the training and testing, the classification and preprocessing steps, including JavaScript code semantic representation and feature selection, put a major impact on the accuracy of detecting cyber threats (Fang et al., 2022).

The experimental findings reveal that ensemble-based classifiers, especially Regular Random Forest, can be effectively used in detecting cryptojacking behaviour in a cloud environment. The high performance of the Random Forest can be explained by the fact that it can merge several decision trees, which diminishes overfitting and proves to be more effective in dealing with variance in network traffic data than single tree or linear models. Conversely, Bagging Logistic failed miserably since logistic regression is a linear classifier, whereas cryptojacking traffic patterns are non linear and are complex, which does not fit the linear classifier.

Multiple experimental runs were conducted across the datasets by applying varied filtering strategies and adjusting the parameter configurations available for each classifier. The outcomes of every test run were evaluated for each classifier based on the number of correctly and incorrectly classified instances, True Positive Rate (TPR), False Positive Rate (FPR), Precision and the time required to build the model. The confusion matrix generated for the Regular Random Forest demonstrates that this classifier accurately identified both the True Positive and True Negative classes, while the False Positive and False Negative instances were misclassified with only minor errors across the

dataset. Based on these observations, it is evident that the machine learning models developed in this study have achieved acceptable classification performance. Conversely, the confusion matrix of Bagging Logistic indicates that this classifier significantly failed to perform well in the correct identification of the True Positive class and the near acceptable performance in the True Negative class. Moreover, the False Positive and the False Negative classes were both vulnerable to significant misclassification errors throughout the dataset and the conclusion that the Bagging Logistic was the worst overall performer out of all of the classifiers tested.

5. CONCLUSION AND RECOMMENDATIONS

5.1 Conclusions

This study introduced a detection framework that leverages machine learning models to monitor and evaluate critical parameters of a cloud server in order to identify cryptojacking incidents. The proposed system is capable of determining both whether a cryptojacking attack is actively occurring and the extent to which the Central Processing Unit (CPU) is being utilized. The machine learning based classification approach delivered enhanced detection outcomes for identifying cryptojacking threats targeting cloud infrastructure. The overall framework operates across three core stages: data pre processing, feature selection and final attack detection. At the initial stage, all attributes associated with incoming network traffic are normalized on a standard scale before being fed into the various machine learning classifiers. Among all classifiers evaluated, Random Forest consistently outperformed the others, achieving an accuracy of 99.87%.

The machine learning classification system proposed had a relatively high detection rate of 99.87% accuracy that is higher than the detection rates reported in previous studies on cryptojacking malware (Tekiner et al., 2021). Since the total number of cryptojacking attacks increased to 1.06 billion around the world in 2023, which is 659% higher than in 2022, effective and precise detection systems such as the one suggested in this paper are becoming more and more essential in securing cloud-based infrastructure (SonicWall, 2023).

5.2 Recommendations

This study might be extended by determining other similar characteristics and adding them to the current dataset. It should be mentioned that the number of machine learning algorithms used in the study, while comprehensive, could be further extended and further research may be enriched with more experiments with the classifier variety to make more comprehensive comparisons. Expanding the dataset used for static analysis would also address one of the key limitations encountered in this work. Additionally, extending the dynamic analysis to assess the performance parameters of multiple operating systems under active cryptojacking conditions would significantly widen the research scope. A practical application of this work could involve the development of a real time antivirus or intrusion detection system that integrates both the static and dynamic approaches explored in this study. Furthermore, tuning hyperparameter settings across various neural network architectures and binary classification algorithms could further strengthen detection performance for cryptojacking attacks on cloud infrastructure. Incorporating cross validation as part of the training strategy is also recommended as an additional optimization measure.

REFERENCES

Barbhuiya, S., Papazachos, Z., Kilpatrick, P., & Nikolopoulos, D. S. (2018). RADS: Real-time anomaly detection system for cloud data centers. 20(43), 36–39.

SonicWall. (2023). 2023 SonicWall Cyber Threat Report Mid-Year Update. SonicWall Inc. Retrieved from https://www.sonicwall.com

Carlin, D., O'kane, P., Sezer, S., & Burgess, J. (2018). Detecting cryptomining using dynamic analysis. In 2018 16th Annual Conference on Privacy, Security and Trust (PST), 5(2), 1–6.

Almajed, H., Alsaqer, A., & Frikha, M. (2022). Imbalance datasets in malware detection: A review of current solutions and future directions. International Journal of Advanced Computer Science and Applications, 13(9). https://doi.org/10.14569/IJACSA.2022.01309XX

Fang, Y., Huang, C., Zeng, M., Zhao, Z., & Huang, C. (2022). JStrong: Malicious JavaScript detection based on code semantic representation and graph neural network. Computers & Security, 118, 102715. https://doi.org/10.1016/j.cose.2022.102715

Reddy, C. K. K., Kaza, V. S., R., M. M., Alamer, A., Alam, S., Shuaib, M., Basudan, S., & Sheneamer, A. (2024). Detecting and forecasting cryptojacking attack trends in Internet of Things and wireless sensor networks devices. PeerJ Computer Science, 10, e2491. https://doi.org/10.7717/peerj-cs.2491

Zheng, Z., Xie, S., Dai, H., Chen, X., & Wang, H. (2022). An overview of blockchain technology: Architecture, consensus and future trends. Future Generation Computer Systems, 107, 841-853. https://doi.org/10.1016/j.future.2019.07.024

Hong, G., Yang, Z., Yang, S., Zhang, L., Nan, Y., Zhang, Z., & Duan, H. (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, 1701–1713.

Tekiner, E., Acar, A., Uluagac, A. S., Kirda, E., & Selcuk, A. A. (2021). In-browser cryptojacking: Characterization, detection and defense. IEEE Transactions on Dependable and Secure Computing, 19(3), 1421–1434. https://doi.org/10.1109/TDSC.2021.3092561

Jayasinghe, K., & Poravi, G. (2020). A survey of attack instances of cryptojacking targeting cloud infrastructure. In Proceedings of the 2020 2nd Asia Pacific Information Technology Conference, 100–107.

Kaur, S., & Kaur, G. (2021). Threat and vulnerability analysis of cloud platform: a user perspective. In 2021 8th International Conference on Computing for Sustainable Global Development (INDIACom), 533-539.

Naseem, F., Aris, A., Babun, L., Tekiner, E., & Uluagac, A. S. (2021). MINOS: A lightweight real-time cryptojacking detection system. In Proceedings of the 2021 Network and Distributed System Security Symposium (NDSS). https://doi.org/10.14722/ndss.2021.24444

Kharraz, A., Ma, Z., Murley, P., Lever, C., Mason, J., Miller, A., & Bailey, M. (2019). Outguard: Detecting in-browser covert cryptocurrency mining in the wild. In The World Wide Web Conference, 840–852.

Konoth, R. K., Vineti, E., Moonsamy, V., Lindorfer, M., Kruegel, C., Bos, H., & Vigna, G. (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, 1714–1730.

Musch, M., Wressnegger, C., Johns, M., & Rieck, K. (2019). Web-based cryptojacking in the wild. In Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security (AsiaCCS), 1–12. https://doi.org/10.1145/3321705.3329793

Meland, P. H., Bayoumy, Y. F. F., & Sindre, G. (2020). The Ransomware-as-a-Service economy within the darknet. Computers & Security, 5(92), 101762.

Rodriguez, J. D. P., & Posegga, J. (2018). Rapid: Resource and API-based detection against in-browser miners. In Proceedings of the 34th Annual Computer Security Applications Conference, 313–326.

Ruth, J., Zimmermann, T., Wolsing, K., & Hohlfeld, O. (2018). Digging into browser-based crypto mining. In Proceedings of the Internet Measurement Conference 2018, 70–76.

Saad, M., Khormali, A., & Mohaisen, A. (2019). Dine and dash: Static, dynamic and economic analysis of in-browser cryptojacking. In 2019 APWG Symposium on Electronic Crime Research (eCrime), 1–12.

Sanda, O., Pavlidis, M., & Polatidis, N. (2023). A deep learning approach for host-based cryptojacking malware detection. Evolving Systems, 1-16.

Aponte-Novoa, F. A., Povedano Álvarez, D., Villanueva-Polanco, R., Sandoval Orozco, A. L., & García Villalba, L. J. (2022). On detecting cryptojacking on websites: Revisiting the use of classifiers. Sensors, 22(23), 1-15. https://doi.org/10.3390/s22239219

Shareef, M. N., Ali, J. H. M. K. A., & Khan, M. A. B. (2023). Crypto Jacking. Mathematical Statistician and Engineering Applications, 72(1), 1581-1586.

Singh, P. K., & Murugan, R. (2024). A systematic review and comparative study of cryptojacking detection via machine learning. ResearchGate. https://doi.org/10.13140/RG.2.2.27893.93921

Tekiner, E., Acar, A., Uluagac, A. S., Kirda, E., & Selcuk, A. A. (2021). SoK: cryptojacking malware. In 2021 IEEE European Symposium on Security and Privacy (EuroS&P) 120-139.

Ying, Q., Yu, Y., Tian, D., et al. (2022). CJSpector: A novel cryptojacking detection method using hardware trace and deep learning. Journal of Grid Computing, 20, 31. https://doi.org/10.1007/s10723-022-09621-2

Wang, W., Ferrell, B., Xu, X., Hamlen, K. W., & Hao, S. (2018). Seismic: Secure in-lined script monitors for interrupting cryptojacks. In Computer Security: 23rd European Symposium on Research in Computer Security, ESORICS 2018, Barcelona, Spain, September 3–7, 2018, Proceedings, 122–142.