An Anomaly Detecting Blockchain Strategy for Secure IoT Networks

Naseem Alsadi*[a], Waleed Hilal[a], Onur Surucu[a], Alessandro Giuliano[a], Stephen A. Gadsden[a], John Yawney[b], Stephan Iskander[c]

[a]McMaster University, 1280 Main St W, Hamilton, ON L8S 4L8; [b]Adastra Corporation, 8500 Leslie St #600, Thornhill, ON L3T 7M8; [c]University of Guelph, Guelph, ON N1G 2W1

ABSTRACT

Highly distributed connected systems, such as the Internet of Things (IoT), have made their way across numerous fields of application. IoT systems present a method for the connection for various heterogeneous devices across the internet, facilitating the efficient distribution, collection and processing of system-related data. However, while system inter-connectivity has aided communication and augmented the effectiveness of integrated technology, it has also increased system vulnerability. To this end, researchers have proposed various security protocols and frameworks for IoT ecosystems. Yet while many suggested approaches augment system security, centralization remains an area of concern within IoT systems. Therefore, we propose the use of a decentralization scheme for IoT ecosystems based on Blockchain technology. The proposed method is inspired by Helium, a public wireless long-range network powered by blockchain. Each network node is characterized by its device properties, which are comprised of local and network-level features. Communication in the network requires the testimony of other companion nodes, ensuring that anomalous behaviour is not accepted and thereby preventing malicious attacks of various sorts.

Keywords: Internet of Things, Blockchain, Cybersecurity

1. INTRODUCTION

Blockchain is a rapidly advancing state of the art technology that, at its core, is a method for the decentralization of contemporary systems. Most notably applied to cryptocurrencies, Blockchain can eliminate the need for any centralization within various systems, including smart systems. Decentralization can improve numerous aspects of a system, including security and efficiency [1].

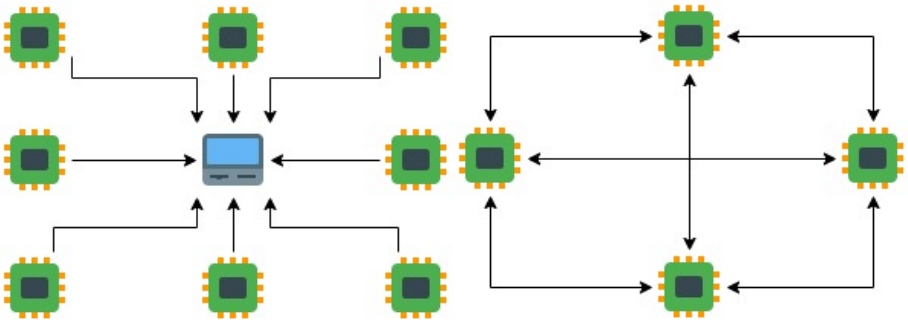

Figure 1: a) Centralized Network b) Decentralized Network

Blockchain is a distributed ledger that comprises records, referred to as blocks. These blocks, which track system transactions, are distributed across numerous devices in the network. Each block within the network contains a cryptographic hash of the block prior to it in the blockchain. The network is constructed in a peer-to-peer manner, supporting the decentralization of the network and equal allocation of privileges amongst network nodes [2]. This is in contrast with network models like the client-server model, where network clients are required to request services and resources from a centralized server.

Disruptive Technologies in Information Sciences VI, edited by Misty Blowers, Russell D. Hall, Venkateswara R. Dasari, Proc. of SPIE Vol. 12117, 121170A · © 2022 SPIE · 0277-786X · doi: 10.1117/12.2618301

The distribution of the chain across numerous network devices, rather than a central authority, prevents the retroactive manipulation of blocks without the recalculation of the hash for all subsequent blocks in the chain, and this must be completed prior to the addition of any new blocks to the chain [3].

The integration of blockchain with the Internet of Things (IoT) enables a greater overall security structure. IoT systems are, by nature, heavily distributed. However, reliance on a central cloud facilitates system vulnerabilities [4]. Therefore, the integration of blockchain and IoT has been explored by numerous researchers. Figure 1 depicts the rapid increase in publications concerning IoT and blockchain, sampled from Google Scholar and IEEE between the years 2013 and 2021.

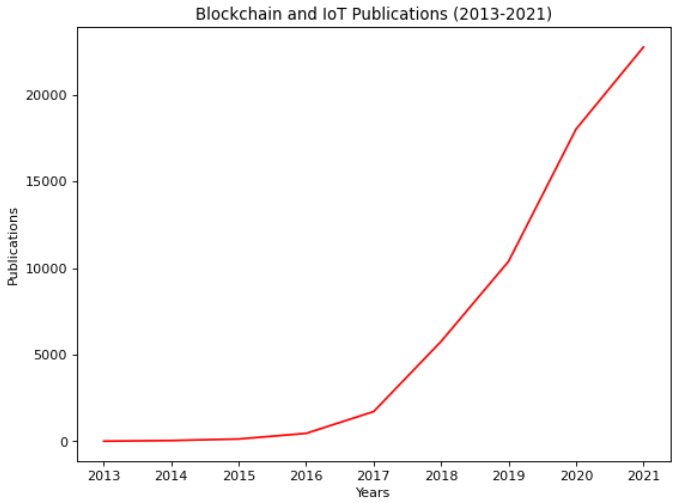

Figure 2: Blockchain and IoT Publications

The core question to be asked of the implementation of blockchain within IoT applications is, what problems does blockchain aim to solve and how does it aim to do so? Fundamentally, the implementation of blockchain in IoT applications will target the decentralization of the overall system network.

Problem Statement

Widespread interconnectivity in IoT ecosystems is a point of concern for network security. Malicious actors can take advantage of a single highly trusted and widely connected network node to jeopardize the integrity, availability and confidentially of an entire system. Contemporary implementations of IoT ecosystems are based on centralized architectures. While this approach reduces design complexity, the potential for single points of failure is significant and the consequences are detrimental to the overall system function. In addition, IoT devices are heavily resource-constrained, therein constricting the utilization of intensive security protocols.

Proposed Solution

In this paper, we propose the employment of deep learning backed blockchain for the purposes of securing an IoT ecosystem. The fundamental objective of the proposed architecture is to ensure the secure and fast transmission of data within an IoT ecosystem. To this end, we posit a single-layered blockchain architecture comprising a set of interconnected network nodes that aim to validate companion nodes' behaviour through a novel consensus algorithm, Proof of Integrity. The architecture is inspired by Helium, a public wireless long-range network powered by blockchain [5].

2. PROPOSED METHOD

The proposed approach is built on top of a highly interconnected environment of network nodes, which represent standard resource-constrained IoT devices. Although the fundamental kernel of communication between these devices is heavily contingent on the application space, the transmission of data, by a node in the network, can only be facilitated through the validation of a node's characteristics, referred to in this paper as its device characteristics. These characteristics are

essentially the node's fingerprint. They are subject to change across applications but must be composed of network relative and individual features, device characteristics are discussed more in detail below.

We postulate that device characteristics can reveal significant information about node intentions. A lot of research has been conducted on the detection of malicious behaviour using various instances of device characteristics [6]–[10] . For example, Azmoodeh *et al.* employed numerous detection techniques to detect malicious behaviour by analyzing device power consumption [11]. Milosevic *et al.* utilize RAM and CPU usage to detect malicious behaviour [12].

Each node that wishes to transfer data throughout the network will need to do such through the process of mining. The novel proof of integrity consensus algorithm ensures that devices appending data to network blockchain are operating in accordance with their precedently established device characteristics. As previously mentioned, each individual node has a set of device characteristics, some network relative and others, individual. Therefore, with the aim of using collective network connectivity and computational power, the consensus algorithm will use the testimony of a set of fellow network nodes to validate the data being appended to the chain.

Each node represented in the network is a device of a potential resource-constrained nature. Devices will be connected via a set of arbitrary IoT communication protocols. Network nodes will have individual and network relative properties. Power consumption, CPU usage, RAM utilization are examples of a node's individual properties. A node's network relative properties are features of the nodes which can be verified independent of that node's testimony. Companion node connectivity, node location, and communication frequency are all instances of a node's network relative properties.

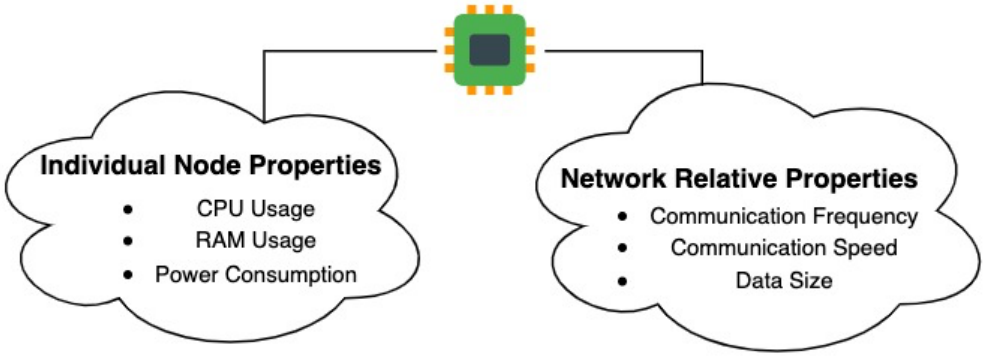

Figure 3: Node Characteristics

It is important to keep in mind that the set of device characteristics selected will be contingent on the application space. Employment of the network within a potentially hostile environment, where device location needs to be kept private, will require specific tailoring of a node's characteristics, avoiding perhaps sensitive information such as node location.

Node responsibility is crucial to the overall functionality and security of the architecture. Each node in the network must take on a responsibility space composed of four variant duties, namely challenger, challengee, witness and validator. The challengee and challenger roles are the only roles that can be self-assigned. The witness and validator roles are randomly assigned when a block insertion request is broadcasted to the network by the challengee. This prevents the challengee from verifying itself or knowing which nodes will be involved in the verification process before making the block insertion request.

The challengee is the device which is attempting to transmit data to one or more nodes in the network. The challengee node will make a request to transfer data to a companion node in the network. That companion node now becomes the challenger and performs two main tasks. The first of which is to ensure that the challengee is operating within regular node characteristics using a smart contract with integrated anomaly detection. The second task is to create a broadcast request to other nodes in the network, inviting them to become witnesses and validators to the addition of the new block. However, it is crucial to note, that without the addition of a stochastic element, the witnesses and validators can be routinely predicted, leading to a severely comprised network. Therefore, the process for selecting witness and validator nodes is completed with stochastic consensus.

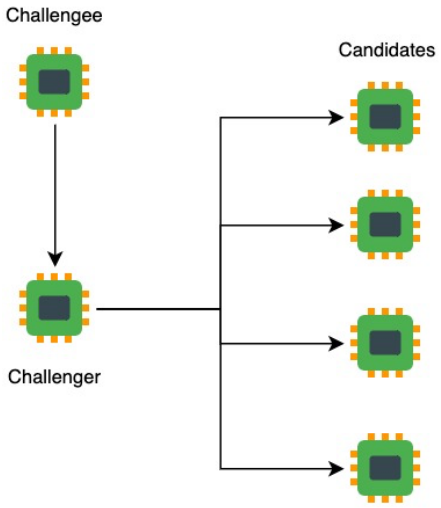

Figure 4: Broadcast Request to Candidates

The procedure begins when the candidate pool receives a broadcast request from the challenger node, asking to partake in the mining process. Each individual node, within the candidate pool, will reply to the challenger with a privately generated pseudorandom number. The pseudorandom generation algorithm chosen for this task is the Mersenne Twister [13]. Although not cryptographically secure, the Mersenne Twister provides the necessary means of randomness. Note that the Mersenne Twister can be replaced with variant methods of cryptographically safe random number generators, however, more complex pseudorandom generators are computationally expensive and can therefore slow down the end-to-end transmission of data [14].

The challenger will wait for φ nodes to reply, where φ is the maximum size of the candidate pool that is baked into the network protocol. Once the candidate pool has been established, the challenger is left with a random numerical sequence, produced by the collective efforts of all the devices in the candidate pool.

$$x_{1\to\varphi} = [x_1, x_2, \dots x_\varphi] \tag{1}$$

Note that the randomness of the model increases with φ, or the maximum size of the candidate pool. However, increasing φ influences the overall energy consumed by the network. The larger the candidate pool, the larger the number of nodes required to make necessary computations, and therefore the more energy dispensed by the network. Tuning φ to ensure an adequate quantity of randomness, while maintaining computational complexity, is crucial to network employment.

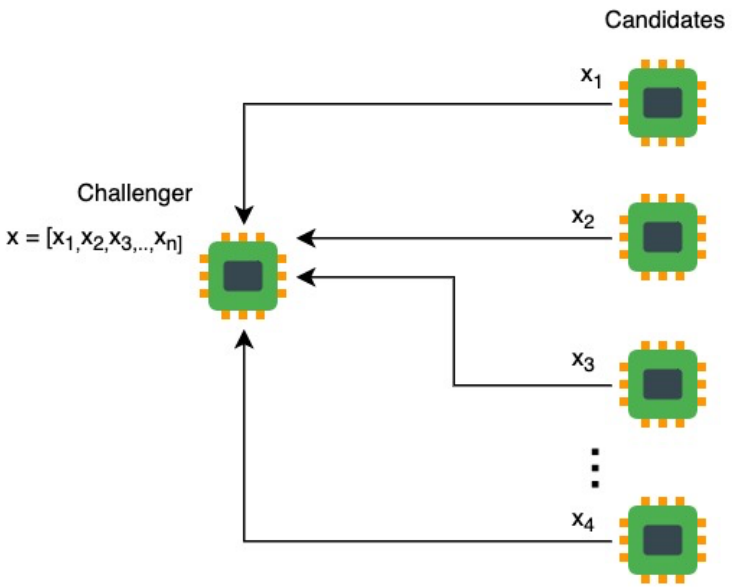

Figure 5: Stochastic Consensus

The concept of the combination of multiple sources of weak randomness generated by the calculation of pseudorandom numbers by individual nodes is inspired by research conducted in 1999 by Santha and Vazirani on the combination of bit streams with weak randomness to generate a complex quasi-random bit stream [15], [16]. When communication entropy is further introduced as a means of sorting the sequence, the randomness of the process is augmented. The summation of the sequence, named the Stochastic Consensus Element (SCE), is utilized to randomly assign witness or validator responsibility to the respective nodes in the candidate pool.

$$SCE = \sum_{\varphi}^{i=1} x_i \tag{2}$$

The SCE will be embedded within the final block mined to the chain, ensuring that validators can check to see that the selection of the respective witnesses and validators was conducted honestly, without manipulation.

The number of witness or validator nodes is calculated with the utilization of the formulas listed below.

$$n_w = Ceil\big((n-2) \times \varepsilon \times \theta_w\big) \tag{3}$$

$$n_v = Ceil\big((n-2) \times \varepsilon \times \theta_v\big) = Ceil\big((n-2) \times \varepsilon \times (1-\theta_w)\big) \tag{4}$$

where n_w, and n_v is the number of witness nodes and validator nodes respectively. ε is the network utilization factor. θ_w and θ_v is witness and validator split factor.

Once the designated candidate nodes have been allocated as either witnesses or validators, they can begin fulfilling their respective responsibilities. The duty of a witness node is to provide sufficient information about the challenged node's network relative properties. These properties will provide insight on the historical and concurrent communication behavior of the node in question. Conducting this is akin to having nodes vouch on behalf of the challenged node. The more witness nodes we have involved in the consensus process, the more testimonies we have, and therefore the more likely the correct verdict is reached.

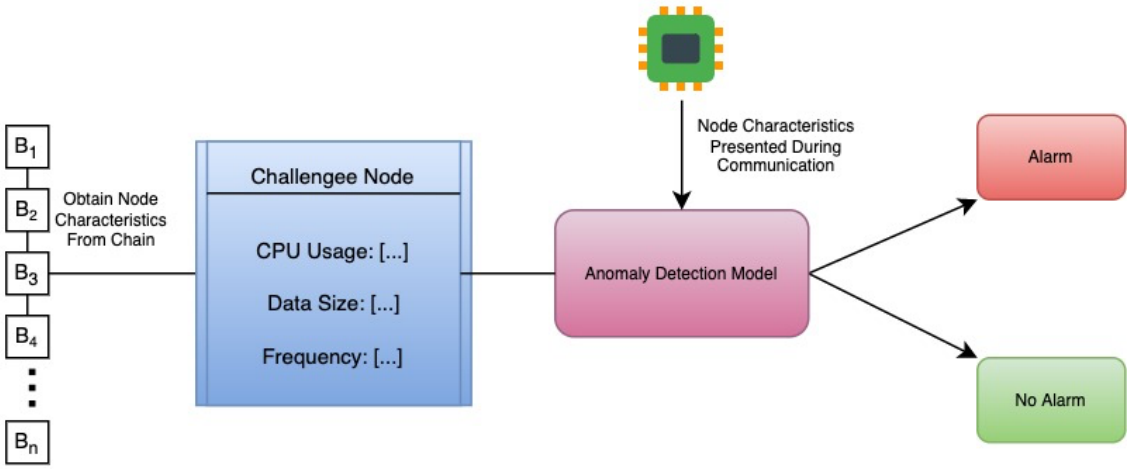

Figure 6: Overview of Proposed Process

As described prior, network relative properties will assist in building a diverse node profile. Let us take a scenario where the challenged node has been hijacked by a malicious agent. The agent is looking to perform a high communication frequency-based attack, with the goal being to target system availability. The first layer of defence is the anomalous behaviour detection-embedded smart contract. The recipient node of the data transaction will need to verify the node's individual properties, however, note that this is only a preliminary method of verification. The malicious agent can easily manipulate the outgoing property values and conceal anomalous behaviour. This is where the network relative properties are vital to the security of the network.

Network relative properties ensure that behavioural information about the node is not influenced by a malicious agent. The information, rather, is taken from the testimony of witnesses, stochastically selected from the network. The random selection of these witnesses is crucial to maintaining the integrity of the network. Returning to our example, and assuming the node has bypassed the anomaly detecting smart contract, the recipient node will broadcast a request for fellow network nodes to join the candidate pool. The key factor is the introduction of the SCE into the candidate pool. Again, the larger the candidate pool, the greater the diversity of entropy in the network, and therefore the more superior the stochasticity. The malicious node will not know which nodes will be selected as witnesses or validators, nor will he know which subset of nodes will be included within the greater scope of the candidate pool. This dramatically increases the computational workload a malicious agent must perform in order to select which nodes to hijack.

In addition, if the malicious agent was to succeed in the meticulous calculation and prediction of which nodes will be assigned witness responsibility, then he does so with an increased computational workload, which is indicated in the node's individual properties. Therefore, when the hijacked node attempts to communicate on the network, the recipient node will be able to identify the anomalous increase in the computational workload of the hijacked node. The hijacked node will not be able to transmit data until the computational workload returns to the regular range of function and therefore, in retrospect, the computational effort will be rendered useless because a new distribution of candidate nodes will be generated for the novel mining iteration.

Nodes append data to the chain in a fundamental data structure known as the block. The block will contain a set of necessary pieces of information about the novel data being appended to the chain, as well as information on the mining process which can be utilized to verify it in the future.

As aforementioned, validators will oversee the process of adding a block to the chain. The validator will begin by validating the SCE based on information conveyed by both the challenger and the candidate node. The ordering of the sequence of stochastic elements, $[x_1, x_2, \dots x_\varphi]$, must match the dispatch time conveyed by the respective candidate and the arrival time conveyed by the challenger. If the verification process of the SCE fails, the block is disregarded.

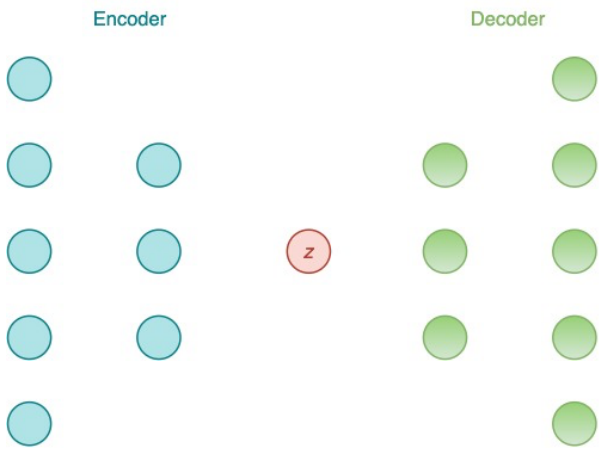

Figure 7: AutoEncoder

If the SCE is valid, the validator proceeds to append the respective block to the chain, containing the SCE, transaction details, and node characteristics, both individual and network relative, of all the devices involved in the mining process. The validator completes the process by broadcasting the new chain to all the network nodes.

The detection of anomalous behaviour in the network is conducted with the utilization of an AutoEncoder. The goal of the AutoEncoder is to encode input data into a low dimensional latent representation and reconstruct the data into its original dimensionality. An autoencoder is chosen for this task due to its ability to accurately recognize anomalous data of various input formats [17]:

$$\phi: X \to Z \tag{5}$$

$$\psi: Z \to X \tag{6}$$

$$\phi, \psi = \arg\min_{\phi,\psi} \|X - (\psi \circ \phi)X\|^2 \tag{7}$$

where X, ϕ, Z, ψ is the input, encoder, latent representation, and decoder, respectively.

Each node in the network will need to process the node characteristics provided by companion nodes in the network and sampled from the blockchain. The anomaly detection algorithm must be capable of accurately processing the device characteristics such that anomalous individual behaviour, like CPU over usage, and network relative properties, like abnormal communication frequency, are detected and flagged.

In addition, when a device is flagged for anomalous behaviour, its communication frequency becomes limited. The limiting factor is implemented with exponential backoff [18]. Formulated as:

$$f = \frac{1}{b^c}, \tag{8}$$

where b is a base factor predefined in every network node and c is the number of times the suspected node has been flagged for anomalous behaviour. Note that this only occurs on the local level, rather than throughout the entire network. This prevents falsified testimonies by malicious actors.

CONCLUSION

In brief, this paper defines an IoT blockchain architecture inspired by the Helium blockchain. Network nodes are characterized by their device properties, which are composed of local and network-level features. Blocks can only be added to the network via a novel consensus protocol, Proof of Integrity. Network security relies on the testimony of various companion nodes, which are stochastically selected. Node characteristics are analyzed with an anomaly detection algorithm. To this end, an AutoEncoder is employed for the detection of anomalous behaviour across a variety of different input forms. Anomalous devices are flagged, and their communication is subsequently limited using

exponential backoff to prevent high frequency-based or recurrent attacks. The project codebase is available at https://github.com/nalsadi/Deep_Blockchain_IoT.

REFERENCES

[1] M. Conoscenti, A. Vetrò, and J. C. De Martin, "Peer to Peer for Privacy and Decentralization in the Internet of Things," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, May 2017, pp. 288–290. doi: 10.1109/ICSE-C.2017.60.

[2] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, "Where Is Current Research on Blockchain Technology?—A Systematic Review," *PLOS ONE*, vol. 11, no. 10, p. e0163477, Oct. 2016, doi: 10.1371/journal.pone.0163477.

[3] M. Wang, M. Duan, and J. Zhu, "Research on the Security Criteria of Hash Functions in the Blockchain," in *Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts*, New York, NY, USA, May 2018, pp. 47–55. doi: 10.1145/3205230.3205238.

[4] M. R. Dorsala, V. N. Sastry, and S. Chapram, "Blockchain-based solutions for cloud computing: A survey," *J. Netw. Comput. Appl.*, vol. 196, p. 103246, Dec. 2021, doi: 10.1016/j.jnca.2021.103246.

[5] "Helium – Introducing The People's Network." https://www.helium.com/ (accessed Mar. 23, 2022).

[6] F. Gomes and M. Correia, "Cryptojacking Detection with CPU Usage Metrics," in *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*, Nov. 2020, pp. 1–10. doi: 10.1109/NCA51143.2020.9306696.

[7] R. Bridges, J. Hernández Jiménez, J. Nichols, K. Goseva-Popstojanova, and S. Prowell, "Towards Malware Detection via CPU Power Consumption: Data Collection Design and Analytics," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, Aug. 2018, pp. 1680–1684. doi: 10.1109/TrustCom/BigDataSE.2018.00250.

[8] P. Luckett, J. T. McDonald, W. B. Glisson, R. Benton, J. Dawson, and B. A. Doyle, "Identifying stealth malware using CPU power consumption and learning algorithms," *J. Comput. Secur.*, vol. 26, no. 5, pp. 589–613, Jan. 2018, doi: 10.3233/JCS-171060.

[9] K. Hausknecht, D. Foit, and J. Burić, "RAM data significance in digital forensics," in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2015, pp. 1372–1375. doi: 10.1109/MIPRO.2015.7160488.

[10] C.-W. Tien, S.-W. Chen, T. Ban, and S.-Y. Kuo, "Machine Learning Framework to Analyze IoT Malware Using ELF and Opcode Features," *Digit. Threats Res. Pract.*, vol. 1, no. 1, p. 5:1-5:19, Mar. 2020, doi: 10.1145/3378448.

[11] A. Azmoodeh, A. Dehghantanha, M. Conti, and K.-K. R. Choo, "Detecting crypto-ransomware in IoT networks based on energy consumption footprint," *J. Ambient Intell. Humaniz. Comput.*, vol. 9, no. 4, pp. 1141–1152, Aug. 2018, doi: 10.1007/s12652-017-0558-5.

[12] J. Milosevic, M. Malek, and A. Ferrante, "A Friend or a Foe? Detecting Malware using Memory and CPU Features:," in *Proceedings of the 13th International Joint Conference on e-Business and Telecommunications*, Lisbon, Portugal, 2016, pp. 73–84. doi: 10.5220/0005964200730084.

[13] M. Matsumoto and T. Nishimura, "Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, Jan. 1998, doi: 10.1145/272991.272995.

[14] A. M. Gergely and B. Crainicu, "A succinct survey on (Pseudo)-random number generators from a cryptographic perspective," in *2017 5th International Symposium on Digital Forensic and Security (ISDFS)*, Apr. 2017, pp. 1–6. doi: 10.1109/ISDFS.2017.7916504.

[15] M. Santha and U. V. Vazirani, "Generating Quasi-Random Sequences From Slightly-Random Sources," in *25th Annual Symposium onFoundations of Computer Science, 1984.*, Singer Island, FL, 1984, pp. 434–440. doi: 10.1109/SFCS.1984.715945.

[16] M. Herrero-Collantes and J. C. Garcia-Escartin, "Quantum Random Number Generators," *Rev. Mod. Phys.*, vol. 89, no. 1, p. 015004, Feb. 2017, doi: 10.1103/RevModPhys.89.015004.

[17] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Autoencoder-based network anomaly detection," in *2018 Wireless Telecommunications Symposium (WTS)*, Apr. 2018, pp. 1–5. doi: 10.1109/WTS.2018.8363930.

[18] B.-J. Kwak, N.-O. Song, and L. E. Miller, “Performance analysis of exponential backoff,” *IEEEACM Trans. Netw.*, vol. 13, no. 2, pp. 343–355, Apr. 2005, doi: 10.1109/TNET.2005.845533.