BOMBAT : Borda Mitigates Ballot Stuffing Attacks in Fog Networks

Rasagna V
Dept of CSIS
Bits Pilani, Hyderabad, India
p20200019@hyderabad.bits-pilani.ac.in

Geethakumari G
Dept of CSIS
Bits Pilani, Hyderabad, India
geetha@hyderabad.bits-pilani.ac.in

Timothy Zachariah Binesh
Dept of EEE
Bits Pilani, Hyderabad, India

J SriPada Reddy
Dept of CSIS
Bits Pilani, Hyderabad, India

***Abstract*—Fog computing extends cloud services to network edges, facilitating real-time processing and low latency. However, its decentralized nature introduces security challenges, including trust-based attacks like ballot stuffing. Ballot stuffing manipulates decision outcomes by injecting biased votes, posing a significant threat in fog environments. Traditional security measures are often inadequate in countering such attacks. To address this, the Borda scoring method emerges as a promising solution. Derived from social choice theory, Borda assigns scores based on option rankings in each ballot, providing a resilient mechanism against manipulation. It functions as a consensus mechanism, aggregating opinions from all nodes in the network to ensure fair decision-making. Integrating Borda scoring into fog computing mitigates trust-based attacks like ballot stuffing, enhancing decision-making reliability and security.**

***Index Terms*—Fog Computing, Ballot Stuffing Attacks, Borda Scoring, Trust Management**

I. Introduction

Fog computing has emerged as a paradigmatic shift in the realm of distributed computing, aiming to extend the capabilities of cloud services to the edge of the network. Unlike traditional cloud computing, which centralizes data processing in distant data centers, fog computing distributes computing resources closer to where data is generated, enabling real-time processing, low latency, and improved efficiency. However, with the distributed nature of fog computing comes a host of security challenges that demand innovative solutions to ensure the integrity and confidentiality of data and operations [1].

Security in fog computing encompasses various aspects, including data privacy, integrity, availability, and trustworthiness [2].

With data processing and storage distributed across a network of edge devices, securing these resources against unauthorized access, data breaches, and malicious activities becomes crucial. Moreover, the dynamic nature of fog environments, characterized by frequent device mobility and network topology changes, adds complexity to security management and enforcement. Trust-based attacks are particularly concerning in fog computing. These attacks exploit vulnerabilities in trust relationships within the fog environment, compromising decision-making processes and undermining system trustworthiness. Given the potential impact of such attacks, robust countermeasures are necessary to ensure the integrity and reliability of fog computing systems. Common trust-based attacks include Bad mouthing attack,Good Mothing Attack,Ballot Stuffing Attack, On off attack, self promotion and White Washing Attack. In this paper we focus on mitigating ballot stuffing attack [3].

Mitigating ballot stuffing attacks in fog computing is essential to uphold the integrity and trustworthiness of decision-making processes. Fog computing is pivotal in real-time applications, where compromised decisions could have severe consequences, particularly in safety-critical scenarios. Malicious fog nodes within the network pose a significant threat by exploiting vulnerabilities to conduct ballot stuffing attacks. These attacks not only skew decision outcomes but also erode trust in the system. Given the widespread adoption of IoT devices in fog computing, the potential for manipulation is heightened, underscoring the urgent need for robust countermeasures. Effectively addressing these vulnerabilities not only protects individual systems but also ensures the overall integrity and reliability of the fog computing ecosystem.

II. Related Work

In recent years, there has been a growing interest in trust-based security solutions within both industry and academia. Trust serves as a valuable tool for identifying and isolating malicious nodes within the network, leveraging the legitimate nodes. Furthermore, trust plays a pivotal role in fostering relationships among various fog nodes, ensuring the preservation of privacy and security [4]. Designing a trust management mechanism for fog nodes presents challenges due to its decentralized architecture. The primary issue stems from the decentralized nature, which complicates the collection and management of evidence and behavior crucial for evaluating the trustworthiness of distributed fog nodes [5].The authors in [6]computed the trustworthiness of fog service providers using a range of trust metrics including reliability, security, feedback, and cost but presumed the honesty of user feedback in their assessment.The authors employed a subjective logic mechanism to identify trust-based attacks, yet they did not specify their approach for addressing ballot stuffing attacks [7]. Authors in [8], [9]proposed a multi-source trust evaluation method to identify malicious nodes in the system. They calculate the final trust of IoT users using a weighted sum function of fog node experiences and trust level values from neighboring fog nodes and IoT devices. Trust level values from reputable neighbors are prioritized, determined using a page rank method for IoT devices. However, all neighboring fog nodes are assumed trustworthy. If the final trust value falls below a threshold, the IoT device is flagged as malicious. Moreover, the system detects malicious users transmitting out-of-context data. Yet, the methods are limited by potentially malicious trust level values from neighbors, highlighting the need to assess their credibility. Other approach [10], CTRUST is proposed which incorporates a belief function to gauge a node's concurrence with recommendations from a third-party node regarding another node. However, the framework relies on quantitative analysis, neglecting imprecision and uncertainty inherent in the Fog environment.The authors did not address the issue of ballot stuffing attacks in their discussion. The authors in [11]introduced a hierarchical trust model for sensor-to-cloud applications based on fog computing. This model enables wireless sensor nodes to establish trust relationships among themselves and with cloud service providers and sensor service providers (or fog providers) to calculate trust scores. The authors demonstrated that implementing the fog-enhanced hierarchical structure enables quick detection of malicious nodes, reduces energy consumption, and facilitates the recovery of nodes engaged in bad-mouthing, good-mouthing and ballot stuffing Attack. However there is no proper mention of how they are mitigating the ballot stuffing attack in there approach.

III. Preliminaries

A. Ballot Stuffing Attack

Ballot stuffing attacks in fog computing, analogous to malicious nodes, pose a significant threat to the integrity and reliability of decision-making processes within the network. In traditional ballot stuffing, malicious actors illegitimately influence the outcome of an election by fraudulently casting multiple votes or manipulating the vote count. Similarly, in fog computing, malicious nodes attempt to subvert the decision-making process by dishonestly influencing the aggregation of trust or preference ratings, thereby compromising the accuracy and trustworthiness of decisions. A typical ballot stuffing attack has been depicted in Fig. 1

The consequences of ballot stuffing attacks in fog computing can be severe, particularly in safety-critical applications or environments where decisions impact human safety or operational integrity. For example, in a smart city deployment relying on fog computing for traffic management, a malicious node orchestrating a ballot stuffing attack could manipulate traffic signal timings, leading to congestion, accidents, or other adverse consequences.

B. Borda scoring

Borda scoring is a preferential voting method utilized to determine the collective preference or ranking of a set of options based on individual preferences or rankings. It was first proposed by the French mathematician Jean-Charles de Borda in the 18th century and has since found applications in various decision-making processes, including elections, ranking systems, and consensus-building mechanisms. The fundamental principle behind Borda scoring is to assign scores to each option based on its ranking by individuals. In a typical Borda scoring system, the highest-ranked option receives the maximum score, while the lowest-ranked option receives the minimum score. The scores assigned to intermediate rankings are determined by a predefined scoring scheme, often following an arithmetic progression. For example, in a scenario with n options, the highest-ranked option would receive a score of n, the second-highest-ranked option would receive a score of $n-1$, and so forth, with the lowest-ranked option receiving a score of 1. Once individual rankings are collected, the scores assigned to each option are aggregated across all participants to calculate the total score for each option. The option with the highest total score is deemed the most preferred or ranked the highest by the collective group.

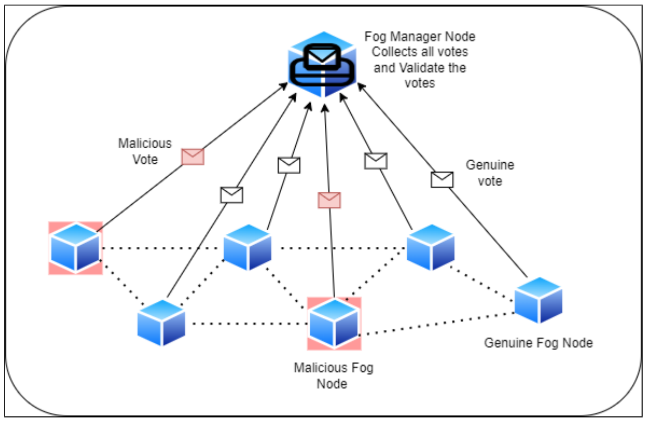


Fig. 1: A Simple Ballot Stuffing Attack Scenario.

Borda scoring is particularly advantageous due to its ability to capture nuanced preferences and provide a comprehensive ranking of options, even in situations with multiple alternatives and diverse individual opinions. It promotes fairness and transparency by considering all preferences and mitigating the influence of outliers or individual biases. Utilizing Borda Scoring Method to prevent Ballot Stuffing Attack : In the context of fog computing and decision-making processes, Borda scoring can be applied to assess the trustworthiness or reliability of nodes based on their interactions and perceived trust levels. By aggregating trust ratings from multiple sources using Borda scoring, a comprehensive assessment of node trustworthiness can be obtained, facilitating informed decision-making and enhancing the security and resilience of fog computing systems.

Utilizing Borda Scoring Method to prevent Ballot Stuffing Attack: In the context of fog computing and decision-making processes, Borda scoring can be applied to assess the trustworthiness or reliability of nodes based on their interactions and perceived trust levels. By aggregating trust ratings from multiple sources using Borda scoring, a comprehensive assessment of node trustworthiness can be obtained, facilitating informed decision-making and enhancing the security and resilience of fog computing systems.

IV. Proposed Methodology

Our methodology is formulated to identify rogue nodes within a fog computing environment systematically. This is accomplished through a series of defined steps, starting with the computation of trust matrices, followed by the analysis of node behavior through voting matrices, and culminating in a systematic identification of potential rogue nodes. Each phase of the proposed methodology is described below.

A. Computation of Trust Matrix

The methodology initiates with the generation of a trust matrix T by the fog manager, modeling the network with a star topology where each node is directly connected to every other node. The trust values for each node are initialized at a baseline level, and the system simulates multiple rounds of node interactions, adjusting trust scores based on the outcomes of these interactions. This simulation helps in establishing a comprehensive trust matrix that encapsulates the trustworthiness of each node.

B. Identification and Management of Rogue Nodes

In adherence to the principles of Byzantine fault tolerance, the system constrains the number of rogue nodes to a fraction of the total nodes to ensure network integrity. A scenario is set up where certain nodes are hypothesized as rogue to demonstrate the detection process. The system employs methods to identify discrepancies that could indicate manipulations typical of rogue nodes.

C. Populating and Analyzing the Voting Matrix

The fog manager initially employs a ranking method to evaluate nodes based on their performance metrics, documented in a voting matrix B. In the event of manipulation, a secondary manipulated voting matrix B' is produced, reflecting the distorted rankings. By comparing the original and manipulated matrices, potential attacks are identified. This comparison is pivotal for determining whether shifts in node rankings represent genuine security threats.

D. Systematic Identification of Rogue Nodes

Following the identification of rank discrepancies, a suspect matrix is formulated, listing all nodes along with the rank changes they induce in others. A scoring system evaluates each node based on the magnitude of these changes, disregarding the direction of change. Nodes with the highest cumulative scores are flagged as potential rogues, indicating a high probability of disruptive behavior.

E. Periodic Monitoring and Reevaluation

The methodology emphasizes the need for periodic monitoring and reevaluation to maintain network integrity. Stable networks show consistent node rankings over time, while the emergence of new discrepancies could signal potential security threats. This ongoing assessment is crucial for adapting to new threats and maintaining the operational efficacy of the system.

This structured and iterative approach ensures continuous monitoring and the robust detection of rogue nodes, thereby safeguarding the network against internal disruptions and external attacks.

Algorithm 1 Proposed Borda Mitigates Ballot Stuffing Attack Algorithm (BOMBAT)

```
1: **Input:** Nodes $N$, Initial trust level $\tau_{init}$
2: **Output:** List of rogue nodes $R$
3: **procedure** ComputeTrust($N$, $\tau_{init}$)
4:     Initialize trust matrix $T$ with $\tau_{init}$
5:     **for** $i \leftarrow 1$ to $num\_rounds$ **do**
6:         Simulate communication, update $T$
7:     **end for**
8:     **return** $T$
9: **end procedure**
10: **procedure** DetectRogues($T$)
11:     Initialize Borda count $B$ and manipulated Borda $B'$
12:     $B \leftarrow$ Rank nodes based on $T$
13:     $B' \leftarrow$ Simulate attack, manipulate $B$
14:     Calculate discrepancies $D$ between $B$ and $B'$
15:     **return** Nodes with significant discrepancies in $D$
16: **end procedure**
17: **procedure** CalculateScores($D$)
18:     Initialize score vector $S$
19:     **for** each $n \in N$ **do**
20:         $S[n] \leftarrow$ Sum of discrepancies affecting $n$
21:     **end for**
22:     Sort $N$ by $S$, identify top suspects in $R$
23:     **return** $R$
24: **end procedure**
25: **procedure** Main
26:     $T \leftarrow$ ComputeTrust($N$, $\tau_{init}$)
27:     $suspects \leftarrow$ DetectRogues($T$)
28:     $rogueNodes \leftarrow$ CalculateScores($suspects$)
29:     **return** $rogueNodes$
30: **end procedure**
```

V. Experimental Results

The subsequent paragraphs detail various settings and the results achieved at each stage of the process undertaken.

- **Computation of Trust Matrix:** The initial input for our proposed approach is the trust matrix, which is generated by the fog manager. We model the network as a star topology, enabling each node to communicate with all other nodes directly. The trust values, ranging between 0 and 1, are initially set at 0.5 for each node. We simulate 200 rounds of communication between the nodes using Matlab. During each communication round, there is a randomly assigned 50% chance for either successful communication or a packet drop. Each successful communication increases a node's trust score by 0.0025, and similarly, it decreases with each packet drop. After completing 200 rounds, the final trust matrix is obtained. It should be noted that extensive research has been conducted on the generation of trust matrices using various methodologies as described in literature [12], [13], and any of these methods could potentially be

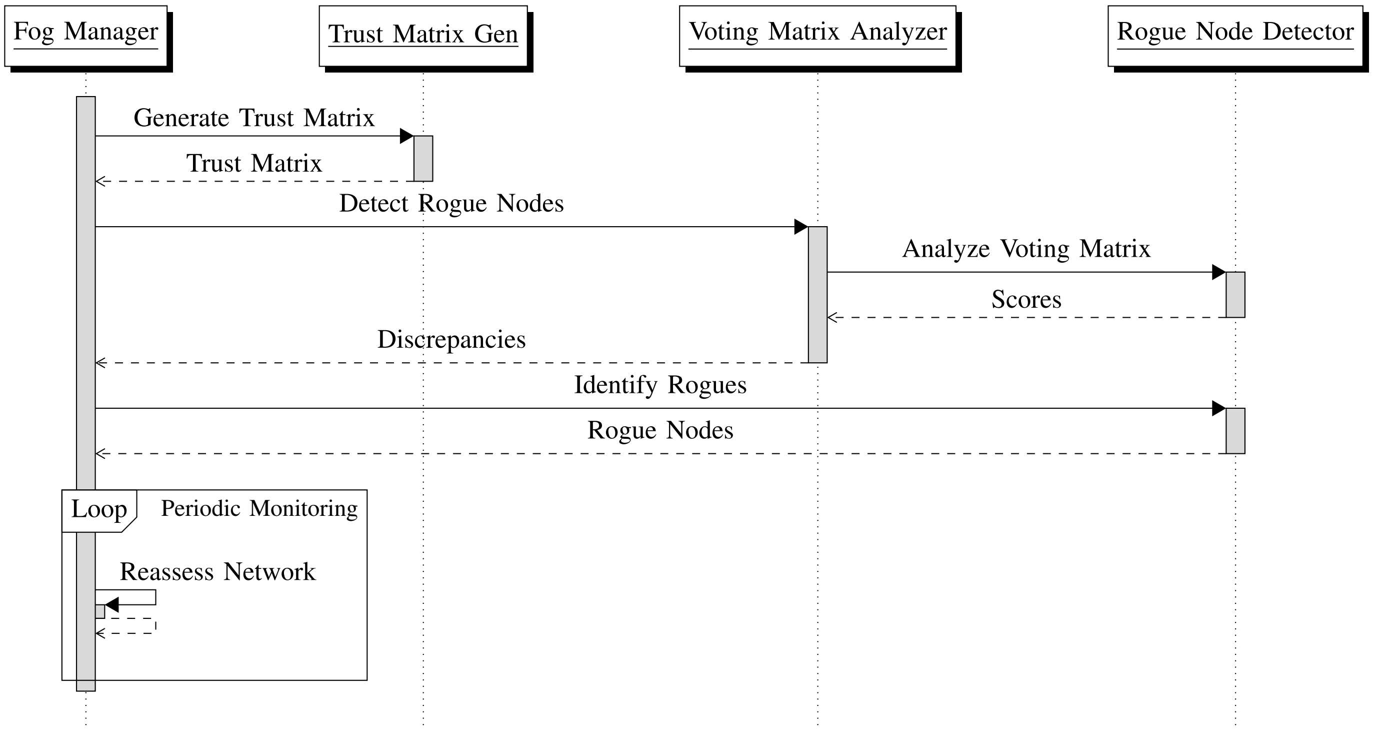


Fig. 2: Sequence Diagram for the proposed BOMBAT methodology.

employed to construct this trust matrix. A sample trust matrix for six nodes is depicted in Fig. I.

TABLE I: Trust Matrix

	N1	N2	N3	N4	N5	N6
N1	0.0000	0.4450	0.4750	0.5200	0.5300	0.4400
N2	0.3950	0.0000	0.5200	0.5800	0.4550	0.5100
N3	0.5400	0.5000	0.0000	0.5250	0.4400	0.5000
N4	0.4400	0.5300	0.4700	0.0000	0.4950	0.5000
N5	0.4650	0.4750	0.4950	0.4700	0.0000	0.4700
N6	0.5350	0.5000	0.5500	0.4500	0.4400	0.0000

- **Number of Rogue Nodes :** According to the Byzantine principle [14], the maximum number of rogue nodes is limited to less than one-third of the total nodes within the system. Thus, in our fog environment comprising six nodes, a maximum of one rogue node is permissible. For this toy example we consider N3 and N5 to be rogue nodes and we shall show how these nodes will be identified through the proposed approach.
- **Populating the Voting Matrix :** : Initially, the fog manager employs the Borda scoring technique to gauge node performance within the network, as illustrated in Table II. Subsequently, in the event of rogue nodes conducting a ballot stuffing attack, the rankings of the nodes might be altered. It is assumed that rogue nodes manipulate the rankings—assigning lower ranks to highly trusted nodes and vice versa, thereby modifying the overall Borda score and the resulting rankings.
  This scenario is depicted in Table III. At any given moment, the fog manager has access to these two sets of rankings and can determine whether an attack has occurred by comparing them. Absence of rank changes indicates no attack, whereas discrepancies signify an attack. Identifying the rogue nodes, however, poses the greatest challenge for the fog manager. To address this, the rank differences for each node are calculated. For instance, if node N1 is ranked 1 before an attack and 3 afterward, the rank difference is -2. Such differences are calculated for all nodes. Additionally, the nodes suspected of manipulating rankings are identified.

TABLE II: Voting Matrix with Borda Scores before Attack

	N1	N2	N3	N4	N5	N6	Borda Scores	Node Ranks
N 1	0	1	2	3	5	1	12	1
N 2	1	0	4	5	3	5	18	6
N 3	5	3	0	4	1	3	16	3
N 4	2	5	1	0	4	4	16	4
N 5	3	2	3	2	0	2	12	2
N 6	4	4	5	1	2	0	16	5

TABLE III: Voting Matrix with Borda Scores and Node Ranks After Attack

	N1	N2	N3	N4	N5	N6	Borda Scores	Node Ranks
N1	0	1	4	3	5	1	14	3
N2	5	0	2	5	3	5	20	2
N3	1	3	0	4	1	3	12	4
N4	4	5	5	0	4	4	22	1
N5	3	2	3	2	0	2	12	5
N6	2	4	1	1	2	0	10	6

Following the Byzantine principle, no more than one-third of the nodes are selected as suspects. Table IV lists the differences in ranks along with the suspected nodes for each node.

- **Identifying Rogue Nodes :** Upon completion of the rank differentiation and the compilation of the suspect matrix, a systematic scoring mechanism is implemented to accurately identify potential rogue nodes within the network. This scoring process involves an in-depth analysis of each node's frequency of appearance on various suspect lists, which are derived from the initial data aggregation. For illustration, refer to Table IV, where Node N3 is implicated by Nodes N1, N2, N4, N5, and N6. The rank changes associated with these implications are recorded as 2, -1, 2, 1, and -2, respectively. In the scoring process, the absolute values of these differences are utilized to negate any directional bias, focusing solely on the magnitude of change, which culminates in a cumulative score of 8 for Node N3. his methodical approach to scoring is uniformly applied across all nodes within the network.

TABLE IV: Rank Change for Each Node and Suspect List of Nodes

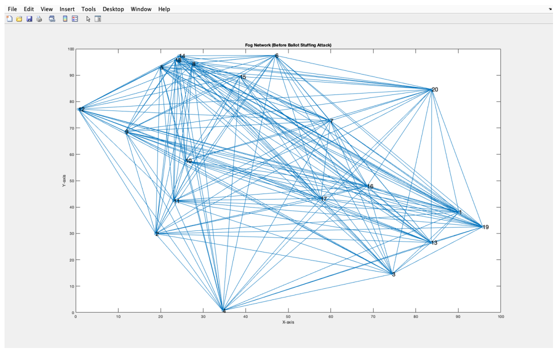
Node	**Rank Change**	**Suspect List of Nodes**
N1	2	N5, N3
N2	-1	N3, N5
N3	-2	N1, N5
N4	2	N2, N3
N5	1	N1, N3
N6	-2	N3, N4

TABLE V: Suspect Scores of Each Node

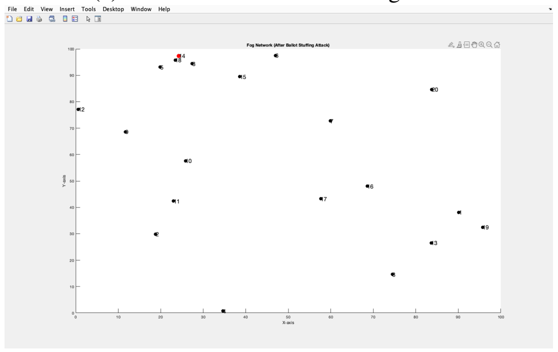
	N1	**N2**	**N3**	**N4**	**N5**	**N6**
Total Occurrences	3	2	8	2	5	0

The results are shown in Table V, which facilitates an objective assessment of which node accumulates the highest score—indicative of the highest level of suspicion and potential disruption to the network's operation. In the presented scenario, Node N3 has the highest score, confirming it as the primary suspect. This substantiates the preliminary hypothesis regarding N3's anomalous behavior, thus identifying it as the potential rogue node.

- **Simulation with 20 Nodes :** We have also experimented with 20 nodes having a maximum of 6 rogue nodes. Table VI summarizes the borda scores before and after the attack. The rank difference, suspect nodes and finally the suspect scores. As we can observe from the table, N14 is considered as the rogue node which is part of initial assumptions. It is important to note that since the fog manager does not have a count of rogue nodes to be eliminated, the process needs to performed iteratively to remove all the rogue nodes. Simulation of node interactions and identification of N14 as rogue node can also be observed from the simulated matlab environment depicted in Fig. 3a.
To ensure the ongoing security and reliability of the network, this analytical procedure should be periodically repeated. Consistent monitoring and reevaluation are crucial for identifying emergent threats or behavioral anomalies among the nodes. A network exhibiting no significant rank differences over time is indicative of stability and absence of internal disruptions. Conversely, the appearance of rank differences signals potential security breaches or attacks, necessitating immediate investigative and corrective actions. It can observed that this approach is robust and also scalable which is important for a for environment.



(a) Interactions between 20 fog nodes



(b) Identification N14 as rogue node post analysis

Fig. 3: Interactions and Detection of Rogue Nodes through BOMBAT

VI. Conclusion

The proposed methodology, implemented and tested in a fog computing environment, effectively identifies rogue nodes through the computation of trust matrices and the analysis of voting discrepancies. The experimental results demonstrated the robustness of the trust matrix generation, employing a star topology where nodes communicated in a simulated setting with dynamic trust adjustments based on the outcomes of these interactions. This foundational step allowed us to capture the essential trust relationships within the network accurately.

Subsequent phases of the methodology leveraged these trust matrices to detect anomalies in node behavior through a sophisticated analysis of voting matrices, further refined by comparing original and manipulated rankings. The detection process was validated using a hypothetical scenario involving rogue nodes, which manipulated rankings to disrupt network integrity. Our approach successfully identified these nodes by calculating discrepancies in rankings and applying a scoring system that quantified the impact of each node's actions on the network's trustworthiness.

TABLE VI: Suspect Scores of Each Node

Nodes	Borda score before attack	Borda score after attack	Rank Difference	Suspect Nodes						Suspect Score
N1	203	261	7	6	12	8	14	19	5	16
N2	187	225	8	17	3	18	5	6	13	21
N3	197	235	7	14	1	11	16	20	7	24
N4	192	160	-8	6	2	14	15	8	18	13
N5	163	167	4	8	9	12	2	11	18	22
N6	146	166	5	2	9	8	12	15	14	31
N7	220	182	-9	18	19	14	20	8	11	16
N8	168	184	5	13	19	14	15	18	1	45
N9	204	194	-1	13	5	12	19	14	18	19
N10	209	201	-1	15	8	5	12	3	16	20
N11	182	200	6	4	7	3	9	19	10	27
N12	206	204	1	4	5	11	2	20	3	20
N13	226	194	-8	18	15	8	10	14	19	21
N14	**210**	**190**	**-6**	**3**	**19**	**15**	**6**	**17**	**13**	**55**
N15	167	161	0	0	0	0	0	0	0	33
N16	221	229	-1	2	18	4	9	11	14	12
N17	191	173	-2	11	20	10	7	8	2	16
N18	180	162	-2	12	6	9	11	17	3	44
N19	153	155	-1	8	9	7	11	13	4	42
N20	175	157	-4	4	10	14	16	1	5	19

This systematic identification and scoring mechanism significantly contribute to the field of fog computing security, providing a scalable and efficient method to maintain network integrity. By adhering to the principles of Byzantine fault tolerance, our methodology ensures that the network can sustain a limited number of rogue nodes without compromising its overall functionality.

A. Future Directions

Moving forward, there are several avenues for enhancing this research. Firstly, integrating machine learning techniques to predict node behavior based on historical data could provide preemptive security measures, potentially stopping attacks before they start. Additionally, exploring the application of blockchain technology could offer a decentralized and transparent method for managing trust and voting records, further securing the network against tampering.

Another promising direction involves extending the methodology to more complex network topologies and larger scales, which would test the robustness and scalability of the current approach in more dynamic and unpredictable environments. Finally, conducting real-world tests in commercial fog computing networks would provide valuable insights into the practical challenges and performance implications of implementing such security measures. These enhancements and expansions could significantly improve the resilience of fog computing networks, ensuring their reliability and security in the face of evolving cyber threats.

References

[1] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, 2012, pp. 13–16.

[2] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in *Wireless Algorithms, Systems, and Applications: 10th International Conference, WASA 2015, Qufu, China, August 10-12, 2015, Proceedings 10*. Springer, 2015, pp. 685–695.

[3] A. K. Junejo, N. Komninos, M. Sathiyanarayanan, and B. S. Chowdhry, "Trustee: A trust management system for fog-enabled cyber physical systems," *IEEE transactions on emerging topics in computing*, vol. 9, no. 4, pp. 2030–2041, 2019.

[4] Y. Wang, Y. Xiang, J. Zhang, W. Zhou, G. Wei, and L. T. Yang, "Internet traffic classification using constrained clustering," *IEEE transactions on parallel and distributed systems*, vol. 25, no. 11, pp. 2932–2943, 2013.

[5] J. Ni, K. Zhang, X. Lin, and X. Shen, "Securing fog computing for internet of things applications: Challenges and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 601–628, 2017.

[6] F. H. Rahman, T.-W. Au, S. S. Newaz, W. S. Suhaili, and G. M. Lee, "Find my trustworthy fogs: A fuzzy-based trust evaluation framework," *Future Generation Computer Systems*, vol. 109, pp. 562–572, 2020.

[7] J. Al Muhtadi, R. A. Alamri, F. A. Khan, and K. Saleem, "Subjective logic-based trust model for fog computing," *Computer Communications*, vol. 178, pp. 221–233, 2021.

[8] Y. Hussain and Z. Huang, "Trfiot: Trust and reputation model for fog-based iot," in *Cloud Computing and Security: 4th International Conference, ICCCS 2018, Haikou, China, June 8-10, 2018, Revised Selected Papers, Part VI 4*. Springer, 2018, pp. 187–198.

[9] Y. Hussain, H. Zhiqiu, M. A. Akbar, A. Alsanad, A. A.-A. Alsanad, A. Nawaz, I. A. Khan, and Z. U. Khan, "Context-aware trust and reputation model for fog-based iot," *IEEE Access*, vol. 8, pp. 31 622–31 632, 2020.

[10] A. A. Adewuyi, H. Cheng, Q. Shi, J. Cao, Á. MacDermott, and X. Wang, "Ctrust: A dynamic trust model for collaborative applications in the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5432–5445, 2019.

[11] T. Wang, G. Zhang, M. Z. A. Bhuiyan, A. Liu, W. Jia, and M. Xie, "A novel trust mechanism based on fog computing in sensor–cloud system," *Future Generation Computer Systems*, vol. 109, pp. 573–582, 2020.

[12] J. Caminha, A. Perkusich, and M. Perkusich, "A smart middleware to perform semantic discovery and trust evaluation for the internet of things," in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2018, pp. 1–2.

[13] W. Najib, S. Sulistyo *et al.*, "Survey on trust calculation methods in internet of things," *Procedia Computer Science*, vol. 161, pp. 1300–1307, 2019.

[14] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *OsDI*, vol. 99, no. 1999, 1999, pp. 173–186.