Educational Administration: Theory and Practice
2024, 30(5), 5051-5062
ISSN:2148-2403
https://kuey.net/

Research Article

A Novel Mobile Agent-Based Intrusion Detection Framework For Network Security Using Sl-Gat And Pp-FQCC

Rajendra Singh Kushwah[1*], Kamal Kishor Prasad[2]

[1*]Professor, Dept. of Computer Science and Engineering, Sri Satya Sai University of Technology and Medical Sciences, Sehore. Email : rajendrasingh.ind@gmail.com
[2]Research scholor, Sri Satya Sai University of Technology and Medical Sciences, Sehore. Email : mr.kamalprsd@gmail.com

Citation: Rajendra Singh Kushwah, et al (2024), A Novel Mobile Agent-Based Intrusion Detection Framework For Network Security Using Sl-Gat And Pp-Fqcc, *Educational Administration: Theory and Practice*, *30*(5), 5051-5062, Doi:10.53555/kuey.v30i5.3748

ARTICLE INFO

ABSTRACT

Mobile Agent is known as a software component that collects the data from hosts in the network. However, security is a major problem in MA. Therefore, IDS is used to detect malicious activities in the network. But, none of the works validated whether the MA is cloned or real. Therefore, the paper presents an MA-based IDS framework for network security using SL-GAT and PP-FQCC. Firstly, the MA and host are registered with the centralized server; in the meantime, UUID is generated for the MA. Afterward, MA is securely localized by using CMP-GAO. Then, MA is authorized and the data is secured. At this point, IDS checks whether the data is attacked or not. Here, IDS is trained based on pre-processing, graph construction, and classification. Finally, the classifier classifies whether the data are attacked or non-attacked. The results proved that the proposed model achieved a high-security level of 98.87%, which outperformed prevailing techniques.

Keywords: Intrusion Detection System (IDS), Mobile Agent (MA), Conway-Maxwell-Poisson-Giant Armadillo Optimization (CMP-GAO), Transposition Cipher–Message Authentication Code (TC-MAC), Partial Public key-based-FourQ Curve Cryptography (PP-FQCC), K-Nearest Neighbors (KNN), Smish Logish-Graph Attention Network (SL-GAT), and Universally Unique Identifier (UUID).

1. INTRODUCTION

With the rapid development of mobile edge computing, mobile data traffic has increased (Garg et al., 2021). So, security is the major concern while sharing data packets through the internet (Lai et al., 2021). Denial of Service (DoS) and zero-day exposure attacks are the most common security threats (Cao et al., 2020). So, paying attention to security attacks is important to avoid data loss and tampering (Sun, 2022). Therefore, IDS is used to detect the anomaly actions in the network (Nie et al., 2022). An IDS is a first-level security system, which works on a per-packet basis (Krishnan et al., 2020).

The existing studies used some anomaly detection methods, such as support vector machines, Gated Recurrent Unit (GRU), and hybrid techniques with signature-based schemes to detect intrusions in the network. But, these models are prone to overfitting problems (Ramaiah et al., 2021, Liu et al., 2021). Also, some authorization schemes, such as fine-grained access control mechanisms and symmetric and asymmetric cryptographic algorithms are used to ensure the security of the networks (Sindjoung et al., 2023, Hou et al., 2020). However, these models are inaccurate in providing network security. Therefore, an efficient model called MA-based IDS framework using SL-GAT and PP-FQCC has been proposed to improve network security.

1.1 Problem Statement

Some limitations of the existing techniques are described below,

* None of the existing works validated whether the MA is cloned or real.
* (Ren et al., 2023) Existing studies do not concentrate on secure localization processes.
* (Singh et al., 2022) In existing works, the data prevention process is not performed in IDS.

* MA-based IDS generates more false positives and false negatives.

1.2 Objectives

The key objectives of the proposed model are given below,

- ❖ TC-MAC is utilized to validate the MA.
- ❖ CMP-GAO is introduced to securely relocate the MA.
- ❖ PP-FQCC is suggested to secure the data in the IDS.
- ❖ KNN graph and SL-GAT classifier are used to reduce the false positives and negatives.

The rest of this paper is organized as: section 2 illustrates the related works and their limits, section 3 summarizes the proposed system, section 4 conveys the results and discussion, and section 5 concludes the proposed work with future recommendations.

2. LITERATURE SURVEY

(Ren et al., 2023) established a Multi-Agent Feature Selection IDS (MAFSIDS) for attack detection in networks. The GCN (Graph Convolutional Network) was used for extracting deep features from the data. The features were extracted accurately using MAFSIDS. However, the edges of computing were not localized, which led to attack detection problems during data transfer.

(Wang et al., 2020) deployed IDS for Empowered Intruders (IDEI) in wireless networks. The mobile nodes were tracked using the IDEI method. The mobile service nodes helped in detecting the intrusion. Thus, the model obtained better intrusion detection. Yet, the nodes were insufficient, which resulted in improper attack detection.

(Chen et al., 2020) presented federated learning-based IDS for wireless edge networks. Here, the Federated Learning-based Attention GRU (FedAGRU) was used for intrusion detection. Hence, the intrusion detection was improved accurately. On the contrary, the FedAGRU was affected by poison attacks, thus reducing the attack detection rates.

(Parsamehr et al., 2020) evaluated IDS for Network Coding-enabled (NC) mobile cells. Here, the detection was based on homomorphic MAC schemes. The Intrusion Detection and Location-Aware Prevention (IDLP) mechanism identified the attacker's exact location and blocked it to prevent an attack. Hence, the NC mobile cell depletion was reduced. But, the data could not be decoded properly, which made the model unsuccessful.

(Gong et al., 2021) introduced a two-phase algorithm for cyber intrusion detection in edge computing. Here, the selected features were classified using the Modified Back-Propagation Neural Network (MBPNN). Hence, a higher detection rate was achieved. Yet, the MOGA method could not get enough features, which affected the performance of attack detection.

3. PROPOSED MA BASED IDS

In the proposed model, the SL-GAT classifier is used to detect security threats, and the PP-FQCC algorithm is used to secure the data. The structural diagram of the proposed system is shown in Figure 1.

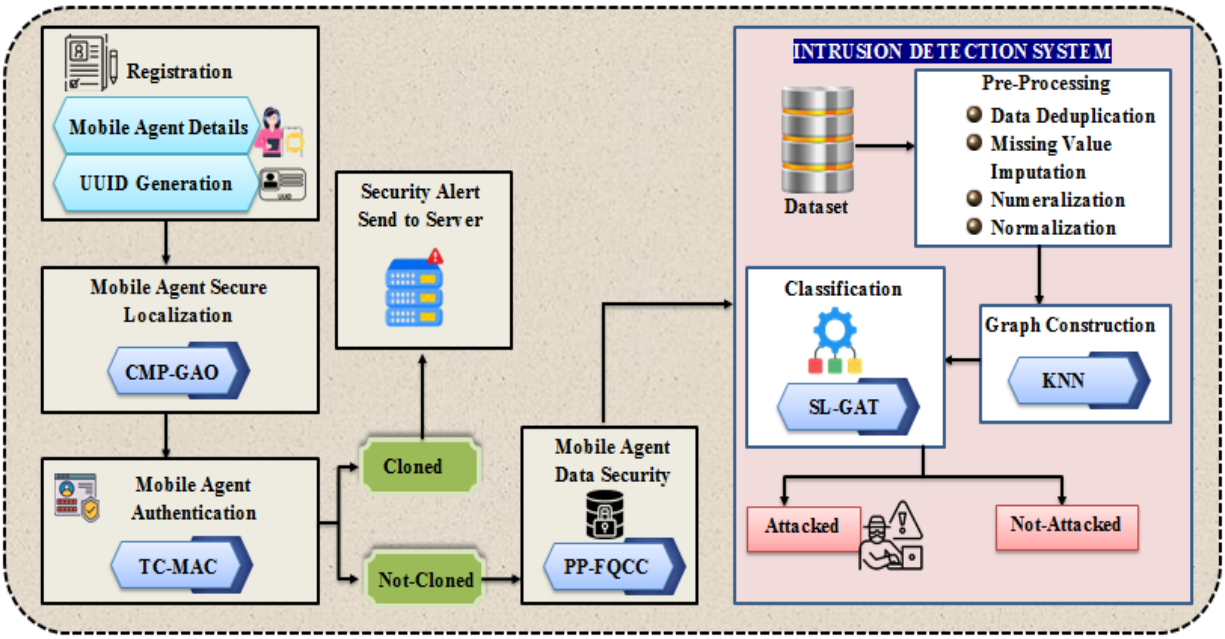

Figure 1: Structural diagram of the proposed model

3.1 Registration

Initially, the MA and the host are registered with the centralized server. MA details, such as trust value, energy, and distance are collected for registration. In the meantime, partial public β and private keys ξ_ℓ are generated for hosts. The key is generated using the PP-FQCC algorithm. Afterward, UUID is generated for the MA. The registered MA χ_{ma} and hosts Θ_{ho} are expressed as,

$$(\chi_{ma}, \Theta_{ho}) = \left(\chi_1, \Theta_1 + \chi_2, \Theta_2 + \chi_3, \Theta_3 + \ldots\ldots + \chi_i, \Theta_a\right) \quad (1)$$

Where, χ_i, Θ_a indicate the total numbers of χ_{ma} and Θ_{ho}.

3.2 MA Secure Localization

Then, the MA is securely localized based on the minimum distance and high trust value of the source and destination hosts. Here, the CMP-GAO algorithm is utilized for secure localization. GAO provides suitable solutions to optimization problems. But, premature convergence problems occurred due to the random position updation. So, CMP is replaced to overcome these issues. Primarily, the population N of Giant Armadillos (GA) is initialized. The N is considered as the registered MA, which is expressed as,

$$N(\chi_{ma}, \Theta_{ho}) = \begin{bmatrix} N_1 \\ \vdots \\ N_p \\ \vdots \\ N_T \end{bmatrix}_{T \times o} = \begin{bmatrix} n_{1,1} & \cdots & n_{1,q} & \cdots & n_{1,o} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ n_{p,1} & \cdots & n_{p,q} & \cdots & n_{p,o} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ n_{T,1} & \cdots & n_{T,q} & \cdots & n_{T,o} \end{bmatrix}_{T \times o} \quad (2)$$

$$n_{p,q} = low_q + R_{p,q} \bullet (up_q - low_q) \quad (3)$$

Where, $n_{p,q}$ indicates the q^{th} dimension of the problem, N_p is the p^{th} GA, which is the candidate solution, T is the total number of GA, o is the total number of problem variables, $p = 1,2,3,\ldots,T$ and $q = 1,2,3,\ldots,o$, $R_{p,q}$ denote random parameters, and low_q and up_q indicate the lower and upper bound, respectively. The fitness function I is calculated by considering the minimum distance $\min(d')$ and high trust value $\max(c)$,

$$I = \max(c) * \min(d') * \begin{bmatrix} I_1 \\ \vdots \\ I_p \\ \vdots \\ I_T \end{bmatrix}_{T \times 1} = \begin{bmatrix} I(N_1) \\ \vdots \\ I(N_p) \\ \vdots \\ I(N_T) \end{bmatrix}_{T \times 1} \quad (4)$$

$$N_{best} = Best(I) \quad (5)$$

$Best(I)$ is the best fitness value for the best candidate solution N_{best}. In the exploration phase, the position is updated based on the attack of GA towards the termite mounds during hunting. Here, the CMP is replaced in the random position updation to make the exploration easier. For each GA, the set of selected Candidate Termite Mounds (CTM) V_i is described as,

$$V_i = \frac{N}{\sum_{p=0}^{\infty} \frac{I_p}{(p!)}(p!)} \quad (6)$$

Here, I_p indicates the fitness of the p^{th} GA and p indicates the number of GA. The GA randomly selects one of the CTMs for hunting. The updated new position $n_{p,q}^{new}$ is formulated as,

$$n_{p,q}^{new} = n_{p,q} + R_{p,q} \bullet (V_i - rn_{p,q} \bullet n_{p,q}) \quad (7)$$

$$N_p = \begin{cases} if \;\; I_p^{new} < I_p & n_{p,q}^{new} \\ else & N_p \end{cases} \quad (8)$$

Where, $rn_{p,q}$ is the random number with set{1,2} and I_p^{new} is the fitness function of $n_{p,q}^{new}$. In the exploitation phase, the position is changed based on GA digging in termite mounds to feed on termites. The new suitable position $n_{p,q}^{Sp}$ is defined as,

$$n_{p,q}^{Sp} = n_{p,q} + \frac{low_q + R_{p,q} \bullet (up_q - low_q)}{iter} \quad (9)$$

$$N_p = \begin{cases} if \;\; I_p^{Sp} < I_p & n_{p,q}^{Sp} \\ else & N_p \end{cases} \quad (10)$$

Where, $iter = 1,2,3,\ldots, iter'$ is the total number of iterations. This process is continued until the last iteration. Finally, the relocated MA L_π is expressed as,

$$L_\pi = \{L_1, L_2, L_3, \ldots\ldots, L_e\} \quad Where \quad \pi = 1,2,3,\ldots, e \quad (11)$$

Where, L_e is the total number of L_π.

Pseudocode for CMP-GAO

Input: Registered MA (χ_{ma}, Θ_{ho})

Output: Relocated MA L_π

Begin
Initialize population N
While $iter \le iter'$
For each N
Compute dimension $n_{p,q}$
Estimate $N_{best} = Best(I)$
Update new position
If $I_p^{new} < I_p$
$n_{p,q}^{new}$
Else
N_p
End If
Update new suitable position
If $I_p^{Sp} < I_p$
$n_{p,q}^{Sp}$
Else
N_p
End If
End For
End While
End

Next, the relocated MA is authorized, which will be briefly explained in the following section.

3.3 MA Authorization

Next, the L_{π} are authorized by using TC-MAC. MAC algorithm efficiently prevents the servers and data packets from man-in-the-middle attacks. MACs don't secure the messages against intruders. On the sender side, the message G is converted into a ciphering format using TC. The ciphered text β' is given below,

$$\beta'(G) = V_z + F(J_z + K_{z+1}) \quad (12)$$

Where, V_z and J_z are the left and right-hand parts of the intermediate cipher, respectively and K_{z+1} denotes the key bits. Then, the secret key ϑ is generated by using the β, which is expressed as,

$$\vartheta \xrightarrow{secret\ key} (\beta) \quad (13)$$

After that, MAC (U) is created on the sender side. β' and ϑ are given as an input to MAC and is defined as,

$$U = \beta' \oplus \vartheta \quad (14)$$

Similarly, the created MAC (E) at the receiver side is checked with (U) for similarity (γ), which is defined as,

$$\gamma = \begin{cases} if\ (U = E) & not-cloned \\ else & cloned \end{cases} \quad (15)$$

If MA is cloned, then a security alert is sent to the centralized server; otherwise, the data Ω_{φ} is secured.

3.4 Data security

Then, the data Ω_{φ} is secured by using the PP-FQCC algorithm. ECC speeds up the key generation, encryption, and decryption process. But, the computation is complex due to the negative points in the variables. So, the FourQ curve is used to overcome these issues. At the time of data security, the key is fully generated. Initially, FourQ curves δ are discovered for generating the keys and are defined as,

$$\delta = x_1 + x_2 a' + x_3 b' + x_4 a' b' \quad (16)$$

Where, a' and b' indicate the horizontal and vertical axis and x, x_1, x_3 and x_4 are constants. Next, the keys are generated from the FourQ curves. The fully generated public key α is formulated as,

$$\alpha \to (P_{\ell}) * Q \quad (17)$$

Where, P_{ℓ} indicates fully generated private key. Then, the data are encrypted, which is derived as,

$$C1 = r \times Q \quad (18)$$

$$C2 = \Omega_{\varphi} + r * (\alpha) \quad (19)$$

Where, $C1$ and $C2$ denote the encrypted data and r defines the random number. Finally, the decryption is performed to decrypt the data, which is expressed as,

$$\Omega_{\varphi} = (C2 - (P_{\ell})) \times C1 \quad (20)$$

Where, Ω_{φ} is the decrypted data.

3.5 IDS

Here, the IDS checks whether the data is attacked or not.

3.5.1 Dataset

The IDS takes UNSW_NB15 datasets to train the classifier, which contains the data, such as service, synack, smean, sload, and sloss. The data in the dataset Φ^u is expressed as,

$$\Phi^u = [\Phi^1, \Phi^2, \Phi^3, \ldots\ldots, \Phi^{\varepsilon}] \quad Here \quad u = 1,2,3,\ldots,\varepsilon \quad (21)$$

Where, Φ^1 is the first data in Φ^u.

3.5.2 Pre-processing

Next, the Φ^u are pre-processed to improve the quality. Firstly, the duplicate data are eliminated. Then, the missing values in Φ^u are replaced randomly, which is defined as Γ. After that, numeralization H_∞° is performed to convert the string data into numerical values, which are expressed as,

$$H_\infty^\circ(\Gamma) = (H_1^\circ + H_2^\circ + H_3^\circ + \ldots\ldots + H_g^\circ) \quad where \quad \infty = 1,2,\ldots,g \tag{22}$$

Then, H_∞° is normalized, which converts all the numbers into 0 or 1. The normalized data M_ν is defined as,

$$M_\nu(H_\infty^\circ) = \frac{H_\infty^\circ - \min(H_\infty^\circ)}{\max(H_\infty^\circ) - \min(H_\infty^\circ)} \tag{23}$$

Finally, the pre-processed data $\Im^\hbar$ is determined as,

$$\Im^\hbar(M_\nu) = \{\Im^1, \Im^2, \Im^3, \ldots\ldots, \Im^Z\} \quad \hbar = 1,2,\ldots,Z \tag{24}$$

Where, $\Im^2$ is the second $\Im^\hbar$.

3.5.3 Graph Construction

Next, the graph is constructed for the $\Im^\hbar$ by using KNN graph algorithm. Primarily, the $\Im^\hbar$ is initialized and given as an input in this process. This process decides the number of clusters by grouping them based on the protocols. The centroids B_l with the total j number of centroids are selected randomly and are expressed as,

$$B_l = \{B_1, B_2, B_3, \ldots, B_j\} \quad where \quad l = 1,2,\ldots,j \tag{25}$$

Then, the Euclidean distance D is calculated, which is defined as,

$$D(B_l, \Im^\hbar) = \left[\sum_{l=1,\hbar=1}^{j,Z} (B_l - \Im^\hbar)\right]^{\frac{1}{2}} \tag{26}$$

Next, a new centroid Π_∂ is computed by calculating the average for all data, which is expressed as,

$$\Pi_\partial = \frac{\sum_{l=1,\hbar=1}^{j,Z} A_l^{\hbar}\, \Im^\hbar}{\sum_{l=1,\hbar=1}^{j,Z} A_l^{\hbar}} \tag{27}$$

Where, $A_l^{\hbar}$ is the average of data. These steps are continued until the best clusters are found. The total ab number of constructed graphs is formulated as,

$$\psi_\kappa = (\psi_1, \psi_2, \psi_3, \ldots\ldots, \psi_{ab}) \quad where \quad \kappa = 1,2,\ldots,ab \tag{28}$$

Here, ψ_κ denotes the constructed graph.

3.5.4 Classification

Afterward, the classifier detects whether the data is attacked or not attacked by using SL-GAT. GAT superiorly performs in the data described in graphs. But, it adds more weight parameters, so training time is increased. So, the SL activation function is added to provide efficient classification. The architectural diagram of the proposed classifier is shown in Figure 2.

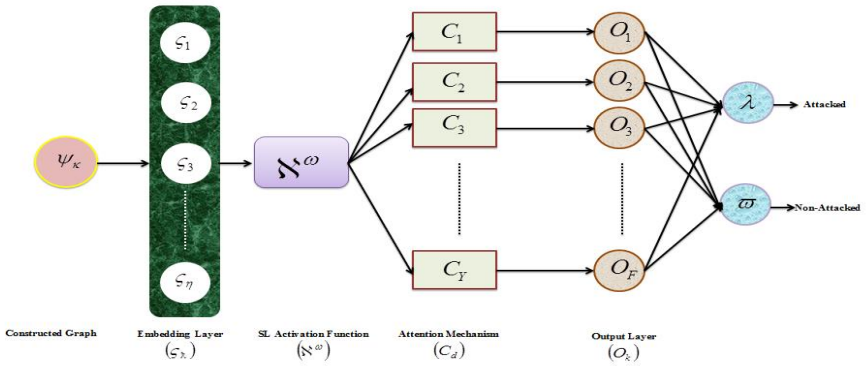

Figure 2: Architectural diagram of SL-GAT

Initially, (ψ_κ) is fed to the embedding layer. The total η number of embedding layers ς_{λ} encoded (ψ_κ) into low dimensional space, which is expressed as,

$$\varsigma_{\lambda} = \{\varsigma_1, \varsigma_2, \varsigma_3, \ldots\ldots, \varsigma_\eta\} \quad where \quad \lambda = 1,2,3,\ldots,\eta \tag{29}$$

Then, the output of ς_{λ} is subjected to the SL activation function $\aleph^{\omega}$ along with the self-attention mechanism. The $\aleph^{\omega}$ reduces the training time of the long sequence inputs, which is derived as,

$$\aleph^{\omega}(\varsigma_{\lambda}) = \phi(\varsigma_{\lambda}) + (\varsigma_{\lambda} - f(\varsigma_{\lambda})\,\phi(\varsigma_{\lambda})) \bullet \frac{\varsigma_{\lambda} \cdot e^{-\varsigma_{\lambda}} (\sigma(\varsigma_{\lambda}))^2}{1 + \sigma(\varsigma_{\lambda})} \tag{30}$$

Where, f and ϕ represent the linear functions and σ indicates the sigmoid function. Then, the attention mechanism C_d calculated with weights W_t and SL is equated by,

$$C_d(\aleph^{\omega}) = \frac{\exp(\aleph^{\omega}(h^{\to T'}[W_t \,\|\, W_t]))}{\sum_{t \in v} \exp(\aleph^{\omega}(h^{\to T'}[W_t \,\|\, W_t])} \tag{31}$$

Where, $h^{\to T'}$ indicates the transpose matrix, $\|$ denotes the concatenation operation, and $d = 1,2,\ldots,Y$, where Y is the total number of output values obtained from C_d. Afterward, the output of C_d is fed to the output layer O_k, which is formulated as,

$$O_k(C_d) = \sigma\left(\frac{1}{d}\sum_{d=1}^{Y}\sum_{t \in v} C_d W_t\right) \tag{32}$$

Here, $k = 1,2,\ldots,F$, where F is the total number of O_k. Finally, the O_k delivers the results, such as Attacked (λ) or Non-attacked (ϖ). Also, O_k classifies the types of attacks, such as reconnaissance, backdoor, DoS, and many others.

Pseudo-code for SL-GAT

Input: Constructed graph (ψ_κ)

Output: Attacked (λ) or Non-attacked (ϖ)

Begin
Initialize (ψ_κ)
Set $[it = 1]$
While $(it \le it_{\max})$
For each (ψ_κ)
Perform ς_{λ}
Compute $\aleph^{\omega}(\varsigma_{\lambda}) = \phi(\varsigma_{\lambda}) + (\varsigma_{\lambda} - f(\varsigma_{\lambda})\,\phi(\varsigma_{\lambda})) \bullet \frac{\varsigma_{\lambda} \cdot e^{-\varsigma_{\lambda}} (\sigma(\varsigma_{\lambda}))^2}{1 + \sigma(\varsigma_{\lambda})}$
Update weight W_t
Estimate SL activation function
Find $O_k(C_d) = \sigma\left(\frac{1}{d}\sum_{d=1}^{Y}\sum_{t \in v} C_d W_t\right)$
End For
End While
End

The pseudo-code given above describes the process of the SL-GAT classifier.

4. RESULTS AND DISCUSSION

Here, the performance of the proposed model is analyzed and compared with the existing methods. The metrics were obtained by working on the PYTHON platform.

4.1 Dataset Description

The evaluation of the proposed work is done using the UNSW_NB15 dataset. From the total data, 80% data is used for training and 20% for testing and is given in Table 1.

Table 1: Dataset Details

Data		**UNSW_NB15**
Total		257674
Normal		93000
Attacked		164673
Testing	Normal	74400
	Attacked	131738
Training	Normal	18600
	Attacked	32934

4.2 Performance Analysis

In this section, the performance of the proposed techniques is validated to show the better performance of the proposed techniques.

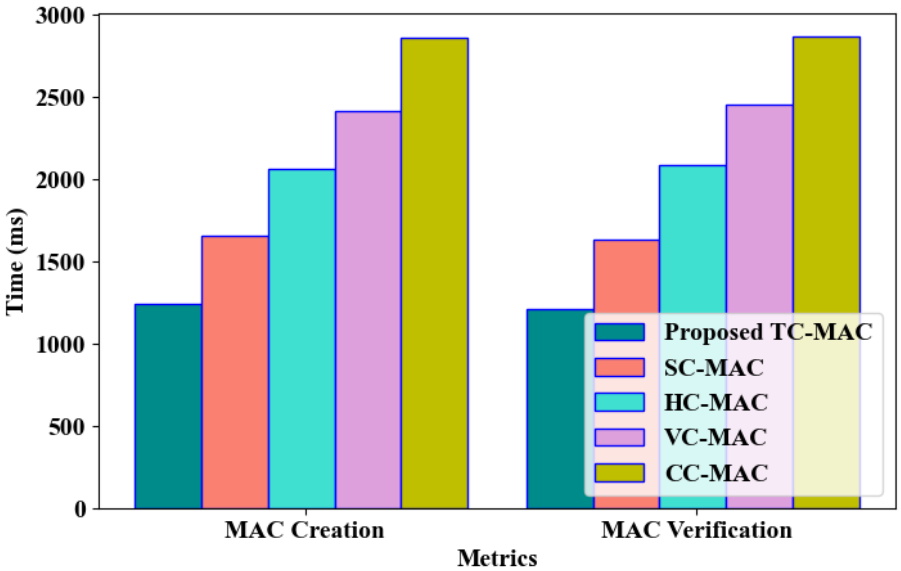

Figure 3: Comparative Analysis for TC-MAC

The proposed TC-MAC algorithm was compared with the existing Stream Cipher-MAC (SC-MAC), Hill Cipher-MAC (HC-MAC), Vignere Cipher-MAC (VC-MAC), and Caesar Cipher-MAC (CC-MAC) as shown in Figure 3. The MA authentication was done based on the sender and receiver host public key. Thus, the MAC was created within 1247ms, and the MA verification was done within 1214ms. But, the existing techniques verified the MA in an average of 2262ms, which is higher than the proposed techniques. Thus, the TC-MAC technique verified the MA more quickly than prevailing models.

Table 2: Comparative Analysis for SL-GAT

Techniques	Precision (%)	Recall (%)	F-Measure (%)
Proposed SL-GAT	98.7456	99.2659	98.4082
GAT	96.1208	97.4587	97.4862
RNN	94.3254	96.2608	95.6324
DBN	91.3265	93.6324	92.1046
DNN	88.7045	91.2347	89.6521

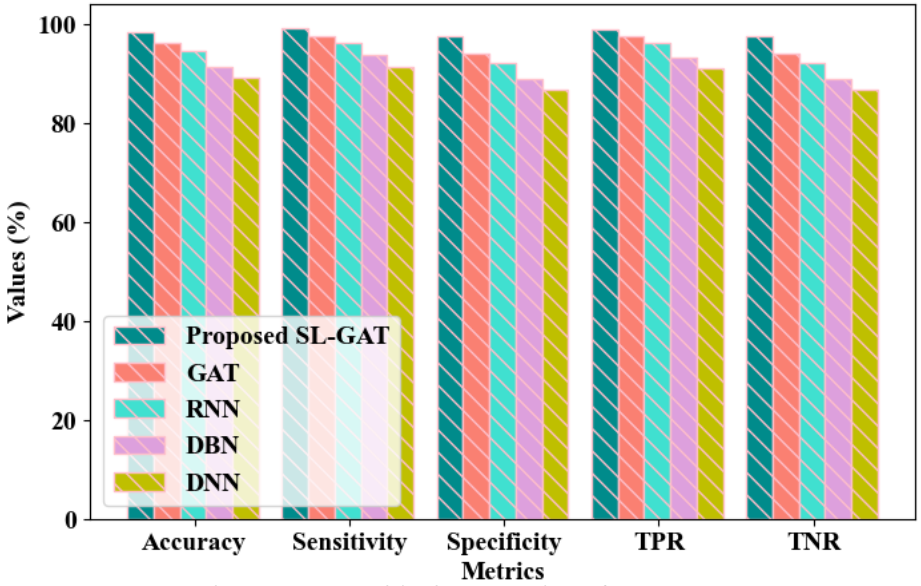

Figure 4: Graphical Comparison for SL-GAT

The metrics achieved by the proposed and existing classifiers are given in Table 2 and Figure 4. The proposed model classified the attack with Precision of 98.7456%, Recall of 99.2659%, F-Measure of 98.4082%, Accuracy of 98.5915%, Sensitivity of 99.2054%, Specificity of 97.5609%, True Positive Rate (TPR) of 99.0737%, and True Negative Rate (TNR) of 97.5609%. The existing GAT, Recurrent Neural Network (RNN), Deep Belief Network (DBN), and Deep Neural Network (DNN) obtained lower metrics values than the proposed work. The proposed model used the SmishLogish activation function, which more precisely classified the attacks than the existing techniques.

Table 3: Comparative Analysis for PP-FQCC

Methods	Encryption Time (ms)	Decryption Time (ms)
Proposed PP-FQCC	968	987
ECC	1357	1411
RSA	1784	1823
ElGamal	2154	2254
AES	2686	2745

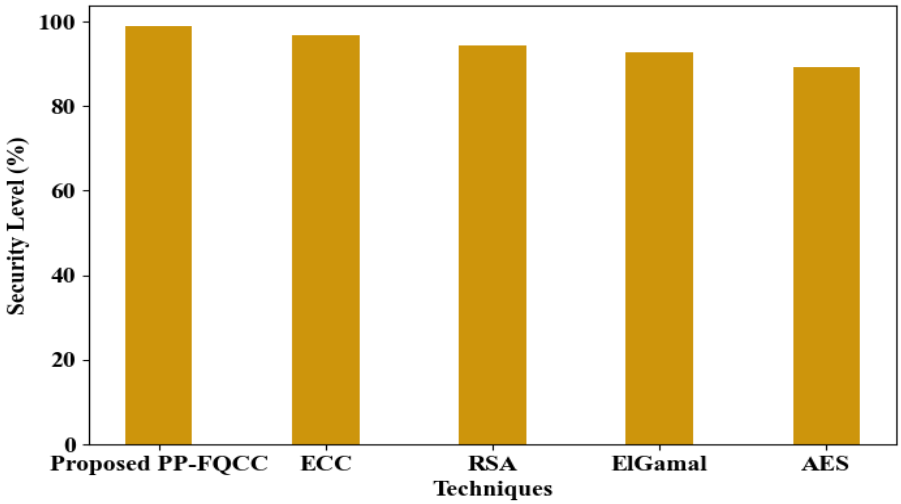

Figure 5: Comparison of Security Level

The PP-FQCC method was compared with existing models regarding Encryption time (ET), Decryption Time (DT), and Security Level (SL) as given in Table 3 and Figure 5. The proposed model secured the data with ET of 968ms, DT of 987ms, and SL of 98.87%. The existing ECC, Rivest–Shamir–Adleman (RSA), ElGamal, and Advanced Encryption Standard (AES) obtained SL of 96.83%, 94.43%, 92.74%, and 89.22% and higher ET and DT than the proposed technique. The usage of the FourQ curve in ECC made the proposed model secure the data superiorly.

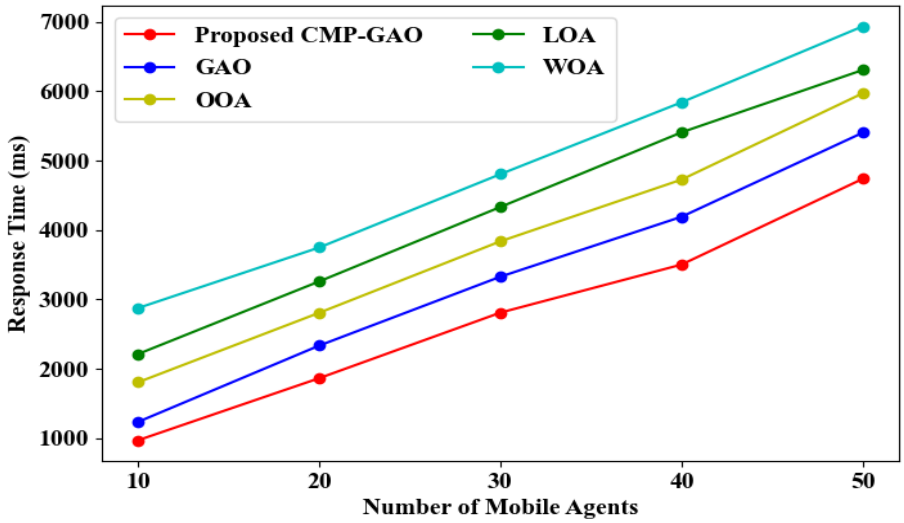

Figure 6: Performance Analysis for CMP-GAO

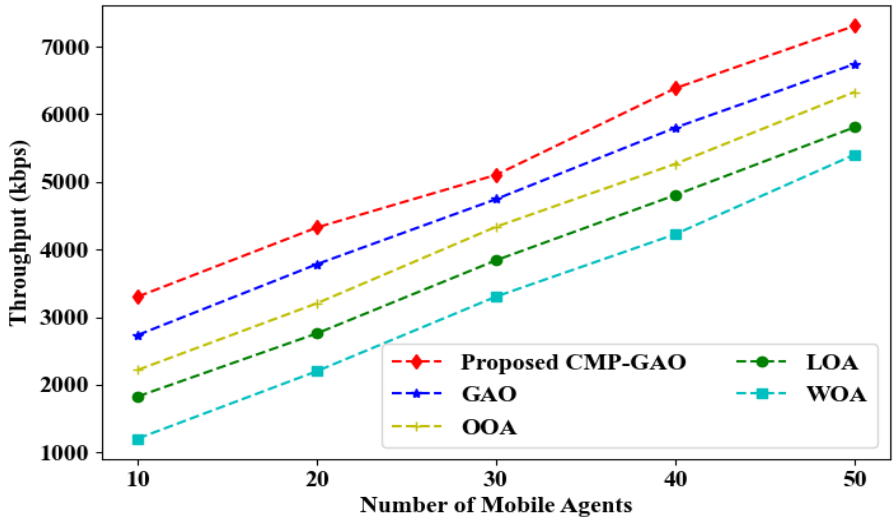

Figure 7: Comparison regarding Throughput

The localization of MA was done using the CMP-GAO technique. Here, the hunting of the GA was done regarding the CMP distribution function. Therefore, the MA was localized in a Response Time of 4738ms and Throughput of 7305 kbps. As depicted in Figures 6 and 7, the existing optimizers GAO, Osprey Optimization Algorithm (OOA), Lyrebird Optimization Algorithm (LOA), and Whale Optimization Algorithm (WOA) obtained higher Response Time and lower Throughput than the proposed technique. Thus, the CMP-GAO performed better than the existing optimizers.

Table 4: Comparison of Related Works

Study	Method	Dataset	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
Proposed Work	SL-GAT	UNSW_NB15	98.5915	98.7456	99.2659	98.4082
(Ribeiro et al., 2020)	OneR	Real Time Data	97.27	97.79	98.60	98.19
(Yadav et al., 2022)	XGBoost	UNSW_NB15	98	97.9	97.5	94.10
(Ajao & Apeh, 2023)	GARL	UNSW_NB15	98.1	97	-	98
(Schmitt, 2023)	GBM	NSL_KDD	97.78	97.59	94.44	94.37
(Singh et al., 2022)	EHIDF	UNSW_NB15	90.25	94.36	97.69	94.35

The comparison of the proposed work with the existing models is given in Table 4. The SL-GAT technique classified the attack with a Precision of 98.7456%. But, the existing Extreme Gradient Boosting (XGBoost) and Edge-based Hybrid Intrusion Detection Framework (EHIDF) used limited features for classification, which obtained an F-Measure of 94.10% and 94.35%. The One Rule (OneR), Genetic Algorithm-Based Reinforcement Learning (GARL), and Gradient Boosting Machine (GBM) could not detect the attack in the network. Thus, these works obtained lower Precision and Recall values than the proposed classifier. Hence, SL-GAT achieved better performance than prevailing classifiers.

5. CONCLUSION

This paper proposed an effective framework for intrusion detection in networks based on Mobile Agents. The authenticated MA was localized using the CMP-GAO technique with a Response Time of 4738ms. Next, the MA was authorized using TC-MAC, which verified the MA in 1214ms. The data of MA was secured with 98.87% SL using the PP-FQCC method. Finally, the intrusion in secured data was detected using SL-GAT with an Accuracy of 98.5915%. Therefore, it can be concluded that the proposed model effectively secured the network based on Mobile Agents.

Future Scope

Even though the attack on the data transferred through MA was detected as an intrusion, the attack on MA was not considered. Hence, in the future, MA attack prevention will be considered for effective data transfer.

REFERENCES

Dataset link: https://www.kaggle.com/datasets/mrwellsdavid/unsw-nb15

1. Ajao, L. A., & Apeh, S. T. (2023). Secure edge computing vulnerabilities in smart cities sustainability using petri net and genetic algorithm-based reinforcement learning. Intelligent Systems with Applications, 18, 1–21. https://doi.org/10.1016/j.iswa.2023.200216
2. Cao, X., Fu, Y., & Chen, B. (2020). Packet-based intrusion detection using Bayesian topic models in mobile edge computing. Security and Communication Networks, 2020, 1–12. https://doi.org/10.1155/2020/8860418
3. Chen, Z., Lv, N., Liu, P., Fang, Y., Chen, K., & Pan, W. (2020). Intrusion detection for wireless edge

networks based on federated learning. IEEE Access, 8, 217463–217472. https://doi.org/10.1109/ACCESS.2020.3041793
4. Garg, S., Kaur, K., Kaddoum, G., Garigipati, P., & Aujla, G. S. (2021). Security in IoT-driven mobile edge computing: New paradigms, challenges, and opportunities. IEEE Network, 35(5), 298–305. https://doi.org/10.1109/MNET.211.2000526
5. Gong, Y., Liu, Y., & Yin, C. (2021). A novel two-phase cycle algorithm for effective cyber intrusion detection in edge computing. Eurasip Journal on Wireless Communications and Networking, 2021(1), 1–22. https://doi.org/10.1186/s13638-021-02016-z
6. Hou, Y., Garg, S., Hui, L., Jayakody, D. N. K., Jin, R., & Hossain, M. S. (2020). A data security enhanced access control mechanism in mobile edge computing. IEEE Access, 8, 136119–136130. https://doi.org/10.1109/ACCESS.2020.3011477
7. Krishnan, R. S., Julie, E. G., Robinson, Y. H., Kumar, R., Son, L. H., Tuan, T. A., & Long, H. V. (2020). Modified zone based intrusion detection system for security enhancement in mobile ad hoc networks. Wireless Networks, 26(2), 1275–1289. https://doi.org/10.1007/s11276-019-02151-y
8. Lai, S., Zhao, R., Tang, S., Xia, J., Zhou, F., & Fan, L. (2021). Intelligent secure mobile edge computing for beyond 5G wireless networks. Physical Communication, 45, 1–8. https://doi.org/10.1016/j.phycom.2021.101283
9. Liu, X., Zhang, W., Zhou, X., & Zhou, Q. (2021). MECGuard: GRU enhanced attack detection in mobile edge computing environment. Computer Communications, 172, 1–9. https://doi.org/10.1016/j.comcom.2021.02.022
10. Nie, L., Wu, Y., Wang, X., Guo, L., Wang, G., Gao, X., & Li, S. (2022). Intrusion detection for secure social internet of things based on collaborative edge computing: A generative adversarial network-based approach. IEEE Transactions on Computational Social Systems, 9(1), 134–145. https://doi.org/10.1109/TCSS.2021.3063538
11. Parsamehr, R., Mantas, G., Rodriguez, J., & Martinez-Ortega, J. F. (2020). IDLP: An efficient intrusion detection and location-aware prevention mechanism for network coding-enabled mobile small cells. IEEE Access, 8, 43863–43875. https://doi.org/10.1109/ACCESS.2020.2977428
12. Ramaiah, M., Chandrasekaran, V., Ravi, V., & Kumar, N. (2021). An intrusion detection system using optimized deep neural network architecture. Transactions on Emerging Telecommunications Technologies, 32(4), 1–17. https://doi.org/10.1002/ett.4221
13. Ren, K., Zeng, Y., Zhong, Y., Sheng, B., & Zhang, Y. (2023). MAFSIDS: A reinforcement learning-based intrusion detection model for multi-agent feature selection networks. Journal of Big Data, 10(1), 1–30. https://doi.org/10.1186/s40537-023-00814-4
14. Ribeiro, J., Saghezchi, F. B., Mantas, G., Rodriguez, J., Shepherd, S. J., & Abd-Alhameed, R. A. (2020). An Autonomous host-based intrusion detection system for android mobile devices. Mobile Networks and Applications, 25(1), 164–172. https://doi.org/10.1007/s11036-019-01220-y
15. Schmitt, M. (2023). Securing the digital world: Protecting smart infrastructures and digital industries with artificial intelligence (AI)-enabled malware and intrusion detection. Journal of Industrial Information Integration, 36, 1–12. https://doi.org/10.1016/j.jii.2023.100520
16. Sindjoung, M. L. F., Velempini, M., & Djamegni, C. T. (2023). A data security and privacy scheme for user quality of experience in a Mobile Edge Computing-based network. Array, 19, 1–11. https://doi.org/10.1016/j.array.2023.100304
17. Singh, A., Chatterjee, K., & Satapathy, S. C. (2022). An edge based hybrid intrusion detection framework for mobile edge computing. Complex and Intelligent Systems, 8(5), 3719–3746. https://doi.org/10.1007/s40747-021-00498-4
18. Sun, J. (2022). Certificateless batch authentication scheme and intrusion detection model based on the mobile edge computing technology NDN-IoT environment. Journal of Function Spaces, 2022, 1–10. https://doi.org/10.1155/2022/5926792
19. Wang, W., Huang, H., Li, Q., He, F., & Sha, C. (2020). Generalized intrusion detection mechanism for empowered intruders in wireless sensor networks. IEEE Access, 8, 25170–25183. https://doi.org/10.1109/ACCESS.2020.2970973
20. Yadav, N., Pande, S., Khamparia, A., & Gupta, D. (2022). Intrusion detection system on IoT with 5G network using deep learning. Wireless Communications and Mobile Computing, 2022, 1–13. https://doi.org/10.1155/2022/9304689