International Journal of Management, IT & Engineering
Vol. 14 Issue 04, April 2024
ISSN: 2249-0558 Impact Factor: 7.119
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

ZERO TRUST-BASED SECURE CREDENTIAL PHISHING DETECTION FRAMEWORK USING Ellsigm-GRU

Rajender Reddy Pell Reddy

Abstract

The practice of stealing one's personal information via a known person or entity in email and any other communication channels is named Credential Phishing (CP). Therefore, the timely detection of CP is very crucial. The CP was effectively detected by numerous prevailing works; however, the false alarm rates were higher. Hence, this paper proposes a zero trust-based secure CP detection model in the cloud. Primarily, by creating a Condition Access Policy (CAP), the Zero-Trust Access (ZTA) is processed. The user can log in to the platform if ZTA is successfully completed; or else, the Multi-Factor Authentication (MFA) gets initiated by producing Quick Response (QR) codes. After that, the Elliott logsigm-Gated Recurrent Unit (Ellsigm-GRU) is employed for CP detection. Lastly, if the site is non-phishing, then the access is further proceeded with Jacobian Koblitz Curve Cryptography (JKCC). The analysis outcomes exhibited the developed model's superiority by attaining an accuracy of 98.87% in CP detection.

Keywords:

Phishing Email Dataset (PED);
Phishing Uniform Resource Locator Dataset (PURLD);
Fishers Interpolated Yates-HAVAL (FIY-HAVAL);
Email & URL phishing;
Phishing Uniform Resource Locator Dataset (PURLD).

Author correspondence:

Rajender Reddy Pell Reddy,
Cyber Security Expert,
Virginia, USA
Email: rpellreddy@gmail.com

1. INTRODUCTION

Currently, the Internet plays an important role in all aspects of people's lives (Tang & Mahmoud, 2021). Also, cybercrimes and several security issues have become more intense with the development of Internet facilities (Salloum et al., 2022). Since CP steals one's sensitive information like user name, ID, passwords, bank account details, and credit card details, it is one of the most important cybercrimes (Faris & Yazid, 2021). The CP is done by emails, instant messages, and phone calls from a known entity or person via any communication channel (Karim et al., 2023) (Salloum et al., 2021). Email phishing is the most common CP; the phishing action easily traps the victims via an authorized email ID (Singh et al., 2020) (Korkmaz et al., 2020). Significant financial and reputational damage to an organization or a person is caused by the CP (Thakur et al., 2023). Theft, credit card fraud, ransom ware attacks, and data breaches are caused by the CP (Kumar & Subba, 2021). Therefore, the detection of these phishing actions is very significant.

For phishing detection, numerous research works have been developed. Machine Learning (ML) methods, Deep Learning (DL) techniques, natural language processing, and spam filtering are some of the research works. However, as the features from the images and URLs present in the email are not investigated, the prevailing models have a high false alarm rate (Aljofey et al., 2020). Thus, in this work, a zero trust-based secure CP detection in the cloud using Ellsigm-GRU is proposed.

1.1. Problem statement

The existing works' drawbacks are:

- Prevailing works didn't concentrate on decentralized user management in the ZTA process.
- In the prevailing models, images and URLs in emails were ignored for CP detection.

- The prevailing systems didn't focus on data security for solving the CP.
- In the prevailing works, only a limited number of features were employed.

The major contributions are:

- ♦ In the proposed work, a decentralized Inter Planetary File System (IPFS) is utilized.
- ♦ The images and URLs in the emails are separated and utilized for CP detection.
- ♦ An effective JKCC-centered data security is employed here.
- ♦ From the data in the Phishing Email Dataset (PED) and Phishing Uniform Resource Locator Dataset(PURLD), more features are extracted

The framework's structure is: the related works are analyzed in Section 2, Section 3 presents the proposed methodology, the results are discussed in Section 4, and the paper is concluded in Section 5.

2. LITERATURE SURVEY

(Butt et al., 2023) introduced a cloud-based phishing attack detection framework using ML and DL algorithms. To detect the cloud-based phishing attacks, classifiers, namely Support Vector Machine (SVM), Naïve Bayes (NB), and Long-Short Term Memory (LSTM) were used in this work. Yet, the URLs were ignored in the detection process, which might augment the false alarm rate.
(Ahammad et al., 2022) used ML methods for phishing URL identification. In this, to detect malicious URLs, ML approaches like Random Forests (RF), Decision Trees (DT), Light Gradient Boosting Machine, and Logistic Regression (LR) were used. Nevertheless, this research had inadequate resources and knowledge for ensuring security.
(Mostafa et al., 2023) presented an innovative multi-factor multi-layer authentication framework for cloud user authentication. For concealing the login information in the cloud, this work used the Advanced Encryption Standard (AES) algorithm. Moreover, the AES encryption was harder for implementing, thus hindering the performance.
(Butnaru et al., 2021) showed the lightweight URL-based phishing detection. Initially, the features were extracted. Then, to block the phishing attacks, the supervised ML methods, namely RF and SVM were used. Furthermore, the over fitting issues were increased by the biggest parameter, which might result in false identification.
(Ozcan et al., 2021) described the detection of phishing URLs by utilizing DL models. In this, to detect phishing URLs, hybrid DL models, namely LSTM and Deep Neural Network (DNN) algorithms were utilized. Hence, superior results were achieved by the presented model. Moreover, the performance of the model could be still affected by a few noisy instances present in the work.

3. PROPOSED METHODOLOGY

Based on Ellsigm-GRU and ZTA, the CP detection framework is proposed in this section. In Figure 1, the proposed framework's architecture is signified.

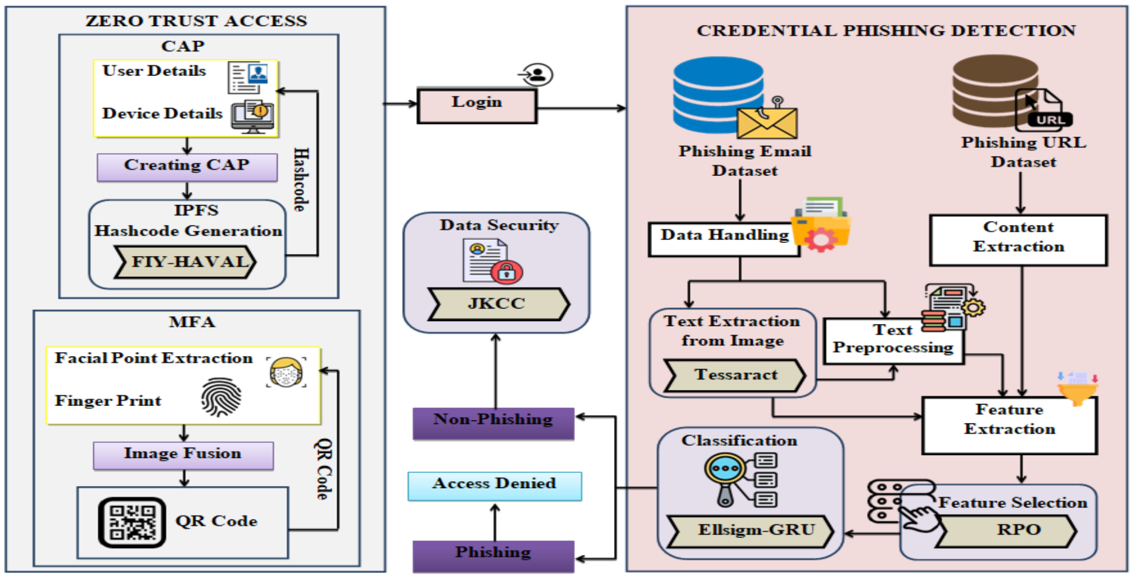


Figure 1: Structure of the proposed framework

3.1. Zero-trust access

The ZTA is first processed by getting the user and device details when a user registers in a platform for data sharing. After that, CAP is created centered on these details. The ZTA is a centralized security framework, where the whole system fails when one network point fails. Thus, a decentralized IPFS is used. Also, it generates a hash code for user verification. In this, the Fishers Interpolated Yates-HAVAL (FIY-HAVAL) approach is used due to its ability to generate complex hash codes; however, it has a problem with the fixed length of the consonant string. Therefore, the Fishers Interpolated Yates (FIY) method is utilized, which aids in creating complex hash codes in numerous scenarios with less time. Let, (U) be the user data. Also, the user data is padded so that the length of U becomes a multiple of 1024. The last block of the padded U gives the number of bits in unpadded U, the required length of the hash code for U, the number of passes $(\delta_1,\delta_2,\delta_3,\delta_4,\beta_5)$ each block in U has to process, and the version of HAVAL. The input (ϖ) rendered to the hashing function (δ) is explained by,

$$\varpi = \{\Omega_{in}, U\} \tag{1}$$

Where, the initial block is signified as Ω_{in}. After that, to compute the output (Ω_{out}), δ is preprocessed, and it is compressed in the 3, 4, or 5 passes, which is explained as,

$$\sum_{w=0}^{5}\varpi_w = \begin{cases} \varpi_0 = \Omega_{in} \\ \varpi_1 = \delta_1(\varpi_0, U) \\ \varpi_2 = \delta_2(\varpi_1, U) \\ \varpi_3 = \delta_3(\varpi_2, U) \\ \varpi_4 = \delta_4(\varpi_3, U), \quad (if \quad \Psi = 4,5) \\ \varpi_5 = \delta_5(\varpi_4, U), \quad (if \quad \Psi = 5) \end{cases} \tag{2}$$

$$\Omega_{out} = \begin{cases} \varpi_3 \oplus \varpi_0, & if \quad \Psi = 3 \\ \varpi_4 \oplus \varpi_0, & if \quad \Psi = 4 \\ \varpi_5 \oplus \varpi_0, & if \quad \Psi = 5 \end{cases} \tag{3}$$

Here, the total number of rounds is specified as w. Thereafter, the Boolean function $(\Delta_1,\Delta_2,\Delta_3,\Delta_4,\Delta_5)$ employed by Fishers Interpolated Yates technique for $\delta_1,\delta_2,\delta_3,\delta_4,\beta_5$ is described as, $\Delta_1\delta_1(m_6,m_5,m_4,m_3,m_2,m_1,m_0) = m_0 \oplus m_1m_4 \oplus \frac{m_2m_5}{m_3m_6} \oplus m_0m_1$ (4)

$$\Delta_2\delta_2(m_6,m_5,m_4,m_3,m_2,m_1,m_0) = m_0 \oplus m_1m_2m_3 \oplus \frac{m_1m_2 \oplus m_1m_4 \oplus m_2m_6}{m_3m_5 \oplus m_4m_5 \oplus m_0m_2} \oplus m_2m_4m_5 \tag{5}$$

$$\Delta_3\delta_3(m_6,m_5,m_4,m_3,m_2,m_1,m_0) = m_0 \oplus \frac{m_1m_4 \oplus m_2m_5}{m_3m_6 \oplus m_0m_3} \oplus m_1m_2m_3 \tag{6}$$

$$\Delta_4\delta_4(m_6,m_5,m_4,m_3,m_2,m_1,m_0) = m_0 \oplus m_1m_2m_3 \oplus m_2m_4m_5 \oplus \frac{m_1m_4 \oplus m_2m_6 \oplus m_3m_4 \oplus m_3m_5}{m_3m_6 \oplus m_4m_5 \oplus m_4m_6 \oplus m_0m_4} \oplus m_3m_4m_6 \tag{7}$$

$\Delta_5\delta_5(m_6,m_5,m_4,m_3,m_2,m_1,m_0) = m_0 \oplus m_0m_1m_2m_3 \oplus \frac{m_1m_4 \oplus m_2m_5}{m_3m_6 \oplus m_0m_5}$ (8)Here, the strings in U are represented as $(m_0,m_1,m_2,m_3,m_4,m_5)$. Lastly, the FIY-HAVAL function adjusts the last 256-bit string in U to the length specified in the last block of the padded U. This approach outputs the adjusted value as the hash code (U_{hash}) of U. Next, for verification, U_{hash} is given to the user. The user can log in to the platform if it is verified; or else, the MFA is initiated by generating a QR code for verification. To generate QR codes, the MFA utilizes the facial points and fingerprints of the users and fuses the images. The user can log in to the platform after the user access is verified successfully.

Pseudo code for FIY-HAVAL

```
Input: User data $(U)$
Output: Hash code $(U_{hash})$
Begin
    Initialize $U$, number of passes $(\delta_1, \delta_2, \delta_3, \delta_4, \beta_5)$
    For $\varpi$ do
        Evaluate $\sum_{w=0}^{5} \varpi_w$
        Estimate compressing function
        Obtained $\Omega_{out} = \begin{cases} \varpi_3 \oplus \varpi_0, & if \;\; \Psi = 3 \\ \varpi_4 \oplus \varpi_0, & if \;\; \Psi = 4 \\ \varpi_5 \oplus \varpi_0, & if \;\; \Psi = 5 \end{cases}$
        Formulate Boolean function $(\Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_5)$
        Adjust last string
    End For
    Return $U_{hash}$
End
```

3.2. Credential phishing detection

After that, by utilizing data from two datasets, such as PED and PURLD, the CP detection model is trained.

3.2.1. Data handling

The detection efficiency is improved by the data handling (Φ) process under three processes, such as subject, body, and text separation (Φ_1), URL identification (Φ_2), and image identification (Φ_3), and it is explained as,

$$\Phi = \{\Phi_1, \Phi_2, \Phi_3\} \tag{9}$$

3.2.2. Text extraction

Thereafter, by using Tesseract, the texts (ω_{tei}) in the images (Φ_3) are further extracted, and it is provided as,

$$\Phi_3 \xrightarrow{Tessaract} \omega_{tei} \tag{10}$$

3.2.3. Text preprocessing

Likewise, the texts in Φ and ω_{tei} are preprocessed by replacing the sensitive data with a surrogate value as a token in the tokenization process. Afterward, the stop words in the text are removed under the stop words removal process. At last, the words in the text are reduced to their keywords in the stemming process. ω_{pp} denotes the preprocessed text.

3.2.4. Content Extraction

Next, the contents $\left(co_{url}\right)$ in the PURLD are extracted. The URL has the contents, namely the protocol name to access the resource and the resource name.

3.3. Feature extraction

After that, the features, such as gray-level co-occurrence matrix, histogram of oriented gradients, texture from Φ_3, part of speech, positive count, negative count from ω_{pp}, lexical, host, and content from co_{url} are further extracted, and it is explained by,

$$\ell \to \{\omega_{tei}, \omega_{pp}, co_{url}\} \tag{11}$$

Where, the extracted features are signified as ℓ.

3.4. Feature selection

Subsequently, the optimal features are selected from ℓ utilizing Red Panda Optimization(RPO), which is used to effectively balance the exploration and exploitation process. Initially, the population (P) of the red pandas (ℓ) is initialized in the search dimension (n) as,

$$P = \begin{bmatrix} P_1 \\ \vdots \\ P_a \\ \vdots \\ P_m \end{bmatrix}, \quad a = \{1,2,\ldots\ldots,m\} \tag{12}$$

$$P = \begin{bmatrix} P_{1,1} & \cdots & P_{1,b} & \cdots & P_{1,n} \\ \vdots & \ddots & \vdots & \unicode{x22F0} & \vdots \\ P_{a,1} & \cdots & P_{a,b} & \cdots & P_{a,n} \\ \vdots & \unicode{x22F0} & \vdots & \ddots & \vdots \\ P_{m,1} & \cdots & P_{m,b} & \cdots & P_{m,n} \end{bmatrix}_{m\times n}, \quad b = \{1,2,\ldots\ldots,n\} \tag{13}$$

Here, $P_{a,b}$ and m denote the a^{th} red panda in the b^{th} dimension and the number of red pandas in P, respectively. Thereafter, the position of (ℓ) is randomly initialized as,

$$P_{a,b} = L_b + \iota_{a,b}^1 \cdot (U_b - L_b) \tag{14}$$

Here, the upper and lower bounds of n are represented as U_b and L_b, respectively, and the random number is signified as $\iota_{a,b}^1$. The fitness function (γ) of the RPO is the maximum $(\max)$ accuracy (Φ), and it is rendered as,

$$\gamma = \max\{\Phi\} \tag{15}$$

Next, the location of the food source (F_a) contrasted with γ is provided as,

$$F_a = \{P_c \mid \gamma_c < \gamma\} \cup \{P_{best}\}, \quad c \in a = \{1,2,\ldots\ldots,m\} \tag{16}$$

Where, the c^{th} red panda is denoted as P_c, the best solution is signified as P_{best}, and the fitness of the c^{th} red panda is specified as γ_c. After that, the new position $\left(P_{a,b}^1\right)$ of (ℓ) is computed as,

$$P_{a,b}^1 = P_{a,b} + \iota_{a,b}^2 \cdot \left(F_{a,b}^s - \iota \cdot P_{a,b}\right) \tag{17}$$

Here, a random number is specified as $\iota^2_{a,b}$, the selected food source is indicated as $F^s_{a,b}$, and a random number is represented as ι. The position updation strategy is displayed as,

$$P_a = \begin{cases} P^1_{a,b}, & if \;\; \gamma^1_{a,b} < \gamma \\ P_a, & else \end{cases} \tag{18}$$

Where, the fitness value for $\left(P^1_{a,b}\right)$ is signified as $\gamma^1_{a,b}$. The position changes in the search dimension $\left(P^2_{a,b}\right)$ after the foraging period, which is explained as,

$$P^2_{a,b} = P_{a,b} + \frac{L_b + \iota^3_{a,b} \cdot (U_b - L_b)}{t} \tag{19}$$

Here, $\iota^3_{a,b}$ and t denote a random number and the iterations number, respectively. After that, the position updation of the a^{th} red panda towards $\left(P^2_{a,b}\right)$ becomes,

$$P_a = \begin{cases} P^2_{a,b}, & if \;\; \gamma^2_{a,b} < \gamma \\ P_a, & else \end{cases} \tag{20}$$

Where, the fitness value for the new position of the a^{th} red panda in b^{th} dimension is signified as $\gamma^1_{a,b}$. The RPO process continues till all iterations are completed. Lastly, the optimal feature $\left(\ell_{opt}\right)$ is the best position of the red pandas that reaches a high fitness value from γ.

3.5. Classification

Then, ℓ_{opt} is utilized for training the Ellsigm-GRU classifier, which is utilized to categorize the phishing and non-phishing sites. The GRU has over fitting problems and low learning efficiency. Thus, Ellsigm activation is utilized due to its faster training process and greater approximation capability. The Ellsigm-GRU classifier has two input features, namely the input vector $\left(\ell_{opt}\right)$ and the previous output vector from the hidden state $\left(h_{q-1}\right)$. The Ellsigm-GRU classifier has two gates named the reset gate $\left(r_q\right)$ and the update gate $\left(u_q\right)$. In Figure 2, the Ellsigm-GRU structure is presented.

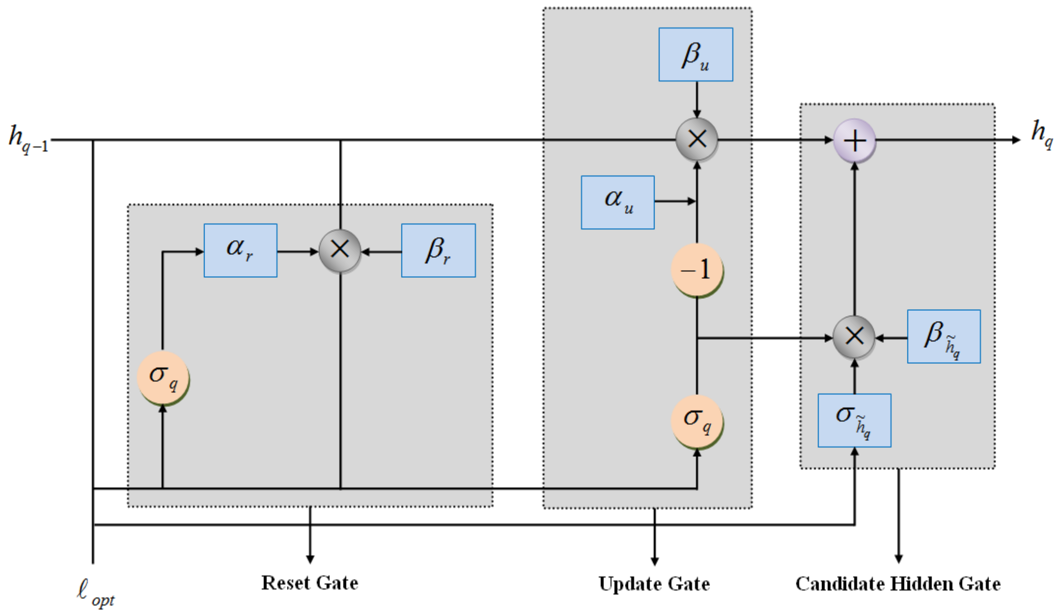


Figure 2: Ellsigm-GRU structure

The $\left(r_q\right)$ in Ellsigm-GRU is utilized for determining how much previous $\left(h_{q-1}\right)$ should be forgotten, which is calculated as,

$$r_q = \sigma_q \cdot \left(\alpha_r \left[\ell_{opt} + h_{q-1}\right] + \beta_r\right) \tag{21}$$

Where, the weight and bias terms of (r_q) are specified as α_r *and* β_r, and the sigmoid function is signified as σ_q.

$$\sigma_q = \left(\frac{0.5q}{1 + |q \exp^{-q}|} \right)^2 + 0.5 \tag{22}$$

Here, the standard exponential function is signified as $\exp$. Subsequently, the (u_q) is utilized for determining how much ℓ_{opt} is utilized to update the (h_{q-1}),

$$u_q = \sigma_q \cdot \left(\alpha_u \lfloor \ell_{opt} + h_{q-1} \rfloor + \beta_u \right) \tag{23}$$

Where, the weight and bias terms of (u_q) are denoted as α_u *and* β_u. After that, the relationship between the input and the output is articulated as,

$$h_q = [1 - u_q] h_{q-1} + u_q \tilde{h}_q \tag{24}$$

Here, $\tilde{h}_q$ and h_q denote the candidate's (h_{q-1}) and the new (h_{q-1}), respectively. The $(\tilde{h}_q)$ is shown as,

$$\tilde{h}_q = \alpha_{\tilde{h}_q} (\ell_{opt}) + \alpha_{\tilde{h}_q} (r_q * h_{q-1}) + \beta_{\tilde{h}_q} \tag{25}$$

Where, the weight and bias terms of $\tilde{h}_q$ are denoted as $\alpha_{\tilde{v}_q}$ *and* $\beta_{\tilde{v}_q}$. Therefore, the output is obtained as phishing (μ_1) or non-phishing (μ_2) in the new hidden state (h_q). The access is denied if phishing is detected; or else, the access is further proceeded with secured data transmission.

Pseudocode for Ellsigm-GRU

```
Input: Optimal features $(\ell_{opt})$
Output: Phishing $(\mu_1)$ or non-phishing $(\mu_2)$
Begin
    Initialize $r_q$, $u_q$, weights $(\alpha_r, \alpha_u, \alpha_{\tilde{v}_q})$, and bias $(\beta_r, \beta_u, \beta_{\tilde{v}_q})$
    For each $\ell_{opt}$ do
        Compute, $r_q = \sigma_q \cdot (\alpha_r \lfloor \ell_{opt} + h_{q-1} \rfloor + \beta_r)$
        Formulate sigmoid function
        Evaluate, $u_q = \sigma_q \cdot (\alpha_u \lfloor \ell_{opt} + h_{q-1} \rfloor + \beta_u)$
        Estimate the candidate hidden state,
    End For
End
```

3.6. Data security

Next, secured data transmission is processed with JKCC. Nevertheless, owing to the negative points obtained with the variables, the complexity is increased. Thus, the Jacobian Koblitz Curve (JKC) is utilized. Also, JKC could defend against simple and differential power analysis attacks. The public (K_1) and private keys (K_2) are generated by the ECC via the property of the JKC equation. The data on the curve is within the (X, Y) pair centered on the constants $(a' \text{ and } b')$, and the pair is explained as,

$$Y^2 = -\pi^{J_K(a')} X^4 + \Gamma_K \left[\frac{a'_l}{L-1} \right] 2b' X^2 + 1 \tag{26}$$

Here, the sum of digits is denoted as $J_K(a')$, the p-adic gamma function is represented as Γ_K, the integer is signified as a'_l, and the power of prime K is specified as L. The encryption is processed centered on the generated keys to convert non-phishing data to cipher text (ξ) utilizing K_1 and a master key (S_k), and it is rendered as,

$$\xi = S_k \cdot K_1(I) \tag{27}$$

After that, at the receiver end, the cipher text (ξ) is decrypted with the help of K_2, and it is articulated as,

$$I = K_2 \cdot \xi \tag{28}$$

Hence, the data is securely transmitted. Also, this framework's results and discussion are discussed further.

4. RESULT AND DISCUSSION

In this section, the proposed system's performance is assessed by analogizing it with prevailing approaches. Furthermore, the proposed system is implemented on the working platform of PYTHON.

4.1 Dataset description

By utilizing two datasets, such as the PED and PURLD, the proposed model is evaluated. The data like URLs and types are comprised in the PURLD. Moreover, PURLD has a total of 6,51,191 number of data. The data like sender, receiver, subject, body, and URLs are encompassed in the PED. Also, Ped has a total of 71,487 number of data. From the whole data, 80% and 20% of data are utilized for training and testing, respectively.

4.2 Performance analysis of proposed FYI-HAVAL

In this section, the proposed FYI-HAVAL's performance is investigated by analogizing it with the prevailing methods, namely HAVAL, Tiger, Secure Hash Algorithm-512 (SHA 512), and Message-Digest Algorithm 5 (MD5).

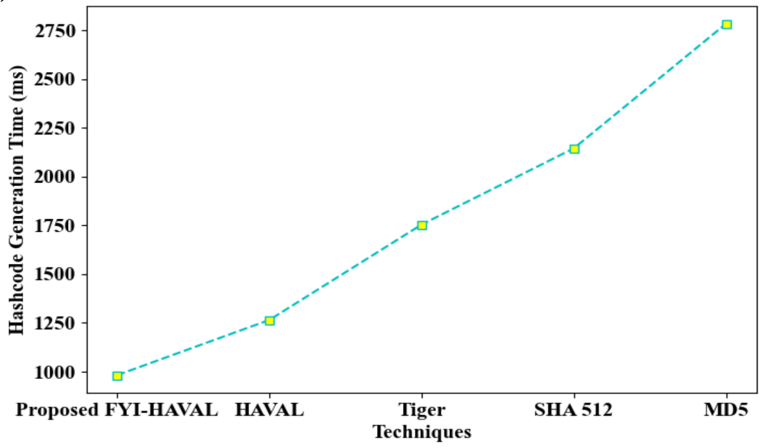


Figure 3: Graphical analysis of hash code generation time

The graphical analysis of the proposed FYI-HAVAL with traditional algorithms regarding Hash code Generation Time (HGT) is shown in Figure 3. In this, a low HGT (982ms) is achieved by the proposed FYI-HAVAL. Nevertheless, a high HGT of 1265ms is reached by the prevailing HAVAL. Here, to solve the issues with the fixed length of the consonant string, the FIY technique is added with HAVAL. Hence, the hash codes are efficiently generated by the proposed FYI-HAVAL with less time.

4.3 Performance estimation of the proposed JKCC

In this section, the proposed framework's performance estimation is validated to prove the model's security.

Table 1: Performance assessment

Technique	Encryption time (ms)	Decryption time (ms)
Proposed JKCC	986	996
ECC	1284	1254
RSA	1865	1754
ElGamal	2236	2135
AES	2639	2569

In Table 1, the performance assessment of the proposed JKCC compared with prevailing methods, namely Elliptical Curve Cryptography (ECC), Rivest, Shamir, Adleman (RSA), ElGamal, and AES is exhibited. The proposed JKCC effectively reduced the encryption time to 986ms and decryption times to 996ms. Nevertheless, the existing works take an average encryption time of 2006ms and a decryption time of 1928ms. Hence, the JKCC takes less encryption and decryption time, thereby increasing computational efficiency.

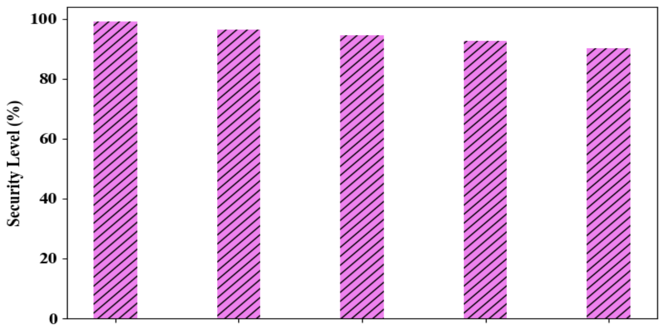
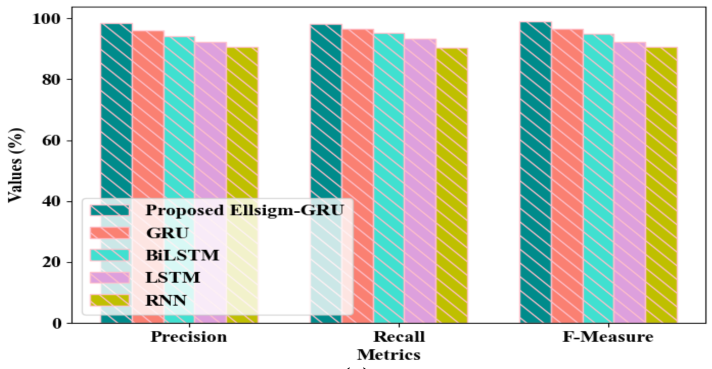


Figure 4: Performance validation of security level

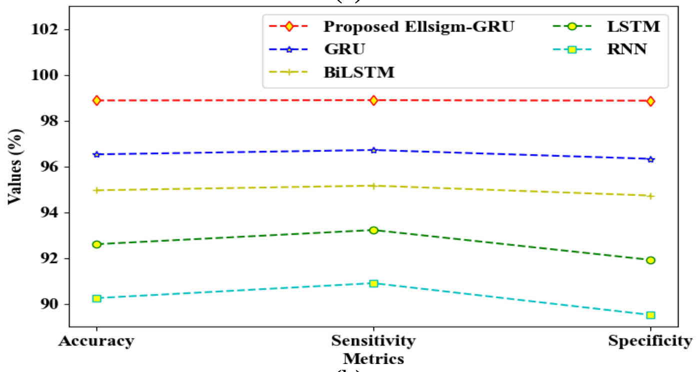
The performance validation of the proposed JKCC and prevailing methods concerning security level is demonstrated in Figure 4. A high-security level (98.97%) is achieved by the proposed JKCC. However, low-security levels of 96.37%, 94.45%, 92.65%, and 90.25% are obtained by the prevailing security methods like ECC, RSA, ElGamal, and AES, respectively. For enhancing the proposed model's security level, the proposed JKCC uses the JKC.

4.4 Performance evaluation of the proposed Ellsigm-GRU

In this section, the proposed Ellsigm-GRU classifier' performance is assessed by analogizing it with traditional methods, such as GRU, Bidirectional LSTM (BiLSTM), LSTM, and Recurrent Neural Network (RNN).



(a)



(b)

Figure 5: Performance analysis of the proposed model

The performance of the proposed Ellsigm-GRU contrasted with traditional classifiers is shown in Figures 5 (a) and (b). To overcome the overfitting and low learning efficiency problems, the proposed model employs an effective Ellsigm activation function. In this, the proposed Ellsigm-GRU achieved a high precision of 98.48%, recall of 98.02%, f-measure of 98.87%, accuracy of 98.87%, sensitivity of 98.88%, and specificity values of 98.86%. Similarly, low average precision, recall, f-measure, accuracy, sensitivity, and specificity values are obtained by the prevailing approaches. Hence, the proposed Ellsigm-GRU performed well in zero trust-centric CP detection.

Table 2: Comparison of TPR, TNR, FPR, and FNR

Technique	TPR (%)	TNR (%)	FPR	FNR
Proposed Ellsigm-GRU	98.81	98.86	1.1346	1.1126
GRU	96.71	96.39	3.6743	3.2863
BiLSTM	95.15	94.72	5.2563	4.8475
LSTM	93.22	91.91	8.0837	6.7034
RNN	90.89	89.51	10.4805	9.1045

In Table 2, the comparison of the proposed Ellsigm-GRU and prevailing works according to True Positive Rate (TPR), True Negative Rate (TNR), False Positive Rate (FPR), and False Negative Rate (FNR) is illustrated. The proposed Ellsigm-GRU attained a TPR of 98.81%, TNR of 98.86%, FPR of 1.1346, and FNR of 1.1126. Similarly, the prevailing BiLSTM obtained a TPR of 95.15%, TNR of 94.72%, FPR of 5.2563, and FNR of 4.8475. Hence, the proposed work has proved as a less error-prone model.

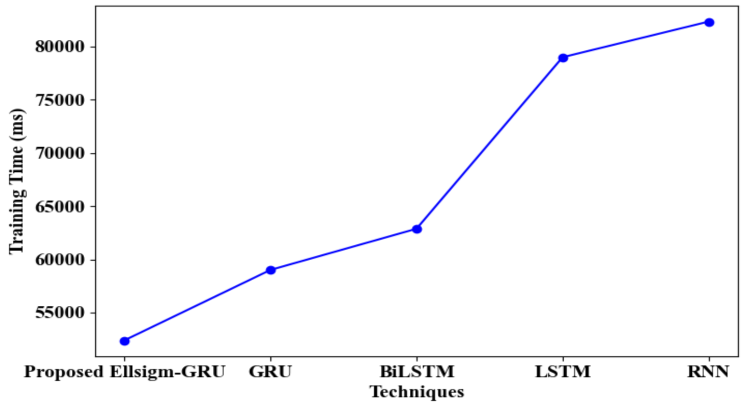


Figure 6: Training time evaluation

The training time validation of the proposed Ellsigm-GRU and traditional approaches is exhibited in Figure 6. A low training time (37845ms) is achieved by the proposed Ellsigm-GRU. Likewise, a mean training time of 50583.25ms is attained by the prevailing models. Therefore, when compared to the prevailing works, the proposed Ellsigm-GRU had higher computational efficiency.

4.5 Comparative analysis of the proposed model

In this section, to prove the framework's reliability, the proposed model's comparative analysis is performed.

Table 3: Comparative analysis of the proposed model

Author name	Algorithms	Dataset used	Accuracy (%)	Precision (%)	F-measure (%)
Proposed model	Ellsigm-GRU& JKCC	PURLD and PED	98.87	98.48	98.87
(Gupta et al., 2021)	Hybrid ML	ISCXURL-2016	97.95	97.55	97.98
(Mohamed et al., 2022)	Hybrid ML	Dataset collected from recent phishing cases	86.50	86.52	84.41
(Sanchez-Paniagua et al., 2022)	Hybrid ML	PILWD	97.95	98.30	98
(Alhogail&Alsabih, 2021)	GCN	Fraud dataset	98.2	98.25	98.5
(El Aassal et al., 2020)	Hybrid ML	PhishTank dataset	96.80	-	93.63

Table 3 provides a comparison of the proposed model with prevailing works. The hybrid ML techniques, such as SVM, Neural Networks (NN), and K-Nearest Neighbor are incorporated by the related works. Also, to achieve secure CP detection, the related works utilize the Graph Convolutional Network (GCN). The proposed Ellsigm-GRU & JKCC achieved a high accuracy of 98.87%, a precision of 98.48%, and an f-measure of 98.87%. Other related works, namely GCN methods obtained a low accuracy of 98.2%, a precision of 98.25%, and an f-measure of 98.5%, thereby causing misclassifications. Likewise, low accuracy, precision, and f-measure are obtained by other related works also. Thus, effective outcomes are achieved by the proposed model.

5. CONCLUSION

In this paper, an effective framework named zero trust-centric secure CP detection in the cloud utilizing Ellsigm-GRU is proposed. In this, for generating the hashcode, the FIY-HAVAL was implemented. Similarly, for data security, the JKCC was used. Also, the phishing and not-phishing attacks are proficiently categorized by the proposed Ellsigm-GRU. Hence, as per the experimental outcomes, the proposed Ellsigm-GRU achieved a recall of 98.02% and an f-measure of 98.87%, thus revealing the system's reliability. Likewise, the security level obtained by the proposed JKCC is 98.97%, which demonstrates a high level of security. Therefore, the phishing attacks in the cloud are effectively detected by the proposed work. Nevertheless, for phishing detection, the proposed work focused only on zero trust-centric authentication.

Future work

To further enhance the model in the future, this work will be improved by considering smart contract-centric user authentication with an intrusion detection system.

REFERENCES

Dataset link: https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset/data
https://zenodo.org/records/8339691

Ahammad, S. H., Kale, S. D., Upadhye, G. D., Pande, S. D., Babu, E. V., Dhumane, A. V., & Bahadur, M. D. K. J. (2022). Phishing URL detection using machine learning methods. *Advances in Engineering Software*, *173*, 1–5. https://doi.org/10.1016/j.advengsoft.2022.103288

Alhogail, A., &Alsabih, A. (2021). Applying machine learning and natural language processing to detect phishing email. *Computers and Security*, *110*, 1–11. https://doi.org/10.1016/j.cose.2021.102414

Aljofey, A., Jiang, Q., Qu, Q., Huang, M., &Niyigena, J. P. (2020). An effective phishing detection model based on character level convolutional neural network from URL. *Electronics*, *9*(9), 1–24. https://doi.org/10.3390/electronics9091514

Butnaru, A., Mylonas, A., &Pitropakis, N. (2021). Towards lightweight url-based phishing detection. *Future Internet*, *13*(6), 1–15. https://doi.org/10.3390/fi13060154

Butt, U. A., Amin, R., Aldabbas, H., Mohan, S., Alouffi, B., & Ahmadian, A. (2023). Cloud-based email phishing attack using machine and deep learning algorithm. *Complex and Intelligent Systems*, *9*(3), 3043–3070. https://doi.org/10.1007/s40747-022-00760-3

El Aassal, A., Baki, S., Das, A., & Verma, R. M. (2020). An In-Depth Benchmarking and Evaluation of Phishing Detection Research for Security Needs. *IEEE Access*, *8*, 22170–22192. https://doi.org/10.1109/ACCESS.2020.2969780

Faris, H., & Yazid, S. (2021). Phishing Web Page Detection Methods: URL and HTML Features Detection. *Proceedings: 2020 IEEE International Conference on Internet of Things and Intelligence Systems*, 167–171. https://doi.org/10.1109/IoTaIS50849.2021.9359694

Gupta, B. B., Yadav, K., Razzak, I., Psannis, K., Castiglione, A., & Chang, X. (2021). A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment. *Computer Communications*, *175*, 47–57. https://doi.org/10.1016/j.comcom.2021.04.023

Karim, A., Shahroz, M., Mustofa, K., Belhaouari, S. B., & Joga, S. R. K. (2023). Phishing Detection System Through Hybrid Machine Learning Based on URL. *IEEE Access*, *11*, 36805–36822. https://doi.org/10.1109/ACCESS.2023.3252366

Korkmaz, M., Sahingoz, O. K., & DIri, B. (2020). Detection of Phishing Websites by Using Machine Learning-Based URL Analysis. *11th International Conference on Computing, Communication and Networking Technologies*, 1–7. https://doi.org/10.1109/ICCCNT49239.2020.9225561

Kumar, Y., & Subba, B. (2021). A lightweight machine learning based security framework for detecting phishing attacks. *International Conference on COMmunication Systems and Networks*, 184–188. https://doi.org/10.1109/COMSNETS51098.2021.9352828

Mohamed, G., Visumathi, J., Mahdal, M., Anand, J., & Elangovan, M. (2022). An Effective and Secure Mechanism for Phishing Attacks Using a Machine Learning Approach. *Processes*, *10*(7), 1–14. https://doi.org/10.3390/pr10071356

Mostafa, A. M., Ezz, M., Elbashir, M. K., Alruily, M., Hamouda, E., Alsarhani, M., & Said, W. (2023). Strengthening Cloud Security: An Innovative Multi-Factor Multi-Layer Authentication Framework for Cloud User Authentication. *Applied Sciences*, *13*(19), 1–24. https://doi.org/10.3390/app131910871

Ozcan, A., Catal, C., Donmez, E., & Senturk, B. (2021). A hybrid DNN–LSTM model for detecting phishing URLs. *Neural Computing and Applications*, *35*(7), 4957–4973. https://doi.org/10.1007/s00521-021-06401-z

Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2021). Phishing Email Detection Using Natural Language Processing Techniques: A Literature Survey. *Procedia Computer Science*, *189*, 19–28. https://doi.org/10.1016/j.procs.2021.05.077

Salloum, S., Gaber, T., Vadera, S., & Shaalan, K. (2022). A Systematic Literature Review on Phishing Email Detection Using Natural Language Processing Techniques. *IEEE Access*, *10*, 65703–65727. https://doi.org/10.1109/ACCESS.2022.3183083

Sanchez-Paniagua, M., Fidalgo, E., Alegre, E., &Alaiz-Rodriguez, R. (2022). Phishing websites detection using a novel multipurpose dataset and web technologies features. *Expert Systems with Applications*, 207, 1–16. https://doi.org/10.1016/j.eswa.2022.118010

Singh, S., Singh, M. P., & Pandey, R. (2020). Phishing detection from URLs using deep learning approach. *Proceedings of the 2020 International Conference on Computing, Communication and Security*, 16–19. https://doi.org/10.1109/ICCCS49678.2020.9277459

Tang, L., & Mahmoud, Q. H. (2021). A Survey of Machine Learning-Based Solutions for Phishing Website Detection. *Machine Learning and Knowledge Extraction*, *3*(3), 672–694. https://doi.org/10.3390/make3030034

Thakur, K., Ali, M. L., Obaidat, M. A., &Kamruzzaman, A. (2023). A Systematic Review on Deep-Learning-Based Phishing Email Detection. *Electronics*, *12*(21), 1–26. https://doi.org/10.3390/electronics12214545