Computer Systems Science & Engineering
DOI: 10.32604/csse.2023.024086
Article

Tech Science Press

Phish Block: A Blockchain Framework for Phish Detection in Cloud

R. N. Karthika*, C. Valliyammai and M. Naveena

Department of Computer Technology, Madras Institute of Technology Campus, Anna University, Chrompet, 600044, India
*Corresponding Author: R. N. Karthika. Email: karthirn@mitindia.edu
Received: 03 October 2021; Accepted: 03 December 2021

Abstract: The data in the cloud is protected by various mechanisms to ensure security aspects and user's privacy. But, deceptive attacks like phishing might obtain the user's data and use it for malicious purposes. In Spite of much technological advancement, phishing acts as the first step in a series of attacks. With technological advancements, availability and access to the phishing kits has improved drastically, thus making it an ideal tool for the hackers to execute the attacks. The phishing cases indicate use of foreign characters to disguise the original Uniform Resource Locator (URL), typosquatting the popular domain names, using reserved characters for re directions and multi-chain phishing. Such phishing URLs can be stored as a part of the document and uploaded in the cloud, providing a nudge to hackers in cloud storage. The cloud servers are becoming the trusted tool for executing these attacks. The prevailing software for blacklisting phishing URLs lacks the security for multi-level phishing and expects security from the client's end (browser). At the same time, the avalanche effect and immutability of block-chain proves to be a strong source of security. Considering these trends in technology, a block-chain based filtering implementation for preserving the integrity of user data stored in the cloud is proposed. The proposed Phish Block detects the homographic phishing URLs with accuracy of 91% which assures the security in cloud storage.

Keywords: Cloud server; phishing URLs; phish detection blockchain; safe files; smart contract

1 Introduction

The rising demand for cloud resources has pulled the majority of technology users including financial services. This might provoke the attackers and make cloud servers a hot spot for security attacks. Security of the documents in the cloud is in question due to deceptive and malicious content uploaded by fellow cloud users. Phishing attacks happen in many ways. Most phishing happens in emails. Angler phishing is another method through which phishing happens. It is a relatively new attack vector, social media offers a number of ways for attackers to trick people. Fake URLs, cloned websites, posts, tweets, and instant messaging are used to persuade people to divulge sensitive information or download malware. Attackers can use the data that people willingly post on social media to create highly targeted attacks. The main attack that is possible in a cloud computing environment is the usage of phishing URLs that can mislead or misguide the users of

the cloud. Since the main objective of phishing is to steal the data of the user, the attackers try to access the user's information without the user's knowledge. Most of phishing attacks start with a specially crafted URL. When clicked on, phishing URLs get directed to fake websites, download malware, or prompt for credentials. The fake URL looks very alike to the original URL and this tricks the user into using such websites. Cloud computing in simple terms means accessing and storing data over the interconnected network instead of the computer's hard drive.

The data or information that gets stored in the cloud is relatively safe. The data stored in the cloud is created by the cloud user but the Cloud Service Provider (CSP) has the ultimate control of the contents in the cloud. The people prefer cloud not just because it has file storage, but using cloud service also permits the user to share files and documents between different users. The cloud service can also be used for creating a backup for the data in order to protect important files. This way, even if anything happens to the computer, the files can be recovered from the cloud storage. Data sharing in cloud computing enables multiple participants to freely share the group data, which improves the efficiency of work in cooperative environments and has widespread potential applications [1]. In Spite of developing several encryption techniques, the security aspects of cloud storage continue to be difficult. A Blockchain is a list of growing records. It is a digital record of transactions.

The name comes from its structure Fig. 1, in which individual records, called blocks, are linked together in a single list, called a chain. Blockchains are used for recording transactions made with cryptocurrencies. A cryptocurrency is a digital asset that can be used as a medium of exchange. It follows and maintains a ledger or a record containing the individual coin ownership records in a digitally automated database. The database is encrypted using strong cryptography to secure, create and verify transaction records and their respective ownerships. It is neither issued or approved by any central authority and is thus termed as distributed ledgers.

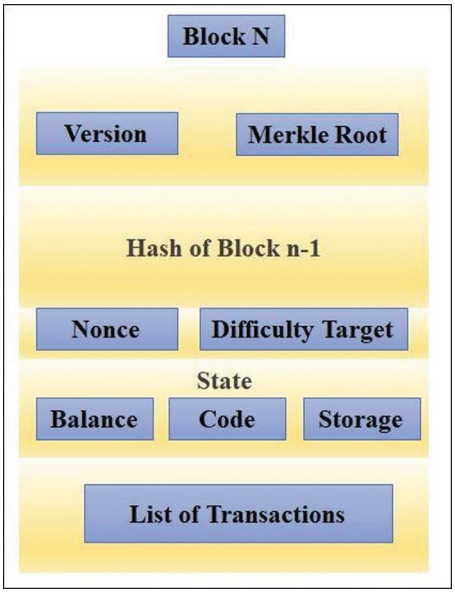

Figure 1: Structure of a block

The avalanche effect is a property in cryptography where if the input is changed slightly, the output changes many-fold. In terms of blockchain, any change in the content of one block would lead to drastic changes in the hash value of that block. As the hash value generated in successive blocks includes that of the previous block as well, any change in the content of a single block leads to a change in the hash

value of all the blocks. Thus, the blockchain becomes resistant to changes and updates in preserving the integrity of the data stored. The term "avalanche effect" was first used by Horst Feistel, although the concept was used in Shannon's diffusion. It is a primary design objective for a hash function that uses the "butterfly effect". It allows small changes to propagate rapidly into a huge effect through iterations of the hash algorithm. The Avalanche effect makes the blockchain completely resistant to changes due to which content once inserted as a block can only be viewed and not changed. After passing the security system provided by the cloud service, the user goes for the upload. The credentials of the user stored inside the blockchain again follow the avalanche effect that ensures non-repudiation. The data access and acquisition processes of the service provider can be modelled as a sequence of access records that present details regarding the data generated and utilized for the service, However, access records [2] are not useful when they cannot be trusted, and it is inadvisable to trust access records without receiving proper protection. It is crucial to guarantee the integrity and unforgeability of access records.

These security aspects have directed in choosing the blockchain as a platform for securing cloud servers. By embedding the blockchain framework into the cloud computing platform, a NutBaaS platform can leverage the deployment and management advantages of cloud service infrastructure to provide developers with convenient, high-performance blockchain ecosystems [3] and related services. A phish detecting blockchain has not been in use so far. Existing phishing URL blocking is done on the browser level using domain certifications. There also exists some utilities that let client applications check URLs against constantly updated lists of unsafe web resources to prevent the cloud users from becoming victims of phishing attacks. The idea of using blockchain to store phishing content is useful to trace back the criminal. The non-repudiation property of the cloud combined with the integrity of the documents in the blockchain helps in identifying the malicious users. As the content of the malicious documents is made visible, the users can be aware of phishing strategies that the criminals may indulge in. The proposed Phish block would reduce the responsibility of using safe browsers on the user's end. It acts as a utility between the cloud users and storage while insertions and updates, removing the necessity of a dual check during data access from the cloud.

The proposed Phish Block aims at differentiating between Safe homographic URLs and malicious homographic URLs from the data that is being stored in the cloud environment. Using a blockchain service to the cloud environment adds an extra layer of security which will be beneficial for the users. The proposed blockchain-based filtering implementation would ensure only legitimate documents get stored on a cloud. The Phish Block system blocks the uploaded malicious documents from entering the cloud by detecting URL-based phishing strategies.

2 Literature Review

Blockchain has become an integral part of various systems. A proposed structure of groupchain with group block and vice block is a scalable blockchain in fog computing. Transactions in the created environment are verified and approved by a leader group through a round robin mechanism which reduces the confirmation latency and transaction throughput. The implementation avoids selfish mining and prevent double spending [4]. Blockchain based CloudEx can be useful to resolve concerns like privacy, Reputation system and Transaction negotiation [5]. Policy Driven permissioned blockchain network has been designed for transport systems with a set of policies which contains the signing key of each user and these keys are associated with a policy set [6]. A permissioned blockchain based decentralized management has been used to ensure the big data in the process of managing the IoT. To increase the quality of the supplied data, a blockchain based token reward mechanism has also been used. This blockchain designed can be feasible even for enormous amount of data [7]. Blockchain-as-a-Service platform called NutBaaS has been developed as a layered architecture design that can provide blockchain

service to the cloud computing environments. Blockchain environments are also developed for ensuring security services. The blockchain can provide IoT security and address the confidentiality, integrity and availability of IoT security using a multi-layered approach. Secure Hash Algorithm 2 (SHA–2) algorithms is implemented using hash tables and merkle trees to ensure the security [8]. In addition to IoT security, cryptocurrency security has been provided using blockchain taxonomies like consensus protocols, smart contracts and forks. The immutable blockchain provides authorization access for all the transaction history and multi-token transactions [9].

A hybrid blockchain model created that ensures mutual authentication to enhance the security measures in the wireless sensor nodes. The hybrid model using base stations, cluster head nodes and ordinary nodes provide integrity, non-repudiation and elasticity [10]. To enable secure auditing, third party auditor can be replaced by blockchain based smart contract fair payment. Storage preserves the privacy and ensures that the parties need not interact during auditing [11]. Cloud environments are well known to use blockchains for preserving security. The blockchain containing the data can be preserved in a cloud computing environment. This cloud computing environment ensures data integrity by sending a block and response request to the cloud owner when data is stolen to ensure data integrity. This protection was provided using hash trees and encryption algorithms for fast and secure crypto transactions [12]. Many works have concentrated on the smart contracts for customizing blockchain for specific applications. A comprehensive overview of the Smart Contracts using Ethereum and Hyperledger blockchain frameworks has been explained along with the proposal of a six layered framework covering the key elements of the smart contracts [13].

A smart contract framework, with Access Control Contracts (ACCs), Judge Contract (JC) and Register Contract (RC) has been developed. Many techniques aim at preventing and detecting phishing attacks. The phishing attacks that come through internet and electronic mails can be averted by using a SAFE–PC (Semi Automated Feature Generation for Phish Classification) model that performs keyword extraction, feature engineering and natural language processing for filtration. SAFE-PC employs the fastest boosting algorithm as a classifier and handles real-world challenges with a portable feature selection, proving better performance than other filtering software like sopho and spam assassin [14]. To detect web phishing, an Adaptive Neuro Fuzzy system was designed using integrated features of text, image and frames in a layered approach. Adaptive Neuro-Fuzzy Inference System ANFIS feature classification, Support Vector Machine and K Nearest Neighbour are used to detect the hybrid qualities in phishing websites [15]. A SAFE-PC (Semi-Automated Feature generation for Phish Classification), a system to extract features, elevating some to higher level features, that are meant to defeat common phishing email detection strategies [16]. Phishing prediction is done on the commonly used set of 12 features that is obtained from third party studies. These features are the set of patterns of URL used in legitimate sites with the intention of phishing the site [17].

Executing a phishing attack gives more learning on post-occurrence of the attack as well. Extreme phishing attacks that have almost identical look and feel as those of the targeted legitimate websites has been created and demonstrated to evaluate effectiveness. 92% of the participants have found to be non-suspicious [18]. Discussions are done about the development of different kinds of updated URLs and updated contents to deceive people. It has been stated that not only URLs, but also logos and graphics are phished in spam mails, making it very difficult to detect the presence of phishing. Solution for phishing has been provided in 3 steps as prevention, detection and stakeholder training [19]. Extensive analysis of the unique characteristics that differentiates between phishing and spear phishing is done along with the detailed explanation about the lack of countermeasures to prevent spear phishing [20]. There are proposals with different detection techniques for various types of phishing including researches on the different types of phishing and spear phishing attacks [21].

3 Proposed Phish Block

The idea of using blockchain to store phishing content is useful to trace back the criminal. The non-repudiation property of the cloud combined with the integrity of the documents in the blockchain helps in finding out the malicious user. As the content of the malicious documents is made visible, the users can become more aware of phishing strategies that the criminals may follow. The proposed Phish block would reduce the responsibility of using safe browsers on the user's end. It acts as a utility between the cloud users and storage while insertions and updates, removing the necessity of a dual check during data access from the cloud. The proposed Phish Block system is used to filter the documents entering the cloud storage. The employed framework of smart contract algorithms identifies the documents with phishing content using a homographic phishing URL detector and withholds it in the blockchain which has the property of the avalanche effect. The enhanced Proof of Work (PoW) algorithm is used to choose the miner of the block, among the cloud users. The contents in the block are made visible to all the cloud users once the block gets mined. This makes legitimate cloud users aware of phishing. Once the filtering gets completed, the remaining documents which are not added in the blockchain are considered as safe. Once there is an input detected, the smart contract framework running on the latest block of Phish Block. In case of the successful compilation of the contracts, the block containing the content of the malicious input document gets mined through the Enhanced PoW into the Phish Block. The remaining documents are uploaded to the cloud server. The Phish Block module first detects phishing URLs using the smart contract framework. Those documents that are found with the presence of phishing URLs are created as blocks. The Enhanced PoW creates a process for the users to mine the block upon the validity of a smart contract. The compiled contract is considered valid only upon the successful detection of a phishing URL. Deploying an invalid contract does not lead to the mining of the block. In this mining process, one user is selected as the miner, whichever user is selected as the miner will add the document with the malicious content to the blockchain.

The safe documents are encrypted using Secure Hash Algorithm (SHA-3) and sent to the respective cloud storage centers. As shown in Fig. 2, the input documents are obtained from cloud users.

Once the document is added to the blockchain, its block contents are made visible to all the users. The documents that were identified as safe are for encryption. A user-friendly interface is created to communicate with the blockchain and display the content of the documents that were added to the blockchain in the Enhanced PoW process. The safe documents are encrypted and sent to the cloud servers. Fig. 3 shows the flow of functionalities incorporated by the modules of detection inside the contract framework. Check homograph is responsible for checking whether the given URL is a homographic phishing URL or not. Three strategies of homographic URL detection are considered, namely, Internationalized Domain Name in Applications (IDNA), Typosquatting, and Reserved Character Usage (RCU). If any of the URL detection techniques returns true, then URL is directly considered to be phishing; otherwise sent for detecting chained phishing.

i) IDNA-It extracts the domain name from the given URL and checks for homographs using multilingual characters.
ii) Typosquatting-It extracts the domain name from the given URL and checks for homographs using deceptive spellings.
iii) RCU-It searches for the reserved characters on an URL that can be used as an escape for redirections.

Web crawling is done with the URLs detected from the documents for the web page content to find the possible hyperlinks. The found hyperlink URLs are checked for homographs recursively until no more hyperlink is found.

Algorithm 1: PHISHING DETECTION BLOCKCHAIN

// n represents the number of documents, doc represents the list containing the n documents, a block is the genesis block

Input:

List of documents (LD)

Output:

Phish Block

Procedure:

```
input 'n' documents in a list 'doc'
initialize i = 0
initialize j = 1
 do
 Scan doc[i]
 if (doc[i].CHECK_HOMOGRAPH() == TRUE) then
    create_block()
    block[j] = doc[i]
    increment j
    PhishBlock_PoW
 end
if(User_solves_PoW AND user_details IS VALID) then
    user ←minercreate_block()
  if (add_block == TRUE) then
     initialize k = i
     for k in n − 1:
     doc[k] = doc[k + 1]
     increment k
  end
end
increment i
while (doc.next != NULL)
display Block Contents
initialize x = 0
do:
  encrypt_doc = Encrypt doc[x]
  add encrypt_doc to cloud
  increment x
  while (doc.next != NULL)
end
```

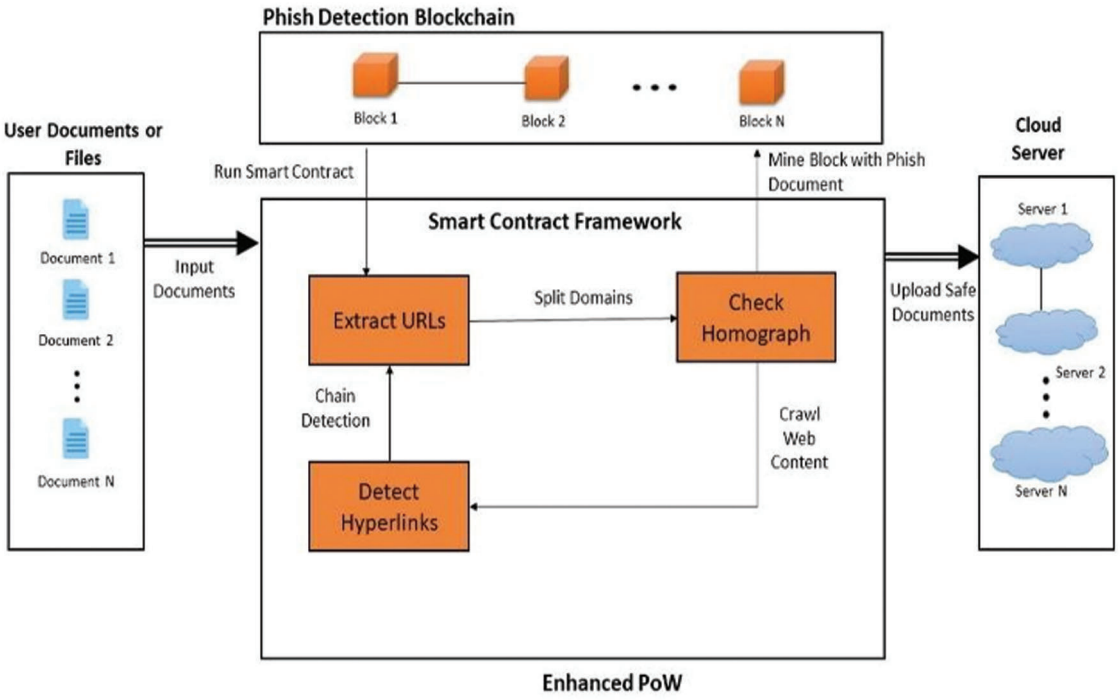

Figure 2: Proposed phish detection blockchain architecture

As shown in Algorithm 1, the phish block algorithm takes a random list of documents as input from the dataset. The dataset contains 200 documents with and without phishing URLs. Phishing URLs are coinedor referred from wandera and phishbank for creating .txt files. The input documents in the list are considered as those from the cloud users trying to be uploaded onto the cloud. The list is traversed to obtain each document. The document is scanned for the presence or absence of a phishing URL. The presence of phishing URL confirms the validity of the contract and deploys the same for creating a block with the document content as the entry. Each block contains a nonce value, hash value, difficulty, coin base, timestamp, file data, gas limit and configuration details as fields. The content found to have the phishing URL is placed as the file data of the block. The count of blocks in the Phish Block increases as the number of malicious documents in the input list increases. Once the block is created, the Proof of Work employed by the Ethereum blockchain chooses the miner for Phish Block and the block gets mined.

Algorithm 2: CHECK_HOMOGRAPH

Input: Document, D with x lines

Output: Boolean value

Procedure:

initialize i = 0

do:

if*(D[i].is_URL == true)***then**

RCU = pattern_search(URL, reserved characters)

if*(RCU == true)*

return true

(Continued)

Algorithm 2: (continued).

```
    end
    DN = extract_domain_name(URL)
    IDNA = verify_punycode(DN)
    if(IDNA == true)then
        return true
    end
    TS = autocorrection_probability(DN)
    if(TS >= 0.4)then
        return true
    end
    MC = Multichain_Phishing(URL)
    if(MC is_not_null)then
        D.append(MC)
    end
end
    while(is_lesser_than x)
    return false
```

The input list is traversed again to remove the documents corresponding to the created blocks from the list. The entire process is repeated until the list reaches the end. All the malicious documents are removed from the list after the completion of these iterations, having the list size to either remain the same or decreased. The list size remains the same if there are no malicious documents among the list of selected documents. The list size gets reduced according to the respective number of malicious documents in the randomly selected input list. The documents remaining in the list are safe to be uploaded in the cloud. An encrypted version of these files is ready for cloud storage. Algorithm 2 shows the traversal of each and every inputted document to check the presence of phishing URL. This punycode convertor is used for detecting IDNA, a text distance algorithm called Jaccard distance for detecting Typosquatting and Boyer More string search algorithm for RCU detection. Punycode convertor has been incorporated by various browsers for phish detection and is the conversion tool that can be incorporated in python. It is a simple and efficient transfer coding syntax designed to be used with IDNA [22]. Under autocorrection using python, models like error model and candidate model are available. Error model sticks to the proximity of the characters in the keypad for suggesting autocorrection whereas candidate models use distance calculation of the words against a dictionary. Under text distance calculation there are several categories like edit-based, token-based, sequence-based, phonetic-based and so on. Taking into consideration computational efficiency, Jaccard distance algorithm, a token-based technique has been used. Reserved characters like ';', ',' and '@' are used in URLs for redirection and are considered as escape characters. An URL with any other domain name followed by reserved characters can be a phishing URL [23]. A Naive pattern search algorithm can detect the same.

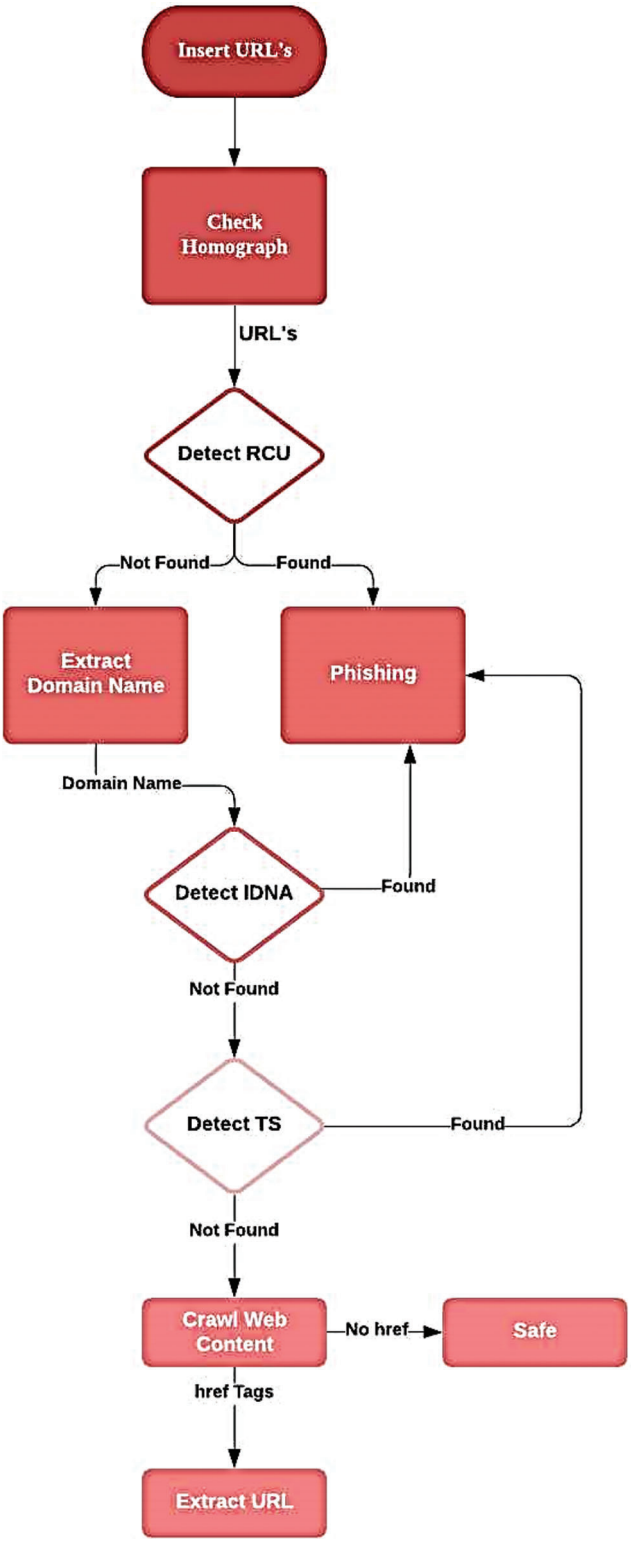

Figure 3: Flow of detection by proposed phish block smart contract

Algorithm 3: MULTICHAIN_PHISHING

//href_list represents the list of hyperlinks obtained from the source code of the web content corresponding to the extracted URL, EU

Input: Extracted URL, EU from Document, D

Output: href_list

Procedure:

initialize href_list = empty
crawl the HTML source code of EU
extract hyperlinks
href_list.add(hyperlinks)
return href_list

As shown in Algorithm 3, the multichain phishing is implemented using recursive calls. An empty list is given as the input for web crawling and web contents are stored as .txt files into the list if hyperlinks are detected. The items in the returned list are appended to scan for homographic phishing URLs again and again, until no such is found. Web crawling uses Beautiful Soup, a web crawling framework in Python. Beautiful Soup enables the detection of multichain phishing. It also enables extracting URLs from the webpages. It is used to visit webpages corresponding to the extracted URL and crawl through the webpage for retrieving other available hyperlinks.

The proposed mathematical procedure aims at materializing the efficiency of the selected features for phish detection. The features used by Phish Block have been selected based on the ability to integrate with blockchain. The impact of the feature detection on the performance of Phish Block and certain other logical factors such that the system can perform a decent rate of detection. This set of mathematical equations is used to prove the same.

Let w be the document that needs classification as safe or phishing,

$$w \xrightarrow{X} \{safe,\ \ phishing\}$$

Then X is the anti-phishing Phish Block system that considers features $f_i \in w$, such that,

$$w = \sum_{i}^{n} f_i,\ n > 0 \tag{1}$$

w is a non-empty set. The input document is having a minimum of single feature for classification. Depending on the complexity of the features, feature frequency assessment is done and depicted as; $X = \{x_1, x_2, \ldots x_n\}$ which assigns the result for each $f_i \in w$ as,

$$y = \begin{cases} 1, & phishing \\ 0, & safe \end{cases}$$

then as,

$$x_i : f(w) \rightarrow y$$

In order to analyze the frequency of phishing features the Frequency Information (FI) approach is selected. By determining the FI value, a statistical information on phishing inputs are known. The phishing input is represented by the total Document Files (DF). The undertaken list of features F_{URL} is

considered to evaluate FI as,

$$FI = F_{URL} / \sum DF \tag{2}$$

$$0 \ll FI \ll 1$$

where; 0 means no occurrence and, 1 means found in all occurrences

The threshold for FI values is set as;

$$1.00 \ll FI_{phishing} \ll 0.10$$

$$0.00 \ll FI_{safe} \ll 0.20$$

The two possible classifications are taken as phish division and safe division. In order to ensure the feature-to-feature inter correlation, a heuristic evaluation is represented as;

$$M = \frac{k.a}{\sqrt{k + k(k-1)b}} \tag{3}$$

where, M is considered as the merit value of considered features,

k is the number of features,

a is the mean feature-division correlation and,

b is the average feature-feature inter correlation

Based on the obtained value, 0 indicates safe status and >0 indicates chances of phishing. A threshold of 20% occurrence is fixed to confirm phishing status.

For checking International Domain Name in Application,

$$F_1 = \begin{cases} 0, & Unity\ language \\ >0, & Otherwise \end{cases}$$

For checking Typosquatting in domain names,

$$F_2 = \begin{cases} 0, & TS < 0.4 \\ >0, & Otherwise \end{cases}$$

For finding reserved character usage,

$$F_3 = \begin{cases} 0, & no\ reserved\ characters \\ >0, & Otherwise \end{cases}$$

For finding the presence of URL in extracted web page content,

$$F_4 = \begin{cases} 0, & no\ redirections \\ >0, & Otherwise \end{cases}$$

The standards of classifications are applied to the system to analyze the accuracy. The used term are shown in Tab. 1.

Table 1: Terms and description

Terms	Description
PSR	Phish success rates
PFR	Phish failure rates
SSR	Safe success rates
SFR	Safe failure rates
P_P	Phishing sites classified as phishing
Ps	Phishing sites classified as safe
S_P	Safe sites classified as phishing
Ss	Safe sites classified as safe
P	Total number of phishing sites
S	Total number of safe sites

$$\text{PSR} = \frac{P_P}{P} \times 100 \tag{4}$$

$$\text{PFR} = \frac{P_S}{P} \times 100 \tag{5}$$

$$\text{SFR} = \frac{S_P}{S} \times 100 \tag{6}$$

$$\text{SSR} = \frac{S_S}{S} \times 100 \tag{7}$$

Accuracy of detection is calculated by,

$$A = \frac{P_P + S_S}{S + P} \times 100 \tag{8}$$

The reliability of the Phish Block is calculated using Mathew's Correlation Coefficient (MCC). When the chosen MCC approaches the value 1, the detection is considered chosen to perfection.

$$MCC = \frac{P_P \times S_S - P_S \times S_P}{\sqrt{(P_P + P_S)(P_P + S_P)(P_S + S_S)(S_P + S_S)}} \tag{9}$$

The standards of classifications are applied to the system to analyse the accuracy.

4 Implementation Details

4.1 Experimental Setup

The experimental setup for the implementation of the proposed Phish Block involves-Metamask, Rinkeby, Remix, Truffle and Go Ethereum. Metamask acts as a gateway to access Phish Block through Firefox browser. Rinkeby is a test network used to collect ethers for compiling the contracts in Phish block, accessed via Metamask. Remix is the Integrated Development Environment used to run and deploy the Phish Block smart contract. Truffle framework is used to integrate the driver code in python with the ethereum smart contract in solidity. Go Ethereum is the client where the accounts can be created

and smart contracts for Phish Block can be implemented through Truffle suite. web3.py library is used for interacting with the blocks. SHA-3 algorithm is used for encrypting the safe documents. It is connected to metamask to interact with the private ethereum blockchain. The interaction with the console is tested.

Rinkeby test network is used for collecting ethers Fig. 4 shows the collected ethers are then transferred to the metamask account. Web3 is used to call ethereum smart contracts using python.

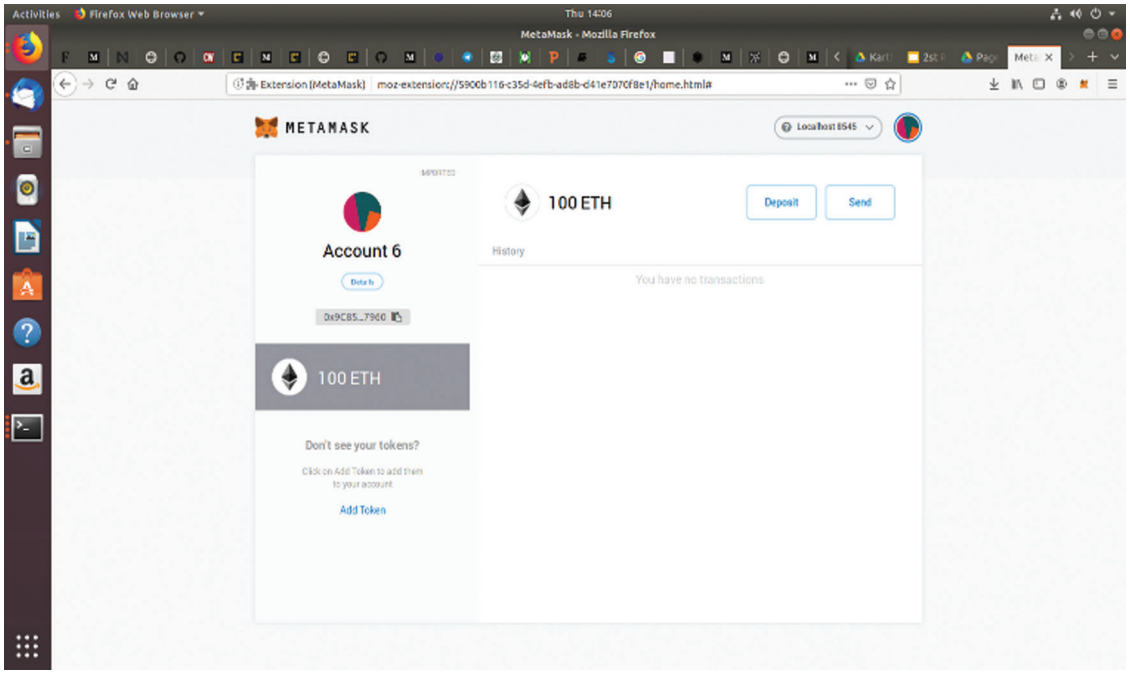

Figure 4: Ethers transferred to the accounts of the private blockchain

The web interface uses python, Java Script Object Notation (JSON), Java Script and google scripts API.

4.2 Dataset

Dataset has been generated with and without URLs. Documents containing URLs consist of safe and phishing URLs. Phishing URLs are framed as homographs belonging to all the three strategies for detection. Phishing URLs are coined or referred from wandera and phishbank for creating .txt files.

Total documents generated-200 (Safe-50, Phishing-150)

Documents with no URL-25

Documents with safe URL-15

Documents with multiple safe URLs-10

Documents with phishing URL (IDNA)-25

Documents with phishing URL (Typosquatting)-25

Documents with phishing URL (RCU)-25

Documents with multiple phishing URLs (IDNA)-25

Documents with multiple phishing URLs (Typosquatting)-25

Documents with multiple phishing URLs (RCU)–25

Implementation of Phish Block

As shown in Fig. 5, the geth client is accessed via truffle framework. When the complete set of 200 documents are given as input the files with safe content are encrypted and those with phishing content are added as blocks. The block content, type of phishing as well as the block times are displayed as shown in Fig. 6.

```
naveena@naveena-Lenovo-ideapad-330-15IKB:~/Desktop/final_project$ truffle develop
Truffle Develop started at http://127.0.0.1:8545/

Accounts:
(0) 0x08af63d4c9b58929280a4843129d775b46446765
(1) 0x923f664d4c23500be03bd0009888acee0b9b1f2e
(2) 0x8a95d295d5c5831eb5ea316cf88fcdc98c5ed105
(3) 0x3bd3cf7e496b4d7e19452671bf084cdf8a4b4cd5
(4) 0x5152d7ad355058ce06c88656909d8895d591194e
(5) 0x0078569c4736296b06b57d99292ff2e6c3614812
(6) 0x4e127d5b83928cbb455a917823fac6d2454cb6d9
(7) 0xd3725eb5bc762490b080b404fa2a6ef45e7d51f7
(8) 0xbc1728de2d79b0505b1ae2d419bb457b17a96966
(9) 0x43e125e4ac0cc1fa4ba4818e7bc66d16486b10b8

Private Keys:
(0) 015da7f52a3f6fe9303f352794a45d13ec756265107bceabe0ebc977d8ff272c
(1) 7ee193dc3c808e8e5b201c9a0bff2739fd4ceabf8ffa5545703a83ef6f3be535
(2) 425f78715cb9a158452d0628c166f505fe7fe6e9be83414bdffda1fdc94e7d17
(3) e2e1582d9f4a843773ab7e3e61bfb56083531cb69594ee9ce52bddd549c1e62d
(4) 3869b77213e992675af93c7be8ec126ce90e8b50b96b8f0e85fbf78327de7168
(5) 7f827ec15fc68da0dde2ee2295ac71425d9589e013fa594994b4295488ef8962
(6) 51123fc6b553b1fc94803d0e0228b63e84b97c18b4e066c30e6837964687a700
(7) b2d1684e6ba501d4df702b77906056aebe5ceebc6c5fd745d2634ee1737f87d3
(8) 449941c889f47ba8b7f8b5be76fe2f1baac64866f06701d71a0f8ade8ba28513
(9) c174e093b5bcefcb186405cad8604ea60afe691558b95f58955d0e763f9e28cb

Mnemonic: vocal guide excess private sugar lounge because sure motor viable body almost

⚠️  Important ⚠️  : This mnemonic was created for you by Truffle. It is not secure.
Ensure you do not use it on production blockchains, or else you risk losing funds.
```

Figure 5: Compiling smart contracts using truffle

```
200) m_office
RCU Phishing- http://www.office.com@fake-auction.com
File Number 200: Phish content
Content of the added block:
Collaborate for free with online versions of Microsoft Word, PowerPoint, Excel, and OneNote. Save
documents, spreadsheets, and presentations online,


At Microsoft our mission and values are to help people and businesses throughout the world realize
their full potential.
Microsoft account · Microsoft 365 · Download Center · Xbox

Office 365 Login | Microsoft Office

http://www.office.com@fake-auction.com


Block added with block time: 1617616548


Total Number of Files: 200
Detected Safe Files: 68
Detected Phish Files: 132
```

Figure 6: Classification of 200 safe and phishing files by phish block

An Ethereum interface has been developed for the interaction to the front-end. It allows the safe documents to get uploaded to google drive (considered as cloud). The proposed Phish block system is tested through 4 test cases. The number of the input documents are increased gradually for each case. Test case 1 takes 50 documents as input, test case 2 takes 100 documents as input, test case 3 takes 150 documents as input and test case 4 takes 200 documents as input. These documents are randomly chosen by the driver program from the generated dataset containing 200 documents. For evaluating the system, two different measures have been considered. First measure to be calculated is the accuracy of phish detection by phish block which is shown in Tab. 2, for each test case. The phish block system gives approximately 91% accuracy upon the generated dataset as an average. Fig. 7 also clearly shows the misclassification of 9% of the actual files.

Table 2: Accuracy of phish block

Test cases (files)	Input documents		Detection by PHISH BLOCK		Accuracy (%)
	Phish	Safe	Phish	Safe	
Test case 1 (50 files)	37	13	34	16	91.89
Test case 2 (100 files)	75	25	68	32	90.67
Test case 3 (150 files)	113	37	103	47	91.15
Test case 4 (200 files)	150	50	132	68	88
Average accuracy:					90.42 = ∼91

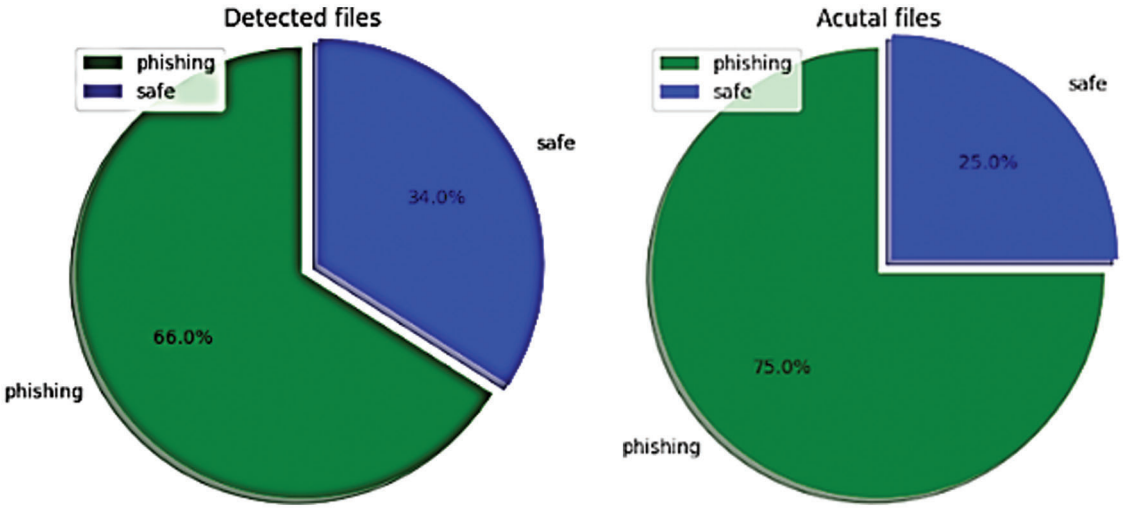

Figure 7: Displaying the ratio of the phishing files and safe files

The other measure to be calculated is the block time. It is the time taken for the consecutive blocks to get mined to the blockchain. Block time for Phish Block is calculated by the difference between the timestamps of the successive blocks getting added to the blockchain. Timestamp of the added block is obtained through a function call to the smart contract that returns back the current block's timestamp to the driver program. The overall block time for the last block added to Phish Block is 327 s. The obtained value is the difference between the timestamp values of the first block and the last block added in the test case 4.

5 Result Analysis

The results shown in Fig. 8, shows the misclassified inputs at each test case. Maximum number of misclassifications has occurred in test case 4 and test case 1 has proven to be most accurate among the four. The misclassification has taken place because of the highly misspelt URLs that escaped typosquatting detection. As our typosquatting detection is based on the autocorrection tool, high levels of variations cannot be detected. Nevertheless, the efficacy of the system remains unaffected as highly misspelt URLs can be detected by human eyes easily. Phish Block has achieved the maximum accuracy of 91.89 percentages in test case 1 and a minimum accuracy of 88 percent in test case 4 owing to the number of misclassifications and the number of documents given as input in the respective test cases. The system gives 90.67 percentage accuracy when tested with 100 files test case 2 and an accuracy of 91.15 percentage in test case 3.

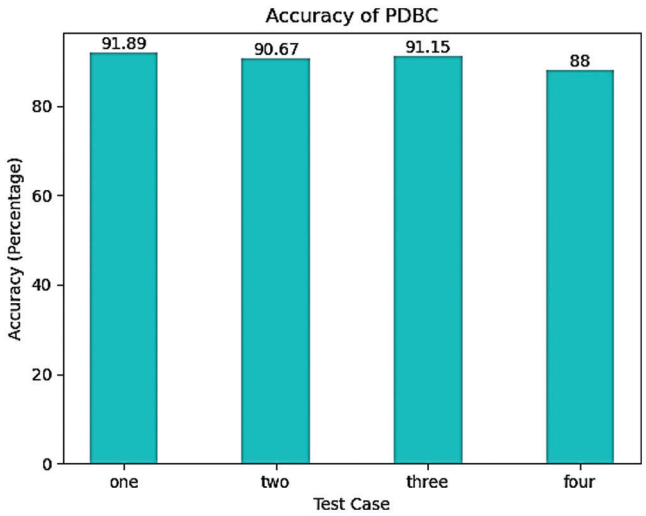

Figure 8: Accuracy shown by phish block for different test cases

The Fig. 9 shows the block times of successive blocks added to the phish block during each test case. The block times have been recorded at an interval of every 25 files given as input. The files are plotted against their respective block times to get themselves mined into Phish block. The block time is recorded as zero seconds when the successive blocks are added to the Phish block during the same timestamp or the corresponding files are not added to the chain. The files without phishing URLs are not added to the blockchain and their block time values are zero seconds. It is evident that the block times of the blocks added in test case 1 has experienced maximum fluctuations among non-zero values. This may be accounted for the initial run of the deployed contracts and the high accuracy of detecting the phishing files. Test cases 2 and 4 have recorded block time values each having the least mode values of 147 s and 148 s respectively. Test cases 2 and 4 are also similar in the minimum block times as 11 s and 10 s for 25 files respectively. Test case 3 hits the peak with the highest block time as 175 s. Test case one hits the maximum among the lowest block times as 20 s. Reduction in the block time can be attributed to the successive blocks having the same timestamp values. The blocks had begun to get mined to Phish block faster at the end of test case 4. Observing the arrived pattern, the blocks start getting mined slowly and shoot up speed towards the end leading to decrease in the block time values. It is observed that test case 3 has taken the maximum block time for processing files from 50–75. It can be inferred that the 25 files consisted of multichain phishing analysis or alternative safe files. As the test cases are continued, the system has seemed to grow consistent. Test cases 2 and 4 have taken consistent block time for the

blocks mined during 25–50 and 75–100 respectively. Test case 1 has seemed to achieve maximum block time during the initial batch of 0–25 files. It is noticed that in all the four cases maximum value always occurs till the first 100 files, after which the system works faster. Analyzing the trends, it is inferred that block time of an added block might shoot up after a long stagnant time of without mining scenario. Anyhow, the presence of malicious files in the inputs obtained from the cloud users is not playing the majority in real time, thus the block times of the blocks are expected to be high.

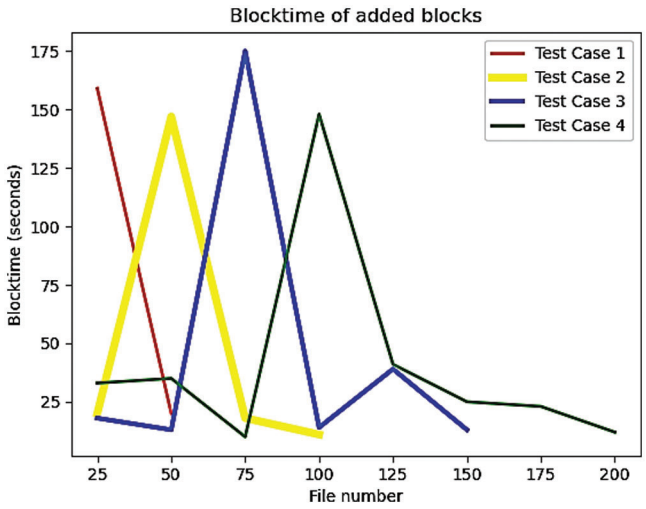

Figure 9: Graph representing the block time for different test cases

6 Conclusion and Future Work

The proposed Phish block has been implemented on a private ethereum blockchain is successful in retaining the homographic phishing URLs (~91 percent), whereas documents that are undetected contained other types of phishing URLs (~9 percent). The proposed phish block also has some limitations the default difficulty level meant for ethereum platform has been an overhead in block time which is modifiable where the Phish Block algorithm is implemented on a self-configured private blockchain that the CSP could afford and results in better block time. Not only providing safety to the cloud storage and the cloud consumers but adding the Phish Block system as a utility would give an added value as a trust factor in the Service Level Agreement provided by the CSP. Therefore, the proposed phish block can bring a massive impact on the customer's selection of the cloud services among the competitive CSPs. Future works include making the Phish Block more resistant to documents with phish content through the incorporation of detecting other types of phishing with adequate compensation for the overhead.

Acknowledgement: I would like to extend my thanks to Big Data Analytics Research and Development Laboratory in Department of Computer Technology, MIT Campus, Anna University Chennai for providing the technical support.

Funding Statement: The authors received no specific funding for this study

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] J. Shen, T. Zhou, D. He, Y. Zhang, X. Sun *et al.,* "Block design-based key agreement for group data sharing in cloud computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 6, pp. 996–1010, 2019.

[2] J. Xue, C. Xu and Y. Zhang, "Private blockchain-based secure access control for smart home systems," *KSII Transactions on Internet and Information Systems*, vol. 12, no. 12, pp. 6057–6078, 2018.

[3] W. Zheng, Z. Zheng, X. Chen, K. Dai, P. Li *et al.,* "Nutbaas: A blockchain-as-a-service platform," *IEEE Access*, vol. 7, pp. 134422–134433, 2019.

[4] K. Lei, M. Du, J. Huang and T. Jin, "Groupchain: Towards a scalable public blockchain in fog computing of iot services computing," *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 252–262, 2020.

[5] S. Xie, Z. Zheng, W. Chen, J. Wu, H. -N. Dai *et al.,* "Blockchain for cloud exchange: A survey," *Computers and Electrical Engineering*, vol. 81, no. 106526, pp. 1–12, 2020.

[6] Y. Mu, F. Rezaeibagha and K. Huang, "Policy-driven blockchain and its applications for transport systems," *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 230–240, 2020.

[7] M. Zhaofeng, W. Lingyun, W. Xiaochang, W. Zhen and Z. Weizhe, "Blockchain-enabled decentralized trust management and secure usage control of iot big data," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4000–4015, 2020.

[8] D. Minoli and B. Occhiogrosso, "Blockchain mechanisms for IoT security," *Internet of Things*, vol. 1, no. 2, pp. 1–13, 2018.

[9] A. Ghosh, S. Gupta, A. Dua and N. Kumar, "Security of cryptocurrencies in blockchain technology: State-of-art, challenges and future prospects," *Journal of Network and Computer Applications*, vol. 163, no. 102635, pp. 1–35, 2020.

[10] Z. Cui, X. U. E. Fei, Z. Shiqiang, C. Xingjian, C. Yang *et al.,* "A hybrid blockchain-based identity authentication scheme for multi-wsn," *IEEE Transactions on Services Computing*, vol. 13, no. 2, pp. 241–251, 2020.

[11] W. Hao, Q. Hong, Z. Minghao, W. Xiaochao *et al.,* "Blockchain-based fair payment smart contract for public cloud storage auditing," *Information Sciences*, vol. 519, pp. 348–362, 2020.

[12] P. Wei, D. Wang, Y. Zhao, S. K. S. Tyagi and N. Kumar, "Blockchain data-based cloud data integrity protection mechanism," *Future Generation Computer Systems*, vol. 102, pp. 902–911, 2020.

[13] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han *et al.*, "Blockchain-enabled smart contracts: Architecture, applications, and future trends," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 11, pp. 2266–2277, 2019.

[14] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang and J. Wan, "Smart contract-based access control for the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1594–1605, 2019.

[15] M. A. Adebowale, K. T. Lwin, E. Sánchez and M. A. Hossain, "Intelligent web-phishing detection and protection scheme using integrated features of images, frames and text," *Expert Systems with Applications*, vol. 115, pp. 300–313, 2019.

[16] C. N. Gutierrez, K. Taegyu, D. C. Raffaele, A. Jeffrey, G. Dan *et al.,* "Learning from the ones that got away: Detecting new forms of phishing attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, pp. 988–1001, 2018.

[17] C. Pham, L. A. T. Nguyen, N. H. Tran, E. N. Huh and C. S. Hong, "Phishing-aware: A neuro-fuzzy approach for anti-phishing on fog networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 1076–1089, 2018.

[18] C. M. R. da Silva, E. L. Feitosa and V. C. Garcia, "Heuristic-based strategy for phishing prediction: A survey of url-based approach," *Computers & Security*, vol. 88, no. 101613, pp. 1–20, 2020.

[19] R. Zhao, S. John, S. Karas, C. Bussell, J. Roberts *et al.,* "Design and evaluation of the highly insidious extreme phishing attacks," *Computers & Security*, vol. 70, pp. 634–647, 2017.

[20] I. Vayansky and S. Kumar, "Phishing–challenges and solutions," *Computer Fraud & Security*, vol. 18, pp. 15–20, 2018.

[21] L. Allodi, T. Chotza, E. Panina and N. Zannone, "The need for new antiphishing measures against spear-phishing attacks," *IEEE Security & Privacy*, vol. 18, no. 2, pp. 23–34, 2020.

[22] M. Uwais, A. Sharma, A. Kumar and L. Singh, "Homograph attack warning system," *International Journal of Innovative Technology and Exploring Engineering*, vol. 10, no. 2, pp. 1–4, 2020.

[23] R. Bouslimi and J. Akaichi, "Automatic medical image annotation on social network of physician collaboration," *Network Modeling Analysis in Health Informatics and Bioinformatics*, vol. 5, no. 1, pp. 1–17, 2016.