PRIVACY PROTECTION OF USER BROWSING DETAILS AND UNSAFE URL DETECTION

Praisy Evangelin A[1], Jeenath Laila N[2]

[1]*PG Scholar, Dept. Of Computer Science, GCE, Tirunelveli Tamil Nadu, India.*
[2] *Assistant Professor, Dept. Of Computer Science, GCE, Tirunelveli, Tamil Nadu, India.*

Abstract:

Malicious URL (or) malicious website is a common and serious threat to cyber security. Naturally, backbone of information management is search engine. Nevertheless, the flooding of large number of malicious websites on search engine has posed tremendous threat to our users. Nowadays exiting systems mostly used to detect malicious websites focus on specific attack. And also that available browser extensions based on blacklist are not Highly be used to detect countless websites. Therefore, it is essential that any data leaving the client side should be effectively masked such that the server cannot interpret any valuable information from the masked data. Here propose the first PPSB service. It provides strong security guarantees that are missing in existing SB services. In particular, it inherits the capability of detecting unsafe URLs with the help of blacklist generation, while at the same time protects both the user's privacy (browsing history) and blacklist provider's proprietary assets (the list of unsafe URLs). In this work, proposed a model which encrypts the users' sensitive data to prevent privacy from both outside analysts and service provider. Also, completely supports selective aggregate functions for online user behaviour analysis and guaranteeing differential privacy. AES algorithm is used for encrypting users' online behaviour data. Implementation is done with the help of browsing based web application development using ASP.NET as front end and SQL for back end. And also extend this blacklist storage with keyword based malicious prediction approach.

Keywords — **Malicious URL Detection, Blacklist Creation, History encryption using AES, URL Recommendation, Key verification, History Access.**

I. INTRODUCTION

Malicious SB service provider wants to know whether a user is visiting a particular web page, e.g., some political news. One way to achieve this is that the web browser sends all the visited URLs to a remote server, either in the plaintext, hash value or encrypted format. However, this behaviour can be detected by monitoring and analysing the browser, e.g., using the taint analysis technique. Specifically, in order to track a particular URL the SB service provider can insert the 32-bit hash prefixes of all its decompositions, e.g., c01e362f, and then push this newly updated prefix filter to the clients. Later, once a user visits the web page (or similar URLs that share some decompositions), the matched hash prefixes would be sent to the remote SB server. Based on the prior knowledge of the prefix filter (i.e., the mappings between the hash prefixes and their corresponding URLs), the server can infer the URL (or domain) navigated by the user. It provides strong security guarantees that are missing in existing SB services. In particular, it inherits the capability of detecting unsafe URLs, while at the same time protects both the user's privacy (browsing history) and blacklist provider's proprietary assets (the list of unsafe URLs). This method has some disadvantages such as; creating metadata of URLs fails when the server receives multiple prefixes for a URL and there is a chance that other URLs may have the same hash prefixes this makes collision between URLs.

A malicious party might leverage PPSB to degrade the client-side user experience, like inserting a number of fake or safe URLs or increasing the server-side delay. To address this potential issue, PPSB provides a flexible mechanism for users to add or remove blacklist providers. Admin could add the fake URL and keyword to this blacklist storage. User can also allowed suggesting the malicious website details regarding black list. In this system malware detection system uses a supervised machine learning approach for discovering malwares. The SVM based malware detection system extends the idea of signature based detection system with a combination of behaviour monitoring approach. It utilizes static and dynamic analysis of malwares by taking the run time traces of the executables. Image based malicious detection also provide to compare the image features based on original website and malicious website. This model also provides search data security which encrypts the users' sensitive data to prevent privacy from both outside analysts and the aggregation service provider. Also, completely supports selective aggregate functions for online user behaviour analysis and guaranteeing differential privacy.

II. RELATED WORK

Cui et al. Propose the fingerprint techniques and locality-sensitive hashing to convert the problem of NDD into the keyword search. We then adopt an efficient multi-key searchable encryption scheme, which requires only one encrypted query from the user even the data are from multiple content providers encrypted with different keys. Initially, the CPs encrypts their data items with a standard encryption scheme, e.g., AES. And the ISP will attach these metadata together with corresponding encrypted data items and deliver them to ISs that are close to the users. The user will be able to access the encrypted data by the CP and generate encrypted queries for secure NDD with her own key. Stage Two - Secure Detection: In order to locate near duplicate data items from the encrypted in-network storage, the user needs to generate an encrypted query tq from the interested data with her own key and send it to the nearest IS. Stage Three - In order to further improve the quality of query results, the IS needs to filter out the potential false positives from the located candidates. Hence, the IS and the ESP will conduct a secure evaluation procedure via Yao's garbled circuits protocol. In particular, the ESP (as the garbled-circuit generator) prepares a garbled circuit for the IS (as the garbled-circuit evaluator), where the circuits function checks if the query item and each candidate are indeed near-duplicates based on a particular distance metric. For those qualified candidates, the IS will return them directly to the user.

Cui et al. Propose a privacy-preserving malware detection system for Android, in which the privacy (or assets) of phone vendors, users, and security service providers are protected. It detects malicious apps in phone vendor's app stores and on users' phones, without directly sharing apps, apps' runtime behaviours, and malware signatures to other parties. Proposed design goal is to allow the SPs perform malware detection without holding the apps. Thus, the above-mentioned features, including the permissions, behavioural footprints, and file hashes are necessarily minimal leakage about the apps. The SPs cannot (easily) recover the original value thanks to the one-way property of the selected cryptographic hash functions. Even a SP may guess the original values using the brute-force attack, the cost of the process and the value of the recovered features do not deserve this attempt. Remember the code of the app is not shared and cannot be recovered at all.

Shengshan et al. Propose the first practical system for privacy-preserving cross-media retrieval by utilizing trusted processors. Proposed scheme enables secure aggregation of the data from distinct parties, and secures canonical correlation analysis (CCA) over collaborated data to obtain semantic models. Verification mechanisms are designed to defend against active attacks from a malicious adversary. SGX provides data sealing and unsealing functions to protect private data outside the boundary of an enclave. The sealing facilities provide each enclave with keys that are unique to the processor and the enclave digest. When the enclave is closed, the private data can be sealed and stored on the platform in the encrypted form. It aims at addressing the problem of how to securely conduct cross-media retrieval over encrypted data strategically using a remote SGX-enabled server.

Wang et al. Proposed system investigates the problem of training high quality word vectors over large-scale encrypted data with the privacy-preserving collaborative neural network learning algorithms. It contains multiple participating users (i.e., data owners), a central remote server S and a crypto service provider CSP. More specifically, there are n users in total, denoted as ui (i = 1 . . . n), each of which owns a private data file fi and wants to perform collaborative neural network learning with all other participating users. In other words, they will contribute their private data files in the encrypted form to the central remote server. After receiving the encrypted data, the remote server S performs training over the contributed data and produces high-quality word vectors along with a model, which can be used later for various natural language processing (NLP) tasks.

Yuan et al. Propose the frequency hiding query scheme which allows the server to see the flattened query distribution only. To improve the scalability, further design the result sharing query scheme, which processes a small portion of query points and shares the results with other nearby points. Besides, we set up a strict constraint to carefully select query points to achieve “as-strong-as-possible” guarantees. It consists of three parties, i.e., the client of the authorized user, the data owner who has the source dataset, and the server in the public cloud. Before using our system for secure similarity join queries, a setup procedure is required. The data owner will build an encrypted LSH-based index I, encrypt the dataset S, and upload them to the cloud server. After that, the client will pre-process the query set Q and generate secure tokens t from LSH hash values of query points. When the server receives t, it will process them over I to get a set of ids of collided data points. When the number of data point’s hash collisions to a query point is greater than a pre-defined collision threshold α, these two data and query points will be considered as a candidate pair.

III. EXISTING SYSTEM

Phishing is the fraudulent attempt to obtain sensitive information such as usernames, passwords and credit card details, often for malicious reasons, by disguising as a trustworthy entity in an electronic communication. Phishing attack can be implemented in various form like Email phishing, Website phishing, spear phishing, Whaling, Tab napping, Evil twin phishing etc. To avoid this phishing attack various anti-phishing solutions should be use. There are various anti phishing solutions such as Blacklist, heuristic, visual similarity, machine learning etc.

This is most commonly used approach in which list of phishing URL is stored in database and then if URL is found in database, it is known as phishing U and gives warning otherwise it is called legitimate. This approach is easy and faster to implement as it see URL is in db or not. But limitations are small change in URL is sufficient to bypass the list based technique and frequent update of list is necessary to counter new attack.

Phishing imitates the characteristics and features of emails and makes it look the same as the original one. It appears similar to that of the legitimate source. The user thinks that this email has come from a genuine company or an organisation. This makes the user to forcefully visit the phishing website through the links given in the phishing email. These phishing websites are made to mock the appearance of an original organisation website. The phishers force user to fill up the personal information by giving alarming messages or validate account messages etc so that they fill up the required information which can be used by them to misuse it. They make the situation such that the user is not left with any other option but to visit their spoofed website.

In the training phase, we should use the labelled data in which there are samples such as phish area and legitimate area. If we do this then classification will not be a problem for detecting the phishing domain. To do a working detection model it is very crucial to use data set in the training phase. We should use samples whose classes are known to us, which means the samples that we label as phishing should be detected only as phishing. Similarly the samples which are labeled as legitimate will be detected as legitimate URL. The dataset to be used for machine learning must actually consist these features. There so many machine learning algorithms and each algorithm has its own working mechanism which we have already seen in the previous chapter. The existing system uses any one of the suitable machine learning algorithms for the detection of phishing URL and predicts its accuracy. The existing system has good accuracy but it is still not the best as phishing attack is a very crucial; we have to find a best solution to eliminate this. In the currently existing system, only one machine learning algorithm is used to predict the accuracy, using only one algorithm is not a good approach to improve the prediction accuracy

IV MALICIOUS URL DETECTION WITH HISTORY ENCRYPTION APPROACH

A malicious party might leverage PPSB to degrade the client-side user experience, like inserting a number of fake or safe URLs or increasing the server-side delay. To address this potential issue, PPSB provides a flexible mechanism for users to add or remove blacklist providers. Admin could add the fake URL and keyword to this blacklist storage. User can also allowed suggesting the malicious website details regarding black list. In this system malware detection system uses a supervised machine learning approach for discovering malwares. The SVM based malware detection system extends the idea of signature based detection system with a combination of behavior monitoring approach. It utilizes static and dynamic analysis of malwares by taking the run time traces of the executables. Image based malicious detection also provide to compare the image features based on original website and malicious website. This model also provides search data security which encrypts the users' sensitive data to prevent privacy from both outside analysts and the aggregation service provider. Also, completely supports selective aggregate functions for online user behavior analysis and guaranteeing differential privacy.

In this system malware detection system uses a supervised machine learning approach for discovering malwares. The SVM based malware detection system extends the idea of signature based detection system with a combination of behavior monitoring approach. Also, completely supports selective aggregate functions for online user behavior analysis and guaranteeing differential privacy. A malicious party might leverage PPSB (Privacy Preserving Safe Browsing) to degrade the client-side user experience, like inserting a number of fake or safe URLs or increasing the server-side delay. To address this potential issue, PPSB provides a flexible mechanism for users to add or remove blacklist providers. Admin could add the fake URL and keyword to this blacklist storage. User can also allowed suggesting the malicious website details regarding black list.

The purpose of website security is to prevent these (or any) sorts of attacks. The more formal definition of website security *is the act/practice of protecting websites from unauthorized access, use, modification, destruction, or disruption*. Effective website security requires design effort across the whole of the website: in your web application, the configuration of the web server, your policies for creating and renewing passwords, and the client-side code. The proposed project detects Malicious or Fake URLs to prevent the users accessing from Unsafe URLs. Also provide secure encryption method to encrypt the user search data before stored on the server.

ALGORITHM

AES Encryption

The AES cipher is also known as the block cipher. No successful attack has been reported on AES. Some advantages of AES are easy to implement on 8-bit architecture processors and effective implementation on 32-bit architecture processors. In addition, all operations are simple (e.g, XOR, permutation and substitution). AES encryption is performed in multiple rounds. Each round has four main steps including sub-byte, shift row, mix column and add round key. Sub-byte is the substitution of bytes from a look-up table. Shift row is the shifting of rows per byte length. Mix column is multiplication over Galois field matrix. Finally, in the add round key step, the output matrix of mix column is XORed with the round key. The number of rounds used for encryption depends on the key size. For a 128-bit key, these four steps are applied to 9 rounds, where the 10th round does not consider the mix column step. Since all steps are recursive, decryption is the reverse of encryption.

Algorithm Procedure

The algorithm begins with an **Add round key** stage followed by 9 rounds of four stages and a tenth round of three stages. This applies for both encryption and decryption with the exception that each stage of a round the decryption algorithm is the inverse of its counterpart in the encryption algorithm. The four stages are as follows:

1. Substitute bytes
2. Shift rows
3. Mix Columns

4. Add Round Key
The tenth round simply leaves out the **Mix Columns** stage. The first nine rounds of the decryption algorithm consist of the following:
1. Inverse Shift rows
2. Inverse Substitute bytes
3. Inverse Add Round Key
4. Inverse Mix Columns
Again, the tenth round simply leaves out the **Inverse Mix Columns** stage. Each of these stages will now be considered in more detail.

Support Vector Machine

Support Vector Machine (SVM) is a supervised algorithm based on machine learning which can be used for both classification and regression problems. However, it's far ordinarily used in classification work. In this work, plot each data item as a point in n-dimensional space with the value of every feature being the count of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well. Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is best for segregates the two classes (hyper-plane/ line). The hyperplane is the line with the biggest margin to both groups.

Support Vector Machines has higher effectiveness in higher dimensional spaces. It is even very effective on data sets where number of dimensions is greater than the number of samples. This is mainly because of the kernel trick, which we talk about it later. Further advantages of Support Vector Machines are the memory efficiency, speed and general accuracy in comparison to other classification methods like k-nearest neighbour or deep neural networks.

Step1: Malicious URLs and keywords have been collected and stored on blacklist storage.

Step 2: For the collection of Malicious URLs number of features could be used like URL length, the number of dots, ip Address, SSL connection, at symbol(@) and dash symbol(-).

Step 3: The selected features identified from URL then stored on blacklist.

Step 4: User could enter the URL or Keyword for searching details.

Step 5: Input data classified with trained dataset with the help of SVM classifier.

Step 6: SVM classifier returns either an URL is phishing or non-phishing.

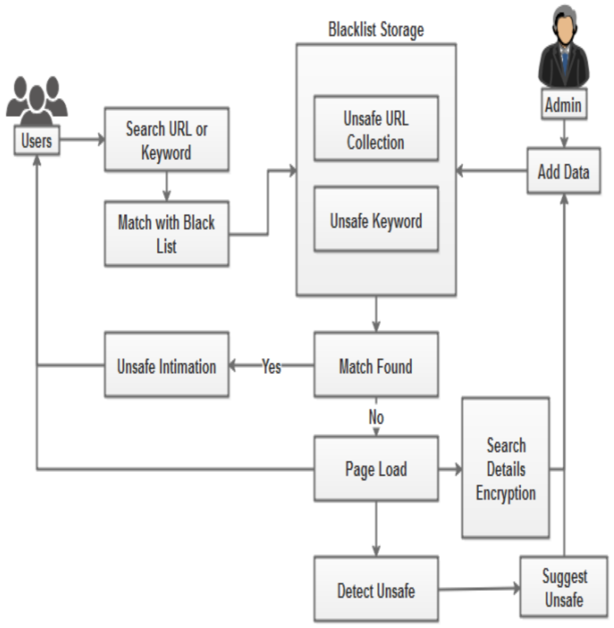

Fig 4.1 proposed work Architecture

The blacklist providers have the incentive to collect and publish unsafe URLs and keywords for helping users to avoid websites that contain malware or phishing and deceptive content, e.g., for the better marketing purpose. Assume that these blacklist providers and the corresponding PPSB servers are semi-trusted. They faithfully perform the designed procedures, i.e., the database preparation/ update. But they should not be aware of the queried URLs from users. In proposed work a service provider that owns a high-quality blacklist, which may be more frequently updated or simply contains more items. User also allowed to directly sharing blacklists with servers in an uncontrollable way could make these dataset be obtained by every user, including the competitors. The client needs to search into the list of unsafe URLs and keywords. The searched URL could be matched with blacklist providers. Once match could be found, that will show the malicious alert to the user. Otherwise the page will be loaded to show the details to searched user. In further the searched URLs or Keywords are stored in encrypted format, which will not reveal the users sensitive information to the server or unknown person.
The proposed work consists of the following

modules,

- Framework Construction
- URL Encryption
- User Registration and Login
- Unsafe URL Detection
- History Access
- Malicious URL Suggestion

MODULE DESCRIPTIONS

FRAMEWORK CONSTRUCTION

The detection of malicious URLs limits web-based attacks by preventing web users from visiting malicious URLs and warning web users prior to accessing content located at a malicious URL. Thus, malicious URL detection protects computing system hardware/software from computer viruses, prevents execution of malicious or unwanted software, and helps avoid accessing malicious URLs web users do not want to visit. This proposed framework uses SVM classification models to detect a malicious URL and categorize the malicious URL as one of a phishing URL.

URL ENCRYPTION

The blacklist storage models by using a set of training data (unsafe URLs and keywords) and machine learning algorithms. The training data includes a known set of unsafe URLs and a known set of malicious keywords. This framework also supports URL encryption process, to avoid the unauthorized prediction of URL details. In blacklist storage, keyword and URL will be encrypted and stored in the intermediate. AES encryption algorithm used for encrypting data before stored on blacklist. This encryption also provides the security for user search history.

USER REGISTRATION AND LOGIN

Users have to register with their name, password and Email id. These details will be saved in the database. The user have to login with the name and password. The entered data will be compared with the available data. If match found, the user can proceed. If no match found, the user have to re enter the details again. This process will helps to protect from unauthorized access in search engine and also helps to predict the user who add the blacklist.

Once the login procedure is succeeded, the user can search details using URLs and keywords. The user will enter a URL or keyword in the search box and click the submit button. When the user clicks the search button, the request was processed and related details are shown to the user.

UNSAFE URL DETECTION

The verification of URLs and keywords is very essential in order to ensure that user should be prevented from visiting malicious websites. SVM mechanisms have been proposed to detect the malicious URLs. One of the basic features that a mechanism should posses is to allow the fake URLs that are requested by the client and prevent the malicious URLs before reaching the user. This is achieved by notifying the user that it was a malicious website. The techniques extract features associated with the known URLs, and use the machine learning algorithms to train the classification models to detect and categorize an unknown malicious URL. A database updation is performed every time the systems come across a new URL. Here, the new URL will be matched and tested with every previously known malicious URL in the black list. The update has to be made in black list whenever system comes across a new malicious URL. This also allows users to provide suggestions to add malicious URLs.

HISTORY ACCESS

All the users of the same application will not be allowed to access the user search data. Privileges are given by the administrator to the users. A malicious user in this case can be any user in the network who is not authorized to access the users search data but can intrude by using others credentials. This problem of intrusion can be overcome by providing Secret Key Sharing process generated by the system which will be forwarded only to authorized users. The intruder cannot be able to access the database without the shared secret key. If user wants to access search history, they will share request to the admin for access permission. After getting permission from admin, they will allow accessing their search history in plain text format. This approach will enhance the security of history protection process.

MALICIOUS URL SUGGESTION

In proposed work, the URL suggestion process could be implementing to enhance the performance of blacklist storage. When user finds any malicious URL during searching process, they will allow to suggestion process. Here user should send URL details to admin, to add blacklist storage. This frequent update in blacklist improves the performance of unsafe (or) malicious URL detection.

V CONCLUSION

In this proposed work, implement a Malicious URL Detection process using machine learning techniques. This focuses on detecting unsafe website URLs and keywords with the help of encrypted blacklist storage. According to few selected features can be used to differentiate between legitimate and malicious web pages. These selected features are many such as URLs and Keywords. In proposed work a service provider that owns a high-quality blacklist, which may be more frequently updated or simply contains more items. User also allowed to directly sharing blacklists with servers in an uncontrollable way could make these dataset be obtained by every user. With the help of efficient classification approach will detect the fake websites accurately and prevent the users from accessing that websites. This also provides the secure encryption approach avoid the unknown access of search history. The security is provided to the search data which has been stored in the database.

REFERENCES

[1] Cui, Helei, Xingliang Yuan, Yifeng Zheng, and Cong Wang. "Towards Encrypted In-Network Storage Services with Secure Near-Duplicate Detection." IEEE Transactions on Services Computing (2018).

[2] Cui, Helei, Yajin Zhou, Cong Wang, Qi Li, and Kui Ren. "Towards Privacy-Preserving Malware Detection Systems for Android." In 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS), pp. 545-552. IEEE, 2018.

[3] Hu, Shengshan, Leo Yu Zhang, Qian Wang, Zhan Qin, and Cong Wang. "Towards private and scalable cross-media retrieval." IEEE Transactions on Dependable and Secure Computing (2019).

[4] Wang, Qian, Minxin Du, Xiuying Chen, Yanjiao Chen, Pan Zhou, Xiaofeng Chen, and Xinyi Huang. "Privacy-preserving collaborative model learning: The case of word vector training." IEEE Transactions on Knowledge and Data Engineering 30, no. 12 (2018): 2381-2393.

[5] Yuan, Xingliang, Xinyu Wang, Cong Wang, Chenyun Yu, and Sarana Nutanong. "Privacy-preserving similarity joins over encrypted data." IEEE Transactions on Information Forensics and Security 12, no. 11 (2017): 2763-2775.

[6] Ramezanian, Sara, Tommi Meskanen, Masoud Naderpour, Ville Junnila, and Valtteri Niemi. "Private membership test protocol with low communication complexity." Digital Communications and Networks (2019).

[7] Keelveedhi, Sriram, Mihir Bellare, and Thomas Ristenpart. "DupLESS: server-aided encryption for deduplicated storage." In Presented as part of the 22nd {USENIX} Security Symposium ({USENIX} Security 13), pp. 179-194. 2013.

[8] Armknecht, Frederik, Jens-Matthias Bohli, Ghassan O. Karame, and Franck Youssef. "Transparent data deduplication in the cloud." In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 886-900. ACM, 2015.