File-Based Deception Technology for Impeding Malicious Users

Navyashree R[1], Dr Guruprakash C D[2], Dr M Siddappa[3]

[1]Dept. of Computer Science and Engineering, Sri Siddhartha Institute of Technology, Karnataka, India.
[2]Dept. of Computer Science and Engineering, Sri Siddhartha Institute of Technology, Karnataka, India.
[3]Dept. of Computer Science and Engineering, Sri Siddhartha Institute of Technology, Karnataka, India.

Abstract - *When adversaries have already successfully obtained access to a host while eluding intrusion detection systems, file-based deception methods can be utilized as an enhanced security barrier. Whenever adversaries tries to access files, they are discovered. This research paper has concentrated on the usage of user data files as decoys. It is anticipated to be successful in finding malicious individuals since launching an attack without access to system files is extremely difficult. Legitimate users also may unintentionally view false files. This problem is addressed in this paper by proposing a hidden interface. Legitimate users gain access to files via the hidden interface. The hidden interface may also be used to conceal sensitive files from the standard interface. The suggested approach has a minimal performance overhead, is effective against numerous attack scenarios, and is functional since it does not cause false alerts for genuine users.*

Key Words: Deception technology, file system, honeypot.

1.INTRODUCTION

Deception technology (also known as honeypot technology) is an information security strategy that identifies, deflects, or counterbalance cyber threats by deploying a false system or information [1]. It is believed to be more successful in insider threats, social engineering, and 0-day assaults than standard perimeter or signature-based intrusion detection and anomaly detection systems [2].

When malicious attackers have successfully acquired access to a host, file-based deception techniques can be used to stop them. When intrusion detection/prevention systems fail to detect/prevent unauthorized users, file-based deception technology acts as an extra security barrier. It may be used to identify rogue individuals and prohibit them from obtaining sensitive data. When malevolent users or software access the false user data files, it is called an intrusion indication. It is extremely difficult to carry out an assault without access to even a single file. Even if there are temporary attacks that do not modify any files, this does not indicate that they will not read any files. It is critical to obtain knowledge about the victim system in order for assaults to be successful.

Though it is supposed to be extremely powerful, there is indeed a significant disadvantage to using fake records to identify malevolent users: false alarm. Deception technologies are projected to have a low false alarm rate in general [2]. Fake files, on the other hand, may cause an excessive number of false alerts [3]. Legitimate users might unintentionally view fake files.

In this paper, we present the hidden interface. The hidden interface is being used to retrieve legitimate files rather than fake files. Fake files are not visible using the hidden interface. As a result, genuine users as well as system utilities that use the veiled interface cannot cause false warnings. Malicious users might attempt to acquire sensitive data. We may also conceal crucial files using the hidden interface.

2. RELATED WORKS

2.1 Deception technology

The deception technique has been used to a variety of system entities. Network-based deception techniques deflect cyber assaults by displaying false network entities such as false network setups, servers, services, and messages [4]. They make it difficult for adversaries to locate the actual target within the network perimeter. Host-based deception methods have been developed in order to fool attackers who have acquired access to a host. To entice adversaries [2] fake databases [5, 6], passwords [7, 5], accounts [5, 8] are employed.

One of the host-based deception strategies is the use of false files. When an opponent accesses a fraudulent file, it is notified as a possible attack [3], [9]–[11]. One of the issues addressed in past studies is how to make false files appear as authentic as conceivable. HoneyGen is presented as a solution to this problem[12]. Fake files are often used in the detection of ransomware [13]. These tactics are successful in increasing the efficacy of alluring opponents, but we have discovered no previous work on dealing with false alarms caused by fraudulent files.

Former file-based deception methods simply offered simulation, however this one enables not just simulation, but additionally dissimulation and a combination of these methods. It increases the likelihood of effective deception by utilizing these characteristics.

2.2 Different file views

The notion of displaying different files to various users on the same host may appear similar to what virtual

machines provide. The virtual machines as well as containers, on the other hand, are not meant for deception. Other virtual machines on the same host might not be affected if one of the virtual machines is attacked, however the data in the affected virtual machine might be disclosed or destroyed. Furthermore, it lets users to manually adjust the access interface of files, whereas virtual machines as well as containers do not provide such fine-grained control.

2.4 Intrusion detection systems

Intrusion detection systems (IDSs) are broadly classified as network-based IDSs (NIDSs) [14–15] and host-based IDSs (HIDSs) [16–18]. NIDSs monitor network packets, whereas HIDSs monitor host actions to identify malicious behaviour. They often identify harmful behaviour by matching the signature of malicious actions [17] or by detecting aberrant behaviours that differ from the known normal profile [18]. Artificial intelligence has also been investigated in IDSs [14–16]. IDSs are often used to deter malicious users from entering hosts and interfering with routine operations.

3. PROBLEM STATEMENT

The goal of deception technologies is to keep a cybercriminal who has infiltrated a network from causing significant harm. The technology generates traps or deceptive decoys that seem like actual assets. No security mechanism can prevent all network attacks however deception technology could provide attackers with a false feeling of security through making them think they have achieved a foothold within your network. Here the objective is to make sure the attacker will not get their hands on important sensitive files and even If the intruder has compromised the system only false file which is a decoy will be available to attacker.

4. REQUIREMENT SPECIFICATION

4.1 ECLIPSE IDE FOR JAVA DEVELOPMENT

Eclipse is a significant tool in computer programming. It is an integrated development environment (IDE) with a workspace [19] and a large extendable plug-in enabling environment creation. It is often written using Java programming which is used to create java-based applications. It is also used to create programmes in other programming languages such as Ada, C, C++, C#, COBOL, PHP, Python, and others. The original Eclipse codebase was derived from IBM VisualAge. It is indeed a software development tool kit that includes several packages. Eclipse [20] was among the IDEs that worked with GNU Classpath.

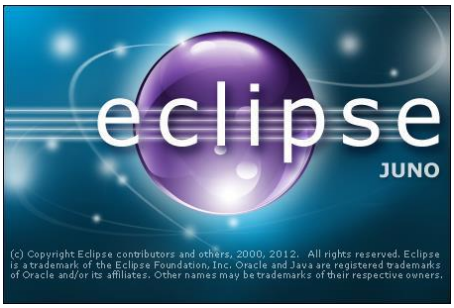

Fig -1: Eclipse IDE

4.2 SQLYOG

SQLyog is a graphical user interface tool for the RDBMS MYSQL that was created by Webyog [21][22]. It may be released as both free software and commercial software. SQLyog's key features include the following:

- It supports a variety of formatting choices as well as intelligent code completion.
- On a spreadsheet-like interface, data manipulation operations such as INSERT, DELETE, and UPDATE may be done.
- Visual Schema Designer and query formatting are both supported

Fig -2: SQLyog

5. DESIGN

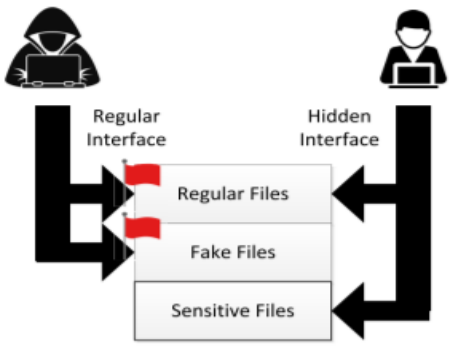

Fig -3: It provides a hidden interface in addition to the regular interface. The view of file varies with the interface.

As illustrated in Figure 3, In addition to the standard interface, it has a hidden interface. The files displayed by the concealed interface are genuine files utilised by genuine users and apps. The files displayed in the usual interface are distinct and are being monitored. It enables the administrator to specify the interface a file should be shown through. The administrator can additionally choose the sort of access that should be notified. In our system design, we use the web application as hidden interface. Initially the legitimate user logins and creates a profile through registering themselves. There are four types of users such as operator, manager, executive and helper. After each registration the user has to wait until administrator generates a unique secret key to the specified user. This key will be then stored in database. When the operator wants to shares a file he has to login using his account and upload the file and have to select the to whom the file will be shared to that is either the manager or executive. We can share the file to multiple people to view too. The shared file contents are also encrypted. Later the others that is like manager can view the file when he has both the public key and the secret key of the up loader of the file. If there is a mistake in providing the either of the keys a fake file will be generated. Algorithm used for key generation is AES.

It consists of the following modules:

5.1.1. Authority

Here the administrator will generate a key for each user after the user registration is completed.

5.1.2. Operator

In this module operator will initially generate content key and public key. After the generation of the keys the file which needs to be shared will be uploaded. Then to whom the file needs to be shared will also be selected. It can be shared with multiple people or users at the same time.

5.1.3. Executive/Manager

Here when the user login after registering themselves the files uploaded and shared by the operator is available in table. When the user selects a file to be viewed he has to enter public key of the file and the secret key of the operator to download the file. If either of the entered key is wrong then the file downloaded will be a fake one.

5.1.4. Helper

Here when the user logins on the interface only the list will be available of the shared documents. The user cannot view the content details.

5.2 ALGORITHM USED

AES ALGORITHM

The AES algorithm (also known as the Rijndael algorithm) is a symmetrical block cypher technique that turns plain text into ciphertext utilising keys of 128, 192, and 256 bits. To generate ciphertext, the AES algorithm employs a substitution-permutation, or SP, network with many rounds. The number of rounds is determined on the key size utilised. A 128-bit key size requires 10 rounds, a 192-bit key size requires twelve rounds, and a 256-bit key size requires fourteen rounds. Each one of these rounds requires a round key, however because the method only accepts one key, this key must be extended to obtain keys for each round, including round 0.

6. RESULT

If a genuine authorized person tries to access the file uploaded by the operator then the original uploaded file be downloaded and can be accessed by the user. If an unauthorized person tries to access the file that person will also be able to download the file but it would be a file which will be a fake. It will not contain any authenticate information. It is used to mislead the unauthorized person.

7. CONCLUSION

In this paper, we propose a file-based deception solution in this research to deter hostile users who have acquired access to a host. It offers the idea of employing a hidden interface to deal with the problem of false alarms. Legitimate users and apps can avoid accessing bogus files by using the concealed interface. It may also be used to conceal sensitive files from the standard interface, preventing unauthorized people from accessing them. We prototype and show that the overhead is insignificant. In the future, we will look at ways to prevent clever opponents from becoming aware and attempting to disrupt it.

REFERENCES

1) Wikipedia. Honeypot (Computing). Accessed: 2020.

2) D. Fraunholz, S. D. Anton, C. Lipps, D. Reti, D. Krohmer, F. Pohl, M. Tammen, and H. D. Schotten, “Demystifying Deception Technology:A Survey”, 2018.

3) J. Yuill, M. Zappe, D. Denning, and F. Feer, “Honeyfiles: Deceptive files for intrusion detection,” in Proc. 5th Annu. IEEE SMC Inf. Assurance Workshop, Jun. 2005, pp.116-122.

4) L. Spitzner, ``The honeynet project: Trapping the hackers,'' IEEE Secur. Privacy, vol. 1, no. 2, pp. 15-23, Mar. 2003.

5) D. Fraunholz, D. Krohmer, F. Pohl, and H. D. Schotten,"On the detection and handling of security incidents and perimeter breaches_A modular and flexible honeytoken based framework", in Proc. 9th IFIP Int. Conf. New Technol., Mobility Secur. (NTMS), Feb. 2018, pp. 1-4.

6) M. Bercovitch, M. Renford, L. Hasson, A. Shabtai, L. Rokach, and Y. Elovici, "HoneyGen: An automated honeytokens generator," in Proc. IEEE Int. Conf. Intell. Secur. Informat., Jul. 2011, pp. 131-136.

7) Juels and R. L. Rivest, "Honeywords: Making password-cracking detectable," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS). New York, NY, USA: ACM, 2013, pp. 145-160.

8) F. Araujo, K. W. Hamlen, S. Biedermann, and S. Katzenbeisser,"From patches to honey-patches: Lightweight attacker misdirection, deception, and disinformation,'' in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS). NewYork,NY, USA: Association for Computing Machinery, 2014, pp. 942-953.

9) B. M. Bowen, S. Hershkop, A. D. Keromytis, and S. J. Stolfo, "Baiting inside attackers using decoy documents,'' in Security and Privacy in Communication Networks, Y. Chen, T. D. Dimitriou, and J. Zhou, Eds. Berlin, Germany: Springer, 2009, pp. 51-70.

10) B. Whitham, "Canary files: Generating fake files to detect critical data loss from complex computer networks,'' in Proc. 2nd Int. Conf. Cyber Secur., Cyber Peacefare Digit. Forensic (CyberSec), Mar. 2013, pp. 1-10.

11) M. Lazarov, J. Onaolapo, and G. Stringhini, "Honey sheets: What happens to leaked google spreadsheets?'' in Proc. 9th Workshop Cyber Secur. Experimentation Test (CSET). Austin, TX, USA: USENIX Association, 2016, pp. 1-8.

12) B. Whitham, "Automating the generation of enticing text content for highinteraction honeyfiles,'' in Proc. Hawaii Int. Conf. Syst. Sci., Jan. 2017, pp. 1-10.

13) C. Moore, "Detecting ransomware with honeypot techniques,'' in Proc. Cybersecurity Cyberforensics Conf. (CCC), Aug. 2016, pp. 77-81.

14) S. Ganapathy, P. Yogesh, and A. Kannan, "An intelligent intrusion detection system for mobile ad-hoc networks using classification techniques,'' in Advances in Power Electronics and Instrumentation Engineering, V. V. Das, N. Thankachan, and N. C. Debnath, Eds. Berlin, Germany: Springer, 2011, pp. 117-122

15) D. S. Vijayakumar and S. Ganapathy, "Machine learning approach to combat false alarms in wireless intrusion detection system,'' Comput. Inf. Sci., vol. 11, no. 3, pp. 67-81, Jul. 2018.

16) G. Creech and J. Hu, "A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns,'' IEEE Trans. Comput., vol. 63, no. 4, pp. 807-819, Apr. 2014.

17) S. N. Chari and P.-C. Cheng, "BlueBox: A policy-driven, host-based intrusion detection system,'' ACM Trans. Inf. Syst. Secur., vol. 6, no. 2, pp. 173-200, May 2003

18) D.-Y. Yeung and Y. Ding, "Host-based intrusion detection using dynamic and static behavioral models,'' Pattern Recognit., vol. 36, no. 1, pp. 229-243, Jan. 2003.

19) "Eclipse Tools Project". archive.eclipse.org. Retrieved 28 December 2018.

20) https://projects.eclipse.org/projects/eclipse/releases/4.20.0

21) "SQLyog MySQL GUI 13.1.7 Released". sqlyog.com. Retrieved 2020-10-08.

22) "Webyog | Manage, Monitor and Secure Your MySQL Servers". www.webyog.com. Retrieved 2019-11-20.