IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 12, Special Issue 2, April 2023

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

|| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | SJIF: 8.423|| Volume 12, Special Issue 2, April 2023 ||

8th National Conference on Advanced Computing Technologies (NCACT'23)

Organized by

Department of CSE, Velammal Engineering College, Chennnai, India

System State-based Authentication

Mohamed Riyaz M. I. [1]

U.G. Student, Department of Computer Engineering, Velammal Engineering College, Chennai, Tamil Nadu, India[1]

ABSTRACT: Authentication is the process of proving the identity of a user to a computer system, so that the user may access either the system itself or certain functionality that would otherwise remain blocked. A common method of authentication is key-based authentication, which places a "lock" on the system and unlocksit by using a "key" that a user possesses. The key may be, for example, a password, a biometric signature, or an authentication token. These keys are "static", as they do not change during every session. Similar to how a physical key can be duplicated, so too can a virtual key. This paperproposesan alternative procedureby providing a "dynamic", randomly generated key during every session, by recording the "state" of the system's files before a session is closed. This is done by recording a trace of the system in question before it is terminated, saving it to an external location, and verifying the snapshot on the next authentication attempt. Thismethod should, in theory, dispel most attempts of tampering and forceful entry.This method could prove effective for systems that do not change their states when inactive, such as single-user computers when powered off. System state-based authenticationhas two components. The first creates a uniquely identifiable "hash" of the system using changes made to it during the current session. The second saves and retrieves this hash to/from an external location, an USB drive for example during the next login attempt.

KEYWORDS: Checksums, Snapshots, A-B root, System immutability, Full-Disk Encryption [FDE], Linux Unified Key System [LUKS]

I. INTRODUCTION

In an real world, attackers commonly take advantage of flawed authentication methods using many forms of attacks, namely social engineering (phishing), misconfigured systems, or brute-forcing their way into the system. These methods are dependent on the reliability of the system and its user(s), and a misstep by either party could prove detrimental to both. For instance, a user may unknowingly leak their personal credentials over a phishing attack, granting the attacker a perfect gateway for infiltrating the system. System state-based authentication overcomes this by providing a robust method of authentication that does not require the user to enter credentials manually.Rather, the state of the system before it is closed is used.

A rather rudimentary implementation of this process would be to calculate the checksum of all the files that have changed during the current session, store it in a safe location, and request the checksum during the next authentication attempt. The reliability of the checksum algorithm used and the "safe" location must be taken into consideration. The list of modified files would be tracked by using an "A-B root" method, a recent innovation, where the entire system is duplicated and a copy of the system is allowed to be modified by the user, while another copy is kept read-only so as to provide a rollback to the previous copy should a failure occur. Only the modified files are taken note of, in order to reduce the work done by the checksum process and thus minimise overhead.

The checksum or hashing algorithm used is the very crux of this method. A perfect algorithm would be virtually irreversible (where the checksum value cannot be used to determine the original content), and overcome all collisions while taking minimum time. While no perfect implementations exist, SHA-256 and its successors come close, being irreversible in nature by most if not all computers in existence. However, due to the large volume of data that changes between sessions, SHA-256 would prove slow and infeasible. Hence, the implementation uses a faster yet less effective hash algorithm, MD5.

The paper is organised as follows. Section II outlines how the snapshot is captured before the session is terminated. The steps of the process is explained in the flow diagram. Section III showcases examples of a basic implementation of the method. Section IV summarises the paper with a conclusion.

|| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | SJIF: 8.423|| Volume 12, Special Issue 2, April 2023 ||

8th National Conference on Advanced Computing Technologies (NCACT'23)

Organized by

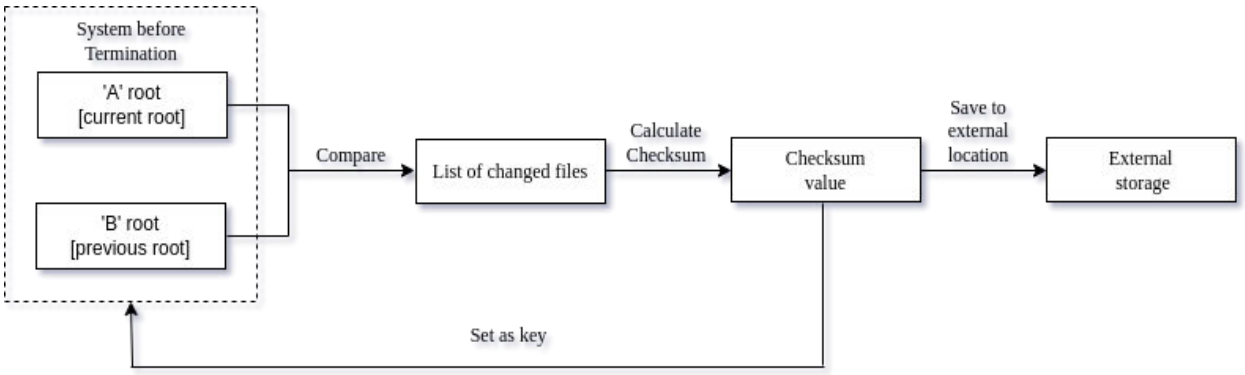
Department of CSE, Velammal Engineering College, Chennnai, India

II. METHODOLOGY

A-B root is a method of implementing system immutability. The system is duplicated into two copies. One copy, named the 'A' root, is exposed to the user, who may use it similar to any normal system. The other copy, named the 'B' root is kept read-only, for rollback purposes, should an error occur in the 'A' root. This guarantees an error-free work environment for both the user and the system.
System state-based authentication takes advantage of this technique for authentication. First a list of changed files during the current session is captured. The list of changed files is obtained simply by comparing the 'A' and 'B' roots. Once a list is obtained, the checksum of these files is calculated using MD5 or any relevant hash algorithm and written to an external storage location, in this case, a USB drive. Additionally, this checksum acts as a key to the system. The filesystem is encrypted using this key.

Full-Disk encryption is a method of obfuscating the entire disk using a key, commonly a passphrase or a special file carried around in a USB drive. The procedure of using a static key to unlock a system is flawed, as an attacker may potentially compromise this setup regardless of how complex the encryption method is. Additionally, once a key is compromised, the attacker may gain access in successive attempts. The proposed method overcomes this by generating a truly random key during every session, as a user's behaviour in a system is unpredictable. The proposed method also provides a fallback system that does not include the changes made by the attacker (by use of the 'B' root), should a system be compromised. This way, potential threats are minimised. The encryption technology used may vary depending in the nature of the system and supported functionality. This paper uses LUKS [Linux Unified Key Setup] as a means of setting up FDE.

The most significant advantages over traditional methods of authentication are [1] The key is randomised during every session. As the list of files and changes to them cannot be predicted or simulated with ease, the key generated is entirely random. This by itself takes care of several attack methods, such as brute-forcing. The same does not apply to traditional full-disk encryption. [2] The system can be verified to be unaltered between sessions. This increases the reliability provided by A-B root while maintaining the traditional method of authentication. In a traditional method of FDE, should an attacker succeed in obtaining access, no trace is left behind. In the proposed method, however, access is hard to obtain and traces are captured by the authentication mechanism itself. [3] The implementation provides a fallback to an unmodified system in the event of a compromise, a feature that the traditional method lacks.



III. EXPERIMENTAL IMPLEMENTATIONS

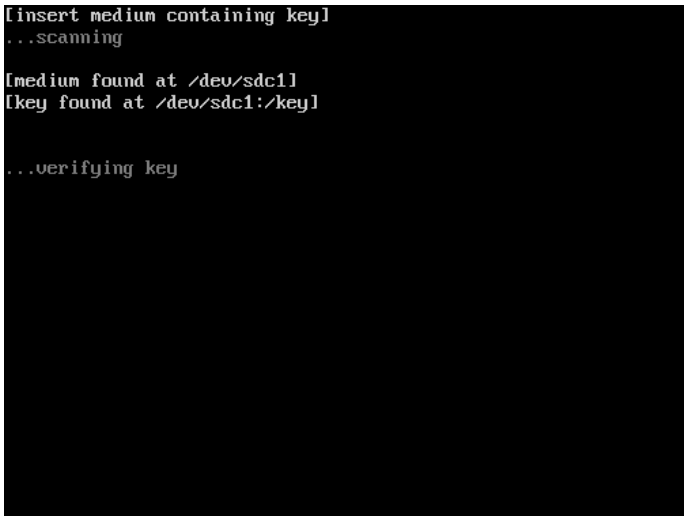
Figures 1(a), 1(b), and1(c) show the boot-up process and the verification process of the system. The system waits for an external storage disk containing the keyfile in 1(a). It verifies the key, if found, with the key written to the disk in 1(b). Boot continues in 1(c) as normal.

|| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | SJIF: 8.423|| Volume 12, Special Issue 2, April 2023 ||

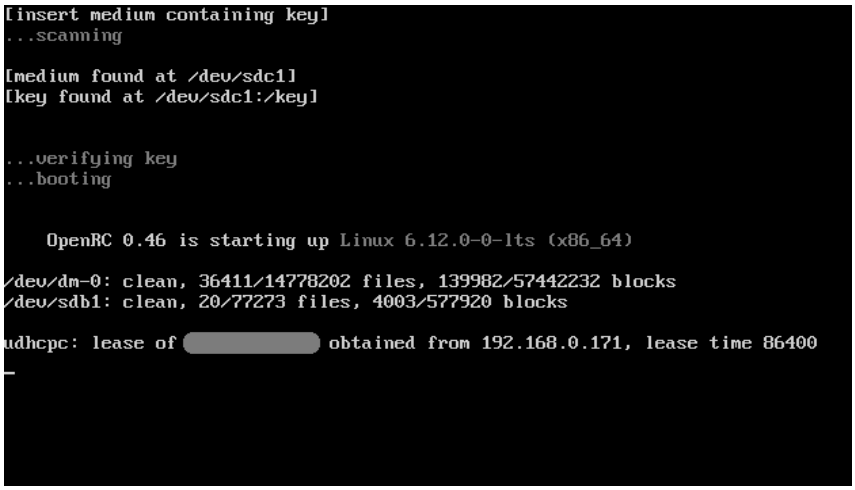
8th National Conference on Advanced Computing Technologies (NCACT'23)

Organized by

Department of CSE, Velammal Engineering College, Chennnai, India



1(b)



1(c)

Fig. 1. System boot-up process: (a) Waiting for storage device (b) Verifying keyfile (c) Booting as normal into system

|| e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | SJIF: 8.423|| Volume 12, Special Issue 2, April 2023 ||

8th National Conference on Advanced Computing Technologies (NCACT'23)

Organized by

Department of CSE, Velammal Engineering College, Chennnai, India

Fig. 2. Rebooting the system. The screenshot shows the instant when the checksum is calculated, saved to an external medium and set as the key for the next session.

Figure 2 Shows the system being rebooted. The modified files are listed and their checksums are calculated and stored to an external storage device. It is also used as a key to the primary storage disk of the system after the disk is unmounted.

IV. CONCLUSION

The above outlined implementation is simply a proof-of-concept, but provides a working example of the authentication method. The System state-based authentication method works effectively, fixing a few caveats that occur in traditional methods. It also preserves the concept of immutability from A-B root by providing a rollback to an unaffected system in case of an error or an attack. The System state-based authentication method is successful in providing an attack-proof and failure-proof authentication method.

REFERENCES

[1] S. Ibrokhimov, K. L. Hui, A. Abdulhakim Al-Absi, h. j. lee and M. Sain, "Multi-Factor Authentication in Cyber Physical System: A State of Art Survey," 2019 21st International Conference on Advanced Communication Technology (ICACT), PyeongChang, Korea (South), 2019, pp. 279-284, doi: 10.23919/ICACT.2019.8701960.

[2] S. Alwahaishi and J. Zdrálek, "Biometric Authentication Security: An Overview," 2020 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bengaluru, India, 2020, pp. 87-91, doi: 10.1109/CCEM50674.2020.00027.

[3] J. Stone, M. Greenwald, C. Partridge and J. Hughes, "Performance of checksums and CRCs over real data," in IEEE/ACM Transactions on Networking, vol. 6, no. 5, pp. 529-543, Oct. 1998, doi: 10.1109/90.731187.

[4] N. -F. Huang, C. -N. Kao and R. -T. Liu, "A novel software-based MD5 checksum lookup scheme for anti-virus systems," 2011 7th International Wireless Communications and Mobile Computing Conference, Istanbul, Turkey, 2011, pp. 207-212, doi: 10.1109/IWCMC.2011.5982419.

[5] ABRoot, Mirko Brombin, URL: https://documentation.vanillaos.org/docs/ABRoot/

[6] D. Anton and E. Simion, "Linux Unified Key Setup (LUKS) - The Good, the Bad, the Ugly," 2018 10th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Iasi, Romania, 2018, pp. 1-6, doi: 10.1109/ECAI.2018.8678978.

[7] A. Fujimoto, P. Peterson and P. Reiher, "Comparing the Power of Full Disk Encryption Alternatives," 2012 International Green Computing Conference (IGCC), San Jose, CA, USA, 2012, pp. 1-6, doi: 10.1109/IGCC.2012.6322245.

INNO SPACE
SJIF Scientific Journal Impact Factor

SJIF: 8.423

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INTERNATIONAL CENTRE

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details