A non-Interactive One-Time Password-based Method to Enhance the Vault Security

Juarez Oliveira, Altair Santin, Eduardo Viegas, and Pedro Horchulhack

Pontifícia Universidade Católica do Paraná | Pontifical Catholic University of Parana — PUCPR, Graduate Program in Computer Science — PPGIa, Curitiba, Brazil
{juarez.oliveira,santin,eduardo.viegas,pedro.horchulhack}@ppgia.pucpr.br

Abstract. Multi-factor authentication (MFA) is recommended to access sensitive data applications. A password Vault protects secrets by storing privileged user credentials and access codes. The combination of MFA and Trusted Execution Environment (TEE) by multiple communication channels reduces the attack surface of secrets and enables secure periodic code updates from the password Vault. In this paper, we propose all these layers of protection and add a one-time password (OTP) mechanism to enhance the security of the Vault without human intervention. The expiration time of the code in the Vault remains unchanged. Finally, we show that web applications and an interactive remote shell are effectively secured by penetration testing from an adversary's point of view.

Keywords: One-Time Password · Trusted Execution Environment · Password' Vault · Mitre Att&ck

1 Introduction

Traditionally, cybersecurity focused on securing computer networks by strongly relying on cryptographic algorithms implemented in hardware and software to safeguard data storage and transportation. In response, the industry prioritized building hardware that facilitates secure software execution. Trusted Execution Environments (TEEs) enables the execution of source code that manipulates sensitive data, such as credentials, which remains encrypted within enclaves [28]. The primary purpose of these enclaves is to thwart unauthorized access to memory regions housing sensitive information.

Unfortunately, credential and session stealing are increasingly common, even in environments safeguarded by security mechanisms, including those implemented with cryptographic techniques. To address such a challenge, a widely adopted strategy involves implementing Multi-Factor Authentication (MFA) mechanisms [19]. However, MFA is susceptible to social engineering attacks, a risk that can only be mitigated through adequate user training.

Centralized authentication, particularly in the Single Sign-On (SSO) model, is advocated as a best practice by the Center for Internet Security (CIS) Controls [4] and MITRE ATT&CK [10]. This approach is recommended due to the complexities associated with managing user credentials for each application

in an enterprise environment. In the centralized Identity and Access Management (IAM) model, network administrators exert greater control over password expiration times, MFA devices, and the ability to block suspicious devices attempting unauthorized access. Notwithstanding, they also oversee users' permissions within a system. In this scenario, the IAM infrastructure can either opt for an on-premise solution (usually not recommended) or choose the preferable external alternative for the Identity Provider (IdP). This is because controlling an IAM server is challenging for an attacker due to the lateral movement difficulty, particularly when dealing with a remote and well-secured IdP domain. Through lateral movement, adversaries can pave the way for unauthorized access to a device or system via application flaws or credential theft, expanding their influence across other systems or devices and gaining control over privileged credentials. Privileged Access Management (PAM) offers effective protection, particularly through a password vault component [14].

To mitigate the risk of credential theft, imposing an expiration time for credentials (passwords/codes) is a common practice [25]. Implementing a credential expiration time can booster security but might simultaneously reduce usability and productivity. This underscores the need for adopting new best practices to ensure the safeguarding of these credentials. For example, a common industry practice involves setting a secret expiration time, often at 8 hours, implying that within this timeframe, an attacker could exploit system vulnerabilities by concealing their actions or acting as a malicious insider. Another scenario involves a hacker targeting the PAM to obtain the secret for system access, with an 8-hour window to attempt unauthorized access. This complicates the management of credential validity because if administrators reduce this time, users will need to change their secrets (passwords/codes) more frequently as soon as the validity period expires. As a result, enhancing security could potentially negatively impact productivity due to the increased frequency of token renewal.

In light of this, this paper proposes a new secure mechanism to maintain usability and productivity while also improving vault security in a non-interactive approach. Our proposed scheme combines MFA with TEE to reduce the attack surface for application secrets. Notwithstanding, we introduce an integrated approach that features the implementation of a One-Time Password (OTP) mechanism aimed at augmenting the vault security seamlessly without requiring human intervention. This innovative method enables the system administrator to maintain the expiration time of codes in the vault while not affecting the system's usability and keeping its security.

In summary, our main paper contributions are:

- A new non-interactive mechanism for vault security without compromising usability. The model combines MFA with TEE to reduce the attack surface and introduces an OTP method to seamlessly augment vault security.
- A rigorous prototype penetration testing that validates the effectiveness of our model, ensuring robust security for web applications and an interactive remote shell from an adversary's viewpoint.

2 Background

Intel Software Guard Extension (SGX) implements TEE by leveraging hardware-centric security protocols [16]. By delineating secure enclaves within the processor architecture, SGX facilitates the establishment of isolated domains where confidential computations occur with heightened security. These enclaves function as secure environments, shielding sensitive data and code from unauthorized access, even by privileged software layers. This ensures protection through cryptographic measures, employing encryption to secure the enclave's memory contents. Furthermore, robust secure initialization procedures fortify the enclave's integrity. Aligned with the overarching TEE framework, SGX stands as a concrete and effective solution for creating secure computational environments within processors.

In practical implementation, leveraging SGX involves meticulously considering enclave design, secure coding practices, and enclave attestation. Enclave design necessitates a judicious delineation of the specific computations requiring heightened security, ensuring they are encapsulated within the enclave. Secure coding practices involve adhering to SGX-specific guidelines, employing appropriate cryptographic primitives, and meticulously managing memory within the enclave. Enclave attestation, a critical component, involves verifying enclaves' integrity, confirming that they have not been compromised.

CIS Controls is a well-recognized framework that features 18 control groups designed to support the development of robust mitigation and prevention controls. As an example, the integration of Vault as a PAM system is advocated, as it paves the way for security through password and passwordless mechanisms. In particular, the framework enforces MFA for Vault access, which can be implemented using PyOTP as a standard Python library [12], while also exploring OTP variants, such as Time-Based One-Time Password (TOTP) and the HMAC-based One-Time Password (HOTP) into PAM architectures.

Apart from introducing strong cybersecurity mechanisms, frameworks also focus on identifying and addressing security vulnerabilities. As an example, Common Vulnerabilities and Exposures (CVE) is used as a comprehensive vulnerability catalog that is linked to the NIST Vulnerability Database[11]. In such a case, each identified vulnerability is mapped and measured based on their Common Vulnerability Scoring System (CVSS) score, which tracks the vulnerability impact on the system. On a similar path, the Mitre ATT&CK framework is a globally accessible knowledge base detailing adversaries' techniques and tactics, utilizing the Tactics, Techniques, and Procedures (TTP) framework [10]. In such a context, the Penetration Testing Execution Standard (PTES), which is a systematic and repeatable security assessment framework that covers various stages in the cyber kill chain, plays a crucial role [1].

3 Related Works

Wu et al. introduced SGX-UAM, a unified authentication scheme that leverages Intel SGX and OTP to address Man-in-the-Middle (MitM) and replay attacks.

Despite acknowledging potential hardware costs associated with Intel SGX, the authors underscored the heightened resilience achieved in Unified Access Management. In a performance comparison, the proposed scheme demonstrated comparable authentication times to OpenID and OAuth2. However, it is noteworthy that a dedicated security evaluation was not conducted in the study [27].

The Confidential Computing Consortium assesses contemporary confidential computing technologies, with a particular focus on hardware features within TEE, such as code confidentiality. While the objective is to limit access for platform operators, it's important to note that trusted computing has limitations and may not provide a comprehensive defense against all potential attack vectors, as highlighted in the evaluation [15].

Henricks and Kettani [19] explore the merits of MFA, highlighting its strength as an additional layer beyond password authentication. Their discussion extends to potential future considerations, particularly the prospect of using human DNA characteristics for authentication. Emphasizing the importance of prudent security considerations, the authors underscore the need for careful evaluation in implementing such advanced authentication methods.

Fisher [17] underscores the significance of Vault as a central component in PAM, providing protection against a range of security threats, including credential abuse and unauthorized privilege escalation. Despite its pivotal role, Fisher highlights the importance of complementing Vault with additional secure programming techniques to attain optimal security outcomes, particularly in intricate computing environments.

4 Modeling the Vault Security Enhancement

4.1 A non-Interactive One-Time Password-based Method

We propose a mechanism that consistently employs a Vault for storing and managing secrets, encompassing tokens, codes, and passwords. This approach minimizes the exposure surface by mitigating reliance on credentials stored in files and environment variables.

The web server, exposed to the Internet, connects to other machines via API (see Fig. 1). User authentication, in steps 1 to 3 of Fig. 1, can be an internal or external IdP, based on integration needs and adherence to internal policies.

The OTP token generation (Fig. 1, steps 7 and 11) occurs on the TEE server for MFA, preventing unauthorized PAM access and securing the OTP generator's seed. This approach, integrated into our proposal, adds complexity for potential adversaries, especially in critical environments. An alternative technique involves storing the OTP generator seed in a Vault, eliminating the need for writing local secrets to a file. In both cases, steganography is used to protect code in transit through APIs, highlighting the value of a TEE implementation for swift and secure control implementation.

Post-authentication, whether for a web application user, secure shell user, or other non-interactive service, the validity time of the credential or token (Fig. 1,

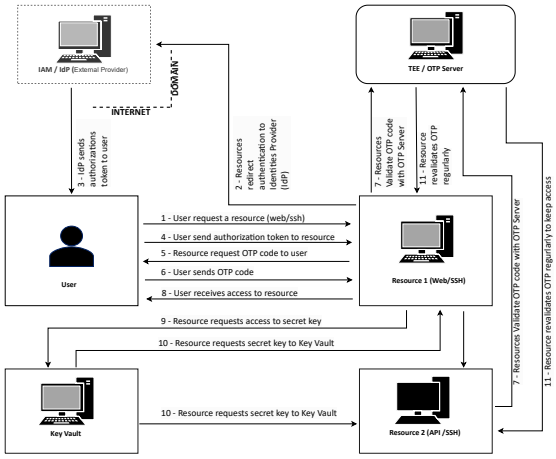

Fig. 1: Overview of our proposed enhanced credential Vault security scheme.

steps 9 and 10) is crucial. It determines the vulnerability window for potential adversaries, representing the timeframe for token exploitation. Accurate calculation of this period and the inclusion of additional authentication factors beyond the password are pivotal in maintaining service security. The dual validation of credentials, performed in addition to the initial authentication process (Fig. 1, steps 1 to 3, and outlined in Fig. 2a:, detailed events from *userToken* to Data and from *SendOTPCode* to *authOTPuser*), occurs periodically on the webserver for API access (Fig. 1, steps 5 to 7 and steps 8 to 11). This dual validation enhances the overall security level and ensures that credentials or even the active session remain safeguarded against potential theft.

Continuous OTP monitoring facilitates quick responses to security issues, like terminating sessions or SSH services. Post-SSH authentication (Fig. 2b:, events from *requestUserAuth* to *SSHConnection*) in the IdP, a script for OTP verification is triggered (event *sendOTPcode*), awaiting the user's second validation (event *authOTPuser*). Access to the Vault for a single-use access credential occurs during SSH authentication

This MFA mechanism is embedded in a TEE, with a second validation during web and SSH authentication. The OTP code is transmitted non-interactively and independently after successful IdP authentication. Even if an adversary steals a credential from the Vault, they cannot use it without completing steps 1 to 4 in Fig. 1. In case of OTP-based authentication failure, the connection or session is automatically blocked, adding an extra layer of security for critical resources.

4.2 Threat Model

The adversary's goals include reconnaissance of the environment (TA0043 in MITRE ATT&CK) [2], involving identifying services, open ports, and under-

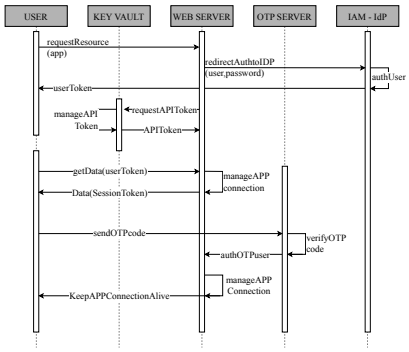

Fig. 2a: WEB Auth with centralized OTP server.

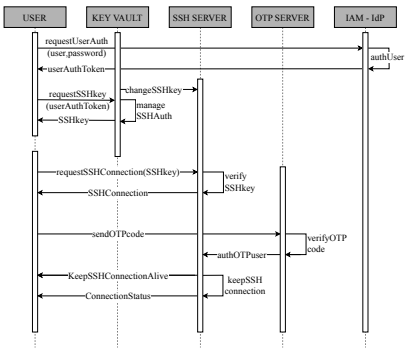

Fig. 2b: SSH auth with centralized OTP server.

standing authentication processes [22]. Once familiar with the target's characteristics and weaknesses, the attacker develops a strategy, potentially using automated tools for password cracking or social engineering tactics to acquire valid credentials from users.

The attacker's direct actions depend on the level of asset protection, including firewalls, web server restrictions, MFA, connection limits, and Web Application Firewall (WAF). To assess the adversary's capabilities, two scenarios are considered: (*i*) an Internet-based attacker with limited access to the web server and (*ii*) an internal network-based attacker with access to both SSH and web ports.

We acknowledge the potential interception of MFA authentication through keyloggers on compromised systems. External channel interception, like email or SMS, is possible without adequate service provider protection. Mitigation includes user training (TTP 1111 [21]) to remove authentication cards when not in use. Monitoring system APIs is crucial to detect malicious actions, albeit requiring additional effort [18].

In an alternative adversary action, authentication assets like password hashes, Kerberos tokens, or application OTP tokens can be stolen through improper access to RAM or configuration files. To mitigate, managing privileged accounts by enforcing the principle of least privilege (aligned with TTP 1550 [3]) is crucial. This extends to ordinary user accounts to prevent unnecessary access and privileges. Detection involves monitoring session creation and resource access logs.

In the context of Remote Services, specifically Mitre's SSH (ID: T1021.004) [13], adversaries exploit legitimate user information for unauthorized network access via the SSH protocol. APT39 is a known threat group using this technique. Mitigation includes turning off unnecessary SSH services, implementing MFA, and limiting user access. Detection involves monitoring session and user creation, and network connections. Utilizing a Vault aids in mitigation, and MFA with OTP facilitates monitoring, allowing service deactivation in case of misuse. Therefore, considering such a threat model, in this work, we want to answer the following

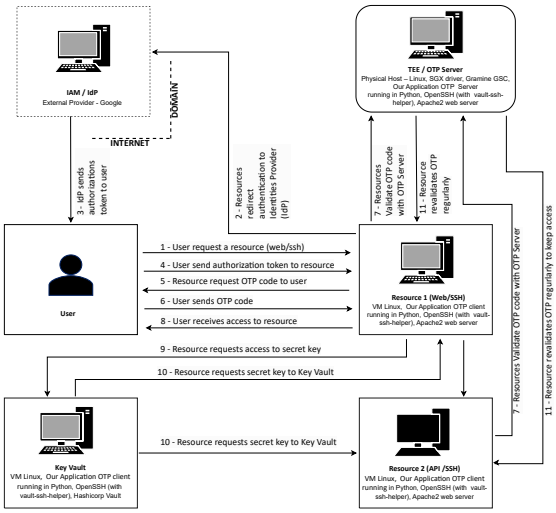

Fig. 3: Prototype Overview

research question: *How is it possible to provide a secure mechanism to increase the security of the Vault in a non-interactive way, aiming at non-interference in the usability of the approaches in use?*

5 Prototype

We implemented a prototype with a web application server on a virtual machine using Ubuntu Server 22.04 (Fig. 3). The setup includes an Apache2 web server, openSSH server, Python-based OTP client, and a Vault SSH client agent (vault-ssh-helper). The IdP is considered an external entity in this configuration.

The Vault, utilizing HashiCorp Vault solution [7], is deployed on a virtual machine with Ubuntu Server 22.04. The configuration includes a Vault SSH client (vault-ssh-helper) and a Python-based OTP client application. Integrating Vault aligns with best practices for Access Management, enhancing protection against unauthorized system access by implementing PAM, as per CIS controls.

The OTP Server application server, developed in Python with the PyOTP library [12], is on a physical machine running CentOS 8 [26]. The hardware includes a 2.70 GHz Intel(R) Core(TM) i7-7500U processor with INTEL Gramine GSC, operating within a Docker container [20]. The system utilizes Intel/SGX drivers for the TEE environment. This server also hosts a critical Python-based API and a Vault SSH client (vault-ssh-helper).

Concurrently with API authentication, the user undergoes IdP authentication. A distinct OTP authentication step follows to ensure access credential integrity and validate against compromise. OTP re-validation can occur at initial login and periodically during the session.

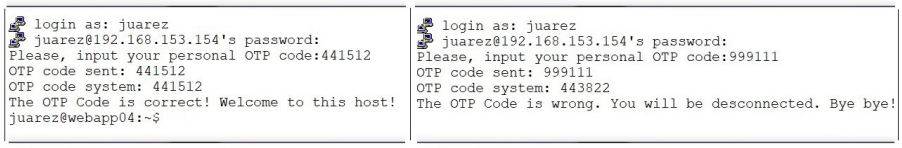

Fig. 4: Validation screen displayed to the user

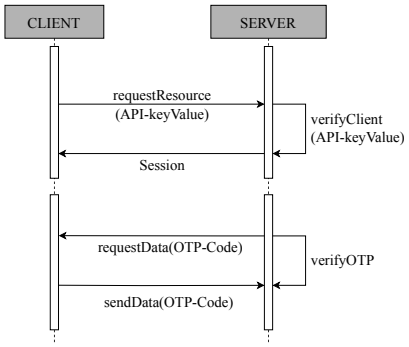

Fig. 5a: API Authentication metaprotocol

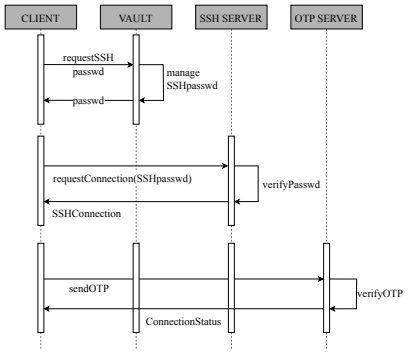

Fig. 5b: SSH Auth metaprotocol

SSH connection utilizes a one-time code/password from the Vault. After standard authentication, an internal post-script on the target server validates the OTP within the secure server (SGX-Gramine). The SSH connection finalizes only on successful secondary validation, with periodic re-validation possible. The validation screen (Fig. 4, left) displays the correct code for debugging.

The integration between the OTP application on the secure server and the Vault utilized the HVAC library [8], as shown in Fig. 5b:. This approach enables password input for the initial connection without storing it in a file or environment variable. An alternative is steganography, concealing connection information within a file for added complexity against potential attackers. If credentials are entered once, the application prompts re-entry upon each restart. A Docker container with Intel Gramine GSC [6] enhances portability and reduces dependence on the host operating system, chosen for its increased portability [9].

Installation of OTP clients and monitoring application usage was successful within a local network. For external web clients, continuous authentication without interaction is challenging, necessitating periodic entry of the OTP code. Although introducing inconvenience, this measure significantly enhances security, as depicted in Fig. 5b:.

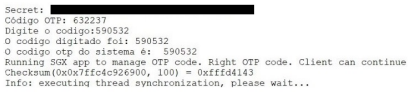

Fig. 6a: OTP App Server – Auth OK.

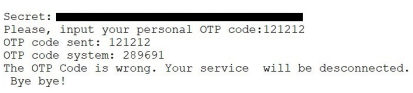

Fig. 6b: OTP App Server – Auth Not OK.

6 Evaluation

In the case of an Internet-based attacker limited to port 443, the implemented MFA by IdP Single Sign-On substantially reduces the attack surface. Even if the attacker captures login credentials or gains access, OTP token software triggers automatic disconnection after the configured time limit, limiting the window of opportunity.

For API authentication on a separate host without external Internet access, exclusive OTP token authentication without human interaction prevents the attacker from sustaining the connection, even with authentication data (key, secret). The OTP token, transmitted with the request, can use steganography for added complexity.

An alternative in this work is sending the TOTP code out-of-band, parallel and over another channel. Every authentic code request using an API triggers the sending of a HOTP code based on the OTP code sequence. If an attacker attempts to invoke an API without the correct HOTP sequence code, the operation will not be confirmed. Our proposal includes a manual validation step for the HOTP sequence code to monitor these actions.

We have implemented manual validation of the HOTP sequence code in our proposal to track these actions. For more detail, see Fig. Fig. 6a: – verification of an OTP code from an OTP application running on the TEE, which has been successfully validated, and Fig. 6b: – verification of an OTP code from an OTP application running on the TEE, which has been successfully validated.

7 Security Analysis

Following cybersecurity best practices, our model emphasizes layer-based security. Procedures are structured to prevent the failure of one layer from compromising overall security. The design includes two additional layers: (i) robust data protection within a TEE for system applications and (ii) an extra authentication factor with OTP implemented out-of-band. This multi-layered approach enhances the overall security framework.

External access rules via the firewall are implemented, allowing access to TCP ports exclusively for the web service externally and for SSH, Vault, and the OTP server internally. The application client's external access is restricted to the web server, minimizing the exposure surface for enhanced security.

To evaluate the security achieved, penetration tests were conducted using established methodologies like Metasploit [23] [24]. These tests aimed to identify potential attacker capabilities and assess the exposure surface.

An external NMAP scan revealed ports 443 (HTTPS) on the web server for internet connections and internal ports 4443 (OTP application), 8201 (key vault), and 22 (SSH). The accessibility of these ports indicates successful firewall-based security measures.

Additional tests simulated adversary actions using tools from PTES [1], focusing on *Customized Exploitation* and *External Footprints*. These tests provided insights into the system's resilience against exploitation and external footprinting attempts.

In assessing credential theft potential over the local network, TCPDump analysis indicated data encryption in transit, particularly over HTTPS and TLS connections to the OTP Socket, reinforcing security measures.

A memory dump test to access OTP in memory demonstrated encryption by SGX, making direct access challenging. Subsequent tests with the Volatility tool affirmed the unavailability of clear-text memory contents due to encryption.

The initial penetration test targeting SSH server configured without OTP revealed vulnerability to brute-force dictionary attacks. After access was gained, the connection was promptly revoked. Notably, there were no additional security measures on the SSH server, such as attempt limits, source IP blocks, or MFA authentication in PAM.

In a subsequent SSH server test to evaluate OTP effectiveness, a simulated insider attack with Metasploit was executed. Initial connection success was followed by a password change due to Vault's dynamic password generation. Valid SSH authentication led to OTP verification on TEE, terminating the connection. While Metasploit sessions could be created with correct credentials, shell access was not established.

A Hydra brute-force attack on the web server, assuming an Internet-based attacker on port 443, yielded no results without OTP configuration. Acknowledging potential credential acquisition through alternative means, we activated OTP as a second factor, prompting re-authentication to terminate unsuccessful connections. Balancing heightened security with usability and productivity considerations requires careful analysis due to the risk of unavailability, posing a security concern.

In assessing continuous authentication via APIs without human interaction, we tested the OTP client periodically dispatching codes. The OTP server in the TEE scrutinized the HOTP sequence's validity, successfully executing a pre-configured action for an artificially transmitted incorrect code.

Authentication check intervals vary based on the service; SSH might require a one-minute check, while databases or external API access may need longer intervals. Administrators must configure session time parameters based on specific use case requirements.

For SSH access on Linux servers, implementing tools like Fail2ban [5] or EDR protection helps thwart brute-force attempts. Additional security measures include configuring hosts.deny, hosts.allow, Apache2 web server location match rules, and applying firewall rules and host hardening practices, contributing to a layered security approach.

Vault integration has effectively reduced access to credentials traditionally stored as plain text, enhancing overall security by mitigating risks associated with long credential validity times. Secure storage in Vault aligns with best practices, reducing exposure surface and potential vulnerabilities.

Implementing multiple layers of protection challenges intruders by restricting the exposure surface and introducing temporal constraints through frequent code or password changes. This multi-layered approach reinforces overall system resilience against potential threats.

The research aimed to enhance system security without compromising productivity or user-friendliness. The proposed model successfully integrated an OTP mechanism concurrently with the conventional approach, maintaining PAM-based workflow.

While PAM credential validity remained around 8 hours, the OTP mechanism introduced more frequent out-of-band validations in minutes, significantly raising the bar for attackers. Configurable time validity adapts to specific application needs, contributing to strengthened security while maintaining operational efficiency.

8 Conclusion

Implementing applications within a trusted execution environment introduces complexities for system administrators, especially in routine vulnerability management. To address this challenge, our choice of utilizing libOS Gramine helps navigate this intricate landscape.

The use of Vault for credential management aligns with established practices in major cybersecurity frameworks, underlining its significance as a robust security measure. Although adjusting session times for applications and services poses a considerable challenge, it remains imperative for overall environment protection. In our work, employing an out-of-band OTP validation strategy proved effective even with low session times, demonstrating reconfigurability. This approach fortifies security and poses challenges for potential attackers aiming to compromise the proposed security based on OTP.

The viability of the proposed mechanism aligns with project objectives, which were observed through security analyses and penetration tests. The findings indicate an enhancement in security without altering the time validity in the PAM, thereby favoring the tradeof between security, usability, and user productivity.

References

1. The penetration testing execution standard (2014), http://www.pentest-standard.org/index.php/Main_Page
2. Reconnaissance – tactic ta0043 (2020), https://attack.mitre.org/tactics/TA0043/
3. Use alternate authentication material (2022), https://attack.mitre.org/techniques/T1550/
4. Cis (2023), https://www.cisecurity.org/controls

5. Fail2ban: ban hosts that cause multiple authentication errors (2023), https://github.com/fail2ban/fail2ban
6. Gramine - a library os for unmodified applications (2023), gramineproject.io/
7. Hashicorp developer (2023), https://developer.hashicorp.com/
8. hvac – hvac 1.2.1 documentation (2023), https://hvac.readthedocs.io/en/stable/
9. Intel software guard extensions – developer guide (2023), https://download.01.org/intel-sgx/latest/linux-latest/docs/Intel_SGX_Developer_Guide.pdf
10. Mitre att&ck (2023), https://attack.mitre.org/
11. National vulnerability database (2023), https://nvd.nist.gov/
12. Pyotp (2023), https://pyauth.github.io/pyotp/
13. Remote services: Ssh (2023), https://attack.mitre.org/techniques/T1021/004/
14. Cheng, H., Li, W., Wang, P., Chu, C.H., Liang, K.: Incrementally updateable honey password vaults. In: 30th USENIX Security 21. pp. 857–874 (2021)
15. Consortium, C.C.: A technical analysis of confidential computing. Tech. rep. (2023), https://confidentialcomputing.io/wp-content/uploads/sites/10/2023/03/CCC-A-Technical-Analysis-of-Confidential-Computing-v1.3_unlocked.pdf
16. Fei, S., Yan, Z., Ding, W., Xie, H.: Security vulnerabilities of sgx and countermeasures: A survey. ACM Computing Surveys **54**(6), 1–36 (Jul 2021)
17. Fisher, P.: Privileged access management (pam) demystified (2023), https://www.oneidentity.com/what-is-privileged-access-management/
18. Geremias, J., Viegas, E.K., Santin, A.O., Britto, A., Horchulhack, P.: Towards multi-view android malware detection through image-based deep learning. In: 2022 International Wireless Communications and Mobile Computing (IWCMC). IEEE (May 2022)
19. Henricks, A., Kettani, H.: On data protection using multi-factor authentication. In: Proceedings of the 2019 International Conference on Information System and System Management. ISSM 2019, ACM (Oct 2019)
20. Horchulhack, P., Viegas, E.K., Santin, A.O., Ramos, F.V., Tedeschi, P.: Detection of quality of service degradation on multi-tenant containerized services. Journal of Network and Computer Applications **224**, 103839 (Apr 2024)
21. Lambert, J.: Multi-factor authentication interception (2023), https://attack.mitre.org/techniques/T1111/
22. dos Santos, R.R., Viegas, E.K., Santin, A.O.: A reminiscent intrusion detection model based on deep autoencoders and transfer learning. In: 2021 IEEE Global Communications Conference (GLOBECOM). IEEE (Dec 2021)
23. dos Santos, R.R., Viegas, E.K., Santin, A.O., Tedeschi, P.: Federated learning for reliable model updates in network-based intrusion detection. Elsevier Computers & Security **133**, 103413 (Oct 2023)
24. Santos, R.R.d., Viegas, E.K., Santin, A.O., Cogo, V.V.: Reinforcement learning for intrusion detection: More model longness and fewer updates. IEEE Transactions on Network and Service Management **20**(2), 2040–2055 (Jun 2023)
25. Taherdoost, H.: Understanding cybersecurity frameworks and information security standards—a review and comprehensive overview. Electronics **11**(14) (07 2022)
26. Viegas, E., Santin, A., Bachtold, J., Segalin, D., Stihler, M., Marcon, A., Maziero, C.: Enhancing service maintainability by monitoring and auditing sla in cloud computing. Cluster Computing **24**(3), 1659–1674 (Nov 2020)
27. Wu, L., Cai, H.J., Li, H.: Sgx-uam: A secure unified access management scheme with one time passwords via intel sgx. IEEE Access **9**, 38029–38042 (2021)
28. Xia, K., Luo, Y., Xu, X., Wei, S.: Sgx-fpga: Trusted execution environment for cpu-fpga heterogeneous architecture. In: 2021 58th ACM/IEEE DAC (Dec 2021)