Privacy-Preserving Service for Secure Storage of Passwords Based on Singularization

Simona David
Orange Services
Bucharest, Romania
simona1.david@orange.com

Mihail Plesa
Orange Services
Bucharest, Romania
mihail.plesa@orange.com

Sebastian Irimia
Orange Services
Bucharest, Romania
sebastian.irimia@orange.com

Robert Poenaru
Orange Services
Bucharest, Romania
robert.poenaru@orange.com

***Abstract*—In the realm of cybersecurity, enterprises managing their own authentication services are increasingly vulnerable to offline attacks that target password databases. Traditional password storage methods, including hashing with salt, often fall short against modern threats due to the inherently low entropy of user passwords. This paper presents a novel solution that enhances password security through a privacy-preserving service known as the Singularization Service. The proposed method significantly increases password entropy to cryptographic standards, ensuring a minimum of 128 bits of entropy, while maintaining the confidentiality of user credentials. By employing a homomorphic encryption scheme and a unique randomization process, Singularization Service mitigates the risks associated with offline attacks, enabling secure password management without exposing plaintext passwords. This innovative approach not only strengthens the security posture of enterprises but also addresses critical privacy concerns associated with third-party authentication services.**

***Index Terms*—authentification, passwords, homomorphic encryption**

I. Introduction

Password-based authentication remains a cornerstone of enterprise security, yet the widespread use of low-entropy passwords continues to expose organizations to significant vulnerabilities. Several mitigation strategies have emerged to address this challenge, including the adoption of password managers, the implementation of multi-factor authentication (MFA), and the outsourcing of authentication to third-party identity providers. Password managers offer a means to strengthen password entropy and minimize reuse, while MFA adds additional verification layers to reduce reliance on passwords alone. Third-party authentication services can further enhance security and simplify credential management but introduce critical privacy risks by requiring the sharing of sensitive internal data with external entities. This paper surveys the current landscape of enterprise authentication solutions, analyzes the security and privacy trade-offs inherent to each approach, and underscores the need for a careful, context-specific evaluation when designing robust authentication architectures. We highlight emerging trends aimed at reconciling security, usability, and privacy, including decentralized identity systems and privacy-preserving authentication mechanisms.

II. Related Work

The challenge of securing enterprise authentication systems has been extensively studied, particularly concerning the vulnerabilities posed by low-entropy passwords. Several mitigation strategies have been proposed in the literature, encompassing password managers, multi-factor authentication (MFA), and third-party authentication services, each offering distinct trade-offs between security, usability, and privacy.

Password managers represent a widely endorsed solution to the problem of password entropy. By generating and securely storing complex, high-entropy passwords, these systems reduce the risks associated with user-chosen weak credentials [1, 2]. Numerous studies have evaluated the usability and security benefits of password managers, highlighting their role in minimizing credential reuse and preventing password guessing attacks [3, 4]. However, reliance on password managers introduces concerns regarding single points of failure and potential vulnerabilities if the manager itself is compromised [5].

Another critical advancement is the adoption of multi-factor authentication (MFA), which augments password-based systems with additional verification factors such as biometrics, hardware tokens, or one-time passwords (OTPs) [6, 7]. MFA schemes have been demonstrated to significantly mitigate the risks associated with credential theft and phishing attacks [8]. Despite their security benefits, MFA implementations often face challenges related to user adoption, increased authentication friction, and additional infrastructure costs [9].

Enterprises are increasingly turning to third-party authentication services, including OAuth providers and federated identity management solutions, to enhance authentication security and streamline access control [10, 11]. OAuth 2.0, for instance, has become a de facto standard for delegated authorization on the web, facilitating seamless access without necessitating repeated credential entry [10]. Identity Federation initiatives such as SAML and OpenID Connect further aim to unify authentication across diverse platforms [12, 13].

Nevertheless, the integration of external authentication services introduces significant privacy implications. Relying on third-party providers requires enterprises to share sensitive user information beyond organizational boundaries, thereby increasing the risk of data exposure and surveillance [14],

[15]. Furthermore, the concentration of identity management responsibilities in a few dominant providers has raised concerns regarding centralization and vendor lock-in [16].

Recent research efforts have sought to balance the trade-offs between security, usability, and privacy by proposing novel authentication paradigms. These include decentralized identity frameworks based on blockchain technology [17, 18] and privacy-preserving authentication protocols that minimize data disclosure during the authentication process [19, 20]. Such approaches aim to grant users greater control over their credentials while maintaining robust security guarantees.

Although considerable progress has been made toward enhancing enterprise authentication mechanisms, achieving an optimal balance between security, usability, and privacy remains an ongoing research challenge. A careful evaluation of the available solutions, each customized to meet enterprise needs and risk profiles, is imperative for an effective authentication system design.

III. Preliminaries

A fundamental component of implementation is the ElGamal encryption scheme [21], which consists of three algorithms:

1) $\mathsf{KeyGen}\left(1^{\lambda}\right)$: The key generation algorithm takes the security parameter as input and proceeds as follows:
   - Generate a group G of order q with a generator g.
   - Select a random integer $x \in 1, 2, \ldots, q-1$.
   - Compute $h = g^x$.
   - Output the public key $pk = (G, q, g, h)$ and the private key x.
2) $\mathsf{Enc}\,(pk, m)$: The encryption algorithm takes as input the public key pk and a plaintext message $m \in G$, and performs the following steps:
   - Select a random integer $y \in \{1, 2, \ldots, q-1\}$.
   - Compute $s = h^y$, $c_1 = g^y$, and $c_2 = m \cdot s$.
   - Output the ciphertext $c = (c_1, c_2)$.
3) $\mathsf{Dec}\,(x, c)$: The decryption algorithm takes as input the private key x and the ciphertext c, and performs the following:
   - Compute $s = c_1^x$.
   - Recover the plaintext $m = c_2 \cdot s^{-1}$.

A remarkable characteristic of this cryptosystem is its ability to perform operations directly on ciphertexts. Let $g^{m_a} \in G$ and $g^{m_b} \in G$ represent two plaintext messages, with c_a and c_b as their corresponding ciphertexts generated using the public key pk. For large values of x, the function g^x behaves as a pre-image resistant hash function, relying on the hardness of the discrete logarithm problem [22]. However, for small values of x, the function can be brute-forced, effectively allowing g^x to be treated as a plaintext.

Consider the ciphertext $c^* = (c_1^*, c_2^*)$, defined as:

$$c_1^* = c_{a_1} \cdot c_{b_1},$$
$$c_2^* = c_{a_2} \cdot c_{b_2}.$$

Using the encryption algorithm, we compute:

$$c_1^* = g^{y_a} \cdot g^{y_b} = g^{y_a + y_b},$$
$$c_2^* = g^{m_a} \cdot g^{m_b} \cdot h^{y_a} \cdot h^{y_b} = g^{m_a + m_b} \cdot h^{y_a + y_b}.$$

Following the decryption algorithm, the ciphertext c^* decrypts to $g^{m_a+m_b}$. Thus, given the encryption of g^{m_a} and g^{m_b}, we obtain a ciphertext that encrypts $g^{m_a+m_b}$. This demonstrates that the scheme is homomorphic with respect to addition.

IV. Problem Description

Consider a classic password-based authentication scheme for an application deployed as an online service, which can be accessed by users. Such a protocol is depicted in Figure 1 and it mainly consists of two stages: *registration* and *loging*.

1) **Registration Phase** The registration phase is composed of the following steps:
   - The user sends the user ID (U_{ID}) and the chosen password (P) to the identity server (ID server).
   - The ID server generates a random salt (unique per user, denoted by $\text{SALT}_{\text{user}}$ in Figure 1). A hash function computed over the concatenation of this salt and the password is then stored in a database, together with the user ID.
2) **Login Phase** The login phase is composed of the following steps:
   - The user sends the user ID and the password to the identity server.
   - The ID server retrieves the corresponding salt from the database and computes the hash over the concatenation of the salt and the received password.
   - The ID server checks if the computed hash corresponds to the one stored in the database for the user ID. If the hashes are identical, then authentication will be validated.

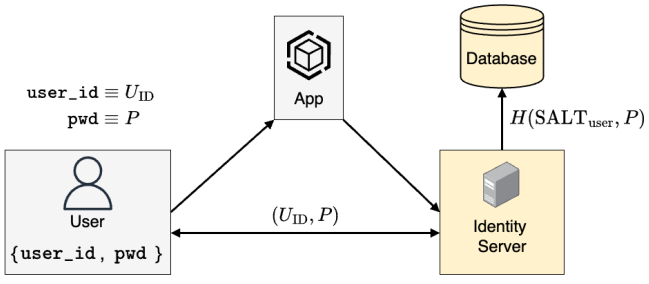

Fig. 1. A workflow diagram depicting a classical authentication flow, where a user attempts to access an application. Upon access, the application redirects the user to the identity server, which queries its database to validate the user's credentials. After a successful login, the identity server redirects the user back to the application, where the user is now authenticated. The database of the identity server contains cryptographic hashes generated by a hashing function H, which requires the password and a salt.

The primary limitation of this approach is that, in the event of a database breach, an attacker can dedicate unlimited time to brute-forcing the password hashes. Given the low entropy of passwords and the public nature of the salts, such an attack

becomes computationally feasible. To mitigate these risks, an enhanced protocol has been developed, building upon the previously described approach. The following section presents this protocol in detail.

V. Singularization based protocol

Our solution employs the concept of *singularization*, a dynamic defense strategy that ensures each system instance is unique [23]. Originally introduced to enhance the security of applets in legacy infrastructures, this approach has recently been adapted to strengthen the resilience of symmetric ciphers against cryptanalysis attacks [24]. A key benefit of this strategy is its ability to improve the security of existing systems without requiring extensive upgrades.

The proposed idea integrates seamlessly with existing authentication infrastructures based on identity (ID) servers. At its core is the Singularization Service, which increases the entropy of user passwords. This enhancement makes brute-force attacks computationally impractical, even if hashed passwords are compromised from the identity server. The overall architecture is illustrated in Figure 2, in which several notations are introduced:

1) P represents the user's password. Without loss of generality, we assume $P \in G$, where G is the group component of an ElGamal public key.
2) $\mathcal{H}$ denotes a secure cryptographic hash function [25].
3) The proxy refers to the entity acting on behalf of the user, conforming with the OAuth protocol.
4) U_{ID} is a unique identifier assigned to the user by the identity server, ensuring user distinction within the system.

A. Singularization process

The workflow proceeds as follows:

1) The user sends their unique identifier $\mathcal{H}(U_{\text{ID}})$ and password P to a stateless proxy (e.g., can be a containerized service). The proxy acts as an intermediary to interact with the Singularization Service itself, ensuring minimal modifications on the user side.
2) The stateless proxy generates an ElGamal key pair, encrypts the password using the public key, and sends the resulting ciphertext along with the U_{ID} to the Singularization Service:

$$\begin{aligned}(x, pk) &\leftarrow \mathsf{KeyGen}\left(1^{\lambda}\right)\\ c_p &\leftarrow \mathsf{Enc}\left(pk, g^{p}\right)\\ \text{Proxy} \rightarrow \text{SingServ} &: (U_{\text{ID}}, c_p)\end{aligned}$$

3) The Singularization Service generates a large random value $r \in G$ (e.g., 128 bits), computes its ElGamal encryption under the proxy's public key, and uses the homomorphic properties of the encryption scheme to compute a ciphertext corresponding to the sum of the password and r. The random value r is securely stored by the Singularization Service, associated with the user ID. The resulting ciphertext is sent back to the proxy:

$$\begin{aligned}r &\xleftarrow{\$} G\\ c_r &\leftarrow \mathsf{Enc}\left(pk, g^{r}\right)\\ c_s &\leftarrow \left(c_{p_1} \cdot c_{r_1}, c_{p_2} \cdot c_{r_2}\right)\\ \text{SingServ} \rightarrow \text{Proxy} &: c_s\end{aligned}$$

4) The proxy decrypts the received ciphertext to recover the hash of the sum of the password and the random value (g^{p+r}). The user's original password is replaced with this hash value, and the existing authentication process with the ID server continues without further changes:

$$\begin{aligned}g^{p+r} &\leftarrow \mathsf{Dec}\left(x, c_s\right)\\ p &\leftarrow g^{p+r}\end{aligned}$$

B. Comparison with existing work

From the perspective of the classical protocol described in Figure 1, the proxy replaces the user's password with a high-entropy password. This significantly mitigates offline attacks, as brute-forcing a password with high entropy (e.g., 128 bits chosen uniformly at random) is computationally infeasible. For an attacker to perform a brute-force attack, they would need access to both the ID server's database and the Singularization Service's database. This scenario is highly unlikely in practice, as these databases are stored in entirely separate locations. Furthermore, the Singularization Service cannot impersonate the user because it does not have access to the plaintext password, thanks to the security guarantees of the ElGamal cryptosystem. Additionally, the proxy is stateless, meaning it does not store the encryption key used for the password.

Our approach is distinguished from similar solutions [6, 7, 10] by two key characteristics:

1) Ease of Integration: Our solution requires minimal modifications to the existing authentication flow. For example, in OAuth, authentication is performed through a third party, which entirely replaces the current ID server, necessitating significant changes to the flow. In the case of multi-factor authentication (MFA), users are required to synchronize a second factor, such as a one-time password (OTP), and infrastructure owners must implement additional verification steps. In contrast, our approach avoids these complexities, making integration seamless.
2) Privacy-Preserving: The Singularization Service does not gain any meaningful information about the user's password or identity during the authentication process. In OAuth, the third-party provider is aware of the user's identity at all times. Similarly, MFA requires an enrollment step where the user must register with the service provider, potentially exposing sensitive information. In our approach, the Singularization Service is stateless and does not require prior user registration, ensuring enhanced privacy.

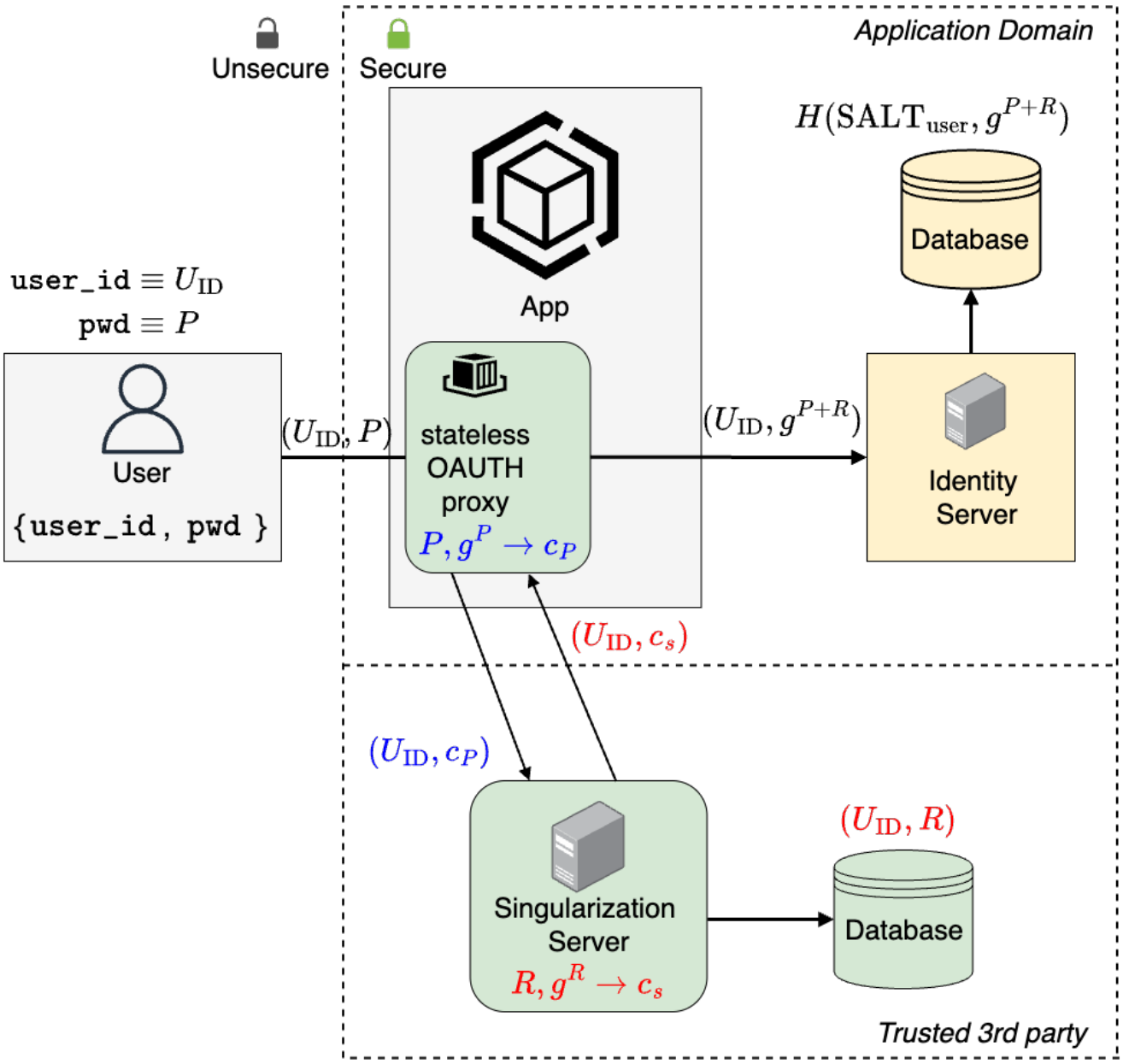

Fig. 2. (Color Online) A diagram showing the flow of an application that is connected to a *Singularized Identity Server*, which is an identity server with a proxy that handles authentication and authorization reuqests through a dedicated (trusted) singularization service. The parameters in blue color correspond to the proxy and with red color for the identity server. Notations are consistent with the ElGamal scheme defined in Section V-A.

VI. Conclusion

The Singularization Service addresses critical vulnerabilities associated with low-entropy passwords in enterprise authentication systems. By implementing this innovative solution, organizations can significantly enhance password security, thereby reducing the risk of unauthorized access. Furthermore, the service prioritizes user privacy, ensuring that security measures do not compromise individual data protection. Unlike traditional methods such as OAuth and multi-factor authentication, which necessitate significant alterations to existing authentication flows, our solution allows for a seamless integration process. This minimizes disruption for both users and infrastructure owners, facilitating a smoother transition to enhanced security measures. Overall, the Singularization Service represents a meaningful advancement in cybersecurity practices, paving the way for more secure and user-friendly authentication methods in the enterprise landscape.

References

[1] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano, "The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes," in *IEEE Symposium on Security and Privacy*, 2012.

[2] P. Gasti and K. B. Rasmussen, "Security analysis of password managers," in *Proceedings of the IEEE International Conference on Security and Privacy in Communication Networks*, 2012.

[3] S. Chiasson, P. C. van Oorschot, and R. Biddle, "Usability studies of password authentication schemes," in *Proceedings of the USENIX Security Symposium*, 2009.

[4] R. W. Reeder and S. Schechter, "Password managers: Advantages, challenges, and improvements," in *IEEE Security & Privacy Workshops*, 2011.

[5] A. Karole, N. Saxena, and N. Christin, "Assessing the security of smartphone password managers," in *Proceedings of the International Conference on Information Systems Security*, 2010.

[6] F. A. Aloul, "Two Factor Authentication Using Mobile Phones," in *Proceedings of the IEEE International Conference on Computer Systems and Applications*, 2009.

[7] S. Das, A. Dingman, and L. J. Camp, "Two-Factor Authentication in the Wild: A Usability Study," in *Proceedings of the Symposium on Usable Privacy and Security (SOUPS)*, 2018.

[8] S. Tang and K. W. Ross, "Phishing in real life: An evaluation of real-world phishing attacks," *IEEE Transactions on Information Forensics and Security*, 2014.

[9] R. Biddle, S. Chiasson, and P. C. van Oorschot, "Graphical passwords: Learning from the first twelve years," *ACM Computing Surveys*, 2012.

[10] D. Hardt, "The OAuth 2.0 Authorization Framework," *IETF*, 2012.

[11] E. Maler and D. Reed, "The Venn of Identity: Options and Issues in Federated Identity Management," *IEEE Security & Privacy*, 2008.

[12] S. Cantor, "SAML v2.0 Technical Overview," *OASIS SSTC Working Draft*, 2005.

[13] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore, "OpenID Connect Core 1.0," *The OpenID Foundation*, 2014.

[14] J. Sun, X. Zhang, and Y. Fang, "Privacy-aware access control system for enterprise identity management," *IEEE Transactions on Information Forensics and Security*, 2014.

[15] R. Chow, P. Golle, and M. Jakobsson, "Privacy and identity management for emerging Internet applications: challenges and trends," *Springer Briefs in Computer Science*, 2017.

[16] F. Brunton and H. Nissenbaum, "Obfuscating data to protect privacy," *Springer*, 2014.

[17] G. Zyskind, O. Nathan, and A. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *Proceedings of the IEEE Security and Privacy Workshops*, 2015.

[18] M. C. Towner and K. Salah, "Self-sovereign identity systems: Standards and future outlook," *IEEE Access*, 2019.

[19] J. Camenisch and A. Lysyanskaya, "Privacy-enhancing authentication and credential systems," in *Proceedings of the International Workshop on Privacy Enhancing Technologies*, 2002.

[20] J. Zhang et al., "Privacy-preserving authentication protocols: Survey and challenges," *IEEE Communications Surveys & Tutorials*, 2019.

[21] ElGamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions On Information Theory*. **31**, 469-472 (1985)

[22] W. Diffie and M. E. Hellman, New Directions in Cryptography, *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[23] Gaber, C., Macariot-Rat, G., David, S., Wary, J. & Cuaboz, A. Position Paper: Strengthening Applets on Legacy SIM Cards with Singularization, a New Moving Target Defense Strategy. *International Conference On Mobile, Secure, And Programmable Networking*. pp. 71-74 (2023)

[24] Macario-Rat, G. & Plesa, M. Singularization: A New Approach to Designing Block Ciphers for Resource-Constrained Devices. *International Conference On Attacks And Defenses For Internet-of-Things*. pp. 155-167 (2024)

[25] Dworkin, M. & Others SHA-3 standard: Permutation-based hash and extendable-output functions. (National Institute of Standards,2015)