3rd Global Conference on Engineering and Technology (GCOET), 4-5 December 2017, Bangkok, Thailand.

SUPA: Strewn User-Preserved Authentication

Monjur Ahmed

Centre for Business, Information Technology and Enterprise (CBITE)

Waikato Institute of Technology (Wintec), New Zealand

Email: Monjur.Ahmed@wintec.ac.nz

ABSTRACT

Central storage and management of user credentials or passwords leaves a single tempting repository for the attackers. If the credentials are not stored by a system at all, there will be no stored 'vault' to allure the attackers. At the same time, there will be no single resource that holds the credentials of all users of a system. This paper presents the high level conceptual architecture of SUPA, an authentication system that would allow a system to authenticate users without having its own repository of users' credentials. SUPA enables a system to authenticate its users without having their credentials stored in it. The proposed authentication system uses the features of asymmetric encryption as part of its authentication process.

Type of Paper: Conceptual – Original Research.

***Keywords*:** Authentication, Cloud Computing, Credential, Distribution, IoT, Password, Security.

1. Introduction

Security is a historic concern in computing. Recent developments in Cloud Computing and IoT, and the evolution of Internet and Internet-based technologies let information reside in remote unknown locations (Liu et al., 2015). Such technologies help the dispersion of user-information to third party computers that are out of organizational boundary. In the age of IoT and Cloud Computing, the attackers have more doors to penetrate into systems. Traditionally systems store authentication related data or information in their servers or computers (Kumar, Anjala, & Sharma, 2014). This results in single and bulk repository of sensitive user-credential related data that might be of interest to the attackers. With distributed computing and distributed approaches, the volume of data is split into sub-volumes. This only creates smaller volumes of the repository; it does not eliminate the existence of information 'vault' from the scenario. Compromising a computer with any kind of information repository would mean compromising all the accounts on which the credentials are kept in that repository. If a mechanism can be developed where users' credentials are not required to be stored by a system to authenticate the users, it would ensure added layer of challenge for the attackers to compromise those accounts. In this paper, we propose SUPA – a mechanism that authenticates users without having their credentials stored in any system servers.

The rest of the paper is structured as follows: 'Related Study' section discusses the literature review related to authentication. SUPA architecture is illustrated in the following section. The section 'Validation' shows the logical validation of the proposed mechanism, followed by the discussion on planned further and future developments for SUPA.

2. Related Study

Existing authentication systems store user's credentials at the system's or server's end. Different approaches to user authentication have been proposed. The existing authentication systems discuss how to securely store users credential and how to ensure that the authentication credentials are not abused (i.e. stolen) and used by any unwanted parties. Khalid et al. (2013) argue that there exist authentication protocols that promote anonymity, but anonymity is contradictory with the concept of authentication itself, and thus authentication should mean to identify a credible party.

Vaithyasubramanian, Christy and Lalitha (2015) discuss the use of array password for two-factor secured login, where they state that passwords are central in processing. In their proposed system, users may choose the length of the array of the password, and the validation is done based on the information stored at the service providers' end. The password authentication scheme based on single block hash function - as proposed by Wang, Wang and Li (2013) - authenticates users by comparing the credentials stored at the server end. A few examples of authentication systems that uses servers based users' stored credentials are found in Hwang and Li (2000), Liu, Zhou and Gao (2008), Li et al. (2011), Khanjan et al. (2015). Malempati and Mogalla (2011) propose user authentication using native language passwords – the passwords are verified by the stored password at the system's end. Hybrid authentication techniques stated by Sreelatha and Shashi (2011), and hybrid password scheme proposed by Zhang et al. (2010) takes similar approach for verification of users' credentials or passwords. The approach of users' identity and password for authentication present in a number of works that are discussed by Conklin, Dietrich and Walz (2004). Banne and Shedge (2012) present a review on a number graphical password based authentication scheme that use graphical stored credentials for authentication. Discussions and proposals on authentication systems that use credentials stored at servers' side are also found in Singh, Gour and Thakur (2014), Sahu and Singh (2014), Varghese et al. (2014), Vaithyasubramanian1, Christy and Saravanan (2015), Sayed et al. (2016) and Kumar and Bilandi (2014).

3. SUPA Architecture

The novelty of SUPA stands within its approach of not storing any secret user credentials at all, and thus eliminating the probability of user credentials being compromised. By the term 'credential', we refer to the users' authentication-related information that are kept secret and are supposed to be known by the respective user only – where the classic example of a 'credential' is a password, pin number or passphrase. SUPA-based authentication systems would not store any secret credentials in any system, and thus there will be no 'vault' of user credentials to allure the attackers. Discussion on SUPA architecture follows.

SUPA uses asymmetric encryption to authenticate users. The illustration assumes the users on a system registers themselves; but the method of user creation (i.e. whether user initiated registration and created by system administrator) is not significant in discussing SUPA architecture.

Once a user is created and a username for that user is created, allocated and reserved; the username is considered as the respective user's public key. Since SUPA uses public key encryption, there needs to be the counterpart private key. Let us assume that a user's public key is client public key (CPK), and the private key is client private key (CPvtK). Figure-1 shows the user creation process. In Figure-1, the application server refers to the system that holds the system or application for which users exist.

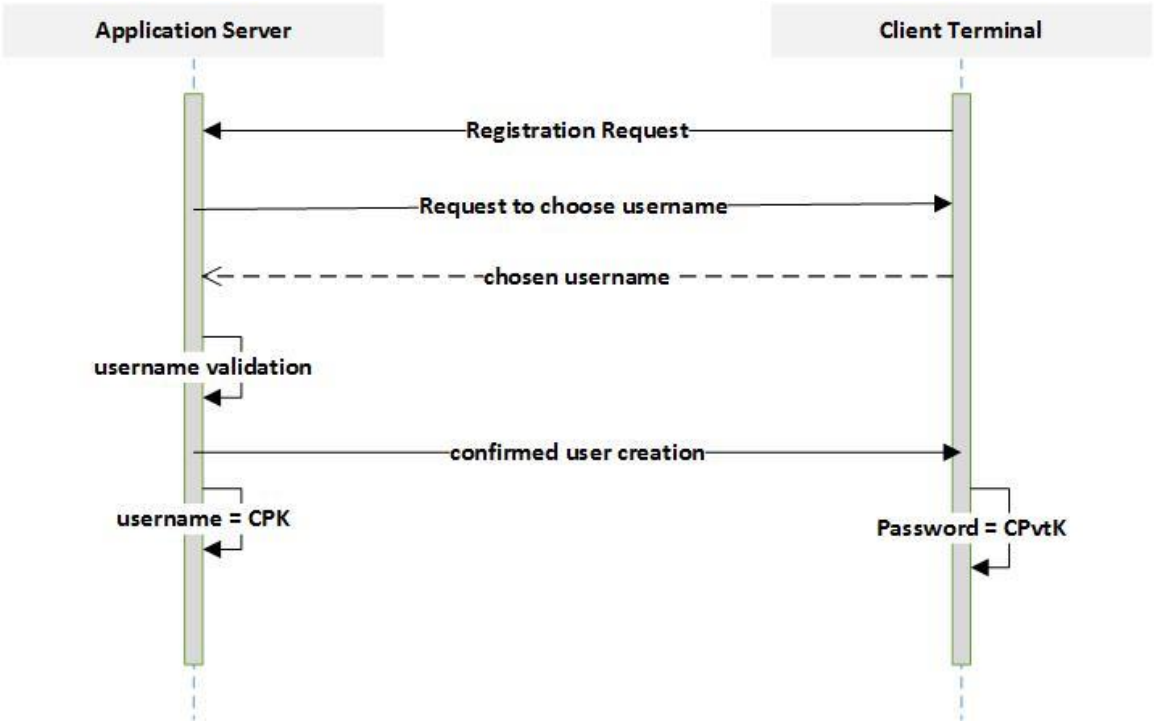

Figure 1. Defining CPK and CPvtK upon User Creation

Figure-1 implies the concept of user creation, where a unique identifier (e.g. username) for a user is determined. This is not different from any existing systems where a username or unique identifier is required for each user. However, upon user creation, SUPA considers the username as the primary key and asks user to choose password that is used as the private key for the user. Thus, the password is not shared or transmitted to the application server, as transferring only public key suffice in asymmetric encryption.

The application or system for which the user exists also has its own public-private key pair. Let us assume these public and private keys for the system or server are respectively server public key (SPK) and server private key (SPvtK). Figure-2 shows the steps involved in SUPA for user authentication.

As illustrated in Figure-2, when a user needs to be authenticated, the username is provided, which is also the public key (CPK) of the user. The server then takes any environment variable

from within its context and uses it as the environment parameter (ParamEnv). Let us assume that the environment parameter for the server is ParamEnv(s). An environment parameter might be anything existing in the server, for example, a random process identifier (PID) of the operating system of the computer where the application server resides, the checksum or hash value of any existing file or string, and so on. Any randomizing algorithm may be used to ensure that different ParamEnv is generated for every new user to be authenticated.

The server users the chosen ParamEnv(s) and encrypts it using its own private key (SPvtK) and the client's public key (CPK). The encrypted ParamEnv(s) then becomes the challenge string that is offered to the client.

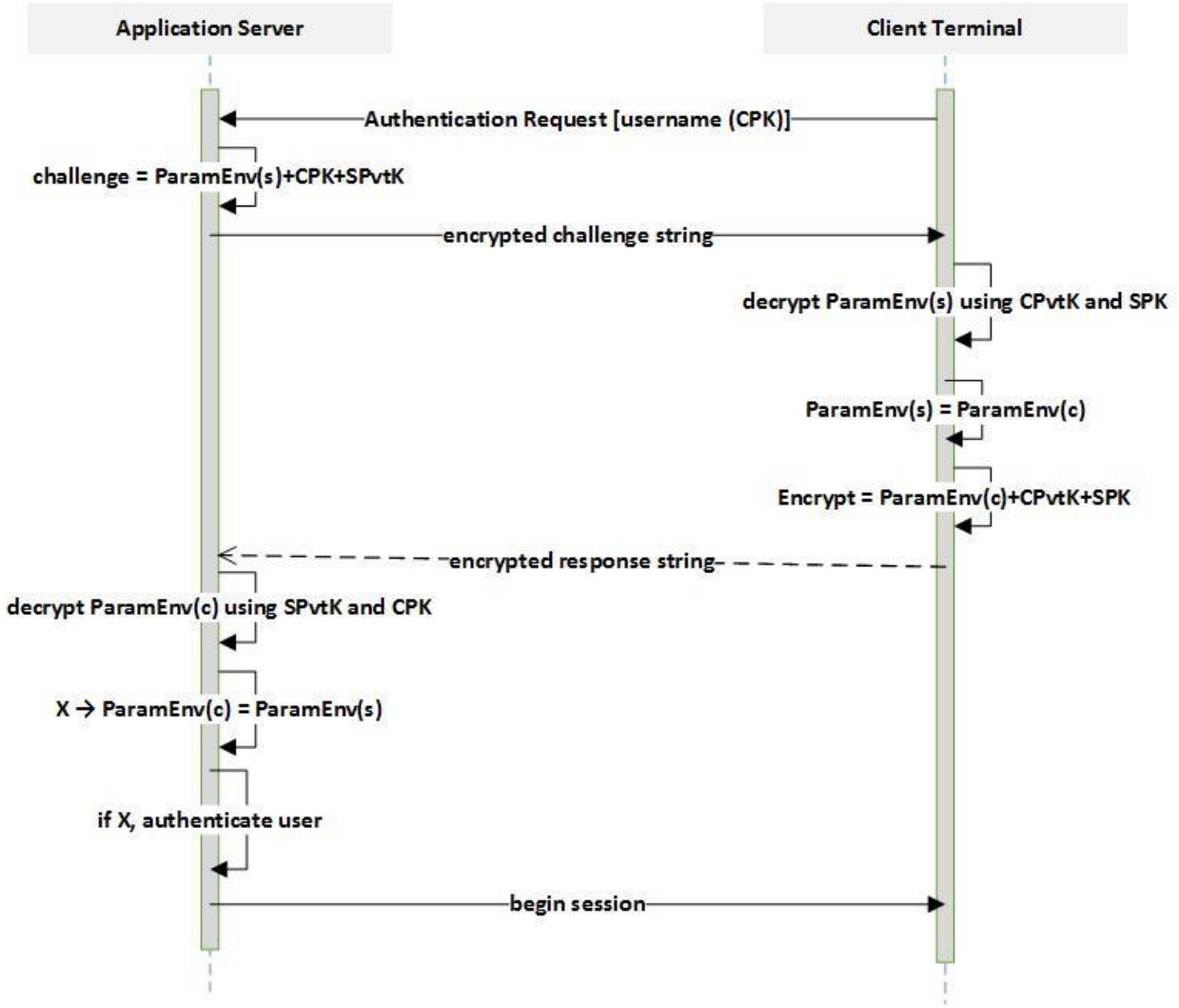

Figure 2. Authentication Process in SUPA

Upon receiving the encrypted challenge string, the client decrypts it using the server's public key (SPK) and its own private key (CPvtK), to get ParamEnv(s). The decrypted environment parameter then becomes the client's version of the server's environment parameter. Let us

assume that the client's version of the server's environment parameter is ParamEnv(c). Thus, for successful authentication, it must always be true that ParamEnv(c) is an exact replica of ParamEnv(s). However, the client encrypts ParamEnv(c) using its own private key (CPvtK) and server's public key (SPK). The response string is thus an encrypted version of ParamEnv(c). Upon receiving the response string from client, the server decrypts it using its own private key (SPvtK) and client's public key (CPK) to get ParamEnv(c); which the server then compares with ParamEnv(s). If ParamEnv(c) is an exact replica of ParamEnv(s), the server indicates the user as authenticated user. From this point onward, the session is established for the intended information sharing between the client and the server.

4. Validation

As explained above,

CPK = Client's public key

CPvtK = Client's private key

SPK = Server's public key

SPvtK= Server's private key

ParamEnv(s) = Server's environment parameter

ParamEnv(c) = Client's version of Server's environment parameter

Let us also assume,

S = Challenge string

R = Response String

Z = Successful authentication

X = successful encryption

Y = successful decryption

Thus,

$S \rightarrow \forall X [CPK \wedge SPvtK \wedge ParamEnv(s)]$

$R \rightarrow \forall X [SPK \wedge CPvtK \wedge ParamEnv(c)]$

$Z \rightarrow \forall Y [\exists! [S \wedge R]: ParamEnv(c) = ParamEnv(s)]$

Failed authentication is a situation where 'Z' does not hold true. Thus, a decryption with differing value of ParamEnv(c) and ParamEnv(s) would result in failed authentication attempt.

5. Future Development

SUPA is a research in progress. The architecture presented in this paper is the high level conceptual view of SUPA. A number of future developments are envisioned for the proposed authentication system. The planned future development for SUPA is to test it in simulated environment. The testing would include performance measurement and tolerance of SUPA against different types of attacks, for which detailed specifications for each step of the authentication processes are to be constructed. Developing an initial working prototype of SUPA would be achieved afterwards. Besides, how SUPA can complement other technologies and likewise is another aspect to investigate in future. The feasibility of SUPA in ad-hoc networking scenarios that are becoming more and more common in IoT based computing, is also to be examined.

6. Conclusions

Authentication systems that do not store users' credentials for authentication purpose, does not exist to date to the best of the author's knowledge – this is where the novelty of SUPA stands. However, SUPA stores some user information (e.g. username) that, as discussed earlier, are used as the public key for SUPA-based authentication. Since usernames are used as public key, they do not form any part of authentication related secret credentials, and thus do not violate the SUPA principle of not storing users' secret credentials for authentication. The application of SUPA is not limited to scenarios of system users. SUPA may be used in scenarios where interfacing between two different entities (e.g. human-system, or system-system) requires authentication as crucial part of trust establishment.

References

Banne, S.S., & Shedge, K.N. (2012). *A Review of the Graphical Password Based Authentication Schemes*. International Journal of Science and Research (IJSR), 3(10), 2137-2139.

Baruah, K.C., Banerjee, S., Dutta, M.P., & Bhunia, C.T. (2015). A New Remote User Authentication Scheme based on Graphical Password using Smart Card. *International Journal of Security and Its Applications*, 9(12), 237-244.

Conklin, A., Dietrich, G., & Walz, D. (2004). Password-Based Authentication: A System Perspective. *Proceedings of the 37th Hawaii International Conference on System Sciences*, 1-10.

Hwang, M.S., & Li, L.H. (2000). A new remote user authentication scheme using smart cards. *IEEE Transactions on Consumer Electronics*, 46(1), 28–30.

Khalid, U., Ghafoor, A., Irum, M., & Shibli, M.A. (2013). *Cloud based Secure and Privacy Enhanced Authentication & Authorization Protocol*. 17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems (KES2013), Procedia Computer Science, 22, 680 – 688.

Kumar, A., & Bilandi, N. (2014). *A Graphical Password based Authentication based System for Mobile Devices*. International Journal of Computer Science and Mobile Computing, 3(4), 744-754.

Kumar, G., Anjala, & Sharma, J. (2014). Authentication Techniques in Computer Networks. *International Advanced Research Journal in Science, Engineering and Technology*, 1(2), 76-79.

Li, X., Niu, J.W., Ma, J., Wang, W.D., & Liu, C.L. (2011). Cryptanalysis and improvement of a biometric-based remote authentication scheme using smart cards. *Journal of Network and Computer Applications*, 34, 73–79.

Liu, J.Y., Zhou, A.M., & Gao, M.X. (2008). A new mutual authentication scheme based on nonce and smart cards. *Computer Communications*, 31(10), 2205–2209.

Liu, Y., Dong, B., Guo, B., Yang, J., & Peng, W. (2015). Combination of Cloud Computing and Internet of Things (IOT) in Medical Monitoring Systems. *International Journal of Hybrid Information Technology*, 8(12), 367-376.

Malempati, S., & Mogalla, S. (2011). *User Authentication using Native Language Passwords*. International Journal of Network Security & Its Applications (IJNSA), 3(6), 149-160.

Sahu, S.B., & Singh, A. (2014). *Survey on Various Techniques of User Authentication and Graphical Password*. International Journal of Computer Trends and Technology (IJCTT), 16(3), 97-102.

Sayed, S., Mohid, A., Pal, M., & Haji, M. (2016). *Graphical Password based Authentication System with Sound Sequence*. International Journal of Computer Applications, 138(12), 38-43.

Singh, D., Gour, P., & Thakur, R. (2014). *User Security in Cloud Using Password Authentication*. Int. Journal of Engineering Research and Applications, 4(6), 39-44.

Sreelatha, M. & Shashi, M. (2011). Modified schemes for authentication based on shape and text. *International Journal of Electrical, Electronics and Computer Systems (IJEECS)*, 1(2).

Vaithyasubramanian, S., Christy, A., & Lalitha, D. (2015). *Two factor Authentication for Secured Login Using Array Password Engender by Petri net*. International Conference on Intelligent Computing, Communication & Convergence (ICCC-2015), Procedia Computer Science, 48, 313 – 318.

Vaithyasubramanian, S., Christy, A., & Saravanan, D. (2015). *Two Factor Authentications for Secured Login in Support of Effective Information Preservation and Network Security*. ARPN Journal of Engineering and Applied Sciences. 10(5), 2053-2056.

Varghese, L., Mathew, N., Saju, S, & Prasad, V.K. (2014). *3-level Password Authentication System*. International Journal of Recent Development in Engineering and Technology, 2(4), 127-131.

Wang, S.Q., Wang, J.Y., & Li, Y.Z. (2013). *The Web Security Password Authentication based the Single-Block Hash Function*. International Conference on Electronic Engineering and Computer Science, IERI Procedia, 4, 2-7.

Zheng, Z., Liu, X., Yin, L., & Liu, Z. (2010). A Hybrid password authentication scheme based on shape and text. *Journal of Computers*, 5(5).