Asian Journal of Information Technology 5 (12): 1338-1342, 2006
© Medwell Journals, 2006

Server-Aided Signature Scheme Based on Password without Servers' Public Keys

Xianping Mao and Tianjie Cao
School of Computer Science and Technology, China University of Mining and Technology,
Xuzhou 221008, China

Abstract: In a roaming system, users who access the network system from different client machines would do digital signatures for some security goals. One of the major concerns in this system is how to protect the user's private data (such as long-term key) from disclosure. In this study, we present a server-aided RSA signature scheme based on password. In this scheme, the user's private key is split into two parts which are stored in two servers respectively and a roaming user who wants to make the RSA signature on a message only need to provide a password. If not all of the servers are compromised, the password and the private key would not be disclosed. Compared with He *et al.*'s protocol, our scheme has four advantages. First of all, the involved servers do not need public keys; second, our scheme has less communication rounds; third, the servers have symmetrical position which provides convenience for implementation; last, our scheme could run more efficiently.

Key words: Password, server-aided signature, RSA

INTRODUCTION

Suppose the following real-world scenario: A senior manager, who is far from his company for conference or vocation, wants to make digital signature on an important electronic-file. In this roaming application, some solutions to the problem could be suggested. Portable computer or smartcard in which the private key stored can be used. But this traditional method has disadvantages, this means is not a convenience way and an adversary could break into the portable computer or filch the smartcard to obtain user's private key. Another alternative is password-only schemes. A well-known fact in password-only mechanisms is that human typically choose weak, low-entropy passwords from a relatively small space of possibilities. The above characteristics could be leveraged by an eavesdropping adversary, who can mount an off-line dictionary attack easily if he derives useful information about the password from storage in servers/clients or login sessions. Consequently, the eavesdropping adversary could derive the users' private keys or forge users' signatures.

Because of the human-memorable characteristic, this convenience factor of password-only mechanisms lead people into temptation in the design of authentication or signature protocols. Many of the presented protocols are about authentication. Bellovin and Merritt[1] presented Password-only Authenticated Key Exchange protocol (PAKE), who is the first to suggest that in protocols only use a short password and no additional information. Jablon proposed a simple password-only authentication protocol in[2]. By using only a small password, his methods provide authentication and key establishment over an insecure channel and could defeat off-line dictionary attack. Ford and Kaliski[3] proposed a threshold protocol to provide the authentication functionality with multiple servers, an adversary learns nothing about user's password other than what it learns from its on-line password guesses as long as not of all servers are compromised. For each of one or more servers, the user engages in a password-hardening protocol with each server to obtain a hardened password from the password, none of the servers learns the password and the hardened passwords. The hardened passwords can then be combined to obtain additional strong secrets which none of the servers can determine alone. Jablon[4] subsequently presented a protocol with similar functionality in the password-only setting. In Jablon's protocol, the user can authenticate servers and retrieve his private key for temporary use on a client machine using just a password.

In 2005, He *et al.* proposed a signature protocol based on password[5]. The user's signature private key is split into two parts and storing in two well-protected servers. When a user needs to sign some documents,

Corresponding Author: Xianping Mao, School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221008, China

only his or her password is needed, which provides great mobility. The user first contacts one server called backup server and gets the first part of the private key, then the user asks the second server called signature server to sign the message with the second part of the private key, finally the user gets the signature by signing the output of the signature server with the first part of the private key. In He *et al.*'s protocol, the involved servers must have their own public keys, the servers and client should do some time-consuming operations such as constructing keyed message authentication codes, servers' signatures and doing symmetry/asymmetry cryptographic transformations. Beside those, it is more difficult for programming that servers in their protocol do not have the symmetrical position. More details about He *et al.*'s protocol could be found in the original study[5].

In this study, we present a server-aided RSA signature protocol which is based on Ford *et al.*'s scheme[3]. In our scheme, the private key is split into two parts which are stored in two servers respectively and a roaming user who wants to make the RSA signature on a message only need to provide a password. An adversary could not get any useful information about users' passwords and their private keys by the passive attacks and the active attacks if not of all servers are compromised. Compared with He *et al.*'s protocol[5], our scheme has four advantages. First of all, the involved servers do not need public keys; second, our scheme has less communication rounds; third, the servers have symmetrical position which provides convenience for implementation; last, message authentication codes, servers' signatures and symmetry/asymmetry cryptographic transformations do not be used in our protocol which can make our scheme run more efficiently than He *et al.*'s protocol.

We organize this study as the following: in the next section, we describe our server-aided password-only signature protocol in details. Then we make security analysis of our proposed protocol in study. At last, we conclude this study.

Proposed scheme: A digital signature scheme is a protocol that produces the same effect as the signature in real world. One of the most popular digital signatures is the RSA scheme proposed by Rivest *et al.*[6]. To set up the standard RSA scheme, two large, random and secure primes P and Q are introduced. Let $N = PQ$ and $\varphi(N) = (P-1)(Q-1)$. Two more integers e and d are chosen such that $ed = 1 \bmod \varphi(N)$. The pair (e, N) is made publicly known. However, d which is kept secret is the private key. The digital signature created by a user Alice for message $M \in Z^*_N$ is the integer C such that $C = H(M)^d \bmod N$ where H() is a secure hash function. Anyone can verify the correctness of the signature by checking whether $H(M) = C^e \bmod N$ or not.

In our scheme, for instance with the help of two servers, the RSA signature private key is split into two parts and then those two parts are stored in those two servers respectively. We use password harden protocol described in[3] to enhance the security of our password and then we derive some secure secret parameters for protecting user's private key, for example through key derivation function KDF[3,7].

We introduce firstly the following notations to express our description clearly and then we will give the protocol model with the help of two servers.

Notation:

A, ID_A: a user Alice and her identity
ω: the user Alice's weak password
M: a message to be signed
H: a secure hash family
h: Alice's personalized secure hash function, $h \in H$
H: a secure hash function to hash the message M, $H \in H$
S_i: the ith server, i = 1, 2
$Sig_A(M)$: user Alice's signature on message M

Initialization: The user Alice chooses a memorialized password ω and then she chooses some random integers, r_1 and r_2. Alice also generates a secure prime p where $p = 2p' + 1$, p' is also a large prime. We assume that Alice have a pairs for RSA signature key, (e, N) as public key and d as private key. N, which is the module of the RSA signature scheme, is the product of two large safe prime numbers. Alice randomly chooses two random integers d_1 and d_2 such that $\sum_{i=1}^{2} d_i = d \bmod \varphi(N)$. Last, Alice chooses a personalized secure hash function $h \in H$, where H is a secure hash family.

Alice uses her password to achieve some secret parameters. Firstly, she get secure harden passwords $R_i = h(h(\omega)^{r_i} \bmod p)$, then she get two secret parameters K_1 and K_2 from R_1, R_2, for instance with a Key Derivation Function (KDF): $K_i = KDF(R_1, R_2, i)$, i = 1, 2. One simple version of such function may be described as this: $K_i = h(h(R_1) \| h(R_2) \| h(\omega) \| h(i) \bmod p) \bmod p$, i =1, 2.

After initialization, Alice only needs to learn a memorialized password ω. In the client machine, certificates for p, N and e is stored. Alice can verify all those certificates.

In the storage of the ith server S_i where i =1, 2, information about Alice includes the following parameters, ID_A, r_i and $q_i = K_i^{-1} d_i$ where K_i^{-1} is the inverse of $K_i \bmod \varphi(N)$. Note that to ensure K_i has its inverse, the random integers, r_1 and r_2, may be chose more than once.

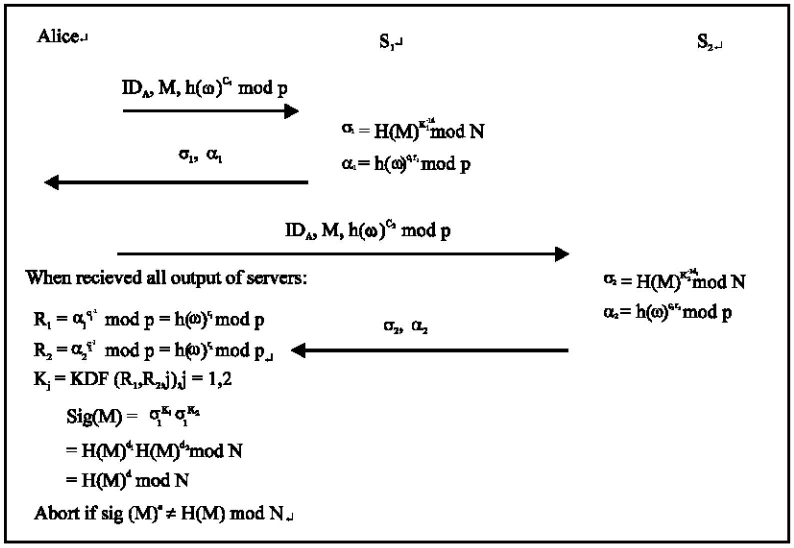
Alice
S_1
S_2
$ID_A, M, h(\omega)^{C_1} \bmod p$
$\sigma_1 = H(M)^{K_1^{-1}d_1} \bmod N$
$\alpha_1 = h(\omega)^{c_1 r_1} \bmod p$
σ_1, α_1
$ID_A, M, h(\omega)^{C_2} \bmod p$
When recieved all output of servers:
$\sigma_2 = H(M)^{K_2^{-1}d_2} \bmod N$
$\alpha_2 = h(\omega)^{c_2 r_2} \bmod p$
$R_1 = \alpha_1^{c_1^{-1}} \bmod p = h(\omega)^{r_1} \bmod p$
$R_2 = \alpha_2^{c_2^{-1}} \bmod p = h(\omega)^{r_2} \bmod p$
σ_2, α_2
$K_j = KDF (R_1, R_2, j), j = 1,2$
$Sig(M) = \sigma_1^{K_1} \sigma_1^{K_2}$
$= H(M)^{d_1} H(M)^{d_2} \bmod N$
$= H(M)^{d} \bmod N$
Abort if $sig (M)^{e} \neq H(M) \bmod N$

Fig. 1: Two-server protocol execution

Protocol execution: For each of the servers S_1 and S_2, the user Alice engages in the following process with the ith server S_i to obtain S_i's partial signature on message M, where i = 1, 2.

- $A \to S_i$: ID_A, M, $h(\omega)^{c_i} \bmod p$

Alice chooses a random integer c_i for server S_i, computes:

$$b_i = h(\omega)^{c_i} \bmod p \qquad (1)$$

Then she sends triple including her identity, the message M to be signed and b_i, $<ID_A, M, b_i>$ to server S_i.

- $Si \to A : \sigma_i = H(M)^{K_i^{-1} d^i} \bmod N \ \alpha_i = h(\omega)^{c_i d_{A_i}} \bmod p$

After S_i receives $<ID_A, M, b_i>$, S_i uses ID_A to find out the storage of Alice, $K_i^{-1} d_i$ and r_i, subsequently computes:

$$\sigma_i = H(M)^{K_i^{-1} d^i} \bmod N \qquad (2)$$

$$\alpha_i = h(\omega)^{c_i d_{A_i}} \bmod p \qquad (3)$$

At last, S_i returns $<\sigma_i, \alpha_i>$ to Alice.

Upon receiving $<\sigma_1, \alpha_1>$, $<\sigma_2, \alpha_2>$ from the servers S_1 and S_2 respectively, Alice computes:

$$R_i = h(\alpha_i^{c_i^{-1}} \bmod p) = h(h(\omega)^{r_i} \bmod p) \qquad (4)$$

and then derives secrets K_1, K_2 from the hardened passwords R_1, R_2, for instance with a key derivation function:

$$K_j = KDF (R_1, R_2, j), j = 1, 2. \qquad (5)$$

Alice computes S_1 and S_2's partial signatures on message M as:

$$Sig_1 = \sigma_1^{K_1} = H(M)^{d_1} \bmod N, \qquad (6)$$

$$Sig_2 = \sigma_2^{K_2} = H(M)^{d2} \bmod N \qquad (7)$$

then she derives the RSA signature on message M:

$$Sig_A(M) = \sigma_1^{K_1} \sigma_2^{K_2} \bmod N = H(M)^{d_1} H(M)^{d_2} \bmod N. \qquad (8)$$

At last, Alice verifies the signature on message M with her public key. If it is correct, the signature $Sig_A(M)$ is valid. Figure 1 shows the process of two-server protocol execution. We can see clearly that our protocol is easily generalized to multiple servers.

Compare with the existing protocol: Compared with the two-server signature protocol (SADS) described in[5], our new scheme's advantages are four-folds. First of all, in He *et al.*'s scheme, there is a flaw that the servers should have their own private/public keys, while in our scheme, the involved servers do not need public keys; second, our scheme has less communication rounds than He *et al.*'s scheme; third, in our scheme, the servers have symmetrical position which provides convenience for implementation while the two involved servers in He *et al.*'s scheme do not have this advantage; at last, there are not message authentication codes, servers' signatures and symmetry/asymmetry cryptographic transformations in our protocol which can make our scheme run more efficiently than He *et al.*'s protocol.

Security analysis of our protocol: All our security analysis is done under the framework of the Dolev-Yao Threat Model[8]. In this Model, we consider that network is the vulnerable environment and could just think of the open network as the adversary. But we do not consider the adversary could do all things, such as guessing a random number, etc. In this section, we also assume that the adversaries do not know the password initially and not all of the servers are compromised. For instance, in the two-server model, at most one of the servers could be compromised. Under those assumptions, we can show informally that our protocol is secure. We only do our analysis with the two-server model. The analysis for more servers' model is similar.

We first describe the adversarial model before our analysis. There exist two kinds of attacks, one of which is passive attacks. In the passive attacks, a corrupted server continues to operate according to the protocol, but the adversary may monitor its internal state. Another one where an adversary controls all interactive messages between user and servers to obtain the useful information, are called active attack.

We claim that an adversary can not achieve his malevolent goal, namely getting users' passwords or forging users' signatures through the passive attack or the active attack. Firstly, we point out that the password could not be compromised by the passive attack if not of all the servers being corrupted. Because of the servers' symmetrical position, we assume that Server S_1 is compromised, thus all of the storage in Server S_1 for Alice, r_1 and $q_1 = K_1^{-1}d_1$, is open to the adversary. We also make a reasonable assumption that the adversary could obtain the transcripts of interactions between user and Servers. We could see clearly that from the information stored in server, the adversary could not mount an off-line dictionary attack to obtain user's password. The adversary could get $h(w)^{c_1 r_1} \bmod p$ for interactions and r_1 from storage in servers, but the random exponent c_1 serves as a blinding factor, so the adversary does not learn any information about the user's password from the above two values. And K1, which is generated from the harden passwords R_1 and R2, could not be confirmed when only one server is compromised[3], thus the adversary could not derive the partial private key from q_1 stored in Server S_1.

We also show that an adversary could not get any useful information about password or secret key and could not forge a signature via the active attack. Because S_2 is not compromised, we can see that K_1 and K_2 is secure only S_1 is compromised. We take into account the following two situations. If the adversary impersonate user Alice to spoof servers, he can send a pseudo message M' to servers, then he can get the outputs of servers: $H(M')^{K_1^{-1}d_1} \bmod N$ and $H(M')^{K_2^{-1}d_2} \bmod N$, but he could not get any useful information about the secret K_1 and K_2 which are derived from the password ω, thus he could not obtain the servers' valid partial signatures on message M', then he could not derives the right signature on message M'. A verifier can detect the false signature with Alice's public key. Another consideration is that the adversary assumes as the servers to cheat user Alice. In this situation, the adversary also could not obtain Alice's password from b_1 because of the random exponent c_1 and the adversary do not know the secret exponent $K_1^{-1}d_1$ and $K_2^{-1}d_2$, then he may generate error partial signature on message M such as $H(M)^{a_1} \bmod N$ and $H(M)^{a_1} \bmod N$. When Alice generates signature on message M and verifies it, she will detect the error.

CONCLUSION

In this study, we present a server-aided RSA signature protocol based on password for a roaming user. In our scheme, the private key is split into two parts which are stored in two servers respectively and a roaming user who wants to make the RSA signature on a message only need to provide a password. An adversary could not get any useful information about users' passwords and their private keys or forge a signature if not of all servers are compromised. The protocol is easily generalized to multiple servers. Compared with He *et al.*'s protocol, our scheme has four advantages: the involved servers do not need public keys; less communication rounds; the servers have symmetrical position which provides convenience for implementation; and more efficiency.

ACKNOWLEDGMENT

We thank the support of Open Research Foundation of the State Key Laboratory of information Security, China.

REFERENCES

1. Bellovin, S.M. and M. Merritt, 1992. Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks, IEEE Symposium on Research in Security and Privacy, IEEE, pp: 72-84.

2. Jablon, D., 1996. Strong Password-Only Authenti-cated Key Exchange, Computer Communication Review, ACM SIGCOMM, pp: 5-26.
3. Ford, W. and B.S. Kaliski Jr, 2000. Server-Assisted Generation of A Strong Secret From a Password, In 9th International Workshops on Enabling Technologies (WET ICE 2000), pp: 176-180.
4. Jablon, D., 2001. Password Authentication Using Multiple Servers, RSA Cryptographers' Track 2001, LNCS vol. 2020, Springer-Verlag, pp: 344-360.
5. He, Y., C. Wu and D. Feng, 2005. Server-Aided Digital Signature Protocol Based On Password, 39th Annual 2005 International Carnahan Conference on Security Technology (CCST '05), pp: 89-92.
6. Rivest, R., A. Shamir and L. Adleman, 1978. A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM, pp: 120-126.
7. RFC2898-PKCS #5: Password-Based Cryptography Specification, available at: http://www.ietf.org/rfc/rfc2898.txt
8. Dolev, D. and A.C. Yao, 1983. On the Security of Public Key Protocols, IEEE Transactions on Information Theory, pp: 198-208.