EC Transactions Use Different Web-Based Platforms

Whe Dar Lin

The Overseas Chinese Institute of Technology
Dept of Information Management,
No. 100, Chiao Kwang Road Taichung 40721, Taiwan

Abstract. A review of the methodological aspects of the EC transactions use different Web-based platforms is presented. Our design techniques for complex EC environments are present to avoid the viruses infected EC transaction files. When an executive service code is ordered it may get infected with some viruses before the signature is attached to it. The infected EC transaction files cannot be detected by signature verification and the origin of the infection order cannot be specified by different Web-based platforms. Applied our EC transactions use different Web-based platforms, these techniques lead to an EC framework with clients, agents, EC application servers and EC system management. The paper discusses how Web-based platforms can use signature method together with detection infected in the preprocessing step enable us to specify the origin of the infection side. The impact of our proposed method can improve Web-based platforms relationships between EC transaction agents.

1 Introduction

In EC era, there has been a general trend towards partnership Web-based platforms together with a reduction and consolidation of the supply based in order to have better EC application relationships with fewer Web-based platforms. It is an important step that an execution service code can be detected by checking the consistency of original transaction order with its accompanying digital signature [1], [2], [5], [6], [8]. The cooperative Web-based platforms approach to supplier is characterized by long term contracts, integrated key protocols, share marketing database, and a commitment to partner relationships.

We don't think this is a good architecture using in Web-based platform. [12][13][14] Some other efforts through the directory services, name services, library service to reduce complex about viruses impaction. This approach yields an efficient solution for simple and static point-to-point interaction in collaborative EC transaction order. Nevertheless, we need face dynamic communication commit from different Web-based platforms for multiple relationships between one another on supply chain management. We will choice the ID-based scheme [10], a large public-key file is not required because each user's public key is nothing but an identity and communication cost is low using in EC applications for different Web-based platforms.

Our proposed signature method is based on a public key scheme. We assume that special entities, such as transaction suppliers in supply chain are not reliable for

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2658, pp. 1059–1068, 2003.

concerning the signature forgery or computer viruses. The rest of this paper is organized as follows. Sect. 2 gives some general assumptions about viruses and transaction supplier and Web-based platform. The proposed our security Web-based platform scheme will be described in Sect. 3 and Sect. 4 the correctness shall be discussion. Our proposed scheme security analysis is described in Sect. 5, and we discuss some implementation Web-based platform issue in Sect. 6. We conclude with some final remarks shall be stated in Sect.7.

2 Assumptions and Model

Here, we provide several assumptions about viruses and transaction supplier, and security in general using in our protocol. A protocol is a set of rules and conventions that define the communication framework between two or more EC transactions. After that, we shall describe the Web-based platforms.

2.1 General Assumptions about Viruses and Transaction Supplier

Our proposed scheme relies on the existence of a hash function h. Specifically,

Assumption 1: We assume there exist a function h such that:

On random input (r_i, m_i), it is difficult to generate (e_i) such that $h(r_i, m_i)= (e_i)$. More generally, it is difficult to generate such (e_i) on input (r_i, m_i) and samples of signature on random messages signed with EC-based platforms.

Assumption 2: Web-based platform act according to the following conditions:

- They create transaction supplier honestly in marketing channels.
- They do not refuse to reply to the requests or questions from the EC agent.
- Viruses can infect and incubate warehouse files and EC order files on Web-based platforms.
- Viruses can damage both warehouse files and EC order files on Web-based platforms.

Our proposed automatic signature scheme of protection against viruses works upon the following security assumptions [1], [2], [3], [4], [9], [11]:

- Solving the discrete logarithm problem is difficult, so we use in cryptographic protocols.
- Inverse calculation of one-way hash functions is difficult that we use assumption is often made in cryptographic protocols.
- Distributed operating system in Web-based executes a verification program properly. This assumption is important because improper events must be ruled out in the verification procedure so that one can rely on the results of the verification.

2.2 Client-Server Transaction Model for Procurement

In this paper, the transaction supplier on the Web-based platform will reduce the amount of complete purchased work items and the turnaround time the deliver the order transaction services.

- Creates the service system library, good library, customer library and included files,
- Signs all library files following every subroutine, procedure and initial data structure,
- Calculates the fingerprints of the included files with a one-way hash function and creates the database of the fingerprints (fingerprint data file),
- Computes proxy integers and attaches them to the Web-based platform,
- Creates the signatures for the shops on Web-based platform and the fingerprint data file.

The included files and customer relation data are text files and are accessed during preprocessing. The transaction includes all processes from preprocessing to linking. The fingerprint of an included file is an output of a one-way hash function taking the included file as an input file. The fingerprint data file is composed of pairs of the names of the included files and their fingerprints. From this file, one can authenticate the validity of the included files.

3 Our Proposed Automatic Signature Scheme

In this section, we shall illustrate the structure of our proposed method. In our proposed protocol, The transaction supplier is denoted by u_m, the EC server u_s and the mobile user, client u_c.

They follow the steps below.
Step 1: The EC server u_s sends the request the Web-based platform R to the transaction supplier u_m.
Step 2: The transaction supplier u_m sends the Web-based platform R to the EC server u_s.
Step 3: The mobile user, client u_c sends the request the Web-based platform R for the transaction order P to the EC server u_s.
Step 4:The EC server u_s sends the executable transaction bill service code M to the client u_c.

3.1 The Initialization Phase

The system parameter are listed as follows:
(1)We use a prime p with $2^{511} < p < 2^{512}$, a prime divisor q|p – 1, a generator g with order q over GF(p), and
(2) the security information of the transaction supplier u_m: x_m
(3) the public information of the transaction supplier u_m: y_m

$$y_m \equiv g^{x_m} \bmod p \tag{1}$$

(4) the security information of the EC server u_s: x_s
(5) the public information of the EC server u_s: y_s,

$$y_s \equiv g^{x_s} \bmod p \tag{2}$$

(6) the security information of the mobile user, client u_c: x_c
(7) the public information of the mobile user, client u_c: y_c,

$$y_c \equiv g^{x_c} \bmod p \tag{3}$$

3.2 The EC Server Requests the Delivery of a Web-Based Platform

When a server u_s requests the delivery of a Web-based platform named R. The server sends his secret key x_s along with four integers calculated from a random number k_1. The computation is as follows.

(1). Generate a random number k_1 satisfying $0<k_1<q-1$
(2). Compute

$$r_1 \equiv g^{k1} \bmod p \tag{4}$$

$$m_1 \equiv u_s \| R \bmod p \tag{5}$$

$$e_1 \equiv h(r_1, m_1) \bmod q \tag{6}$$

$$s_1 \equiv k_1 - x_s e_1 \bmod q \tag{7}$$

where h is a one-way function and || denotes concatenation.
(3). Send (u_s, R, e_1, s_1) with the request to the transaction supplier u_m.

3.3 The Transaction Supplier Sends the Web-Based Platform to the EC Server

When the transaction supplier u_m receives the integers (u_s, R, e_1, s_1), he verifies the server u_s requests and then delivers a Web-based platform C_R. The transaction supplier u_m verifies (u_s, R, e_1, s_1) by computing the following equations.

(1). compute

$$r^* \equiv g^{s1} y_s^{e1} \tag{8}$$

check whether the congruence $e_1=h(r^*, u_s\|R)$ holds.
If it holds, the transaction supplier u_m does as follow.
(2). Generate a random number k_2 satisfying $0<k_2<q-1$
(3). Compute

$$r_2 \equiv g^{k2} \bmod p \tag{9}$$

$$t_m \equiv x_m + k_2 * r_2 \bmod q \tag{10}$$

$$m_2 \equiv C_R \| t_m \bmod p \tag{11}$$

$$e_2 \equiv h(r_2, m_2) \bmod q \tag{12}$$

$$s_2 \equiv k_2 - x_m e_2 \bmod q \tag{13}$$

(4). Send the Web-based platform C_R and its signature (C_R, t_m, e_2, s_2) to u_s

3.4 The Client Requests to the EC Server to Send Service Code for an Order P

A client user u_c wants to get a service code for an order P, and the server creates an executable service code M and its signature. When the server u_s receives the integers (C_R, t_m, e_2, s_2), he verifies that a server u_m sends then accept the Web-based platform C_R. The server u_s verifies (C_R, t_m, e_2, s_2) by computing the following equations.

(1). Compute

$$r^* \equiv g^{s2} y_m^{e2} \tag{14}$$

check whether the congruence $e_2=h(r^*, C_R \| t_m)$ holds. If it holds, the server u_s does as the client wishes. In the process of calculating a signature, the client executes the following steps:

(2). Generate a random number k_3 satisfying $0 < k_3 < q-1$

(3). Compute

$$r_3 \equiv g^{k3} \bmod p \tag{15}$$

$$m_3 \equiv u_c \| P \bmod p \tag{16}$$

$$e_3 \equiv h(r_3, m_3) \bmod q \tag{17}$$

$$s_3 \equiv k_3 - x_s e_3 \bmod q \tag{18}$$

(4). Send (u_c, P, e_3, s_3) with the request to the server u_s.

3.5 The Server Processes the Order and Sends the Execution Service Code to the Client

When the server u_s receives the integers (u_c, P, e_3, s_3), he verifies that a client u_c requests that the Web-based platform C_R processes the order P. The server u_s verifies (u_c, P, e_3, s_3) by computing the following equations

(1). compute

$$r^* \equiv g^{s3} y_s^{e3} \bmod p \tag{19}$$

check whether the congruence $e_3=h(r^*, u_c||P)$ holds. If it holds, in the process of calculating a signature, the server processes the order P and executes the following steps:

(2). Generate a random number k_4, satisfying $0 < k_4 < q-1$

(3). Compute

$$r_4 \equiv Pg^{-k4} \mod p \tag{20}$$

$$s_4 \equiv k_4 - r_4 x_s \mod q \tag{21}$$

$$M \equiv C_R(P) \mod p \tag{22}$$

$$m_4 \equiv u_c||P||M \mod p \tag{23}$$

$$e_4 \equiv h(r_3, m_3) \mod q \tag{24}$$

$$t_s * h(M) \equiv t_m + r_2 x_s \mod q \tag{25}$$

$$r_2 \equiv g^{s2} y_m^{e2} \mod p \tag{26}$$

(4). Send (r_4, s_4, t_s, M) with the request to the server u_s.

The Web-based platform can be used only by the server u_s because the integer t_m is the proxy key of the transaction supplier u_m. However, if another user happens to know u_m's secret key, then he can use this u_m's Web-based platform and create a bad executable service code with this proper proxy. Then, the executable service code may damage someone's computer. The basic rule for this event is that the owner u_m of the Web-based platform storing the proxy integer key t_m is responsible for the damage so that the transaction supplier must hide his secret key.

3.6 Signature Verification

When a user executes a service code M made by u_s, the use verifies the signature with the transaction supplier's public key and server's public key. That he checks whether the signature satisfies the congruence such that

$$g^{t_s h(M)} \equiv y_m (r_2 y_s)^{r_2} \bmod p \tag{27}$$

Since the signature (r_4, s_4, t_s, M) is automatically created, an infection of a generated executable service code can be detected in the above verification process. Using a signature can possibly be created for a contaminated executable service code, and in this case, the infection can be detected by the verification. We can determine who causes the injection.

4 Correctness of Our Scheme

Theorem 1:
1. The transaction supplier u_m can get the correctness request from the server u_s.
2. The server u_s can get the correctness from transaction supplier u_m.
3. The server u_s can get the correctness request from client u_c.

Proof:
(1) The transaction supplier u_m, receiving (u_s, R, e_1, s_1) from u_s can compute
$r^* = g^{s1} y_s^{e1} = g^{k1-xse1} y_s^{e1} = g^{k1-xse1} g^{xse1} = g^{k1} = r_1$
$h(r^*, u_s||R) = h(r_1, u_s||R) = h(r_1, m_1) = e_1$
Then we can be sure the signature of (u_s, R) is (e_1, s_1)
(2) The server u_s, receiving (C_r, t_m, e_2, s_2) from u_m can compute
$r^* = g^{s2} y_m^{e2} = g^{k2-xme2} y_m^{e2} = g^{k2-xme2} g^{xme2} = g^{k2} = r_2$
$h(r^*, C_r||t_m) = h(r_2, C_r||t_m) = h(r_2, m_2) = e_2$
Then we can be sure the signature of ($C_r||t_m$) is (e_2, s_2)
(3) The server u_s, receiving (u_c, P, e_3, s_3) from client u_c can compute
$r^* = g^{s3} y_c^{e3} = g^{k3-xce3} y_c^{e3} = g^{k3-xce3} g^{xce3} = g^{k3} = r_3$
$h(r^*, u_c||P) = h(r_3, u_c||P) = h(r_3, m_3) = e_3$
Then we can be sure the signature of (u_c, P) is (e_3, s_3)

Theorem 2:The server u_s can processes the correctness transaction order request from the client u_c.

Proof: The client u_c, receiving (r_4, s_4, t_s, M) from server u_s can compute

$$g^{t_s h(M)} \equiv y_m (r_2 y_s)^{r_2}$$

$$where \quad g^{t_s h(M)} \equiv g^{t_m + x_s r_2} \equiv g^{x_m + k_2 r_2 + x_s r_2} \equiv y_m r_2^{r_2} y_s^{r_2} \equiv y_m (r_2 y_s)^{r_2}$$ Then we can be sure the signature of M is (t_s, M) and use the public key of transaction supplier and server.

5 Security Considerations

Theorem 3: The crackers want to reveal the secret key from public key is computing impossible.

Proof: The crackers want to from the public key $y_m \equiv g^{xm} \bmod p$, $y_s \equiv g^{xs} \bmod p$, and $y_c \equiv g^{xc} \bmod p$ to reveal x_m, x_s, and x_c, they have to solve the discrete logarithm problem.

Theorem 4: The malicious users want to forger the request from the proposed scheme is computing impossible.
(1) The transaction supplier u_m get the false receive from malicious users.
(2) The server u_s can get the false request from malicious users.

Proof:
(1) The transaction supplier can use (u_s, R, e_1, s_1) to verify
$r^* = g^{s1}y_s^{e1} = g^{k1\text{-}xse1}y_s^{e1} = g^{k1\text{-}xse1}g^{xse1} = g^{k1} = r_1$, But the malicious users construct (u_s, R, e_1', s_1') without know x_s, the congruence, $r^* = g^{s1'}y_s^{e1'} = r_1$. They have to solve the discrete logarithm problem.
(2) The proof method is similar as part (1).

Theorem 5: The malicious users want to forger the signature from the proposed scheme is computing impossible.
(1) The server u_s can get the false Web-based platform from malicious user.
(2) The client u_c can get the false executable service code from malicious user.

(1) Proof: The server u_s the receive (C_r, t_m, e_2, s_2) from u_m can compute
$r^* = g^{s2}y_m^{e2} = g^{k2\text{-}xme2}y_m^{e2} = g^{k2\text{-}xme2}g^{xme2} = g^{k2} = r_2$
$h(r^*, C_r||t_m) = h(r_2, C_r||t_m) = h(r_2, m_2) = e_2$
We will sure the signature of ($C_r||t_m$) is (e_2, s_2)
Case I: The malicious users use the false (C_r', t_m, e_2, s_2) signature send to server u_s
$r^* = g^{s2}y_m^{e2} = g^{k2\text{-}xme2}y_m^{e2} = g^{k2\text{-}xme2}g^{xme2} = g^{k2} = r_2$
$h(r^*, C_r'||t_m) = h(r_2, C_r'||t_m) = h(r_2, m') = e' \neq e_2$
We will sure the signature of (C_r', t_m, e_2, s_2) is false signature.
Case II: The malicious users use the false (C_r, t_m', e_2, s_2) signature send to server u_s
$r^* = g^{s2}y_m^{e2} = g^{k2\text{-}xme2}y_m^{e2} = g^{k2\text{-}xme2}g^{xme2} = g^{k2} = r_2$
$h(r^*, C_r||t_m') = h(r_2, C_r||t_m') = h(r_2, m') = e' \neq e_2$
We will sure the signature of (C_r, t_m', e_2, s_2) is false signature.
Case III: The malicious users use the false (C_r', t_m', e_2', s_2') signature send to server u_s
$r^* = g^{s2'}y_m^{e2'} = r_2'$, $h(r_2', C_r'||t_m') = e_2'$
But the malicious users choose another Web-based platform C_r' and another proxy key t_m' without know x_m, the congruence equation that they have to solve the discrete logarithm problem.

(2) The proof method is similar as part (1).

Theorem 6: The computational complexity for an intruder to cryptanalyze a new session key in our scheme, after having received previous transaction order data, the public key, the old messages and the new message is as hard to cryptanalyze a plaintext in the ElGamal scheme when the order of g is a prime.

Proof. Let B_O correspond with the problem of breaking our scheme and B_E with breaking the ElGamal encryption scheme with the order of g is prime. There is an existence of a polynomial time B_E implies the existence of a polynomial time B_O.

We suppose it is possible to compute the discrete log of an output of G after seeing a sequence of all transaction orders. Without loss of generality we can assume success rate is an ε. We will use B_E by B_O as a procedure processing in computing feasible by a polynomial time. So, the input parameters for B_E is K_i=(r_i, s_i, t_j, M_k).

Now B_E computes H_I and $h(r_I, m_I)$ which will be used as input parameters for B_O as follows. B_E chooses $x_1, x_2, \ldots, x_k \in Z_q$ at random and computes $y_1 \equiv g^{x_1}, y_2 \equiv g^{x_2}, \ldots, y_k \equiv g^{x_k}$ Then B_E has obtained a public message H_I=(p, q, g

$y_1,\ldots, y_{k-1}, y_k$). So B_E computes $y_i^{r_j} \equiv g^{x_i r_j}$ $for\ i = 1,2,..., p,\quad j = 1,2,..., q.$ Thus, uses the H_i=(p, q, g $y_1,\ldots, y_{k-1}, y_k$) and K_i=(r_i, s_i, t_j, M_k) to B_O. In computing feasible condition for any queried transaction order by B_O, B_E computes by using H_i=(p, q, g $y_1,\ldots, y_{k-1}, y_k$) and sends h(r_1, m_1) message to B_O. The EC transaction protocol will complete. Finally, B_E outputs the result of B_O, which is an $1/\varepsilon$ with non-negligible probability cases. We would find the discrete log in expected $1/\varepsilon$ steps. We prove under reasonable assumptions, that our proposed protocol is computationally secure even on public network within different Web-based platforms.

6 Performance

In this section, we shall calculate performance of our new method
I: The transmission cost on the delivery of the Web-based platform
In our method
Step 1: Send (u_s, R, e_1, s_1) with the request to the transaction supplier u_m.
Step 2:Send (C_R, t_m, e_2, s_2) with the response to the server u_s. Send the Web-based platform C_R and its signature (C_R, t_m, e_2, s_2) to u_s
II: The computation cost on verifying the signature of an executable program
In our proposed method verifies the signature.

$$g^{t_s h(M)} \equiv y_m (r_2 y_s)^{r_2} \bmod p$$

Our method uses 2 modular exponentiation computing time and 1 hash function computing time.
III: The security model comparison in implement.
In our proposed method: The client checks if the signature satisfies the congruence

$$g^{t_s h(M)} \equiv y_m (r_2 y_s)^{r_2} \bmod p$$

The server checks if the signature satisfies the congruence
e_3=h(r^*, u_c||P) and $M \equiv C_R(P) \bmod p$.

7 Conclusion

We proposed a protocol that developed to allow clients to get committing Web-based platform signature. In our signature scheme the public key of the Web-based platform owner is used for signature verification that a signature is created from both the public key of the transaction supplier and that of the server Web-based platform owner. In our proposed method, transaction suppliers no use any table to store EC Web-based servers' information that will more efficient and safety than any others EC Web-based systems. Without identify creator, our effective approach on the public keys lead to a verification procedure, and created signatures are checked relatively fast. The most nature extension to this novel protocol scheme is a server-based signature that integrated together with EC application package will allow client and the server to commit with one another.

References

1. T. ElGamal, "A public key cryptosystem and a signature ,scheme based on discrete logarithms," IEEE Trans. Inf. Theory, vol.IT-31, no.4, pp.469–472, 1985.
2. C,P, Schnorr, "Efficient identification and signatures for smart Cards," Advances in Cryptology-CRYPTO'89, LNCS 435, pp.239–252, Springer-Veriag, 1990.
3. R.C, Merklc, "A fast software one-way hash function," J. of Cryptology, vol.3, no.1, pp.43–58, 1990.
4. C. Park, "A Fiat-Shamir-like identification protocol without a highly reliable trusted center," Proc. the 1992 Symp. on Cryptography and lnf. Security, no.6D, 1992.
5. Y. Desmedt and Y. Frankel, "Multi-signatures for virus protection," Proc, the 5th lnt'l Computer Virus and Security Conf., 1992.
6. E. Okamoto, "Integrated security system and its application to anti-viral methods," Proc. the 6th Virus and Security Conf, 1993.
7. M. Cyirault, "Self-certified public keys," Advances in Cryptology-EUROCRYP'FO'91, LNCS 547, pp.490–497, Springer-Veriag, 1991.
8. K. Usuda, M. Mambo, T. Uyematsu and E. Okamoto “Proposal of an automatic signature scheme using a Web-based platform”, IEICE Trans. Fundamentals E79-A (1) (1996) 94–101
9. K. Nyberg, R.A. Rueppel, “Message recovery for signature scheme based on the discrete logarithm problem,” Designs, Codes and Cryptography, No 7, pp.61–81, 1996
10. A. Shamir, “Identity-based cryptosystem based on the discrete logarithm problem,” Proc. CRYPTO'84, 1985, pp. 47–53.
11. D. W. Manchala, “E-Commerce Trust Metrics and Models,” J. of IEEE Internet Computing, March 2000, pp. 36–44.
12. R. Sherwood, B. Bhattacharjee and A. Srinivasan, “A Protocol for Scalable Anonymous Communication,” Proc. the IEEE Symposium on Security and Privacy, 2002, pp.1–12.
13. A. Ginige and S. Murugesan, “Web Engineering: An Introduction,” J. of IEEE MultiMedia, January 2001, pp.14–18.
14. J. B. Lim and A. R. Hurson, “Transaction Processing in Mobile, Heterogeneous Database Systems,” IEEE Trans. On Knowledge and data Engineering, Vol. 14, No. 6, 2002, pp.1330–1346.
15. Jan, J.K. and Whe Dar Lin :An Efficent Anonymous Channel Protocol in Wireless Communications, IEICE Trans. on Communications, Vol.E84-B, No.3, PP.484–491, 2001.
16. Whe Dar Lin, “Using Marketing Factors Associated with Web site in E-commerce,” The International Conference on ASEMA, pp7~11, 2002.