2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing

A Low-level Security Solving Method in Grid

Weifeng Sun, Boxiang Dong, Zhenquan Qin*, Juanyun Wang, Mingchu Li
School of Software
Dalian University of Technology
Dalian Liaoning, China
wfsun@dlut.edu.cn, bxdong7@gmail.com, qzq@dlut.edu.cn,wjy48981581@gmail.com, mingchuli@dlut.edu.cn

***Abstract*—Computational grids are a promising platform for solving platform for solving large-scale resource intensive problems. Security problems become an urgent and complex undertaking for the application of grid computing. Traditional approaches to proving certificates validity such as CRL and OCSP are problematic in some areas. A new mechanism LSSM(Low-level Security Solving Method) which can efficiently solve problems of security, time accuracy and overhead is introduced in detail. LSSM is able to realize better security and faster revocation without bringing about too much cost.**

Keywords- low-level; grid computing; certificate revocation

I. INTRODUCTION

Because grid based computational infrastructure[1] involves a wide variety of geographically distributed computational resources , storage systems, data sources and databases and presents them as a unified integrated resource, the mutual trust relationship needs to be established and removed in a dynamic manner in grid environments. As a result, security problems[2] become an urgent and complex undertaking before the widespread applications of gird computing.

Certificate revocation[3] is the action of eliminating the relationship between the public key and attributes embodied in a certificate. There are two traditional and widely used approaches to proving certificates' validity: Certificate Revocation List (CRL) and the Online Certificate Status Protocol (OCSP). However, these methods only focus on designing at high level in network, mainly arranging on application layer and presentation layer, which makes them problematic in areas of security, time accuracy and cost. In the paper, we put forward a security solving method LSSM from both low level and high level which is able to ealize better security and faster revocation without bringing about too much cost.

This paper is organized as follows. Section II is related work. The technical details of the architecture are presented in Section III. Section IV shows the analysis of LSSM, followed by a conclusion in Section V.

Supported by Nature Science Foundation of China under grant No.: 60673046，90715037，University Doctor Subject Fund of Education ministry of China under grant No.: 200801410028，National 973 Plan of China under grant No.: 2007CB714205 and Natural Science Foundation Project of Chongqing, CSTC under grant No.: 2007BA2024.

*Corresponding author.

II. RELATED WORK

A certificate revocation list (CRL)[4] is a list of certificates (or more specifically, a list of serial numbers for certificates) that have been revoked or are no longer valid. Certification authority (CA) associates a list with a time stamp and its own signature. However, CRL[5] can be so large in scale that the cost of CRL management and distribution will be too high. Besides, the update of certificates can not be done in real time.

The Online Certificate Status Protocol (OCSP)[6] is another Internet protocol used for obtaining the revocation status of an X.509 digital certificate. A CA answers a query about a certificate by returning its own digital signature of the certificate's validity status at the current time. Each validity proof generated by the OCSP has a great length and a digital signature is a computationally complex and expensiveoperation. In certain large applications, the OCSP may require CA to comput millions of signatures in a short time, which may lead OCSP being vulnerable to denial-of-service attacks.

These traditional methods are subject to serious problems in areas of security, time accuracy and cost because they are aimed at realizing certificate revocation just from high level in network such as application layer and presentation layer, without taking any consideration of arrangement from low level such as physical layer and transport layer.

Silvio Micali has proposed NOVOMODO[7] to deal with the certificate revocation. NOVOMODO uses a one-way hash function H enjoying the following properties: H is at least 10,000 times faster to compute than a digital signature; H produces 20-byte outputs, no matter how long its inputs and H is hard to invert. As a result, NOVOMODO is more efficient and faster. NOVOMODO enjoys higher security because it's almost impossible to forge a proof of validity or invalidity.

Dan Boneh[8] has provided Mediated RSA to realize fine-grained control of security capabilities. RSA signature generation is composed of two steps: message encoding and cryptographic primitive computation. The second step requires SEM's involvement since. In mRSA, a client does not possess its entire private key. The fact that the private key is not held in its entirety by any one party is transparent to the outside world, i.e., to the those who use the corresponding public key. Therefore, neither the client nor

978-0-7695-4222-5/10 $26.00 © 2010 IEEE
DOI 10.1109/IIHMSP.2010.184

IEEE computer society

the SEM can decrypt or sign a message without mutual consent. Besides, mRSA enables fast revocation.

In HMAC[9], the hardware tokens are made with unique serial number, capacity of on-board HMAC computations and capacity to keep some hidden parameters (HMAC secret keys) inside the token which must not be known to outside world, except known the party who need to authenticate a message or an user. HMAC deals with security problems based on physical layer and transport layer. The only way to be authenticated is to own the legitimate token (registered and key inserted by the TA) and know the client's password. Thus, attackers with just one of them can't undermine the system.

The three innovative mechanisms which can partially solve some existing problems introduced above lay the foundation of LSSM which can solve existing problems brought by traditional methods with both of high level and low level arrangement.

III. SYSTEM DESIGN

A. System Architecture

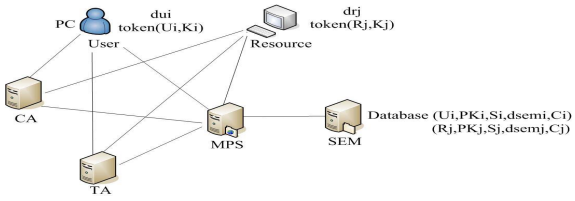

Figure1. Structure of the LSSM

Figure 1 shows the architecture of our system. In LSSM, there exists a Trusted Authority(TA), a Certificate Authority, a MyProxy Server (MPS)[10], a Semi-trusted Mediator (SEM), a number of users and resources.

TA mainly deals with the affair about tokens. The token can be owned by authenticated users or hardware resources in grid infrastructure such as PCs. Here we regard HMAC(a , b) as applying HMAC using "a" as the key and "b" is the data we want to compute the digest. A token has a unique serial number, denoted as i or j, and we use this serial number as the index to a specific user or resource. First of all, TA must first select a random generated master identity secret (for identification purpose), denoted as $S_{identity}$, which must also be known by SEM which need to validate a user or incoming message. TA needs to compute $K_{identification} = HMAC(S_{identity}, i)$ and then insert this key into the token. This is the secret key keep inside the token which should not be known to outside. Then TA sends the specific token to a user or hardware resource.

CA mainly deals with certificate authentication affairs. CA is responsible for sending certificates to users and resources and updating certificates and notifying SEM the latest information.

SEM is responsible for restoring important information about users and resources. It maintains a database including the serial numbers of information, their public key PK, $S_{identit}$, part of private key d_{sem} and certificate C published by CA.

RSA is connected and cooperated with SEM and responsible for association with users, resources, CA and TA.

B. Certificate Issuance

Any user and resource needs to apply CA for its own certificate. And the process for a resource is similar to that of a user. So in this section, the application process of a user named U_i in detail is introduced. Fisrt, U_i applies CA for a certificate. Then CA checks U_i 's identity and decides whether to approve the application. If CA decides to issue U_i a certificate, it will randomly select two different 20-byte values, Y_0 and X_0, and from them computes two 20-byte values, Y_1 and X_N, as follows. Value Y_1 is computed by hashing Y_0 once; and X_N by hashing X_0 N times: $X_1 = H(X_0)$, ..., $X_N = H(X_{N-1})$. Then the certificate $C = SIG_{CA}(S_N, PK, U, D_1, D_2, ..., Y_1, X_N)$ includes a serial number S_N, a public key PK, a user name U, an issue date D_1, an expiration date $D_2 = D_1 + N$,forms. At the same time, CA splits U_i's private key into d_{sem} and d_u. After that, CA sends d_u to U_i and d_{sem} and C to SEM through secure channel.

After the process introduced above, U_i successfully gets the certificate and SEM gets all the information it needs about U_i, which includes U_i, PK_i, S_i and d_{sem}.

C. The Construction of Mutual Trust Relationship

In LSSM, certificated and authenticated users need to apply for the right to utilize the computational resources. Here we take the process of user U_i's application for the particular resource R_j as an example and talk about the process concretely.

As discussed above, either the user U_i or resource R_j has applied for the certificate successfully and got the token including its serial number(U_i , R_j), part of private key (d_{ui}, d_{rj}) and $K_{identificatioon}$ (K_i, K_j) . Besides, SEM maintains the information about U_i and R_j. We denote by EC() and DC() the encoding and decoding functions.

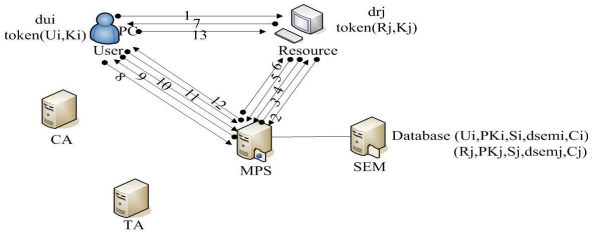

Figure2. Interaction in LSSM

Figure 2 shows the interaction of U_i, R_j, RSA and SEM during the process of mutual trust relationship construction.

(1) U_i encrypts random message m and his own id U_i with $R_j's$ public key PK_j to generate $C_1 = EC_{PK_j}(m, U_i)$. U_i sends C_1 to R_j.

(2) R_j receives C_1. R_j sends a request to MPS.

(3) MPS sends a random generated value RC to R_j.

(4) R_j have to response by comput r= HMAC(K_j, RC) using the token and then compute R_{rj}=HMAC(PK_j, r). Then R_j sends R_{rj} back to MPS for verification. At the MPS side, it must know which resource is trying to request, then it will retrieve the required parameters from the database on SEM (i.e. the resource specific token serial number R_j and PK_j), and compute the correct response value R_{mps} locally .Then, MPS can compare R_{rj} with R_{mps} to determine if the resource is a legitimate resource and owns the corresponding legitimate token.

(5) R_j forwards C_1 to MPS. At the same time, R_j computes $PC_r = C_1 d_{rj} \bmod n_j$.

(6) If R_{rj} equals R_{mps}, MPS will check if R_j is not revoked and, if so, compute a partial clear-text $PC_{sem} = C_1 d_{semj} \bmod n_j$ and send PC_{sem} to R_j. Else, MPS will not deal with C_1.

(7) R_j receives PC_{sem} and computes $m_0 = PC_{sem} * PC_r \bmod n_j$. Now R_j gets the message m_0 it computes with the help from MPS. R encrypts m_0, his own serial number R_j and random conversation key k_s using PK_i to generate $C_2 = EC_{PKi}(m_0, R_j, k_s)$. R_j transmits C_2 to U_i.

(8) U_i receives C_2. U_i sends a request to MPS.

(9) MPS sends a random generated challenge value RC' to U_i.

(10) U_i will have to response by compute $u = HMAC(K_i, RC')$ using the token and then compute $R_{ui} = HMAC(PK_i, u)$. Then U_i sends R_{ui} back to MPS for verification. At the MPS side, it must know which resource is trying to request, then it will retrieve the required parameters from the database on SEM (i.e. the user specific token serial number U_i and PK_i), and compute the correct response value R_{mps}' locally . MPS can compare R_{rj} with R_{mps}' to determine if the user is a legitimate user and own the corresponding legitimate token. With the low level arrangement of tokens, any user or resource which can get the assistance from MPS must has been authenticated and attackers are not able to compromise MPS.

(11) U_i forwards C_2 to MPS. At the same time, U_i computes $PC_u = C_2 d_{ri} \bmod n_i$.

(12) If R_{ui} equals R_{mps}', MPS will check that U_i is not revoked and, if so, compute a partial clear-text $PC_{sem}' = C_2 d_{semi} \bmod n_i$ and send PC_{sem}' to U_i. Else, MPS will not deal with C_2.

(13) U_i receives PC_{sem}' and gets the message m_1 and conversation key k_s which U_i computes with the help from MPS. Then U_i will check whether m equals m_1. If $m = m_1$, U_i is convinced that that both of $R_j's$ identity and current status of $R_j's$ EEC(the End Entity Certificate) are valid, because MPS will not help R_j decrypt C_1 only if R_j is valid and authenticated. Then U_i encrypts m with conversation key k_s to generate $C_2 = EC_{ks}(m)$. U_i sends C_2 to R_j. R_j receives C_2 and decrypts C_2 with k_s to generate m_3. R_j will check if $m_1 = m_3$ or not. If $m_1 = m_3$, R_j is convinced that both of $U_i's$ identity and current status of $U_i's$ EEC are valid, because MPS will not help U_i decrypt C_2 only if U_i is valid and authenticated. Now, the mutual trust relationship is constructed.

D. Certificated Revocation

In this section, we will focus on the process of update and revocation of $U_i's$ certificate. Normally, the CA updates C's proof of status by computing and transmitting an up-to-time proof to SEM. M time intervals after C's issuance CA hashes X_0 m($m \le N$) times: $X_1 = H(X_0)$, $X_2 = H(X_1)$, ..., $X_m = H(X_{m-1})$. Then CA gets X_m and transmits it with $U_i's$ serial number to SEM. SEM receives them and check whether if $U_i's$ certificate is valid or revoked. MPS selects $U_i's$ certificate $C = SIG_{CA}(S_N, PK, U, D_1, D_2, ..., Y_1, X_N)$ from the database in SEM and hashes X_m N-m times. So MPS gets the result of X_N and finds out whether if the result equals the X_N in $U_i's$ certificate. If so, SEM is convinced that $U_i's$ certificate is valid during the m*th* time interval after C's issuance. However, if CA wants to revoke $U_i's$ certificate C, the authority just needs to transmit the revocation proof Y_0 of C saved in CA and $U_i's$ serial number to SEM. After SEM receives the information, MPS gets the result Y_1 by hashing Y_0 one time. Then MPS selects $U_i's$ certificate $C = SIG_{CA}(S_N, PK, U, D_1, D_2, ..., Y_1, X_N)$ and finds out that the computation result equals the revocation proof Y_1 in C. Thus SEM is convinced that $U_i's$ certificate should be revoked and no longer be permitted to participate in grid computing.

IV. LSSM ANALYSIS

A. Security in Certificate Issuance

In the process of certificate issuance, CA keeps values Y_0 , X_0 , X_1 ,..., X_{N-1} in secret, while Y_1 and X_N are included in the certificate and transmitted to SEM. Besides, CA splits $U_i's$ private key into d_{sem} and d_u and sends du to U_i and d_{sem} and C to SEM through secure channel. Because the hash function H is essentially impossible to invert, attackers who only gets the knowledge of Y_1 or X_N can not forge the proof of revocation Y_0 or the proof of validity X_i. And neither U_i nor the SEM can decrypt or sign a message without mutual consent. As a result, any attacker who has compromised SEM and got part of the private key d_{sem} can not visit the grid services successfully.

B. Security and Overhead in Construction of Trust

In the process of the construction of mutual trust relationship, if U_i wants to apply for the right to utilize the computational resources R_j , both U_i and R_j needs the help from MPS. If any applicant who sends the request is revoked or invalid, MPS will not provide the assistance. So only if both U_i and R_j are valid can they trust each other and participate in the grid computing process, which leads to great improvement in security, making users and resources whose certificates have expired or been revoked and malicious attackers can not participate in grid computing. Besides, with the introduction of low-level deployment mechanism such as tokens in HMAC and certification policy between users or resources and MPS before interaction, if U_i or R_j wants to get the help from MPS, the user or resource must convince MPS that U_i or R_j has got the authenticated token with the specific $K_{identificatioon}$. Otherwise, MPS will not give the assistance. As a result, most probably attackers who intends to breach MPS or make denial-of-service attacks can not threat our system because they don't have an authenticated token and MPS will not deal with their requests. This process may also require to a more complex interaction between U_i or R_j with MPS because every time of cooperation between them needs validation. But this overhead is acceptable.

C. Time Accuracy and Overhead in Certificate Update

In the process of certificate update, CA needs to transmit $U_i's$ and $R_j's$ proof of status to SEM during every time interval. Because H always produces 20-byte outputs, Y_1, X_N, and all intermediate values X_i are 20-byte long. So the corresponding overhead and requirement for bandwidth are negligible. Silvio Micali[9] has proved that with proper mechanism, CA can handle extreme accuracy such as a time accuracy of 15 seconds, so LSSM can specify time with a predetermined accuracy: one day, one hour, one minute, etc, making it possessed of great praticability.

D. Comprehensive Analysis

In sum, with the low-level arrangement including physical layer deployment(the hardware tokens) and transport layer deployment(the certification policy between users or resources and MPS), LSSM realize secure and efficient certificate revocation not only from high level, but also uniquely and creatively from low level in network. It can avoid the attacks based on delay brought by CRL and denial-of-service attacks in OCSP. Also, it makes attackers who has breached SEM and possessed part of the private key unable to visit the grid services and impossible to compromise grid system. At the same time, our system has much better time accuracy than CRL and requires far less overhead than both of CRL and OCSP.

V. CONCLUSION

LSSM based on both of low-level and high-level arrangement is introduced in detail. LSSM can realize better security and faster revocation without bringing about too much cost. LSSM is an efficient and secure mechanism to update certificate and revoke invalid certificate in grid infrastructure. Up to now, our research on this system remains in theory. In future work, we will implement this system and do relevant experiments based on practical application to implement the method and improve the quality of the system.

VI. REFERENCES

[1] Erin Cody, Raj Sharman, Raghav H.Rao, Shambhu Upadhyaya. Security in grid computing: A review and synthesis. Decision Support Systems,2008-Elsevier.

[2] K. Hwang, Y. Chen, and H. Liu, “Protecting Network-Centric Computing System from Intrusive and Anomalous Attacks,” Proc. IEEE Workshop on Security in Systems and Networks (SSN’05), in conjunction with IPDPS 2005, April 8, 2005.

[3] M Naor, K Nissim. Certificate revocation and certificate update. IEEE Journal on Selected Areas in Communications, 2000.

[4] Patrick McDaniel, Aviel Rubin. A Response to "Can We Eliminate Certificate revocation Lists?". Financial Cryptography, 2001 - Springer.

[5] Paul C. Kocher. On certificate revocation and validation. Financial Cryptography, 1998 - Springer.

[6] X.509 Internet Public Key Infrastructure Online Certificate Status Protocol (OCSP), IETF RFC2560, http://www.ietf.org/rfc/rfc2560.txt.

[7] Silvio Micali. NOVOMODO: Scalable certificate validation and simplified PKI management. Proc. of 1st Annual PKI Research Workshop, 2002.

[8] Dan Boneh, Xuhua Ding, Gene Tsudik. Fine-grained Control of Security Capabilities. ACM Journal, February 2004.

[9] Shanhui Tan. Integrate HMAC Capable Token into User Authentication Mechanism and Public Key Infrastructure. SANS Institute InfoSec Reading Room, 2003.

[10] Jim Basney, Marty Humphrey, Von Welch. The MyProxy online credential repository. Software: Practice and Experience, 2005.