DISTRIBUTED THRESHOLD CRYPTOGRAPHY CERTIFICATION WITH NO TRUSTED DEALER

Apostolos P. Fournaris
Electrical and Computer Engineering Department, University of Patras, Rio Campus, Patras, Greece

Keywords: Threshold Cryptography, Elliptic Curve Cryptography, Distributed system, Certificate Authority.

Abstract: Threshold cryptography offers an elegant approach in evenly sharing certificate responsibilities to all participants of a distributed system through Shamir's secret sharing scheme, where a secret (the Certificate Authority's (CA) private key) is split and shared among all participants. However, existing threshold cryptography distributed key generation and certification systems still rely on a single, centralized, trusted entity at some point during the certification process (usually during initialization) to split the secret and distribute it to all distributed system participants. This centralized entity, denoted as trusted dealer, can cancel participant equality and can become a single point of failure. In this paper, we deal with this problem by extending the a key generation scheme of Noack and Spitz (2009) and by proposing a certification scheme that has no need for a trusted dealer to create, split and distribute the proposed certification scheme's private-public key pair. The proposed scheme uses the participant addition-removal procedure described in (Noack and Spitz, 2009) that does not affect the scheme's public key (used for certificate verification) and has small interference to the certification process as a whole. To reduce the computational cost the proposed system employs Elliptic Curve Cryptography (ECC) principles.

1 INTRODUCTION

CAs are widely used in network security following the client-server model. However, when using CAs for authenticity on distributed environments, like p2p, Ad-Hoc networks, and MANETs, the centralization introduced by the CA comes in contrast to the distributed nature of the network and constitutes a single point of failure.

To solve this issue, the idea of distributed certification schemes has risen. In such schemes, the role of a centralized entity is minimized and each participant of the scheme is assigned some task related to the certification services that are provided as a whole. Threshold cryptography for distributed CAs is based on the work of A. Shamir (1979), who proposed the concept of a *(t,n)* threshold scheme. In such approach, a methodology is developed for splitting a secret into n shares, so that, for a certain threshold $t < n$, any t components-parts of the secret can be combined to reconstitute the secret, whereas any combination of $t-1$ or less shares is incapable of reconstructing the secret. This idea, providing a way to save a secret in a distributed manner, is very attractive to systems where no centralized control is administered. However, Shamir's scheme, needs some sort of trusted entity for generating the secret value, splitting it into shares and distributing them to all the remaining participants. This entity, usually denoted as a trusted dealer, has enhanced responsibilities compared to the remaining system participants and most importantly it always needs to be trusted as well as protected because it has full knowledge of the secret.

Desmedt and Frankel (1989) as well as Frankel et al. (1997) were among the first to use the idea of Shamir's secret share to design threshold cryptosystems based on ElGamal. Pedersen (1991) attempted to avoid the need for a trusted dealer was made. His work was complemented by Shoup (2000) and was further supported by Damgård et al (2001) where the trusted dealer intervention was minimized. In the above schemes significant problems occur when new participants are added or removed to the system as pointed by Noack and Spitz (2009). So, Noack and Spitz (2009) proposed a discrete logarithm key distribution scheme with no trusted dealer that has a simple participant addition-removal mechanism.

In this paper, the work of Noack and Spitz

P. Fournaris A..
DISTRIBUTED THRESHOLD CRYPTOGRAPHY CERTIFICATION WITH NO TRUSTED DEALER.
DOI: 10.5220/0003525304000404
In *Proceedings of the International Conference on Security and Cryptography* (SECRYPT-2011), pages 400-404
ISBN: 978-989-8425-71-3

(2009) is extended and an ECC based distributed certificate authority is proposed that has a public key known to all participants but a private key known to none. Since there is no single CA entity or trusted dealer, the private key is always kept secret to all involved parties. The associated public key is used for certificate verification of any involved party and remains unchanged regardless of possible addition or removal of participants. As a result, the integrity of the proposed certification scheme is always retained while system compromise is very difficult as long as less than *t* participants are susceptible to secret information leakage.

The paper is organized as follows. In section 2 the proposed scheme is presented and analysed. In section 3, certificate management is described. In section 4, participant addition and removal is outlined. Finally, section 5 concludes the paper.

2 PROPOSED SCHEME

We assume that a group $\boldsymbol{U} = \{U^1, U^2, .. U^n\}$ of n participants U^i wish to establish a common Public Key $Pub = \{T, Q\}$ and a corresponding private key $priv$. To recover *priv*, at least *t+1* participants need to cooperate (threshold cryptography principle) where $t < n$. We also assume that all participants have agreed on a set of EC public parameters $T = \{p, a, b, G, q, h, H(x)\}$ defining an Elliptic curve E over a prime field F_p.with base point G: (x_G, y_G)

Initially, all involved participants generate local public-private key pairs (one for each participant) and agree on a global public private key pair (Pub – $priv$). This stage is denoted as key generation, establishment and distribution.

When all participants have contributed to the key generation, they need to obtain legitimate certificates for themselves. By proofing knowledge of their private information as well as proofing their contribution to the global key generation, the participants issue a certificate request in order to obtain a legitimate certificate of identity. When all participants obtain such certificates, the certification scheme reaches a stable state and is able to operate fully offering certificate issuing, certificate reissuing and certificate revocation. In a stable state, new participants can be added or removed without any additional overhead to the proposed scheme's functionality.

2.1 Key Generation, Establishment and Distribution

Initially, each participant U^i generates a local public-private key pair similar to the ElGamal Elliptic Curve scheme by choosing randomly a $pr_i \in F_p$ and computing $Pu_i = pr_i \cdot G \ \in E(F_p)$

This local key pair constitutes, in the initialization stage, U^i*'s* contribution to the master secret generation i.e. the global public-private key pair. Global keys can be produced using Shamir's secret sharing of creating a *(t,n)* threshold scheme:

1. Choose *t* random elements $\{s_1, s_2, \dots s_t\} \in F_p$
2. Construct a *t* degree secret polynomial
   $f_i(x) = s_t x^t + s_{t-1} x^{t-1} + \cdots + s_1 x \quad + s_0$
   where $s_0 = priv_{i,0} = pr_i$
3. Generate for all $U^j \in U$, $priv_{i,j} = f_i(j)$ where $j \in \{0,1, \dots t | j \neq i\}$
4. $Q_i = priv_{i,0} \cdot G = pr_i \cdot G = Pu_i$
5. Send to node U_j
   $\langle nonce_i, Q_i, Encr_{pu_j = Q_j}(priv_{i,j}, H(Q_i, priv_{i,j}, nonce_i)\rangle$

The above actions are performed for each participant of the system. After all messages are sent to all involved parties, each participant U^i calculates:

$$Q = \sum_{j=1}^{n} Q_j = \sum_{j=1}^{n} priv_{i,0} \cdot G = \sum_{j=1}^{n} pr_i \cdot G$$

$priv = \sum_{j=1}^{n} pr_i$ and $priv_i = \sum_{j=1}^{n} priv_{i,0} = \sum_{j=1}^{n} f_j(i)$

The point Q along with the EC parameters constitutes the global public key $Pub = \{T, Q\}$, the value $priv$ is the global private key while $priv_i$ is the private key share.

2.2 Proof of Knowledge Stage

After key generation completion, each participant knows the public key of the distributed CA, the private key share $priv_i$, a portion of the CA's private key ($priv_{i,0}$) and the corresponding partial public key Q_j. However, each participant has to obtain a certified identification from the CA through a certificate. Each participant has already chosen a local public-private key pair Pu_i, pr_i that is identical at this point with the partial key pair Q_i, $priv_{i,0}$. This pair Pu_i, pr_i can be used in identification as long as it's certified by the distributed CA. For simplicity reasons, we will denote the local public-private key pair as Q_i, $priv_{i,0}$ and discuss the possible difference between local key pair and partial key pair further in section 4. To avoid impersonation attacks, the CA must verify that the requesting participant is the true holder of the local private key and, most

importantly, that the participant was actively involved in the global public-private key generation. We propose a proof of knowledge protocol to verify this information. When Participant U^i (denoted as *prover*) issues a certification request, it initially performs the following two independent operations:

Local private key proof of Knowledge

1. Choose two random numbers r, $nonce_1 \in F_p$
2. Calculate $R = r \cdot G$
3. Use Hash Function *H(x)* and calculate $c = H(nonce_1|Q_i|R)$
4. Calculate $s = c \cdot priv_{i,0} + r$
5. Broadcast to all other participants
$$P = [nonce_1, s, Q_i, R]$$

Key share proof of Knowledge

1. Choose a random numbers r_j for at least *t+1* different participants U^j and a $nonce_2 \in F_p$
2. Choose a random point $V \in E(F_p)$
3. Calculate $R_j = r_j \cdot V$ for at least *t+1* different participants U^j
4. Use Hash Function *H(x)* and calculate $c_j = H(nonce_2|V|R_j)$ for all j
5. Calculate $s_j = c_j \cdot f_j(i) + r_j$
6. Transmit to at least *t+1* random participants U^j, a corresponding value $P_j = [nonce_2, s_j, V, R_j]$

The participants (denoted as *verifiers*) that are involved in the certification process of *prover* U^i, form the group U_{sign} and receive two messages, broadcasted message P that is common to all verifiers and message P_j that is unique for each *verifier* U^j. The proof of knowledge by the verifiers is achieved by performing the following operations:

Local private key proof of Knowledge

1. Receive message $P = [nonce_1, s, Q_i, R]$
2. Verify freshness of the $nonce_1$ value
3. Use Hash Function *H(x)* and calculate $\acute{c} = H(nonce_1|Q_i|R)$
4. Check the validity of the following equation: $s \cdot G = \acute{c} \cdot Q_i + R$

Key share proof of Knowledge

1. Receive message $P_j = [nonce_2, s_j, V, R_j]$
2. Verify freshness of the $nonce_2$ value
3. calculate $\acute{c}_j = H(nonce_2|V|R_j)$
4. Use secret polynomial $f_j(x)$ to calculate the value $\acute{f}_j(i)$
5. Check the validity of the following equation: $s_j \cdot V = \acute{c}_j \cdot \acute{f}_j(i) \cdot V + R_j$

If both the above validation tests produce true answers then each verifier U^j is persuaded that *prover* U^i has knowledge of its local private key $priv_{i,0}$ and knowledge of $priv_{i,j}$, that the verifier provided to the *prover* during key generation. If at least *k* verifiers $j,\ 1 \le j \le n,\ j \ne i$ provided with a message P_j from *prover i,* are persuaded that *i* has knowledge of the associated $priv_{i,j}$ value then participant *i* has knowledge of the key share $priv_i$.

2.3 Certificate Generation Stage

Successful participant's proof of knowledge described in section 2.2, triggers the certificate generation mechanism of the distributed certification scheme. Goal of this mechanism is to provide each U^i with a verifiable certificate of its local public key as well as to assign a legitimate, unique identity through an ID number and ID attributes.

At the initialization phase of the certification mechanism, each participant U^j, $1 \le j \le n,\ j \ne i$, acting as *verifier*, attains the role of *signer* and uses the already acquired knowledge on the certification candidate participant U^i to produce a certificate share containing a digitally signed Q_i and identification share ID^j. The identification share ID^j includes the ID number (ID^j_{no}) and a set of *w* Identification attributes AT^j_m, where $1 \le m \le w$. More specifically, each *signer* participant U^j performs the following actions:

1. Choose a random number $k_j \in F_p$
2. Calculate $Y^1_j = k_j \cdot G$
3. Generate $ID^j_{no} = \mathrm{H}(Q_i) \cdot k_j + priv_j \cdot x_j$ where $x_j = \prod_{t \in U_{sign}, t \ne j} \left(\frac{-t}{t-j}\right)$
4. Generate a set of ID attributes $AT^j = \{AT^j_m \in F_p | 1 \le m \le w\}$
5. Calculate $s_j = H\left(\sum_{m=1}^{w} AT^j_m\right) \cdot k_j - priv_{j,0}$
6. Send to Participant U^i $M_j = \left[ID^j_{no}, AT^j, [Y^1_j, s_j]\right]$ where $[Y^1_j, s_j]$ is the certificate share for *signer* U^j

Upon receipt of the certificate shares of every signer on the group U_{sign}, the certificate requesting participant U^i verifies the shares and generates the certificate. Determination of the Identification Attributes is achieved by performing a proposed operation called *random consultant advice*. During this operation, the certificate requesting participant chooses randomly a *signer* participant as its *consultant* U^{cons} and adopts this *consultant's* sent Identification attribute vector AT^{cons}

1. Choose a random number $r_i \in F_p$
2. Calculate $Y^2 = r_i \cdot G$
3. Calculate $Y^1 = \sum_{j \in U_{sign}} Y^1_j = \sum_{j \in U_{sign}} k_j \cdot G$
4. Calculate ID number $ID_{no} = \sum_{j \in U_{sign}} ID^j_{no}$
5. *Random Consultant Advice*:
   a. *Generate vector*
$$V = \left[V_j = Y^1 + Q_j \middle| for\ all\ j \in U_{sign}\right]$$

b. *Generate vector*
$$C = \left[C_j = H(V_j) \,\middle|\, for\ all\ j \in U_{sign}\right]$$
c. $signC = (HashChain(C)) \cdot \mathrm{r}_i + priv_{i,0}$
d. Find $C_{cons} = min(C)$ and choose associated U^j as U_{cons}, (*min(C)* is the minimum value of *C)*
e. Verify $s_j \cdot G = H\left(\sum_{m=1}^{w} AT_m^j\right) \cdot Y_j^1 - Q_j$ for $U^j = U_{cons}$ *(j=cons)*
f. If step 5e is valid then assign $AT = \left[AT_m^j \middle| for\ all\ 1 \le m \le w, j = cons\right]$
g. Generate Signing Vector: $SignQ = \left[Q_j \middle| for\ all\ j \in U_{sign}\right]$

6. Publish Certificate as:
$$Cert_i = \begin{Bmatrix} ID_{no}, Q_i, AT, SignQ, \\ \{Y^1, Y^2, s, signC\}, \{Q_{cons}, Y_{cons}^1, s_{cons}\} \end{Bmatrix}$$

The operation *HashChain( )* provides a digest of a value or a series of values as well as integrity measure on the sequence series of values. In our case, we define *HashChain( )* as an iterative process where $H_i = H(C_{i-1} + H_{i-1})$, $H_1 = H(C_1)$ and $HashChain(C) = H_{t+1}$ for all *i* in $U^i \in U_{sign}$.

The above operations are repeated for every participant in the group U. Upon completion, each participant of the proposed scheme has a legitimate signed certificate of its characteristics $(ID_{no}, Q_i, AT, SignQ)$. At this point the scheme has reached a stable state.

The verification of the participant's ID characteristic using its certificate can be achieved by performing the following operations:

1. Verify Q_i and ID_{no} by checking the validity of equation $ID_{no} \cdot G = H(Q_i) \cdot Y^1 + Q$
2. Verify choice of random *consultant*:
   a. From vector *SignQ* generate vector
   $$\acute{V} = \left\{\acute{V}_j = Y^1 + Q_j \,\middle|\, for\ all\ j \in U_{sign}\right\}$$
   b. *Generate vector*
   $$\acute{C} = \left\{\acute{C}_j = H\left(\acute{V}_j\right) \,\middle|\, for\ all\ j \in U_{sign}\right\}$$
   c. Verify equation $signC \cdot G = HashChain(\acute{C}) \cdot Y^2 + Q_i$
   d. Find $\acute{C}_{cons} = min(\acute{C})$ and verify if associated Q_j is equal to Q_{cons}
3. Verify AT by checking the validity of equation $s_{cons} \cdot G = H(\sum_{m=1}^{w} AT_m^{cons}) \cdot Y_{cons}^1 - Q_{cons}$

The *random consultant advice* verification main goal is to prove randomness of participant's U_i choice of consultant. Retracing the steps of this choice by any certificate verifier should always lead to the same participant and this participant should be the participant U_{cons} included in the certificate. The distributed generation of the EC point Y^1 added to the public key of each participant in U_{sign} can guarantee randomness of each Hash Function outcome. As a result, the minimum value of the Hashing results cannot be retraced (due to one way function property) and is different in each issued certificate resulting to different consultant advice.

3 CERTIFICATE MANAGEMENT

When a legitimate, fully functional, certificate reaches its validity end, the certificate owner must request a certificate reissuing or update before this validity time limit is reached. During reissuing, the requesting participant retains his local public - private key pair and his ID_{no} as well as some of his Identification attributes depending on the system at hand. In the proposed scheme certificate reissuing is done by broadcasting a reissue request from a Participant accompanied by his existing certificate to a signer set $\acute{U}_{sign}$. Each Participant U^j of $\acute{U}_{sign}$ verify the existing certificate and generate a new ID Attribute set $\acute{A}T^j$, digitally signs it by performing $\acute{s}_j = H\left(\sum_{m=1}^{w} \acute{A}T_m^j\right) \cdot \acute{k}_j - priv_{j,0}$ using a random number $\acute{k}_j$ and transmits to the requesting participant the new $\left\{\acute{A}T^j, \acute{s}_j, \acute{Y}_j^1\right\}$ where $\acute{Y}_j^1 = \acute{k}_j \cdot G$. The requesting participant performs the *Random Consultant Advice* operation using as random point $Y_{rev}^1 = \sum_{j \in U_{sign}} \acute{Y}_j^1$ and publishes the new certificate

$$Cert_i = \begin{Bmatrix} ID_{no}, Q_i, \acute{A}T, S\acute{\imath}gnQ, \\ \{Y^1, Y_{rev}^1, Y^2, s, s\acute{\imath}gnC\}, \{\acute{Q}_{cons}, \acute{Y}_{cons}^1, \acute{s}_{cons}\} \end{Bmatrix}$$

where $AT, \left\{\acute{Q}_{cons}, \acute{Y}_{cons}^1, \acute{s}_{cons}\right\}$ are the *Random Consultant Advice* operation results. Note, that the reissued certificate includes the value Y_{rev}^1, indicating certificate reissuing. In future reissuing, Y_{rev}^1 value is compared to the original Y^1 value. For valid reissued certificates, $Y_{rev}^1 \neq Y^1$.

The certificate reissuing process occurring after a successful participant addition or removal is different and is further discussed in subsection 4.1.

Certificate revocation must be performed in a timely and efficient manner and cannot be overlooked during certificate verification. Common distributed revocation schemes are based on participant voting on the credibility of a certificate (Crépeau, C. & Davis, C.R., 2003), (Arboit, G. et al., 2008) (Kyul Park et al., 2010). Each participant can place an accusation vote for another participant's certificate. If the number of participants accusing a particular entity exceeds a predefined threshold R_T, the accused participant's certificate is revoked.

In the proposed scheme, based on threshold cryptography, the use of Participant revocation voting can be successfully included, since the threshold mechanism infrastructure is already

present. We suggest using the scheme described by Arboit, G. et al., (2008) where the certificate revocation scheme requires that participants of the system monitor the behavior of the other nodes.

4 PARTICIPANT ADDITION-REMOVAL

One of the important benefits of the proposed certification scheme is its ability to easily add and remove Participants in the group *U*. To achieve that, we adopt the participant addition-deletion mechanism proposed by Noack and Spitz (2009). We assume that the certification scheme has been already established, that every participant has his local public-private key pair, his partial public key pair as well as his legitimate certificate and that he has contributed successfully to the generation of the global public-private key pair of the distributed CA.

We employ the share renewal technique of Noack and Spitz (2009), based on the PSS scheme of Herzberg, A. et al., (1995). PSS updates already distributed shares of all n members to provide proactive security. While adding a participant, $t+1$ members of *U,* forming a subset U_{splt}, split off a part of their secret and share this part with the new member. Removing a participant is done by computing and redistributing the participant's secret to some remaining *U* members.

4.1 Addition-Removal Certificate Management

Addition - Removal of Participants has no effect in the global public key of the distributed certification scheme. So, certificates remain valid even after the partial public-private key pairs change values and can still be issued or verified. This happens due to the fact that the local public private key pairs that handle secure communication between participants are different in principle from the partial public – private key pairs. Local key pairs are only changing in a certificate reissuing operation after a participant addition –removal. In that case, the reissuing certificate procedure is different than the one described in 3. In certificate reissuing after participant addition –removal, the partial key pair replaces the local key pair of a requesting participant and subsection 2.2, 2.3 processes are executed.

5 CONCLUSIONS

In this paper, an ECC based fully distributed Threshold cryptography certification scheme was proposed that eliminates the need for trusted dealer for secret sharing. The certification infrastructure is self-organized and fully decentralized. Also, using the mechanism described by Noack and Spitz (2009), addition and removal of participants is achieved while keeping the global public key unchanged..

ACKNOWLEDGEMENTS

The reported work is supported through the SECRICOM FP7 FP7 SEC 218123European project

REFERENCES

Shamir, A. auth., 1979. How to share a secret. *Communications of the ACM, 22, σ.612–613.*

Desmedt, Y& Frankel Y., 1989: *Threshold Cryptosystems. CRYPTO 1989:307-315*

Frankel, Y. et al. auth., 1997. Optimal-resilience proactive public-key cryptosystems. *Proceedings of the 38th Annual Symposium on Foundations of Computer Science, pp.384–.*

Pedersen, 1991: A Threshold cryptosystem without a trusted third party, *proc. of EuroCrypt 91, Springer Verlag LNCS nr. 547.*

Shoup, V. auth., Practical threshold signatures. , 1807 pages, pp.207-220.

Damgård, I. & Koprowski, M. auth., 2000. Practical Threshold RSA Signatures *Without a Trusted Dealer. , pp.152-165.*

Andreas Noack, Stefan Spitz, 2009, Dynamic Threshold Cryptosystem without Group Manager *Network Protocols and Algorithms 1: 1. pp 108-121*

Crépeau, C. & Davis, C. R. auth., 2003. A certificate revocation scheme for wireless ad hoc networks. in Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks. *SASN '03. New York, NY, USA: ACM, pp 54–61.*

Arboit, G. et al. auth., 2008. A localized certificate revocation scheme for mobile ad hoc networks. *Ad Hoc Networks, 6, pp.17–31.*

Kyul Park et al. auth., 2010. Certificate Revocation to Cope with False Accusations in Mobile Ad Hoc Networks. in Vehicular Technology Conference (VTC 2010-Spring), *2010 IEEE 71st. Vehicular Technology Conference (VTC 2010-Spring),. pp 1-5.*

Herzberg, A. et al. et al., 1995. *Proactive Secret Sharing Or: How to Cope With Perpetual Leakage. Lecture Notes in Computer Science, 963:339, pp.339--352.*