Decentralization of the current PKI infrastructure without losing backward compatibility

Vesselin Tzvetkov
Arcor AG&Co KG
Alfred-Herrhausen-Alle 1, 65817 Eschborn, Germany
vesselin.tzvetkov@arcor.net

***Abstract-* The Public Key Infrastructure (PKI) is an important part of almost all security implementations, from secure portals for banks and e-shops to vpn devices. Although it has advantages, there is a critical design issue due the single point of failure of the root (CA) certificate. This issue is solved by decentralization of the infrastructure. Since a lot of the PKI infrastructure is already active, implementing such a solution means rebuilding the entire system from scratch. In this paper we introduce this problem and propose an industry and user friendly solution without loosing the already built PKI infrastructure. The stress of the proposed solution is on backward compatibility to the current PKI so that the smooth migration of the clients is achieved. Here we present x509v3 extensions and define the policy algorithms. Our target is to achieve an efficient solution with minimum changes in the current standards.**

***Index Terms-* Public Key Infrastructure, x509v3, Accredited CA, Certification Authority**

I. Introduction

The PKI is an evolutional step in cryptography due to asymmetric encryption algorithms such as DSA/RSA and hierarchical scalability. The use of a Digital Signature is a reliable and reasonable alternative to the classical hand signature. This is reflected in the legislation of many countries, that principally Digital Signature is as authentic as a classical signature. This supports its wide implementation in official structures for passports, personal IDs, bank cards, etc. At the beginning of the millennium it was predicted, that the PKI would be implemented rapidly in all these personal IDs by official institutions. However, six years later, there are only a few examples worldwide of implementing a PKI structure in official government structures.

An Achilles' heel or the crux of the PKI is its extremely strong hierarchy, which results in blind trust of the credibility and authenticity of the CA. When the CA key is compromised, the whole PKI structure collapses and there is no standard PKI way to recover from this disaster. The structure should be initialized from the very beginning - all issued certificates should be re-issued. For millions of issued certificates it could be an enormous challenge with unpredictable costs.

This horror scenario, when the CA key is stolen or destroyed, cannot be treated and principally solved by the current concept of PKI. The users trust only the CA regarding user certificates and only this CA is responsible for all issued certificates. It is even a difficult task to determine, if the key is destroyed or stolen, what is conclusive for further actions.

A distribution of the root responsibility to multiple equitable Authorities (multiple CAs with the same administrative rights) resolves the single point of failure problem. The major difficulty in protecting the investments and the existing infrastructure is the migration from the current infrastructure. In our view, the only practical solution must be a smooth migration in uninterrupted service with backward compatibility. In the following text we are suggest the needed enhancements and procedures to achieve this.

The use of the current PKI infrastructure with a few RFC conform extensions is the target for achieving decentralization and in this way resistance against root key single point of failure. A backward compatibility, so that the already issued x509v3 certificates can be further used, is the main goal of this solution.

II. Overview of the existing technologies and related mechanisms

A. *Current protection of the root CA*

The major CAs secure their root keys using multiple physical systems protected by statefull inspection firewalls and multiple access mechanisms. Furthermore they implement high physical security by using safe and protected areas with minimum access and utilization. This certainly increases the security, but there is still a single point of failure. Even highly protected areas are accessed by staff and can be physically attacked. Backup copies of keys also enhance the risk.

B. *Threshold cryptography*

In many companies, banks for example, one document should be signed not only by one person, but by a group of people. The document is valid only when it is signed by this defined group of signers. Encryption methods dealing with group signing and group authentication models are called threshold cryptography [1]. Using threshold cryptography does not change the administrative PKI structure and the risks from it.

C. *Cross Certification and Certificate Trust List*

Cross Certification [6] and Certificate Trust List (CTL) are standards defining the possibility for integration of different

PKI structures without changing the issued certificates and trust authority. In Cross-Certification the CA issues a cross certificate to the partner authority. Cross certificates can include the same private and public key as the partner self-signed certificate. The user checks online if there is a cross certificate for the received certificate. In this way, without changing his trust CA, the user can deal with different PKI's certificates.

Generally Cross Certification and CTL do not change the responsibility domains of the CA – only the certificate issuer is responsible for revocation of his certificates and the CA root certificate cannot be revoked by any partner.

D. *Decentralization Methods in Ad-Hoc networks based on Threshold Cryptography*

Solutions for distributing the CA responsibility were proposed based on threshold cryptography [1], since the problem was treated in the Ad-Hoc network [3]. These solutions assume the building of new PKI structure with new CA servers and clients. This is practically almost not implementable.

E. *Models based on forward-secure signature scheme (FSS)*

Some proposals make use of a FSS algorithm [4] which also requires new PKI clients and CA servers thus complicating the practical implementation and rebuilding a new structure.

F. *Quorum System*

In the distributed computing the quorum system are wide spread [7][8]. The idea is maintaining an information piece in distributed storages with multiple copies. The operations "read" and "write" are performed on quorum group of servers. The quorum systems increase the efficiency and availability of the replicated data. The information in the distributed system is consistent to some practical degrees, sufficient for the application. Through the quorum principle this degree of consistency is achieved, which is tradeoff the efficiency. These works on quorum systems are possible enhancements of the here proposed model. At first line we suggest x509v3 extensions, which make possible working with quorum principles at all. The suggested policy is simple quorum, but for sure other quorum principles can be implemented.

III. QUORUM PRINCIPLE AND MIGRATION TO QUORUM PRINCIPLE

In order to verify administratively a certificate the users download the CRLs from different accredited Certification Authorities. If a quorum of the CAs acknowledges the validity of the certificate thus it is accepted. When a minority (not quorum) of the CA is not physically accessible or distributes malicious information, then their information is not considered. The user information for certificates revocation is correct until a quorum of the CAs distribute valid information.

The legacy PKI clients canot interpret the new extensions, so they proceed as described in the current PKI. They access the CRL only from the root CA and verify the certificate. In this case only the issuer root CA is responsible for the issued certificates.

The existing PKI structure is smoothly upgraded in the following steps: first the accredited CAs are installed and activated. Second the root CA is configured to issue extensions *AccreditedCA* to all new certificates and interact with the accredited CAs. Third the clients are exchanged, so the new installation can access the new *accreditedCAs*. After these steps the legacy and new clients are coexisting. There is service interruption only during CA server and clients upgrade.

IV. TRUST RELATION ESTABLISHMENT AND ROLES

The first step should be to set a relation between an odd number of authorities. There should be an Issuer CA and an even number of Accredited CAs. Using odd number prevents of running in no quorum situation, where no strict reliable decision can be met. All authorities should regularly authenticate each other in authentication periods. The authentication should be based on a not-PKI method; for example PINs, passwords, biometrics, etc. If there is a group of n authorities, where n = {2k+1}, k non-negative integer are n.(n-1)/2 authentications per authentication period. The number of authentications increase with the square of n and correspond to the complexity. To reduce this complexity, it is recommended to use a reasonable number of authorities.

The Issuer CA generates the x509 certificates and should support CRLv2 (Certificate Revocation Lists) or OCSP (Online Sertificate Status Protcol) as described in the current PKI standard [2].

The Accredited CAs generate CRL [2], where certificates from the Issuer CA and itself can be revoked. The Accredited CA can also implement an OCSP protocol.

V. CERTIFICATE X509V3 EXTENSION

When an Issuer CA generates a new certificate, it should include the extension *accreditedCA* defined here. This extension identifies the Accredited CA and also contains the signed user's public key. The extension includes multiple ANSI sequences in the certificate – one *AccreditedAuthKeyId* for every Accredited CA. For the notation description use [2].

```
AccreditedAuthorities ::= {

        AccreditedAuthKeyIds }

AccreditedAuthKeyIds::= SEQUENCE SIZE (1..MAX) OF
                              AccreditedAuthKeyId

AccreditedAuthKeyId = SEQUENCE {

        keyIdentifier[0]              KeyIdentifier       OPT
        authorityCertIssuer[1]        GeneralNames        OPT
        authCertSerialNumber[2]       CertSerialNumber    OPT
        SignedUserKeyIden[3]          KeyIdentifier
}
```

```
KeyIdentifier ::=           OCTET STRING
GeneralNames::=             SEQUENCE SIZE (1..MAX) OF
                            GeneralNames
CertSerialNumber ::=         INTEGER
```

The extension is built very similarly to *authorityKeyIdentifier* with additional the signed user's key identifier attribute. The user key identifier attribute is signed by Accredited CA and shows that the certificate is trusted according to the Accredited CA. The AccreditedAuthKeyId sequence is repeated for every Accredited CA.

The certificate owner should have installed and trust not only the certificate of Issuer CA, but also the certificate of the Accredited CA. Using the attributes authorityCertIssuer, authCertSerialNumber and keyIdentifier from *AccreditedAuthKeyId* the right certificate can be found and the an the signature of the SignedUserKeyIden be verified.

VI. Certificate revocation list Extension

At each authentication period, all authorities issue new CRLs with which they can revoke the Issuer/Accredited CA and user certificates. To access the CRL, the clients use the information in the "CRL Distribution Points" extension. In this extension, the CRLs must be listed by Issuer CA and Accredited CA. In this way the client can download all issued CRLs. It is important to stress that the client must download all CRLs listed in the extensions and not only the first one in the list. To keep compatibility to legacy PKI infrastructure in the extension CRL Distribution Point the Issuer CA attribute should be at first place. The reason is, that current PKI clients download only the first possible CRL. It should be assured the first downloaded is to be form the Issuer CRL.

Since the Accredited CAs are not issuers of the certificates, they issue indirect CRLs [2]. Further more the CRLv2 may include the *RevokedAccreditedCA* extension, which revokes an Accredited CAs. For clarity, the definition of CRL as in [2] is listed:

```
CertificateList ::= SEQUENCE {
        TbsCertList                 TBSCertList,
        SignatureAlgorithm          AlgorithmIdentifier,
        SignatureValue              BIT STRING }

TBSCertList ::= SEQUENCE {
        version                     Version           OPT,
        signature                   AlgorithmIdentifier,
        issuer                      Name,
        thisUpdate                  Time,
        nextUpdate                  Time              OPT,

 revokedCertificates SEQUENCE OF SEQUENCE {

        userCertificate             CertificateSerialNumber,
        revocationDate              Time,
        crlEntryExtensions          Extensions        OPT
} OPTIONAL,

 crlExtensions [0]  Extensions OPTIONAL
```

Generally, only certificates from one CA can be revoked in one CRL. In our case, the Accredited CA have probably issued a self signed certificate, so we define a new extension *RevokedAccreditedCA* allowing the revoking of certificates issued by a multiple CA in the same CRL.

The *RevokedAccreditedCA* identifies the Accredited CA using the subject name or KeyId as in *authorityKeyIdentifier* [2] extension, it is defined as:

```
RevokedAccreditedCA ::= authorityKeyIdentifier
```

To identify the origin of a revoked certificate the following rule is applied: if there is not *RevokedAccreditedCA* extension under *crlEntryExtensions*, the revoked certificate is Issuer by Issuer CA. If there is a *RevokedAccreditedCA* extension, the extension attributes determine which Accredited CA is revoked.

Keeping in mind that the backward compatibility must be considered, the legacy clients will ignore this extension. This can cause revoking certificates from an incorrect issuer if there is an overlapping certificate number with the Accredited CA certificate number. We suggest the reservation of serial numbers of Accredited CA certificates in the Issuer CA infrastructure.

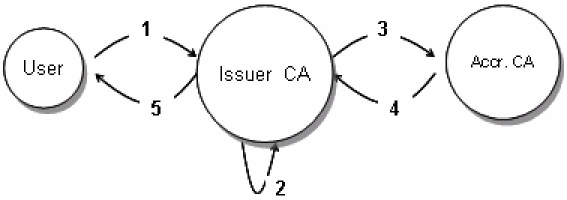
VII. Handling Behavior and Policy in Major Trust.

The major change in using Accredited CA compared to the current PKI standard is not the x509 extensions but the processes and behaviour handling certificates. These are described in policies.

A. Issuing the user certificate

The user generates a certificate request and provides it to the Issuer CA, which authenticates the user with credentials such as password, pin, etc.(Step 1 at figure 1). The Issuer CA adds the request to the issuing queue (Step 2 at figure 1). The queue is sent regularly to the Accredited CAs (Step 3 at figure 1). If the Issuer CA and Accredited CA successfully authenticate themselves, the Accredited CA singes a hash of the user's public key and returns it to Issuer CA as a certificate attribute **SignedUserKeyIden.**´(Step 4 at figure 1). If the authentication fails, a zero attribute **SignedUserKeyIden** (value null) is used. The attribute **SignedUserKeyIden** is set in the extension **AccreditedAuthKeyId** and the user certificate is issued and distributed to the user (Step 5 at figure 1).

Figure 1 Issuing process



Steps 3 and 4 are repeated for every Accredited CA and one *AccreditedAuthKeyId* extension is added to the user certificated for every Accredited CA. Between step 1 and 5 the user request is in status pending and the user should wait and regularly check if the certificate is issued.

B. Accredited CA Policy

The main task of the Accredited CA is to guarantee the trust of the Issuer CA. The number of Accredited CAs is p, *where* $p = \{2k+1\}, k \in N$.

Every Accredited CAs issue its own CRL - which certificates are revoked and which are valid. This information is independent and cannot be influenced by other CAs.

At each authentication period, all entities authenticate each other and if an authentication fails then the entity is revoked and listed in the CRL using the *RevokedAccreditedCA* extension or by a serial number for the Issuer CA. The authentication fails, when the non-PKI authentication using PIN, TAN etc. has been done (see IV). There is no reverse option after revocation. In figure 2, the policy for CRL issuing is presented.

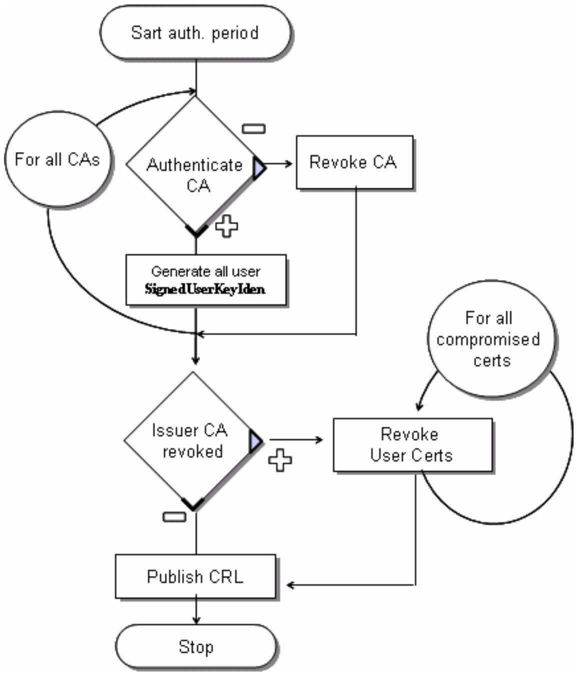


Figure 2 Accredited Policy

The Accredited CA can revoke user certificates, but only if the Issuer CA has been already revoked. The information for user revocation should be provided to every Accredited CA separately in non-PKI way.

C. Issuer CA policy

The CA operates regarding [2] for user authentication, certificate distribution, publishing, CRL etc. The Issuer CA should authenticate every authentication period of all Accredited CA and revoke them, if necessary, in the CRL as described above. The policy for this part is the same as that for Accredited CAs in figure 2.

D. User Policy

In the user policy, the Majority Trust principle determines if a certificate is revoked or not. The user compares the different CA's views, expressed in CRL, and decides based on the quorum principle.

In the current PKI implementation, the user downloads only the first possible CRL. In the new policy, the user must download the CRLs from all Accredited CAs in addition to the Issuer CA. The quorum principle: if one CA (Accredited or Issuer) is revoked in more then $n/2$ CRLs, the CA is untrusted-revoked. The CRLs from revoked CAs are no longer considered. The last trusted CRL from one revoked CA is called the *Testament* CRL. If there are m untrustworthy CAs, there are $n-m$ actual CRLs and m *Testament* CRLs.

If the Issuer CA is revoked from a quorum of CAs, the PKI is in state *Capsulated* – no more new certificates are accepted. All user certificates issued after the revoking date (Capsulation Date) of Issuer CA are untrusted, since they will all have zero value **SignedUserKeyIden** in all **AccreditedAuthKeyId** extentions.

By verifying a certificate, the user checks that there are more than $(n-1)/2$ non zero valued **AccreditedAuthKeyId** extensions from a trusted Accredited CA .

If the PKI is not in the *Capsulated* state, the user certificates are revoked only from Issuer CA's CRL. If the PKI is in the Capsulated state then the user certificates can be revoked in the last Testament CRL of the Issuer CA or only by a quorum of more than $(n-1)/2$ Accredited CAs.

The following figure 3 shows the decision algorithm if the user certificate is trusted. The legacy client, not implementing this policy, will download the CRL only from the Issuer CA and keep working as before.

VIII. DISASTER SCENARIOS

A. Compromising of Issuer CA

Let us consider that the Issuer CA is compromised. In the next authentication period the CA can not authenticate to the Accredited CAs. The Accredited CAs will revoke the Issuer CA by adding it in their CRL. After the user refreshes his CRLs he will set the status of the PKI as Capsulated and will not accept any certificate issued after this capsulation date.

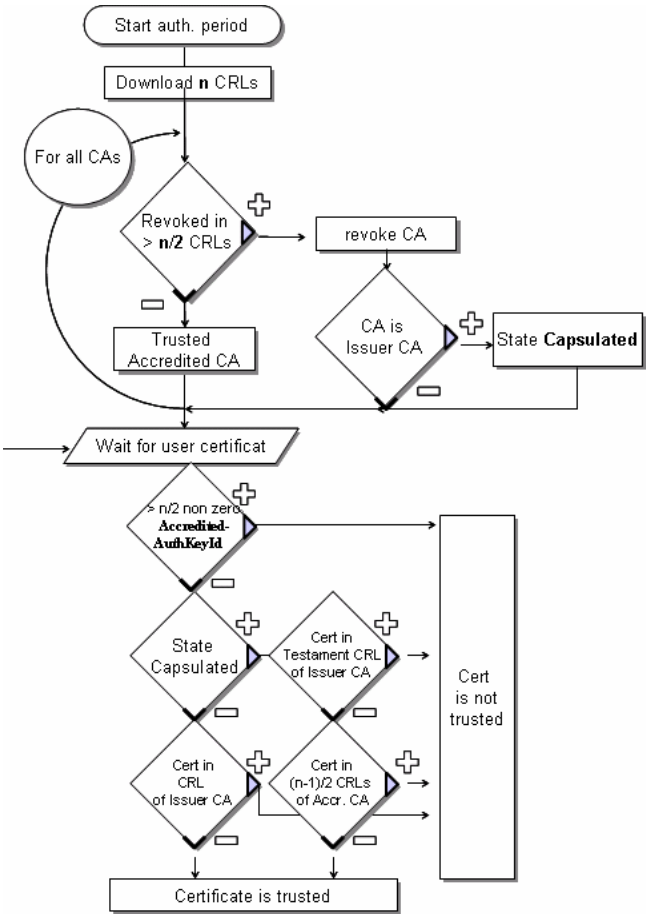


Figure 3 User policy

When the system is in status capsulated state, user certificates can be revoked only by a quorum of more than *(n-1)/2* Accredited CA or in the Testament CRLs of a Issuer CA.

When the Issuer CA is compromised, the PKI structure operates stable until a majority of (n-1)/2 authorities are trusted.

B. Compromising of an Accredited CA

Like in the previous case, when an Accredited CA is compromised, in the following authentication period it will be revoked in CA's CRL. After downloading the new CRLs the user will revoke the Accredited CA and it will not consider any further information from this CA.

C. Revoking user certificate

User certificate is revoked only be the Issuer CA in its CRL as in the classical PKI, when the Issuer CA is not compromised. If the Issuer CA is compromised then the user certificate must be revoked in CRL of the AccreditedCAs.

IX. Further properties of quorum PKI

The Majority Trust PKI will collapse when more than *(n-1)/2* CAs are compromised or physically on reachable. The probability of losing authenticity and credibility in this decentralised model is significantly better then in the current PK infrastructure. Clear disadvantage of the model is the increasing complexity of the system because of the involvement of more authorities.

The authentication period is the arbitrative variable for the reaction time of changes for the system. Smaller values decrease the reaction time of the system, which is the time to revoke compromised CA certificates. Unfortunately, smaller authentication periods increase the cost of the PKI and a reasonable value should be used, since every authentication period there are n(n-1)/2 authentications

Instead of CRL, the users can use OCSP to determine if a certificate is revoked, which will help for rapid certificate validation. CRLs and OCSP can be used together, as in the most current PKI, depending on the authentication importance.

X. Conclusion

Compromise of CA root key was not treated deeply enough in the current PKI, which makes the current PKI unattractive for some official structures. Here, we proposed enhancements to the current PKI for improved resistance against CA's root key attacks. The technology is cryptographically simple and does not include new algorithms; it is more policy and organization enhancement than format change of the x509v3 certificate.

The quorum solution is designed to be friendly and compatible to the current PKI structure, which is a very important part for the practical implementation. Compatibility is arbitrage characteristic for the industry, because all current PKI nodes can not be simultaneously changed without loosing functionality. Before every practical migration of existing infrastructures, the current client's and server's policy must be deeply studied and verified, because we are considering the RFC behavior and not some vendor specific differences.

References

[1] "Some Recent Research Aspects of Threshold Cryptography." In E. Okamoto, G. Davida at. al Springer-Verlag, 1997.

[2] "Internet x.509 Public Key Infrastructure. Certificate and CRL (Profile)" Housley,et.al., RFC 3280, April 2002

[3] "Key Management for Heterogeneous Ad Hoc Wireless Networks", Seung Yi Robin Kravets, 2002

[4] "Decentralization Methods of Certification Authority Using the Digital Signature Schemes", S.Koga, K.Sakurai 2003

[5] "How to Share a Secret". A. Shamir. ACM, 1979.

[6] "Cross-Certification and PKI Policy Networking", J. Turnbull, 2000

[7] "Weighted Voting for Replicated Data", David K. Gifford, ACM, 1979

[8] "Probabilistic Quorum Systems", Dahlia Malkhi, Michael K. Reiter, Avishai Wool, Rebecca N. Wright