

The Certificate Revocation Framework

O. Kessler, Open Systems AG
<http://www.open.ch>



March 2000

Abstract

Although technologies for Public Key Infrastructures (PKI) are well developed nowadays and almost ready to be used, some issues are unsolved so far. The lack of a mechanism that provides scalable certificate revocation is a fundamental problem inhibiting the wide acceptance of a PKI in the Internet.

This paper will compare some existing solutions and propose a novel mechanism called *Certificate Revocation Framework (CRF)*. The CRF provides an infrastructure capable of delivering certificate revocation information almost immediately to a large community of end entities using cryptography applications. The Client Access Points (CAP) serving certificate status information to end entities provide the interface to gain information about any certificate whose issuing Certificate Authority (CA) is integrated in the CRF. The internal structure guarantees for freshness and correctness while being completely scalable to follow the growth of a PKI.

1 Introduction

Certificate validation is a basic PKI operation and its importance has long been underestimated. An entity using certificates needs to be sure that a certificate's data is indeed correct and thus must be granted access to certificate status information. Certificate revocation is an unexpected invalidity of the certificate and thus all potential users must be informed as soon as possible to prevent further usage of the certificate in question.

Seen from an operational standpoint, we must provide a scheme to transport revocation information (a) from the Certificate Authority (CA) to the repository or to various repositories and (b) from the repository or the Verification Authority (VA) to the end entity. This will split up our problem, as there are different needs for these two paths which in turn request different solutions.

In this paper, we will focus on the task of distributing once created revocation information and not on the interfaces provided to the user for actually fulfilling the task of revocation. This may be solved elsewhere, either as a CA administrator's task or as a sort of online service provided to the user to issue a revocation request.

1.1 The Importance of Certificate Revocation

As early as in the 80's already, a study by the MITRE corporation¹ in order of the NIST² (National Institute of Standards and Technology) has assisted Certificate Revocation to have the potential of being the most costly aspect of running a large scale PKI [23].

This has already proven to be true in terms of the delay that was introduced so far in implementing PKIs around the globe by the lack of a reasonable method handling certificate revocation. It seems clear nowadays that this is the most vexing task in a PKI environment.

2 Requirements for a Solution

Before comparing existing solutions and proposals, let's assemble the design parameters a certificate revocation scheme has to conform to.

The main task of a scheme solving the revocation problem is spreading the revocation information as fast as possible and as convenient as possible to the (interested) entities.

For simplicity (and taking into account the remark about the problem falling apart into two different challenges) we distinguish the following two paths of information:

CA-REP This labels the communication path between the CA and the repository.

REP-EE This is the mnemonic for the path between the repository and the end entity.

As already stated, the two parts of the potentially long way from the CA to the end entity have to be dealt with separately. Nevertheless, both are important and we will use the same criteria for both of them. Note that we do not make any statement about the location of the repository here. It might be located very close to the CA — or very close to the end entity. These recommendations will be provided later on.

Security

Security is of course the ultimate ambition — that's what the whole PKI is all about. Revocation information has to be secured against tampering as entities depend on correct certificate validation for granting access to services. The weakest point within a public key infrastructure is the CA itself with its private key and certificate generation facilities which will render it a major target for an attack.

¹<http://www.mitre.org>

²<http://www.nist.gov>

Entities exchanging revocation information want to verify their communication partners in order to make sure they are not fooled by faked service facilities and supplied with incorrect information. This requires authentication mechanisms, which are fortunately to a large degree already provided by the digital signatures mentioned earlier.

Latency

What could be an upper limit for someone to wait for accurate and fresh certificate status information? The faster, the better. Apart from the use within personal communication most application will rely on fresh status information. Online banking and other financial actions as well as access authentication are obvious examples of applications, where latency is as crucial as availability

Both paths need to keep the latency as low as possible. For the CA-REP path, this means that the CA has to publish new revocation information instantly to the repository. On the REP-EE path, a suitable scheme preventing long latency has to be found.

Correctness

All entities within the PKI must be able to correctly determine the revocation state of a certificate within well-known (time) bounds. This applies to both paths just outlined and requires taking measures to exchange revocation information in a tamper-proof way (e.g. by digitally signing the message).

Availability

A certificate revocation scheme should always be available. Imagine a PKI within an online-banking environment: Every party involved wants to be able to verify certificates immediately.

The more critical a PKI becomes, the higher the needs for availability will grow. If a company uses certificates as credentials to access all of its resources, a PKI environment becomes quickly mission critical. This situation is perfectly suited for high availability solutions incorporating technologies like redundancy, load-balancing, standby-servers and replication.

The repository has the greatest demand for availability as it is the access point for the end entity needing information. As far as the CA is concerned, it needs to be available for actually reporting incidents that will lead to certificate revocation. However, this is not our concern here as we assume the certificate revocation to be already triggered by the CA once our solution takes over.

Scalability

A practical solution for a PKI has to scale with the growth of the online-community and not remain a bottleneck. This touches the mechanism, how the revocation information is spread effectively, as well as the network topology providing access to it.

Scalability should be considered especially when adding new CAs to the PKI and when providing more access points to the end entities. This should let us focus on the role of the repository being the message hub between the two mentioned entities.

Network Use

The network use is critical, for company-wide networks as well as dial-up users at home. This implies that the message size and the introduced overhead must be minimized in order

to deliver the client exactly the information that suits its need. We should avoid passing messages (and using precious CPU-time and network bandwidth) which render almost useless at the client side.

While network bandwidth is precious, one has to be aware of the fact, that a solution reaching our goal will always put a certain amount of extra-traffic on public networks. As the importance of PKI grows in time (with the security needs of our modern, networked society), we will probably have to deal with the fact that PKI-related traffic grows with at least similar speed.

Computing Time

Cryptographic operations are always time-consuming and servers may rapidly become bottlenecks. This should be avoided by minimizing the load of an entity providing revocation information. On the client side, following a certification path may be needed to verify the integrity of the downloaded revocation information, which in turn requires heavy cryptographic operations as well.

One should consider using a combination of both symmetric and asymmetric cryptography for encryption. For our application, far more important than confidentiality is data integrity, which can be achieved with hash functions and digital signatures. These operations perform rather fast in general.

Compatibility and Integration

It is important for a revocation scheme to be compatible to existing certificate standards and key management schemes, particularly to the X.509 recommendations, which have been designed with Certificate Revocation Lists (CRL, see Section 3.1 later on) in mind. Keeping compatibility and nevertheless integrate additional information into a certificate can be done by using certificate extensions as defined in X.509v3.

On the end entity side, seamless integration in existing PKI-aware applications is a must and major companies should be asked for official support. The task of gaining revocation information should be completely transparent to the end user.

3 Existing Solutions

This Section will outline existing approaches to the certificate revocation problem and consider different environments and needs. Each solution will be analyzed according to the criteria presented in the last chapter.

3.1 Certificate Revocation Lists (CRL)

CRLs have been proposed with the first version of the ITU's X.509 recommendations in 1988 (X.509v1) and have been improved in subsequent versions (X.509v2 and X.509v3). They are the first mechanism proposed for solving the revocation problem and have gained a promising acceptance.

CRLs are simple lists of entries corresponding to revoked certificates, each indicating the certificate's serial number, the time the revocation was made and other information such as the revocation reason. The CRL includes the CA's unique identity and is signed for data integrity purposes. A revoked certificate will appear only on the CRL originating from the CA which issued the certificate.

The CRL also contains the date and time when it was generated, thus providing an indication of its freshness, and a next-update indicator which announces the time when a new update of the CRL will be available. Once a certificate is on a CRL, subsequent

updates of this CRL will still contain its entry as long as the certificate would be valid according to the date indicated on the certificate itself (*valid until*).

Distributing the CRL may happen either with a push or a pull model. The former will allow the client to tune the request interval dependent on the needs of the application. The CRL itself is a typical PUSH-model of communication. Typically, it will be published through a kind of repository serving it to interested end entities. The standard extension “CRL Distribution Points” provides information where to get the latest CRL a client can use to check a certain certificate.

Analysis

The main drawback of a CRL is its growth in size. This leads to a network bandwidth use that is, given an application with a desire for instant revocation information such as online-banking, out of the acceptable range. The second striking factor is the freshness. CRLs are normally published in intervals which means there will not be any revocation information available between the update and the publishing of the new CRL. Newer revocations will thus be delayed until the next update occurs. This is also known as the *time-granularity problem* the CRL suffers by design. This is a fact high-security applications can not cope with and render the CRL approach almost useless in these environments.

There is no scalable solution for the delivery problem available within X.509’s CRL. We do have a list we can generate and even adjust the generation rate for providing better latency rates but can not really get the list to the potentially interested end entities. All we can do is publish the CRL in our local repository and leave the other tasks to the end entity.

Although the delivery of the lists to the end entities might be improved, the overall network use remains too high and the scheme is not scalable. CRL may do a reasonable job in closed environments like company-internal networks with low requirements for latency but fail to provide an Internet-wide community with certificate validation services.

3.2 CRL Segmentation

For operational efficiency when large CRLs may arise, a scheme for dividing the CRL into smaller lists has been proposed. The base for the segmentation may be the revocation reason. For example, the CA could issue one CRL for routine revocations (e.g. a change in a certificate subject’s identifying information) and another list for revocations due to security compromise. Similarly, a CA could issue a CRL for its end entity subjects and another for the CAs it may certify. The dividing may also take place based on pre-assigned sublists with new segments created on demand.

Segmenting the CRL must be transparent to the user and for each certificate, available information must indicate which segment must be consulted.

Analysis

Segmenting the CRL does not solve the scalability and handling problem of the CRL, as it still deals with lists in the same format. While creating sublists seems comprehensible and extends the granularity of the CRLs, it still has the problem of coping with various needs in freshness and can’t solve the time-granularity problem.

3.3 Delta CRL

This approach again tries to reduce the size of the CRL issued by a CA. Full lists are distributed within large intervals but upon each CRL update, only new entries which have been revoked since the last issued complete certificate revocation list are included. Thus only delta information is published reducing the size and network use of the CRL. The

end entity needs to maintain and update a local copy of the current CRL which requires a reliable cache.

Delta CRL information can be generated and distributed each time a new revocation occurs or be spread in regular intervals.

Analysis

This approach clearly reduces network load. It will probably not be the preferred solution for the REP-EE path as some end entities may not be online for 24 hours and miss a lot of delta-information which will force them to download entire list afterwards.

Still, one has to adjust the time interval between publishing the revocation information or even publish only when new data has arrived, meaning new revocation information is available at the CA. This solution will also not prevent end entities from getting more information than they need for fulfilling their tasks, which can be seen as unnecessary network use.

On the other hand, delta-CRL might be an interesting alternative for the CA-REP path. Here, the interval for full CRLs might be extended at will as we count on the repository being available whenever the CA wants to publish new revocation information and the delta-CRL might be published whenever a new certificate revocation occurs. Thus benefits in network use are guaranteed and we will not lose precious time between the CA and the repository and therefore eliminate another potential source of increased latency.

3.4 Windowed Key Revocation

This approach tries to overcome the weaknesses of CRLs by changing various parameters inside the CRL approach. A revocation window is introduced, bounding the time over which a revocation of a certificate is announced. This limits the size of the CRL as revocation information about a certificate only lasts a relatively short period in the CRL (during the so called "Revocation Window").

Further improvements incorporate push delivery to the verifiers (the end entities in our terminology) and multicast communication. The proposition further contains a feature called revocation aggregation allowing announcements (CRLs) from multiple sources to be aggregated by higher level authorities. This is the first attempt an existing solution makes in providing mechanisms for large scale distribution of revocation information.

Analysis

Again, reducing network load is the main advantage we gain over the original CRL approach. Only the fact that a verifier is able to detect a missing revocation information allows for bounding the time the certificate (its unique identifier actually) is on the CRL to the revocation window. However, it is not stated in the approach where to get the missing revocation information after the detection.

Using push delivery and multicast communication will not be able to serve the needs of a very large community concerning revocation information. Again, no distribution concept is available as pushing new revocation information to all clients is certainly not feasible.

Although the aggregation approach is promising, the concept remains inside the barrier of a push delivery system. Aggregating from different sources means in this context only that a higher level authority will carry out the delivery of various CAs.

3.5 Online Certificate Status Protocol (OCSP)

The OCSP is a simple request/reply protocol allowing an end entity to query a server for information about a certificate's status. It will issue a request and send the serial number of the certificate in question to the OCSP-server which in turn will answer indicating the

state of the certificate (e.g. revoked). Response extensions can be used to obtain detailed information about the certificate (e.g. issuance, validity). The protocol is described in RFC 2560 [11]. Various implementations exist in form of add-ons for CAs and general verification servers (targeting at VAs).

Analysis

This is a typical PULL-model allowing the client to control when and how much information it gets. Its performance is fine as long as the amount of requests is not too high for one single server. A simple check without extended information will only involve two messages and provide exactly the desired information.

An OCSP-server may soon become a bottleneck if it is the only access point for a CA. Thus, multiple OCSP-server should be able to fulfill requests for certificates from a certain CA which will require replication. Note that the RFC 2560 states nothing about the mechanism revocation information gets from the CA to the OCSP-server. This has to be done using another approach and incorporating other protocols. Most recent implementations support CRLs or CRL-derived methods for doing this.

OCSP seems the right approach into the direction of an infrastructure providing access points who can be queried about certificate revocation information. Here, the protocol would fit perfectly.

3.6 Simple Certificate Verification Protocol (SCVP)

SCVP³ is a kind of framework incorporating OCSP and other services to provide clients with certificate chains needed for path validation and revocation information. The protocol uses a simple request-response model as OCSP does and its typical use is expected to be over HTTP.

A trusted SCVP server can even perform full certificate validation for a client. If a client uses these services, it inherently trusts the SCVP server as much as it would trust its own path validation component. Clients may do this within an enterprise for security checking during path validation and enforcing security policies or they just lack the needed software for validation.

Analysis

Since the SCVP was first mentioned in an Internet draft in August 1999 and it is work in progress, there's not much sense in trying to analyze it. Nevertheless, we can apply the same thoughts as stated above in analyzing OCSP.

3.7 CRL in the DNS

RFC 2538 [10] describes a system for storing certificates and CRLs in the Domain Name System (DNS). To do so, a special CERT resource record (RR) is specified. No special statement about timing or freshness is made and the approach is just a formal specification.

Analysis

The idea of using the DNS-system is reasonable and DNS proves to be functional within the whole Internet. However, DNS is optimized for small packets, typically not bigger than 512 bytes including overhead. RFC 2538 advises to keep certificates minimized. This will definitely be a problem with CRLs growing in a linear manner.

³Complete reference is available in [5]

Using the DNS system will only solve the distribution problem but neither improve the latency nor reduce the network use. Nevertheless, DNS may be used as a good model for distributing revocation information within a similar architecture.

3.8 Conclusions

The previous sections have shown that there is no solution that can really solve the problem in an elegant way. All concepts either focus on ideas that can only solve one half of the problem by design or have drawbacks in the maximum latency and thus freshness of the information they provide.

Furthermore, none of the propositions takes into consideration the question of scalability proportionately to its importance. Fundamental scalability drawbacks result from design and, as the improvements for CRLs prove, are almost impossible to overcome. Most of the existing schemes expect the end entities to come very close to the CA to get revocation information which in turn creates potential bottlenecks. In addition, freshness is not considered to be a fundamental need for a revocation solution. This can have substantial impact on high security applications and prevent the technology from being used.

To overcome this situation, a new approach is needed which has the means of making guarantees about the maximum latency, availability and provides an architecture for distributing information in the large. This leads us to another view of the problem. What we really need is a protocol tight to an infrastructure providing access points for potential certificate users to collect information about a certain certificate anytime they want. This information should be guaranteed to meet certain freshness criteria and the infrastructure will thus be responsible for delivering fresh information about certificates for every access point as fast as possible.

4 The Certificate Revocation Framework (CRF)

The CRF is a new approach in trying to solve the certificate revocation problem which combines an infrastructure with suitable protocols to publish certificate revocation information to a number of client access points. We provide a description of the conceptual approach in this Section without going into the details here but lay our focus on the functionality and features of the CRF.

4.1 Overview

As stated in previous chapters, our goal will be to propose an infrastructure with client access points (CAPs) providing information about the status of any desired certificate. This will require us to build a distribution system capable of spreading fresh CRIMs (Certificate Revocation Information Message) to every CAP within a very short period of time.

Figure 1 gives the black-box approach where we do not (yet) look into the CRF itself but rather at the connections to its environment, the client access points. The figure shows a global PKI (where our CRF is included) and a client having a trusted relationship to a limited number of CAPs. Typically, its CA will be at one of these CAPs as there is already a relationship of trust between the CA and the end entity for certificate generation and management purposes. The settings just outlined would even allow the client to delegate the process of path validation to the CAP, an idea we will pursue later on.

4.2 The Structure

We propose a tree as the base topology of the CRF with the CAPs located at its leaf nodes. All certificate revocation information (CRI) will traverse the tree from their origin up to the root node and then down again to all leaves attached to the tree which are in fact our

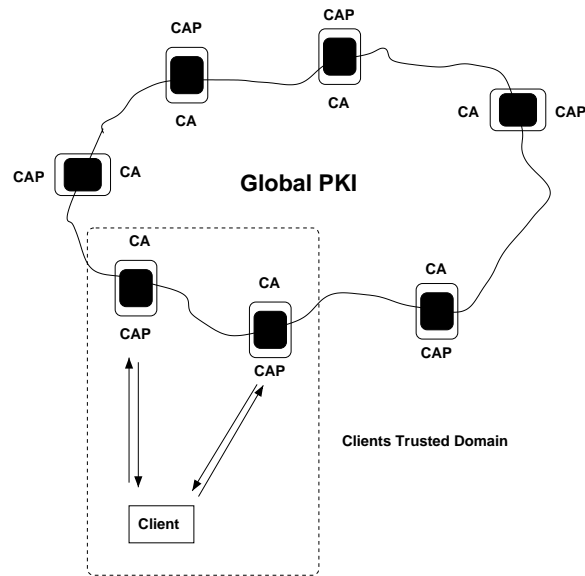


Figure 1: A black-box approach to the CRF

CAPs. This does not require the tree to be balanced in any way and allows to attach new subtrees at will without changing the structure above. The tree's internals can be seen as a large message hub enabling a CA to inform all attached CAPs almost instantly (see also Figure 2).

If you compare this structure to the ITU's X.509 recommendations, you will find little differences between their hierarchical model incorporating primarily CAs and our CRF proposition. This is volitional because we are interested in a seamless integration into the existing PKI model. The CAPs will thus typically contain a CA component that manages certificates of arbitrary end entities.

The topology proposed has the following advantages:

- Parallel message transfer (top-down) speeds up information distribution. This is a feature by design not by optimization.
- The number of hops needed for one message to be spread to all leaves is a linear function of the number of levels, not the number of partaking nodes. This is a major scalability advantage.
- The tree can be seen as a number of connected, autonomous logical rings operating at different tree levels. This should allow for just connecting rings together to form any kind of tree. Thus, an ideal corporation of autonomous subsystems is reached.
- Redundancy can be achieved by dynamically changing parent nodes of subtrees once their designated parent is down. This is part of ring management which is completely transparent to higher levels.
- The topology is completely scalable as new subtrees can be connected anytime at will and will not affect the performance of the net seen from the existing nodes in the tree.

See figure 2 for a picture of the internal tree structure. It will also show the partitioning into logical rings, a feature explained in section 4.5.

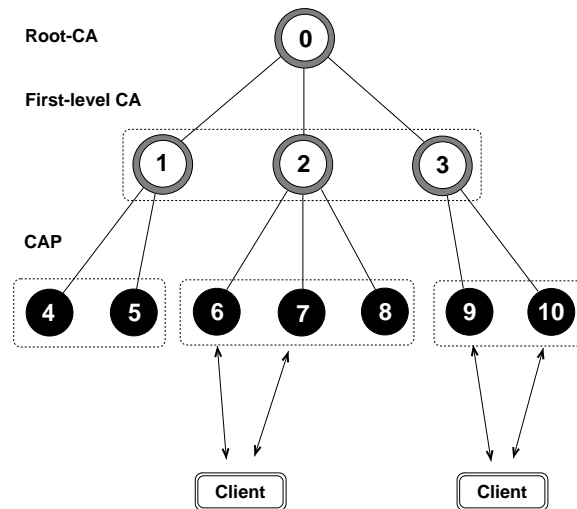


Figure 2: Internal tree structure of the CRF

4.3 Publishing Revocation Information

Publishing new CRI typically starts at one of the CAPs. The appropriate CA (the CA that issued the certificate in question) triggers the revocation action. First, it generates the required data structure packed with the certificate information, the revocation indications and identification data about itself. This structure is then signed by the CA and launched for distribution.

The distribution process starts by sending the CRIM (the generated data structure called *Certificate Revocation Information Message*) to the parent node of the CAP. Therewith, the CA's duty is carried out and the CRF will now take over. The CRIM will climb up the tree until it reaches the root node and then step down the tree to each CAP connected to the topology. The CRIM is cloned at will on its journey and distributed in parallel according to the tree's structure.

Note that once a node anywhere in the tree has received a CRIM it is sent to all attached nodes except the one from which it arrived. Consider for example the CAP number 8 in Figure 2 evoking a CRIM. It will first arrive at node 2 where it will be cloned for simultaneous distribution to the parent node (the root node in our example) and the nodes 6 and 7.

4.4 Reliability and Failure-Tolerance

The guarantees about correctness, availability and latency require sophisticated measures for tolerating failures of nodes inside the tree. This is a must for providing acceptable reliability with our infrastructure.

A mechanism capable of intercepting node failures and reconfiguring dynamically parts of the tree is needed. A node in the tree has to rely on a working parent node for being able to distribute new CRIM instantly. A failing parent node is fatal for both ways of communication (tree up and down), as the path to the root node is broken. This situation can be compared to a network partition.

We must also have to cope with multiple node failures inside the tree and making guarantees of CRIM delivery in this situation. This will introduce additional communication between the nodes monitoring each other for failures - this is the price for redundancy.

Figure 3 compares between publishing and distributing a CRIM in an error less tree (a) and distributing while one node is down (b).

In step number 3, the failure of node C1 is detected in (b) by mean of a mechanism we will describe later on. In turn, the subtree of node C1 is reconfigured and the direct child nodes get attached to a new parent. The concept of taking over a failed neighbour node is the key to a stable and reliable infrastructure.

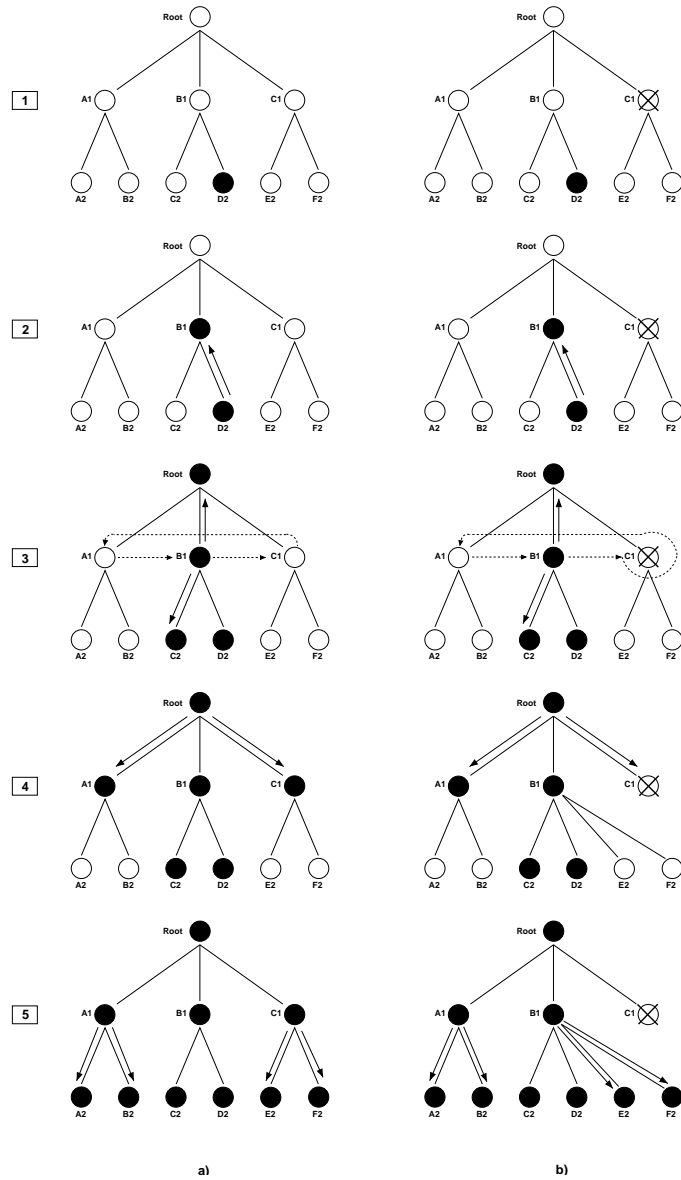


Figure 3: Distributing a CRIM with (b) and without (a) node failure

4.5 Logical Rings

Redundancy inside the CRF is achieved by splitting up the whole tree in small logical units, called *logical rings*. All nodes in a logical ring have the same parent node and are located on the same level in the tree. This will split up our sample tree in figure 2 into three subtrees (the root node is not a logical level) consisting of the following nodes:

1. On the 1st level: {1, 2, 3}

2. On the 2nd level: {4, 5}
3. On the 2nd level: {6, 7, 8}
4. On the 2nd level: {9, 10}

Logical rings will be responsible for constantly monitoring their members and take appropriate actions when one of them fails. This is achieved by using a protocol that lets the ring detect the (communication) failures of its members (the dotted line in Figure 3, step 3, is in fact the protocol message detecting the failure of node C1). Once a node is down, another node in the ring should take care of the children of the failed node and guarantee their integration in the tree by taking them over and informing them about the parent change. This requires profound knowledge of the neighbours child nodes which in turn causes network traffic. We simply can't take over a node's role if we lack the knowledge of its environment and must therefore provide a solution for spreading neighbour knowledge in the logical ring.

The tree will thus change dynamically depending on the state of the nodes composing it. Note that we try to keep the knowledge a single node needs to have of the tree as small as possible. This is necessary for scalability reasons and will keep our protocol as simple as possible. So far, a node only has to be aware of its parent node, its neighbours in the logical ring and their children.

4.6 The Client Access Point (CAP)

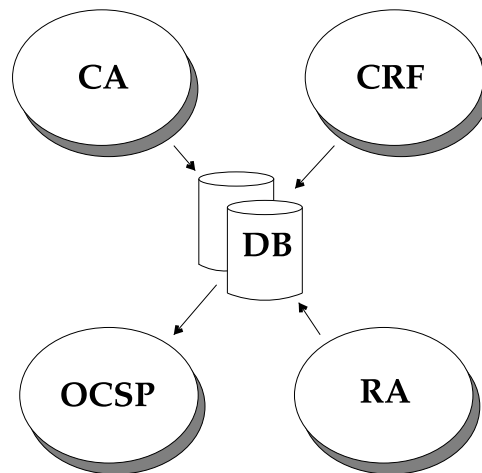


Figure 4: Internal structure of a client access point

As stated earlier, the hierarchical structure corresponds to the topology of a PKI proposed by X.509. This lets us make use of the full amount of synergy we will receive hereby. Each CAP can incorporate a full range of PKI services as indicated in figure 4.

So far we only provided an approach for sending CRIMs to every CAP but not beyond. The protocol between end entities (we will call them clients for simplicity) will be a typical request/reply protocol for querying about certain certificates. Here we can make use of existing work and therefore propose OCSF for this part of the communication. OCSF allows for placing an OCSF-server component at the CAP and lets the client run queries at will.

The question of storing CRIM is still unsolved. The CAP has to keep certificate revocation information as long as the affected certificate is valid according to the validity period

stated on the certificate itself. This requires the use of a database to store the CRIM and let the OCSP server access this data storage at will getting the information it needs for the client's query. There is also a security task involved here: Every CRIM has to be checked for a correct digital signature of the CA issuing the CRIM. If this check fails, the CRIM will never make it into the local store.

Figure 4 shows the typical components of a CAP. The CA managing the certificates and also triggering revocation actions, the CRF component responsible for interacting with the CRF, the RA and finally the OCSP server. All components will interact with one data store modeled as a database in the figure.

Potentially, each CAP is capable of providing information about any certificate that has been revoked (We assume silently that a CRIM had been generated and distributed inside the CRF). What if a CAP is queried about certificate X and has no suitable information? Our model of distributing only bad news (CRIM in our case) will lead us to the conclusion that every certificate we do not explicitly have revocation information about is valid. Is this feasible?

Intermediate nodes will typically also contain CAs for certifying the CAs at the client access points. Thus the full path validation is easily achieved by simply following the tree structure.

5 Analysis and Conclusions

We will attempt to analyze the CRF in this section and use the means provided in former sections of this paper. An attempt to gain answers to the question whether our solution is a pure overkill or not is also made in the following section.

5.1 Compared to the Requirements

In comparison with the requirements outlined in Section 2, we can make the following statements about the CRF:

Security and Correctness The CRIM transporting revocation information are secured by applying digital signatures to the data – this ensures for both, security and correctness. Each involved party will thus be able to verify the CRIM's origin. However, the OCSP does not provide for authenticating communication partners. The end entity will therefore not be able to verify the CAP's identity, a drawback that has to be overcome.

Latency As already stated earlier, the latency of the CRF is depending on the tree's height and nothing else (apart from failure conditions, of course.) By choosing an appropriate tree structure, we can keep the latency very low and almost deliver real-time information to the CAPs. This is an immense advantage our structure has and probably the key for a wide acceptance.

If we compare our latency to existing solutions, only pure OCSP would reach the same freshness. By incorporating OCSP and moving the OCSP-server close to the client we can make use of this advantage OCSP offers.

Availability The redundancy in our structure enables the CRF to provide maximum availability. This holds true for the tree internas but not the CAPs who might fail and can not be replaced as their actual clients are not known. Therefore, a client coming across a failed CAP will require an alternative access point. This is provided by either choosing one at will or, if further service delegation is invoked, choosing another CAP in the client's trusted domain.

Scalability This is one of the main advantages the CRF offers by design. Enlarging the CRF can be done by just connecting another subtree to one of the CAPs or attaching a new node to a logical ring with a separate branch. None of these operations will have an impact on the latency of existing CAPs.

In terms of scalability, our solution turns out to be superior than any existing approach and thus provides the basis for a global PKI.

Network Use We have to distinguish between the traffic, the management generates and the CRIM traffic. The redundancy protocol is our base network use we can not eliminate as it is crucial for availability. In comparison with other approaches, this can be seen as a drawback our solution introduces here.

The traffic a CRIM generates is depending on the number of nodes our CRF consists of. We try to keep the data structure as small as possible to reach approximately the same size as a delta CRL message. This calculation is limited to the internals of the CRF and network use for communication between end entity and CAP have to be considered separately. Here, OCSP determines the amount of traffic generated.

In general, our solution will provoke more traffic than any existing solution because we do have network use not directly related to client queries. This is actually the price we have to pay for increased stability and availability.

Computing time The computing time introduced with our solution is minimal. The intermediate node's only operation besides message routing and dealing with the redundancy protocol is verifying the digital signature of a CRIM. The CAP will do the same and also manage the local data store. The overhead introduced by OCSP is minimal but the database interaction involved in client queries may take some time.

Compatibility and Integration We keep complete compatibility with the X.509 recommendations. The CRF can be integrated seamlessly into hierarchical public key infrastructures and we do not even specify new certificate extensions.

Integrating OCSP into the end entity is not a critical issue and simplifies the handling of certificate revocation in comparison with the widely implemented CRL approach. OCSP servers are already commercially available.

5.2 Is it Overkill?

One might consider setting up a whole infrastructure consisting of a network of nodes as an overkill for the certificate revocation problem. We will try to give some answers below:

- Seen from the PKI's point of view, the nodes incorporating our CRF are already in place. We do just add new functionality.
- There is no need in creating a new network. We merely use existing connections and create a logical network on top of them.
- The redundancy protocol, provoking the base network use, can be tuned by parameters allowing to lower the traffic generated. This will impact the latency but provides a remedy to prevent charging the network too much. These possibilities have just been explained above.
- Certificate revocation is a major problem in a PKI and its correct operation an important part of the whole security concept. A client not being able to verify a certificate's status will lose a considerable amount of trust into the system. This certainly vindicates the additional expenditures.
- A solution providing accurate availability and freshness will always incorporate a higher effort. Coping with failure conditions is never for free and requires to spread additional knowledge in the system, something we can discard in simpler systems.

5.3 Conclusion

In our opinion, a scheme has been found that proved to be suitable to satisfy the needs of an ever growing PKI. It is scalable up to the size of a global PKI and the integration into existing structure is not a complicated task.

The analysis given in the previous chapter has shown the advantages we can gain from the introduced tree structures. Namely latency and scalability are not any longer an issue preventing the growth of a PKI serving the global community.

The drawbacks introduced are to be found in the additional traffic our infrastructure generates and the expenditure of managing a distributed environment of CRF-nodes. However, this has to be considered in relation to the benefit we encounter. Reliability, the main cause of the additional network load, will always request for additional effort.

We are convinced, that the gains outnumber the disadvantages outlined above and that our solution is in fact capable to solve the PKI's certificate revocation problem in a feasible manner.

References

- [1] Bruce Schneier, *Applied Cryptography*, John Wiley & Son, Second Edition, 1996, ISBN 0-471-11709-9.
- [2] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone *Handbook of Applied Cryptography*, CRC Press LLC, 1997, ISBN 0-8493-8523-7.
- [3] G. Coulouris, J. Dollimore, T. Kindberg *Distributed Systems – Concepts and Design*, Addison-Wesley, Second Edition, 1994, ISBN 0-201-62433-8.
- [4] Phillip Hallam-Baker, *OCSP Extensions*, Internet Draft, September 1999 (work in progress).
- [5] A. Malpani, P. Hoffman, *Simple Certificate Validation Protocol (SCVP)*, Internet Draft, August 1999 (work in progress).
- [6] R. Housley, W. Ford, W. Polk, D. Solo, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, Internet Draft, October 1999 (work in progress).
- [7] D. Eastlake, C. Kaufman, *Domain Name System Security Extensions*, RFC 2065, January 1997.
- [8] R. Housley, W. Ford, W. Polk, D. Solo, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, RFC 2459, January 1999.
- [9] C. Adams, S. Farrell, *Internet X.509 Public Key Infrastructure Certificate Management Protocols*, RFC 2510, March 1999.
- [10] D. Eastlake, O. Gudmundsson, *Storing Certificates in the Domain Name System (DNS)*, RFC 2538, March 1999.
- [11] M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, *Online Certificate Status Protocol - OCSP*, RFC 2560, June 1999.
- [12] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, T. Ylonen, *SPKI Certificate Theory*, RFC 2693, September 1999.
- [13] ITU/ISO, *ITU-T Recommendation X.500 - Information Technology - The Directory: Overview of Concepts, Models and Services*, 1988.

- [14] ITU/ISO, *ITU-T Recommendation X.509 - Information Technology - Open Systems Interconnection - The directory: Authentication Framework*, November 1993.
- [15] ITU/ISO, *ITU-T Recommendation X.509 - Information Technology - Open Systems Interconnection - The directory: Authentication Framework*, November 1997.
- [16] ITU/ISO, *Final Text of Draft Amendments DAM 4 to ISO/IEC 9594-2 on Certificate Extensions*, April 1996.
- [17] Ian Curry, Entrust Technologies, *Version 3 X.509 Certificates*, July 1996.
- [18] Petra Glöckner, *X.509v3 Certificate*, July 1996.
- [19] Silvio Micali, *Efficient Certificate Revocation*, Laboratory for Computer Science, MIT, March 1996.
- [20] Patrick McDaniel, Sugih Jamin, *A Scalable Key Distribution Hierarchy*, Electrical Engineering and Computer Science Department, University of Michigan, April 1998.
- [21] Patrick MacDaniel, Sugih Jamin, *Windowed Key Revocation in Public Key Infrastructures*, Electrical Engineering and Computer Science Department, University of Michigan, October 1998.
- [22] Stuart G. Stubblebine, *Enforcing Revocation in Distributed Systems*, AT&T Bell Laboratories.
- [23] S. Berkovits, S. Chokhani, J.A. Furlongm, A. Geiter and J.C. Guild *Public Key Infrastructure Study: Final Report*, Produced by the MITRE Corporation for NIST, April 1994.
- [24] Ueli Maurer, *Modelling a Public-Key Infrastructure*, Department of Computer Science, Swiss Federal Institute of Technology (ETH), 1996.
- [25] David A. Cooper, *A Model of Certificate Revocation*, Computer Security Division, National Institute of Standards and Technology, October 1999.
- [26] Patrick McDaniel, Aviel Rubin, *A Response to "Can We Eliminate Certificate Revocation Lists?"*, EECS Department University of Michigan and AT&T Research Labs, August 1999.
- [27] C. Adams, R. Zuccherato, *A General, Flexible Approach to Certificate Revocation*, Entrust Technologies, June 1998.
- [28] David A. Cooper, *A More Efficient Use of Delta-CRLs*, Computer Security Division, National Institute of Standards and Technology, October 1999.
- [29] Mastercard, Visa, *Secure Electronic Transaction LLC (SETCo)*, <http://www.setco.org>.