Selecting Revocation Solutions for PKI

André Årnes, Norwegian University of Science and Technology,
Mike Just, Entrust Technologies,
Svein J. Knapskog, Norwegian University of Science and Technology,
Steve Lloyd, Entrust Technologies,
Henk Meijer, Queen's University

Keywords

Certificate revocation, PKI, X.509, CRL, OCSP, CRT

Abstract

This paper is based on the work with the thesis "Public Key Certificate Revocation Schemes" written at Queen's University 1999-2000 and submitted to the Norwegian University of Science and Technology March 2000. The paper provides a survey and a comparative analysis of available revocation schemes for PKI certificates. Performance simulations on some revocations schemes are also presented. As a conclusion, the schemes are evaluated with respect to combining different schemes in order to achieve better performance.

1 Introduction

Certificate Revocation is the action of declaring a certificate invalid before its validity period is at an end. This can be viewed as "blacklisting" the certificate. A method for revoking certificates and distributing the revocation information to all involved parties is necessary in a PKI. There are several possible reasons requiring the invalidation of certificates before the expiry date, including suspected key compromise, detected key compromise, change of subject name and change of relationship between a subject and an authority [1].

A major problem with revocation of certificates is that each end entity has to be able to obtain real-time revocation information in order to be able be determine the current revocation status of any certificate at any time.

One approach to reducing the resource demand involved in revocation status checks is to allow caching for a certain period of time, as well as grouping the revocation information into lists. One final approach that is presented in this paper is to reduce the size of each revocation information entry, as well as using lightweight signatures in order to reduce processing.

2 PKI

Public key infrastructure (PKI) is defined to be "the set of hardware, software, people, policies and procedures needed to create, manage, store, distribute, and revoke public key certificates based on public key cryptography" in the IETF PKIX Roadmap [2].

2.0.1 Public key certificates

A public key certificate is a user's public key, identity and some other information digitally signed by a trusted authority [3]. Certificates are used to prevent the use of a forged key in place of a legitimate key. A trusted authority, or *certification authority* (CA), is a third party that is trusted by all its users. The CA certifies the correctness of the content in certificates by digitally signing them.

2.1 X.509

One of the most popular standards for public key certificates is included in the ITU (International Telecommunication Union) X.509 standard [4, 1]. The X.509 standard specifies how authentication information is created and stored by a directory server, how user authentication information can be obtained from a directory, and by which methods applications may perform authentication [5]. The CA issues certificates signed under the CA's private key, and the users are expected to know the CA's public key in advance.

2.1.1 Certification paths

If more than one CAs are involved, two users may be certified by different CAs. It can not be assumed that the users have the public key of the CA that certified the other user. In order to verify this public key, the user has to obtain a certificate for the other user's CA from a CA whose public key is already securely obtained. CA certificates are certificates for a CA issued by another CA, which implies that there is a trust relationship between the two CAs [1]. This technique can be applied recursively to obtain an increasing number of CA certificates until the public key of the CA in question is obtained. In this way, there is a *certification path* to the other user involving all the intermediate CAs whose public keys have to be securely obtained.

2.1.2 Certificate Validation

A certificate validation is a process of different verifications of a certificate, including the verification of the validity period and integrity of a certificate. Also, if applicable, a *certification path validation* is performed. Finally, the process checks the *revocation status* to ensure that the certificate in question has not been revoked [6].

3 Criteria for analyzing Revocation Schemes

In order to perform a comparative analysis of available schemes, the most important criteria have to be identified. Table 1 lists eight different criteria that were identified in [7].

Criteria	Description
Performance	The network and processing demand of a scheme in terms of Peak Load/ Request Rate, Load Distribution, Delay, and Size
Timeliness	Maximum time between the actual revocation and the time when the revocation is known to all parties
Scalability	How the scheme behaves if the amount of users and the revocation rate increase. This criteria is highly related to performance and timeliness
Security	Non-repudiation issues and whether the scheme's integrity depends on 2 or 3 parties (trusted parties)
Standards Compliance	Whether the scheme is standardized or proprietary, and whether it is actually implemented
Expressiveness	Level of granularity of information on a per certificate basis
Scheme Management	Issues related to managing the scheme, like automatization, secure archiving, scheme complexity
On-line vs. Off-line	Whether the scheme is suitable for off-line use

Table 1: Evaluation criteria for revocation schemes

4 An Overview of Available Solutions

This section presents a number of revocation schemes and describes their advantages and disadvantages. For a comprehensive treatment, see [7].

4.1 Revocation Lists

The traditional means of managing certificate revocation is to periodically publish a digitally signed data structure called a *certificate revocation list* (CRL). A CRL is a time-stamped, digitally signed list containing the *certificate serial numbers* of all revoked certificates within a CA domain. The CRL scheme is included as part of the X.509 standard [4]. The standard discussed in this chapter is the current 1997 X.509 standard with Version 2 CRL. This text refers to the CRL scheme without any modifications as the *basic CRL scheme*.

CRLs have been criticized for not being able to provide the required service and being too costly [8, 9]. However, the CRL scheme is standardized and widely used, and the many modifications and extensions allow CRL based schemes to be deployed in more efficient manners. The CRL based schemes are now considered a suitable basis for many (but not all) certificate revocation systems [10].

4.1.1 CRL Distribution Points

The CRL Distribution Points scheme extends the CRL scheme by addressing the maximum size of a CRL [6, 1]. The size of a *CRL subject population* is limited by dividing the total population for one CA into a number of *segments*. Each segment is associated with a CRL distribution point, which can be located on different hosts and/ or directories on the same host. Each certificate has a pointer to the location of its CRL distribution point, so there is no need to either search through distribution points or have a priori knowledge of revocation information locations.

With this scheme, both the maximum and average load is generally reduced compared to full CRL postings, whereas the average request rate will increase since each end entity may require more than one segment. This scheme is more scalable than the basic CRL scheme, since the segments can be distributed over several hosts, and each end entity is likely to request less than all segments.

4.1.2 Delta CRL

The Delta CRL scheme is another mechanism that addresses the size of the CRLs. A delta CRL is a digitally signed list of incremental changes that have occurred since the last complete posting of a CRL [1]. Delta CRLs are periodically posted, and act as updates to the last full CRL postings. Delta CRLs are generally significantly smaller than the full CRLs, and they can be posted much frequently. Full CRL postings are cached on the end entities and referred to as *base postings*, while the delta CRLs are to be considered *incremental postings*. Note that it is possible to post multiple delta CRLs for the same base CRL. A delta CRL is an increment to a single base CRL only (and never to another delta CRL). The newest revocation information is obtained from the *newest* base CRL posting and the *newest* Delta CRL.

With delta CRLs, the validity period of base CRLs can be increased, since the delta CRLs ensure higher timeliness with significantly smaller postings. Consequently, the average network load can be reduced compared to the basic CRL scheme.

4.1.3 Over-Issued CRL

It is possible to reduce the peak request rate by allowing multiple CRLs to have overlapping validity times. In this way, there are multiple valid CRLs available, each with a different expiration time. This scheme can be implemented by issuing a new CRL earlier than the "Next Update" time for the previous CRL [11]. By doing this, there will exist multiple not-yet-expired CRLs with overlapping lifetimes. An end entity will always request the *newest CRL*, but the lifetime of that CRL is longer than the period to the next CRL issuance.

The over-issued CRL scheme distributes the peak request rates from the CRL scheme in such a way that the peak request rates are reduced, but occur more frequently. The resources will be better utilized, and the timeliness and scalability can be improved compared to the basic CRL scheme. For further improvements of performance, over-issuing can be combined with CRL distribution points and/ or delta CRLs. In the case of the delta CRL scheme, it is possible to over-issue both the base CRL and the delta CRL. Note that there may be non-repudiation consideration when allowing multiple valid sources of revocation information.

4.1.4 Authority Revocation Lists

An *Authority Revocation List* (ARL)[6] is a CRL that is used to revoke CA certificates exclusively (i.e. there are no end entity revocations present). ARLs are obtained from CA-certifying CAs, and reduce the load involved in verifying the certificate path.

When validating a certificate path, a valid ARL must be available for each CA that has signed certificates in the path with the exception of self-signed trust anchors. With ARLs, each list is either empty or small (CA certificate revocation rarely occurs), significantly improving the performance of a system relative to full CRLs.

4.1.5 Indirect CRL

In the Indirect CRL scheme, a different authority than the certificate issuer can issue a CRL [6]. A CRL can be configured to contain revocation information from multiple certification authorities, so that the number of CRLs needed by end entitites during certificate validation can be reduced. The overhead for obtaining multiple CRLs is reduced, and the recursive certificate verification for a certification path may be reduced. One way to deploy this scheme is by having the Indirect CRL issuer obtain CRLs from each individual CA and compile them into one CRL.

The main performance gain in using indirect CRLs is that there is less overhead in requesting multiple CRLs from different sources. This can lower the request rate, as well as the average load. An important consideration for this scheme is that the indirect CRLs never should grow so large that the performance benefit of combining the CRLs is lost. Note that since the indirect CRL issuer has to be trusted, this scheme is a 3-party scheme.

4.1.6 Dynamic CRL Distribution Points

A problem with the use of CRL distribution points is that the CRL segmentation is permanently fixed for the lifetime of the certificates involved. The issuing CA is required to define a static partitioning before issuing certificates.

A scheme for dynamic partitioning of CRL distribution points was suggested in [12]. The scheme has previously been referred to as *Enhanced CRL Distribution Points* scheme [7], and it implements dynamic partitioning by providing a a set of scope statements with their associated pointers. A *scope* statement indicates a range of certificates that are covered by the associated CRL partition. Two CRL extensions are defined [12]. The *CRL Scope Field* allows scope statements to be defined and associated with CRL distribution points through *Status Referral Fields.*

An end-entity validating a certificate can obtain a CRL containing a Status Referral extension when the certificate's enhanced CRL distribution point is referred. This extensions may include a scope that covers the certificate in question and a pointer to a new location for the CRL for the certificate in question.

4.1.7 Freshest Revocation Information

With different timeliness requirements for different end entities, it has been suggested to have a separate reference to the freshest revocation information available for end entities with high timeliness demands. A new certificate extension called *Freshest CRL* [12] identifies the revocation information to which a certificate user should refer to obtain the freshest revocation information (typically the most updated delta CRL). In this way, it is possible to have a system that discriminates between applications that require timely revocation information and applications that do not have such requirements. The performance and scalability of this scheme is improved over the basic CRL and the delta CRL scheme by providing only sufficiently timely revocation information for a particular application or end entity.

4.2 Online Revocation Solutions

As a response to the low timeliness of some periodically updated certificate revocation schemes, protocols for *on-line status checking* have been developed. Some certificate-based systems cannot tolerate the revocation delay resulting from the periodically updated schemes. With *real-time* revocation checking, an end entity can confirm the certificate validity by performing an on-line transaction that returns the current revocation status for a certificate.

Common for the on-line revocation checking protocols is that they do not represent full revocation schemes. The mechanisms described in this chapter are only protocols that are limited to provide an on-line response from a *server*. The mechanisms used to provide the server itself with fresh revocation information is not within the scope of the protocols. The Server can rely on on-line protocols itself, obtain the information directly from a database, or even request periodically updated information like CRLs. In this way, the protocol used is *no more timely than the back-end mechanism* [6] used to retrieve the necessary revocation information. Note also that the on-line protocols generally introduce another trusted party, so that the scheme is 3-party rather then 2-party.

4.2.1 OCSP

The *On-Line Certificate Status Protocol* (OCSP) [13] is a protocol in which on-line revocation information is available from an *OCSP responder* through a request/ response mechanism. OCSP is designed to check the certificate revocation status exclusively.

In this protocol, an end entity requests the revocation information for one or more certificates (bulk requests are allowed) in an OCSP request. An *OCSP responder* checks the revocation status information for the involved certificate(s) against its available revocation information, and issues an OCSP response to the end entity. The OCSP response must be digitally signed (an expensive operation) in order to ensure the integrity of the response and the authenticity of the trusted entity. Optionally, the OCSP request can be signed as well.

With the basic OCSP scheme, there is one request for one or more simultaneous certificate revocation status checks at any given end entity. The peak request rate is likely to be at its maximum when the end entities are most heavily used. Due to the fact that the OCSP responder has to sign each reply, a significant processing load is introduced at both the responder and the end entity. In addition, the performance of a scheme using OCSP is dependent on the back end mechanism.

In an expired Internet Draft [14], Branchaud suggests a back end for obtaining and caching certificate status information from intermediate Servers. The method is based on the concept of an *OCSP CA*, an OCSP responder that is *co-located* with a CA. The CA operates the OCSP responder in such a way that the OCSP responder has direct access to the CA's own authoritative revocation information. In addition to the OCSP CA, there is a number of ordinary OCSP responders (not associated with any particular CA). These OCSP responders are capable of caching revocation information, and accept OCSP requests. When getting an OCSP request, an OCSP responder forwards the request to another responder unless it has the valid revocation information in question in its repository or cache. A number of *intermediate* OCSP responders may be involved before the request reaches either the OCSP CA or some non-expired cache with the necessary information.

It is also possible to *precompute* or *preconstruct* OCSP responses in order to improve the performance of the scheme at the cost of reduced timeliness. This means that an OCSP response for each certificate is created and signed with a predefined validity period. In this way, the OCSP responder does not have to perform the processing intensive job of signing an OCSP response for every OCSP request. A problem with this method is that an OCSP request is allowed to include multiple certificate identifiers. This makes the job of pre-computing every possible OCSP response very large.

4.2.2 OCSP-X, SCVP, and DCS

There is a number of on-line protocols that are more extensive than OCSP.

OCSP-X [15], or OCSP extensions, provides a richer set of semantics for OCSP. With these extensions, an end entity is able to delegate the full task of deciding whether a certificate should be relied upon and whether it is acceptable for for a particular operation.

The *Simple Certificate Verification Protocol (SCVP)* [16] is a separate protocol that is capable of handling (parts of or) the entire certificate validation process. With SCVP, end entities can avoid the overhead involved in processing the certificate validation locally. The protocol may also be used to centrally enforce some validation policy.

The *Data Certification Server (DCS)* [17] is a trusted third party that can be used as a component in building non-repudiation services. DCS is capable of verifying the correctness of specific data submitted to it. This service may, for example, be used to verify the correctness of a signature, the full certification path, and the revocation status of a certificate. Note that DCS provides more general services than OCSP-X and SCVP.

4.3 Reducing the load

There are a number of certificate revocation schemes that are designed to provide revocation information with reduced data structures compared to full revocation lists. In order to do this, the end entities obtain information in support of validating a single certificate rather than a complete list. A problem with this method (as with OCSP) is that the process of digitally signing each revocation reply is processing intensive. The schemes with reduced data-structures (called *trusted dictionaries* in [10]) attempt to solve this problem by using one-way hash functions in order to provide *lightweight digital signatures*. Common for most of these schemes are that they are not standardized like CRL and OCSP based schemes, and only CRT has actually been implemented for common use. Additionally, these schemes do not (with the exception of CRT) support the expressiveness found in CRL.

4.3.1 Certificate Revocation Status Directory

The *Certificate Revocation Status Directory* (CRS Directory) [9] was an early suggestion for altering the underlying data-structure of raw lists of revocation information to a reduced and more scalable data-structure. In the paper originally describing the CRS Directory scheme [9], Micali suggests the following strategy:

- "Increase the amount of information sent by CAs to the Directory during an update, but design this larger information so as to enable the Directory to answer more succinctly to certificate queries."

This follows from an observation that the communication due to CRL updating the directories is relatively small, whereas the directory to end entity communication is significantly larger.

This scheme suggests extending the certificates with two quantities, namely a 100-bit value Y and a 100-bit value N. The CA generates Y for each certificate by selecting a secret 100-bit value Y_0 for each certificate and then evaluating it with a one-way function F 365 times, such that $Y = Y_{365} = F^{365}(Y_0)$. N is generated for each certificate by selecting a secret N_0 for each certificate and evaluating F on it once, such that $N = F(N_0)$.

The CA then proceeds to update the CRS Directory with:

- An authenticated and updated list of all serial numbers corresponding to issued and *not-yet-expired* certificates. This list can simply be implemented as a bit-string, where each bit c_n represents the certificate serial number n.
- For each not-yet-expired certificate, for a given day i, the CA sends a 100-bit value $Y_{365-i} = F^{365-i}(Y_0)$. If the certificate has been revoked, CA simply sends N_0.

The CRS Directory now stores all not-yet-expired certificates with its latest Y value for all valid certificates. Otherwise, the CRS Directory stores N_0 with the certificate in question. For every certificate, the CRS Directory checks that either the 100-bit value V gives either $F(V) = Y_{365-i+1}$ (which means that $F(V)$ equals the 100-bit value received the day before, such that the certificate is not revoked) or $F(V) = N$ (which means that the certificate is revoked).

When a user inquires about a certificate revocation status, the CRS Directory sends the user the latest 100-bit value V for that certificate. At a current date i, the user checks whether $F^i(V) = Y$ (certificate is valid), or if $F(V) = N$ (in which case the certificate is revoked).

4.3.2 Hierarchical Certificate Revocation Scheme

The *Hierarchical Certificate Revocation Scheme* (HCRS) [18] is an extension of the CRS Directory. It was observed that CRS sends the same amount of information about each certificate. HCRS proposes a method of reducing the amount of data transferred by organizing the necessary revocation information in a tree. Note that the paper first presenting HCRS [18] also suggested a generalization of the scheme that is not hierarchical that increases the performance of the scheme. The name HCRS was used in [19], but it is not applicable to the generalized scheme.

4.3.3 Certificate Revocation Trees

Certificate Revocation Trees (CRTs) is a revocation technology based on Merkle hash trees in which a tree represents all certificate revocation information known to some PKI domain [20]. CRTs are based on a sorted list of unique revoked certificate serial numbers.

The list of revoked certificates is placed into adjacent ranges where the lower end point of each range contains the unique number of a revoked certificate. The *CRT issuer* builds a data structure representing each range, and builds a binary hash tree with the data structures as leaves. The leaves are then used to compute their parent nodes, where a parent is computed as the secure hash (with a size of about 20 bytes) of the concatenation of its two child nodes. By recursively applying this computation, the root node will be a function of all leaf nodes. The issuer digitally signs the root node to ensure authenticity and integrity of the CRT. An issue with CRTs is that they have to be recomputed at every update, increasing the processor load and potentially reducing the timeliness.

At the CRT issuer, the revocation status of a certificate can be determined by looking up a certificate's unique number in the leaf nodes. If the certificate is at the lower end point of some range, then the verifier finds cryptographic proof of the revocation by traversing the tree and verifying that the root node and the intermediate nodes are properly signed. Similarly, if the certificate's unique number is in a range, but not at the lower end point, it can be proved that the certificate is not revoked.

At the end entity, this verification is done by obtaining the *supporting nodes* required to compute each intermediate node up to and including the root node. A supporting node is the second child needed to perform the computation of a parent, and one supporting node is needed for each level of depth in the tree. All supporting nodes are provided by the CRT issuer.

Note that the supporting nodes also can *piggyback* on its associated certificate. This information is called a *Freshness Proof*, and allows a revocation check to be performed without requesting the CRT Issuer. The timeliness of the CRT scheme with piggybacking is determined by time-stamping the freshness proof. CRT is implemented and in commercial use, but it is not an open standard. Also, CRT is a 3-party scheme, requiring a trusted CRT issuer in addition to the CA.

4.3.4 Naor's and Nissim's scheme

Naor's and Nissim's scheme [21] extends the idea of CRTs by using a more suitable structure than a binary tree. Naor and Nissim suggest using a 2-3 tree, in which every interior node has 2 or 3 children and the paths from the root to the leaves always have the same length. The main operations on this tree require minimal recomputations in the tree.

5 Modelling

In the papers [11] and [22], David A. Cooper developed a mathematical model for analyzing certificate revocation schemes and applied this model to several CRL-based schemes. These models assume that the number of end entities is reasonably large and that the certificate validation attempt times are independent, so that an exponential inter-arrival probability density can be used to model the CRL scheme. The models primarily study the peak request rate.

An important observation from the models is that the peak request rate for CRLs will occur at the time of the CRL issuance. At this time, there will be a rush to obtain the new revocation information. After this instance, the request rate is exponentially decreasing, as each end entity caches the revocation information and only has to obtain a CRL at most once each period.

This paper will not restate the models as described in Cooper's papers, but the request rate graph for CRL (fig. 1), CRL distribution points (fig. 2), delta CRLs (fig. 3), and over-issued CRLs (fig. 4) will be shown for an example environment with 10000 users, 25 certificate validations per end entity per day, daily CRL updates, and a revocation rate of 10% of all certificates per year.

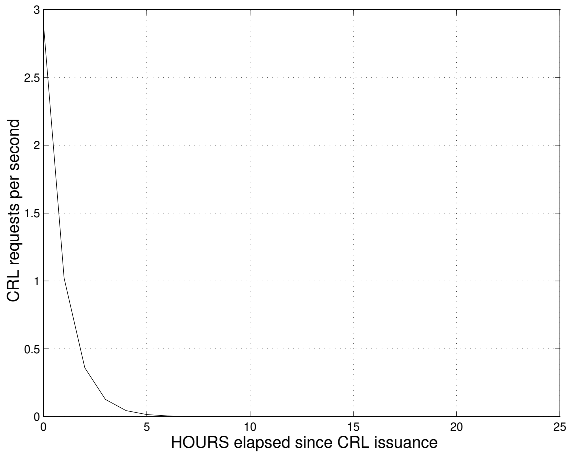

Figure 1: Request Rate following CRL issuance

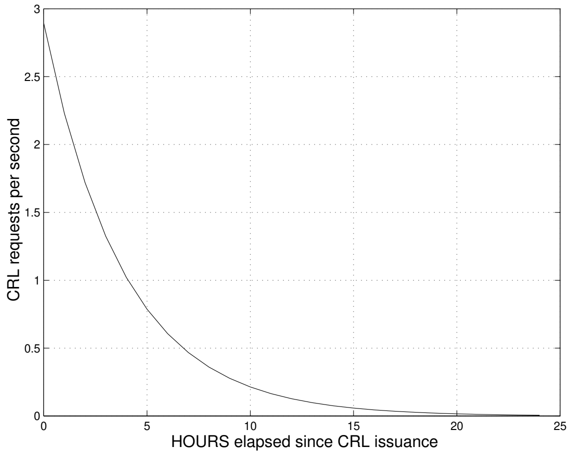

Figure 2: Request Rate following CRL issuance with 4 distribution points

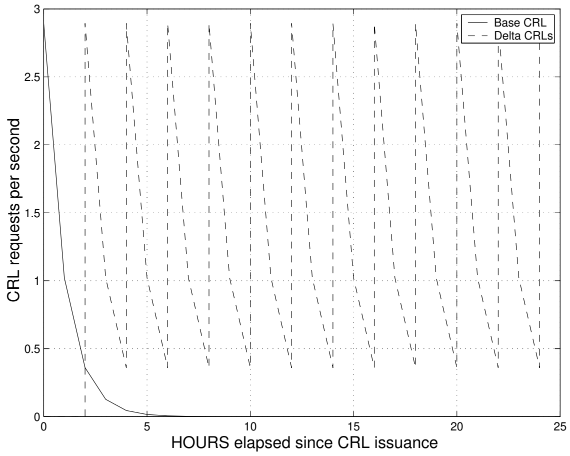

Figure 3: Request Rate following base CRL issuance with 12 Delta periods

As demonstrated in the graphs, the peak request rate for CRL is at its maximum at the time of the issuance. For CRL distribution points, the peak request rate is unchanged, but the graph decreases slower. However, each segment is smaller than a full CRL, so that both the peak and average network load is likely to be smaller for CRL distribution points than for the basic CRL scheme. For delta CRL, the graph shows that the base CRL request rate is unchanged, and that the delta CRL request rate follows the same pattern as CRLs, except with shorter intervals. To take full advantage of delta CRLs, the validity period of the base CRL can be increased. For over-issued CRLs, the graphs demonstrates that there are several peaks which are smaller than the CRL peak request rate.

6 Simulations

The work with [7] demonstrated that there is little publicly available quantitative material on revocation schemes, and some performance simulations was performed in an effort to provide some quantitative analysis. The simulation was implemented as an object-oriented, discrete-event simulation using JSIM for Java. The simulation covers CRL, delta CRL, CRL distribution points, delta CRL with distribution points, and OCSP.

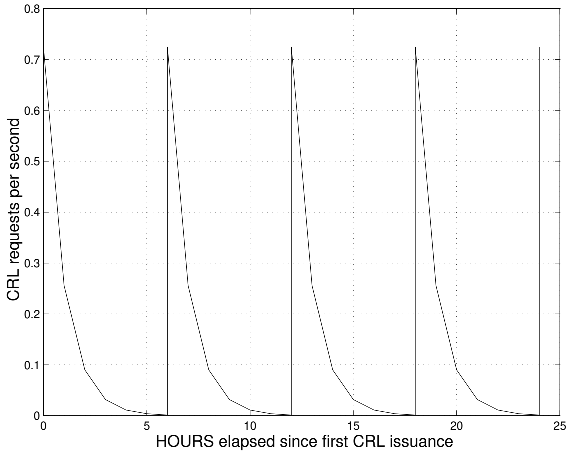

Figure 4: Request Rate following over-issued CRL issuance with 4 overlapping CRLs

6.1 Assumption and model constraints

- The certificate validity time is one year. Since the revocation rate in percent revocations is set to one year as well, the *average* percentage of revocations is r_r.
- The bandwidth of the system is set to 10Mbit ($BW = 10Mbit$). The processing load is measured in units. The processing capacity is set to 1000, so that a unit describes the processing equivalent of 1ms. If the processing load is larger than 1000 units per second, the work will queue up, causing delay. It requires 1ms to process a request for a presigned CRL. It takes 43ms to process an OCSP request.
- The size and processing requirements of delta CRLs and CRL segments are computed in the same way as for full CRLs.
- The responsibility of each Distribution Point is random and equally large as any other DP. Hence, any revocation information request is equally likely to be directed to any of the Distribution Points.
- Time periods are given in minutes, validation frequency is per day, the request rates are measured per second, whereas delay is given in milliseconds. The metrics are chosen in order to follow standard measurements and to provide more realism to the simulation variables.
- If the maximum network load is higher than the bandwidth, the network will be congested. If the average network load is higher than the bandwidth, the network is in average congested, and the scheme can not work. The same is true for the processing time, where a processing time above 1000 indicates a congested repository.

6.2 Simulation Variables

6.2.1 Input variables

The simulation accepts seven input variables: the simulation timespan in minutes, the system size in number of End Entities, the average number of validations per day, the revocation rate in percent revocations per year, the revocation information validity period in minutes, the number of distribution points/ segments, and the delta CRLs per validity period.

6.2.2 Output variables

There are five output variables from the simulator. One of the variables are only relevant if delta CRLs are used. Note that only maximum values are shown. This is done in order to show the schemes in their worst case, so that a minimal performance can be guaranteed. The output variables are: the maximum request rate, the maximum delta CRL request rate, the maximum network load, the maximum processing load, and the maximum delay.

6.3 Simulation Results

The results are presented for all the 5 simulated schemes in separate tables for different environments. Note that the simulations demonstrate increased quality of the results as the timespan of the scheme, the size of the scheme or the certificate validation rate increases. This is because a higher number of revocation requests are more likely to follow a statistical distribution.

6.3.1 Environments

Due to resource constraints, it was necessary to limit the total number of revocations during the simulation. Three environments were designed to meet this restriction with different input variables. These environments assume 4 distribution points and/ or 12 delta CRL periods for the relevant schemes. Table 2 shows the simulated environments.

Environment	Size	Validation rate	Validity Period	Revocation Rate
A	100	5/day	2 weeks	10% / year
B	1000	100/day	60 min	10% / year
C	10000	25/day	30 min	10% / year

Table 2: Simulated environments

6.3.2 Simulation runs

The simulation was performed for all five schemes for the environments in table 2. The results of the simulations are listed in table 3, 4, and 5.

Scheme	Max. req. rate	Max delta req. rate	Max nw load	Max proc load	Max delay
CRL	0.03rq/s	0rq/s	71.73b/s	0.02unit/s	1.22ms
CRL DP	0.07rq/s	0rq/s	107.47b/s	0.07unit/s	1.16ms
Delta CRL	0.05rq/s	0.05rq/s	140.64b/s	0.05unit/s	1.22ms
Delta CRL DP	0.05rq/s	0.07rq/s	118.48b/s	0.07unit/s	1.16ms
OCSP	0.07rq/s	0rq/s	66.67b/s	2.87unit/s	42.1ms

Table 3: Simulations on Environment A

Scheme	Max. req. rate	Max delta req. rate	Max nw load	Max proc load	Max delay
CRL	1.12rq/s	0rq/s	9639.07b/s	1.12unit/s	1.86ms
CRL DP	1.23rq/s	0rq/s	3986.13b/s	1.23unit/s	1.32ms
Delta CRL	1.27rq/s	1.3rq/S	10933.87b/s	1.87unit/s	1.86ms
Delta CRL DP	1.35rq/s	1.42rq/s	5232.06b/s	2.28unit/s	1.32ms
OCSP	1.47rq/s	0rq/s	1466.67b/s	63.07unit/s	42.1ms

Table 4: Simulations on Environment B

6.3.3 Analysis

These simulations show that the schemes do not demonstrate any particular pattern for a small number of requests (e.g. with timespan=60s, size=100s, and validation rate=25). However, as the number of requests

Scheme	Max. req. rate	Max delta req. rate	Max nw load	Max proc load	Max delay
CRL	3.25rq/s	0rq/s	238654b/s	3.25unit/s	8.34ms
CRL DP	3.12rq/s	0rq/s	60563.07b/s	3.12unit/s	2.94ms
Delta CRL	3.42rq/s	3.42rq/s	256487.39b/s	6.83unit/s	8.34ms
Delta CRL DP	3.77rq/s	3.77rq/s	78782.23b/s	7.53unit/s	2.94ms
OCSP	3.43	0rq/s	3433.33b/s	147.63unit/s	43.1

Table 5: Simulations on Environment C

increases, all the schemes exhibit the behavior outlined in [7].

The maximum request rate and maximum delta request rate varies slightly in the simulation runs, but the variation is not large. These variations most likely occur due to the randomness of the simulation. Obviously, the delta request rates for CRL, CRL DP, and OCSP are zero.

The maximum network load varies widely, but environment A does not seem to behave in the same way as environment B and C. Since environment A has a very small size and a low validation rate, we can not expect A to exhibit a representative behavior, so we choose to study environment B and C. For both of these environments, CRL has a high maximum network load that is significantly reduced (up to 75%) by using CRL Distribution Points. The network load for delta CRL is slightly higher (about 10%) than CRL. The reason why the difference is not larger is that the delta CRLs tend to be very small compared to the full CRLs. By using delta CRLs with distribution points, the timeliness of delta CRLs can be obtained with a reduced maximum network load not much higher than for regular CRL distribution points. OCSP has a significantly smaller maximum network load than any of the other schemes.

The simulation runs show that CRL and CRL distribution points have similar maximum processing loads. The processing loads for the delta CRL scheme and the delta CRL with DP scheme are higher (about twice as big). This is because delta CRLs have to be processed in addition to the CRLs. The processing load for OCSP is, as expected, significantly higher than any of the other schemes, because the OCSP responder has to sign every response.

Since the maximum network load never exceeds the bandwidth (10Mbit) and the maximum processing load never exceeds the processing capacity (1000 units/s), there is no delay due to queueing in these simulation runs. Consequently, the delay is determined by the transmission time (decided by revocation information size) and processing time alone. As can be seen in the tables, CRLs and delta CRLs have the same delay for the base CRLs, whereas the delay for the schemes using distribution points is slightly smaller due to reduced revocation information size. Because of the high processing load involved in digitally signing the responses, OCSP has a much higher delay.

The simulation software was implemented to plot a graph of the request rate for a simulation run. For a sufficient number of certificate revocations (as in environment B and C), these graphs qualitatively confirmed Cooper's model [11]. In particular, the maximum request rate for CRL occurs at time zero, and it decreases exponentially. Also, the graph for the CRL distribution points scheme is similar to the graph for CRL, except that it decreases slower. Each delta CRL period demonstrates the same properties as a base CRL period. For delta CRL DP, the request rate for the delta CRLs decreases slower than for regular delta CRLs. Finally, the request rate for OCSP is relatively stable over the simulation run.

7 Comparison of Schemes

As shown in the previous chapters, the different certificate revocation schemes have different properties. This section compares the different schemes based on the previous analysis and highlights the weaknesses and strengths of the different schemes.

The previous chapters have discussed certificate revocation schemes based on different methods. The CRLs are periodically issued, digitally signed lists of all revocations. The main criticism of the CRL scheme is its poor performance. Three general methods and several other techniques that improves the general performance have also been described. The scalability can be improved by segmenting the CRLs, the timeliness can be improved by issuing delta-updates of the CRLs, and the performance can be improved by lowering the peak request rate with over-issuing. Also, there are several other techniques for CRLs that improve management and performance. The CRL based schemes demonstrate high flexibility and expressiveness, and it is possible to select from or combine a number of CRL extensions.

Another class of schemes that has been presented are the on-line revocation status checking protocols, in which the end entities can request the revocation status of a particular certificate and receive an informative reply. These protocols are limited in the sense that they only provide a real-time request/ reply mechanism, rather than a full real-time revocation scheme. These protocols have to be equipped with a back-end mechanism, which might for instance be a CRL based scheme. Hence, the on-line scheme cannot be more timely than its back-end mechanism. Another consideration with the on-line protocols is that these schemes introduce another trusted party, namely the responder.

The final class of schemes that has been presented are the schemes with reduced data-structures. These schemes present the revocation information in a compressed form (e.g. as hashes in a tree), and attempt to provide the end entity with a minimal proof of revocation status. These schemes focus on avoiding the processing load involved in e.g. OCSP, as well as the communication load involved in e.g. CRL. Most of these schemes are proprietary, and only CRT is implemented and in commercial use. Also, most of these schemes (except CRT) lack the expressiveness to include information specific to a particular revocation.

8 Combining multiple Revocation Schemes

As suggested above, combining or using multiple revocation schemes can have advantageous effects on the performance, scalability, and timeliness of a revocation system at the cost of added complexity. Note, however, that redundancy of simultaneous data sources may introduce non-repudiation problems, since it may be difficult or impossible to prove what source an end entity has been using at a particular time. Consequently, an end entity may obtain revocation information from several sources, select the information that is more suitable, and then claim that this was the only revocation information available.

Cooper [11] demonstrated that a combination of over-issuing and delta CRLs or CRL segmenting can reduce the peak and average request rate of a revocation system.

The Freshest Revocation Scheme allows one form of combining schemes where the key is to discriminate the users with respect to critical and non-critical tasks. By providing high timeliness for critical tasks and less timely information for non-critical tasks, the timeliness requirements are better met and the overall performance is increased. The Freshest Revocation Scheme suggests using delta CRLs for the critical tasks, while the non-critical tasks only requires the base CRLs. The same approach could be taken with other schemes. For instance, OCSP could provide real-time revocation information, while CRLs, CRTs or some other scheme could provide less timely information. This approach is not limited to only two levels of timeliness, several schemes could be utilized.

Finally, the performance simulations described above demonstrated that the combination of delta CRLs and CRL distribution points may improve the performance of a CRL based system.

9 Selecting a revocation solution

This section briefly discusses some of the properties and parameters relevant to selecting a revocation solution. All the criteria in table 1 are relevant with respect to selecting a revocation solution. The requirements for each criteria should be evaluated for a revocation system in order to avoid an unsatisfactory or too complex revocation system. A general principle that applies to the selection is the *protection criteria*, which states that "the protection need only be as strong as the value of what it protects" [23].

An important consideration is to estimate the system parameters, in particular the system size, the certificate verification rate, the revocation information update rate, and the revocation rate. These parameters determine the resource demand for a system, and some schemes may be deemed unsuitable on this basis. Two other aspects are whether a revocation system is homogeneous or heterogeneous, and whether the system is centralized or distributed.

The following guidelines should give the reader an idea of suitable revocation solutions for a given scenario:

1. Specify the policy requirements for performance (resource usage) for the scenario in question. If high peak request rates are prohibited, CRL, CRL distribution points and delta CRLs should be avoided unless over-issuing is used. Instead, apply OCSP or take advantage of over-issuing CRLs combined with either CRL distribution points or delta CRLs. If minimizing the resource usage is crucial, CRT with piggy-backing may be an attractive option.

2. Specify the policy requirements for timeliness for the scenario in question. If no exposure to expired certificates can be allowed, a near real-time revocation solution is necessary. In this case, a delta CRL based scheme may be sufficient, but OCSP and CRT (without piggy-backing) may be more suitable. If a modest exposure to expired certificates can be accepted, delta CRL based schemes can be expected to be sufficient.

3. Specify the policy requirements for scalability for the scenario in question. If the size of the system is large or the system can be expected to grow fast, segmentation is recommended in the case of a CRL based system (i.e. CRL distribution points or enhanced CRL distribution points), as well as ARLs. In large systems, full CRL postings implies extremely high peak request rates. In this case it is important to take advantage of segmentation, delta-CRLs and/ or over-issuing. CRT can also provide high scalability.

4. Specify the policy requirements for security (2 or 3 party and non-repudiation) for the scenario in question. If a 2 party solution is required, avoid Indirect CRLs, OCSP and CRT (although an OCSP CA may be 2-party). If non-repudiation is considered an important issue, avoid schemes that allow multiple simultaneous sources of revocation information, i.e. Over-Issued CRL based schemes, the Fresh Revocation Information scheme or any combination of schemes.

5. Specify the policy requirements for standards compliance and interoperability for the scenario in question. If standards compliance or interoperability is important, CRT should be avoided, as it is a proprietary solution that is not supported by all vendors. If the system is expected to interoperate with other PKIs using CRL based schemes, ARLs are recommended.

6. Specify the policy requirements for expressiveness for the scenario in question. Most of the considered schemes support high expressiveness, but a high expressiveness requirement could prohibit some schemes (e.g. CRS, HCRS and Naor/ Nissim's scheme). Of the schemes with reduced data structures, only CRT is based on data structures that allows information about particular revocations.

7. Specify the policy requirements for scheme management (complexity and secure archiving) for the scenario in question. If it is considered important to keep the implementation and management complexity low, avoid combinations of schemes if possible. Also, keep in mind that some schemes require a separate mechanism for secure archiving of revocation information (in particular OCSP and CRT). CRL based schemes generally support secure archiving with modest modifications.

8. Specify the policy requirements regarding on-line/ off-line use for the scenario in question. If the user base is assumed to be off-line, this does not only prohibit OCSP, but it suggests a lower timeliness in general. CRL distribution points and enhanced CRL distribution points are not very suitable, but delta CRL may be suitable given relatively long validity periods. Also, CRT can be used off-line if piggybacking of freshness proofs (where time-stamped revocation information is attached to the certificate itself) is allowed. Note also that Indirect CRL may be advantageous if there is a number of small CRLs and ARLs involved.

9. If the scenario in question has a heterogeneous infrastructure or heterogeneous user patterns, multiple certificate solution may be advisable. Fresh Revocation Information is a standardized scheme supporting two different levels of timeliness. Here, the users requiring higher timeliness use the freshest revocation information pointer to request an updated delta CRL, while the other users only request the base CRL maximum once each validity period. Other combinations, like using both a CRL based scheme and OCSP could provide similar effects based on other differences in the system.

10. Whether the scenario in question is distributed or centralized is also important in deciding the revocation solution. For a highly distributed scenario, a higher number of CAs and repositories/ servers should be expected. OCSP or CRT servers can be placed where necessary, or a solution based on ARL and enhanced CRL distribution points should be considered. A highly centralized scenario is simpler to handle, so that all the considered schemes may be considered.

10 Conclusion

This paper provides a survey and a comparison of existing public key revocation schemes. It is demonstrated that the different schemes address different problems, and none of the schemes have proven generally superior. Rather than finding an optimal scheme, a designer or developer has to evaluate the requirements for the system in question, and select the revocation scheme that is most suitable. It is also demonstrated that the combination of different schemes may improve the performance of a revocation system at the cost of added management complexity as well as possible non-repudiation vulnerableness. It is the authors opinion that the existing schemes provides a good foundation for engineering revocation solutions for most environments.

References

[1] Warwick Ford and Michael S. Baum. *Secure Electronic Commerce.* Prentice Hall PTR, 1997.

[2] A. Arsenault and S. Turner. PKIX Roadmap. Internet Draft, "Work in progress, IETF PKIX working group", October 1999.

[3] Bruce Schneier. *Applied Cryptography, 2.nd ed.* John Wiley & Sons, 1995.

[4] ITU. X.509. ITU-T Recommendation, June 1997.

[5] R. Chow and T. Johnson. *Distributed Operating Systems & Algorithms.* Addison Wesley, 1997.

[6] Carlisle Adams and Steve Lloyd. *Understanding Public-Key Infrastructure: Concepts, Standards and Deployment Considerations.* Macmillan Technical Publishing, 1999.

[7] Andre Arnes. Public Key Certificate Revocation Schemes. Master's thesis, Department of Telematics, Norwegian University of Science and Technology, February 2000.

[8] Ronald L. Rivest. Can We Eliminate Certificate Revocation Lists? In *Proceedings of Financial Cryptography 1998.* Springer, February 1998.

[9] Silvio Micali. Efficient Certificate Revocation. Technical report, Massachusetts Institute of Technology, March 1996.

[10] Patrick McDaniel and Aviel Rubin. A Response to “Can We Eliminate Certificate Revocation Lists?”. Technical Report 99.8.1, AT&T Labs, February 2000.

[11] David A. Cooper. A model of certificate revocation. In *Proceedings of the Fifteenth Annual Computer Security Applications Conference*, December 1999.

[12] ITU and ISO/IEC. Final Proposed Draft Amendment on Certificate Extensions, April 1999.

[13] Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Carlisle Adams. X.509 Internet Public Key Infrastructure: Online Certificate Status Protocol. IETF RFC2560, June 1999.

[14] M. Branchaud. Caching the Online Certificate Status Protocol. Internet Draft, "Work in progress, IETF PKIX working group", April 1999.

[15] Phillip Hallam-Baker. OCSP Extensions. Internet Draft, "Work in progress, IETF PKIX working group", September 1999.

[16] Ambarish Malpani and Paul Hoffman. Simple Certificate Validation Protocol. Internet Draft, "Work in progress, IETF PKIX working group", August 1999.

[17] Carlisle Adams and Robert Zuccherato. Data Certification Server Protocols. Internet Draft, "Work in progress, IETF PKIX working group", September 1998.

[18] William Aiello, Sachin Lodha, and Rafail Ostrovsky. Fast Digital Identity Revocation. In *Advances in Cryptology - CRYPTO'98*. Springer, 1998.

[19] Jan Willemson. Certificate Revocation Paradigms. Technical report, Cybernetica Estonia, September 1999.

[20] Paul Kocher. A Quick Introduction to Certificate Revocation Trees(CRTs). Technical report, ValiCert, 1999.

[21] Moni Naor and Kobbi Nissim. Certificate Revocation and Certificate Update. In *Proceedings of the 7th USENIX Security Symposium*, 1998.

[22] David A. Cooper. A more efficient use of Delta-CRLs. Draft for IEEE Symposium on security and privay, May 2000.

[23] D. Kahn. *The Codebreakers.* Macmillan Publishing Co., 1967.