Public Key Certificate Revocation Schemes

by

André Årnes

A thesis submitted to the
Department of Telematics,
Norwegian University of Science and Technology
in conformity with the requirements for
the degree of Sivilingeniør

Queen's University
Kingston, Ontario, Canada
February 2000

Abstract

Public Key Infrastructure (PKI) is increasingly popular as the security infrastructure of choice in corporate information systems and Internet applications. In particular, certificates are used to distribute public keys associated with the identity of its owner.

One of the main concerns associated with PKI is that mechanisms for revocation of certificates are required. Certificate Revocation is the action of declaring a certificate invalid before its validity period is at an end. Investigations of PKI have demonstrated that the traditional means of certificate revocation are the most expensive part of a PKI system. In response to this problem, several different schemes and protocols have been suggested to provide more efficient certificate revocation.

This thesis provides a survey and an analysis of existing schemes for public key certificate revocation. The analysis includes the traditional certificate revocation lists, protocols that provide on-line certificate revocation, as well as revocation systems with reduced data structures. Some techniques for improving the existing schemes are also suggested. Based on the analysis, the different schemes are compared in order to highlight advantages and disadvantages. A guideline for using this information in selecting a revocation solution is developed and applied to different example scenarios.

This thesis is intended to provide a comprehensive background in certificate revocation as a basis for further research and development in the area, as well as a guide for PKI system developers in need of a certificate revocation system.

Acknowledgments

I want to thank Professor Henk Meijer at the Department of Computing and Information Science at Queen's University for supervising the writing of this thesis and providing essential guidance and feedback. Our weekly meetings have proved very beneficial in the progress of this thesis.

Also, I thank Steve Lloyd and Mike Just at Entrust Technologies for providing an opportunity to work with representatives from the leading industry in the field. They have shown high interest in the project, and they have provided valuable feedback based on their experience and insight in both standardization work and implementations.

I acknowledge Entrust Technologies and CITO (Communications and Information Technology Ontario) for the financial support of this project.

Finally, I appreciate the assistance of Associate Professor Svein J. Knapskog at the Department of Telematics at the Norwegian University of Science and Technology. As my home university supervisor, he has been very helpful in providing help with administrative issues, as well as given general advise on my thesis and its progression.

Contents

List of Tables

List of Figures

List of Symbols

2.1.1 K - Symmetric encryption key

2.1.1 M - Plaintext message

2.1.1 C - Encrypted messages (Ciphertext)

2.1.1 A_s - Symmetric encryption algorithm

2.1.1 A_s^{-1} - Symmetric decryption algorithm

2.1.2 E - Private key

2.1.2 D - Public key

2.1.2 M_1, M_2 - Plaintext messages

2.1.2 C_1, C_2 - Encrypted messages (Ciphertexts)

2.1.2 A_e - Public key encryption algorithm

2.1.2 A_d - Public key decryption algorithm

3.2.2 i - Environment identifier

3.2.2 N_i - Number of End-Entities in environment i

3.2.2 v_i - Validation rate in environment i

3.2.2 $r_{r,i}$ - Revocation rate in environment i

3.2.2 $r_{u,i}$ - Revocation information update rate in environment i

4.1.1 t - Time elapsed since last issuance

4.1.1 $P(t)$ - Probability of an End-Entity performing its first validation within the interval $[t...t+dt]$

4.1.1 $T(t)$ - Total number of requests within the interval $[t...t+dt]$

4.1.1 $R(t)$ - Request rate for CRLs at time t

4.2 s - Number of Distribution Points (segments)

4.2 $R_s(t)$ - Total request rate for all segments

4.3.1$R_b(t)$ - Request rate for base CRLs

4.3.1$R_\Delta(t)$ - Request rate for delta CRLs

4.4.1 O - Number of concurrent CRLs

4.4.1 l - Validity time of CRLs

4.4.1 $R_o(t)$ - Request rate during over-issued interval

4.5.1 $R_{so,1}(t)$ - Request rate for over-issued segments

4.5.1 $R_{so,2}(t)$ - Request rate for staggered segments

4.6.1 w - Window size for delta CRLs

4.6.1 $R_w(t)$ - Request rate for base CRLs with sliding window delta CRLs

4.7.1 $R_{o\Delta}(t)$ - Request rate for overissued delta CRLs

4.7.1 $R_{ob}(t)$ - Request rate for base CRLs with overissued delta CRLs

6.1 N - The 100 bit value representing revoked included in a CRS certificate

6.1 Y - The 100 bit value representing non-revoked included in a CRS certificate

6.1 F - A cryptographic one-way hash function

6.1 N_0 - The 100 bit secret value that indicates revocation if disclosed

6.1 Y_0 - The 100 bit secret value that indicates non-revocation

6.1 c_n - A bit in a bit string representing certificate n

6.1 i - Time elapsed in day since certificate issuance

6.1 V - The 100 bit value provided by the CRS Directory

6.1 n - A certificate identity serial number

6.2 l - There are 2^l certificates in HCSR

6.2 ϕ - The empty root of the HCRS tree

6.2 F - A cryptographic one-way hash function

6.2 D - The validity time of the certificates

6.2 r - Random initial value for a node

6.2 v - A leaf node/ certificate

6.2 i - Time elapsed since issuance

6.2 R_i - The list of revoked certificates up to day i

List of Abbreviations

ARL: Authority Revocation List

CA: Certification Authority

CITO: Communications and Information Technology Ontario

CRL: Certificate Revocation List

CRS: Certificate Revocation Status

CRT: Certificate Revocation Tree

DCS: Data Certification Server

DN: Distinguished Name

DP: Distribution Point

EE: End Entity

FPDAM: Final Proposed Draft Amendment

HCRS: Hierarchical Certificate Revocation Scheme

IEC: International Electrotechnical Commission

IETF: Internet Engineering Task Force

ISO: International Organization for Standardization

ITU: International Telecommunication Union

OCSP: Online Certificate Status Protocol

OCSP-X: Online Certificate Status Protocol Extensions

PEM: Privacy Enhanced Mail

PKI: Public Key Infrastructure

PKIX: Public Key Infrastructure (X.509)

RA: Registration Authority

SCVP: Simple Certificate Verification Protocol

Chapter 1

Introduction

1.1 Background

This thesis was written at the Department of Computing and Information Science at Queen's University, and it serves as the diploma thesis for the degree of Sivilingeniør at the Department of Telematics at the Norwegian University of Science and Technology. The work was done in cooperation with Entrust Technologies in Ottawa. Entrust Technologies provides PKI products and services to a wide range of customers, and they have significant interest in the area of certificates and certificate revocation.

1.2 Objectives

The first objective of this thesis is to develop a suitable framework for comparison of public key certificate revocation schemes, and to perform a comprehensive survey and analysis of existing revocation schemes. Furthermore, it is an objective to suggest a guideline for selecting revocation solutions and demonstrate this through different example scenarios. It is also an objective to point out possible improvements in the existing certificate revocation schemes.

1.3 About this Thesis

This thesis includes the following chapters:

Chapter 2, Introduction to PKI and Certificate Revocation, provides an overview of public key infrastructure and certificates, and introduces the reader to the problem of certificate revocation.

Chapter 3, A Framework for Analysis of Revocation Schemes, provides a reference model, some reference environments, as well as evaluation criteria for the comparative analysis.

Chapter 4, Certificate Revocation Lists with Extensions, describes and analyzes the most common and traditional means of certificate revocation, as well as its extensions and modifications.

Chapter 5, On-line Revocation Status Checking Protocols, describes and analyzes a number of protocols that are capable of providing on-line certificate revocation information.

Chapter 6, Revocation Schemes with Reduced Data-Structures, describes and analyzes a number of certificate revocation schemes that take advantage of reduced data-structures in order to provide more efficient systems.

Chapter 7, Selected Other Certificate Revocation Mechanisms, describes some additional techniques related to certificate revocation.

Chapter 8, Suggested Improvements for Existing Schemes, provides some techniques for improving the efficiency of some of the schemes in chapter 4 and 5.

Chapter 9, Comparison of Existing Certificate Revocation Schemes, utilizes the analysis from chapter 4, 5 and 6, and provides a summarized comparison of the schemes.

Chapter 10, Selecting Certificate Revocation Solutions, provides certificate revocation solutions for a number of scenarios based on the comparative analysis.

Chapter 11, Conclusion, concludes the work in the previous chapters and highlights the most important points from the thesis.

1.4 Contributions

This thesis provides a thorough survey and a comparative analysis of certificate revocation schemes. A comprehensive survey of certificate revocation schemes was not publicly available elsewhere when the thesis was written.

In order to facilitate the comparative analysis, a framework for analysis of certificate revocation schemes was developed. This framework is intended to define the reference model to be used, relevant reference environments, as well as the reference criteria for the comparative analysis.

A number of suggestions for improving existing certificate revocation schemes was developed during the work with the thesis.

Finally, the comparative analysis is summarized and utilized in developing a guideline for selecting revocation solutions, which is demonstrated for some example scenarios. This highlights the considerations that have to be taken in designing a certificate revocation system, and provides examples on how to do it.

Chapter 2

Introduction to PKI and Certificate Revocation

This chapter provides a brief introduction to cryptography, public key cryptography and certificate revocation. The introduction is intended for the reader that does not have sufficient background to read this thesis. For more thorough explanations, see "Applied Cryptography" by Bruce Schneier [1] for a comprehensive approach, "Cryptography - Theory and Practice" by Douglas R. Stinson [2] for a formal approach, "Security in Computing" by Charles P. Pfleeger [3] for an introductory approach or "The Codebreakers" [4] by D. Kahn for a historic approach.

2.1 Cryptography

This section explains some of the basic concepts in cryptography. These concepts are essential to the material discussed later in the book, but it can be skipped by readers that are familiar with cryptography.

2.1.1 Symmetric Key Encryption

The traditional way of encrypting messages is called symmetric key encryption. This technique assumes that the sender and the recipient share the knowledge of a secret key K and an encryption algorithm A_s. The message M is *encrypted* by applying the algorithm A_s to the message M using the key K:

$$C = A_{s,K}(M) \tag{2.1}$$

The secret message C is *decrypted* by applying the inverse algorithm A_s^{-1} to the secret message C with the key K:

$$M = A_{s,K}^{-1}(C) \tag{2.2}$$

2.1.2 Public Key Encryption

In a public key encryption scheme [5], each user has a secret key D, the *private key*, and a non-secret key E, the *public key*. A plain-text message M_1 that is encrypted with the public key E can only be decrypted with the private key D. The cryptographic algorithm A_e is used for encryption, and A_d is used for decryption. In some public key encryption schemes, the same algorithm can be used for both encryption and decryption (i.e. $A_e = A_D$). The encryption and decryption is performed as follows:

$$C_1 = A_{e,E}(M_1) \tag{2.3}$$

$$M_1 = A_{d,D}(C_1) \tag{2.4}$$

Similarly, a message that is encrypted with the private key D can only be decrypted with the public key E:

$$C_2 = A_{e,D}(M_2) \tag{2.5}$$

$$M_2 = A_{d,E}(C_2) \tag{2.6}$$

2.1.3 Digital Signatures

A digital signature [3] emulates a real, physical signature by generating a digital proof that only the creator/ sender of a message can make, but everyone can identify as belonging to the creator. Public key encryption systems are well suited to digital signatures, as an encryption under the private key of the user serves as a signature that only the owner of the private key can create, but everyone with the public key can verify.

2.1.4 Digital Hash Functions

A one-way hash function is a function that takes a variable length input (*pre-image*), and computes a fixed-length output string called the *hash value*, *digest* or *check value* [1]. The objective of this function is to produce a value that indicates whether a candidate pre-image is likely to be the same as the real pre-image. Due to the fact that the hash value in general is smaller than the pre-image, the function is many-to-one, but a good one-way hash function makes it hard to generate two pre-images with the same hash-value.

2.2 Public Key Infrastructure (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 [6]. PKIX is an Internet Engineering Task Force (IETF) effort to standardize such a PKI.

2.2.1 Public Key Certificates

A public key certificate is a user's public key, identity and some other information digitally signed by a trusted authority [1]. Certificates are used to prevent the substitution of one forged key for a legitimate key. The 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.

In the terminology of PKIX, a public key certificate is defined to be "a data structure containing the public key of an End-Entity and some other information, which is digitally signed with the private key of the CA which issued it" [6].

2.2.2 X.509

One of the most popular standards for public key certificates is included in the ITU (International Telecommunication Union) X.509 standard. The X.509 standard [7, 8] provides an authentication framework with public key certificate distribution to the X.500 directory standards series [9]. The X.509 standard specifies how authentication information is created and stored by a directory server, describes how user authentication information can be obtained from a directory, and also defines methods in which applications may use the authentication information to perform authentication [10]. X.509 also defines a simple, password based authentication mechanism, but this mechanism is irrelevant to this text.

In the X.509 structure for public key certificates, each user has a distinguished name (DN) assigned by a trusted Certification Authority (CA) [1]. The CA issues certificates signed under the CA's private key. A X.509 certificate includes a version number that identifies the message format, a serial number that is unique within the CA's domain, an algorithm identifier, the name of the CA, the period of validity for the certificate, the name of the user, the user's public key, and the CA's signature. When a user A wishes to communicate with a user B, A gets B's certificate from a directory (or by another method) and verifies its authenticity with the CA's public key. The users are expected to know the CA's public key in advance.

2.2.3 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 obtain this public key, the user have to obtain a certificate for the other user's CA from another 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 [8].

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 obtained.

There are three main methods for creating systems with trust-relationships between CAs [11]. The first model is the *hierarchical model*, in which there is a tree of CAs with one single *root certification node*. The second model is based on *cross certifications*, where all CA certifications are bilateral. The third model is the *hybrid model*, where both hierarchical certifications as well as cross certifications. The third model is suitable for creating trust-relationships between two different organizations that do not belong in the same hierarchy.

2.2.4 Certificate Validation

Certificate Validation is the process of verifying that a certificate is valid. This process checks that the certificate is within its *validity period*, and performs an *integrity check* based on the digital signature of the CA. Also, if applicable, a *certification path validation* is performed, and the *trust anchors* or root nodes have to be recognized and trusted. The verification process also checks whether there is a consistency with intended key usage and certificate policy restrictions. Finally, the process checks the *revocation status* to ensure that the certificate in question has not been revoked [11].

2.3 Certificate Revocation

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 [8] suspected key compromise, detected key compromise, change of subject name and change of relationship between the subject and the CA.

There are several approaches to certificate revocation. The traditional approach is to periodically publish a *certificate revocation list*, which is a list of all revoked certificates within a domain. The main problem with this approach is that the list can grow large for large domains, and the network load involved in all clients downloading the list from all clients can become unacceptable.

It is possible to cache the lists at the clients, but the frequency of list updates is limited in such a way that the list obtained may not always be fresh. In order to provide higher efficiency with the certificate revocation lists, several modifications have been suggested.

A different, also standardized approach is to provide an on-line server and use protocols for obtaining on-line revocation information. In this case, a client issues a request for every encountered certificate instead of obtaining a full revocation lists. The main problem with this approach is that each response has to be digitally signed, so that there is a significant processing and network overload for each response.

An issue with the methods described above is that they publish the full revocation information for each certificate. Micali [12] first suggested a system where the information is sufficient to prove the revocation status of a particular certificate, without publishing the full revocation information for each certificate. Also, it was implicit in Micali's model that a client requested the revocation information from the server for each certificate rather than once each period. Several systems have taken similar approaches in order to optimize the size of the revocation information and reduce network load. Some of the main problems with most of these systems is that they are theoretical models that have not been standardized and that they provide little flexibility since they do not provide the per certificate revocation information used in the other systems.

This thesis performs a comparative analysis of the systems described above, and highlights the strengths and weaknesses of the different systems. It is not the intention of this text to solve the problems inherent in the systems, but to provide a framework for selecting the right system for the right environment. Also, it is suggested that different systems can be combined in order to address different problems. The combination of schemes is particularly attractive in heterogeneous systems.

Chapter 3

A Framework for Analysis of Revocation Schemes

In order to compare the different certificate revocation schemes from a common stand-point, it is necessary to define a framework for analysis and comparison. The schemes themselves are presented with different models and terminologies that do not necessarily compare directly. This framework consists of the following elements:

- A *Reference Model* that all schemes can be superimposed on.
- *Reference Environments* on which each scheme can be evaluated.
- Common *Criteria* for analyzing every scheme.

3.1 Reference Model

In general, the literature on public key certificate revocation is based on two different models. The first model is commonly used in academic papers [12], and it is referred to as the simplified reference model in this text. The second model is used in the IETF standardization framework [6], and it is referred to as the IETF model in this text.

3.1.1 Simplified Reference Model

In the simplified reference model (figure 3.1), the CA issues certificates to the users and certificate information (e.g. CRLs) to the Directory. The Directory is, in turn, responsible for the distribution of certificate information to the users of the services. The CA does not issue certificate information directly to the users. The following entities are involved [12]:

- CA (Certification Authority): A trusted agent that issues and revokes certificates.
- Directory: A non-trusted (or less trusted) agent that receives certificate information from the CAs and handles user queries.
- User: Non-trusted agents who receive certificates from the Directory, produce and verify digital signatures, and request certificate information from the Directory.

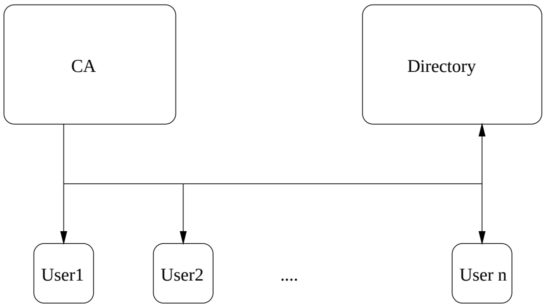

Figure 3.1: Simplified Reference Model

3.1.2 IETF Model

The IETF model is similar to the simplified reference model, but it is more detailed and uses different terms. In this model (figure 3.2), the following entities are involved [6]:

- End-Entity (EE): User of PKI certificates and/ or end user system that is the subject of a certificate. An End-Entity is often referred to as a user, a client or a relying party.
- CA (Certification Authority): Authority that issues and revokes certificates.
- RA (Registration authority): An optional system to which a CA can delegate certain management functions.

- Repository: A system or collection of distributed systems that store certificates and CRLs and serves as a means of distributing these certificates and CRLs to End-Entities

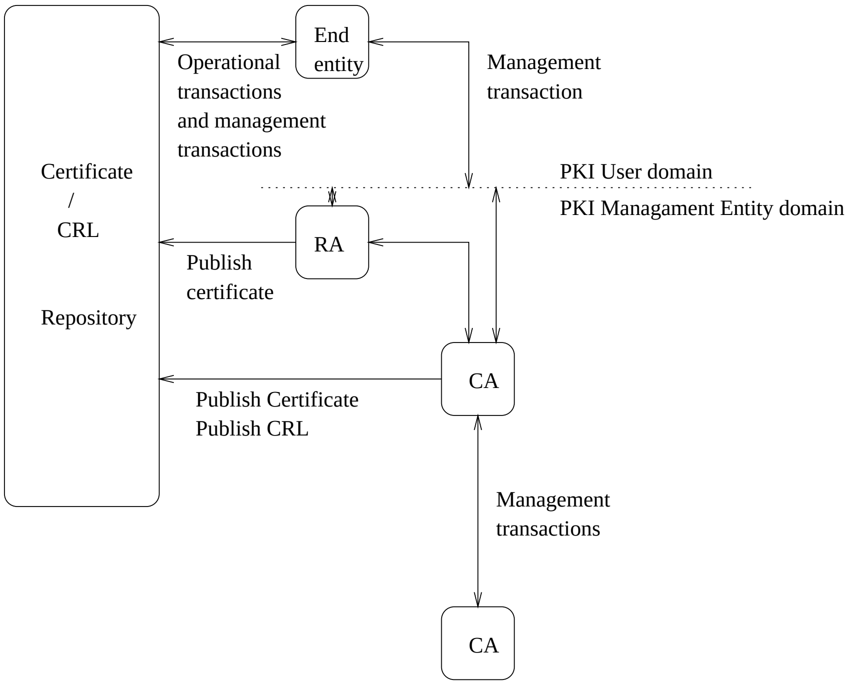

Figure 3.2: IETF Model

3.1.3 Selected Reference Model

For the purpose of a comparative analysis, it is important that the reference model is flexible enough to be able to express different schemes on similar premises. Furthermore, the model must be capable of expressing both a centralized model (with one central certification authority and one central certificate revocation list) and a distributed model (where the number of certificate authorities and certification revocation lists are arbitrary).

The IETF model is more generic, as it models *End-Entities* rather than *Users* and *a Repository* rather than *a Directory*. The term Repository is more general, as the Directory usually is associated with the X.500 Directory. Furthermore, the IETF model also models the RA and communication with other CAs (potentially supporting distributed services). Note also that this allows the model in figure 3.1 to be modeled as a subset of the model in figure 3.2.

Based on this comparison of the models, the IETF model is used as a base for further analysis of Certificate Revocation Schemes. It is, however, necessary to extend this model to include on-line *Certificate Revocation Servers* to support protocols for on-line revocation status checking protocols. The modified reference model for this analysis is thus as shown in figure 3.3.

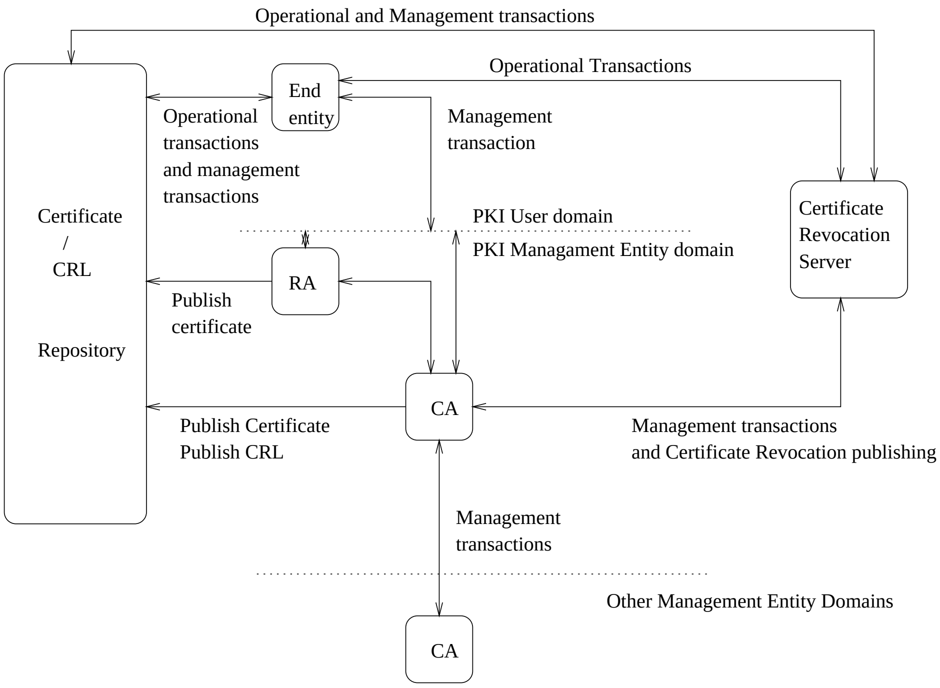

Figure 3.3: Reference Model

3.2 Reference Environments

Much of the work on certificate revocation is theoretically or conceptually based, and there is little statistical material available. There are, however, a few analysises of theoretically assumed PKI system environments. This section presents the most frequently cited environment, as well as three different constructed environments for the purpose of the analysis in this thesis.

3.2.1 The MITRE Environment

A simple analysis of the costs of the CRL scheme was published by Micali [12], and recently referred to in a paper by Willemson [13]. The analysis was based on assumptions from the US Federal PKI MITRE report from 1994 [14], and they are presented here in lack of more updated parameters. These parameters are originally assumed for *version 1* CRLs without any extensions. There are multiple CAs, each responsible for a limited End-Entity domain each, although each revocation list is distributed to all users (through the CAs). These are the revocation parameters from the MITRE report [14]:

1. 3 Million users
2. Each CA serves 30000 users
3. 10% of all certificates are revoked during one year; 5% from key compromise and 5% because of End-Entities loosing associations with CAs
4. CRLs are sent out bi-weekly
5. There are 4 different scenarios for how many certificate revocation requests each End-Entity issues a day: 5, 25, 50, 100
6. The communication costs are 2 cents per kilobyte

The analysis described above fails to see that the CRL can be cached on the End-Entity, and assumes that each End-Entity must re-obtain the CRL for each new certificate that is used. With a cached CRL on the End-Entity, assumption 5 is only relevant for calculating the *request rates* in a CRL validity period. Furthermore, the following analysis will not consider the costs of the systems in dollars, since these prices are highly fluctuating. For this reason, assumption 6 is considered irrelevant (and outdated). The PKI market today also demand increasingly timely systems, such that a bi-weekly CRL update is not sufficient.

3.2.2 Selected Reference Environments

For the purpose of the comparative analysis of revocation schemes, it is useful to define a number of environments under which the properties of the different schemes can be studied. In particular, the size of the environment is varied in order to study the scalability (section 3.3.2) and timeliness (section 3.3.2) of the different schemes. To simplify the analysis, it is assumed that each environment has only one certificate revocation information issuer that serves all End-Entities rather than having multiple issuers (CAs). In this way, all revocations in the system are available to or delivered to each End-Entity. This issue will only be brought up where CA certification and cross-certification are particularly relevant.

Now let N_i represent the number of End-Entities in environment i, v_i represent the validation rate in environment i, $r_{u,i}$ represent the certificate revocation information update rate in environment i, and $r_{r,i}$ represent the revocation rate in environment i. Environments are defined such that $N_{small} = 100$, $N_{medium} = 10000$, and $N_{large} = 1000000$. The validation rate of the environments are selected from the estimates in the MITRE environments to be $v_{low} = 5$, $v_{medium} = 25$, and $v_{high} = 100$. The certificate information update rates are defined such that $r_{u,low} = biweekly$, $r_{u,medium} = daily$, and $r_{u,high} = hourly$. The revocation rate is kept constant at $r_{r,i} = 10\%/year$. These parameters are selected to be reasonably descriptive of different types of systems and user patterns with different demands for scalability and timeliness.

Table 3.1 shows the environments E_1, E_2 and E_3 based on the above variables. Note that this is not an exhaustive representation of the possible environments. Only three environments have been defined. Environment 1 has minimal complexity, environment 2 has medium complexity, and environment 3 has maximal complexity within the given variables. This allows an analysis of the most extreme and the medium case without having to iterate through all $3^3 = 27$ combinations. These environments are in particular utilized to demonstrate request rates in sections 4.1 to 4.7.

Environment (E_i)	Size (N_i)	Val.rate (v_i)	Update rate ($r_{u,i}$)	Rev.rate ($r_{r,i}$)
1	100	5/ day	Bi-weekly	10%/ year
2	10000	25/ day	Daily	10%/ year
3	1000000	100/ day	Hourly	10%/ year

Table 3.1: Reference Environments

3.3 Evaluation Criteria

In order to perform a comparative analysis on the different certificate revocation schemes, it is necessary to define the criteria involved. The list of criteria should be a comprehensive collection of all relevant criteria that are important for choosing a particular scheme. Such a list includes both qualitative and quantitative elements, and some criteria can be seen as categories that expands to several more specific criteria.

3.3.1 General Criteria

The criteria presented in this section is based on a list of general criteria from [11]:

- Timeliness: Is the information supplied in a timely fashion as dictated by policy?
- Performance: Is the scheme viable from a network bandwidth and component-level perspective?
- Scalability: Does the scheme itself scale?
- Integrity: Is the revocation information from a trusted source, and is the integrity maintained?
- Standards-based: Is the solution based on industry accepted standards?

Note that timeliness, performance and scalability are highly interrelated. In particular, both the timeliness and the scalability are dependent on the performance of a scheme. If a scheme causes a processing or network bottleneck, timeliness and/ or scalability necessarily degrades as well.

3.3.2 Selected Evaluation Criteria

The performance properties of a scheme is simpler to analyze and measure systematically than timeliness and scalability, so performance is studied first in this treatment. Also, *security* is substituted for the less general criterion integrity in order to include other issues like trust and non-repudiation. In addition to the criteria above, the criteria *expressiveness*, *scheme management* and *on-line vs. off-line* are introduced. Also, *standards compliance* is substituted for *standards-based*, as standards compliance is a more commonly used term.

Performance

The cost of using a certificate revocation scheme is a critical consideration when choosing certificate revocation schemes. Similarly, delays in a system can be an important factor. Certificate revocation information can, based on the system used, have high communication or processing demand, as well as long delay periods due to communication or processing.

These are the specific parameters that are considered:

- Peak Load and Peak Request Rate: In the case where a reliable, guaranteed performance is expected, it is necessary to look at the peak load and the peak request rate. The peak request period is interesting because it represents the time where the resource demands and consequently the delays are the highest. The quantitative measures of these parameters are maximum bit-rate and maximum hit-rate.

- Average Load and Average Request Rate: The average load and request rate can be used in determining the expected costs of a scheme, but the peak values are more interesting in designing a system where a guaranteed performance is expected at all times. The quantitative measures of these parameters are average bit-rate and average hit-rate.

- Load distribution: When designing a public key infrastructure, it is important to know how the load and request rates behave in different parts of the system. For the reference model as shown in figure 3.3, the load distribution can be represented by CA processing load, CA to Repository/ Server communications, Repository/ Server processing load, Repository/ Server to End-Entity communications, and finally End-Entity processing load. The network and processing load distributions can be measured by finding the respective loads at different parts of the system.

- Delay: For the End-Entities, the delay is an important aspect of a scheme. The delay is related to the peak load and request rate, and can be measured by maximum and average waiting time, where maximum waiting time is the most relevant factor in selecting and implementing a certificate revocation scheme.

- Size: For some End-Entities, the storage capacity may be too low to store/ cache large amount of revocation information. Also, the network load is partially determined by the revocation information size. The measure used for this parameter is maximum and average size in bytes. Again, the maximum size is most interesting for determining the bottlenecks and maximum delays in a system.

Timeliness

An issue with several certificate revocation schemes is that they allow the use of revocation information for a certain period of time after each revocation information update. There is an increasing demand for more timely information, as users of PKI based systems expect a revocation decision to be implemented instantly, or at least sufficiently fast. Note that timeliness is also referred to as *freshness*. A measure for the timeliness in a system is the maximal time between the actual revocation and when the revocation information is made available to the End-Entities.

Scalability

Scalability is a criteria that is highly related to both performance and timeliness. Some of the certificate revocation schemes are suitable for handling domains with a small number of End-Entities, whereas the performance is unacceptable for larger domains. On the other hand, it is unnecessary to deploy complex schemes when a simpler scheme is sufficient. For a proper choice of a certificate revocation scheme, it is necessary to determine the size and growth potential of the domain. The scalability of a scheme is best demonstrated by finding the complexity of the scheme in terms of communication, size and processing.

Security

Certificate revocation is a part of the architecture intended to provide security to an End-Entity domain. It is important that the security properties of a certificate revocation scheme are no weaker than the rest of this architecture. In particular, it should protect against unauthorized *creation*, *manipulations* or *removal* of certificate revocations. This is usually implemented by having the revocation information issuer sign the revocation information, hence ensuring authenticity and integrity.

Optionally, the scheme could also provide data encryption to provide confidentiality of the certificate revocations. This is, in general, not preferred, since the revocation information must be known to every End-Entity that tries to use a revoked certificate. Note that this approach may change as PKI and certificate revocation mature. It can be expected that a revocation information provider may charge for its revocation information, and that authentication and encryption mechanisms can be deployed to protect the information.

Also, some schemes address the problem that the Repository should be able to prove that a non-revoked, non-expired certificate is not issued by a given CA if it was indeed not issued [12]. This is, however, not a traditional responsibility of certificate revocation schemes, and the importance of this is arguable.

By identifying the entities in the scheme in question, it is possible to determine a minimum level of trust. In some schemes, all revocation information is digitally signed by the CA, providing guaranteed integrity and authenticity of the list. These schemes are frequently referred to as 2-party schemes. In other schemes, the Repository/ Server signs the revocation information provided to the End-Entity. In this case, both the CA and the Repository/ Server must be *trusted.* These schemes are frequently referred to as 3-party schemes.

Another security issue that is frequently discussed in the PKI community is non-repudiation. Non-repudiation ensures the availability of sufficient evidence to support a fast resolution of disagreements based on repudiation [8]. Repudiation is a threat, in which a party in a transaction later denies that the transaction took place. Repudiation is a potential problem for certificate revocation when multiple simultaneous sources of revocation information is available. An End-Entity may obtain revocation information from several sources, select the that that is more suitable and then claim that this was the only revocation information available to this End-Entity at the time.

Standards Compliance

The price, availability and interoperability of a certificate revocation scheme heavily depend on whether the scheme is standardized or proprietary. If a system is standardized, the specifications are readily available, and interoperability with other systems is simplified. If the system is patented, license fees are potentially an additional cost, and the scheme in question is less likely to be compatible with other products.

Expressiveness

Depending on the usage of the system, it may be preferable to include information regarding a specific revocation (e.g. revocation date, revocation reason, etc.). This is easily implemented on a per entry basis in full revocation lists, but it is not necessarily available in the systems that reduce the size of the revocation information in order to provide more efficient schemes.

Scheme Management

The extent of management for a given scheme is also important in choosing a scheme for a certain environment. Important issues are the number of servers (CAs, Repositories and Revocation Servers) and the amount of management (installations, maintenance, supervision) on all involved entities. Furthermore it is an advantage if the mechanisms required can be easily integrated into existing software. A critical factor is also whether the scheme is fully automated and whether separate mechanisms for *secure archiving* is necessary. Some mechanisms (like certificate revocation lists) lend itself to archiving, whereas other mechanisms lack the expressiveness and structure to serve as a long term archive. In the last case, a separate mechanism may be necessary in order to provide an archive for judicial and other reasons.

On-line vs. Off-line

One of the advantages and initial purposes of certificates is that they can be used in an off-line environment. In early non-certificate systems, each End-Entity was on-line, and revocation status could be checked directly. The recent trend is, however, moving back towards allowing on-line End-Entities. Some End-Entities, like Internet connected computers, can always be assumed to be on-line. Other, more restricted units, are in general off-line (e.g. Palm Pilots). Another consideration is that the storage-, communication-, and processing- load can be offloaded to more powerful servers in the case of End-Entities with limited resources. This opens up for certificate revocation schemes that are designed for an on-line environment in addition to combined systems that could choose either solution based on price and availability.

Note that a system that is considered off-line cannot be more timely than the time between it's connections. The timeliness demands of a scheme is a policy-based issue, and off-line schemes may conflict with the policy timeliness requirements. The number of connections per revocation update period can be used as a quantitative measure of "off-lineness". The number of connections is a reasonable measure because each connection can be assumed to have a cost as well as a noticeable delay. In this way, one connection per revocation update period (as dictated by policy) is optimal from an off-line perspective, whereas one connection per certificate (independent of the revocation update period) is more suitable for an on-line environment.

3.3.3 Summary of Selected Revocation Criteria

Table 3.2 summarizes the criteria above and lists the most important keyword for each criteria.

Criteria	Keywords
Performance	Peak Load/ Request Rate, Load Distribution, Delay, Size
Timeliness	Max. time between the revocation and revocation information distribution
Scalability	Related to performance and timeliness, demonstrated by scheme complexity
Security	Trusted entities and integrity of revocation information, non-repudiation
Standards Compliance	Standardized or proprietary, theoretical or implemented
Expressiveness	Level of granularity of information on a per certificate basis
Scheme Management	Automated, secure archiving, amount of management, complexity
On-line vs. Off-line	Frequency of connections

Table 3.2: Evaluation Criteria

Chapter 4

Certificate Revocation Lists with Extensions

The most standardized and developed certificate revocation scheme is the Certificate Revocation Lists with its many modifications and extensions for improving its performance. This chapter provides an analysis of the basic CRL scheme with its extensions and alternative techniques.

4.1 Certificate Revocation Lists (CRLs)

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 [7], and Version 2 CRL is the current version. Also, there is a CRL standard for use with PEM (Privacy Enhanced Mail) called PEM CRL. The standard discussed in this chapter is the 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 to costly [15, 12]. 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 viewed by many as a suitable basis for many (but not all) certificate revocation systems [16].

Figure 4.1 shows the basic syntax of a Version 2 CRL Structure, whereas figure 4.2 shows the syntax of a single Version 2 CRL Revoked Certificate entry. The *Version* field in figure 4.1 indicates the version of the CRL. The *Signature Algorithm* field provides the identifier of the digital signature algorithm used to produce the CRL signature. The *Issuer* field provides the name of the CA that issued the CRL. The *This Update* field indicates the CRL issue time, while *Next Update* indicates the next time of CRL issuance. The *Revoked Certificate List* is a list of revoked certificates specified by their unique serial numbers and their revocation date on a per entry basis, as shown in figure 4.2. Extensions are available both as *CRL extensions* for the whole CRL and as *CRL entry extensions* for a single CRL entry.

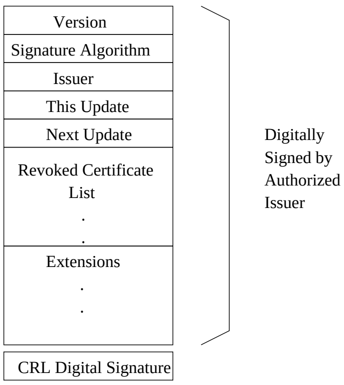

Figure 4.1: Version 2 CRL Structure

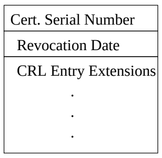

Figure 4.2: Version 2 Revoked Certificate Entry

Figure 4.3 shows a simple CRL example system with a CA, a Repository and 3 End-Entities

(EEs). First, the CA publishes the CRL to the Repository, then End-Entity 1 requests, obtains and caches the CRL from the Repository. At a later time, End-Entity 2 and 3 may request the CRL, and the next CRL is typically published after one CRL validity period.

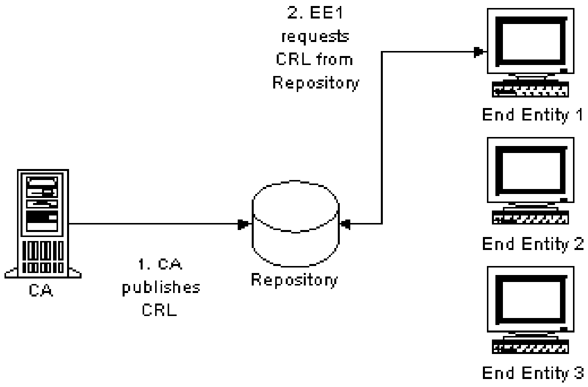

Figure 4.3: Example CRL system

CRL Extensions

Version 2 CRL introduced the concept of extensions to CRLs. An extension is defined on a per revoked certificate entry basis or on a per CRL basis. Extensions are furthermore flagged as either critical or non-critical. It is optional whether non-critical extensions need to be processed, whereas critical extensions always should be processed and understood. Note that the extensions and the critical flag use the same syntax as for X.509 certificates. The available CRL extensions are listed in table 4.1, and they are defined as follows [7]:

- Authority Key identifier: unique key identifier for the public key used for validating the CRL.
- Issuer Alternative Name: used for any name formats for the issuer other than a X.500 Distinguished Name (DN).
- CRL Number: unique serial number for a CRL to ensure detection of missing CRLs or CRL partitions.
- Hold instruction code: indicates action to be taken on encountering a suspended certificate.

The following CRL entry extensions are also available [7]:

- Reason Code: identifies a reason for the certificate revocation, respectively *unspecified, key compromise, CA compromise, affiliation change, superseded certificate, cessation of operation, certificate held/ suspended, or certificate removed from CRL* (used in delta CRLs).
- Invalidity date: indicates the date at which it is known or suspected that the certificate was to be considered invalid.

In addition, each *certificate*, can have a *CRL Distribution Point* field as an extension. This field identifies a given CRL distribution point associated with the certificate in question. This enables the use of *CRL Distribution Points* (section 4.2). The following three extensions are available in *CRLs* in order to support more advanced CRL schemes:

- Issuing Distribution Point: identifies the CRL distribution point for a particular certificate, and indicates whether the CRL is limited to revocations for End-Entity certificates only, for CA-certificates only, or for a limited set of reasons only
- Certificate Issuer: identifies the certificate issuer associated with an entry in an *indirect CRL* (section 4.9)
- Delta CRL Indicator: identifies a CRL as being a *delta-CRL* (section 4.3) only

In a Final Proposed Draft Amendment on Certificate Extensions (FPDAM) [17], two new mechanisms were suggested incorporated into the X.509 standard with the help of three new extensions:

- CRL scope field: CRL extension that indicates the scope/ applicability of a CRL
- Status referral field: CRL extension that provides information about CRLs, including the issuer and location of a CRL, issuance dates, scope definitions and pointers to associated delta CRLs.
- Freshest CRL field: Certificate extension that provides a pointer to the freshest revocation information available.

Extension Name	Standard Document	Applicability
Authority Key Identifier	X.509	CRL
Issuer Alternative Name	X.509	CRL
CRL Number	X.509	CRL
Reason Code	X.509	CRL Entry
Hold Instruction Code	X.509	CRL
Invalidity Date	X.509	CRL Entry
CRL Distribution Point	X.509	Certificate
Issuing Distribution Point	X.509	CRL Entry
Certificate Issuer	X.509	Indirect CRL Entry
Delta CRL Indicator	X.509	Delta-CRL
CRL Scope Field	FPDAM	CRL
Status Referral field	FPDAM	CRL
Freshest CRL Field	FPDAM	Certificate

Table 4.1: Certificate Revocation Extensions

4.1.1 Analysis of CRL

Performance

A CRL is issued at time thisUpdate, and it expires at time nextUpdate. For the basic CRL scheme, "This Update" of the next CRL equals "Next Update" of the present CRL. There is one and only one valid CRL at any time, although a single CRL can be distributed to multiple Repositories. An End-Entity without a valid CRL requests a CRL whenever it has to validate a certificate. The CRL is obtained from the Repository, and *cached* at the End-Entity until it expires.

The MITRE report [14] estimated that each PEM CRL has a size of 51 bytes plus 9 bytes for each listed certificate. Here, the 51 bytes header contains the issuer's unique name (32 bytes), three dates (18 bytes), and a signature algorithm identifier (1 byte). Each certificate entry contains minimally a 20 bit serial number and a 6 byte revocation date, totaling 9 bytes. Note that the MITRE report [14] does not consider the size of the digital signature.

For some implementations based on X.509 CRLs, each CRL has a size of 450 bytes plus 39 bytes for each normal revocation with a reason code. The size varies based on the use of extensions or in different implementations.

Cooper [18] observes that the peak request rate for CRL based systems is more important than the average request rate. The reason for this is that the system should be able to provide a minimum service response time at all times, also during the peak request times. In this case, the average request rate is of less interest. Given the assumption that the number of End-Entities is reasonably large, Cooper [18] assumes that the certificate validation attempt times are independent of each other, so that an exponential inter-arrival probability density can be used to model the CRL scheme. The probability that an End-Entity's first certificate validation attempt (and hence CRL request) occurs within the interval $[t...t + dt]$ as $dt \to 0$, is

$$P(t) = v \exp^{-vt} dt \tag{4.1}$$

where v represents the validation rate [18]. The total number of requests within the interval $[t...t+dt]$ is given by $P(t)$ multiplied with the number of End-Entities

$$T(t) = NP(t) \tag{4.2}$$

where N represents the number of End-Entities. Given $T(t)$, the request rate $R(t)$ at time t is

$$R(t) = \frac{T(t)}{dt} = Nv \exp^{-vt} \tag{4.3}$$

The validity period of a CRL is $\frac{1}{r_u}$, where r_u represents the CRL update rate. A new CRL is issued at the beginning of each validity period, and t is initialized to $t = 0$ for every validity period. The peak request rate is at the start of each validity period, and it is given by

$$R(t)_{peak} = R(0) = Nv \tag{4.4}$$

The CRL request rate during the CRL period is shown in figure 4.4, 4.5 and 4.6 for environments 1, 2 and 3 from table 3.1 respectively. These graphs demonstrates the peak request rate at the start of the CRL validity periods. It is also apparent that the request rate decreases much faster for the simpler system in environment 1 than in the more complex case of environment 3. By looking at the peak request rates on the graphs, it can be derived that CRL is usable for environment 1 and 2, whereas the complex environment 3 virtually prohibits the use of the basic CRL scheme.

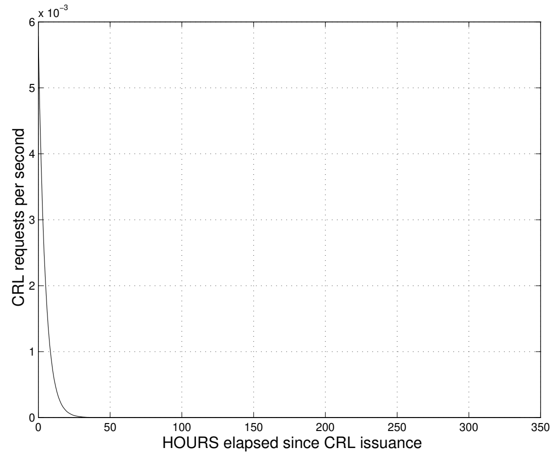

Figure 4.4: Request Rate following CRL issuance for E_1

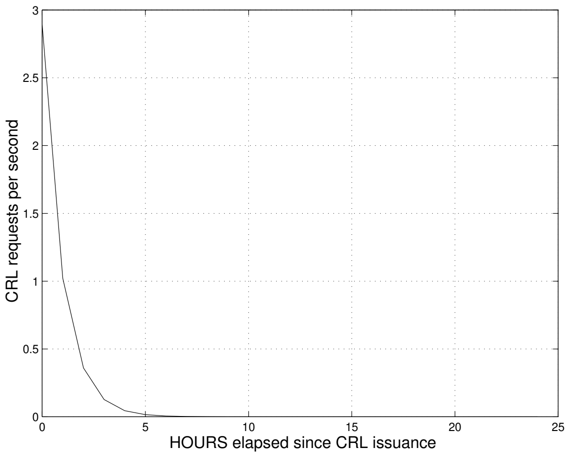

Figure 4.5: Request Rate following CRL issuance for E_2

Timeliness

The timeliness of CRLs is dependent of the length of the periods between updates. A CRL can at worst be as old as one period. In some old discussions on CRLs, the period is defined to be in the order of days, or even weeks [14], but contemporary systems are expected to have stricter timeliness demands. The problem with improved timeliness is, however, that the shorter the period, the more often CRLs will have to be requested, and the network load increases significantly. If the period is too short (relative to the size of the domain), the average request rate will be close to its peak rate. Consequently, the network may be unable to deliver the CRLs to all the requiring End-Entities

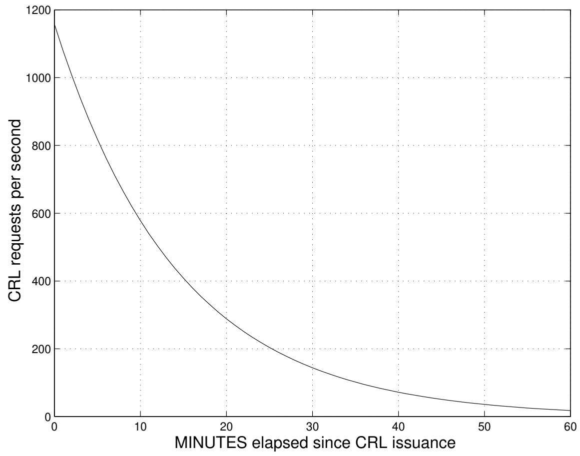

Figure 4.6: Request Rate following CRL issuance for E_3

within reasonable waiting time.

Scalability

The number of End-Entities in a domain impacts the scalability in two ways. The number of revocations is a function of the number of End-Entities given a constant revocation rate, so that the actual size of the CRL increases for larger domains. Similarly, the request rate will increase for larger domains, as shown above. The CRLs are not broadcasted, so every single request leads to a download of a complete CRL. Since the majority of requests are at the start of a period, a large number of End-Entities may lead to unacceptable delays.

Security

CRL provides a 2-party security scheme where the CA signs the CRL which is in turn distributed by the Repository. The End-Entities verify that the CRL is properly signed with the CA's digital signature. In this way, the End-Entity are cryptographically ensured that neither the Repository nor anyone else has tampered with the CRL.

Standards Compliance

CRL version 2 is standardized within the X.509 standard [7]. As described in the MITRE report [14], another version of CRL is also standardized within the PEM framework.

Expressiveness

CRL has high expressiveness in the sense that information regarding specific revocations can be associated with each entry in the list. Furthermore, the CRL scheme is easily extendible to include further extensions, even customized extensions for a domain or product.

Scheme Management

The CRL scheme is easily automated by having a CA compile a new list of current revocations for each update period and distribute this CRL to its Repository. For the purpose of secure archiving, CRLs can easily be adapted for this purpose. Each updated CRL can be stored, with the disadvantage of having to store and facilitate searches in a large number of CRLs. Alternatively, a separate CRL that keeps expired revoked certificates can be maintained. The last alternative requires extra processing in creating and signing the separate list, but it requires less storage.

On-line vs. Off-line

CRL is well suited as an off-line certificate revocation scheme since an End-Entity only have to download a CRL for each period. In this case, each End-Entity can connect for a relatively short time, download a CRL, and then stay off-line for a relatively long time. Note that this scheme (including its off-line use) may be unsuitable when policy dictates a high timeliness.

4.2 CRL Distribution Points

The CRL Distribution Points scheme extends the CRL scheme by addressing the maximum size of a CRL [11, 8]. The size of a *CRL subject population* is limited by dividing the total population for one CA into a number of *segments.* The CRL is segmented, and 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.

This scheme is incorporated into version 3 X.509 certificates and version 2 CRLs [7]. A certificate uses the *CRL Distribution Point* extension in order to point to the applicable distribution point. Also, each CRL segment uses the *Issuing Distribution Point* extension which includes an attribute that is required to match the value in the certificate extension.

Figure 4.7 shows a simple CRL Distribution Point example system with a CA, 4 distribution points (DPs) and 3 End-Entities (EEs). First, the CA publishes the CRL segments to the distribution points, then EE1 requests segment 2 from DP2, then EE3 requests segment 3 from DP3. At a later time, any of the End-Entities may request any of the DP segments, unless the segment is already cached on the End-Entity. After the current CRL validity period is over, the CA publishes new segments.

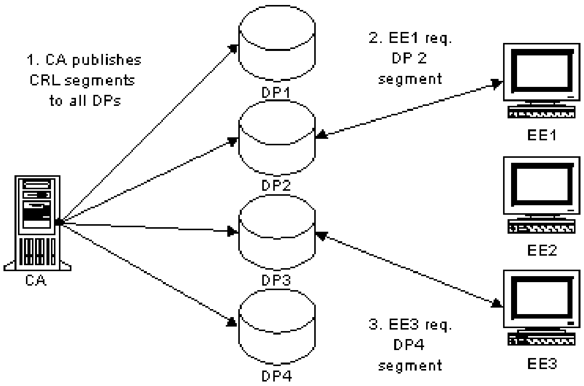

Figure 4.7: Example CRL Distribution Point system

4.2.1 Analysis of CRL Distribution Points

Performance

With this scheme, the number of requests and amount of communication for each segment is reduced compared to full CRL postings. If the End-Entities in general only need a fraction of the segments, the average network load will be significantly smaller than the full CRL network load. Note that if an End-Entity is expected to use different certificates from all possible CRL partitions frequently, this scheme will not reduce the average load, since all or nearly all the partitions will be obtained anyway. In this case, the peak load is likely to be reduced, but the average load will remain at the same level [13] since at least as much data as in a full CRL posting would be downloaded.

Based on a similar analysis as in 4.1.1, Cooper [18] calculates the total request rate $R_s(t)$ for all segment. To simplify the analysis, the calculations makes the following assumptions:

1. The certificate revocation information is randomly distributed over the distribution points, so that any revocation information request is equally likely to refer to any of the distribution points.
2. All the CRL partitions are issued at the same time.

For a given segment, Cooper [18] shows that the total request rate is

$$R_s(t) = Nv \exp^{\frac{-vt}{s}} \tag{4.5}$$

where s is the number of segments. The equation shows that the peak request rate in the system is, in fact, the same (Nv) as for the basic CRL scheme. The average request rate is actually higher for the CRL Distribution Points scheme. This is because each End-Entity is likely to request more than one segment during a validity period. However, as long as all segments are generally not requested, both the peak load and the average load are reduced compared to full CRL postings.

The request rate for 4 distribution points is shown in the figures 4.8, 4.9 and 4.10 for environment 1, 2 and 3 from table 3.1. As shown in the figures, there is no improvement in the request rate, which actually increases in average. However, there may be a significant improvement to the basic CRL scheme in reducing the average network load and revocation information size. This is because each single CRL segment is only a fraction of the full CRL.

Cooper's model does not necessarily present the optimal usage of CRL Distribution Points. This is because the 2 assumptions above do not always hold:

1. With knowledge of the End-Entities and user profiles, a more efficient scheme could be implemented by dividing the revocation information in a way that minimizes the number of CRL

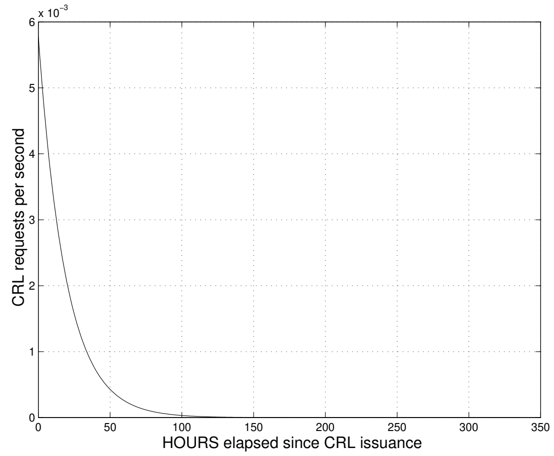

Figure 4.8: Request Rate following CRL issuance for E_1 with $s = 4$

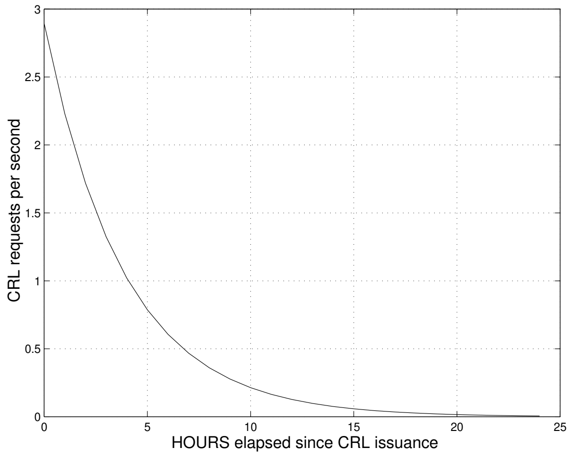

Figure 4.9: Request Rate following CRL issuance for E_2 with $s = 4$

segments needed by each End-Entity. As an example, it may be possible to identify "user cliques" that communicates heavily between themselves, but not often to other End-Entities. In this case, a good solution could be to collect all revocation information on this clique into a separate distribution point.

2. It is possible to *stagger* the issuance and expiration of the different CRL partitions in such a way that they are issued and expire at different times. In this way, the peak request rate is smaller, but occurs more frequently. This approach is described in section 4.5.

A possible problem with this scheme is that the CRL partitioning is permanently defined at

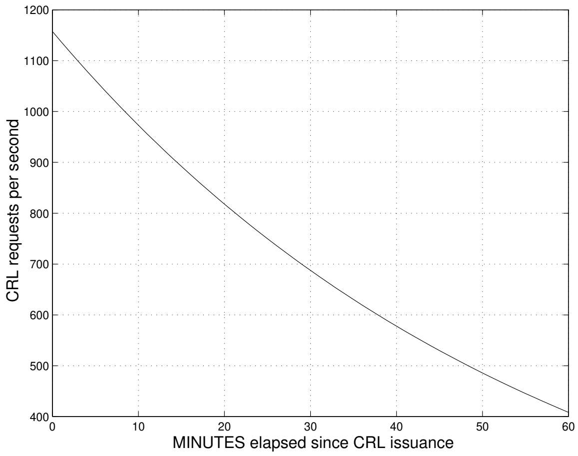

Figure 4.10: Request Rate following CRL issuance for E_3 with $s = 4$

certificate creation time. This can potentially lead to unwanted load unbalances on the CRL distribution point hosts . Another consideration is that the End-Entities must wait for a response in the case when the CRL Segment in question is not in the cache. With basic CRL postings, this delay only occurs one time each period, namely when the CRL is new. If this problem is solved by downloading all segments at the beginning of each period, the CRL Distribution Point scheme will behave *worse* than full CRLs with respect to request rate and network load because of the overhead involved in the multiple requests involved.

Timeliness

The timeliness of the CRL Distribution Point scheme is similar to the scheme for CRLs, except that the the cost of acquiring a new segment is smaller than the cost of acquiring a full CRL. Consequently, the CRL segment validity periods can be reduced with less impact than with full CRL postings. Also, with shorter periods, the probability that a particular segment will be requested in a given period is decreasing.

Scalability

Scalability is the major advantage of CRL Distribution Points compared to full CRL postings. With distribution points, any number of servers can be used to provide CRL segments. Each End-Entity is likely to request less than all segments, so that the network load will be reduced and the request rate for each distribution point is decreased compared to the basic CRL repository.

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted.

Standards Compliance

This scheme is standardized within the current X.509 standard [7].

Expressiveness

The CRL Distribution Points scheme uses the same syntax for certificate revocation entries as CRLs. Consequently, the CRL Distribution Points scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

The main management problem with CRL Distribution Points is that a certificate's distribution point has to be decided as it is issued. There are no means of changing the association between a certificate and a distribution point before the certificate expires. This association have to be decided manually or by some rule when it is issued.

On-line vs. Off-line

It is the intention of this scheme that an End-Entity does not have to download a CRL segment unless the segment is required. In general, the required CRL segments are not known in advance, so that downloads will occur irregularly. In an on-line environment, a new segment is requested whenever it is needed, but this would require a number of irregular connections in an off-line environment. As mentioned above, if this problem is solved by downloading all segments, the CRL Distribution Points scheme performs worse than full CRL postings. Unless an End-Entity knows or is able to guess which segments will be needed in advance, full CRL postings is a more practical scheme for off-line use.

4.3 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 [8]. Delta CRLs are periodically posted, and acts as updates to the last full CRL postings. Delta CRLs are generally significantly smaller than the full CRLs, and they can be posted much more frequently. In this scheme, 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.

This mechanism is implemented using the Delta CRL Indicator extension field in an X.509 CRL. This field identifies the CRL in question to be a delta CRL that only carries updates to an identified CRL base posting. A delta CRL is only applicable to the base CRL identified in its extension field.

4.3.1 Analysis of Delta CRL

Performance

With delta CRLs, the validity period of a base CRL can be increased, since the delta CRLs ensures higher timeliness with significantly smaller postings. Consequently, the average network load can be significantly reduced. The request rate for the both the base CRLs and the delta CRLs follow the same pattern as for the basic CRL scheme for their respective validity period. In this way, there is a peak request rate for the base CRL posting at the start of the CRL validity period, and another delta CRL peak in the start of each delta CRL validity period. Small delta CRL validity periods may introduce a large load, but with a significantly improved timeliness compared to the basic CRL scheme.

The request rate for base CRLs is the same as for the basic CRL scheme:

$$R_b(t) = Nvexp^{-vt} \tag{4.6}$$

where N is the number of End-Entities, v is the validation rate, t is the time passed since last base

CRL issuance. The delta CRL request rate is given by the same equation, with the modification that t is the time passed since last delta CRL issuance:

$$R_{\Delta}(t) = Nvexp^{-vt} \tag{4.7}$$

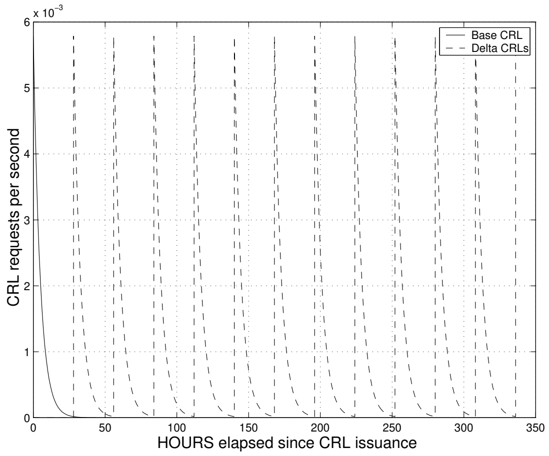

Figure 4.11: Request Rate following base CRL issuance for E_1 with 12 Delta periods

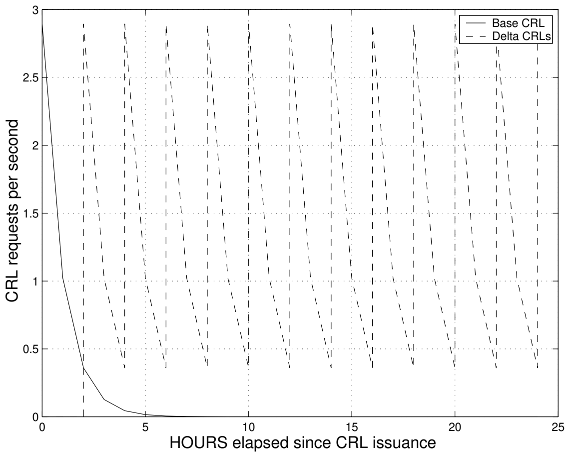

Figure 4.12: Request Rate following base CRL issuance for E_2 with 12 Delta periods

The request rate for a Delta CRL system with the same properties as the CRL model is shown with a delta CRL period of 1/12 of the CRL validity period in figure 4.11, figure 4.12 and figure 4.13 for environments 1 to 3. The dotted line shows the delta CRL requests, and the continuous line shows the base CRL request rate.

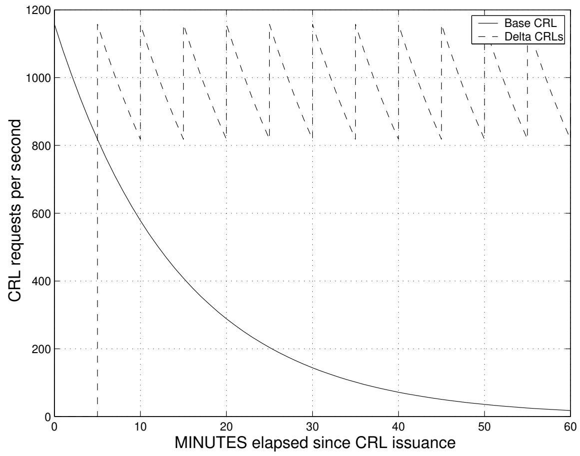

Figure 4.13: Request Rate following base CRL issuance for E_3 with 12 Delta periods

A method for reducing the peak base CRL request rate was described in [19]. In this method, each End-Entity request a new delta CRL rather than the new full CRL posting in the first delta validity period if the previous full CRL was acquired. This first delta period is called a *synchronization interval* (or "synch" interval). This term refers to the transition to using the new full CRL posting, and the peak request rate is reduced, since the End-Entities that did not request the previous full CRL posting and the End-Entities that did request it request the new posting at different times. Also, the scheme can be further improved in terms of performance by allowing the End-Entities to locally create the new full CRL based on the previous full CRL and the delta CRL from the synchronization interval.

Timeliness

Timeliness is the main advantage of using delta CRLs. Since delta CRLs are significantly smaller than full CRL postings, they can be posted far more frequently.

Scalability

Considering the improvement in timeliness and performance, the Delta CRL scheme is more scalable than the basic CRL scheme. However, the problem of the initial peak load of a base CRL posting is not reduced. With large systems, relatively high delays in the initial phase can be expected.

Security

As with the basic CRL scheme, this scheme is 2-party, and only the CA have to be trusted.

Standards Compliance

Delta-CRLs is standardized within CRL version 2 in the X.509 standard [7].

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

This scheme can be automated and implemented in a similar way to the basic CRL scheme. In addition to secure archiving of the base CRL information, the delta CRLs have to be archived as well in order to demonstrate when revocation information was first available.

On-line vs. Off-line

It is the intention of this scheme that base CRL postings are requested for long periods, and delta CRLs are requested for smaller periods. If the scheme is used to provide revocation information with high timeliness, a connection for requesting new delta CRLs have to be set up frequently. However, this scheme could be deployed in a combined scheme, where off-line End-Entities were *allowed* to use a base CRL posting without requesting the delta CRLs. The on-line End-Entities could request Delta-CRLs as described above. This concept is expanded in the Fresh Revocation Information scheme (section 4.11).

4.4 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 not waiting until the "Next Update" time given by the CRL, but to issue a new CRL earlier [18]. 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.

As an example corresponding to Environment 2, a CA could issue a new CRL with a lifetime of 24 hours every sixth hour. In this case, there would always be 4 valid CRLs with different expiration times. The sequence of CRLs could be issued as demonstrated in table 4.2.

cRLNumber	thisUpdate	nextUpdate
0	0:00	24:00
1	6:00	30:00
2	12:00	36:00
3	18:00	40:00
⋮	⋮	⋮

Table 4.2: CRL Over-Issued Schedule with $O = 4$, Environment 2

4.4.1 Analysis of Over-Issued CRL

Performance

In the analysis of the basic CRL scheme, it was demonstrated that all cached CRLs expired at the same time, so that a high peak request time occurred at the start of each validity period. The Over-Issued CRL scheme distributes the requests in such a way that the peak request rates are always relatively smaller, but occur more frequently.

If an *interval* is the time between an over-issued CRL posting and the next, [18] shows that the request rate during the interval is

$$R_o(t) = \frac{Nve^{-vt}}{(O-1)(1-e^{vl/O})+1} \tag{4.8}$$

where N is the number of End-Entities, v is the certificate validation rate, O is the number of *currently* valid CRLs and l is the validity time of a single CRL. This scheme is demonstrated in figure 4.14, figure 4.15 and figure 4.16 for environment 1, 2 and 3 with $O = 4$ and l as given by the update rates.

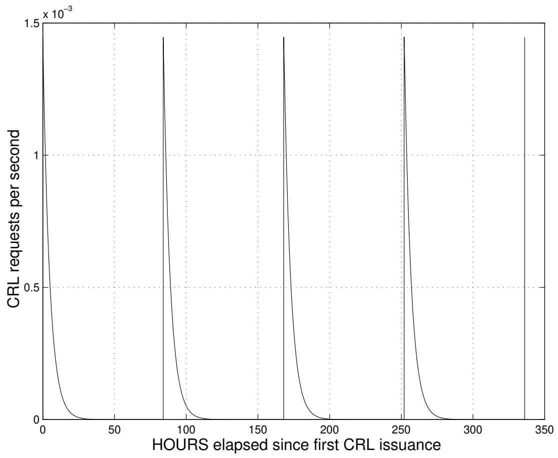

Figure 4.14: Request Rate following over-issued CRL issuance for E_1 with $O = 4$

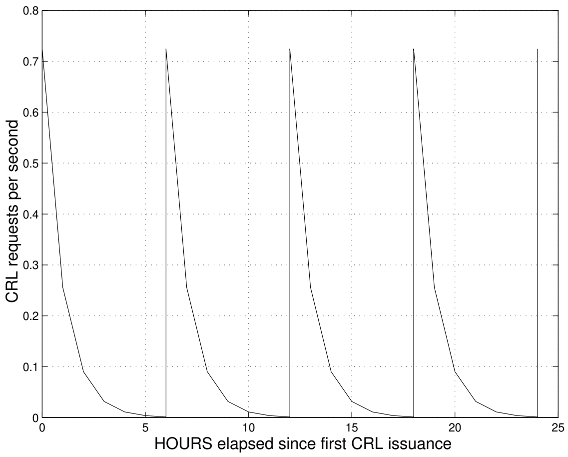

Figure 4.15: Request Rate following over-issued CRL issuance for E_2 with $O = 4$

Timeliness

This scheme is not in general more timely than the basic CRL scheme. An acquired CRL is cached for the full validity period. However, when an End-Entity requests a new CRL, this CRL is guaranteed to be less than l/O old. This provides a more timely response. Also, timeliness may be improved with this scheme by decreasing the CRL validity period compared to the basic CRL scheme. This

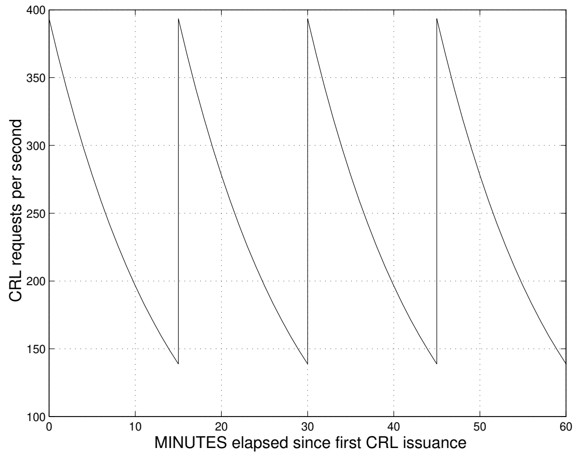

Figure 4.16: Request Rate following over-issued CRL issuance for E_3 with $O = 4$

is feasible since the peak request rates are lower. Note that if all End-Entities requested a new CRL for each interval, the scheme would degenerate into the basic CRL scheme.

Scalability

The Over-Issued CRL scheme provides a more scalable system by evening out the peak request rates. By the same argument as for timeliness, it is possible to increase the number of End-Entities without getting unacceptable delays due to high network load because of the decreased peak request rates.

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted.

Note that a potential controversy around this system is the possible problems involved in having multiple CRLs where some are outdated compared to the newest posting. An End-Entity could potentially collect all valid CRLs and, if a certificate was revoked on the newest CRL but not an older (still valid) CRL, claim to have been using the oldest valid CRL when taking some action involving that particular certificate. In this way, the End-Entity could claim that it was acting in good faith, and blame some other party for any damage or loss incurred. This is a non-repudiation problem, and the same problem can occur in any case where multiple valid sources of revocation information with different degrees of freshness is available.

Standards Compliance

This scheme can be implemented within the current X.509 standard [7].

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

An attractive attribute with over-issued CRLs is that they can be implemented without any changes to CRL version 2 in the X.509 standard or to existing client applications.

On-line vs. Off-line

Over-Issued CRLs are, like the basic CRL scheme, suited to off-line use.

4.5 Over-issued segmented CRLs

As described in section 4.2, the CRL Distribution Points scheme does not reduce the high peak request rate compared to full CRL postings. By taking the approach of over-issued CRLs and applying it to the CRL Distribution Points scheme, the peak request rate can be significantly reduced. Over-issuing can be combined with CRL segmentation in two ways:

1. All CRL segments can be issued at the same time with a validity period longer than the time to next issuance (i.e. the segments can be over-issued).

2. Each CRL segment can be staggered relative to the other segments (i.e. issued at different times), so that the peak request rates occur at different times.

4.5.1 Analysis of Over-Issued segmented CRLs

Performance

Both of the methods outlined above distributes and reduces the peak request rates an peak network load in a similar manner as over-issued CRLs. Cooper [18] shows that the request rate during the interval between staggered CRL segments issued at evenly separated intervals (according to method 2) is

$$R_{sI,2}(t) = \frac{Nv \exp^{-vt'/s}(1 - \exp^{-vl/s})}{s(1 - \exp^{-vl/s^2})} \tag{4.9}$$

where N is the number of End-Entities, v is the validation rate, s is the number of segments, $t' = t \pmod{\frac{l}{s}}$, and l is the validity time of a CRL segment. Here, $t \pmod{\frac{l}{s}}$ denotes the modular function extended to be applicable to real numbers.

A higher reduction in the peak request rate can be achieved by over-issuing the CRL segments according to method 1. Cooper [18] demonstrates that the request rate during the interval between two over-issued CRL segments is

$$R_{sI,1}(t) = \frac{Nv \exp^{-vt/s}}{(O-1)(1 - \exp^{-vl/sO}) + 1} \tag{4.10}$$

where O is the degree of overlapping for each segment. As for over-issued CRLs, the gain increases as O approaches infinity. Both the methods are demonstrated in figure 4.17, 4.18 and 4.19 with $O = 4$ and $s = 4$ for environments 1, 2 and 3 respectively. The continuous line represents the more efficient method 1, whereas the dotted line represents method 2.

Timeliness

This scheme shows the same timeliness properties as over-issued CRLs. This scheme is not primarily designed to increase timeliness, but the reduced request rate can be utilized to improve timeliness.

Scalability

Over-issued CRL segments takes advantage of both the CRL Distribution Points scheme and the Over-issued CRLs scheme to improve scalability. This scheme has both the advantage of decreased

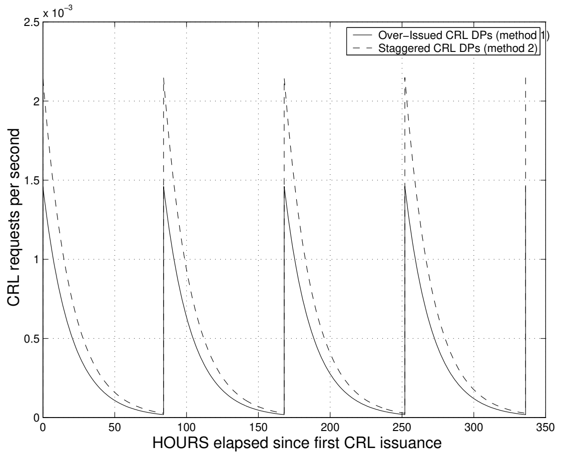

Figure 4.17: Request Rate following overlapping CRL segment issuance for E_1 with $O = 4$ and $s = 4$

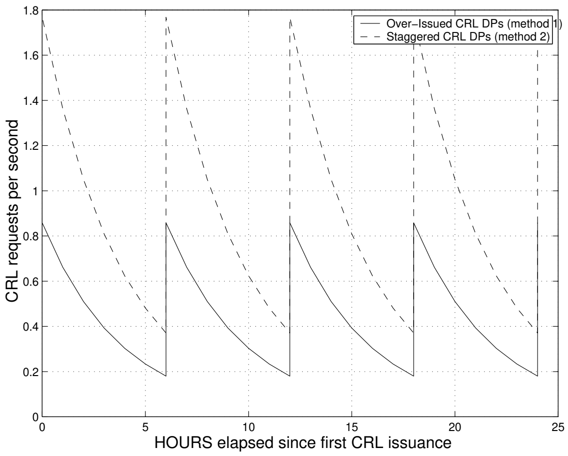

Figure 4.18: Request Rate following overlapping CRL segment issuance for E_2 with $O = 4$ and $s = 4$

peak request rates and the option of using multiple physical computers as CRL Distribution Points.

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted. Note, however, that the non-repudiation issue presented in 4.4.1 is valid for method 1 (but generally not for method 2).

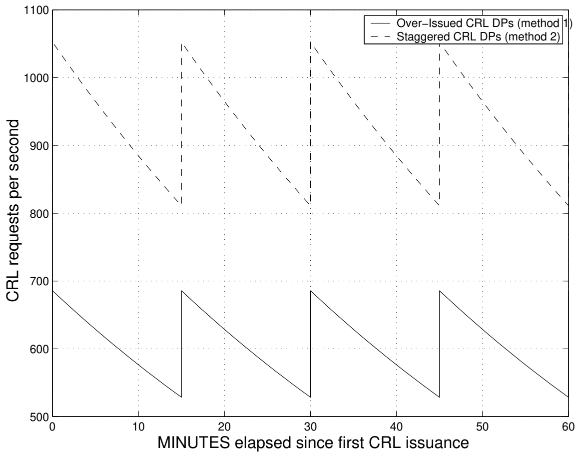

Figure 4.19: Request Rate following overlapping CRL segment issuance for E_3 with $O = 4$ and $s = 4$

Standards Compliance

This scheme can be implemented within the current X.509 standard [7].

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

The management consideration regarding the static segmentation in CRL Distribution Points is not changed. Secure archiving can be achieved by having each CRL Distribution Point log their CRLs as in the basic CRL scheme. For method 1, the amount of archiving is increased, as there are O times as many CRL segments for each Distribution Point.

4.6 Sliding window Delta CRLs

This scheme is based on the observation that delta CRLs provide revocation information for a certain *window* of time following the issuance of the base CRL, and that these window sizes vary for each delta CRL. With cached base CRLs, large windows sizes result in lower request rates for the base CRLs.

The Sliding Window Delta CRL scheme takes advantage of this by having a constant, large window size rather than allowing the window size vary. In addition, this scheme allows new base CRLs to be *created locally* based on a valid base CRL and a valid delta CRL, as suggested in section 4.3. A fixed delta CRL window size can be implemented by issuing a new *over-issued* base CRL every time a delta CRL is issued. Each delta CRL refers to the base CRL that was issued exactly one base CRL validity period ago, and it is applicable to all base CRLs issued within this period (i.e. all base CRLs that have not expired).

The idea behind this scheme is that the long window size allows most End-Entities to request a delta CRL before their certificate revocation information has expired. Following a request, the End-Entities create a new base CRL locally (with an updated validity period), and the base CRL request becomes unnecessary. In this way, the total requests for base CRLs are significantly reduced, as such a request only occurs if an End-Entity's last created (or requested) base CRL has expired by a time interval exceeding the window size.

4.6.1 Analysis of Sliding window Delta-CRLs

Performance

This scheme takes advantage of both the properties of over-issued (base) CRLs and delta CRLs with local creation of updated base CRLs. The request rate for delta CRLs in this scheme is exactly the same as for the Delta CRL scheme. The request rate for base CRLs is demonstrated by Cooper [18] to be

$$R_w(t) = Nv \exp^{-v(t+w)} \tag{4.11}$$

where w represents the window size of the current Delta-CRL, N is the number of End-Entities and v is the validation rate. Note that the base CRL peak request rate can be made arbitrarily small by increasing the window size. However, such an increase will also increase the size of the delta CRLs. Consequently, it is advantageous to find an optimal window size.

The request rates of this scheme are demonstrated in figure 4.20, 4.21 and 4.22 with $O = 4$, the window size equal to 1/6 of the validity period, and 12 delta-periods for each CRL validity period for environments 1, 2 and 3 respectively.

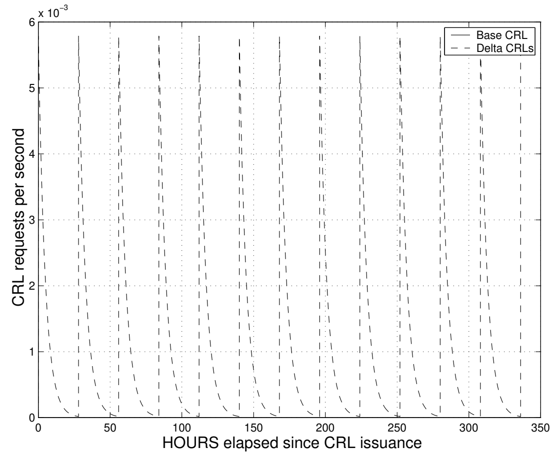

Figure 4.20: Req Rate following issuance for E_1 using Sliding window delta CRLs, $w = 56hrs$

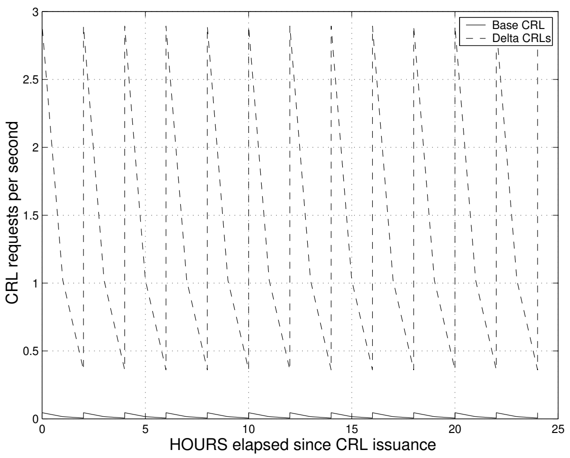

Figure 4.21: Req Rate following issuance for E_2 using Sliding window delta CRLs, $w = 4hrs$

Timeliness

Timeliness can be improved due to the performance gain.

Scalability

This scheme improves scalability compared to delta CRLs due to the reduced base CRL request rates.

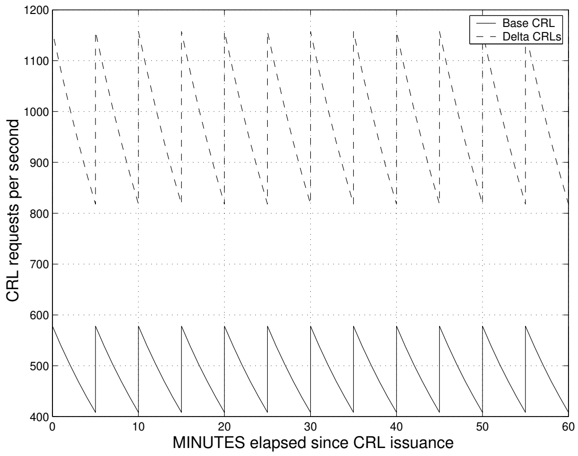

Figure 4.22: Req Rate following issuance for E_3 using Sliding window delta CRLs, $w = 10min$

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted. However, this scheme potentially faces the same non-repudiation considerations as Over-Issued CRLs.

Standards Compliance

This scheme can not be implemented within the current X.509 framework. The scheme is not directly supported by the X.509 Delta CRL scheme, as this scheme requires each delta CRL to be applicable to all base CRLs newer than or corresponding to the base CRL explicitly pointed at. The X.509 version 2 CRL standard states that a delta CRL is only applicable to its respective base CRL. Furthermore, this scheme requires local generation of new base CRLs based on the newest delta CRL and some valid base CRL. This is not included as part of the original delta CRL scheme. In order to implement this scheme, the standard must be modified or violated.

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

This scheme is managed in the same way as delta CRLs, except for the fact that more base CRLs have to be issued and securely archived. Also, the End-Entities must be capable of generating the new base CRLs.

On-line vs. Off-line

As for delta CRLs, this scheme generally provides more timely revocation information than the basic CRL scheme. An off-line End-Entity either have to connect more frequently compared to a less frequently updated scheme, or trust the base CRLs to be sufficient for their validity period. Note that the local creation of the base CRLs can be an attractive feature for an off-line End-Entity.

4.7 Over-issuing Delta-CRLs

The Delta-CRLs themselves can also be over-issued in order to reduce the peak request rate for Delta-CRLs. This scheme introduces the concept of delta CRLs with overlapping issuance and expiration times. This scheme shares the properties of the Sliding Window Delta CRL scheme, except that this scheme also over-issues the delta CRLs.

4.7.1 Analysis of Over-issuing Delta-CRLs

Performance

Cooper [18] shows that the request rate for over-issued Delta-CRLs is described by the same function as the request rate for over-issued CRLs in a system without Delta-CRLs:

$$R_{o\Delta}(t) = \frac{Nv\exp^{-vt}}{(O-1)(1-\exp^{-vl/O})+1} \tag{4.12}$$

The request rate for the base CRLs in this scheme is

$$R_{ob}(t) = \frac{Nv\exp^{-(t+w+l/O-l)v}}{(O-1)(1-\exp^{-vl/O})+1} \tag{4.13}$$

It is possible to significantly reduce the peak request rates when the validity periods of Delta-CRLs are relatively long (e.g. 1 hour), but as the validity period decreases (e.g. 10 minutes), this method is less efficient. This is because the scheme becomes increasingly unable to take advantage of the CRL caching.

The request rates of this scheme are demonstrated in figure 4.23, 4.24 and 4.25 with $O = 4$, the window size equal to 1/6 of the validity period and 12 delta-periods for each CRL validity period for environments 1, 2 and 3 respectively.

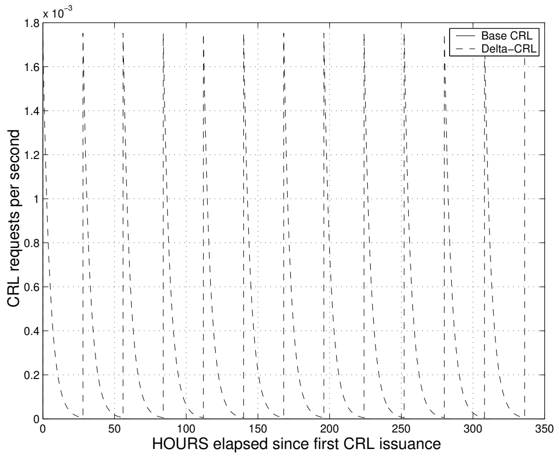

Figure 4.23: Req Rate following issuance for E_1 using over-issued delta CRLs, $w = 56hrs$, $O = 4$

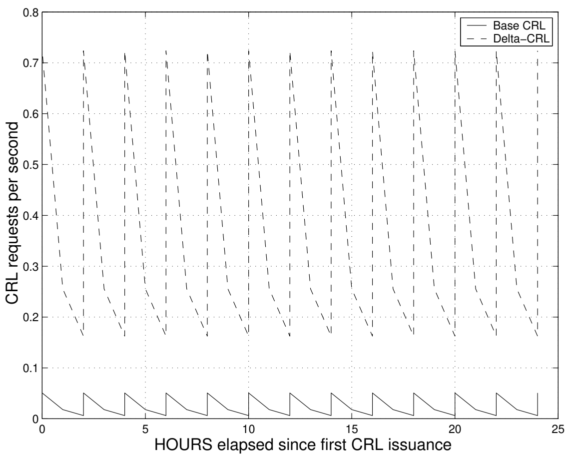

Figure 4.24: Req Rate following issuance for E_2 using over-issued delta CRLs, $w = 4hrs$, $O = 4$

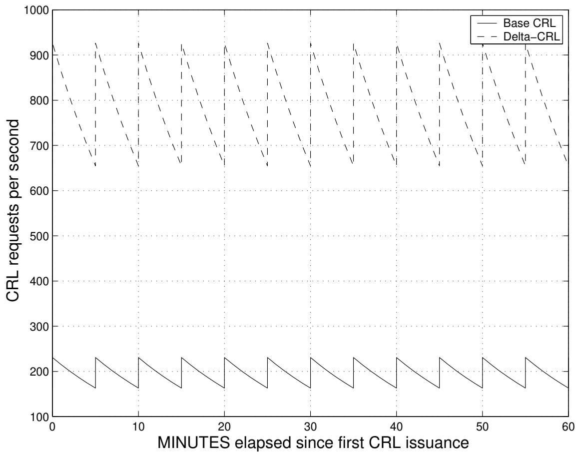

Figure 4.25: Req Rate following issuance for E_3 using over-issued delta CRLs, $w = 10min$, $O = 4$

Timeliness

As with the other over-issued and delta CRL based schemes, this scheme allows improved timeliness through improved performance.

Scalability

With a reduced peak request rate, this scheme is more scalable than over-issued CRLs and delta CRLs.

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted, but the same non-repudiation considerations as for over-issued CRLs are still valid.

Standards Compliance

As for the Sliding Window Delta-CRL scheme, this scheme can not be implemented within the current X.509 framework.

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

This scheme is managed in the same way as sliding window Delta-CRL, except for the fact that more base Delta-CRLs have to be issued and persistently logged. There is no difference in managing the End-Entity.

On-line vs. Off-line

This scheme has the same properties regarding off-line use as the Sliding Window Delta CRL scheme.

4.8 Authority Revocation Lists (ARLs)

An *Authority Revocation List* (ARL)[11] 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 reduces the load in verifying the certificate path. This scheme is incorporated into X.509 version 2 CRLs by providing the Issuing Distribution Point for the CRL and specifying that the CRL is limited to revocation for CA-certificates exclusively by setting the value "onlyContainsCACerts" to True.

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, the size of each list is either zero or small (CA certificate revocation rarely occurs), significantly improving the performance of a system relative to full CRLs.

For an End-Entity certificate user, the ARL scheme implies that there will be two tasks involved in checking certificate revocation status:

- Check that no CA certificates in the certificate path are revoked. This is done by obtaining the respective ARLs.
- Check that the certificate in question is not revoked. This is done by obtaining the appropriate End-Entity CRL.

Figure 4.26 shows a simple ARL example system with two CAs, two Repositories and 3 End-Entities (EEs). CA1 has signed CA2's certificate, and it publishes the revocation information concerning CA2 in an ARL to Repository 1. CA2 publishes the revocation information for its subjects in a CRL and publishes it to Repository 2. In order to verify a certificate, EE3 has to request the ARL from Repository 1 before trusting the CRL published by CA2. At a later time, EE1 and EE2 may perform the same operation. CA1 publishes a new ARL after one ARL validity period, and CA2 publishes a new CRL after one CRL validity period. The two validity periods may be different.

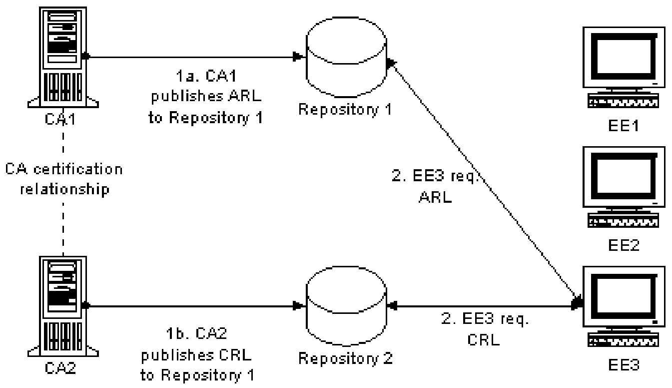

Figure 4.26: Example ARL system

4.8.1 Analysis of ARL

Performance

The ARL scheme improves the performance for certification systems where CAs may be certified by other CAs. In this case, a valid ARL is requested for each CA that has signed one or more certificates in the certificate path. Since ARLs can be considered marginally small, this is a significant improvement to using full CRLs for all subordinate entities, CAs and End-Entities alike. The performance gain is advantageous for both hierarchical, cross-certifying and hybrid certification systems, but the gain is greatest for hierarchical systems, where the certification path can be expected to be longer.

Timeliness

ARL by itself does not improve timeliness, but the validation of the issuing CA takes shorter time than with full CRLs. Each validation period can be shortened, and more timely CRLs can be issued.

Scalability

The ARL scheme is an improvement in scalability compared to the basic CRL scheme. ARLs allow systems with several CAs to be implemented without having to obtain excessive amounts of revocation information.

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted.

Standards Compliance

This scheme is standardized within the current X.509 standard [7].

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

The implementation of the ARL scheme is quite to the CRL scheme, except that each CA that have certified other CAs operate with both CRLs and ARLs. This is also true for the purpose of secure archiving.

On-line vs. Off-line

To be able to execute certificate verifications off-line, an End-Entity must download the relevant CRLs as well as all ARLs in its certificate path. Since the ARLs are small and this can be done off-line as well, the use of ARLs is suitable for off-line use. In particular, it is more suitable than full CRLs, since a smaller amount of data have to be downloaded for a given system of CAs.

4.9 Indirect CRL

In the Indirect CRL scheme, a different authority than the certificate issuer can issue a CRL [11]. A CRL can be configured to contain revocations from multiple certification authorities, so that the overall number of CRLs needed by relying parties during the certificate validation can be reduced. The overhead for obtaining multiple CRLs is reduced, and the recursive certificate verification for a certification path (where every step except one are certification authority certificates) 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 Issuing Distribution Point extension for X.509 CRLs is set to refer to the Indirect CRL Distribution point, and each CRL entry contain the Certificate Issuer extension to refer to the certificate issuer in question.

Figure 4.27 shows a simple Indirect CRL example system that collects the information from 3 CRLs into one indirect CRL. The indirect CRL issuer then obtains the 3 CRLs whenever the validity period of one of them expires. The issuer then compiles an indirect CRL consisting of the 3 CRLs and an overhead that corresponds to the overhead in a basic CRL.

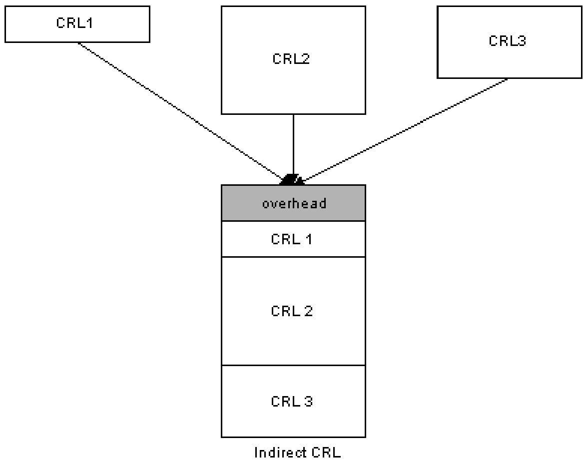

Figure 4.27: Example Indirect CRL system

4.9.1 Analysis of Indirect CRL

Performance

The main performance gain in using Indirect CRLs is that there is less overhead in requesting multiple CRLs from different sources. In particular, this can lower the request rate, as well as the average load. An important consideration for this scheme is that the Indirect CRL never should grow so large that the performance benefit of combining the CRLs is lost. Note also that this is somewhat the opposite approach compared to CRL Distribution Points, where CRLs are split up rather than merged. The two schemes are appropriate for different environments.

Timeliness

The different sources of revocation information may have different validity periods. In this way, it is necessary to compile and publish a new Indirect CRL every time any of the validity periods expires. Consequently, an End-Entity may have to download *all* the information at the end of the shortest validity periods. The revocation information with the longer validity periods will then be downloaded multiple times. An even worse scenario occurs if the revocation information sources are out of synchronization, so that the starts and ends of the different validity periods do not correspond. If indirect CRLs are to be used, the validity periods of the different sources of revocation information should be adjusted so that unnecessary indirect CRLs issuances do not occur.

Scalability

The Indirect CRL scheme simplifies the use of revocation information from multiple sources relative to using full CRL postings. For a system consisting of many small CRLs and ARLs, this scheme may be more scalable by allowing domains to interoperate in a more efficient way.

Security

This scheme introduces a security consideration by allowing another party than the original CA to issue the CRLs. This scheme is 3-party, and both the indirect CRL issuer and the CA have to be trusted. The indirect CRL issuer verifies the integrity of its acquired revocation information and signs the indirect CRL with its own digital signature. Note that this problem can be avoided if the indirect CRL issuer is already trusted (e.g. as a CA).

Standards Compliance

This scheme is standardized within the current X.509 standard [7].

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

Indirect CRLs simplifies management on the End-Entity, since it only have to request CRLs from a single or at least fewer providers compared to requesting the revocation information from the individual providers. Note that all relevant CRLs are not necessary included in a single indirect CRL, so the End-Entity may have to request more than one indirect CRLs or other CRLs. The integration of the individual CRLs is done on the CRL issuer, which is an additional resource that have to be configured and managed. This scheme does not have to perform secure archiving of revocations, since this may be handled at the individual CRL issuers. An Indirect CRL Issuer only have to log which CRLs it combined and when they where retrieved in order to demonstrate what has occurred.

On-line vs. Off-line

Indirect CRLs provides a scheme that is suitable for off-line use. The End-Entity only have to periodically request a single indirect CRL from a single repository rather than connecting to several repositories at potentially different times.

4.10 Enhanced CRL Distribution Points

A problem with the use of the CRL Distribution Points scheme 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 dynamic solution may be desirable for the following reasons:

- The load on the distribution points can be unbalanced because some distribution points have more actual revocations than others.
- The load on the distribution points can be unbalanced because some CRL segments are requested more frequently than others.
- Partitioning strategies could be based on other properties than certificate serial number ranges, like revocation reasons, certificate types or other criteria.

The *Enhanced CRL Distribution Points* scheme [17] 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 [17] in order to enable the use of Enhanced CRL Distribution Points. The *CRL Scope Field* allows scope statements to be defined and associated with CRL Distribution Points. The *Status Referral Field* can be used to modify the CRL partitioning for CRL Distribution Points by using the scope statements to refer to a CRL Distribution Point.

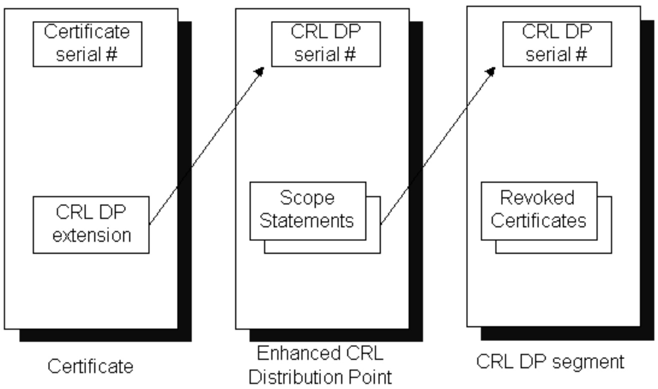

Figure 4.28: Enhanced Distribution Point functionality

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.

4.10.1 Analysis of Enhanced CRL Distribution Points

Performance

The main performance benefit in using Enhanced CRL Distribution Points is the ability to balance the load between the CRL Distribution Points involved. This can reduce performance drops due to particularly overloaded servers.

Timeliness

The Enhanced CRL Distribution Points scheme provides approximately the same timeliness as CRL Distribution Points.

Scalability

The load balancing benefit is particularly advantageous in larger systems. It is difficult to estimate the load on the different distribution points at certificate issuance, and scalability can be improved by allowing the scopes of the distribution points to be updated dynamically.

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted.

Standards Compliance

This scheme is documented in the Final Proposed Draft Amendment (FPDAM) on Certificate Extensions [17]. It is expected to be incorporated into the next update to X.509 later this year.

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

Scheme Management

The Enhanced CRL Distribution Points scheme enables dynamic and flexible management of the distribution points. This management have to be performed through a management application or an automated program. Secure archiving can be performed at each CRL distribution point in the same was as described for the CRL Distribution Points scheme.

On-line vs. Off-line

For off-line End-Entities, it is of little relevance which CRL Distribution Points are used. Enhanced CRL distribution points has the same properties as regular CRL distribution points in this case. Note, however, that it may be possible to define the scope of some distribution points to approximate (or guess) what revocation information the off-line users are most likely to use.

4.11 Freshest Revocation Information

With different timeliness requirements for different End-Entities, it has been suggested to have a reference to the freshest revocation information available for End-Entities with high timeliness demands. A new certificate extension called *Freshest CRL* [17] 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 high timeliness requirements.

4.11.1 Analysis of Fresh Revocation Information

Performance

This scheme has a positive effect on performance by allowing a discrimination of End-Entities based on the timeliness demands. Only the information that is sufficient according to different timeliness requirements is distributed, rather than providing the best timeliness information available to everyone.

Timeliness

This scheme provides improved timeliness for the End-Entities that obtains the freshest CRLs, and less timely revocation information for the others. The timeliness of this scheme is designed to efficiently meet the requirements of different End-Entities with different policy requirements.

Scalability

For the same reasons as above, this scheme scales better than a non-discriminating scheme for a diverse system with both critical and non-critical systems.

Security

As with the basic CRL scheme, this scheme is 2-party. Only the CA have to be trusted. There is, however, a non-repudiation consideration associated with having two sources of revocation information with different timeliness.

Standards Compliance

This scheme is documented in the Final Proposed Draft Amendment (FPDAM) on Certificate Extensions [17]. It is expected to be incorporated into the next update to X.509 later this year.

Scheme Management

The Freshest Revocation Information scheme allows the integration of systems with different timeliness demands into the same management system. Furthermore, the End-Entities can have the same front-end whether they demand high timeliness or not. In particular, this makes it simple to change timeliness demands. For secure archiving purposes, the freshest revocation information have to be archived as well as the standard revocation information.

Expressiveness

This scheme maintains the same expressiveness as the basic CRL scheme.

On-line vs. Off-line

This scheme is like CRLs well suited for use in an off-line environment. In particular, non-critical off-line equipment can connect relatively infrequently in order to get the less timely information.

Chapter 5

On-line Revocation Status Checking Protocols

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 time granularity factor in 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 insure that the Server itself has 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* [11] used to retrieve the necessary revocation information. If the on-line Server relies on any kind of precomputation or caching, the revocation information will inevitably be outdated to some degree.

5.1 On-Line Certificate Status Protocol (OCSP)

The *On-Line Certificate Status Protocol* (OCSP) [20] 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. This is different from SCVP below, which is a protocol to fully validate a certificate based on all certificate validation criteria. This section describes the OCSP protocol itself, as well as a number of methods suggested as back-end mechanisms for the scheme.

In this protocol, an End-Entity requests the revocation information for one or more certificates in an OCSP request to the OCSP responder. The OCSP request contains the current OCSP version number, the service request type, as well as one or more certificate identifiers (bulk requests are allowed). The 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 contains the certificate identifier, the certificate status and a validity interval for the response. The OCSP response must be digitally signed 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.

In an expired Internet Draft [21], 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 and immediate 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 take one of the following actions:

1. An OCSP CA issues an immediate response itself

2. If the necessary information is in the non-expired cache, the OCSP responder responds with cached information

3. If there is no non-expired cache, the request is forwarded to one or more neighboring OCSP responders

This course of actions allows for intermediate OCSP responders, where a number of OCSP responders are involved before the request reaches either the OCSP CA or some non-expired cache with the necessary information. In order to control the caching, the method meets the following criteria:

- An End-Entity may override any intermediary caches between it and the CA in order to obtain fresh revocation information
- An End-Entity must be given the information to decide whether the revocation information is fresh enough
- A CA must be able to specify the caching policy for the OCSP responders

It is also possible to *precompute* or *preconstruct* OCSP responses in order to improve the performance of the scheme. 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. The 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.

Note also that there is a proposal to support additional functionality through the OCSP extension mechanism (OCSP-X) [22] as described in section 5.3.

5.1.1 Analysis of OCSP

Performance

With the basic OCSP scheme, there is one request for one or more simultaneous certificate revocation status checks at a given End-Entity. The peak request rate is likely to be at its maximum when the End-Entities are most heavily used. If the average number of certificate identifiers in the requests is close to one, the revocation rate is approximately the same as the certificate validation rate at any time. Due to the fact that the OCSP responder/ Server have to sign each reply, a significant processing load is introduced at both the Server and the End-Entity. The delay is partly dependent on the processing time for the digital signature.

In addition, the performance of a scheme using OCSP is dependent on the back-end mechanism. In particular, caching and pre-computation can be used to improve performance at the cost of timeliness. Caching can be performed both at the OCSP responders as well as on the End-Entities.

Timeliness

The timeliness of OCSP is heavily dependent on what method the OCSP responder uses to gather its revocation information, and whether caching is allowed. Even though OCSP provides real-time replies, the information it carries may not be fresh if the OCSP responder acquires its information through the use of CRLs or any other cached mechanism. On the other hand, the timeliness may be very high if the OCSP responder acquires its information directly from the CAs central revocation information. The revocation information from OCSP is no more timely than the OCSP responder's source of information.

Scalability

OCSP is designed to scale well (linearly) by increasing the number of OCSP Servers as the amount of End-Entities increases. The performance of each individual Server is,however, quite limited due to the digital signing of the replies. Note also that the properties of the back-end mechanism are important for the scalability.

Security

This scheme is a 3-party scheme, since both the CA and the OCSP Server have to be trusted. The OCSP Server have to be trusted to gather authenticated revocation information and produce correct and digitally signed responses to each request. When receiving an OCSP reply, the End-Entity is required to possess a trusted copy of the OCSP responder's public key in order to verify the digital signature. In addition, other sources of revocation information (e.g. intermediate OCSP responders) may have to be trusted. Note that it is not necessarily apparent to the End-Entity what other sources are trusted. This is at the OCSP responders discretion.

Standards Compliance

OCSP is being standardized in the IETF PKIX working group. The basic protocol is documented in RFC 2560 [20].

Expressiveness

The OCSP standard [20] specifies that all extensions available as CRL entry extensions are also available as OCSP extensions.

Scheme Management

The OCSP scheme allows other certificate revocation schemes to be used as the back-end mechanism. OCSP only specifies the behavior of the OCSP Server and the OCSP End-Entities. The management of the End-Entities is quite light, since they only have to request information and verify the reply. The management of the OCSP Servers can, however, be a large job, since the number of Servers can be quite high.

on-line vs./ Off-line

Being an on-line request/ response protocol, OCSP is obviously not suitable for off-line use. OCSP responses could potentially be cached, but this would compromise a high timeliness. Also, a high number of OCSP responses would have to be cached, so that the list based approaches most likely would perform better.

5.2 Simple Certificate Verification Protocol (SCVP)

The *Simple Certificate Verification Protocol (SCVP)* [23] is a more general request/ response protocol than OCSP. It handles the entire certificate verification process rather than only the certificate revocation checking. The mandate for designing this protocol is that certificate validation is a complex process, and SCVP is designed to support a wide certificate deployment in a variety of applications and environments. There are two main motivations for using SCVP [23]:

- The End-Entity may want to avoid the overhead of processing the certificate validation locally
- SCVP may centrally enforce a validation policy

The SCVP is designed in such a way that an application can perform any parts of or the full protocol. In the extreme cases, this allows the End-Entity to perform all processing itself and use the SCVP Server only as an untrusted information-gatherer, or to offload all the certificate validation work to a completely trusted Server that will guarantee that policies are enforced in a consistent fashion. In between these extremes, SCVP can be used to obtain some but not all trusted information needed for a certificate validation. A *Query* is used to specify the object of a request, and it can include one or more certificates.

This scheme is not analyzed in this text, as the properties of this protocol are outside the scope of certificate revocation. The properties with respect to certificate revocation can be expected to be fairly similar to OCSP. SCVP is being standardized in the IETF PKIX working group, and it is documented in an Internet Draft [23].

5.3 OCSP Extensions (OCSP-X)

OCSP Extensions (OCSP-X) [22] is another effort in designing a more general request-response protocol than OCSP. OCSP-X provides a richer set of semantics for OCSP through its optional extensions, so that 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 a particular operation. The End-Entity can utilize all or parts of the protocol extensions as required. Functionally, OCSP-X has many similarities with SCVP.

As for SCVP, this scheme is not analyzed in this text, as its properties with respect to certificate revocation are similar to the properties of OCSP. OCSP-X is being standardized in the IETF working group, and it is documented in an Internet Draft [22].

5.4 Data Certification Server (DCS)

The Data Certification Server (DCS) [24] 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. With this service, DCS can, for example, verify the correctness of a signature and check the full certification path. DCS can check the revocation status of a certificate, based on CRLs, ARLs or some other (typically more timely) mechanism. Note that DCS provides services that are more general than SCVP and OCSP-X.

As for SCVP, this scheme is not analyzed in this text, as its properties with respect to certificate revocation are similar to the properties of OCSP. DCS is being standardized in the IETF working group, and it is documented in an (expired) Internet Draft [24].

Chapter 6

Revocation Schemes with Reduced Data-Structures

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 full revocation certificates and OCSP without multiple certificate identities) is that the process of digitally signing each revocation reply is processing intensive. The schemes with reduced data-structures attempt to solve this problem by using one-way hash functions in order to provide *lightweight digital signatures* in different ways. This class or revocation schemes are also referred to as *trusted dictionaries* [16].

6.1 Certificate Revocation Status Directory (CRS Directory)

The *Certificate Revocation Status Directory* (CRS Directory) [12] 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 [12], 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). Note that the CRS Directory behaves like a Certificate Revocation Server rather than a Repository.

6.1.1 Analysis of CRS

Performance

CRS is intended to increase performance by using a more compact data structure than the full lists in CRL based systems. There is an overall performance gain compared to CRLs by decreasing the communication load between the Server and the End-Entities at the cost of an increased communication load between the CA and the Server. Since the communication to the End-Entities represents the major communication load, this implies an overall performance gain compared to CRLs. Note that the scheme requires the CA to compute and store the Y and N values for each certificate for a full year in advance.

By the argument above, CRS can be expected to have an average request rate comparable to the on-line revocation status checking protocols. There is a request each time an End-Entity validates every certificate for the first time in a day (the Y and N for a certain certificate can be cached until the next day). The overhead of each reply is, however, limited to 100 bit (either an Y or a N). In this way, the communication load is reduced compared to both CRLs and the on-line revocation status checking protocols. Compared to OCSP, CRS has the advantage that its replies are smaller and always preprocessed.

When CRS was first described in 1996 [12], this scheme was estimated to be 900 times cheaper than CRL in terms of communications costs. Note, however, that these calculations did not consider that the CRLs can be cached on the End-Entities. Furthermore, none of the CRL based mechanisms using CRL extensions was considered. In the example used in [12], there were 5 revocations per End-Entity per day. For this example, CRS is only 180 times cheaper than CRL if CRL caching is utilized. The cost of the schemes become more comparable if the CRL extensions are taken into consideration.

Since CRS takes the approach of using an on-line server, CRS is also comparable to OCSP. CRS has the advantage that its replies are always preprocessed, and its replies are intended to be smaller.

Timeliness

As initially described [12], the CRS scheme is intended to update the revocation status of each certificate once a day. However, at the cost of more processing in both the CA and in the CRS Directory, as well as more communication between them, the update rate could easily be increased to provide higher timeliness.

Scalability

The increased performance alone suggests that the CRS scheme is more scalable than CRLs. However, it is important to realize that the processing involved in the CA and in the CRS Directory are potential bottlenecks in a large system if the timeliness of the scheme is improved. Also, the CRS Directory must be capable of handling a request for each certificate validation.

Furthermore, CRS is more scalable than OCSP, as the replies are preprocessed. There is no need to calculate replies with full digital signatures, which is one of the main bottlenecks in OCSP. In this way, a smaller number of servers is necessary with CRS.

Security

As the lightweight digital signatures are computed at the CA, the CRS Directory is not able to manipulate the revocation status of a certificate. The scheme is 2-party, and only the CA have to be trusted.

Micali [12] claimed that CRS demonstrates a property that he called *completeness*. Here, completeness means that the CRS Directory is capable of providing a complete and satisfactory answer to any possible query. In particular, the CRS Directory is capable of proving to an End-Entity that a certificate serial number n does not belong to any not-yet-expired certificates issued by a given CA by providing the authenticated bit-string representing all certificates. However, the importance of this criteria is arguable, and it is not considered essential in this thesis, as it is strictly spoken outside the scope of the certificate revocation schemes. Also, with the current certificate standards, the certificate itself identifies the CA that issued it and the relevant revocation information source.

Standards Compliance

The CRS scheme presented the idea of reducing the size of the data structures for certificate revocations, and it was patented [25]. There are no commercial CRS products, and CRS have not been through a standardization path like the CRL based schemes.

Expressiveness

The CRS scheme does not provide any means of informing the End-Entities about the circumstances of a revocation (e.g. revocation reason, revocation date etc.).

Scheme Management

With this scheme, both the CA and the CRS Directory are required to process each certificate. In particular, the CRS Directory verifies each certificate revocation. This implies that the Server have to have knowledge of the scheme, and it have to be managed as a part of it. This scheme cannot by itself provide secure archiving of revocation information because of the lack of expressiveness. Another mechanism (e.g. a proprietary mechanism or CRL) may have to be deployed.

On-line vs. Off-line

Since CRS is based on per-certificate inquiries, the End-Entity have to make a connection to the CRS Repository for each certificate. This scheme is consequently not suitable for off-line use.

6.2 Hierarchical Certificate Revocation Scheme (HCRS)

The *Hierarchical Certificate Revocation Scheme* (HCRS) [26] 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. In HCRS, assume that 2^l certificates are denoted by binary vectors of length l. Another 2^{l-1} vectors of length $l-1$ are made parents of the first 2^l vectors. This pattern is recursively repeated until a tree has been created with an empty root ϕ. A simple example of such a tree is show in in figure 6.1 with 8 certificates.

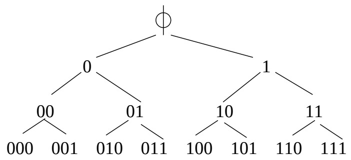

Figure 6.1: HCRS tree structure with 8 certificates

Let F be a one-way hash-function and D the validity time of the certificates. For each node, calculate the sequence $(r, F(r), F^2(r), ..., F^D(r))$, where r is a random initial value for that node. This construct is similar to the CRS Y and N computations, except that the values $F^D(r)$ of all the nodes on the path from the leaf v to the root ϕ are included in v's certificate.

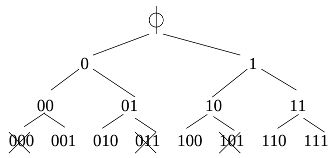

Figure 6.2: HCRS with 3 revoked certificates

Let the set R_i be the list of revoked certificates up to day i. Figure 6.2 shows the example with 8 certificates where $R_i = \{000, 011, 101\}$. At the end of day i, a CA calculates a subset S of *day-i verification nodes* of the binary tree according to the following two criteria:

- For every certificate $v \notin R_i$ there is at least one node on the path from v to the root ϕ which is not a day-i verification node

- For every certificate $w \in R_i$ none of the nodes on the path from v to the root ϕ is a day-i verification node

This subset is called the set of *day-i verification nodes*. The day-i verification nodes for the example above is shown in figure 6.3.

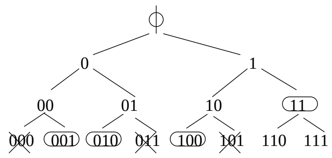

Figure 6.3: HCRS with 4 day-i verification nodes

With this scheme, each non-revoked certificate are exactly those for which there exist a day-i verification ancestor (where ancestor includes the node itself), and the revoked certificates are exactly those for which none of the ancestors are day-i verification nodes. It is possible to use an algorithm for selecting a *minimal* subset S in order to optimize the size of the data-structure.

The CA now sends the values $F^{D-i}(r)$ for each day-i verification node to the Server. The validity of a certificate on day i is demonstrated by sending any of the values $F^{D-i}(r)$ from the path of the certificate in question to the root. The Server will not be able to produce such a value if the certificate is revoked. The End-Entities can verify the day-i verification nodes by calculating $F^{D}(r)$ from $F^{D-i}(r)$ by applying the function F i times.

Note that the paper first presenting HCRS [26] also suggested a generalization of the scheme that increases the performance of the scheme. The name HCRS was used in [13], but it is not applicable to the generalized scheme. The generalized scheme is not hierarchical.

6.2.1 Analysis of HCRS

Performance

HCRS improves CRS by reducing the overall CA to Repository communication, while maintaining the low communication load (about 128 bits per reply) between the Repository and the End-Entities. HCRS is demonstrated to always require less communication from the CA to the Repository compared to CRS [26]. Note that the peak request rate from the End-Entities is the same as for CRS.

Timeliness

HCRS is, similarly to CRS, designed to provide new certificate revocation information once every day. There is, however, nothing in the scheme that provides it from operating with shorter periods at the cost of more communication and processing.

Scalability

Since HCRS improves the performance of CRS, it is to be expected that HCRS is potentially more scalable. However, the same considerations regarding CA and Server processing as well as Server requests are the same for HCRS as for CRS.

Security

HCRS is only capable of providing proofs that certificates are *not* revoked. There is no concept similar to the N in CRS, so both the CA and the Server have to be trusted to some degree. The scheme is basically a 3-party scheme.

Standards Compliance

No standardization work or reference implementations of this scheme has been published.

Expressiveness

The HCRS scheme does not provide any means of informing the End-Entities about the circumstances of a revocation (e.g. reason, date etc.).

Scheme Management

The same management considerations as for CRS apply.

On-line vs. Off-line

Since HCRS is based on per-certificate requests, the End-Entity have to make a connection to the Server for each certificate. This scheme is consequently not suitable for off-line use.

6.3 Certificate Revocation Trees (CRTs)

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 [27]. CRTs are based on a sorted list of revoked certificates. The sorted list can be built by obtaining CRLs (possibly from severals CAs) and concatenating the CA's public key or Distinguished Name (DN) in front of each certificate serial number to make each entry unique.

The list of all 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 its two child nodes concatenated. 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.

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 cryptographically proved that the certificate is not revoked.

At the End-Entity, this verification is done by obtaining the *supporting nodes* 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 be time-stamped and *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.

6.3.1 Analysis of CRT

Performance

The CRT provides a mechanism for verifying the revocation status of a single certificate rather than a list. This leads to more requests than the CRL based schemes, but the replies are considerably smaller. An estimate from ValiCert [28] claims that the Freshness Proof is typically less than 8000 bits. Also, if the scheme allows an End-Entity to *piggyback* the freshness proof with its certificate, there is no extra connections for the verification. However, the certificate itself is increased in size with the size of the Freshness Proof.

Furthermore, the authenticity of the CRT is demonstrated by the iterative hashing into a tree followed by the digital signing of the root node. The only information needed to verify the status of a certificate is the data-structure associated with it, as well as all hashes on the path up to and including the root node. Only one public key operation on a relatively small data element is required in addition to the (inexpensive) hash operations.

Compared to CRS and HCRS, the CA to Server communication is superior, but the verification on the End-Entity is smaller in CRS and HCRS [26]. The sizes of CRS replies are in the order of 100 bits, whereas the CRT Freshness Proof is in the order of 8000 bits.

Note that a major disadvantage with the CRT tree is that it have to be recomputed every time it is changed. This places a significant processing load on the CRT Issuer whenever the certificate revocation information is updated.

Timeliness

The CRT scheme provides different levels of timeliness based on the CRT validity period. Very timely revocation information can be acquired by accepting only very recent timestamps in the freshness proofs. In this case, one connection and one public key operation is required for each certificate to be checked, as the piggybacking of revocation information would be likely to always provide expired information. Note that the recomputation of the CRT is a limitation on the timeliness as well.

Scalability

This system is designed to scale well, as the number of CRT issuers can be increased. Also, the amount of supportive nodes and operations on the nodes are restricted to $20log_2(N)$ [27] (N is the total number of certificate, or in this case, End-Entities), so that the size of the data required to prove the revocation status of a certificate is always relatively small.

Security

Compared to CRLs, the major issue regarding CRT security is that the CRT Issuers accepts revocations from some other source. This implies that the CRT issuer have to be trusted to verify the authenticity of its source information and compute the CRT without changing it. In other words, this scheme is 3-party, and both the CAs and the CRT issuers have to be trusted.

Standards Compliance

CRTs are patented and implemented in a toolkit by ValiCert.

Expressiveness

ValiCert claims [27] that all supporting information found in CRLs can be included in the revocation information.

Scheme Management

With this scheme, it is necessary to manage both CAs and CRT issuers in a secure manner. The CA can issue revocation information to the CRT issuers, and the CRT issuers computes CRTs based on this information. As with OCSP, this scheme needs a *back-end mechanism* unless the CRT issuer is closely integrated with the CA itself. Since CRT has a high expressiveness, either the back-end mechanism or the CRTs can be used for the purpose of secure archiving.

Note that each End-Entity must have the necessary software toolkit to compute the necessary hashes and verify the root of the tree with the CRT issuer's public key.

On-line vs./ Off-line

CRTs are not well suited for off-line use unless it can be assumed that each certificate piggybacks valid revocation information about itself. If this is the case, it places a considerable restriction on the timeliness of the scheme.

6.4 Naor's and Nissim's Scheme

Naor's and Nissim's scheme [29] extends CRTs by using a more suitable structure than a binary tree. A problem with CRTs is that they have to be recomputed at every update. 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 2-3 tree has the following properties that can be utilized in this scheme:

- Testing membership, inserting or deleting a single element are done in logarithmic time
- Inserting and deleting an element only affect the nodes on the insertion/ deletion path

With this scheme, every insertion or deletion of a leaf only affects a few interior points, so that there is no need to recalculate the entire tree. Furthermore, it is not necessary to use any kind of additional data structure for the statements. The tree issuer (in this case the CA) inserts the plain serial numbers of the revoked certificates into the leaves and computes the values on the paths of these nodes. The result is sent to the Revocation Server which also performs the computations itself and checks the result against the CA signed root hash value. The signed root hash value is used by the End-Entities to ensure the authenticity of the tree.

In order to obtain the revocation status of a certificate, an End-Entity sends a request to the Revocation Server with the certificate's unique identity. If the certificate is revoked, the Revocation Server replies with the leaf (with the unique number) and its supporting nodes (as for CRT). If the certificate is not revoked, the Revocation Server replies with two neighboring leaves that are smaller than and greater than the unique number respectively accompanied by their supporting nodes. In this way, the End-Entity can calculate all nodes on the path to the root using the hash function, and then verify the digital signature of the root node. Then this is completed, the certificate's revocation status is proved by the presence or absence of the certificate's unique number in the revocation tree.

6.4.1 Analysis of Naor's and Nissim's Scheme

Performance

This scheme improves on the CRT scheme by using a *2-3 tree* rather than a binary tree. With this data structure it is not necessary to recompute the tree for each revocation information update, and the CA to Server communication is lower. Testing membership, inserting and deleting a single element are done in logarithmic time. Also, insertion and deletion of an element only affect the nodes on the path in question.

Naor and Nissim [29] claim in their evaluation that the communication load of this scheme is similar to the load for CRT. In a simple evaluation, it is also indicated that this scheme requires far less communication between the CA and the Revocation Server than both CRL and CRS, whereas the communication load between the Revocation Server and the End-Entities is larger than CRS but smaller than CRL.

Timeliness

This scheme is, like CRT, based on the updating of a tree. This updating can be done frequently at the cost of higher processing. Note that the processing task is much smaller than for CRT.

Scalability

Naor's and Nissim's Scheme is also designed to scale well. In particular, it is an improvement of the CRT scheme when it comes to processing.

Security

In this scheme, the Revocation Server is capable of proving either that a certificate is revoked or that a certificate is not revoked. The CA's digital signature on the root node provides security against tampering from the Revocation Server or other parties. The scheme is 2-party, and only the CA have to be trusted.

Standards Compliance

No standardization work or reference implementations of this scheme has been published.

Expressiveness

Like CRS and HCRS, this scheme compresses the revocation information in such a way that information regarding each certificate revocation is not preserved.

Scheme Management

Both the CA and the Revocation Servers must be managed with respect to the scheme. The CA produces and updates the 2-3 tree, whereas the Revocation Server verifies and implements the changes in its own 2-3 tree (a copy of the CAs tree). This scheme is inadequate for secure archiving, as it has a lack of expressiveness. A separate mechanism (e.g. CRLs) have to be used for logging purposes.

On-line vs./ Off-line

This scheme does not define piggy-backing as in CRT or any other form of caching. Consequently, one connection must be made for each certificate validation. This scheme is not suitable for on-line use.

Chapter 7

Selected Other Certificate Revocation Mechanisms

This chapter presents some additional techniques that have been suggested as complements to revocation schemes or as individual revocation mechanisms in themselves. Some of these techniques are in regular use (e.g. removing expired certificates), whereas others are considerations that may be taken in any revocation system (e.g. the certificate validity period and the issue of certificate suspension).

7.1 Short Certificate validity period

It is possible to obtain more timely certificate revocation by decreasing the certificate validity period. If the validity period is very short, the certificate End-Entity is forced to request a new certificate often. Instead of revoking a certificate, the CA can simply refuse to issue a new certificate. The certificate validity period is determined by policy, and it is in this case based on the maximal time the organization is willing to risk exposure to outdated revocation information.

This method may involve a high management and operational costs. There is more inconvenience involved (including for the End-Entities), and such a system may require a large secure archiving resource.

Note, however, that the expiration date of the certificates in general should be used in a way so that unnecessary certificate revocations do not occur. In this way, it is possible to *reduce the amount of revocation information* in any scheme by choosing appropriate certificate expiration dates. As an example, a temporary employee in an organization should get a certificate expiring no later than his last expected day at work. If some default expiration date is used (e.g. 1 year), a revocation *will* occur when a short-term employee is finished. In this way, the minimal expected validity of a certificate can be used as its expiration date.

7.2 Certificate Suspension

In some cases, it might be unclear whether a certificate revocation is required or not. This would be true in the case of a *suspected* public key compromise. There are also other situations where a temporary revocation of a certificate could be required, as during a job leave, suspension etc. Rather than revoking the certificates in question permanently, a certificate suspension may be advantageous. This would not only help reducing the number of revoked certificates, but also reduce management costs in issuing new certificates. Note, however, that certificate suspension is a controversial technique, and the usefulness is arguable.

An issue with certificate suspension is how the End-Entity should manage a suspended certificate. The End-Entity has to have a policy for how to handle a certificate suspension and how to continue with a certificate validation. As an example of certificate suspension, a CRL issuer can suspend a certificate in a CRL. The certificate in question will either expire suspended, be permanently revoked or be removed from the CRL before expiry (indicating reinstatement). The CRL itself has a *hold instruction code* extension that instructs the End-Entity on how to proceed with the certificate validation in the case of suspension.

In the case where a certificate can be reinstated, the advantages of certificate suspension are that there is no need for certificate re-issuance, and that the revocation information for the certificate is removed from any revocation lists. In the case where a certificate have to be permanently revoked, the revocation entry is simply changed, and the revocation will be known from the start of the next revocation information validity period.

7.3 Removing expired certificates from revocation lists

It is important to have a system for removing expired certificates from the list of revoked certificates. Keeping old certificates in the list would be a waste of resources, since the certificate validation process reads the expiry date from the certificate itself. Note that this is common practice in existing certificate revocation schemes. An important consideration related to this mechanism is that there may be requirements for having a repository for secure archiving of expired, revoked certificates. CRL or some other scheme appropriate for archiving may be used for this purpose.

7.4 Full Revocation Certificates

For an expressive and individual treatment of certificate revocations, Full Revocation Certificates can be used [12]. A Full Revocation Certificate is a data-structure containing the revoked certificate's serial number, revocation date, revocation reason etc., and it is digitally signed with the CA's digital signature. This scheme might reduce the communication needs compared to the basic CRL scheme, but the processing load from signing the revocation certificates is significant. Also, this mechanism is likely to behave like an on-line protocol (like OCSP), and the End-Entity either have to request the revocation certificate from the Repository or request it from the CA (and increase the CA load).

Chapter 8

Suggested Improvements for Existing Schemes

Some mechanisms that attempt to improve existing certificate revocation schemes are developed in this chapter.

8.1 Improvements for the CRL scheme and its extensions

The main problem with CRLs is their size, which in turn causes problems with scalability and timeliness. In order to improve the CRL scheme, reducing the CRL size is a critical issue. Several improvements to CRL have been standardized, and many of these can be combined. This section explores several ways of further improvements on the CRL scheme. Most of these improvements are also compatible with the extensions to CRL.

8.1.1 Communication Improvements with Sloppy CRL Broadcasting

The current CRL scheme and its extensions are designed in such a way that each End-Entity requests a new CRL whenever it validates its first certificate in a CRL validity period. If the CA *Broadcasts* or *Multicasts* the CRL rather than waits for individual requests, significant network resources can be saved. CRL Broadcasting was discussed in [8], and it was referred to as a *push* mechanism as opposed to the *pull* mechanism usually used in CRL based schemes. The following problems were identified in [8]:

1. Ensuring that CRLs reach their intended destinations undamaged
2. A massive amount of traffic is generated if broadcasting were to be used to notify about all revocations
3. Deciding which revocations need to be advised to which certificate-using systems

These problems can all be addressed by having a *sloppy* CRL Broadcast or Multicast mechanism. In this context, the sloppy mechanism refers to a mechanism that is not always successful in distributing the revocation information. For the cases that are not successful, there have to be a backup mechanism.

In the Sloppy CRL Broadcast scheme, the CA compiles and broadcasts the CRL to a network of End-Entities known to be users of the CA's certificates slightly before the previous CRL expires (in order to allow the broadcasting to be completed before the expiry). All active End-Entities accept the CRL, verify that it is correct or delete it if this could not be verified. At this point, a significant part of the CRL-using End-Entities can be expected to have a new and correct CRL. End-Entities that were not on the address list or that have invalid CRLs have to use the backup mechanism and request a new, valid CRL when the old CRL expires. In this way, problems 1 and 3 above have been solved by allowing the broadcast to be sloppy. Problem 2 refers to the case where the CA broadcasts revocations immediately. Given that the broadcasts are periodical, this problem is not relevant.

If the End-Entities are not collected on a single network, but distributed over several networks, a Sloppy CRL Multicast scheme can be used instead. This scheme behaves much like the Sloppy CRL Broadcast scheme, except that the CRL is addressed to a list of End-Entities or networks rather than being addressed to all End-Entities in a single subnetwork. The disadvantage of multicasting is that it is more complex and less efficient than broadcasting.

Note that these sloppy schemes are most useful where the broadcasting or multicasting is efficient. This is the case when the End-Entities are located on a minimal number of subnets (i.e. in a LAN or Intranet environment). If this is not the case, multicasts are split at the routers and directed towards a high number of subnets. However, the load at the Repository subnet will be unchanged, and there will be few or no requests for the rest of the period. In this way, this scheme is most efficient in a LAN environment, but it should still be considered even for WAN environments.

In the best case, a sloppy CRL broadcast/ multicast would result in a single CRL transmission for each CRL validity period rather than approximately 1 CRL transmission per End-Entity. The improved performance can be used to provide higher scalability, higher timeliness, or simply a more streamlined scheme. This method can also be combined with most of the CRL based schemes.

As an example of the Sloppy CRL Broadcasting scheme, assume an Ethernet office LAN on one subnet using TCP/IP. The basic CRL scheme is used as the backup mechanism and delta CRLs are used to achieve high timeliness. The CRLs are broadcasted every morning, whereas the delta CRLs are updated every hour. If the office has a policy of keeping the computers on at night, the CRL can be multicasted every morning before office hours. At 7AM every morning, the CRLs are broadcasted, and all computers that are available accept the CRL broadcast. Each computer must have a server program running in order to accept a broadcast. The network load of the broadcast operation is exactly the same as transmitting a single CRL to any one individual computer, because any transmission in an Ethernet network is transmitted to all its hosts, but only accepted by its addressee. In this case, all the hosts are addressed. When the employees come to work, some computers might not have received the multicast, and they have to request the CRL. Also a new delta CRL is available for download every hour (although the delta CRLs could be multicasted as well).

Another Sloppy CRL Broadcasting example is illustrated in figure 8.1, where the CA publishes the CRL to the Repository, which in turn Broadcasts the CRL to the 6 End-Entities. Figure 8.2 shows how End-Entity 3 (EE3) failed to receive the CRL Broadcast, and requests the CRL from the Repository. Figure 8.3 demonstrates the situation when End-Entity 3 (EE2) and End-Entity 6

(EE6) requests the first Delta CRL in order to obtain timely revocation information.

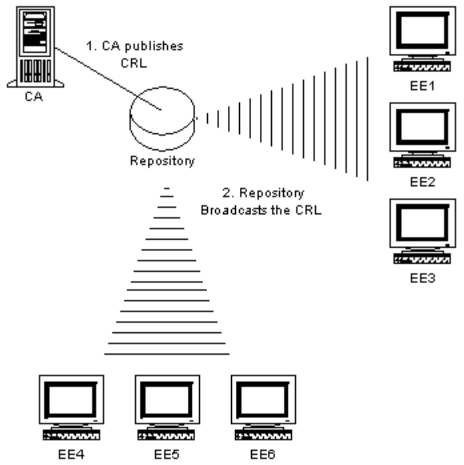

Figure 8.1: Example Sloppy CRL Broadcast system, Broadcast phase

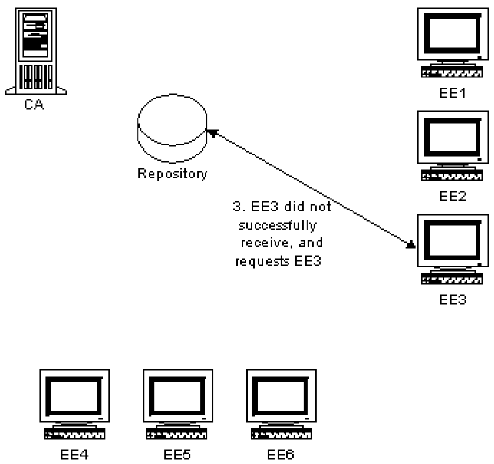

Figure 8.2: Example Sloppy CRL Broadcast system, Request phase

As an example of the Sloppy CRL Multicasting scheme, assume a corporate WAN consisting of multiple Ethernet subnets using TCP/IP. Only some of the hosts on each subnet are supposed to receive the multicast. As before, the basic CRL scheme is used as the backup mechanism and delta CRLs are used to achieve high timeliness. The CRLs are multicasted at 7AM every morning, and delta CRLs are published every hour. The multicast is transmitted to the routers of the involved subnets, and the routers transmit the CRL to each computer that exist on the respective subnet and that are listed in the multicast address list. A multicast is less efficient than a broadcast, but

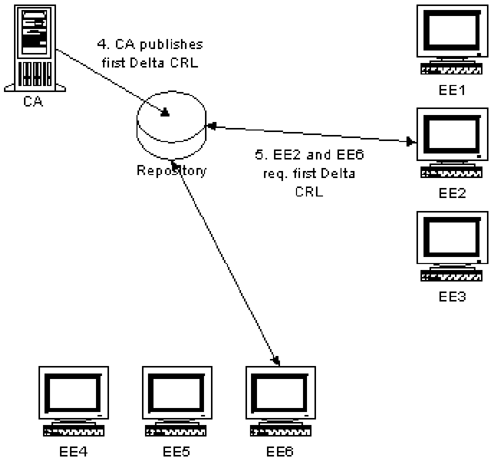

Figure 8.3: Example Sloppy CRL Broadcast system, Delta CRL request phase

it may be more versatile.

8.1.2 Communication Improvements with CRL Compression

A simple technique that can be used to reduce the network load from certificate revocation lists is to compress the data in the CRLs. The compression can be done either by the CA or by the Repository, and it only needs to be done once per CRL. The decompression is done at the End-Entities, and can either be done in memory or written to disk. This method is particularly advantageous for networks where the End-Entities are personal computers with the processing power to perform fast decompressions.

8.1.3 Reducing the number of CRL Entries with Probabilistic CRL Entries

The CRL scheme can potentially be improved by allowing each CRL entry to represent multiple certificates where one or more certificates are revoked and some are non-revoked. In this scheme, the absence of a certificate serial number means that the the certificate is not revoked, whereas if the certificate matches an entry it might be revoked. In the last case, the End-Entity is required to obtain revocation specifically for the certificate in question to check its revocation status. For this purpose, the End-Entity may request the full CRL, request the status from an OCSP Server or use some other method. The loss of expressiveness in the Probabilistic CRL Entries is irrelevant, since an expressive source is consulted in the case of an actual revocation.

The entries can be produced by calculating a non-cryptographic hash of a serial number or using only a limited amount of the bits in the number. Each of these techniques will make an entry represent at least one revoked (possibly several) certificate(s). The savings of this system is that the periodic distributions are much smaller than CRL, while the number of requests for the full information is significantly reduced.

8.1.4 Reducing the number of CRL entries with CRL Filters

The size of CRLs can also be reduced by computing entries that covers multiple revocations. Each entry would be built up of a *base* serial number with a *filter* that indicates how the entry covers multiple certificate revocations. The base serial number has a pattern that matches one or more revoked serial numbers, whereas the filter modifies the base serial number to describe multiple revocation by identifying bits in the base serial number that does not matter (i.e. “don’t care bits”).

The main problem with this scheme is the amount of computation involved to build the filters. Also, the serial numbers for the involved certificates have to be similar (counting upwards rather than randomly chosen) so that there is a high chance of finding revoked certificates whose serial numbers deviate in only one bit. Note that this method implies a degrading of the expressiveness in the CRL scheme. The data-structure is reduced (like CRS, HCRS and Naor and Nissim’s scheme), and individual information about each revocation is not available.

The algorithm needed to compute the filters have to recursively find couples of serial numbers that deviate in one and only one bit, and substitute this with an X. The X indicates that it does not matter whether the corresponding bit in the base serial number is 0 or 1. This can be implemented in the CRLs by adding a CRL entry extension called a *filter*. This filter is a bit-string with the same length as the serial numbers, in which 0 is ignored while 1 means that the corresponding bit in the base serial number is an X.

As an example, the revoked certificates 0000 and 0100 can be presented as the revoked certificate 0000 with the filter 0100. If the certificates 0101 and 0001 were revoked as well, the four revocations {0000,0001,0100,0101} can be represented as the revoked certificate 0000 with the filter 0101. Also, there may be more than one way to build the base serial number and the filter. For more complex cases, there is a trade-off in whether the algorithm should be capable of optimizing the filters (with more processing) or allow the CRL to be longer than optimal (with higher network load).

8.1.5 Reducing entry size with Implicit bits

If the range of serial numbers is larger than the amount of certificates managed by a CA, it may be the case that some of the bits in the serial numbers are always the same. In this case, these bits could be specified elsewhere (e.g. in a CRL extensions) and considered implicit for each CRL entry. In particular, this method could be useful for the CRL Distribution Point schemes with segmentation based on a range of serial numbers. In this case, the range of serial numbers for a distribution point is, by definition, smaller than the amount of certificates managed by the distribution point.

8.2 A back-end for OCSP

OCSP is intended as a real-time protocol, but the specification is focused on providing a real-time reply to the End-Entities rather than to keep the Servers themselves updated. This section discusses some possibilities for providing an efficient back-end for OCSP that is capable of providing fresh revocation information.

8.2.1 OCSP Replication

A number of replication protocols have been developed to ensure proper replication of data-bases and file systems. Database management system (DBMS) providers like Oracle and Sybase have developed highly efficient and secure replication mechanisms for replicating data between distributed computers. Such a replication protocol or system could be used to provide the same services to OCSP Responders in order to provide revocation information with very high timeliness. As an example, a DBMS could be managed at the same computer as the CA, and be continuously updated by the CA. Any changes to this DBMS would then be replicated to all other DBMSs in the domain. These DBMSs could then be equipped with an OCSP interface that allowed them to reply to OCSP requests as required.

8.2.2 An OCSP Pre-computation method

The pre-computation of OCSP responses has been proposed, and was discussed in chapter 5. OCSP pre-computation reduces the problem of signing every response real-time, and it solves the problem of signing multiple responses containing the same information about the same certificate. One of the main disadvantages with pre-computation, however, is that OCSP allows for bulk requests, so that the number of possible responses becomes prohibitive for any system of size. Some systems have solved this problem by not allowing bulk OCSP requests.

For a system where both the number of single and multiple certificate requests are large, a method that takes advantage of pre-computation, yet allows bulk OCSP requests could prove useful. One such system can be implemented using the following rules:

- Precompute all responses with single entries at the start of a validity period
- If a response is pre-computed, use the pre-computed response
- As responses with multiple entries are computed, cache them as precomputed replies

Note that the pre-computation of OCSP replies implies the use of caching mechanisms. Consequently, a timeliness consideration is raised. A policy should dictate how long OCSP replies can be cached.

Chapter 9

Comparison of Existing Certificate Revocation Schemes

As shown in the previous chapters, the different certificate revocation schemes have different properties. This chapter 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 three general methods and several other techniques have been suggested to improve its general performance. 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 have 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.

The final class of schemes that has been presented is the schemes with reduced data-structures. These schemes present the revocation information in 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 and full revocation certificates, as well as the communication load involved in e.g. CRL.

9.1 Comparison of Performance, Timeliness and Scalability

The tables 9.1, 9.2 and 9.3 compare the performance properties of the analyzed certificate revocation schemes. The timeliness and scalability criteria are closely related to the system performance, and they are included in the same tables. Note that a performance gain can be used to improve timeliness or scalability, even though this is not the primary improvement of the scheme. This comparison mainly shows the *primary* improvements, whereas the *secondary* improvements are noted with “allows improvement”. As an example, Over-Issued CRL provides a reduced peak request rate which in turn enables a higher timeliness and/ or scalability within the same network constraints. This secondary effect is shown in the table 9.1 as “allows improvement over CRL”.

9.2 Comparison of Security, Standards Compliance, Expressiveness, Scheme Management and Off-line/ On-Line

The tables 9.4, 9.5 and 9.6 compare all the analyzed schemes with respect to the criteria that are not directly related to the performance. These criteria are security, standards compliance, expressiveness, scheme management and on-line vs. off-line. These criteria are relevant in deciding whether a certificate revocation scheme is suitable based on the nature of an organization or its policies.

Scheme	Performance	Timeliness	Scalability
CRL	High peak req. rate/load	Low	Low
CRL Distribution Points	High peak req. rate/load	Allows improvement over CRL	Improves CRL
Delta CRL	High peak req. rate/load	Improves CRL	Allows improvement over CRL
Over-Issued CRL	Improves CRL	Allows improvement over CRL	Allows improvement over CRL
Over-issued segmented CRL	Improves CRL Distr. Points	Allows improvement over CRL DP	Allows improvement over CRL DP
Sliding window Delta-CRL	Improves Delta-CRL	Allows improvement over Delta CRL	Allows improvement over Delta CRL
Over-issued Delta-CRL	Improves Sliding Wind. Delta-CRL	Allows improvement over Sliding wind.	Allows improvement over Sliding wind.
ARL	Improves systems w. CA certs	Allows improvement over CRL w. CA certs	Improves CRL systems w. CA certs
Indirect CRL	Improves req.rate w. many small CRLs	Allows improvement over CRL systems w. many small CRLs	Improves CRL systems w. small CRLs
Enhanced CRL Distr Points	Provides load balanced CRL Distr Points	Allows improvement over CRL DP	Improves CRL Distr Points
Fresh Revocation Info	Better utilization	Improved over CRL for critical End-Entities	Allows improvement over CRL or Delta CRL

Table 9.1: CRL and CRL extensions Performance Comparison

9.3 Summarizing the comparisons

Table 9.7 summarizes the comparison of the different schemes according to the criteria. This table is intended to provide a basis for selecting appropriate schemes for different scenarios. The different schemes are ranked on a subjective scale from 1 to 3 for each criteria:

1. Poor or none
2. Modest or improved
3. Good or deliberately improved

For the *security* criteria, rank 1 implies that the scheme has security flaws, rank 2 implies that the scheme has a non-repudiation problem or is 3-party, whereas rank 3 implies that the scheme does not have any significant security issues and is 2-party. For the *management* criteria, 1 implies that the

Scheme	Performance	Timeliness	Scalability
OCSP	High processing load	High (depending on back-end)	Linear growth, but many OCSP Responders
SCVP	As OCSP	As OCSP	As OCSP
OCSP-X	As OCSP	As OCSP	As OCSP
DCS	As OCSP	As OCSP	As OCSP

Table 9.2: On-line Revocation Status Checking Protocols Performance Comparison

Scheme	Performance	Timeliness	Scalability
CRS	Reduced data, increased req. rate	1 day / low	Higher than CRL
HCRS	Improves CRS	1 day / low	As CRS
CRT	Improves CRS in some ways	high unless caching is allowed	High
Naor/Nissim	Improves CRT	Allows improvement over CRT	Allows improvement over CRT

Table 9.3: Reduced Data Structure Performance Comparison

scheme is very complex to implement and manage and that a separate secure archiving scheme has to be deployed, 2 implies that the scheme have to be complemented with a separate secure archiving scheme or is fairly complex, whereas 3 implies that the scheme is fairly non-complex and inherently supports secure archiving.

Note that the rankings are highly subjective. The justification for the ranking should be apparent from the earlier tables in this chapter or from the analysis of the individual schemes in earlier chapters. Also, two schemes with the same high ranking for the same criteria do not necessarily perform the same. The ranking only indicates that both have been designed to perform well for that particular criteria.

Scheme	Security	Standards Compl	Expressiveness	Scheme Mgm	Off-line/ On-line
CRL	CA trusted	X.509	High		Suitable for off-line
CRL Distribution Points	CA trusted	X.509	As CRL		Not suitable for off-line
Delta CRL	CA trusted	X.509	As CRL		High timeliness, many connections
Over-Issued CRL	CA trusted	X.509	As CRL		As CRL
Over-issued segmented CRL	CA trusted	X.509	As CRL		As CRL Distribution Points
Sliding window Delta-CRL	CA trusted		As CRL		As Delta-CRL
Over-issued Delta-CRL	CA trusted		As CRL		As Delta-CRL
ARL	CA trusted	X.509	As CRL	Simpl. CA hierarchies	As CRL
Indirect CRL	CA and Issuer trusted (3-party)	X.509	As CRL	Simpl. End-Entity	As CRL
Enhanced CRL Distr Points	CA trusted	FPDAM	As CRL	Reconfigurable	As CRL Distribution Points
Fresh Revocation Info	CA trusted	FPDAM	As CRL	Allows discrimination	Enables differences for on-line and off-line users

Table 9.4: CRL and CRL extensions Properties

Scheme	Security	Standards Compl	Expressiveness	Scheme Mgm	Off-line/ On-line
OCSP	CA and Server trusted	IETF PKIX RFC2560	High (as CRL)	Requires back-end mechanism	Not suitable
SCVP	CA and Server trusted	IETF PKIX Internet Draft	High	Requires back-end mechanism	Not suitable
OCSP-X	CA and Server trusted	IETF PKIX Internet Draft	High	Requires back-end mechanism	Not suitable
DCS	CA and Server trusted	IETF PKIX Internet Draft	High	Requires back-end mechanism	Not suitable

Table 9.5: On-line Revocation Status Checking Protocols scheme Properties

Scheme	Security	Standards Compl	Expressiveness	Scheme Mgm	Off-line/ On-line
CRS	CA trusted	Patented	None	Requires arch. mech	Not suitable for off-line
HCRS	CA and Issuer trusted (3-party)		None	Requires arch. mech	Not suitable for off-line
CRT	CA and Issuer trusted (3-party)	Patented and implemented	As CRL		Usable off-line if piggy-backing
Naor/Nissim	CA trusted		None	Requires arch. mech	Not suitable for off-line

Table 9.6: Reduced Data Structure scheme Properties

Scheme	Perf.	Time.	Scal.	Sec.	Standard	Expr.	Mgm	Off-line
CRL	1	1	1	3	3	3	3	3
CRL Distr Points	2	2	3	3	3	3	2	2
Delta CRL	2	3	2	3	3	3	3	2
Over-Issued CRL	3	2	2	2	3	3	3	3
Over-issued segm CRL	3	2	3	2	3	3	2	2
Slid. wind. Delta-CRL	3	3	2	2	1	3	3	2
Over-issued Delta-CRL	3	3	2	2	1	3	3	2
ARL	3	2	2	3	3	3	3	3
Indirect CRL	3	2	2	2	3	3	3	3
Enh. CRL Distr Point	3	2	3	3	3	3	3	2
Fresh Rev. Info	3	3	2	2	3	3	2	3
OCSP	2	3	3	2	3	3	2	1
SCVP	2	3	3	2	3	3	2	1
OCSP-X	2	3	3	2	3	3	2	1
DCS	2	3	3	3	2	3	2	1
CRS	3	3	3	3	1	1	2	1
HCRS	2	3	3	3	1	1	2	1
CRT	2	3	3	3	2	3	2	2
Naor/ Nissim	3	3	3	3	1	1	2	1

Table 9.7: Selecting a Certificate Revocation Scheme

Chapter 10

Selecting Certificate Revocation Solutions

The discussion in previous chapters (in particular chapter 9) has demonstrated the widely varying attributes of the existing revocation schemes. Different environments have different demands for the revocation schemes, so that a universally optimal revocation scheme can not be expected to exist. In order to select a suitable certificate revocation solution, different schemes have to be surveyed, and the more suitable scheme or combination of schemes have to be selected.

This chapter discusses some of the most important parameters relevant to selecting a certificate revocation scheme, and a guideline for selecting a revocation system is developed. To demonstrate the guideline, 4 example scenarios are presented, and the guideline is utilized in designing certificate revocation solutions for the scenarios.

Note that these scenarios do not directly correspond to the environments in table 3.1. The environments only takes the basic revocation parameters into consideration, whereas the scenarios are developed in order to better model real world systems. However, scenario 1 and 2 have closest resemblance to environment 1, scenario 3 has closest resemblance to environment 2, whereas scenario 4 has closest resemblance to environment 3.

10.1 Revocation Scenario Variables

This section briefly discusses some of the relevant properties and parameters used in designing some interesting scenarios. The scenario variables for each scenario are selected such that real world situations are approximated.

10.1.1 Evaluation Criteria

All the criteria in section 3.3.2 are relevant scenario criteria, and the demands with respect to each criteria vary for different scenarios. In general, the requirements for each criteria should be carefully analyzed when designing a certificate revocation solution. Having requirements that are too low may lead too poor performance, possible serious system exploits or security breaches, or other problems. Similarly, requirements that are too high may result in an unnecessarily complex and expensive solution. Furthermore, some of the criteria are working against each other, like off-line vs. timeliness. In these cases, a trade-off is necessary.

10.1.2 Revocation Parameters

As shown in section 3.2, an environment is described by the relevant revocation parameters. As for the defined environments, the following revocation parameters are used in discussing the revocation scenarios: Size (N), Certificate Verification Rate (v), Revocation Information Update Rate (r_u) and Revocation Rate (r_r).

10.1.3 Homogeneous/ Heterogeneous Systems

In selecting schemes based on the evaluation criteria and revocation parameters above, it is important to know whether the End-Entities are homogeneous or heterogeneous. If the End-Entities are homogeneous, a single revocation scheme may be sufficient. In a heterogeneous system, End-Entities can be expected to have different certificate revocation requirements. In this case, a single scheme might be suitable for some but not all End-Entities, and additional schemes or combinations of schemes should be considered.

10.1.4 Central/ Distributed Systems

Another interesting property with certificate revocation systems is whether it is central or distributed. In a central scheme, management of the End-Entities and the server (CA, Repositories or Revocation Servers) configuration is simplified compared to a distributed system. In a distributed system, management of the End-Entities might not be possible, and the configuration of the servers are more dependent on geographic parameters. Also, it might be impossible to manage the network load in a highly distributed system, as publicly available networks often are used.

10.2 A guideline for selecting a scenario solution

This section takes advantage of the results in chapter 9 in order to create a guideline (or method) for selecting a revocation solution for different scenarios. This guideline is intended to be useful for security infrastructure designers and PKI product developers in need of a certificate revocation solution. The schemes CRS, HCRS and Naor/ Nissim's schemes are not considered in this guideline, as they are neither standardized nor available on the market. Also, SCVP, OCSP-X and DCS are not considered in this guideline, as they are more general in their functionality and not dedicated certificate revocation status revocation protocol. These protocols are in many respects similar to OCSP. The Enhanced CRL Distribution Point and the Freshest Revocation Information schemes are expected to be included in the X.509 standard this year, and are included in the guideline for this reason.

Note also that some schemes are designed to supplement other schemes, or they could be implemented to do so. If this is taken advantage of, the efficiency (performance, timeliness, scalability) can be expected to improve at the price of increased complexity (and hence implementation and management costs). In general, if any CRL based scheme is considered, over-issuing and ARLs may provide better resource utilization and allow higher timeliness and scalability. Also, the Enhanced CRL Distribution Points scheme is more flexible and allows better load balancing than the CRL Distribution Points scheme. These mechanisms are examples of schemes that increase the complexity while improving the performance, timeliness and scalability.

Some methods for improving the efficiency at the cost of complexity was also suggested in chapter 8. In particular, Sloppy CRL Broadcasting, CRL Compression and Probabilistic CRL Entries may provide significant performance benefits to any CRL based scheme.

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" [4]. Consequently, do not specify requirements that are too strict for the purpose of the security infrastructure.

The following guideline should give the reader an idea of suitable revocation solutions for any 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. OCSP and 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. 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 (interoperability) for the scenario in question. If standards compliance and interoperability is important, CRT should be avoided, as it is a proprietary solution that is not widely supported. If the system is expected to interoperate with other PKIs using CRL based schemes, ARLs are highly recommended.

6. Specify the policy requirements for expressiveness for the scenario in question. All of the considered schemes support high expressiveness, but a high expressiveness requirement could prohibit some other schemes (e.g. CRS, HCRS and Naor/ Nissim's scheme).

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 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.

10.3 Scenario 1: A small, homogeneous and centralized scenario

The first scenario models a small business placed in a single office where the users have fairly similar information system needs. This scenario is characterized by a small number of on-line End-Entities ($N = 100$) in a centralized system with homogeneous End-Entities (with similar user-patterns and revocation parameters). Typically, such a system may be an Ethernet LAN, as shown on figure 10.1.

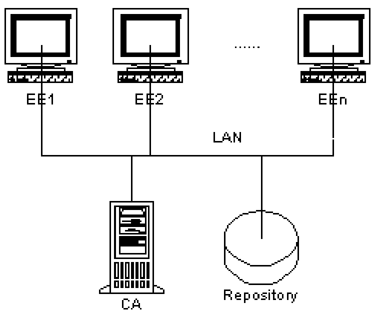

Figure 10.1: Scenario 1: Certificate revocation for a LAN System

This is a very simple scenario, and most of the schemes analyzed are in general capable of handling this scenario. The scenario has no strict requirements relative to the guideline, and the basic CRL scheme is sufficient.

Alternatively, the scheme may have some additional properties that implies some stricter requirements. Here are some relevant suggestions:

1. If the system is high-critical in some sense, the system might require *high timeliness* of revocation information according to point 2 in the guideline. A single *OCSP* Responder can be co-located with the CA and provide real-time revocation status checking. Delta CRLs can also provide a good level of timeliness for such a small system.

2. If the business is expected to grow rapidly, *scalability* should be considered in accordance with point 3 in the guideline. A single CRL distribution point can be implemented, allowing more distribution points to be transparently implemented as the system grows.

3. If the system is intended to be integrated with another PKI system or the Internet, a standardized scheme should be prioritized for interoperability in accordance with point 5 in the guideline. The CRL based schemes (with ARL) in particular, are open and tested standards that simplifies interoperability with other providers.

10.4 Scenario 2: A small, heterogeneous, distributed and mobile scenario

The second scenario models a small business ($N = 100$) based on field work (like some news/ media-companies and police stations). The employees are users of laptops, cellular phones, palmtops, and even digital cameras and recorders with digital signing capabilities. This scenario is characterized by a small number of both on-line and off-line End-Entities ($N = 100$) in a highly distributed and heterogeneous environment. Also, a relatively small number of End-Entities can be expected to be centralized at the same network as the CA and Repository/ Revocation Server. Note that some of the restricted units (e.g. digital cameras) do not need certificate revocation status checking capabilities. These units are only capable of signing information, but not reading signed data. Typically, such a system may be an Ethernet LAN with a wireless LAN or remote access extension, as shown on figure 10.2

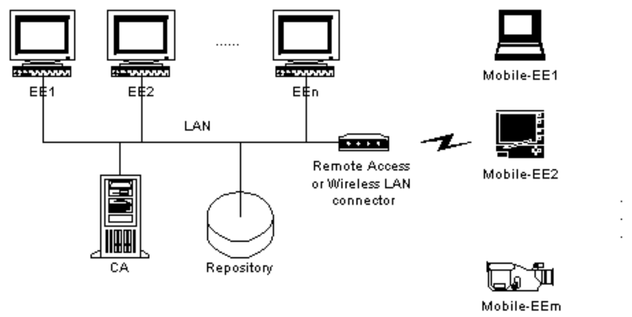

Figure 10.2: Scenario 2: Certificate revocation for a LAN with a wireless/remote access extension

Even though this scenario is small, the demands for the revocation system is far more complex than scenario 1. Here are some suggestions for selecting a certificate revocation solution for this scenario:

1. If the system is used for judicial documentation (e.g. police field work), it is important that the revocation information can be securely archived with information on a per certificate level. Also, the users of the system need updated revocation information in order to avoid falsified documentation from compromised End-Entities. In this case, *high expressiveness* and *high timeliness* is required in accordance with point 2 and 6 in the guideline. The high timeliness constraint is, however, only applicable to the central End-Entities, and possibly to a few distributed End-Entities in the case where they verify critical data. Full CRLs can be posted to periodically update all End-Entities. More timely revocation information can be implemented with optional delta CRLs with the Fresh Revocation Information scheme as described in point 8 in the guideline. The latter revocation information should always be obtained by the central End-Entities, and should be available for download to the distributed End-Entities when critical information is processed. The distributed End-Entities could decide whether timely revocation information is needed either at the users discretion or by tagging critical data so that downloading the freshest revocation information is mandatory. The fresh revocation information would be obtained by making a connection (e.g. by modem and cellular phone) to the Repository.

2. For a less critical system, the basic CRL scheme may still cover the need for a system of this size. The same considerations as for Scenario 1 is valid for a system that is expected to grow

rapidly or integrate with other PKI systems, except that OCSP and CRL distribution points are not recommended because of the off-line considerations.

10.5 Scenario 3: A medium, mostly homogeneous scenario with multiple sites

Scenario 3 models a big corporation ($N = 10000$) with one headquarter and a limited number of office locations. Also, about 10% of the corporation are frequent travelers, and are located at different locations in the world. This scheme is characterized by having multiple central sites, as well as a smaller number of distributed End-Entities. Also the user-patterns and revocation parameters can be expected to be relatively homogeneous for the central locations and the distributed End-Entities respectively. Typically, such a system may be a corporate Intranet or Virtual Private Network over TCP/IP, as shown on figure 10.3

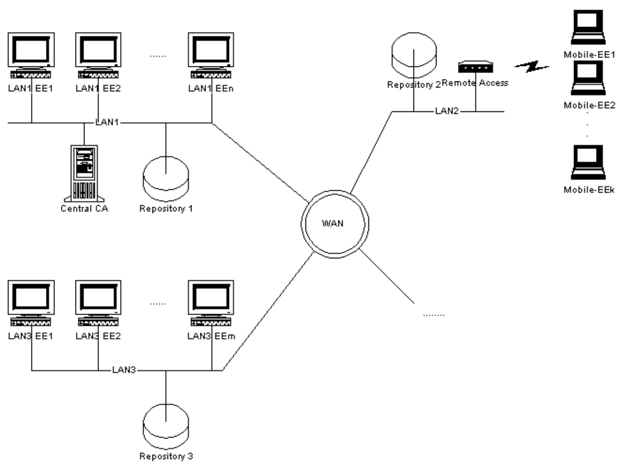

Figure 10.3: Scenario 3: Certificate revocation for a corporate Intranet

Since the corporation is partitioned into a limited number of office locations, it makes sense to place one or more distribution points or servers at each location. Here are some suggestions for selecting a certificate revocation solution for this scenario:

1. If *timeliness* is not a major constraint, CRL Distribution Points or Enhanced CRL Distribution

Points would cover the needs of the on-line End-Entities fairly well. The distribution points are, however, not very suitable for off-line users, so an additional scheme may have to be deployed. For this use, the off-line users could have the option of downloading the full CRLs.

2. If *timeliness* is important, solution 1 could still be used with the exception that delta CRLs optionally can be requested through the Fresh Revocation Information scheme. Alternatively, OCSP could be provided for the on-line End-Entities.

10.6 Scenario 4: A large, heterogeneous and highly distributed scenario

Scenario 4 models a large Internet user base with approximately 1 million End-Entities ($N = 1000000$). The PKI in question supports multiple Internet services, and a number of different client applications. This scheme is characterized by being highly distributed and highly heterogeneous with varying user-patterns and revocation parameters. This system is typically running on the Internet over TCP/IP, as shown on figure 10.4

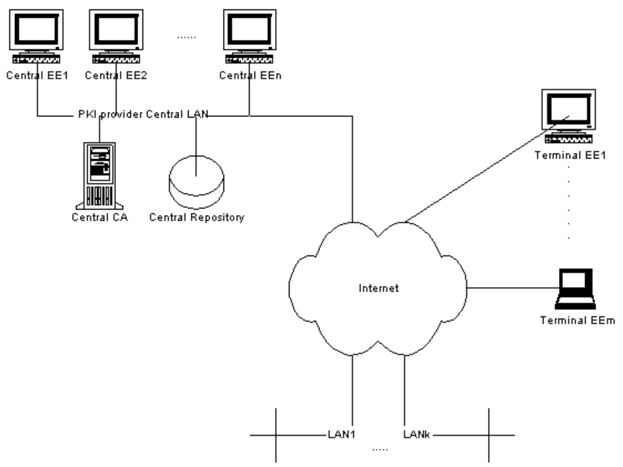

Figure 10.4: Scenario 4: Certificate revocation for the Internet

Here are some suggestions for selecting a certificate revocation solution for this scenario:

1. Such a large system will most likely be a *hybrid, cross-certified* network of CAs, so that ARLs

would be appropriate in any CRL based system.

2. If the system is used for high-critical purposes (e.g. financial), the demands for *high security* and *high timeliness* are critical. In this case, any performance bottleneck is a potential problem, so that a performance optimization in the form of reduced data-structures may be appropriate. CRT is an implemented and tested scheme with good performance and high timeliness. In order to support off-line users (off-line should not necessarily be supported), freshness proofs with some validity time can be issued to piggyback on the certificates.

3. If this system was to be implemented with CRL based schemes, all the three concepts of delta CRLs, over-issuing and CRL distribution points could be combined in order to obtain high performance, timeliness and scalability for CRL based schemes. For instance, the system could deploy over-issued CRL segments, and delta CRLs could be obtained by a fresh revocation information pointer in order to provide some differentiation.

10.7 Certificate Revocation Scenarios Summary

Table 10.1 summarizes the scenarios, their main properties and the proposed revocation solutions.

Scenario	Size	Homogeneous	Centralized	Revocation Solution
1	100 (small)	Yes	Yes	CRL
2	100 (small)	No	No	Freshest Revocation Information delta CRLs
3	10000 (medium)	Mostly	Partitioned	CRL Distribution Point and full CRLs
4	1000000 (large)	No	No	CRT or Over-issued CRL segments with Freshest Revocation Information delta CRLs

Table 10.1: Certificate Revocation Scenarios

Chapter 11

Conclusion

This thesis provides a comprehensive survey and analysis of existing public key revocation schemes. A framework was developed for performing the comparative analysis, and the main findings of the analysis are presented in tables representing all the schemes and their most important properties. Based on the assertion that the system environment is critical to the selection of certificate revocation schemes, a guideline for selecting certificate revocation schemes has been suggested. This guideline is also used in several examples involving different system scenarios.

It is demonstrated that the different schemes address different problems, and none of the schemes have proven to be superior. Rather than finding an optimal scheme, a designer or developer have to evaluate the requirements for the system in question, and select the revocation scheme that is more suitable. It is also shown that combinations of different schemes can be advantageous in order to meet multiple demands in a heterogeneous user environment. Based on the analysis in this thesis, it is expected that the existing certificate revocation schemes are a sufficient foundation for creating revocation solutions for most environments. Better schemes might be invented, but it is in general possible to engineer good solutions based on the existing revocation schemes.

11.1 Future Work

This thesis has taken the approach of a theoretical, comparative analysis of existing public key certificate revocation schemes in order to provide a background for selecting suitable revocation schemes for different scenarios. As a possible extension of this thesis, it is suggested that data is obtained in order to provide a different perspectives on the problems.

The following methods are suggested as options for obtaining this data:

- Gather statistical data from actual PKI systems (preferably in actual business settings) with a significant number of users. These data could prove valuable in demonstrating how the environment variables in section 3.2 impact the evaluation criteria in section 3.3.2 (particularly performance, timeliness and scalability).
- Simulations of the revocation schemes could provide some of the same information as statistical data, except that simplifications and assumptions in the simulation model add an uncertainty to the findings. The results would, however, still be interesting.
- The mathematical models developed by Cooper [18, 19] could be expanded to include more schemes, and a thorough analysis based on these models could demonstrate the properties of the different schemes to a higher degree than in this thesis. In particular, it would be interesting to compare values obtained with the mathematical models with results from the statistical data or simulations described above.

Also, more work can be done in developing and implementing the ideas in chapter 8. In particular, the approaches with *Sloppy CRL Broadcasting/ Multicasting* and *Probabilistic CRL Entries* are interesting modifications to CRL based schemes.

Some of the protocols that provide on-line revocation status checking capabilities are still being standardized. They have not been covered in depth in this thesis, as some of these protocols offer revocation status checking only as one of many services. A survey and comparison of these protocols (OCSP-X, SCVP and DCS) is not available. Also, more work is in designing a good back-end solution for OCSP.

Bibliography

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

[2] Douglas R. Stinson. *Cryptography - Theory and Practice*. CRC Press, 1995.

[3] Charles P. Pfleeger. *Security in Computing*. Prentice Hall PTR, 1997.

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

[5] W. Diffie and M. Hellman. New Directions in Cryptography. *IEEE Transactions of Information Theory*, November 1976.

[6] A. Arsenault and S. Turner. PKIX Roadmap. IETF Internet Draft, October 1999.

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

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

[9] ITU. X.500. ITU-T Recommendation, August 1997.

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

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

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

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

[14] Shimshon Berkovits, Santosh Chokhani, Judith A. Furlong, Jisoo A. Geiter, and Jonathan C. Guild. *Public Key Infrastructure Study: Final Report.* MITRE Corporation, April 1994.

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

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

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

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

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

[20] 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.

[21] M. Branchaud. Caching the Online Certificate Status Protocol. IETF Internet Draft, April 1999.

[22] Phillip Hallam-Baker. OCSP Extensions. IETF Internet Draft, September 1999.

[23] Ambarish Malpani and Paul Hoffman. Simple Certificate Validation Protocol. IETF Internet Draft, August 1999.

[24] Carlisle Adams and Robert Zuccherato. Data Certification Server Protocols. IETF Internet Draft, September 1998.

[25] Silvio Micali. Certificate Revocation System. U.S. Patent number 5666416, September 1997.

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

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

[28] ValiCert Technology Overview. Technical report, ValiCert, 1999.

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

Index

Vita

Name	André Årnes
Place and year of birth	Oslo, Norway, 1976
Education	Queen's University (exchange), 1998–2000 Norwegian University of Science and Technology, 1995–1998
Experience	Summer job, Andersen Consulting Norway, 1999 Summer job, Bull Norway, 1997 and 1998 Student assistant, Department of Engineering Cybernetics, Norwegian University of Science and Technology, January 1997–May 1998 Summer job, Telenor Internet café, Hunderfossen, Norway, 1996