Fast Checking of Individual Certificate Revocation on Small Systems

Selwyn Russell
Information Security Research Centre
School of Data Communications (Information Technology)
Queensland University of Technology
2 George Street, Brisbane 4000, Australia
S.Russell@qut.edu.au *

Abstract

High security network transactions require the checking of the revocation status of public key certificates. On mobile systems this may lead to excessive delays and unacceptable performance. This paper examines small system requirements and options with a view to improving performance. It is shown that the use of keyed hash functions (message authentication codes) with a pre-registration option reduces network latency and allows stateless servers.

1. Introduction

The Internet and its large public audience have stimulated interest in commercial transactions across Wide Area Networks. The World Wide Web and its stateless HyperText Transfer Protocol were devised for access from heterogeneous platforms to public documents and are unsuitable for commercial transactions where confidentiality, access control, authorization, financial transactions, etc are concerned. Security enhancements generally involve the use of digital certificates as discussed in the next section. Mostly a certificate contains information regarding the start and finish of its intended valid life. After a certificate has been created and circulated, it is possible for it to be revoked before its internally specified expiry date. In the absence of a revocation status, a revoked certificate may be mistakenly accepted as valid for a transaction. Verification of the authenticity of a certificate in very important transactions will probably require checking of the revocation status. Revocation notification practices and proposals are considered in section 3. While the processing associated with these status checking techniques is a minor burden on desktop and larger systems, their use of digital signatures can lead to unacceptable delays on small systems such as pocket-sized mobile units because of their limited processing power, limited storage, and relatively low speed network connections.

This paper investigates the use of message authentication codes as alternatives to digital signatures for authenticating revocation status replies from a server by the enquiring client. Section 4 discusses requirements for small systems, and proposes four options for authenticable responses not requiring digital signatures. Two of these options which do not require a prearranged relationship between the two parties are investigated in Section 5. The other two options make use of a prearranged relationship are discussed in Section 6.

2. Background

Recent interest in making transactions across public WANs commercially secure, with assured confidentiality, integrity and authentication, has lead to various proposals for a public key infrastructure (PKI). These proposals commonly involve the concept of a certificate to provide some reliable information about the other party. A "certificate" [10] [5] is usually thought of as a block of data which provides information on the public key of one party and which has been attested to by another party which creates the certificate and is known as the Certification Authority (CA) for that certificate, but the Simple Public Key Infrastructure (SPKI) project of the Internet Engineering Task Force has generalized [7] it to include any required information about the first party, which may not include the public key in some applications. A CA may be a "well known" entity or may be an entity within an organization which provides certificates for others within the organization. A certificate may be self-certified by an individual and accepted by friends and other individuals but would not be accepted in electronic commerce for example.

*This research is part of the co-operative project "Security Technologies in Wireless Communications" between Queensland University of Technology and Korea Telecom.

The life cycle of a certificate begins when an individual approaches a CA and provides certain identification information. In some cases, the individual may provide details of a pre-selected public key; in others, the CA generates a private and public key pair for the applicant. An individual would probably use a CA which provides services to the public. For a member of a small organization, the approach to the public CA is more likely to be made by authorized officers of the organization on behalf of the individual. For a large enterprise the use of a public CA is not cost effective, and the enterprise would more likely have its own internal CA for issuing certificates to its employees or clients.

The top level internal CA would have acquired a certificate from a publicly accepted CA. There may be several levels of internal CAs, particularly in a geographically dispersed enterprise so that verification of the initial identification of the applicant is more certain.

An issued certificate contains an expiry time after which it is to be regarded as unacceptable for further usage. After that expiry time, the certificate should not be accepted by a recipient for performing transactions or other operations, but it may be required after its expiry date for audit, taxation, or other legal purposes. The maximum lifetime might be specified by the applicant but public CAs commonly impose limits such as one year.

If it is decided by the certificate holder or the issuing CA or some other authorized person that the certificate is not to be used anymore before the expiry date, e.g. the person has left the organization or the customer has closed the account, the CA who issued the certificate is approached and notified of the desire for termination. The CA maintains a publicly accessible database of revoked certificates, and newly revoked identities are added in accordance with the CA's policies.

3. Revocation Notification, Practice and Proposals

There are numerous well known methods of notifying the public of identities of certificates which have been revoked before their expiry date. This section covers the better known ones and compares their benefits.

3.1. Certificate Revocation List

Under the X.509 framework[5], a CA will periodically publish a Certificate Revocation List (CRL) containing the identities of all revoked but unexpired certificates issued by that CA. The size of the list is unlimited.

For validation of an incoming certificate by using CRLs, the computer will obtain, either immediately or at some regular or irregular intervals, the most recently published CRL from each of the issuing CAs, of which there are probably at least two, one master CA for the enterprise and the well known public CA who issued a certificate to the enterprise master CA. If there are other internal CAs involved in a certificate chain, they may have to be contacted for their latest CRLs.

For a large computer, a CRL may be stored locally once fetched from the source. Depending on the transaction, a CRL known to be a week old may be considered satisfactory but for a mission-critical financial transactions, anything older than one hour may be rejected. Fetching the latest CRL is not a great burden for a large computer attached to a high speed network but can lead to unacceptable elapsed times for mobile units.

To determine if the certificate in question is in a CRL, the computer needs to

1. determine the class of CRL
2. determine the version
3. extract the individual items of data (issuer name, signature, certificate list, etc)
4. identify the validity period
5. if expired, reject the CRL and take security actions
6. identify the signature
7. identify the signature algorithm used
8. reconstruct the original unsigned CRL
9. calculate the corresponding hash
10. identify the signer
11. obtain the signer's public key
12. from the signer's key and the signature algorithm, calculate the hash signed by the signer
13. if that hash is different from that of the local calculation, reject the CRL and take security actions
14. search through the list of identifiers of revoked certificates, looking for a match with the identifier of the certificate under analysis.

The elapsed time for this process is greatly influenced by the format of the individual elements, the structure of the certificate, and any encoding of the structure.

A CRL is a poor way for a client to obtain information on a single certificate because of the size of the download and the storage and processing required, but is very efficient if many certificates can be checked from the extracted certificate identifiers before the next version of that CRL is available, as is possible if a mainframe in a large enterprise is involved. A benefit to the CA server is that the number

of CRL fetches will be lower than if only information on single certificates were available. The CA will only build the CRL at scheduled intervals and will have a relatively low processing load. However there is still a problem for the irregular enquirer of a single certificate and this is being investigated by various groups.

Although the CRL is current at the time of publication, the CRL will grow stale as further certificates are revoked by the holders before the next publication date. In some scenarios, this is of no consequence, but will be a serious security flaw for others.

Consequently, other means of determining currency of certificate have been investigated including Certificate Revocation Trees (CRTs) [9] and special purpose protocols to access special purpose online databases such as those discussed below.

3.2. Certificate Revocation Tree (CRT)

A Certificate Revocation Tree (CRT)[9] is a patented method which allows a receiver of a certificate to obtain information on its status in the form of fragments of a Merkle tree built and signed by a CA. To reduce the enquiry load on the CA, the tree may be replicated at convenient distributed locations.

The fragments may accompany the certificate or may be obtained from one of the distributed licensed servers ("Confirmation Issuer") which has the replicated complete tree. The fragments relating to the one certificate (actually the fragments define a range of certificate serial numbers) are smaller in size than a typical X.509 CRL with information on all revoked certificates, and can be processed by a sequence of calculations of hash values plus the validation of the signature of the revoking CA.

For the server replying to enquiries on the status of a single certificate, the benefit to the server is that it does not have to sign each reply. It selects the appropriate fragments of the complete CRT which it has obtained from the CA, and forwards them with the signature of the CA on the complete tree. There is no encryption required of the server and its working efficiency is therefore quite high and it can support high hit rates. To check the certificate status, the client must calculate hashes, the number of which is approximately the binary logarithm of the number of leaves in the tree. For a tree of 100,000 leaves containing information on the status of possibly millions of certificates, about 17 hash calculations would be involved. A benefit of the client is that the reply from the server is much smaller than the corresponding CRL. The principle benefit appears to be to the server, which does not have to sign any reply. Network traffic can be reduced by locating mirror servers where convenient.

3.3. Realtime Enquiry of the CA Database

In some cases, it is necessary to verify that the certificate is still active prior to conducting or completing a transaction. CRLs and CRTs cannot provide this realtime assurance and direct enquiry of the certificate database is necessary. The X.500 Directory provides for realtime enquiries using the Directory Access Protocol on OSI networks, as does the derived LDAP system using the Lightweight Directory Access Protocol on TCP/IP networks. There are a number of proposed protocols for authenticated realtime enquiries which are underway as part of the IETF's Public Key Infrastructure using X.509 (PKIX) project.

- Online Certificate Status Protocol (OCSP)

  This proposal [4] is designed around ASN.1 and its related Distinguished Encoding Rules (DER) encoding and uses signed responses.

- Web-based Integrated CA services Protocol – ICAP

  This proposal [13] is designed around the HyperText Transfer Protocol and does not use DER encoding and allows for signed or unsigned responses. OCSP is not designed to be transport protocol dependent but ICAP is designed to use HTTP to ensure it can traverse security firewalls. On the other hand, OCSP points out that HTTP caching can lead to unexpected results if configurations are not carefully set.

- Real Time Certificate Status Protocol – RCSP

  The draft proposal [1] has expired and this proposal seems to have been superceded by OCSP.

- WEB based Certificate Access Protocol – WebCAP/1.0

  This proposal [12] is similar to ICAP in that it uses HTTP but uses XML to encode the contents of the HTTP replies.

3.4. Suitability for use with small systems

Small systems, such as hand held mobile units and palmtop computers, by their nature have limitations on processing power, memory, local storage, speed of network link, which do not apply to enterprise servers or high powered desktop work stations. For these small systems, their performance in processing security functions is critical. If the processing is perceived as unacceptably long by the user, they will be turned off.

In this paper we are concerned with the issue of performing revocation checks from small systems. The issue of processing certificates is beyond the scope of this paper. Certificate structure and processing are being considered by

groups such as the IETF's PKIX and SPKI groups. The former is proceeding with X.509 certificates with extended attributes, ASN.1 specifications, DER encoding and decoding; the latter avoids ASN.1 and DER and favours multiple application-specific certificates rather than one multipurpose certificate. For small systems, the SPKI proposals offer the superior processing speed.

PKI proposals are generally not explicitly platform dependent, but they assume that the client has all the necessary computing power and storage to perform the required operations. Typically, authenticated responses require the client to verify at least one digital signature on the response from a CA or agent, which is relatively time consuming, especially if DER decoding is also involved, as with X.509 certificates and associated systems.

If a certificate chain is involved, then each certificate in the chain will need to be tested for revocation, adding noticeably to the elapsed time for a small system with a low speed network connection. An X.509 CRL is the worst in that it may require the small system with limited local storage to download a 50K CRL, perform DER decoding and analysis and a digital signature verification, for each certificate in the chain, before a transaction can proceed.

4. General Requirements and Options

In this section we will look closer at some of the requirements of clients and servers involved with information on status of certificates.

If a revoked certificate is used by mistake, the fact will be discovered during routine checking and no harm will be done. The certificate holder will probably be notified and will submit a current certificate. The security problem arises when a revoked certificate is used deliberately, implying that the attacker has a compromised key and is impersonating the certificate owner. In this situation, the attacker will try to prevent certificate status checks from revealing that the certificate has been revoked, i.e. will either modify the entries in the database or will actively modify network messages so that the client will receive a good reply rather than the notice of invalid status.

For slow network links, short requests and short responses are desirable. To reduce the elapsed time at the client, short processing time at client and server are desirable. The structure of a request should minimize the work to process it, i.e. not involve processing which could be avoided with a more efficient design. The client requires correct reply, authentication of source, and integrity.

In most cases, the certificate will be valid and the response from the CA will indicate so. This leads to the possibility of a precalculation on the client of a valid response to reduce elapsed time, easy with threaded operating systems. The received response can be compared with the precalculated value of a valid reply. If they match, the transaction can continue. An example where this can be used effectively is the case of a pre-arranged relationship treated below in section 6.1.

If the time for a hash calculation is t, and the most likely response occurs with probability p, two or more hashes will be needed at most only with a probability of $1 - p$. Even on small systems, t is very likely less than the transmission latency.

Because of uncontrollable network latency, the design should minimize the number of message exchanges across the network, preferably using only one request message and one reply message.

4.1. Options

Options for authentication of responses include

- unauthenticated response from CA
- conventionally signed response
- authenticatable but non-signed response

    1. no pre-arranged relationship between client and server, anyone can authenticate the response
    2. no pre-arranged relationship between client and server, only the client can authenticate the response
    3. pre-arranged relationship between client and server based upon a persistent shared value
    4. pre-arranged relationship between client and server using a transient or one-time shared value

We will not discuss insecure responses which cannot be authenticated or responses which have a conventional digital signature, but will seek methods where the elapsed time is much shorter. The remainder of the paper discusses the four classes of authenticatable but non-signed responses.

5. No Relationship Between the Parties

In this section we consider the case where the parties have not made prior arrangements before beginning transactions, e.g. an Internet shopper chances across a shop and begins impulsive purchasing. We will use CA to represent the authority providing replies regarding the current status of a certificate, even though this may not necessarily be the issuing Certification Authority.

5.1. Anyone Can Authenticate the Response

The most desirable situation is where anyone can authenticate the status checking response without having made prior arrangements with the CA but no one other than the CA can construct one. The client can easily share the response as received with others and they can verify the origin independently. In legal investigations, the client does not have to reveal any secret information.

One implementation of this asymmetric situation would be the use of an asymmetric cryptosystem, which is undesirable on a small system. Some proposals, e.g. OCSP, use public key signatures. An advantages is that a stateless protocol can be utilized.

The Guy Fawkes protocol [2] has been proposed for signing bidirectional digital streams. It uses hash functions rather than conventional digital signatures. This protocol requires a signed delivery of initial values but subsequent stages do not need digital signatures, only hash calculations. It could be used for a single enquiry only, and restarted from the beginning on the next enquiry, requiring another digital signature creation and one verification. For multiple enquiries over a period of time, the subsequent digital signatures could be avoided but the state would need to be stored. A disadvantage in either case is that a stateful server is required for the steps in the protocol. Another disadvantage is the number of steps involved, leading to increased elapsed time.

5.2. Only the Client Can Authenticate the Response

This group satisfies the client's request but the response cannot be independently verified by other parties.

One possible implementation is the use of a Diffie-Hellman shared key [6] [11], which requires knowledge of the enquiring client's public key and the published value of the key of the CA (or agent) server. If the client has public value $g^c \ mod \ p$ and the server uses $g^s \ mod \ p$, the shared value, which either can calculate, is $shVal = g^{cs} \ mod \ p$. An efficiency consideration is that the client could calculate the shared key once and store it for reuse. Alternatively, other one pass mutual key values, such as an El Gamal key agreement value [11], could be used. With more network traffic, stateful application programs, and more time, a two pass key agreement [11] could be used.

Assuming we have a $shVal$, we will briefly outline two general ways of using it for authentication.

5.2.1 Symmetric Key Based Authentication

Using a key agreement value from the above calculation, either the value $shVal$ or a derivative, e.g. its hash, could be used as a symmetric key to encrypt a response, or to encrypt a data encrypting key which is then used to encrypt the response. Conventional symmetric methods for message authentication can then be used.

5.2.2 Hash Based Authentication

The value $shVal$ calculated above could be used in the calculation of a hash value, effectively performing a keyed message authentication code. In the following, $TrNum$ is a transaction number used here to thwart replay of reply attacks, and CID is the identifier of the certificate in question. Party A represents the enquiring client and party B represents the CA server.

$$\begin{aligned} A \rightarrow B: &\quad TrNum, CID \\ A \leftarrow B: &\quad TrNum, response, \\ &\quad H(shVal, response, TrNum, CID) \end{aligned}$$

Here there is one hash calculation $H(\ldots)$ each and one calculation each of the shared value $shVal$. Some small systems would be able to calculate the shared value for a most-likely-to-be-accessed-CA once and store it or may have it built into a SIM type device provided with the unit, as with mobile phones.

This principle appears to be suitable for small systems in some environments, mainly when $shVal$ is precalculated and stored securely as in a smart card.

6. Prearranged Relationship Between Client and Server

Many Internet servers require a client to register and establish a user identity and password with it before providing information, even if there is no charge for the subsequent accesses. One reason is to obtain statistics on the usage of the server. A revocation-enquiring client could register with a CA server and obtain a user-specific password (shared value) which would be used in subsequent enquiries. This value could be permanent, i.e. persist unchanged from one access to the next, as is usual with human enquirers, or it could form the basis of a calculated one-time value. These two cases are investigated in the remainder of this section.

6.1. Permanent Shared Value for Each Transaction

If there is a large persistent random value $shRVal$ known to both parties, it can be used in the hash preimage to implement a keyed message authentication code:

$$A \rightarrow B: \quad TrNum, CID$$

$$A \leftarrow B: \quad TrNum, response, H(shRVal, response, TrNum, CID)$$

A nice feature of this is that the client can calculate the response corresponding to the most likely status, viz. $VALID$, while waiting for the server to reply. If the received value corresponds to the precalculated value, then the elapsed time at the client is reduced link, to the comparison time plus the maximum of the network latency and the (pre)calculation time rather than the sum of all three as would be the case if digitally signed responses were used. Calculating a keyed message authentication code will be significantly quicker than verifying a digital signature supplied with a response followed by extracting the status content. Having to wait for a reply from the CA before digital signature processing can begin makes the total elapsed time for the digital signature method even worse. If there is a certificate chain to be worked through, the hash based method is even more desirable.

As mentioned above, an attacker will attempt to send a valid response to hide the fact that a certificate has been revoked. In such a case, if the attacker intercepts the reply and alters the plaintext $response$, the reply will be

$$\begin{aligned} Attacker \leftarrow B: \quad & TrNum, INVALID, \\ & H(shRVal, INVALID, \\ & TrNum, CID) \\ A \leftarrow Attacker: \quad & TrNum, VALID, \\ & H(shRVal, INVALID, \\ & TrNum, CID) \end{aligned}$$

The client will note that the plaintext response contains $VALID$ and that the hash does not correspond to the precalculated hash corresponding to $VALID$. The next step is to calculate the hash corresponding to $INVALID$. If this matches the received hash, the fraud attempt is exposed. If it fails to match, the client will either repeat the request assuming transmission errors or server errors, or abandon the transaction.

If the most likely response, $VALID$, occurs with probability p, two hashes will be needed at most only with a probability of $1 - p$. The average number of hashes per enquiry is $2 - p$, which is close to unity.

One hash calculation is required of the client and the server. The server can be stateless so HTTP can be used. The server needs to know the identity of the enquirer to use the appropriate $shRVal$ and such identities are commonly provided via a cookie in the header of the HTTP request. Alternatively party A, the client, can provide its identity explicitly in the request. If a brute force attack is considered a threat, the actions could be extended to provide a one-time nonce from the client:

$$\begin{aligned} A \rightarrow B: \quad & TrNum, CID, nonce_c \\ A \leftarrow B: \quad & TrNum, response, \\ & H(shRVal, response, TrNum, \\ & CID, nonce_c) \end{aligned}$$

If a chosen plaintext attack is considered a threat, the actions could be extended to provide a one-time nonce from the server but the client will not be able to perform a precalculation while waiting for a reply:

$$\begin{aligned} A \rightarrow B: \quad & TrNum, CID \\ A \leftarrow B: \quad & TrNum, response, nonce_s, \\ & H(shRVal, response, TrNum, \\ & CID, nonce_s) \end{aligned}$$

An alternative is to use a modification of the well known KryptoKnight [3] family of protocols developed at IBM which provide for hash-based authentication and form the basis of some IBM products. Modifying the three steps of the KryptoKnight "secure two-way authentication protocol" by adding a transaction number $TrNum$ and a certificate identification CID in the enquiry, and adding a status reply $response$ from the server, we obtain the following, where $H(shRVal, \ldots)$ represents the Message Authentication Codes in the original:

$$\begin{aligned} A \rightarrow B: \quad & TrNum, CID, A, B, Nonce_A \\ A \leftarrow B: \quad & TrNum, response, B, A, Nonce_B, \\ & H(shRVal, Nonce_A, Nonce_B, B) \\ A \rightarrow B: \quad & A, B, H(shRVal, Nonce_A, Nonce_B) \end{aligned}$$

$shRVal$ is either a shared value or some derived value available to both parties, e.g. $shRVal = H(TrNum, PSV)$ where PSV is the permanent shared value arranged in a prior transaction. In this latter example, each party must calculate three hashes. If $shRVal$ requires no hash calculation, e.g. it is a stored value, there are two hash calculations required for each enquiry. Because of the three steps involved, a server using this KryptoKnight protocol would not be able to function using the simple stateless HTTP but would require a state-aware application running on the server machine.

6.2. Different Value for Each Transaction

The principle is to initially arrange a shared value, then, for each transaction, derive a working value from it which is used as the message authentication key.

Synchronous modification of an initial prearranged value is workable on a small system. An example is a simple extension of the above shared constant value method of section 6.1 is to increment the stored value by a prearranged amount after each enquiry. Because of the avalanche property of the hash function, this will dramatically alter the observed value even if other preimage components were unchanged.

A more complicated option is to use a variation of S-key, almost a reverse S-key. In the S-key system [8] of authenticating a client to a server, there is a value stored at the server for which an authentic client is able to provide the preimage. At successful login, the supplied preimage becomes the stored value to be used for the next login attempt. For our purposes, the client wants to authenticate a communication (the revocation status reply) from the server. An authentic server will know the preimage of the value used on the previous reply, and will be able to use it for the current reply.

Set N as the intial value, each can calculate $F_i(N)$ for any value i, where F is a one way function, and $F_n = F(F_{n-1}(N))$.

$$
\begin{aligned}
A :\rightarrow B: \quad & TrNum, CID, accessNum \\
A :\leftarrow B: \quad & TrNum, response, \\
& H(TrNum, response, CID, accessNum, \\
& F_{N-accessNum}(N))
\end{aligned}
$$

The value $accessNum$ is incremented by the client at each enquiry.

This system places more computation load on the client, which has to calculate $N - accessNum$ hash values at the time of the enquiry, or requires sufficient local storage for a table of precalculated hash values (as is done in some S-key systems).

Another option is a modified Guy Fawkes system, where the server at each stage provides a hash value whose preimage is revealed at a later stage. However this protocol involves numerous stages of networked communications and stateful behaviour of both parties and hence fails to meet our requirements for small systems.

7. Summary

Realtime status verification of certificates is necessary in some situations. If an attempt is made to submit a revoked certificate, it may be a mistake or fraud may be involved. If fraud is involved, authentication is essential. Authentication involving digital signatures can be a performance problem on small systems connected to low speed network links, such as hand held mobile digital units. As an alternative more suitable for small systems, this paper has proposed some uses of hash functions in keyed message authentication codes to provide shorter elapsed times and faster processing with satisfactory security assurance. The use of pre-registration with a constant value used in a hash function allows precalculation during network latency and allows stateless servers.

References

[1] C. Adams, A. Malpani, R. Ankney, and S. Galperin. Internet Public Key Infrastructure Real Time Certificate Status Protocol – RCSP. Technical report, IETF, Mar. 1998. Internet Draft draft-malpani-rcsp-00.txt.

[2] R. Anderson, F. Bergadano, B. Crispo, J.-H. Lee, C. Manifavas, and R. Needham. *A New Family of Authentication Protocols*. Available at http://www.cl.cam.ac.uk/ftp/users/-rja14/fawkes.ps.gz.

[3] R. Bird, I. Gopal, A. Herezberg, P. Janson, S. Kutten, R. Molva, and M. Yung. The KryptoKnight Family of Light-Weight Protocols for Authentication and Key Distribution. Technical report, Dec. 1993.

[4] M. Branchard. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP. Technical report, IETF, Mar. 1998. Draft RFC draft-ietf-pkix-ocsp-08.txt.

[5] CCITT. *The Directory - Authentication Framework*. Number CCITT X.509. International Telegraph and Telephone Consultative Committee, Switzerland, Nov. 1988.

[6] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22(6):644–654, Nov. 1976.

[7] C. Ellison. SPKI requirements. Technical report, IETF, Oct. 1998. IETF draft RFC draft-ietf-spki-cert-req-02.txt.

[8] N. M. Haller. The s/key one-time password system. In *Proceedings of the ISOC Symposium on Network and Distributed System Security*, pages 151–157, San Diego, California, Feb. 1994. The Internet Society. (Available from National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161, USA.

[9] P. C. Kocher. *On Certificate Revocation and Validation*. Springer-Verlag, Anguilla, British West Indies, Feb. 1998. Proc. of the Second International Conference Financial Cryptography, ISBN 3-540-64951-4.

[10] L. M. Kohnfelder. *Towards a Practical Public-key Cryptosystem*. May 1978. MIT B.S. Thesis.

[11] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997. ISBN 0-8493-8523-7.

[12] S. Reddy. WEB based Certificate Access Protocol – WebCAP/1.0. Technical report, IETF, Apr. 1998. IETF draft RFC draft-ietf-pkix-webcap-00.txt.

[13] Y. Sameshima, H. Kikuchi, M. Sakurai, H. Hattori, and H. Kumagai. Web-based Integrated CA services Protocol – ICAP. Technical report, IETF, Jan. 1999. IETF draft RFC draft-sakurai-pkix-icap-01.txt.