Public Key Infrastructures
A Tutorial

Dieter Gollmann
Microsoft Research
Cambridge, United Kingdom
diego@microsoft.com

Preliminary Version

Introduction

Over the last few years, public key infrastructures (PKIs) have became a familiar sight at security exhibitions and received a lot of attention in the security literature, both in academic and commercial circles. To some, 1999 was the year of the PKI and the widespread deployment of PKIs was thought to be imminent.

PKIs are being advertised as the patent solution to many Internet security problems, in particular as a means of creating trust for global e-commerce. This tutorial will not explain what one should do to correctly install a PKI or try to evaluate the PKI *products* currently on offer – the speaker cannot lay claim to any qualification for lecturing on these subjects – but will analyse what lies behind the acronym 'PKI'. We will examine the security problems PKIs claim to address and assess to which extent these problems can be solved at all by technical means. By the end of the tutorial, we hope the audience will have a better understanding for the security services one can expect from a PKI, and equally the limitations of this particular technology.

The tutorial is structured into four parts. Part 1 will lay the foundations for the tutorial, searching for a definition of the term PKI, giving a brief summary of the cryptographic background, and introducing the constituting elements of a PKI. Part 2 compares PKI standards like X.509, PKIX, PKCS, and SPKI. It also refers to legal matters relevant for trust in e-commerce. Part 3 examines policy issues like key generation and storage, naming, certificate validity, and revocation, and also takes a quick look at the use of formal methods in the analysis of a PKI. Part 4 asks whether PKIs are necessary or sufficient to solve security problems in some typical applications areas.

I have no experience with PKI products but have participated in relevant working groups and have collected information from those active in the field. If my attempts to keep the information in this tutorial accurate and recent have failed, I apologize to the reader. In particular, Internet Drafts and the URLs included for further reference are subject to frequent changes.

1 Foundations

1.1 What is a PKI?

To have a meaningful technical discussion on any subject, one first has to define the object of the conversation. In this spirit, we will search for a useful working definition of the term 'public key infrastructure'. We take our cue from the web pages of major PKI vendors and from some PKI users (government bodies piloting PKI projects).

> PKI products consist of software products designed to register, issue, and manage the public and private keys related to digital certificates throughout the life cycle of the certificate [Entrust].

This definition concentrates on the software support for public key cryptography. Compared to other proposals, it is to be commended for its precise focus. However, security is rarely ever achieved by technical means only, so one needs a wider infrastructure to reap the potential benefits of public key cryptography, as expressed by the next definition.

> A Public-Key Infrastructure (PKI) is a comprehensive structure consisting of protocols, services and standards that facilitate the use of public-key cryptography by allowing the secure distribution of public keys between communicating parties, and thereby supporting the exchange of sensitive information over an unsecured network such as the Internet [Secure Solutions Experts].

This is again a consistent definition, adding standards and services as components to the PKI. The next two quotes come from the same source.

> PKI provides the core framework for a wide variety of components, applications, policies and practices to combine and achieve the four principal security functions [confidentiality, integrity, authentication, non-repudiation] for commercial transactions.
>
> The technology to secure the Internet is known as Public Key Infrastructure (PKI) [Baltimore].

This quote makes a bold promise, and implies that public key cryptography is a prerequisite for securing the Internet. The same sentiment is reflected in the next pair of quotes.

> ... sensitive information and communications ... will require a range of security assurances. Fully and properly implemented, a PKI is a system of hardware, software, policies and people that can provide these assurances.
>
> The basis of PKI's security assurances is a sophisticated cryptographic technique known as public key cryptography. [United States General Accounting Office].

Indeed, any comprehensive security infrastructure has to consider the people involved in it and the procedures they are asked to follow. However, there is no logical reason why public key cryptography has to be used to protect sensitive information and communications.

> Federal agencies considering the use of public key technology will benefit by proceeding promptly to participate in building the Federal portion of the evolving worldwide PKI [NIST].

This final quote leads into yet another direction, envisaging a worldwide PKI as an infrastructure for issuing something akin to digital identity cards. The answers to the question 'What is a PKI?' are thus manifold. A PKI may be

- software for managing digital certificates,
- a system of hardware, software, policies and people providing security assurances,
- the technology for securing the Internet,
- a worldwide system of digital ID cards?

None of these is the 'correct' answer. Some may be more well-defined than others but all have some merit. It is important to remember the differences, and whenever a PKI is encountered one has to establish which interpretation is intended at that particular moment before drawing any further conclusions.

1.2 Public Key Cryptography

It is assumed that the reader is already familiar with the basic concepts of public key cryptography. This summary is included as a brief reminder, and it creates an opportunity for pointing out some common misconceptions surrounding public key cryptography. The main technical reference for the cryptographic mechanisms mentioned is [14].

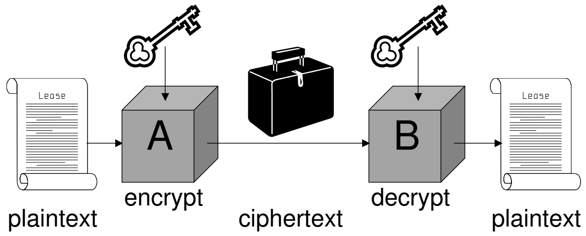

Figure 1: Schematic description of symmetric key cryptography

Figure 1 depicts the fundamental idea of symmetric key cryptography. Encryption protects documents on the way from A to B. A and B share a secret key. A procedure is required that allows A and B to obtain their shared key. For n parties to communicate, about n^2 keys are needed ($n(n-1)/2$ to be precise). Familiar symmetric key encryption algorithms are DES, IDEA, and now AES.

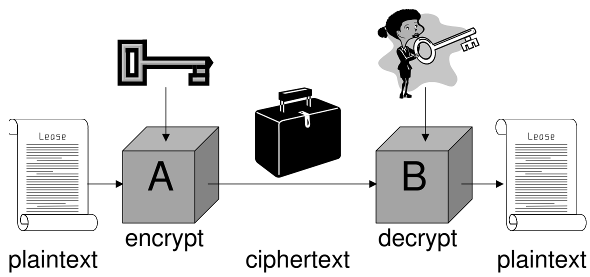

Figure 2: Schematic description of public key cryptography

Figure 2 depicts the fundamental idea of public key cryptography. Encryption protects documents on the way from A to B. B has a public encryption key and a private decryption key. A procedure is required that allows A to get an authentic copy of B's public key. For n parties to communicate, n key pairs are needed. The most familiar public key encryption algorithm is RSA. We assume the reader is familiar with its details.

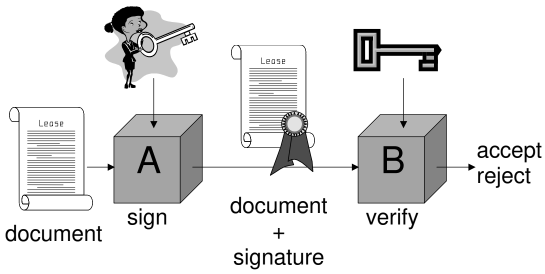

Figure 3: Schematic description of digital key signatures

Digital signatures protect the authenticity of documents. A has a public verification key and a private signature key (Figure 3). A procedure is required for B to get an authentic copy of A's public verification key. Successful verification of a signed document links the document to A's public key and implies that it had not been tampered with. Familiar digital signature algorithms are RSA and DSA.

1.3 Misconceptions about Digital Signatures

A number of misconceptions about digital signatures are prevalent in the cryptographic literature. The first is a technical point. Often the verification of a digital signature is explained as encryption with the public key, plus a comparison of the result with the document being verified. This description can be even found in X.509, which is probably a main source for the propagation of this fallacy.

The claim is not even true for RSA, which uses a public exponent e both for signature verification and for encryption. However, signature verification also requires some redundancy checks on the document. Otherwise, *existential forgeries* are possible where an attacker raises an arbitrary value s to the public exponent e and presents the result $m = s^e \bmod n$ as a document with signature s.

The claim certainly does not hold for DSA. Signature generation in DSA takes as input a private key a, public values α, p, q, and the hash $h(m)$ of the document m. Guidance on the choice of parameters can be found in [14]. To sign m,

- select k at random, $0 < k < q$,
- compute $r = (\alpha^k \bmod p) \bmod q$,
- compute $s = k^{-1}(h(m) + ar) \bmod q$,
- the signature is the pair (r, s).

Signature verification takes as its input the signature (r, s), public values $p, q, \alpha, y = \alpha^a \bmod p$, and the hash $h(m)$, and proceeds as follows:

- verify $0 < r < q$, $0 < s < q$,
- compute $w = s^{-1} \bmod q$,
- compute $u_1 = w \cdot h(m) \bmod q$, $u_2 = r \cdot w \bmod q$,
- compute $v = (\alpha^{u_1} y^{u_2} \bmod p) \bmod q$,
- accept if and only if $v = r$.

Beside this specific counterexample, observe in general that encryption takes as its input a document and a key and produces another document (the ciphertext) as output. In contrast, signature verification takes as its input a document, a verification key, and a signature, and produces a Boolean answer ('yes' or 'no').

The second misconception claims that a digital signature binds the signer (a person) to the document. As we have seen, digital signatures are mathematical evidence linking a document to a public verification key. The link between a public key and a person has to be established by procedural means, and could be recorded in a *certificate* (see Section 1.5). The holder of a private signature key has to protect the key from compromise, relying again on procedural means or on tamper-resistant devices. Finally, there has to be evidence that the signer actually wanted to sign the document the digital signature was computed for.

The third misconception claims that digital signatures are legally binding. Even when digital signatures are recognized by law, there is no guarantee that there is a court with jurisdiction. We return to this issue in Section 4.5.

The fourth misconception assumes that a certificate is needed to verify a digital signature. In fact, an authentic copy of the verification key is needed. The verification key may be stored in a certificate but can also be stored in protected memory. Furthermore, to check a certificate, another verification key is needed. Ultimately, you need a root verification key whose authenticity is not guaranteed by a certificate.

1.4 Electronic signatures

To differentiate between digital signatures as a cryptographic mechanism and signatures as a concept at the application level, a convention is emerging that refers to digital signatures to indicate a cryptographic mechanism for associating documents with verification keys, and *electronic signature* to refer to any security service for associating documents with persons. Electronic signature services usually employ digital signatures as a building block but could be implemented without them.

1.5 Certificates

Public key cryptosystems often rely on a public directory of user names and keys. Kohnfelder implemented the directory as a set of digitally signed data records containing a name and a public key, coining the term certificate for these records already in 1978 [13]. Certificates originally had a single function, binding between names and keys. Today, the term certificate is often used to denote a digitally signed statement [17]. Certificates are used to meet a wider range of objectives:

> Certificate: A signed instrument that empowers the Subject. It contains at least an Issuer and a Subject. It can contain validity conditions, authorization and delegation information [8]

A certificate is simply a signed document binding a *subject* to other information. Subjects can be people, keys, names, etc. Indeed, as the use of certificates changes, our view of what constitutes the subject in a given certificate may change too. We can distinguish between different types of certificates.

- *Identity (ID) certificate*: provide a binding (name, key); sometimes this is the default interpretation;
- *Attribute certificate*: provide a binding (authorization, name);
- *Authorization certificate*: provide a binding (authorization, key); authorization is a new asepct in our security considerations.

1.6 Certification Authorities

Certificates are signed by an *Issuer*. *Certification Authority* (CA) is just another name for Issuer. Sometimes CA is used more narrowly for organizations issuing ID certificates [11]. Section 4 will touch on trust requirements for CAs. We write $\langle\langle X\rangle\rangle Y$ for a certificate for entity X issued by Y. *End entity* (EE) is a name for a PKI entity that does not issue certificates. *Cross certificates* are certificates issued by one CA for another CA to shorten certification paths (Figure 4).

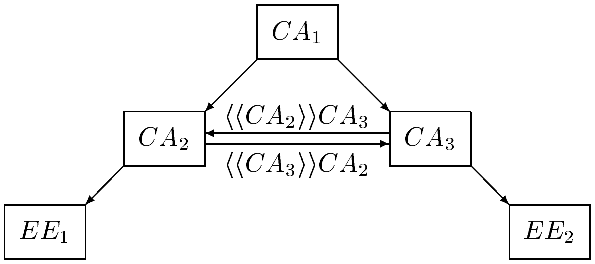

Figure 4: Cross certificates

In general terms, a *Trusted Third Party* (TTP) is an entity providing a security service other parties rely on. A party maintaining a directory of public keys is a typical TTP. A party generating keys for others is a TTP. A CA that issues certificates relied on by others is a TTP. However, it is important to keep in mind that you do not have to be a TTP to issue certificates. Not every CA is a TTP.

1.7 Registration Authorities

Before issuing an identity certificate to a person, the identity of that person has to be checked. This first step could be performed by a *Registration Authority* (RA). The CA would then only manage the certificate life cycle. For example, German post offices are RAs for Signtrust[1]. Note that RA and CA need not be separate entities. The tasks of a RA can be summarized as follows.

- Confirm the person's identity.
- Validate that the person is entitled to have the values requested in the certificate.
- Optionally, verify that the subject has possession of the private key associated with the public key requested for a certificate.

1.8 Time stamping

Applications may need independent evidence about the time a document has been signed. A *Time Stamp Authority* (TSA) is a TTP that provides a "proof-of-existence" for a particular datum at an instant in time. A TSA does not check the documents it certifies. An example of a time stamping protocol is TSP, the Time Stamp Authority [2].

Summary

Security is a multi-faceted challenge.

[1] `http://www.signtrust.de`

- Do you want to send confidential messages?
- Do you want to request access to a service?
- Do you want to create evidence to be prepared for a potential dispute?
- Do you want protection from outsiders?
- Do you want protection from the party you are dealing with?

PKIs can help you address all these issues, but it is not necessarily a single PKI that solves all your problems, and some of these problems can be addressed quite efficiently without a PKI.

2 PKI Standards

This section gives an overview of some of the most important standards for public key infrastructures.

2.1 X.509 – ISO|IEC/ITU-T 9594-8

Today, X.509 version 3 is the most commonly used PKI standard. The original ITU-T Recommendation X.509 [12] was part of the X.500 Directory [6], which has since also been adopted as IS 9594-1. X.500 was intended as a global, distributed database of named entities such as people, computers, printers, etc, i.e. a global, on-line telephone book.

> The information held by the Directory is typically used to facilitate communication between, with, or about objects such as application-entities, people, terminals and distribution lists.

X.509 certificates were intended to bind public keys (originally passwords) to X.500 path names (Distinguished Names) to note who has permission to modify X.500 directory nodes. X.500 was geared towards identity based access control:

> Virtually all security services are dependent upon the identities of communicating parties being reliably known, i.e. authentication.

This view of the world predates applets and many new e-commerce scenarios, where a different kind of access control is more appropriate.

X.500 Distinguished Names identify object entries in the X.500 Directory. A distinguished name is defined as a sequence of *relative distinguished names* (attribute value assertions) representing an object and all its superior entities. For illustration, we mention some typical attributes:

- C ... country
- L ... locality
- O ... organisation
- OU ... organizational unit
- CN ... common name

X.509 uses the following terminology:

- User certificate (public key certificate, certificate): contains the public key of a user, together with some information, rendered unforgeable by encipherment with the secret key of the certification authority which issued it.
- Attribute certificate: contains a set of attributes of a user together with some other information, digitally signed under the private key of the CA.
- Certification authority: an authority trusted by one or more users to create and assign certificates.

Authorization attributes are linked to a cryptographic key via a common name in a user certificate and an attribute certificate (Figure 5).

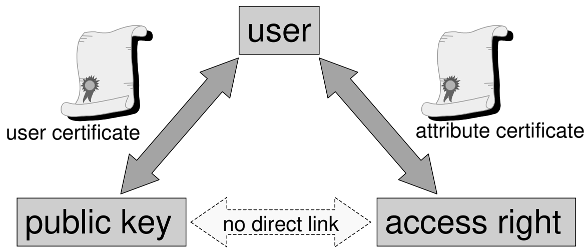

Figure 5: Access control in X.509

Compared to previous versions, the X.509 v3 certificate format (Figure 6) includes extensions to increase flexibility. Extensions can be marked as *critical.* If a critical extension cannot be processed by an implementation, the certificate must be rejected. Non-critical extensions may be ignored. Critical extensions can be used to standardize policy with respect to the use of certificates.

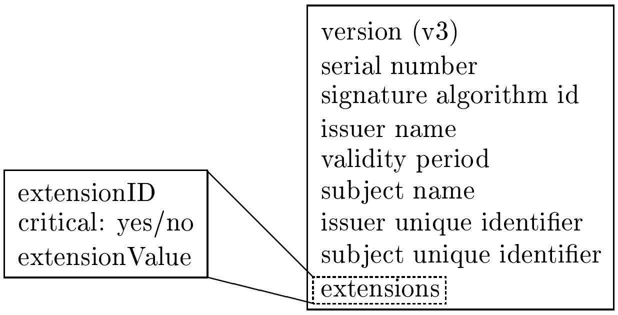

Figure 6: X.509 v3 certificate format

X.509 v3 – Evaluation

X.509 is not a good source to learn about public key cryptography. Many explanations are RSA-centric and concepts like *data recipient entity authentication* appear with out justification and proper definitions. X.509 is geared towards identity certificates. For access control, access rights are assigned to names, not to keys. Only certificate revocation through CRLs is specified. CAs are 'trusted' and have to be 'authorized'. CAs are distinguished from end entities. X.509 no longer assumes that CAs are organized in a global hierarchy.

X.509 has been criticised for using ASN.1 notation, which was deemed to be too resource consuming. Today, XML is often the format of choice for defining data structures so this argument hardly applies anymore. X.509 has been criticised for not being flexible enough, so extensions have been added. X.509 has been criticised for becoming too flexible by including extensions, so that implementations compliant with the standard are not necessarily interoperable. The European Forum for Electronic Business (EEMA) has recently started a pki-Challenge to advance interoperability between PKI products[2].

Distinguish between the X.509 certificate format and the application X.509 was intended for. X.509 v3 is simply a data structure, which has the potential to be used for more than one purpose. The data structure does not tell you how you have to use it.

2.2 PKIX

PKIX is the Internet X.509 Public Key Infrastructure [11]. It adapts X.509 v3 to the Internet by specifying appropriate extensions. The terminology is slightly modified from X.509.

- Public Key Certificate (PKC): 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.
- Attribute Certificate (AC): A data structure containing a set of attributes for an end-entity and some other information, which is digitally signed with the private key of the Attribute Authority which issued it.
- Public Key Infrastructure (PKI): The set of hardware, software, people, policies and procedures needed to create, manage, store, distribute, and revoke PKCs based on public-key cryptography.
- Privilege Management Infrastructure (PMI): A collection of ACs, with their issuing Attribute Authorities, subjects, relying parties, and repositories.

PKC extensions include:

- Authority Key Identifier: for issuers with multiple signing keys
- Subject Key Identifier: typically a hash of the key
- Key Usage: encipher, sign, non-repudiation, ...
- Private Key Usage Period: if different from certificate validity
- Certificate Policies: policy under which certificate was issued and purpose for which it may be used
- Subject Alternative Name: additional identities bound to certificate subject

PKIX authorization follows the X.509 model. Access control requires an AC and a PKC. In X.509, 'authorization conveys privilege from one entity that holds such privilege, to another entity. Users of public key-based systems must be confident that, any time they rely on a public key, the subject that they are communicating with owns the associated private key.' Certificates can provide this assurance if 'owns' means 'is entitled to use'. Certificates cannot provide this assurance if 'owns' means 'is in possession of'.

[2] `http://www.eema.org/pki-challenge/index.asp`

PKIX – Evaluation

PKIX is a 'name centric' PKI. It adapts X.509 v3 for the Internet by standardizing extensions and specifying profiles for certificates, CRLs (certificate revocation lists), and CRL extensions. There is also a profile for the use of X.509 attribute certificates in Internet protocols and there are many protocols for certificate management. For PKIX related documents, see `http://www.ietf.org/ids.by.wg/pkix.html`. Among those, the PKIX roadmap [3] contains a useful discussion of PKI theory.

2.3 SPKI

The Simple Public Key Infrastructure (SPKI) [8] was developed to address some of the perceived shortcomings of X.509. In particular, there were concerns about the global name spaces once assumed in X.509, and it was also felt that for authorization it would be cleaner to assign authorization attributes directly to cryptographic keys rather than taking the detour via a user names. SPKI specifies

> a standard form for digital certificates whose main purpose is authorization rather than authentication.

Trust management takes the same approach, as expressed in this quote from [9]:

> In answering the question "is the key used to sign this request authorized to take this action?", a trust management system should not have to answer the question "whose key is it?".

In this world, *authorization certificates* link keys to authorization attributes. An identity certificate linking a name to a key would still have a role when auditing access decisions, but it is no longer necessary for access control (Figure 7).

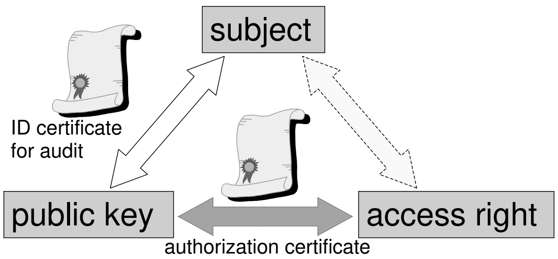

Figure 7: Access control in SPKI

The following definitions are given in [8]:

- Certificate: a signed instrument that empowers the subject; contains at least issuer and subject
- Issuer: signer of the certificate and source of empowerment
- Subject: the thing empowered by a certificate or ACL entry

The SPKI policy is expressed in its *tuple reduction algorithms*. Tuples are abstract notation for certificates or ACL entries. For example, a SDSI 5-tuple has the following fields:

- Issuer: public key (or "Self")
- Subject: public key, name identifying a public key, hash of an object, ...

- Delegation: TRUE or FALSE
- Authorization: access rights
- Validity dates: not-before date and not-after date

A tuple reduction algorithm evaluates ‘certificate chains’. No algorithm for finding relevant tuples is specified, the algorithm processes the tuples presented to it. Authorizations and validity periods can only be reduced. Here is an example of a reduction rule:

```
Input: <Issuer1,Subject1,D1,Auth1,Val1>
       <Issuer2,Subject2,D2,Auth2,Val2>

IF Subject1 = Issuer2 AND D1 = TRUE
   THEN output <Issuer1, Subject2, D2, AIntersect(Auth1,Auth2),
                                       VIntersect(Val1,Val2)>
```

SPKI – Evaluation

The SPKI Certificate Theory [8] is recommended reading on names, access control, etc. SPKI is oriented towards access control and away from global CA hierarchies. There is a separation of concerns. ID certificates are used for accountability. Attribute and authorisation certificates are used for access control. Certificates facilitate distributed storage of Access Control Lists. SPKI standardizes (prescribes?) some policy decisions. For example, only permissions held by the delegator can be delegated. Such a scheme does not support separation of duties very well.

2.4 PKCS

PKCS is set of industry standards published by RSA Inc.[3]. For example, PKCS#10 gives a syntax for certification request. A request contains a distinguished name, a public key, and optionally a set of attributes, collectively signed by the entity requesting certification. Certification requests are sent to a CA, which transforms the request into an X.509 public-key certificate

2.5 Legal Matters

Many countries have enacted legislation aimed at electronic commerce that is also relevant for PKIs and for the role digital signatures can play in securing electronic transactions. We only give a brief and very selective overview.

The European Union has adopted the Directive 1999/93/EC of 13 December 1999 on a Community framework for electronic signatures. Its aim is to facilitate the use of electronic signatures and contribute to their legal recognition. The Directive is not a law itself but has to be implemented in national law by EU states. For our purpose, the annexes of the Directive have to be mentioned.

- Annex I: requirements for qualified certificates
- Annex II: requirements for certification-service-providers issuing qualified certificates
- Annex III: requirements for secure signature devices

In the US, there is relevant legislation both at the federal and at the state level:

[3] `http://www.rsasecurity.com/rsalabs/pkcs/index.html`

- E-SIGN: Electronic Signatures in Global and National Commerce Act [7] (federal law)
- UCITA: Uniform Computer Information Transactions Act: uniform commercial code for software licenses and other computer information transactions [15] (state law passed in Virginia and Maryland)
- UETA: Uniform Electronic Transactions Act [16] (state law passed in a number of states)

2.6 Qualified Certificates

In the parlance of RFC 3039 [18], a *qualified certificate* is a PKC issued to a natural person. The primary purpose is identifying a person with high level of assurance in public non-repudiation services. A qualified certificate contains an unmistakable identity based on a real name or a pseudonym of the subject and exclusively indicates non-repudiation as key usage. The CA has to include a statement about the certificate purpose in the certificate policy. Two ways of constructing a qualified certificate are accepted:

- by a policy in the certificate policies extension
- by a private Qualified Certificates Statements extension

Once again, implementations compliant with RFC 3039 may not be interoperable.

Qualified Certificate in the sense of the EU Directive vouch for *advanced electronic signatures*. Advanced electronic signatures created on secure-signature-creation devices should 'satisfy the legal requirements of a signature in relation to data in electronic form in the same manner as a handwritten signature satisfies those requirements in relation to paper-based data' [ETSI TS 101 862]. This document specifies a qualified certificate profile to help he implementing the EU Directive. The following requirements are stated.

- The issuer field must contain country name.
- There must be an indication that the certificate is issued as a qualified certificate.
- There must be a limit to the value of transactions.
- There must be a limit to the retention period of registration information.
- The certificate contains the name of the signatory or a pseudonym, which shall be identified as such.

RFC 3039 defines a certificate format but cannot imply compliance with the Directive.

2.7 Summary

Standards for certificate formats like X.509 v3 are widely supported. Should there be standards for more than certificate formats? How much should be assumed about the usage of certificates? Different applications have different requirements. The more aspects are considered, the more extensive the documents become (see PKIX). Different (partial) implementations of the same standard are then less likely to be interoperable.

3 Points of Policy

The previous section covered certificate formats. This section turns to the *certificate policies* and operational rules that may apply in a PKI. Background for this discussion is the certificate life cycle shown in Figure 8.

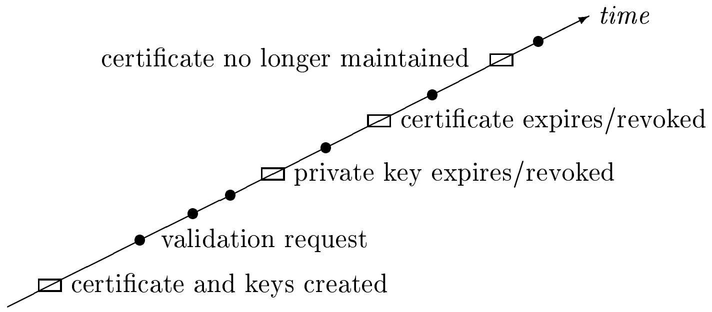

Figure 8: Certificate life cycle

3.1 Key generation and storage

Key generation is an important security aspect. Strong cryptography fails if keys are predictable. Weak pseudo random generators are a frequent cause of failure.

- Keys could be generated by the users (certificate subjects), but do they have the competence to do so?
- Keys could be generated by the CA or some other TTP, but can the CA be trusted with the users' private keys?

As so often, there is no correct answer and it will depend on the actual situation which solution is preferable. When users generate their own keys, *proof of possession* (POP) of the private key may be required when a certificate is being issued. POP is recommended when the certificate is to be used for non-repudiation.

Key storage is equally an important security aspect. If a private signing key is compromised, the link to the certificate subject is broken. If a bogus verification key is inserted, users are misled when verifying certificates. The integrity of verification keys can be protected through:

- cryptographic means: e.g. digital signatures, but root verification keys are not protected by cryptographic means;
- physical and operating system security;
- out-of-band checks: fingerprints of keys published in some other medium, e.g. in a book.

The confidentiality of private keys can be protected by means of physical security or operating system security. Keys can be stored on tokens like smart cards, but you have to consider the consequences of the token being lost or stolen. Keys can be stored on a PC protected by a password. Cryptographic protection is then only as strong as security on the PC, and to a large degree limited by the strength of the password. Long keys protected under a short password could be as insecure as the password.

3.2 Naming

Subjects have to be given names. For the interoperability of PKI products, there needs to be agreement on syntax. Examples from PKIX are

- X.500 distinguished name
- RFC822 e-mail address
- DNS domain name
- URI: uniform resource identifier

For interoperability of systems, there needs to be agreement on semantics. SPKI makes a strong case that globally unique names are relatively useless in open systems like the Internet. Names have only local meaning.

3.3 Validity

Certificates have expiry dates and validity periods. It is a common misconception to think that a certificate cannot be used after it has expired. Deciding what should be done with expired certificates is a policy decision. As an example from the physical world, consider possible entry policies for holders of an EU passport:

- The passport has to be valid n months beyond entry.
- The passport has to be valid until exit.
- The passport has expired less than a year ago (entry into EU countries).

When evaluating a certificate chain we have to consider that certificates expire or may be revoked. The *shell model* takes a conservative approach. All certificates have to be valid at the time of evaluation. In the *chain model* the issuer's certificate has to be valid at the time it issues certificates.

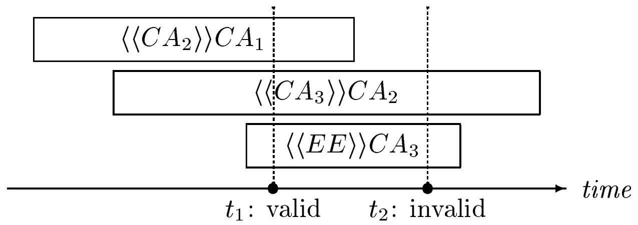

Figure 9: Certificate validity in the shell model

Figure 9 illustrates the shell model. Certificate $\langle\langle EE\rangle\rangle CA_3$ is valid at time t_1 as all three certificates are valid, but is invalid at time t_2 as certificate $\langle\langle CA_2\rangle\rangle CA_1$ has expired. With the shell model, CAs should only issue certificates that expire before their own certificate. If a top level certificate expires or is revoked, all certificates signed by the corresponding private key have to be re-issued under a new key. The shell model is appropriate for certificates defining hierarchical address spaces. It is also the policy implemented in SPKI.

Figure 10 illustrates the chain model. Certificate $\langle\langle EE\rangle\rangle CA_3$ is valid at times t_1 and t_2 as $\langle\langle CA_3\rangle\rangle CA_2$ was valid when $\langle\langle EE\rangle\rangle CA_3$ was issued, and $\langle\langle CA_2\rangle\rangle CA_1$ was valid when $\langle\langle CA_3\rangle\rangle CA_2$ was issued. The chain model requires a time-stamping service (some means of reliably establishing when a certificate was issued). If a top level certificate expires or is revoked, certificates signed by the corresponding

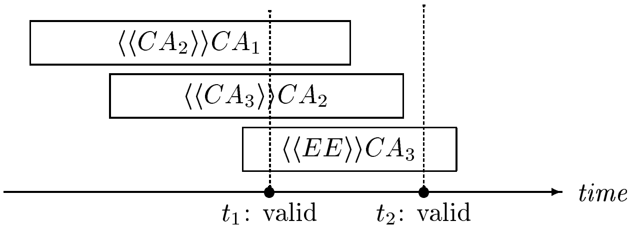

Figure 10: Certificate validity in the chain model

private key remain valid. For example, consider an organisation issuing membership certificates signed by a manager. When the manager leaves and his certificate is revoked, it should not be necessary to re-issue all membership certificates.

Policies beyond the shell and chain model have been proposed, and we can generalize our observations for end entity certificates to other types of signed documents.

Never forget that a certificate cannot tell the end user what the end user's policy is. A certificate can tell the end user what the CA's policy is and it may limit the CA's liability. Be aware that policy decisions have consequences. In the shell model, certificates have to be re-issued when a top level certificate has been revoked. In the chain model certificates should be time-stamped.

3.4 Revocation

Certificates may have to be revoked if a corresponding private key is compromised or if a fact the certificate vouches for no longer is valid. *Certification Revocation Lists* (CRLs) are a solution to this problem and are used in X.509. CRLs are distributed in regular intervals or on demand. A *Delta-CRL* records only the changes since the issue of the last CRL. CRLs make sense if on-line checks are not possible or too expensive.

When on-line checks are feasible, CRLs can be queried on-line. Indeed, when on-line checks are feasible, certificate status can be queried on-line. Examples are the PKIX Online Certificate Status Protocol – OCSP [RFC 2560] and so called *positive lists* of valid certificates in the German signature infrastructure. Today, on-line checks are often the preferred solution.

When a CA issues certificates for its own use, e.g. for access control, only a local CRL is required.

Figure 11 poses a question about revocation and attribute certificates. Bob had acquired an attribute certificate on the basis of an identity certificate IDcert1[Bob,K_B] that is no longer valid. Bob uses this attribute certificate and another identity certificate, IDcert2[Bob,K'_B], when requesting access to the resource *Doc*. Should the request be granted? Should the attribute certificate be revoked?

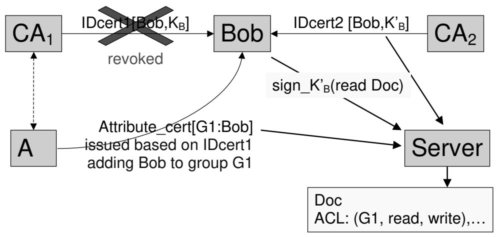

Figure 11: Revocation – a scenario

The answers depend on your policy. Is the attribute certificate linked to Bob's name or to an identity certificate? Does the server differentiate between CA_1 and CA_2? Would your policy change if IDcert1 and IDcert2 had been issued by the same CA? You cannot look at Figure 11 to learn your policy. You can look at it to learn about issues your policy may have to address.

3.5 Formal methods and PKIs

Formal methods can help to analyse the effects of complex policy rules. We will use SDSI [17] as an example. SDSI is a PKI for access control, using groups (and access control lists) to express security policies. We write a SDSI certificate as $k : n \to p$, denoting that principal p is added to group n in the local name space of k, or that n is a name for principal p in the local name space of k. (We will sometimes omit the identifier of the local name space in our examples.) We write k's n for the name n in the local name space of k. SDSI specifies an algorithm for deciding whether a principal is a member of a given group. We will give brief overviews of two formal analyses of SDSI. In the first case, a logic of access control was tailored to SDSI [1]. The second case is a 'precise' logic (LLNS) [10] for SDSI.

In the access control logic, we informally equate names with access rights. A group certificate $n \to p$ gives p the access rights of group n. A member has more rights than the group. Again informally, we equate the interpretations $[[n]]$ and $[[p]]$ in the formal semantics with the set of rights given to n and p. For this logic, we have a theorem matching our intuition.

Theorem 1 *We can derive $n \to p$ in the logic if and only if* $[[n]] \subset [[p]]$.

It is furthermore the case that all results of the SDSI group membership algorithm can be derived in the logic, but the converse does not hold.

To address this apparent deficiency, LLNS was intended as a 'precise' logic for SDSI. LLNS is silent on the purpose of certificates. A group certificate $n \to p$ is simply read as 'p is member of n'. For our purpose, we equate groups with email lists. The group certificate $n \to p$ adds p to the list n. The group contains more entries than a member. Informally, we equate the interpretations $[[n]]$ and $[[p]]$ in the formal semantics with the set of addresses in n and p. Also for this logic, we have a theorem matching our intuition.

Theorem 2 *We can derive $n \to p$ in the logic if and only if* $[[n]] \supset [[p]]$.

It is the case that LLNS captures SDSI group membership exactly. We are now in the interesting situation of having two logics that make contradicting observations about the world. This raises a few questions.

- Is LLNS the more reasonable formalisation of SDSI as it models SDSI precisely?
- Is LLNS a logic for access control?
- Are the SDSI rules compatible with the SDSI goals?

To see which logic is 'right', we tell a short story.

> MIT is an authority for *Ron*, a cryptographer who changes keys weekly to reduce key exposure. In week 1, MIT issues MIT: *Ron* $\to k_1$ and *Ron* issues k_1: *Butler* $\to$ *Lampson*. In week 2, MIT issues MIT: *Ron* $\to k_2$ and *Ron* issues k_2: *Lampson* $\to k_3$. Re-issuing *Ron*'s first certificate with k_2 would defeat the purpose of changing keys. Later, *Ron* receives a request signed by key k_3 for a resource *Butler* is allowed to access. Should the request be granted?

The conclusion MIT's *Ron's Butler* $\to k_3$ that would grant access follows in the access control logic but not from SDSI group membership, and it cannot be derived in LLNS. In SDSI, a local name can be bound to more than one key. The wording in [17] suggests that the above conclusion would be acceptable.

The conclusions we must reach is that SDSI group membership does not define a complete access control scheme, and that one has to take care when interpreting the results of formal analysis.

3.6 Summary

The rules your PKI implements are given by your policy, not by a law of nature. Most PKI standards make assumptions about the world and standardize policy to some extent. Complex relationships between PKI entities may imply complex evaluation and revocation rules. Formal methods can help in the analysis of complex systems, but you have to make sure that the formal analysis captures your intentions.

4 PKI as a Solution

Having dealt with the technical aspects internal to a PKI, we finally turn to applications that seem to be promising areas for PKI deployment. We will check whether PKIs are necessary or sufficient to deal with the security problems encountered.

4.1 Secure email

Messages arrive from arbitrary sources. There is usually some indication of the name of the sender, but you may not know that person or organisation. Authenticating the sender requires an authentic copy of the verification key, e.g. a certificate. Certificates bind a name and/or email address to an encryption and/or verification key. Three classes of certificates are common today:

- Class-1: online enrolment, no identity checks; some CAs therefore do not include a name in the certificate;
- Class-2: CA checks identity and address against third party information; checks that the name is 'real' but not that it is really the applicant;
- Class-3: applicant has to appear in person at CA.

Email is signed to authenticate the sender. Signed email is authenticated on receipt, either when it arrives in the mailbox or when it is first opened. Signed email is rarely a contract that has to be archived or re-authenticated at a later date or be presented as evidence in a dispute. (Signed) email may be evidence held against you.

Messages are being sent to known addresses. End-to-end encryption of email requires an authentic copy of the recipient's encryption key. Alternatively, email may be encrypted between the sender (receiver) and a mail server.

Consider now the special case of secure company email. Staff has external access to the company mail server. To secure access to the mail server each user gets a private key/public key pair and a certificate for their public keys, which is presented on access to the server. A PKI is deployed to support this application.

The company could issue the certificates, so no TTP is needed. The public keys could be stored in a protected list at the mail server, so no certificates are needed. Users could be issued with secret keys (one key per user), so no public key cryptography is needed. The company can secure access to its mail server without deploying a PKI.

The secret key solution and the public key solutions both use one key per user. For the stated security goal, compromise of secret keys and private keys are equivalent. The PKI solution has no advantage compared to a conventional secret key solution. This situation may change when the requirements change, e.g.

- staff should get access to company resources isolated from a central server (revocation will have to be addressed),
- support for collaboration between companies,
- end-to-end secure email,
- support of non-repudiation.

4.2 Non-repudiation

Handwritten signatures on paper documents signify the intent of the signer and thus bind the signer to the document. In the electronic world, non-repudiation protocols should create evidence that binds a (legal) person to a document.

As discussed in Section 1.3, digital signatures bind a document to a public verification key. A certificate can bind the public verification key to a name. Operational procedures at the CA check the correspondence between the name and a person. Further operational procedures check that the person holds the private key matching the public key in the certificate. Computer security protects the private key from compromise and misuse. With this level of indirection, it becomes doubtful whether digital signatures can really prove intent [19].

Proper non-repudiation requires a complex infrastructure. The concept of a PKI is relevant when seen as 'a system of hardware, software, policies and people'. CAs are part of this system. This is generally appreciated and policies for CAs are receiving considerable attention. End users are also part of this system. Can they fulfil their security obligations? Can they plead ignorance or incompetence if they want to renege on a commitment? The answers will again depend on the type of end user envisaged. Companies participating in a PKI may well be expected to meet higher standards of professionalism than home users. On the technical level, end user devices are part of the system. Securing end user devices is a difficult task and may prove impossible when the devices become too complex.

It is important to keep in mind that signing email is inherently different from signing a contract. Email is signed to authenticate the source, not to indicate intent. Automatic signing of email for authentication is reasonable, automatic contract signing is not. Contracts are signed to indicate intent (non-repudiation). In this case, the certificate and the key have to be valid at time contract is signed. Hence, a time stamping service is required. A contract will be verified at some time in the future. Thus, contracts have to be archived. Certificates (verification keys) have to be archived. Software for displaying the contract has to be archived.

4.3 Access control

X.509-style PKIs support identity-based access control. Identity-based access control is a legacy from previous eras when *closed systems* were dominant. In the Internet age, we are dealing frequently with *open systems* requiring different models of access control. Moreover, identity-based access control raises privacy and data protection issues. Still, organisations are used to identity-based access control, maybe augmented by Role Based Access Control (RBAC), so there is a demand for this type of systems.

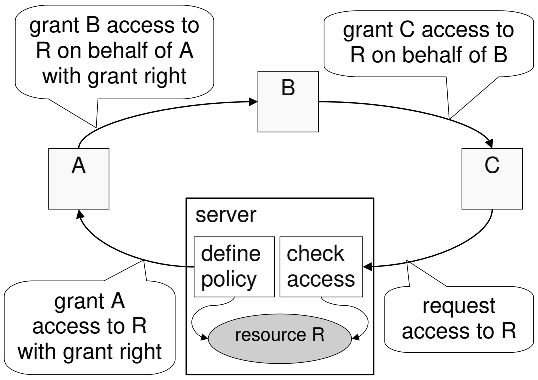

Figure 12: Certificate loop in access control

For access control on the Internet, the world is still searching for appropriate security policies that are not based on user identities. We have the Java sandbox model and code signing. We have a choice between authorization certificates assigning rights to keys and attribute certificates assigning rights to names. We may consider 'anonymous' access to services paid for in e-cash. Trust management systems such as KeyNote [5] 'delegate' access decisions to third parties.

In access control, certificates form a loop starting from the party that makes the final access decision (Figure 12). Identity certificates can be used for auditing purposes to augment access control. CAs are part of your local PKI and should adhere to your security policy. For access control within an organisation, you can be your own CA, and would not even need certificates.

The previous applications, like secure email, are concerned with communications security. The theoretical investigations described in Section 3.5 could imply that at a fundamental level communications security and access control are following incompatible rules. They could be read as a warning sign of potential pitfalls when using a single PKI to support both access control and communications security. In particular, it is potentially much more difficult to federate the access control systems of several organisations than to federate their secure email systems.

4.4 Digital identity cards

Specific certificates such as the qualified certificates of the EU Directive are intended to vouch for the identity of a person conducting an electronic transaction, thus taking the role of identity cards on the Internet. Such 'digital identity cards' can be useful for e-government, e.g. to protect access to personal data (health information) or to control access to services and benefits. By their very nature, government services are mainly offered locally to residents of a country.

Digital identity cards do not seem to be directly useful for e-commerce. It is often unnecessary to know the precise identity of an entity, relying on identities (of persons) may create privacy problems, and the necessary infrastructure may be too expensive to support.

When CAs act as TTPs issuing identity (qualified) certificates, the parties relying on these certificates may want to have some guarantees that the CA performs its job properly. A regulatory framework may therefore state the liability of a CA or even provide for the accreditation of CAs. Accreditation is not required in general by the EU Directive. In specific cases a CA may be governed by rules pertinent to the application. This point is raised, for example, in a discussion paper recently published by the UK Financial Services Authority [4].

4.5 Legal assurance

Assume that electronic (digital) signatures are accepted as evidence in any court of law. Assume properly regulated CAs issue qualified certificates. Assume there is a dispute. Which court has jurisdiction? Depending on the circumstances, there may be no court, there may be exactly one court (it helps when all parties reside in the same country), or there may be several courts.

In our legal systems location is an important factor, but e-commerce and the Internet are notorious for 'ignoring' national boundaries. Where does an electronic transaction take place? To adjudicate on electronic transactions, the problem is less the recognition of electronic signatures but determining a court with jurisdiction.

Consider the current situation in the United States regarding the electronic delivery of goods. UCITA gives the *licensor* (provider) the right to determine where to litigate a dispute. However, Iowa has passed UETA with an explicit provision that for consumers in Iowa, the case is heard in Iowa. Legal uncertainty is not caused by doubts about the validity of digital signatures but by conflicting laws.

This example illustrates a general dilemma faced by those having to regulate e-commerce. Is it more important to protect the interests of consumers or the interest of (small) companies?

4.6 Trust

Lack of trust supposedly inhibits e-commerce, and PKIs are being marketed as the means of creating trust. Unfortunately, trust is a word with too many meanings. Trust can refer to a quality of a personal relationship, or to a lack of fear from the unknown, or to expectation of behaviour. In a commercial setting, trust can be related to the fact that there is recourse to arbitration or to the legal system. In access control (trust management), trust stands for the right to access a service or resource. In many security discussions these aspects get hopelessly mixed up.

Can a PKI create trust? Would you trust someone because he has a Verisign[4] certificate (or an ID card)? A certificate from a trusted CA can certainly reduce uncertainty. Trust in a CA may mean that it is expected to follow proper procedures. Trust in a CA may mean that there is legal recourse in case of damages. If the CA has authority over certificate subjects, uncertainty may be reduced even more.

Trust management is a fashionable term for certain types of access control systems, and a PKI can be part of such an access control system. Access rights may be based on trust but may also be assigned for other reasons, e.g. on the need to do a job. It is not wise to pretend that all authorizations are expressions of trust.

In general, PKIs will *support* existing trust relationships, whatever they may be, more than *creating* trust relationships on their own.

4.7 The PKI Paradox

PKIs promise to put trust into *global* e-commerce so that you can securely interact with parties you have never met before from all places in the world. Paradoxically, PKIs work best *locally*. Concerning trust between people, you know most about the people and institutions closest to you. Concerning legal issues, the situation is clearest when all parties, CA included, are under the same jurisdiction. For access control, you can be your own CA.

Commerce uses chains of local 'trust' relations, it does not need a global dispenser of trust. For example, Identrus[5] is setting up a PKI for B2B that is built upon local relationships (Figure 13).

[4] `http://www.verisign.com`
[5] `http://www.identrus.com`

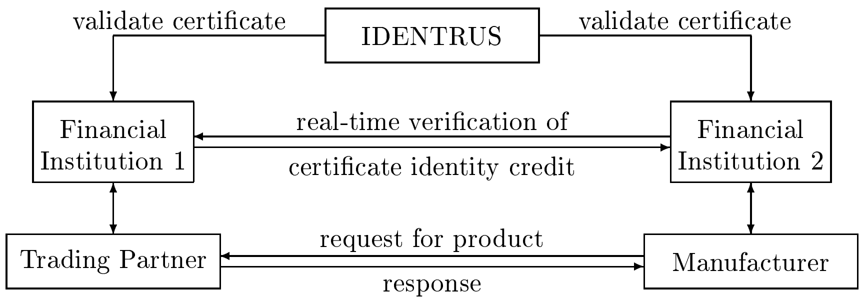

Figure 13: The Identrus model

Summary

There are several options for deploying a PKI. There are *local PKIs*, e.g. when using public key cryptography for access control. In this case, all that is required is a protected local list of user names and verification keys. Neither certificates nor a third party are needed.

The credit card system is an example for a *private PKI*. Customer and merchant have a pre-established relationship with the card issuer, who is the CA issuing certificates. The CA is party to the transactions, not an independent TTP.

Qualified certificates are intended for use in a *public PKI*. Certificates are issued by CAs to certificate holders. The certificate verifiers have no pre-established relationships with the CA.

The PKI sector is facing a number of challenges. At the technical level, the interoperability of PKI products has to be addressed. In this area, the EEMA pki-Challenge has already been mentioned. At the application level, e-government, secure email, single log-on for access control, and non-repudiation for e-commerce are the main areas of promise for PKIs. In commercial settings, the business case for a PKI has to be examined. Does your application need a PKI? There are numerous cases of technically successful PKI pilots that did not lead to a full PKI deployment. At Information Security Solutions Europe 2001 a rumour went round that PKIs are dead.

As concluding advice, analyse your security problems first, then look for a solution that fits your situation. Do not buy a 'solution' and then try to fit your organisation around it. PKIs have many applications. The application tells you how to use a PKI, if a PKI is useful at all. The PKI should not tell you what you ought to do.

Without a killer application, the PKI will become the application killer.

Acronyms

AA	Attribute Authority [11]
AC	Attribute Certificate [11]
CA	Certification Authority
CRL	Certificate Revocation List
EE	End Entity
OCSP	PKIX Online Certificate Status Protocol
PKC	Public Key Certificate [11]
PKCS	Public-Key Cryptography Standards (RSA Security Inc.)
PKI	Public Key Infrastructure
PKIX	Internet X.509 Public Key Infrastructure [11]
PMI	Privilege Management Infrastructure [11]
RA	Registration Authority
SDSI	Simple Distributed Security Infrastructure [17]
SPKI	Simple Public Key Infrastructure [8]
TSA	Time Stamp Authority
TSP	Time Stamp Authority
TTP	Trusted Third Party

Certificate Types

- Attribute certificate (X.509, PKIX)
- Authorization certificate (SPKI)
- Identity certificate (frequently mentioned, e.g. in SPKI, probably equivalent to user certificate)
- Public key certificate (PKIX)
- Qualified certificate (EU Directive, RFC 3039)
- User certificate (X.509) (also public key certificate, certificate)

References

[1] Martín Abadi. On SDSI's linked local name spaces. *Journal of Computer Security*, 6:3–21, 1998.

[2] C. Adams, P. Cain, D. Pinkas, and R. Zuccherato. Internet X.509 Public Key Infrastructure Time Stamp Protocols, October 2000. Internet Draft draft-ietf-pkix-time-stamp-11.txt.

[3] Alfred W. Aresenault and Sean Turner. Internet X.509 Public Key Infrastructure – Roadmap, November 2000. Internet Draft draft-ietf-pkix-roadmap-06.txt.

[4] Financial Serives Authority. The FSA's approach to the regulation of e-commerce, June 2001. `http://www.fsa.gov.uk/pubs/discussion/dp6.pdf`.

[5] Matt Blaze, Joan Feigenbaum, John Ioannidis, and Angelos D. Keromytis. The KeyNote Trust-Management System Version 2, September 1999. RFC 2704.

[6] CCITT. *The Directory – - OVERVIEW OF CONCEPTS, MODELS AND SERVICES*, 1988. CCITT Rec X.500.

[7] US Congress. *Electronic Signatures in Global and National Commerce Act*, 2000. Pub. L No. 106-229, 114 Stat. 464 (2000) (codified as 15 U.S.C. 7001-7006, 7021, 7031) (enacted S. 761).

[8] Carl M. Ellison, Bill Frantz, Butler Lampson, Ron Rivest, Brian M. Thomas, and Tatu Ylonen. *SPKI Certificate Theory*, November 1998. Internet Draft.

[9] Joan Feigenbaum. Overview of the AT&T Labs trust-management project. In *Security Protocols, LNCS 1550*, pages 45–50. Springer Verlag, 1998.

[10] J. Y. Halpern and R. van der Meyden. A logic for SDSI linked local name spaces. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop*, pages 111–122, 1999.

[11] Russell Housley, Warwick Ford, Tim Polk, and David Solo. Internet X.509 Public Key Infrastructure – Certificate and CRL Profile, January 1999. RFC 2459.

[12] International Organization for Standardization. *Information technology – Open Systems Interconnection – The Directory-Authentication Framework*. Genève, Switzerland, June 1997. ISO/IEC 9594-8—ITU-T Rec X.509 (1997 E).

[13] Loren M. Kohnfelder. Towards a practical public-key cryptosystem, May 1978. MIT S.B. Thesis.

[14] A.J. Menezes, P.C. van Oorschot, and Vanstone S.A. *Handbook of Applied Cryptography*. CRC Press, Boca Raton, FA, 1997.

[15] National Conference of Commissioners of Uniform State Laws. *Uniform Computer Information Transactions Act*, 1999. see also `www.ucitaonline.com`.

[16] National Conference of Commissioners of Uniform State Laws. *Uniform Electronic Transactions Act*, 1999. see also `www.uetaonline.com`.

[17] Ron Rivest and Butler Lampson. SDSI – a Simple Distributed Security Infrastructure. Technical report, 1996. `http://theory.lcs.mit.edu/~cis/sdsi.html`.

[18] Stefan Santesson, Tim Polk, Petra Barzin, and Magnus Nystrom. Internet X.509 Public Key Infrastructure – Qualified Certificates Profile, January 2001. RFC 3039.

[19] Jane K. Winn. The emperor's new clothes: The shocking truth about digital signatures and Internet commerce, 2001. Revised Draft, `faculty.smu.edu/jwinn/shocking-truth.htm`.