Reasoning About Public-Key Certification: On Bindings Between Entities and Public Keys*

Reto Kohlas Ueli Maurer

Department of Computer Science
Swiss Federal Institute of Technology (ETH)
CH-8092 Zürich, Switzerland
{kohlas,maurer}@inf.ethz.ch

Abstract. Public-key certification is of crucial importance for advancing the global information infrastructure, yet it suffers from certain ambiguities and lack of understanding and precision. This paper suggests a few steps towards basing public-key certification and public-key infrastructures on firmer theoretical grounds. In particular, we investigate the notion of binding a public to an entity.

We propose a calculus for deriving conclusions from a given entity Alice's (for instance a judge's) view consisting of evidence and inference rules valid in Alice's world. The evidence consists of statements made by public keys (e.g., certificates, authorizations, or recommendations), statements made physically towards Alice by other entities, and trust assumptions. Conclusions are about who says a statement, who owns or is committed to a public key, and who transfers a right or authorization to another entity, and are derived by applying the inference rules.

1 Introduction

1.1 Motivation

In recent years, cryptography has become a key technology for digitizing business processes, and this development is going to continue both with increased intensity and with global impact. While technical aspects of cryptographic mechanisms (security and efficiency of cryptographic algorithms, standardization of APIs and platforms, etc.) have received substantial attention in the past years, a precise understanding of their role in business processes is still in development.

A key aspect that is not precisely understood is the notion of *binding* a public key to an entity in order to achieve authenticity and/or non-repudiation. This paper's contribution is to propose some initial steps in the direction of a formal understanding of the role of such bindings and of digital signatures in a global public-key infrastructure (PKI).

* In *IEEE Journal on Selected Areas in Communications, vol. 18, no. 4, Apr, 2000.*

1.2 Previous Work

There exists a large body of literature on formal methods related to information security (e.g. [8, 3, 18, 30, 15, 9, 7, 29]). Much of this research is aimed at analyzing and designing protocols or security mechanisms, most importantly bilateral authentication protocols. In contrast, the goal of the formal approach taken in this paper is to illustrate a number of important points in the context of public-key certification that have previously not been made precise. Our calculus models desirable conclusions when using digital signatures and public-key certificates as evidence and hence contributes to a more precise understanding of what one tries to achieve with a PKI.

Previous Work on PKIs. From a high-level point of view, a possible classification of problem areas and research contributions related to PKIs is as follows:

- *Certificate format and content.* It has been recognized that PGP- and X.509-certificates are not well-suited for electronic commerce, and several research efforts have been made to define new certificate formats and semantics [12, 13, 19, 16, 11].
- *Revocation.* One particular area of interest is to design efficient mechanisms for withdrawing public keys or certificates that have become invalid for a certain reason [26, 14, 23, 17, 29, 24, 22].
- *Name spaces.* It is non-trivial to establish and implement a global, context- and application-independent naming scheme as in the X.500 recommendation. To avoid this problem, SDSI proposes the linking of local name spaces [27, 2].
- *Trust models and uncertainty management.* Most evidence in real life is to some degree uncertain. It can therefore be desirable to model uncertainty by assigning confidence values to evidence and conclusions, in particular regarding the trustworthiness of entities.[1] Similarly, the reliability of the identification underlying certification can depend on the procedure that is used. The term trust model is often used to refer to models and techniques for dealing with uncertainty in these contexts [25, 32, 5, 4, 10, 31, 20, 21, 6].
- *Information systems aspects.* A global PKI will be a complex distributed information system whose design and implementation are a research area of independent interest.

Notions of Bindings Between Entities and Public Keys. This paper's contributions should be seen as an extension of the work in the first of the above research areas.

In the literature on public-key certification, "binding" is a term that is often used to indicate that a public key represents an entity. A digitally signed statement that asserts a binding between an entity and a public key is sometimes

[1] In PGP for instance, trustworthiness can be estimated by five different values: (unknown, no trust, marginally trusted, fully trusted, ultimate trust)

called an "identity-based certificate" [19], but the exact nature of the binding is often unclear. We summarize different notions of bindings in the literature.

In X.509 [1], a public key is bound to the entity if the entity "possesses" (or controls or owns in our terminology) the public key. It is expected explicitly that the key owner keeps the secret key confidential. It is unclear whether a X.509 certificate should be interpreted as an assertion by the certificate issuer that the entity controls the contained public key or simply claimed that it controls the key. It is further unclear to what extent the key owner was instructed by the certification authority to assure that the secret key remains confidential and to what extent he or she will be liable for messages (or statements according to our terminology) signed by the secret key.

"A PGP public-key certificate is, in essence, a public key together with a user ID testifying that the user ID associated with this public key is valid" [28]. It is not specified in the documentation when a public key is valid for an entity, and the notion of binding is left unspecified. Common practice of PGP users is to issue a certificate when another user claims ownership of a key.[2]

SDSI [27, 19] explicitly assumes that every key is controlled by at most one entity; hence the binding of SDSI certificates is similar to that of X.509 certificates. SPKI certificates are sometimes called *authorization-based* certificates [13, 19]. SPKI also explicitly assumes exclusive ownership of the keys. A prominent example in SPKI is that an issuer (i.e., a public key) delegates an authorization to another key. The entity controlling the key can obtain the authorization by digitally signing the requests. SPKI naming is based on SDSI.

Lampson, Abadi, Burrows, and Wobber [18] model, among other things, the role of public-key certificates and certification authorities (possibly organized hierarchically) in the context of access control. Because the focus is on access control, where the access control mechanism is trusted by default, trust is in this paper not a resource explicitly expressed. The only type of binding between public keys and an entities is the "speaks for" relation, meaning that if a statement s is digitally signed with a key speaking for the entity, then the entity says s. It is not considered that different entities might have different views; in particular, certification authorities are always universally trusted. Thus, all entities will draw the same conclusions.

1.3 Contributions of this Paper

We illustrate a number of points that have previously not been made explicit:

- Because sole ownership can inherently not be verified (except in a context where a physical device internally generates and uniquely controls a secret key), one should not rely on such a claim or assumption. While in certain contexts exclusive ownership is not needed and a claim of ownership suffices, in other contexts the liability of a claimed key owner must be made explicit,

[2] In general, the user presents a hash value (the "fingerprint") of the public key to the introducer.

making it the owner's interest to be and remain the sole owner, i.e. to protect the secret key.

For instance, in the case of encryption and signature keys, an entity (say Alice) can be trusted by default to be the exclusive owner of a particular key. Alice has no interest in claiming false ownership, since otherwise another entity could sign in Alice's name or decrypt messages sent to Alice, which, if it were desirable for Alice for some reason, Alice could also achieve by performing herself the operations requiring the secret key on behalf of another entity.

In contrast, in a legal context a claim by Alice to be the exclusive key owner of a signature key is not sufficient evidence; the key could have been intended for other applications (e.g. PGP), and it would be inappropriate to silently let the entity be legally bound to statements signed by the key. Rather, Alice must declare explicitly her intention to use the key in a legal context. We call such a statement to accept the legal consequences of using a signature key a *commitment* of an entity to a key. A commitment to a key can be seen as a special kind of ownership claim.

- The main role of a certification authority is to make statements, through its own public key, about statements (claim of ownership or commitment) made by other entities.
- The role of self-certificates. If a right or privilege is transferred to a key owner of the signature public key, then Alice's mere claim of ownership of the key or the commitment of Alice to the key cannot be used to conclude that Alice is authorized to receive the right. Rather, Alice must prove ownership of the key, for instance by digitally signing the statement that Alice is a owner of the key with the same key. This is a self-certificate. On the other hand, in contrast to the recommendations made in PGP (e.g. see [28] (p. 97)), self-certificates do not contribute to drawing conclusions when exchanging signature or encryption keys.

1.4 Outline

In Section 2 we informally introduce the basic concepts of the calculus, and in Section 3 we state basic definitions. The inference rules are discussed in Section 4. The purpose of Section 5 is to illustrate the use of the calculus in a few typical scenarios. Some open problems and directions for future research are mentioned in Section 6.

2 Concepts

2.1 Principals: Entities and Public Keys

The two types of principals[3] in our model are *public-key pairs* and *entities*. A public-key pair consists of two parts. The public part is used to verify signatures or to encrypt data. The private part allows to generate digital signatures or to decrypt data; it must therefore be protected from unauthorized access. For simplicity, we will refer to public-key pairs by "public keys" or "keys".

Entities are human beings or legal entities. An entity *controls* (or *owns*) the public key if it can use the public key to decrypt or sign data. In practice, a public key is controlled through knowledge (e.g. of the secret key or a pass phrase protecting the secret key), or through possession of a physical token (e.g. a smart card).

2.2 Statements

In any formal system one defines the basic units of concern, often called formulas, which can take on different truth values, most often the values true and false. In our calculus, these basic units are called *statements*, and the truth values are *valid* and *not valid* rather than true and false, for reasons explained in Section 2.3. The term statement refers (1) to what principals "say", (2) to the pieces of evidence, and (3) to what is concluded in the calculus.

One kind of statements that one may wish to conclude is that an entity Alice made a certain statement s (i.e. Alice said s), in particular in a case where s is signed by a certain signature public key. This process of deriving the statement that Alice said s is traditionally known as *authentication* in the case where an entity draws this conclusion in her own view. It is also known as *non-repudiation* in the case where a legal system (e.g. a court) draws this conclusion or, more precisely, where an entity takes the court's view and convinces herself that the statement Alice said s would be derived by the court from the given evidence.

Note that in the case of non-repudiation, the final goal is often not to conclude that an entity said something, but rather that a certain privilege, value, or right (e.g. a payment agreement) has been transferred to another entity. We capture all of them in concept of a *right*. Transferring a right is a basic statement.

2.3 Views

The basic idea behind our calculus is that an entity (say Bob) draws conclusions from his evidence, consisting of statements, by applying certain inference rules.

[3] The term "principal" is sometimes used differently in the cryptographic literature. While in SPKI/SDSI public keys are called principals, in the calculus of Lampson et al. [18] and [3] all "sources for request" (users, machines, cryptographic keys) are called principals.

We call the collected evidence and the inference rules of a given entity his *view*.[4] The view reflects Bob's initial belief from which further believed statements can be derived. The fact that a statement s is in Bob's view does not imply that it is true in an absolute sense, only that in Bob's context it is reasonable to make this assumption. Similarly, the fact that s is not in Bob's view does not mean that s is false; it could be that Bob simply does not pay attention to s. Statements in an entity's view or derived from it according to the inference rules are therefore called *valid* rather than true. All statements that cannot be derived could be called "not valid" to define the second truth value, but this term will not be used.

The views (beliefs) of different entities are generally different. Trust relations in particular can be subject to an entity's experience, context, or policy. For instance, one can think of the legal system as consisting of a certain agreed-upon and published set of rules and statements (when neglecting many aspects related to the fact that not everything can be formalized and that human beings (e.g. judges) play an important role.) In a dispute, these statements and rules can be taken into a judge's view. Such statements could for instance be about the trustworthiness of certain legally or otherwise appointed certification authorities. An entity must therefore adopt a different view depending on whether the conclusions should be convincing only for herself or for a legal entity, say a judge.

3 Definitions

We clearly distinguish between the syntactic definitions of our calculus (which can be glanced through in a first reading) and the meaning of the statements.

3.1 Syntactic Definitions

Let $\mathcal{K}^e$ and $\mathcal{K}^s$ denote the (not necessarily distinct) sets of encryption public keys and signature public keys, respectively, and let $\mathcal{K}$ denote denote the set of all keys: $\mathcal{K} = \mathcal{K}^e \cup \mathcal{K}^s$. Let $\mathcal{E}$ be the set of entities. The set of principals consists of the public keys and entities and is denoted by $\mathcal{P}$: $\mathcal{P} = \mathcal{E} \cup \mathcal{K}$. By $\mathcal{R}$ we denote the set of rights. Elements of a set are written in small letters and variables in capital letters. Key variables and values carrying as subscript the name of an entity stand for the key allegedly belonging to the entity.

For describing the syntax of the statements and the rules, we use the Extended Backus-Naur Formalism (EBNF).[5] Each statement consists of a predefined name , together with a list of arguments in parenthesis. Each argument is

[4] The concept of a view was introduced in [20], and four different statements have been defined. In contrast, we recursively define a set of statements. Moreover, because different entities might interpret the evidence in different ways, also inference rules are part of a view.

[5] AB stands for the concatenation of the expressions A and B, $A|B$ for the the alternative between A and B, and $\{A\}$ for the repeated concatenation of A (including the empty word). Let *element*($\mathcal{T}$) stand for an element of a set $\mathcal{T}$ and *variable*($\mathcal{T}$) for a variable of type $\mathcal{T}$ (a type is a set).

a term of a certain type. All statements are in prefix notation, with exception of the statement says.

Definition 1.

$$
\begin{array}{lcl}
\mathit{term}(\mathcal{T}) & = & \mathit{element}(\mathcal{T}) \mid \mathit{variable}(\mathcal{T}) \\
\mathit{statement} & = & \mathsf{own}(\mathit{term}(\mathcal{E}), \mathit{term}(\mathcal{K})) \mid \\
 & & \mathsf{transfer}(\mathit{term}(\mathcal{E}), \mathit{term}(\mathcal{E} \cup \mathcal{K}^s), \mathit{term}(\mathcal{R})) \mid \\
 & & \mathit{term}(\mathcal{E} \cup \mathcal{K}^s)\ \mathsf{says}\ \mathit{statement} \mid \\
 & & \mathsf{commit}(\mathit{term}(\mathcal{E}), \mathit{term}(\mathcal{K}^s), \mathit{statement}) \mid \\
 & & \mathsf{trust}(\mathit{term}(\mathcal{E}), \mathit{statement}) \\
\mathit{inference\text{-}rule} & = & \text{“}\langle\text{”}\,\mathit{statement}\{\text{“,”}\,\mathit{statement}\}\,\text{“}\vdash\text{”}\,\mathit{statement}\,\text{“}\rangle\text{”}
\end{array}
$$

For convenience, we allow an equivalent notation for rules:

$$\mathit{inference\text{-}rule} \;=\; \frac{\mathit{statement}\{\text{“,”}\,\mathit{statement}\}}{\mathit{statement}}$$

Extending the above definition (this can be seen as syntactic sugar), we interpret a statement in which a set of values of the corresponding type (rather than a single term) is used as an argument, to be the set of statements resulting from plugging in, one by one, every element of the set.

Definition 2. The *view* of an entity a, denoted by View_a, is a set consisting of statements and inference rules. ∘

The following definitions describe the mechanism of deriving a statement from a view.

Definition 3. Let V be a variable of type $\mathcal{T}$ occurring in the statement s and let v be an element in $\mathcal{T}$. Let W be a variable of the same type as V. Then $s[V = v]$ denotes the statement that is obtained by replacing all occurrences of V in s by v (instantiation of V). Similarly, if s is a statement in which W does not occur, then $s[V = W]$ is the statement where V is renamed to W (variable substitution). ∘

Definition 4. Let $s_1, \ldots, s_i$ be statements. A *variable assignment* $\mathcal{I}$ for an inference rule $\langle s_1,\ s_2,\ \ldots,\ s_{i-1} \vdash\ s_i\rangle$ maps each variable occurring in the inference rule to an element of the corresponding type. $s_j^{\mathcal{I}}$ is the statement that is obtained by replacing all occurrences of the variables in s_j by the values specified in $\mathcal{I}$. ∘

Definition 5. A statement is *valid* in View_a if it is either contained in View_a or if it can be derived from View_a by repeated application of the inference rules in View_a. More precisely,

- All statements in View_a are valid.
- Let V be a variable of type $\mathcal{T}$ occurring in s, let W be a variable of the same type not occurring in s, and let v be any element of $\mathcal{T}$. If s is valid, then the statements $s[V = v]$ and $s[V = W]$ are also valid.[6]

[6] Hence if an argument is a variable, all the occurrences of the variable are universally quantified (∀).

– Let s_1, ..., s_i be statements. If for a rule of the form $\langle s_1, \ldots, s_{i-1} \vdash s_i \rangle$ in View_a there is a variable assignment $\mathcal{I}$ such that $s_j^{\mathcal{I}}$ is valid for $j = 1, \ldots, i-1$, then $s_i^{\mathcal{I}}$ is also valid. ◦

Example. We illustrate the mechanisms of variable instantiation and variable substitution in a simple example. Note that in the examples of Section 5, these formal steps are not shown explicitly. Consider the following view:

$$\begin{aligned}\mathsf{View}_A = \{\ &\langle X \text{ says } S,\ \mathsf{trust}(X, S) \vdash S\rangle,\\ &\mathsf{trust}(b, \mathsf{trust}(Y, X \text{ says } \mathsf{own}(X, K))),\\ &b \text{ says } \mathsf{trust}(c, Y \text{ says } \mathsf{own}(Y, K))\ \}\end{aligned}$$

One can derive $\mathsf{trust}(c, X \text{ says } \mathsf{own}(X, K))$ by the following sequence of application of Definition 5.

– $\mathsf{trust}(b, \mathsf{trust}(Y, X \text{ says } \mathsf{own}(X, K)))$ is valid, because it is in the view.

– $\mathsf{trust}(b, \mathsf{trust}(Y, X \text{ says } \mathsf{own}(X, K)))[Y{=}c] = \mathsf{trust}(b, \mathsf{trust}(c, X \text{ says } \mathsf{own}(X, K)))$ is valid.

– $\mathsf{trust}(b, \mathsf{trust}(c, X \text{ says } \mathsf{own}(X, K)))[X{=}Y] = \mathsf{trust}(b, \mathsf{trust}(c, Y \text{ says } \mathsf{own}(Y, K)))$ is valid.

– The inference rule $\langle X \text{ says } S, \mathsf{trust}(X, S) \vdash S\rangle$ can now be applied. Let $X = b$ and $S = Y \text{ says } \mathsf{own}(Y, K)$ be a variable assignment of the rule. The statements $\mathsf{trust}(b, \mathsf{trust}(c, Y \text{ says } \mathsf{own}(Y, K)))$ and $b \text{ says } \mathsf{trust}(c, Y \text{ says } \mathsf{own}(Y, K))$ are valid and therefore $\mathsf{trust}(c, X \text{ says } \mathsf{own}(X, K))$ is also valid.

3.2 Meaning of Statements

We now discuss the meaning of the statements. We only describe the case where all arguments are elements of a set ($x \in \mathcal{E}$, $k \in \mathcal{K}$, $k^s \in \mathcal{K}^s$, $e \in \mathcal{E} \cup \mathcal{K}^s$, $r \in \mathcal{R}$), rather than variables. The symbol s stands for a specific statement. If a statement contains variables as arguments, the meaning is implied by the meaning for specific values through the above definition of validity. Variable substitutions do not change the meaning of a statement.

$\mathsf{trust}(x, s)$ Entity x is trustworthy with respect to the statement s. The statement $\mathsf{trust}(x)$ means that x is trustworthy with respect to all statements.

k^s says s The statement s is digitally signed with the signature key k^s. We assume that every entity has a trusted computing base for checking digital signatures, i.e., every entity can verify whether k^s says s is valid.

x says s An entity x can say s by a variety of mechanisms. Saying s can mean to pronounce s, to write s on a piece of paper, to type s into a computer, to initiate a process that generates a string corresponding to s, or to create any other unique representation of s. Note that the act of signing s is not an act of saying s but rather of creating evidence for other entities to conclude that x says s.
Note that for instance in authentication protocols (e.g., for access control), it is important to know that a statement is fresh. In contrast to the BAN logic, it is not the goal of this paper to model authentication protocols, and hence freshness is not modeled. However, it is a possible extension of this work to include time aspects like freshness.

$\mathsf{own}(x, k)$ Entity x is an exclusive owner of the public key k or, equivalently, x solely controls the public key k in the sense that he can perform any operation requiring access to the secret key.

$\mathsf{commit}(x, k^s, s)$ Entity x is committed to the signature public key k^s with respect to statement s. A commitment of x to k^s with respect to s means that x agrees to be liable in a legal sense to have said s, provided s is digitally signed with k^s. $\mathsf{commit}(x, k^s)$ stands for the statement that x is committed to k^s with respect to all statements.

$\mathsf{transfer}(x, e, r)$ Entity x delegates a right r to e (or owes a right to e). If e is a signature public key k^s, then $\mathsf{transfer}(x, k^s, r)$ means that the right is delegated to an entity owning the public key k^s.[7]

4 Inference rules

Throughout the paper, $K \in \mathcal{K}^e \cup \mathcal{K}^s$, $K^s \in \mathcal{K}^s$ and $K^e \in \mathcal{K}^e$ are variables standing for any public key, for a signature public key and for an encryption public key, respectively. Similarly, X, $Y \in \mathcal{E}$, $E \in \mathcal{E} \cup \mathcal{K}^s$, and $R \in \mathcal{R}$ are variables for an entity, for an entity or a signature public key, and for a right, respectively. Furthermore, S is a variable of type $\mathcal{S}$, where $\mathcal{S}$ is the set of all syntactically correct statements.
We introduce inference rules that illustrate the conclusions that a can draw from the statements contained in her view View_a. We discuss rules for using

[7] In fact, a right could also be delegated to an encryption key: an entity can prove ownership by challenge-response protocol.

trust and rules applicable in the two settings of bilateral communications and legally binding statements. We do not claim in any sense that the rules proposed here are the only reasonable way for an entity to interpret the evidence. The rules might for instance differ from one legal environment to another; as explained in the previous section, we therefore allow the rules to be part of an entity's view.

4.1 Trust

Trust can be seen as a basic mechanism which allows one entity to believe a statement made by another. If one trusts X on S, and if X says S, then one has reason to believe S. Note that the statement X says S makes no sense because it means that every entity says every statement. However, it makes sense to use it within a rule because S will always be instantiated to a concrete statement or a set of statements.

Rule 1a.

$$\frac{\mathsf{trust}(X, S),\ X\ \mathsf{says}\ S}{S}$$

One can have a more detailed understanding of trust. In the BAN logic for instance, a server cannot only be trusted to provide the correct keys, but also to generate good keys.

Trust is a fundamental and rare resource. Many of the trust relations used in real life are not based on personal experience but rather on explicit or implicit recommendations, a concept first formalized in [4]. A recommendation is a statement of an entity about the trustworthiness of another entity. a could for instance recommend b for making statements about the ownership of keys: a says $\mathsf{trust}(b, \mathsf{own}(Z, K^s))$. An entity must explicitly be trusted to issue recommendations, otherwise the recommendation cannot be used as evidence to draw further conclusions.

4.2 Bilateral Communication

Assume that one wants to encrypt a message for X or verify whether a digitally signed message has been sent by X. In both cases, one needs to establish that X eexclusively owns a certain key. We argue that ownership need not be proven (in fact sole ownership cannot be proven), but what is needed is a statement by X that he solely owns the key. By making a false claim of ownership (either for a key not controlled or not exclusively controlled), X cannot achieve more than what he could achieve by other means. Namely, instead of violating secrecy by making a false claim of ownership for an encryption public key, X could achieve the same goal by revealing the contents of an encrypted (for X) message to another entity. Similarly, X could sign a message provided by another entity, thus violating authenticity. Hence, if X says that he solely controls K, one can derive that K is exclusively owned by X.

Rule 2a.

$$\frac{X \text{ says } \mathsf{own}(X, K)}{\mathsf{own}(X, K)}$$

Note that this rule could be replaced by the statement $\mathsf{trust}(X, \mathsf{own}(X, K))$. If one believes that X solely owns the signature key K^s, and if there is a statement S digitally signed with K^s, then one has reason to believe that X says S:

Rule 2b.

$$\frac{\mathsf{own}(X, K^s),\ K^s \text{ says } S}{X \text{ says } S}$$

Similarly, if one believes that X solely owns the encryption key K^e, then one can use K^e to encrypt a message for X. However, secrecy is not captured as a concept in this first version of our calculus, and hence there is no rule corresponding to Rule 2b.

4.3 Commitments and Non-Repudiation

As has been recognized by the legislative bodies issuing digital signature legislation, one has to be very cautious when taking digital signatures as evidence in a legal sense. For one thing, there are various technical risks (key compromise or system misuse due to flaws and bugs), cryptographic risks (e.g. the discovery of a fast factoring algorithm), and risks due to incompetence and negligence by the users.

On the other hand, there is also the problem that ownership statements are strongly context-dependent, and that this is not reflected in previous technical proposals for public-key certification. For instance, the statement by X that he controls a certain PGP public key K^s cannot be used as evidence that X agrees in a legal sense to be liable for statements signed by K^s. Rather, X must declare explicitly his intention to use the key in a legal context and, furthermore, for which kinds of statements (e.g. for any statements implying liability up to $1000). We call such a statement to accept liability for statements made by a signature key a *commitment* of an entity to the key. A commitment by X to a signature key K^s with respect to the statement S can be seen as a special kind of ownership claim.[8]

Rule 3a.

$$\frac{X \text{ says } \mathsf{commit}(X, K^s, S)}{\mathsf{commit}(X, K^s, S)}$$

Rule 3b.

$$\frac{\mathsf{commit}(X, K^s, S),\ K^s \text{ says } S}{X \text{ says } S}$$

Note that in a legal context there will be a restriction of what constitutes "saying" in our sense. In particular, the statement of being committed must be made in a specific way (e.g. by an ordinary signature on a paper contract).

[8] Note that by replacing $\mathsf{commit}(X, K^s, S)$ with $\mathsf{own}(X, K^s)$, Rules 3a and 3b exactly correspond to Rules 2b and 2a.

4.4 Transfer of Rights

In many legal scenarios, the statement of interest is not whether an entity X said something, but whether X delegated a right to another entity Y. A right is transferred from X to E if X says so.

Rule 4a.

$$\frac{X \text{ says } \mathsf{transfer}(X, E, R)}{\mathsf{transfer}(X, E, R)}$$

Privacy and anonymity are increasingly important issues in electronic commerce. It will become more and more common to transfer privileges to a key or pseudonym, without knowing the identity of the person controlling the key except in case of a dispute (for an example, see Section 5). Therefore one needs a mechanism for a key owner to claim the right transferred to a public key. Note that if a right is reproducible (e.g., access to a database), then several entities could control the public key. For rights that must inherently be restricted to one receiving entity (e.g., a payment), it must be assured in the first place by the receiving entity that the rights are transferred to a key it solely owns.

Rule 4b.

$$\frac{\mathsf{transfer}(X, K^s, R),\ \mathsf{own}(Y, K^s)}{\mathsf{transfer}(X, Y, R)}$$

One possibility to prove ownership for redeeming a right is a self-certificate, i.e., a statement that is digitally signed with K^s and asserts who the key owner of K^s is. Only a key owner of K^s can issue a self-certificate. One can postulate the rule that a self-certificate proves ownership of a key, i.e. the rule

Rule 4c.

$$\frac{K^s \text{ says } \mathsf{own}(Y, K^s)}{\mathsf{own}(Y, K^s)}$$

However, this rule makes sense only in the context of redeeming a right transferred to K^s, but of course not in the context of bilateral communication. An attacker could attach, by a self-certificate, an arbitrary identity to a public key it generated, and hence this statement should not be applicable in that context. In the sequel, we denote by

$$\mathsf{IR_P} = \{1\text{a}, 1\text{b}, 2\text{b}, 2\text{a}\}$$

the set of inference rules applicable in a private or bilateral scenario and by

$$\mathsf{IR_L} = \{1\text{a}, 1\text{b}, 3\text{a}, 3\text{b}, 4\text{a}, 4\text{b}, 4\text{c}\}$$

the rules that can be applied in a legal context.

5 Scenarios

We illustrate our rules on some concrete scenarios. The first example shows various ways in which an entity can derive that a certain public key is owned by a

certain entity. The purpose of the second example is to give a simplistic and idealized framework for concluding digital contracts (e.g., for electronic commerce). In the third scenario an entity delegates a right to a key owner, without knowing the identity of the key owner.
In order to keep the examples readable, we will not explicitly show the variable instantiations and substitutions made in the derivations. Instead, we only show, for every application of an inference rule, the instantiated preconditions and the consequence of the rule, separated by the symbol $\Rightarrow$ to indicate that the latter follows from the former.

5.1 Exchange of Public Keys for Authenticating and Encrypting Mail

Alice (denoted by a) wants to check whether a digitally signed e-mail message (which contains the message s) has been sent by Bob (denoted by b). Since the mails are for private use only, a will not require b to commit to the key k_b^s. In order to apply Rule 2a, a must somehow derive the statement b says $\mathsf{own}(b, k_b^s)$. This can happen in various ways.

Exchange of Public Keys by Non-Digital Means. a could directly obtain b's public key k_b^s from b. a could meet b in person, or b could provide his key during a telephone call; in this case a can verify that b says $\mathsf{own}(b, k_b^s)$ is valid. Assume that a later gets the digitally signed message s. Thus, her view contains the following two statements.

$$\mathsf{View}_a = \{\ k_b^s \text{ says } s,\ b \text{ says } \mathsf{own}(b, k_b^s)\ \} \cup \mathsf{IR_P}$$

a can apply the following two rules to derive b says s.

$$b \text{ says } \mathsf{own}(b, k_b^s) \ \Rightarrow\ \mathsf{own}(b, k_b^s) \qquad (2a)$$
$$\mathsf{own}(b, k_b^s),\ k_b^s \text{ says } s \ \Rightarrow\ b \text{ says } s \qquad (2b)$$

a could also rely on the statements of entities she trusts in order to derive b says $\mathsf{own}(b, k_b^s)$. In the following, let

$$s_1 = X \text{ says } \mathsf{own}(X, K^s)$$

If her friend Carol (c) says that b says $\mathsf{own}(b, k_b^s)$, and if a trusts c to provide the keys of other entities (i.e., c is trusted on s_1), a can authenticate b's key.

$$\mathsf{View}_a = \{\ k_b^s \text{ says } s,\ c \text{ says } b \text{ says } \mathsf{own}(b, k_b^s),\ \mathsf{trust}(c, s_1)\ \} \cup \mathsf{IR_P}$$

The statement $\mathsf{trust}(c, s_1)$ means that c is trusted on the statements that some entity X has claimed exclusive ownership of some key K^s. Thus c is trusted on b says $\mathsf{own}(b, k_b^s)$ (variable instantiation), and because c says b says $\mathsf{own}(b, k_b^s)$ is valid in a's view, a can apply Rule 1a to derive the statement b says $\mathsf{own}(b, k_b^s)$. From this, a can apply the same two rules as above to conclude that b says s.

Certificates and Certificate Chains. Instead of making the statement c says b says $\mathsf{own}(b, k_b^s)$ directly towards a, this is achieved, in a typical scenario, with a certificate containing this statement. Of course, we also need the statement c says $\mathsf{own}(c, k_c^s)$.

$$\begin{aligned}\mathsf{View}_a = \{\ & k_b^s \text{ says } s,\ k_c^s \text{ says } b \text{ says } \mathsf{own}(b, k_b^s),\ c \text{ says } \mathsf{own}(c, k_c^s),\\ & \mathsf{trust}(c, s_1)\ \} \cup \mathsf{IR_P}\end{aligned}$$

Since c is trusted to provide her own signature key, a concludes $\mathsf{own}(c, k_c^s)$ by Rule 2a. Thus, by Rule 2b, a can derive c says b says $\mathsf{own}(b, k_b^s)$. Finally, a can again apply the same rules as in the previous view to derive b says s.
Perhaps another friend of Alice, Dave (d), knows that c says $\mathsf{own}(c, k_c^s)$. If a trusts him to provide the keys of other entities, and if a has d's signature key (d says $\mathsf{own}(d, k_d^s)$), a can obtain c's key.

$$\begin{aligned}\mathsf{View}_a = \{\ & k_b^s \text{ says } s,\ k_c^s \text{ says } b \text{ says } \mathsf{own}(b, k_b^s),\ d \text{ says } \mathsf{own}(d, k_d^s),\\ & k_d^s \text{ says } c \text{ says } \mathsf{own}(c, k_c^s),\ \mathsf{trust}(c, s_1),\ \mathsf{trust}(d, s_1)\ \} \cup \mathsf{IR_P}\end{aligned}$$

Here d issues a certificate for c (digitally signed with k_d^s) and c for b (with k_c^s). Since a trusts both c and d to provide the keys of other entities, a can retrieve the signature key of b. The following sequence of rules is applied:

$$\begin{aligned}
d \text{ says } \mathsf{own}(d, k_d^s) &\Rightarrow \mathsf{own}(d, k_d^s) && \text{(2a)}\\
\mathsf{own}(d, k_d^s),\ k_d^s \text{ says } c \text{ says } \mathsf{own}(c, k_c^s) &\Rightarrow d \text{ says } c \text{ says } \mathsf{own}(c, k_c^s) && \text{(2b)}\\
d \text{ says } c \text{ says } \mathsf{own}(c, k_c^s),\ \mathsf{trust}(d, s_1) &\Rightarrow c \text{ says } \mathsf{own}(c, k_c^s) && \text{(1a)}\\
c \text{ says } \mathsf{own}(c, k_c^s) &\Rightarrow \mathsf{own}(c, k_c^s) && \text{(2a)}\\
\mathsf{own}(c, k_c^s),\ k_c^s \text{ says } b \text{ says } \mathsf{own}(b, k_b^s) &\Rightarrow c \text{ says } b \text{ says } \mathsf{own}(b, k_b^s) && \text{(2b)}\\
c \text{ says } b \text{ says } \mathsf{own}(b, k_b^s),\ \mathsf{trust}(c, s_1) &\Rightarrow b \text{ says } \mathsf{own}(b, k_b^s) && \text{(1a)}\\
b \text{ says } \mathsf{own}(b, k_b^s) &\Rightarrow \mathsf{own}(b, k_b^s) && \text{(2a)}\\
\mathsf{own}(b, k_b^s),\ k_b^s \text{ says s} &\Rightarrow b \text{ says s} && \text{(2b)}
\end{aligned}$$

Such a scheme where one entity certifies the key of another entity is often referred to as a certificate chain.
Trust in c could also be established by a recommendation of d for c. d could for instance recommend c for certifying ownership of keys. In this case, a must explicitly trust d for recommending other entities.

$$\begin{aligned}\mathsf{View}_a = \{\ & k_b^s \text{ says } s,\ k_c^s \text{ says } b \text{ says } \mathsf{own}(b, k_b^s),\ \mathsf{trust}(d, s_1),\ \mathsf{trust}(d, s_2)\\ & k_d^s \text{ says } c \text{ says } \mathsf{own}(c, k_c^s),\ d \text{ says } \mathsf{trust}(c, s_1),\ d \text{ says } \mathsf{own}(d, k_d^s)\ \} \cup \mathsf{IR_P}\end{aligned}$$

where

$$s_2 = \mathsf{trust}(X, Y \text{ says } \mathsf{own}(Y, K^s))$$

The statement $\mathsf{trust}(d, s_2)$ stands for a's trust in d to issue recommendations for other entities.

Exchanging an Encryption Key. While in some systems (for instance PGP), the same key is used for signing and encrypting messages, it makes sense to separate these two functions. The authentication of an encryption key is identical to the authentication of a signature key, i.e., the statement b says $\mathsf{own}(b, k_b^e)$ must be derived.

Self-Certificates. The main purpose of PGP is to encrypt and authenticate mail messages. There seems to be a misconception about the role of self-certificates. For instance, [28] suggests to append self-certificates to a public key. However, as illustrated by the rules, self-certificates do not occur as a precondition of a rule and are not needed for exchanging encryption and signature keys on a bilateral basis. Hence they can at most mislead an entity a to believe that a key is b's, if a draws incorrect conclusions. More precisely, neither the statement b says $\mathsf{own}(b, k_b^s)$ nor the statement $\mathsf{own}(b, k_b^s)$ can be derived from the following view

$$\mathsf{View}_a = \{\ k_b^s \text{ says } \mathsf{own}(b, k_b^s)\ \} \cup \mathsf{IR_P}$$

because an impostor e could of course generate a public key k_e^s and sign with it the self-certificate: k_e^s says $\mathsf{own}(b, k_e^s)$.

5.2 A Simple Legal Framework

In the following example we describe an infrastructure allowing two entities to conclude a contract with digital signatures. A contract can be seen as a transfer of a right r_1 from a to b ($\mathsf{transfer}(a, b, r_1)$), and possibly also a right r_2 from b to a ($\mathsf{transfer}(b, a, r_2)$). In order to achieve non-repudiation, each entity's goal is to be able to derive in the legal system's view that the other party has transferred the corresponding right.
Unlike in the scenarios described in Section 5.1, it does not matter for a whether she believes that a certification authority is trustworthy, but rather whether the legal system does. It can be assumed that the legal system's policy and trust relations, i.e. its view, is clearly specified and publicly known, and that also the public keys root CAs are publicly known in an authenticated manner.
In the sequel, we adopt the legal system's (e.g., a judge's) view who wants to verify (i.e. derive) the statement $\mathsf{transfer}(a, b, r_1)$.
It is conceivable that the legislator defining the legal system appoints a set of distinguished authorities that are trusted by default (i.e. in the legal system's view): certification authorities, naming authorities registration authorities, key revocation authorities (if not part of the CA), etc., subsumed below under the term CA. In the sequel, let

$$s_1 = \mathsf{transfer}(Y, Z, \{R | R \text{ stands for a cash value up to \$1000}\})$$
$$s_2 = Y \text{ says } \mathsf{commit}(Y, K^s, s_1)$$
$$s_3 = \mathsf{trust}(X, s_2).$$

Note that from a syntactical point of view, s_1 is not a statement but stands for a set of statements. In our scenario, b tries to collect evidence that a owes

him \$500. The judge trusts one root certification authority ca_1 to license other certification authorities (this corresponds to the statement $\mathsf{trust}(ca_1, s_3)$ in the view shown below). Furthermore, he trusts ca_1 to correctly certify the keys of other entities ($\mathsf{trust}(ca_1, s_2)$). Additionally, the judge believes that ca_1 uses k_1^s as its signature key.
Assuming that ca_1 has licensed ca_2 (ca_1 says $\mathsf{trust}(ca_2, s_2)$), b can retrieve a digitally signed statement of ca_2 asserting that a is committed to k_a^s with respect to s_1 (ca_2 says a says $\mathsf{commit}(a, k_a^s, s_1)$ The statement k_a^s says $\mathsf{transfer}(a, b, r_1)$ is in the judge's view because a signed the contract and b produced the fact as evidence in the dispute. Thus, the judge's view is:

$$\begin{aligned}\mathsf{View}_{judge} = \{\ & ca_1 \text{ says } \mathsf{commit}(ca_1, k_1^s),\ k_1^s \text{ says } \mathsf{trust}(ca_2, s_2),\\ & k_1^s \text{ says } ca_2 \text{ says } \mathsf{commit}(ca_2, k_2^s),\ k_2^s \text{ says } a \text{ says } \mathsf{commit}(a, k_a^s, s_1),\\ & k_a^s \text{ says } \mathsf{transfer}(a, b, r_1),\ \mathsf{trust}(ca_1, s_2), \mathsf{trust}(ca_1, s_3)\ \} \cup \mathsf{IR_L}\end{aligned}$$

The following table summarizes the complete sequence of applications of inference rules for deriving $\mathsf{transfer}(a, b, r_1)$ in the judge's view. Note again that it is irrelevant in this context whether an actual dispute is being resolved by the judge or whether b convinces himself that in case of a dispute, he would be able to produce a convincing collection of evidence for $\mathsf{transfer}(a, b, r_1)$.

$$\begin{aligned}
ca_1 \text{ says } \mathsf{commit}(ca_1, k_1^s) &\Rightarrow \mathsf{commit}(ca_1, k_1^s)\\
\mathsf{commit}(ca_1, k_1^s),\ k_1^s \text{ says } \mathsf{trust}(ca_2, s_2) &\Rightarrow ca_1 \text{ says } \mathsf{trust}(ca_2, s_2)\\
ca_1 \text{ says } \mathsf{trust}(ca_2, s_2),\ \mathsf{trust}(ca_1, s_3) &\Rightarrow \mathsf{trust}(ca_2, s_2)\\
\mathsf{commit}(ca_1, k_1^s),\ k_1^s \text{ says } ca_2 \text{ says } \mathsf{commit}(ca_2, k_2^s) &\Rightarrow ca_1 \text{ says } ca_2 \text{ says } \mathsf{commit}(ca_2, k_2^s)\\
ca_1 \text{ says } ca_2 \text{ says } \mathsf{commit}(ca_2, k_2^s),\ \mathsf{trust}(ca_1, s_2) &\Rightarrow ca_2 \text{ says } \mathsf{commit}(ca_2, k_2^s)\\
ca_2 \text{ says } \mathsf{commit}(ca_2, k_2^s) &\Rightarrow \mathsf{commit}(ca_2, k_2^s)\\
\mathsf{commit}(ca_2, k_2^s),\ k_2^s \text{ says } a \text{ says } \mathsf{commit}(a, k_a^s, s_1) &\Rightarrow ca_2 \text{ says } a \text{ says } \mathsf{commit}(a, k_a^s, s_1)\\
ca_2 \text{ says } a \text{ says } \mathsf{commit}(a, k_a^s, s_1),\ \mathsf{trust}(ca_2, s_2) &\Rightarrow a \text{ says } \mathsf{commit}(a, k_a^s, s_1)\\
a \text{ says } \mathsf{commit}(a, k_a^s, s_1) &\Rightarrow \mathsf{commit}(a, k_a^s, s_1)\\
\mathsf{commit}(a, k_a^s),\ k_a^s \text{ says } \mathsf{transfer}(a, b, r_1) &\Rightarrow a \text{ says } \mathsf{transfer}(a, b, r_1)\\
a \text{ says } \mathsf{transfer}(a, b, r_1) &\Rightarrow \mathsf{transfer}(a, b, r_1)
\end{aligned}$$

5.3 Transferring a Right to a Key Owner

We sketch a scenario where a delegates a right to a key owner, without knowing who the key owner is.[9] It is likely that a marketplace on the Internet for certain services involving digital information (e.g. designing clip art, programming a shell script, etc.) will emerge in the near future. In this context, it may be quite possible that some of the service providers are known only by a pseudonym which could typically be the public key itself. The following example illustrates how a right (e.g. a payment obligation) can be transferred to a public key.
Assume that b, known by the public key k_b^s has done a job for a and that a has written a digital cheque (called r) payable to k_b^s. Adopting the judge's view as

[9] This example was first given by C.Ellison in similar form.

in the previous example, b collects the following evidence:

$$\begin{aligned}\mathsf{View}_{judge} = \{\ & ca\ \mathsf{says}\ \mathsf{commit}(ca, k_1^s),\ \mathsf{trust}(ca),\ k_1^s\ \mathsf{says}\ a\ \mathsf{says}\ \mathsf{commit}(a, k_a^s),\\ & k_a^s\ \mathsf{says}\ a\ \mathsf{transfer}(a, k_b^s, r)\ k_b^s\ \mathsf{says}\ \mathsf{own}(b, k_b^s)\ \} \cup \mathsf{IR}_{\mathsf{L}}\end{aligned}$$

The following sequence of derivations convinces b that the evidence is sufficient.

$$\begin{aligned}
ca\ \mathsf{says}\ \mathsf{commit}(ca, k_1^s) &\Rightarrow \mathsf{commit}(ca, k_1^s) && (3a)\\
\mathsf{commit}(ca, k_1^s),\ k_1^s\ \mathsf{says}\ a\ \mathsf{says}\ \mathsf{commit}(a, k_a^s) &\Rightarrow ca\ \mathsf{says}\ a\ \mathsf{says}\ \mathsf{commit}(a, k_a^s) && (3b)\\
ca\ \mathsf{says}\ a\ \mathsf{says}\ \mathsf{commit}(a, k_a^s),\ \mathsf{trust}(ca) &\Rightarrow a\ \mathsf{says}\ \mathsf{commit}(a, k_a^s) && (1a)\\
a\ \mathsf{says}\ \mathsf{commit}(a, k_a^s) &\Rightarrow \mathsf{commit}(a, k_a^s) && (3a)\\
\mathsf{commit}(a, k_a^s),\ k_a^s\ \mathsf{says}\ a\ \mathsf{transfer}(a, k_b^s, r) &\Rightarrow a\ \mathsf{says}\ \mathsf{transfer}(a, k_b^s, r) && (3b)\\
a\ \mathsf{says}\ \mathsf{transfer}(a, k_b^s, r) &\Rightarrow \mathsf{transfer}(a, k_b^s, r) && (4a)\\
k_b^s\ \mathsf{says}\ \mathsf{own}(b, k_b^s) &\Rightarrow \mathsf{own}(b, k_b^s) && (4b)\\
\mathsf{transfer}(a, k_b^s, r),\ \mathsf{own}(b, k_b^s) &\Rightarrow \mathsf{transfer}(a, b, r) && (4c)
\end{aligned}$$

6 Concluding Remarks and Open Problems

The proposed calculus captures a number of important aspects of public-key certification, but it is by no means a formalism that could directly be used in a concrete legal system. However, we hope that it is a possible starting point for research into formalizing and reasoning about processes in the digital economy. Many issues remain open including time aspects (e.g. the concept of freshness, time stamping, etc.), modeling certificate and public-key revocation (more generally the revocation of any statement), and extending the model to capture degrees of belief and contradicting evidence.

Acknowledgments

We wish to thank Christian Cachin and Thomas Kühne for interesting discussions and comments, and the anonymous referees for helpful suggestions.

References

1. I. I. S. 9594-8. Information technology, open systems interconnection, the directory, part 8: Authentication framework, 1990.
2. M. Abadi. On SDSI's linked local name spaces. In *Proceedings of the 10th IEEE Computer Security Foundations Workshop*, pages 98–108. IEEE Computer Society, 1997.
3. M. Abadi, M. Burrows, B. Lampson, and G. Plotkin. A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706–734, September 1993.
4. T. Beth, M. Borcherding, and B. Klein. Valuation of trust in open systems. In D. Gollmann, editor, *Computer Security - Esorics '94*, volume 875 of *Lecture Notes in Computer Science*, pages 3–18. Springer Verlag, Berlin, 1994.

5. M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *Proceedings of the Symposium on Security and Privacy*, pages 164–173. IEEE Computer Society Press, 1996.
6. M. Blaze, J. Feigenbaum, and M. Strauss. Compliance checking in the policymaker trust management system. In R. Hirschfeld, editor, *Financial Cryptography*, volume 1465 of *Lecture Notes in Computer Science*, pages 254–274. Springer Verlag, Berlin, 1998.
7. C. Boyd. Security architectures using formal methods. *IEEE Journal on Selected Areas in Communications*, 11(5):694–701, 1993.
8. M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.
9. E. Campbell, R. Safavi-Naini, and P. Pleasants. Partial belief and probabilistic reasoning in the analysis of secure protocols. In *The computer Security Foundations Workshop V*, pages 84–91, 1992.
10. D. Chadwick and A. Young. Merging and extending the PGP and PEM trust models. *IEEE Network Magazine*, May 1997.
11. S. Consortium. Basic services, architecture and design, available at http://www.semper.org/info/index.html. Technical report, SEMPER, 1996.
12. C. Ellison. Establishing identity without certification authorities. In USENIX Association, editor, *6th USENIX Security Symposium, July 22–25, 1996. San Jose, CA*, pages 67–76. USENIX, July 1996.
13. C. E. et al. SPKI http://www.clark.net/pub/cme/html/spki.html. Internet Draft, 1998. Expires: 16 September 1998.
14. B. Fox and B. LaMaccia. Certificate revocation: Mechanisms and meaning. In R. Hirschfeld, editor, *Financial Cryptography*, volume 1465 of *Lecture Notes in Computer Science*, pages 158–164. Springer Verlag, Berlin, 1998.
15. J. Glasgow, G. MacEwen, and P. Panagaden. A logic for reasoning about security. *ACM transactions on Computer Systems*, 10(3):226–264, 1992.
16. T. M. C. Group. MCG - internet open group on certification and security, http://mcg.org.br/, 1998.
17. P. Kocher. On certificate revocation and validation. In R. Hirschfeld, editor, *Financial Cryptography*, volume 1465 of *Lecture Notes in Computer Science*, pages 172–177. Springer Verlag, Berlin, 1998.
18. B. Lampson, M. Abadi, M. Burrows, and E. Wobber. Authentication in distributed systems: Theory and practice. *ACM Transactions on Computer Systems*, 10(4):265–310, November 1992.
19. I. Lehti and P. Nikander. Certifying trust. In H. Imai and Y. Theng, editors, *Proceedings of the first international workshop on Practice and Theory in Public Key Cryptography, PKC'98*, pages 83–98, 1998.
20. U. Maurer. Modelling a public-key infrastructure. In E. Bertino, H. Kurth, G. Martella, and E. Montolivo, editors, *Proceedings 1996 European Symposium on Research in Computer Security (ESORICS' 96), Lecture Notes in Computer Science, Springer*, LNCS, pages 325–350, 1996.
21. U. Maurer and P. Schmid. A calculus for secure channel establishment in open networks. In D. Gollmann, editor, *Proc. 1994 European Symposium on Research in Computer Security (ESORICS' 94)*, volume 875, pages 175–192. Lecture Notes in Computer Science, 1994.
22. S. Micali. Efficient certificate revocation. Technical report, Technical Memo MIT/LCS/TM-542b, 1996.

23. M. Myers. Revocation: Options and challenges. In R. Hirschfeld, editor, *Financial Cryptography*, volume 1465 of *Lecture Notes in Computer Science*, pages 165–172. Springer Verlag, Berlin, 1998.
24. M. Naor and K. Nissim. Certificate revocation and certificate update. *Proceedings of Usenix '98*, pages 217–228, January 1998.
25. M. Reiter and S. Stubblebine. Path independence for authentication in large-scale systems. *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 57–66, 1997.
26. R. Rivest. Can we eliminate certificate revocation lists? In R. Hirschfeld, editor, *Proceedings of Financial Cryptography 1998*, pages 178–183, 1998.
27. R. Rivest and B. Lampson. SDSI – A simple distributed security infrastructure, http://theory.lcs.mit.edu/~cis/sdsi.html. Presented at CRYPTO'96 Rumpsession, April 1996.
28. W. Stallings. *Protect your privacy*. Prentice Hall, 1996.
29. S. Stubblebine and R. Wright. An authentication logic supporting synchronization, revocation, and recency. In *SIGSAC: 3rd ACM Conference on Computer and Communications Security*. ACM SIGSAC, 1996.
30. P. Syverson and C. Meadows. A logical language for specifying cryptographic protocols requirements. In *IEEE Conferences on Research in Security and Privacy*, pages 165–180, 1993.
31. R. Yaholem, B. Klein, and T. Beth. Trust relationships in secure systems - a distributed authentication perspective. In *Proceedings of the IEEE Conference on Research in Security and Privacy*, pages 150–164, 1993.
32. P. R. Zimmermann. *The Official PGP User's Guide*. MIT Press, Cambridge, MA, USA, 1995.