Modeling Public Key Infrastructure in the Real World

John Marchesini and Sean Smith

BindView Corporation and Department of Computer Science, Dartmouth College
john.marchesini@bindview.com, sws@cs.dartmouth.edu

Abstract. PKIs are complex distributed systems that are responsible for giving users enough information to make reasonable trust judgments about one another. Since the currencies of PKI are trust and certificates, users who make trust decisions (often called *relying parties*) must do so using only some initial trust beliefs about the PKI and some pile of certificates (and other assertions) they received from the PKI. Given a certificate, a relying party needs to conclude that the keyholder described by the certificate actually possesses the properties described by the certificate. In this paper, we present a calculus that allows relying parties to make such trust judgements. Our calculus extends Maurer's deterministic model, and is focused on real world issues such as time, revocation, delegation, and heterogeneous certificate formats. We then demonstrate how our calculus can be used to reason about numerous situations that arise in practice.

1 Introduction

PKIs are complex distributed systems that are responsible for giving users enough information to make reasonable trust judgments about one another. While there are a number of metrics we can use to reason about PKIs, one measure stands out: we say a PKI is *correct* if it allows Alice to conclude about Bob what she should, and disallows her from concluding things she should not. PKI designers need tools which can accurately evaluate the correctness of their designs and clearly illustrate what types of trust judgments their systems enable. The literature contains a number of approaches for applying formal methods to the PKI problem (e.g., [11, 13, 15, 20, 23]). The modeling work of Ueli Maurer [20] stands out, as it is simple, flexible, and is used to reason about PKI.

We are primarily concerned with designing, building, and deploying PKI systems which allow relying parties to make reasonable trust judgments. We have applied Maurer's calculus to model some of the systems we have seen in the wild as well as systems we have built in the lab. However, the real world is messy. Repeatedly, we find that the calculus cannot model some of the concepts we see in practice. For example:

- Usually, what matters about a public key is not some innate "authenticity" of it, but whether the keyholder has the properties to which the certificate attests.
- Certificates carry more than names; they carry extensions, use policies, attributes, etc. Some types of certificates (e.g., X.509 Attribute Certificates [8]) bind a key to a set of properties, and other types (e.g., SDSI/SPKI [7]) do not require names at all. In many real-world PKI applications, a globally unique name is not even the relevant parameter [5, 6].

- Certificates and beliefs expire and/or get revoked. Some systems use short certificate lifespans as a security advantage. Systems which use multiple certificates to describe an entity can have lifespan mismatches. For example, an Attribute Certificate that contains the courses Alice is enrolled in this term may expire well before her Identity Certificate.
- Some systems allow users to delegate some or all of their authority to other users.
- Some systems use a combination of multiple certificate types. The Grid community's MyProxy [22] system uses X.509 certificates in conjunction with short-lived Proxy Certificates [24, 26] for authentication and dynamic delegation. Greenpass uses an X.509 certificate in conjunction with a SDSI/SPKI certificate to express delegation.
- Many federated PKI systems (such as the Federal Bridge Certification Authority, the Higher Education Bridge Certification Authority, and SAFE) involve multiple entities issuing multiple statements about the trustworthiness of multiple users.

In this paper, rather than start with a calculus and attempt to make all of the PKIs we see fit into the calculus, we start with the things we have seen, and rework Maurer's calculus to allow us to reason about all of them. We begin by reviewing Maurer's calculus in Section 2. Then, we extend the calculus in Section 3 in order to make a more powerful tool for evaluating PKIs. Section 4 uses the extended calculus to reason about a number of real-world PKI scenarios, and relates this work to ideas in *trust management.* Section 5 concludes.

2 Maurer's Calculus

In 1996, Ueli Maurer's seminal "Modelling a Public-Key Infrastructure" [20] presented a deterministic model for PKI. In this model, a relying party Alice can use a certificate issued by Certification Authority (CA) X for user Bob if and only if Alice knows the public key for X and believes that it is *authentic*, and Alice *trusts* X to be honest and to correctly authenticate the owner of a public key before signing it. To determine whether Alice can deduce these facts, the calculus contains four types of *statements* and two *inference rules*. Alice can use her *initial view* (her axioms) and the rules to derive new statements. A *valid statement* is one contained in Alice's *derived view.*

The calculus introduced two concepts which are worth clarifying. First, a *recommendation*, transfers trust. Similar to a certificate, a recommendation grants the power to issue certificates and/or further recommendations. For example, if entity X has issued a recommendation to entity Y, then X is stating that it believes Y is trustworthy enough to issue certificates and further recommendations. Second, a *trust level parameter* limits the length of recommendations and certificate chains. For instance, if Alice trusts X at level 3, then she will accept certificate chains with a maximum length of 3.

Maurer also presents a useful graphical notation for the calculus, but given space constraints, we do not reproduce the details of Maurer's definitions here.

Maurer's deterministic model is appealing because it is simple and flexible. However, when we apply the model to the types of systems we deal with in practice, we discover limits of its applicability.

Authenticity Maurer's "Authenticity of public keys" is the wrong concept. In practice, we find that a relying party does not care about some innate "authenticity" of a public key, but rather about the binding between the public key and the information in the certificate. Often, the portion of the certificate information that defines the subject's name is not what Alice cares about—she may instead care about key usage policies, constraints, or other extensions.

Time In real-world PKIs, certificates expire, beliefs expire, and certificates get revoked. Without any concept of time, Maurer's model makes it impossible for relying parties to take such events into consideration when making trust decisions.

Delegation Sometimes, Bob would like to give another party the right to claim some of the attributes in his certificate. For instance, Bob may want to let Charlie claim to be Bob, so that Charlie can act as Bob. Diane may want to issue a certificate to Frank which indicates he is one of her teaching assistants. Maurer's recommendations are all-or-nothing, meaning that if Alice has issued a recommendation to Bob, then Alice is claiming Bob is trustworthy for the same set of operations that Alice is trustworthy for. In practice, Alice may want to limit what properties she gives to Bob.

Verification Maurer claims that certificates and recommendations are *allegedly* issued by an entity, because verification is outside the scope of the calculus. However, verification—including the various contending approaches to checking revocation and expiration—is an important (and messy) part of real world PKI [4, 10, 12, 19, 21]. For example, assume that a relying party Alice has the following initial view:

$$\mathit{View}_A = \{\mathit{Aut}_{A,X},\ \mathit{Trust}_{A,X,1},\ \mathit{Cert}_{X,Y}\}\ .$$

Even if $\mathit{Cert}_{X,Y}$ is invalid for some reason (e.g., revocation, expiration, usage violation), Alice can still derive the authenticity of Y's public key:

$$\mathit{Aut}_{A,X},\ \mathit{Trust}_{A,X,1},\ \mathit{Cert}_{X,Y} \vdash \mathit{Aut}_{A,Y}$$

Thus Alice draws an incorrect conclusion.

3 A Model for the Real World

Our revised model is rooted in Maurer's deterministic model, but extends it in order to deal with the complexity of real-world PKIs. From a high level, our extensions involve several elements.

1. We generalize Maurer's *Authenticity of public keys* to capture the notion of the authenticity of the binding between a public key and the certificate information.
2. We add the concept of time to Maurer's calculus so that we can model expiration and revocation.

3. We replace Maurer's *Recommendation* with a *Trust Transfer* which allows an entity A to give entity B the right to claim some or all of A's certificate information. This replacement allows us to remove the non-intuitive trust level parameter from the calculus, and to explicitly handle the various forms of trust transfer that occur in real-world PKI.
4. We introduce the notion of *validity templates* which are used to capture format-specific definitions of a statement's validity.
5. We redefine the inference rules to utilize these extensions.

(We also change the notation to use postfix instead of subscripts for the arguments, to improve readability.)

3.1 The Model

Informally, we use two concepts to make Maurer's deterministic model time-aware. The *lifespan* of a statement s is the time interval from t_j to t_k on which s can be used in trust calculations. We denote lifespans as the interval $\mathcal{I}$ where $\mathcal{I}$ is the time interval $[t_j, t_k]$. At time $t > t_k$, we say that s has *expired* and is no longer usable in trust calculations. We say statement s is *active* at time t if and only if $t \in \mathcal{I}$, the lifespan of s. (In theory, we could also add two levels of time: the time period during which the assertion is true, and the time period during which a party may believe and use this assertion. However, we found the simpler approach sufficed.)

We use the concept of a *domain* to indicate the set of properties that a certificate issuing entity may assign to its subjects. Intuitively, the domain of an entity is what it is allowed to vouch for. For example, the Dartmouth College CA can bind names, Dartmouth-specific attributes, and other extensions to public keys. Thus, the CA's domain (denoted as the set $\mathcal{D}$) is the set of names, Dartmouth-specific attributes, and extensions it can bind to public keys. The Dartmouth CA cannot bind Department of Defense (DoD)-specific attributes to public keys because it is not authorized to vouch for the DoD—i.e., the DoD-specific attributes are not in $\mathcal{D}$.

With these three concepts, we can formally define our model with the following two definitions.

Definition 1. In our model, statements and their representations are one of the following forms:

– **Authenticity of binding.** *Aut*$(A, X, \mathcal{P}, \mathcal{I})$ denotes A's belief that, during the interval $\mathcal{I}$, entity X (i.e., the entity holding the private key K_X) has the properties defined by the set $\mathcal{P}$.

  The symbol is an edge from A to X labeled with $\mathcal{P}, \mathcal{I}$: $A \xrightarrow{\mathcal{P},\mathcal{I}} X$.
– **Trust.** *Trust*$(A, X, \mathcal{D}, \mathcal{I})$ denotes A's belief that, during the interval $\mathcal{I}$, entity X is trustworthy for issuing certificates over domain $\mathcal{D}$.

  The symbol is a dashed edge from A to X labeled $\mathcal{D}, \mathcal{I}$: $A \overset{\mathcal{D},\mathcal{I}}{\dashrightarrow} X$.
– **Certificates.** *Cert*$(X, Y, \mathcal{P}, \mathcal{I})$ denotes the fact that X has issued a certificate to Y which, during the interval $\mathcal{I}$, binds Y's public key to the set of properties $\mathcal{P}$.

  The symbol is an edge from X to Y labeled with $\mathcal{P}, \mathcal{I}$: $X \xrightarrow{\mathcal{P},\mathcal{I}} Y$.

- **Trust Transfers.** *Tran*$(X, Y, \mathcal{P}, \mathcal{I})$ denotes that A holds a trust transfer issued by X which, during the interval $\mathcal{I}$, binds Y's public key to the set of properties $\mathcal{P}$. The symbol is a dashed edge from X to Y labeled with $\mathcal{P}, \mathcal{I}$: $X \overset{\mathcal{P},\mathcal{I}}{\dashrightarrow} Y$.
- **Certificate Validity Templates.** *Valid*$\langle A, C, t\rangle$ denotes A's belief that certificate C is valid at evaluation time t according to the definition of validity appropriate for C's format. Minimally, the issuer's signature over C must be verified and C must be active.
- **Transfer Validity Templates.** *Valid*$\langle A, T, t\rangle$ denotes A's belief that trust transfer T is valid at evaluation time t according to the definition of validity appropriate for T's format. Minimally, the issuer's signature over T must be verified and T must be active.

As with Maurer's model, some of the symbols are identical because of the similarity in meaning. Authenticity can be thought of as a certificate signed by one's own private key; trust can be thought of as a recommendation signed by one's own private key.

We introduce the notion of *template* because different PKI approaches have different (and non-trivial) ways of expressing validity of certificates and transfers. For example, validity for X.509 identity certificates may be determined by expiration dates and the absence of the certificate on a currently valid *certificate revocation list (CRL)*; validity of transfer in an X.509 identity certificate may be determined by basic constraints and usage bits in a certificate held by the source party.

Alice's initial view is denoted *View*$_A$, as in Maurer's deterministic model. Under our model, if Alice wishes to verify that Bob had some property p at time t, she must be able to derive the statement *Aut*$(A, B, \mathcal{P}, \mathcal{I})$ where $t \in \mathcal{I}$ and $p \in \mathcal{P}$. In many cases, the evaluation time t is the current time, meaning that Alice wants to verify that Bob currently has some property p. It should be noted, however, that the model still functions if the evaluation time t is some time in the past or the future. Such scenarios will be examined more closely in Section 4.

Definition 2. In our model, a statement is valid if and only if it is either contained in *View*$_A$ or if it can be derived from *View*$_A$ by applications of the following inference rules:

$$\forall X, Y, t \in \{\mathcal{I}_0 \cap \mathcal{I}_1\}, \mathcal{Q} \subseteq \mathcal{D}: \tag{1}$$

$$\textit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \textit{Trust}(A, X, \mathcal{D}, \mathcal{I}_1),\ \textit{Valid}\langle A, \textit{Cert}(X, Y, \mathcal{Q}, \mathcal{I}_2), t\rangle \vdash \textit{Aut}(A, Y, \mathcal{Q}, \mathcal{I}_2)$$

$$\forall X, Y, t \in \{\mathcal{I}_0 \cap \mathcal{I}_1\}, \mathcal{Q} \subseteq \mathcal{D}: \tag{2}$$

$$\textit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \textit{Trust}(A, X, \mathcal{D}, \mathcal{I}_1),\ \textit{Valid}\langle A, \textit{Tran}(X, Y, \mathcal{Q}, \mathcal{I}_2), t\rangle \vdash \textit{Trust}(A, Y, \mathcal{Q}, \mathcal{I}_2)$$

As with Maurer's deterministic model, for a finite set S of statements, $\overline{S}$ denotes the closure of S under applications of the inference rules (1) and (2), i.e., the set of statements derivable from S. The *evaluation time* t is the time that Alice is attempting to

reason about. Alice's *derived view at evaluation time* t is the set of statements derivable from her initial view at evaluation time t. Alice's derived view is defined by the function $\overline{\mathit{View}_A(t)}$ where $\overline{\mathit{View}_A} : t \longrightarrow \overline{S}$. Under the model, a statement s is *valid at evaluation time* t if and only if $s \in \overline{\mathit{View}_A(t)}$, and invalid otherwise.

3.2 Semantic Sugar

We explain the intuition for the definitions of the model and highlight some of the semantic difference between our model and Maurer's.

Maurer's notion of "authenticity" establishes that entity X holds the private key corresponding to the public key in X's certificate. We extend authenticity to establish that some entity X not only holds the private key corresponding to the public key in X's certificate, but also has the other properties $\mathcal{P}$, such as attributes, roles, key attributes, extended key usage, etc.

With "level," Maurer limits trust vertically (i.e., how deep trust may propagate). With "domain," we limit trust horizontally (i.e., how wide the trust may span). A trusted entity should only be allowed to vouch (either via a certificates or trust transfer statement) for properties that it is authorized to speak for. Entities may be allowed to vouch for a specific domain for a number of reasons. In many cases, the assignment of a domain to a trusted entity is done out-of-band of the PKI, and the users it a priori. For example, users almost always trust their CA to vouch for the organization's population. In our calculus, this fact is represented by the inclusion of the CA's authenticity and trust statements in every user's initial view. In other cases, the assignment of a domain to a trusted entity is done implicitly. Delegation scenarios are an example of this type of binding. Typically, if Alice trusts Bob to delegate some of his privileges to another entity, Alice would require Bob to have had the privilege in the first place. In the model, this is represented by Bob's trust statement having a subset of the properties in his authenticity statement, i.e. for $\mathcal{Q} \subseteq \mathcal{P}$:

$$\mathit{View}_A = \{\mathit{Aut}(A, B, \mathcal{P}, \mathcal{I}),\ \mathit{Trust}(A, B, \mathcal{Q}, \mathcal{I})\} .$$

Generalizing Maurer's "recommendation," our trust transfer statement can be used to model different types of transactions such as when a CA certifies a subordinate CA, or when Alice delegates some or all of her properties to Bob. Moreover, a trust transfer may be an explicit statement (such as a certificate) or an implicit statement (e.g., by activating a certificate extension such as the X.509 "basicConstraints" extension).

As discussed in Section 2, Maurer's calculus does not include checking for certificate validity as part of the calculus. Our model deals with this is through validity templates: a meta-statements whose validity checking algorithm depends on the argument type. Templates allow us to reason about different certificate formats without having to handle every format's specifics. For example, assume that $\mathit{Valid}\langle A, C, t\rangle$ is being evaluated, and C is an X.509 Identity Certificate. In order for $\mathit{Valid}\langle A, C, t\rangle$ to be true, the template instantiation should check that C's signature verifies, that C has not expired, that C has not been revoked (e.g., by having in one's belief set a properly signed, active copy of the Certificate Revocation List (CRL) to which C points), that C's key attributes allow the requested operation, that the certificate chain length has not been exceeded,

etc. If C were an X.509 Attribute Certificate, *Valid*$\langle A, C, t\rangle$ may also check that C was signed by an attribute authority. Evaluating trust transfer statements is similar.

Note that the level parameter of Maurer's deterministic model has been omitted in our model. The use of validity templates allows relying parties to directly check the properties in certificates and trust transfer statements for things like certificate chain length, delegation depth, "pathLenConstraint", etc. This way, relying parties may draw such conclusions using the context of the certificate format rather than an artificial level parameter.

3.3 An Example

To illustrate the basic concepts of our model, we extend an example from Section 3 in Maurer's paper.

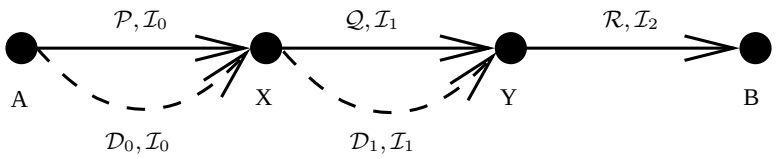

Fig. 1. A simple PKI.

Consider the PKI depicted in Figure 1. The figure indicates that Alice (A) believes that X's public key is bound to the set of properties $\mathcal{P}$ during the interval $\mathcal{I}_0$ (depicted by the solid edge from A to X). She trusts X to issue certificates over domain $\mathcal{D}_0$ (the dashed edge). (For simplicity, we set the lifespans of the trust statements to match the authenticity and certificate statements, but this is not necessary.) Her view contains a trust transfer from X to Y (the dashed edge), and two certificates (solid edges): one from X to Y which binds Y's public key to the set of properties $\mathcal{Q}$ during the interval $\mathcal{I}_1$, and one from Y to B which binds B's public key to the set of properties $\mathcal{R}$ during the interval $\mathcal{I}_2$. The trust transfer from X to Y could be an explicit statement issued by X indicating that it trusts Y to issue certificates; in practice, it is more likely to be expressed implicitly in the certificate issued from X to Y (e.g., by X setting the "basicConstraints" field of Y's X.509 certificate or the "delegation" flag of Y's SDSI/SPKI certificate).

In order for Alice to be able to believe Bob's (B) certificate (either the public key or the properties in $\mathcal{R}$) at evaluation time t, she needs to derive the statement *Aut*$(A, B, \mathcal{R}, \mathcal{I}_2)$. In this scenario, Alice's initial view is the following set of statements:

$$\mathit{View}_A = \left\{ \begin{array}{l} \mathit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, X, \mathcal{D}_0, \mathcal{I}_0),\ \mathit{Tran}(X, Y, \mathcal{D}_1, \mathcal{I}_1), \\ \mathit{Cert}(X, Y, \mathcal{Q}, \mathcal{I}_1), \quad \mathit{Cert}(Y, B, \mathcal{R}, \mathcal{I}_2) \end{array} \right\} .$$

Note that Alice's view does not contain any validity templates. Since validity templates take an evaluation time as input, they are not instantiated until evaluation time t.

Now, since Alice does not trust Y to issue certificates directly, she must derive her trust in Y using rule (2). Suppose that evaluation time $t \in \mathcal{I}_0$ and $\mathcal{D}_1 \subseteq \mathcal{D}_0$, we have:

$$\textit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \textit{Trust}(A, X, \mathcal{D}_0, \mathcal{I}_0),\ \textit{Valid}\langle A, \textit{Tran}(X, Y, \mathcal{D}_1, \mathcal{I}_1), t\rangle \vdash \textit{Trust}(A, Y, \mathcal{D}_1, \mathcal{I}_1)\ .$$

Once Alice trusts Y, she can use rule (1) to establish the authenticity of the binding expressed in Bob's certificate at evaluation time t assuming $t \in \mathcal{I}_1$, $\mathcal{Q} \subseteq \mathcal{D}_0$, and $\mathcal{R} \subseteq \mathcal{D}_1$ (in addition to our previous supposition that $t \in \mathcal{I}_0$, and $\mathcal{D}_1 \subseteq \mathcal{D}_0$):

$$\textit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \textit{Trust}(A, X, \mathcal{D}_0, \mathcal{I}_0),\ \textit{Valid}\langle A, \textit{Cert}(X, Y, \mathcal{Q}, \mathcal{I}_1), t\rangle \vdash \textit{Aut}(A, Y, \mathcal{Q}, \mathcal{I}_1)$$

$$\textit{Aut}(A, Y, \mathcal{Q}, \mathcal{I}_1),\ \textit{Trust}(A, Y, \mathcal{D}_1, \mathcal{I}_1),\ \textit{Valid}\langle A, \textit{Cert}(Y, B, \mathcal{R}, \mathcal{I}_2), t\rangle \vdash \textit{Aut}(A, B, \mathcal{R}, \mathcal{I}_2)\ .$$

Thus, Alice's derived view at evaluation time t is given by:

$$\overline{\textit{View}_A(t)} = \textit{View}_A \cup \{\textit{Trust}(A, Y, \mathcal{D}_1, \mathcal{I}_1),\ \textit{Aut}(A, Y, \mathcal{Q}, \mathcal{I}_1),\ \textit{Aut}(A, B, \mathcal{R}, \mathcal{I}_2)\}\ .$$

Alice believes that the binding between Bob's public key and his certificate properties is authentic during the time interval $\mathcal{I}_2$. Alice may stop believing this fact when Bob's certificate expires or gets revoked.

4 Using this New Model

Our motivation is to give PKI designers a tool which can be used to reason about a wide range of PKI systems. In this section, we apply the model to an array of real-world situations in order to illustrate its applicability.

4.1 Modeling Multiple Certificate Families

The new model's certificate statement binds an entity's public key to some set of properties $\mathcal{P}$ for some lifespan $\mathcal{I}$. The power of the new model stems from the fact that it is agnostic with respect to the semantics of the properties in $\mathcal{P}$, and yet still builds a calculus which allows relying parties to reason about these sets.

In standard X.509 Identity Certificate [10], the property sets may include the subject's Distinguished Name, Alternative names, name constraints, her key attributes, information about where to retrieve CRLs, and any number of domain-specific policies. The property set may also include information as to whether the subject is allowed to sign other certificates (i.e., via the "basicConstraints" field).

X.509 Attribute Certificates (ACs) [8] contain a very different set of properties than X.509 Identity Certificates. ACs typically use domain-specific properties which are used by relying parties to make authorization decisions. Some common examples of attributes include: identity, group membership, role, clearance level, etc. Other differences include the fact that an AC's subject may delegate to another party the right to

claim some of the delegator's attributes, and that ACs may not be used to form certificate chains.

An X.509-based Proxy Certificate (PC) [24] is similar to an X.509 Identity Certificate, except that PCs have a Proxy Certificate Information (PCI) extension and are signed by standard X.509 Identity Certificates. The PC standard allows any type of policy statement expressed in any language (such as eXtensible Access Control Markup Language) to be placed in the PCI. Thus, the set of properties for a PC could contain a large family of policy statements.

The SDSI/SPKI certificate format [7] takes an entirely different approach to certificates. The set of properties placed in a SDSI/SPKI certificate does not contain a global name for the subject, as SDSI/SPKI uses the public key as the subject's unique identifier. (If there is any name at all, it would be part of a linked local namespace.) Further, a SDSI/SPKI certificate contains attributes much like X.509 attributes, except they are expressed as S-expressions as opposed to ASN.1. In contrast to X.509 ACs, SDSI/SPKI certificates are allowed to be chained.

Our new model enables reasoning about all of these diverse certificate formats and semantics within one calculus. The addition of properties to the calculus allows relying parties to reason about different types of information contained in the different certificate families.

4.2 Modeling Revocation

Validity templates play an important role in our model: they allow users to reason about different *types* of signed statements. As an example, consider the case when Alice needs to make a trust decision about Bob. The instantiation of the validity template used to check Bob's certificate may require that Alice check a CRL to ensure that Bob's certificate is not included in the list of revoked certificates.

Thus, Alice's first step is to make a trust decision about a signed CRL. CRLs contain a list of revoked certificates, a lifespan (noted by the "thisUpdate" and "nextUpdate" fields), and are signed by the organization's CA. Formally, we can represent a CRL as a kind of certificate which is issued by a CA X and contains a list of revoked certificates $\mathcal{L}$, and a lifespan $\mathcal{I}$: $\mathit{Cert}(X, \emptyset, \mathcal{L}, \mathcal{I})$. Since CRLs do not contain a public key, we use the empty set notation to indicate the absence of a key. In order for Alice to use the CRL at evaluation time t, she needs to deduce that it is authentic, i.e., $\mathit{Aut}(A, \emptyset, \mathcal{L}, \mathcal{I}) \in \overline{\mathit{View}_A(t)}$.

Assuming that Alice believes that the binding expressed in CA X's certificate is authentic, and that she trusts CA X to issue certificates over domain $\mathcal{D}$, her initial view would be:

$$\mathit{View}_A = \{\mathit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, X, \mathcal{D}, \mathcal{I}_0),\ \mathit{Cert}(X, \emptyset, \mathcal{L}, \mathcal{I}_1)\}\ .$$

If X is authorized to vouch for the revocation status of all the certificates in $\mathcal{L}$ (i.e., $\mathcal{L} \subseteq \mathcal{D}$), and all of the statements are active (i.e., $t \in \mathcal{I}_0$), then Alice can deduce the CRL's authenticity by applying rule (1):

$$\mathit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, X, \mathcal{D}, \mathcal{I}_0),\ \mathit{Valid}\langle A, \mathit{Cert}(X, \emptyset, \mathcal{L}, \mathcal{I}_1), t\rangle \vdash \mathit{Aut}(A, \emptyset, \mathcal{L}, \mathcal{I}_1)\ .$$

For the CRL, the instantiation of the validity template $\mathit{Valid}\langle A, \mathit{Cert}(X, \emptyset, \mathcal{L}, \mathcal{I}_1), t\rangle$ must check that $t \in \mathcal{I}_1$, and that X's signature is verifiable. If the conditions are met, we have $\mathit{Aut}(A, \emptyset, \mathcal{L}, \mathcal{I}_1) \in \overline{\mathit{View}_A(t)}$, which indicates Alice's belief that $\mathcal{L}$ accurately represents the list of revoked certificates during the interval $\mathcal{I}_1$.

Once Alice believes the CRL, she must make a trust decision about Bob's certificate: $\mathit{Cert}(X, B, \mathcal{Q}, \mathcal{I}_2)$. Assuming that CA X can vouch for Bob's certificate information (i.e., $\mathcal{Q} \subseteq \mathcal{D}$), and all of the statements are active (i.e., $t \in \mathcal{I}_0$), then Alice can deduce Bob's authenticity by applying rule (1):

$$\mathit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, X, \mathcal{D}, \mathcal{I}_0),\ \mathit{Valid}\langle A, \mathit{Cert}(X, B, \mathcal{Q}, \mathcal{I}_2), t\rangle \vdash \mathit{Aut}(A, B, \mathcal{Q}, \mathcal{I}_2)\ .$$

In this case, the instantiation of the validity template $\mathit{Valid}\langle A, \mathit{Cert}(X, B, \mathcal{Q}, \mathcal{I}_2), t\rangle$ is being used to establish the validity of a certificate, not a CRL. As before, the template instantiation must check that $t \in \mathcal{I}_2$, and that X's signature over Bob's certificate verifies. However, in this case, the instantiation should also check that Bob's certificate has not been revoked, i.e., $\mathit{Cert}(X, B, \mathcal{Q}, \mathcal{I}_2) \notin \mathcal{L}$ as well as any other certificate information which is relevant to the requested operation.

4.3 Authorization-based Scenarios and Trust Management

In many modern distributed systems, access to some resource is granted based on authorization rather that authentication. Systems such as PERMIS [3] use the attributes contained in ACs to determine whether an entity should have access to a resource (other Trust Management systems such as KeyNote [1, 2] and PolicyMaker [17, 18] have their own certificate formats for expressing credentials). This approach simplifies the management of ACLs at the resource. For example, if Bob wants to access Alice's file, he presents his AC to Alice. Alice first decides if the AC (or set of credentials) is authentic, and if so, she examines Bob's attributes to check if he should have access (e.g., if the file is accessible to the group "developers", then Bob's attributes must state that he is a member of the group).

Maurer's deterministic model cannot handle this scenario, primarily because it cannot handle ACs. Under the deterministic model, if Alice were to deduce the authenticity of Bob's public key, she still has learned nothing about Bob (i.e., his attributes). All she has established is that the entity named Bob really has the private key corresponding to the public key found in the certificate.

There are a number of *Trust Management (TM)* languages which do handle this scenario, such as *Delegation Logic* [14] and others [16]. These TM languages can not only tell Alice that Bob has a certain set of credentials, but can also evaluate Bob's credentials and Alice's policy to determine whether Alice should allow the file access. While TM languages are typically framework-specific (i.e., KeyNote, PolicyMaker, and SDSI/SPKI have their own policy languages), there have been efforts to generalize across languages [25]. Since our model is aimed at reasoning about PKI systems, and not TM systems, such policy evaluation is outside the scope of our model's abilities. However, our model can be used to model these different certificate and credential

formats (e.g., ACs, credentials, SDSI/SPKI certificates), as well as reason about the authenticity of the core trust statements.

Under our model, Bob would first present his AC to Alice (e.g., $\mathit{Cert}(X, B, \mathcal{P}, \mathcal{I})$). Assume that Alice can then derive the authenticity of the binding between Bob's public key and the properties in the certificate—i.e., $\mathit{Aut}(A, B, \mathcal{P}, \mathcal{I}) \in \overline{\mathit{View}_A(t)}$. Since Bob's certificate is an AC, Alice needs to determine if an attribute placing Bob in the "developers" group is in the set $\mathcal{P}$. If so, then Bob is allowed to access the file.

4.4 Delegation

Some systems allow users to delegate some or all of their properties to another entity. Maurer's deterministic model allows users to issue recommendations and certificates to other entities, but this is insufficient to capture the notion of delegation. Maurer's model allows Alice to vouch for Bob, but she is limited to vouching for Bob's identity.

In our model, Alice can give some or all of her properties to Bob (possibly including identity), provided she has the properties in the first place (i.e., the she can only give Bob the properties in her domain). In the calculus, Alice would issue a certificate to Bob (i.e., $\mathit{Cert}(A, B, \mathcal{P}, \mathcal{I})$).

If a relying party Charlie has established that the binding between Alice's public key and her properties is authentic, trusts Alice to delegate, and receives a delegation from Alice to Bob, then his view will be:

$$\mathit{View}_C = \{\mathit{Aut}(C, A, \mathcal{P}, \mathcal{I}),\ \mathit{Trust}(C, A, \mathcal{P}, \mathcal{I}),\ \mathit{Cert}(A, B, \mathcal{P}, \mathcal{I})\}\ .$$

He can then derive the authenticity of the binding between Bob's public key and the delegated properties (i.e., the statement $\mathit{Aut}(A, B, \mathcal{P}, \mathcal{I})$) by applying rule (1):

$$\mathit{Aut}(C, A, \mathcal{P}, \mathcal{I}),\ \mathit{Trust}(C, A, \mathcal{P}, \mathcal{I}),\ \mathit{Valid}\langle C, \mathit{Cert}(A, B, \mathcal{P}, \mathcal{I}), t\rangle \vdash \mathit{Aut}(C, B, \mathcal{P}, \mathcal{I})\ .$$

Thus, we have $\mathit{Aut}(C, B, \mathcal{P}, \mathcal{I}) \in \overline{\mathit{View}_C(t)}$.

4.5 Modeling MyProxy

The Grid community's MyProxy credential repository [22] uses a chain of certificates for authentication. When Bob (or some process to which Bob delegates) wants to access a resource on the Grid, he generates a temporary keypair, logs on to the MyProxy server, and requests that a Proxy Certificate (PC) [24, 26] be generated which contains the public portion of the temporary keypair and some subset of Bob's privileges. The new PC is then signed with the private portion of the keypair described by Bob's long term X.509 Identity Certificate, thus forming a chain of certificates.

As Figure 2 shows, entity X is the CA which issued Bob's X.509 Identity Certificate, and T is the entity which will own the temporary keypair (possibly Bob or some other delegated entity or process). Initially, Alice believes that the binding between X's public key and properties is authentic during $\mathcal{I}_0$, and she trusts X to issue certificates and trust transfers for the domain $\mathcal{D}$. X has issued Bob a certificate binding his public key to the set of properties $\mathcal{Q}$ during $\mathcal{I}_1$. X has also issued a trust transfer to Bob,

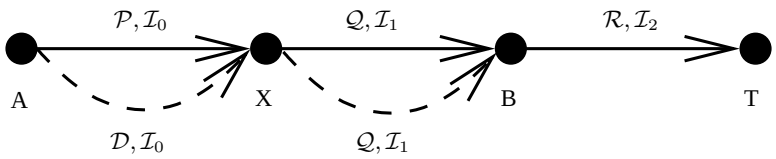

Fig. 2. The statement graph for the MyProxy system.

so that he may use his private key to sign his PC. The trust transfer is not a separate certificate in this scenario; it is an implicit statement which X makes by setting the "basicConstraints" field in Bob's X.509 Identity Certificate which allows him to sign certificates. Finally, Bob has issued a certificate (the PC) to the entity possessing the temporary keypair T for some subset $\mathcal{R}$ of his properties. The PC is valid over the interval $\mathcal{I}_2$, which in practice, is on the order of eight hours. In order for Alice to accept Bob's PC, she must derive the statement $\mathit{Aut}(A, T, \mathcal{R}, \mathcal{I}_2)$. This scenario can be reduced to the example discussed in Section 3.3.

4.6 Discovering Requirements: Greenpass

The Greenpass [9] system uses delegation to give guests inside access to a campus-wide wireless network. Further, it relies on an X.509 certificate in conjunction with SDSI/SPKI certificates to express delegation. To gain some insight as to why the designers chose this hybrid approach, we can model the problem with the calculus.

Let us assume that a relying party Alice is a member of that college, which we denote C. Let us also assume that another member of C, named Bob, has invited his colleague George from the University of Wisconsin (denoted W) to come for a visit. Bob would like to give George some guest access to the network, so that he can access some resources protected by Alice. In order for Alice to grant access to George, she must make a trust decision about George. Since there is no trust relationship between C and W (i.e., they are not cross-certified or participating in the Higher Education Bridge CA), Alice cannot simply reason about George based on statements made by George's CA. Since George is Bob's guest, Bob is in a position to vouch for George.

So, initially, Alice's view consists of her authenticity and trust beliefs about her CA, a certificate issued by her CA to Bob, and a certificate issued by George's CA to George:

$$\mathit{View}_A = \{\mathit{Aut}(A, C, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, C, \mathcal{D}, \mathcal{I}_0),\ \mathit{Cert}(C, B, \mathcal{Q}, \mathcal{I}_1), \mathit{Cert}(W, G, \mathcal{R}, \mathcal{I}_2)\}\ .$$

Since Bob has a certificate issued by a CA which Alice trusts, she can deduce the authenticity of Bob's certificate information (assuming that $t \in \mathcal{I}_0$, $\mathcal{Q} \subseteq \mathcal{D}$, and Bob's certificate is valid), i.e.,

$$\mathit{Aut}(A, C, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, C, \mathcal{D}, \mathcal{I}_0),\ \mathit{Valid}\langle A, \mathit{Cert}(C, B, \mathcal{Q}, \mathcal{I}_1), t\rangle \vdash \mathit{Aut}(A, B, \mathcal{Q}, \mathcal{I}_1)\ .$$

Now, in order for Alice to grant George access to her resources, she needs to believe the binding between George's public key and the properties in his certificate, and then that the properties grant him authorization. However, since Alice does not trust W (and has no reason to), she has no reason to trust any of the properties about George expressed in his certificate (namely, in the set of properties $\mathcal{R}$).

Since George is Bob's guest, Bob is in a position to delegate some of his privileges to George. In order for Alice to believe this delegation, Alice first needs to believe that Bob is in a position to delegate, and she then needs to believe that Bob actually delegated to George. The first condition requires the CA to transfer trust to Bob (i.e., $\mathit{Tran}(C, B, \mathcal{Q}, \mathcal{I}_1) \in \mathit{View}_A$).[1] The second condition requires that Bob issue a certificate which delegates some of his properties to George (i.e., $\mathit{Cert}(B, G, \mathcal{S}, \mathcal{I}_2) \in \mathit{View}_A$ where $\mathcal{S} \subseteq \mathcal{Q}$).

Assuming all of the preconditions are met, and the certificates and trust transfer are valid, Alice can deduce Bob's trustworthiness, and the authenticity of the certificate issued from Bob to George, i.e.,

$$\mathit{Aut}(A, C, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, C, \mathcal{D}, \mathcal{I}_0),\ \mathit{Valid}\langle A, \mathit{Tran}(C, B, \mathcal{Q}, \mathcal{I}_1), t\rangle \vdash \mathit{Trust}(A, B, \mathcal{Q}, \mathcal{I}_1)$$

$$\mathit{Aut}(A, B, \mathcal{Q}, \mathcal{I}_1),\ \mathit{Trust}(A, B, \mathcal{Q}, \mathcal{I}_1),\ \mathit{Valid}\langle A, \mathit{Cert}(B, G, \mathcal{S}, \mathcal{I}_3), t\rangle \vdash \mathit{Aut}(A, G, \mathcal{S}, \mathcal{I}_3)\ .$$

Thus Alice can reason about George because $\mathit{Aut}(A, G, \mathcal{S}, \mathcal{I}_3) \in \overline{\mathit{View}_A(t)}$.

The last question that the system designer is faced with is: "what type of certificate format should be used for the certificate issued from Bob to George?" The first consideration is that if George already has a public key, the system should reuse it. The second consideration is that Alice is not concerned with George's identity, but rather his authorization. Finally, we need to reason about what type of certificate format would allow this type of delegation scenario, and still make $\mathit{Valid}\langle A, C, t\rangle$ evaluate to true. Proxy Certificates would not allow George to have the public key of his regular certificate (i.e., $\mathit{Cert}(W, G, \mathcal{R}, \mathcal{I}_3)$) also used in his Proxy Certificate, resulting in $\mathit{Valid}\langle A, C, t\rangle$ never being true. An X.509 Attribute Certificate would not make $\mathit{Valid}\langle A, C, t\rangle$ true unless Bob was an Attribute Authority proper. This leaves us with the choice to use SDSI/SPKI certificates, which is what Greenpass implemented.

4.7 Time Travel

There may be times when a relying party would like to reason about an event that has past or one that has not happened yet (because some statements are not yet active). Maurer's model lacks of the concept of time. In the new model, we can reason about such events by manipulating the evaluation time t.

[1] In the Greenpass prototype, this trust transfer is expressed as a SDSI/SPKI certificate issued to Bob's public key and allowing him to delegate. It could also have been implicit, by setting the "basicConstraints" field of Bob's X.509 certificate, but this would require reissuing Bob's certificate.

For example, assume that a relying party Alice is trying to verify a signature that Bob generated on April 21, 1984. (Note that Alice would need a mechanism such as a timestamping service in order to know that the signature existed on April 21, 1984). Further, assume that Alice believed that the CA X had an authentic binding between its public key and certificate information, and that it trusted the CA during that time period. Last, Alice would have to possess a certificate for Bob's which was valid during that period.

More formally, let $\mathcal{I}_0$ be the time period from January 1, 1984 to December 31, 1984. Let $\mathcal{I}_1$ be the time period from April 1, 1984 to April 30, 1984. Finally, let $\mathcal{Q} \subseteq \mathcal{D}$. Alice's initial view is given by:

$$\mathit{View}_A = \{\mathit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, X, \mathcal{D}, \mathcal{I}_0),\ \mathit{Cert}(X, B, \mathcal{Q}, \mathcal{I}_1)\}\ .$$

Now, at evaluation time t where $t \in \{\mathcal{I}_0 \cap \mathcal{I}_1\}$ (i.e., t is some time in April, 1984), Alice can use rule (1) to derive the authenticity of the binding between Bob's public key and his certificate information:

$$\mathit{Aut}(A, X, \mathcal{P}, \mathcal{I}_0),\ \mathit{Trust}(A, X, \mathcal{P}, \mathcal{I}_0),\ \mathit{Valid}\langle A, \mathit{Cert}(X, B, \mathcal{Q}, \mathcal{I}_1), t\rangle \vdash \mathit{Aut}(A, B, \mathcal{Q}, \mathcal{I}_1)\ .$$

Thus, Alice can use Bob's public key to verify the signature (she could also use any other of Bob's properties in the set $\mathcal{Q}$) because $\mathit{Aut}(A, B, \mathcal{Q}, \mathcal{I}_1) \in \overline{\mathit{View}_A(t)}$ when t is some time in April, 1984. If she were to try and derive the same statement in May of 1984, she would fail because Bob's certificate expired after April, 1984. Since the evaluation time t would be in May, 1984, we have $t \notin \{\mathcal{I}_0 \cap \mathcal{I}_1\}$, and the validity template instantiation would fail. Thus, $\mathit{Aut}(A, B, \mathcal{Q}, \mathcal{I}_1) \notin \overline{\mathit{View}_A(t)}$.

4.8 Comparison

PKI System	Maurer's model	Our model	Enabling feature
Multiple formats	no	yes	properties
Revocation	no	yes	time
Authorization	no	yes	properties
Delegation	some	yes	domains
MyProxy	some	yes	time, domains
Greenpass	no	yes	properties, domains
Time travel	no	yes	time

Table 1. A comparison of the Maurer's model and ours.

Table 1 shows how our model and Maurer's model handle the systems discussed in this section. Since Maurer's model relies on the use of names instead of properties, his model cannot be used to reason about certificate formats which do not use names

(such as SDSI/SPKI and X.509 Attribute Certificates). With no notion of time, Maurer's model cannot handle revocation, time travel, and the MyProxy system which relies on short-lived Proxy Certificates. Finally, our concept of domain permits delegation scenarios where a subset of the delegator's privileges are given to the delegatee. This level of granularity is necessary for most real world systems, such as MyProxy and Greenpass.

5 Conclusions and Future Work

While our new model makes it possible to reason about a number of different types of PKIs and has been useful in practice, it is not perfect. There are a number of interesting potential future directions.

First, our model does not describe how well the properties in the certificates match the real world properties of certificate's subject. A similar issue arises in the field of program verification. One might determine how well the program fits the specification. However, this does not answer the question "Is my specification any good?" Such approaches yield a program which is at most as correct as the specification. In determining authenticity of a binding between a set of properties and a public key, the relying party trusts the attributes at most as much as it trusts the issuer. If an issuer is careless (or malicious), and binds false properties to Bob's public key then, under the new model (and in the real world), Alice will accept false properties about Bob. As an alternative view, we might consider whether the building blocks of a particular certificate scheme are in fact sufficiently *articulate* for relying parties to make the correct decision, or consider the size of the fraction of the space where relying parties make the wrong decisions. Further investigation into this issue, perhaps including automated formal methods, is an area for future work.

Second, the inclusion of time in the new model makes it *nonmonotonic*: true statements can become false over time. Nonmonotonicity can have a fatal side effect: a relying party may deduce authenticity when it should not. Some statement may have expired or been revoked, and the relying party has not received the revocation information yet. Li and Feigenbaum [15] introduce a concept of "fresh time" which could be used either in the certificate's properties, or possibly as an explicit parameter to make the system monotonic. Using fresh times in our model is another area for future work.

Last, certification and trust transfer statements in our new model are similar to Jon Howell's "speaks-for-regarding" operator [11]. However, our statements go beyond Howell's because they are applicable to a number of certificate formats (not just SDSI/SPKI), and they allow cases where transfers of trust are expressed implicitly (e.g., via the X.509 "basicConstraints"). If a relying party Alice receives multiple certificates about Bob, and she successfully deduces their authenticity (i.e., the authenticity of the bindings they contain), then Alice may hold multiple sets of properties assigned to Bob. What kind of set operations should we allow on these sets of properties? Howell disallows the relying party to use the union operation, but allows intersection. Considering the universe of allowable set operations is another area for future work.

In sum, we briefly reviewed Maurer's calculus for reasoning about PKI systems, and illustrated its limitations. We then introduced a new model which generalized and

extended Maurer's calculus to handle many real-world PKI concepts, such as the notion of authentic bindings, the consideration of certificate information, and the concept of time. Next, we used the new model to illustrate how it can be used to reason about real-world PKI systems that we have seen in the wild as well as in our lab. Finally, we discussed some of the model's limitations and directions for future work.

Acknowledgements

This work was supported in part by the Mellon Foundation, by the NSF (CCR-0209144), by Internet2/AT&T, by Sun, by Cisco, by Intel, and by the Office for Domestic Preparedness, U.S. Dept of Homeland Security (2000-DT-CX-K001). The views and conclusions do not necessarily represent those of the sponsors.

References

1. M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The KeyNote Trust-Management System, version 2. IETF RFC 2704, September 1999.
2. M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. The Role of Trust Management in Distributed Systems. *Secure Internet Programming*, 1603:185–210, 1999. Lecture Notes in Computer Science.
3. D. Chadwick, A. Otenko, and E. Ball. Role-Based Access Control with X.509 Attribute Certificates. *IEEE Internet Computing*, March-April 2003.
4. D. Cooper. A Model of Certificate Revocation. In *15th Annual Computer Security Applications Conference (ACSAC '99)*, pages 256–264, Phoenix, Arizona, USA, December 1999. IEEE Computer Society.
5. C. Ellison. The nature of a usable pki. *Computer Networks*, pages 823–830, 1999.
6. C. Ellison. Improvements on Conventional PKI Wisdom. In *Proceedings of the 1st Annual PKI Research Workshop*, April 2002.
7. C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. SPKI Certificate Theory. IETF RFC 2693, September 1999.
8. S. Farrell and R. Housley. An Internet Attribute Certificate Profile for Authorization. IETF RFC 3281, April 2002.
9. N. Goffee, S. Kim, S.W. Smith, P. Taylor, M. Zhao, and J. Marchesini. Greenpass: Decentralized, PKI-based Authorization for Wireless LANs. In *3rd Annual PKI Research and Development Workshop*. NIST, April 2004.
10. R. Housley, W. Polk, W. Ford, and D. Solo. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. IETF RFC 3280, April 2002.
11. J. Howell and D. Kotz. A formal semantics for SPKI. In *Proceedings of the Sixth European Symposium on Research in Computer Security (ESORICS 2000)*, volume 1895 of *Lecture Notes in Computer Science*, pages 140–158. Springer-Verlag, October 2000.
12. P. Kocher. On Certificate Revocation and Validation. In *Proceedings of the Second International Conference on Financia l Cryptography (FC'98)*, volume 1465 of *LNCS*, pages 172–177. Springer-Verlag, 1998.
13. R. Kohlas and U. Maurer. Reasoning About Public-Key Certification: On Bindings Between Entities and Public Keys. *Journal on Selected Areas in Communications*, pages 551–560, 2000.

17

14. N. Li, B. Grosof, and J. Feigenbaum. Delegation logic: A logic-based approach to distributed authorization. *ACM Transactions on Information and System Security (TISSEC)*, 6(1):128–171, February 2003.
15. N. Li and Feigenbaum J. Nonmonotonicity, User Interfaces, and Risk Assessment in Certificate Revocation. In *Proceedings of the 5th International Conference on Financial Cryptography*, pages 166–177. Springer-Verlag, 2002.
16. N. Li, W. Winsborough, and J. Mitchell. Beyond Proof-of-compliance: Safety and Availability Analysis in Trust Management. In *Proceedings of 2003 IEEE Symposium on Security and Privacy,*, pages 123–139. IEEE Computer Society Press, May 2003.
17. M. Blaze and J. Feigenbaum and J. Lacy. Decentralized trust management. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 164–173. IEEE Computer Society Press, May 1996.
18. M. Blaze and J. Feigenbaum and M. Strauss. Compliance-checking in the PolicyMaker Trust Management System. In *Proceedings of Second International Conference on Financial Cryptography (FC '98)*, volume 1465, pages 254–274, 1998.
19. A. Malpani, S. Galperin, M. Mayers, R. Ankney, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol. RFC2560, `http://www.ietf.org/rfc/rfc2560.txt`, June 1999.
20. U. Maurer. Modelling a Public-Key Infrastructure. In *ESORICS*. Springer-Verlag LNCS, 1996.
21. M. Naor and K. Nissim. Certificate Revocation and Certificate Update. *IEEE Journal on Selected Areas in Communications*, 18(4):561–570, April 2000.
22. J. Novotny, S. Tueke, and V. Welch. An Online Credential Repository for the Grid: MyProxy. In *Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10)*. IEEE Press, August 2001.
23. S.W. Smith. Outbound Authentication for Programmable Secure Coprocessors. *International Journal on Information Security*, 2004.
24. S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Internet X.509 Public Key Infrastructure Proxy Certificate Profile. `http://www.ietf.org/internet-drafts/draft-ietf-pkix-proxy-10.txt`, 2003.
25. S. Weeks. Understanding Trust Management Systems. In *Proceedings of 2001 IEEE Symposium on Security and Privacy*, pages 94–105. IEEE Computer Society Press, May 2001.
26. V. Welch, I. Foster, C. Kesselman, O. Mulmo, L. Pearlman, S. Tuecke, J. Gawor, S. Meder, and F. Siebenlist. X.509 Proxy Certificates for Dynamic Delegation. In *3rd Annual PKI Research and Development Workshop*, April 2004.