Implementation of Trust Metrics in X.509 Public Key Infrastructure

Lucas Gonçalves Martins and Ricardo Felipe Custódio
Departamento de Informática e Estatística
Universidade Federal de Santa Catarina
Santa Catarina, Brazil
Email: {lucasgm,custodio}@inf.ufsc.br

***Abstract*—The *X.509* hierarchical public key infrastructure model is used to distribute trust and to decentralize the responsibility of managing digital certificates among certification authorities. However, the trust indiscriminately flows through these certification authorities, allowing any of them to issue trusted certificates. Many works propose trust quantification and calculation as a solution for this problem, but most of them apply their proposed methods in hypothetical public key infrastructure networks. In this paper, we propose a plausible implementation of quantification and calculation of trust for the *X.509* public key infrastructure, specifying *ASN.1* structures and trust management procedures to initialize and update trust values in this model's relationships.**

Keywords*—PKI; Trust; Certificate; Metrics; Calculation; X.509; ASN.1; Hierarchy.

I. Introduction

Nowadays, the *X.509* standards are the most used Public Key Infrastructure (PKI) model. However, this model has been subject of concern about its security and usability [1], [2]. Nevertheless, the usage of digital certificates, as a method for identification and authentication in the digital environment, has been growing over the years. With this growth, there have also been an increasing of attempts to obtain fraudulent digital certificates to impersonate big companies on the Internet [3], [4], [5].

Most of the attacks are done to Certification Authorities (CA) at the end of the PKI hierarchy, which usually use semi-automatic on-line applications to manage their certificates. The attackers use a combination of social engineering along with the exploitation of these application flaws to obtain valid certificates. The only way to avoid the attacks is to guarantee the security of the technology stack used by all CA's applications, and to take the human factor off from its sensitive procedures; a virtually impossible task, considering the number of trusted CAs (in the most popular repositories) and the diversity of applications used by them. Since we cannot guarantee the security of all CA applications, we need to focus on the reason why certification authorities are chosen by hackers.

The *X.509* PKI trust model follows a hierarchical structure to distribute trust among PKI entities and decentralize the responsibility of managing digital certificates. Also, if a node of the hierarchy is compromised, only its adjacent nodes will be compromised, too. This characteristic makes the attacks on lower levels of the PKI less harmful to the whole PKI.

However, when a CA delegates a services to another CA, it also gives its trust to that CA. The trustworthiness is transitive in the *X.509* PKI [6] and all certificates in the hierarchy have the same trust value. As a result, all delegated CAs are as trustworthy as the root CA, but most of them are not as secure as it is. This allows an attacker to attack the entities at the bottom of the hierarchy to obtain a certificate as trustworthy as a certificate issued by the entity at the top of it. Burmester [2] also wrote about this issue: *"The problem with X509 is that it cannot tolerate even one penetration: each node [in the hierarchy] is a single point of failure."*

This flaw leads to another, which allows the attacker to profit from the stolen certificate. The entities in a hierarchy should be organized through a measurable organization criteria, which defines in which level of the hierarchy the entity should be placed. However, the *X.509* hierarchy does not have a well-defined organization criteria, allowing any certificate to be positioned below any CA, regardless of the importance of the certificate owner. For example, when a certificate is issued to identify a company, the company will be considered inferior to the CA, which is not necessarily true. Thereby, an attacker can attack a less important and, probably, more insecure company (the CA) to jeopardize a bigger one.

Several works propose the usage of trust quantification and calculation as a solution for these problems. Through the measurement of trust, it is possible for a certificate verifier to decide if the calculated trust level of a certificate is enough for the context in which he is using it. However, most of these works apply their method to hypothetical web PKIs. In this paper, we propose a plausible implementation of trust quantification and calculation for *X.509* PKI. We specify *ASN.1* structures to represent trust values, as well as trust management procedures for the initialization and update of these values. The model is specified to be independent of the trust calculation method. However, we use Jøsang's trust model [7] to interpret the trust relationships, and use his quantification and calculation methods as an example of how to measure trust in a PKI [8].

Through our proposal, we want to bring all the benefits of trust metrics to the *X.509* PKI. With trust metrics we can build a stronger PKI against attacks and introduce semantics in the certificate verification procedures, that may help end-users to decide if he should trust in a specific digital certificate or not. We also tried to make the trust metrics implementation the less impacting possible to the *X.509* standards, keeping it compatible to the existent PKI-enabled applications.

The remainder of this paper is organized as follows. In Section II, we present the related works found in our research. In Section III, we present the Jøsang definitions of trust, as well as his notations for trust networks and his method to calculate trust. Also in this section, we specify a method of interpreting a PKI as a trust network. In Section IV, we propose the procedures to initialize and update the trust values in the *X.509* PKI. In Section V, we show how our method can be used to solve the problems discussed in Section I. Finally, in Section VI, we give our final considerations and future works.

II. Related Work

In this section, we present the related works found in our research. First, we give a brief overview about the studies directly associated with trust models and the generic view of trust. Then, we discuss PKI trust models and the application of trust metrics to improve them.

A. Trust Concepts and Models

Trust modeling is a topic easily found in the literature. Several works study the semantics of trust and its transitivity, with a generic view about the subject [9]. Other works defined notations for the specification of trust networks that can be used to evaluate and measure trust [7][10][11][12][13]. Among these studies we highlight the following.

Ruohomaa [9] presented the concepts of trust and its applications in computer science. She defines trust as the extent to which one party is willing to participate in a given action with a given partner, considering the risks and incentives involved. She also discusses trust management life-cycle, which she defines in three steps: determining initial trust, observing the trustee's actual behavior and updating trust accordingly. We use this definition in our work to propose the trust management in the *X.509* PKI.

Jøsang [7][10] is an active researcher in the trust model field. He published several works about basic trust concepts, proposed notations for trust network specification and applied subjective logic with belief calculus to measure trust in trust networks. His notations and trust calculus are better explained in Section III.

Trust metrics have a visible application in PKI models, such that most of the work in this field uses PKI as a practical example for their proposals. However, they usually give a superficial view of the problem, and do not define real solutions for a hierarchical PKI. In the next sections, we present the studies of PKI trust models and the existent proposal that applies trust metrics in PKI.

B. PKI Trust Models

The *X.509* PKI is specified by several documents called *Request For Comments* (RFC), which are maintained by the *Internet Engineering Task Force* (IETF) [14]. The RFC 5280 - *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile* - specifies the data structures of digital certificates and CRL, as well as the interpretation of these structures and an algorithm for certification path validation [15]. We use these specifications to understand the semantics of trust in *X.509* and how the trust flows through the PKI entities.

There are several works in the literature that specify different trust models for PKI [16][17][18][19]. However, most of them address structural characteristics and procedures for certification path validation, with a superficial approach to the concepts and semantics of trust. As the proposal of this paper does not change the *X.509* structure, we focus on the works that apply trust metrics in PKI [20][21][8][22].

Maurer [20] proposed in his work a deterministic PKI model, based on recommendations and confidence levels. This model specifies trust relationships as predicates of authenticity, trust and recommendation. He also defined inference rules to be applied to an initial set of predicates, generating new predicates of trust and authenticity (these inferences are similar to certification path validation that uses path size limitations). Based on the deterministic model, he proposed a probabilistic model, which uses probabilistic logic to measure the confidence of his predicates. Through these values, he calculates the resultant trust of predicates generated by inferences.

Jøsang [8] uses his trust model to interpret public key infrastructures and measure their trust relationships. He also specifies an algebra to calculate trust transitivity in the PKI, which we discuss in Section V of this paper. Levien [22], by contrast, addresses the efficiency of PKI trust models that use trust metrics to resist to attacks. He proposes an attack model to be used as a framework to calculate the index of attack-resistance efficiency. He also demonstrates that the closest attacks to the certificate verifier have more chances of success.

III. Specifying Trust Networks

In this section, we present Jøsang's [7][10] notation for trust network specification, and his proposed methods to measure trust and calculate its transitivity. We use these tools to define a certificate interpretation method that allows us to build a trust network based on PKI hierarchy.

Jøsang uses Gambetta's definition of trust, which he calls *reliability trust*. He represents the reliability trust as a tuple (A, B, P, μ, τ), that can be interpreted as: A trusts B for the purpose P with the measure value μ in the specific time τ. A and B are principals, or nodes, in the trust network. P is the purpose of the trust (e.g., "being a good mechanic"). The kind of the measurement μ is arbitrary, and τ is any representation of time. In our examples, we may use a simplified notation, making μ and τ implicit, as in (A, B, P).

> **Definition 1 (Reliability Trust)** *"Trust is the subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends [23]."*

The purpose P can be categorized by prefixes in its identifier, as xyP. The category x identifies if the trust is direct (d) or indirect (i), while the category y identifies if the trust is functional (f) or for referral (r). A functional trust means that A trusts in B for the purpose P (e.g., A trusts B to be a good mechanic), while a referral trust means that A trust in who B recommends for the purpose P (e.g., A trusts B to recommend a good mechanic). The direct trust is used when

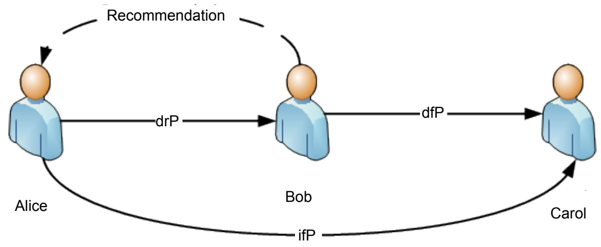

Fig. 1. Trust Network

A direct trusts B, and the indirect trust is used when A trusts B because of a recommendation of an entity X that A trusts. These modifiers generate four kinds of purpose: dfP, drP, ifP and irP.

Figure 1 shows a graph that represents a simple trust network. Considering the node Alice as A, Bob as B and Carol as C, and using ":" as a transitive connection between two consecutive trust edges of the graph, we can specify this network as follow:

$$\begin{aligned}&(A, C, ifP, \mu_1, \tau_1)\\&\quad = (A, B, drP, \mu_2, \tau_2) : (B, C, dfP, \mu_3, \tau_3)\end{aligned} \tag{1}$$

To calculate the measurement μ_1 based on μ_2 and μ_3, Jøsang proposed the usage of a belief metric called opinion, explained below:

> *"Subjective logic represents a specific belief calculus that uses a belief metric called opinion to express beliefs. An opinion denoted by $\omega_x^A = (b_x^A, d_x^A, u_x^A, a_x^A)$ expresses the relying party A's belief in the truth of statement x. Here b, d, and u represent belief, disbelief and uncertainty, and relative atomicity respectively where $b_x^A, d_x^A, u_x^A, a_x^A \in [0, 1]$ and the following equation holds:*
>
> $$b_x^A + d_x^A + u_x^A = 1 \tag{2}$$
>
> *The parameter a_x^A reflects the size of the state space from which the statement x is taken. In most cases the state space is binary, in which case $a_x^A = 0.5$".*

Assume that the values of μ_2 and μ_3 are $\omega_B^A = (b_B^A, d_B^A, u_B^A, a_B^A)$ and $\omega_x^B = (b_x^B, d_x^B, u_x^B, a_x^B)$, respectively. We use the discount ($\otimes$) operator, from subjective logic, to calculate the transitive opinion $\omega_x^{A:B} = (b_x^{A:B}, d_x^{A:B}, u_x^{A:B}, a_x^{A:B})$, using the definitions below:

$$\omega_B^A \otimes \omega_x^B = \omega_x^{A:B} = \begin{cases} b_x^{A:B} = b_B^A b_x^B \\ d_x^{A:B} = b_B^A d_x^B \\ u_x^{A:B} = d_B^A + u_B^A + b_B^A u_x^B \\ a_x^{A:B} = a_x^B \end{cases} \tag{3}$$

We also define the conjunction ($\wedge$) operator, from subjective logic, to calculate the trust of an entity in two different statements. Assume the following opinions $\omega_x^A = (b_x^A, d_x^A, u_x^A, a_x^A)$ and $\omega_y^A = (b_y^A, d_y^A, u_y^A, a_y^A)$. To calculate the resultant conjunction $\omega_x^A \wedge \omega_y^A$, we use the following definition:

$$\omega_x^A \wedge \omega_y^A = \omega_{x\wedge y}^A = \begin{cases} b_{x\wedge y}^A = b_x^A b_y^A \\ d_{x\wedge y}^A = d_x^A + d_y^A - d_x^A d_y^A \\ u_{x\wedge y}^A = b_x^A u_y^A + u_x^A b_y^A + u_x^A u_y^A \\ a_{x\wedge y}^A = a_x^A a_y^A \end{cases} \tag{4}$$

Now, we will demonstrate how to specify an *X.509* PKI in the Jøsang's notation. We use the PKI represented by the directed graph in Figure 2, as an example. We simplify a certificate as a tuple $C = (X, Y, k, p, s)$, where X is the certificate issuer, Y the certificate subject, k a public key, p a certificate policy, and s a signature done by X over these data. To interpret the certificate as a trust relationship, we read this tuple as follows: (a) X trusts that Y is responsible for the public key k; (b) X trusts that Y follows the certificate policy p; and (c) s proves the authenticity of X's trust in (a) and (b).

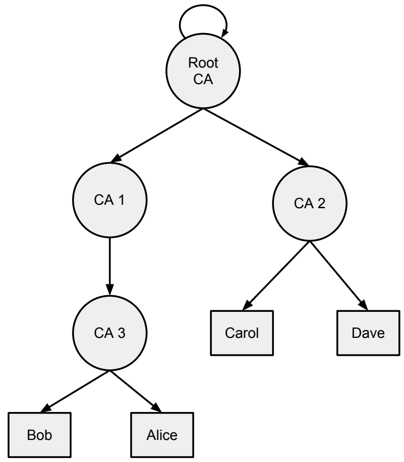

Fig. 2. X.509 PKI Hieararchy Example

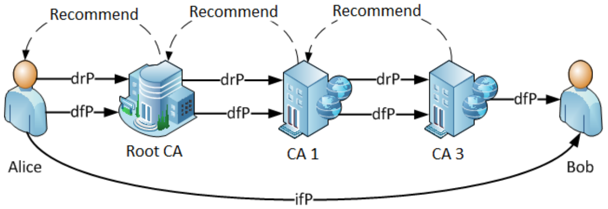

Fig. 3. Certification Path's Trust Network

We interpret (a) as a direct functional trust, specified as (X, Y, dfP), where the purpose P is "being responsible for the public key k". The interpretation of (b) is more complicated. We could interpret it as a functional trust with the purpose of "following the certificate policy p"; however, it would be different from the purpose in (i). As the purposes need to be the same for the transitivity to be possible [7], we interpret (ii) as follow: if X trusts that Y follows its certificate policy, and Y is a certification authority, X trusts that Y correctly issues certificates. In other words, X trusts in the certificates that Y recommends. This means that X has a direct referral trust in Y, specified as (X, Y, drP), where the purpose P is "recommend some entity as responsible for the public key k".

To give an example of how to interpret a certification path as a trust network, we assume that Alice (A) wants to verify the authenticity of Bob's certificate (B), using the PKI in figure 2. Following the certification path validation procedure [15], we build the certification path shown in table I, where the *Root CA* is R, CA 1 is I and CA 3 is F.

TABLE I: CERTIFICATION PATH

C^R_R	$=$	(R, R, k_r, p_r, s_r)
C^R_I	$=$	(R, I, k_i, p_i, s_r)
C^I_F	$=$	(I, F, k_f, p_f, s_i)
C^F_B	$=$	(F, B, k_b, p_b, s_f)

Alice's trust in the *Root CA* has the same meaning of the trust in a certificate, except for its signature (Alice does not need proof of her trust relationships). So, we interpret her trust in the root CA as (A, R, dfP) and (A, R, drP). Interpreting the certificates in the certification path, we build the trust network illustrated in Figure 3. Following Jøsang's proposal [8], we calculate the resulting trust transitivity value as shown bellow, the identifier k being for functional trust and the identifier p for referral trust.

$$\omega^{A:F}_{B_k} = (\omega^{A}_{R_k} \wedge \omega^{A}_{R_p}) \otimes (\omega^{R}_{I_k} \wedge \omega^{R}_{I_p}) \otimes (\omega^{I}_{F_k} \wedge \omega^{I}_{F_p}) \otimes \omega^{F}_{B_k} \tag{5}$$

IV. Trust Management in *X.509*

In this section, we demonstrate how to implement trust quantification and calculation in the *X.509* PKI model. We define *ASN.1* structures that represent trust values, to be used in the existent structures of the *X.509* model. We also define three kinds of trust in the *X.509* PKI: authentication trust, *policy trust* and *PKI trust*. For each of these, we describe procedures that cover the life-cycle of trust management, defined by Ruohomaa: (i) determining initial trust; (ii) observing the trustee's actual behavior and; (iii) updating trust accordingly [9]. The correct semantic interpretation of trust measures is important because trust has a strong context-dependence, and any misinterpretation can lead to a different sample space [11].

```
Trust ::= CHOICE {
   opinon   Opinion}
```

Fig. 4. *ASN.1 Trust* structure

```
Opinion ::= SEQUENCE {
   belief      INTEGER (0..100),
   disbelief   INTEGER (0..100),
   uncertain   INTEGER (0..100)}
```

Fig. 5. *ASN.1 Opinion* structure

To represent a generic type of trust measurement, we define a *Trust ASN.1* structure of type *CHOICE*, which is composed by a set of trust measurement methods. We also define another *ASN.1* structure that represents the opinion based trust measurement. This structure name is *Opinion* and it is a *SEQUENCE* of *INTEGER* values, identified as *belief*, *disbelief* and *uncertain*. Each of these values can be set in the integer interval $[0, 100]$, to represent the percentage of belief, disbelief and uncertainty, from the belief calculus of subjective logic. We show the *ASN.1* specification of these structures in Figures 4 and 5.

A. Authenticity Trust

We call the direct functional trust *authenticity trust*, because it is formed when a certification authority issues a certificate, creating proofs for the bind between the certificate's subject and its public key. In other words, the CA trust in the certificate authenticity. To define the initial trust in this kind of trust relationship, we need to answer the following question: How much does a CA need to believe in the bind between an entity and its public key to issue a certificate for it?

Considering that the certification authority is responsible for the identification and authentication of certificate applicants, the value of CA's trust over the bind is equivalent to the trust over their own procedures. Therefore, in our interpretation, the answer for the question is 100% belief, because the CA can't have doubts about their own procedure (this scenario changes when the CA uses an RA, because the CA have a direct referral trust on the RA). Hence, we use binary value of trust, that in the opinion metrics is expressed as $(1.0, 0.0, 0.0)$ for all issued and not revoked certificates; $(0.0, 1.0, 0.0)$ for all revoked certificates; and $(0.0, 0.0, 1.0)$ for all expired and not issued certificates.

These quantification rules do not change the already used procedures for certificate issuance and revocation (initialization and update), neither do they need new *ASN.1* structures to represent them. During the certificate validation, the values are assumed according to the certificate status (valid, revoked, expired). Moreover, these trust value constraints prevents an attack over a final certification authority to create certificates with differentiated values, which would give to the attacker the power to manipulate the final trust value of the certificate.

If the PKI wants to define different values for the *authenticity trust*, it can use a certificate extension that includes a *Trust* structure as an extension value. However, the CA also needs to specify an update method for this kind of trust, which the certificate owner needs to know how to use. In the next section, we specify some methods for this purpose, however they are specified for certification authorities and may be complicated for an end user to understand. So, we discourage its use in end-user certificates.

B. Policy Trust

The *policy trust* refers to the direct referral trust relationship that is formed when a CA issues a digital certificate for another CA, trusting that the CA will follow its certificate policies. This trust relationship is different from the *authenticity trust*, because an entity needs to trust in another entity behavior, which we interpret as expectation. Therefore, we measure this kind of trust as follows: the trustor CA defines a belief value, b, that the trustee CA will follow its certificate policies. The disbelief value has to be zero, $d = 0$, as the trustor CA should not issue a certificate if it has any belief that the trustee CA will not follow the policies. Thus, if $d = 0$ and $b+d+u = 1$, then $u = 1-b$, reflecting the uncertain value (not disbelief) about the trustee CA following the policies. Thereby, we define the *policy trust* values as: $(b, 0, 1 - b)$.

The *policy trust* initialization happens when a certification authority issues a certificate for another CA. As the values may be different for each CA certificate, the issuer CA needs to use the proposed *Trust ASN.1* structure to bind the trust value to the issued certificate. As the value represents the trust in the CA following a certificate policy, we extend the *Qualifier ASN.1* structure, which is used to define a *PolicyQualifier* for a policy in the certificate policies extension, as defined by the RFC 5280 [15]. We include the *Trust* structure as a possible *Qualifier* choice, as shown in Figure 6. With this structure, the issuer CA can define a trust value as a *PolicyQualifier* of a certificate policy in the certificate. Thereby, the CA may also define different values for each of the policies followed by the subject CA.

```
Qualifier ::= CHOICE {
   cPSuri        CPSuri,
   userNotice    UserNotice,
   trustValue    Trust}
```

Fig. 6. *ASN.1 Qualifier* structure

Updating the *policy trust* is a big challenge. The initial trust value is set in the certificate structure before being issued. As the certificate is an immutable structure, the trust updates need to be done in other ways. The first and simpler way is through a certificate revocation list (CRL). We could expand the CRL usage to support *policy trust* updates. This way, it is possible to recovery the newest trust values from the latest issued CRLs. However, CRLs are considered one of the biggest problems in the *X.509* PKI, and encouraging the usage of CRLs for other purposes, beyond certificate revocation, is not a good practice.

The second way to update the *policy trust* is through recommendations. Recommendation is defined in Maurer's work as a mechanism for a supposed entity X prove his trustworthiness in another entity Y. However, the recommendation does not need to give proofs of Y's authenticity [20]. We can represent this relationship in Jøsang's model as a direct referral trust: (X, Y, drP). We can implement a recommendation as an attribute certificate that defines a *PolicyInformation* as an attribute. The truster CA can issue attribute certificates for all its trustee CAs, defining a *PolicyInformation* attribute with a *Trust Qualifier*, to be used as a *policy trust* update. Thereby, the trustee CAs become responsible for distributing, with its certificates, its newest recommendations.

This brings us to the third way of *policy trust* update, the time. With a decay function, we can reduce the trust value as time passes. This calculation can be done during the certificate path validation. This function may be useful to force the certification authorities to update and distribute their newest recommendations, to ensure that their confidence levels will always be high.

C. *PKI Trust*

The last kind of trust is the *PKI trust*, which is formed when an end-user trusts in a trust anchor. This kind of trust is composed by the *authenticity* and *policy trust*, which follow the same value rules defined to these kinds of trust. However, their management is different because the end-user does not issue a certificate to the trust anchor. The definition of the trust values need to be done through an out-of-bands method. To discuss how to do this management, we consider two different scenarios: in the first, Alice, an end-user, relies in just one trust anchor; and in the second, Alice relies in a set of trust anchors.

In the first scenario, any value defined by Alice to her relationship with the trust anchor will equally affect all certificates in the PKI. So the trust value of this relationship does not create any evidence that can be used in Alice's decision-making. Therefore, we define the *PKI trust* value, in this scenario, as $(1, 0, 0)$, which represents the *policy trust* of Alice on the trust anchor. This value does not reduce the resultant value of the transitivity trust from the trust anchor to the certificates below. As a fixed value, Alice does not need to manage it. In the certification path validation, the value is assumed when the trust anchor is defined.

By contrast, in the second scenario, the values defined by Alice for each trust anchor will affect only the certificates under the respective trust anchor. So, Alice might use these different values to help in her decision-making. However, Alice has to initialize and update her trust values for every trust anchor that she trusts, what may be a hard task for a end-user. But, as it is already done nowadays, a relying party can be responsible for this management, setting and updating the trust values in a certificate repository that Alice fully trusts.

V. Analysis

In this section, we present an example of using calculation of trust in a *X.509* PKI, following our proposed trust management. All our calculations are based on the work of Jøsang about trust algebra in PKI [8] and in the arbitrary values set for the trust relationships in the PKI illustrated in Figure 2, represented in Table II. Using the table values, we calculate the trust value through the following formula (The calculations were made with the calculator available at [24]:

$$\omega^{A:F}_{B_k} = (\omega^{A}_{R_k} \wedge \omega^{A}_{R_p}) \otimes (\omega^{R}_{I_k} \wedge \omega^{R}_{I_p}) \otimes (\omega^{I}_{F_k} \wedge \omega^{I}_{F_p}) \otimes \omega^{F}_{B_k} = (0.81, 0.00, 0.19)$$

To evaluate the behavior of the trust calculation when a certificate is revoked, we use the update values, ω'^{F}_{B} e ω'^{I}_{F}, with value $(0, 1, 0)$, that represents the Bob's and CA-3's certificate revocation, respectively. the calculus is given below:

$$\omega^{A:F}_{B_k} = (\omega^{A}_{R_k} \wedge \omega^{A}_{R_p}) \otimes (\omega^{R}_{I_k} \wedge \omega^{R}_{I_p}) \otimes (\omega^{I}_{F_k} \wedge \omega^{I}_{F_p}) \otimes \omega'^{F}_{B_k} = (0.00, 0.81, 0.19)$$

$$\omega^{A:F}_{B_k} = (\omega^{A}_{R_k} \wedge \omega^{A}_{R_p}) \otimes (\omega^{R}_{I_k} \wedge \omega^{R}_{I_p}) \otimes (\omega'^{I}_{F_k} \wedge \omega^{I}_{F_p}) \otimes \omega'^{F}_{B_k} = (0.00, 0.00, 1.00)$$

As can be seen, when the *CA 3* revokes Bob's certificate, the result value of belief in the trust transitivity calculation changes to disbelief, while the value of uncertainty remains the same. These values are correct, because we set a value of uncertainty for *CA 3*, which may have incorrectly revoked Bob's certificate. The second scenario shows that when the *CA*

TABLE II: ARBITRARY *POLICY TRUST* VALUES

$$\omega^{A}_{R_p} = (1.00,\ 0.00,\ 0.00)$$
$$\omega^{R}_{I_p} = (0.90,\ 0.00,\ 0.10)$$
$$\omega^{R}_{O_p} = (0.90,\ 0.00,\ 0.10)$$
$$\omega^{I}_{F_p} = (0.80,\ 0.00,\ 0.20)$$

3 has its certificate revoked by the *CA 1*, Bob has its belief and disbelief values set at 0, while the value of uncertainty is set at 1. As the *CA 3* certificate's authenticity is no longer reliable, it is impossible to define which certificate is valid or invalid, making it impossible for us to make any conclusion about Bob's certificate.

In our model, we can define an organizational criteria for the PKI hierarchy, and use it to graduate the trust value through its levels. We can see an example of this graduation in the Table II. The *Root CA* has a belief value of one, its subordinates CAs have a belief value of 0.9, and so on. With this graduation of trust, we can establish an organizational criteria. For example, offline CAs usually are more secure than online CAs. So, offline CAs should appear on the tops levels of the PKI hierarchy (with a higher belief value), while the online CAs should be close to the bottom of the hierarchy (with lower belief value).

Following this organizational criteria, we can give an example of how the PKI can resist to an attack, using the PKI Illustrated in Figure 2. Assume that the *CA 3* is an online CA, and the *CA 2* is an offline one. Alice is an end-user that accesses Carol's web site that has a certificate issued by the *CA 2* (*O*). Using Levien's [22] attack model, we can verify how Alice will be protected if the *CA 3* gets attacked.

Levien defines two attack types, which he calls edge attack and node attack. In the edge attack the attacker can issue a certificate (create an edge) from a specific node, while in the node attack, the attacker can create any number of certificates (edges) from the attacked node. However, in any of these scenarios, if the attacked node is the *CA 3*, the attacker will not be able to issue any certificate with a transitivity trust value higher than $(0.00, 0.81, 0.19)$, because the trust values are defined in the CAs certificates and the hierarchy does not have multiple certification paths for one certificate. Even if the attacker issues several CA certificates with full trust value under the *CA 3*, the resultant transitivity value will be decreased to the value of $(0.00, 0.81, 0.19)$, when it passes through the *CA 3*'s certificate.

Assuming that Alice has already verified Carol's certificate before the attack over the *CA 3* happened, she will know that the certificate has a trust value of $(0.93, 0.00, 0.07)$. So, if the attacker uses a fraudulent certificate, in name of Carol, issued by the *CA 3*, Alice will be able to identify that it is a fraudulent certificate, because of its lower trust value. The only way for the attacker to be successful, is attacking the CA 2. However, it is a much harder task, considering that it is an offline CA.

$$\omega^{A:O}_{C_k} = (\omega^{A}_{R_k} \wedge \omega^{A}_{R_p}) \otimes (\omega^{R}_{O_k} \wedge \omega^{R}_{O_p}) \otimes \omega^{O}_{C_k} = (0.93, 0.00, 0.07)$$

If Alice does not know the trust value of Carol's certificate before the attack, she can establish a minimum value of trust for the context in which she is using the certificate. For example, if Alice is accessing an online bank, she can establish a minimum belief value of 0.9, while if she is accessing a university web site, she can establish a belief value of 0.8. Through these acceptance values, we can use an adaptation of Levien's [22] attack model, to calculate the efficiency of the PKI's organization criteria to resist attacks.

Now, we analyse the efficiency of our model to resist attacks compared to the classical X.509 model. For that, we use a certificate sample retrieved from the last available snapshot (March 2011) of *SSL Observatory*'s certificate database [25]. This database contains a large number of real certificates, which are validated under *Microsoft* and *Firefox* trust anchor repositories. Table III show the number of CAs and the number and percentage of final CAs (a final CA only issues certificates to end-users), for each hierarchy level of the PKIs in the sample.

TABLE III: CA DISTRIBUTION

Level	Total	Final	Final/Total
First (root)	176	50	28.41%
Second	422	364	86.26%
Third	365	352	96.44%
Fourth	21	16	76.19%
Fifth	5	5	100%
Total	989	787	79.57%

To calculate the efficiency of a PKI to resist attacks, we use an adaptation of Levien's attack model. Levien considers all PKI's nodes (certificates) as eligible to attack. We only consider final CAs as eligible nodes, as this kind of CA usually use online systems to issue certificates, which are preferable targets for hackers. In this sample, we have 787 final CAs, which represents 79.57% of all CAs. In the conventional X.509 PKI model, the decision to trust or not in a certificate is made when the certificate is validated by the certification path validation algorithm. As the certificates of our sample were already validated, all 747 final CAs can be a target to issue certificates that will be trusted by an user that trust in *Microsoft* and *Firefox* repositories.

To compare this result with our model, we need to define trust values for the CAs in the sample. As we can not determine which CA is more trustworthy than other, we assume that each level has a trust value lower than the level above. Besides this assignment being arbitrary, it can significantly reduce the number of eligible CAs. For example, to forge a certificate with trust value equals to a certificate issued by a CA at the second level, the attacker needs to successful attack a CA of that level or above. It reduces the number of eligible CAs to 414, representing a reduction of 47.39% compared to the conventional X.509 PKI. However, as the X.509 PKI does not have a real organization criteria, we cannot determine if a certificate at upper levels of the hierarchy will be securer than certificates at the bottom levels. A single compromised CA, at the first or second level, can jeopardize all the PKI, even with trust metrics.

VI. Conclusion and Future Work

In this paper, we demonstrated how to use trust calculation to solve the trust transitivity flaw of the *X.509* PKI. Thereby giving end-users a new tool that helps in his decision-making, allowing him to verify the trust level of a digital certificate and identify possible frauds. As a result, attackers will be discouraged from attacking certification authorities with lower trust level, by reducing the cost-benefit of these attacks. Our work differs from other works by focusing on the interpretation of trust in the context of *X.509* PKI, and by proposing structures and procedures necessary to support the calculation of trust, without jeopardizing its standards and maintaining the compatibility with existing applications.

For future work, we have the studies about the organization criteria that should be used to evaluate the trust level of PKI entities. This criteria must ensure a greater security for the PKI, which can be assessed through the Levien's work [22]. We also need to analyse the behavior of our model when a CA uses a registration authority and when the PKI uses cross and bridge certification to integrate other PKIs. In this case, we can use all existing knowledge about the trust calculus in web PKIs.

References

[1] C. Ellison and B. Schneier, "Ten risks of pki: What you're not being told about public key infrastructure," *Comput Secur J*, vol. 16, no. 1, pp. 1–7, 2000.

[2] M. Burmester and Y. G. Desmedt, "Is hierarchical public-key certification the next target for hackers?" *Communications of the ACM*, vol. 47, no. 8, pp. 68–74, 2004.

[3] Comodo, "Comodo fraud incident," 2011, retrieved: June, 2013. [Online]. Available: http://www.comodo.com/Comodo-Fraud-Incident-2011-03-23.html

[4] Vasco, "Diginotar reports security incident," 2011, retrieved: June, 2013. [Online]. Available: http://www.vasco.com/

[5] J. Segura, "Digital certificates and malware: a dangerous mix," 2013, retrieved: June, 2013. [Online]. Available: http://blog.malwarebytes.org

[6] L. H. Shenchangxiang, "Hierarchy-Distribution Combined PKI Trust Model," no. I 00044, pp. 121–124, 1996.

[7] A. Jøsang and S. Pope, "Semantic constraints for trust transitivity," in *Proceedings of the 2nd Asia-Pacific conference on Conceptual modelling-Volume 43*. Australian Computer Society, Inc., 2005, pp. 59–68.

[8] A. Jøsang, "An algebra for assessing trust in certification chains," in *Proceedings of the Network and Distributed Systems Security Symposium (NDSS99). The Internet Society*, 1999.

[9] S. Ruohomaa and L. Kutvonen, "Trust management survey," *Trust Management*, pp. 77–92, 2005.

[10] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision support systems*, vol. 43, no. 2, pp. 618–644, 2007.

[11] J. Huang and D. Nicol, "A formal-semantics-based calculus of trust," *Internet Computing, IEEE*, vol. 14, no. 5, pp. 38–46, 2010.

[12] H. El Bakkali and B. Kaitouni, "A logic-based reasoning about PKI trust model," *Proceedings. Sixth IEEE Symposium on Computers and Communications*, pp. 42–48, 2001. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=935353

[13] C.-N. Ziegler and G. Lausen, "Propagation models for trust and distrust in social networks," *Information Systems Frontiers*, vol. 7, no. 4, pp. 337–358, 2005.

[14] IETF, "Public-key infrastructure (x.509) (pkix)," Internet Engineering Task Force, 2013. [Online]. Available: http://datatracker.ietf.org/wg/pkix/

[15] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC 5280 (Proposed Standard), Internet Engineering Task Force, May 2008, retrieved: June, 2013. [Online]. Available: http://www.ietf.org/rfc/rfc5280.txt

[16] R. Perlman, "An overview of pki trust models," *Network, IEEE*, vol. 13, no. 6, pp. 38–43, 1999.

[17] J. Linn, "Trust models and management in public-key infrastructures," *RSA Laboratories*, vol. 12, 2000.

[18] R. Housley and T. Polk, *Planning for PKI: best practices guide for deploying public key infrastructure*. John Wiley & Sons, Inc., 2001.

[19] C. Adams and S. Lloyd, *Understanding PKI: concepts, standards, and deployment considerations*. Addison-Wesley Professional, 2003.

[20] U. Maurer, "Modelling a public-key infrastructure," in *Computer SecurityESORICS 96*. Springer, 1996, pp. 325–350.

[21] J. Huang and D. Nicol, "A calculus of trust and its application to pki and identity management," in *Proceedings of the 8th Symposium on Identity and Trust on the Internet*. ACM, 2009, pp. 23–37.

[22] R. Levien and A. Aiken, "Attack-resistant trust metrics for public key certification," in *7th USENIX Security Symposium*, 1998, pp. 229–242.

[23] D. Gambetta, "Can we trust trust," *Trust: Making and breaking cooperative relations*, vol. 2000, pp. 213–237, 2000.

[24] S. H. Simon Pope and M. Davey, "Subjective Logic Operators Demo," University of Oslo, 2011, retrieved: June, 2013. [Online]. Available: http://folk.uio.no/josang/sl/Op.html

[25] EFF, "SSL Observatory," Electronic Frontier Foundation, 2013, retrieved: June, 2013. [Online]. Available: https://www.eff.org/observatory