An Interdomain PKI Model Based on Trust Lists

Helena Rifà-Pous and Jordi Herrera-Joancomartí

Universitat Oberta de Catalunya, Rbla del Poblenou, 156
08018 Barcelona, Spain
{hrifa, jherreraj}@uoc.edu

Abstract. The penetration of PKI technology in the market is moving slowly due to interoperability concerns. Main causes are not technical but political and social since there is no trust development model that appropriately deals with multidomain PKIs. We propose a new architecture that on one hand considers that trust is not an homogeneous property but tied to a particular relation, and on the other hand, trust management must be performed through specialized entities that can evaluate its risks and threads. The model is based on trust certificate lists that allows users to hold a personalized trust view without having to get involved in technical details. The model dynamically adapts to the context changes thanks to a new certificate extension, we have called TrustProviderLink (TPL).

Keywords: trust lists, reliability in PKI, interoperability, certificate extension

1 Introduction

PKI technology has been widely accepted as the best solution to provide secure electronic transactions through an insecure channel. However, its global market penetration to common applications of general use it is not yet a fact.

The key reason for the slow adoption of PKI solutions in mass media products [1, 2] is due to interoperability concerns. Interoperability can not be simply characterized as a technology-only issue. In fact, it encompasses a wide range of technical, legal and political issues.

The PKI industry has addressed technical interoperability problems through a standardization process of data types and protocols. Nowadays, despite the flexibility of the specifications, PKI technology has achieved maturity and the basic interoperability goals between different vendor solutions are guaranteed.

Therefore, today's major drawback for the PKI adoption is the difficulty to deploy a cross-border solution due to differences in the countries legal and political framework. Governments are reluctant to recognize other nation's CAs if they do not take part in the quality control and, on the other hand, the scope of liabilities of a CA is not clear if the jurisdiction does not regulate it by law. Several proposals have been presented to overcome these problems such as cross-certification and Bridge Certification Authority. Yet, none of them has succeed

because of the complexities involved in the management of such infrastructures and the generalist perspective in which they are based.

The contribution of this paper is defining a procedure to facilitate the integration and interoperability of different PKI islands. The aim is being able to deal with elements of external security domains without creating a unique and monolithic infrastructure that is unable to adapt to any change. Users are the last responsible entities of the trust assignment within their context. Facilities are provided to identify the scope and attributions of each authority so that end entities can easily take the more appropriate decision for themselves.

The rest of the paper is structured as follows. In Section 2, the main trust models and their challenges are reviewed. Section 3 presents the proposed architecture and describes the entities and elements involved. In Section 4 we specify the functionality of the proposed architecture. Finally, section 5 concludes the paper and outlines some ideas for future research.

2 Trust models and challenges

PKI is intended to establish and maintain trusted relationships. In order to reach such objectives, mechanisms to propagate trust from credited organisms to unknown entities have to be built. See in [3] a survey on interoperability issues of multi-domain PKI. Next, we review the main trust model proposals and we identify their most relevant challenges.

2.1 Trust models

Single CA

PKI trust development has been studied and analyzed from PKI origins. The most simple topology architecture is the single Certification Authority (CA) model, that is, all certificates are issued by a unique CA. Although simple, this design neither scales well nor adapts to the society patterns, so it leads to the appearance of multiple interconnected CAs which manage communities of users that can not interoperate ones with the others.

Hierarchical PKI

The first attempt to solve the problem of having multiple interconnected trust islands was the hierarchical PKI structure that is managed by a Root Certification Authority (RCA). Trust is established in a tree-like fashion and flows from top to bottom. The RCA public key is the fundamental point of trust, or trust anchor, for evaluating certificate acceptability. In this model the path construction procedure is very simple, as a single path exists from any end entity up to the RCA.

However, deploying a global unique RCA is inappropriate for political reasons. There is not a consensus about whom it would manage the RCA and how it would do it. Thus the conclusion is that this model is only directly applicable within one domain, which is generally supported in one or several communities

generally forming different security domains, but always within one unique administrative domain.

Mesh PKI

There has been multiple efforts to bring trust development to inter-domain levels. In the cross-certification model, also known as mesh model, two CAs cross-certify each other once they agree to trust and rely on each other's issued public key certificates as if they had generated them themselves. Pairs of CAs exchange cross-certificates and enable users from one administrative domain to interact electronically and securely with users from the other.

However, the number of cross-certificates tends to grow exponentially with the number of CAs and policy mappings are very complex. Therefore, there appear scalability problems.

On the other hand, if a PKI domain A wants to join another PKI domain B but restrict or deny trust in one or more other domains that B may have joined, A has to issue a cross-certificate to B where policy constraints shall be explicitly included. This makes the building of certification paths between two generic end entities still more unmanageable.

Mesh model can be also build upon a hierarchical PKI architecture so that the number of required cross-certificates can be reduced and the complexity of the system lightened. Although IETF has included CA cross-certification in its Certificate Management Protocol [4] and there are some implementations of it, cross-certification is still not well supported by common general-purpose applications like browsers or email clients.

Trust Lists

There is not an homogenous way to define or formalize trust lists. While for ones is just a list of certificates (i.e. stores of certificates used in web browsers), for others is a signed list that can contain any trusted information and, in particular, certificates; this is the case of the Certificate Trust List (CTL) format from Microsoft.

In this paper we use the term trust list to designate a signed set of trusted certificates plus information defining properties and constraints on how to apply this trust [5].

There are two types of trust lists:

- User trust lists: managed by a single user.
- Provider trust lists: managed by a trust provider.

The user trust list model is the most common trust development architecture in use today, offered by operating systems and web applications. An example is the list of more than a hundred CAs included in distributions of Microsoft OS. The requirement to appear in this list is paying a fee to Microsoft. On the other hand, end users can modify this list by adding or deleting CAs as needed.

User trust list model does not present technical complexities. However, it has to be noticed that users do not have the means or skills to construct their

own trust list from scratch because they do not know the CAs nor are able to evaluate the risks that entails accepting them. This is, for example, the case of web browser applications, that are distributed with a predefined trust list attached to them so that users do not have to create it.

On the other hand, provider trust lists are created and managed by trust providers (TP). Its aim is to serve as a reference for users; the trust on the certificates of the list is recommended by a TP. From an inter-domain interoperability perspective, the provider trust list essentially replaces the cross-certificate pair in the Mesh model. The user trusts the issuer of the list, adopts the list, and then the trust is extended to the CAs conveyed within it.

Bridge CA

Bridge CA (BCA) trust model was first introduced by the U.S. Federal Government [6] as a way to facilitate the interconnection of CAs through a cross-certification process. Each user only trusts its own CA which in turn trusts the bridge that finally trusts the remote CA, so that each member needs only to maintain a single cross-certification with the BCA and then it is automatically able to build certification paths across all spokes.

It must be noted that the BCA is not intended to be used as a trust anchor by the users of the PKI. It simply acts as a gateway between isolated CAs. Even so, BCA is responsible to map certificate policies and guarantee PKI equivalences adequately. Therefore, users must rely on it regarding these mappings.

Although this model is quite simple from the end user perspective, in fact it presents technical difficulties because the path construction is intrinsically complex and several checks (e.g. policy and name constraints, certificate status, policy mappings) must be performed throughout the certificate chain. Nowadays, the Federal BCA is trying to interconnect with other BCAs, but it is reporting both technical and operational problems (see Top 10 issues from [7]).

European Community took a project for deploying a BCA for the member states. In a way to facilitate the interoperability, their proposal [8] was to create an hybrid BCA that combines the distribution of accredited CA certificates under the form of a signed list, and the cross-certification of the CA with the BCA for those users that do not want to download trust lists.

The European BCA pilot finished in 2004. They concluded [5] cross-certification was more tedious than trust lists and that the model should be restricted to the use of these last ones.

Bridge VA

The Bridge Validation Authority (BVA) trust model [9, 10] is a further step to the BCA in which the central entity is not a CA but a Validation Authority (VA). VAs are trusted third parties that offer online services on certificate validation. In general terms, they are responsible of building the certification path, evaluating the quality of the certificates, validating their status, and ensuring they are trustworthy.

In the BVA model, the element that links multiple PKI islands is an VA that gathers and classifies status information of certificates from multiple domains. BVA becomes the trust anchor for users and admits liabilities for the certificates it works with.

This model solves some of the technical complexities of BCA, for instance, when dealing with path construction, and offers to the users a more comfortable service. Although simple, users are totally relegated to the decisions of the BVA and do not receive any information about the characteristics, quality or application context associated to the CA they are working with.

On the other hand, as we have already seen in other models, it can not exist a unique BVA in the world due to political reasons, but multiple instances of them. Whether users trust one or more BVA, and how they combine the results stated by each authority has to be solved for each user.

2.2 Trust Development Challenges

As we have described above, several solutions for building trust have been proposed, however, trust development still presents challenges. In the following use case, we expose the common problems users face up when dealing with security services.

Lets suppose a user, Bob, who has traveled to a foreign country and is visiting the city. While he is walking in the park, he sees a publicity announcing a music show that night in the theater. He wants to buy a ticket, so he gets his PDA an tries to connect to a wifi network. He finds an ad hoc network that reaches Internet and makes a request for establishing a connection.

Charles is another member of the ad hoc network. He is Bob's neighbor and can provide him with forwarding services so that Bob can connect to the theater. However, Charles would not forward Bob's packets unless he can assure Bob is not a selfish user of ad hoc networks, and that he will be rewarded afterward through a reputation system. Therefore, he wants to verify Bob's identity and that he is registered to a CA that takes liabilities in case of problems.

Charles verifies the signature of Bob's certificate and checks all data is correct. However Charles is unable to verify the certification chain of Bob up to a root CA because he does not know that authority and does not trust it. If Charles' CA has not a binding relationship with Bob's PKI domain, he has no further means to obtain relevant information for extending trust to Bob.

In this example case, let's suppose Charles takes the risk of giving forwarding services to Bob. Now, Bob can reach the Internet and tries to buy theater tickets. He can successfully build the vendor's certification path and validate the certification chain is correct. Still, he does not have any information on the quality of the certificate and if it is appropriate for using it in e-commerce applications. Bob does not want taking the risk of buying the tickets in a foreign country to a vendor that uses a certificate that is not qualified, so pitifully, he will not be able to go to the theater.

The example presented above shows that the challenges of trust models based on PKI can be summarized in:

- Processing certification paths, that is, finding an ordered sequence of certificates from the end entity certificate to a trust anchor.
- Determining the quality of the certificate, which can be usually derived from the certificate policy.
- Deciding if the certificate is trustworthy for the purpose at hand.

Some models like BVA and Trust Lists have a simple certification path management, however, they do not offer users details about quality of service parameters. If a CA issues two types of certificates with different security policies, the BVA party will extend the same type of responses for the two of them, without including quality information that can be used by the user as a qualifying parameter for the trust decision.

BCA model offers some quality of service through cross-certification policies. The BCA is responsible for certification mapping and it has to publish the rules that follows to accept new CAs. However, such architecture presents technical complexities that can harden interoperability.

Several proposals have emerged to formalize a certification policy format [11, 12] and define a way to interpret the CA liabilities and the quality of the certificates it issues. In [13] authors define an automatic way to compare certificates from different CAs and determine its relative quality. The use of standard description rules will facilitate certificate evaluation. However, users neither have the knowledge nor the capacity to interpret all the parameters and take a decision about the quality of the certificate. So, they have to delegate this operation in some external entities, which we call Trust Providers (TP).

Another challenge of PKI trust models is how to extend trust and decide if a certificate is trustworthy for the purpose at hand. In spite of some technical complexities, the main problem of existing trust models is that they do not follow the trust building guidelines we use in personal relationships.

Setting up a global uniform network of trust is not viable. The models proposed so far that admit some personalization are the ones based on trust lists. However, current trust lists implementations are very simple and only cover certificate's classification in a few categories: trusted CA certificates, web server certificates, code validation certificates and end user certificates.

3 Architecture description

In this section we describe the architecture of our trust model aimed to facilitate the multidomain PKI interoperability. The main characteristics of the proposed architecture are:

- Built upon a centralized PKI model that extends trust from well know authorities in which users trust, the TP.
- Let users configure its own trust list based on recommendations, that can be accepted or not.
- Facilitate trust list dissemination and management using core PKI mechanisms.

Different entities interoperate in our architecture using different elements (see figure 1).

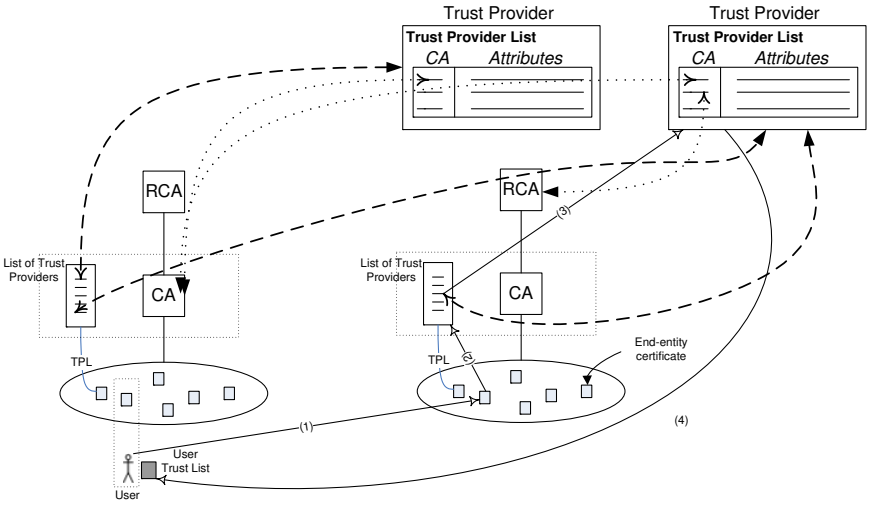

Fig. 1. System architecture

3.1 Architecture entities

The main architecture entities of our model are users, Certification Authorities (CA) and Trust Providers (TP).

In our architecture we use the term user in a wider sense. A **user** is the one that has a certificate but also the entity that wants to validate a certificate. Notice that the latter does not need to have a certificate, and in some environments entities that validate certificates are called relying parties [14]. However we use the term user in both cases in order to simplify the description.

Certification Authorities (CA) are entities that issue certificates and vouch for the binding between the data items in these certificates. CAs manage the whole life cycle of certificates, revoking them if they are compromised before the validity period, renewing if the validity has to be extended, and publishing updated status information so that users can be accurately informed. Root Certification Authorities (RCA) are a special case of CA with a top hierarchical level.

Trust Providers (TP) are well known entities that have accredited political, legal or social impact (i.e. the Ministry of Law or recognized private enterprise). They manage lists of CA certificates that they consider reliable and that have the required quality for being used in some specific actions. The ap-

plication context of the certificates in the list is confined to the influence scope of the Trust Provider.

3.2 Architecture main elements

Our architecture main elements are certificates, list of trust providers, trust lists, and a trust list enforcement engine.

Certificates

Certificates are the central point of any PKI based model. Our architecture uses X.509 certificates [15] in which a new certificate extension called Trust Provider Link (TPL) has been included.

The TPL is a noncritical certificate extension that contains URL locations where a list of trust providers can be retrieved. The TPL extension shall be identified by a unique object identifier:

```
id-pe-trustProvidersLink OBJECT IDENTIFIER  ::=  { id-pe 20 }
```

The ASN.1 specification of the extension is the following:

```
trustProvidersLink ::= SEQUENCE SIZE (1..MAX) OF
                           TrustProviderLink

trustProviderLink  ::=  SEQUENCE {
        accessMethod          OBJECT IDENTIFIER,
        accessLocation        GeneralName  }
```

`trustProvidersLink` is a list of `trustProviderLink` objects that contain the location where the CA that issued the certificate publishes the list of TP.

List of trust providers

A list of trust providers is the relation of different TP that support the CA policy. Such list is formatted in XML and includes a brief description of the TP included. Table 1 shows the elements of the list in XPath language [16].

Notice that lists of trust providers are signed to avoid fake manipulations in the data. A specific example of a list of providers is shown in figure 2.

Trust lists

The main element of our architecture are trust lists. Two different trust lists can be identified: user trust lists and provider trust lists, which are managed by TPs so that they are also known as TP trust lists.

Trust lists contains three groups of information:

- List owner data: It identifies the entity responsible of the trust list. In case such entity is a TP the list is more reliable and it includes the services offered by the TP, the number of clients that it has, the purpose of the list and its target. It also states if the TP is refining an existing trust list, or it creates it from scratch.

Table 1. List of trust providers parameters

XPath	Value
`/List`	List container
`/List/TrustProvider[n]`	Information of a n'th Trust Provider (TP) of the list
`/List/TrustProvider[n]/Name`	TP's distinguished name
`/List/TrustProvider[n]/Context`	The operational context of the TP, i.e, public administrations, e-commerce, ..
`/List/TrustProvider[n]/Scope`	TP's influence ambit. It can be global (i.e. Microsoft) or local (trust list of a European member state)
`/List/TrustProvider[n]/CertProvider`	TP's certification provider, in case it exists
`/List/TrustProvider[n]/Reference`	URL location of the trust lists
`/List/ds:Signature`	XML Signature of the whole list

```
- <List>
  - <TrustProvider>
      <Name>CN=TrustProvider, O=Ministry of Law, C=ES</Name>
      <Context>Law</Context>
      <Scope>Spain</Scope>
      <CertProvider>OU=FNMT Clase 2 CA, O=FNMT, C=ES</CertProvider>
      <Reference>https://www.minlaw.es/trust_provider/trust_list</Reference>
    </TrustProvider>
  - <TrustProvider>
      <Name>CN=TrustProvider, O=UOC, C=ES</Name>
      <Context>E-learning</Context>
      <Scope>Europe South-America</Scope>
      <CertProvider>O=UOC, C=ES</CertProvider>
      <Reference>https://www.uoc.edu/trust_provider/trust_list</Reference>
    </TrustProvider>
  - <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    - <ds:SignedInfo>
        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
        <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" />
      - <ds:Reference URI="">
        - <ds:Transforms>
            <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1" />
          <ds:DigestValue>WScEsaC0y3PzG3cC9nUqLmTrnN8=</ds:DigestValue>
        </ds:Reference>
      </ds:SignedInfo>
      <ds:SignatureValue>gXqJmcIquM0PQjMWIziSXDiLHa6YhEmROorIMh65TEVdvA0c5bH1LJLVinC
        9od5nmarv8fIfF9Y/nDVW7Ad5NHcTYO+EOhyDA==</ds:SignatureValue>
    - <ds:KeyInfo>
      - <ds:X509Data>
          <ds:X509Certificate>MIIEIzCCAwugAwIBAgIBBDANBgkqhkiG9w0BAQUFADCBnjELMAkGAU
            G0+l2k2Y9fzfBqwx7OekTR+OQsMiTHuU/PvQq03uCcZHILF7Qw==</ds:X509Certificate>
        </ds:X509Data>
      </ds:KeyInfo>
    </ds:Signature>
  </List>
```

Fig. 2. List of trust providers example

– Certificate list: List of trusted certificates and qualifying information.

– Authenticity data: It includes parameters to validate the integrity and authenticity of the data contained in the list. Such parameters include a hash of the data, a signature and the signing certificate.

We introduce a format of trust lists that allows defining quality parameters for the certificates. Table 2 shows the elements of the list in XPath language. The format of the list is based on the ETSI specification [17] that has been extended to include certificates data.

Table 2. Trust lists parameters

XPath	**Value**
`/TrustList`	Trust List (TL) container
`/TrustList/ThisUpdate`	Publication date
`/TrustList/Target`	Public target
`/TrustList/Purpose`	Purpose
`/TrustList/From`	List of Trust Providers (TP) that have contributed to the creation of the present TL
`/TrustList/From/TrustProvider[n]`	Information of a n'th TP
`/TrustList/From/TrustProvider[n]/Id`	Distinguished Name
`/TrustList/From/TrustProvider[n]/Update`	Publication date of the provider's list
`/TrustList/Owner`	Information of the responsible of the TL
`/TrustList/Owner/Id`	Distinguished Name
`/TrustList/Owner/Name`	Name of the responsible legal entity
`/TrustList/Owner/Address`	Postal address
`/TrustList/Owner/URI`	Web address
`/TrustList/Owner/Services`	List of services provided by the entity
`/TrustList/Owner/BeginDate`	Legal entity creation date
`/TrustList/Owner/Context`	The operational context of the owner, i.e, public administrations, e-commerce, ..
`/TrustList/Owner/Market`	Market of the owner
`/TrustList/Owner/Status`	Financial range of the owner
`/TrustList/CertList`	List of trusted certificates
`/TrustList/CertList/CA[n]`	Information of a n'th CA. This element includes trust parameters in semantic languages
`/TrustList/CertList/CA[n]/CertProperties`	Encoded certificate and certificate fields
`/TrustList/CertList/CA[n]Constraints`	Information to qualify the certificate
`/TrustList/ds:Signature`	XML Signature of the whole list

TP evaluate certificates they want to include in their trust list based on their knowledge of the CA holder and the quality of the issued certificates (that is, the CA certificate policy). Results from the evaluation are expressed on the list using an ontology language [18]. Ontology languages are formal languages used

to construct ontologies, that is, specifications of some concepts. They allow the encoding of knowledge about specific domains and often include reasoning rules that support the processing of that knowledge.

We use ontologies to include any properties and constraints for the certificates issued by the CAs in the list. Categorization of certificates is achieved with information of two orthogonal fields (see figure 3):

- **Service**: the security services that shall be granted, like identification, data authentication, access control, non-repudiation or establishment of a confidential channel.
- **Sector**: the ambit of the target application, i.e., government, pharmaceutical, banking, transport, leisure, etc.

Finally, TP define a certificate Trust Level for using the stated certificates in each specific environment.

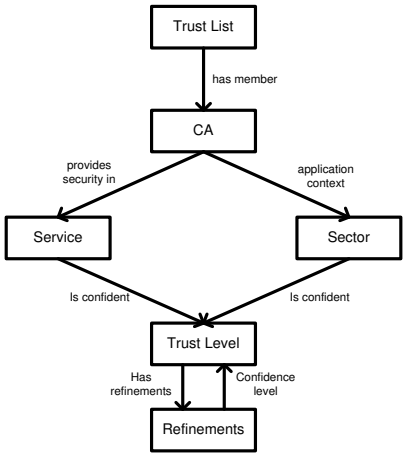

Fig. 3. Trust list ontology schema

Ontologies provide means to define relations between the objects in a domain. This information is exploited in trust lists as a way to automatically propagate trust on certificates used in similar environments. For example, a TP may recommend a certificate to be accepted for non-repudiation services in governmental applications. By extension, if a user asks if he shall accept this certificate for authentication purposes in an e-administration environment, the response will be true.

Extending the context of use of a certificate entails some risks, for this reason, the confidence level of this kind of recommendations will always be lower than the primary goals for it was intended. The client application is responsible to define a threshold above which certificates will be accepted, and rejected otherwise.

```
- <TrustList>
    <ThisUpdate>2007-03-02T09:16:16Z</ThisUpdate>
    <Target>Spanish citizens</Target>
    <Purpose>Social services access</Purpose>
    <From created="true" />
  - <Owner>
      <Id>CN=Medicus, C=ES</Id>
      <Name>Medicus, S.A.</Name>
      <Address>Calle Alcala, 43. 3-2 Madrid</Address>
      <URI>http://www.medicus.es</URI>
    - <Services>
        <Service>Health</Service>
        <Service>Insurances</Service>
      </Services>
      <BeginDate>1985-05-23T08:00:00Z</BeginDate>
      <Context>Public Health</Context>
    </Owner>
  - <CertList>
    - <CA rdf:ID="AC Raiz DNIE 01" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
      - <CertProperties rdf:resource="#x509cert">
          <Subject>CN=AC Raiz DNIE, OU=DNIE, O=Direccion general de la policia, C=ES</Subject>
          <Issuer>CN=AC Raiz DNIE, OU=DNIE, O=Direccion general de la policia, C=ES</Issuer>
        - <Validity>
            <NotBefore>2006-02-27T11:54:38Z</NotBefore>
            <NotAfter>2021-02-26T23:59:59Z</NotAfter>
          </Validity>
          <EncodedCert>MIIFxTCCA62gAwIBAgIQZCBmyZl7ruFEAtpupCL9w0BAQUFADBdUE9MSU
            MQswCQYDVQQGEwJFUzEoMCYGA1UECgwfRElSRUNQUwgREUgTEEg</EncodedCert>
        </CertProperties>
      - <Constraints>
          <ProvidesSecurity rdf:resource="#NonRepudiation" />
          <ApplicationContext rdf:resource="#eHealth" />
          <TrustLevel>8</TrustLevel>
        </Constraints>
      </CA>
    </CertList>
  + <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  </TrustList>
```

Fig. 4. Trust list example

Figure 4 shows an example of a TP trust list. The TP is a medical enterprise that offers health services and insurances. It has defined a trust list of certificates to be used in social services. In particular, the example depicts a CA that can be used for non repudiation operations in an e-health context.

Trust list enforcement engine

Since there are many independent PKI domains in the world and new ones are appearing every day, user trust lists have to be updated regularly to meet the changes. New methods have to be provided in order to modify and include the appropriate CAs in the list and trust list enforcement engines are the appropriate tool.

Trust lists are tied with trust list enforcement engines, that are used to encode users requirements and check if certificates fulfill the stated rules according to the list. The engine is the interface of trust lists that can interpret users requirements in natural language and automatize the process of list modification and update.

4 Architecture function

4.1 User trust list creation

Users trust lists are initially created importing the information from a provider trust list of an entity whom the user knows and trusts. Therefore, users construct they own trust list based on TP recommendations, that are formalized in an TP trust list.

TP can offer a downloading interface that allows exporting only some parts of the list (some selected CAs). Anyway, the information users accept from the provider trust lists, can be then refined and modified.

4.2 User trust list management

Trust providers value and classify certificates in the lists they publish. Users can refine or modify this classification in the lists managed by themselves. When a user imports new providers trust lists to his own list, the trust list enforcement engine merges the data with the contents of the other providers.

In our model, the use of user trust lists is not mandatory. There is a way to fasten the trust validation of a certificate and guarantee the service is available anytime. However, users that need to check a certificate and are not using their personal device nor having access to their trust list, can perform the validation directly. The trust list enforcement engine directs the user to the list of TP referenced in the certificate under validation, the user will have to indicate if he trusts any of the TP on the list and, if positive, the trust list of the selected TP will be checked and its recommendations imported.

4.3 Certificate evaluation

In the general model, users maintain a trust list that contains the PKI domains they trust. When they are involved in an electronic transaction and need to decide the acceptance of a certificate, they delegate to the trust list enforcement engine the certification path construction and the evaluation of whether the certificate is sufficiently trusted and qualified for that specific purpose. The inputs to the engine are the certificate itself, and a description of the context in use. For example, a user can request if a certificate is admissible for non-repudiation purposes in a work contract.

Certificate evaluation can be directly requested from the client application itself (mail client, web browser, document reader, ...). In this case, the client application is the one that passes contextual inputs to the trust list enforcement engine.

Figure 1 shows the certificate evaluation steps:

Step 1 The enforcement engine tries to build a trust certification path from the user certificate to a trust CA in the list. If it is unable to do it, it can be because it does not know that authority, or because it does know it but the

user does not explicitly trust in it. In the first case, as the certificate under evaluation holds a TPL extension, the engine can obtain more information about the PKI domain where the certificate belongs through the recommendations made by some TPs. The TPL extension points to a list of TPs that can supply more information about the unrecognized certification chain.

Step 2 The enforcement engine connects to the web page in which the CA that issued the certificate publishes a list of TPs. All TPs in the list support the CA, however, the context and scope of each provider is different. If the list of TPs is not available, the certificate under evaluation is considered untrustful and the process is finished.
Otherwise, after getting the list of TPs and verifying that is authentic and has not been manipulated, the engine searches if the user knows some of the providers, that is, checks if TPs are registered in the user trust list. If this is the case, the recommendation of the registered TPs is accepted. Otherwise, the engine asks the user if he want to import some new TP in the repository and acts accordingly.

Step 3 TPs are reached. The client verifies the signature of the TP trust lists.

Step 4 TPs trust lists are downloaded and merged with the current information in the user trust list. When recommendations on a PKI domain come from several trust providers, the best trust level values are the ones that prevail. If information of trust providers has to be merged with data stated by the user owning the list, the user opinion takes preference.

Finally, we review the use case example we have introduced in section 2.2 to explain how it can be resolved using our trust list model. Users Bob and Charles are connected in an ad hoc network. Bob requests Charles forwarding access through his node so that he can reach Internet. Charles is not able to construct Bob's certification chain and he does not trust the authority. As Bob's certificate holds a TPL extension, he can get information about the TP that recommend the acceptance of the certificate.

Charles recognizes some of the TP on the list. TP are reached in order, starting by the provider that best meets the requirements of the situation, and ending with the remotest one. For each provider that recommends the use of the certificate, the client application alerts the user and asks for a confirmation.

In our particular case, Charles seeks confidence parameters in the context of ad hoc networks, and particularly in the access control operation. TP that support Bob certification chain are unknown to Charles, and are in the public administration domain. However, because the number of known TP that recommend the use of the certificate is very high, and because the risk of the operation is low, Charles accepts the identity of Bob and grants him the access.

So far, Bob can browse the Internet. He connects to the theater for buying the ticket. However, the vendor presents him a certificate that he does not recognize because does not maintain a user trust list in his handheld device due to its

limited capacity. He asks the trust list enforcement engine to verify the validity of the certificate for e-commerce services.

The engine verifies that the vendor certificate is issued by a CA that is member of the certificate trust lists of the major software vendors, and that the context of this CA is e-commerce. It asks Bob if he relies on the TP on the list, and as the response is affirmative, the engine validates positively the vendor certificate and Bob is able to perform the purchase.

5 Conclusions

Interoperability issues of interdomain PKIs are one of the key problems to be solved in order to allow a massive deployment of PKI technology. Although several solutions have been proposed so far, none of them has succeed in the market due to technical, political and social constraints.

In this paper we have presented a new trust development model more flexible than the former ones from the user point of view. The proposed model is based on trust lists in order to resolve interoperability issues of interdomain PKIs. It is built upon a hierarchical PKI model and extends trust using TPs. As in the BVA model, the trust anchor is not a CA, but the TP.

The use of TPs facilitates the adoption of interdomain CAs by the users because they are close entities which users really know and treat. TPs have gained their reputation in a certain community by their demonstrated know-how, activities and projects, and they are trusted in their area of expertise. TPs are used to give recommendations, help users to interpret CAs security policies and give them information about the PKI domains. On the other hand, the promotion and dissemination of TPs is achieved thanks to a new certificate extension that provides information of the entities that support its PKI domain.

Moreover, the presented architecture deals with categorized trust lists expressed in semantic language so that a more specific approach about when is appropriate or not to accept a certificate can be surely stated.

The use of ontologies is twofold: first it permits to describe the complex relations of the real world in a language that is interpretable by computers; Second, it provides the base to deploy natural language interfaces for the TPs so that user-friendly applications can be developed and consumed by anyone, although they do not have technical skills.

Further research will be focused on defining specific ontologies for the trust list enforcement engine to allow a highly configurable and automatic user list management.

Acknowledgement

The work described in this paper has been supported in part by the Spanish MCYT with a grant for the project PROPRIETAS-WIRELESS SEG2004-04352-C04-04.

References

1. Backhouse, J., Hsu, C., Baptista, J., Tseng, J.C.: The key to trust? signalling quality in the PKI market. In: Proceedings of the 11th European Conference on Information Systems, ECIS. (2003)
2. Doyle, P., Hanna, S.: Analysis of June 2003 Survey on Obstacles to PKI Deployment and Usage. OASIS Public Key Infrastructure (PKI) Technical Committee (TC) (2003)
3. Shimaoka, M., Hastings, N., Nielsen, R.: Memorandum for multi-domain Public Key Infrastructure Interoperability. IETF Internet Draft (2007)
4. Adams, C., Farrell, S., Kause, T., Mononen, T.: Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP). IETF RFC 4210. Standards Track (2005)
5. Certipost: Trust List Usage Recommendations for a European IDA Bridge/Gateway CA Pilot for Public Administrations. IDA PKI II / EBGCA / WP1.2 (2004)
6. Burr, W.E.: Public Key Infrastructure (PKI) technical Specification: Part A – Technical Concept of Operations. NIST Working Draft (1998)
7. Blanchard, D.: I-CIDM Bridge to Bridge Interoperations. 5th Annual PKI R&D Workshop Making PKI Easy to Use (2006)
8. EDS: A bridge CA for Europe's Public Administrations - Feasibility study. European Commission - Enterprise DG. Public Key Infrastructure for Closed User Groups Project (2002)
9. Malpani, A.: Bridge Validation Authority. White Paper, ValiCert (2001)
10. Ølnes, J.: PKI Interoperability by an Independent, Trusted Validation Authority. 5th Annual PKI R&D Workshop Making PKI Easy to Use (2006)
11. Casola, V., Mazzeo, A., Mazzocca, N., V.Vittorini: Policy Formalization to combine separate systems into larger connected network of trust. Proc. of Int. Conf. on Network Control and Engineering for QoS, Security and Mobility (Net-Con) (2002)
12. Chokhani, S., Ford, W., Sabett, R., Merrill, C., Wu, S.: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. IETF RFC 3647. Informational (2003)
13. Casola, V., Mazzeo, A., Mazzocca, N., Rak, M.: An Innovative Policy-Based Cross Certification Methodology for Public Key Infrastructures. In: Public Key Infrastructure, Second European PKI Workshop: Research and Applications, EuroPKI. Volume 3545 of Lecture Notes in Computer Science., Springer (2005) 100–117
14. Cooper, M., Dzambasow, Y., Hesse, P., Joseph, S., Nicholas, R.: Internet X.509 Public Key Infrastructure: Certification Path Building. IETF RFC 4158. Informational (2005)
15. Housley, R., Polk, W., Ford, W., Solo, D.: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. IETF RFC 3280. Standards Track (2002)
16. Berglund, A., Boag, S., Chamberlin, D., Fernndez, M.F., Kay, M., Robie, J., Simon, J.: XML Path Language (XPath) 2.0. W3C Recommendation (2007)
17. ETSI: Electronic Signatures and Infrastructures (ESI); Provision of harmonized Trust Service Provider status information. Draft ETSI TS 102 231 V1.2.1 (2005)
18. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A.: OWL Web Ontology Language. W3C Recommendation (2004)