School of Computing Science,
University of Newcastle upon Tyne

An Approach to the Formalisation of a Certification Policy

Omar Batarfi and C. R. Snow

Technical Report Series

CS-TR-918

July 2005

Published by the University of Newcastle upon Tyne,
School of Computing Science, Claremont Tower, Claremont Road,
Newcastle upon Tyne, NE1 7RU, UK.

An Approach to the Formalisation of a Certification Policy

Omar Batarfi Omar.Batarfi@ncl.ac.uk

C.R.Snow c.r.snow@ncl.ac.uk

School of Computing Science, University of Newcastle

Abstract

In this paper, we present an initial approach to the problem of specifying formally the requirements of the Certification Policy as published by a Certification Authority. The approach uses the XML language to describe the structure of the policy document, and attempts to provide a canonical representation of the document in order to allow the semantics of the policy to be described. The ultimate aim is to permit policies to be compared, thus allowing a degree of trust to be inferred between entities holding certificates issued by otherwise unconnected Certification Authorities.

Introduction

A common and increasingly popular method of establishing trust between entities in cyberspace is by the use of digital certificates. Digital certificates are electronic documents essentially confirming the identity of the holder, and relating this identity to a set of additional attributes "owned" by the identity.

The *de facto* standard for digital certificates is X.509 [1], of which the current latest version is Version 3. The trust that can be placed in a certificate comes about through the fact that the digital certificate (also called a public-key certificate, or simply a certificate) is issued by a trusted entity known as a Certification Authority (CA). The CA will receive a certificate signing request from a subject, and after making a number of checks to establish the credentials of the applicant, will issue the certificate to an end-entity (e.g., end-user, device, Web server, process, etc.), or another CA, which is essentially a digitally signed copy of the information supplied by the requestor, together with some additional information supplied by the issuer (typically the validity period of the certificate falls into this category). The signature provides the proof that the CA agrees that the information contained in the certificate is correct.

With the increasing use of the Internet for transactions between entities who have not previously interacted with one another, the use of certificates to establish trust will also increase, and this creates a problem of when one entity (the *relying party*)[1], can reasonably trust another entity (the *target*).

Trusting certificate (actually, trusting the binding embodied in the certificate) could be examined by the relying party, and the degree of the trust depends on several factors [1]. These

[1] As defined in [11] "A user or agent that relies on the data in a certificate in making decisions".

factors include CA practices for authenticating the subject; the CA's operating policy, procedures, security controls, the subject's obligations such as protecting the private key and finally, CA’s legal obligations and the stated undertakings (for example, warranties and limitations on liability).

Every CA is required [2] to provide a Certification Policy (CP) and a Certificate Practice Statement (CPS), which define exactly the circumstances under which certificates will be issued, and what reliance can be placed on the certificates issued by that CA. It can be assumed that any entity requesting a certificate will be prepared to accept the terms and conditions laid down by these two documents. In fact, the CA will make it a condition of issuing the certificate that the recipient does so accept them. Clearly, any (non-root) CA can only adopt a CP/CPS for its own use if it also satisfies the terms of the CAs it in the chain from itself to its root CA. (It is possible, however, for a CA to refine the terms of the CP/CPS for its own use, provided it in no way weakens the requirements of the CP/CPS it inherits.)

Because the relying party already has a trust relationship with one (or more) of the CAs, and the current situation is that certain world-wide Certification Authorities – Verisign, Thawte, etc. – are almost universally the root CA of every certificate issued. (Why this should be, and why these particular companies should be regarded as the ultimate trusted third party is unclear. Schneier and Ellison [3] address this interesting question in their now infamous “Ten Risks of PKI” paper.) However, we foresee a situation in which a trust relationship between relying party and target cannot be established, because the trust path for the target has no point in common with the trusted entities known to the relying party.

One possible approach to this problem is for the relying party to examine the CP/CPS of the target certificate, and try to decide the extent to which the policy of the target’s CA “matches” the policy of the relying party’s own CA. Clearly, study of the target policy by hand would be a feasible, if long-winded, approach, but the ultimate aim of our work is to explore the possibilities of carrying out this task without human intervention.

Public-Key Infrastructures

The term Public Key Infrastructure (PKI) is usually taken to mean a set of techniques, procedures and policies which all members of the Infrastructure have agreed to follow. A PKI will typically include a number of CAs, together with all the entities to which certificates have been issued. A PKI will normally be expected to include a repository for certificates, and a Certificate Revocation List (CRL). Due to the nature of the PKI, the certificate policy formats the PKI domain in the sense that there are numbers of CAs and end-entities follow a certain CP, and as stated in [4] there are CAs who admitted more than one policies which implies the CA becomes a subordinate to more than one root CA. A Version 3 X.509 certificate may contain a field declaring that one or more specific certificate policies applies to that certificate [1].

A PKI domain is the environment that connects nodes (CAs and end-entities) together. When an authentication request is issued by a relying party to validate a target, the process of attempting to construct a certification path will start. The certification path processing is the mechanism for authentication of any certificate, but this processing is applicable for domains where:

1. The target and relying party reside on the same domain. See Figure 1.
2. The target and relying party reside on different domains, but these domains are connected together directly or indirectly by some previous *cross-certification* (see Figure 2). This cross-certification may take a number of forms as described in [2].

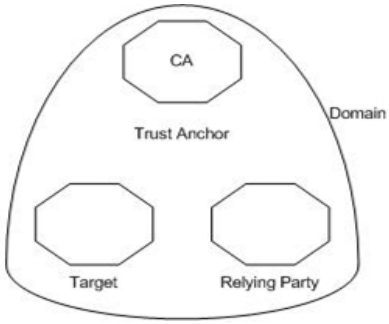


Figure 1: Relying party and the target on the same domain

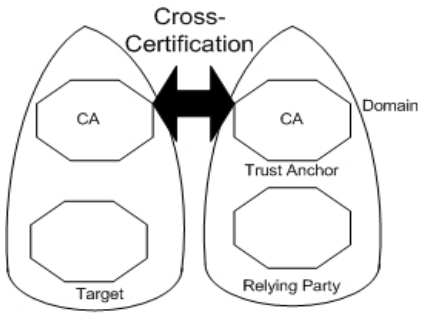


Figure 2: Relying party and the target on different domains

This paper considers the problem where there is no such certification path exists, i.e. neither of the two situations described above apply [5].

Certificate policy

A Certificate Policy (CP) is defined in [1] as: a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements. For example, a particular CP might indicate applicability of a type of certificate to the authentication of electronic data interchange transactions for the trading of goods within a given price range.

A certificate may include a field that has a value called Object Identifier (OID) which refers to a specific CP; this helps also to distinguish one CP from another. For better qualification, Certificate Policies certificate extension could have one or more OIDs; for example, RFC3280 suggests two Certificate Policy qualifiers: a User Notice and a pointer to a Certification Practice Statement (CPS) [4].

A CP is a statement of agreement between the certificate issuer and the end-entity of certificate which means both of their certificates should have the same CP. In the cross-certificates domain the Policy Mappings extension will point to equivalent CPs in different domains[4].

In [1] it is stated that a CP is considered as an accreditation of a CA in the way that CA will be accredited against the CP. Moreover the CP has a significant part to play in the cross-certification, where the issuing CA relies on its CP to allow trust to be placed in the target's CA; an entry in the target's certificate refers back to the CA and hence to the CP.

A relying party may use the CP of a target to decide whether the target's certificate is trustworthy for a particular application [2].

It is clear from the importance of the Certificate Policy within the PKI, that much care must be taken in creating it. Conversely, the lack of a well-formed CP will have a big influence on security, resulting in possible compromise of electronic transactions and loss of trust.

Organizations that fail to provide a CP and a CPS will be unable to be involved in secure communications, and cannot be included in a cross-certification environment [2].

Methodology

In our attempt to formalise the CP, we have defined certain conventions to be used as guidelines during the process. This will lead to a common approach to the formalisation of any CP. The adopted conventions are:

1. The framework defined in RFC 2527 is to be used, which outlines the contents of a set of provisions of the CP in terms of eight primary components. These components are :

    - Introduction;
    - General Provisions;
    - Identification and Authentication;
    - Operational Requirements;
    - Physical, Procedural, and Personnel Security Controls;
    - Technical Security Controls;
    - Certificate and CRL Profile; and
    - Specification Administration.

2. Where a phrase is used as a section heading, the XML tag consists of those same words, without spaces, and with the first letter of each word (except the first) being capitalised. Thus, for example, the phrase “Community and Applicability” becomes “`communityAndApplicability”.`

3. We use a fixed value for any expression clause. The value is in string format, space separate between words. An example is `"the sections of EuroPKI-CP".`

During this work we have chosen an example CP to illustrate the formalisation process, and the “EuroPKI Certificate Policy” has been selected for this purpose. There are two reasons for this choice:

1. EuroPKI, which issued this policy, is non-profit organization.
2. It has been accepted as a Europe-wide standard CP.

Formalisation process

Our formatisation focuses on representing the semantic meaning of a CP, and we will perform this process using the XML language. As stated in [6] eXtensible Markup Language (XML) is a language used to describe data structures. XML provides the means for pieces (elements) of data to be labelled. Similar work in the PERMIS system, which uses XML to encode policies for stating the rules for assigning roles to users, and permissions to roles, has been reported by Chadwick [7].

Within each CP, there are a number of statements which define what relying parties may expect of certificates issued under the policy. These statements contain words indicating the level of obligation placed on holders of certificates issues under the policy. These are listed in the introduction of EuroPKI CP as:
MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY and OPTIONAL. The meaning of these so-called "obligation words", is given in RFC 2119 [ref]. In detail, the meaning of these obligation words is as described in [8]:

1. **MUST –** (alternatively "REQUIRED" or "SHALL") means that the statement is an absolute requirement of the specification.
2. **MUST NOT –** ("SHALL NOT") means that the definition is an absolute prohibition of the specification.
3. **SHOULD** – ("RECOMMENDED") means that there may exist valid reasons in particular circumstances for ignoring this particular item, but the full implications must be understood and carefully weighed before doing so.
4. **SHOULD NOT** – ("NOT RECOMMENDED") means that there may exist valid reasons in particular circumstances when this particular statement is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any such statement.
5. **MAY** – ("OPTIONAL") means that the item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option MUST be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides.)

We have used these obligation words as a measure of the importance of different CP sections, and as a result, we have introduced in our XML description, a tag corresponding to each of the obligation words:

1. **REQUIREMENT:** MUST, REQUIRED and SHALL.
2. **PROHIBITION:** MUST NOT and SHALL NOT.
3. **PREFERRD:** SHOULD and RECOMMENDED.
4. **NOT PREFERRD:** SHOULD NOT and NOT RECOMMEND.
5. **POSSIBLE:** MAY and OPTIONAL.

These five words will appear as main sub-sections under each section of CP, hereafter referred to as **obligation title**. All the obligations will be listed under one of the obligation title according to their importance as defined in the CP. For example, the obligation section has sub-sections in CP called CAObligations. The CAObligations section will look like in XML as the following:

```
<obligations>
        <CAObligations>
                <requirement>
                </requirement>

                <prohibition>
```

```
            </prohibition>

            <preferred>
            </preferred>

            <notPreferred>
            </notPreferred>

            <possible>
            </possible>
        </CAObligations>

</obligations>
```

As illustrated in the above example, the internal sections headed by the obligation title show the obligations upon the CA; therefore if there is a required action that should be taken by the CA, it will be listed under the requirement section. Each obligation title contains a description of what is required under this clause. The presentation of these obligations will be in the following format:

<CP section title tag>
    <Entity name tag>
        <Obligation title tag>
            <Description of the obligation>

The description section of the obligation will start with a *verb* which describes the action to be taken to perform the obligation, which is then followed by operands and other values. For example, one of the CA obligations states that “CA SHALL operate a certification authority service”, and XML representation for this will be as the following:

```
<obligations>
      <CAObligations>
            <requirement>
                  <operate>
                        <certificationAuthorityService/>
                  </operate>
            </requirement>
      </CAObligations>
</obligations>
```

We observe that the description section of the obligation starts with the verb “operate” under requirement section, and the operand is “certificationAuthorityService”. In the case where there is another obligation which acts as a constraint on the, we can use the obligation tag as a sub-section under the verb. As an example, let us recall the previous example where an obligation was put on the CA to operate a certification authority service. There is a further obligation on this service:

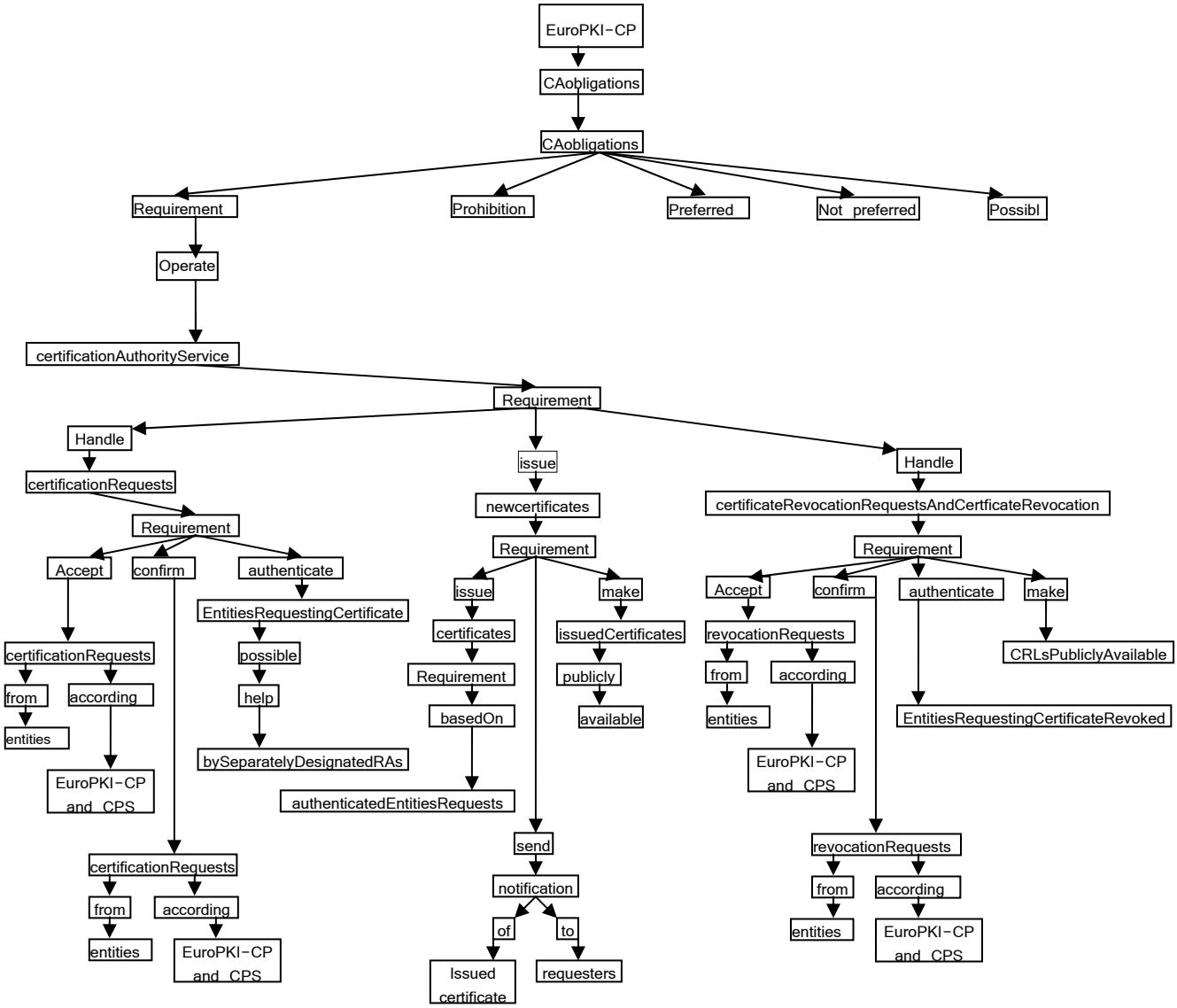
Handle certificate request

which represents an additional obligation on the operation of the previously mentioned certification authority service.

This obligation will be written with the previous representation as the following:

```
<obligations>
      <CAObligations>
            <requirement>
                  <operate>
                        <certificationAuthorityService/>
                     <requirement>
                        <handle>
                           <certificateRequests/>
                        </handle>
                     <requirement>
                  </operate>
            </requirement>
      </CAObligations>
</obligations>
```

We can continue using this technique to a greater depth if there are inner obligations. This way of presentation defines a tree structure in which we lay down all the entities of the XML representation. The tree root is the name of the CP (EuroPKI-CP in this example - we add CP to distinguish it from the CPS), and the next level will be the CP section names. The third level from top will be the obligation titles and followed it by verb tags. The rest of levels will be either the description of the obligation or sub-obligation titles. The following diagram explains the tree presentation for the obligation section about the CAobligations tag:



Comparison Result

When the activity of creating the formal representation of the CP is complete, it may be used to compare different CPs to measure the trustworthiness of the target from the point of view of the relying party. We have to bear in mind that “trust” as a concept can only be interpreted in the context of the particular task or service which the relying party is expecting the target to provide. In that sense, the result of the comparison of two policies may depend upon a part of each policy only.

Initially, the comparison will be done manually; i.e. a visual comparison of the two CPs will be performed. Our aim is, however, to be able to perform an automatic comparison.

Taking account of this, we may consider the result of two policies as yielding one of four situations:

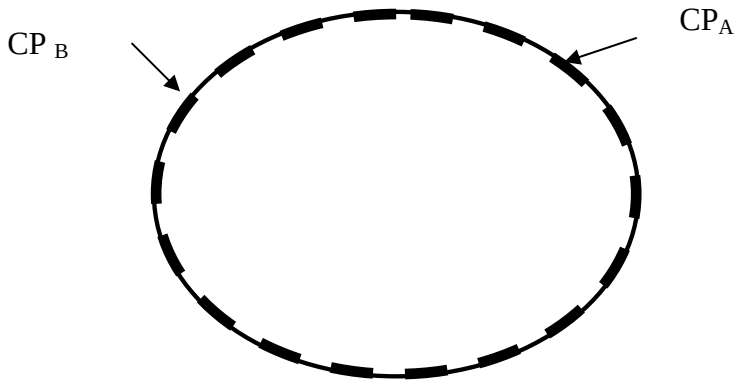
1. Absolute overlap.
2. Inclusive overlap.

3. Partial overlap.
4. No overlap.

More illustration of the four cases is in the following paragraphs:

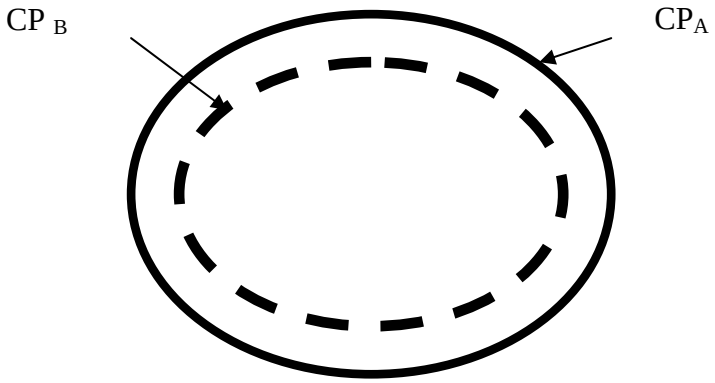
Absolute overlap

The relevant sections of each CP impose the same constraints on the certificates within the two domains. Thus, every obligation, prohibition, etc. in policy A (CP_A) is matched by a corresponding obligation, prohibition, etc. in policy B (CP_B). In this case, certificates in the respective domains should always have trust in each other. This is represented in the following diagram, in which CP_A is represented by the solid line and CP_B by the dashed line:



Inclusive overlap

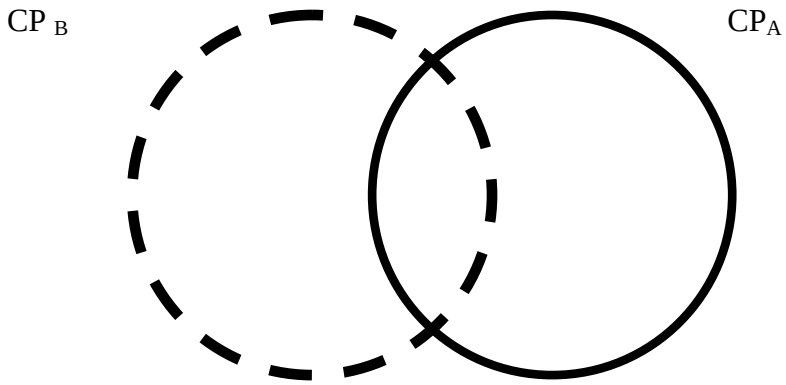
In this case one of the CPs is "included" in the other; in other words certificates in one domain can always trust certificates in the other domain, but not vice versa. In the following diagram, which depicts two policies, CP_A and CP_B again, certificates in domain B can always be trusted by domain A, but not (necessarily) the other way around. This circumstance may arise (for example) as a result of requirement in policy B being only optional in policy A.



The inner CP (CP_B) will have the same trust of the outer one (CP_A); therefore CP_B will trust the CP_A because all its policy is appears in the CP_A. If we want to relate that to the authorization concept, CP_B has less authorization authority than CP_A which means there are more constrains on CP_B than CP_A.

Partial overlap

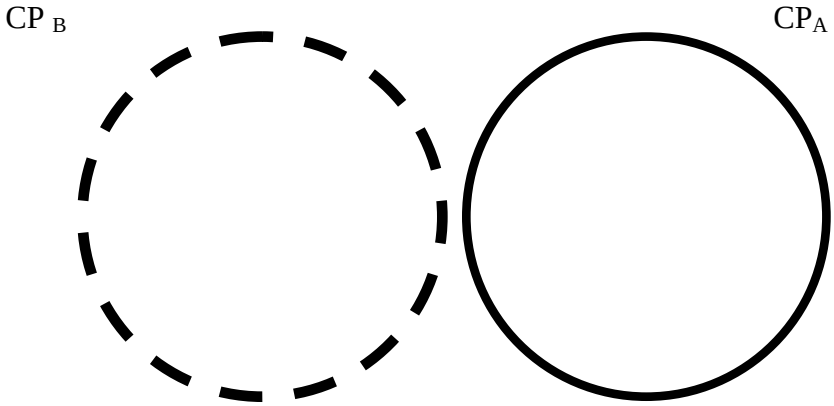
Partial overlap means that the policies in charge are having an intersection between them. The following diagram explains this case:



In this case CP $_B$ intersects with CP_A, indicating that the two policies have some aspects in common, but each has constraints which do not apply to the other policy. Both can trust each other, but only in certain activities.

No overlap

Both policies are disjoint which leads to the conclusion that no trust path exists. This situation may be represented by the following diagram:



Conclusion

We have set out in this paper to describe the first steps towards our goal of being able to establish, automatically, trust relationships between otherwise untrusted entities. Clearly we have only taken a small step in this direction, but we see the use of XML to describe not only the syntax, but also the semantics, of a Certification Policy as a promising way to approach the problem of providing a formal description of the policy. Once this has been achieved, it will be possible to make use of a wide range of XML-based tools to compare policies, and to decide

whether or not an appropriate trust relationship can exist between relying party and target, given the policies to which they both (respectively) are required to adhere.

References

1. S. Chokhani, O.S.S., Inc., et al., *Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework.* 2003.
2. Austin, T., *PKI*. A Wiley Tech Brie f, ed. M. Hendrey. 2001: John Wiley & Sins, Inc. 270.
3. Ellison, C. and B. Schneier, *Ten Risks of PKI.* p. 8.
4. Carlisle Adams, S.L., *Understanding PKI: Concepts, Standards, and Deployment Considerations, Second Edition*. SECOND ed. 2003: Addison Wesley. 352.
5. Batarfi, O. *Certificate Validation in Untrusted Domains*. in *On The Move to Meaningful Internet Systems 2003: OTM 2003Workshops*. 2003. Catania, Sicily, Italy: Springer-Verlag Heidelberg.
6. (W3C), t.W.W.W.C., *Why use XML?*
7. D.W.Chadwick and A. Otenko, *RBAC POLICIES IN XML FOR X.509 BASED PRIVILEGE MANAGEMENT.* p. 15.
8. Bradner, S., *Key words for use in RFCs to Indicate Requirement Levels.* 1997.