Liberty Trust Models

Draft Version 1.0-14

13 April 2003

Editor:

John Linn, RSA Laboratories

Contributors:

Sharon Boeyen, Entrust
Gary Ellison, Sun Microsystems
Niina Karhuluoma, Nokia
William MacGregor, Schlumberger
Paul Madsen, Entrust
Senthil Sengodan, Nokia
Serge Shinkar, Communicator

Abstract:

This specification is non-normative. Its purpose is to provide guidance on a variety of models that can be applied to establish trust among Liberty components, discussing their characteristics and implications. Its emphasis is on authentication and business relationships among components performing Liberty protocols, rather than on other components within supporting infrastructures. The discussion considers Liberty Phase 1 circle-of-trust environments as well as extended models appropriate to support the inter-IdP interaction requirements established within Phase 2. Its intended audience includes designers of Liberty protocols and deployers of Liberty implementations.

Notice

Liberty Alliance Project
Licensing Administrator
c/o IEEE-ISTO
445 Hoes Lane
Piscataway, NJ 08855-1331, USA
info@projectliberty.org

55 # Document History

Rev	Date	By Whom	Description
00	24-Jun-02	John Linn, RSA Laboratories	Initial draft (annotated outline only)
01	17-Jul-02	John Linn	Added taxonomy section, identified contributors per Vancouver meeting discussion, various other edits
02	6-Aug-02	John Linn	Renamed taxonomy table columns, extended introduction, added definitions, inserted contributions for Common Trust and Trust Establishment Mechanisms sections
03	12-Aug-02	Paul Madsen	Modified definitions, added new content for sections
04	3-Sep-02	John Linn, Niina Karhuluoma	Miscellaneous editorial changes, updated Trust Establishment Mechanisms section
05	3-Sep-02	John Linn, Serge Shinkar	Added contribution for Pairwise Trust section
06	9-Sep-02	John Linn, Bill MacGregor	Deleted Unsecured Operation section, began Comparison Among Models, integrated revised Common Trust section.
07	Sep-02	John Linn, Paul Madsen	Restructuring per Bedford meeting, inclusion of figures in Sec. 2.
08	7-Oct-02	John Linn, contributors	Condensed and added examples, other editing and restructuring
09	11-Oct-02	John Linn, Paul Madsen	Various corrections and additions, including new example
10	8-Nov-02	John Linn	Various edits
11	26-Nov-02	John Linn	Revised Sec. 4.2.4.
12	17-Dec-02	John Linn, Paul Madsen	Moved delegated trust example to 5.2, updated Section 9, removed references to Authentication Domain
13	10-Mar-03	John Kemp	Revised logo; added legal notice; edited footer & header.
14	13-April-03	Tom Wason	Added legal notice, abstract.

56

Table of Contents

97

98 # Figures

109

110

1 Introduction

This specification is non-normative. Its purpose is to provide guidance on a variety of models that can be applied to establish trust among Liberty components, discussing their characteristics and implications. Its emphasis is on authentication and business relationships among components performing Liberty protocols, rather than on other components within supporting infrastructures. The discussion considers Liberty Phase 1 circle-of-trust environments as well as extended models appropriate to support the inter-IdP interaction requirements established within Phase 2. Its intended audience includes designers of Liberty protocols and deployers of Liberty implementations.

The models identified can be applied as parallel alternatives, and can be hybridized with one another. Through use of different models, it is possible for a given entity to obtain trust in other entities through different means and to different levels. While this document discusses and compares characteristics of the different models, it does not attempt to specify a universal strength ordering among them.

The document's structure is as follows. Following this Introduction, Section 2 presents a taxonomy to organize discussion of different alternatives for trust establishment, defines relevant terms, and discusses a conceptual procedure for trust-related processing. The next sections present examples of various models for establishing business trust between Liberty entities:

Section 3 considers trust establishment on a pairwise basis, as is applied in Liberty's Phase 1 circles of trust.

Section 4 considers the use of active brokering entities as intermediaries to support transactions involving multiple IdPs. This corresponds to the introducer model contemplated for support in Phase 2.

Section 5 considers interactions among Liberty components in a mode where interoperability is enabled through the use of a common authentication infrastructure, and on business-level trust gained through that infrastructure's administrative and enrollment processes, rather than on business agreements established independently of the authentication infrastructure.

Within each of sections 3-5, alternative approaches for establishment of authentication trust are considered. Section 6 compares the presented models. Section 7 provides a comparative overview of cryptographic trust establishment methods, and Section 8 discusses aspects of their application in the context of Liberty. Section 9 considers the prospect of metadata-based facilities for automated establishment of trust paths. Section 10 provides references.

2 Definitions, Taxonomy, and Conceptual Processing Procedure

This section defines relevant terms as used within this document, establishes a taxonomy to structure the discussion of different trust model alternatives, and describes a conceptual processing procedure supporting the determination of trust among communicating Liberty entities.

2.1 Definitions

Authentication Enrollment Agreement: An agreement between an authentication infrastructure provider and an entity registering in order to be authenticable through that provider's services. For the case of PKI, where a CA acts as the infrastructure provider, provisions of an authentication enrollment agreement will normally correspond to aspects of the CA's applicable Certification Practice Statement (CPS).

Brokered Trust: Brokered Trust describes the case where two entities do not have direct business agreements with each other, but do have agreements with one or more intermediaries so as to enable a business trust path to be constructed between the entities. The intermediary brokers operate as active entities, and are invoked dynamically via protocol facilities when new paths are to be established.

Business Agreement: An agreement among parties providing the commercial prerequisites that the parties require in order to engage in business transactions. Such agreements may be negotiated bilaterally, or may be presented unilaterally by an issuer and accepted by a recipient.

Business Anchor (BA): A business anchor represents an entity with which its holder has a direct business relationship. If an entity requires direct business agreements in order to interoperate with other peers, those peers must be listed in the entity's business anchor list. If an entity accepts indirect business agreements in order to interoperate with peers, its business anchor list must identify an intermediary through which a business agreement path can be derived leading towards those peers. A Business Anchor entry may be qualified by the associated business agreement and other potential information such as the subset of the TAL that applies to it.

Business Anchor List (BAL): Entities requiring business agreements in order to interoperate with other entities will maintain business anchor lists identifying the entities with which direct business trust relationships have been established. In some cases, these lists may correspond with the trust anchor lists used to represent entities trusted for authentication purposes; nonetheless, their semantics are distinct. Normally, entries in business anchor lists will be added and removed only as a result of explicit administrative action, reflecting changes to business agreements with direct partners.

Community Trust: Community Trust applies when the business trust between a pair of entities is derived from their enrollment in a common authentication infrastructure and acceptance of its practices, without reliance on other business agreement paths. As such, the entities' mutual trust in a business sense is based on their membership in a community constructed and linked for authentication purposes.

Direct Trust: Direct Trust is obtained when communicating entities hold each other's keys within their TALs, so that their validity is established without reliance on intermediaries.

Indirect Trust: Indirect Trust is obtained when communicating entities ascertain the validity of each others' keys based on pre-existing trust established with an intermediary, as represented by a trust anchor.

Pairwise Trust: Pairwise Trust describes the case where two entities have direct business agreements with each other.

Trust Anchor (TA): A trust anchor represents an entity and key that the anchor's holder has determined to trust directly for cryptographic authentication purposes. In some cases, the TA is qualified by an associated agreement between the represented entity and the TA's holder. This qualification may affect the set of entities that can be authenticated through the TA.

Trust Anchor List (TAL): Entities accepting cryptographic authentication of other entities will maintain trust anchor lists, identifying the entities and associated keys that they trust for authentication purposes and upon which validations will be based. In some cases, these lists may correspond with the business anchor lists used to represent entities trusted for business purposes; nonetheless, their semantics are distinct. Normally, entries in trust anchor lists will be added and removed only as a result of explicit administrative action reflecting changes in trust relationships.

2.2 Taxonomy

When issues of trust in distributed systems are discussed, confusion often results from ambiguities concerning particular aspects for which entities are to be trusted. Figure 1 distinguishes two dimensions of trust, dimensions introduced for clarification purposes.

The figure's columns distinguish the types of cryptographic infrastructures applied to support authentication among components, ensuring that the identities of named entities are authentic. Proceeding along the horizontal axis, we consider direct authentication (pairwise exchange of cryptographic keys), and indirect authentication (facilitated through the involvement of off-line or on-line trusted intermediaries); since Liberty specifications require the use of authentication facilities, no column is provided to represent unauthenticated cases. In the indirect case, it is common for participants to accept authentication enrollment agreements issued unilaterally by the authentication infrastructure providers; these help to ensure procedural integrity of the infrastructure, but are distinct from business-level agreements executed between Liberty participant entities with the purpose of supporting Liberty-enabled services.

The figure's rows distinguish among the types of business agreements established between participants as a basis to support transactions. Proceeding along the vertical access, we consider direct agreements (exchanged between the

participants), indirect agreements (facilitated by business intermediaries), and the absence of business agreements linking participants. Generally, it is assumed that business agreements will be negotiated between entities on a bilateral basis[1].

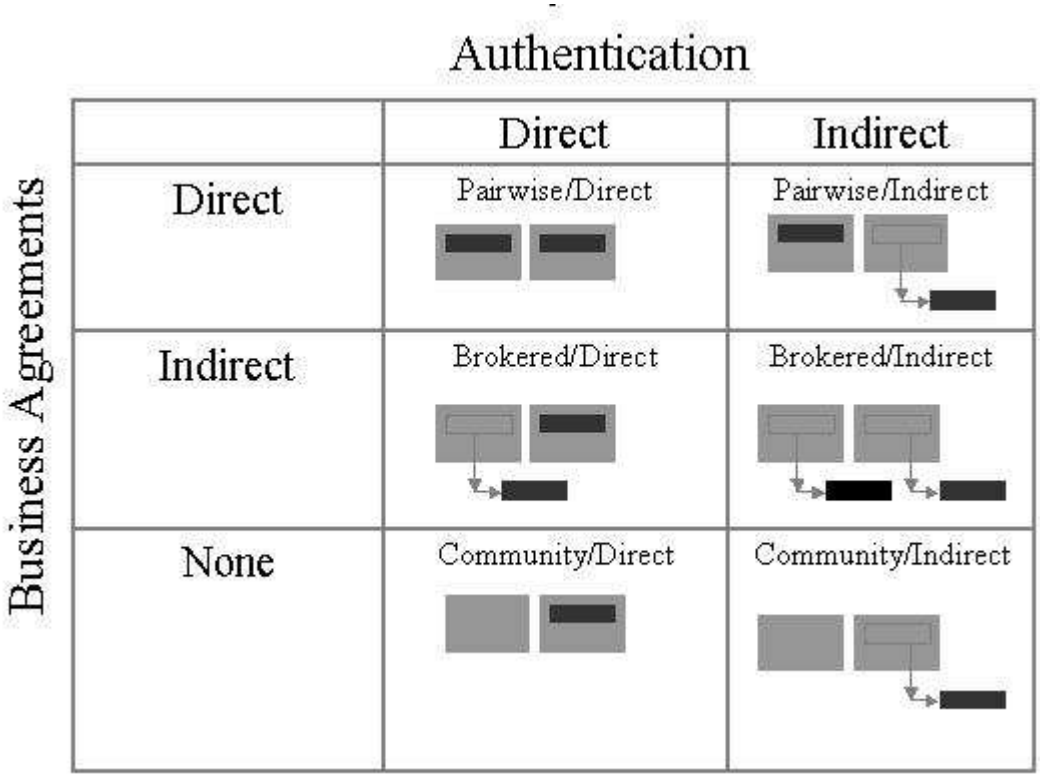


Figure 1: Trust Model Taxonomy

As the figure's structure suggests, approaches providing authenticated naming may vary independently from approaches providing business-level trust. Titles within the figure's cells correspond to subsequent sections within the document, where supporting discussion will be provided. Within the cells, graphic elements represent applicable contents of the BAL (on left) and TAL (on right) corresponding to that case. In each graphic, the business entity in question is identified by a black horizontal rectangle. The cells indicate whether business agreement and authentication trust paths are direct, indirect, or absent using the following graphic conventions:

- For a direct path, by illustrating the black rectangle representing the business entity within either or both of the lists representing BAL and TAL,
- For an indirect path, by illustrating the black rectangle outside the applicable list but reachable through a link from some other entity (represented by a gray horizontal rectangle) located in the applicable list, or
- For an absent path, by the absence of a black rectangle or link thereto within the applicable list.

[1] It has been suggested that certain intermediaries might provide unilateral business agreements to participants, facilitating establishment of indirect business agreement paths. This prospect requires further study, and may comprise a subcase of the Indirect Business Agreement table row.

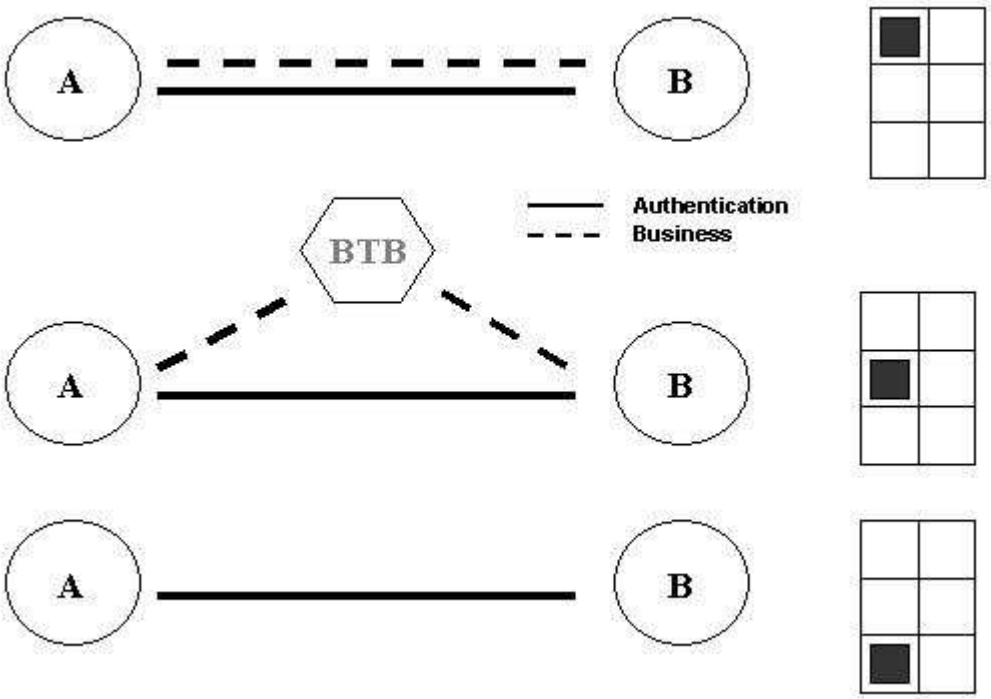


Figure 2: Direct Authentication Models

Figure 2 illustrates the three models based on direct authentication, associating them with their corresponding cells in Figure 1.

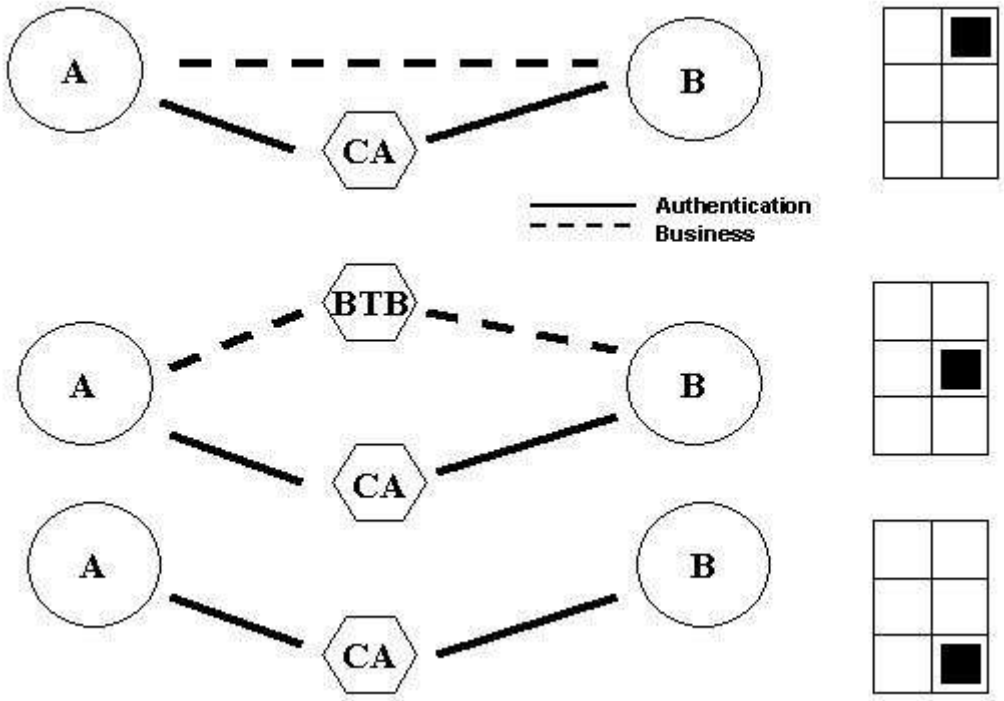


Figure 3: Indirect Authentication Models

Figure 3 illustrates the three models based on indirect authentication (using a PKI CA as an example intermediary), associating them with their corresponding cells in Figure 1.

2.2.1 Characteristics of Pairwise Trust Models

Liberty Phase 1 circles of trust exemplify Pairwise Trust models. These models afford strong trust in a business sense, but have relatively limited scalability. Cryptographic authentication within these models may be based on pairwise out-of-band exchange of shared secret keys or public-key certificates, in conjunction with business/legal agreements; this exemplifies the Pairwise/Direct case. It is also possible for Phase 1 entities to authenticate each other via an infrastructure involving intermediary entities (e.g., PKI CAs); such infrastructure usage exemplifies the Pairwise/Indirect case.

In the Pairwise Trust models, relationship and business trust between all interoperating participants is exclusively governed by signed business agreements. The strong trust established via business agreements is not technically extendable which results in the forming of closed communities.

The determination of the level of trust in these communities is managed by business agreements, which generally take precedence over trust established via authentication infrastructure. A new entity may not interact within such a community without first entering into a business agreement with the existing participants and being added to the BAL.

2.2.2 Characteristics of Brokered Trust Models

In Liberty's Brokered Trust models, active intermediaries are invoked and involved when federation and/or authentication transactions span multiple administrative domains. These approaches constrain the set of components that must be involved in interdomain trust management, but require the use of additional protocol facilities beyond those defined in Phase 1. Further, Brokered Trust models depend on availability of appropriate intermediaries in order to construct a path to federate a user's relationship and/or to authenticate a particular session.

As an example situation Brokered Trust may be applicable, an SP associated with IdP A receives an assertion to be processed from IdP B, with which it shares no prior relationship. The assertion may be an authentication assertion, a federation request, or an attribute assertion (in examples we will refer to authentication assertion but it should be understood that this is merely representative of a more general message). The SP must decide whether to trust IdP B's assertion. Overall trust is made up of the combination of business trust, based on direct/indirect business agreements, and authentication trust, based on direct/indirect cryptographic authentication infrastructure.

In Brokered Trust models, there is no direct business trust; i.e., the remote IdP is not directly represented in the BAL of the local SP. However, there must be at least one entity represented in the local SP's BAL that can act as an intermediary for the local SP. Two subcases are possible, depending on the business agreements involved:

1. In the first subcase, it is assumed that the business agreement between the local SP and the intermediary explicitly identifies the remote IdP as an entity with which the intermediary has a direct business agreement and that this agreement can be used transitively with the agreement between the local SP and intermediary. This model enables the formation of a business agreement chain that satisfies the business needs of the local SP such that it may place trust in an assertion received from that remote IdP. No dynamic update protocol for the set of such remote entities per local business agreement is anticipated. Requiring explicit identification of remote entities with which an intermediary has direct agreements limits the length of possible chains of business agreements to two. If longer business agreement chains become necessary, then some repository service would be required to enable identification of remote business agreements that can be used as links in a path between two communicating entities.
2. In the second subcase, the business agreement between the local SP and the intermediary places broader trust in the intermediary, allowing it to act as an agent for the SP and to establish paths to other parties without requiring that those parties be identified in advance in the business agreement between the local SP and the intermediary. This subcase can allow business trust to be established more dynamically and to a broader range of peers.

In some cases the establishment of indirect business trust with a remote entity will not require any additional anchors to be added to the BAL. In these cases, an entity that is already represented in that list acts as the intermediary to broker business trust with the remote entity. In other cases, if no such intermediary is listed in the local entity's BAL,

an additional anchor will need to be added. This additional anchor could be either another intermediary or a Liberty provider directly (implying that subsequent transactions would be Pairwise Trust). It is assumed that the addition of an entity to the BAL is a serious decision and is not undertaken without ensuring that the new entity is properly vetted in accordance with security, operational, and business policies.

2.2.3 Characteristics of Community Trust Models

Community Trust models presume neither direct nor indirect business agreement paths between communicating entities. Instead, they rely on shared membership in a community defined by a cryptographic trust establishment infrastructure as a basis to enable communication between entities for purposes of federation and/or authentication. Public Key Infrastructure (PKI), Kerberos realms and inter-realm relationships, and PGP webs of trust represent examples of available trust establishment infrastructures. In these models, a trust establishment infrastructure is used in lieu of direct business agreements or intermediary entities acting as trust brokers.

When Community Trust applies between a pair of entities, trust establishment is not based on identification of BAL entries corresponding to the communicating peers. Instead, entries within the entities' TALs identify an authentication trust path. Aspects of that authentication trust path are governed by the infrastructure's Authentication Enrollment Agreements, and can be applied as a basis to achieve business-level trust.

Hybrid models are also possible, where aspects of business-level trust obtained through the agreements of the Pairwise/Indirect or Brokered/Indirect models are complemented with additional aspects obtained through participation in a common trust establishment infrastructure. Trust establishment infrastructures are essential to support these models for authentication purposes, and can be leveraged to offer additional value for business purposes.

2.3 Conceptual Processing Procedure

For an entity A to determine whether a suitable basis exists to carry out trusted transactions with another entity B, it operates on the following data:

- B's identity
- A's BAL
- A's TAL
- A's operational policies, indicating the types of paths it accepts

This section describes the necessary processing at a conceptual level; it is intended for descriptive purposes, not to constrain individual implementations.

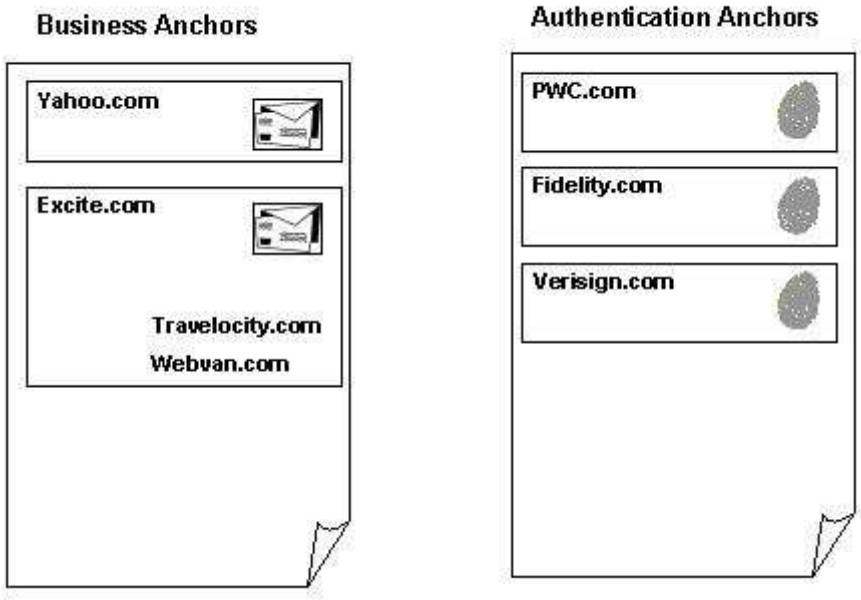


Figure 4: Example BAL and TAL

The process of validating an authentication trust path begins by determining whether A's TAL contains an entry for B. If so (e.g., in the Figure 4 example, if B's identity is Fidelity.com), Direct Trust applies, and A possesses the key required to authenticate messages and/or connections received from B. If not, A must determine whether one or more of the entries in its TAL enables it to construct an authentication path to B. Path construction and validation algorithms are well known, though their specifics vary for different types of infrastructures. If an authentication path can be constructed and validated, Indirect Trust applies, and A can traverse that path to obtain the key required to authenticate messages and/or connections received from B. If no path can be constructed, then A is unable to authenticate B and the Liberty-specified prerequisites for communication cannot be satisfied. Assuming that A holds or obtains the key necessary to authenticate B, it applies it as it processes B's communications, in order to validate B's authenticity.

The process of validating a business agreement path begins by determining whether A's BAL contains an entry for B. If so (e.g., in the Figure 4 example, if B is Yahoo.com), Pairwise Trust applies. If not, A must determine whether one or more of the entries in its BAL enables it to construct a business agreement path to B. It appears that the process of constructing business agreement paths has received less study in an algorithmic sense than that of constructing authentication paths, so its procedures may often be more ad hoc in nature. If a business agreement path can be constructed (e.g., in the Figure 4 example, a path to Travelocity.com via Excite.com), Brokered Trust applies. If not, no business agreement applies between A and B, and any transactions must be carried out based on a Community Trust model.

At this stage in the process, A has identified the "shortest" applicable type of authentication path (Direct or Indirect) and of business agreement path (Pairwise, Brokered, or Community) reaching to B. It must now determine whether these paths satisfy its policies and, if so, whether they dictate any limits or constraints on the transactions that it will be willing to undertake with B; a peer reachable via Pairwise Trust, e.g, might be accorded broader rights than one reachable only at the Community Trust level.

Note that some or all of A's BAL, TAL, and policy data may be kept confidential to A; it is not assumed that their contents must be shared with B in order to enable transactions to proceed. It is possible, however, that sharing of some of this information may simplify the task of identifying a suitable authentication and/or business agreement path.

3 Pairwise Trust Model Examples

3.1 Pairwise/Direct Model

In this model, an entity receives an assertion from another entity in its local circle of trust with which it has a direct authentication trust established and business trust enabled. This direct authentication trust can be established by exchanging keys using a means that is out-of-band with respect to Liberty specifications. The assertion recipient has the assertion's originator in its TAL and BAL.

3.1.1 Example

As an example, an SP signs a Business Agreement with an IdP as part of which it agrees to use the services of the IdP to authenticate its users. The SP adds the IdP to its BAL. The SP and IdP also set up a mechanism to exchange keys on a periodic basis. For each period, the SP picks up the key and stores the key in its TAL. The IdP sends signed assertions to the SP, and the SP uses the key it obtained in order to authenticate the IdP.

3.2 Pairwise/Indirect Model

In this model, an entity receives an assertion from another entity with which it does not have direct authentication trust established. As such, the remote entity's key is not present in the local entity's TAL. The receiving entity does have a Business Agreement with the sending entity and hence the sending entity is present in its BAL.

3.2.1 Example

Considering a PKI-based example, an SP receives a signed authentication assertion from an IdP. Business trust exists between the two parties. If there is a valid certification path from one of the CA's in the local SP's TAL through a chain of intermediate CA's to the IdP's certificate then the signature on the assertion can be trusted.

3.3 No Authentication Infrastructure

This case is not conformant to Liberty specifications and is not recommended for operational use, but is described briefly in the interests of clarification and completeness. Here, there exists no Authentication Infrastructure between the SP and IdP but the IdP and SP have a business agreement. This is likely to be a temporary state and not a likely permanent method unless one of the parties decides to forego verification since it considers the services it provides of low value and not worth securing. This can occur temporarily when existing infrastructure becomes unavailable due to it being compromised or broken. Hence the SP will not be able to authenticate the IdP and will not be able to validate the assertions. The SP may determine that such an assertion can be used to provide service as the level it would be offered to users anonymously or with unsigned authentication assertions from an IdP.

4 Brokered Trust Model Examples

Each of the following subclauses describes a distinct model for authentication trust that is used in conjunction with indirect business trust. These authentication trust models include direct authentication trust, indirect authentication trust and no authentication trust.

4.1 Brokered/Direct Model

In this model, the local entity that receives an assertion from a remote entity has direct authentication trust established with that remote entity. As such, the remote entity's key is included in the local entity's TAL. Because this model deals with indirect business trust, the remote entity is not represented in the local entity's BAL.

4.1.1 Example

Considering an example, a local SP receives a signed authentication assertion from a remote IdP. The local SP has a local IdP in its BAL. The business agreement between these two does not explicitly state that the local IdP has a business agreement with the remote IdP. The local IdP provides business trust only among the SPs with which it is affiliated. Another IdP does have a business agreement with the remote IdP and offers to act as an intermediary for the local SP. Such an IdP may have as its primary role that of an intermediary broker. Many Liberty entities could make use of such intermediaries to establish business agreement chains with remote entities. Because of the generic nature that such business agreements would likely have, it may be that the services of such brokers would be used primarily for lower value business transactions than those where a local IdP is used as the intermediary for business trust. The indirect business agreement chain includes the business agreement between the local SP and generic remote IdP broker, as well as the business agreement between the remote IdP broker and the remote IdP that initiated the authentication assertion.

Because the local SP already has the key of the remote IdP that issued the authentication assertion in its TAL, no intermediary is required for cryptographic authentication trust.

The SP has established indirect business trust and direct authentication trust. Together these enable overall trust to be placed in the authentication assertion received from the remote IdP. In this example, an additional business anchor for the generic remote IdP broker must be added to the local SP's BAL. No new trust anchors need to be added to its TAL.

4.2 Brokered/Indirect Model

In this model, the local entity that receives an assertion from a remote entity does not have direct authentication trust established with that remote entity. As such, the remote entity's key is not present in the local entity's TAL. Because this model deals with indirect business trust, the remote entity is also not represented in the local entity's BAL. The examples vary in the authentication technologies they employ, and in whether their infrastructure components are involved actively or passively in the authentication process. They include a PKI case, a Kerberos case, and a case where SAML assertions are used as a basis for establishment of trust in a remote IdP.

4.2.1 Example 1: PKI

To facilitate comparison of the examples in Sections 4.2.1-4.2.3, the same basic scenario is used. A local SP receives a signed authentication assertion from a remote IdP. In this example, indirect business trust is established using one of the techniques described in the previous section.

Public-key infrastructure (PKI) is the authentication infrastructure in this example. The local SP has in its TAL the key of the CA that issued a public-key certificate used to verify the digital signature of the local IdP. If this same CA issued a certificate to the remote IdP, then the signature on the authentication assertion issued by the remote IdP can be verified using that same trust anchor. Even if the same CA did not issue a certificate to the remote IdP, if there is a valid certification path from the local trust anchor, through one or more intermediate CAs, to the certificate issued by some other CA to the remote IdP, the signature on the authentication assertion can be trusted.

The SP has established indirect business trust and indirect authentication trust. Together these enable overall trust to be placed in the authentication assertion received from the remote IdP. Depending on whether indirect business trust was established as in example 1 or example 2 of 4.1, the SP may/may not need to add a new anchor to its BAL. Because one of the CAs whose key is already in the local SP's TAL either issued a certificate directly to the remote IdP or issued a certificate to an intermediary CA that is used to form a valid certification path to the remote IdP, no new anchor needs to be added to the local SP's TAL.

4.2.2 Example 2: Kerberos

As with the previous example, a local SP receives a signed authentication assertion from a remote IdP. In this example, indirect business trust is established using one of the techniques described in the examples in 4.1.

Kerberos is the indirect authentication infrastructure in this example. The local SP's TAL contains the symmetric key that it shares with its local KDC but does not contain a symmetric key for the remote IdP. In order for the local SP to place authentication trust in the signed (HMACed) assertion from the remote IdP; that remote IdP will have to demonstrate that it was trusted (directly - if it shares the KDC with the local SP or indirectly – if it belongs to another Kerberos realm). The remote IdP is able to demonstrate this trust by proving that it has possession of a short-lived symmetric key that was also delivered to the remote SP encrypted by the long-lived symmetric key shared between the local SP and its KDC.

The SP has established indirect business trust and indirect authentication trust. Together these enable overall trust to be placed in the authentication assertion received from the remote IdP. Depending on whether indirect business trust was established as in example 1 or example 2 of 4.1, the SP may/may not need to add a new anchor to its BAL. If inter-realm Ticket-Granting Tickets (TGTs) traversing the path from the remote IdP's KDC to the local SP's KDC are obtained and used, the local SP can authenticate the remote SP's communications without adding a new TA to its TAL.

4.2.3 Example 3: SAML

Just as SAML Authentication Assertions enable indirect authentication trust between Principals and SPs (with the IdP playing the role of TTP), SAML can play a similar role enabling indirect authentication trust between local SPs and remote IdPs.

Logically very similar to the Kerberos example above, the local SP will be able to derive trust in the remote IdP through the active involvement of a TTP playing the logical role of the Kerberos KDC, i.e. issuing authentication tokens to the remote IdP that will be trusted by the local SP because of the trust the SP has in the TTP. While in the previous example these authentication tokens are binary Kerberos tickets, in this example they are SAML Authentication Assertions.

The local SP's TAL either directly contains the public key of the TTP or contains the key of a CA that has issued a certificate to that TTP such that the SP can verify SAML Authentication Assertions signed by the TTP's associated private key. By definition, the local SP's TAL does not contain a key for the remote IdP.

The remote IdP authenticates to the TTP (SAML Authentication Authority) in order to be issued a SAML Authentication Assertion, signed by the TTP. The remote IdP then presents the SAML assertion as a 'letter of introduction' to the local SP. The SAML Authentication Assertion will likely contain keying information encrypted for the local SP. The remote IdP is able to demonstrate its trustworthiness to the remote SP by proving that it has possession of the same key. This shared secret will allow the remote IdP and the SP to securely establish a session key for their subsequent transaction. Following completion of this processing, the SP has established indirect business trust and indirect authentication trust. Together, these enable overall trust to be placed in the authentication assertion received from the remote IdP.

Like the Kerberos example, this use of SAML relies on a TTP playing an active role in the derivation of indirect trust through the real-time issuance of authentication tokens. Unlike the Kerberos example, this SAML scenario depends on asymmetric cryptography. The authenticity of the SAML Authentication Assertions is determined by private key signatures rather than a secret key MAC.

4.3 No Authentication Infrastructure

This case is not conformant to Liberty specifications and is not recommended for operational use, but is described briefly in the interests of clarification and completeness. In some situations, an entity in one domain may need to establish trust with an entity in another domain, even though there is no supporting cryptographic authentication infrastructure (direct or indirect) in place. For example, in a situation where one company purchases another, the subsumed organization may inherit the business agreements of the parent company but not yet have cryptographic

authentication infrastructure established to support those business agreements. Given the same scenario as above, where an SP in the subsumed company receives a signed authentication assertion from an IdP in another domain, the SP may be able to establish indirect business trust, but no authentication trust. As such, the local SP may still be able to use that authentication assertion, although the level of overall trust in that assertion would be reduced. The local SP may determine that such an assertion can be used to provide service as the level it would be offered to users anonymously or with unsigned authentication assertions from an IdP.

5 Community Trust Model Examples

In the Community Trust model, an organization (e.g., an industry consortium or a community) sponsors, endorses, or adopts one or more trust establishment services to provide and manage the credentials needed by entities to create and maintain authentication trust among themselves. The service(s) could be operated by the sponsoring organization, or could be provided by an independent service delivery organization. In Community Trust, some level of business trust, although not provided by either direct or brokered business agreements, can be derived from participation in a shared authentication infrastructure. The assumption is that the authentication infrastructure will, in addition to allowing entities to be identified, further identify them as belonging to some community.

Various service options are possible; with PKI technology, e.g., the set of selected services could include one, some, or all of:

- Certification Authorities (CAs)
- Publication repositories for certificates and CRLs, whether generated by sponsored services or obtained from other sources (e.g., from independent CAs maintained by participants rather than a community-level facility)
- On-line facilities for certificate status checking

Different options imply different degrees of organizational involvement and, potentially, of organizational liability. Generally, a broader set of services will incur greater costs than a narrower set, but will also afford more value in terms of enabling trusted connectivity among participant entities and of ensuring consistent assurance across the participant community.

5.1 Community/Direct Model

The simplest cases of direct authentication involve small configurations and manual keying, and a privileged officer responsible for all key management actions. Direct authentication becomes unwieldy as the number of managed entities grows, and consolidated repositories of key material, especially symmetric key material, can create a significant security risk.

Considering an example, a small, multi-site, hub-and-spoke Liberty community agrees to rely on the direct exchange of self-signed certificates to establish communications and authentication trust. Participants accord each other community-level business trust based on their enrollment in this process. The operator of each entity has a software tool that will create PKI key pairs and create self-signed X.509v3 certificates. The IdP operator creates two key pairs, one for SSL/TLS and one for XML-Signature use, and delivers the corresponding certificates to each of the SP operators in a secure manner (e.g., by personal meeting, or by email and subsequent out-of-band verification of the certificate fingerprints). Each SP operator creates one key pair for XML-Signature use, and delivers the corresponding certificate to the IdP operator in a secure manner.

This example uses the technical mechanisms of PKI, in the form of asymmetric key pairs and certificates, without reliance on a Trusted Third Party or Certification Authority. It is therefore an intermediate step, benefiting from ubiquitous technology but not leveraging the advantages of an available TTP service. This approach can be used effectively, but has three major drawbacks:

1. without the stabilizing effect of a TTP and its policies, the necessary discipline and rigor for trusted operation is easily lost (e.g., certificates are exchanged via email but the fingerprint verification may never be done);
2. the trust establishment process is straightforward, but trust disestablishment, when an SP operator goes out of business, for example, requires extreme diligence among participants; and
3. each party assumes full responsibility for identity verification of the other parties.

5.2 Community/Indirect Model

Indirect authentication implies the use of trust infrastructure services outside of the Liberty model. Available trust establishment services can improve the assurance level of Liberty operations, and/or reduce the cost of operations, because they potentially deliver identity verification, credential lifecycle management, and credit checks and other qualification ratings, obtained under well-defined, implemented, and audited policies and procedures. These aspects can be important in the acceptance of corresponding community-level trust relationships for business purposes. Under the assumption that a trust infrastructure service is already available and the participating entities are already enrolled in the infrastructure in other capacities, use of an available trust infrastructure service may also avoid duplication of effort.

5.2.1 Example: PKI Certification Hierarchies

Considering one example, a Liberty community agrees on a list of TTPs offering PKI certificate services. In addition to conventional Certification Authorities (CAs), Bridging Authorities may also be included. In the latter case, each Bridging Authority cross certifies with participating CAs and with other Bridging Authorities. Two types of approaches can be applied (or hybridized) to establish trust among community members:

- Individual entities' trusted CAs establish cross-certification paths to other CAs within the community, and the entities employ their existing trust anchors that reference their trusted CAs
- A list of selected trust roots representing the set of the Community's CAs becomes a Community TAL. This TAL is distributed to all of the entities in the community in a secure manner.

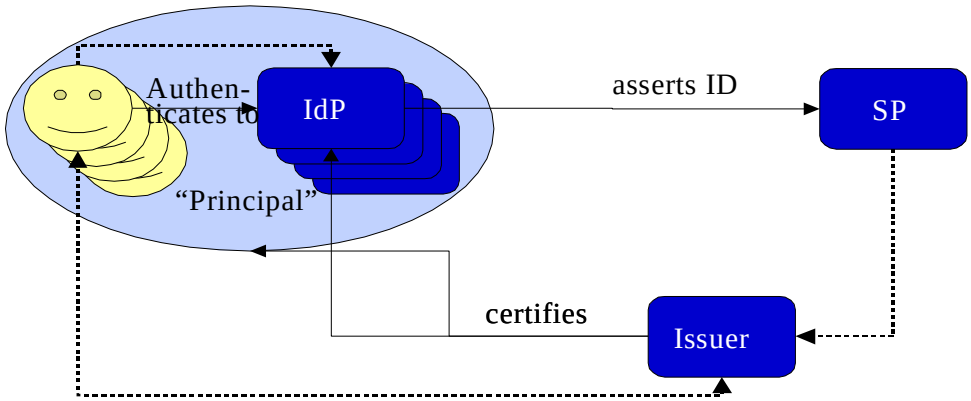
For each trust root there is a certificate verification procedure known to the participating entities. Given any certificate, an entity can apply the certificate verification procedures, and positively determine if the certificate in question was issued in accordance with the policies of one or more of the TTPs trusted by the community. The entity can also determine, according to the policies of the TTPs, if the certificate is still valid (i.e., has not expired, and has not been revoked).

This example represents a "full PKI" case. The selected TTPs may be commercial, government operated, or closed community service providers, and the TAL creates flexibility to adjust the mix over time. As a matter of community policy, the trust anchors could be required to share a single certificate verification procedure, simplifying the implementations of the participating entities; or multiple procedures could be allowed to increase the pre-enrolled population or enable technology migration.

The advantages of the "full PKI" case derive from the long experience with PKI technology, deployment, and services, the substantial number of PKI TTPs and enterprise CAs, and the best practice qualities of PKI for key management in large populations. For these reasons, modern high- and medium-assurance trust management infrastructures tend to be constructed around PKI.

5.2.2 Example: Delegated Trust Scenario

The general Liberty architecture model is that a principal authenticates to an SP via an IdP. This example IdP model describes a case where the IdP function is distributed and colocated with individual principals. For this case, new trust aspects must be taken into account because this model introduces a new element in the trust chain. Indirect trust is applied through certification, to enable individual IdPs to be validated by the entities accepting their assertions.



534

535 **Figure 5: Delegated IdP Model**

536 In this model, the SP does not have a direct agreement with the principal's IdP, but trusts the issuer (acting as a CA) to
537 establish indirect trust. The issuer uses its key to certify the principal's IdP, thereby establishing a chain that can be
538 verified by any entity obtaining the issuer's public key. Typically, certification of principals' IdPs by issuers would
539 take place as part of the registration process between the principal and the infrastructure that the issuer represents. An
540 SP can trust a principal based on the certificate that his/her IdP presents, when the SP has a (direct or indirect) trust
541 relationship with the issuer. Note that several issuers may certify a single principal's IdP.

542 The principal's IdP must store the private key corresponding to its certificate in a secure way, because it is essential to
543 guarantee that no one can masquerade as the principal. In practice, this will require the usage of smart cards or other
544 tamper resistant media to securely support the distributed IdP case.

545 One practical example of this kind of model is a mobile Liberty client, where the IdP provides its certificate to the
546 mobile terminal and the SP trusts the issuer. Based on this trust, the SP can also trust the certificate stored in the
547 mobile client.

548 # 6 Comparison Among Models

549 As the preceding sections demonstrate, a variety of methods can be employed to establish trust among Liberty
550 processing components, achieving different types and levels of assurance. Cryptographic authentication may be based
551 on direct exchange of keys between peers or may be indirect through one or more intermediaries, and may employ a
552 variety of public-key and secret-key technologies. Similarly, the business agreements enabling transactions may be
553 directly exchanged between peers, may be indirect through one or more intermediaries, may be absent or unnecessary
554 for particular transactions, and/or may be derived from enrollment and participation in a shared authentication
555 infrastructure. Authentication trust and business trust may vary independently, thereby supporting a broad range of
556 operational environments.

557 Liberty Phase 1 presumes direct business agreements among the set of entities comprising a circle of trust, employing
558 the Pairwise Trust model. It requires certificate-based authentication of IdPs, and recommends its use for other
559 purposes (authentication of SPs, signing of assertions), but is silent as to whether the trust model applied to verify
560 those certificates is direct or indirect. Pairwise Trust enables strong bonds of mutual trust to be developed, but
561 impedes connectivity beyond small, closed communities. Brokered Trust and Community Trust represent two
562 alternative strategies to enable broader sets of entities to interoperate with one another.

Liberty Phase 2 introduces the prospect that IdPs may operate as intermediaries, introducing SPs with which they share relationships to other IdPs; this comprises the Brokered Trust model. Relative to Pairwise or Community Trust, it adds complexity by interposing active, trusted entities into the protocol transactions performed to accomplish federation. On the positive side, it centralizes the management of interdomain relationships at a relatively small number of entities.

Cryptographic trust establishment infrastructures can be used to enable broader secure interoperability than would be practical if direct authentication trust needed to be established among pairs of participants; this approach exemplifies the Community Trust model. Relative to Brokered Trust, it simplifies federation transactions, at the cost of making larger numbers of entities responsible for assessing and managing cross-domain relationships. Where business requirements permit, use of Community Trust can obviate the need to deploy and invoke the intermediary IdPs that are characteristic of Brokered Trust. If independent organizations interested in facilitating communications among entities (e.g., a community or an industry consortium) were to deploy or sponsor infrastructure facilities, such resources could help to facilitate and encourage the growth of Liberty-based connectivity.

For Liberty to achieve its potential benefits, interoperability beyond the scope of small, closed communities must be possible. Deployers should recognize the prospects of the Brokered and Community Trust models, and should select the choice that best fits their business and operational requirements.

7 Trust Establishment Mechanisms

This chapter introduces an overview and essential characteristics about trust establishment mechanisms applicable to Liberty. PKI and Kerberos can be seen as the primary candidate methods for this purpose.

7.1 Public Key Infrastructure, PKI

PKI-based approaches can provide secure, trusted and efficient key and certificate lifecycle management. This facilitates security services like authentication, data integrity, confidentiality and non-repudiation, which are often seen as essential components and building blocks of modern security. Discussions of PKI and its deployment and usage are available in numerous publications, e.g. [Adam99] [Hous01] [Nash01].

In PKI, a certificate serves to bind a named entity to a public key. The most common and standardized certificate format is ITU-T X.509 (currently version 3) [X.509], discussed in Section 7.1.1. PKI system deployments using standard X.509v3 certificates include the following main components (not all of which are required in all configurations):

- Public-key certificate;
- Certification Authority (CA);
- Registration Authority (RA);
- Certificate Repository;
- End entity (user).

This section introduces these building blocks and section 7.1.2 outlines the various trust models currently in use.

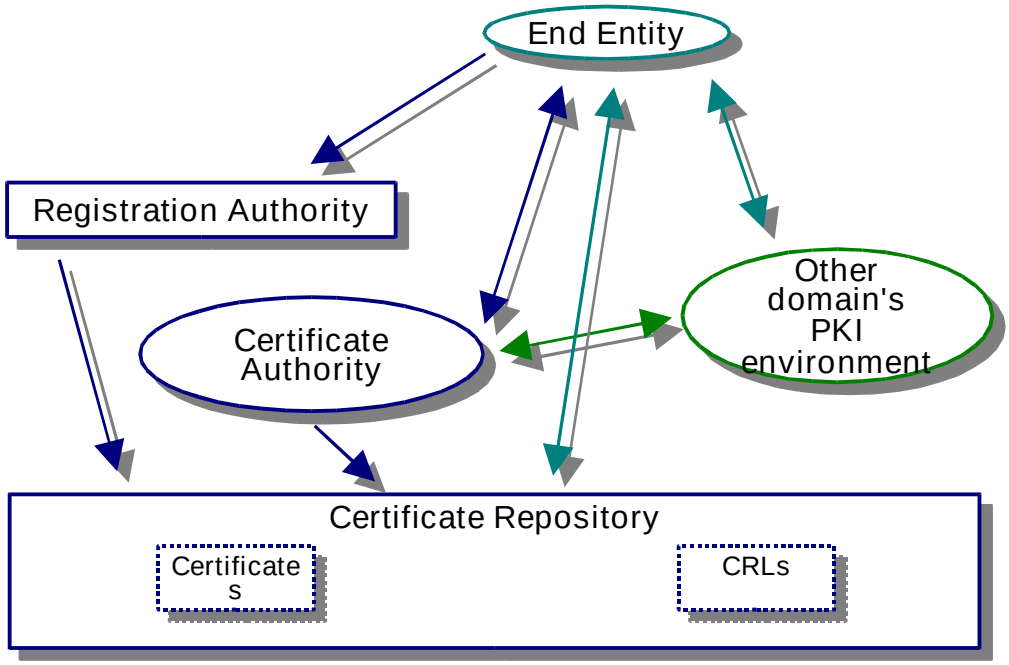


Figure 6: PKI Elements

End-Entity (EE): a user of PKI certificates and/or end-user system that is the subject of a certificate.

Certificate Authority (CA): Acts as the signer of certificates. Primary tasks include the issuance of certificate, renewal of certificate and revocation of certificate.

Certificate Repository (CR): Stores the issued certificates and Certificate Revocation Lists (CRL). Usually provides an interface for users to search directory (such as LDAP interface or HTTP)

Registration Authority (RA): Optional element in PKI system and can be combined with the CA. RA can do some of the CA's management functions and can therefore take some of the load off from CA. RA registers users into the PKI infrastructure. It is particularly useful to separate the RA component when the CA is remote and the RA registers the users in person on behalf of the CA.

7.1.1 X.509

X.509 is the common name by which the International Standard defining the PKI Framework is known. It is also the term that is generally used to identify public-key certificates formatted in accordance with the standard. The X.509 standard has been updated and enhanced several times. Some of the revised editions of the standard enhanced the fundamental structure of a public-key certificate and therefore resulted in a new “version” of public-key certificates. The 1st edition of the X.509 standard was first published in 1988 and the certificates defined in that edition were known as X.509 v1 certificates. The 2nd edition of the X.509 standard was published in 1993. It enhanced the certificate structure, resulting in X.509v2 certificates, by adding two new elements (issuerUniqueID and subjectUniqueID). The 3rd edition of the X.509 standard was published in 1997 and resulted in the definition of the X.509 v3 certificate format. V3 certificates extended the v2 format by adding a general extensions mechanism. As a result of this mechanism, no further certificate versions are anticipated. A number of certificate extensions were defined in the 3rd edition. The 4th edition of X.509 was published in 2000 and although it defined an additional set of certificate extensions, no new certificate format was required. Certificates that include these new extensions are X.509 v3 certificates. The X.509 v3 specification is profiled for Internet usage in IETF RFC-3280 [Hous02].

The main purpose of an X.509 certificate is to establish a link between an identified entity and a public key (and, indirectly, with the corresponding private key held confidentially by the entity). This is accomplished by signing the certificate using the private key of a CA, so that the certificate can subsequently be verified by any entity holding or obtaining the CA’s public key.

The public keys carried in certificates can be used for signature or encryption purposes. When signatures are required, a principal applies a private key and relying parties verify that signature using the public key in the entity's certificate. To perform encryption, the public key in a subject's certificate is used and the subject may decrypt the data using their corresponding private key. Commonly, public-key encryption is used to transfer a symmetric key, which is used in turn for encryption of message data. Typically, users will have two public key pairs (and two corresponding certificates), one for digital signature purposes and a separate set for encryption purposes.

When a certificate is issued, it asserts a binding between a named subject and a public key for a predetermined validity period. When a certificate is used, it is important to determine that its contents remain valid. Two classes of approaches have been specified for this purpose: Certificate Revocation Lists (CRLs, defined within the X.509 specification) and on-line certificate status checking services (e.g., Online Certificate Status Protocol [Myer99] and XML Key Management Specification [Hbak02]). Generally, on-line services can offer more timely detection of revocation events, but require access to trusted and available responders; CRLs are best suited to providing revocation information on a scheduled basis.

Information included in a X.509v3 certificate:

1. Public key of certificate owner;
2. Issuer's (CA) individual name;
3. Validity time of certificate;
4. Subject, name of the certificate owner;
5. Digital signature of the issuer;
6. Extensions.

7.1.2 Trust establishment in PKI system

PKI enables a variety of different trust models. The selection of a trust model for a certain environment depends on several different factors and the requirements for one environment can vary greatly from those in another. Trust models for Liberty were introduced in previous chapters of this document.

The three primary trust models used in PKI are hierarchical, distributed and bridge. *Hierarchical trust* is a common PKI trust model. In this model the trust is established as a tree structure from top to bottom. At the top of the whole trust model is the root CA that has sub-CAs, with sub-CAs providing CA services to their end entities. In the hierarchical trust model, there is a single trust anchor, the public key of the root CA, that is used by all relying parties within the hierarchy.

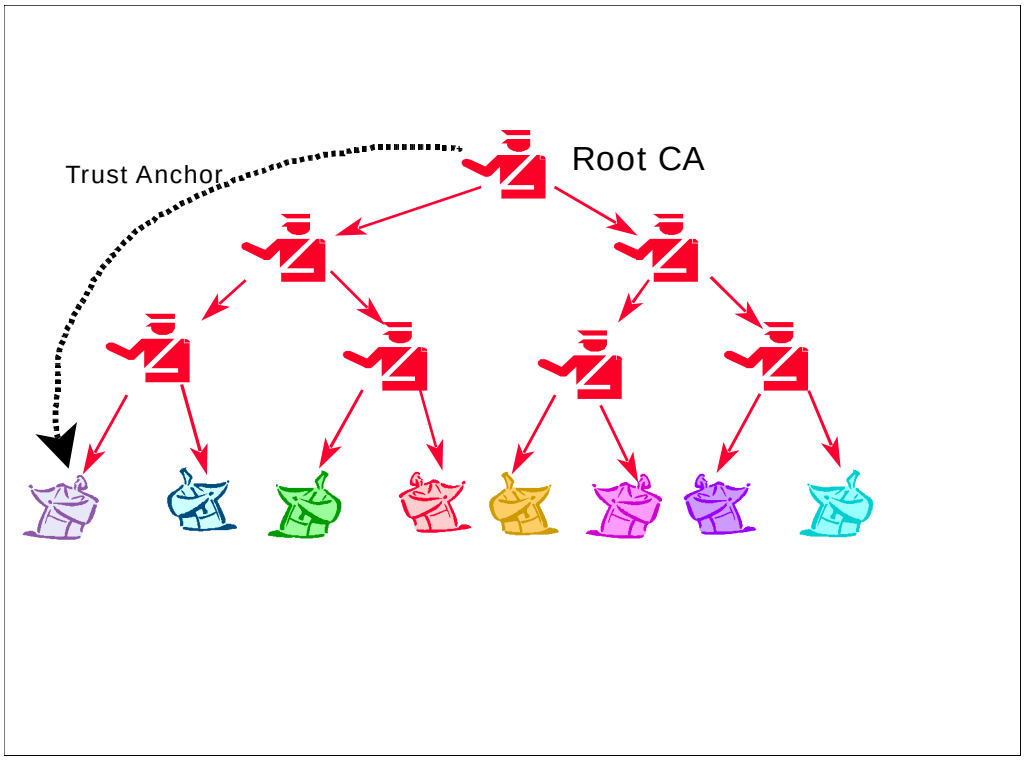


Figure 7: Hierarchical PKI Model

This kind of trust model makes possible to delegate trust and CA operations to sub authorities. When the trust chain is built in this model, it is done by backtracking. The path must be built from the end entity up to the root CA. Once the path is built, however, processing of the certificates must be done in order from the trust anchor down to the end-entity certificate.

The hierarchical trust model is best suited to environments where there is a natural root identified for the business environment and there is a fully established development process for the architecture in place.

The distributed trust model is one where no single CA roots all trust. Rather, typically the key used as a trust anchor for a given user is the public key of the CA that issues certificates to that user. In this model, there is a distributed network of trust anchors. One advantage of this model is that there is no single point of failure as there is with the single trust anchor in the hierarchical model. Also, in this model the CAs are able to act fairly autonomously without being bound by policy delegated from a root CA.

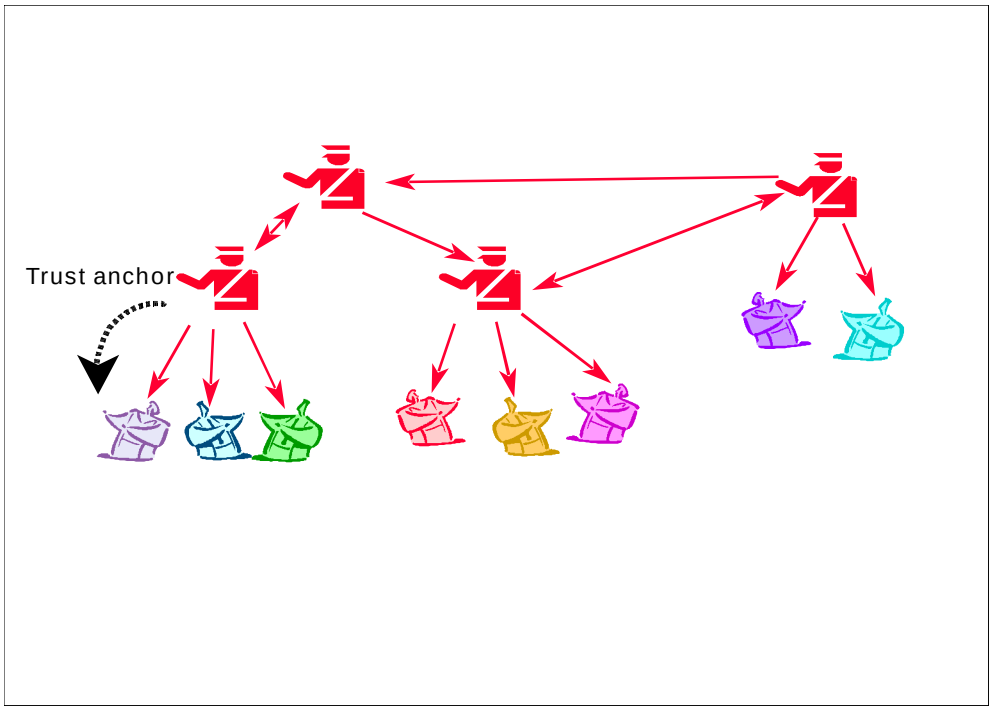


Figure 8: Distributed PKI Model

In the distributed trust model, certification paths can be built in either direction, or a combination of both, however, processing of the certificates must always be done from the local trust anchor to the end-entity certificate. The distributed trust model is best suited to business-to-business environments where there are a relatively small number of CAs that need to be inter-connected.

The Bridge trust model is similar to the distributed model in that there is no single Root CA and no single trust anchor common to all users. In the bridge model there is a single CA that acts purely as a facilitator to interconnect other CAs. A bridge CA typically does not issue certificates to any end entities, but is used, as a hub, to interconnect the spokes which can be individual CAs, PKIs that use the hierarchical trust model, and PKIs that use the distributed trust model. The primary benefit provided by a bridge CA is that each spoke need only maintain a single cross-certification with the bridge CA and they are automatically able to build certification paths across all spokes in the model.

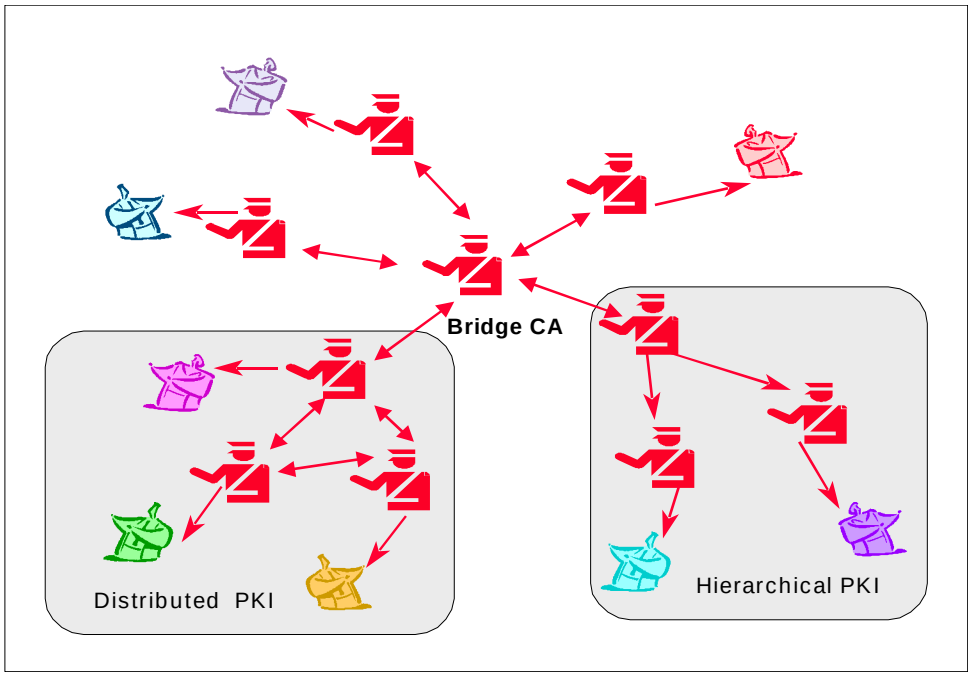


Figure 9: Bridge PKI Model

In the bridge model, certification paths can be built in a combination of directions. If the path includes certificates in a hierarchical PKI, those portions of the path would be built from the end-entity to the root of that hierarchy. Other portions of the path can be built in either direction. Processing of the certificates in the path, as with the other trust models, must always be done from the trust anchor to the end-entity certificate. The bridge trust model is best suited to environments where a large mesh of cross-certificates would otherwise be needed to establish the required trusted environment, such as the U.S. Federal Government and its agencies. The bridge CA can also provide a single point of interconnection for all its spokes to external PKIs.

In all models described above, trust between CAs is established by using *cross-certification*. In the hierarchical model, cross-certification is used to delegate responsibility to subordinate CAs. It is also used for connecting the hierarchical PKI to other certification domains. In the distributed model cross-certification is used to connect the CAs within a domain and similarly to connect the spoke CAs with the hub, the bridge CA. Cross-certification can be seen as representing a peer-to-peer contract between two CAs.

In cross-certification trust establishment, CAs create trust to each other so that CA A's entities are trusted by CA B's entities and issue cross-certificates to represent these trust relationships. Cross certificates can include extensions that impose constraints on the set of certificates in the remote domain that are acceptable to be trusted by relying parties in the local domain. Depending on applicable policies, cross certificates may be issued by root CAs or by sub-CAs within their hierarchies.

Additional discussion on PKI trust models can be found, e.g., in [Elle01], [Linn00], and [Perl99].

7.1.3 Conclusions

Public key cryptography enables strong methods for entity authentication and PKI provides many methods to establish trust relations between different entities. The appropriate architecture for each situation can be determined based on numbers of entities, numbers of CAs, and their organizational relationships and associated policies.

7.2 Kerberos

Kerberos [Kohl93] is the most common method to provide strong authentication between users and servers by using secret key cryptography, based on a protocol developed by MIT. After the identity is proved both entities can communicate using encryption and integrity protection.

Kerberos provides key freshness, i.e., a new session key is created whenever two entities want to communicate with each other. Since new keys are generated for each session, an attacker that determines the key used for one session cannot use it to decrypt subsequent traffic.

In Kerberos, each participating user and server shares a distinct long-term secret key with a trusted authority, the Key Distribution Center (KDC). For the user case, the shared secret is derived from a password. These secrets are used for processing at their respective entities, but are not transmitted over the network. Session keys are generated and delivered by the KDC within protocol elements called tickets, when communication between two entities is starting. In most current Kerberos deployments, the key shared between entity and KDC is a (56-bit) DES key, with DES CBC mode used for encryption. Specification activities incorporating triple-DES (112-bit key) and AES (128-bit and longer keys) are currently in progress, and use of these newer algorithms appears prudent from a cryptographic perspective once corresponding implementations are available.

Once registered with a KDC, a user's Kerberos interactions proceed as follows. The user's client requests a special type of ticket (the Ticket Granting Ticket, or TGT) from the KDC, receives the TGT and an encrypted representation of the corresponding TGT session key, and applies the user's password to decrypt the TGT session key. Once this step is complete, the user's password can be deleted from memory, as it is not required for subsequent use of the TGT. When the user wishes to communicate with a particular server, it sends the KDC a message with its TGT, an authenticator based on knowledge of the TGT session key, and an indication of the server with which the user wishes to communicate. If the KDC successfully validates the authenticator, it generates a service ticket for the user to use in communication with the requested server and returns it to the user's client along with a representation of the service ticket's session key, encrypted using the TGT session key. Based on this data, an authentic client can now generate an authenticator with the service ticket session key and can send it to the server along with the service ticket, thereby authenticating its user to the server.

7.2.1 Kerberos processing

This section describes the basic Kerberos cryptographic protocol based on Kerberos version 5. As preconditions, both the user and server have keys that are registered with the KDC. The user's key is generated from the password he/she has chosen, and the server's key is randomly selected and stored at the server.

Processing in Kerberos:

- User A sends a message to KDC and tells the KDC that it wants to communicate with server B
- KDC creates a random session key, K and makes two copies of it. KDC creates two encrypted messages, where message 1 (m_1) is encrypted with user's key and message 2 ("ticket") with server's key. Both messages are sent to user.

$$m_1 = e_{KA}(K, ID(B))$$
$$m_2 = e_{KB}(K, ID(A))$$

- User decrypts the message 1 with his/her own key and gets the session key.
- User creates new message ("authenticator"), m_3, and encrypts it with new session key. This new message includes the timestamp T. Timestamp is included to prevent the sending the message 2 again later by attacker who impersonates the user.

$$m_3 = e_K(ID(A), T)$$

- User sends messages 2 and 3 to the server

- Server decrypts message 2 with the key it shares with KDC and gets the session key. Then it decrypts the message 3 with new session key.
- If the user wants the server to be authenticated as well, an additional message is needed. In this case the server takes the timestamp, T, from the message 3 and creates new message, m4, which is encrypted with session key.

$$m_4 = \left(ID\left(B\right), T \right)$$

7.2.2 Conclusions

The basic Kerberos protocol is vulnerable to password guessing attacks against TGTs, as a TGT can be requested and obtained without first demonstrating possession of the password; the optional preauthentication facility provides a countermeasure against this attack. Interoperability between different realms can be accomplished using inter-realm protocol facilities and shared inter-KDC keys, but trust models for inter-realm Kerberos operation have received less evaluation and standardization than corresponding models for PKI environments.

8 Integrating Trust Establishment Infrastructures with Liberty

In practice, trust establishment technologies would be applied in a layered fashion to support Liberty requirements. At the lowest level, a bootstrapping process would be used to create and maintain authentication trust among the participating entities: an entity would initially be enrolled in a trusted relationship with a trust establishment service. The trust establishment service would then facilitate introductions between this and other enrolled entities.

The nature of enrollment with the trust establishment service and the mechanisms for authentication trust between a Liberty entity and the trust establishment service are unspecified by Liberty. Authentication trust could be achieved by any technical mechanism that provides message authenticity, integrity, and confidentiality, e.g., physically secure channels, PKI, manual SKI, or Kerberos for authentication, together with SSL/TLS, IPSEC, S/MIME, or SSH for integrity and confidentiality.

The nature of authentication trust between Liberty entities, as delivered by the trust establishment service, is partially defined in the Liberty specifications. Some entities are required to accept SSL/TLS sessions, and all are required to verify XML-Signatures [East02] on messages if present. Although the Phase 1 Liberty specifications do not require all XML messages to be signed, it is best practice for senders to sign all messages, and the Phase 1 Liberty specifications note that vulnerabilities may be introduced if messages are not signed. Some entities may initiate SSL/TLS sessions with certificate-based authentication. Liberty entities may use additional mechanisms that are permitted, but not required, in the Liberty specs, for example, IPSEC security associations. All of these security mechanisms require the distribution of cryptographic keys (public/private key material and/or symmetric key material). The primary function of the trust establishment service is the distribution and management of this key material. Once private or symmetric keys are distributed, secure processing depends on protection of the stored keys against compromise; while such protection mechanisms are implementation-specific and are not defined by Liberty specifications, they are important aspects of secure processing components.

The Liberty Phase 1 specifications do not mandate XML-Signatures on all messages, nor do they constrain the technical options present in XML-Signature when it is used. These options include, for example, signature systems based on both PKI (using asymmetric key pairs) and HMAC algorithms (using symmetric keys). Implementation requirements may favor one or another approach, however, because of the advantages of PKI for key distribution and non-repudiation, best practice for large-scale deployments will generally use PKI mechanisms. PKI may imply, however, some additional system complexity and costs. Small-scale systems, or systems that create no questions of legal liability (e.g., a Liberty deployment entirely within a single company), might rely on secured channels between Liberty elements, or manual, symmetric keying for signatures.

Since X.509v3 certificates can be used to implement authentication trust in the SSL/TLS and XML-Signature protocols named in the Liberty Phase 1 specifications, the trust establishment service may, in fact, be an X.509v3 Certification Authority, providing usual and customary CA services. Advantages of this approach are the significant number of commercial Trusted Third Parties (TTPs) already providing these services, the large number of compatible

software implementations available, and the broad dissemination of technical knowledge concerning PKI. TTP services include the ability to certify participating Liberty providers, distribute issued certificates, and update and distribute Certificate Revocation Lists. Additionally, on-line validation services (e.g., through the OCSP or XKMS protocols) could be provided for the certificates. This model offers scalable trust at a strong level (though somewhat less than that of the Circle of Trust Model), but requires organizational involvement to establish and manage infrastructure.

The term “trust establishment service” is used in a general sense, because although the service could, in fact, be a Certification Authority, the service need not operate as a conventional CA. Instead, it could be a broker for several CAs (e.g., a PKI bridge). It could deliver private keys and certificates through protocols not conventionally associated with CAs (e.g., in files through a shared file system). It could construct certificates in unconventional ways (e.g., all participating entities use the same private key and the same short-lived certificate, replaced daily). Researchers, companies, and governments continually seek improvements to the technology of trust management, and many new alternatives will appear and be tested by the marketplace.

9 Metadata and Trust Discovery

If two entities attempt to communicate without previous awareness of membership in a common trust infrastructure, the following outcomes are possible:

1. the entities communicate insecurely without authentication
2. the entities transfer data enabling them to perform authentication
3. the entities do not interoperate

In the second scenario, an entity wishes to communicate with another entity in order to perform some transaction but has no pre-existing basis for the required technical trust. Nevertheless, the entities may be able to establish trust between themselves through exchange of trust metadata.

One such mechanism would be for the involved entities to publish their public keys along with their approved usages, the commitments the key owner makes with respect to that key, and the obligations that a relying party must accept (either implicitly or explicitly) if were to use that key. The key owner would publish this statement to potential relying parties; an XML Signature calculated over it would both ensure its integrity and bind the associated private key to those statements. A relying-party, once it discovered this signed statement, would be able to examine the approved applications, commitments and obligations associated with that key and determine whether or not the key was appropriate to an intended application. If the result of this analysis were positive, the relying party would install the public key into some trusted store – the stored key indexed by the application usages for which it was appropriate. This process is shown in the following diagram.

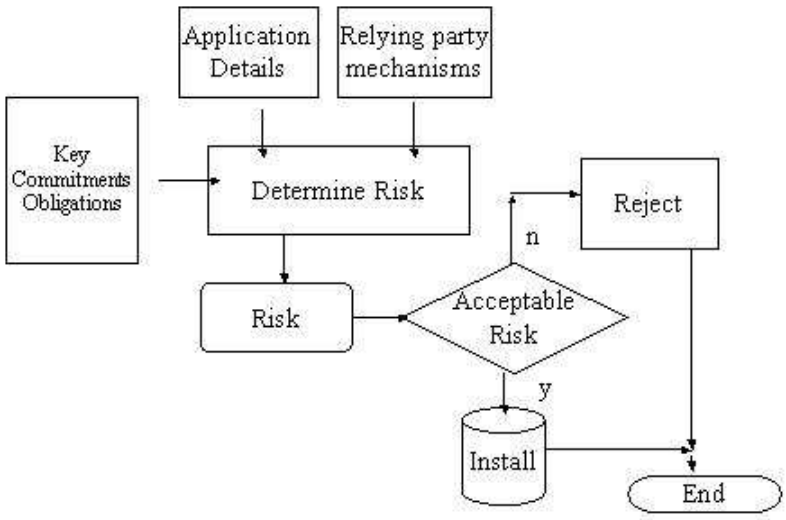


Figure 10: Validation of Key from Metadata

As the public key is distributed along with the associated business commitments and obligations, exchange of Trust Metadata in this scenario can be thought of enabling both business and authentication trust (e.g. a decision to install a key will result in the addition of the key-owner to both of the relying-party's BAL and TAL).

10 References

[Adam99] C. Adams, S. Lloyd, "Understanding the Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations", New Riders Publishing, 1999.

[East02] D. Eastlake, J. Reagle, D. Solo, "XML-Signature Syntax and Processing", Internet RFC-3275, March 2002. http://www.ietf.org/rfc/rfc3275.txt.

[Elle01] Y. Elley, A. Anderson, S. Hanna, S. Mullan, R. Perlman, S. Proctor, "Building Certification Paths: Forward vs. Reverse", ISOC NDSS, 2001.

[HBak02] P. Hallam-Baker, ed., "XML Key Management Specification (XKMS 2.0)", World-Wide Web Consortium (W3C) Working Draft, 31 January 2002. Currently available at http://www.w3.org/2001/XKMS/.

[Hous01] R. Housley, T. Polk, "Planning for PKI: Best Practices Guide for Deploying Public Key Infrastructure", John Wiley & Sons, New York, 2001.

[Hous02] R. Housley, W. Ford, W. Polk, and D. Solo, "Internet X.509 Public Key Infrastructure: Certificate and CRL Profile", Internet RFC-3280, April 2002. http://www.ietf.org/rfc/rfc3280.txt.

[Kohl93] J. Kohl, C. Neuman, "The Kerberos Network Authentication Service (V5)", Internet RFC-1510, September 1993. http://www.ietf.org/rfc/rfc1510.txt.

[Linn00] J. Linn, "Trust Models and Management in Public-Key Infrastructures", RSA Laboratories Technical Note, 6 November 2000. Available via http://www.rsasecurity.com/rsalabs/technotes/index.html.

[Myer99] M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, "X.509 Public Key Infrastructure: Online Certificate Status Protocol – OCSP", Internet RFC-2560, June 1999, http://www.ietf.org/rfc/rfc2560.txt.

[Nash01] A. Nash, W. Duane, C. Joseph, D. Brink, "PKI: Implementing and Managing E-Security", Osborne/McGraw-Hill, New York, 2001.

[Perl99] R. Perlman, "Overview of PKI Trust Models", IEEE Network, November/December 1999.

[X.509] ITU-T Recommendation X.509 (2000) | ISO/IEC 9594-8:2000, Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks.