

X.509 PROFILES FOR VARIOUS CA SCENARIOS

Version 3.0

Author: Sharon Boeyen
Date: June 2004

Entrust is a registered trademark of Entrust, Inc. in the United States and certain other countries. In Canada, Entrust is a registered trademark of Entrust Limited. All Entrust product names are trademarks of Entrust, Inc. or Entrust Limited. All other company and product names are trademarks or registered trademarks of their respective owners. The material provided in this document is for information purposes only. It is not intended to be advice. ENTRUST DOES NOT WARRANT THE QUALITY, ACCURACY OR COMPLETENESS OF THE INFORMATION CONTAINED IN THIS ARTICLE. SUCH INFORMATION IS PROVIDED "AS IS" WITHOUT ANY REPRESENTATIONS, WARRANTIES AND/OR CONDITIONS OF ANY KIND, WHETHER EXPRESS, IMPLIED, STATUTORY, BY USAGE OF TRADE, OR OTHERWISE, AND ENTRUST SPECIFICALLY DISCLAIMS ANY AND ALL REPRESENTATIONS, WARRANTIES AND/OR CONDITIONS OF MERCHANTABILITY, SATISFACTORY QUALITY, TITLE, NON-INFRINGEMENT, OR FITNESS FOR A SPECIFIC PURPOSE.

TABLE OF CONTENTS

1. ABSTRACT	3
2. INTRODUCTION	4
2.1 SCOPE	4
2.2 TERMINOLOGY	5
3. COMMON RECOMMENDATIONS	6
3.1 CERTIFICATE CONTENT.....	6
3.1.1 Base Certificate Fields	6
3.1.2 Certificate Extensions	8
3.1.3 Extension Criticality.....	9
3.2 CRL CONTENT.....	10
3.2.1 Base CRL Fields	10
3.2.2 CRL Extensions	11
4. COMMON PROFILES	13
4.1 USER CERTIFICATES	13
4.2 CERTIFICATE REVOCATION LISTS	14
5. DISTRIBUTED TRUST MODEL	17
5.1 USER CERTIFICATES	17
5.2 CERTIFICATE REVOCATION LISTS	17
5.3 CROSS CERTIFICATES.....	17
5.4 PROCESS CONSTRAINTS	19
6. HIERARCHICAL TRUST MODEL	21
6.1 USER CERTIFICATES	21
6.2 CERTIFICATE REVOCATION LISTS	21
6.3 SUBORDINATE CERTIFICATES.....	21
6.4 ROOT CA – CROSS CERTIFICATES.....	23
6.5 PROCESS CONSTRAINTS	23
7. BRIDGE TRUST MODEL	25
7.1 USER CERTIFICATES	25
7.2 CERTIFICATE REVOCATION LISTS.....	25
7.3 SPOKE CA: CERTIFICATES ISSUED TO BRIDGE CA	25
7.4 BRIDGE CA: CERTIFICATES ISSUED TO SPOKE CA	27
7.5 BRIDGE CA: CROSS CERTIFICATES ISSUED TO EXTERNAL CA.....	29
7.6 PROCESS CONSTRAINTS	29
8. SUMMARY	30
9. ABOUT ENTRUST	31
10. REFERENCES	32
11. ACRONYMS	33
ANNEX A: KEY LENGTHS AND HASH FUNCTIONS	34

1. ABSTRACT

There are a number of options with respect to the fields and extensions that a CA includes in certificates it issues. Numerous profiles have emerged that specify the content to include for particular applications or operating environments. This paper provides recommendations for certificate and CRL content, as well as for constraints on relying party path validation processes, for a number of common scenarios. The scenarios are based on the type of CA issuing the certificates and CRLs. These profiles are intended to assist deployment of PKI solutions where there are no specific profiles that need to be adhered to and are expected to be applicable in many different operating environments.

2. INTRODUCTION

[PKIX] and [X.509] specify the standard elements that comprise public-key certificates and CRLs as well as a set of initial conditions for path validation that can be used to constrain the set of certificates trusted by a given certificate user/relying party (RP). Certificate elements include a set of base certificate fields as well as a set of extensions that can be included. Depending on the environment in which a CA is operating, and on the type of entity a certificate is issued to, the content of certificates differs. Also, depending on the environment in which a RP is operating, the conditions that may be imposed to constrain trust to a subset of the potential candidate certificates may also differ.

2.1 SCOPE

This paper provides a set of recommendations for certificate and CRL content profiles and initial conditions for path validation for a number of scenarios. The advice in this paper is intended for entities deploying PKI in a general environment that is not specifically covered by its own set of profiles. There are no specific certificate policies to which these profiles align, however they are expected to be generally applicable in many environments. Even where there are existing policies and profiles, this paper may provide additional advice in narrowing the choices among options in those specifications. These profiles are based on the following assumptions about the operating environment of a CA:

- An LDAP directory is being used as the PKI repository;
- Certificates are being issued to end-users as well as to other CAs;
- CRLs are used as the revocation scheme;
- CRLs may be partitioned by certificate type (user/CA) and by number of certificates, but not by reason code;
- Indirect CRLs and Delta CRLs are not used;
- The deployment is not covered by any existing profile;
- Certificates are intended to be applicable to multiple applications;
- Users have 2 key pairs (one for encryption and the other for digital signatures); and
- Use of the wildcard “anyPolicy” policy identifier is to be inhibited

CAs typically participate in one of three general trust model environments; mesh/distributed, hierarchical or bridge. This paper proposes profile recommendations for CAs that operate in the following scenarios:

- Mesh/Distributed Trust Model
 - Standalone CA
- Hierarchical Trust Model
 - Root CA
 - Subordinate CA
- Bridge Trust Model
 - Bridge CA
 - Spoke CA

For these scenarios, the following set of profiles is defined:

- User certificate content
- CA cross/subordinate certificate content
- CRL content
- Relying party initialization of path validation variables

Section 3 introduces the elements (fields and extensions) that comprise certificates and CRLs and provides recommendations that are common to all profiles in this paper.

Section 4 specifies profiles for user certificates and for CRLs. These two profiles apply to these entities in all trust model environments.

Sections 5-7 specify profiles for CA certificate types and for process constraints for RP path validation. Each of these profiles is specific to the trust model for which it is provided.

2.2 TERMINOLOGY

The profiles contained in sections 4-7 use the following terminology to indicate the presence/absence of a certificate/CRL field or extension:

P (present) Indicates that the field/extension must be present in all certificates/CRLs issued in accordance with the profile

A (absent) Indicates that the field/extension must be absent from all certificates/CRLs issued in accordance with the profile; and

O (optional) Indicates that the field/extension may be present in some, but not necessarily all, certificates/CRLs issued in accordance with the profile.

The profiles contained in sections 4-7 use the following terminology to indicate the value of the criticality component of a certificate/CRL extension:

c Indicates that the extension is set to critical;

nc Indicates that the extension is set to non-critical.

3. COMMON RECOMMENDATIONS

This section provides an introduction to the specific elements of certificate/CRL content and relying party initialization of path validation variables that are common to each of the profiles. In many cases the recommendations for a particular element are identical across all profiles. These common recommendations are also included in this section.

3.1 CERTIFICATE CONTENT

Public-key certificates consist of a set of base fields plus a set of extensions.

3.1.1 Base Certificate Fields

[PKIX] and [X.509] mandate inclusion of most base certificate fields in all certificates of all types. This section makes recommendations on the population of those base fields across certificate types. These recommendations apply to all specific certificate profiles included in this paper.

3.1.1.1 Version

To date, there have been three versions of the syntax for public-key certificates defined in [X.509]. The only version that allows extensions to be included in certificates is version 3. Because of the key role extensions play in determining the trustworthiness of a certificate and the appropriateness of a given certificate to a particular transaction, all certificates should now be issued as version 3 certificates.

3.1.1.2 Serial Number

The only requirement placed on serial numbers by [X.509] is that the value be a unique integer for each certificate issued by a given CA. The combination of serial number plus certificate issuer name provides a unique identifier for each certificate, provided issuer names are unique. In order to avoid any interoperability issues with systems that may not support negative integers in this field, it is recommended that the values of the serial number be assigned starting with the integer "1" and increasing by the value of "1" for each subsequently issued certificate.

3.1.1.3 Signature

This field contains the identifier for the algorithm used by the CA to sign the certificate. It must contain the same algorithm identifier as the **signatureAlgorithm** field in the **Certificate** ASN.1 sequence. For interoperability reasons, it is recommended that these fields be populated with the algorithm identifier for the RSA PKCS #1 Version 1.5 signature algorithm along with the appropriate SHA hash function for the desired security level. See Annex A for information on security levels and which hash function to use. Based on Annex A and the validity periods contained in these profiles, all certificates should be signed by CA public keys that are 2048 bits long and the signatures should be created using the SHA-256 hash function. The Object Identifier (OID) for **sha256WithRSAEncryption** for these signatures is { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 11 }.

While there are no known attacks on the RSA PKCS #1 Version 1.5 algorithm, some high security applications may wish to use the RSA-PSS algorithm which has more attractive security characteristics. Similarly, some applications may want to use the ECDSA signature algorithm, which uses elliptic curve cryptography, because of the performance advantages that it provides at high security levels. See [ECC] for a discussion of when it is appropriate to use elliptic curve based techniques.

These recommendations apply at the time of writing (2004) and will require updating in accordance with updates to the advice in Annex A.

3.1.1.4 Issuer/Subject

These fields contain the Distinguished Name (DN) of the certificate issuer and the certificate subject respectively. There are two schemes for formulating DNs that are generally used.

The geopolitical name scheme formulates DNs based on the geographic location of the entity combined with organizational and/or personal attributes. For example, a U.S. based organization might choose to name its CA "ou=ABC CA; o=ABC; c=us". In some countries there is a national registration authorities set up to register organization names. If the geopolitical name scheme is used, it is recommended that the organization name be registered with such an authority to ensure uniqueness. If a name dispute does arise at some point in the future, this registration provides an additional safeguard to defend the right to use that name. This name scheme is most applicable to environments where there is an existing LDAP or X.500 directory infrastructure in place that makes use of geopolitical names, or where a national identity is important. If geopolitical names are to be used, it is recommended that, as a minimum, the DN of a CA include the **countryName** attribute, the **organizationName** attribute and the **organizationalUnitName** attribute. For end-users it is recommended that the DNs include the same attribute types plus at least the **commonName** attribute. If this is insufficient to unambiguously identify an end-user the **serialNumber** attribute can also be used. While it is true that technically an organization may elect to use the **organizationName** attribute without a **countryName** attribute, this is not recommended as name clashes would likely occur more frequently.

The domain-component name scheme formulates DNs based, at least in part, on [RFC 2247]. That RFC provides a mapping of domain names to DNs by using the **domainComponent** (dc) attribute type to represent each component of a domain name. This naming scheme is most applicable in environments where the domain name is expected to remain unchanged, the domain name is sufficient to identify the organization, and there is no pre-existing LDAP/X.500 directory in place. For such environments, basing at least the higher levels of the naming infrastructure on this scheme is recommended. For example a CA name might be "cn=ABC CA; dc=ABC; dc=com".

Setting up a dc-based name scheme can have the following advantages:

- No additional name registration required;
- Names are automatically globally unambiguous – no name clashes; and
- Users can use the domain name directly to review a web site and learn more about the entity.

While it is recognized that geopolitical names will be more appropriate in certain environments, the profiles in this paper recommend the dc-based scheme for the reasons given above.

In terms of encodings, at the present time it is recommended that PrintableString encoding be used for all names that can be encoded using it. UTF8 is recommended only for names that cannot be encoded using PrintableString. These recommendations are to facilitate interoperability with the broadest installed base. It is possible that in the future, as UTF8String becomes more widely implemented, this recommendation would change.

3.1.1.5 Validity

Both the **notBefore** and **notAfter** elements of validity need to be present in all certificates. [X.509] allows these times to be represented as either **utcTime** or **generalizedTime**, however [PKIX] requires that dates through the year 2049 be encoded as **utcTime** and dates in 2050 or later be encoded as **generalizedTime**. Therefore, given that certificates are normally issued with a much smaller validity period than 45 years, all certificates should currently be issued with their validity periods in **utcTime** format. These times must be expressed as Greenwich Mean Time (Zulu) and be of the format (YYMMDDHHMMSSZ). If the value of "YY" is greater than or equal to "50", this represents the year "19YY". If the value of "YY" is less than "50", this represents the year "20YY".

For most applications user certificate lifetimes will tend to be 1 or 2 years, and certainly less than 5 years. CA certificates will tend to be in the 5 to 10 year timeframe and root certificates will tend to be in the 10 to 20 year timeframe. If any certificate content needs to be modified during the validity period of a certificate, that certificate needs to be revoked and replaced with a new one. To balance the amount of time such

certificates need to remain on CRLs, all profiles in this paper recommend a validity period of 2 years for user certificates, 5 years for subordinate and cross-certificates and 10 years for hierarchical root certificates.

3.1.1.6 Subject Public Key Info

This field is used to carry the public key and identify the algorithm with which the key is used. For interoperability, this field should contain the algorithm identifier for RSA public keys, **rsaEncryption**. The parameters field has value NULL for this algorithm. This algorithm identifier is used in public key certificates for both RSA signature keys and RSA encryption keys. The intended application for the key will be indicated in the key usage field. See the Annex for information on appropriate key sizes. Given the validity periods in these profiles, user keys should be 1024 bits long and CA keys should be 2048 bits long. The OID for **rsaEncryption** for these keys is { iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 1}. This recommendation is common to all profiles in this paper.

Certain high security applications may wish to restrict the use of the RSA public keys to the RSA-PSS algorithm (for signature) or RSA-OAEP (for encryption). These algorithms are less widely deployed than the RSA PKCS #1 Version 1.5 algorithm, but have better theoretical security characteristics. Similarly, some applications may wish to indicate an elliptic curve public key in this field because of the performance advantages that they provide at high security levels. See [ECC] for a discussion of when it is appropriate to use elliptic curve based techniques.

These recommendations apply at the time of writing (2004) and will require updating in accordance with updates to the advice in Annex A.

3.1.1.7 Issuer / Subject Unique Identifiers

These are the only two base fields that are optional in [X.509]. They were added in version 2 certificate syntax, prior to the extension capability. However, these are not widely supported and seldom used, if ever. These two fields should be excluded from all certificates.

3.1.2 Certificate Extensions

There are a number of standard extensions defined in [X.509] and a few additional ones in [PKIX]. Specific recommendations for each extension are addressed in the individual profiles later in this paper. In terms of general requirements, there are a number of certificate extensions that should be included in all certificates. Common requirements for these extensions are outlined below. In addition to these common extensions, there are extensions that are recommended to be included in some certificate types but not others.

3.1.2.1 Key Identifiers

The **authorityKeyIdentifier** extension identifies the public-key to be used in verifying the signature on the certificate. This extension should be included in all certificates. The **keyIdentifier** component of the extension should always be present and include a 160 bit SHA-1 hash of just the public key. The **authorityCertIssuer** and **authorityCertSerialNumber** components can be excluded.

The **subjectKeyIdentifier** extension identifies the public-key being certified in the certificate. This extension should be included in all certificates. The extension should include a 160 bit SHA-1 hash of the **subjectPublicKeyInfo** field of the certificate, or whatever the requester asks for.

These recommendations apply at the time of writing (2004) and will require updating in accordance with updates to the advice in Annex A.

3.1.2.2 Basic Constraints

The **basicConstraints** extension identifies the certificate subject as either a CA or a user. This is an important property of each certificate when building certification paths. As only CAs are permitted to issue

certificates, all certificates, except the final certificate in a path, must have this extension present and indicate that the subject is a CA. Because version 1 and version 2 certificates had no way to differentiate between CA and user subjects, it is also recommended to include this extension in user certificates. This extension also includes the ability to constrain the length of a certification path. It may be useful in some environments to include such a constraint (e.g. to ensure the path development process doesn't run for an unnecessarily long period of time). However, effective use of this constraint would require knowledge of the specific CA network architecture in order to determine the right value to include in the constraint. Also, unknown changes in remote CA network architectures (e.g. the addition or deletion of a CA) could cause unintended impacts on the set of certificates that will/won't be trusted under this constraint. Therefore this constraint is not recommended in the profiles in this paper.

3.1.2.3 Key Usage

The **keyUsage** extension identifies the purposes for which the certified key is to be used. In all certificates issued to CAs, this extension should be present with the **keyCertSign** bit set. In most cases CAs also issue CRLs. Therefore the **cRLSign** bit should also be set. This extension should also be present in all certificates issued to users. User encryption public key certificates should have the **keyEncipherment** bit set and user signature verification certificates should have the **digitalSignature** bit set. Note that these specific settings for the user certificates are based on the recommended algorithm **rsaEncryption** for all profiles in this paper. If other algorithms are used, e.g. ECDSA, the **keyUsage** extension values would also change.

3.1.2.4 Certificate Policies

The **certificatePolicies** extension indicates the policies under which the certificate is to be used. Typically policies indicate either a set of applications or an assurance level associated with the certificate. Because policy may be required in the path validation process it should be included in all certificates. Use of the special value "**anyPolicy**" in this extension should be avoided as it may invalidate the certificate, based on the conditions under which the validation process is being conducted. Specific recommendations regarding population of this extension are included in each of the profiles in this paper.

3.1.2.5 Authority Info Access

The **authorityInfoAccess** extension has two primary uses. It is used to point to the single repository location where certificates issued to the issuing CA can be located. To aid in path development it is recommended that this extension be included in all certificates and that an LDAP URI be used for this purpose. The other purpose of the extension is to provide pointers to OCSP servers. Because profiles in this paper assume CRLs are used instead of OCSP, this second pointer is excluded from these profiles.

3.1.2.6 CRL Distribution Point

The **cRLDistributionPoints** extension provides a mechanism to point to the location of the CRL(s) that would list the certificate in the event that it were to become revoked. All profiles in this paper recommend inclusion of this extension with a **directoryName** name component and an LDAP URI pointing to the CRL. The reasons and **cRLIssuer** components of this extension are excluded in all profiles.

3.1.3 Extension Criticality

Extensions include a component that indicates the criticality of that extension to the use of the certificate.

If this component is set to critical, it is mandatory that the RP understand and process that extension as part of the validation to determine the trustworthiness of the certificate.

If the component is set to non-critical, an RP that does not understand and cannot process that extension may ignore the extension and proceed with the validation as if the extension were not present. However,

if the RP does understand the extension, it must be processed in the same way as it would if the component had been set to critical.

In the interest of interoperability it is recommended that most extensions be set to non-critical at the present time. However, because of their vital role in ensuring that only CAs issue certificates, it is recommended that **basicConstraints** and **keyUsage** extensions be always set to critical in CA certificates and **keyUsage** be set to critical in user certificates. Note that the **certificatePolicies** extension is somewhat unique. Even though these profiles recommend that it be set to non-critical, there are other mechanisms in these profiles that allow users to require policy to be present and processed in certificates.

3.2 CRL CONTENT

CRLs consist of a set of base fields plus a set of extensions. Some extensions apply to the complete CRL and others apply only to a single certificate listed as revoked on the CRL. This section includes some generic background that applies to population of the base CRL fields and extensions.

3.2.1 Base CRL Fields

3.2.1.1 Version

There have been two versions of the syntax for CRLs defined in X.509. The only version that allows extensions to be included in CRLs is version 2. Because of the key role extensions play in determining the scope of a CRL and in providing details about specific revocations, all CRLs should now be issued as version 2 CRLs.

3.2.1.2 Signature

This field contains the algorithm identifier for the algorithm used by the CA to sign the CRL. It must contain the same algorithm identifier as the **signatureAlgorithm** field in the ASN.1 sequence **CertificateList**.

For interoperability reasons, it is recommended that this field contain the algorithm identifier for the RSA PKCS #1 Version 1.5 signature algorithm along with the appropriate SHA hash function for the desired security level. See the Annex for information on security levels and which hash function to use. Based on Annex A and the validity periods of CA certificates contained in these profiles, all CRLs should be signed by CA public keys that are 2048 bits long and the signatures should be created using the SHA-256 hash function. This recommendation is common to all profiles in this paper.

While there are no known attacks on the RSA PKCS #1 Version 1.5 algorithm, some high security applications may wish to use the RSA-PSS algorithm which has more attractive security characteristics. Similarly, some applications may want to use the ECDSA signature algorithm, which uses elliptic curve cryptography, because of the performance advantages that it provides at high security levels. See [ECC] for a discussion of when it is appropriate to use elliptic curve based techniques.

These recommendations apply at the time of writing (2004) and will require updating in accordance with updates to the advice in Annex A.

3.2.1.3 Issuer

This field contains the Distinguished Name (DN) of the CRL issuer. The standards allow a scheme of indirect CRLs where an entity other than the CA that issued a certificate issues the relevant CRL(s). In general, however, the same CA that issues a certificate issues those CRLs. Therefore, in all profiles in this paper, the value of the **issuer** field should be the same as the value of the corresponding **issuer** field in certificates.

3.2.1.4 This Update

[X.509] allows this time to be represented as either **utcTime** or **generalizedTime**, however [PKIX] requires that dates through the year 2049 be encoded as **utcTime** and dates in 2050 or later be encoded as **generalizedTime**. Therefore all CRLs should currently be issued with their **thisUpdate** time in **utcTime** format, expressed as Greenwich Mean Time (Zulu) and in the format (YYMMDDHHMMSSZ). The value of “YY” should be less than “50”.

3.2.1.5 Next Update

Although this field is optional in [X.509], [PKIX] mandates its inclusion and it plays a role in enabling relying parties to determine whether the CRL they are processing is appropriate for use at the current time. Therefore this field should be present in all CRLs. [X.509] allows this time to be represented as either **utcTime** or **generalizedTime**, however [PKIX] requires that dates through the year 2049 be encoded as **utcTime** and dates in 2050 or later be encoded as **generalizedTime**. Therefore all CRLs should currently be issued with their **nextUpdate** time in **utcTime** format, expressed as Greenwich Mean Time (Zulu) and in the format (YYMMDDHHMMSSZ). The value of “YY” should be less than “50”. It is recommended that the value in this field be no greater than 24 hours after the value in the “this update” field.

3.2.1.6 Revoked Certificates

This field is a list of entries for revoked certificates. Each entry always includes the serial number of a revoked certificate as well as its date/time of revocation. [X.509] allows this time to be represented as either **utcTime** or **generalizedTime**, however [PKIX] requires that dates through the year 2049 be encoded as **utcTime**. Therefore each CRL entry should currently include **revocationDate** in **utcTime** format, expressed as Greenwich Mean Time (Zulu) and in the format (YYMMDDHHMMSSZ). The value of “YY” should be less than “50”. In addition **revokedCertificates** may include one or more extensions in the **crEntryExtensions** component. If the revocation reason is known, it should be included in the **reasonCode** CRL entry extension. Also, if the certificate was revoked for a key compromise reason, the **invalidityDate** CRL entry extension should be included. While this information does not impact validations done against the current time, it may impact future validations performed against a time of interest in the past. Because indirect CRLs are not widely supported, the **certificateIssuer** CRL entry extension should not be included. Because there is not yet widespread support for processing the **holdInstructionCode** CRL entry extension, it is recommended that this extension be excluded from CRLs.

3.2.2 CRL Extensions

There are a number of extensions standardized in [X.509] that apply to an entire CRL. Specific recommendations for each extension are addressed in the individual profiles later in this paper. In terms of general requirements, there are 2 CRL extensions that should be included in all CRLs.

3.2.2.1 Authority Key Identifier

The **authorityKeyIdentifier** extension identifies the public-key to be used in verifying the signature on the CRL. This extension should be included in all CRLs. The **keyIdentifier** component of the extension should always be present and include a 160 bit SHA-1 hash of just the public key. The **authorityCertIssuer** and **authorityCertSerialNumber** components can be excluded.

These recommendations apply at the time of writing (2004) and will require updating in accordance with updates to the advice in Annex A.

3.2.2.2 Issuing Distribution Point

The **issuingDistributionPoint** extension is relevant for partitioned as well as combined/full CRLs and therefore should be included in all CRLs. The **distributionPoint** component should be present and include at least the DN that indicates the directory entry in which the CRL was stored and an LDAP URL for retrieval of the same CRL. Either the **onlyContainsUserCerts** component or the **onlyContainsCACerts** components should be included. The **onlySomeReasons**, **indirectCRL** and **onlyContainsAttributeCerts** components should be excluded. Separate distribution point locations should be used for each CRL stream.

Other CRL extensions are covered in the specific profiles.

4. COMMON PROFILES

The recommendations for user certificate content and CRL content are applicable to all three trust models discussed in this paper. These profiles are documented in this section and referenced from each of the trust model sections.

4.1 USER CERTIFICATES

The following profile provides recommendations for the content of user certificates regardless of the trust model. Many of these recommendations (e.g. the particular algorithms, naming attributes etc) are made with an emphasis on interoperability and broad applicability. The particular certificate policy under which each certificate is issued determines its actual appropriateness to a given application or transaction.

TABLE 1: USER CERTIFICATES

FIELD/EXTENSION	P	C	VALUE/COMMENTS
Base certificate fields			
version	P	N/A	2 (indicates v3 certificate)
serialNumber	P	N/A	Monotonically increasing integer, beginning with "1" for the first certificate
signature	P	N/A	algorithm is present and includes the OID: 1.2.840.113549.1.1.11 parameters is present with the NULL value
issuer	P	N/A	CA DN encoded in PrintableString . Attributes include domainComponent (dc) and commonName (cn)
validity	P	N/A	notBefore set to the time of certificate issuance notAfter set to 2 years beyond the time of certificate issuance encoded as UTCTime format YYMMDDHHMMSSZ
subject	P	N/A	User DN encoded in PrintableString . Attributes include domainComponent (dc) and commonName (cn)
subjectPublicKeyInfo	P	N/A	The algorithm component of AlgorithmIdentifier includes the OID 1.2.840.113549.1.1.1 The parameters component of AlgorithmIdentifier is present and includes the NULL value The subjectPublicKey includes a 1024 bit RSA public key
issuerUniqueIdIdentifier	A	N/A	
subjectUniqueIdIdentifier	A	N/A	
Extensions			
authorityKeyIdentifier	P	nc	keyIdentifier includes 160 bit SHA-1 hash of the public key used to verify the CA signature on the certificate authorityCertIssuer and authorityCertSerialNumber are excluded
subjectKeyIdentifier	P	nc	Includes 160 bit SHA-1 hash of the subjectPublicKey component of the subjectPublicKeyInfo field of the certificate

FIELD/EXTENSION	P	C	VALUE/COMMENTS
authorityInfoAccess	P	nc	caIssuers is present and includes LDAP URI pointing to the directory entry of the issuing CA
basicConstraints	P	nc	ca boolean set to FALSE pathLenConstraint is excluded
keyUsage	P	c	digitalSignature bit is set for verification certificates keyEncipherment bit is set for encryption certificates
cRLDistributionPoints	P	nc	distributionPoint is present <ul style="list-style-type: none"> directoryName includes the DN where the CRL is located (immediately subordinate to the CA entry in the directory) LDAP URI is included and points to the directory entry holding the CRL reasons and cRLIssuer are excluded
certificatePolicies	P	nc	policyIdentifier includes at least one OID representing the policy/policies under which this certificate was issued policyQualifiers is excluded
subjectAltName	P	nc	rfc822Name is included
privateKeyUsagePeriod	O	nc	Included in verification certificates only. notBefore set to the time of certificate issuance notAfter set to 70% of the certificate lifetime (17 months beyond the time of certificate issuance) allowing sufficient time for rekeying before expiry of the certificate encoded as UTCTime format YYMMDDHHMMSSZ
extKeyUsage	O	nc	Included only if necessary for specific applications. Values restricted to those defined in RFC 3280
subjectDirectoryAttributes	A	N/A	
freshestCRL	A	N/A	
policyConstraints	A	N/A	
inhibitAnyPolicy	A	N/A	
policyMappings	A	N/A	
nameConstraints	A	N/A	
issuerAltName	A	N/A	
subjectInfoAccess	A	N/A	

4.2 CERTIFICATE REVOCATION LISTS

The following profile provides recommendations for the content of Certificate Revocation Lists (CRL) regardless of the trust model. Many of these recommendations (e.g. the particular algorithms, naming attributes etc) are made with an emphasis on interoperability and broad applicability. While this profile

does recommend partitioning CRLs to reduce their size, the same profile can be used in environments where partitioning is not supported.

TABLE 2: CRLS

FIELD/EXTENSION	P	C	VALUE
Base CRL fields			
version	P	N/A	1 (indicates v2 CRL)
signature	P	N/A	algorithm is present and includes the OID: 1.2.840.113549.1.1.11 parameters is present with the NULL value
issuer	P	N/A	Issuing CA DN identical to the value in the issuer field of certificates issued by the same CA
thisUpdate	P	N/A	The date/time of CRL issuance in UTCTime format YYMMDDHHMMSSZ
nextUpdate	P	N/A	Value is no greater than 24 hours beyond the value of thisUpdate in UTCTime format YYMMDDHHMMSSZ
revokedCertificates	P	N/A	serialNumber and revocationDate components included for each entry crEntryExtensions present in each entry as outlined below
CRL Entry Extensions (crEntryExtensions)			
reasonCode	P	nc	Included unless the reason is unspecified. If the reason is unspecified the extension is excluded
invalidityDate	O	nc	Included only if reasonCode extension is present and indicates keyCompromise or cACompromise
holdInstructionCode	A	N/A	
certificateIssuer	A	N/A	
CRL Extensions (crExtensions)			
authorityKeyIdentifier	P	nc	keyIdentifier is present and includes 160 bit SHA-1 hash of the public key used to verify the CA signature on the CRL authorityCertIssuer and authorityCertSerialNumber are excluded
cRLNumber	P	nc	Monotonically increasing positive integer starting at value '1'
issuingDistributionPoint	P	c	distributionPoint component is included with the fullName element for at least the DN name of the directory entry containing the CRL nameRelativeToCRLIssuer is excluded onlyContainsUserCerts and onlyContainsAuthorityCerts components are set accordingly for CRLs and ARLs onlySomeReasons , indirectCRL and onlyContainsAttributeCerts components are excluded
issuerAltName	A	N/A	
crIScope	A	N/A	
statusReferrals	A	N/A	
cRLStreamIdentifier	A	N/A	

FIELD/EXTENSION	P	C	VALUE
orderedList	A	N/A	
deltaInfo	A	N/A	
deltaCRLIndicator	A	N/A	
baseUpdateTime	A	N/A	
freshestCRL	A	N/A	

5. DISTRIBUTED TRUST MODEL

In the distributed trust model (sometimes referred to as the “mesh” trust model), each CA operates autonomously as a standalone CA, providing services to its own community of users (subscribers and relying parties). The CA issues certificates to those users, provides its public key to anchor trust decisions made by those users, and sets the security policy that determines the conditions under which those users trust certificates (internal and external).

Each standalone CA establishes independent direct cross-certification relationships with other CAs as required to facilitate trusted transactions between its own community of users and the community of users managed by remote CAs. The cross-certification relationships need to reflect the business needs of both communities. Each CA is primarily responsible for protecting the interests of its own user community.

Because the standalone CA establishes a separate cross-certification relationship with each and every external CA of interest directly, the cross-certificates issued by the standalone CA can each be tailored to the specific needs of that relationship. Name constraints are an effective tool, in this trust model, to constrain remote certificates to only those issued to users in the specific organizations of interest and are therefore recommended for inclusion in cross-certificates in the distributed trust model. Name constraints can also be used to increase the efficiency of path development algorithms.

Because each CA in a distributed trust model is a standalone CA it is unlikely that it will issue certificates under the same certificate policies as other standalone CAs. However, it is likely that at least a subset of the policies may be deemed equivalent. In these cases, a combination of the certificate policies extension and the policy mappings extension is recommended to enable equivalent policies to be acceptable replacements for local users.

Specific requirements for policy may vary, even among the local community of users. For example, some applications may require a certificate to be valid for a “high assurance” policy while others may only require a medium assurance policy. If this is the case, the infrastructure controls established through extensions in the cross-certificates must be supplemented with process constraints that initialize the conditions for path validation for each relevant subset of the user community. Further information about the relationship between infrastructure constraints and process constraints can be found in [PVAL].

The recommendations specific to the distributed trust model are contained in two profiles. The first recommends content for cross-certificates issued by a standalone CA to other CAs. The second recommends process constraints to handle differing user requirements for policy.

5.1 USER CERTIFICATES

User certificates in the distributed trust model are covered by the common profile for user certificates in section 4.1 of this paper.

5.2 CERTIFICATE REVOCATION LISTS

CRLs in the distributed trust model are covered by the common profile for CRLs in section 4.2 of this paper.

5.3 CROSS CERTIFICATES

This profile recommends content for cross-certificates issued in a distributed trust model environment. Note that the recommended constraints include a name constraint to help contain the bounds of the federation established through the cross-certification. The rationale for recommending a name constraint rather than a path length constraint is that a path length constraint requires knowledge of the internal network of CAs in remote organizations in order to establish an effective constraint. Not only may this

information not be shared, but the topology of the remote network may change from time to time, unknown to a CA that issued a cross-certificate to one in that network. Name constraints can provide effective control over the community of users whose certificates should be considered trusted, independent of the CA topology.

TABLE 3: CROSS CERTIFICATES

FIELD/EXTENSION	P	C	VALUE/COMMENTS
Base certificate fields			
version	P	N/A	2 (indicates v3 certificate)
serialNumber	P	N/A	Monotonically increasing integer
signature	P	N/A	algorithm is present and includes the OID: 1.2.840.113549.1.1.11 parameters is present with the NULL value
issuer	P	N/A	CA DN encoded in PrintableString . Attributes include domainComponent (dc) and commonName (cn)
validity	P	N/A	notBefore set to the time of certificate issuance notAfter set to 5 years beyond the time of certificate issuance encoded as UTCTime format YYMMDDHHMMSSZ
subject	P	N/A	Subject CA DN. This is the same name used by the subject CA when it is the issuer of certificates
subjectPublicKeyInfo	P	N/A	The algorithm component of AlgorithmIdentifier includes the OID 1.2.840.113549.1.1.1 The parameters component of AlgorithmIdentifier is present and includes the NULL value The subjectPublicKey includes a 2048 bit RSA public key
issuerUniqueId	A	N/A	
subjectUniqueId	A	N/A	
Extensions			
authorityKeyIdentifier	P	nc	keyIdentifier includes 160 bit SHA-1 hash of the public key used to verify the CA signature on the cross certificate authorityCertIssuer and authorityCertSerialNumber are excluded
subjectKeyIdentifier	P	nc	Includes 160 bit SHA-1 hash of the subjectPublicKey component of the subjectPublicKeyInfo field of the certificate, or the value provided by the certificate requester
authorityInfoAccess	P	nc	caIssuers accessMethod is present and includes LDAP URI pointing to the directory entry and appropriate attributes that include inbound and outbound CA certificates for the issuer CA
subjectInfoAccess	P	nc	caRepository accessMethod is present and includes LDAP URI pointing to the directory entry and appropriate attributes that include inbound and outbound CA certificates for the subject CA
basicConstraints	P	c	ca boolean set to TRUE pathLenConstraint is excluded

FIELD/EXTENSION	P	C	VALUE/COMMENTS
keyUsage	P	c	keyCertSign and crIISign bits are set
cRLDistributionPoints	P	nc	distributionPoint is present <ul style="list-style-type: none"> directoryName includes the DN where the CRL is located (immediately subordinate to the CA entry in the directory) LDAP URI is included and points to the directory entry holding the CRL reasons and cRLIssuer are excluded
certificatePolicies	P	nc	policyIdentifier includes OIDs for all issuer CA policies relevant to the specific cross-certification relationship policyQualifiers is excluded
inhibitAnyPolicy	P	nc	SkipCerts is set to "0"
policyMappings	P	nc	Includes a mapping from each of the issuer policy OIDs in the certificatePolicies extension to its equivalent policy OID in the subject CA's domain.
nameConstraints	P	nc	Includes at least one permittedSubtree component that constrains the DN namespace to that which includes subscribers and any necessary intermediate CAs in the subject CA domain.
policyConstraints	O	nc	requireExplicitPolicy may be included if this requirement applies to ALL relying parties in the issuer CA domain. If present, skipCerts should be set to "0". If the requirement applies only to some relying parties (e.g. the purchasing requires high assurance policies in all certificates used for purchases, but other employees do not require policy for inter-company email exchanges), this extension should be excluded inhibitPolicyMapping should be excluded
privateKeyUsagePeriod	A	N/A	
extKeyUsage	A	N/A	
subjectAltName	A	N/A	
subjectDirectoryAttributes	A	N/A	
freshestCRL	A	N/A	
issuerAltName	A	N/A	

5.4 PROCESS CONSTRAINTS

Process constraints enable subsets of users to impose different constraints on the set of certificates that should be considered valid. Process constraints are implemented as initial settings for the standard set of variable inputs to the path validation process. Because policy mappings are expected to be used in cross-certificates these are not inhibited. Because the special value "anyPolicy" is inhibited for all users through the recommendation that **inhibitAnyPolicy** be included in content for cross-certificates, the corresponding process constraint need not be set. If policy requirements differ among the users, process constraints are recommended for users that require policy. Table 4 assumes this is the case. Note that in this situation, the **policyConstraints** extension would not be necessary in cross certificates.

TABLE 4: DISTRIBUTED TRUST MODEL – PROCESS CONSTRAINTS

PATH PROCESSING CONFIGURABLE INPUTS	VALUE/COMMENTS
<i>initial-policy-set</i>	Set to the specific policies of interest for the relying party, if any
<i>initial-explicit-policy</i>	Set (if policy is important to the particular relying party: i.e. some certificates may be rejected on the basis of policy)
<i>initial-policy-mapping-inhibit</i>	Unset
<i>initial-inhibit-policy</i>	Unset

6. HIERARCHICAL TRUST MODEL

In the hierarchical trust model there is a single “root CA” that serves as the trust anchor for the complete hierarchy. The Root CA issues certificates to “subordinate CAs”. Depending on the depth of the hierarchy, these first level subordinate CAs may issue certificates directly to users, or may in turn issue certificates to another level of subordinate CAs. Regardless of the depth of the hierarchy, users obtain certificates from the subordinate CAs at the lowest level of the hierarchy but all users have the single root CA as their trust anchor. A single common set of policies are applied to the entire hierarchy, although some branches may operate under only a subset of those policies. Subordinate CAs that issue certificates to users are responsible for management and specific policy associated with those users and process constraints under which those users perform path validation.

The root CA may also establish cross-certification relationships with CAs outside of the hierarchy to facilitate secure transactions across that federation. In all cases, for both users internal to the hierarchy and users outside the hierarchy, all certification paths that are built to a hierarchical user's certificate include exactly the same subset of certificates from the hierarchical root CA through any subordinate CAs to that user certificate. This means that any constraints applied in subordinate CA certificates within the hierarchy impact users inside that hierarchy as well as external users.

6.1 USER CERTIFICATES

User certificates in the hierarchical trust model are covered by the common profile for user certificates in section 4.1 of this paper.

6.2 CERTIFICATE REVOCATION LISTS

CRLs in the hierarchical trust model are covered by the common profile for CRLs in section 4.2 of this paper.

6.3 SUBORDINATE CERTIFICATES

This profile recommends content for certificates issued to subordinate CAs by a root CA or by a higher level subordinate CA. Because of the nature of hierarchies and the fact that external cross-certification only happens at the root, there is little need for constraints to be included in these certificates in order to control federation. Also, because each entity in a hierarchy can only have one superior entity, path development is generally performed from the subscriber certificate toward the root CA. This greatly reduces the value of constraints in certificates for path development purposes as well.

Policy constraints are recommended to be excluded from subordinate certificates, primarily because their presence affects both internal users in the hierarchy as well as external users cross-certified with the hierarchy, because all paths to the end-user certificates include the subordinate certificates regardless of where the relying parties are located. Such constraints would cause problems for external users that cannot process policy and it is primarily the task of their root CA to determine constraints on their use of remote certificates.

Policy mappings are also excluded from this profile. This is because hierarchies operate under a uniform set of policies and mappings are therefore unnecessary.

Name constraints are recommended to be included in subordinate certificates, primarily because they can help improve the efficiency of some path development algorithms. Although this may not be as useful within the hierarchy itself, as paths tend to be built from the subscriber certificate toward the trust anchor, it can still be useful when a hierarchy is cross-certified with another domain and paths may be built in the opposite direction. They can also be helpful in limiting the set of users to whom subordinate CAs can issue certificates. If they issue certificates to users outside the name constraints, this constraint will cause

certificate users to fail validation on those certificates.

TABLE 5: SUBORDINATE CA CERTIFICATES

FIELD/EXTENSION	P	C	VALUE/COMMENTS
Base certificate fields			
version	P	N/A	2 (indicates v3 certificate)
serialNumber	P	N/A	Monotonically increasing integer
signature	P	N/A	algorithm is present and includes the OID: 1.2.840.113549.1.1.11 parameters is present with the NULL value
issuer	P	N/A	CA DN encoded in PrintableString . Attributes include domainComponent (dc) and commonName (cn)
validity	P	N/A	notBefore set to the time of certificate issuance notAfter set to 5 years beyond the time of certificate issuance encoded as UTCTime format YYMMDDHHMMSSZ
subject	P	N/A	Subject CA DN exactly as in the issuer field of certificates issued by the subject CA
subjectPublicKeyInfo	P	N/A	The algorithm component of AlgorithmIdentifier includes the OID 1.2.840.113549.1.1.1 The parameters component of AlgorithmIdentifier is present and includes the NULL value The subjectPublicKey includes a 2048 bit RSA public key
issuerUniqueId	A	N/A	
subjectUniqueId	A	N/A	
Extensions			
authorityKeyIdentifier	P	nc	keyIdentifier includes 160 bit SHA-1 hash of the public key used to verify the CA signature on the certificate authorityCertIssuer and authorityCertSerialNumber are excluded
subjectKeyIdentifier	P	nc	Includes 160 bit SHA-1 hash of the subjectPublicKey component of the subjectPublicKeyInfo field of the certificate or includes the value received from the requester
authorityInfoAccess	P	nc	caIssuers accessMethod is present and includes LDAP URI pointing to the directory entry for the issuer CA
subjectInfoAccess	P	nc	caRepository accessMethod is present and includes LDAP URI pointing to the directory entry for the subject CA
basicConstraints	P	c	cA boolean set to TRUE pathLenConstraint is excluded
keyUsage	P	c	keyCertSign and crISign bits are set
cRLDistributionPoints	P	nc	distributionPoint is present <ul style="list-style-type: none"> directoryName includes the DN where the CRL is located

FIELD/EXTENSION	P	C	VALUE/COMMENTS
			(immediately subordinate to the CA entry in the directory) <ul style="list-style-type: none"> LDAP URI is included and points to the directory entry holding the CRL reasons and cRLIssuer are excluded
certificatePolicies	P	nc	policyIdentifier includes OIDs for all issuer CA policies under which the subordinate CA may issue certificates policyQualifiers is excluded
inhibitAnyPolicy	P	nc	SkipCerts is set to "0"
nameConstraints	P	nc	Includes at least one permittedSubtree component that constraints the DN namespace to that within which the subordinate CA may issue certificates
policyConstraints	A	N/A	
policyMappings	A	N/A	
privateKeyUsagePeriod	A	N/A	
extKeyUsage	A	N/A	
subjectAltName	A	N/A	
subjectDirectoryAttributes	A	N/A	
freshestCRL	A	N/A	
issuerAltName	A	N/A	

6.4 ROOT CA – CROSS CERTIFICATES

In a hierarchy, the root CA establishes any necessary cross-certification relationships with CAs outside the hierarchy. As with the standalone CA in a distributed trust model, the root CA can establish direct cross-certifications with each CA of interest. The hierarchical root CA is responsible for protecting the interests of the user community within its entire hierarchy when establishing such cross-certifications. Cross-certificates issued by hierarchical root CAs to external CAs are covered by the profile for cross-certificates in the distributed trust model in section 5.3 of this paper.

6.5 PROCESS CONSTRAINTS

Process constraints enable subsets of users to impose different constraints on the set of certificates that should be considered valid. Process constraints are implemented as initial settings for the standard set of variable inputs to the path validation process. Because policy mappings are expected to be used in cross-certificates issued by the root CA to external CAs these are not inhibited for many users (however may be useful for some). Because the special value "anyPolicy" is inhibited for all users through the recommendation that the **inhibitAnyPolicy** extension be included in content for cross-certificates, the corresponding process constraint need not be set. Because policy constraints are excluded from the subordinate certificates in these profiles, and because requirements may differ among the users within the hierarchy, process constraints are recommended for users that need to discriminate on the basis of policy.

TABLE 6: HIERARCHICAL TRUST MODEL – PROCESS CONSTRAINTS

PATH PROCESSING CONFIGURABLE INPUTS	VALUE/COMMENTS
<i>initial-policy-set</i>	Set to the specific policies of interest for the relying party, if any
<i>initial-explicit-policy</i>	Set (if policy is important to the particular relying party)
<i>initial-policy-mapping-inhibit</i>	Set (if mappings are to be inhibited for this particular relying party (e.g. if only certificates in the hierarchy are to be used and not external certificates from domains with the root CA has cross-certified))
<i>initial-inhibit-policy</i>	Unset

7. BRIDGE TRUST MODEL

In this environment, a single CA acts as a 'bridge' to connect PKI communities together under a common infrastructure. This model reduces the need for standalone CAs and hierarchical root CAs to enter into cross-certification relationships directly with each external CA of interest. Instead, those CAs enter into a cross-certification relationship with a bridge CA and that automatically provides them with relationships with all other CAs that are also connected to that same bridge. The CAs that cross-certify with a bridge CA are called "spoke" CAs.

A bridge CA typically connects spoke CAs that have a common bond. For example, a bridge CA operated on behalf of a federal government would typically have all federal agency CAs as its spoke CAs. In a particular industry vertical, there could be a bridge CA interconnecting all CAs in that sector (e.g. pharmaceuticals, banks etc). Note that, in general, any particular entity (e.g. agency/bank) would have a single spoke CA. If that entity has multiple standalone CAs operating in a local distributed trust model, then only one of those would act as the spoke CA for that entity.

The bridge CA is generally managed by a policy authority that includes representation from the spoke CA stakeholders, ensuring that it serves the needs of the whole community. In addition to interconnecting the spoke CAs, a bridge CA can enter into cross-certification relationships with external CAs of interest to the overall bridge community (e.g. a federal government bridge cross-certifying with a state government). While these external CAs appear the same as spoke CAs in the architecture, there may be additional constraints imposed to satisfy the business needs of the bridge CA community. Therefore these certificates typically resemble the cross-certificates profiled for the mesh/distributed trust model, more than the spoke/bridge certificates. Constraints applying to names are therefore best suited to be imposed by the bridge CA. However, each entity represented by a spoke CA may have differing requirements for policy handling. Therefore the bridge CA needs to ensure that the appropriate policies are represented in certificates it issues but the spoke CAs are generally responsible for imposing policy related constraints. The bridge CA is generally liberal in certificates it issues, based on its role as a facilitator, while spoke CAs are generally more restrictive based on their role as a protector. Because it is even less likely, in the bridge trust model, that spoke CAs or the bridge CA have detailed and current information about the CA topology of each entity, path length constraints are excluded from the profiles for this trust model.

7.1 USER CERTIFICATES

User certificates in the bridge trust model are covered by the common profile for user certificates in section 4.1 of this paper.

7.2 CERTIFICATE REVOCATION LISTS

CRLs in the bridge trust model are covered by the common profile for CRLs in section 4.2 of this paper.

7.3 SPOKE CA: CERTIFICATES ISSUED TO BRIDGE CA

In general, certificates issued to a bridge CA contain any policy-related constraints that apply to the complete user community represented by that spoke CA. Name constraints are typically handled by the bridge CA and excluded from certificates issued by the spoke CAs.

TABLE 7: SPOKE CA: CERTIFICATES ISSUED TO BRIDGE CA

FIELD/EXTENSION	P	C	VALUE/COMMENTS
Base certificate fields			
version	P	N/A	2 (indicates v3 certificate)
serialNumber	P	N/A	Monotonically increasing integer
signature	P	N/A	algorithm is present and includes the OID: 1.2.840.113549.1.1.11 parameters is present with the NULL value
issuer	P	N/A	CA DN encoded in PrintableString . Attributes include domainComponent (dc) and commonName (cn)
validity	P	N/A	notBefore set to the time of certificate issuance notAfter set to 5 years beyond the time of certificate issuance encoded as UTCTime format YYMMDDHHMMSSZ
subject	P	N/A	Subject CA DN exactly as in the issuer field of certificates issued by the subject CA
subjectPublicKeyInfo	P	N/A	The algorithm component of AlgorithmIdentifier includes the OID 1.2.840.113549.1.1.1 The parameters component of AlgorithmIdentifier is present and includes the NULL value The subjectPublicKey includes a 2048 bit RSA public key
issuerUniqueId	A	N/A	
subjectUniqueId	A	N/A	
Extensions			
authorityKeyIdentifier	P	nc	keyIdentifier includes 160 bit SHA-1 hash of the public key used to verify the CA signature on the certificate authorityCertIssuer and authorityCertSerialNumber are excluded
subjectKeyIdentifier	P	nc	Includes 160 bit SHA-1 hash of the subjectPublicKey component of the subjectPublicKeyInfo field of the certificate or the value provided by the requester
authorityInfoAccess	P	nc	caIssuers accessMethod is present and includes LDAP URI pointing to the directory entry for the issuer CA
subjectInfoAccess	P	nc	caRepository accessMethod is present and includes LDAP URI pointing to the directory entry for the subject CA
basicConstraints	P	c	ca boolean set to TRUE pathLenConstraint is excluded
keyUsage	P	c	keyCertSign and crISign bits are set
cRLDistributionPoints	P	nc	distributionPoint is present <ul style="list-style-type: none"> directoryName includes the DN where the CRL is located

FIELD/EXTENSION	P	C	VALUE/COMMENTS
			(immediately subordinate to the CA entry in the directory) <ul style="list-style-type: none"> LDAP URI is included and points to the directory entry holding the CRL reasons and cRLIssuer are excluded
certificatePolicies	P	nc	policyIdentifier includes OIDs for all spoke CA policies relevant to the bridge CA relationship policyQualifiers is excluded
inhibitAnyPolicy	P	nc	SkipCerts is set to "0"
policyConstraints	O	nc	requireExplicitPolicy may be included if this requirement applies to ALL relying parties in the spoke CA domain. If present, skipCerts should be set to "0"
policyMappings	O	nc	If the spoke CA and the bridge CA do not use common policy identifiers, this extension should be present and include a mapping from each of the spoke CA policy OIDs in the certificatePolicies extension to its equivalent policy OID in the subject CA's domain. Otherwise this extension is excluded
nameConstraints	A	N/A	
privateKeyUsagePeriod	A	N/A	
extKeyUsage	A	N/A	
subjectAltName	A	N/A	
subjectDirectoryAttributes	A	N/A	
freshestCRL	A	N/A	
issuerAltName	A	N/A	

7.4 BRIDGE CA: CERTIFICATES ISSUED TO SPOKE CA

In general, certificates issued to a spoke CA include necessary policy-related information to enable spoke CAs and RPs to enforce the policy constraints they deem necessary, but don't impose policy constraints of their own. These certificates typically include constraints on the federation supported by the bridge through name constraints. These certificates constrain the set of identities that can be trusted through the bridge, but facilitate any validations regardless of additional constraints. This satisfies the role of the bridge CA as a facilitator and not a controller of trust.

TABLE 8: BRIDGE CA: CERTIFICATES ISSUED TO SPOKE CA

FIELD/EXTENSION	P	C	VALUE/COMMENTS
Base certificate fields			
version	P	N/A	2 (indicates v3 certificate)
serialNumber	P	N/A	Monotonically increasing integer
signature	P	N/A	algorithm is present and includes the OID:

FIELD/EXTENSION	P	C	VALUE/COMMENTS
			1.2.840.113549.1.1.11 parameters is present with the NULL value
issuer	P	N/A	CA DN encoded in PrintableString . Attributes include domainComponent (dc) and commonName (cn)
validity	P	N/A	notBefore set to the time of certificate issuance notAfter set to 5 years beyond the time of certificate issuance encoded as UTCTime format YYMMDDHHMMSSZ
subject	P	N/A	Subject CA DN exactly as in the issuer field of certificates issued by the subject CA
subjectPublicKeyInfo	P	N/A	The algorithm component of AlgorithmIdentifier includes the OID 1.2.840.113549.1.1.1 The parameters component of AlgorithmIdentifier is present and includes the NULL value The subjectPublicKey includes a 2048 bit RSA public key
issuerUniqueIdIdentifier	A	N/A	
subjectUniqueIdIdentifier	A	N/A	
Extensions			
authorityKeyIdentifier	P	nc	keyIdentifier includes 160 bit SHA-1 hash of the public key used to verify the CA signature on the certificate authorityCertIssuer and authorityCertSerialNumber are excluded
subjectKeyIdentifier	P	nc	Includes 160 bit SHA-1 hash of the subjectPublicKey component of the subjectPublicKeyInfo field of the certificate
authorityInfoAccess	P	nc	caIssuers accessMethod is present and includes LDAP URI pointing to the directory entry for the issuer CA
subjectInfoAccess	P	nc	caRepository accessMethod is present and includes LDAP URI pointing to the directory entry for the subject CA
basicConstraints	P	c	ca boolean set to TRUE pathLenConstraint is excluded
keyUsage	P	c	keyCertSign and crISign bits are set
cRLDistributionPoints	P	nc	distributionPoint is present <ul style="list-style-type: none"> directoryName includes the DN where the CRL is located (immediately subordinate to the CA entry in the directory) LDAP URI is included and points to the directory entry holding the CRL reasons and cRLIssuer are excluded
certificatePolicies	P	nc	policyIdentifier includes OIDs for all bridge CA policies relevant to the particular spoke CA relationship policyQualifiers is excluded
nameConstraints	P	nc	Includes at least one permittedSubtree component that constraints

FIELD/EXTENSION	P	C	VALUE/COMMENTS
			the DN namespace to that which includes subscribers and any necessary intermediate CAs in the subject CA domain.
inhibitAnyPolicy	A	N/A	
policyConstraints	A	N/A	
policyMappings	A	N/A	
privateKeyUsagePeriod	A	N/A	
extKeyUsage	A	N/A	
subjectAltName	A	N/A	
subjectDirectoryAttributes	A	N/A	
freshestCRL	A	N/A	
issuerAltName	A	N/A	

7.5 BRIDGE CA: CROSS CERTIFICATES ISSUED TO EXTERNAL CA

If a bridge CA issues cross-certificates to external CAs outside the bridge community, including other bridge CAs, those certificates are covered by the cross-certificates profile for the distributed trust model in section 5.3 of this paper.

7.6 PROCESS CONSTRAINTS

Process constraints enable the local CA to manage a set of variable conditions under which certificates should be considered valid by the users served by that CA. Process constraints are implemented as initial settings for the standard set of variable inputs to the path validation process. If policy mappings are required within the bridge community, then initial-policy-mapping-inhibit should not be set for any relying parties. However, if policy mappings are not required within the bridge community, this setting may be useful for some relying parties. Because the special value “anyPolicy” is inhibited for all users through the recommended content for cross-certificates issued to the bridge CA, the corresponding process constraint need not be set. Because policy requirements may differ among the users, process constraints are recommended for users that require policy.

TABLE 9: BRIDGE TRUST MODEL – PROCESS CONSTRAINTS

PATH PROCESSING CONFIGURABLE INPUTS	VALUE/COMMENTS
<i>initial-policy-set</i>	Set to the specific policies of interest for the relying party, if any
<i>initial-explicit-policy</i>	Set (if policy is important to the particular relying party)
<i>initial-policy-mapping-inhibit</i>	Set (if mappings are to be inhibited for this particular relying party (e.g. if only certificates in a local hierarchy are to be used and not external certificates such as those through the bridge))
<i>initial-inhibit-policy</i>	Unset

8. SUMMARY

There are a large number of options and extensions that can be contained in certificates and CRLs. The base standards, including [PKIX] leave a great amount of flexibility in this area and entities deploying a CA need to make their own decisions about certificate and CRL content. In some environments, specific profiles have been defined, but in many cases these have not. Even where these profiles exist, they may leave several options for the deploying entity. In general, such profiles focus solely on certificate and CRL content, ignoring process constraints and their relationship with infrastructure constraints. Deployments may be slowed while study is undertaken on which extensions and constraints are appropriate for a given domain.

The profiles in this paper provide recommendations for certificate and CRL content as well as process constraints in a number of scenarios based on the popular trust models. These profiles, based on the assumptions in section 2.1 are general recommendations that may be invalidated by circumstances for a given CA deployment.

9. ABOUT ENTRUST

Entrust, Inc. [Nasdaq: ENTU] is a world leader in securing digital identities and information, enabling businesses and governments to transform the way they conduct online transactions and manage relationships with customers, partners and employees. Entrust's solutions promote a proactive approach to security that provides accountability and privacy to online transactions and information. Over 1,200 enterprises and government agencies in more than 50 countries use Entrust's portfolio of security software solutions that integrate into the broad range of applications organizations use today to leverage the Internet and enterprise networks.

For more information, please visit <http://www.entrust.com>.

10. REFERENCES

[X.509] ITU-T Recommendation X.509 (2000 E): Information Technology, Open systems interconnection - The Directory: Public-key and attribute certificate frameworks.

[PKIX] RFC 3280: Internet X.509 Public Key Infrastructure Certificate and CRL Profile

[RFC 2247] Using Domains in LDAP/X.500 Distinguished Names

[PKITS]: Public Key Interoperability Test Suite - available at:

<http://csrc.nist.gov/pki/testing/x509paths.html>

[KMG] National Institute of Standards and Technology, Special Publication 800-57 (Draft): Recommendation for Key Management - Part 1: General Guideline, January 2003.

[ECC] Zuccherato, Robert, *Using A PKI Based Upon Elliptic Curve Cryptography - Examining the Benefits and Difficulties*. - available at

http://www.entrust.com/resources/download.cfm/21245/wp_EllipticCurveCryptography.pdf

11. ACRONYMS

CA	Certification Authority
CRL	Certificate Revocation List
DN	Distinguished Name
LDAP	Lightweight Directory Access Protocol
OCSP	Online Certificate Status Check Protocol
PKI	Public Key Infrastructure
PKIX	Public Key Infrastructure (X.509) IETF Working Group
RFC	Request for Comment
RP	Relying Party
URI	Uniform Resource Identifier

ANNEX A: KEY LENGTHS AND HASH FUNCTIONS

Based upon recommendations provided by NIST [KMG] the following are recommendations for appropriate key sizes and hash functions that will provide consistent security levels. These are general guidelines only. Individual implementers should their own security analysis to determine which key lengths are appropriate in their environment.

User RSA public keys which will only be used until about the year 2010 should be at least 1024 bits long. User elliptic curve public keys which will be used no later than the year 2010 should be at least 160 bits. Signatures created with these keys should use the SHA-1 hash function.

User and CA RSA public keys which will be used no later than the year 2030 should be at least 2048 bits long. User and CA elliptic curve public keys which will be used no later than the year 2030 should be at least 224 bits long. Signatures created with these keys should use the SHA-256 hash function.

User and CA RSA public keys which will be used beyond the year 2030 should be at least 3072 bits long. User and CA elliptic curve public keys which will beyond the year 2030 should be at least 256 bits long. Signatures created with these keys should use the SHA-256 hash function.

If longer key lengths are desired, then the SHA-384 hash function should be used with RSA keys that are 7680 bits long and with elliptic curve keys that are 384 bits. The SHA-512 hash function should be used once RSA keys are 15360 bits long and elliptic curve keys are 512 bits.