

Planning and Implementing Cross-Certification and Qualified Subordination Using Windows Server 2003

By Brian Komar and David Cross

Microsoft Corporation

Abstract

Microsoft® Windows® XP Professional and Microsoft Windows Server 2003® provide an integrated, public key infrastructure (PKI) that enables you to securely exchange information across the Internet, extranets, intranets, and applications. This white paper provides a technical reference and planning guide for PKI administrators who wish to perform PKI cross-certification, deploy bridge Certification Authorities (CAs), and understand how to implement qualified subordination in Windows Server 2003.

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein. The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2003 Microsoft Corporation. All rights reserved. Microsoft, Windows, and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Other product and company names mentioned herein may be the trademarks of their respective owners.

Microsoft Corporation • One Microsoft Way • Redmond, WA 98052-6399 • USA

Contents

[Acknowledgements v](#_Toc33527609)

[Introduction 1](#_Toc33527610)

[Scope 1](#_Toc33527611)

[Terms Used in This White Paper 2](#_Toc33527612)

[Overview 4](#_Toc33527613)

[Understanding Constraints 5](#_Toc33527614)

[Basic Constraints 6](#_Toc33527615)

[Name Constraints 7](#_Toc33527616)

[Name Constraint Processing 8](#_Toc33527617)

[Applying Name Constraints 10](#_Toc33527618)

* [Configuration 10](#_Toc33527619)
* [Relative Distinguished Name Constraints 11](#_Toc33527620)
* [DNS Name Constraints 12](#_Toc33527621)
* [Uniform Resource Identifier (URI) Constraints 13](#_Toc33527622)
* [RFC 822, E-mail, and UPN Constraints 14](#_Toc33527623)
* [IP Address Constraints 15](#_Toc33527624)

[Issuance Policy 15](#_Toc33527625)

[Policy Mapping 17](#_Toc33527626)

[Constraining Policy Mapping to Prevent Unexpected Trust 19](#_Toc33527627)

[Policy Qualifiers 21](#_Toc33527628)

[Application Policy 22](#_Toc33527629)

[Qualified Subordination Deployment Scenarios 24](#_Toc33527630)

[Restricting Namespaces 24](#_Toc33527631)

[Creating Trust Between Two Organizations 27](#_Toc33527632)

[Implementing Qualified Subordination to Trust All CAs 27](#_Toc33527633)

* [Cross-Certification Between Root CAs 28](#_Toc33527634)
* [Cross-Certification Between Subordinate and Root CAs 31](#_Toc33527635)

[Implementing Qualified Subordination to Limit Trust to Specific CAs 34](#_Toc33527636)

[Types of Constraints 36](#_Toc33527637)

[Creating Trust Between Multiple Organizations 37](#_Toc33527638)

[Viewing Certificates from the Three Organizations 39](#_Toc33527639)

[Adding New CAs to Existing CA Hierarchies 40](#_Toc33527640)

[Using Name Constraints with Bridge CAs 41](#_Toc33527641)

[Policy Constraints and Policy Constraint Mapping 41](#_Toc33527642)

[Application Policy 44](#_Toc33527643)

[Walkthrough 45](#_Toc33527644)

[Creating an Offline Bridge CA 45](#_Toc33527645)

[Configuring the CAPolicy.inf File 45](#_Toc33527646)

[Installing Certificate Services 46](#_Toc33527647)

[Disabling Delta CRLs at the Root CA 51](#_Toc33527648)

[Removing the Non-Required AIA and CDP Extensions 51](#_Toc33527649)

[Adding Required AIA and CDP Extensions 52](#_Toc33527650)

* [Adding the LDAP CDP Extension 52](#_Toc33527651)
* [Adding the HTTP CDP Extension 56](#_Toc33527652)
* [Adding the HTTP CDP Extension 56](#_Toc33527653)
* [Adding an HTTP AIA Extension 57](#_Toc33527654)

[Publishing CRLs and Certificates 59](#_Toc33527655)

* [Publishing the CRL and CA Certificate to a Web Server 60](#_Toc33527656)
* [Publishing the CRL and CA Certificate into Active Directory 61](#_Toc33527657)

[Installing the Enterprise Subordinate CAs 62](#_Toc33527658)

[Configuring the CAPolicy.inf File 62](#_Toc33527659)

[Installing Certificate Services 63](#_Toc33527660)

* [Installing the Enterprise Subordinate CA 63](#_Toc33527661)
* [Signing the Offline Certificate Request 68](#_Toc33527662)
* [Installing the Certificate at the Enterprise Subordinate CA 69](#_Toc33527663)

[Renewing an Existing Enterprise Subordinate CA 69](#_Toc33527664)

[Verifying the Inclusion of All Issuance Policies 71](#_Toc33527665)

[Creating a Qualified Subordination Signing Certificate at a Stand-alone CA 72](#_Toc33527666)

[Performing the Initial Certificate Request 72](#_Toc33527667)

[Issuing the Pending Certificate Request 73](#_Toc33527668)

* [Installing the Qualified Subordination Signing Certificate 74](#_Toc33527669)
* [Verifying the Qualified Subordination Signing Certificate 74](#_Toc33527670)

[Creating a Qualified Subordination Signing Certificate at an Enterprise CA 75](#_Toc33527671)

[Creating a Qualified Subordination version 2 Certificate Template 75](#_Toc33527672)

[Modifying the Cross-Certification Authority version 2 Certificate Template 79](#_Toc33527673)

[Configuring Windows Server 2003 Enterprise Edition to Issue the Certificate Templates 81](#_Toc33527674)

[Acquiring the Qualified Subordination Signing Certificate 82](#_Toc33527675)

* [Performing the Initial Certificate Request 82](#_Toc33527676)

[Issuing the Pending Certificate Request 83](#_Toc33527677)

* [Installing the Qualified Subordination Signing Certificate 83](#_Toc33527678)
* [Verifying the Qualified Subordination Signing Certificate 84](#_Toc33527679)

[Performing Cross-Certification 85](#_Toc33527680)

[Gathering the Necessary Files 85](#_Toc33527681)

[Generating the Cross-Certification Authority Certificate Request 86](#_Toc33527682)

[Processing the Cross-Certification Authority Certificate Request 90](#_Toc33527683)

[Verifying the Existence of the Cross Certificate 92](#_Toc33527684)

[Completing Cross-Certification 96](#_Toc33527685)

[Completing the Cross-Certification When a Bridge CA Is Implemented 97](#_Toc33527686)

[Appendix A – Policy.Inf 98](#_Toc33527687)

[Appendix B – CAPolicy.inf 101](#_Toc33527688)

[Appendix C – CMC Dump of a Qualified Subordination Request 103](#_Toc33527689)

# Acknowledgements

David Cross, Program Manager, Microsoft Corporation.

Trevor Freeman, Program Manager, Microsoft Corporation

Darren Canavor, Software Test Engineer, Microsoft Corporation

Sergio Dutra, Software Design Engineer, Microsoft Corporation

Krish Shenoy, Software Design Engineer, Microsoft Corporation

Jan De Clercq, Senior Consultant, Compaq Global Services

# Introduction

Qualified subordination is the process of cross-certifying Certification Authority (CA) hierarchies using basic, policy, naming, and application constraints to limit which certificates are accepted from partner CA hierarchies or a secondary hierarchy within the same organization. True cross-certification of CA hierarchies in a Windows 2000 network was not possible. The only available alternative was to define Certificate Trust Lists (CTLs) that trusted specific CAs and restricted certificate usage. By using qualified subordination, a CA administrator can clearly define which certificates issued by a partner’s PKI are trusted by the CA administrator’s organization. Qualified subordination also provides methods for compartmentalizing and controlling certificate issuance within an organization according to policy guidelines. Examples of both scenarios will be explained in this white paper.

For the purposes of this white paper, the term qualified subordination refers to a cross-certification between two CAs that implements basic constraints, name constraints, application constraints, policy constraints, or a mix of the four constraints. This limits which certificates are trusted from a partner or secondary CA hierarchy according to the rules and definitions defined in RFC 2459 and subsequently RFC 3280.

## Scope

The scope of this white paper is to describe the ways that qualified subordination is used to govern the relationship between multiple organizations’ PKI hierarchies. The white paper describes the various constraints that can be implemented to define the relationship between PKI hierarchies, presents qualified subordination scenarios, and provides walkthroughs of the qualified subordination process.

Terms Used in This White Paper

**Application Constraints** A constraint that limits what purposes a certificate can be used for in a qualified subordination configuration. A presented certificate must contain the required application constraint to be accepted by the partner organization.

**Authority Information Access (AIA)** A certificate extension that contains URL locations where the issuing CA’s certificate can be retrieved. The AIA extension can contain HTTP, FTP, LDAP, or FILE URLs.

**Authority Key Identifier (AKI)** A certificate extension used by the certificate chaining engine to determine what certificate was used to sign a presented certificate. The AKI can contain the issuer name and serial number, public key information, or no information at all. By matching the information in a certificate’s AKI extension to a CA certificate’s Subject Key Identifier (SKI) extension, a certificate chain can be built.

**CaPolicy.inf** A configuration file stored in the %SystemRoot% folder that defines configuration settings for CAs when they are installed and when the CA’s certificate is renewed.

**CRL Distribution Point (CDP)** A certificate extension that indicates where the certificate revocation list for a CA can be retrieved. This extension can contain multiple HTTP, FTP, File, or LDAP URLs for the retrieval of the CRL.

**Certificate Trust List (CTL)** A method of restricting certificates chaining to a designated CA for limited time periods or usages. It is used more prevalently in a Windows 2000 network. In a Windows Server 2003 environment, qualified subordination is the preferred method for restricting certificate usage between organizations.

**Certificate Revocation List (CRL)** A digitally signed list issued by a CA that contains a list of certificates issued by the CA that have been revoked. The listing includes the serial number of the certificate, the date that the certificate was revoked, and the revocation reason. Applications can perform CRL checking to determine a presented certificate’s revocation status.

**Cross-Certification** The process of issuing subordinate CA certificates for existing CAs that link two root CAs.

**Cross-Certification Authority Certificate** A certificate issued by one CA for another CA's signing key pair (that is, for another CA's public verification key).

**Name Constraint** A constraint that limits what names are permitted or excluded in certificate requests submitted to a CA.

**Issuance Policy Constraint** A constraint that defines what issuance practices must be followed for certificates to be trusted by your organization. Issuance policy object identifiers (OIDs) in your organization are mapped to the matching object identifiers in a partner organization, so that object identifiers in presented certificates are recognized by your PKI.

**Policy.inf** A configuration file that defines the constraints that are applied to a CA certificate when qualified subordination is defined.

**Public Key Infrastructure (PKI)** A PKI provides an organization with the ability to securely exchange data over a public network using public-key cryptography. A PKI consists of CAs that issue digital certificates, directories that store the certificates (including Active Directory in Windows 2000 and Windows Server 2003), and X.509 certificates that are issued to security entities on the network. The PKI provides validation of certificate-based credentials and ensures that the credentials are not revoked, corrupted, or modified.

**Qualified Subordination** The process of configuring cross-certification with basic constraints, name constraints, application constraints, and issuance policy constraints to govern what certificates are trusted from a partner organization.

# Overview

Qualified subordination allows an organization to extend its PKI trust hierarchy to other organizations and secondary hierarchies within the same organization. Qualified subordination is also sometimes referred to as “cross-certification”. While extension of PKI trust was possible in a Windows 2000 network using CTLs, CTLs were limited in how restrictions could be defined between the two organizations. Qualified subordination provides a more flexible and manageable trust mechanism.

When implemented using qualified subordination, each qualified subordinate CA can have rules defined that do the following:

* Define the namespaces for which your PKI hierarchy will issue and accept trusted certificates
* Specify the acceptable uses of certificates issued by a qualified subordinate CA
* Define what issuance practices must be followed for a certificate issued by the qualified subordinate CA to be considered valid
* Create a managed trust between separate certification hierarchies

Windows Server 2003 Enterprise Edition provides the necessary tools to configure qualified subordination between CAs, so that two organizations can define how certificates will be trusted between the organizations. These tools include the following:

* **Version 2 certificate template** Allows CA administrators to modify certificate templates to meet their business purposes. Qualified subordination requires the use of version 2 certificate templates to include policy constraints. Modifications to version 2 templates can include:
* Creating a new certificate template by duplicating and renaming an existing template
* Modifying template properties such as certificate validity period, renewal period, cryptographic service provider (CSP), key size, and key archival settings
* Establishing and applying enrollment policies, issuance policies, and application policies
* **Cross-Certification Authority Certificate** A version 2 certificate template that is issued from one CA to another CA establishing qualified subordination between the two certificates. The certificate used to sign the cross-CA certificate enforces the constraints defined in the certificate.
* **Qualified Subordination Signing Certificate** A version 2 certificate template that must be manually created; it contains the Qualified Subordination application policy OID. This certificate template validates that the holder is approved to sign cross-certification authority certificates.
* **Certutil.exe** A command-line program that is installed as part of Certificate Services. It is used to dump and display CA configuration information, configure Certificate Services, back up and restore CA components, and verify certificates, key pairs, and certificate chains.
* **Certreq.exe** A command-line program that is installed as part of Certificate Services. It is used to request certificates from a CA. In qualified subordination, certreq.exe is used to request the cross-certification authority certificate at the issuing CA.

# Understanding Constraints

The key to qualified subordination is restricting what certificates issued by the qualified subordinate CA, or by CAs that chain through the qualified subordinate CA, are trusted by your organization. This is accomplished by defining basic constraints, name constraints, issuance policy constraints, or application policy constraints in one of two ways:

* Constraints can be defined during the installation of the CA. By including the sections defined below in the CAPolicy.inf file, the basic constraints, name constraints, issuance policy constraints, and application policy constraints are defined at the CA during the installation of the CA or during the certificate renewal process at a CA.
* Constraints can also be defined during the cross-certificate request process by defining the constraints in the Policy.inf file. The Policy.inf file defines the constraints implemented in the cross-certification certificate used to define the qualified subordination.

Appendix A contains an example of a Policy.inf file. Appendix B contains a sample of the CAPolicy.inf file.

**Note** Constraints are defined in section 4.2 "Standard Certificate Extensions" in RFC 2459.

**Note** For more information on how the certificate chaining engine uses constraint information when building certificate chains, see the Microsoft white paper, Troubleshooting Certificate Status and Revocation**,** at http://www.microsoft.com/technet/security/prodtech/tshtcrl.asp

## Basic Constraints

Basic constraints allow a CA administrator to limit path length for a certificate chain. You can specify a basic constraint that defines the maximum number of CAs that can exist below the CA where the basic constraint is assigned. For example, if you define a path length of zero, the CA can only issue end-entity certificates and is not permitted to issue SubCA certificates. Figure 1 shows a CA where a basic constraint is applied that limits the path length to two.

Issuer: **RegionCA**Subject: **AsiaCA**

Issuer: **AsiaCA**Subject: **JapanCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **CorpCA**Subject: **RegionCA**

Issuer: **AsiaCA**Subject: **user1**

****

****

[BasicConstraintsExtension]

PathLength = 2

Figure 1: Applying Basic Constraints

The basic constraint ensures that the maximum path length for certificates that chain to the CorpCA that are trusted is two deep. This means that only two more layers of CAs can exist below the CA where the basic constraint is defined. In this example, the AsiaCA is restricted so that it can only issue trusted end-entity certificates. If a certificate issued to the JapanCA or any certificate issued by the JapanCA is presented, the chain would fail chain verification, because the certificate would result in a chain with a path length greater than two. Whenever a CA issues a certificate to another CA, it can reduce the path length, but never increase the permitted length.

**Note** Basic constraints are fully defined in section 4.2.1.10 of RFC 2459.

**Best Practice** A best practice that can be applied in regards to basic constraints is to only restrict the path length in a subordinate CA certificate, not a root CA certificate. The reason this is more commonly employed is that a change in the root requires a complete redeployment of the hierarchy and the root certificate should a change in the path length be desired. Therefore, in the previous example, the RegionCA certificate would be issued with a PathLength = 1 and have no length defined in the root CA certificate.

## Name Constraints

Name constraints allow you to designate which namespaces are either permitted or excluded for certificates issued by CA where qualified subordination is defined. When a request is received at the qualified CA, the names present in the subject and the subject alternate name fields are compared to the configured name constraints to determine whether the namespace is permitted or excluded. In other words, the CA enforces all constraints defined in the CA certificate.

**Note** If the name presented in the request is not present in the list of constraints, the request will be rejected by the qualified CA. All names in a certificate request must be within permitted namespaces and none may be within an excluded namespace for the certificate to be issued.

Name constraints allow you to control what namespaces are managed by each CA in your organization and those that you trust from other organizations. When you deploy a qualified subordinate CA, you must carefully consider what namespaces you wish to allow the CA to issue, and more importantly in some cases, what namespaces you wish to prevent the qualified subordinate CA from supporting.

For example, when you configure qualified subordination between your organization and a partner organization, you would typically not want your partner’s CA infrastructure to issue certificates containing names in your organization’s namespace. Remember that a valid certificate that contains a UPN for your organization is mapped to the user account with the UPN as an attribute. It does not matter which CA issues the certificate, only that the certificate chains to a CA certificate in the NTAuth store. The use of name constraints can ensure that your namespace, and all recognized formats of your namespace, are excluded for certificates issued by your partner’s CA hierarchy.

In the case where both permitted and excluded name constraints exist, the excluded name constraints will always take precedence. For example, if you create a permitted DNS name constraint for the namespace .microsoft.com and an excluded DNS name constraint for .subdomain.microsoft.com, all certificates for subdomain.microsoft.com will be rejected, even though the microsoft.com namespace is permitted.

**Important** The default policy of a Microsoft client validating a certificate with name constraints asserted is that all names have to be explicitly permitted. For example, if a name constraint does not specify the e-mail name as a permitted type, and a certificate request contains an e-mail name, the request will be rejected. It is possible to relax this policy and implicitly allow names not defined in the name constraint extension by configuring the CA policy in the registry.

The certificate request is only accepted if each name in the certificate request matches at least one of the permitted name constraints configured at the qualified subordinate CA. In other words, if the certificate request contains the requestor’s name in both an LDAP distinguished name format and in a UPN format, both names must match permitted name constraints. If one of the presented names does not match, the certificate request fails.

### Name Constraint Processing

When a certificate request is presented to a subordinate CA under qualified subordination rules, all name forms in the certificate request must be within the permitted namespaces. In addition, the subject names must not be within an excluded namespace. The permitted and excluded name constraints are processed separately.

The following rules are followed by the qualified subordinate CA:

* A certificate request is successful if all names in the certificate request succeed in matching the corresponding permitted name constraints.
* A certificate request is unsuccessful if any names in the certificate request succeed in matching the corresponding excluded name constraint.
* A certificate request containing names that cannot be matched with either the permitted or excluded names is considered excluded and fails.
* Comparison of the name identifying the subject entity with the qualified subordinate CA's name constraints is performed using the following rules:
* The comparison is not case-sensitive if the name or part of the name is encoded with an IA5 string. If it is encoded as a Unicode or UTF8 string, the comparison will be a binary match of the content.
* Name constraints are applied to the Subject name field and any existing Subject Alternate Name extensions.
* A qualified subordinate CA's name constraints cannot permit names excluded by its parent CA. For example, if the parent CA excludes the DNS name .microsoft.com, then its qualified subordinate CA cannot permit the DNS sub-domain .example.microsoft.com. The name constraints are checked by the CA to ensure that its chain is valid by calling CryptoAPI.
* Name constraints apply to all names contained in subject and subject alternate name extensions in an end certificate. Each name in the subject or subject alternate name extensions must match at least one of the name constraints listed for that name type. A subject name or subject alternate name that does not match a listed name type is rejected.
* Constraints apply only when the namespace types specified as name constraints are present in the certificate request. If no namespace of the specified types is in the certificate request, the certificate is acceptable. For example, a CA is configured with the following name constraints:

[NameConstraintsExtension]

Include = NameConstraintsPermitted

Exclude = NameConstraintsExcluded

Critical = TrUe

[NameConstraintsPermitted]

DirectoryName = "DC=Microsoft, DC=Com"

email = @microsoft.com

UPN = .microsoft.com

UPN = @microsoft.com

[NameConstraintsExcluded]

In this configuration, if the CA receives a request with the e-mail name of user1@northwindtraders.com, the request would be rejected. Likewise, the request would be rejected if the subject alternate name contained user1@northwindtraders.com while the subject name contained CN=user1,OU=Contractors,DC=Microsoft,DC=Com. Remember that all subject names must match the name constraints for a certificate to be issued. The default policy as mentioned previously requires all names to be defined unless otherwise configured on the certificate authority.

**Note** If you do not specify permitted or excluded names, you are creating a wildcard (\*) name constraint for those name constraint formats.

When the name constraint processing is completed, the validation process will result in one of the following outcomes:

* **Permitted** The end certificate contains a name that is listed as permitted in an issuer’s name constraints extension.
* **Not permitted** The end certificate contains a name that is not listed as permitted in an issuer’s name constraints extension.
* **Excluded** The end certificate contains a name that is listed as excluded in an issuer’s name constraints extension.
* **Not Defined** The issuer certificate does not list a constraint for a specific name type (such as Directory Name or IP Address).

**Note** Windows Server 2003 CAs will have constraints set to permit all namespaces for a particular name type, unless specified otherwise in the name constraints extension in the Policy.inf file.

### Applying Name Constraints

The following naming and addressing format are natively supported for use in either CAPolicy.inf or Policy.inf when defining name constraints:

* Relative distinguished name
* DNS domain name
* Universal Resource Identifiers (URI)
* E-mail name and User principal name (UPN)
* IP address

Additional name constraint forms may be enforced using the “Other Name” forms, which are identified by name and OID that are UTF8 or ASN.1 encoded. Other Names can be used to represent names that are outside of the standard name types such as Rfc822Name, DNS, or X.500 Directory Name. It consists of an OID and a binary blob. The most common Other Name is the Universal Principal Name (UPN) Name, which has the constant OID\_NT\_PRINCIPAL\_NAME as the OID. The version 2 "Domain Email Replication" template certificates and the Version 1 "Domain controller" certificates have a subjectaltname extension that contains the constant OID\_NTDS\_REPLICATION as other name type and the GUID representation of the DC as the blob. In Windows 2000, only the UPN was present in the other name constraint. However, now in Windows Server 2003, the other name constraint has been expanded to support even the domain controller GUIDs. The following is the syntax where the first part is the OID, followed by a qualifier identifying the encoding of the following data as UTF8 or octet or ASN.1:

OtherName=1.2.3.4.99.100,{utf8}ssss

OtherName=1.2.3.4.99.101,{octet}ABCD

OtherName=1.2.3.4.99.102,"{asn}BAgAAQIDBAUGBw=="

OtherName=1.3.6.1.4.1.311.25.1

One scenario where this may be useful is when you have cross forest smartcard logon, wherein the other domain has Windows 2000 domain controllers. In this case, you can provide an additional name constraint as "OtherName=1.3.6.1.4.1.311.25.1" to allow all Domain e-mail replication certificates and restrict the namespace using the DNS name constraint. It is technically possible to restrict the GUID of a domain controller as another name but that would not be a generic constraint but a specific constraint to only one domain controller.

#### Configuration

The name constraints are configured in one of two locations. When creating a new CA, you can define name constraints for that CA by configuring CAPolicy.inf to impose name constraints. Likewise, if you are creating a qualified subordinate CA certificate, you would define name constraints in the Policy.inf file. In both cases, the following syntax is used:

[NameConstraintsExtension]

Include = NameConstraintsPermitted

Exclude = NameConstraintsExcluded

Critical = TrUe

[NameConstraintsPermitted]

DNS = ""

email=""

UPN=""

[NameConstraintsExcluded]

DNS = .nwtraders.com

email = @nwtraders.com

UPN = .nwtraders.com

UPN = @nwtraders.com

URI = ftp://.nwtraders.com

DIRECTORYNAME = "DC=NWtraders, DC=com"

**Note** The critical = True in the [NameConstraintsExtension] indicates that this extension is marked as critical. If the validating computer cannot parse the extension, it must reject the certificate chain.

#### Relative Distinguished Name Constraints

Relative distinguished names are used to identify the names of objects stored in directories such as Active Directory or X.500 directories. Using Relative distinguished name constraints allows you to restrict a qualified subordinate CA to issue certificates only to specific users or computers in Active Directory by using the object’s Relative distinguished name. The distinguished name can be very specific, where the permitted or excluded Relative distinguished name references a specific object. Or, the distinguished name can be a wildcard value that references all the objects within a specific container, domain, or organizational unit.

The name constraint must be included in either the Policy.inf file used to create a qualified subordinate CA or in the CAPolicy.inf file used when installing a new subordinate CA. Table 1 shows some examples of LDAP name constraints and defines what objects the constraints refer to.

Table 1: Name Constraint Examples

|  |  |
| --- | --- |
| **Relative Distinguished Name Constraint** | **Includes** |
| DC=example,DC=microsoft,DC=com | All objects in the example.microsoft.com domain |
| CN=Users,DC=example,DC=Microsoft,DC=com | All objects in the Users container in the example.microsoft.com domain |
| CN=Brian Komar,CN=Users,DC=microsoft,DC=com | The user object Brian Komar in the Users container in the microsoft.com domain |
| OU=Marketing,DC=Microsoft,DC=Com | All objects in the Marketing OU of the Microsoft.com domain |
| CN=Servers,CN=Sites,CN=Configuration,DC=Microsoft,DC=Com | All domain controllers in the microsoft.com forest |
| OU=Domain Controllers, DC=Redmond,DC=Microsoft,DC=Com | All domain controllers in the Redmond.microsoft.Com domain. |

The range of characters in the allowed name constraint is defined by the RDN type. Most RDNs (such as O=, OU=,CN=, and so on) support international (UTF8) characters, where RDNs that include (DC=,C=,email, and so on) support on IA5 character types. A Relative name constraint will appear in either the permitted or excluded name constraints using the following format:

DIRECTORYNAME = "DC=NWtraders, DC=com"

#### DNS Name Constraints

DNS name constraints compare any DNS names contained in certificate requests to the list of permitted and excluded DNS name constraints. As with LDAP distinguished name constraints, the name in the certificate request must match one of the permitted names for the certificate to be issued.

The DNS domain names accepted by the subordinate CA must follow the standard DNS naming conventions as specified in RFCs 1034 and 1035, and are listed in either the Policy.inf or CAPolicy.inf file used. Therefore, the DNS name constraint cannot contain international character sets.

When configuring a DNS name constraint, you can either designate a specific DNS host name, for example, host.example.microsoft.com, or you can designate a DNS namespace. For example, the DNS name constraint .example.microsoft.com indicates all hosts that have a DNS name that ends with example.microsoft.com. The preceding period (.) indicates that any DNS name that ends with example.microsoft.com matches the name constraint.

**Note** This does not include a host that does not have a period preceding example.microsoft.com. For example, the host named ourexample.microsoft.com would not result in a match when compared to the name constraint .example.microsoft.com.

Table 2 demonstrates the results for submitted DNS names against configured DNS name constraints.

Table 2: Evaluating DNS Name Constraints

|  |  |  |
| --- | --- | --- |
| **Submitted DNS Name** | **DNS Name Constraint** | **Evaluation** |
| www.example.microsoft.com | .example.microsoft.com | Match |
| www.south.example.microsoft.com | .example.microsoft.com | Match |
| example.microsoft.com | .example.microsoft.com | No Match |
| Ourexample.microsoft.com | example.microsoft.com | No Match |
| www.example.microsoft.com | example.microsoft.com | Match |

**Important** Evaluate your DNS constraints carefully. Remember that all excluded name constraints take precedence over a permitted name constraint. For example, if you permitted .example.microsoft.com and excluded .microsoft.com, a request from www.example.microsoft.com will be rejected. This DNS name matches both the excluded and permitted name constraints. Because there is a match on the excluded name constraint, the permitted name constraint is not observed.

A DNS name constraint will appear in either the permitted or excluded name constraints using the following format:

DNS = .nwtraders.com

#### Uniform Resource Identifier (URI) Constraints

Uniform Resource Identifiers (URIs) identify resources on the Internet that use identifiers such as URL, FTP, HTTP, telnet, mailto, news, and gopher. The URI naming conventions supported for name constraints must follow the syntax specified in RFC 2396.

When a qualified subordinate CA validates a URI name, the protocol element in the URI is ignored. For example, if the submitted URI is http://www.example.microsoft.com and the URI name constraint is URL=ftp://.example.microsoft.com, the actual comparison performed is between the host name www.example.microsoft.com and the name constraint .example.microsoft.com. In this case, the result would be a match, even though the protocols differ.

As with DNS constraints, a URI constraint can reference either a host or a domain. If a preceding period is included in the URI constraint, the domain name must include the entire suffix defined in the name constraint. For example, if the URI constraint is .example.microsoft.com, then matches are achieved for west.example.microsoft.com and for east.example.microsoft.com.

The most frequent application of URI name constraints is the validation of certificate requests for Web Server certificates, where the Web server’s URL is submitted in the certificate request. For example, if the Web Server certificate request has a name of http://www.microsoft.com and the qualified subordinate CA's permitted URI name constraint is URL=http://www.microsoft.com, then this URI constraint would permit the certificate request for the Web Server certificate.

A URI name constraint will appear in either the permitted or excluded name constraints using the following format:

URL = "http://.microsoft.com"

#### RFC 822, E-mail, and UPN Constraints

RFC 822 and email constraints must follow RFC 822 naming conventions and use IA5 encoding of character strings. User Principal Names (UPNs) use the same syntax but are encoded using UTF8. Although RFC 822 names, UPNs, and E-mail addresses share the same syntax, you must provide separate examples in the name constraints listing of Policy.inf to differentiate between requests using e-mail addresses and requests using UPN names.

When defining RFC 822, E-mail, or UPN constraints, you can specify constraints for individual addresses or for addresses that end with a specific domain.

For example, to create a name constraint for the e-mail address for a Joe Smith at Northwind Traders, you could create a E-mail address constraint for jsmith@nwtraders.com. To create a name constraint for all e-mail users at Northwind Traders, you would set the e-mail constraint to be @nwtraders.com. By not placing any text to the left of the ampersand, the e-mail constraint becomes a wildcard representing all e-mail addresses in the nwtraders.com domain.

Table 3 shows some examples of RFC822, E-mail, and UPN constraints and how they evaluate.

Table 3: RFC 822, E-mail, and UPN Constraint Evaluation

|  |  |  |
| --- | --- | --- |
| **Submitted Address** | **UPN/E-Mail Constraint** | **Evaluation** |
| jsmith@nwtraders.com | jsmith@nwtraders.com | Match |
| bjsmith@nwtraders.com | jsmith@nwtraders.com | Match |
| jsmith@nwtraders.com | @nwtraders.com | Match |
| jbsmith@nwtraders.com | bjsmith@nwtraders.com | No Match |

**Note** UPN name constraints should always be entered as two separate entries in the list of name constraints. One entry should include the ampersand character (such as @nwtraders.com); the second entry should replace the ampersand with a period (.), so this entry would be .nwtraders.com. This format allows for the possibility of having a UPN of subdomain.nwtraders.msft.

E-mail and UPN name constraints will appear in either the permitted or excluded name constraints using the following format:

email = @nwtraders.com

UPN = .nwtraders.com

UPN = @nwtraders.com

#### IP Address Constraints

IP address constraints allow you to specify either specific IP addresses or ranges of IP addresses that a CA can successfully receive certificates from a qualified subordinate CA. The format of the IP addresses in the constraints must follow either RFC 791 (for IPv4 addresses) or RFC 2460.

When a certificate request is received by a qualified subordinate CA, the source IP address in the certificate request is compared to the IP address constraints to determine if the IP address is permitted or excluded by the name constraints.

To specify a range of IP addresses, you must specify both the IP address and associated subnet mask in each constraint. For example, if you wanted to create an IP address constraint for the 192.168.3.0 network, the constraint would be 192.168.3.0/255.255.255.0.

An IP address constraint will appear in either the permitted or excluded name constraints using the following format:

IPADDRESS = 192.168.3.0/255.255.255.0

IPADDRESS = 192.168.2.254/255.255.255.255

IPADDRESS = 172.16.8.0/255.255.248.0

The first constraint includes all computers in the 192.168.3.0 network. The second constraint is a host-specific constraint that only includes the host with IP address 192.168.2.254. The final constraint is a Classless Internet Domain Routing (CIDR) example that includes all hosts with addresses between 172.16.8.1 and 172.16.15.254.

## Issuance Policy

Issuance policies are used to identify the extent to which your organization trusts the identity presented in a certificate issued by another organization’s CA hierarchy. For example, an issuance policy may describe that you only trust certificates that were issued during a face-to-face meeting with a network administrator, such as the issuance of a smartcard certificate. Each issuance policy is described by an OID. The inclusion of an issuance policy OID in an issued certificate indicates that the certificate was issued meeting the issuance requirements associated with the issuance policy OID.

**Note** Issuance policy is the Microsoft term for Certificate Policies defined in part 4.2.1.5 in RFC 2459 and is contained in the certificatePolicies extension of a certificate.

For a specific certificate template, you can define one or more issuance policy OIDs that are included in issued certificates. There is a special issuance policy, the *All issuance Policies* OID, commonly reserved for CA certificates, that indicates the issuance policy contains all other issuance policies.

Windows Server 2003 includes four predefined issuance policies:

* **All Issuance (2.5.29.32.0)** The all issuance Policy indicates that the issuance policy contains all other issuance policies. Typically, this OID is only assigned to CA certificates.
* **Low Assurance (1.3.6.1.4.1.311.21.8.*x.y.z*.1.400)** The low assurance OID is used to represent certificates that are issued with no additional security requirements. The *x.y.z* portion of the OID is a randomly generated numeric sequence that is unique for each Windows Server 2003 forest.
* **Medium Assurance (1.3.6.1.4.1.311.21.8.*x.y.z*.1.401)** The medium assurance OID is used to represent certificates that may have additional security requirements for issuance. For example, a smartcard certificate that is issued in a face-to-face meeting with a smartcard issuer could be considered a medium assurance certificate and contain the medium assurance OID. The *x.y.z* portion of the OID is a randomly generated numeric sequence that is unique for each Windows Server 2003 forest.
* **High Assurance (1.3.6.1.4.1.311.21.8.*x.y.z*.1.402)** The high assurance OID is used to represent certificates that are issued with the utmost security. For example, a key recovery agent certificate allows the holder to recover private key material from a Windows Server 2003 Enterprise Edition CA. The issuance of a key recovery agent certificate may require additional background checks and a digital signature from a designated approver. The *x.y.z* portion of the OID is a randomly generated numeric sequence that is unique for each Windows Server 2003 forest.

**Best Practice** Technically, Issuance Policies can be added to any version 2 template that is issued from a Windows Server 2003 certificate authority running Enterprise Edition. However, it is a best practice to apply issuance policies in the Policy.inf file for the qualified subordinate CA to ensure consistent policy for the CA. This is to avoid issuing templates with different policies unintentionally.

**Note** A root CA automatically implies that all issuance policies are defined. For understanding how policy is evaluated in certificate chains, see the Certificate Status and Revocation white paper at <http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/prodtech/tshtcrl.asp>

If your organization has existing OIDs issued, they may be used for this purpose. In addition, you can create your own OIDs to represent custom issuance policies. For example, two organizations involved in a purchaser/seller relationship may define custom OIDs to represent digital signature certificates for specific purchase amounts. For example, OIDs may be defined for purchase between $100,000 and $500,000, and another OID for purchases greater than $500,000. These OIDs could then be used by applications to recognize whether a person had the appropriate signing authority for a specific volume purchase.

[PolicyStatementExtension]

Policies = HighAssurancePolicy, MediumAssurancePolicy, LowAssurancePolicy

CRITICAL = FALSE

[HighAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.402

[MediumAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.401

[LowAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.400

**Note** Issuance policy extensions are only recognized by Windows XP and Windows Server 2003 clients. If the extension is marked critical, CAPI passes the extensions to the application. It is up to the calling application to understand the extension when it calls with a specific policy provider. The behavior will vary by application. In the case of Outlook on Windows 2000, a critical issuance policy will cause Outlook to generate an error as it does not understand the extension. But on Windows XP, the issuance policy is recognized and does not cause an error.

### Policy Mapping

When qualified subordination is configured between two CAs that use issuance policy, the OIDs between the two organizations must be mapped. Policy mapping establishes what OIDs in one organization are equivalent to OIDs defined in a different organization's CA hierarchy. The policy mapping allows the OIDs generated in certificates in one organization to be recognized in another organization through the creation of cross-certification certificates.

Policy mapping ensures that only authorized OIDs from a partner organization are allowed in certificates issued by the partner organization. The policy mapping will associate the partner organization's OID with an OID defined in your organization's PKI hierarchy.

Figure 2 shows how policy mapping is defined in the Policy.inf file when cross certificates are issued to allow certificates to be used between the two CA hierarchies.

Figure 2: Policy Mapping

In this figure, the Northwind Traders is configured with an issuance policy named MillionDollar that has been assigned the OID 1.3.6.4.1.311.21.8.1.124. This issuance policy is mapped to the issuance policy named BigOrder configured at the Contoso Consulting CA.

If a certificate is issued by the Northwind Traders CA with an issuance policy other than the MillionDollar policy, the certificate is rejected by the Contoso CA because the policy does not map to an approved issuance policy OID.

When cross certificates are created to link the two organizations' CA hierarchies, the code snippets shown in the figure must be included in the Policy.inf files used to generate the cross-certification certificate at each CA. The Policy.inf files exist at the *issuer* CA*.* The OID that is mapped is the OID from the *subject* CA that is the OID for the matching issuance policy at the partner organization.

Before you perform the policy mapping, you must configure the issuer CA with the issuance policies and associated object IDs. This is done by configuring the CAPolicy.inf file and either installing the CA or renewing the CA certificate using the configured CAPolicy.inf file with the required object identifiers as shown in the following code sample:

[PolicyStatementExtension]

Policies = HighAssurancePolicy, MediumAssurancePolicy, LowAssurancePolicy

CRITICAL = FALSE

[HighAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.402

[MediumAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.401

[LowAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.400

In this example, the High, Medium, and Low Assurance issuance policies are defined at the subject CA. The OIDs can be retrieved from the Certificate Templates console by right-clicking Certificate Templates in the console tree and selecting View Object Identifiers from the pop-up menu.

**Note** Appendix B contains a sample CAPolicy.inf file.

The renewal of the CA certificate using the CAPolicy.inf file ensures that the issuance policies are now defined at the CA. At this point, the cross certificates can be generated that perform the policy mapping between the two organizations. The following process is used to perform the qualified subordination between the two CA hierarchies:

1. Determine the organization with which you want to establish a trust relationship. If more than two organizations exist in the relationship, the trust relationships must be defined in pairs.
2. Through discussions with the partner organization, establish the equivalence between the assurance levels used by the two organizations involved in the trust relationship.
3. Exchange the issuance policy object identifiers defined for each assurance level used in both organizations involved in the trust relationship.
4. Map the issuance policy object identifiers in the separate organizations in the Policy.inf file at each CA and define their policy constraints in the CA certificate request for the qualified subordinate CA you are installing for your organization.
5. Install the qualified subordinate CA with the policies, policy mappings, and policy constraints you wish to deploy in your organization.

When a user or computer in the organization where the *issuing* CA exists receives a file signed by a user in the organization where the *subject* CA exists, the certificate will be chained to the root CA in the *issuing* CA using the qualified subordination path. If the object identifier for the issuance policy in the certificate maps to the required issuance policy object ID in your organization, the certificate chain is considered valid. If the user or computer in the issuing CA’s organization receives a certificate with a non-mapped object identifier, the certificate chain is assigned a lesser value by the validation process and is typically rejected by the application.

### Constraining Policy Mapping to Prevent Unexpected Trust

You can further constrain issuance policy by defining parameters that define how the issuance policy defined in the qualified subordination affects other CAs below the qualified subordinate CA. There are two settings that can define the relationship:

* Require explicit policy Specifies the number of certificates that can exist in the hierarchy below the certificate with the policy constraint extension. For example, if the explicit policy is configured with a value of three, this means that the defined issuance policy must exist for three layers of the hierarchy with the CA, where the qualified subordination is defined being the first level.
* Inhibit policy mapping Specifies the number of additional certificates that can appear in the path before policy mapping is no longer permitted. An inhibit policy mapping value of three only restricts the policy mapping to three levels of CAs below the qualified subordinate CA.

These settings exist to prevent unexpected trust relationships. For example, Figure 3 shows the same qualified subordination configured previously between Northwind Traders and Contoso Consulting with the addition of a new CA at Fabrikam Corporation.

Figure 3: Preventing Unexpected Trust

If qualified subordination is configured between the Contoso Consulting CA and the Fabrikam Corp CA, mapping the BigOrder issuance policy to the SpecialSig issuance policy, it is possible that a certificate issued by the Fabrikam Corp CA could be accepted by the Northwind Traders CA. This is because the OID for SpecialSig would be mapped first to the BigOrder OID. Then, the BigOrder OID would be mapped to the MillionDollar OID.

This behavior is prevented by configuring the following Policy Constraint Extensions at the Northwind Traders CA.

[PolicyConstraintsExtension]

RequireExplicitPolicy = 1

InhibitPolicyMapping = 1

By assigning InhibitPolicyMapping a value of one, only certificates issued by the ContosoCA are recognized through the MillionDollar OID mapping defined at the Northwind Traders CA. A certificate issued by the Fabrikam Corp CA with the SpecialSig OID will fail the InhibitPolicyMapping =1 setting defined at the Northwind Traders CA. This prevents the certificate issued by the Fabrikam Corp CA from being accepted by the Northwind Traders CA.

### Policy Qualifiers

You can provide additional information about the Issuance policies implemented at a CA by configuring policy qualifiers. Policy qualifiers provide information directly, or provide links to information, that describe the purpose of the issuance policy. The following code shows both formats of the policy qualifier:

[LegalPolicy]

OID = 1.3.6.1.4.1.311.21.43

Notice = "Legal policy statement text"

URL = "http://www.example.microsoft.com/policy/isspolicy.asp"

When viewing the certificate in an application, a person can follow the URL and read the description of the issuance policy in a Web browser. If a text message is used as a policy qualifier, the application will display the message.

**Best Practice:** It is recommended that URLs be implemented for certificate practice statements, rather than to use notice text. This provides several key benefits.

* It allows modification of the practice statement without having to re-issue the CA certificate.
* URLs do not dramatically increase the size of the certificate, unlike notice text.

## Application Policy

Application policies included in certificates allow applications to determine if a certificate can be used when authenticating a user, encrypting data, or signing a device driver. An application can be coded to only accept certificates that contain specific application policies. When the application receives signed information from a user, the application will review the certificate associated with the private key used to sign the information and verify that the certificate chain has the required OID as a valid application policy.

Application policies are similar to the Extended Key Usage (EKU) extension in a certificate as both use one or more OIDs to prescribe how the public key in a certificate must be used. EKU is still supported by Windows XP and the Windows Server 2003 family to provide for PKIs that use this extension, but application policies are a replacement. The following rules are used when combining EKU and Application Policy extensions:

* If a certificate is presented that contains an EKU extension, but does not contain a separate Application Policy extension, the EKU extension will be treated as an Application Policy extension by Windows XP and Windows Server 2003 clients.
* If a certificate has an extension containing an application policy and also has an EKU extension, the EKU extension is ignored.
* If a certificate has an Application Policy extension and an EKU property, the effective policy for the certificate is the common policy between the EKU property object identifiers and the application policy object identifiers.

**Note** If issuing certificates that include both Application Policy and EKU extensions, ensure that the two extensions are identical in their assignment of OIDs so that the two extensions do not conflict with each other. This ensures consistent application of policy, when either Application Policy or EKU extensions are used. This is especially important in issuing cross certificates or in using templates that do not have predefined application policies.

Application policies may be defined in qualified subordination certificates to limit what purposes certificates from a trusted organization can be used for. For example, you can create an application policy that only allows certificates with the Secure Mail OID to be recognized if they chain through the qualified subordinate CA.

When application policy is defined for a CA, the OIDs associated with the application policy are included in all issued certificates.

**Note** The All Applications OID indicates that the application policy includes all application policies. This application policy is normally reserved for certificates issued to CAs. For understanding how policy is evaluated in certificate chains, see the Certificate Status and Revocation white paper at <http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/prodtech/tshtcrl.asp>

The following Application Policy OIDs are recognized by Windows Server 2003.

|  |  |
| --- | --- |
| * Server Authentication (1.3.6.1.5.5.7.3.1) | * Client Authentication (1.3.6.1.5.5.7.3.2) |
| * Code Signing (1.3.6.1.5.5.7.3.3) | * Secure Email (1.3.6.1.5.5.7.3.4) |
| * IP Security End System (1.3.6.1.5.5.7.3.5) | * IP Security Tunnel Endpoint (1.3.6.1.5.5.7.3.6) |
| * IP Security User (1.3.6.1.5.5.7.3.7) | * Time Stamping (1.3.6.1.5.5.7.3.8) |
| * IP Security IKE Intermediate (1.3.6.1.5.5.8.2.2) | * All Application Policies (1.3.6.1.4.1.311.10.12.1) |
| * Microsoft Trust List Signing (1.3.6.1.4.1.311.10.3.1) | * Qualified Subordination (1.3.6.1.4.1.311.10.3.10) |
| * Key Recovery (1.3.6.1.4.1.311.10.3.11) | * Document Signing (1.3.6.1.4.1.311.10.3.12) |

**Note** A list of available object identifiers can be viewed in the Certificate Templates console by right-clicking Certificate Templates in the console tree and selecting View Object Identifiers from the pop-up menu. Alternatively, custom application policy OIDS can be defined and used in custom-developed applications.

To define application policies in a Policy.inf file, the following sections must be created:

[ApplicationPolicyStatementExtension]

Policies = AppEmailPolicy, AppCodeSignPolicy, AppAuthPolicy

CRITICAL = FALSE

[AppEmailPolicy]

OID = 1.3.6.1.5.5.7.3.4 ; Secure Email

[AppCodeSignPolicy]

OID = 11.3.6.1.5.5.7.3.3 ; Code Signing

[AppClAuthPolicy]

OID = 1.3.6.1.5.5.7.3.2 ; Client Authentication

The [ApplicationPolicyStatementExtension] section defines all application policy setting sections that exist in the Policy.inf file. In this case, three application policy sections are defined. For each [*AppPolicy*] section, the OID associated with the application policy is defined.

If a custom application policy OID is defined, an [ApplicationPolicyMappingsExtension] section must be defined. This section uses the same format, where the local OID is mapped to the OID used by the other organization participating in the qualified subordination as shown in the following code sample:

[ApplicationPolicyMappingsExtension]

1.3.6.1.4.1.311.21.64 = 1.2.3.4.98

1.3.6.1.4.1.311.21.65 = 1.2.3.4.100

critical = true

As with issuance policies, you can also configure limitations on how deep in the hierarchy explicit policy must exist and whether the defined application policies can be mapped to other application policies. These settings are defined in the [ApplicationPolicyConstraintsExtension] section as follows:

[ApplicationPolicyConstraintsExtension]

; consists of two optional DWORDs

; They refer to the depth of the CA hierarchy that requires explicit policy

; and inhibits Policy Mapping

RequireExplicitPolicy = 6

InhibitPolicyMapping = 10

# Qualified Subordination Deployment Scenarios

This section looks at some of the more common certificate services deployments that use qualified subordination, including the following scenarios:

* Using qualified subordination to restrict certificate issuance to specific namespaces
* Configuring qualified subordination between two organizations; specifically two configurations:
* Allowing trust between all CAs in the two organizations
* Limiting trust to specific CAs in the CA hierarchy
* Configuring qualified subordination using a bridge CA

This section covers both the design of the scenarios and how to deploy the scenarios.

**Note** This section looks at the design process of qualified subordination. The Walkthrough section provides the actual deployment steps involved when defining qualified subordination between two CA hierarchies.

## Restricting Namespaces

The first example is a common scenario in large organizations where certificate services administration is decentralized. In a decentralized environment, it may be desired to restrict certificate issuance to only specific objects in Active Directory. Use of name constraints is an effective way to ensure specific CAs can only issue to a well-defined subset of users framed by a namespace.

Assume that Contoso uses the forest structure in Figure 4 to provide decentralization of administration to regional subsidiaries.

Figure 4: Contoso.com’s Forest and Domain Structure

Within Contoso.com, several CA’s have been deployed for certificate services deployments. Figure 5 shows the current CA hierarchy used by Contoso.com.

Figure 5: A Proposed CA Hierarchy for Contoso Corporation

In this example, the CA named RegionCA exists to allow decentralized management of certificate deployment for Europe and Asia. The EuropeCA and AsiaCA are installed on member servers in the europe.contoso.com and asia.contoso.com domains. Qualified subordination can be used to restrict the issuance of certificates by the AsiaCA to only security principals (users, computers, services) located in the asia.contoso.com domain.

To do this, the AsiaCA must be installed with a CAPolicy.inf file in the %SystemRoot% folder with the following settings:

[NameConstraintsExtension]

Include = NameConstraintsPermitted

Exclude = NameConstraintsExcluded

Critical = True

[NameConstraintsPermitted]

DNS = .asia.contoso.com

email = @asia. contoso.com

UPN = .asia.contoso.com

UPN = @asia.contoso.com

DIRECTORYNAME = "DC=asia,DC=contoso,DC=com“

[NameConstraintsExcluded]

These name constraints will only allow security principals from the asia.contoso.com domain to receive certificates issued by the AsiaCA certification authority. If a request is received from a security principal from a different namespace, the certificate request will be denied due to the configured name constraints.

**Note** Alternatively, as mentioned earlier when defining the types of constraints, the AsiaCA can obtain a subordinate CA certificate from the RegionCA that includes the defined name constraints. Either method would provide the necessary name constraints to issued certificates.

For example, Figure 6 shows the name constraints in place for AsiaCA restricting certificate issuance to the asia.contoso.com namespace.

Issuer: **RegionCA**Subject: **AsiaCA**

Issuer: **RegionCA**Subject: **EuropeCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **ProjectCA**Subject: **EdiCA**

Issuer: **CorpCA**Subject: **RegionCA**

Issuer: **AsiaCA**Subject: **user3@asia.contoso.com**

Issuer: **AsiaCA**Subject: **user5@europe.contoso.com**

****

****

[NameConstraintsExtension]

Include = NameConstraintsPermitted

Exclude = NameConstraintsExcluded

Critical = True

[NameConstraintsPermitted]

DNS = .asia.contoso.com

email = @asia.contoso.com

UPN = .asia.contoso.com

UPN = @asia.contoso.com

DIRECTORYNAME = "DC=asia,DC=contoso,DC=com“

[NameConstraintsExcluded]

Figure 6: Applying Name Constraints at AsiaCA

In this example, the certificate request by user3@asia.contoso.com is approved because Noel’s UPN is included in the permitted name constraints. In comparison, the request by user5@europe.contoso.com is denied because the UPN Europe.contoso.com is not listed in the permitted name constraints and is considered to be an excluded name constraint.

Enterprise CAs enforce name constraints during certificate requests based on the authenticated security principal performing the certificate request. The credentials of the security principal are used to extract the names of the user from Active Directory and these names are compared with the defined name constraints. In this case, user3 would be logged on to the network and performing the successful certificate request.

## Creating Trust Between Two Organizations

The second scenario where cross-certification may be used is to allow certificates to be used and trusted between two organizations. Before the cross-certification is performed between the two organizations, the first thing that must be decided is what certificate usage will be trusted between the organizations. For example, you could require that certificates be trusted from the other organization(s) only for secure e-mail. Or, you could allow secure e-mail, client authentication, and server authentication.

This must be decided and included in the Policy.inf file when the qualified subordination is configured between CAs in the two organizations.

**Note** For more information on how to perform the qualified subordination, see the Walkthrough section.

When configuring the cross-certification between the two organizations, you also must decide whether all CAs in the other organization’s CA hierarchy are to be trusted or only a specific grouping of CAs in the hierarchy.

**Important** Cross-certification of a CA hierarchy may create very long certificate chains that can increase the size of network traffic for IPSEC, Secure Mail (s/mime), SSL, and so on. Use caution so as to not increase the chain length and overall certificate chain byte size to exceed an application limit.

### Implementing Qualified Subordination to Trust All CAs

In the first example, qualified subordination will be configured so that all CAs in the two organizations are trusted by the other organization. Figure 7 shows the two CA hierarchies.

Contoso  
Consulting

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**

Subject: **IssuingCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Northwind  
Traders

Figure 7: Two Non-Connected CA Hierarchies

In this scenario, there is a business need for user 1 and user 2 to exchange secure e-mail using Outlook and S/MIME. Due to increased cooperation between Northwind Traders and Contoso Consulting, it is decided that qualified subordination be used to permit secure e-mail certificates issued by either organization to be trusted by both Northwind Traders and Contoso Consulting. In addition, the secure e-mail certificates can be issued by any CA in the individual CA hierarchies.

To accomplish this, two different methods can be used:

* Cross-certification between root CAs  
  In this design, the root CAs issue cross-certification certificates to the other hierarchy's root CA. This is the easiest method to cross-certify organizations and is the easiest to understand when troubleshooting qualified subordination problems.
* Cross-certification between subordinate and Root CAs  
  Many organizations are unwilling to issue certificates from their root CA to other organization's root CA. In this case, the cross-certification certificates can be issued by a subordinate CA to the root CA in the partner organization's CA hierarchy.

Both methods allow for complete trust between the CA hierarchies, subject to the constraints defined in the Policy.inf files. The decision factors will be based on whether your organization is willing to issue a cross-certification certificate to a partner organization's root CA from your root CA. Another consideration is that Root CAs generally publish their CRL less frequently than subordinate CAs. If there is a possibility that the cross-certification authority certificate may be revoked, it is preferable to issue the certificate from a subordinate CA than from a root CA. A certificate issued from a subordinate CA will have less lag time between the time the certificate is revoked and the revocation being recognized by all clients.

#### Cross-Certification Between Root CAs

If your organization is willing to perform cross-certification between the root CAs, the configuration in Figure 8 can be used.

Contoso  
Consulting

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **CorpCA**Subject: **PartnerCA**

Issuer: **PartnerCA**Subject: **CorpCA**

Issuer: **IssuingCA**Subject: **user1**

Northwind  
Traders

Figure 8: Configuring Complete Trust Between Root CAs

In this example, the root CA for each organization is cross-certified with the root CA in the other organization. To create this configuration, the qualified subordination must be performed twice:

1. The PartnerCA at Contoso Consulting must be cross-certified using qualified subordination with the CorpCA at Northwind Traders.
2. The CorpCA at Northwind Traders must be cross-certified using qualified subordination with the PartnerCA at Contoso Consulting.

The qualified subordination will require that the following be in place:

* A Policy.inf is configured at each root CA. The Policy.inf must define any name constraints, issuance constraints, and application constraints defined between the two organizations and is used during the cross-CA certificate request as described in the Walkthrough section.
* The user account that is performing the qualified subordination must obtain a certificate with the Qualified Subordination application policy OID to sign the qualified subordination request between the two organizations. The signing certificate must be obtained from the CorpCA or any CA that is trusted on the CorpCA and PartnerCA computers for the first request and from the PartnerCA or any CA that is trusted on the CorpCA and PartnerCA computers for the second request.

The configuration shown in Figure 8 allows certificates issued by either CA to chain to a root that is trusted by each respective organization. For example, the certificate issued to User1 will validate up to the CorpCA if verified by a security principal at Northwind Traders as follows:

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

If the same certificate is validated by a computer in the Contoso Consulting organization, the certificate will validate up to PartnerCA.

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **PartnerCA**Subject: **CorpCA**

Issuer: **IssuingCA**Subject: **user1**

Similarly, the certificate issued to User2 will chain to a trusted root in each organization. If the certificate is validated by a computer at Northwind Traders, the certificate chain will be as follows:

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **PolicyCA**

Subject: **user2**

Issuer: **PartnerCA**

Subject: **PolicyCA**

Issuer: **CorpCA**Subject: **PartnerCA**

However, the same certificate would validate differently and use the PartnerCA root CA as the trust anchor when validated by a computer in the Contoso Consulting organization:

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **PolicyCA**Subject: user2

Issuer: **PartnerCA**Subject: **PolicyCA**

**Note** In all cross-certification designs, a single certificate will chain to different roots depending on which organization's computers are validating the certificate. In both examples, a certificate validated by computers at Contoso Consulting will chain to the PartnerCA root, while certificates validated by computers at Northwind Traders will chain to the CorpCA root.

The advantage of this design is that it also allows trust of new CAs installed into the CA hierarchies at either organization. For example, Figure 9 shows the addition of a new CA at Northwind Traders.

Contoso  
Consulting

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **IssuingCA**

Northwind  
Traders

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **CorpCA**Subject: **PartnerCA**

Issuer: **PartnerCA**Subject: **CorpCA**

Figure 9: Adding an Additional CA to the Hierarchy

In this case, the certificate issued to User4 is trusted in both organizations. In Northwind Traders, the certificate chains to the CorpCA root CA.

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

While at Contoso Consulting, the certificate chains to the trusted PartnerCA.

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **PartnerCA**Subject: **CorpCA**

#### Cross-Certification Between Subordinate and Root CAs

The same level of trust can be established between two organizations by issuing the cross-certification authority certificate from a subordinate CA in your CA hierarchy. This strategy is recommended if your organization's security policy prohibits the issuance of crossCA certificates by the root CA, your organization is simply unwilling to issue cross-certification certificates at the root CA, or there is a possibility that the cross-certification authority certificate may be revoked, and the CRL publication period for the root CA is a long interval. In these cases, the configuration shown in Figure 10 can be used to create complete trust between the two CA hierarchies, subject to defined qualified subordination constraints.

Contoso  
Consulting

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **PolicyCA**Subject: **CorpCA**

Issuer: **IssuingCA**Subject: **PartnerCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Northwind  
Traders

Figure 10: Configuring Complete Trust Between Subordinate and Root CAs

In this example, the Root CA for each organization is cross-certified with the subordinate enterprise CA in the other organization. To create this configuration, the qualified subordination must be performed twice:

1. The PartnerCA at Contoso Consulting must be cross-certified using qualified subordination with the IssuingCA at Northwind Traders.
2. The CorpCA at Northwind Traders must be cross-certified using qualified subordination with the PolicyCA at Contoso Consulting.

The qualified subordination will require that the following be in place:

* A Policy.inf is configured at each subordinate CA. The Policy.inf must define any name constraints, issuance constraints, and application constraints defined between the two organizations and is used during the cross-CA certificate request as described in the Walkthrough section.
* The user account that is performing the qualified subordination must obtain a certificate with the Qualified Subordination application policy OID to sign the qualified subordination request between the two organizations. The signing certificate must be obtained from the IssuingCA or any CA that is trusted on the CorpCA and PartnerCA computers for the first request and from the PolicyCA or any CA that is trusted on the CorpCA and PartnerCA computers for the second request.

The following configuration allows certificates issued by either CA to chain to a root that is trusted by each respective organization. For example, the certificate issued to User1 will validate up to the CorpCA if verified by a security principal at Northwind Traders.

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

If the same certificate is validated by a computer in the Contoso Consulting organization, the certificate will validate up to PartnerCA.

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **PolicyCA**Subject: **CorpCA**

Issuer: **PartnerCA**Subject: **PolicyCA**

Similarly, the certificate issued to User2 will chain to a trusted root in each organization. If the certificate is validated by a computer at Northwind Traders, the certificate chain will be as follows:

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **IssuingCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **CorpCA**Subject: **CorpCA**

While, the same certificate would validate up to the PartnerCA root CA when validated by a computer in the Contoso Consulting organization.

Issuer: **PolicyCA**Subject: user2

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

**Note** In all cross-certification designs, a single certificate will chain to different roots depending on which organization's computers are validating the certificate. In both examples, a certificate validated by computers at Contoso Consulting will chain to the PartnerCA root, while certificates validated by computers at Northwind Traders will chain to the CorpCA root.

As with the model where the cross-certification is established between root CAs, this design also allows trust of new CAs installed into the CA hierarchies at either organization. Figure 11 shows the addition of a new CA at Northwind Traders.

Contoso  
Consulting

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **IssuingCA**

Northwind  
Traders

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **PolicyCA**Subject: **CorpCA**

Issuer: **IssuingCA**Subject: **PartnerCA**

Figure 11: Adding an Additional CA to the Hierarchy

In this case, the certificate issued to User4 is trusted in both organizations. In Northwind Traders, the certificate chains to the CorpCA root CA.

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **CorpCA**Subject: **CorpCA**

While at Contoso Consulting, the certificate chains to the trusted PartnerCA.

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **PolicyCA**Subject: **CorpCA**

Issuer: **PartnerCA**Subject: **PolicyCA**

### Implementing Qualified Subordination to Limit Trust to Specific CAs

In some situations, two organizations may wish to restrict the cross-certification to specific CAs or to specific portions of the CA hierarchy, for example, where an organization uses outside consulting services. The organization (Northwind Traders) and consulting firm (Contoso) may wish to restrict the cross-certification to specific CAs in the hierarchy.

Figure 12 shows the scenario where the qualified subordination is performed to limit the trust between specific CAs in the organization.

Contoso  
Consulting

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**

Subject: **CorpCA**

Issuer: **IssingCA**Subject: **PolicyCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Northwind  
Traders

Issuer: **PolicyCA**Subject: **IssuingCA**

Figure 12: Configuring a Limited Trust Relationship

With this configuration, the certificates issued by the IssuingCA and the PolicyCA will validate up to a trusted root CA in both organizations.

At Northwind Traders, the certificate issued to user1 will chain as follows:

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **CorpCA**Subject: **CorpCA**

The certificate issued to user2 will chain as follows:

Issuer: **PolicyCA**Subject: **user2**

Issuer: **IssuingCA**Subject: **PolicyCA**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **CorpCA**Subject: **CorpCA**

At Contoso Consulting, the certificates will chain with the PartnerCA being the trusted Root CA. For the user1 certificate, the following chain will be selected by a validation process:

Issuer: **IssuingCA**Subject: **user1**

Issuer: **PolicyCA**Subject: **IssuingCA**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

For the user2 certificate, the validation process will also select the chain that chains to the PartnerCA root CA.

Issuer: **PolicyCA**Subject: user2

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

The major difference between this qualified subordination design and the design that allowed total trust between CAs in the two hierarchies is best shown by investigating an additional CA added to the Northwind Traders CA hierarchy as shown in Figure 13.

Contoso  
Consulting

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **IssuingCA**

Northwind  
Traders

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **IssingCA**Subject: **PolicyCA**

Issuer: **PolicyCA**Subject: **IssuingCA**

Figure 13: Adding an Additional CA to the Hierarchy

In this case, the ProjectCA was added to the Northwind Traders CA hierarchy. To a computer in the Northwind Traders network, the certificate issued to user4 is valid because it chains to a trusted root CA, the CorpCA.

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **CorpCA**Subject: **CorpCA**

When a computer from the Contoso Consulting organization is presented the user4 certificate, only a partial chain can be built and the chain does not terminate at a trusted root certificate.

### Types of Constraints

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **CorpCA**Subject: **CorpCA**

Not trusted

Qualified subordination allows constraints to be applied to a cross-certification configuration. Rather than implicitly trusting all certificates issued by another organization’s CA hierarchy, qualified subordination allows you to define application, naming, and issuance constraints.

For example, when qualified subordination is established between two organizations, some of the constraints they may wish to configure include:

* **Application policy** Rather than allowing all certificates to be trusted, specific application purposes can be defined for the issued crossCA certificate by defining what OIDs are required in certificates from the other organization. For example, if an organization wishes to only accept S/MIME certificates from a partner organization, an application policy can be defined to only accept certificates with the Secure Email OID (1.3.6.1.5.5.7.3.4) in the application policy extension or in the EKU extension.

**Note** Extended Key Usage is defined in part 4.2.1.13 in RFC 2459.

* **Name Constraints** Whenever qualified subordination is used between two organizations, the organization issuing the CrossCA certificate should include a name constraint that excludes its namespace through the cross-certification. This prevents the partner organization from submitting a certificate using the other organization’s namespace. For example, Northwind Traders would implement a name constraint that prevents Contoso from presenting certificates that are issued to a user from Northwind Traders. In other words, if a CA in the Contoso hierarchy issued a certificate with a subject name of user1@northwindtraders.com, the name constraint would exclude the nortwindtraders.com e-mail address and would invalidate the certificate.

**Note** Name Constraints are defined in part 4.2.1.11 in RFC 2459.

* **Policy Constraints** To ensure a level of trust for partner certificates, policy constraints (sometimes referred to as issuance policy) can be defined and must be included as attributes of presented certificates to be accepted by the partner organization. For example, if two partner organizations are involved in a buyer/seller relationship, a “Million Dollar” object ID can be defined. For purchases of 1,000,000 USD or more, the signing certificate must contain the Million Dollar OID. Because the OID must be defined in both organizations, the Policy.inf file must map the OID from one organization to the other organization’s OID. Likewise, constraints can be applied to whether the OID can be mapped to other OIDs, or whether the OID must be issued from a specific CA or limit how many layers below the cross-certified CA can issue a certificate containing the OID.

**Note** Policy Constraints is defined in part 4.2.1.5 in RFC 2459 and are referred to as the Certificate Policies extension in the RFC.

**Important** If different combinations of policy and name constraints are required for application policies, you may have to perform the qualified subordination multiple times. For example, if you are willing to accept any Secure Email certificates, but will only accept High and Medium Assurance OIDs for Client and Server authentication certificates, two separate Policy.inf files must be defined and the qualified subordination process must be executed twice—once for each Policy.inf file. This results in two separate Cross-Certification Authority certificates being issued—one for Secure Email certificates and one for Client and Server Authentication certificates. The cross-certification certificates are ultimately trusted for different purposes. Applications will verify the policy OIDs when making authentication or authorization decisions.

## Creating Trust Between Multiple Organizations

When you require trust between more than two, three, or more organizations, it is sometimes easier to design the qualified subordination by using a Bridge CA. The Bridge CA will act as a link between the CA hierarchies in each organization. Each organization will validate certificates issued by other organizations that use chains that include the Bridge CA. The Bridge CA design also reduces the complexity involved when new organizations are included in the trust.

A Bridge CA has several advantages over configuring separate trust relationships between multiple organizations, including

* A Bridge CA reduces the complexity for defining trust between CA hierarchies when three or more CA hierarchies exist.
* It is easier to add an additional organization to an existing Bridge CA trust relationship. The only tasks that must be performed are as follows:

1. Issue the Bridge CA a cross-certification certificate using qualified subordination from each organization.
2. Issue a cross-certification certificate from the Bridge CA for the root CA of the new organization.
3. Publish the Bridge CA issued certificate in all other organizations to complete the web of trust created by the BridgeCA.

* The Bridge CA can offer a method to manage trust between subsidiary organizations by a holding company.

**Important** Even if a BridgeCA is implemented, there is still nothing preventing two organizations participating in the Bridge CA structure from using qualified subordination to define a separate relationship between their hierarchies. For example, while Secure Mail certificates may be validated through the Bridge CA for e-mail sent between Northwind Traders and Contoso Consulting, a separate qualified subordination may be established between CAs in the two organizations to define specific policy constraints that must exist in certificates for signing purchase orders for transactions greater than 1 million dollars between the two organizations. Depending on the certificate’s attributes, the validation process will select the appropriate certificate chain based on the certificate’s usage and the defined qualified subordinations between the two organizations.

A common design for the Bridge CA deployment is shown in Figure 14.

Issuer: **BridgeCA**Subject: **PartnerCA**

Issuer: **IssuingCA**Subject: **BridgeCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **SubCA**Subject: **BridgeCA**

Issuer: **PolicyCA**Subject: **BridgeCA**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **SubCA**Subject: **user6**

Issuer: **OrgCA**Subject: **SubCA**

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **BridgeCA**Subject: **BridgeCA**

Issuer: **OrgCA**Subject: **OrgCA**

Issuer: **BridgeCA**Subject: **OrgCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **BridgeCA**Subject: **CorpCA**

Northwind Traders

Fabrikam Inc.

Contoso Consulting

Figure 14: Using a Bridge CA

In this design, total trust is established between all CAs in the three organizations, subject to any name, issuance, and application policies defined in the qualified subordination.

To achieve this, qualified subordination is defined so that the Bridge CA receives a subCA certificate defining all application, naming, and policy constraints from the subordinate enterprise CA in each organization. In addition, the root CA in each organization receives a subCA certificate defining all application, naming, and policy constraints from the Bridge CA.

**Note** Alternatively, the cross-certification can be performed between the root CA of each organization and the BridgeCA. Again, the decision will be based on the security policy of each organization using the bridge CA, whether the organizations are willing to issue a cross-certification certificate from their root CA to the bridge CA, and the CRL publication period of the root CAs.

**Important** Cross-certification of a CA hierarchy may create very long certificate chains that can increase the size of network traffic for IPSEC, Secure Mail (s/mime), SSL, and so on. Use caution so as not to increase the chain length and overall certificate chain byte size to exceed an application limit.

### Viewing Certificates from the Three Organizations

The result is complete trust between the organizations using the BridgeCA for the certificate purposes defined in the qualified subordination. As long as an issued certificate meets all defined application, name, and policy constraints, the certificate will be considered valid in any organization connected to the Bridge CA in this manner.

In this example, the user1 certificate would chain to the CorpCA trusted root for all computers at Northwind Traders as follows:

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **CorpCA**Subject: **CorpCA**

The user1 certificate would chain to the OrgCA trusted root for all computers at Fabrikam Inc. as follows:

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **BridgeCA**Subject: **CorpCA**

Issuer: **SubCA**Subject: **BridgeCA**

Issuer: **OrgCA**Subject: **SubCA**

Issuer: **OrgCA**Subject: **OrgCA**

If the same certificate is evaluated by the certificate chaining engine at a computer in the Contoso organization, the resulting certificate chain would be as follows:

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **IssuingCA**Subject: **user1**

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **BridgeCA**Subject: **CorpCA**

Issuer: **PolicyCA**Subject: **BridgeCA**

Issuer: **PartnerCA**Subject: **PolicyCA**

Likewise, the certificates for user6 and user2 would chain to the trusted root for each of the three organizations, based on what organization the computer evaluating the certificate belongs to.

### Adding New CAs to Existing CA Hierarchies

This design for Bridge CA deployment allows for the addition of new CAs to the hierarchies at any of the participating organizations, without having to modify the existing qualified subordination using the Bridge CA. For example, if the Project CA is added to the Northwind Traders CA hierarchy as shown in Figure 15, the certificate issued to user4 would be trusted by computers in the Northwind Traders, Fabrikam Inc., and Contoso Consulting organizations, subject to application, naming, and policy constraints.

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **IssuingCA**Subject: **BridgeCA**

Issuer: **SubCA**Subject: **BridgeCA**

Issuer: **PolicyCA**Subject: **BridgeCA**

Issuer: **SubCA**Subject: **user6**

Issuer: **OrgCA**Subject: **SubCA**

Issuer: **PolicyCA**Subject: **user2**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **BridgeCA**Subject: **BridgeCA**

Issuer: **OrgCA**Subject: **OrgCA**

Issuer: **BridgeCA**Subject: **OrgCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

Issuer: **BridgeCA**Subject: **PartnerCA**

Issuer: **CorpCA**Subject: **CorpCA**

Issuer: **BridgeCA**Subject: **CorpCA**

Northwind Traders

Fabrikam Inc.

Contoso Consulting

Issuer: **CorpCA**Subject: **IssuingCA**

Issuer: **ProjectCA**Subject: **user4**

Figure 15: Adding a New CA to the Northwind Traders CA Hierarchy

In this case, the user4 certificate would chain to the CorpCA trusted root for the computers in the Northwind Traders organization.

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **CorpCA**Subject: **CorpCA**

The certificate would chain to the OrgCA trusted root for the computers in the Fabrikam Inc. organization.

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **BridgeCA**Subject: **CorpCA**

Issuer: **SubCA**Subject: **BridgeCA**

Issuer: **OrgCA**Subject: **SubCA**

Issuer: **OrgCA**Subject: **OrgCA**

Finally, the certificate would chain to the PartnerCA trusted root for the computers in the Contoso Consulting organization.

Issuer: **ProjectCA**Subject: **user4**

Issuer: **CorpCA**Subject: **ProjectCA**

Issuer: **BridgeCA**Subject: **CorpCA**

Issuer: **PolicyCA**Subject: **BridgeCA**

Issuer: **PartnerCA**Subject: **PolicyCA**

Issuer: **PartnerCA**Subject: **PartnerCA**

### Using Name Constraints with Bridge CAs

To prevent CAs in the other organization from creating certificates with subject names from your organization’s namespace, name constraints should be created to exclude your namespace from certificates trusted through the BridgeCA.

For example, for the cross-certification authority certificate issued to BridgeCA from the CorpCA, the following name constraints could be added to the Policy.inf file at the CorpCA:

[NameConstraintsExtension]

Include = NameConstraintsPermitted

Exclude = NameConstraintsExcluded

Critical = TrUe

[NameConstraintsPermitted]

DNS = ""

email=""

UPN=""

[NameConstraintsExcluded]

DNS = .northwindtraders.com

email = @northwindtraders.com

UPN = .northwindtraders.com

UPN = @northwindtraders.com

DIRECTORYNAME = "DC=Northwindtraders,DC=com"

These name constraints will ensure that all namespaces except the NorthwindTraders namespace will be permitted.

**Note** Additional name constraints could be defined for NameConstraintsPermitted to restrict what specific namespaces are allowed into the organization through the BridgeCA.

### Policy Constraints and Policy Constraint Mapping

The qualified subordination between the organizations may also require specific policy constraints to exist for trusted certificates. If a specified policy constraint is not included in a certificate, the application should not select the chain including that certificate.

The problem with policy constraints is that for each CA hierarchy, the policy constraint may have a different name and it will have a different object identifier (OID). To make the policy constraints interoperate between the CA hierarchies, the policy constraints must be defined and mapped in the Policy.inf file.

For example, if the BridgeCA will function to only allow Low and Medium Assurance certificates to be passed between the organizations, the policy constraints must be defined in each cross-certification certificate issued between CAs.

For the cross-certification authority certificate issued to CorpCA by the BridgeCA, the Policy.inf file must contain the following lines to define the medium and low assurance OIDs. These OIDs may be obtained from the Certificate Templates console or from the IANA:

[PolicyStatementExtension]

Policies = MediumAssurance, LowAssurance

CRITICAL = FALSE

[MediumAssurance]

OID = 1.3.6.1.4.1.311.21.8.1608391590.1259233725.355412102.3300744578.1.401

[LowAssurance]

OID = 1.3.6.1.4.1.311.21.8.1608391590.1259233725.355412102.3300744578.1.400

In addition to defining the OIDs for its CA hierarchy, the Policy.inf file must also map the Medium and Low Assurance OIDS to the OIDS used at the BridgeCA. This is accomplished through the [PolicyMappingsExtension] section of the Policy.inf file:

[PolicyMappingsExtension]

1.3.6.1.4.1.311.21.8.1608391590.1259233725.355412102.3300744578.1.401 = 1.3.6.1.4.1.311.21.8.2812900009.1258984576.2698790533.44255976.1990074901.2398624335

1.3.6.1.4.1.311.21.8.1608391590.1259233725.355412102.3300744578.1.400 = 1.3.6.1.4.1.311.21.8.2812900009.1258984576.2698790533.44255976.572751058.2135922791

critical = yEs

This section maps the OID ID for medium assurance to an object ID defined at the BridgeCA. The OID does not have to be for the MediumAssurance policy constraint. In fact, it can be for a user-defined policy OID, as the previous example shows.

To complete the policy mappings, the following lines must be included in the Policy.inf file used by the BridgeCA to receive a cross-certification authority certificate from the IssuingCA:

[PolicyStatementExtension]

Policies = SecLevel1, SecLevel2

CRITICAL = FALSE

[SecLevel1]

OID = 1.3.6.1.4.1.311.21.8.2812900009.1258984576.2698790533.44255976.1990074901.239862433

[SecLevel2]

OID = 1.3.6.1.4.1.311.21.8.2812900009.1258984576.2698790533.44255976.572751058.2135922791

[PolicyMappingsExtension]

1.3.6.1.4.1.311.21.8.2812900009.1258984576.2698790533.44255976.1990074901.239862433 = 1.3.6.1.4.1.311.21.8.1608391590.1259233725.355412102.3300744578.1.401

1.3.6.1.4.1.311.21.8.2812900009.1258984576.2698790533.44255976.572751058.2135922791 = 1.3.6.1.4.1.311.21.8.1608391590.1259233725.355412102.3300744578.1.400

critical = yEs

As you can see, custom object identifiers were defined for the policies named SecLevel1 and SevLevel2. In the [PolicyMappingsExtension] section, the same OIDs are mapped with the order reversed from the Policy.inf file used by the CorpCA to receive a cross-certification authority certificate from the BridgeCA.

Figure 16 shows another example of policy constraint mapping.

[PolicyStatementExtension]  
Policies=MillionDollar  
Critical=False

[MillionDollar]  
OID=1.3.6.1.4.1.311.21.8.1.124

[PolicyMappingExtension]  
1.3.6.1.4.1.311.21.8.1.124= 1.3.6.4.1.204.22.33.44

[PolicyStatementExtension]  
Policies=BigOrder  
Critical=False

[BigOrder]  
OID=1.3.6.4.1.204.22.33.44

[PolicyMappingExtension]  
1.3.6.4.1.204.22.33.44=1.3.6.1.4.1.311.21.8.1.124

NorthwindCA

ContosoCA

Issuer: **NorthwindCA**Subject: **User1  
Certificate Policy =** 1.3.6.1.4.1.311.21.8.1.124

Issuer: **NorthwindCA**Subject: **User2  
Certificate Policy =**1.3.6.4.1.204.22.33.44

Equivalent

Figure 16: Policy Constraint Mapping

In this figure, the NorthwindCA is configured with a MillionDollar OID, 1.3.6.1.4.1.311.21.8.124. This OID is mapped to the BigOrder OID, 1.3.6.4.1.204.22.33.44, at the ContosoCA. Although the certificate issued to  
v-user1 contains a different issuance policy OID from the certificate issued to user2, the policy constraint mapping makes the OIDs equivalent and the OIDs are recognized by both CA hierarchies.

### Application Policy

The final policy that could be applied in this scenario is the application policy. Application policy will limit which purposes certificates issued by another organization’s CA’s can be used for. By defining application policy, you can prevent certificates for specific usages from being accepted for use in your organization.

For example, if you only want to allow certificates for S/MIME, client authentication, and server authentication to be recognized through the BridgeCA, you can add the following sections to the Policy.inf file:

[ApplicationPolicyStatementExtension]

; list of user defined policies

Policies = AppOIDPolicy1, AppOIDPolicy2, AppOIDPolicy3

[AppOIDPolicy1]

OID = 1.3.6.1.5.5.7.3.4 ; Secure Email

[AppOIDPolicy2]

OID = 1.3.6.1.5.5.7.3.1 ; Server Authentication

[AppOIDPolicy3]

OID = 1.3.6.1.5.5.7.3.2 ; Client Authentication

CRITICAL = FALSE

These application policies would prevent the use of an EFS certificate between the organizations. An EFS certificate would not chain in a trusted manner because the defined application policies define an application policy for EFS encryption services.

**Note** Remember that these application policies also are combined with name constraints and policy constraints when applied to a certificate presented from another CA hierarchy. If you require different combinations, you must request multiple cross-certification authority certificates using separate Policy.inf files for each request. For example, you may not wish to implement policy constraints for e-mail certificates, but want to ensure that low or medium assurance OIDs exist in client and server authentication certificates. This would require two separate cross-certification authority certificate requests—one for Secure Email and one for Client and Server Authentication.

# Walkthrough

This section provides detailed implementation steps involved in defining qualified subordination between two CA hierarchies.

## Creating an Offline Bridge CA

The following process can be used to deploy an offline root CA for your organization. The following is required before you establish your offline Bridge CA:

* Configure a CAPolicy.inf file to define the following settings for the offline Bridge CA.
* Authority Information Access locations
* CRL Distribution Point locations
* Certificate Practice Statement
* All issuance policies required to cross the Bridge CA
* Determine the CA’s distinguished name.
* Determine what Cryptographic Service Provider (CSP) will be used for certificate creation and signing.

### Configuring the CAPolicy.inf File

The Offline Bridge CA must include the following sections in a CAPolicy.inf file stored in the %Systemroot% folder. This example shows an addition of three policy settings for the Offline Bridge CA. The OIDs for the three policies were obtained from the organization's Active Directory. Alternatively, the OIDs could be obtained from the IANA at <http://www.iana.org/cgi-bin/enterprise.pl> or by running OIDGEN.EXE from the Windows Server 2003 Resource Kit.

[Version]

Signature= "$Windows NT$"

[certsrv\_server]

CRLPeriod = weeks

CRLPeriodUnits = 26

[PolicyStatementExtension]

Policies = HighAssurancePolicy, MediumAssurancePolicy, LowAssurancePolicy

CRITICAL = FALSE

[HighAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.402

[MediumAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.401

[LowAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.400

[PolicyConstraintsExtension]

RequireExplicitPolicy = 0

InhibitPolicyMapping = 0

[BasicConstraintsExtension]

PathLength = 0

**Note** As mentioned previously, it is not a best practice to place policy OIDs in root CA certificates. It is also not a best practice to place CDP or AIA extensions in the root CA certificate as most applications do not check for root CA revocation.

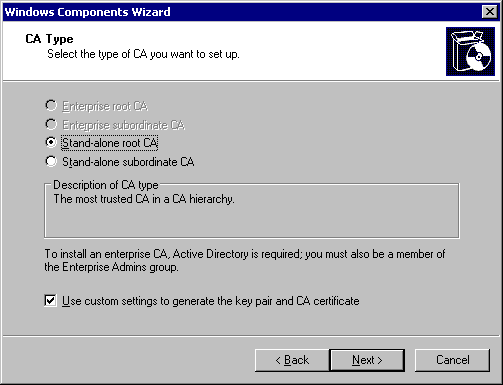
### Installing Certificate Services

The actual installation process of a root CA is performed using the following steps:

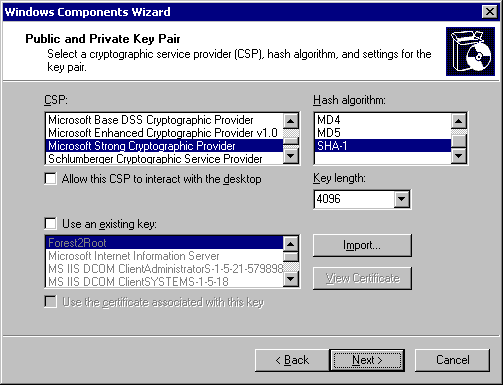
1. Ensure that the updated CAPolicy.inf file is saved to the %systemroot% folder.

**Note** Appendix B contains an example of a CAPolicy.inf file.

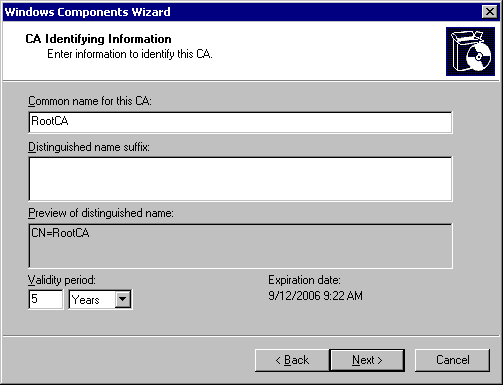
1. On the Start menu, click Settings, and then click Control Panel.
2. On the Control Panel, double-click Add or Remove Programs.
3. In the Add or Remove Programs window, click Add/Remove Windows Components.
4. In the Windows Components Wizard, on the Windows Components page, select the Certificate Services check box.
5. In the Microsoft Certificate Services dialog box, click Yes to accept that you cannot rename the computer or change its domain membership.
6. On the Windows Components page, click Next.
7. In the CA Type page (Figure 17), click Stand-alone root CA, select the Use custom settings to generate the key pair and CA certificatecheck box, and then click Next.

Figure 17: Defining the CA Type Configuration

1. On the Public and Private Key Pair page (Figure 18), select the CSP and the Key length (for the private and public key pair), and then click Next.

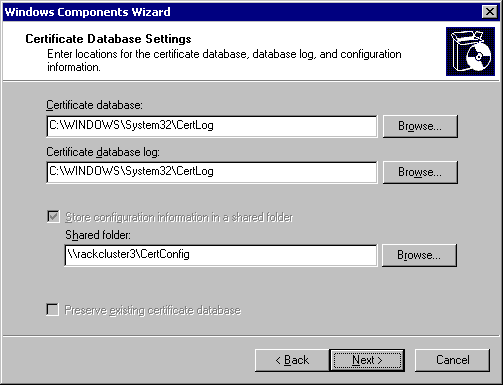
Figure 18: Defining the Public and Private Key Pair Properties

1. On the CA Identifying Information page (Figure 19), enter the Common name and Validity period (for the Root CA certificate), and then click Next.

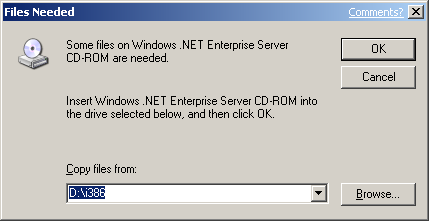
Figure 19: Defining the Common Name of the CA and Setting the Validity Period

**Note** The Cryptographic Key is now generated based on the provided information in the preceding dialog boxes. Depending on the length of key defined on the Public and Private Key Pair page, this may take some time.

1. On the Certificate Database Settings page (Figure 20), accept the default locations, and then click Next.

Figure 20: Defining the Database and Log Storage Locations

1. In the Microsoft Certificate Services dialog box, click Yes to stop the Internet Information Services.  
   The Configuring Components dialog box appears as the installation proceeds.
2. If the Files Neededdialog box appears (Figure 21), in the Copy files from box, enter the CD-ROM drive path or the network share path where the distribution files are located, and then click OK.

Figure 21: Providing the Location of the Installation Files**

1. In the Windows Components Wizard, click Finish.
2. In the Add or Remove Programs dialog box, click Close.
3. Close the Control Panel.

### Disabling Delta CRLs at the Root CA

Typically, the root CA for a CA hierarchy is removed from the network for security purposes. Periodically, the CA must be accessed for the publication of updated CRLs. The goal of delta CRLs is to publish the newly revoked certificates at a more frequent interval than the publication period for the base CRL. By disabling delta CRLs, you reduce the number of times that the offline root CA must be accessed for CRL publication. It is typically not an issue that the delta CRLs are not published, as the number of certificates issued by the root CA is minimal.

**Important** If cross-certification is performed between root CAs, it is possible that there may be a delay between the revocation of a former partner's cross-certification certificate and the recognition of the revocation by the clients accessing the base CRL.

1. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
2. In the Certification Authority console, in the console tree, expand *CAName* (where *CAName* is the name of your offline Bridge CA).
3. In the console tree, right-click Revoked Certificates, and then click Properties.
4. In the Revoked Certificates Properties dialog box, click to clear the Publish Delta CRLs check box, and then click OK.
5. In the console tree, right-click Revoked Certificates, click All Tasks, and then click Publish to publish a new base CRL that does not reference a Delta CRL.
6. In the Publish CRL dialog box, click OK.

### Removing the Non-Required AIA and CDP Extensions

Once the offline Bridge CA is installed, the CDP and AIA locations for issued certificates must be modified to ensure that the certificate chaining engine can find the CRL and certificate for the offline Bridge CA. Because the CA is removed from the network, these default AIA and CDP extensions do not reference valid locations for the publication points. By removing the offline root CA from the network, you effectively remove the default publication points.

To remove non-required CDP and AIA extensions

1. In the Certification Authority console tree, right-click CAName, and then click Properties.
2. In the CAName Properties dialog box, click the Extensions tab.
3. In the Select extension box, ensure that the box contains CRL Distribution Point (CDP).
4. In the list of CDP locations, select the CDP location beginning with ldap:///, and then click Remove.
5. In the Confirm removal dialog box, click Yes to verify the removal.
6. In the list of CDP locations, select the CDP location beginning with http://, and then click Remove.
7. In the Confirm removal dialog box, click Yes to verify the removal.
8. In the list of CDP locations, select the CDP location beginning with file://\\, and then click Remove.
9. In the Confirm removal dialog box, click Yes to verify the removal.

**Important** Do not delete the CDP location referencing c:\windows\System32. This location is required by the CA to locally publish update Certificate Revocation lists.

1. On the Extensions tab, select Authority Information Access in the Select extensionbox.
2. In the list of AIA locations, select the AIA location beginning with ldap:///, and then click Remove.
3. In the Confirm removal dialog box, click Yes to verify the removal.
4. In the list of AIA locations, select the AIA location beginning with http://, and then click Remove.
5. In the Confirm removal dialog box, click Yes to verify the removal.
6. In the list of AIA locations, select the AIA location beginning with file://\\, and then click Remove.
7. In the Confirm removal dialog box, click Yes to verify the removal.

**Important** Do not delete the AIA location referencing c:\windows\System32. This location is required by the CA to locally publish renewed Certificates.

1. Click OK.
2. In the Certification Authority dialog box, click Yesto restart Certificate Services.

### Adding Required AIA and CDP Extensions

The offline Bridge CA must add required CDP and AIA extensions to allow Active Directory clients and other clients to find the CA’s certificate and CRL. You must modify the extensions to include paths that are available when the offline Bridge CA is removed from the network.

The examples provided in this white paper are not mandatory, but should be considered when defining the AIA and CDP extensions for the offline root CA.

**Note** The AIA and CDP extensions are required by the certificate chaining engine to build certificate chains for validation. For more information on the certificate validation process, see the Microsoft white paper Troubleshooting Certificate Status and Revocationat http://www.microsoft.com/technet/security/prodtech/tshtcrl.asp

#### Adding the LDAP CDP Extension

The first CDP extension that needs to be added is the LDAP:// CDP extension. This CDP extension allows Active Directory and other LDAP clients to acquire the CRL from an LDAP service available to the Internet.

**Note** In some Windows 2000 and Windows Server 2003 deployments, a separate Active Directory forest is deployed in the extranet to prevent external access to the corporate Active Directory. If a separate LDAP service is used in the extranet, the LDAP URL must reference this Active Directory service. Otherwise, the LDAP URL must reference a link that is accessible by all clients that require access to the CRL.

To add the CRL to Active Directory, you will need the following information from the forest root domain of Active Directory that will maintain a copy of the offline Root CAs CRL:

*ForestRootDomain* The LDAP representation of the forest root domain. For example security.nwtraders com would be represented as DC=security,DC=Nwtraders,DC=com.

You will also require some information from the offline Bridge CA. Specifically, you require the following:

*CAName*: The name assigned to the CA during installation of the CA.

*CAMachineName*: The NetBIOS name of the computer hosting the CA.

You will need to add the two following LDAP paths to the CDP extensions list:

ldap:///CN=*CAName*,CN=*CAMachineName*,CN=CDP,CN=Public Key Services,  
CN=Services,CN=Configuration,*CAForestName*?certificateRevocationList?base?objectClass=cRLDistributionPoint

and

ldap://ldap.company.com/CN=*CAName*,CN=*CAMachineName*,CN=CDP,CN=Public Key Services,  
CN=Services,CN=Configuration,*CAForestName*

where ldap.company.com is an URL that refers queries to TCP port 389 to the server hosting the LDAP service, typically a Windows 2000 domain controller.

For example, for the following information:

* *CAName*: BridgeCA
* *CAMachineName*: BridgeComp
* *ForestRootDomain*: extranet.nwtraders.com

The following are the LDAP paths you must add to the list of CDP extensions:

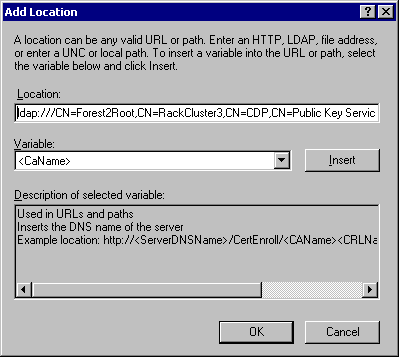
ldap:///CN=BridgeCA,CN=BridgeComp,CN=CDP,CN=Public Key Services,  
CN=Services,CN=Configuration,DC=extranet,DC=nwtraders,DC=com?certificateRevocationList?base?objectClass=cRLDistributionPoint

and

ldap://ldap.extranet.nwtraders.com/CN=BridgeCA,CN=BridgeComp,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=extranet,DC=nwtraders,DC=com

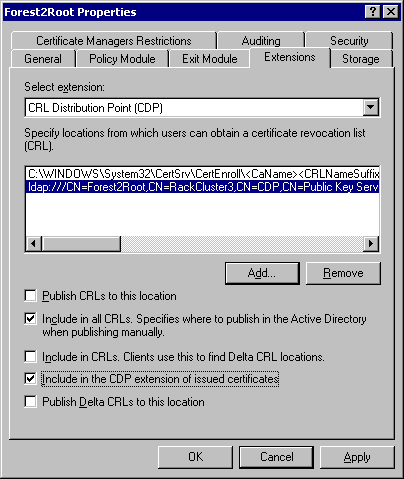
**Note** The LDAP:/// URL is a local LDAP path that binds to the current domain controller used by the client computer. This LDAP URL will not work with external clients, therefore, the second LDAP URL is added to provide the URL to the LDAP server for external clients.

1. In the Certification Authority console tree, right-click CAName, and then click Properties.
2. In the CAName Properties dialog box, click the Extensions tab.
3. Ensure that the Select Extensions list contains CRL Distribution Points (CDP), and then click Add.
4. In the Add Location dialog box (Figure 22), in the Location box, type the LDAP path as described previously, and then click OK.

Figure 22: Adding an LDAP CDP Extension

**Tip** Consider creating the path in a text file beforehand and then copying and pasting the complete LDAP URL into the Location box. There is no control to allow editing of the LDAP path if it is incorrectly typed in the console.

1. On the Extensions tab, select the LDAP CDP extension. Select the Include in all CRLS. Specifics where to publish in the Active Directory when publishing manually and Include in the CDP extension of issued certificates check boxes as shown in Figure 23. The first option is recommended when publishing offline CA CRLs to Active Directory. The second option ensures that all issued certificates include the updated paths to the CRL.

Figure 23: Configuring the Properties of the LDAP Extension

Repeat the process using the LDAP URL ldap://ldap.company.com/CN=*CAName*,CN=*CAMachineName*,CN=CDP,CN=Public Key Services,  
CN=Services,CN=Configuration,*CAForestName*

1. Leave the dialog box open to allow the addition of an HTTP CDP extension.

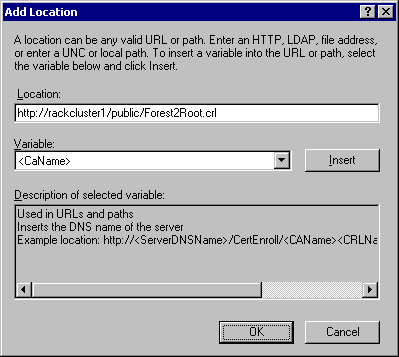
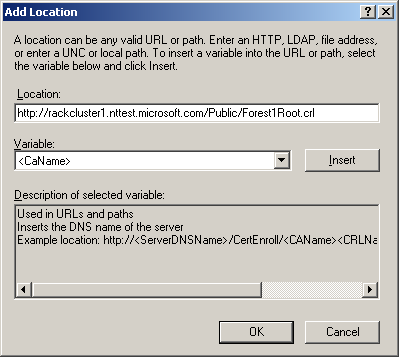
**Note** It is recommended to include an HTTP CDP extension to ensure that non-LDAP clients can access the offline CA's CRL. It also is easier to deploy the CRL to external clients using the HTTP protocol than the LDAP protocol.

#### Adding the HTTP CDP Extension

Additionally, at least one CDP extension should be added for the HTTP protocol that references an available Web server hosting the CRL file for the offline CA. You should provide more than one HTTP location to provide fault tolerance in the event that the Web server referenced in the CDP extension is unavailable. The URL must reference a valid HTTP location to work correctly.

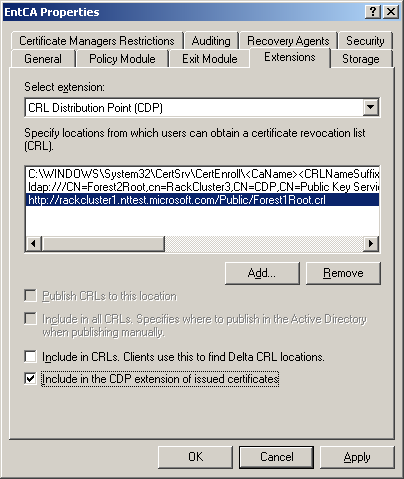
To add an HTTP CDP extension

1. In the CAName Properties dialog box, click the Extensions tab.
2. Ensure that the Select Extensions list contains CRL Distribution Points (CDP), and then click Add.
3. In the Add Location dialog box (Figure 24), in the Location box, type the HTTP path as described previously, and then click OK.

Figure 24: Adding an HTTP CDP Extension

**Tip** Consider creating the path in a text file beforehand and then copying and pasting the complete HTTP URL into the Location box. There is no control to allow editing of the HTTP path if it is incorrectly typed in the console.

1. On the Extensions tab, select the HTTP CDP extension, and then select the Include in the CDP extension of issued certificates check box as shown in Figure 25.

Figure 25: Configuring the Properties of the HTTP Extension

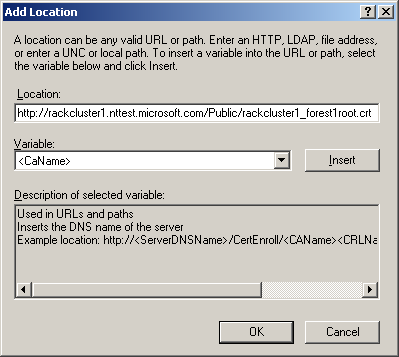
1. In the CAName Properties dialog box, click OK.
2. In the Certification Authority dialog box, click Yes to restart Certificate Services.

#### Adding an HTTP AIA Extension

In addition to CDP extensions, an AIA extension should be added to allow Web clients to access the offline Bridge CA’s certificate from a Web server.

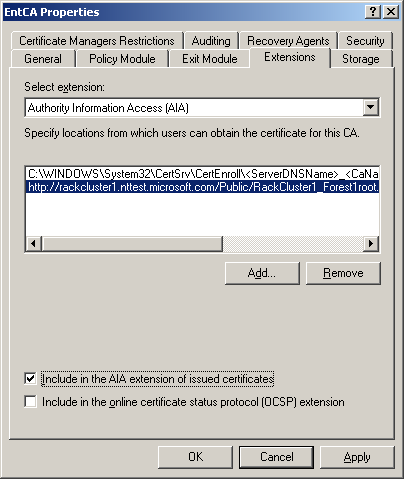
To add an updated HTTP AIA extension

1. In the Certification Authority console tree, right-click CAName, and then click Properties.
2. In the CAName Properties dialog box, click the Extensions tab.
3. Ensure that the Select Extensions list contains Authority Information Access (AIA), and then click Add.
4. In the Add Location dialog box (Figure 26), in the Location box, type the HTTP path to the Root CA certificate, and then click OK.

Figure 26: Adding an HTTP AIA Extension**

**Tip** Consider creating the path in a text file beforehand and then copying and pasting the complete HTTP URL into the Location box. There is no control to allow editing of the HTTP path if it is incorrectly typed in the console.

1. On the Extensions tab, select the HTTP AIA extension, and then select the Include in the AIA extension of issued certificates check boxes as shown in Figure 27.

Figure 27: Configuring the Properties of the LDAP Extension**

1. In the CAName Properties dialog box, click OK.
2. In the Certification Authority dialog box, click Yes to restart Certificate Services.

### Publishing CRLs and Certificates

Once the CDP and AIA extensions are modified, you must publish the updated CRL referencing the new CDP locations. The previous version of the CRL still contains the default locations for the CDP and AIA extensions.

To publish the new base CRL

1. In the console tree, right-click Revoked Certificates, click All Tasks, and then click Publish to publish a new base CRL.
2. In the Publish CRL dialog box, click OK.

Once the CRL is updated, the new version must be published to the locations referenced in the CDP and AIA extensions. This requires manually copying the CRL and Certificate to the Web servers and injecting the CRL and certificate into Active Directory.

#### Publishing the CRL and CA Certificate to a Web Server

The CRL and CA Certificate can be retrieved from the [\\*CAMachineName*\CertEnroll](file:///\\CAMachineName\CertEnroll) share and copied to a disk or to a network location. The CRL and certificate must be published to the location referenced in the CDP and AIA extensions of the CA.

For example, if the CDP extension references <http://www.microsoft.com/public/rootca.crl>, the CRL must be copied to this location so that a Web client can retrieve the latest version of the CRL.

Likewise, if the AIA extension references <http://www.microsoft.com/public/rootca.crt>, the CA’s certificate file must be copied to this location so that a Web client can retrieve the CA’s certificate.

By default, the IIS default Web site is stored in \inetpub\wwwroot in the same drive where the Windows Server 2003 operating system is installed. To create the paths referenced above, you can create a Public subfolder and copy the two files into the Public folder (\inetpub\wwwroot\public).

Alternatively, you can create a virtual folder and have /Public reference any folder on the Web server’s disk.

Once the CRL and certificate are published, you should validate that the paths are correct in the issued certificates by connecting to the CRL and certificate using the CDP and AIA URLs.

#### Publishing the CRL and CA Certificate into Active Directory

The Certutil.exe utility is used to publish the CA’s certificate and CRL into the Configuration naming context of Active Directory. The CA's certificate is published to two locations:

* CN=*CAName*,CN=Certification Authorities,CN=Public Key Services,CN=Services,  
  CN=Configuration,DC=*ForestRootDomainDN*
* CN=*CAName*,CN=AIA,CN=Public Key Services,CN=Services,  
  CN=Configuration, DC=*ForestRootDomainDN*

The CA's CRL is published to the following location in the Configuration naming context:

* CN=*CAName*,CN=*CAComputerName*,CN=CDP,CN=Public Key Services,CN=Services CN=Configuration,DC=*ForestRootDomainDN*

The Certutil.exe utility is only installed at a CA by default, but can be installed at any Windows XP client by installing the Windows Server 2003 Administration Pack (adminpak.msi).

Assuming that the CA’s certificate is named RootCA.crt and that the CA’s CRL is named RootCA.crl, and that the files are stored in the current directory, the following commands can be run to inject the CA’s certificate and the CA’s CRL into Active Directory.

To install the CA’s CRL into Active Directory, use the following command:

certutil.exe -dspublish -f RootCA.crl

If the command is successful, you should see the following output in response to the command:

ldap:///CN=RootCA,CN=RackCluster3,CN=CDP,CN=Public Key Services,CN=Services

,CN=Configuration,DC=bkforest2,DC=nttest,DC=microsoft,DC=com?certificateRevocati

onList?base?objectClass=cRLDistributionPoint?certificateRevocationList

Base CRL added to DS store.

CertUtil: -dsPublish command completed successfully.

To install the CA’s certificate into Active Directory, a variation of the Certutil –dspublish command is used:

Certutil –dspublish –f RootCA.crt RootCA

If successful, the command should produce the following output:

ldap:///CN=RootCA,CN=Certification Authorities,CN=Public Key Services,CN=Se

rvices,CN=Configuration,DC=bkforest2,DC=nttest,DC=microsoft,DC=com?cACertificate

Certificate added to DS store.

ldap:///CN=RootCA,CN=AIA,CN=Public Key Services,CN=Services,CN=Configuratio

n,DC=bkforest2,DC=nttest,DC=microsoft,DC=com?cACertificate

Certificate added to DS store.

CertUtil: -dsPublish command completed successfully.

**Note** The certificate is published to both the Certification Authorities and AIA locations under Public Key Services.

## Installing the Enterprise Subordinate CAs

The enterprise subordinate CAs must be installed with the issuance policies defined for the CA certificate. The following processes describe the installation of the Enterprise Subordinate CA.

### Configuring the CAPolicy.inf File

The Enterprise Subordinate CA requires that a CAPolicy.inf file be created and installed in the %systemroot% folder to apply the correct policies. If the Enterprise CA already exists, you can still use CAPolicy.inf and renew the CA’s certificate to apply the modified settings. The CAPolicy.inf file must contain the following sections:

[RequestAttributes]

CertificateTemplate = SubCA

[PolicyStatementExtension]

Policies = HighAssurancePolicy, MediumAssurancePolicy, LowAssurancePolicy

CRITICAL = FALSE

[HighAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.402

[MediumAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.401

[LowAssurancePolicy]

OID = 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1.400

**Note** The OIDS for the assurance policies must be modified to match the OIDS for the forest where the Enterprise Subordinate CAs exist. Each forest will generate a unique OID that is used for the high, medium, and low assurance issuance policies. In this example, the unique component of each OID is 1.3.6.1.4.1.311.21.8.2473717464.1095930238.502626717.506190032.1, You must change the OIDs shown in this example to match the unique OID representing your forest.

To modify the CAPolicy.inf file

1. Copy the modified CaPolicy.inf file to the %systemroot% folder.
2. Open the CaPolicy.inf file using Notepad.
3. Click Start, and then click Run. In the Openbox, type **certtmpl.msc**.
4. In the console tree, right-click Certificate Templates, and then click View Object Identifiers.
5. In the View Object Identifiers dialog box, select High Assurance, and then click Copy Object Identifier.
6. Paste the updated object identifier for High Assurance into the CaPolicy.inf file.
7. In the View Object Identifiers dialog box, select Medium Assurance, and then click Copy Object Identifier.
8. Paste the updated object identifier for Medium Assurance into the CaPolicy.inf file.
9. In the View Object Identifiers dialog box, select Low Assurance, and then click Copy Object Identifier.
10. Paste the updated object identifier for Low Assurance into the CaPolicy.inf file.
11. In the View Object Identifiers, click Close.
12. Close the Certificate Templates console.

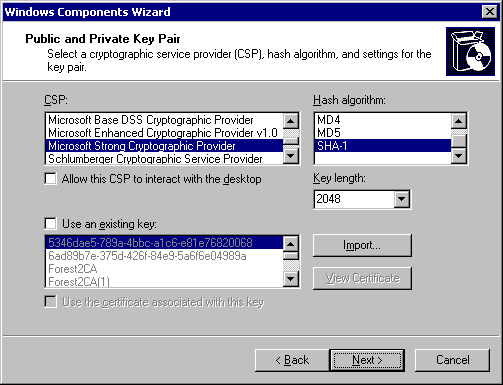
### Installing Certificate Services

Once the CAPolicy.inf file is configured, the installation of the Enterprise CA can begin. The process is made up of three steps: the initial installation of the enterprise subordinate CA, the offline certificate request process, and the installation of the SubCA certificate.

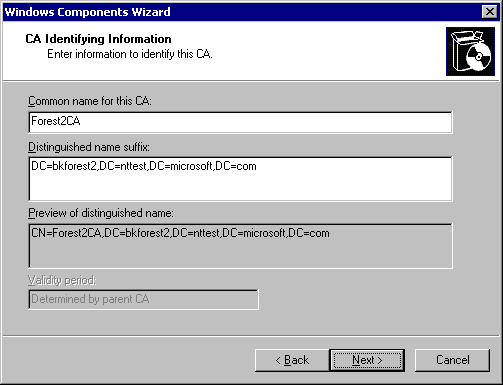
#### Installing the Enterprise Subordinate CA

To perform the initial installation of certificate services on the enterprise subordinate CA

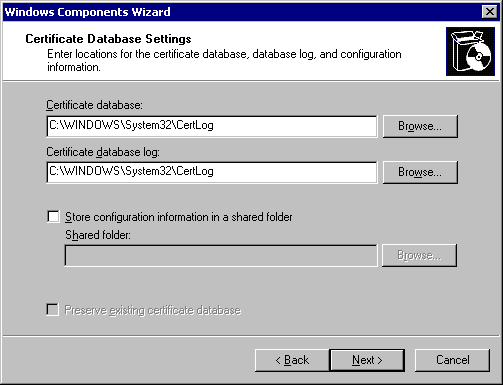
1. On the Start menu, click Settings, and then click Control Panel.
2. In the Control Panel, double-click Add or Remove Programs.
3. In the Add or Remove Programs window, click Add/Remove Windows Components.
4. In the Windows Components Wizard, on the Windows Components page, select the Certificate Services check box.
5. In the Microsoft Certificate Services dialog box, click Yes to accept that you cannot rename the computer or change its domain membership.
6. In the Windows Components page, click Next.
7. In the CA Type page, click Enterprise Subordinate CA, select the Use custom settings to generate the key pair and CA certificate check box, and then click Next.
8. On the Public and Private Key Pair page, define the CSP, Hash algorithm, and Key length the CA will use for the private and public key pair (Figure 28), and then click Next.

Figure 28: Defining the Public and Private Key Settings

1. On the CA Identifying Information page (Figure 29), enter the Common name for this CA, and then click Next.

Figure 29: Defining the CA Name

1. On the Certificate Database Settings page (Figure 30), accept the default settings, and then click Next.

Figure 30: Defining the Certificate Database and Log Locations

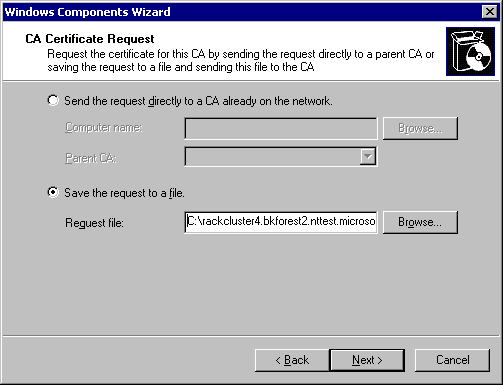
1. On the CA Certificate Request page (Figure 31), select the Save the request to a file option button, ****accept the default Request file name, and then click Next.

Figure 31: Configuring an Offline Certificate Request

**Note** This assumes that the root CA or parent CA is offline and unable to receive online requests.

1. In the Microsoft Certificate Services dialog box, click Yes.
2. If the Files Needed dialog box appears, enter the path to the installation files, and then click OK.

**Note** A dialog box appears indicating that the installation is incomplete and that the request file must be submitted to the parent CA.

1. In the Microsoft Certificate Services dialog box, click OK.
2. In the Windows Components Wizard, click Finish.

#### Signing the Offline Certificate Request

To sign the offline certificate request (These steps are performed at the offline Root CA or the offline parent CA.)

1. Copy the request file to a disk.
2. Take the disk to the offline root CA.
3. Open the request file using Notepad. You should see content similar to the following example:

-----BEGIN NEW CERTIFICATE REQUEST-----

MIICJTCCAY4CAQAwPjETMBEGCgmSJomT8ixkARkWA2NvbTETMBEGCgmSJomT8ixk

ARkWA2FiYzESMBAGA1UEAxMJSXNzdWluZ0NBMIGfMA0GCSqGSIb3DQEBAQUAA4GN

ADCBiQKBgQDD7j/MtDoqG0ZWdGSkF7h+taDOD5fB8JhTqIgx31maN+YQE288n7Vm

xtHH7a6Mo+hTRyNBr9gut3ZD4+CNETE+ek3SAsqu/7yXKPzlURlrniKWSAQ9kseO

9llLNFVAWwE8dwxR/taqMAKW1hxflub7p7qnL95eqLzzLfzPfqHwoQIDAQABoIGm

MCkGCisGAQQBgjcNAgMxGxYZNS4xLjM1OTAuMi5TZXJ2aWNlIFBhY2sgMTB5Bgkq

hkiG9w0BCQ4xbDBqMBAGCSsGAQQBgjcVAQQDAgEAMB0GA1UdDgQWBBS64o3oWvft

Zy8bMazXc1SZsyCx2jAZBgkrBgEEAYI3FAIEDB4KAFMAdQBiAEMAQTALBgNVHQ8E

BAMCAYYwDwYDVR0TAQH/BAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQAl15B50lwN

AfsSgFKTuPELSalkjWmnn11ZPAmGCLHnhZ6yhQwonWntN3nRaU7F1+KfEnvoibb2

DM1x7SLoMEzQrWQ8sWneoBSCtD0Sdg24dIpWwxlnKgsImRlfGnlEQEd/VHTgyxSh

hrS1gQXYQYH0CT8giCZ8PwGsg1qr/8dMIg==

-----END NEW CERTIFICATE REQUEST-----

1. Copy the entire contents of the request file to the Windows Clipboard.
2. Start Internet Explorer.
3. Open the [http://localhost*/*certsrv](http://localhost/certsrv) URL.
4. On the Welcome page, click the Request a certificate link.
5. On the Request a Certificate page, click advanced certificate request.
6. In the Advanced Certificate Request page, click Submit a certificate request by using a base-64-encoded CMC or PKCS#10 file, or submit a renewal request by using a base-64-encoded PKCS#7 file.
7. In the Submit a Certificate Request or Renewal Request page, paste the contents of the request file in the Saved Request box, and then click Submit.
8. The Certificate Pending page appears informing you that an administrator must issue the certificate. You should note the request ID for future reference.
9. Close Internet Explorer.

To issue the certificate, you must use the Certification Authority Console.

1. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
2. In the console tree, expand CAName (where *CAName* is the name of your Certification Authority), and then click the Pending Requests container.
3. In the details pane, right-click the pending request, identified by its request ID, for the previously submitted request, click All Tasks, and then click Issue.
4. Close the Certification Authority console.

To retrieve the issued certificate (These steps are performed at the offline root CA or offline parent CA.)

1. Start Internet Explorer.
2. Open the [http://localhost*/*certsrv](http://localhost/certsrv) URL.
3. On the Welcome page, click the View the status of a pending certificate requestlink.
4. On the View the Status of a Pending Certificate Request page, click the Saved-Request Certificate (Date and Time) link.
5. On the Certificate Issued page, click the Download certificate chain link.
6. In the File Download dialog box, click Save.
7. In the Save As dialog box, in the file name box, type **a:\certnew.p7b**, and then click Save to save the PKCS#7 file to the disk.
8. If the Download complete dialog box appears, click Close.
9. Close Internet Explorer.

#### Installing the Certificate at the Enterprise Subordinate CA

The PKCS#7 file must now be installed at the enterprise subordinate CA using the following steps:

1. Insert the disk containing the PKCS#7 file in the disk drive.
2. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
3. In the console tree, right-click CAName (where *CAName* is the name of your enterprise subordinate CA), point to All Tasks, and then click Install CA Certificate.
4. In the Select file to complete CA Installation dialog box, in the File name box, type **a:\certnew.p7b**, and then click Open.
5. In the console tree, right-click CAName, point to All Tasks, and then click Start Service.

**Note** It may be required to also install the offline root CA's CRL manually if the CRL cannot be accessed directly from the network. You can load the CRL file using the certutil.exe –addstore <*CRL file>* command.

### Renewing an Existing Enterprise Subordinate CA

1. Ensure that the modified CAPolicy.inf file is in the %SystemRoot% folder.  
   This ensures that either the newly defined constraints are applied or that the existing policy constraints are retained.
2. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
3. In the console tree, right-click CAName, and then click Renew CA Certificate.
4. In the Install CA Certificate dialog box, click Yes to stop Certificate Services.

**Note** In addition to defining constraints, the key size, key lifetime, CDP locations, AIA locations, and other settings defined in the CAPolicy.inf file can be redefined at certificate renewal.

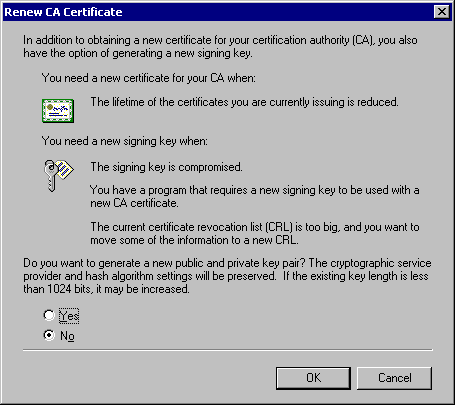
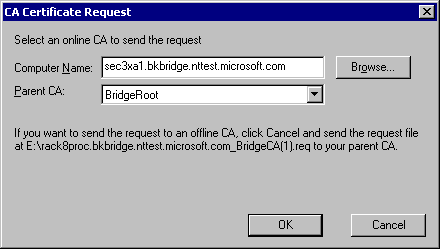
1. In the Renew CA Certificate dialog box (Figure 32), click No so that the same public and private key pair is used, and then click OK.

Figure 32: Selecting Whether to Generate a New Public and Private Key Pair

1. In the CA Certificate Request dialog box (Figure 33), enter the Computer Name and Parent CA name for the issuing CA, and then click Cancel. This will place a *CAMachineName*\_*CAName.reg* file in the root folder of the volume where Windows Server 2003 is installed.

Figure 33: Selecting the CA to Which the CA Will Be a Subordinate CA

**Note** If the CA that issued the SubCAs certificate is online, you can click OK to attempt the certificate request online.

1. Copy the *CAMachineName*\_*CAName.req* file to a disk, and then follow the procedures for performing the offline request and certificate installation used for installing a new subordinate CA certificate.

### Verifying the Inclusion of All Issuance Policies

Once the CA certificate is installed, the Enterprise CA’s certificate should be inspected to ensure that the Issuance policies are included in the certificate.

1. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
2. In the console tree, right-click CAName (where *CAName* is the name of your CA), and then click Properties.
3. In the CAName Properties dialog box, on the General tab, click View Certificate.
4. On the General tab, ensure that the intended purposes list includes:

* High Assurance
* Medium Assurance
* Low Assurance

1. If any are missing, or appear as OIDs rather than text, review the %systemroot%\CAPolicy.inf file and ensure that the correct OIDs for the domain are included in the file. An incorrect OID will result in the OID being listed here, rather than being translated to the display names.
2. In the Certificate dialog box, click OK.
3. In the *CAName* Properties dialog box, click OK.
4. Close the Certification Authority console.

## Creating a Qualified Subordination Signing Certificate at a Stand-alone CA

A stand-alone CA does not use certificate templates. Because certificate templates are not supported by stand-alone CAs, the certificate request must inject the OID for qualified subordination during certificate enrollment. The following steps allow a qualified subordination certificate to be obtained at a stand-alone CA.

### Performing the Initial Certificate Request

To perform the initial request for a qualified subordination signing certificate

1. At the stand-alone CA, ensure that you are logged on as the Administrator of the CA computer.
2. Start Internet Explorer.
3. Open the [**http://localhost/certsrv**](http://localhost/certsrv) URL.
4. On the Welcome page, click Request a certificate.
5. On the Request a Certificate page, click advanced certificate request.
6. On the Advanced Certificate Request page, click Create and submit a request to this CA.
7. On the Advanced Certificate Request page, enter the following information in Identifying Information:

* Name: **Administrator** (or the name of the user account currently in use)
* Email: Leave blank
* Company: ***YourOrganization***(where *YourOrganization* is the name of your organization)
* Department: ***YourDepartment***(where *YourDepartment* is the name of your department)
* City: ***CityName***(where *CityName* is the name of your city)
* State: ***StateName***(where *StateName* is the name of your state or province)
* Country/Region: ***CountryName***(where *CountryName* is the name of your country or region)

1. In Type of Certificate Needed, select Other from the list.
2. In the OID box, type **1.3.6.1.4.1.311.10.3.10**.
3. In Key Options, set the following options:

* **Create a new key set**
* CSP: **Microsoft Enhanced Cryptographic Provider v1.0** (or a different CSP if you are using a hardware CSP).
* Key Usage: **Signature**
* **Automatic Key container name**
* Mark keys as exportable: **unchecked**
* Enable strong private key protection: **unchecked**
* Use local machine store: **unchecked**

1. In Additional Options, set the following options:

* Request format: **CMC**
* Hash Algorithm: **SHA-1**
* Save request to a file: **unchecked**
* Attributes: **empty**
* Friendly name: **Qualified Subordination Signing**

1. Review the entries, and then click Submit.  
   The Certificate Pending page appears.

### Issuing the Pending Certificate Request

The certificate request is set to a pending status by default on a stand-alone CA. To issue the certificate, a CA administrator must use the Certification Authority console.

1. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
2. In the console tree, click the Pending Requests container.
3. In the details pane, right-click the certificate request, point to All Tasks, and then click Issue.
4. Close the Certification Authority console.

#### Installing the Qualified Subordination Signing Certificate

The final step is to install the issued certificate.

1. In Internet Explorer, open the [**http://localhost/certsrv/**](http://localhost/certsrv/) URL.
2. On the Welcome page, click View the status of a pending certificate request.
3. On the View the status of a pending certificate request page, click the User-EKU (1.3.6.4.1.311.10.3.10) Certificate (*Date and Time*) link.
4. On the Certificate Issued page, click Install this certificate.  
   The Certificate Installed page should appear.
5. Close Internet Explorer.

#### Verifying the Qualified Subordination Signing Certificate

Once you have installed the qualified subordination certificate, you should view the certificate to ensure that it is correctly configured.

1. Open an empty MMC console.
2. On the File menu, click Add/Remove Snap-in.
3. In the Add/Remove Snap-in dialog box, click Add.
4. In the Add Standalone Snap-in dialog box, select Certificates, and then click Add.
5. In the Certificates snap-in dialog box, click My user account, and then click Finish.
6. In the Add Standalone Snap-in dialog box, click Close.
7. In the Add/Remove Snap-in dialog box, click OK.
8. In the console tree, expand **Certificates** – Current User, expand Personal, and then click Certificates.
9. In the details panel, double-click the certificate with the Template Friendly Name of “Qualified Subordination Signing”.
10. In the Certificate dialog box, on the General tab, ensure that the intended purposes only list Qualified Subordination.
11. In the Certificate dialog box, on the Details tab, ensure that the Key Usage indicates Digital Signature, Non-Repudiation (c0).
12. Click OK.
13. Close the console without saving changes.

## Creating a Qualified Subordination Signing Certificate at an Enterprise CA

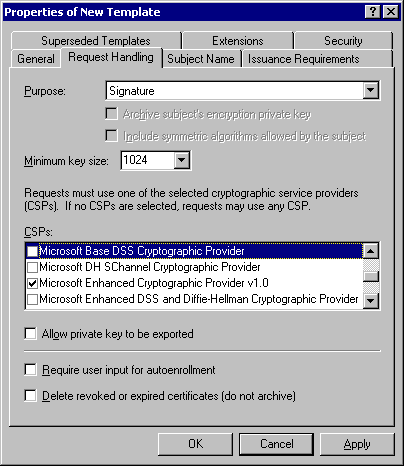
An Enterprise CA running on Windows Server 2003 Enterprise Edition can issue version 2.0 certificate templates. For qualified subordination, a version 2.0 template can be created to allow qualified subordination signing. If you are going to issue the Qualified Subordination Signing certificate from an Enterprise CA, the following steps will allow the creation and retrieval of a qualified subordination signing certificate at an enterprise CA.

**Note** While the goal is the same as the previous section on creating a qualified subordination signing certificate, the method is different when issuing the certificate from an Enterprise CA. The decision on which CA type to use should be based on what CAs are available in the infrastructure.

### Creating a Qualified Subordination version 2 Certificate Template

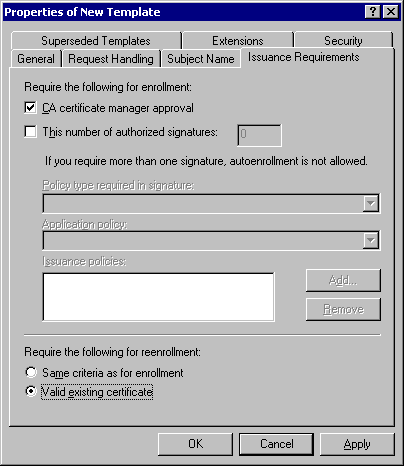
To create a version 2 certificate template for Qualified Subordination Signing

1. Open an empty MMC console.
2. On the File menu, click Add/Remove Snap-in.
3. In the Add/Remove Snap-in dialog box, click Add.
4. In the Add Standalone Snap-in dialog box, select Certificate Templates, and then click Add.
5. In the Add Standalone Snap-in dialog box, click Close.
6. In the Add/Remove Snap-in dialog box, click OK.
7. In the console tree, select Certificate Templates.
8. In the details pane, right-click Enrollment Agent, and then click Duplicate Template.
9. In the Properties of New Template dialog box, on the General tab, in the Template display namebox, type **Qualified Subordination Signing**.
10. On the Request Handling tab (Figure 34), make the following property changes:

Figure 34: Setting Request Handling Attributes

* Purpose: **Signature**
* CSPs: Clear the check boxes for **Microsoft Base Cryptographic Provider 1.0** and **Microsoft Base DSS Cryptographic Provider** and ensure that **Microsoft Enhanced Cryptographic Provider 1.0** is checked.

1. On the Subject Name tab, ensure that the Subject name format is set to Fully distinguished name and that only the User principal name (UPN) check box is selected.
2. On the Issuance Requirements tab (Figure 35), ensure that the CA certificate manager approval check box is selected. For Require the following for reenrollment, ensure that the Valid existing certificate option is set.

Figure 35: Configuring Issuance Requirement Settings

**Note** Depending on the security policy of your organization, you may choose to use different issuance settings. For example, you may choose to require one or more authorized signatures for certificate issuance, or that the same approval process be used for certificate reenrollment.

1. On the Extensions tab, in the Extensions included in this template box, select Application policies, and then click Edit.
2. In the Edit Application Policies Extension dialog box, in the Application policies box, select Certificate Request Agent, and then click Remove.
3. In the Edit Application Policies Extension dialog box, click Add.
4. In the Add Application Policy dialog box, in the Application Policies box, select Qualified Subordination, and then click OK.

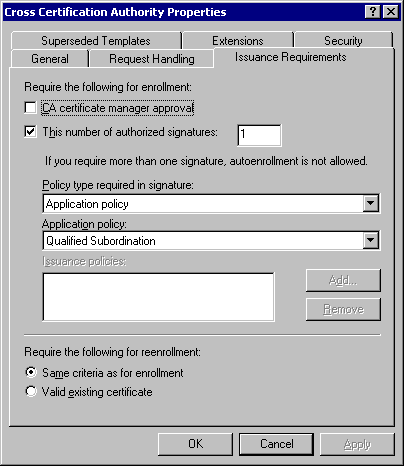
**Note** Alternatively, you could create a custom application policy with its own assigned OID. This will require a modification later in the process where you will configure the XXXX Certificate template to require your custom application policy for a signature.

1. In the Edit Application Policies Extension dialog box, ensure that Qualified Subordination is the only policy that appears in the Application policies dialog box, and then click OK.
2. On the Security tab, ensure that only Domain Admins and Enterprise Admins have the Enroll permission for the certificate template. Alternatively, you can create a custom security group and only assign that security group the Enroll permission.
3. Click OKto apply all configuration changes to the certificate template.

### Modifying the Cross-Certification Authority version 2 Certificate Template

Once the qualified subordination signing template is created, the cross-certificate authority certificate template must be modified.

1. In the Certificate Templates console, in the details pane, double-click the Cross Certification Authority certificate template.
2. In the Cross-Certification Authority Propertiesdialog box, on the Issuance Requirements tab (Figure 36), make the following changes:

Figure 36: Modifying the Issuance Requirements for the Cross CA Certificate

* **This number of authorized signatures** A request can be configured to require more than one signature for issuance. You should require at least one signature.
* **Application Policy** If a custom application policy was created, change the default application policy from Qualified Subordination to the logical name you created for your custom application policy.

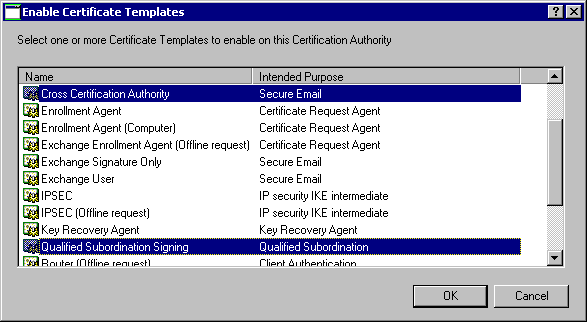
1. Click OK.
2. Close the Certificate Templatesconsole

**Important** By default, the CrossCA template is defined to publish to Active Directory when issued. Ensure that the certificate is formatted and configured properly before issuing. Invalid or unintended constraints may cause unwanted behavior in the environment and pose security risks, if not properly prepared.

### Configuring Windows Server 2003 Enterprise Edition to Issue the Certificate Templates

Since only Windows Server 2003 Enterprise Edition supports version 2 certificate templates, it is required for the certificate deployment. The Windows Server 2003 Enterprise Edition must be configured to issue both the Qualified Subordination Signing and cross-certification authority certificate templates.

1. Log on as a CA Administrator at Windows Server 2003 Enterprise Edition running Certificate Services configured as an Enterprise CA.
2. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
3. In the console tree, expand CAName (where *CAName* is the logical name assigned to your CA).
4. In the console tree, right-click Certificate Templates, point to New, and then click Certificate Template to Issue.
5. In the Enable Certificate Templates dialog box (Figure 37), in the list of available templates, click Cross-Certification Authority, control-click Qualified Subordination Signing, and then click OK.

Figure 37: Loading the Cross-Certification Authority and Qualified Subordination Signing Certificates

1. In the details pane, ensure that the Cross-Certification Authority and Qualified Subordination Signing certificate appear.
2. Close the Certification Authority console.

### Acquiring the Qualified Subordination Signing Certificate

Although you typically request certificates from an Enterprise CA using the Certificates MMC console, the requirement for CA Certificate manager approval in the Qualified Subordination Signing certificate template makes the use of the Certificate Services Web enrollment pages the better method to acquire the Qualified Subordination Signing certificate.

#### Performing the Initial Certificate Request

To perform the initial request for a qualified subordination signing certificate

1. Ensure that you are logged on as a user with the Enroll permissions for the Qualified Subordination Signing Certificate template.
2. Start Internet Explorer.
3. Open the **http://*CAMachineName*/certsrv** URL (where *CAMachineName* is the name of the Windows Server 2003 Enterprise Edition configured to issue the Qualified Subordination Signing Certificates).
4. On the Welcome page, click Request a certificate.
5. On the Request a Certificate page, click advanced certificate request.
6. On the Advanced Certificate Request page, click Create and submit a request to this CA.
7. On the Advanced Certificate Request page, in the Certificate Template section, select Qualified Subordination Signing from the list.
8. In the Key Options section, set the following options:

* **Create a new key set**
* CSP: **Microsoft Enhanced Cryptographic Provider v1.0** (or a different CSP if you are using a hardware CSP).
* Key Usage: **Signature**
* **Automatic Key container name**
* Mark keys as exportable: **unchecked**
* Enable strong private key protection: **unchecked**
* Use local machine store: **unchecked**

1. In the Additional Options section, set the following options:

* Request format: **CMC**
* Hash Algorithm: **SHA-1**
* Save request to a file: **unchecked**
* Attributes: **empty**
* Friendly name: **Qualified Subordination Signing**

1. Review the entries, and then click Submit.  
   The Certificate Pending page appears.

Issuing the Pending Certificate Request

The certificate request is set to a pending status by default on a stand-alone CA. To issue the certificate, a CA administrator must use the Certification Authority console.

1. Ensure that you are logged on to the Windows Server 2003 Enterprise Edition as a CA Certificate Manager.
2. On the Start menu, point to Programs, point to Administrative Tools, and then click Certification Authority.
3. In the console tree, click the Pending Requests container.
4. In the details pane, right-click the certificate request for a Qualified Subordination Signing certificate, point to All Tasks, and then click Issue.
5. Close the Certification Authority console.

#### Installing the Qualified Subordination Signing Certificate

The final step is to install the issued certificate.

1. At the computer where the original certificate request was performed, log on as the user who made the Qualified Subordination Signing certificate request.
2. In Internet Explorer, open the **http://*CAMachineName*/certsrv/** URL (where *CAMachineName* is the name of the Windows Server 2003 Enterprise Edition configured to issue the Qualified Subordination Signing Certificates).
3. On the Welcome page, click View the status of a pending certificate request.
4. On the View the status of a pending certificate request page, click the QualifiedSubordinationSigning Certificate (*Date and Time*) link.
5. On the Certificate Issued page, click Install this certificate.  
   The **Certificate Installed** page should appear.
6. Close Internet Explorer.

#### Verifying the Qualified Subordination Signing Certificate

Once you have installed the qualified subordination certificate, you should view the certificate to ensure that it is correctly configured.

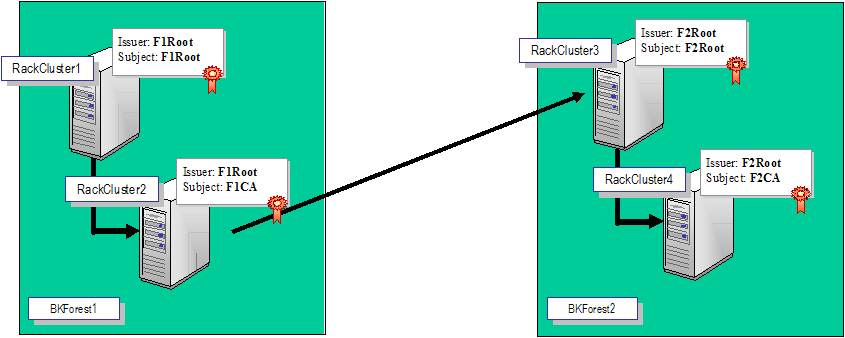
1. Open an empty MMC console.
2. On the File menu, click Add/Remove Snap-in.
3. In the Add/Remove Snap-in dialog box, click Add.
4. In the Add Standalone Snap-in dialog box, select Certificates, and then click Add.
5. In the Certificates snap-in dialog box, click My user account, and then click Finish.
6. In the Add Standalone Snap-in dialog box, click Close.
7. In the Add/Remove Snap-in dialog box, click OK.
8. In the console tree, expand Certificates – Current User, expand Personal, and then click Certificates.
9. In the details panel, double-click the certificate with the template Friendly Name of “Qualified Subordination Signing”.
10. In the Certificate dialog box, on the General tab, ensure that the intended purposes only list Qualified Subordination.

**Note** If you used a custom application policy, then the name of this application policy and its OID will appear instead of Qualified Subordination.

1. In the Certificate dialog box, on the Details tab, ensure that the Key Usage indicates Digital Signature (80).
2. Click OK.
3. Close the console without saving the changes.

## Performing Cross-Certification

Once the qualified subordination signing certificate is successfully installed, you can now request the Cross-Certification Authority certificate from the other CA. In this example, the following CAs shown in Figure 38 will be used.

Figure 38: Performing the Cross-Certification

### Gathering the Necessary Files

The first cross-certification request is to be run from the F1CA certification authority. To perform the qualified subordination, the following is required at the F1CA before starting the process:

* The CA certificate from F2Root.
* The Policy.inf file configured at the F1CA defining the naming constraints, policy constraints, and application constraints defined for the qualified subordination.
* The user performing the qualified subordination must have a qualified subordination signing certificate issued by F1CA.

### Generating the Cross-Certification Authority Certificate Request

Once these items are obtained, the following procedure is used to obtain a Cross-Certification Authority certificate for F2Root from F1CA. The request is performed using Certreq.exe, a command-line tool used from the Windows Server 2003 administrative tools pack that requests certificates from the command line.

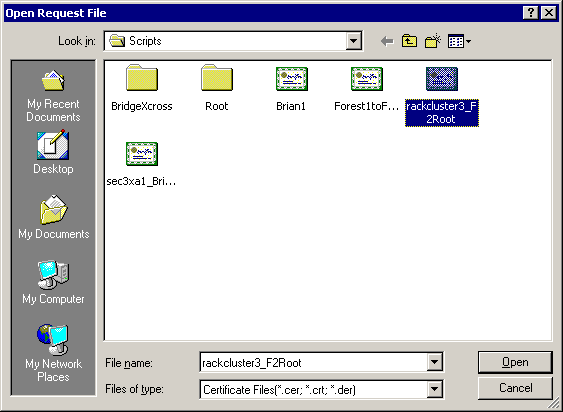
**Important** The constraints defined in Policy.inf are enforced at the signing CA where you generate the request, not at the CA whose CA certificate you use during the request. It is also important to generate the cross-certificate request in the domain in which the cross-cert will be issued.

1. Copy the F2Root certificate and the Policy.inf file into a folder on the F1CA computer.
2. Open a command prompt window.
3. Make the folder containing the F2Root certificate and the Policy.inf file the current directory.
4. Run certreq.exe –policy. This command constructs a cross-certification or qualified subordination request from an existing CA certificate or from an existing request.

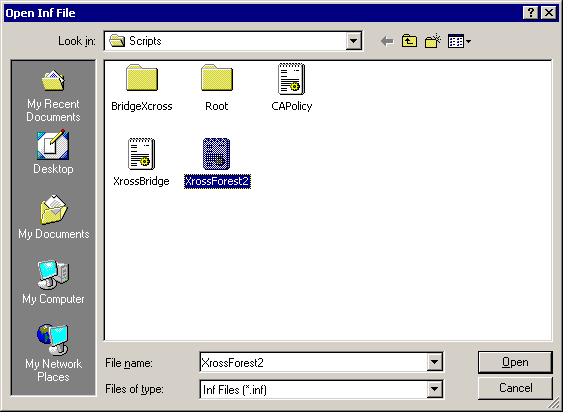
**Note** If creating a cross-certificate request from a stand-alone CA to an Enterprise CA, the following command must be specified to include the proper CrossCA template name in the template. Example:

certreq –policy –attrib “CertificateTemplate:CrossCA”  <CatoXcertify> <inf file> <CMCoutFile>

1. In the Open Request File dialog box (Figure 39), in the Files of Type box, select Certificate Files (\*.cer, \*.crt, \*.der), select the requesting CA’s certificate (RackCluster3\_F2Root in this example), and then click Open.

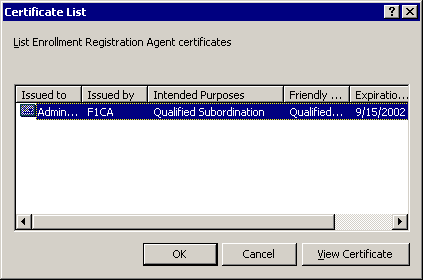
Figure 39: Selecting the Target CA’s Certificate

1. In the Open Inf File dialog box (Figure 40), select the configured Policy.inf file, and then click Open.

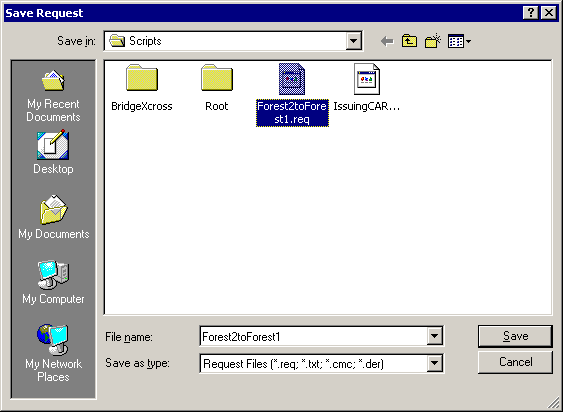
Figure 40: Selecting the Policy.inf File

**Note** You do not have to name the file Policy.inf. In this example, the file was named xrossForest2.inf to aid in determining which Policy.inf file was required.

1. In the Certificate List dialog box (Figure 41), select the Qualified Subordination certificate that you requested earlier, and then click OK.

Figure 41: Selecting the Qualified Subordination Signing Certificate

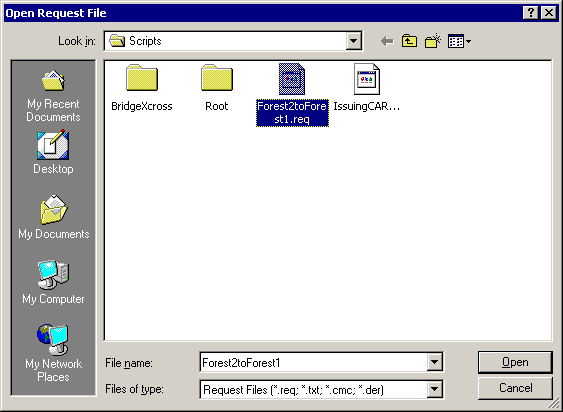
1. In the Save Request dialog box (Figure 42), in the File name box, type a name describing the Cross-Certification Authority certificate request, and then click Save.

Figure 42: Saving the Request File

### Processing the Cross-Certification Authority Certificate Request

At the F1CA, the final step is to process the Cross-Certification Authority certificate request file.

1. In Administrative Tools, open Certification Authority.
2. In the console tree, right-click CAName (where *CAName* is the name of your CA), point to All Tasks, and then click Submit New Request.
3. In the Open Request File dialog box (Figure 43), select the request file that you created in the previous process, and then click Save.

Figure 43: Choosing the Request File

1. In the Save Certificate dialog box, indicate the name of the Certificate File, and then click Save.  
   The certificate file is automatically published into Active Directory, so that the clients in Forest1 can validate certificates issued by the CA’s in Forest2.

### Verifying the Existence of the Cross Certificate

The final step is verifying that the certificate has been successfully published into Active Directory. This process uses Certutil.exe to verify the existence of the cross certificate. The following walkthrough validates the certificate with subject name ASIA SA Root CA that was issued by the Microsoft Intranet CA.

1. Open a command prompt window.
2. Type **certutil -viewstore "CN=<*CAName>*,CN=AIA,CN=Public Key Services,CN=Services,CN=Configuration,DC=*<ForestRootDN>*?crossCertificatePair**

Where:

* *<CAName>* The name of the CA that is issued the cross-certification authority certification. This is the subject of the cross-certification authority certificate.
* *<ForestRootDN>* The name of the forest root LDAP distinguished name. This is the forest in which the cross-certification authority certificate exists.

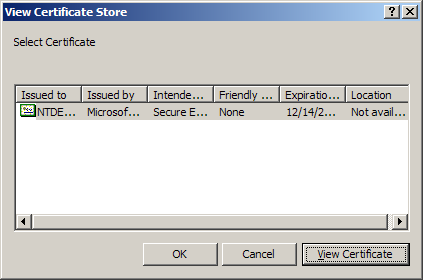
In this example, the following command was used:

certutil -viewstore "CN=ASIA SA Root CA,CN=AIA,CN=Public Key Services,CN=Services,CN=Configuration,DC=corp,DC=microsoft,DC=com?crossCertificatePair

**Note** If the subject name of the CA being cross-certified does not have CN= in the subject name, the Windows Server 2003 CA will generate a hash of the binary name of the CA and publish the cross certificate to that name. For example, to perform the command in the previous step in this case, the syntax would be similar to the following where the common name is the hash of the CA name:

certutil -viewstore CN=12d2cb85c5e62f9aa7591e2ddaa44c987de5abbc,CN=AIA,CN=Public Key Services,CN=Services,CN=Configuration,DC=ntdev,DC=corp,DC=microsoft,DC=com

1. In the View Certificate Store window (Figure 44), all cross-certification authority certificates with the <CAName> in the subject name are listed.

Figure 44: The Cross-Certification Authority Certificates List

**Note** Multiple cross-certification authority certificates can exist when the cross-certification authority certificate is renewed when the previous certificate expires.

1. In the View Certificate Store dialog box, select the cross-certification authority certificate that you wish to view, and then click View Certificate.
2. In the Certificate dialog box, click the Certification Path tab.
3. On the Certification Path tab (Figure 45), ensure that the certification path is the correct path. In this example, the chain is correct, as it shows the ASIA SA Root CA chaining to the Microsoft Corporate Root Authority.

Figure 45: Validating the Certification Path of the Cross-Certification Authority Certificate

1. In the Certificate dialog box, click OK.
2. In the View Certificate Store dialog box, click OK.
3. Close the command prompt window.

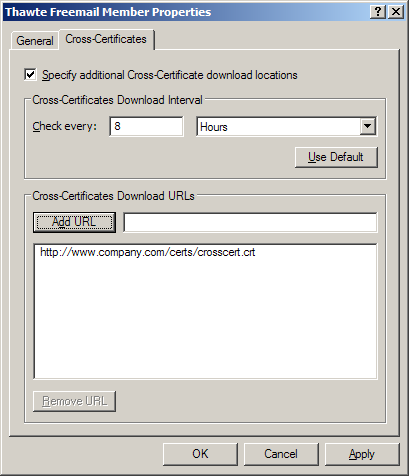
When cross-certifying with some third-party certification authorities, you may have to specify where to retrieve additional cross certificates associated with the issuer of a presented certificate. Typically, a certificate will have a cross-certificate URL encoded in the certificate, but in some cases, you may need to specify additional URLs.

To add the additional URLs

1. Open a blank MMC console.
2. On the File menu, click the Add/Remove snap-in.
3. In the Add/Remove snap-in dialog box, click Add.
4. In the Add Standalone Snap-in dialog box, select Certificates, and then click Add.
5. In the Certificates snap-in dialog box, select either My user account, Service account, or computer account, based on whether the certificate that must chain using a cross certificate is issued to the computer, a service, or to your user account, and then click Finish.

**Note** If you are not an Administrator of the local computer, you will automatically load the Certificates console focused on My user account.

1. In the console tree, expand Certificates – Current User, expand Personal, and then click Certificates.
2. In the details pane, right-click the certificate that must be cross-certified, and then click Properties.
3. In the Certificate Properties dialog box, click the Cross-Certificates tab.
4. In the Cross-Certificates tab (Figure 46), select the Specify additional Cross-Certificate download locations check box. In addition, you can now do the following:

Figure 46: Defining Manual Cross-Certificate Download Locations

* Define the Cross-Certificate download interval. The default is every eight hours.
* Add URLs for Cross-Certificate download locations. The URL can be an LDAP, HTTP, FTP, or FILE location.

### Completing Cross-Certification

The cross-certification must be performed in the opposite direction to ensure that all certificates issued in both organizations are trusted. This requires that the same process be performed at F2CA using the F1Root certificate.

### Completing the Cross-Certification When a Bridge CA Is Implemented

An additional process must be performed when a Bridge CA is used. The certificates that are issued by the BridgeCA must be published in all forests that are connected to the BridgeCA to allow all certificates to be recognized by the organizations using the bridge.

1. From the BridgeCA, copy all issued CrossCA certificates to a common share point.
2. At the first forest connected to the bridge, run **Certutil –dspublish –f *cert1.crt* CrossCA**(where *cert1.crt* is the first crossCA cert).
3. Repeat the process for all certificates issued by the Cross CA, at all forests connected to the Bridge CA.

# Appendix A – Policy.Inf

[Version]

Signature= "$Windows NT$"

; ===========================================================

; Request Attributes

; top level section

; ===========================================================

[RequestAttributes]

CertificateTemplate = CrossCA

AttributeName1 = AttributeValue1

AttributeName2 = AttributeValue2

; ===========================================================

; NameConstraintsExcluded Name Constraints Extension

; szOID\_NAME\_CONSTRAINTS 2.5.29.30

; top level section

; ===========================================================

[NameConstraintsExtension]

Include = NameConstraintsPermitted

Exclude = NameConstraintsExcluded

Critical = TrUe

[NameConstraintsPermitted]

; list of user defined permitted DNS names

; the numeric second and third arguments are optional

; when present, the second argument is the minimum depth

; when present, the third argument is the maximum depth

; NOTE: Crypto APIs fail to process cert chains when the minimum or maximum

; depth is specified!

DNS = foo@domain.com

DNS = domain1.domain.com

email=me@you.com

UPN=user@domain.com

; the first is an IP address, the second is an IP address mask

IPADDRESS=255.255.18.172,255.255.255.0

ipaddress=::255.255.18.172,::255.255.255.0

ipaddress=1234:5678:9abc:def0:3210:7654:ba98:fedc,1234:5678:9abc:def0:3210:7654:ba98:fedc

ipaddress=::5678:9abc:def0:3210:7654:ba98:fedc,1234:5678:9abc:def0:3210:7654:ba98:fedc

ipaddress=1234::def0:3210:7654:ba98:fedc,1234:5678:9abc:def0:3210:7654:ba98:fedc

ipaddress=1234:5678:9abc:def0:3210:7654:ba98::,1234:5678:9abc:def0:3210:7654:ba98:fedc

ipaddress=1234:5678:9abc:def0:3210:7654::,1234:5678:9abc:def0:3210:7654:ba98:fedc

url=http://localhost/certsrv/default.html

url=file://\\localhost\certsrv\default.html

DIRECTORYNAME = "cn=mycn,ou=myou,s=mystate,c=us"

[NameConstraintsExcluded]

; list of user defined excluded DNS names

DNS = domain.com

; ===========================================================

; Policy (CPS) Extension

; szOID\_CERT\_POLICIES 2.5.29.32

; top level section

; ===========================================================

[PolicyStatementExtension]

; list of user defined policies

Policies = LegalPolicy, LimitedUsePolicy, ExtraPolicy, OIDPolicy

CRITICAL = FALSE

[LegalPolicy]

; each policy has one OID, and zero or more Notice and URL keys

OID = 1.3.6.1.4.1.311.21.43

Notice = "Legal policy statement text"

[LimitedUsePolicy]

OID = 1.3.6.1.4.1.311.21.47

URL = "http://http.site.com/some where/default.asp"

URL = "ftp://ftp.site.com/some where else/default.asp"

Notice = "Limited use policy statement text."

URL = "ldap://ldap.site.com/some where else again/default.asp"

[ExtraPolicy]

OID = 1.3.6.1.4.1.311.21.53

URL = http://extra.site.com/Extra Policy/default.asp

[oidpolicy]

OID = 1.3.6.1.4.1.311.21.55

; ===========================================================

; Policy Mapping Extension

; szOID\_POLICY\_MAPPINGS 2.5.29.33

; top level section

; ===========================================================

[PolicyMappingsExtension]

; list of user defined policy mappings

; first OID is Issuer Domain Policy OID, second is Subject Domain Policy OID

; each entry maps one foreign policy OID to local

1.3.6.1.4.1.311.21.53 = 1.2.3.4.87

1.3.6.1.4.1.311.21.54 = 1.2.3.4.89

critical = yEs

; ===========================================================

; Policy Constraints Extension

; szOID\_POLICY\_CONSTRAINTS 2.5.29.36

; top level section

; ===========================================================

[PolicyConstraintsExtension]

; consists of two optional DWORDs

; They refer to the depth of the CA hierarchy that requires explicit policy

; and inhibits Policy Mapping

RequireExplicitPolicy = 3

InhibitPolicyMapping = 5

; ===========================================================

; Application Policy (CPS) Extension

; szOID\_APPLICATION\_CERT\_POLICIES 1.3.6.1.4.1.311.21.10

; top level section

; ===========================================================

[ApplicationPolicyStatementExtension]

; list of user defined policies

Policies = AppLegalPolicy, AppLimitedUsePolicy, AppExtraPolicy, AppOIDPolicy

CRITICAL = FALSE

[AppLegalPolicy]

; each policy has one OID, and zero or more Notice and URL keys

OID = 1.3.6.1.4.1.311.21.54

Notice = "Application Legal policy statement text"

[AppLimitedUsePolicy]

OID = 1.3.6.1.4.1.311.21.58

URL = "http://http.site.com/application some where/default.asp"

URL = "ftp://ftp.site.com/application some where else/default.asp"

Notice = "Application Limited use policy statement text."

URL = "ldap://ldap.site.com/application some where else again/default.asp"

[AppExtraPolicy]

OID = 1.3.6.1.4.1.311.21.64

URL = http://extra.site.com/Application Extra Policy/default.asp

[Appoidpolicy]

OID = 1.3.6.1.4.1.311.21.66

; ===========================================================

; Application Policy Mapping Extension

; szOID\_APPLICATION\_POLICY\_MAPPINGS 1.3.6.1.4.1.311.21.11

; top level section

; ===========================================================

[ApplicationPolicyMappingsExtension]

; list of user defined application policy mappings

; first OID is Issuer Domain Policy OID, second is Subject Domain Policy OID

; each entry maps one foreign policy OID to local

1.3.6.1.4.1.311.21.64 = 1.2.3.4.98

1.3.6.1.4.1.311.21.65 = 1.2.3.4.100

critical = trUE

; ===========================================================

; Application Policy Constraints Extension

; szOID\_APPLICATION\_POLICY\_CONSTRAINTS 1.3.6.1.4.1.311.21.12

; top level section

; ===========================================================

[ApplicationPolicyConstraintsExtension]

; consists of two optional DWORDs

; They refer to the depth of the CA hierarchy that requires explicit policy

; and inhibits Policy Mapping

RequireExplicitPolicy = 6

InhibitPolicyMapping = 10

; ===========================================================

; Basic Constraints Extension

; szOID\_BASIC\_CONSTRAINTS2 2.5.29.19

; top level section

; ===========================================================

[BasicConstraintsExtension]

; Subject Type is not supported always set to CA

; maximum subordinate CA path length

PathLength = 3

[EnhancedKeyUsageExtension]

;OID = 1.3.6.1.4.1.311.21.6 ; szOID\_KP\_KEY\_RECOVERY\_AGENT

;OID = 1.3.6.1.4.1.311.10.3.9 ; szOID\_ROOT\_LIST\_SIGNER

;OID = 1.3.6.1.4.1.311.10.3.1 ; szOID\_KP\_CTL\_USAGE\_SIGNING

; The following match the [ApplicationPolicyStatementExtension] section:

OID = 1.3.6.1.4.1.311.21.54

OID = 1.3.6.1.4.1.311.21.58

OID = 1.3.6.1.4.1.311.21.64

OID = 1.3.6.1.4.1.311.21.66

CriticAL = False

; ===========================================================

; Cross Certificate Distribution Points Extension

; szOID\_CROSS\_CERT\_DIST\_POINTS 1.3.6.1.4.1.311.10.9.1

; top level section

; ===========================================================

[CrossCertificateDistributionPointsExtension]

SyncDeltaTime = 24

URL = http://%1/Public/My CA.crt

URL = ftp://foo.com/Public/MyCA.crt

URL = file://\\%1\Public\My CA.crt

CriticAL = false

# Appendix B – CAPolicy.inf

[Version]

Signature= "$Windows NT$"

;[CAPolicy]

[PolicyStatementExtension]

Policies = LegalPolicy, LimitedUsePolicy, ExtraPolicy, OIDPolicy, EmptyPolicy

Critical = 0

[LegalPolicy]

OID = 1.3.6.1.4.1.311.21.43

Notice = "Legal policy statement text."

[LimitedUsePolicy]

OID = 1.3.6.1.4.1.311.21.47

URL = "http://http.site.com/some where/default.asp"

URL = "ftp://ftp.site.com/some where else/default.asp"

Notice = "Limited use policy statement text."

URL = "ldap://ldap.site.com/some where else again/default.asp"

[ExtraPolicy]

OID = 1.3.6.1.4.1.311.21.53

URL = http://extra.site.com/Extra Policy/default.asp

[oidpolicy]

OID = 1.3.6.1.4.1.311.21.55

[emptypolicy]

; For CRLDistributionPoint, AuthorityInformationAccess and

; CrossCertificateDistributionPointsExtension URLs:

;

; #define wszFCSAPARM\_SERVERDNSNAME L"%1"

; #define wszFCSAPARM\_SERVERSHORTNAME L"%2"

; #define wszFCSAPARM\_SANITIZEDCANAME L"%3"

; #define wszFCSAPARM\_CERTFILENAMESUFFIX L"%4"

; #define wszFCSAPARM\_DOMAINDN L"%5"

; #define wszFCSAPARM\_CONFIGDN L"%6"

; #define wszFCSAPARM\_SANITIZEDCANAMEHASH L"%7"

; #define wszFCSAPARM\_CRLFILENAMESUFFIX L"%8"

; #define wszFCSAPARM\_CRLDELTAFILENAMESUFFIX L"%9"

; #define wszFCSAPARM\_DSCRLATTRIBUTE L"%10"

; #define wszFCSAPARM\_DSCACERTATTRIBUTE L"%11"

; #define wszFCSAPARM\_DSUSERCERTATTRIBUTE L"%12"

; #define wszFCSAPARM\_DSKRACERTATTRIBUTE L"%13"

; #define wszFCSAPARM\_DSCROSSCERTPAIRATTRIBUTE L"%14"

;

; Setup APIs replace all %<number>% sequences with various directory paths.

; %3%8%9 in the first URL below presents two opportunities for string

; replacement with a directory path. To avoid this, use two percent signs

; to escape the setup API string replacement.

;

; URLs with spaces or commas must be quoted to avoid INF parsing problems

;

; default CDP registry URLs:

;

; D:\WINDOWS\System32\CertSrv\CertEnroll\%3%8%9.crl

; ldap:///CN=%7%8,CN=%2,CN=CDP,CN=Public Key Services,CN=Services,%6%10

; http://%1/CertEnroll/%3%8%9.crl

; file://\\%1\CertEnroll\%3%8%9.crl

[AuthorityInformationAccess]

URL = http://%1/Public/My CA.crt

URL = ftp://foo.com/Public/MyCA.crt

URL = file://\\%1\Public\My CA.crt

CriticAL = falSe

[CRLDistributionPoint]

URL = http://%1/Public/My CA.crl

URL = ftp://%1/Public/MyCA.crl

URL = file://\\%1\Public\My CA.crl

CriticAL = trUe

[CrossCertificateDistributionPointsExtension]

SyncDeltaTime = 600 ; in seconds

URL = http://%1/Public/My CCDP.crl

URL = ftp://%1/Public/MyCCDP.crl

URL = file://\\%1\Public\My CCDP.crl

CriticAL = yeS

[EnhancedKeyUsageExtension]

OID = 1.3.6.1.4.1.311.21.6 ; szOID\_KP\_KEY\_RECOVERY\_AGENT

OID = 1.3.6.1.4.1.311.10.3.9 ; szOID\_ROOT\_LIST\_SIGNER

OID = 1.3.6.1.4.1.311.10.3.1 ; szOID\_KP\_CTL\_USAGE\_SIGNING

CriticAL = False

[basicconstraintsextension]

pathlength = 13

criticaL=falsE

[certsrv\_server]

renewalkeylength=2048

RenewalValidityPeriodUnits=0x18

RenewalValidityPeriod=years

CRLPeriod = days

CRLPeriodUnits = 2

CRLDeltaPeriod = hours

CRLDeltaPeriodUnits = 4

# Appendix C – CMC Dump of a Qualified Subordination Request

PKCS7/CMS Message:

CMSG\_SIGNED(2)

CMSG\_SIGNED\_DATA\_CMS\_VERSION(3)

Content Type: 1.3.6.1.5.5.7.12.2 CMC Data

PKCS7 Message Content:

================ Begin Nesting Level 1 ================

CMS Certificate Request:

Tagged Attributes: 2

Body Part Id: 2

1.3.6.1.5.5.7.7.8 CMC Extensions

Value[0]:

Data Reference: 0

Cert Reference[0]: 1

Extensions: 7

2.5.29.36: Flags = 0, Length = 8

Policy Constraints

Required Explicit Policy Skip Certs=0

Inhibit Policy Mapping Skip Certs=0

2.5.29.32: Flags = 0, Length = 10

Certificate Policies

429.195.0: 0x80070002 (WIN32: 2): LDAPFlags

[1]Certificate Policy:

Policy Identifier=Corporate High Assurance

1.3.6.1.4.1.311.21.10: Flags = 0, Length = 6c

Application Policies

[1]Application Certificate Policy:

Policy Identifier=Client Authentication

[2]Application Certificate Policy:

Policy Identifier=Smart Card Logon

[3]Application Certificate Policy:

Policy Identifier=Corporate RAS

[4]Application Certificate Policy:

Policy Identifier=Private Key Archival

[5]Application Certificate Policy:

Policy Identifier=Key Recovery Agent

[6]Application Certificate Policy:

Policy Identifier=Encrypting File System

[7]Application Certificate Policy:

Policy Identifier=Secure Email

[8]Application Certificate Policy:

Policy Identifier=Certificate Request Agent

2.5.29.30: Flags = 0, Length = bf

Name Constraints

Permitted

[1]Subtrees (0..Max):

Other Name:

Principal Name=.asia.northwindtraders.com

[2]Subtrees (0..Max):

RFC822 Name=@northwindtraders.com

[3]Subtrees (0..Max):

RFC822 Name=.northwindtraders.com

[4]Subtrees (0..Max):

DNS Name=.asia.northwindtraders.com

[5]Subtrees (0..Max):

Directory Address:

DC=ASIA

DC=Northwindtraders

DC=com

[6]Subtrees (0..Max):

URL=

[7]Subtrees (0..Max):

IP Address=

Excluded=None

1.3.6.1.4.1.311.20.2: Flags = 0, Length = c

Certificate Template Name

SubCA

2.5.29.15: Flags = 0, Length = 4

Key Usage

Digital Signature, Certificate Signing, Off-line CRL Signing, CRL Signing (86)

2.5.29.19: Flags = 1(Critical), Length = 5

Basic Constraints

Subject Type=CA

Path Length Constraint=None

Body Part Id: 3

1.3.6.1.5.5.7.7.18 Reg Info

Value[0]:

CertificateTemplate: SubCA

Tagged Requests: 1

CMC\_TAGGED\_CERT\_REQUEST\_CHOICE:

Body Part Id: 1

================ Begin Nesting Level 2 ================

Element 0:

PKCS10 Certificate Request:

Version: 1

Subject:

CN=ITG ASIA Corp CA 1

DC=asia

DC=Northwindtraders

DC=com

Public Key Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.1 RSA

Algorithm Parameters:

05 00

Public Key Length: 2048 bits

Public Key: UnusedBits = 0

0000 30 82 01 0a 02 82 01 01 00 c4 92 eb 3d e3 70 52

0010 23 9f 9e a0 6c 9e 1e 26 43 7e 9c a3 d1 82 56 88

0020 5f df 2d a3 c6 f2 0a ae 25 8b 4e c8 7c 2b a2 4a

0030 72 49 ff 48 46 d9 59 6b 9e 1e 76 1a ff a9 1b 29

0040 30 4f a7 00 0f 73 3d 16 6b 4c 57 cd 2b c5 3d 78

0050 82 81 4a 90 26 b7 8b d4 b1 c4 08 ea 77 2a c2 f8

0060 8e e9 93 98 47 21 96 8e f9 9d ac bc 5f 01 f9 09

0070 12 b6 73 70 9a 2e 35 1c 51 d0 74 54 ee 46 7e 92

0080 03 5e d4 86 10 86 02 8b 8c 38 7e 76 10 55 0b 92

0090 1d 85 b9 46 d7 eb c2 42 3d a4 3d 84 d7 1f dd 93

00a0 30 ae 96 57 76 05 5d 2f 6e d0 7f 17 21 c2 87 1b

00b0 82 0c 02 da 10 87 48 ec c6 ba 45 45 75 22 3f 9a

00c0 f8 1f c6 10 05 08 01 d5 fa 56 25 a3 19 2c da e0

00d0 74 f6 43 9a c1 4d ed b6 9e 83 91 35 d0 c9 c3 6b

00e0 72 2f b0 3c fd 05 27 35 7b ea 8b 9d 48 83 96 59

00f0 bf b9 d3 80 b1 14 71 8e 75 e1 c9 da 69 86 4e cc

0100 9a 00 01 83 f5 0f 4b 2e 55 02 03 01 00 01

Request Attributes: 2

2 attributes:

Attribute[0]: 1.3.6.1.4.1.311.13.2.3 (OS Version)

Value[0][0]:

5.1.3541.2.

Attribute[1]: 1.2.840.113549.1.9.14 (Certificate Extensions)

Value[1][0]:

Unknown Attribute type

Certificate Extensions: 6

1.3.6.1.4.1.311.21.1: Flags = 0, Length = 3

CA Version

V1.0

1.3.6.1.4.1.311.21.2: Flags = 0, Length = 16

Previous CA Certificate Hash

d3 e5 cc ef 88 53 0d 13 b7 ae a2 7b 19 5f 57 5e 33 62 b0 ef

2.5.29.14: Flags = 0, Length = 16

Subject Key Identifier

d3 b6 48 1a c0 76 07 ba 35 2b 1c 90 8b bc 1f 2b d3 b9 4d f8

1.3.6.1.4.1.311.20.2: Flags = 0, Length = c

Certificate Template Name

SubCA

2.5.29.15: Flags = 0, Length = 4

Key Usage

Digital Signature, Certificate Signing, Off-line CRL Signing, CRL Signing (86)

2.5.29.19: Flags = 1(Critical), Length = 5

Basic Constraints

Subject Type=CA

Path Length Constraint=None

Signature Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.5 sha1RSA

Algorithm Parameters:

05 00

Signature: UnusedBits=0

0000 a1 4b fa 5b ec 6c 4d e8 e4 a4 2d 7f 4b b9 65 cb

0010 61 1e d5 85 49 e9 07 9c c5 8b 9b e2 a4 b5 18 26

0020 76 33 d9 3d 66 2e 32 2c e8 a5 23 65 9f 07 0a d6

0030 d3 d0 3f 1a ac ef 75 2d 53 0d 79 13 90 6f e3 61

0040 ab bb ad 31 e9 46 31 b4 78 33 3e 7d 4f d4 87 3c

0050 cf 75 0a 03 51 f0 f4 f3 15 25 b8 d8 5f bb d4 78

0060 5d a9 39 34 1d d7 f9 8a 8d 3c 4e 1f ea 79 23 ce

0070 85 42 8f 36 c2 24 d2 9f 90 37 93 80 d4 f2 76 74

0080 ad 65 a0 7e 83 fc 83 21 ea e5 d8 c9 5f 02 ea d0

0090 9f 50 96 3a b5 c3 7f 85 9b b8 fc cd 68 c5 27 c5

00a0 99 d6 5f df 8f 8b 82 7b 0f 21 f0 3d 9f 34 0c d8

00b0 ec 2b b1 a9 55 c7 01 2d 9e f0 89 76 d3 ed d6 33

00c0 55 8a 9a c7 8d 52 3e b1 5d da 35 61 28 f7 07 73

00d0 57 52 80 ac c0 31 ad 9e 81 49 01 1f 48 1f f4 95

00e0 9f 39 4b 1a be 33 c6 4b cf 67 04 5b aa 94 59 4d

00f0 d4 95 42 6f 25 d3 64 fc d6 c8 e7 ca e3 b3 0a ad

Signature matches Public Key

Key Id Hash(sha1): d3 b6 48 1a c0 76 07 ba 35 2b 1c 90 8b bc 1f 2b d3 b9 4d f8

---------------- End Nesting Level 2 ----------------

Tagged Content Info: 0

Tagged Other Messages: 0

---------------- End Nesting Level 1 ----------------

Signer Count: 2

Signer Info[0]:

NULL signature verifies

CMSG\_SIGNER\_INFO\_PKCS\_1\_5\_VERSION(1)

CERT\_ID\_ISSUER\_SERIAL\_NUMBER(1)

Serial Number: 00

Issuer: OID.1.3.6.1.4.1.311.21.9=Dummy Signer

Hash Algorithm:

Algorithm ObjectId: 1.3.14.3.2.26 sha1

Algorithm Parameters: NULL

Encrypted Hash Algorithm:

Algorithm ObjectId: 1.3.6.1.5.5.7.6.2 NO\_SIGN

Algorithm Parameters: NULL

Encrypted Hash:

0000 75 80 82 26 ff 11 77 5b 92 52 ce 2e a2 8e a2 32

0010 98 a7 1a a0

Authenticated Attributes[0]:

3 attributes:

Attribute[0]: 1.2.840.113549.1.9.3 (Content Type)

Value[0][0]:

Unknown Attribute type

1.3.6.1.5.5.7.12.2 CMC Data

Attribute[1]: 1.2.840.113549.1.9.4 (Message Digest)

Value[1][0]:

Unknown Attribute type

Message Digest:

5a 74 12 86 78 3a ab 5b 17 85 6f 4d 44 ea a2 74 2c 86 c1 1f

Attribute[2]: 1.3.6.1.4.1.311.21.20 (Client Information)

Value[2][0]:

Unknown Attribute type

Client Id: = 4

XECI\_CERTREQ -- 4

User: ASIA\user2

Machine: user2.asia.northwindtraders.com

Process: certreq

Unauthenticated Attributes[0]:

0 attributes:

Computed Hash: 75 80 82 26 ff 11 77 5b 92 52 ce 2e a2 8e a2 32 98 a7 1a a0

Signing Certificate Index: 1

-------- CERT\_CHAIN\_CONTEXT --------

ChainContext.dwInfoStatus = CERT\_TRUST\_HAS\_PREFERRED\_ISSUER (0x100)

CertContext.dwRevocationFreshnessTime: 27 Days, 4 Hours, 47 Minutes, 46 Seconds

SimpleChain.dwInfoStatus = CERT\_TRUST\_HAS\_PREFERRED\_ISSUER (0x100)

SimpleChain.dwRevocationFreshnessTime: 27 Days, 4 Hours, 47 Minutes, 46 Seconds

CertContext[0][0]: dwInfoStatus=101 dwErrorStatus=0

Issuer: CN=W2K User Enrollment CA, OU=Test, O=NT Distributed Systems, L=Redmond, S=WA, C=US, E=user1@northwindtraders.com

Subject: CN=Darren Canavor

Serial: 61321aaf00000000061e

Template: EnrollmentAgent

54 8e 6f e3 d6 88 31 29 b6 f0 fb be bd aa 91 12 76 dd 51 a3

Element.dwInfoStatus = CERT\_TRUST\_HAS\_EXACT\_MATCH\_ISSUER (0x1)

Element.dwInfoStatus = CERT\_TRUST\_HAS\_PREFERRED\_ISSUER (0x100)

419.2199.0: 0x80070002 (WIN32: 2)

CRL 0:

Issuer: CN=W2K User Enrollment CA, OU=Test, O=NT Distributed Systems, L=Redmond, S=WA, C=US, E=user1@northwindtraders.com

4b f3 03 53 8b d0 82 f5 bf aa b7 d1 ee fc aa 3e 12 74 b5 fe

Application[0] = 1.3.6.1.4.1.311.20.2.1 Certificate Request Agent

CertContext[0][1]: dwInfoStatus=101 dwErrorStatus=0

Issuer: E=user1@northwindtraders.com, CN=ASIA W2K PCA, OU=Test, O=NT Distributed Systems, L=Redmond, S=WA, C=US

Subject: CN=W2K User Enrollment CA, OU=Test, O=NT Distributed Systems, L=Redmond, S=WA, C=US, E=user1@northwindtraders.com

Serial: 4860d04700020000bc47

Template: SubCA

fa 4b b5 e1 a6 9f 8d e9 1d 69 4b f4 42 9f 76 0b ef a9 c8 d9

Element.dwInfoStatus = CERT\_TRUST\_HAS\_EXACT\_MATCH\_ISSUER (0x1)

Element.dwInfoStatus = CERT\_TRUST\_HAS\_PREFERRED\_ISSUER (0x100)

419.2199.0: 0x80070002 (WIN32: 2)

CRL 0:

Issuer: E=user1@northwindtraders.com, CN=ASIA W2K PCA, OU=Test, O=NT Distributed Systems, L=Redmond, S=WA, C=US

a7 a0 41 a8 90 71 0f 02 60 b1 28 bf 47 3b 4e 48 20 24 58 74

CertContext[0][2]: dwInfoStatus=102 dwErrorStatus=0

Issuer: CN=ASIA SA Root CA, OU=Asia, O=Northwindtraders, C=US

Subject: E=user1@northwindtraders.com, CN=ASIA W2K PCA, OU=Test, O=NT Distributed Systems, L=Redmond, S=WA, C=US

Serial: 1f54dfa100000000001d

Template: SubCA

61 0b 95 b6 06 ba 14 4c ae 89 24 9d 83 fd 06 49 9b ca 82 60

Element.dwInfoStatus = CERT\_TRUST\_HAS\_KEY\_MATCH\_ISSUER (0x2)

Element.dwInfoStatus = CERT\_TRUST\_HAS\_PREFERRED\_ISSUER (0x100)

CRL 49:

Issuer: CN=ASIA SA Root CA, OU=Asia, O=Northwindtraders, C=US

78 9e e3 0f 30 ed 2f d5 6e ec b1 9b 59 93 9b b9 b3 36 bb 8e

CertContext[0][3]: dwInfoStatus=10c dwErrorStatus=0

Issuer: CN=ASIA SA Root CA, OU=Asia, O=Northwindtraders, C=US

Subject: CN=ASIA SA Root CA, OU=Asia, O=Northwindtraders, C=US

Serial: 212566f75e7584b8478f7b59b4a9e212

Template: CA

9e 90 bb 26 24 e4 da dc 63 11 b8 18 2d af ad 39 56 81 66 51

Element.dwInfoStatus = CERT\_TRUST\_HAS\_NAME\_MATCH\_ISSUER (0x4)

Element.dwInfoStatus = CERT\_TRUST\_IS\_SELF\_SIGNED (0x8)

Element.dwInfoStatus = CERT\_TRUST\_HAS\_PREFERRED\_ISSUER (0x100)

CRL 49:

Issuer: CN=ASIA SA Root CA, OU=Asia, O=Northwindtraders, C=US

78 9e e3 0f 30 ed 2f d5 6e ec b1 9b 59 93 9b b9 b3 36 bb 8e

Exclude leaf cert:

ea 88 66 a8 e8 c0 2d 50 c6 b0 21 a8 4d fb 87 2d 0b 8a da 83

Full chain:

7c c0 2c 86 ba 49 40 95 45 4c 0c 7f e1 f7 07 d3 88 f1 8d d4

------------------------------------

Verified Issuance Policies: None

Verified Application Policies:

1.3.6.1.4.1.311.20.2.1 Certificate Request Agent

Signer Info[1]:

Signature matches Public Key

CMSG\_SIGNER\_INFO\_PKCS\_1\_5\_VERSION(1)

CERT\_ID\_ISSUER\_SERIAL\_NUMBER(1)

Serial Number: 61321aaf00000000061e

Issuer: CN=W2K User Enrollment CA, OU=Test, O=NT Distributed Systems, L=Redmond, S=WA, C=US, E=user1@northwindtraders.com

Subject: CN=Darren Canavor

Hash Algorithm:

Algorithm ObjectId: 1.3.14.3.2.26 sha1

Algorithm Parameters: NULL

Encrypted Hash Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.1 RSA

Algorithm Parameters: NULL

Encrypted Hash:

0000 91 d8 40 e2 fc d7 9f dd da 2a 16 ed 6e b4 62 39

0010 08 c9 0b 08 c6 7a 19 8e f4 7a af ee a0 c8 e5 5a

0020 54 90 d3 bb f8 89 cf e8 3f e4 7a 33 45 1e 6b 09

0030 29 a2 4a 3d e0 28 fe d8 45 15 59 67 74 f4 ab 03

0040 82 d8 89 11 e6 bd 1a 5f 3b 73 02 a4 f8 be 9f f9

0050 d2 65 cc 2a b1 47 d4 d1 ce 8f 1d 51 be 5e 5b 92

0060 a7 79 da 80 4e 5e e5 72 3c 76 84 61 34 d4 42 f2

0070 da 4d 4b 17 ec 34 53 9b 2c 86 71 60 82 47 54 1e

Authenticated Attributes[1]:

3 attributes:

Attribute[0]: 1.2.840.113549.1.9.3 (Content Type)

Value[0][0]:

Unknown Attribute type

1.3.6.1.5.5.7.12.2 CMC Data

Attribute[1]: 1.2.840.113549.1.9.4 (Message Digest)

Value[1][0]:

Unknown Attribute type

Message Digest:

5a 74 12 86 78 3a ab 5b 17 85 6f 4d 44 ea a2 74 2c 86 c1 1f

Attribute[2]: 1.3.6.1.4.1.311.21.20 (Client Information)

Value[2][0]:

Unknown Attribute type

Client Id: = 4

XECI\_CERTREQ -- 4

User: ASIA\user2

Machine: user2.asia.northwindtraders.com

Process: certreq

Unauthenticated Attributes[1]:

0 attributes:

Computed Hash: 75 80 82 26 ff 11 77 5b 92 52 ce 2e a2 8e a2 32 98 a7 1a a0

No Recipient

Certificates:

================ Begin Nesting Level 1 ================

Element 0:

X509 Certificate:

Version: 3

Serial Number: 4860d04700020000bc47

Signature Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.5 sha1RSA

Algorithm Parameters:

05 00

Issuer:

E=user1@northwindtraders.com

CN=ASIA W2K PCA

OU=Test

O=NT Distributed Systems

L=Redmond

S=WA

C=US

NotBefore: 6/26/2001 1:03 PM

NotAfter: 6/7/2002 11:41 AM

Subject:

CN=W2K User Enrollment CA

OU=Test

O=NT Distributed Systems

L=Redmond

S=WA

C=US

E=user1@northwindtraders.com

Public Key Algorithm:

Algorithm ObjectId: 1.2.840.10040.4.1 DSA

Algorithm Parameters:

0000 30 82 01 1e 02 81 81 00 a1 13 9a 69 07 1f 39 de

0010 e8 f2 6f 24 0b 80 91 67 a1 f5 b4 d0 e5 24 65 3c

0020 2e f1 48 0c ef 3e e9 5d 1c fe 9a 47 1b 3e d3 41

0030 ba c4 c5 c1 6e 55 9c b2 c4 dc 0b 9a 1a a1 12 f6

0040 45 a5 32 f6 8d d3 50 79 e1 f0 1e 75 70 6b aa a8

0050 75 fb 99 bd 74 75 a8 b1 05 b0 a6 fd 4e 20 fa 9d

0060 9d 5e 79 51 9b 19 b0 c3 62 dd c2 a3 c1 ad af f6

0070 00 6b f7 70 3a 7e 22 e1 a9 d1 df 2f fc 53 d5 04

0080 95 b2 6c 7a 0c c4 52 5f 02 15 00 ee 57 07 f1 1f

0090 44 e7 75 0b d2 a0 f9 65 0a ec b8 dc 7c ad 6d 02

00a0 81 80 38 5d bc e4 06 1c 6c 16 70 54 7c 3a 65 d0

00b0 f3 bb 08 83 90 d0 b1 1b ea 53 90 23 8b b7 2e e2

00c0 a0 16 b7 11 41 31 20 f2 2c 56 a9 f3 8d 2b e8 74

00d0 32 c0 7e f4 90 a1 0f 30 c1 5e df e3 c7 a4 20 90

00e0 73 6a 02 bb eb 46 31 bd 29 70 45 e7 d7 43 22 86

00f0 55 33 e5 b9 d7 ac 4f 0b d4 53 5f ec 9c ae 34 0c

0100 14 35 7e 7f ad 0c 2d 50 4c ea 7d 47 34 1a 19 0b

0110 63 a3 1a 4a 3a 4d b8 4a 8a 7d b1 36 48 64 d0 f7

0120 e2 41

Public Key Length: 1024 bits

Public Key: UnusedBits = 0

0000 02 81 80 6f be 7e 8d a2 4c d2 4a c2 bb dc 71 9c

0010 85 d5 d3 0c e7 df d0 8e a5 85 2b d2 5b a4 a1 0d

0020 a8 55 1b d6 4b 04 2d 56 f8 0a a7 78 8b 1f d1 73

0030 b7 3e 2a af 1e 21 13 c6 4e 98 ce 88 4c 34 60 d1

0040 4c a4 80 4e 1c 76 ad 8e dd 60 6c f0 22 55 47 95

0050 09 3b 93 75 51 11 eb 7c 74 4a a1 72 2c cf d4 28

0060 ef 60 f0 8a 18 eb 4a 19 24 93 c0 27 3f af 55 98

0070 d6 1b 69 63 4a a6 7b f7 69 92 77 4c 28 60 f8 97

0080 6c a2 d0

Certificate Extensions: 7

2.5.29.14: Flags = 0, Length = 16

Subject Key Identifier

0a 2f e7 66 09 39 b6 c3 9d 96 9a 49 ff 75 73 fd 72 80 92 45

1.3.6.1.4.1.311.20.2: Flags = 0, Length = c

Certificate Template Name

SubCA

2.5.29.15: Flags = 0, Length = 4

Key Usage

Certificate Signing, Off-line CRL Signing, CRL Signing (06)

2.5.29.19: Flags = 1(Critical), Length = 5

Basic Constraints

Subject Type=CA

Path Length Constraint=None

2.5.29.35: Flags = 0, Length = 76

Authority Key Identifier

KeyID=7a fc 16 de 56 19 08 a3 39 21 5d 55 0f f2 57 be 8f 5e c7 7f

Certificate Issuer:

Directory Address:

CN=ASIA SA Root CA

OU=Asia

O=Northwindtraders

C=US

Certificate SerialNumber=1f 54 df a1 00 00 00 00 00 1d

2.5.29.31: Flags = 0, Length = 11c

CRL Distribution Points

[1]CRL Distribution Point

Distribution Point Name:

Full Name:

URL=ldap:///CN=ASIA%20W2K%20PCA,CN=W2KPCA,CN=CDP,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=asia,DC=Northwindtraders,DC=com?certificateRevocationList?base?objectclass=cRLDistributionPoint

[2]CRL Distribution Point

Distribution Point Name:

Full Name:

URL=http://w2kpca.asia.northwindtraders.com/CertEnroll/ASIA%20W2K%20PCA.crl

1.3.6.1.5.5.7.1.1: Flags = 0, Length = 130

Authority Information Access

[1]Authority Info Access

Access Method=Certification Authority Issuer (1.3.6.1.5.5.7.48.2)

Alternative Name:

URL=ldap:///CN=ASIA%20W2K%20PCA,CN=AIA,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=asia,DC=Northwindtraders,DC=com?cACertificate?base?objectclass=certificationAuthority

[2]Authority Info Access

Access Method=Certification Authority Issuer (1.3.6.1.5.5.7.48.2)

Alternative Name:

URL=http://w2kpca.asia.northwindtraders.com/CertEnroll/W2KPCA.asia.northwindtraders.com\_ASIA%20W2K%20PCA(2).crt

Signature Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.5 sha1RSA

Algorithm Parameters:

05 00

Signature: UnusedBits=0

0000 70 a2 0e c9 94 30 81 83 43 33 1b b3 6a 83 ae a5

0010 2a 1f fc c9 d8 28 ae 7e b3 a2 fb c2 f0 be 8a 3b

0020 31 e9 bc e2 80 1c 9d 5a 5c 08 db 79 df 7f 88 5e

0030 7c e2 90 fe 9b e5 83 20 b5 7b 2b 2c 86 bb 22 c4

0040 4e 9e 09 3c 6a c1 c0 35 57 85 82 a3 07 b4 ca e5

0050 d8 48 38 33 ec b2 f2 11 56 c4 83 2c 27 2e 9b 19

0060 7e 9c 62 5a 8f 10 78 16 3c 36 c3 19 fc 7f 63 38

0070 b8 eb ab 24 7a 45 86 ec 25 b0 b0 63 b0 2e 98 91

0080 54 13 6d c3 eb d7 c3 42 10 64 49 19 cf 20 d5 55

0090 18 bb a6 fd 5b ff 12 5a 88 62 22 12 5a 02 1e de

00a0 e2 21 e8 fa d2 83 6e 13 fd 55 3b ca 8a 56 27 8d

00b0 28 79 3c 15 df 58 79 4c ce fb d9 44 d4 fc 7f 6b

00c0 92 6b 67 3c e6 29 b2 ed 6a 30 0f 89 75 ab 9e 04

00d0 6b 31 ec e0 79 76 c3 51 cd 91 1e 13 cd 1e 06 8d

00e0 ce c8 c9 9b cb 14 23 88 ae e0 c3 1f 18 56 ae 55

00f0 a6 15 c5 95 18 61 5e 65 b6 24 9e c8 ca 87 fe 20

Non-root Certificate

Key Id Hash(sha1): 0a 2f e7 66 09 39 b6 c3 9d 96 9a 49 ff 75 73 fd 72 80 92 45

Cert Hash(md5): d4 8e c5 46 fa 21 77 2a 32 f0 8c 28 78 9a a2 92

Cert Hash(sha1): fa 4b b5 e1 a6 9f 8d e9 1d 69 4b f4 42 9f 76 0b ef a9 c8 d9

---------------- End Nesting Level 1 ----------------

================ Begin Nesting Level 1 ================

Element 1:

X509 Certificate:

Version: 3

Serial Number: 61321aaf00000000061e

Signature Algorithm:

Algorithm ObjectId: 1.2.840.10040.4.3 sha1DSA

Algorithm Parameters: NULL

Issuer:

CN=W2K User Enrollment CA

OU=Test

O=NT Distributed Systems

L=Redmond

S=WA

C=US

E=user1@northwindtraders.com

NotBefore: 10/29/2001 6:16 PM

NotAfter: 6/7/2002 11:41 AM

Subject:

CN=Darren Canavor

Public Key Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.1 RSA

Algorithm Parameters:

05 00

Public Key Length: 1024 bits

Public Key: UnusedBits = 0

0000 30 81 89 02 81 81 00 eb 8c 61 aa 06 26 3a 0f 74

0010 bb 03 53 94 8e 8b d7 c9 1c b8 af d9 71 12 3a 07

0020 96 fb 15 dd 1f ca d1 14 9e 47 ae aa 79 7d f0 ba

0030 71 3c 02 d7 1a 53 2e fe 56 23 be 64 ee f4 7f 7e

0040 02 68 22 ab f1 8d 94 f6 4f ee e2 45 f6 0e 5f 34

0050 dd c9 60 32 f0 fd 55 6b 4f 3d 5a 8d c3 21 97 ba

0060 a2 6b af 40 b1 ba 59 de 27 15 e4 e4 e3 2f 9f 84

0070 22 92 29 25 88 42 a5 c9 90 84 2e 46 86 32 21 99

0080 1f 52 98 5d 79 d7 eb 02 03 01 00 01

Certificate Extensions: 8

2.5.29.15: Flags = 0, Length = 4

Key Usage

Digital Signature (80)

2.5.29.14: Flags = 0, Length = 16

Subject Key Identifier

3e 09 89 24 b9 c6 f9 68 34 e8 02 1c f5 25 cd 96 6e 13 68 d0

1.3.6.1.4.1.311.20.2: Flags = 0, Length = 20

Certificate Template Name

EnrollmentAgent

2.5.29.35: Flags = 0, Length = ca

Authority Key Identifier

KeyID=0a 2f e7 66 09 39 b6 c3 9d 96 9a 49 ff 75 73 fd 72 80 92 45

Certificate Issuer:

Directory Address:

E=user1@northwindtraders.com

CN=ASIA W2K PCA

OU=Test

O=NT Distributed Systems

L=Redmond

S=WA

C=US

Certificate SerialNumber=48 60 d0 47 00 02 00 00 bc 47

2.5.29.31: Flags = 0, Length = 136

CRL Distribution Points

[1]CRL Distribution Point

Distribution Point Name:

Full Name:

URL=ldap:///CN=W2K%20User%20Enrollment%20CA,CN=w2keobca,CN=CDP,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=asia,DC=Northwindtraders,DC=com?certificateRevocationList?base?objectclass=cRLDistributionPoint

[2]CRL Distribution Point

Distribution Point Name:

Full Name:

URL=http://w2keobca.asia.northwindtraders.com/CertEnroll/W2K%20User%20Enrollment%20CA.crl

1.3.6.1.5.5.7.1.1: Flags = 0, Length = 147

Authority Information Access

[1]Authority Info Access

Access Method=Certification Authority Issuer (1.3.6.1.5.5.7.48.2)

Alternative Name:

URL=ldap:///CN=W2K%20User%20Enrollment%20CA,CN=AIA,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=asia,DC=Northwindtraders,DC=com?cACertificate?base?objectclass=certificationAuthority

[2]Authority Info Access

Access Method=Certification Authority Issuer (1.3.6.1.5.5.7.48.2)

Alternative Name:

URL=http://w2keobca.asia.northwindtraders.com/CertEnroll/w2keobca.asia.northwindtraders.com\_W2K%20User%20Enrollment%20CA.crt

2.5.29.37: Flags = 0, Length = e

Enhanced Key Usage

Certificate Request Agent (1.3.6.1.4.1.311.20.2.1)

2.5.29.17: Flags = 0, Length = 52

Subject Alternative Name

Other Name:

Principal Name=user2@asia.northwindtraders.com

Signature Algorithm:

Algorithm ObjectId: 1.2.840.10040.4.3 sha1DSA

Algorithm Parameters: NULL

Signature: UnusedBits=0

0000 c2 c8 2e 86 c9 32 ca 80 5c f3 ba 09 08 14 fc 01

0010 ab 87 1c 34 14 02 32 aa 7c af d0 36 8b df ac 24

0020 da 2c a6 7e 21 fc f8 49 da 80 00 15 02 2d 30

Non-root Certificate

Key Id Hash(sha1): 3e 09 89 24 b9 c6 f9 68 34 e8 02 1c f5 25 cd 96 6e 13 68 d0

Cert Hash(md5): 59 6d 1c 02 87 8f 91 08 cb 33 82 c1 d2 4a f8 1c

Cert Hash(sha1): 54 8e 6f e3 d6 88 31 29 b6 f0 fb be bd aa 91 12 76 dd 51 a3

---------------- End Nesting Level 1 ----------------

================ Begin Nesting Level 1 ================

Element 2:

X509 Certificate:

Version: 3

Serial Number: 1f54dfa100000000001d

Signature Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.5 sha1RSA

Algorithm Parameters:

05 00

Issuer:

CN=ASIA SA Root CA

OU=Asia

O=Northwindtraders

C=US

NotBefore: 6/7/2001 11:31 AM

NotAfter: 6/7/2002 11:41 AM

Subject:

E=user1@northwindtraders.com

CN=ASIA W2K PCA

OU=Test

O=NT Distributed Systems

L=Redmond

S=WA

C=US

Public Key Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.1 RSA

Algorithm Parameters:

05 00

Public Key Length: 2048 bits

Public Key: UnusedBits = 0

0000 30 82 01 0a 02 82 01 01 00 c2 0b 38 8e 77 00 c5

0010 ec 23 62 2f 1f 5a bd f9 72 5f f4 71 25 87 38 ca

0020 1d 8c 1c 07 cb 02 4e 12 43 7a d9 fe db 40 0f b1

0030 ca 98 72 b0 d9 fb 10 37 42 84 72 24 0f 26 36 f8

0040 ab a7 5b 44 a1 0f d7 18 f5 57 d6 e3 79 36 c5 cc

0050 bc af 47 ae b7 5e 0c 0f 3d 76 e7 06 84 af c1 2e

0060 99 90 e4 82 3f 20 d4 d6 bf 0b 37 9e 7f 31 e4 38

0070 b9 32 46 0b 94 0f b3 b9 55 cb f5 03 c9 c4 3b 1c

0080 6a 3b 78 77 06 71 f1 16 ed f9 a6 4b 94 35 00 a4

0090 9a 13 fe fc 7b 7a 8f cd 6e 9b 87 8d de 19 8e 06

00a0 88 ce b5 04 e4 fd 2a 50 a7 1e d5 7a d2 80 f1 e5

00b0 3f 08 2e 55 5e 05 57 97 0e d6 13 d8 6c 16 7d 5e

00c0 10 65 4e 2a 44 cc 5d f9 3d 52 9c d1 1e 15 e0 4d

00d0 a4 ec a1 0f 2f 5a e6 29 d5 4e 45 04 09 fc 45 0e

00e0 11 0f d0 fa d5 8d 0c 41 0d fd 79 69 e2 2a 09 f7

00f0 92 cd 2d fe 4d 61 13 b9 b9 f4 06 fb 78 9a e6 7c

0100 19 e4 1f 22 64 81 89 c7 29 02 03 01 00 01

Certificate Extensions: 8

1.3.6.1.4.1.311.21.1: Flags = 0, Length = 3

CA Version

V2.0

2.5.29.14: Flags = 0, Length = 16

Subject Key Identifier

7a fc 16 de 56 19 08 a3 39 21 5d 55 0f f2 57 be 8f 5e c7 7f

1.3.6.1.4.1.311.20.2: Flags = 0, Length = c

Certificate Template Name

SubCA

2.5.29.15: Flags = 0, Length = 4

Key Usage

Digital Signature, Non-Repudiation, Certificate Signing, Off-line CRL Signing, CRL Signing (c6)

2.5.29.19: Flags = 1(Critical), Length = 5

Basic Constraints

Subject Type=CA

Path Length Constraint=None

2.5.29.35: Flags = 0, Length = 18

Authority Key Identifier

KeyID=77 c9 74 69 2c 39 fe 38 65 f4 87 05 58 08 ce bd ba 97 da 10

2.5.29.31: Flags = 0, Length = 131

CRL Distribution Points

[1]CRL Distribution Point

Distribution Point Name:

Full Name:

URL=http://whicasarootca.asia.northwindtraders.com/CertEnroll/ASIA%20SA%20Root%20CA.crl

URL=ldap:///CN=ASIA%20SA%20Root%20CA,CN=whicasarootca,CN=CDP,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=asia,DC=Northwindtraders,DC=com?certificateRevocationList?base?objectClass=cRLDistributionPoint

1.3.6.1.5.5.7.1.1: Flags = 0, Length = 145

Authority Information Access

[1]Authority Info Access

Access Method=Certification Authority Issuer (1.3.6.1.5.5.7.48.2)

Alternative Name:

URL=http://whicasarootca.asia.northwindtraders.com/CertEnroll/whicasarootca.asia.northwindtraders.com\_ASIA%20SA%20Root%20CA.crt

[2]Authority Info Access

Access Method=Certification Authority Issuer (1.3.6.1.5.5.7.48.2)

Alternative Name:

URL=ldap:///CN=ASIA%20SA%20Root%20CA,CN=AIA,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=asia,DC=Northwindtraders,DC=com?cACertificate?base?objectClass=certificationAuthority

Signature Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.5 sha1RSA

Algorithm Parameters:

05 00

Signature: UnusedBits=0

0000 1f 1b 35 ad 28 ce 75 25 8b 26 18 19 8a 38 60 1c

0010 95 f6 bf d1 fb de 61 76 ba 24 71 97 f6 1d 48 92

0020 df 11 36 f8 40 de 58 20 b1 6a 55 ac 27 f9 8b f7

0030 c2 b6 ca 76 18 8a 47 69 39 28 e0 fd 81 98 3d 07

0040 df 6f 01 12 76 c3 5b 2a 9b 42 d9 b5 9c 40 fd 15

0050 0b 4a 9c 5f 88 17 f7 3a b8 42 90 58 19 88 10 4d

0060 4f 53 cf d8 29 1b 3c 5b c9 c0 f2 ad 13 61 b0 e7

0070 70 b5 25 df 15 0c 36 2a 50 95 b6 8f b7 1d d5 6e

Non-root Certificate

Key Id Hash(sha1): 7a fc 16 de 56 19 08 a3 39 21 5d 55 0f f2 57 be 8f 5e c7 7f

Cert Hash(md5): f2 bf 51 9f 3a d7 37 ec 03 20 79 b5 69 17 c4 26

Cert Hash(sha1): 61 0b 95 b6 06 ba 14 4c ae 89 24 9d 83 fd 06 49 9b ca 82 60

---------------- End Nesting Level 1 ----------------

================ Begin Nesting Level 1 ================

Element 3:

X509 Certificate:

Version: 3

Serial Number: 212566f75e7584b8478f7b59b4a9e212

Signature Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.5 sha1RSA

Algorithm Parameters:

05 00

Issuer:

CN=ASIA SA Root CA

OU=Asia

O=Northwindtraders

C=US

NotBefore: 9/20/2000 1:24 PM

NotAfter: 9/20/2002 1:33 PM

Subject:

CN=ASIA SA Root CA

OU=Asia

O=Northwindtraders

C=US

Public Key Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.1 RSA

Algorithm Parameters:

05 00

Public Key Length: 1024 bits

Public Key: UnusedBits = 0

0000 30 81 89 02 81 81 00 c7 29 b7 6c 1b 49 f7 77 a9

0010 f5 83 3d 78 5b 6b 25 29 85 03 c3 46 e8 eb 71 4c

0020 a4 4b 2f 2a 2b 5c c6 0d 53 32 ec 76 8c ef 19 67

0030 52 67 09 73 6e f0 13 6a 4c eb ce b8 ae aa ae d0

0040 81 a0 73 26 f4 b4 3a af 32 03 3b 61 a9 fd 23 05

0050 0c ac 1a f4 c7 d4 b1 e2 7a 8d db 98 21 45 38 e5

0060 2d 1a f7 dd 24 66 c4 32 f4 db f1 c4 f4 cb 10 20

0070 3c 9e ce af 45 99 b5 ae fb 7f f0 11 50 d5 96 bf

0080 a8 3b 4c d5 14 85 ed 02 03 01 00 01

Certificate Extensions: 6

1.3.6.1.4.1.311.20.2: Flags = 0, Length = 6

Certificate Template Name

CA

2.5.29.15: Flags = 0, Length = 4

Key Usage

Non-Repudiation, Certificate Signing, Off-line CRL Signing, CRL Signing (46)

2.5.29.19: Flags = 1(Critical), Length = 5

Basic Constraints

Subject Type=CA

Path Length Constraint=None

2.5.29.14: Flags = 0, Length = 16

Subject Key Identifier

77 c9 74 69 2c 39 fe 38 65 f4 87 05 58 08 ce bd ba 97 da 10

2.5.29.31: Flags = 0, Length = ae

CRL Distribution Points

[1]CRL Distribution Point

Distribution Point Name:

Full Name:

URL=http://whicasarootca.asia.northwindtraders.com/CertEnroll/ASIA%20SA%20Root%20CA.crl

URL=file://\\whicasarootca.asia.northwindtraders.com\CertEnroll\ASIA%20SA%20Root%20CA.crl

1.3.6.1.4.1.311.21.1: Flags = 0, Length = 3

CA Version

V0.0

Signature Algorithm:

Algorithm ObjectId: 1.2.840.113549.1.1.5 sha1RSA

Algorithm Parameters:

05 00

Signature: UnusedBits=0

0000 9d 64 9e 14 7c 07 32 06 f7 86 8b a6 fc b9 52 74

0010 31 35 ab 30 98 ee b5 d7 7d 1c 8a 3d f7 a4 89 e2

0020 2c f2 cc f9 ad 93 66 29 95 42 a8 77 a8 1b 7c 1c

0030 4a 4b 25 b1 68 3f 1e db 47 2c e6 46 dd fd c9 b3

0040 28 8a 55 14 c1 a6 64 9d 64 46 90 82 9a 73 55 85

0050 2e 6e 5d ff 19 2c 95 18 fa a1 dc e3 b8 54 bc 9a

0060 c3 3c 1b a7 e0 51 b1 90 3d a7 3b de e3 e8 55 a0

0070 54 40 a4 90 04 37 ff f6 ac a8 cd 24 6f e1 f9 08

Signature matches Public Key

Root Certificate: Subject matches Issuer

Key Id Hash(sha1): 77 c9 74 69 2c 39 fe 38 65 f4 87 05 58 08 ce bd ba 97 da 10

Cert Hash(md5): 51 66 26 77 89 a4 3d 07 f7 62 56 d2 de 0e d8 f6

Cert Hash(sha1): 9e 90 bb 26 24 e4 da dc 63 11 b8 18 2d af ad 39 56 81 66 51

---------------- End Nesting Level 1 ----------------

No CRLs

CertUtil: -dump command completed successfully.