

ITPro™
SERIES

WindowsITPro  **eBooks**

Keeping Your Business
SAFE from Attack:

Encryption and Certificate Services

By Jan De Clercq

Microsoft®



Microsoft®

Contents

Chapter 5: The Certificate Life Cycle Part 2:

Certificate Validation and Revocation	91
Overview of the Certificate Life Cycle	91
Certificate Validation	91
Regular Certificate Chain Processing	93
Chain Construction	94
Chain Validation	96
CTL Certificate Chain Processing	98
Cross-Certification Chain Processing	100
Certificate Retrieval	101
Key and Certificate Update	102
Certificate Revocation	102
Revoking a Certificate	102
PKI-Enabled Applications' Revocation-Checking Support	103
Automated Revocation Checking	104
CRL Distribution Points	104
Complete CRLs	107
Delta CRLs	110
Netscape Revocation Extensions	111
Certificate Expiry and Certificate Lifetimes	112

Chapter 5:

The Certificate Life Cycle Part 2: Certificate Validation and Revocation

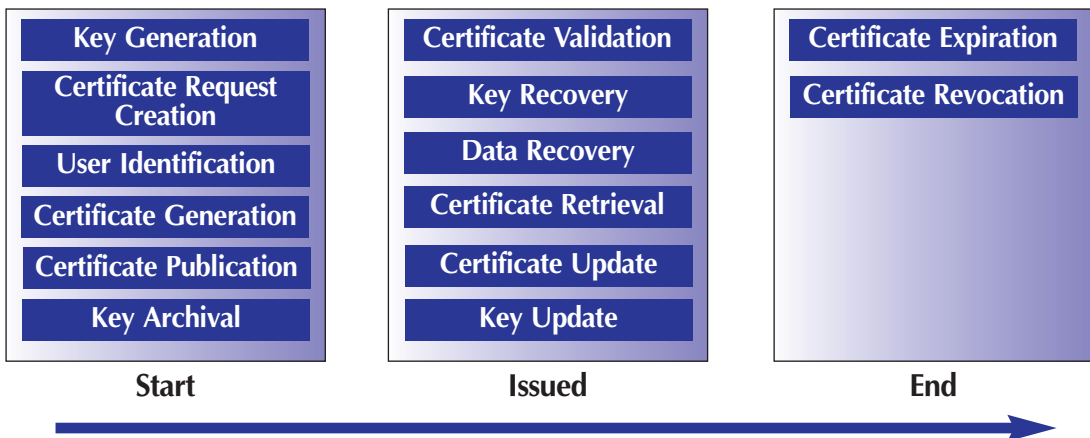
This chapter continues our focus on the Windows Server 2003 PKI certificate life cycle and its different subprocesses. In this chapter, we focus on certificate validation and revocation. We also spend some time on certificate retrieval, key and certificate updating, and certificate lifetime and expiry.

Overview of the Certificate Life Cycle

To review our introduction to the certificate life cycle in Chapter 4, remember that the life of a certificate can be subdivided into three main phases—start, issued, and end—and that different events can occur in each phase. The complete certificate life cycle, including its phases and their events, is illustrated again for your review in Figure 5-1.

Figure 5-1

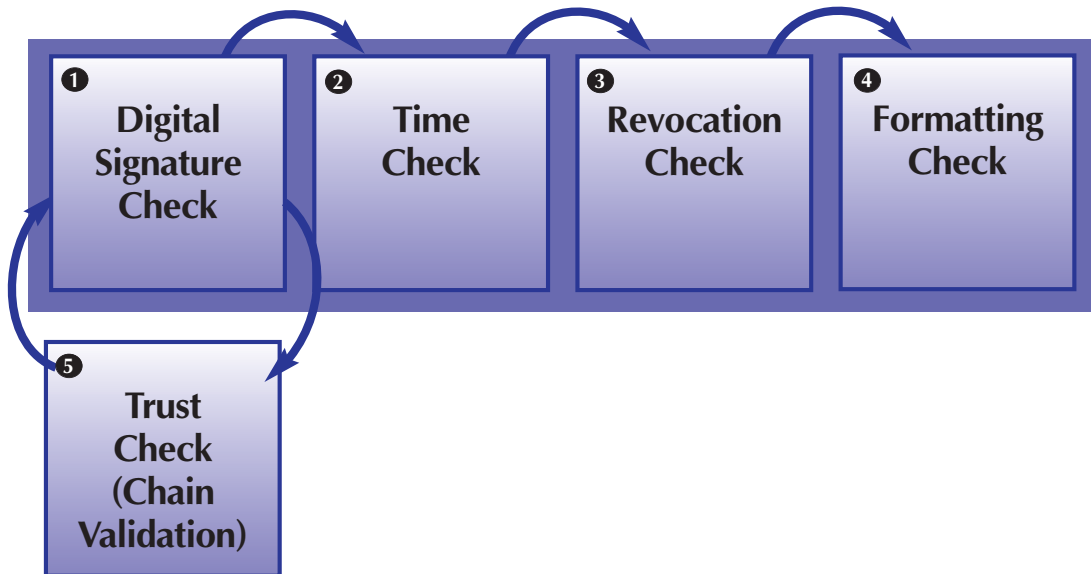
The certificate life cycle



Certificate Validation

Certificate validation is the process by which a PKI-enabled application finds out whether a certificate and the public key it contains are trustworthy. During certificate validation, the following checks are performed: digital signature, trust, time, revocation, and formatting. These checks are illustrated in Figure 5-2.

Figure 5-2
Certificate validation steps



During the *digital signature check*, the certificate-validation logic authenticates the digital signature that the issuer of the certificate has applied to the certificate content. The availability of a public key is not enough to validate a signature: The public key also must be trusted. This trustworthy key can be the public key of the issuing CA or the public key of another CA that is part of the certificate's certificate chain. As we discussed in Chapter 3, in Windows PKI, an explicitly trusted CA certificate and public key are evidence of a *trust anchor*. The trusted CA certificate and public key are available from two specific containers in an entity's certificate store (we explained these containers in Chapter 2). The process of discovering a trusted CA certificate occurs during the *trust check*, which is also referred to as *certificate chain validation*. Certificate chain validation may trigger different certificate-validation loops—one for each certificate in the certificate chain. We explain certificate chain validation in more detail later in this chapter.

During the *time check*, the start-end dates of the certificate are compared to the current time. One reason a certificate's lifetime is limited is to cope with the advances in computer technology. For example, breaking a 512-bit, key-rooted, asymmetric cipher becomes easier every year.

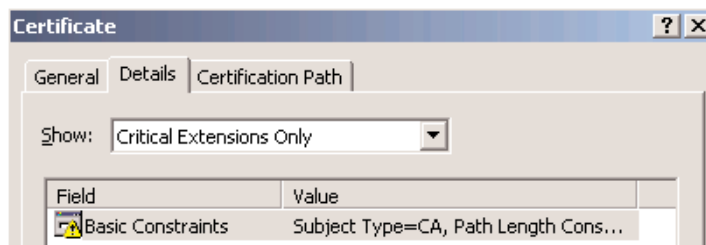
During the *revocation check*, the validation logic checks whether or not the certificate has been revoked. Windows 2000 and Windows Server 2003 PKI support complete certificate revocation lists (CRLs) and CRL distribution points (CDPs). Windows Server 2003 PKI also supports delta CRLs (discussed in detail later in the chapter). CRLs, delta CRLs, and CDPs can provide automated certificate-revocation checking. We discuss revocation in more detail in the "Certificate Revocation" section.

During the *formatting check*, the format of the certificate is checked and validated according to the standard certificate format defined in the ITU-T X.509 standard. This check also includes the validation of the certificate's extensions. Among a certificate's extensions are the basic constraints, application policy, issuance policy, and name-constraint extensions that we discussed in Chapter 3. A certificate also contains a set of critical extensions that, following the X.509 standard, must be validated by every application. The validation or evaluation of the other noncritical extensions depends on the PKI application that is using the certificate. Most Secure MIME (S/MIME) applications, for example, will evaluate the certificate subject's RFC 822 name (e.g., jan.declercq@hp.com). The name will be compared to the sender entry in the header of the SMTP message. In the case of S/MIME, this check protects against impersonation or man-in-the-middle attacks. In such attacks, a malicious entity reuses a user's identity. Most Secure Sockets Layer (SSL) implementations perform a similar check. SSL compares the subject's RFC 822 name to the name that is contained in the URL.

To view the different X.509 certificate extensions, you must use the Details tab in the Certificate properties dialog box (illustrated in Figure 5-3).

Figure 5-3

Bringing up an X.509 certificate's critical extensions



At the top of this tab is a drop-down box called Show: that lets you filter the X.509 field and extension data that are displayed in the bottom part of the tab. The filter lets you differentiate between Version 1 Fields Only, All Extensions, Critical Extensions Only, and Properties Only. To display only the critical extensions, select Critical Extensions Only.

Regular Certificate Chain Processing

So what is a certificate chain, and why do we need to process it during certificate validation? We'll explain this concept using the example of a hierarchical trust model.

In a hierarchical trust model, every end-entity's certificate chain consists of all CA certificates that are lying on the path between the user and the root CA of the PKI hierarchy. In a PKI hierarchy, every certificate contains a pointer to its parent or issuing CA; this pointer is stored in the certificate-issuer field. All of this is illustrated in Figure 5-4.

Figure 5-4
Certificate chain processing

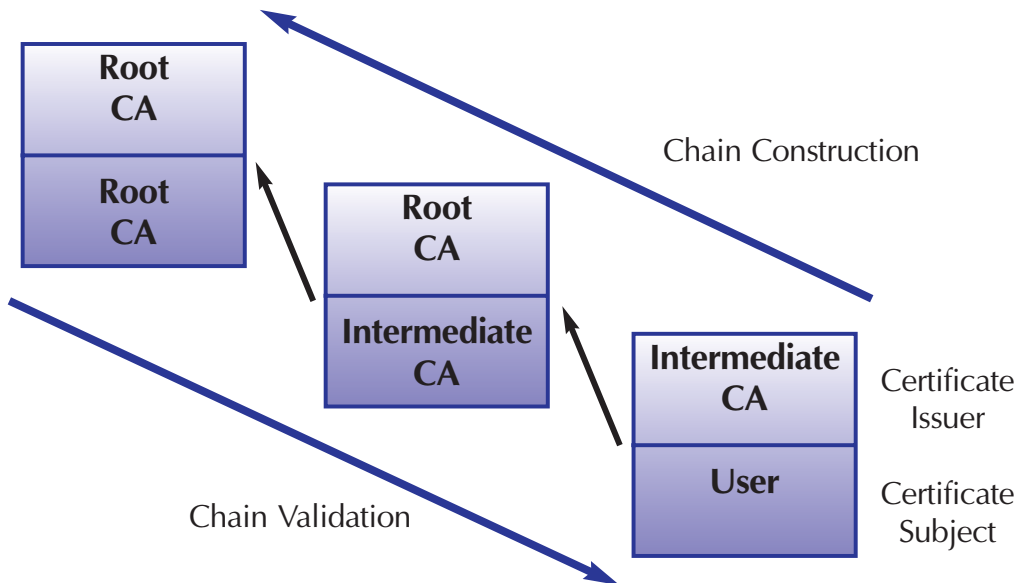


Figure 5-4 shows the certificate chain of a user certificate that has been issued by a CA that is part of a two-level PKI hierarchy. In the figure, certificates are represented in a much-simplified way (to provide an easy explanation). Every certificate is represented using the certificate subject and the certificate issuer. In this example, the user's certificate subject is the user; its issuer is the intermediate CA. The intermediate CA's certificate subject is the intermediate CA; the certificate has been issued by the root CA. In a hierarchy, the root CA always has a self-signed certificate; so, in this example, the certificate subject and issuer are identical.

During certificate validation, the certificate-validation software processes a certificate's certificate chain. This process can be split into two subprocesses: *chain construction* and *chain validation*.

Chain Construction

During *chain construction*, the certificate-validation software runs through the certificate's chain until it finds a trusted CA certificate, also known as a *trust anchor*. In Chapter 3, we explained which CA certificates are considered trust anchors in Windows PKI. In the examples in Figures 5-5 and 5-6, the validation software finds a trust anchor at the root CA level (Figure 5-5) and at the intermediate CA level (Figure 5-6).

Figure 5-5
Certificate chain processing example 1

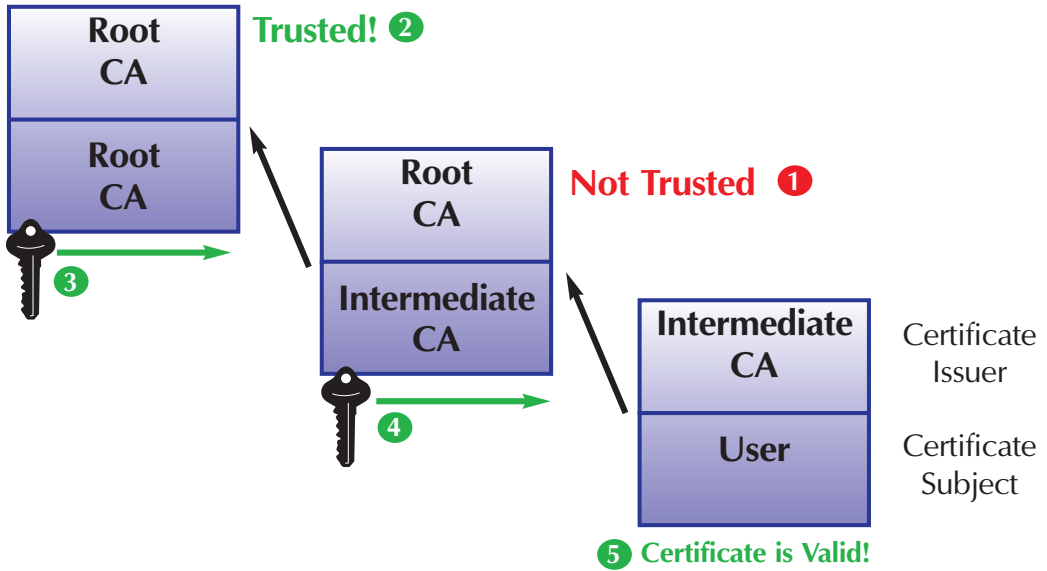
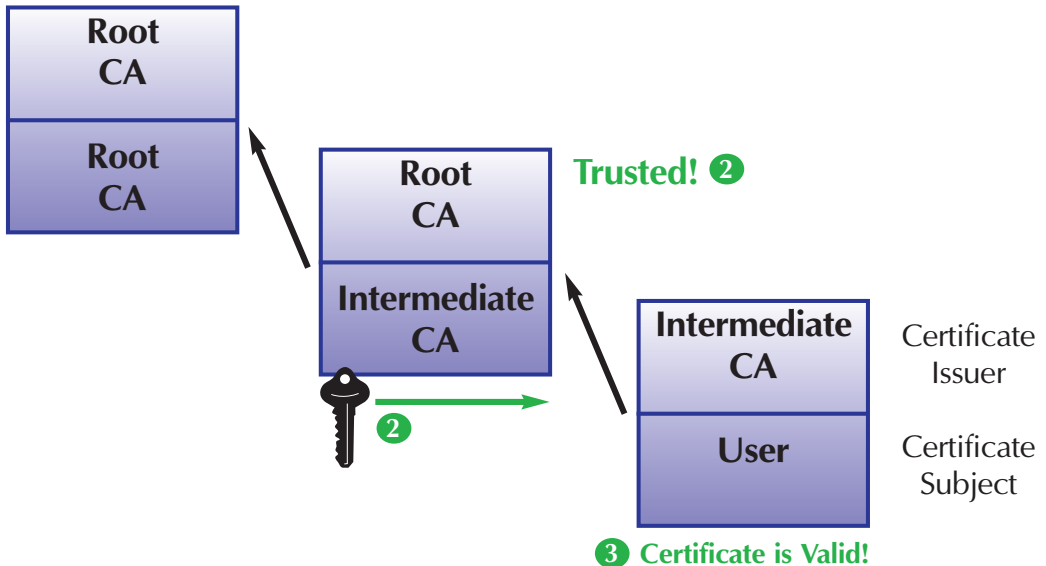
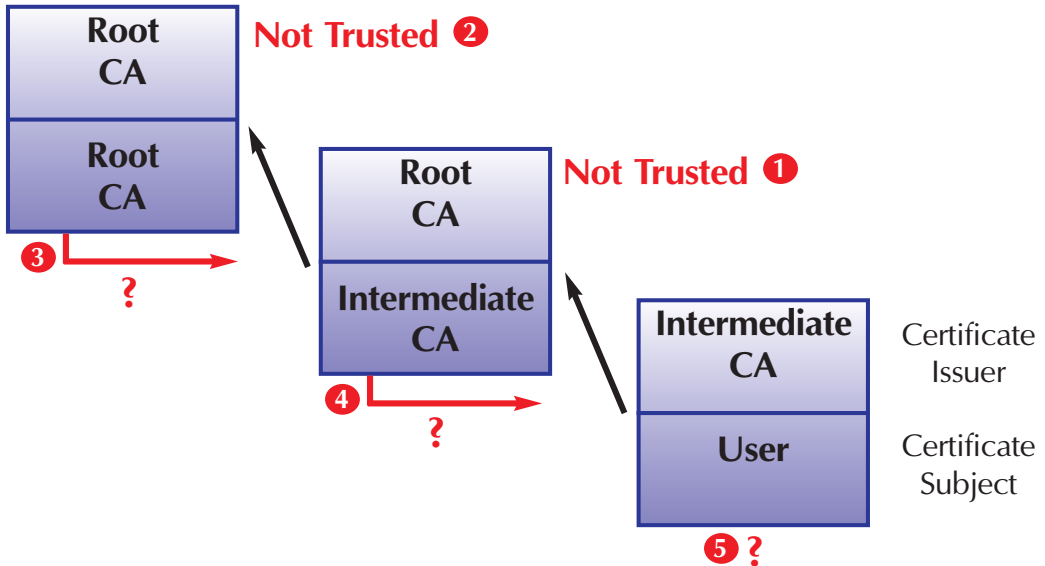


Figure 5-6
Certificate chain processing example 2



When a trust anchor is found, the chain-construction subprocess stops, and the validation logic switches to chain validation. In Figure 5-7, the validation logic cannot find a trust anchor. When this happens, the certificate chain process stops, and no decisions can be made regarding the trust-worthiness of the certificate.

Figure 5-7
Certificate chain processing example 3



Chain Validation

During chain validation, the certificate-validation software walks the chain in the opposite direction (top-down) and validates every CA certificate that is part of the chain. To be validated, the certificate must be available locally in one of the containers of a user's certificate store (see Chapter 2). When a certificate is not available locally, the Windows PKI software will use the Authority Information Access (AIA) method that is explained shortly to obtain a copy of the certificate.

The identification of a CA certificate during chain validation is based on the Authority Key Identifier (AKI) field in the certificate under verification. A certificate's AKI field can contain different types of information:

- The AKI field may contain the issuer name and the serial number of the issuer's certificate. In that case, the chain-validation logic will try to find a matching certificate using a certificate's Serial number and Subject fields. This way of identifying a certificate is called an *exact match*.
- The AKI field may contain the public key identifier (KeyID) of the issuer's certificate. In that case, the chain-validation logic will try to find a matching certificate using a certificate's Subject Key Identifier (SKI) extension. This way of identifying a certificate is called a *key match*.

If the certificate under verification does not contain an AKI field, the chain-validation logic will try to identify the issuing CA's certificate by matching the name in the Issuer field of the certificate under verification with the name in the Subject field. This way of identifying a certificate is called a *name match*.

The *AIA method* that the validation logic uses to obtain a local copy of the certificate simply means that the software will try to download the certificate from an online location. To do this, it will use a certificate's AIA field, which contains an LDAP, an HTTP, or a File System pointer to a location where the CA's certificate is stored. If the AIA field has multiple entries, all entries will be tried out in the order that they are listed in the AIA field. All certificates that are downloaded from an AIA location will be cached in the certificate store and on the file system for future use. On the file system, the certificates are cached in the \Documents and Settings\\Local Settings\Temporary Internet Files folder. Note the use of the <username> variable in the previous file system path: The cache location is dependent upon the user security context under which the calling application is running.

If the certificate is not available, certificate verification will fail. If the certificate is available, the certificate validation logic will run (for every certificate in the chain) through all of the steps explained earlier: digital signature, time, revocation checking, and formatting.

A certificate's certificate chain can be viewed from the Certificate properties dialog box, in the "Certification path" tab. Figures 5-8 and 5-9 show the certificate chain of a certificate that ends in a trusted CA certificate (Figure 5-8) and another that ends in an untrusted CA certificate (Figure 5-9) as they are displayed in the Windows Certificate properties dialog box.

Figure 5-8

Certificate chain viewed from the Certificate properties dialog box: trusted CA certificate

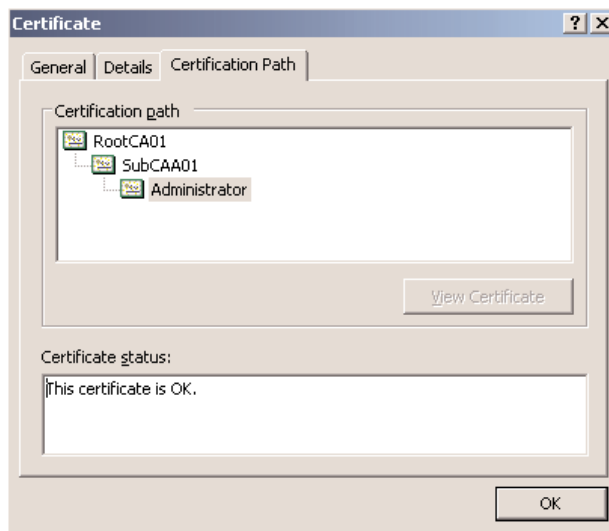
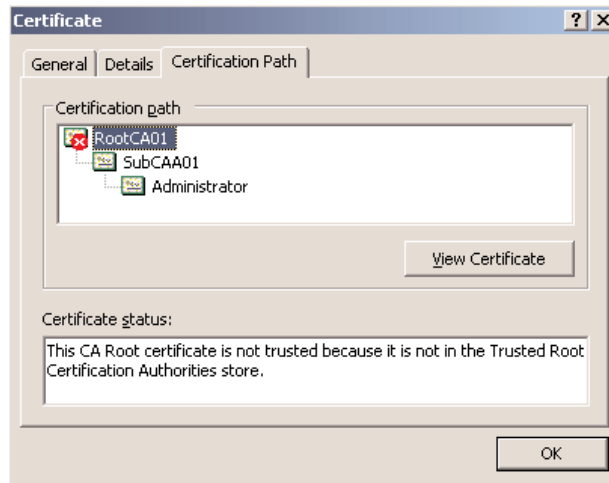


Figure 5-9

Certificate chain viewed from the Certificate properties dialog box: untrusted CA certificate



When a user downloads a certificate using the Windows 2000 or Windows Server 2003 CA Web interface, he or she has the choice to download just the certificate, or the certificate together with all of the certificates that are part of its certificate chain. This option can be interesting, for example, for certificate validation on portable computers. All CA certificates in the certificate chain are made available at once on the client, and so there is no need for the client software to download the certificates in the chain using the AIA pointers.

As this section illustrates, certificates can be invalid for different reasons: expiration or other time problems, invalid signatures, unavailability of a trusted CA certificate, improper use, improper formatting, revocation, and so forth. That is why finding out the exact reason a certificate is not valid is sometimes a tough job.

CTL Certificate Chain Processing

A special case of certificate chain processing is certificate trust list (CTL) certificate chain processing. CTLs are signed lists of trusted root CA certificates: CTLs can contain only self-signed root CA certificates. CTLs can be defined using Windows 2000 or Windows Server 2003 Group Policy Objects (GPOs); they are downloaded to the Enterprise Trust container in an entity's certificate store. The Enterprise Trust container is not a trust anchor container because its content is not considered trusted by default.

For a CTL and its content to be trusted, the CTL signing certificate must be valid. This means that the CTL signing certificate should pass the digital signature, time, revocation, and formatting checks. For the digital signature check to succeed, the CTL signing certificate's certificate chain should contain a certificate that is part of the Trusted Root Certification Authorities container.

Figures 5-10 and 5-11 show the certificate chain of a certificate that is part of a valid CTL (Figure 5-10) and one that is part of an invalid CTL (Figure 5-11) as the chain is displayed in the Windows Certificate properties dialog box.

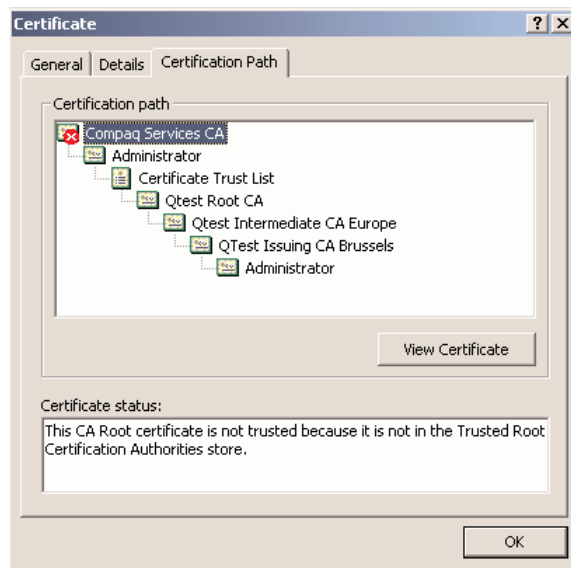
Figure 5-10

Certificate part of a certificate chain starting with a valid CTL



Figure 5-11

Certificate part of a certificate chain starting with an invalid CTL

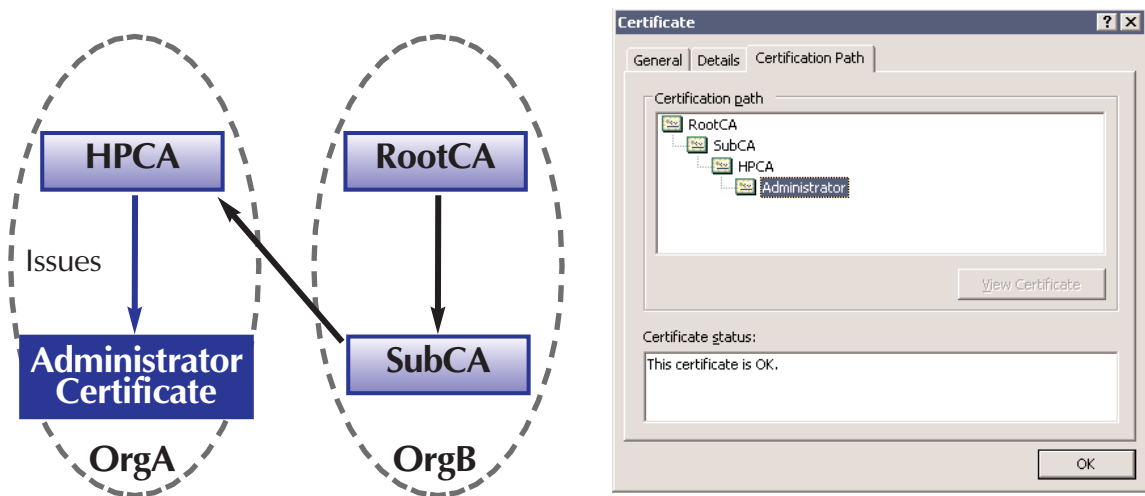


Cross-Certification Chain Processing

Cross-certification is a new Windows Server 2003 PKI trust feature that we explained in detail in Chapter 3. Unlike CTLs, cross-certification allows for very granular PKI trust definitions between different CA entities. When you set up cross-certification between two CA entities, each CA becomes both a parent and a subordinate CA. This relationship has interesting effects on how certificate chain building works in a cross-certification setup.

How a cross-certified trust relationship shows up in the “Certification path” tab of the Certificate properties dialog box is illustrated in Figure 5-12.

Figure 5-12
Cross-certification example



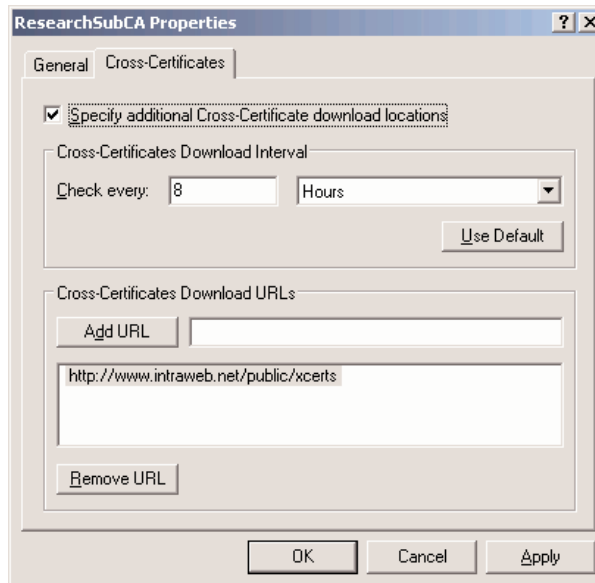
The CA trust relationships that are linked to this setup are shown on the left side of Figure 5-12. In this example, a one-way cross-certification trust is set up between OrgB and OrgA. The SubCA issues a cross-certificate to the HPCA. This configuration will allow users in OrgB to trust a certificate named Administrator that was issued by HPCA. The users in OrgB trust the RootCA, SubCA chains to the RootCA, HPCA is cross-certified by the SubCA, and the Administrator certificate was issued by HPCA.

Cross-certificates are downloaded at every autoenrollment event (just as trusted root CA certificates are) to Windows PKI clients that are members of a domain. In Chapter 4, we explained when this event occurs and how it can be manually enforced. Downloaded cross-certificates are stored in a PKI user’s Intermediate Certification Authorities certificate store container.

For cross-certificates that are stored on locations other than the default AD locations, Windows XP and Windows Server 2003 allow you to define additional cross-certificate download locations for each certificate that is stored in your certificate store. To designate such locations, right-click a

certificate to bring up its Properties dialog box, and then select the Cross-Certificates tab (as illustrated in Figure 5-13). This tab also lets you specify the cross-certificates download interval, as well as the download URLs.

Figure 5-13
Additional cross-certificate download locations



Certificate Retrieval

Certificate retrieval deals with how certificates are retrieved from a repository by a PKI user. In Windows 2000 and Windows Server 2003, certificates can be retrieved manually from any location in which the CA publishes them: AD, a Web site, or a file share.

Windows 2000, Windows XP, and Windows Server 2003 also provide automatic retrieval of CA certificates during certificate validation. CA certificate-download locations are mentioned in the Authority Information Access (AIA) certificate extension.

An interesting way for PKI users to retrieve their personal certificates from AD and store them in their local certificate store is to drag them from the AD User Object to the Personal container in the MMC Certificates snap-in.

Personal certificates issued by a standalone CA can be retrieved from the CA's Web interface. If certificates are not downloaded from the CA's Web site within 10 days, they are purged. You can modify this default behavior by editing the certdat.inc file. You can set the number of days before a certificate is purged from the standalone CA's Web site by modifying the nPendingTimeoutDays setting in the certdat.inc file (located in the c:\winnt\system32\certsrv directory).

Key and Certificate Update

To provide better security, you should update cryptographic keys and certificates regularly. Windows supports both manual and automatic key and certificate updating.

In Windows Server 2003, automatic certificate update is available for both machine and user accounts. Machine and user certificates that are set up for automatic enrollment will also be automatically updated when the autoenrollment event occurs.

To manually update your proper user keys and certificates, you must use the Certificates snap-in. You can choose to renew an existing certificate using the same keys or using a newly generated key pair. To manually update the keys and certificates of a Windows CA requires a special procedure, *CA rollover*, which we discuss in Chapter 7.

Certificate Revocation

Some of the most important aspects in the design of a PKI are certificate revocation and, more particularly, automated revocation checking. *Certificate revocation* assures that a certificate's serial number is added to a blacklist called the "Certificate Revocation List (CRL)" when a PKI user's private key is compromised. Certificate revocation also guarantees that the revocation information is efficiently distributed to PKI clients and PKI-enabled applications. *Automated revocation checking* is critical for PKI systems that deal with confidential information, or valuable information or transactions, or both.

Next, we examine in more detail the process of revoking a certificate, Windows PKI-enabled applications' revocation-checking support, and automated revocation-checking solutions.

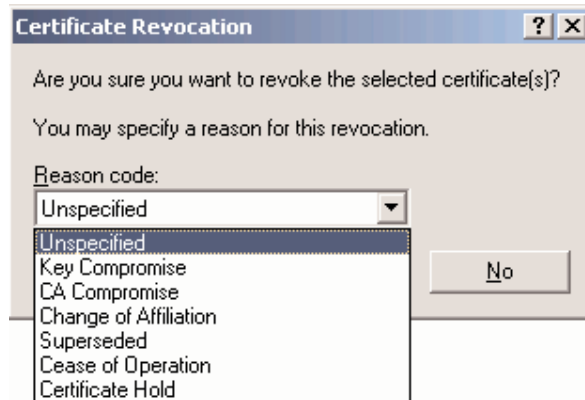
Revoking a Certificate

A Windows CA administrator can revoke a certificate from the MMC Certification Authority snap-in or from the command line. In the Certification Authority snap-in, open the Issued Certificates container, and right-click the certificate you want to revoke. Then select the All Tasks\Revoke Certificate menu option. To revoke a certificate from the command line, use the following certutil command:

```
Certutil revoke <certificate serial number> <reason code>
```

A CA administrator may have different reasons to revoke a certificate: An employee may leave the organization, there may be a compromise or suspected compromise of a user's private key, and so forth. When the CA administrator revokes a certificate, he or she can select a revocation reason code (as illustrated in Figure 5-14). Valid revocation reason codes are Unspecified, Key Compromise, CA Compromise, Change of Affiliation, Superseded, Cease of Operation, and Certificate Hold.

Figure 5-14
Certificate revocation reason codes



PKI-Enabled Applications' Revocation-Checking Support

Not all Windows PKI-enabled applications automatically perform revocation checking. Also, revocation checking sometimes depends on an application-specific configuration setting. Table 5-1 provides an overview of the most commonly used Windows PKI-enabled applications revocation-checking support.

Table 5-1 PKI-enabled applications revocation checking support

PKI-Enabled Application	Revocation-Checking Support
Internet Explorer (SSL-TLS)	Configuration option in Advanced tab of Internet Options: "Check for server certificate revocation (requires restart)."
Internet Explorer (Authenticode Code Signing)	Configuration option in Advanced tab of Internet Options: "Check for publisher's certificate revocation."
IIS (SSL-TLS)	IIS 5.0 and later have revocation checking enabled by default.
Outlook (S/MIME)	Enabled by default in Outlook 2002 and Outlook 2003. Can be enabled in Outlook SR1 using the registry hack outlined in the text.
IPSec	Not supported in Windows 2000. From Windows 2000 SP2 on, can be enabled using the registry values outlined in the text.
EFS	Not supported in Windows 2000 EFS. Supported in Windows XP and Windows Server 2003 EFS when other users are added to the EFS settings of a file set up for EFS file sharing.
Smart Card Logon	The smart card logon authentication logic has revocation checking enabled by default.

To enable revocation checking for S/MIME in Outlook SR1, use the following registry hack: Create a REG_DWORD entry called "PolicyFlags" in the HKLM\SOFTWARE\Microsoft\Cryptography\{7801ebd0-cf4b-11d0-851f-0060979387ea} registry key, and set it to value 00010000.

To enable revocation checking for IPsec in Windows 2000 SP2 and later platforms, use the following registry hack: Create a REG_DWORD entry called “StrongCrlCheck” in the HKLM\System\CurrentControlSet\Services\PolicyAgent\Oakley registry key, and set it to value 1 or 2. This key’s values have the following meaning:

- 0: Disables CRL checking for certificate-based IP Security (IPSec) authentication.
- 1: Enables CRL checking and fails the validation process only if the CRL explicitly indicates that the certificate is revoked. All other failures, including when the CDP URL is unavailable, are ignored.
- 2: Enables CRL checking and fails certificate validation on any CRL-check errors.

Automated Revocation Checking

In the PKI world, different models are available for automated revocation checking. Most of these models, with the exception of Certificate Revocation Trees (CRTs) and the Online Certificate Status Protocol (OCSP), are based on Certificate Revocation Lists (CRLs). Examples are complete CRLs, CDPs, enhanced CRLs, delta CRLs, and indirect CRLs. We will not discuss all of these methods, which are beyond the scope of this eBook. For a good overview of revocation-checking methods, read the book *Understanding Public-Key Infrastructure*, by Carlisle Adams and Steve Lloyd (Que, 1999).

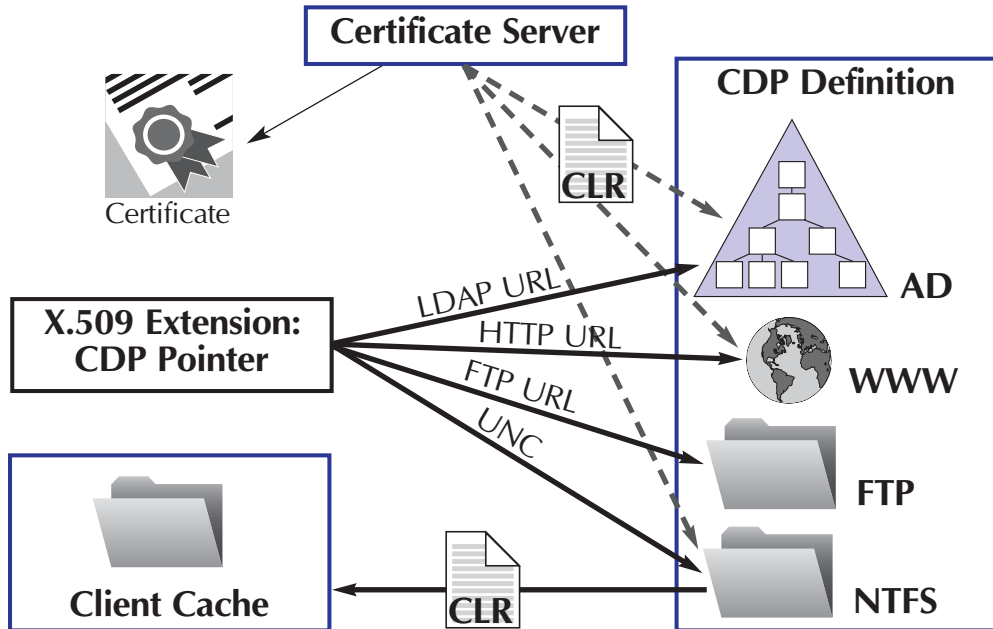
Windows 2000 PKI supports complete CRLs and CDPs. Windows Server 2003 PKI adds support for delta CRLs. Windows 2000 and Windows Server 2003 also support specific Netscape revocation extensions.

Both Windows 2000 and Windows Server 2003 publish CRLs at regular time intervals. In both environments, a CA administrator can also force the publication of a new CRL (or delta CRL in Windows Server 2003). We explain how to configure the CRL publication intervals and how to force CRL publication in the next section of this chapter.

An often-heard revocation requirement is support for the Online Certificate Status Protocol (OCSP). OCSP offers real-time certificate revocation information to PKI users. The OCSP protocol is defined in RFC 2560. Neither Windows 2000 nor Windows Server 2003 support OCSP out of the box. Support can be added using third-party software from vendors such as Alacris (more information is available at <http://www.alacris.com>) or CoreStreet (more information is available at <http://www.corestreet.com>).

CRL Distribution Points

CDPs offer a convenient way to automate revocation checking. Each certificate generated by a Windows 2000 or Windows Server 2003 CA can include one or more CDP pointers. These pointers are stored in the CRL Distribution Points X.509 certificate extension. A CDP can be a URL (HTTP or LDAP) or a file share. Once a certificate has been issued, its CDP pointers cannot be modified. How CDPs work is illustrated in Figure 5-15.

Figure 5-15*Certificate Revocation List Distribution Points (CDPs) operation*

If a Windows PKI-enabled application that is CDP-enabled does not find a local copy of the CRL or delta CRL, it will check the certificate's CDPs for an up-to-date CRL or delta CRL. If a CRL or delta CRL is available from the CDPs, the application will download the CRL or delta CRL and cache it locally for the lifetime of the CRL or delta CRL. If a certificate does not contain any CDPs, the PKI-enabled application will query the certificate's issuing CA for a CRL or delta CRL.

For CDPs to function correctly, not only is certificate and PKI-enabled application support required, but the CA also should support the CDPs. The CA must have an exit module that can publish the CRL or delta CRL to the appropriate file system, Web, or AD CDP. By default, every Windows 2000 and Windows Server 2003 CA includes an exit module that can handle CDP publication. None of these modules can automatically publish CRLs or delta CRLs to HTTP CDPs; you can, however, do this manually (we explained exit modules in Chapter 2).

In addition to automated revocation checking, CDPs also can increase CRL or delta CRL availability. Each certificate can contain different CDPs. If one CDP is unavailable, the PKI logic will try another CDP.

The CDP field content of the certificates a CA issues can be configured from the Properties dialog box of a Windows CA object, in the Extension tab (as illustrated in Figure 5-16). To add a new CDP, click the Add button. The Extensions tab also contains a set of CDP-specific flags (at the bottom of the dialog box) that are explained in Table 5-2. You can configure the CA certificate's proper CDP field using a capolicy.inf configuration file (as we mentioned in Chapter 3).

Figure 5-16
Configuring CDPs

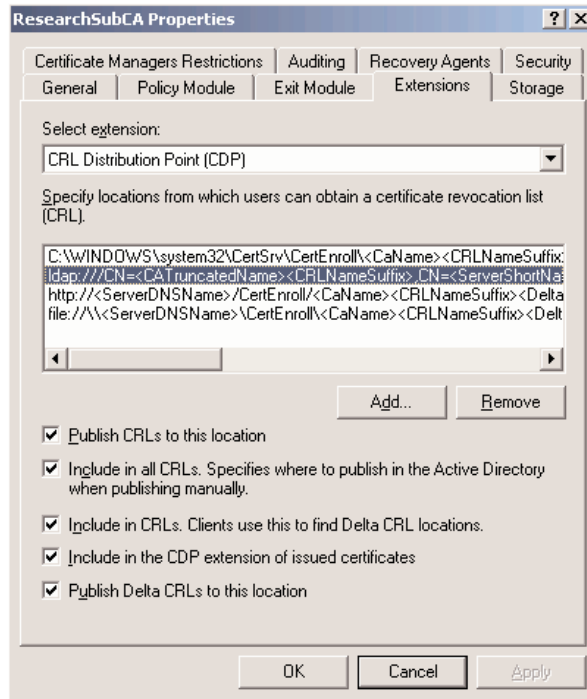


Table 5-2 CDP flags

CDP	Flag Meaning
Publish CRLs to this location.	Used by the CA to determine whether to publish base CRLs to this CDP URL (not available for HTTP CDPs).
Include in all CRLs.	Not used during revocation checking. Specifies where to publish in AD when manually publishing using certutil -dspublish.
Include in CRLs.	Used by clients during revocation checking to find delta CRL locations from base CRLs.
Include in CDP extension of issued certificates.	Used by clients during revocation checking to find base CRL locations.
Publish delta CRLs to this location.	Used by the CA to determine whether to publish delta CRLs to this CDP URL (not available for HTTP CDPs).

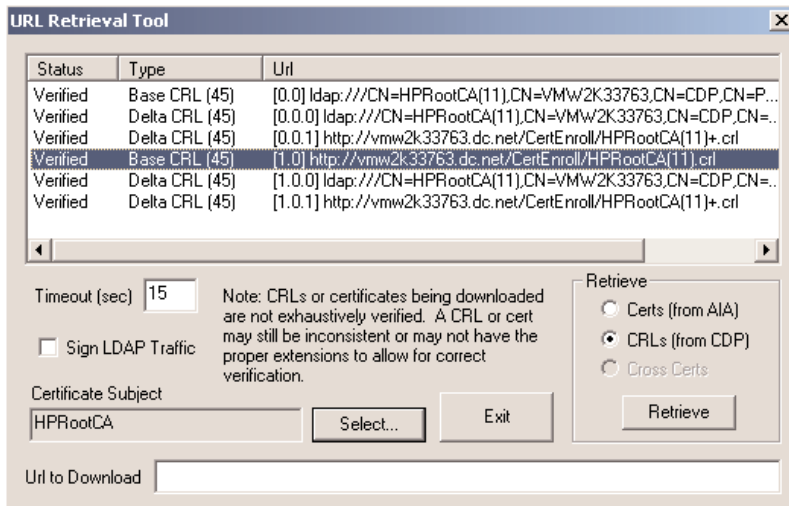
Most organizations do not include CDPs in root CA certificates. Doing so forces the certificate-validation logic to bypass the revocation check for a root CA certificate. If a root CA's private key is compromised, you must rebuild your entire PKI, instead of just revoking its certificate.

A convenient way to test the CDPs embedded in a Windows X.509 certificate is with the URL retrieval tool. This tool comes with the Windows Server 2003 version of the certutil.exe command-line tool. To bring up the URL retrieval tool, type

```
certutil -URL <certificate_file_name>
```

at the command line. This action brings up the URL Retrieval Tool dialog box illustrated in Figure 5-17.

Figure 5-17
URL Retrieval Tool dialog box



To retrieve the CRLs and delta CRLs, click the Retrieve CRLs (from CDP) radio button, then click Retrieve. Double-clicking one of the rows in the upper part of the tool brings up the CRL viewer for the selected CRL. You can also use the tool to retrieve the CA certificates mentioned in a certificate's AIA field.

Complete CRLs

A CRL contains a time-stamped list of revoked certificates, which is signed by a CA and made available to PKI users in a public repository. In a CRL, each revoked certificate is identified by its certificate serial number. CRLs are defined in the ITU-T X.509 standard and in RFC 2459.

CRLs in their most basic format are known as *complete CRLs*. They are also referred to as *base CRLs*, or *full CRLs*. A Windows 2000 and Windows Server 2003 CA generates a complete CRL at predefined intervals. Complete CRLs tend to be huge because revocation information accumulates in each of them. Windows CRLs support versioning, but a new version automatically inherits all revocation information from the preceding version, so a CRL becomes no smaller until a certificate expires. Also, each new CRL version causes the client to download the complete CRL, which is not an efficient use of network bandwidth. As a result, many administrators configure longer CRL lifetimes. But long CRL lifetimes reduce the revocation information's timeliness because new revocation information is not immediately available.

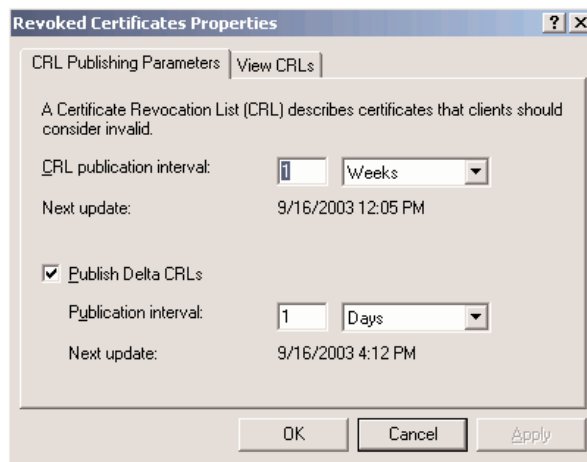
To limit the size of the complete CRLs in a Windows 2000 PKI environment, you can do one of these three things:

1. *Define multiple CAs.* If you define multiple CAs, with each CA having its own CRL, the size of those individual CRLs will be much smaller than the size of the CRL generated if you create just a single CA.
2. *Generate certificates with a short lifetime.* Windows 2000 CRLs are self-cleaning, which means that expired certificates are automatically removed from the CRL.
3. *Generate a new CA key pair.* Every time the CA key pair is renewed, a new CRL will be generated. The new CRL will be signed using the newly generated private key.

In Windows Server 2003, you can use delta CRLs to get around the complete CRL deficiencies. We explain delta CRLs in the next section.

You can configure the complete CRL publication intervals from the Properties dialog box of the Revoked Certificates container in the Certification Authority snap-in (as Figure 5-18 shows). You can force a CRL publication by right-clicking the Revoked Certificates container and selecting All Tasks\Publish. This action brings up the Publish CRL dialog box, which requests the type of CRL you want to manually publish: a new CRL (or complete CRL), or a delta CRL only.

Figure 5-18
Configuring CRL publication intervals



To look at the content and formatting of a CRL, go to the View CRLs tab of the Revoked Certificates Properties dialog box (as illustrated in Figure 5-19).

Figure 5-19
Viewing CRLs

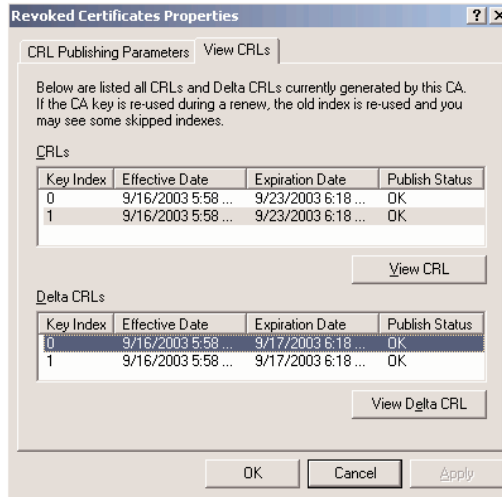


Figure 5-20 shows the layout of a complete CRL issued by a Windows Server 2003 CA as it shows up in the built-in CRL viewer. Notice the presence of some typical CRL extensions: Effective date, Next update, CA Version, CRL Number, Next CRL Publish, Freshest CRL, and Published CRL Locations. A list of the revoked certificates on a CRL is available from the Revocation List tab, shown in Figure 5-21.

Figure 5-20
CRL layout

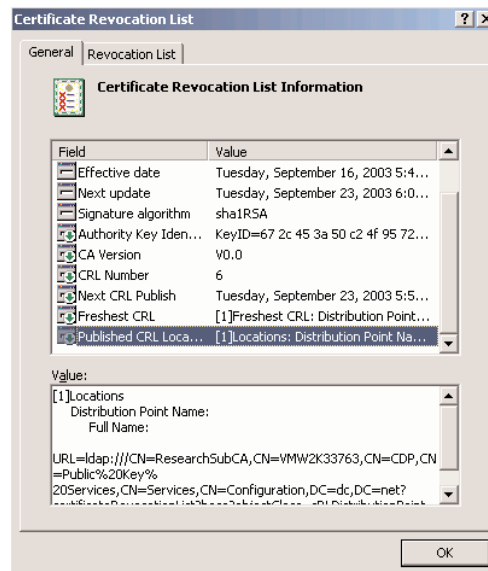
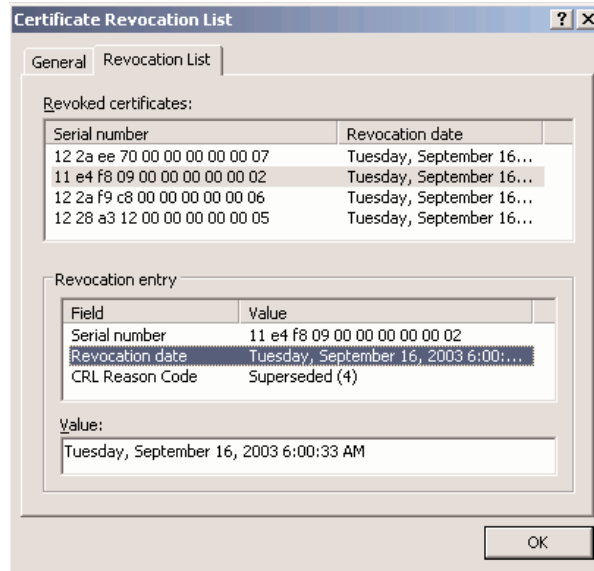


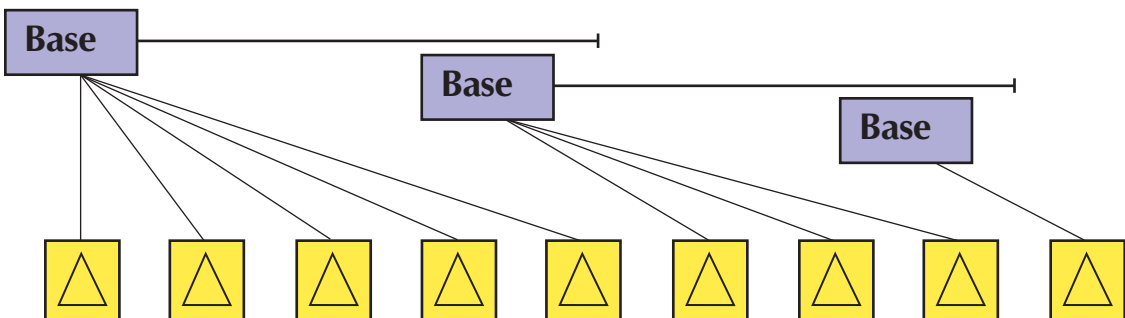
Figure 5-21
CRL content



Delta CRLs

Windows Server 2003 resolves the complete CRL problems (bandwidth impact and revocation information up-to-dateness) by introducing delta CRLs. As Figure 5-22 illustrates, delta CRLs are relatively small CRLs that contain only the revocation changes that have occurred since the most recent base CRL. Because delta CRLs are small, PKI clients can download them more regularly, and the CA can provide more accurate revocation information to its clients. Only Windows XP Professional and later Windows clients can check a certificate's validity against a delta CRL. Again, delta CRLs are not a Microsoft invention; they are defined in RFCs 2459 and 3280.

Figure 5-22
Delta CRL operation

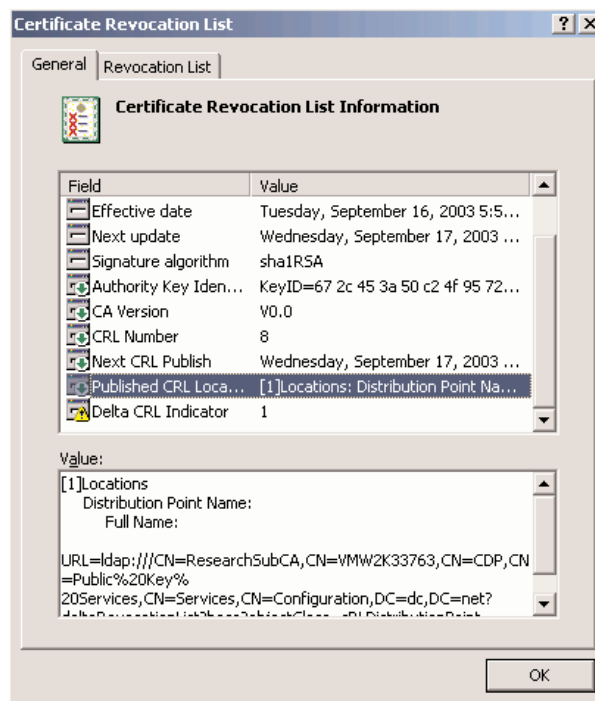


As with complete CRLs, Windows clients also cache delta CRLs. If a base CRL expires, the client retrieves from the CDP a new base CRL that is specified in the certificate. If the base CRL is valid but the cached delta CRL is expired, a Windows client retrieves from the CDP only the delta CRL mentioned in the certificate.

Also as with complete CRLs, you configure delta CRL settings and view delta CRLs' content and formatting in the CA's Revoked Certificates Properties dialog box (illustrated in Figure 5-18). The procedure that is used for manual CRL publication also applies to manual delta CRL publication.

Figure 5-23 shows the layout of a delta CRL issued by a Windows Server 2003 CA as it shows up in the built-in CRL viewer. Notice the presence of the delta CRL Indicator extension, which means that this is a delta CRL, not a complete CRL. The value in the Delta CRL Indicator extension is the CRL number of the base CRL this delta must be associated with. As for a complete CRL, a list of the revoked certificates on a delta CRL is available from the Revocation List tab.

Figure 5-23
Delta CRL layout



Netscape Revocation Extensions

Netscape is using a proprietary online certificate-revocation-checking method. Netscape embeds a custom extension, the netscape-revocation-url, in all its certificates. The netscape-revocation-url points to a Web page where the certificate revocation can be checked. To send the revocation-checking request to the Web page, Netscape is using the HTTP GET method with a URL that is the

concatenation of the `netscape-revocationurl` and the serial number of the certificate that needs to be checked. The response that comes back from the Web server is a document with Content-Type `application/x-Netscape-revocation`. The document contains a single digit that is 1 if the certificate is not valid or 0 if the certificate is valid.

To enable Windows 2000 or Windows Server 2003 to issue certificates that contain this extension, use the `certutil` tool:

```
certutil -setreg Policy\revocationtype +AspEnable
```

You also have to restart the CA service to make this change effective.

Certificate Expiry and Certificate Lifetimes

A certificate expires at the moment in time specified in the certificate's "Valid to" field. Under normal circumstances, the issuing CA decides on the content of this field and of the certificate validity period. Windows 2000 and Windows Server 2003 PKI also allow the certificate requestor to specify the validity period. This feature is disabled by default on enterprise CAs and enabled by default on standalone CAs. To enable this feature, use the following `certutil` command:

```
certutil -setreg policy\EditFlags +EDITF_ ATTRIBUTEENDDATE
```

To disable this feature, use the following `certutil` command:

```
certutil -setreg policy\EditFlags -EDITF_ ATTRIBUTEENDDATE
```

The certificate lifetime preferences are set differently on enterprise and standalone CAs. On a standalone CA, the certificate lifetime is set using a set of registry hacks. Both the `ValidityPeriod` (which can be days, weeks, months, or years) and the `ValidityPeriodUnits` keys (which hold a number) are used to set the certificate lifetime. These keys are located in the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CertSvc\Configuration\<CAName>` registry container. The default lifetime of certificates issued by a standalone CA is one year. To change the certificate lifetime, you can use the following script, which calls on the `certutil` command:

```
certutil -setreg ca\ValidityPeriodUnits 2 certutil setreg ca\ValidityPeriod "Years"
net stop certsvc
net start certsvc
```

On an enterprise CA, the certificate validity period is set based on certificate template properties (from the General tab). The lifetime specified in version 1 certificate templates cannot be changed. The lifetime specified in version 2 certificate templates can easily be changed using the Certificate Templates snap-in. Most certificate templates have one-year lifetimes. Exceptions are the CEP Encryption, Enrollment Agent, IPSec, and Web Server templates that have two-year lifetimes and the EFS Recovery Agent, Root CA, and Subordinate CA templates that have five-year lifetimes.

Windows certificate lifetimes support *nested* validity dates. This means that a certificate can never have a lifetime that is longer than the certificate lifetime of its issuing CA. For example, if the CA's certificate is about to expire in 13 months, and the default certificate lifetime is two years, the CA will issue certificates with a one-year lifetime. The CA administrator should remember to renew the CA certificate early enough not to restrict the lifetime of newly generated certificates.

Within a PKI hierarchy, the lifetime of entities' certificates will differ, depending on the level at which the entity is located in the hierarchy. The higher the entity is in the hierarchy, the more security features will be implemented to safeguard its private key. Remember that CA private-key compromise at a higher level in a hierarchy has much more impact than lower in the hierarchy. Also, consider the nesting-validity-dates feature of Windows PKI. Because an issuing CA's certificate is part of the certificate chain of all certificates it issues, its own CA certificate should be valid for any of the issued certificates to be valid. Thus, the CA's certificate must under all circumstances have a lifetime that is longer than the lifetime of the certificates it issues. We will come back to this topic in the next chapter, which is about Windows PKI design.