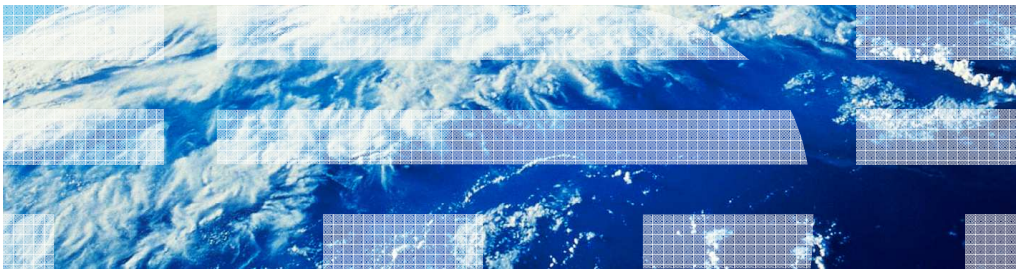
z/OS Communications Server - Security

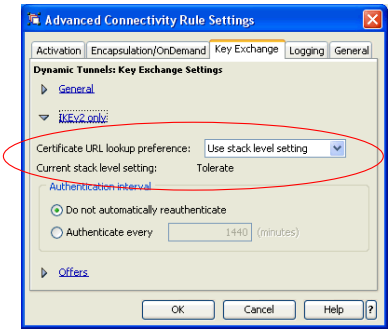
IPSec support for certificate trust chains and certificate revocation lists



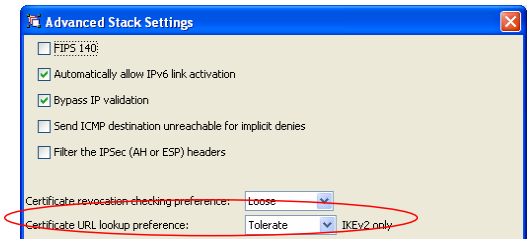
There are many new security functions in z/OS V1R12 Communications Server. This presentation covers the new IPSec support for certificate trust chains and certificate revocation lists.

Function externals: enabling or disabling HTTP lookup

- **CertificateURLLookupPreference** keyword of KeyExchangeAction
  - *Allow* send and request URLs
  - *Tolerate* send URLs, don't request
  - *Disallow* do not send or request



- Also available on KeyExchangePolicy
  - Sets the stack level default

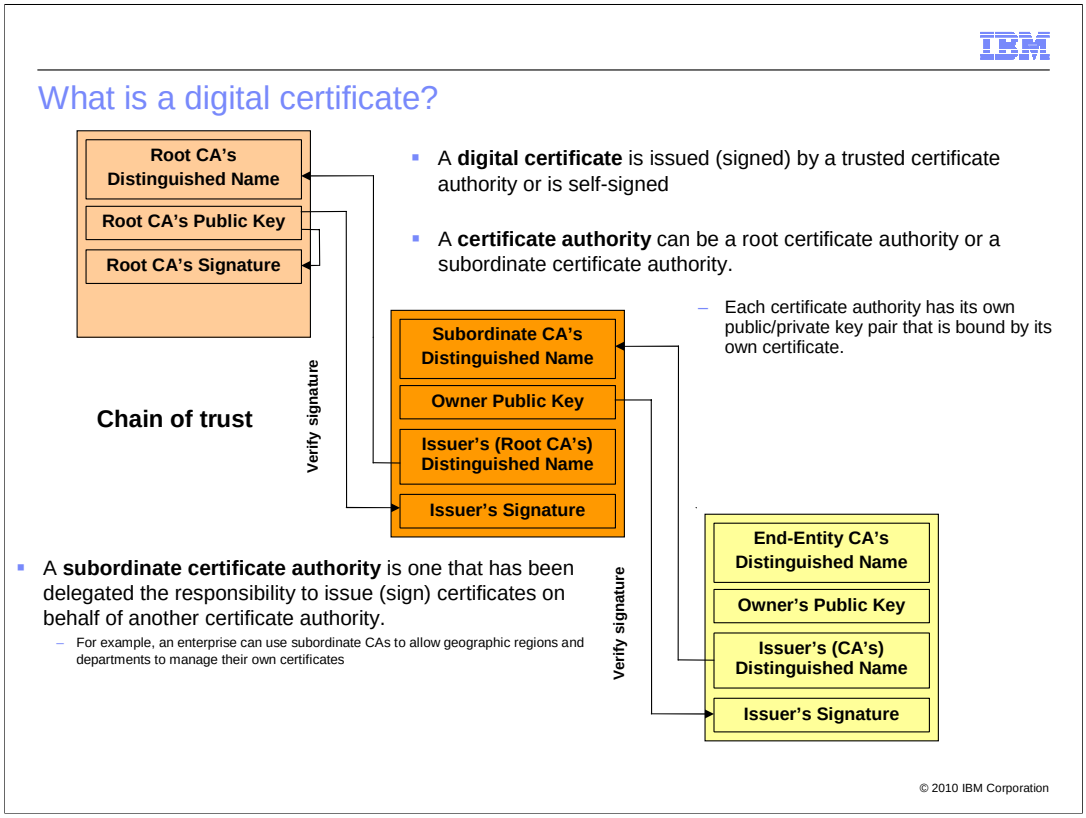


To enable HTTP lookup of certificates and bundles, code the **CertificateURLLookupPreference** keyword of the KeyExchangeAction or KeyExchangePolicy.

If you code Allow, that means IKED will send hash and URL encoded certificate and certificate request payloads to peer nodes, and will request that peer nodes use hash and URL encoding when sending to IKED.

If you code Tolerate, then IKED will send hash and URL encoded certificate and certificate request payloads to peer nodes, but will prevent peer nodes from sending hash and URL encodings in its requests to IKED. This is the default. It gives the system administrator the greatest control over which HTTP servers z/OS will attempt retrieval from, by limiting HTTP retrieval attempts to those URLs that are encoded in the local NSSD configuration file. See the next chart for details.

The CertificateURLLookupPreference option can be specified on a per-IKE-peer basis, and the stack-wide default value can be specified for each stack. These options are also available through the Configuration Assistant.

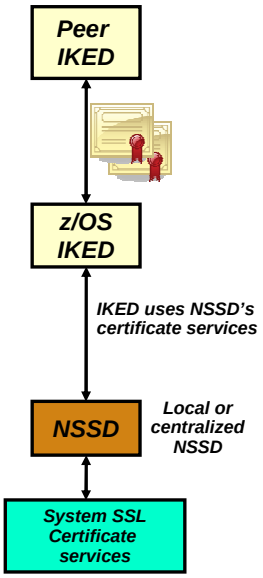


A digital certificate is issued and signed by a certificate authority or can be self-signed. The **certificate authority** (CA) can be the root (originating) authority or a subordinate authority. Each certificate has a public/private key pair that is bound to its identity (the name of a person, company or an IP address). A **subordinate authority** is one that has been delegated the responsibility to issue certificates on behalf of another certificate authority. An example is for an enterprise to use subordinate CAs to allow geographic regions to manage their own certificates. This can reduce the cost and time required to issue a new certificate.

When a certificate authority issues a certificate it signs the certificate using its private key. The public key of the certificate authority can then be used to verify it was the one that issued the certificate.

A certificate trust chain starts with the certificate that signed the end entity certificate (certificate that identifies the entity) and includes all signing certificates up to and including the trusted certificate authority (which is called the root).

NSS IPSec certificate services

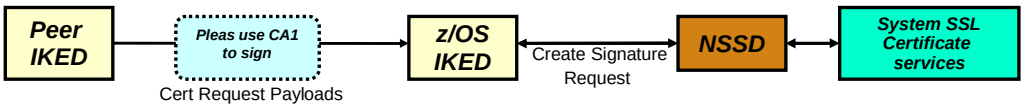


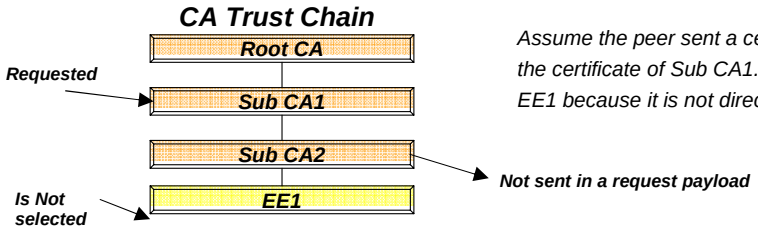
- The IKE protocol for digital signature authentication requires the peers to exchange digital signatures
  - The certificates used to sign them are exchanged between IKE peers in certificate payloads in the IKE messages
- z/OS IKED must perform digital signature creation and verification
  - Can be performed locally by IKED for RSA signatures
  - Can request signature creation and verification from NSS
- Network Security Services (NSS) IPSec discipline certificate services
  - Create a digital signature for an end entity using a certificate, to be sent to a peer node
  - Verify a digital signature using its signing certificate, received from a peer node

The IKE protocol for digital signature authentication requires the peers to exchange digital signatures. IKED will create a signature to send to the peer and verify the signature received from the peer. The certificates used to create and verify a digital signature can be exchanged between IKE peers using the certificate payloads in the IKE messages. IKED can perform the creation and verification of signatures for RSA signatures locally or it can request NSS to create and verify the signatures using its IPSec certificate services.

Certificate request payload processing

- A certificate request payload is a **request** to use a certificate **signed** by the CA whose distinguished name is contained in the payload
- For IKEv2, the CA is the public key hash in the payload





Assume the peer sent a certificate request with the certificate of Sub CA1. IKED will not select EE1 because it is not directly signed by CA1

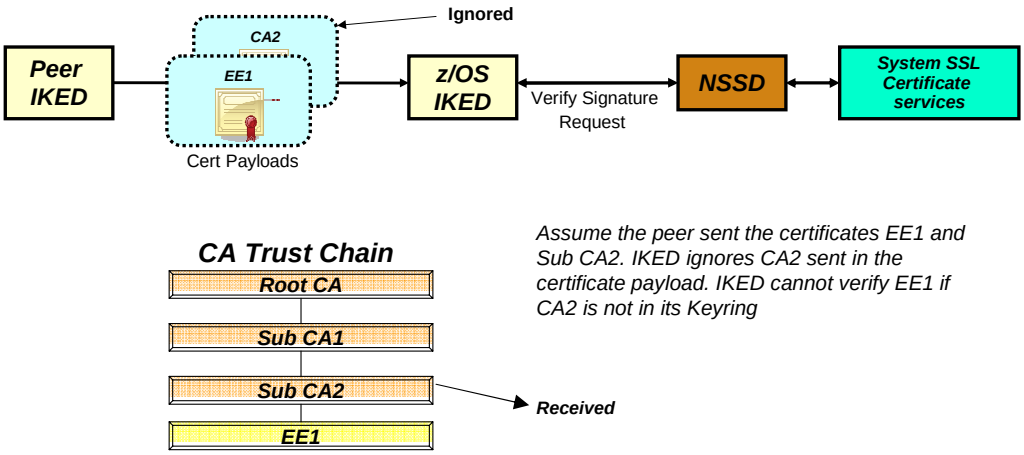
Currently certificate request payloads are treated as a **request** to use an end entity certificate in IKED's or NSSD's keyring. This certificate is **signed** by the CA whose distinguished name or public key hash (IKEv2) is contained in the payload. This does not match the expectations of RFC 4306 and RFC 4945 for certificate request payloads.

For IKEv1, these RFCs state that the payloads be treated as a request to use a certificate **within the trust chain** of the CA whose Distinquished Name (DN) is contained in the payload.

For IKEv2, these RFCs state that the payloads be treated as a **hint** to use a certificate **within the trust chain** of the CA whose public key hash is contained in the payload.

Certificate payload processing

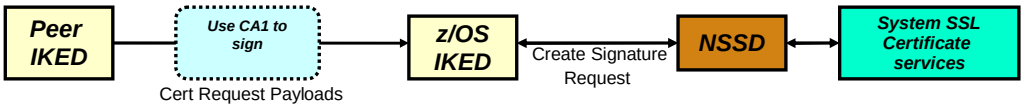
- Currently IKED ignores all certificate payloads, except for the first payload containing an end entity certificate

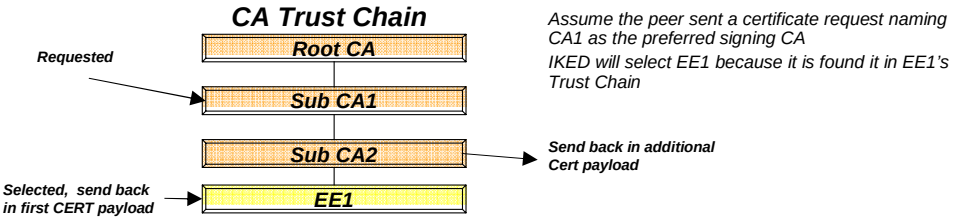


Currently IKED also ignores all certificate payloads, except for the first payload containing the end entity certificate when it needs to verify the Peer's ID. Again this does not match the expectations of RFC 4306 and RFC 4945. These RFCs state that the first certificate payload contain an end-entity certificate and any other certificate payloads contain CA certificates that IKED might not have in the trust chain.

Certificate request processing

- When receiving certificate request payloads for finding a certificate to create a signature
  - Treat the certificate request payloads as a **hint (IKEv2 only)** to use a certificate **within the trust chain** of the CA
  - Continue to use as a **request** for IKEv1





Assume the peer sent a certificate request naming CA1 as the preferred signing CA

IKED will select EE1 because it is found it in EE1's Trust Chain

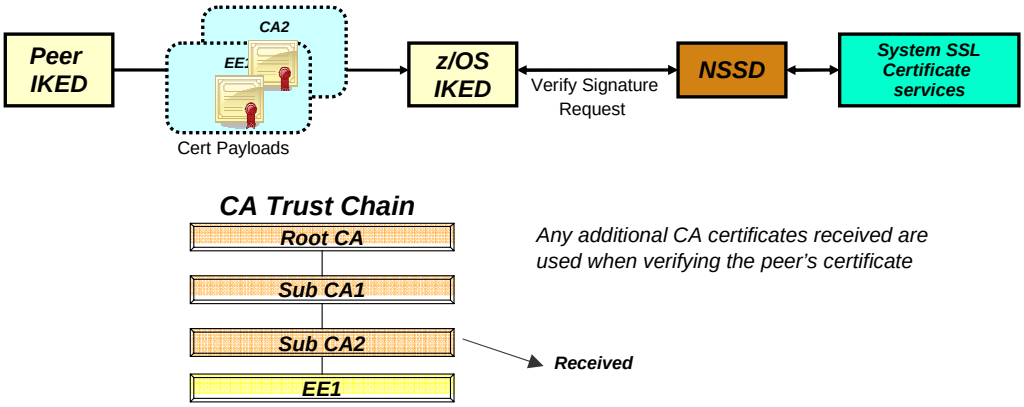
For IKEv1, IKED using NSSD certificate services will treat the payload as a request to select a End Entity (EE) certificate within the trust chain of the CA whose DN is contained in the certificate payload.

For IKEv2, IKED using NSSD certificate services will treat the payload as a hint to select a EE certificate within the trust chain of the CA whose public key hash is contained in the certificate payload.

In the example, IKED will look for a EE certificate that has CA1 in its trust chain. It finds EE1 because it is signed by Sub CA2 and Sub CA2 is within the trust chain of Sub CA1. It sends EE1 along with Sub CA2 back to the peer in certificate payloads. It sends Sub CA2 because it is possible the peer only knows about Root CA and Sub CA1.

Certificate payload processing

- When receiving certificate payloads to use in verifying a signature.
  - The first certificate payload contains an end-entity certificate
    - It is the certificate used to create the signature
  - Use any other certificate payloads received as containing CA certificates that are part of the CA trust chain



A peer might send certificate payloads that contain supporting certificate authority (CA) certificates. In the example IKED received certificates for EE1 and Sub CA2. Since EE1 is in the first certificate payload it is used as the certificate the peer used to create its signature. The certificate Sub CA2, sent in an additional certificate payload, is used when verifying that EE1 is a valid certificate. Before V1R12 Sub CA2 needed to be in the IKED/NSSD keyring in order to be used to validate EE1.

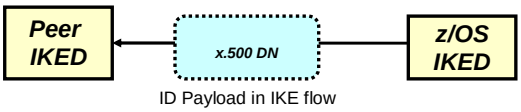
New ID processing requirements

- RFC 4945 provides additional requirements relative to how the ID in a certificate must be used
  - When the local ID is an X.500 distinguished name
    - the ID payload in the IKE message flow MUST be populated with the content of the end-entity certificate's Subject field
  - When the Peer sends an IP address in the ID payload of the IKE message flow
    - MUST be capable of verifying the IP address contained in the ID payload
      - is in the SubjectAltName extension of the peer's certificate
      - is the same as the peer's source address in the IP header

RFC 4945 introduces requirements that are not currently met by IKED. They deal with how the ID payload should be populated when the local ID is an x.500 Distinquished Name (DN) and additional checking that should be performed when the remote ID is an IPv4/6 address.

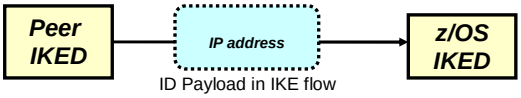
Changes to ID payload processing

- IKED populates the local ID payload using the LocalSecurityEndpoint policy statement
  - X.500 distinguished name might not be a binary match with the end-entity certificate's SubjectName



ID Payload in IKE flow

- IKED does not perform any validation of the IP address when the remote ID is an IP address



ID Payload in IKE flow

Currently IKED populates the local ID payload using the identity configured on the LocalSecurityEndpoint policy statement. When the local ID is an X.500 distinguished name, it might not be a binary match with the certificate's SubjectName field. According to RFC 4945, the ID should be populated from the end-entity certificate's SubjectName extension.

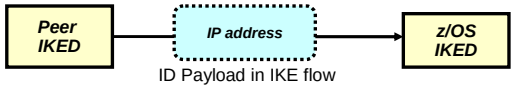
Currently IKED does not perform any validation of the IP address when the remote ID is an IPv4 or IPv6 address. RFC 4945 states that IP address validation should be done although a local option can be provided for it to be disabled.

Enhance IKED ID processing

- IKED is updated to populate the local identity using the binary subject distinguished name field from the signing certificate when:
  - using digital signature authentication and
  - the local identity type is ID_DER_ASN1_DN (X.500 distinguished name)

IKED is updated to validate a remote security endpoint's ID when the ID is an IPv4 or IPv6 address.

- The validation can be optionally disabled using the new BypassIdValidation parameter on the KeyExchangePolicy and KeyExchangeAction policy statements.
- Applies to:
  - IKEv1 and IKEv2 protocols
  - Preshared key and Digital Signature authentication



ID Payload in IKE flow

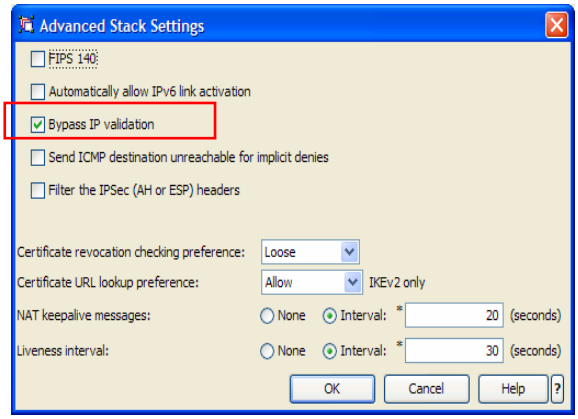
With this release IKED populates the local identity using the binary subject distinguished name field from the signing certificate when using digital signature authentication and the local identity type is ID_DER_ASN1_DN (the X.500 distinguished name).

With this release IKED validates that the remote peer's identity matches the peer's remote IP address when the identity of the peer is an IP address.

Configuring BypassIpValidation

- KeyExchangePolicy statement
  - Indicates whether a check is made to verify that the remote peer's identity matches the peer's remote IP address.
    - If the conditions is not valid then the security association setup is aborted
  - Default is the check is not enforced
- KeyExchangeAction statement.
  - Allowable values are the same as on the KeyExchangePolicy statement
  - KeyExchangeAction statement overrides the KeyExchangePolicy statement
  - Default value is the value from KeyExchangePolicy statement

- The KeyExchangePolicy BypassIpValidation option is set from the Configuration Assistant



The BypassIpValidation parameter can be specified on the KeyExchangePolicy and the KeyExchangeAction statements. The default is set to No to adhere to RFC rules.

On the KeyExchangePolicy statement, the new **BypassIpValidation** parameter indicates whether a check is made to verify that the remote peer's identity matches the peer's remote IP address. If the conditions are not valid then the security association setup is stopped. Allowable values are Yes, which means bypass the check, and No which means the check is enforced. No is the default.

The Configuration Assistant supports the new KeyExchangePolicy BypassIpValidation parameter in the IPSec Perspective, on the Advanced Stack Settings panel.

The GUI will set BypassIpValidation to **yes** for policies migrated from a older release. The default is **no** when generating new policies.

The KeyExchangeAction BypassIpValidation option is set from the Configuration Assistant, IPSec Perspective, Additional Settings, *Advanced Connectivity Rule Settings* panel (which is not shown).

Display command example: pasearch

- BypassIpValidation option reported from the KeyExchangePolicy statement

```
pasearch -c -vk
TCP/IP pasearch CS V1R12                Image Name: TCPCS2
  Date:                 01/28/2010      Time:  14:53:24
  PAPI Version:         9               DLL Version:  9

IPSec Policy Object:
  ConfigLocation:       Local           LDAPServer:       False
  CommonFileName:
  ImageFileName:        /etc/ipsec/V1RC/TSCO2110/S22110
. . .
  IpSecEnabled IPv4:    True            IpSecEnabled IPv6: True
  IpSec3DESEnabled:     True            IpSecAESEnabled:   True
  IpSecAESGCM16Enabled: True
  UpdateInterval:       1800
  InstanceId:           1264707486
  LastPolicyChanged:    Thu Jan 28 14:38:06 2010
  KeyExchange Policy Object:
   Configured:          True
   AllowNat:            No              NatKeepAliveIntvl: 20
   HowToInitiate:       Main            LivenessInterval:  30
   BypassIpValidation:  Yes             CertURLLookupPref: Allow
   RevocationChecking:  Loose
  Policy updated:       Thu Jan 28 14:38:06 2010
```

- BypassIpValidation option also reported from the KeyExchangeAction statement (not shown)

Use the pasearch command with the –c –vk options to report the value of the RevocationChecking option specified on the KeyExchangePolicy statement.

Certificate revocation

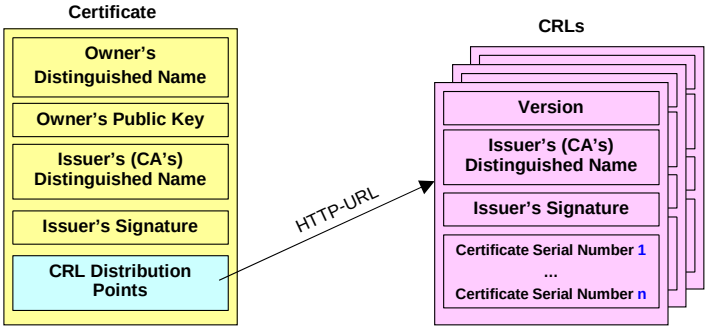
- A Certificate Revocation List is a list of certificates that have been revoked or are no longer valid.
- Common reasons a certificate might be revoked:
  - Private key has been compromised
  - Termination of an affiliation
    - Employment
    - Membership
  - Certificate no longer valid for stated purpose

- Certificate revocations should be taken into account when checking the validity of a certificate

Here are a few common reasons why a certificate can be revoked. In general certificate revocation information should be consulted when validating a certificate; however, consulting revocation information can have performance implications..

Certificate revocation lists

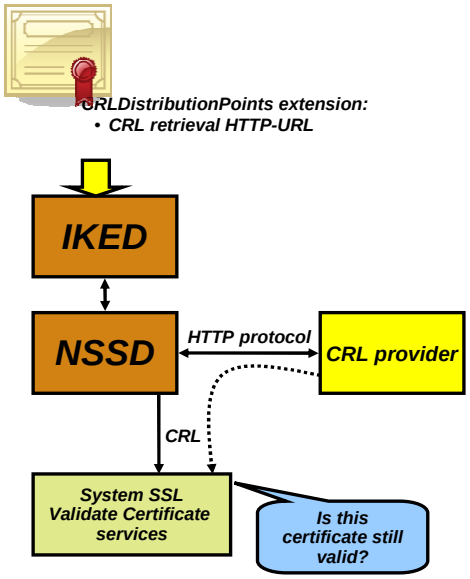
- Certificate Revocation Lists (CRLs)
  - A signed data structure that identifies revoked certificates
  - A direct CRL is signed by the certificate authority that issued the revoked certificates
  - An indirect CRL is signed by a trust authority other than the certificate authority that issued the revoked certificate
  - The signed data includes a list of serial numbers of revoked certificates and the reason for being revoked



Certificate revocation lists (CRLs) are one method to obtain certificate revocation information. The contents of a CRL are signed. Both certificates and CRLs originated as part of the ITU-T's X.509 protocol. The IETF PKIX working group created a profile for X.509 certificates and CRLs. This profile is applicable to IKE.

The x.509 certificate contains one or more CRL Distribution point extensions. As least one of these extension contains a HTTP URL pointing to a CRL. Revoked certificates are listed by their serial numbers in the CRL.

IPSec support for Certificate Revocation Lists (CRLs)

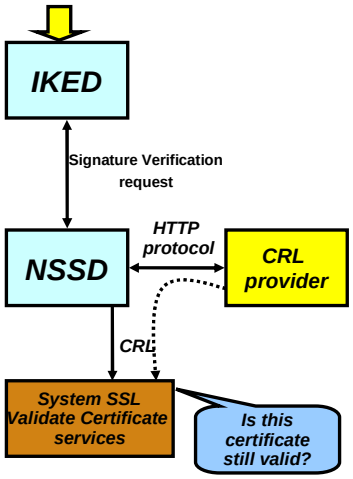


- When IPSec authenticates a digital signature, it needs to ensure the signing certificate is still valid
- NSSD will retrieve CRLs using information in the CRLDistributionPoints extension in a certificate
  - HTTP-URLs only
  - NSSD will not support retrieval of CRLs from LDAP servers
- NSSD will pass CRLs to System SSL
- System SSL will validate the certificate against the CRL
  - To ensure the certificate is still valid
    - Has not expired or been revoked
- IKED depends on NSSD for this function

IPsec must insure that all certificates are valid when it verifies a signature sent in the IKE flow. Support was added to NSSD to retrieve certificate revocation lists (CRLs) referenced in the certificate's CRLDistribiutionPoint extention. The retrieved CRLs along with the certificate trust chain are passed into System SSL when validating a EE certificate in order to insure the certificates have not expired or been revoked.

NSSD and IKED support for CRLs

- Supports the ability to perform CRL based revocation checking when verifying a signature
  - CRL checking is not performed when creating a signature
- Obtains the URL of a certificate's CRL from the CRLDistributionPoints extension
  - For each certificate NSSD uses the first distribution point that contains a HTTP URL
  - Scope (reason) of the CRL must be global
- If a certificate contains neither a CRLDistributionPoint nor a HTTP URL to a direct CRL for all reasons,
  - A CRL is not retrieved
  - If a CRL from a certificate cannot be retrieved then NSSD looks for a CRL in a X.509 certificate bundle provided by IKED
  - NSSD does not support any other CRL retrieval methods described by the RFC



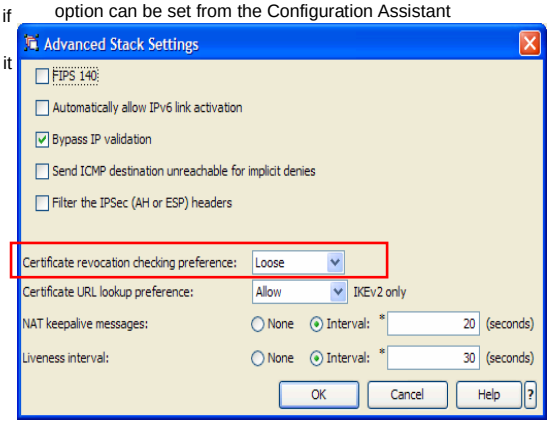
The NSSD provides support to check revocation information using CRLs residing in an HTTP repository. CRLs are obtained by NSSD using a certificate's HTTP CRL distribution point or a CRL in a certificate bundle. IKED must be configured as a network security client to exploit CRL checking.

NSSD only retrieves the CRL if the scope (reason) of the CRL is global. The CRL is global if the reason is not specified in the CRL distribution points extension or it is specified with a value of X'FF".

Certificate revocation checking preference

- KeyExchangePolicy statement
  - new RevocationChecking parameter values
    - **None** – No revocation checking
    - **Loose** (default) – Check revocation information if it can be obtained; otherwise, assume valid
    - **Strict** – Always check revocation information. If it cannot be obtained assume not valid
- KeyExchangeAction statement.
  - new RevocationChecking parameter
    - Allowable values are the same as on the KeyExchangePolicy statement
    - Default value is value from the KeyExchangePolicy statement
    - KeyExchangeAction statement overrides the KeyExchangePolicy statement

- The KeyExchangePolicy RevocationChecking option can be set from the Configuration Assistant



The RevocationChecking parameter can be specified on the KeyExchangePolicy and the KeyExchangeAction statements.

On the KeyExchangePolicy statement, the new RevocationChecking parameter has three possible values. **None** means no revocation checking. **Loose** means check the revocation information if it can be obtained; otherwise, assume it is valid. **Strict** means always check revocation information. If it cannot be obtained, assume it is not valid. The default value is **Loose.**

The Configuration Assistant supports the new KeyExchangePolicy RevocationChecking parameter in the IPSec Perspective, on the Advanced Stack Settings panel.

The Configuration Assistant also supports the new KeyExchangeAction RevocationChecking parameter in the IPSec Perspective, on the connectivity rules Advanced Connectivity Rule Settings panel. This panel can be found by selecting the IPSec Perspective, select the required connectivity rule, then select the AdditionalSettings tab and press the Advanced button.

pasearch

- RevocationChecking option reported from the KeyExchangePolicy statement

```
pasearch -c -vk
TCP/IP pasearch CS V1R12                Image Name: TCPCS2
  Date:                 01/28/2010      Time:  14:53:24
  PAPI Version:         9               DLL Version:  9

IPSec Policy Object:
  ConfigLocation:       Local           LDAPServer:       False
  CommonFileName:
  ImageFileName:        /etc/ipsec/V1RC/TSCO2110/S22110
. . .
  IpSecEnabled IPv4:    True            IpSecEnabled IPv6: True
  IpSec3DESEnabled:     True            IpSecAESEnabled:   True
  IpSecAESGCM16Enabled: True
  UpdateInterval:       1800
  InstanceId:           1264707486
  LastPolicyChanged:    Thu Jan 28 14:38:06 2010
  KeyExchange Policy Object:
   Configured:          True
   AllowNat:            No              NatKeepAliveIntvl: 20
   HowToInitiate:       Main            LivenessInterval:  30
   BypassIpValidation:  Yes             CertURLLookupPref: Allow
   RevocationChecking:  Loose
  Policy updated:       Thu Jan 28 14:38:06 2010
```

- RevocationChecking option also reported from the KeyExchangeAction statement (not shown)

Use the pasearch command with the –c –vk options to report the value of the RevocationChecking option specified on the KeyExchangePolicy statement.

Use the pasearch command with the –a –vk options to report the value of the RevocationChecking option specified on the KeyExchangeAction statement, which is not shown on this page.

Supported IKED configurations for new functions

The table below summarizes the new function added to support Certificate Revocation List processing and identifies the supported IKED configurations

Function	IKEv1 Local	IKEv1 NSS Client	IKEv2 NSS Client
Locate a certificate within a CA's trust chain		✓	✓
Send missing CA certificates to a peer		✓	✓
Use received CA certificates when validating a peer's signature		✓	✓
For IKEv2, treat as having an empty certificate request payload.			✓
Optional IP address check	✓	✓	✓
Populate x.500 DN from certificate	✓	✓	✓
Certificate Revocation Lists		✓	✓

The table shows by protocol (IKEv1 or IKEv2) and configuration which certificate revocation lists are supported.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_SecurityCert.ppt

This module is also available in PDF format at: ../SecurityCert.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, and z/OS are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.