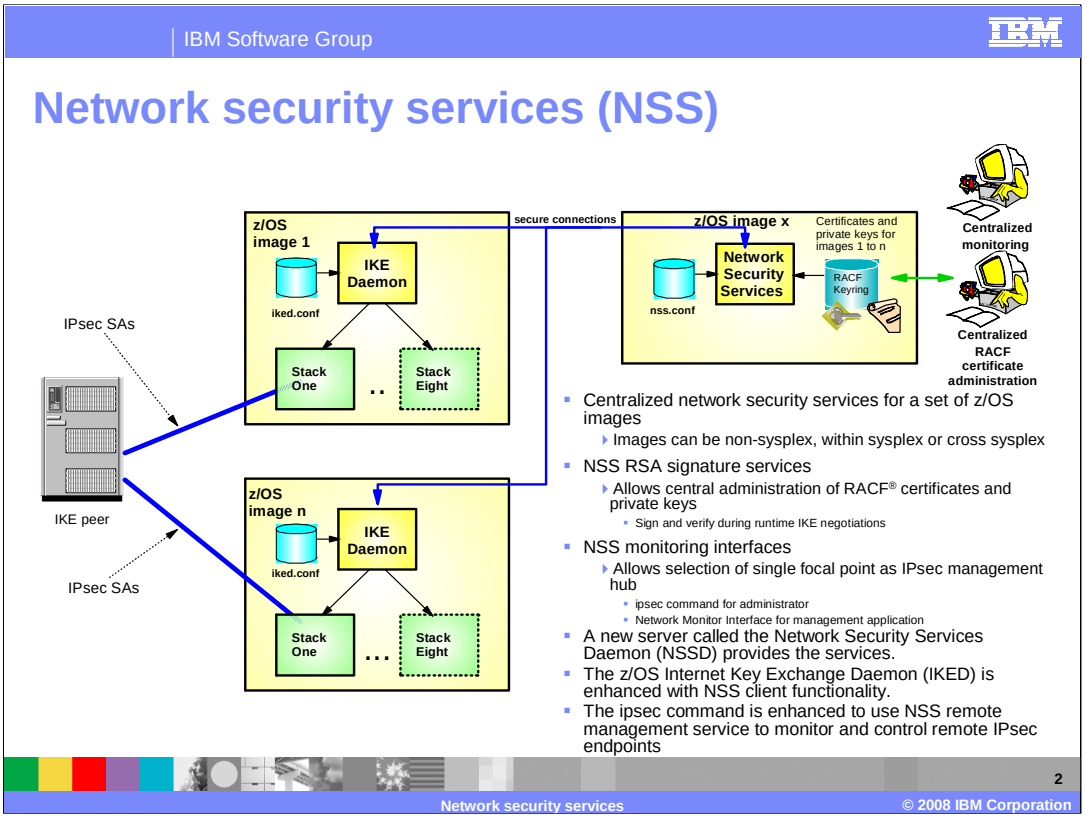
This presentation discusses the Network Security Services function in the z/OS V1R9 Communications Server.



Network Security Services (NSS) centralizes the sensitive keying material that would otherwise need to reside in less secure zones of the network onto a single location in the most secure zone of the network. In addition, NSS allows for centralized configuration and administration of certificates.

Network Security Services provide centralized certificate services, monitoring and management for IPSec security across z/OS systems within and across sysplexes. Network Security Services allow IPSec certificates to be kept in a single location, rather than having them reside on each z/OS node. The z/OS V1R9 Communications Server IKE daemon is enhanced so that it can be configured to act as a Network Security client. Configuration is on a per-stack basis, such that each NSS-enabled stack will appear to the Network Security Server as an independent client. For TCP/IP stacks that are not configured to use Network Security Services, the IKE daemon will continue to manage certificates out of a local keyring.

Specifically, NSS provides a central SAF-enabled repository for RSA certificates along with signature services within the most trusted zones. It eliminates the need to distribute certificates to security endpoints. NSS centralizes and reduce configuration and deployment complexity, especially when used along with Centralized Policy Services. It offloads digital signature operations from IKE daemon (the NSS client) and it enables monitoring and management of remote IPSec endpoints through the ipsec command and a network management programming interface.

The network security services (NSS) server provides a set of network security services for IPSec. These include the certificate (and digital signature) service and the network management service. The certificate service and network management service are used by NSS clients. When an NSS client uses the NSS certificate service, the NSS server creates and verifies RSA signatures on the behalf of the NSS client using RSA certificates that are stored only at the NSS server. When an NSS client uses the network management service, the NSS server routes IPSec network management interface (NMI) requests to that NSS client, which enables the NSS client to be managed remotely. The NSS client provides the NSS server with responses to these requests.

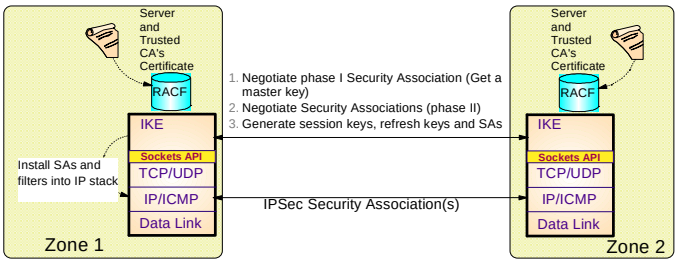
As mention earlier, the IKE daemon can be configured to act as an NSS client on behalf of multiple TCP/IP stacks. A separate connection is maintained to the server for each NSS-enabled TCP/IP stack, so each TCP/IP stack appears as a separate NSS client to the NSS server. The -z option of the ipsec command or the IPSec NMI can be used to manage NSS clients that use the NSS network management service. For details about using the ipsec command to manage NSS clients, see *z/OS V1R9 Communications Server: IP System Administrator's Commands*. For details about using the IPSec NMI to manage NSS clients, see *z/OS V1R9 Communications Server: IP Programmer's Guide and Reference*.

Background: Internet key exchange (IKE)

- Internet Key Exchange (IKE) protocols allow for dynamic, secure key exchange in support of the establishment of IPSec Security Associations (SAs).
- IKE processing involves creation and verification of digital signatures
- In RSA signature mode, signature operations require access to RSA certificates and keys
- Security endpoints may exist in different zones of the network, some more trusted than others

This slide presents the key background points necessary to understand NSS and its role.

Background: SA negotiation using IKE



- IKE negotiates security associations
  - Phase 1 – used by IKE to protect IKE flows
  - Phase 2 – used by IPSec to protect data flows
- Automatically generates keys and non-disruptively refreshes SAs and session keys
- Authentication methods
  - RSA signature mode
  - Pre-shared key (less interesting to NSS solution)

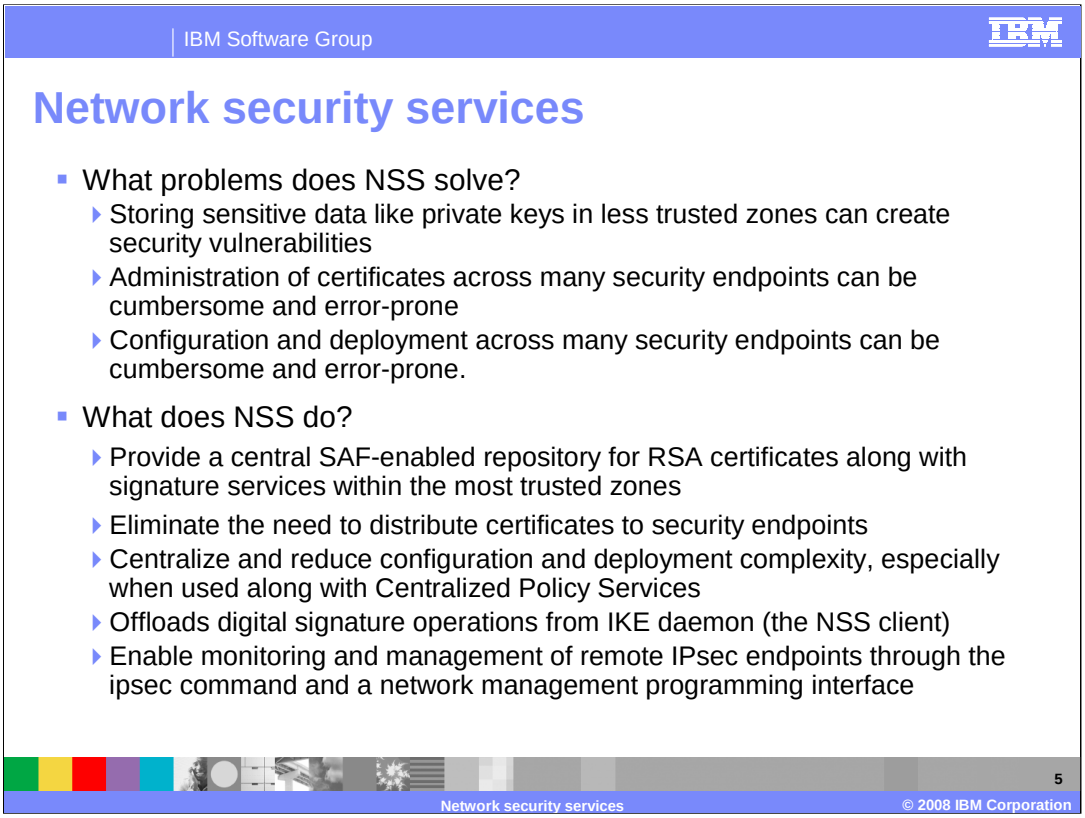
This slide illustrates some of the main points of key distribution using IKE in RSA signature mode, which is where NSS comes into play. NSS is not concerned at all with pre-shared key mode because RSA signature mode is a much more scalable approach.

As a general note, each security endpoint, constituted by an IKE agent, may reside in a separate zone of the IP network. Some zones will be more trusted than others. For example, in this diagram, Zone 1 may be within a highly secure enterprise data center, while Zone 2 may be outside of any corporate firewall and connected to the “open Internet.”

The establishment of IKE tunnels, also called IKE Phase 1, requires RSA signature authentication. To accomplish this, the security endpoints must have access to RSA private and Certificate Authority certificates that can be used to carry out the signature authentication.

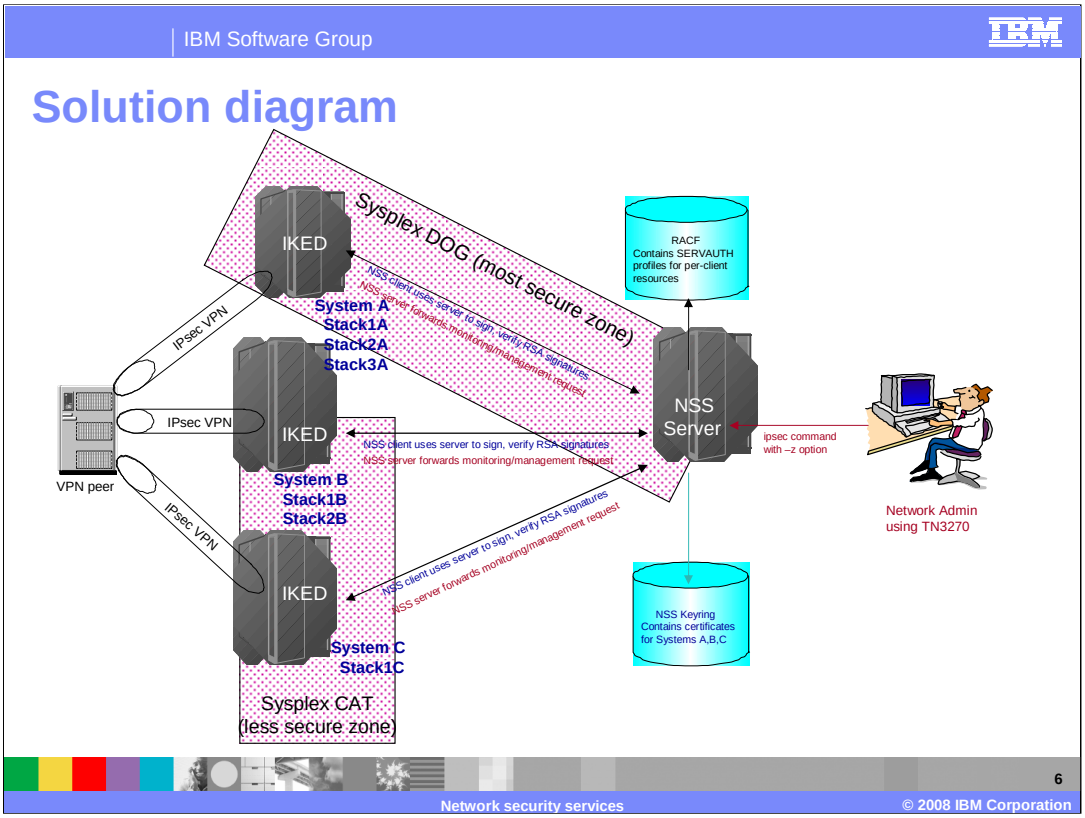
Once the IKE (phase 1) tunnel is established, other protocols are used to generate unique keys for each IPsec (also called Phase 2) tunnel. This phase does not require any further RSA signature operations.

Since each IKE and IPsec tunnel has a limited lifetime (either based on time or amount of data flowed), non-disruptive tunnel refreshes take place over time. For IKE tunnels, this once again requires RSA signature operations.



NSS centralizes the sensitive keying material that would otherwise need to reside in less secure zones of the network onto a single location in the most secure zone of the network. In addition, NSS allows for centralized configuration and administration of certificates.

These are the key functions that NSS provides.



This diagram shows a basic NSS setup involving a single NSS server and several clients. The clients are z/OS IKE daemons. Each IKE daemon can act as an NSS client for up to eight TCP/IP stacks (that is, one for each stack running on that system). In this case:

•z/OS SystemA has three stacks that are enabled for NSS services

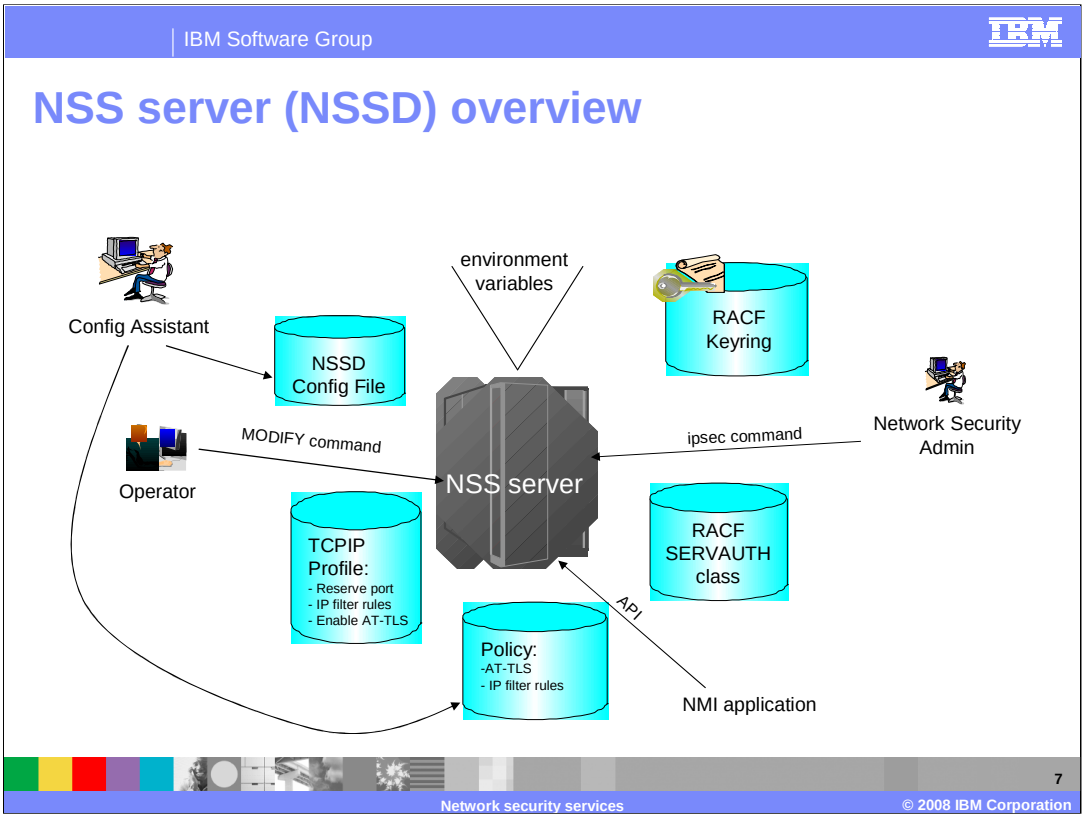
•z/OS SystemB has two NSS-enabled stacks

•z/OS SystemC has one NSS-enabled stack

Note that any of the above z/OS systems may have additional TCP/IP stacks – they just aren't enabled for NSS. Also note that multiple IKE daemons can simultaneously access network security services. These IKE daemons do not have to reside within the same sysplex as the NSS server. In this example, you have two different sysplexes: Sysplex DOG, which contains the NSS server system and SystemA. This sysplex resides in the most secure zone of the network (in a highly secured data center). And Sysplex CAT, which contains systems SystemB and SystemC. This sysplex resides in a less secure zone.

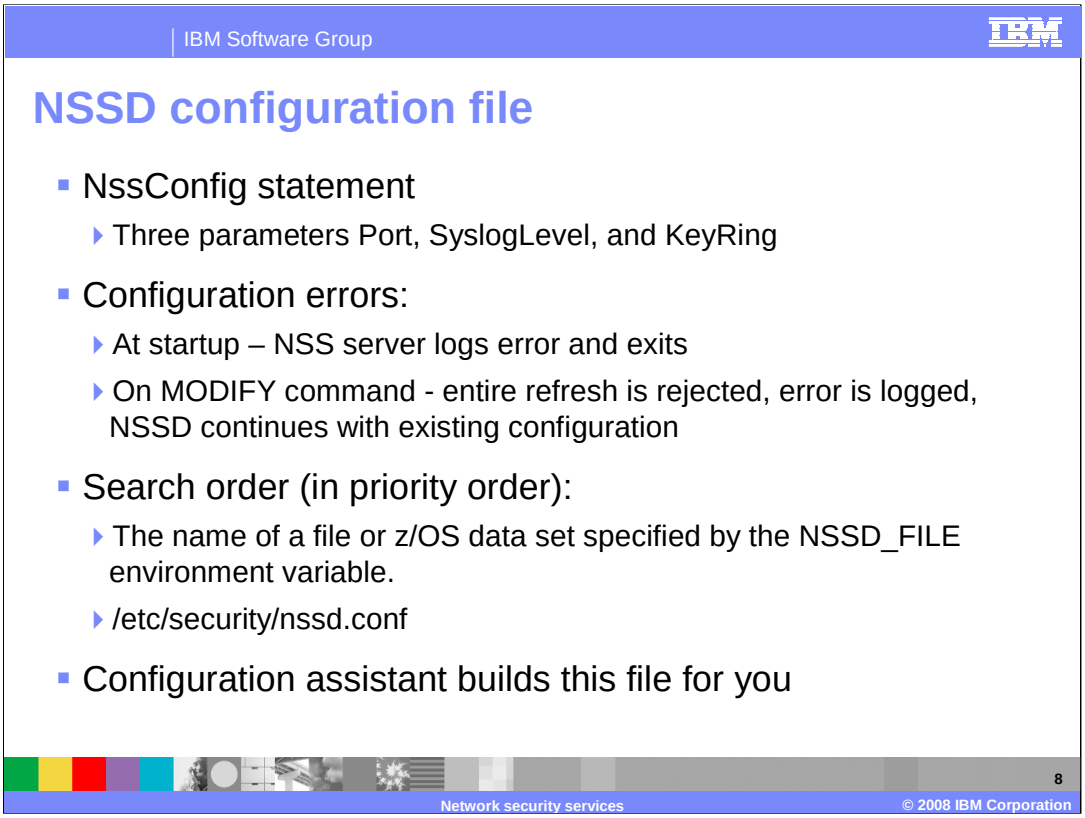
The NSS server is a central repository and signature authority for its clients (represented by blue text in the diagram). The NSS certificate service enables an NSS server to perform RSA signature and verification operations on behalf of its NSS clients. The certificate service is used by an NSS client during a phase 1 negotiation when RSA signature authentication is required. Certificates and private keys for all NSS clients are stored on a single key ring. The NSS server must have access to this key ring and must have access to the certificates and private keys on this key ring. When providing certificate services, the NSS server consults SERVAUTH profiles to verify that an NSS client is authorized to access the certificates involved.

The server also acts as a focal point for IPSec management and monitoring of its clients (represented by red text in the diagram). The NSS network management service enables an NSS server to request IPSec monitoring data from its NSS clients. In addition, the network management service enables the server to make IPSec control requests to its clients. Control requests include the ability to activate, deactivate, or refresh a security association, and to switch between default IP filter policy and IP security filter policy. The ipsec command and the IPSec network management interface (NMI) can be used to direct the NSS server to make such requests from an NSS client.



The NSS server's externals consist of the following:

1. The network security services server – a UNIX daemon named NSSD
2. NSS daemon configuration file – defines the runtime characteristics of the NSS server
3. TCP/IP Profile:
   Reserve the NSS server port (default is 4159)
   Default IP filtering policy can be updated to allow NSS client/server traffic
4. Relevant policy:
   AT-TLS policy to protect the TCP connections between NSS clients and the server
   IP traffic descriptors to allow NSS traffic between client systems and the server system
5. RACF SERVAUTH profiles to control client access to NSS services and to private and Certificate Authority certificates
6. RACF keyring that contains all of the certificates that are available to any of the NSS clients
7. A small set of environment variables that define the location configuration resources
8. The Configuration Assistant program allows you to configure your NSS server (and clients) through a graphical user interface and it then generates the NSSD configuration file and the necessary AT-TLS policy and IP filter rules for NSS client-server traffic. Note that the Configuration Assistant also generates the NSS client configuration in the IKED configuration file (not shown in this server-side-only diagram)
9. The MODIFY command, available to the z/OS operator, to refresh or display the NSSD configuration
10. The ipsec command, available to UNIX shell users, to control and monitor remote NSS clients and the NSS server itself
11. A network management programming interface



The form of the NSSD configuration file is similar to that of the IKE daemon. All of the parameters are contained within the NssConfig statement, enclosed by curly braces. Three parameters are available. The **Port** parameter identifies the TCP port to which the NSS server will bind. The default value is 4159, which is an IANA registered port. All client requests must come in through this TCP port. The **SyslogLevel** parameter specifies the level of logging to obtain from the NSS server. The default value is 1 which is the minimal NSS daemon syslog output. The **KeyRing** parameter specifies the SAF key ring database, which contains certificates and keys used when creating and verifying signatures for NSS clients. There is no default value for this parameter. NSS certificate services are not activated if the KeyRing parameter is not specified.

The NSS server logs an error and exits when configuration errors are detected during startup. When errors in the configuration file, that is being refresh as a result of the Modify command, are detected then the error is logged and the refresh is rejected.

NSSD configuration file example

```
NssConfig
{
  # listen on standard port
  Port 4159

  # minimum tracing
  SyslogLevel 1

  # specify NSS keyring
  KeyRing nssd/keyring
}
```


This is an example of a typical NSSD configuration file. In this example the NSS Server is listening on the IANA-registered port **4159**. The minimum syslog tracking level has been requested. The SAF key ring database containing certificates and keys that is used when creating and verifying signatures for NSS clients is **nssd/keyring**.

NSSD keyring

- Single keyring for NSS server
- All client and Certificate Authority certificates must reside on this keyring
- SERVAUTH profile must be set up for each certificate to permit client access to that certificate

  That is, EZB.NSSCERT.*sysname.clientname*.IPSEC.HOST
- A separate keyring may contain server identity certificate used in AT-TLS handshake. This is specified in the AT-TLS configuration.

The NSS server's key ring serves a similar purpose as the IKE daemon's key ring. It contains certificates that are used in the process of creating and verifying signatures that are exchanged during RSA signature authentication. A personal certificate or site certificate contained on the NSS server's key ring represents the identity of an NSS client, whereas a certificate contained on the IKE daemon's key ring represents a local stack's identity. See chapter 18 of the *z/OS V1R9 Communications Server; IP Configuration Guide* for more details.

Certificates for all NSS clients must reside on this one key ring. The same commands that are used to create and manage the IKE daemon's key ring also apply to the NSS server's key ring. For examples of how to create and manage the IKE daemon's key ring, see the *z/OS V1R9 Communications Server; IP Configuration Guide* Appendix E, "Steps for preparing to run IP security."

You must create a SERVAUTH resource profile for each NSS client certificate that is added to the NSS server's key ring. For details, see chapter 18 of the *z/OS V1R9 Communications Server; IP Configuration Guide*.

NSSD external security manager setup

- Several SERVAUTH profiles control access to NSS server services and certificates
- NSS Clients can use passtickets to authenticate to the server. If passtickets are used:
  - Permit NSSD's user ID to the BPX.DAEMON FACILITY class if that class is already defined (if it's not, no need to define it)
  - Secure signon function must be enabled and at least one profile must be created for the NSS server.
  - Configure PTKTDATA class profiles (profile name is NSSD)
    - ✓ Secure keys are defined in the PTKTDATA profiles
    - ✓ Time of day must be synchronized

    Example:

```
RDEFINE PTKTDATA NSSD SSIGNON(KEYMASKED(E001193519561977)) UACC(NONE)
```


This slide describes the External Security Manager (ESM) setup required for NSS. The name of the SERVAUTH profiles contains a *sysname* which is the name of the z/OS system on which NSSD is running. Some profile names contain a *clientname* which is the symbolic name of the NSS client. A *mappedlabelname* is also present in some profile names and that is the mapped version of a certificate label.

EZB.NSS.*sysname.clientname*.IPSEC.CERT controls whether a NSS client can register with the NSS server for the NSS certificate service. To allow access, the user ID under which the NSS client registers must be permitted READ access to this profile.

EZB.NSS.*sysname.clientname*.IPSEC.NETMGMT controls whether a NSS client can register with the NSS server for NSS network management service. To allow access, the user ID under which the NSS client registers must be permitted READ access to this profile.

EZB.NSSCERT.*sysname.mappedlabelname.*CERTAUTH controls whether a NSS client can access a given CERTAUTH certificate on the NSS server's key ring. This profile controls access to a single certificate, identified by the mappedlabelname. To allow access, the user ID under which the NSS client registers must be permitted READ access to this profile.

EZB.NSSCERT.*sysname.mappedlabelname.*HOST controls whether a NSS client can access a given PERSONAL or SITE certificate on the NSS server's key ring. This profile controls access to a single certificate, identified by the mappedlabelname. To allow access, the user ID under which the NSS client registers must be permitted READ access to this profile.

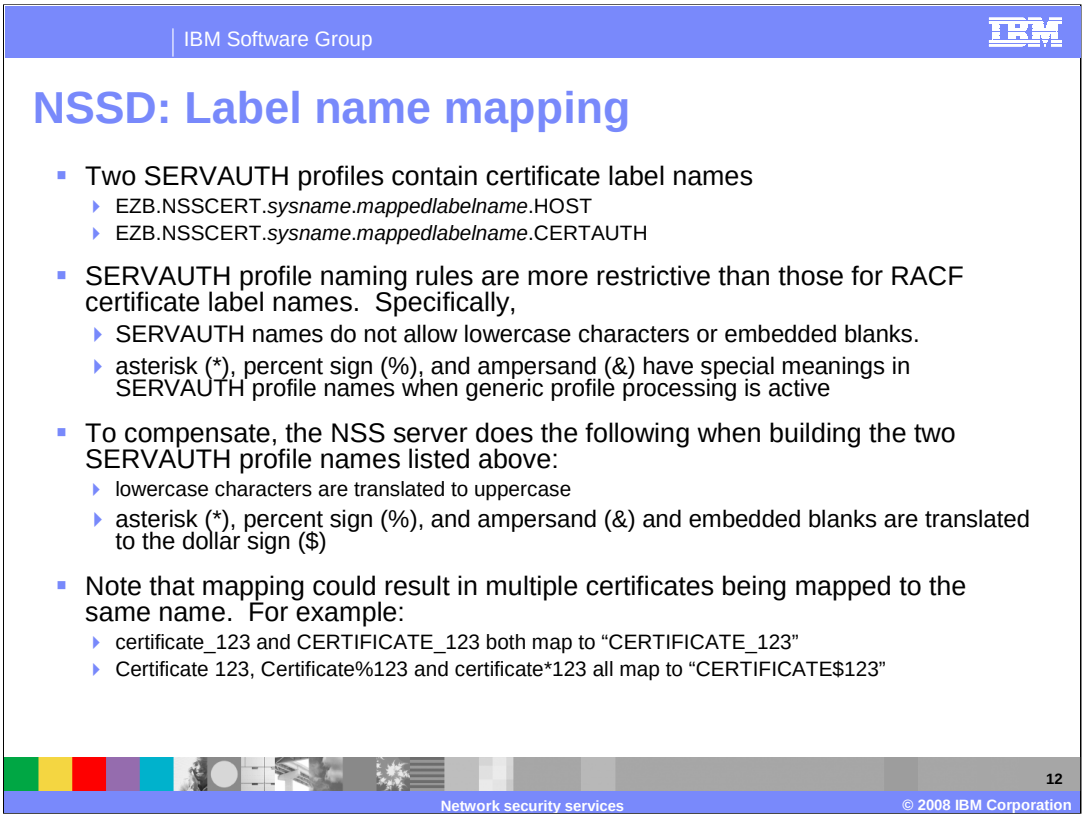
EZB.NETMGMT.*sysname.clientname*.IPSEC.DISPLAY controls whether a z/OS user can issue NMI monitoring requests to the NSS server on behalf of a NSS client (that is. GET_xxx requests) or issue the ipsec command with the -z option to perform a display action for a NSS client (that is. display options) . To allow access, the user ID under which the NMI application or the ipsec command will run must be permitted READ access to this profile.

EZB.NETMGMT.*sysname.clientname*.IPSEC.CONTROL controls whether a z/OS user can issue NMI management requests to the NSS server on behalf of a NSS client (for example, activate/deactivate requests) or issue the ipsec command with the -z option to perform a management action to a NSS client (for example, activate/deactivate options). To allow access, the user ID under which the NMI application or the ipsec command will run must be permitted READ access to this profile.

EZB.NETMGMT.*sysname.sysname*.NSS.DISPLAY controls whether a z/OS user can issue NMI requests to display connections to the NSS server or issue the ipsec command with the -x option to display connections to the NSS server. To allow access, the user ID under which the NMI application or the ipsec command will run must be permitted READ access to this profile.

If passtickets are used to authenticate clients, then the NSS user ID must be permitted to the BPX.DAEMON class and the secured sign-on function of RACF must be enabled (by activating the PTKTDATA class). To store the application key in the local external security manager database, a PTKTDATA profile must be created. This key must be associated with an application ID of NSSD.

For more information on secure sign-on setup for NSS, refer to chapter 18 of the IP Configuration Guide. For specific information about enabling the secure sign on function and defining profiles to be used by the single sign on function, refer to the z/OS Security Server RACF Security Administrator's Guide.



During the processing of certificate operations, the NSS server validates that an NSS client is authorized to access the certificates required to complete the operation. The NSS server consults SERVAUTH profiles to perform this validation. The profile names consulted by the NSS server are dynamically constructed by the NSS server using the following information:

•The system name on which the NSS server is running

•The label of the certificate that is used during a certificate operation

•The certificate operation that is being performed:

- •When processing a request to create a signature, the format of the profile that is consulted is EZB.NSSCERT.*sysname*.*mappedlabelname*.HOST.
- •When processing a request to obtain a list of CA certificates, the format of the profile consulted is EZB.NSSCERT.*sysname*.*mappedlabelname*.CERTAUTH.

The NSS server creates a mapped label name using the following algorithm:

•All lowercase alphabetic characters in a certificate’s label are changed to uppercase. This is necessary because the class descriptor table for the SERVAUTH profile permits only uppercase profile names.

•The asterisk (*), percent sign (%), and ampersand (&) are replaced by a dollar sign ($). This is necessary because these characters have special meaning when generic profile processing is active.

•All embedded blanks are also replaced by a dollar sign ($). This is necessary because blanks are not allowed in SERVAUTH profile names.

Note that the administrator of the NSS server must define profiles using the mapped label names generated by this algorithm. When the certificate’s label name contains lowercase characters, the administrator must change each lowercase character to uppercase. When the certificate’s label name contains the characters *, %, &, or a blank character, the administrator must replace each occurrence with a dollar sign ($) character.

Using this algorithm, it is possible that multiple certificates can result in the same mapped name, as shown on the slide.

NSSD: Policy and TCP/IP profile

- Relevant policy
  - IP filtering policy to allow NSS client/server traffic
  - AT-TLS policy to protect NSS client/server traffic
    - ✓ NSS client-server TCP connections **must** be protected with AT-TLS
    - ✓ AT-TLS policy must be provided for each stack through which NSS clients will communicate with the NSS server.
    - ✓ NSS server provides a certificate to prove its identity to client during initial handshake
    - ✓ NSS client uses server's certificate to verify the server's identity
  - Configuration assistant can set this up for you
- TCP/IP Profile
  - Optionally update PORT statement to reserve the NSS server listening port
  - Optionally add default IP filter rules to allow NSS client/server traffic even when configured policy (via Policy Agent) is not loaded

The NSS server communicates with NSS clients using the TCP protocol. The NSS server binds to all stacks using INADDR_ANY. IP filters rules must be defined to permit NSS client/server traffic for any IP security stacks that contain an interface to which the NSS client will connect. In addition to the default IP filters, the configured policy which is delivered through the z/OS Policy Agent must also be updated to allow this traffic. Policy-based filters are in effect when a stack initializes with the Policy Agent or when the **ipsec -f reload** command has been issued. IP security filter policy is defined in Policy Agent configuration files. For details about defining IP security policy files, see the Policy Agent and policy applications chapter of *z/OS V1R9 Communications Server: IP Configuration Reference*. Note that the SourcePortRange value on the IpService statements must include the value specified on the port parameter of the NssConfig statement in the NSS server configuration file.

The NSS server and the IKE daemon require that communications between the NSS server and NSS clients be secured using AT-TLS. You must define AT-TLS rules to secure this communication. Enable AT-TLS processing for a stack by specifying the TTLS parameter on the TCPCONFIG statement in the TCP/IP profile. Specific AT-TLS policy is configured in Policy Agent configuration files. For details about enabling AT-TLS and configuring AT-TLS policy, see Chapter 19 of the *z/OS V1R9 Communications Server; IP Configuration Guide*. You should define AT-TLS policy such that only cipher suites requiring TLS encryption are exchanged with NSS clients. Failure to restrict the cipher suites to those requiring encryption can result in sensitive information flowing in the clear across an untrusted network. You must define AT-TLS policy for each stack through which the NSS server will communicate with an NSS client.

The NSS server acts as the server during an SSL handshake. To act in the server role of an SSL handshake, the NSS server must have access to a private key and certificate verifying its ownership of that private key. For information about creating and managing keys and certificates for servers using AT-TLS, see Appendix B of the *z/OS V1R9 Communications Server; IP Configuration Guide*. Note that NMI applications use AF_UNIX sockets, so AT-TLS protection does not apply to those connections.

By default the NSS server uses TCP port 4159, but this value is configurable using the Port parameter of the NssConfig statement in the NSS server configuration file. It is also a good idea to update the PORT statement in the TCP/IP profile to reserve the port that the NSS server will use when listening for client connections.

Default IP filter policy is defined in the TCP/IP profile. Updating default IP filter policy to permit communications between the NSS server and NSS clients is optional. Default IP filter policy is in effect only when IP security filter policy cannot be loaded or when the **ipsec -f default** command has been issued. For details about defining default IP filter policy in the TCP/IP profile, see *z/OS V1R9 Communications Server; IP Configuration Reference*. Note that the SRCport value in the filter rules must include the value specified on the port parameter of the NssConfig statement in the NSS server configuration file.

NSSD policy example: IP filtering

```
IpFilterRule            NssTrafficIPv4
  {
      IpSourceAddr             all4
      IpDestAddr               all4
      IpService
      {
         SourcePortRange        4159
         DestinationPortRange   1024 65535
         Protocol               tcp
         Direction              bidirectional InboundConnect
         Routing                local
      }
      IpGenericFilterActionRef  permit-nolog
  }
IpFilterRule            NssTrafficIPv6
  {
     IpSourceAddr             all6
     IpDestAddr               all6
     IpService
     {
        SourcePortRange        4159
        DestinationPortRange   1024 65535
        Protocol               tcp
        Direction              bidirectional InboundConnect
        Routing                local
     }
     IpGenericFilterActionRef  permit-nolog
  }
IpGenericFilterAction    permit-nolog
  {
     IpFilterAction           permit
     IpFilterLogging          no
  }
```


This is an example of a valid IP filter policy definition for use with NSS.

NSSD policy example: AT-TLS

```
TTLSRule                    NssRule
{
   LocalPortRange                   4159
   JobName                          NSSD
   Direction                        Inbound
   TTLSGroupActionRef               NSSGroup
   TTLSEnvironmentActionRef         NSSManager
}
TTLSGroupAction             NSSGroup
{
   TTLSEnabled                      On
}
TTLSEnvironmentAction       NSSManager
{
   TTLSKeyRingParms
   {
      Keyring                       NSSD/keyring
   }
   TTLSCipherParmsRef               RequireEncryption
   HandshakeRole                    SERVER
}
TTLSCipherParms             RequireEncryption
{
 V3CipherSuites             TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
 V3CipherSuites             TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
 .
 .
 .
 V3CipherSuites             TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5
 V3CipherSuites             TLS_RSA_EXPORT_WITH_RC4_40_MD5
}
```


This is an example of a valid AT-TLS policy definition for use with NSS.

NSSD: Other considerations

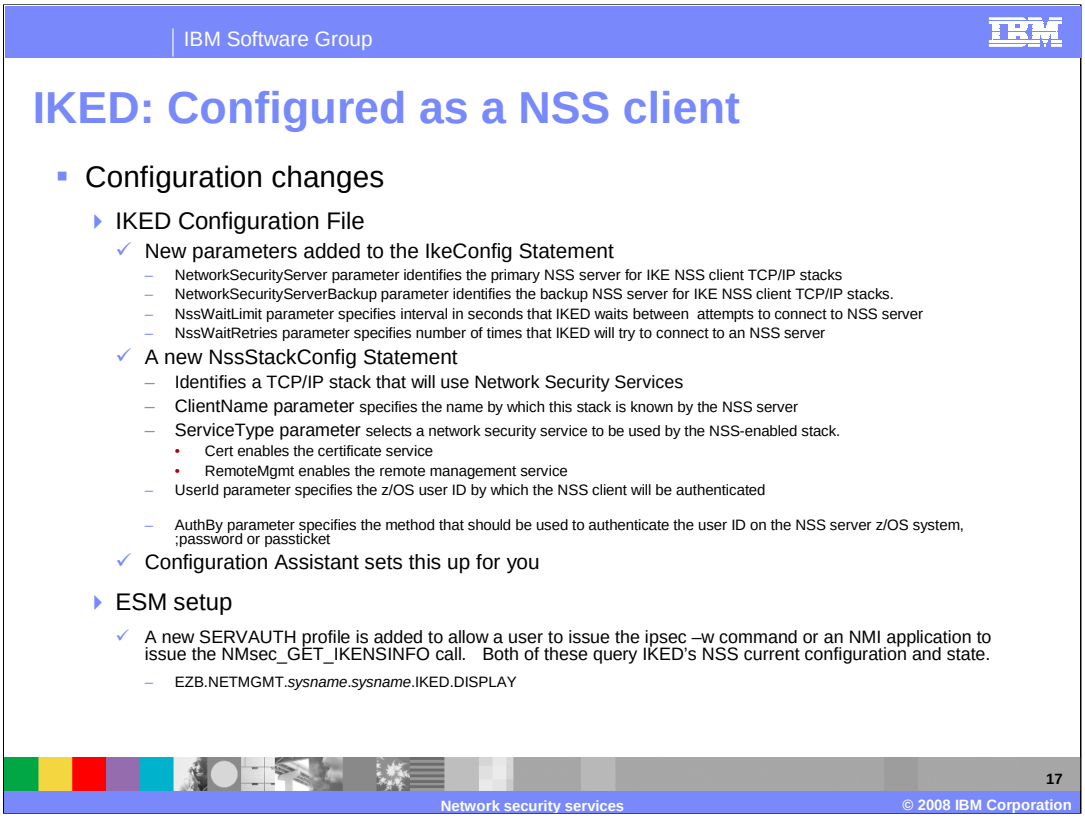
- MODIFY command
  - Display current NSSD configuration
  - Refresh the NSSD configuration
- Failover considerations
  - Sysplex-based failover
    - ✓Use a *non-distributed* dynamic VIPA (do not use a distributed DVIPA)
    - ✓Backup NSS must be on another sysplex node
    - ✓Transparent to NSS clients
  - Explicit backup server
    - ✓Specified in client configuration (hence, not transparent)
    - ✓Upon lost connection to primary server, client will connect to backup
    - ✓Completely independent of any sysplex configuration
  - Any backup server must have compatible (or even identical) ESM definitions and certificates

The NSS server provides a modify command. This command can be used to display configuration values that are currently being used by the NSS server and to reread the configuration file. Use the MODIFY procname,DISPLAY command to display configuration values currently in use by the NSS server. Use the MODIFY procname,REFRESH command to reread the NSS server configuration file. *procname* is the member name of the cataloged procedure that is used to start the network security services server daemon (NSSD). For more information on the MODIFY command, see *z/OS V1R9 Communications Server; IP System Administrator's Commands*.

NSS clients can use the NSS certificate service when negotiating phase 1 security associations. Network monitoring applications can use the NSS network management service to display information about NSS clients. As such, the NSS server should be treated as an application that requires high availability. Take steps to quickly recover from an outage that impacts the NSS server's ability to respond to clients. Recovery configurations for the NSS server include the following:

For recovery of NSS server workload by another NSS server within a sysplex, configure NSS clients to connect to the NSS server on a non-distributed dynamic VIPA. TCP/IP stacks configured as backup for the dynamic VIPA must have the necessary external security manager definitions and certificates to support the NSS clients, and an NSS server must be running on the z/OS system hosting the TCP/IP stack configured as backup. Do not configure NSS clients to connect to a distributed DVIPA address on the NSS server. If a distributed DVIPA is used, the **ipsec** command and IPSec NMI can manage only NSS clients that have been distributed to the system on which the **ipsec** command is being run or the system on which the IPSec NMI is invoked.

Alternatively, an IKE daemon running as an NSS client can be configured to connect to a backup NSS server with the NetworkSecurityServerBackup parameter on the IkeConfig statement in the IKED.CONF file. When the IKE daemon is unable to connect to the primary NSS server, or when it loses its connection with the primary server, the IKE daemon attempts to connect to the server configured as backup. This recovery configuration can be used regardless of sysplex configurations. The backup server must be configured with all necessary external security manager definitions and certificates to support the NSS clients. For additional details about the IkeConfig statement, see *z/OS V1R9 Communications Server: IP Configuration Reference.*



Four new parameters are added to the IkeConfig statement in the IKE daemon configuration file. All of these are focused on NSS exploitation and apply to all stacks that use NSS through this IKE daemon.

The **NetworkSecurityServer** parameter identifies the primary NSS server for IKE NSS client TCP/IP stacks. A single server is used for all of the TCP/IP stacks configured as NSS clients. Stacks can be configured individually as NSS clients. Stacks with a corresponding NssStackConfig statement are treated as NSS clients; stacks without a corresponding NssStackConfig statement rely solely on local IKE resources. While the NetworkSecurityServer parameter is optional, you must specify at least one of the NetworkSecurityServer and NetworkSecurityServerBackup parameters in order for any of the TCP/IP stacks to use an NSS server. The value of the NetworkSecurityServer parameter can be changed using the MODIFY IKED,REFRESH command. However, existing connections remain in place and new connection attempts will use the new value.

The **NetworkSecurityServerBackup** parameter identifies the backup NSS server for IKE NSS client TCP/IP stacks. Network Security clients switch between the primary and the backup NSS servers whenever their current server becomes unresponsive. If both the primary and the backup become unresponsive, the Network Security client attempts to connect to the primary and the backup in a round-robin fashion until a successful connection is made.

The **NssWaitLimit** parameter specifies the number of seconds that a NSS client waits between connection attempts when trying to establish a connection with a NSS server. The product of the NssWaitLimit value multiplied by the NssWaitRetries value defines the maximum number of seconds that a NSS client attempts to connect to a NSS server before switching to another server.

The new statement, **NssStackConfig,** should be coded for each TCP/IP stack that will use NSS. Only stacks with a corresponding NssStackConfig statement are eligible for services provided by a NSS server. Stacks that are not configured with an NssStackConfig statement are locally managed. NssStackConfig statements require that a valid NSS server is described in the IkeConfig statement. It is a configuration error to have a NssStackConfig statement without also specifying a NetworkSecurityServer parameter, a NetworkSecurityServerBackup parameter, or both. You can use the MODIFY IKED,REFRESH command to change which TCP/IP stacks are configured as NSS clients, as follows:

•Deleting a NSS client: If it is determined after a refresh that a NssStackConfig statement was removed, then the connection associated with the removed NssStackConfig statement is closed

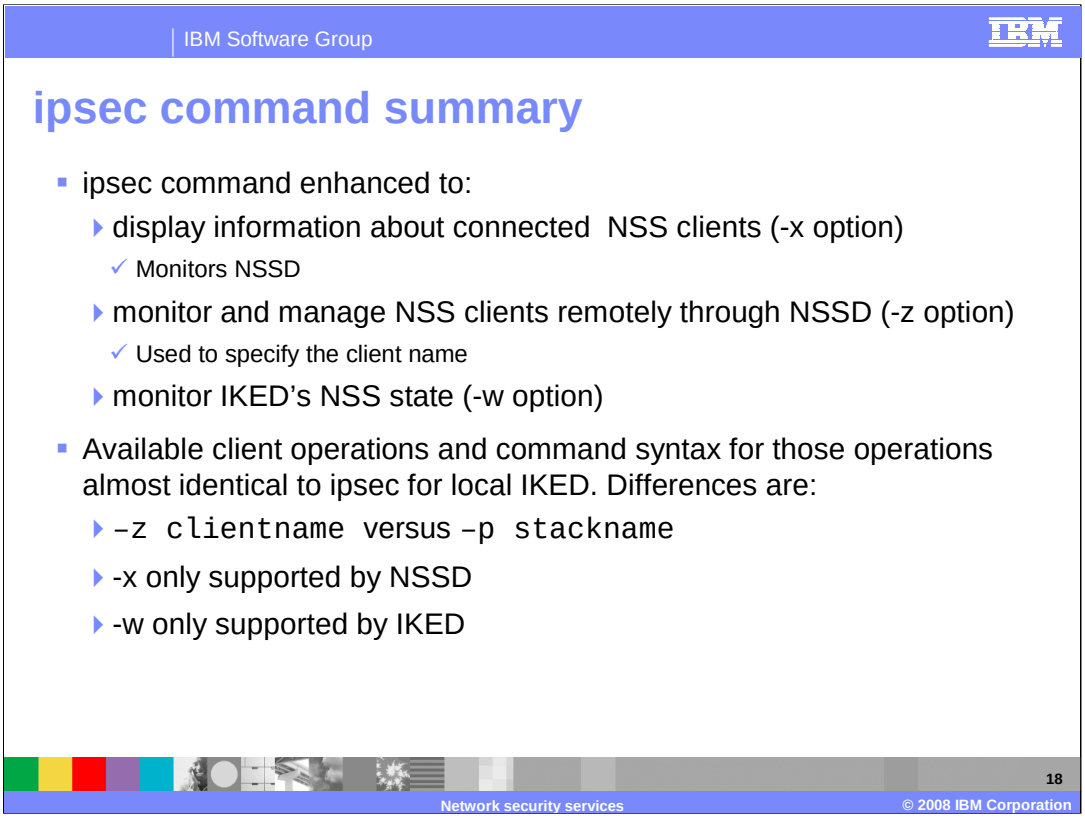
•Adding NSS client: If it is determined after a refresh that a new NssStackConfig statement was added, then the connection for the new stack is opened.

•Changing internal NssStackConfig values: Any change to an internal parameter of the NssStackConfig statement results in a disconnect followed by a reconnect.

The **ClientName** parameter, if not specified, is constructed by the IKE daemon from the z/OS system name and the TCP/IP stack name, as follows: *sysname_stackname.* Regardless of how name is established, it must match the clientname portion of associated SERVAUTH profiles on the NSS server.

The specified user ID, on the **UserId** parameter, must be defined on the z/OS system where the NSS server runs. Furthermore, the user ID must be granted read access to any of the associated SERVAUTH profiles that control access to network security services or certificates on the NSS keyring.

sysname in the EZB.NETMGMT.*sysname*.*sysname*.IKED.DISPLAY profile is the name of the z/OS system on which IKED is running.



The role of the ipsec command expands in z/OS V1R9 Communications Server to encompass NSS environments. You can use the -x primary option on the ipsec command to display connection information about NSS clients connected to the NSS server.

You can Use the -z option on the ipsec command to specify the name of an NSS client rather than a name of a local TCP/IP stack. When the -z option is specified, the ipsec command obtains information about the NSS client from the NSS server. The -z option is valid only on the system running the NSS server. The NSS client identified by the -z option must be connected to the NSS server. The NSS client must also be enabled to use the NSS network management service. The new –z option directs the ipsec command to a local NSS server, which will forward the request to the specified NSS client (assuming that client is currently connected to the server). Almost all of the existing ipsec options work with –z. The only exceptions are:

•-p, which directs the command to a local TCP/IP stack

•-x, which requests information about the local NSS server

•-w, which requests information about the local IKE daemon

Refer to *z/OS V1R9 Communications Server; IP System Administrator's Commands* for details and examples of each ipsec command option.

ipsec command –x option

```
ipsec -x display

CS V1R9 ipsec  NS Client Name: n/a  Mon Nov 27 12:40:02 2006
Primary:  NS Server       Function: Display           Format:   Detail
Source:   Server          Scope:    n/a               TotAvail: 1
SystemName: MVS052

ClientName:                     client4
StackName:                      TCPCS4
SystemName:                     MVS052
ClientIPAddress:                ::ffff:10.10.10.1
ClientPort:                     50003
ServerIPAddress:                ::ffff:10.10.10.99
ServerPort:                     4159
UserID:                         USER1
RemoteManagementSelected:       Yes
RemoteManagementEnabled:        Yes
CertificateServicesSelected:    Yes
CertificateServicesEnabled:     Yes
ConnectState:                   connected
TimeConnected:                  2006/11/27 12:37:08
TimeOfLastMessageFromClient:    2006/11/27 12:37:08
*******************************************************************************

1 entries selected
```


Here are a few noteworthy points regarding the output of this command.

The summary lines at the top are quite similar to that of most other ipsec command options. Nothing too exciting here.

The first several detail lines (ClientName through UserID) display the client identity and address information.

The next four lines describe the client configuration and the services that are actually enabled (per SERVAUTH profiles):

- RemoteManagementSelected indicates whether the client is configured to use the NSS network management service
- RemoteManagementEnabled displays “yes” when the client has selected this service and it is also permitted to the service per the governing SERVAUTH profile. Otherwise, “no” will appear.
- CertificateServicesSelected indicates whether or not the client is configured to use the NSS certificate service
- CertificateServicesEnabled displays “yes” when the client has selected this service and it is also permitted to the service per the governing SERVAUTH profile. Otherwise, “no” will appear.

The final few lines indicate attributes of the current client connection state.

ipsec command –z option

```
ipsec -y display -z client4

CS V1R9 ipsec  NS Client Name: client4  Mon Nov 27 12:44:35 2006
Primary:  Dynamic tunnel  Function: Display           Format:   Detail
Source:   Stack           Scope:    Current           TotAvail: 1

TunnelID:                    Y2
ParentIKETunnelID:           K1
VpnActionName:               Dvpn
LocalDynVpnRule:             mvs052_192
State:                       Active
HowToEncap:                  Tunnel
LocalEndPoint:               10.10.10.1
RemoteEndPoint:              10.10.10.2
LocalAddressBase:            10.10.10.1
LocalAddressPrefix:          n/a
LocalAddressRange:           n/a
RemoteAddressBase:           10.10.10.2
RemoteAddressPrefix:         n/a
RemoteAddressRange:          n/a
HowToAuth:                   AH
 AuthAlgorithm:              Hmac_Sha
 AuthInboundSpi:             2401615039
 AuthOutboundSpi:            1971620597
HowToEncrypt:                3DES
 EncryptInboundSpi:          4088723240
 EncryptOutboundSpi:         445063417
```


This slide shows an example using the -z option to display phase 2 security association information about the NSS client client4, where the name client4 was obtained from the previous ipsec -x display command. As you can see, the output looks exactly as it would if the same command were issued locally against the TCP/IP stack using the –p option. The only difference is in the summary header information that describes the target as an NSS client rather than simply a TCP/IP stack.

ipsec command –z option (continued)

```
Protocol:                   ALL(0)
LocalPort:                  0
RemotePort:                 0
OutboundPackets:            0
OutboundBytes:              0
InboundPackets:             0
InboundBytes:               0
Lifesize:                   0K
LifesizeRefresh:            0K
CurrentByteCount:           0b
LifetimeRefresh:            2006/11/27 14:09:19
LifetimeExpires:            2006/11/27 14:44:19
CurrentTime:                2006/11/27 12:44:35
VPNLifeExpires:             2007/03/07 12:44:19
NAT Traversal Topology:
  UdpEncapMode:             No
  LclNATDetected:           No
  RmtNATDetected:           No
  RmtNAPTDetected:          No
  RmtIsGw:                  n/a
  RmtIsZOS:                 n/a
  zOSCanInitP2SA:           n/a
  RmtUdpEncapPort:          n/a
  SrcNATOARcvd:             n/a
  DstNATOARcvd:             n/a
*******************************************************************************

1 entries selected
```


This slide contains the remainder of the ipsec –z output. This is the information that is normally shown when displaying phase 2 security associations.

ipsec command –w option

```
ipsec -w display

CS V1R9 ipsec  NS Client Name: n/a  Fri Nov 17 11:20:05 2006
Primary:  Stack NS        Function: Display           Format:   Detail
Source:   IKED            Scope:    n/a               TotAvail: 3
SystemName: MVS052

StackName:                      TCPCS
ClientName:                     n/a
NSServicesSupported:            No
RemoteManagementSelected:       No
RemoteManagementEnabled:        n/a
CertificateServicesSelected:    No
CertificateServicesEnabled:     n/a
NSClientIPAddress:              n/a
NSClientPort:                   n/a
NSServerIPAddress:              n/a
NSServerPort:                   n/a
NSServerSystemName:             n/a
UserID:                         n/a
ConnectionState:                n/a
TimeConnectedToNSServer:        n/a
TimeOfLastMessageToNSServer:    n/a
*******************************************************************************
StackName:                      TCPCS3
ClientName:                     client3
NSServicesSupported:            Yes
RemoteManagementSelected:       Yes
RemoteManagementEnabled:        Yes
CertificateServicesSelected:    Yes
CertificateServicesEnabled:     Yes
```


You can use the -w primary option on the ipsec command to query a local IKE daemon to determine which active stacks are configured as NSS clients, and their current status.

This slide illustrates the output of an ipsec –w command for a z/OS system that has three active TCP/IP stacks. Two of these (TCPCS3 and TCPCS4) are enabled for NSS, while the first one (TCPCS) is not. Here are a few noteworthy points regarding the output of this command:

- The summary lines at the top are quite similar to that of most other ipsec command options. Nothing too exciting here.
- The first two detail lines for each stack indicate the stack identity
- The next five lines describe the client configuration as well as the services that are actually enabled (per SERVAUTH profiles at the server):
  - NSServicesSupported indicates whether or not the IKE daemon itself is configured to use NSS.
  - RemoteManagementSelected indicates whether or not the client is configured to use the NSS network management service
  - RemoteManagementEnabled displays “yes” when the client has selected this service and it is also permitted to the service per the governing SERVAUTH profile. Otherwise, “no” will appear.
  - CertificateServicesSelected indicates whether or not the client is configured to use the NSS certificate service
  - CertificateServicesEnabled displays “yes” when the client has selected this service and it is also permitted to the service per the governing SERVAUTH profile. Otherwise, “no” will appear.
  - The remaining lines describe the client and server addresses and indicate attributes of the current client connection state

Note that output for each stack is separated by a line of asterisks

ipsec command –w option (continued)

```
NSClientIPAddress:              10.10.10.1
NSClientPort:                   50105
NSServerIPAddress:              10.10.10.3
NSServerPort:                   4159
NSServerSystemName:             MVS052
UserID:                         USER3
ConnectionState:                connected
TimeConnectedToNSServer:        2006/11/17 11:19:09
TimeOfLastMessageToNSServer:    2006/11/17 11:19:09
*************************************************************************
StackName:                      TCPCS4
ClientName:                     client4
NSServicesSupported:            Yes
RemoteManagementSelected:       Yes
RemoteManagementEnabled:        Yes
CertificateServicesSelected:    Yes
CertificateServicesEnabled:     Yes
NSClientIPAddress:              10.10.10.2
NSClientPort:                   50104
NSServerIPAddress:              10.10.10.3
NSServerPort:                   4159
NSServerSystemName:             MVS052
UserID:                         USER1
ConnectionState:                connected
TimeConnectedToNSServer:        2006/11/17 11:19:09
TimeOfLastMessageToNSServer:    2006/11/17 11:19:09
*************************************************************************


3 entries selected
```


This slide contains the remainder of the ipsec –w output.

NSS: NMI programming interface

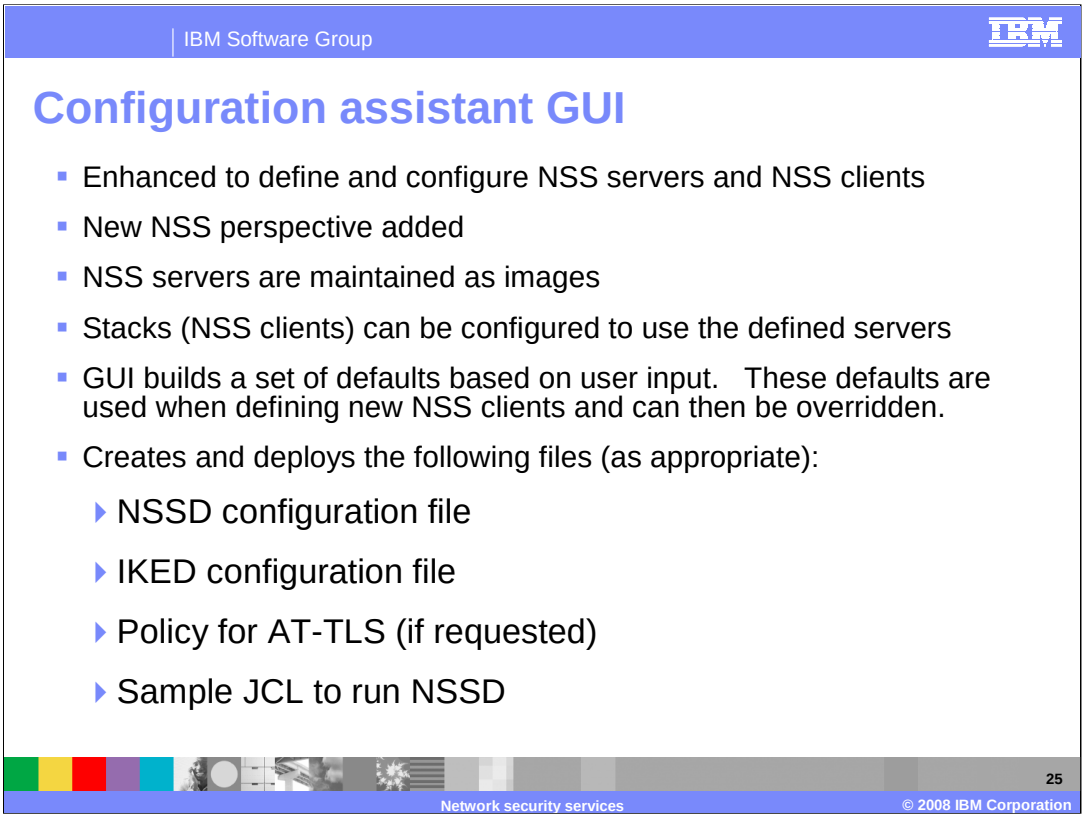
- Based on the IPSec Network Management Interface (NMI)
- Network management applications connect to an AF_UNIX listening socket
  - /var/sock/nss
- The NSS NMI supports 18 of the 20 calls that the IPSec NMI supports. Exceptions are
  - NMsec_GET_STACKINFO (summary description of TCP/IP stacks)
  - NMsec_GET_IKENSINFO (summary of IKED's NSS config and state)
- The NSS NMI supports one unique call
  - NMsec_GET_CLIENTINFO (summary description of connected NSS clients)
- One formatting difference
  - Applications connected to NSSD put the NSS clientname in the NMsMTarget field rather than a TCP/IP stack name.
- Up to 10 simultaneous network management clients are supported.

The NSS server supports a message format that is almost identical to that used by the IKE daemon for local IPSec monitoring and control. Like the local monitoring/control interface, these messages are exchanged over an AF_UNIX socket using a request-response model. NSSD's AF_UNIX socket is named /var/sock/nss.

The NSS server supports all of the request messages described for the IKE daemon except for the NMsec_GET_STACKINFO and NMsec_GET_IKENSINFO requests (see "Application interfaces for monitoring IP filtering and IPSec" on page 442). In addition, the NSS server also supports the NMsec_GET_CLIENTINFO request message. The NMsec_GET_CLIENTINFO request obtains a list of NSS clients that are currently connected to the NSS server and summary information about each client. This message does not allow a filtering record. If the NMsMTarget field in the message header is blank, then information for all of the currently connected clients is returned. If a client name is specified in the NMsMTarget field, then information for only that client is returned as long as the client is connected. If the specified client is not connected, then no records are returned in the response message. Access to this function is controlled through the EZB.NETMGMT.*sysname.sysname*.NSS.DISPLAY resource definition in the SERVAUTH class

The only difference between the NSS and IPSec NMI message format is that when an NMI message is sent to the NSS server, the NMsMTarget string in the message header identifies the remote NSS client to which the request is directed. Use the *clientname* field of the target NSS client in the NMsMTarget string, padded on the right with blanks. You can obtain the *clientname* values of each client connected to the NSS server by issuing the NMsec_GET_CLIENTINFO request. The NMsMTarget field can be set to blanks for an NMsec_GET_CLIENTINFO request. If this field is set to blanks for any other request, the request is rejected with an appropriate error code the reply header.

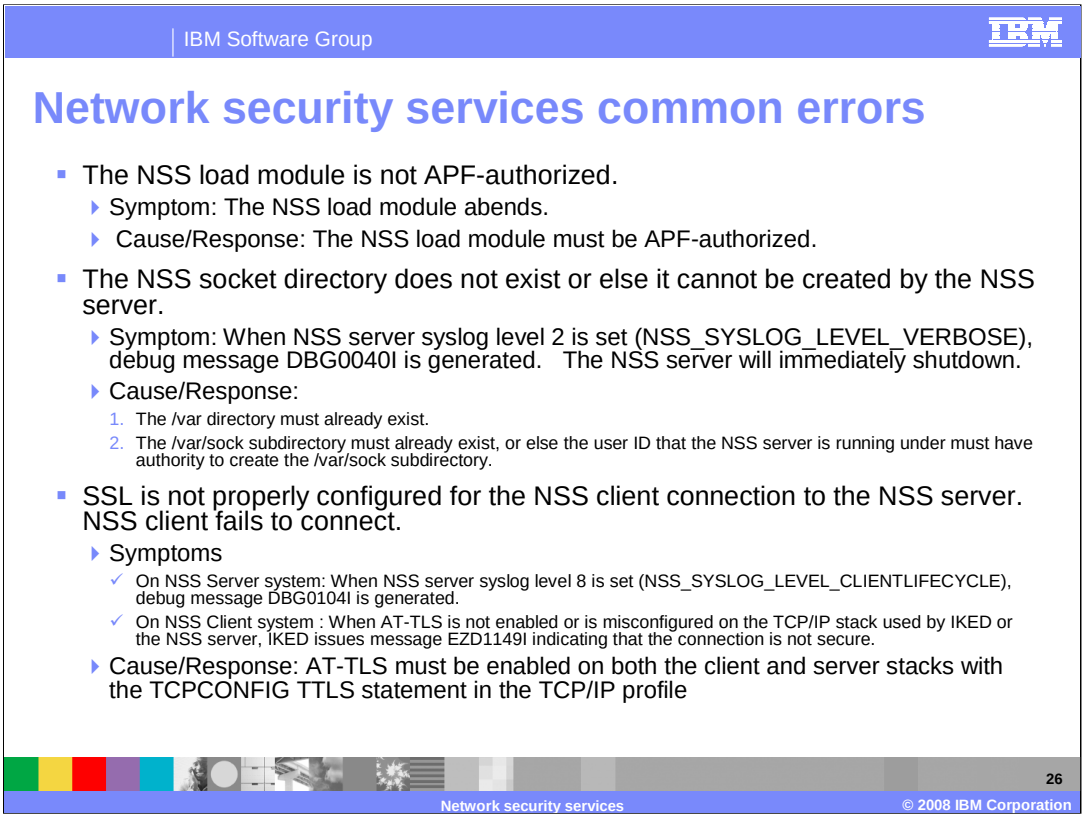
Message layouts are defined in SEZANMAC(EZBNMSEA) and /usr/include/ezbnmsec.h. See the *z/OS V1R9 Communications Server; IP Programmer's Guide and Reference* for details of each message.



A new perspective has been added to the current list of perspectives available for the user to configure on the main panel of the Configuration Assistant GUI.

From the NSS perspective the user is able to create images and have them be an NSS server, NSS client, or both. Stacks can also be created under an image that is either an NSS server or client. Currently the only technology to take advantage of the NSS function is IPSec. From the IPSec perspective the user can also set their NSS client image and stack settings. The design of the panels have been made in such a way to encourage the user to set as many defaults at the image level as they can and then have all of the stacks take those settings as their defaults. Each stack can override the image level defaults if they need to. The user can use either the NSS or IPSec perspective to setup NSS.

As with most other perspectives, key configuration files and excerpts of other files can be generated and deployed to target machines. This includes a sample RACF job that includes the RACF commands that are required to get all the stacks and images setup to use the NSS services.



When the NSS load module is not APF-authorized, an abend occurs. The following message will be logged to the console:

IEF450I NSSD STEP1 - ABEND=S000 U4087 REASON=00000000

To APF-authorize a data set, add an APF ADD statement for the data set to a PROG*xx* member of parmlib that is used for IPL. To immediately APF-authorize the data set, use the SETPROG APF z/OS command.

You will get the following message if the /var directory does not exist or the NSSDuser ID does not have authority to create the /var/sock subdirectory when the NSS server syslog level 2 is set:

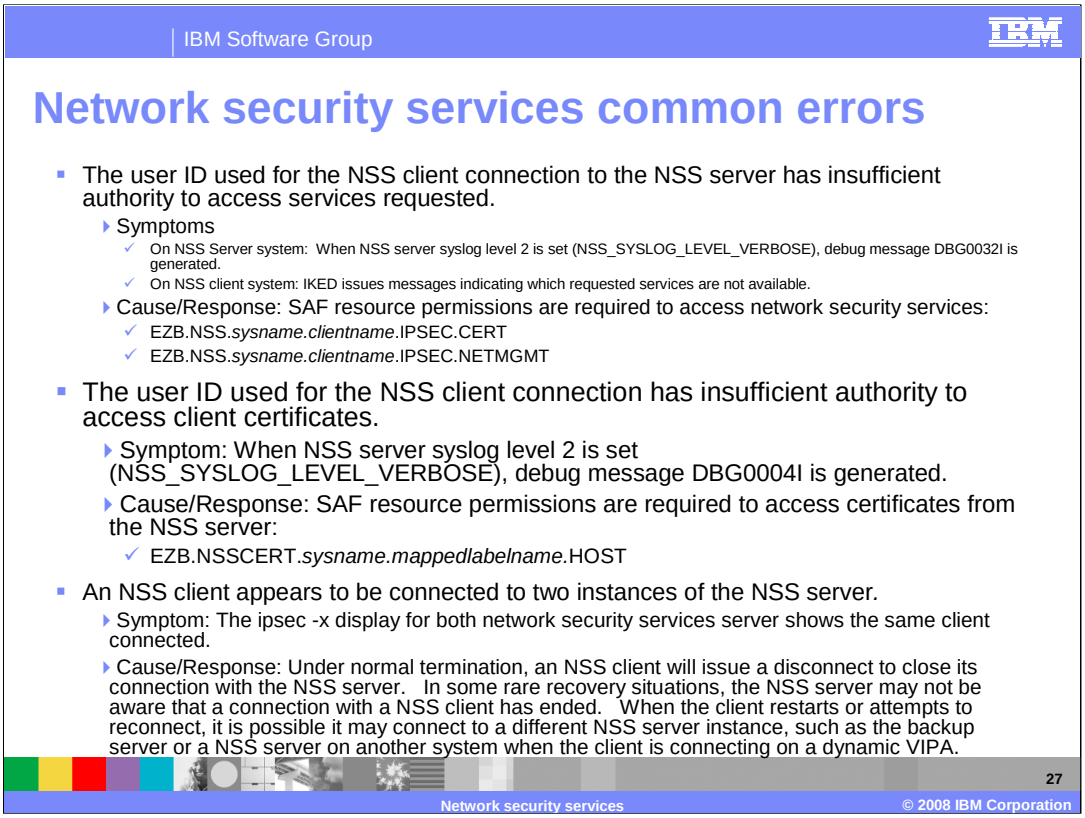
DBG0040I NSS_VERBOSE Cannot create socket directory /var/sock - rc -1 errno 135 EDC5135I Not a directory.

Write access to the /var directory is controlled through standard UNIX file permissions, so the user ID under which NSSD runs needs to have write permissions according to those flags.

When SSL is not properly configured, you may get the following message on the NSS server system when the NSS server syslog level 8 is set:

DBG0104I NSS_LIFECYCLE NSS connID 1 - the connection is not secure - the connection will be closed

IKED, acting as the NSS client, will issue message EZD1149I indicating that the connection is not secure. AT-TLS policies must be defined for both the client and the server to secure the connection. Refer to "AT-TLS policy" in chapter 18 "Providing network security services" of the IP Configuration Guide. If AT-TLS is enabled and the definitions are configured on the client and server stacks but these errors still occur then refer to the *z/OS V1R9 Communications Server; IP Diagnosis Guide* chapter 30 "Diagnosing Application Transparent Transport Layer Security (AT-TLS)."



When the SAF resources, EZB.NSS.*sysname.clientname*.IPSEC.CERT or EZB.NSS.*sysname.clientname*.IPSEC.NETMGMT, are not defined on the NSS server system or the user ID of the NSS client trying to request the service has not been permitted read access to the resource then you will get a message similar to the following, on the NSS server system, when the NSS server syslog level 2 is set:

DBG0032I NSS_VERBOSE ServauthCheck(USER2 ,EZB.NSS.MVS093.CLIENT2.IPSEC.CERT) rc 4 (DENY) racfRC 4 racfRsn 0

On the NSS client system, IKED will issue a message similar to the following messages:

EZD1145I The network security certificate service is not available for stack TCPCS2

EZD1147I The network security remote management service is not available for stack TCPCS2

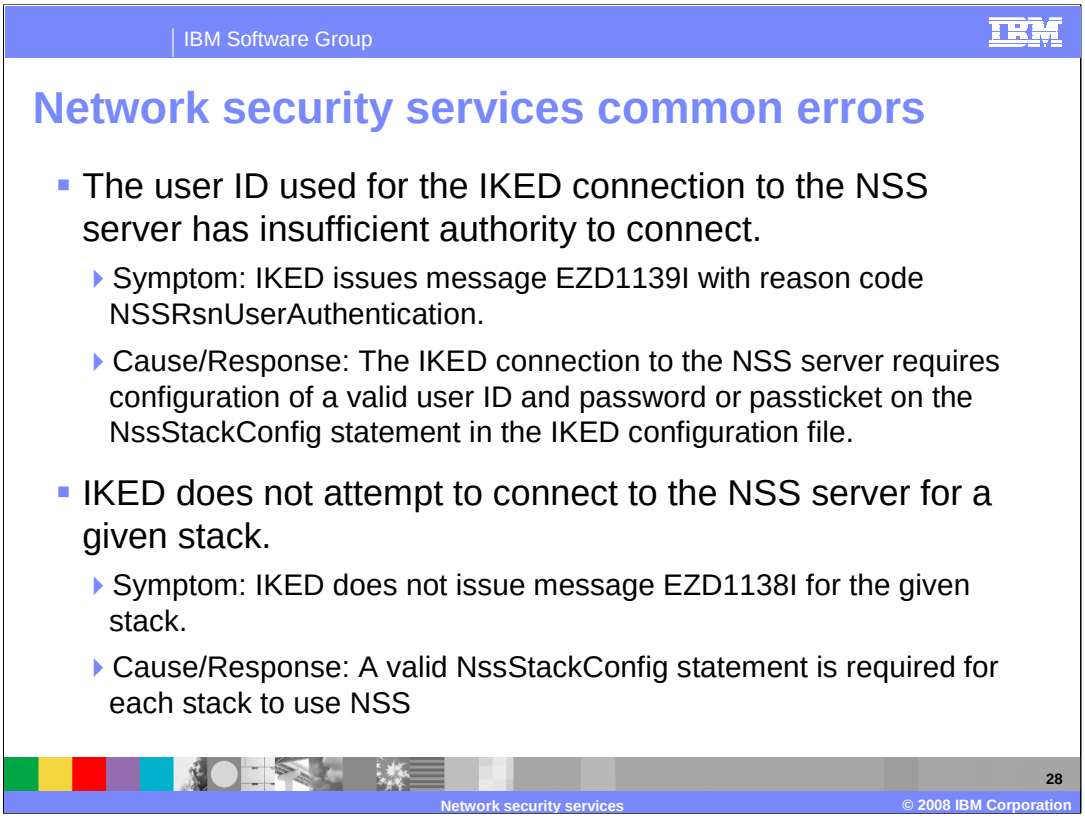
These resources must be defined on the NSS server system and the user ID configured on the NssStackConfig statement in the IKED configuration file must be permitted read access to them. Refer to "Steps for authorizing resources for NSS" in chapter 18 "Providing network security services" of the *z/OS V1R9 Communications Server; IP Configuration Guide*.

When the SAF resource, EZB.NSSCERT.*sysname.mappedlabelname.*HOST, is not defined or the user ID of the NSS Client trying to access the certificate is not permitted read access to the resource profile then a message similar to the following is issued when the NSS server syslog level 2 is set:

DBG0004I NSS_CERTINFO Client MVS093_TCPCS3 connected as userid USER1 is not authorized to profile EZB.NSSCERT.VIC012.NSCLIENT3.HOST associated with matching certificate (NSCLIENT3) for request 00000000000000150000000000000000

This resource must be defined on the NSS server system and the client user ID must be permitted read access to it.

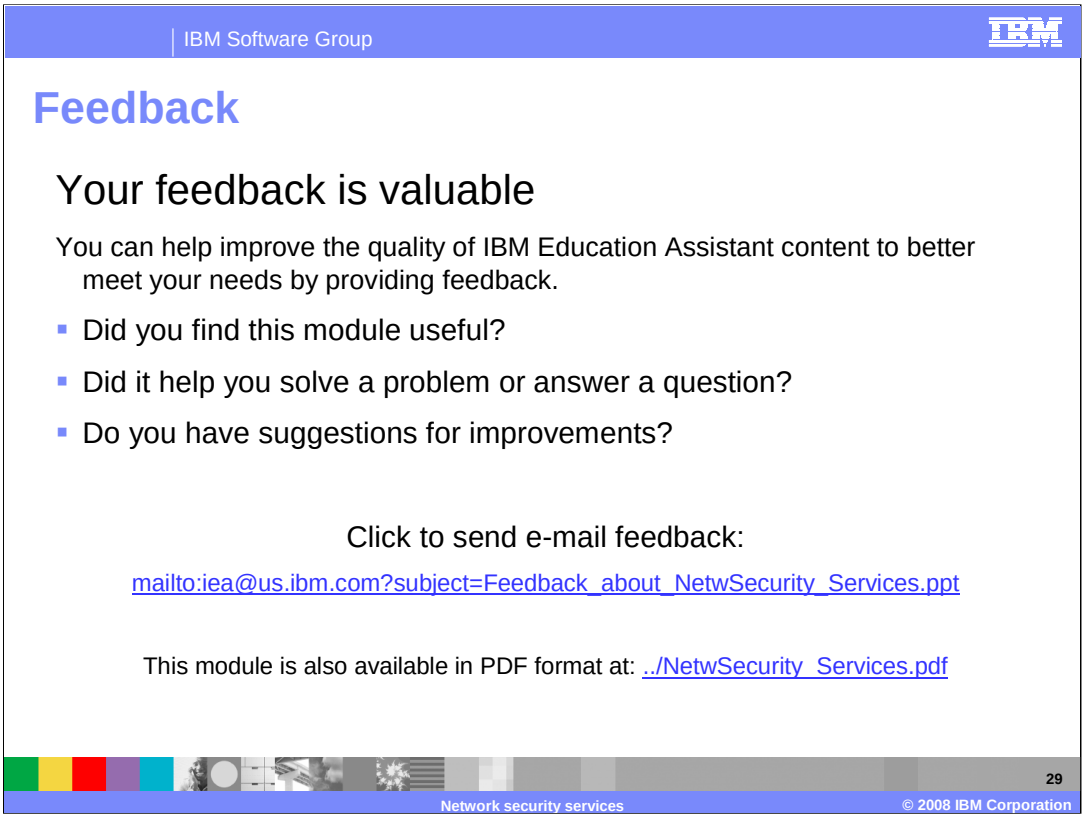
If a NSS client appears to be connected to two instances of an NSS server then issue the ipsec -w display on the system running the affected NSS client to determine to which NSS server the client is actually connected. Optionally, use the netstat drop command to close out the old connection on the other NSS server.



IKED, acting as the NSS client, issues the following message when the user ID associated with the NSS client can not be authenticated.

> **EZD1139I Request type NSS_ConnectClientReqToSrv with correlator ID 00000000000000040000000000000000 for stack TCPCS2 failed - return code EACCES reason code NSSRsnUserAuthentication**

A valid NssStackConfig statement must be configured for each stack that will act as a NSS client. Refer to chapter 8 "IKE daemon" in the *z/OS V1R9 Communications Server; IP Configuration Reference* for information about configuring the NssStackConfig statement.



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

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM RACF z/OS

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.