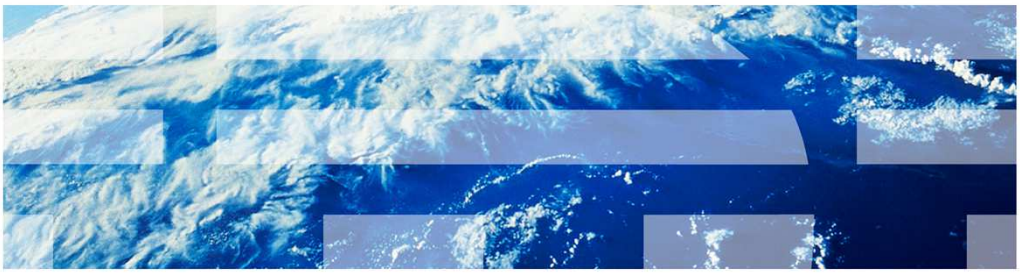
z/OS V2R1 Communications Server

Security

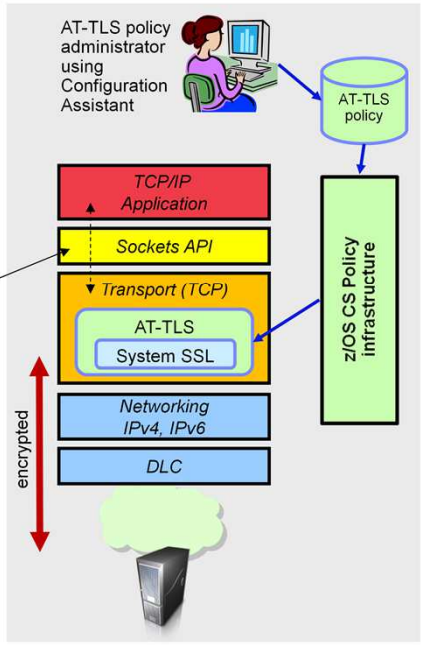


© 2013 IBM Corporation

This presentation provides an overview of the z/OS® V2R1 Communications Server enhancements for Security.

z/OS Application Transparent TLS overview (1 of 2)

- Stack-based Transport Layer Security (TLS)
  - TLS processing performed in TCP layer using System SSL without requiring any application change (transparent)
  - Application Transparent Transport Layer Security (AT-TLS) policy specifies which TCP traffic is to be TLS protected, based on a variety of criteria
- Application transparency
  - Can be fully-transparent to application
  - Optionally, an API allows applications to inspect or control certain aspects of AT-TLS processing – "application-aware" and "application-controlled" AT-TLS
- Key IBM AT-TLS exploiters:
  - FTP, TN3270, DB2®, IMS™ Connect, IMS SOAP GW, CICS® Sockets
  - NJE IP, RACF® (RRSF over TCP/IP), CIMOM, Netview

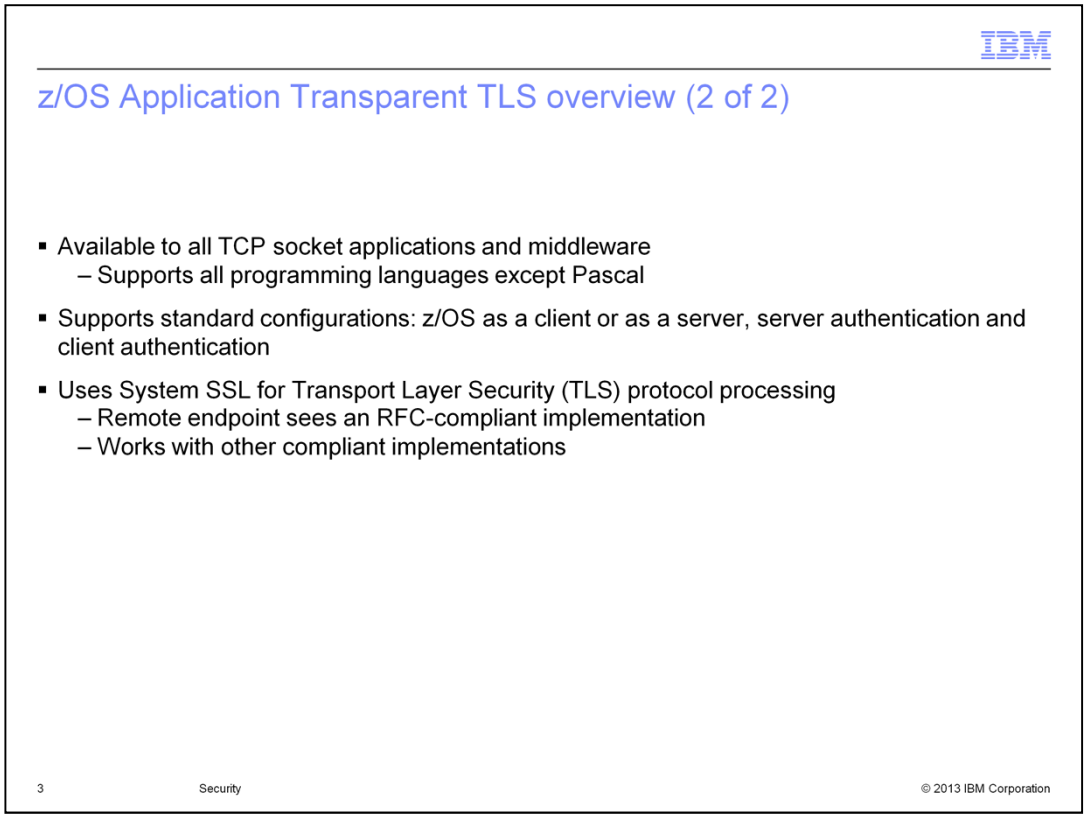


© 2013 IBM Corporation

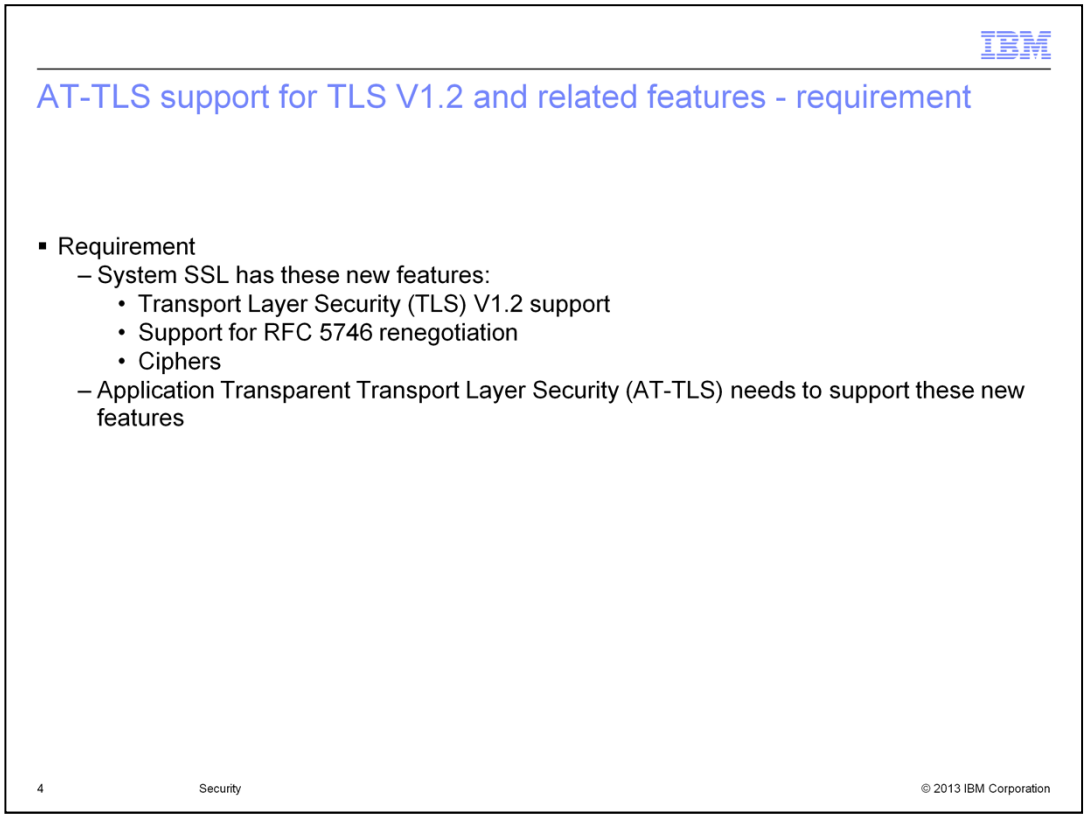
Application Transparent Transport Layer Security applies Transport Layer Security protection, using System SSL, to specific application traffic from within the TCP/IP stack, based on policies that you configure. Application Transparent Transport Layer Security policies specify which TCP traffic is to be protected. These policies are based on things such as the local and remote IP addresses and ports, whether the connections are inbound or outbound, the identity under which the z/OS application is running, and the time of day. As data passes through the stack, it is compared against the Application Transparent Transport Layer Security policy. If a match is found, the Transport Layer Security protection specified by the matching policy is applied by calling System SSL under the application's identity.

Application Transparent Transport Layer Security policies are created using Configuration Assistant for z/OS Communications Server, which is a GUI-based tool that runs under z/OSMF and accessed through a browser. Configuration Assistant creates the policy files that are then read by the z/OS policy agent and installed into the TCP/IP stack.

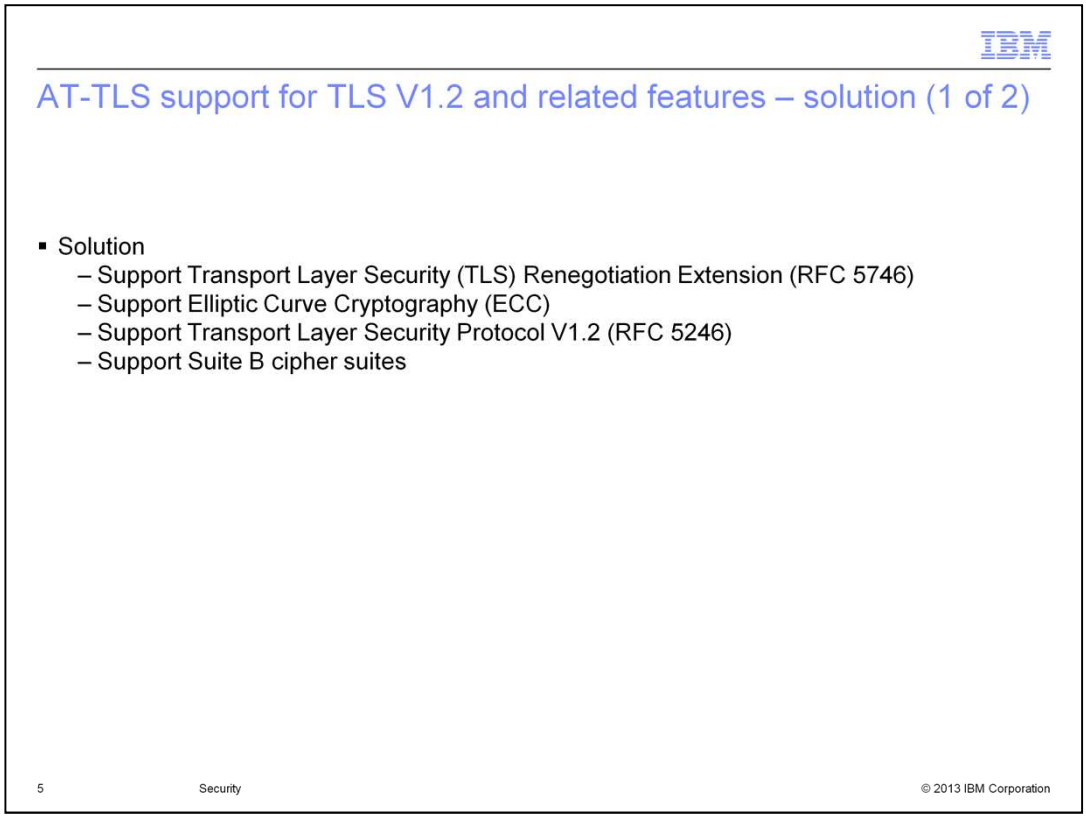
In many cases, Application Transparent Transport Layer Security is completely transparent to the application. However, for applications that want some involvement in the state of the Transport Layer Security session, Application Transparent Transport Layer Security provides an optional application programming interface that allows for awareness, or even control, over the Transport Layer Security session. Some applications need to inspect certain aspects of the session, for example, you might want to examine the certificate of the Transport Layer Security partner before proceeding to use the Transport Layer Security connection. These are called "Application Transparent Transport Layer Security aware" applications. Other applications need to control when the Transport Layer Security handshake occurs or terminates. FTP is a good example of such an application. This is also possible through the Application Transparent Transport Layer Security API. These applications are called "Application Transparent Transport Layer Security controlling" applications.



Transport Layer Security works only over TCP connections. Application Transparent Transport Layer Security supports all programming languages except for Pascal. Application Transparent Transport Layer Security supports the full range of Transport Layer Security configurations. So, your z/OS application can act as a Transport Layer Security client or a Transport Layer Security server. And both server and client authentication are supported. The remote Transport Layer Security partner sees z/OS as an RFC-compliant Transport Layer Security implementation. And z/OS works with other RFC-compliant Transport Layer Security implementations. So, even from the remote endpoint's perspective, Application Transparent Transport Layer Security is completely transparent.



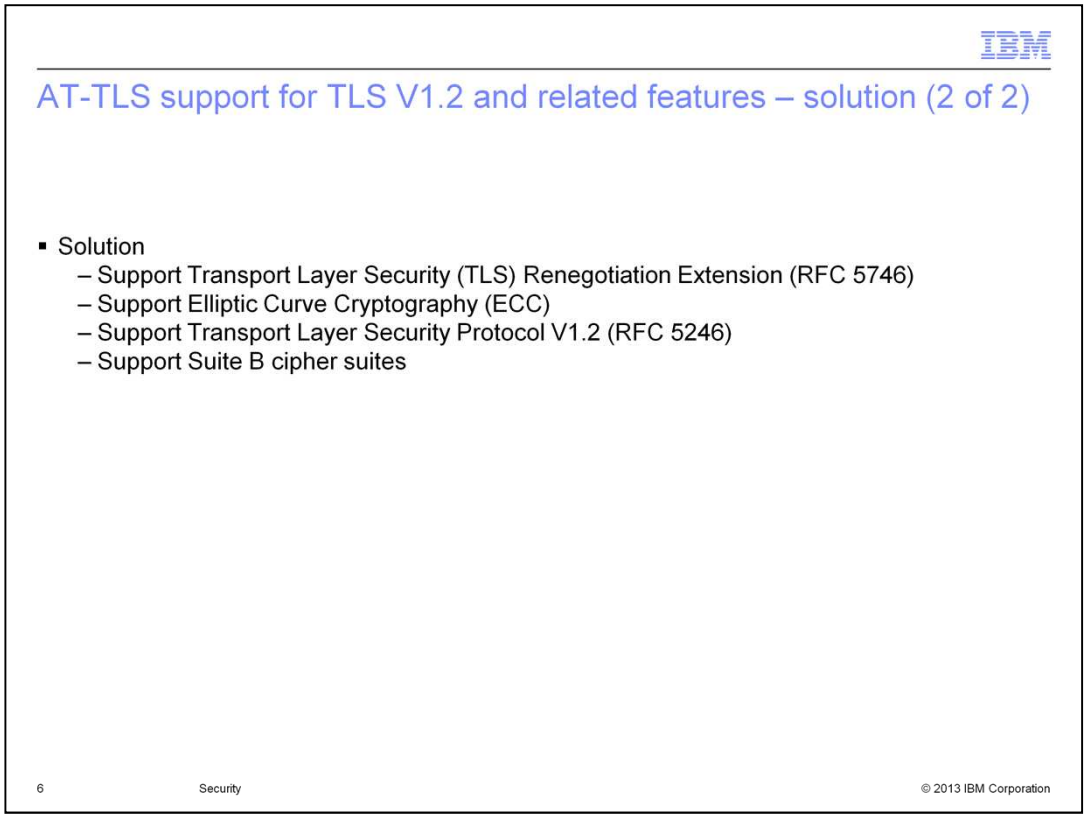
z/OS Communications Server promotes the use of Application Transparent Transport Layer Security. One advantage is that Application Transparent Transport Layer Security, for the most part, keeps up with System SSL so exploiting products get new features with no development effort. System SSL has shipped some new features that Application Transparent Transport Layer Security needs to support. Specifically, it needs to support RFC 5746 renegotiation or Elliptic Curve Cryptography. Also, in z/OS V2R1, System SSL introduces support for Transport Layer Security V1.2 and Suite B cipher suites.



The Application Transparent Transport Layer Security implementation in V2R1 supports all the features offered by System SSL.

This includes renegotiation. SSL connections can refresh the SSL keys on an existing SSL session. When a new SSL connection is started, the client can request to resume a previous SSL session. This is called renegotiation. Renegotiation is supported by SSL V3 and Transport Layer Security version 1.0 and higher. RFC 5746 defines new renegotiation flows between the client and server. It provides a mechanism to protect peers that permit re-handshakes. When supported, it enables both peers to validate that the re-handshake is a continuation of the previous handshake. Both the client and the server must support RFC 5746 for renegotiation to be used.

V2R1 also supports Elliptic Curve Cryptography, which is defined in RFCs 4492 and 5289. Elliptic Curve Cryptography offers equivalent security to other encryption methods, but with smaller key sizes. This is attractive in mobile and wireless environments. Elliptic Curve Cryptography-support is defined for both certificates and ciphers. Elliptic Curve Cryptography is supported in Transport Layer Security versions 1.0, 1.1 and 1.2. However, some ciphers are supported only in Transport Layer Security V1.2. Integrated Cryptographic Service Facility must be active to use Elliptic Curve Cryptography-support on z/OS.

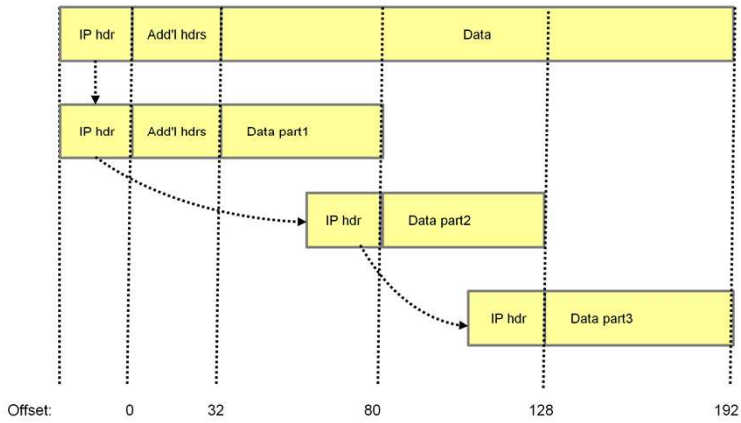


Transport Layer Security V1.2, as defined by RFC 5246, is also supported by V2R1. When Transport Layer Security V1.2 is enabled, SSL V2 sessions are not supported. If the remote partner can only use SSL V2, then the connection will fail during SSL negotiation. Transport Layer Security V1.2 also deprecates ciphers that are based on Data Encryption Standard.

V2R1 also supports Suite B cipher suites. RFC 5430 defines the cryptographic algorithms that can be used on a Suite B-compliant Transport Layer Security V1.2 session. Suite B requires that the key establishment and authentication algorithms that are used in Transport Layer Security V1.2 sessions be based on Elliptic Curve Cryptography and that the encryption algorithm be Advanced Encryption Standard Galois Counter Mode.

IP fragmentation – background information

IP packets can be legitimately fragmented within an IP network. This is to accommodate the Maximum Transmission Unit (MTU) of a link in the network.

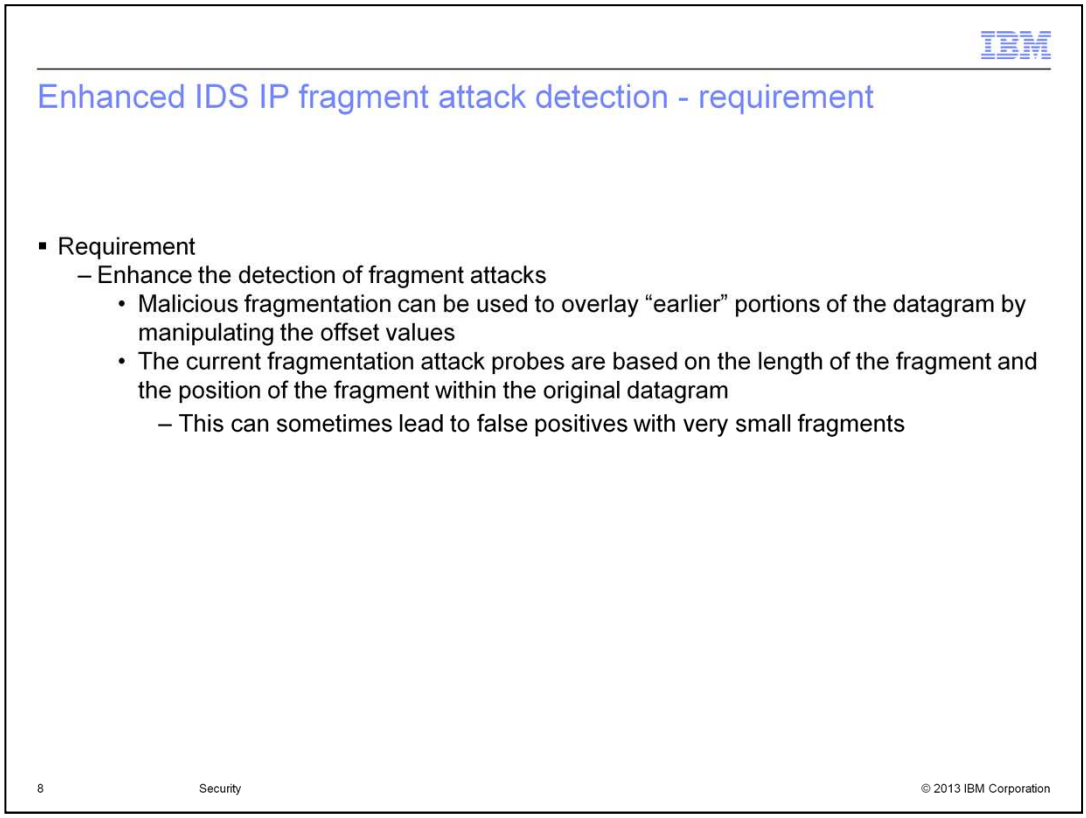


7 Security © 2013 IBM Corporation

This slide provides some background information on IP fragmentation.

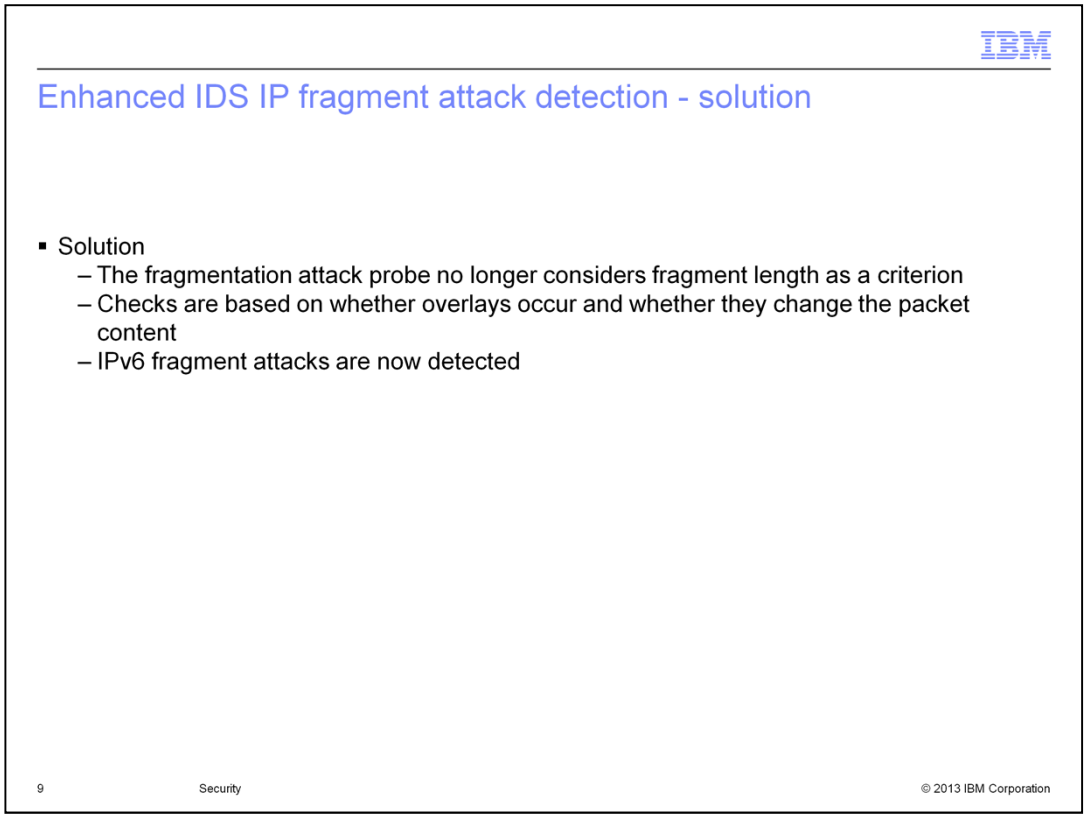
TCP/IP might have to fragment a packet to accommodate the Maximum Transmission Unit of a link in the network. TCP/IP can fragment the packet into two or more fragments to send a packet to the next node. Each fragment has its own IP header or a fragmentation extension header for IPv6 that indicates which IP datagram the fragment belongs to. It has the beginning offset, within that datagram, of the data carried in the fragment. Additional headers are part of the data and, with the exception of some IPv6 extension headers, can be fragmented.

In IPv4, any node along the route can fragment a packet. In IPv6, fragmentation is only allowed at the packet origin. Thus, the Maximum Transmission Unit of the entire path must be known by the sender.



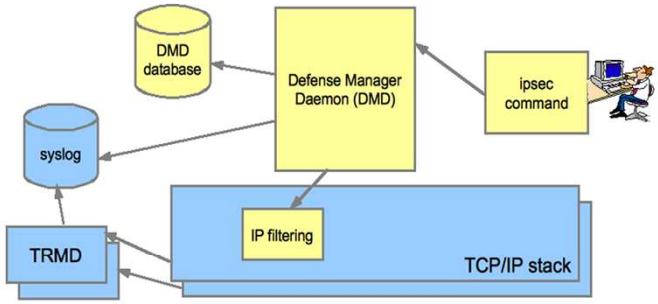
In previous versions, TCP/IP detects fragment attacks by using the length of the packet. It indicates that the fragment is suspicious if the first fragment is less than 88 bytes long or, for a fragment that is not the first fragment, if the offset is less than 88. The length of 88 was chosen because it represents the maximum length of the IP transport headers for IPv4.

Using this approach, valid packets can match the criteria and result in false positives. And fragments that overlay previous fragments and change the data are not detected as attacks. Fragments that change the length of the original packet, as indicated by previous fragments, are also not detected. Furthermore, IPv6 fragmentation attacks are not detected at all.



Starting in V2R1, TCP/IP does not use the fragment length to detect fragment attacks. It indicates a fragment attack for both IPv4 and IPv6 if the current fragment overlays the data of a previous fragment with different data. It also indicates an attack if the current fragment specifies a different length for the original packet than what was specified by a previous fragment.

Limit defensive filter logging – background information



- Defensive filtering provides a mechanism to install temporary defensive filters into a TCP/IP stack to block a specific attack or pattern of attacks
- Two modes of defensive filters:
  - Blocking mode – Denies packets that match the defensive filter and logs, if logging requested for this filter
  - Simulate mode – Writes a log record to syslogd for a packet that matches the defensive filter, but filtering of the packet continues
    - Often used initially to gauge the effect of implementing defensive filtering, before enabling blocking mode

10 Security © 2013 IBM Corporation

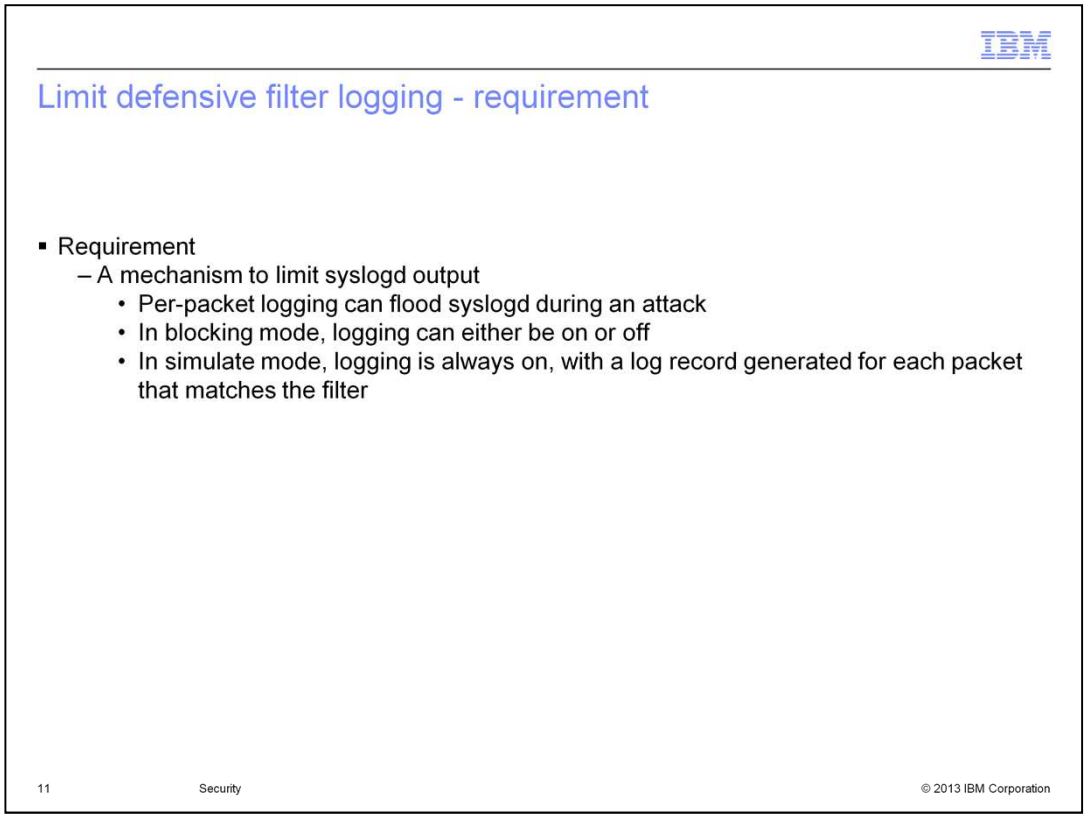
Defensive filtering provides a mechanism to install temporary defensive filters into a TCP/IP stack to block a specific traffic pattern. Defensive filters are added, updated, deleted and displayed with the z/OS UNIX ipsec command. Defensive filters are not configured in policy.

Defense Manager Daemon manages defensive filters. A single instance of Defense Manager Daemon runs per LPAR and can manage defensive filters for one or more TCP/IP stacks. Defense Manager Daemon maintains a database of currently-defined defensive filters to ensure that the filters are applied after an outage or restart.

To use defensive filtering, IP security must be enabled. If you do not have IP security enabled, you can configure a minimal “permit all” policy in the TCP/IP profile. See the section named “Enabling defensive filtering” in the Communication Server IP Configuration Guide for more information.

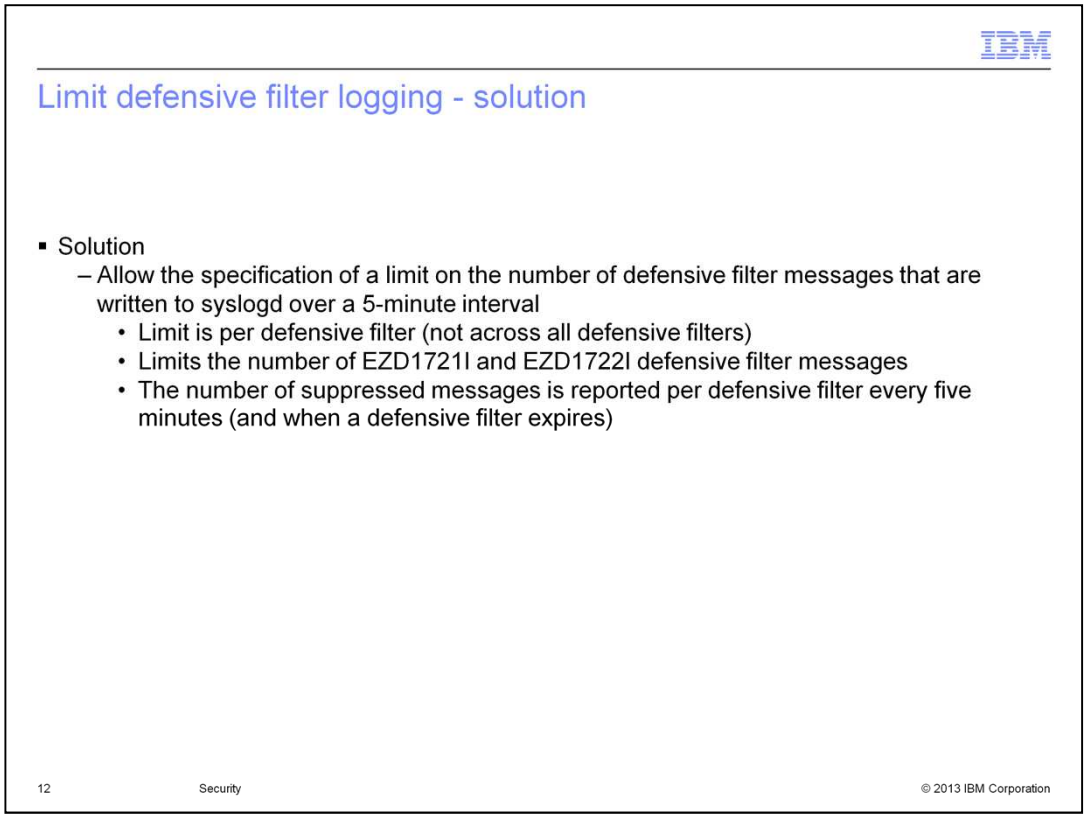
Several components are involved in defensive filtering. The ipsec command is used to add a defensive filter manually or through automation. An *ipsec -F add* request is processed by Defense Manager Daemon. Defense Manager Daemon installs the defensive filter in the TCP/IP stack and saves a copy of the filter to persistent storage in the Defense Manager Daemon database.

Once installed, the defensive filter is checked as part of IP filtering of inbound and outbound packets. If a packet matches a blocking defensive filter, the packet is discarded and a log message is written to syslogd, if logging is requested. Traffic Regulation Manager Daemon must be started for a stack to write a defensive filter log message to syslogd.



In previous releases, per-packet logging is either turned on or off. There is no way to request limited logging to protect against a flood of syslogd messages. For a defensive filter in blocking mode, logging can either be on or off. For a defensive filter in simulate mode, logging is always on.

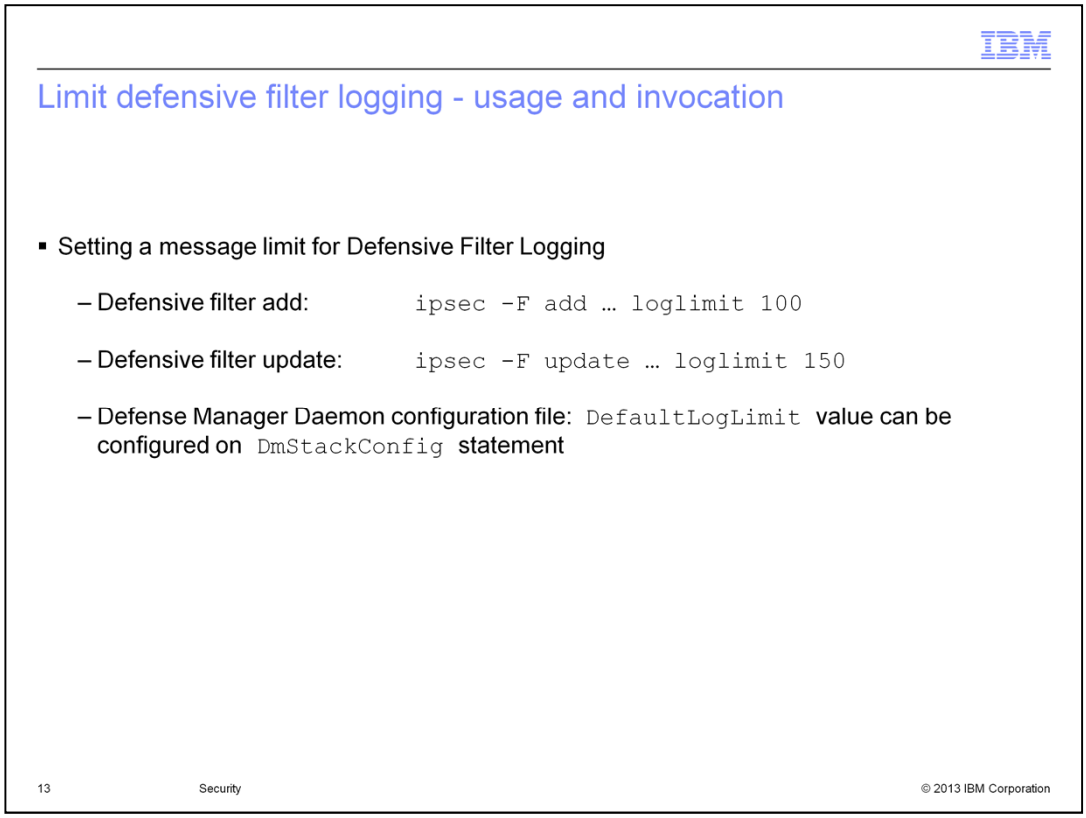
The information in the messages can be helpful in understanding the type of traffic that is matching the defensive filters and being dropped in the case of a blocking filter. However, if you added the defensive filter in response to a detected attack, you do not want a flood of packets from the attacker to overwhelm your syslogd.



Starting in z/OS V2R1 Communications Server, you can specify a limit on the number of defensive filter messages logged in a 5-minute interval. Each defensive filter has its own limit. The limit applies to defensive filter messages EZD1721I and EZD1722I that are written on a per-packet basis.

A limit of 0 - 9999 can be specified. A value of 0 indicates that there is no limit; a message is written to syslogd for each packet that matches the defensive filter. A value of 1 – 9999 indicates the limit for this defensive filter.

If messages are suppressed during a 5-minute interval, the number of suppressed messages is reported at the end of the interval.



Logging can be limited for a defensive filter either by setting a limit for an individual filter or by specifying a default limit in the Defense Manager Daemon configuration file. This default limit applies to all defensive filters that are added to the stack.

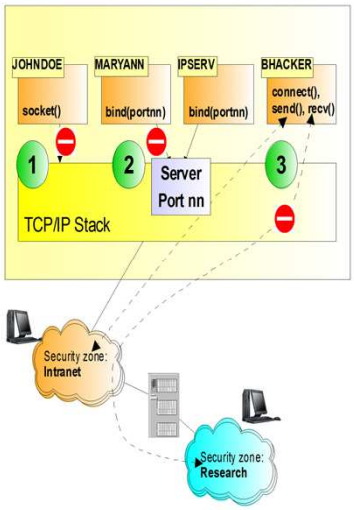
To set a limit for an individual filter, specify the new loglimit parameter when you add the defensive filter using ipsec -F add. You can also update the loglimit setting with the ipsec -F update command.

A default limit can be set for all defensive filters that are added to a TCP/IP stack by setting the new DefaultLogLimit parameter on the DmStackConfig statement in the Defense Manager Daemon configuration file.

You can combine the two approaches by setting DefaultLogLimit in the Defense Manager Daemon configuration file and overriding the default for certain filters through the ipsec command.

Improve auditing of NetAccess rules – background (1 of 2)

- NetAccess provides the ability to control z/OS user access to certain security zones
  - Via NETACCESS statement in TCP/IP profile
- Access to security zone allowed if user permitted to access Security Access Facility (SAF) resource



14 Security © 2013 IBM Corporation

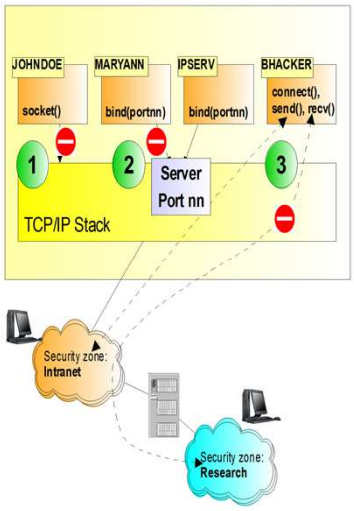
NetAccess, or Network Access, is the Communications Server function that provides you the ability to control user access to networks, sub networks, and hosts. These are referred to as security zones. Using NetAccess, you can specify whether or not individual users should be permitted to access the security zones. User access to a security zone determines their ability to send and receive data between z/OS and the IP addresses in the zone.

There are three setup steps needed to control access to security zones. First, use the NETACCESS statement in the TCP/IP profile to define a name for each of the security zones. Next, in the SERVAUTH class, define a Security Access Facility resource profile for each zone name. Finally, for the set users that should have access to a security zone, provide them read access to the corresponding Security Access Facility resource profile.

You can use the NETACCESS statement to enable network access control for inbound traffic only, outbound traffic only, or both. The NetAccess function does not apply to traffic that is being routed by the stack, only to traffic that originates or terminates at the stack.

Improve auditing of NetAccess rules - background (2 of 2)

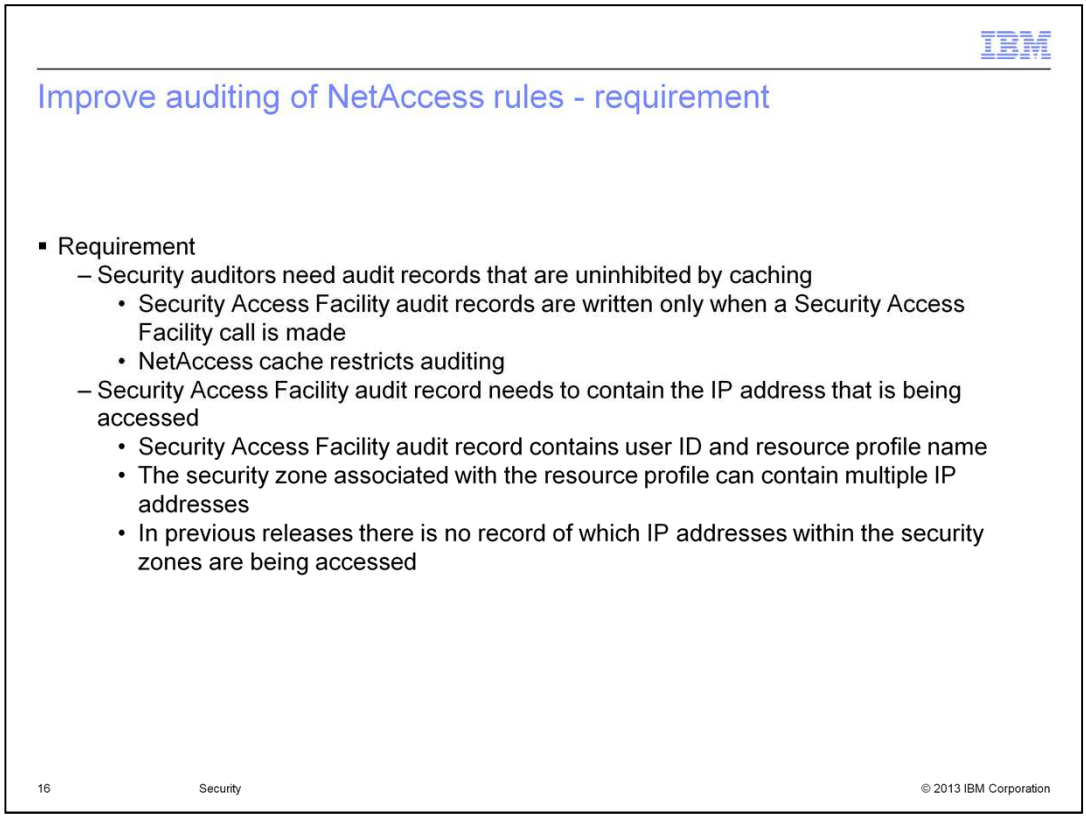
- Results from Security Access Facility calls to check if a user can access a security zone are cached
  - Only the first access check for a user to a security zone results in a Security Access Facility call
  - No Security Access Facility call for subsequent access checks for the same user to the same or different IP addresses in the same zone
- NetAccess provides a log string on calls made to Security Access Facility
  - Security Access Facility includes that log string in its audit records
  - Audit records are used by security auditors



15 Security © 2013 IBM Corporation

User access information that is learned from calls to Security Access Facility is cached. Once a user's access to a particular security zone has been determined, subsequent checks of that user's access to IP addresses in the same security zone are completed using the information in the cache. This caching reduces the number of calls to Security Access Facility. TCP/IP maintains a cache for each user whose access has been checked. The cache can hold the access information for up to 24 different security zones. The cache for a user is deleted when the last active connection is closed. It is also deleted if the NETACCESS statement is updated using a VARY TCPIP,,OBEYFILE command or if the Security Access Facility resource profiles in the SERVAUTH class are refreshed.

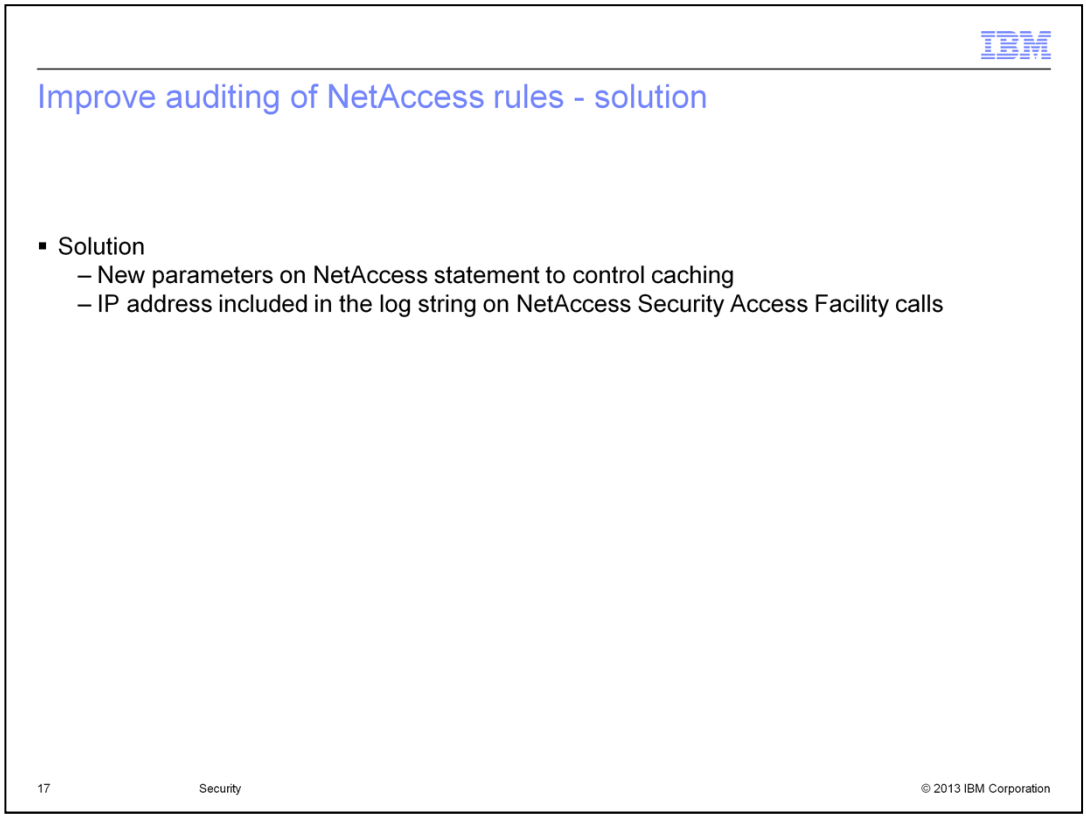
A log string is provided each time TCP/IP calls Security Access Facility to check a user's access to a resource profile. Security Access Facility includes this log string in the audit records that it writes. The audit record contains information like user ID, the name of the resource profile, and the log string provided by the stack. These Security Access Facility audit records are used by security auditors to audit network resources that users are attempting to access and network resources that are attempting to access the stack.



The caching performed by NetAccess restricts the information available to security auditors. When the cache is used, no call is made to Security Access Facility.

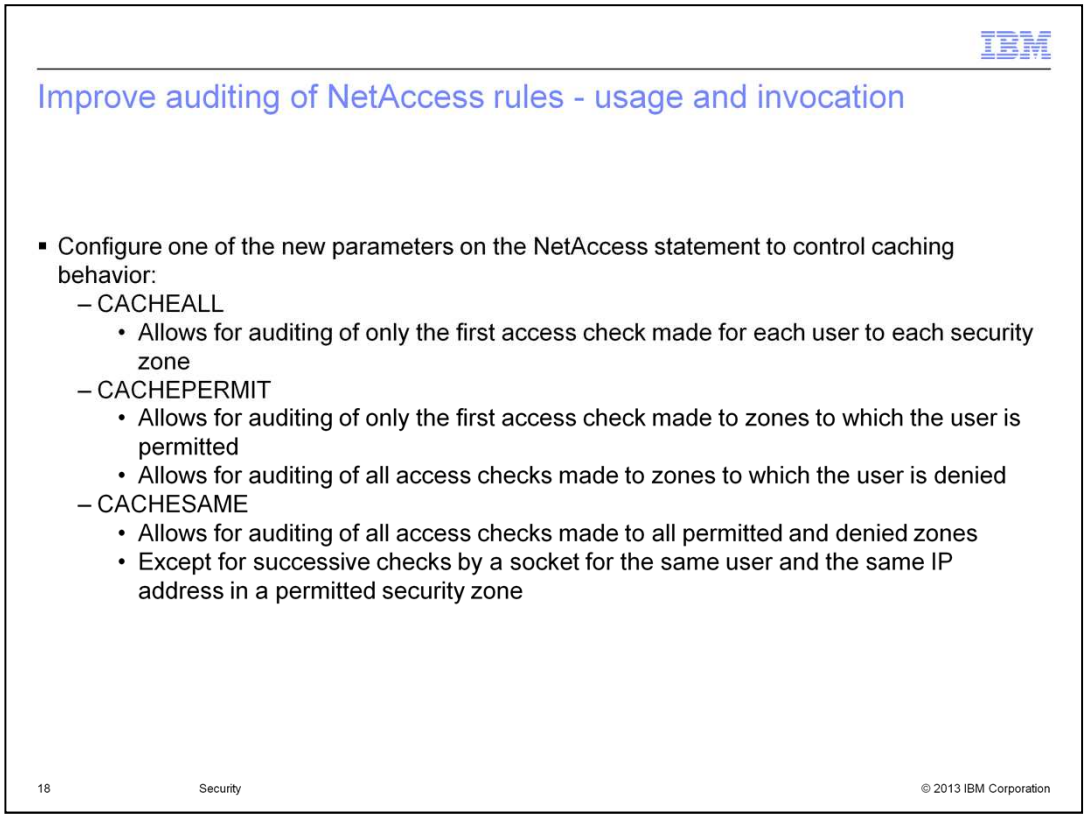
Security auditors use Security Access Facility audit records to determine the network resources that users are attempting to access and the network resources that are attempting to access the stack. These audit records are written only by Security Access Facility when it is called. Because NetAccess caching reduces the number of NetAccess-related audit records, it is difficult for security auditors to get a full picture of the resources being accessed and the resources accessing the stack.

In previous releases, the Security Access Facility audit records that are written for each call to check a user's access to a resource profile do not contain the IP address that the user is attempting to access. The included information, user ID, and resource profile name allow a security auditor to determine which security zones are being accessed by users. However, a network zone can contain multiple IP addresses. Without the IP address in the records, the auditor cannot determine the IP addresses that are being accessed. The IP address information is especially important when access is not permitted to the security zone.



In V2R1, there are three new parameters on the NETACCESS configuration statement in the TCP/IP profile. These parameters, CACHEALL, CACHEPERMIT, and CACHESAME, allow you to control the level of caching that is performed by NetAccess.

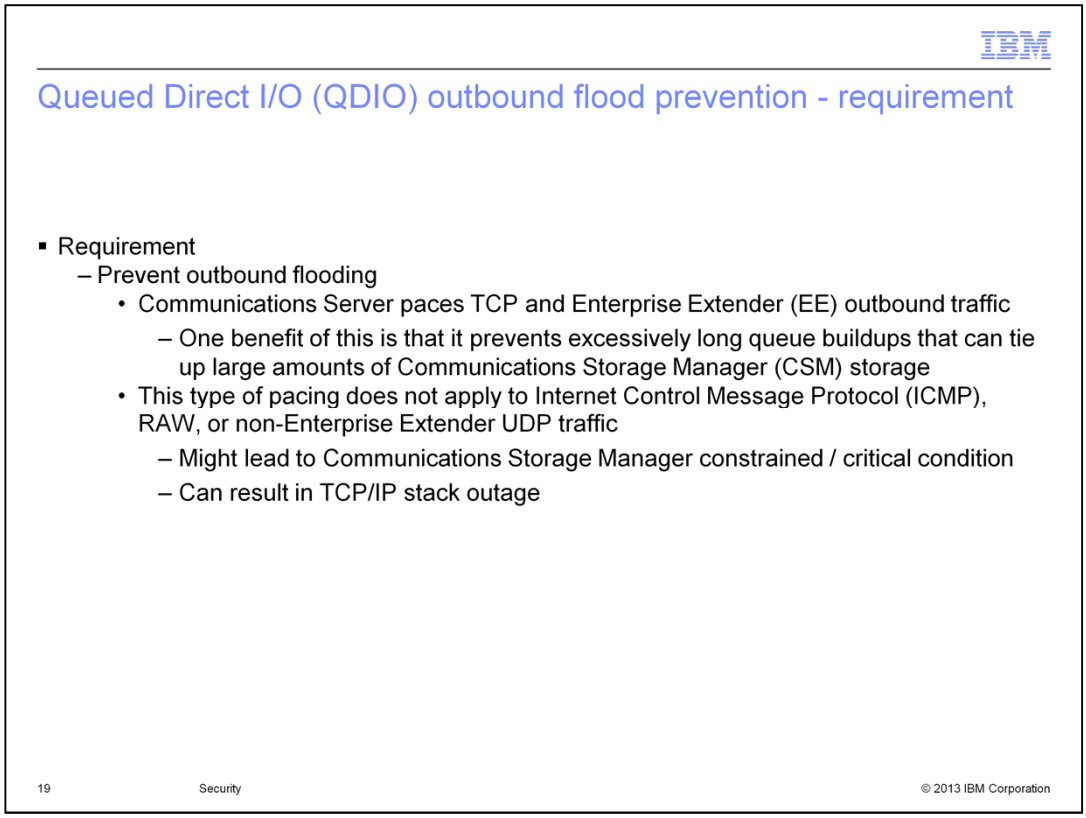
All calls made to Security Access Facility for NetAccess now include the IP address that triggered the call. The IP address is included in the log string that is provided as a parameter on the Security Access Facility call. And Security Access Facility includes that log string in the audit record that it writes.



When the CACHEALL parameter is specified, the result is the same behavior as prior releases. The auditor can audit only the first access check made for each user to each security zone. This is the default behavior.

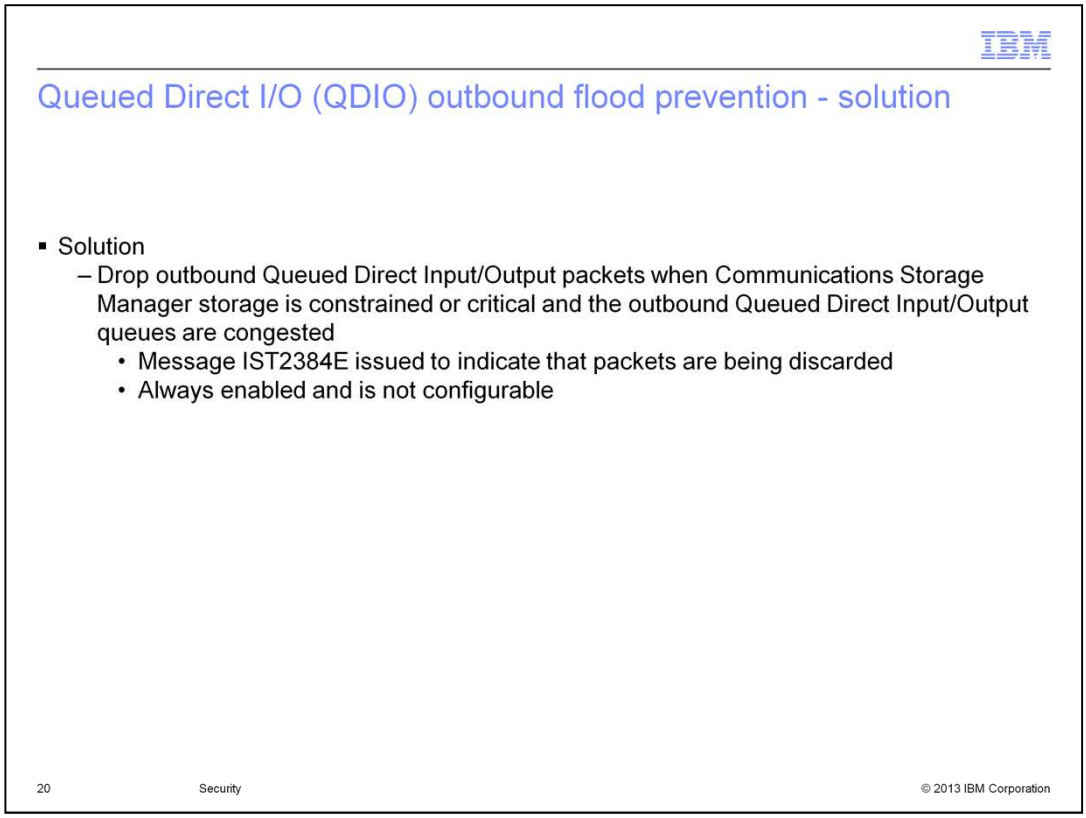
When the CACHEPERMIT parameter is specified, the auditor can audit only the first access check made for each user to security zones to which that user is permitted access. However, the auditor can audit all access checks made to security zones to which that user is denied access. This is important because access attempts that are not permitted are typically of more interest to an auditor.

When the CACHESAME parameter is specified, the auditor can, with one exception, audit all access checks made to all zones. This includes both zones to which the user is permitted and zones to which the user is denied. The one exception is for successive checks by a socket for the same user and the same IP address in a permitted security zone. Repeated audit records for the same socket user accessing the same IP address to which it is permitted access are not of great interest to an auditor. Providing such records can greatly increase the number of audit records written and can affect performance.

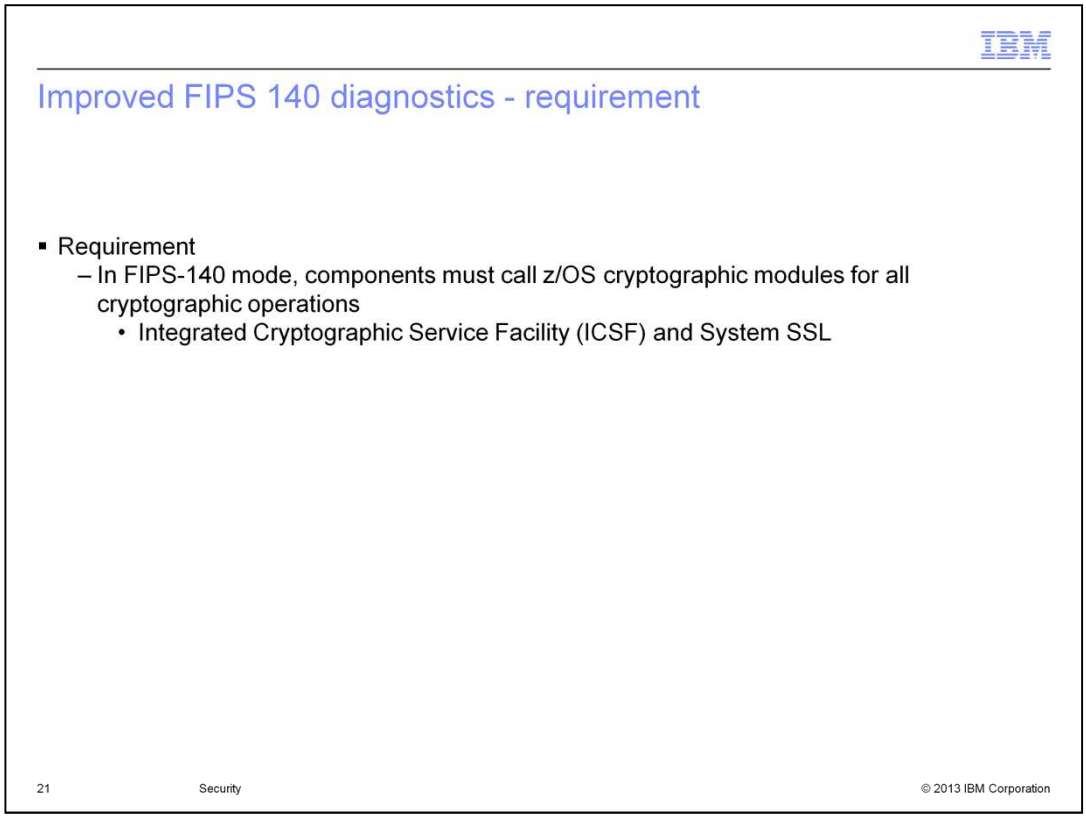


z/OS Communications Server will pace outbound traffic as needed for TCP and Enterprise Extender traffic. Both cases rely on existing flow-control features of the upper-layer protocols to achieve the pacing goals. An additional z/OS Queued Direct Input/Output feature is called Random Early Slowdown. It allows the outbound Queued Direct Input/Output processing to detect outbound queue congestion and provide local feedback to the upper layer protocols, TCP and Enterprise Extender, thus initiating a decrease in transmission rates. This function applies only to Open Systems Adapter-Express Queued Direct Input/Output. No such features exist in Internet Control Message Protocol, UDP or raw IP protocols.

The z/OS TCP/IP stack always replies to Internet Control Message Protocol timestamp requests. When a flood of such requests are sent to z/OS, under the right conditions, the stack can back up because it cannot get the responses out quickly enough. The result is a constrained or critical Communications Storage Manager condition. If the condition is not relieved in a timely fashion, a stack outage can occur. Note that this behavior can happen with other Internet Control Message Protocol requests that always generate a response or UDP requests to an application that behaves in a similar manner. One example of such Internet Control Message Protocol requests are echo requests.

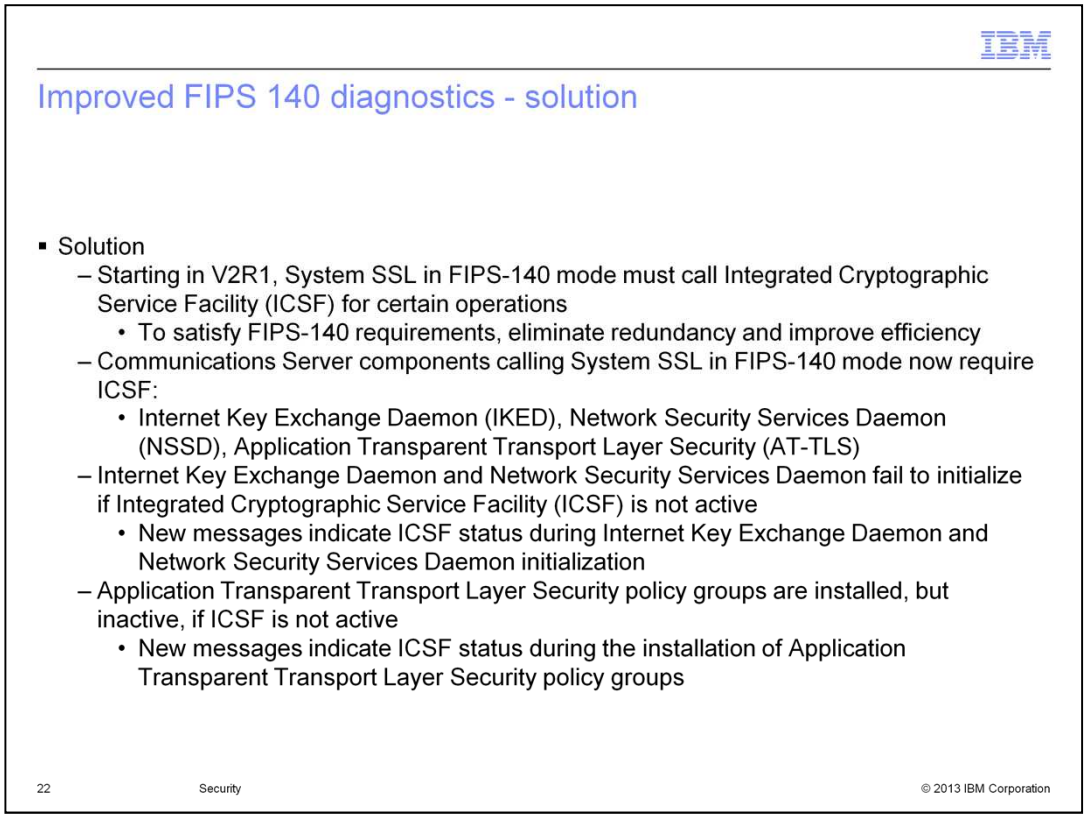


Starting in V2R1, outbound Queued Direct Input/Output packets are dropped when Communications Storage Manager is in a constrained or critical state. This is done for all packet types. The support is always on and is not configurable. It affects only Open Systems Adapter and HiperSockets™ interfaces. A new message, IST2384E, is issued to indicate that packets are being discarded.



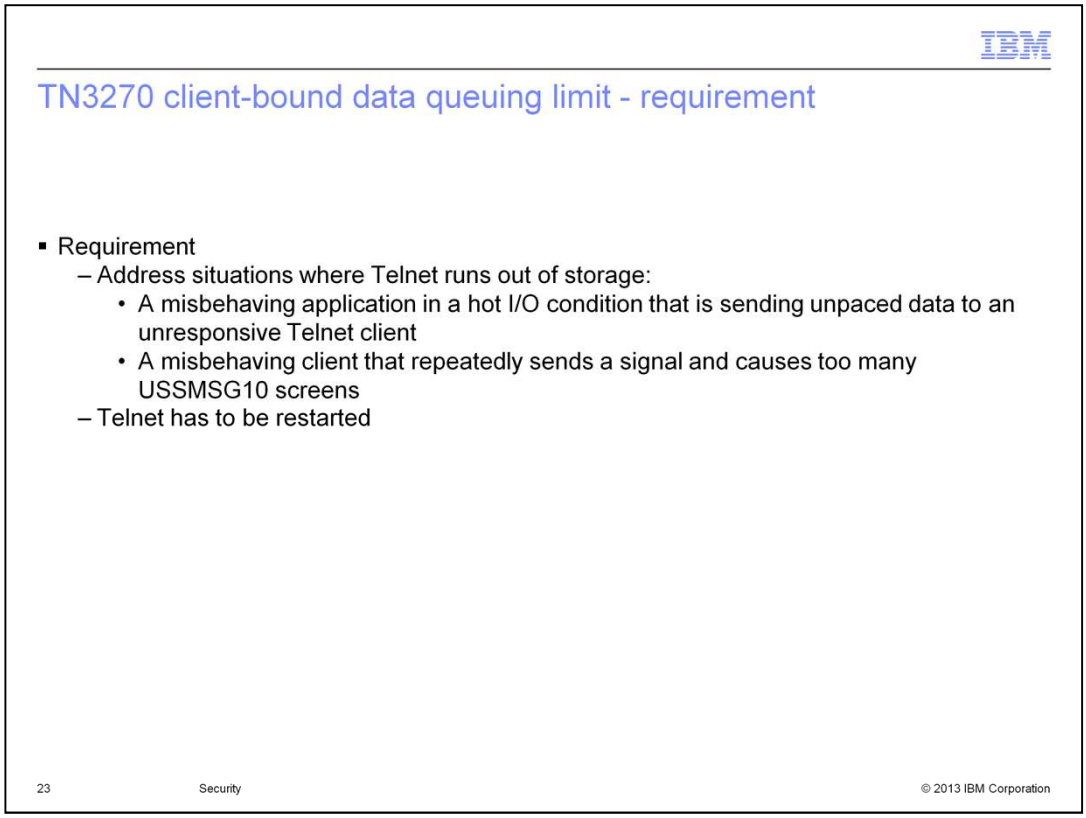
FIPS stands for Federal Information Processing Standards. It is a collection of documents published by the federal government of the United States. FIPS 140 provides specifications for cryptographic modules. Cryptographic modules implement cryptographic algorithms and perform cryptographic key operations. Integrated Cryptographic Service Facility and System SSL are the two cryptographic modules on z/OS.

Internet Key Exchange Daemon, Network Security Services Daemon, Application Transparent Transport Layer Security, and the TCP/IP stack are the z/OS Communications Server components that provide FIPS-140 modes of operation. These components must call the certified cryptographic modules, Integrated Cryptographic Service Facility and System SSL, when operating in FIPS-140 mode.

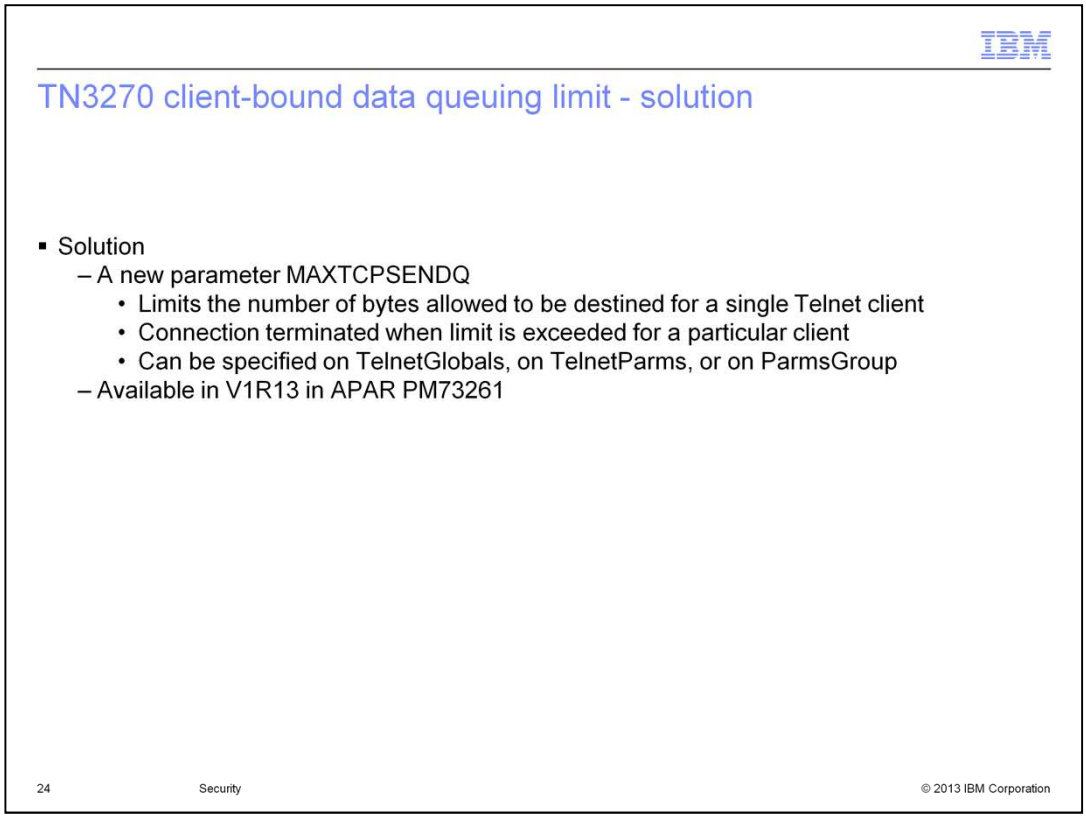


Starting in V2R1, System SSL, when operating in FIPS-140 mode, must call Integrated Cryptographic Service Facility for various cryptographic operations. This change was made to eliminate redundancy and improve efficiency. As a result, three of the four z/OS Communications Server components that provide a FIPS-140 mode of operation now depend on Integrated Cryptographic Service Facility.

New messages have been introduced to better communicate the status of Integrated Cryptographic Service Facility as it relates to Internet Key Exchange Daemon, Network Security Services Daemon, and Application Transparent Transport Layer Security policy groups. Internet Key Exchange Daemon and Network Security Services Daemon running in FIPS-140 mode will fail to initialize if Integrated Cryptographic Service Facility is not active. They will initialize without Integrated Cryptographic Service Facility if they are not running in FIPS-140 mode. FIPS-140 Application Transparent Transport Layer Security policy groups are installed, but inactive in the stack, if Integrated Cryptographic Service Facility is not active. The new messages contain clear information about the problem and what you should do to resolve the problem.

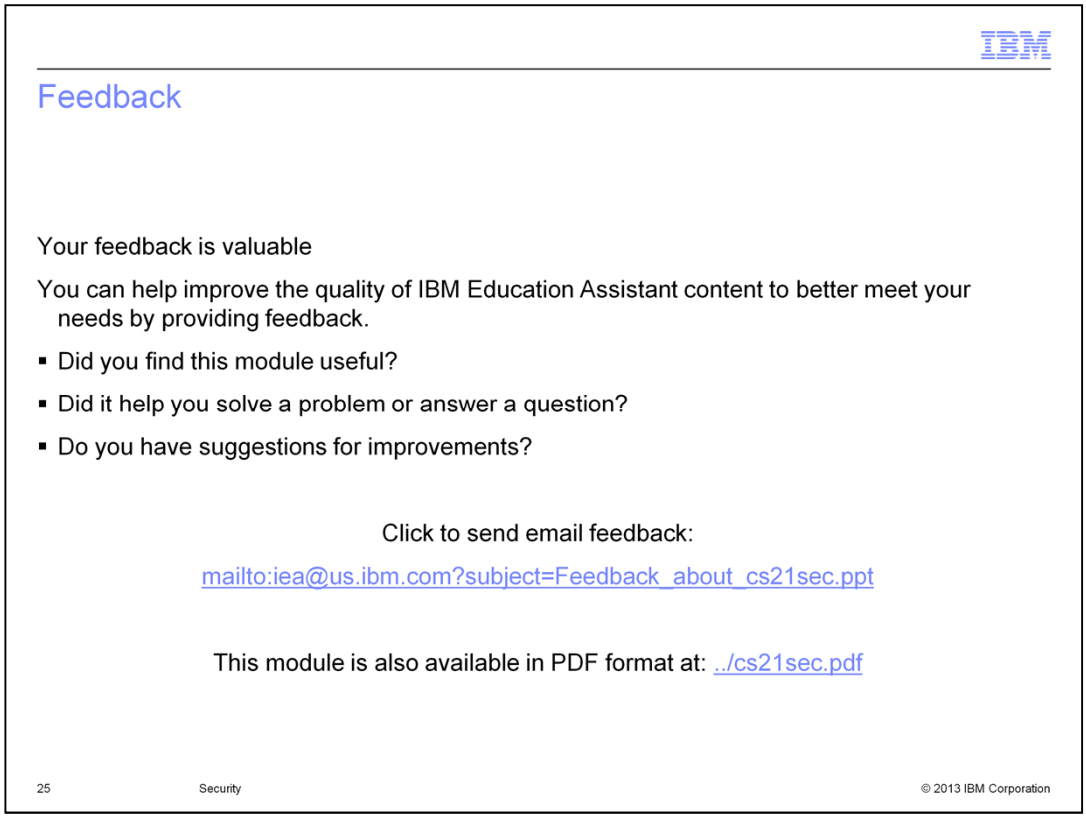


Here are two examples of situations that cause Telnet to run out of storage. The first example is an application that is sending data on a session that is not paced. This allows the data to queue up in Telnet faster than TCP can deliver it to the client. When Telnet runs out of storage, it has to be recycled. The second example is a client that repeatedly sends a signal that causes Telnet to send a USSMSG10 screen to it. These screens back up and cause Telnet to run out of storage. Again, Telnet has to be recycled to relieve the situation.



A new Telnet parameter, MAXTCPSENDQ, is introduced in V2R1. It limits the number of bytes that can be queued up in Telnet for delivery to a single client. It can be specified as part of the Telnet Globals, TelnetParms, and ParmsGroup statements. When the limit is reached for a connection, the connection is terminated, thereby preventing the out-of-storage condition.

This support is also available in V1R13 in APAR PM73261.



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, CICS, DB2, HiperSockets, IMS, RACF, 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

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

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.

© Copyright International Business Machines Corporation 2013. All rights reserved.

© 2013 IBM Corporation