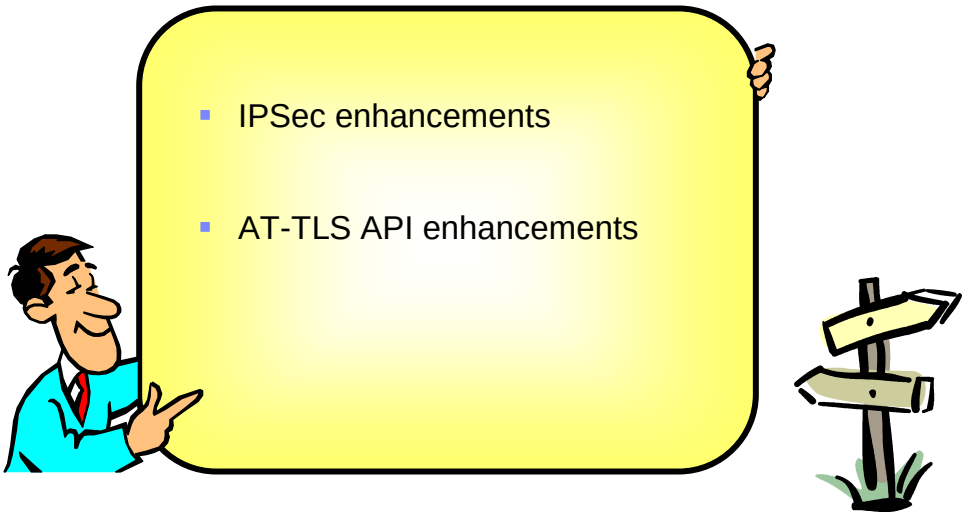

This presentation discusses the Security enhancements for the z/OS V1R9 Communications Server.

# Agenda







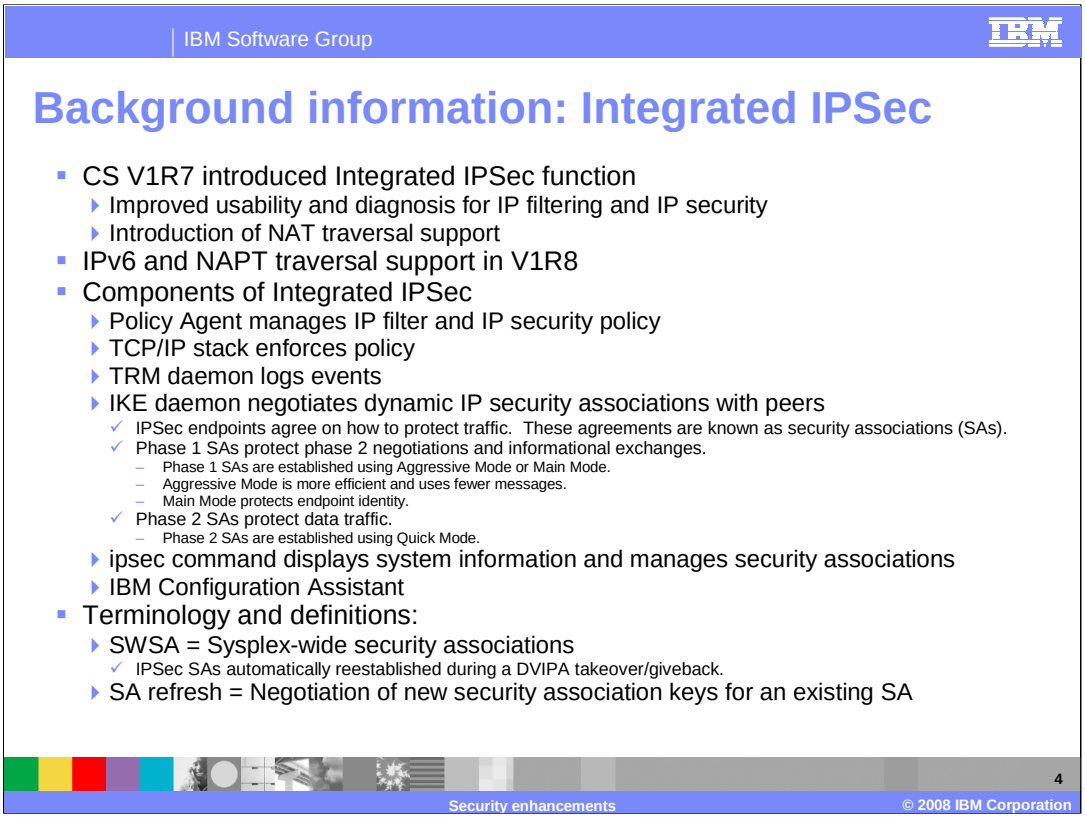
Enhancements have been made to the z/OS V1R9 Communications Server for IP Security and the AT-TLS API.

## Section

***IPSec enhancements***



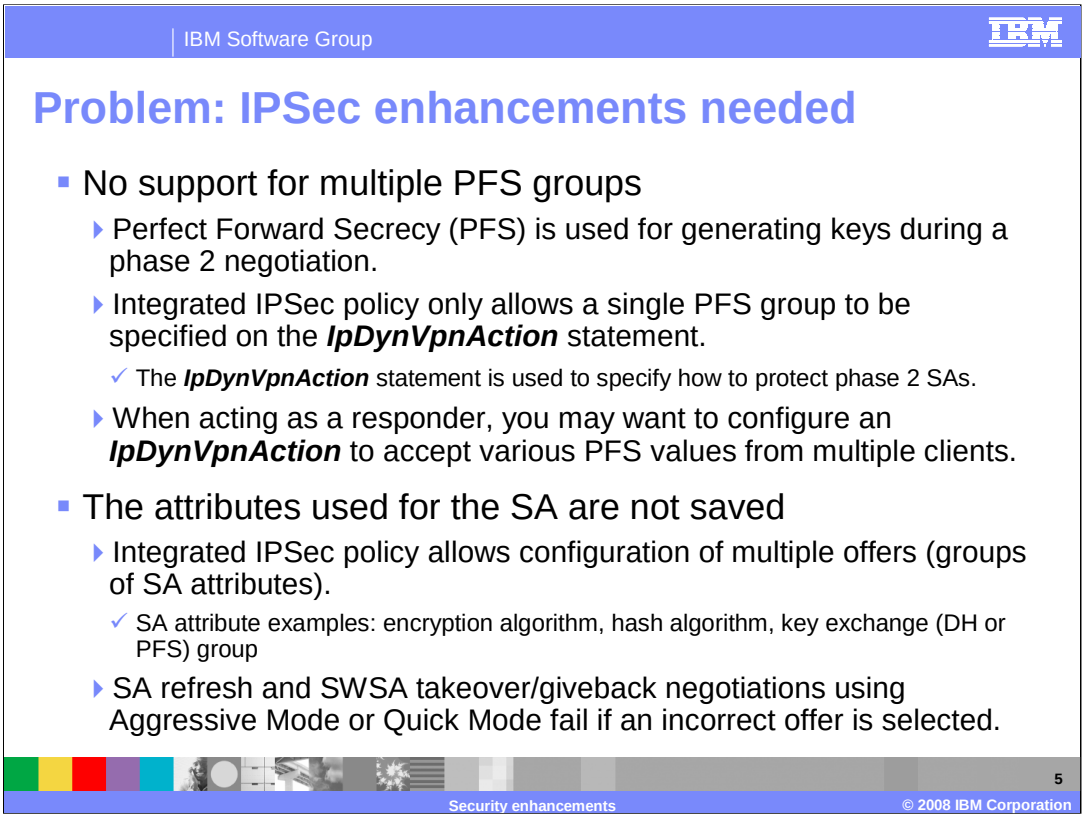

This section describes the enhancements to the existing IPSec function in V1R9.




The Integrated IPSec function was introduced in z/OS V1R7 Communications Server as a replacement for the Security Server Firewall Technologies IP filtering and IP security function. The Integrated IPSec function has improved usability over Firewall Technologies. The Integrated IPSec function has Network Address Translation (NAT) support. It also has Network Address Port Translation (NAPT) traversal support. NAPT translates multiple internal IP addresses to a single public address and translates the TCP or UDP port to make the connection unique.

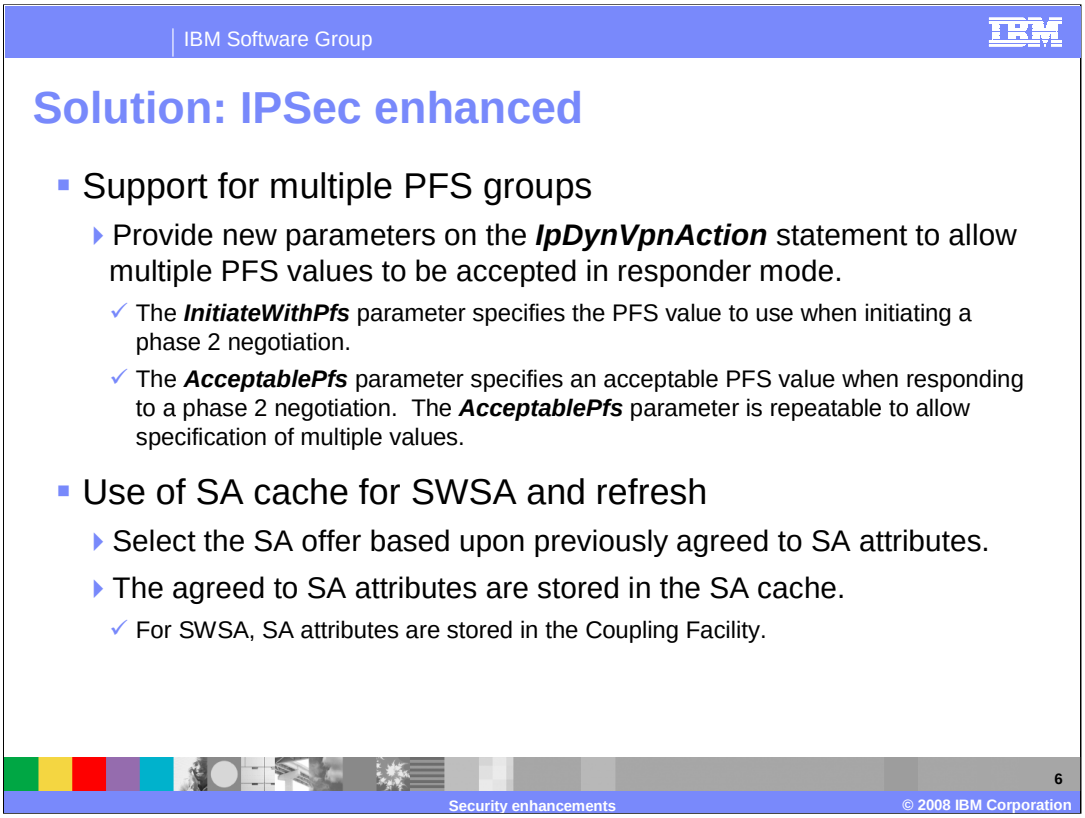
The IKE daemon negotiates the dynamic IP security associations. Security associations are negotiated in two phases. Phase 1 SAs are established first and they protect the phase 2 negotiations. You have a choice of two modes: Aggressive Mode or Main Mode. Phase 2 SAs protect data traffic and uses Quick Mode.

SWSA allows for IPSec SAs to be automatically reestablished during a DVIPA takeover or giveback. SA refreshes are performed before the expiration of the existing SA.




The usage of PFS is optional in a phase 2 negotiation. For maximum interoperability, servers should be able to accept multiple PFS values. Some clients may only be able to support lower PFS groups, while other clients may support higher PFS groups for higher security.

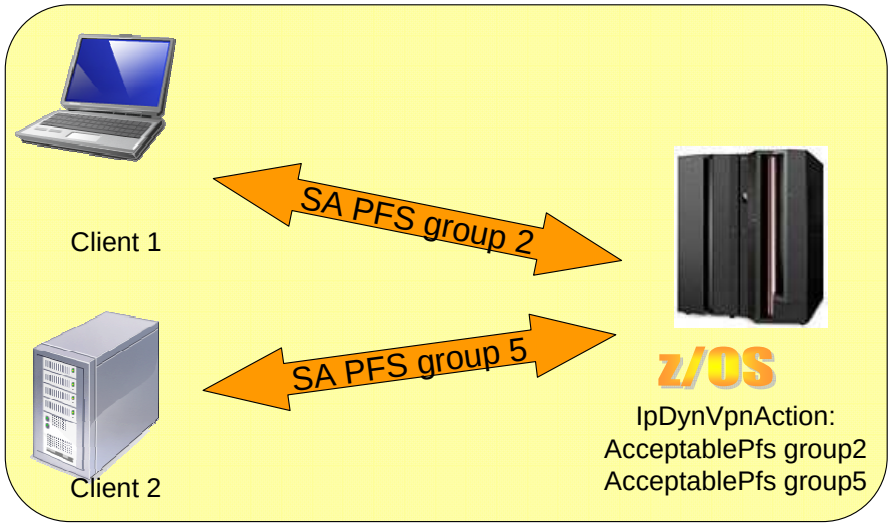
During a takeover/giveback or refresh using Aggressive or quick mode, the first offer is used which may not be the offer used in the establishment of the SA when multiple offers are configured. In Main mode all of the SA offers are sent to the peer and the peer can select an acceptable offer.




New parameters are added to the IpDynVpnAction statement to allow you to configure multiple PFS values to be accepted in responder mode. The InitiateWIthPfs and AcceptablePfs parameters should now be used instead of the Pfs parameter.

With a SA cache the correct offer is used for SWSA takeover and giveback and a SA refresh. The SA cache is located in the IKED private storage.

# Example using multiple PFS groups

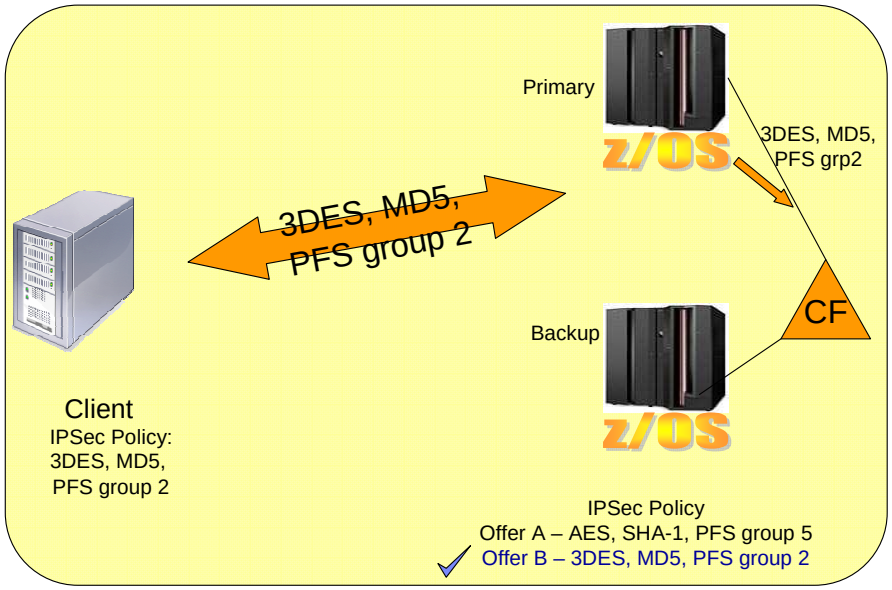






The two clients use different PFS groups when establishing a SA with the z/OS server. The z/OS server is able to respond to both clients by configuring a single IpDynVpnAction to support multiple PFS groups.

# SWSA example - Initial SA establishment

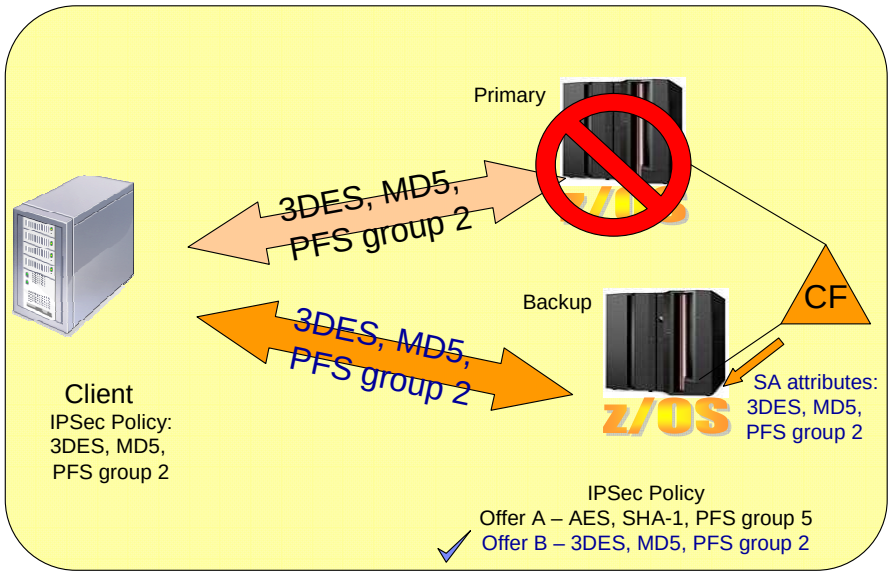






The client on the left establishes an SA with the z/OS sysplex. The SA information is stored in the coupling facility.

# SWSA example takeover







When the primary z/OS box becomes unavailable, the backup performs a SWSA takeover to re-establish the SA with the client. The SA attributes are retrieved from the coupling facility and the correct SA offer is used for the negotiation.

## Diagnosis

- Policy configuration rules:
  - New ***InitiateWithPfs*** and ***AcceptablePfs*** parameters cannot be used with ***Pfs*** parameter in the same ***IpDynVpnAction*** statement.
  - The ***InitiateWithPfs*** value must be specified as one of the values specified by ***AcceptablePfs***.

```
InitiatePfs     Group5
AcceptablePfs   Group2
AcceptablePfs   Group5
```



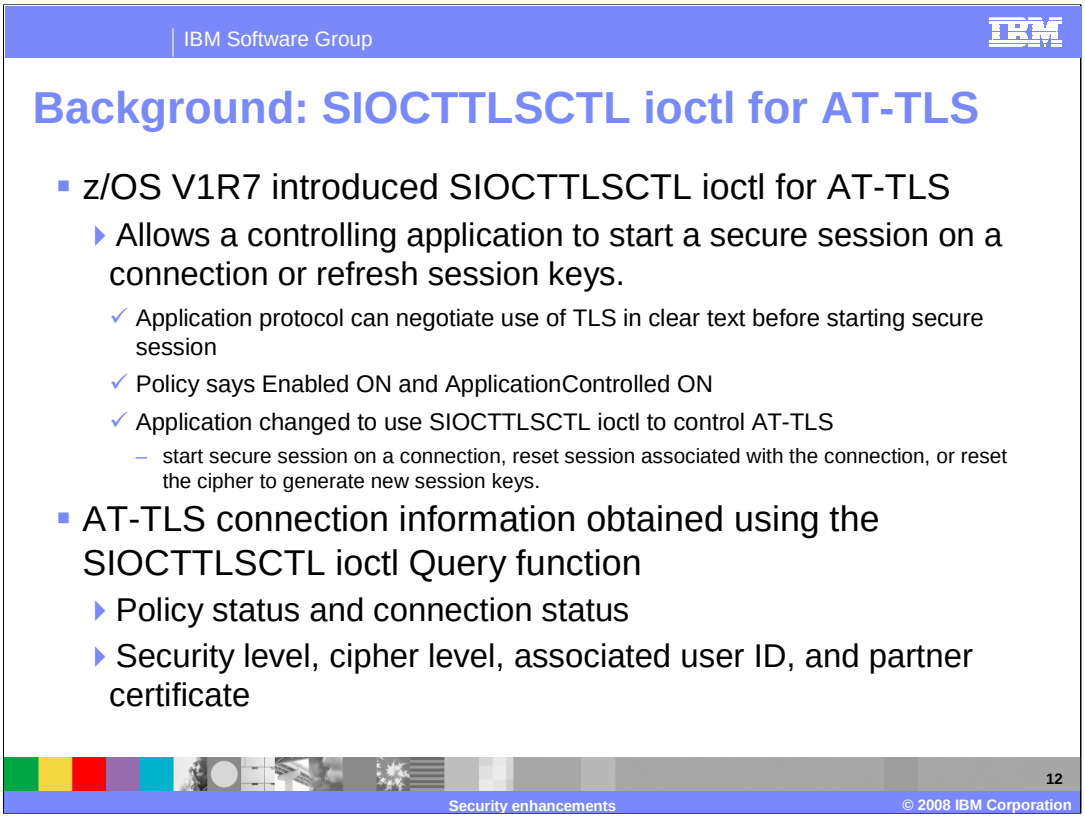

Policy agent enforces that the *InitiateWithPfs* value must be specified as one of the values specified by *AcceptablePfs*. Otherwise, SA refreshes and SWSA takeovers would fail.

## Section

***AT-TLS API enhancements***

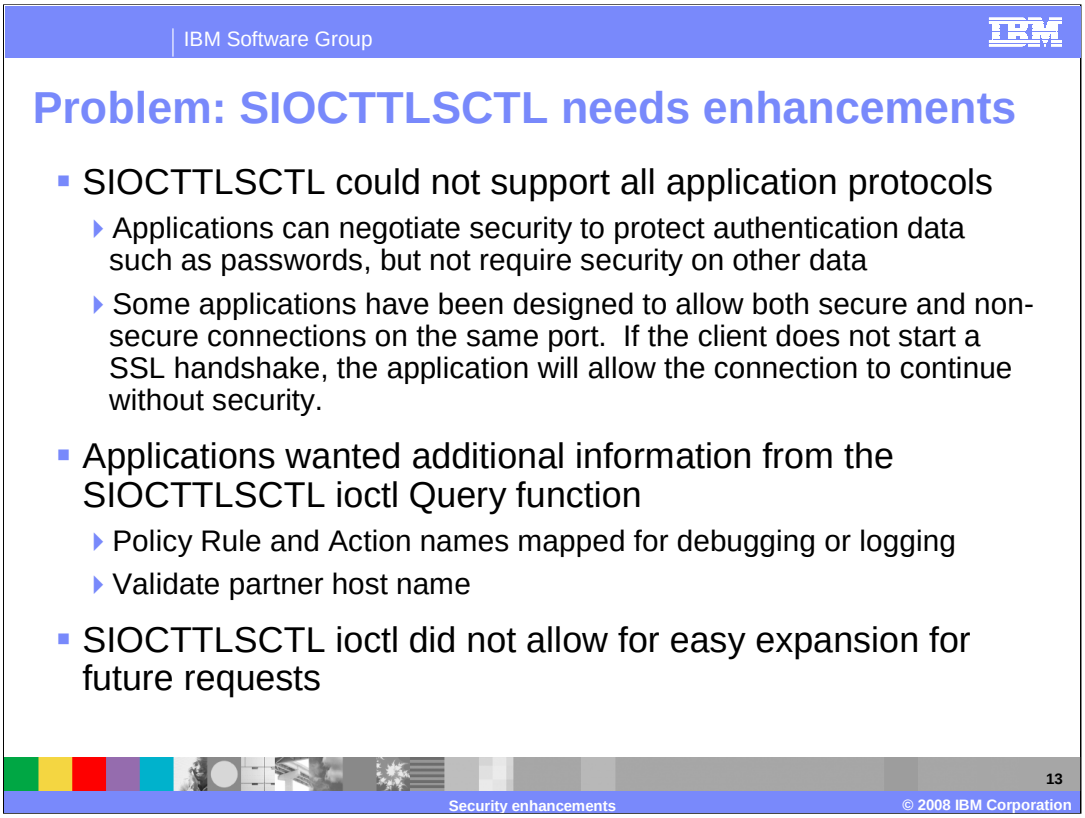



This section covers enhancements to the AT-TLS API using the existing SIOCTTLSCTL ioctl.




z/OS 1.7 introduced Application Transparent Transport Layer Security (AT-TLS) and the SIOCTTLSCTL ioctl. This allowed applications to control AT-TLS security on a connection. The application starts security on the connection. The application can also reset the cipher being used to generate new session keys for the connection or reset the session associated with the connection to force a full SSL handshake. This type of application is called a controlling application. The AT-TLS policy must be defined with ApplicationControlled On.

The SIOCTTLSCTL ioctl currently can be used to obtain information about the connection. The state of the connection (secure, not secure, or handshake in progress) and the policy status (unknown, client, server or server with client authentication) can be obtained. For secure connections, the security level(SSLv2, SSLv3 or TLSv1) and the negotiated cipher can be obtained. For connections which the certificate has been received, the certificate and associated user ID can be obtained.




Many applications use a secure connection for sensitive data during the connection. After this data exchange, security is no longer needed for the connection. The application will stop security on the connection, reducing the processor overhead of security. Some applications also support both secure and non-secure connections on the same port. These applications detect which type of client has connected and act accordingly. These type of applications could not use the SIOCTTLSCTL ioctl to implement security.

Additional information is available about the connection using netstat. Applications can use the policy rule and action names for debugging purposes. Some applications need to validate the host name received in the partner certificate against the host name they have connected to as described in RFC 2818. The SIOCTTLSCTL ioctl did not allow for additional functions to be easily defined.

# Solution: SIOCTTLSCTL ioctl enhanced

- New SIOCTTLSCTL Request options
  - TTLS_Stop_Connection – Stops security on a connection, allowing clear text to be sent
    - The connection returns to clear text communication after the stop completes
    - Restrictions
      - Stop not supported on SSLv2 connections
      - All application data must be read before the stop is issued.
  - TTLS_Allow_HSTimeout – If non-SSL data is received or the SSL handshake times out because no SSL data is received, the connection is allowed to continue.
    - This option only applies to a SSL handshake on a clear text connection, so it must be combined with TTLS_Init_Connection option.
    - Restrictions
      - The AT-TLS policy must specify a non-zero HandshakeTimeout value.
      - The AT-TLS policy HandshakeRole must be Server or ServerWithClientAuth
- Additional information can be requested on the Query function
  - A new structure defined to pass requests
    - TTLSHeader – defines structure
    - TTLSQuadruplet defines each request
  - New requests
    - Retrieve TtlsRule name, TtlsGroupAction name, TtlsEnvironmentAction name and TtlsConnectionAction name
    - Validate partner host name
- SIOCTTLSCTL IOCTL data structures updated




Two new options now are defined for the SIOCTTLSCTL ioctl. TTLS_Stop_Connection allows the application to stop security on a connection. The SSL security on the connection is stopped and subsequent data is sent as clear text. The TTLS_Stop_Connection request behaves differently for blocking and non-blocking sockets. For blocking sockets, the ioctl will return once the stop completes. For non-blocking sockets, the ioctl will return immediately with EInProgress. The application can use a select for write to determine when the stop is complete. Subsequent data on the socket is sent in clear text. SSLv2 does not support any type of close notification, so stop is not supported on SSLv2 connections. All application data needs to be read before the stop is issued.

TTLS_Allow_HSTimeout will allow the SSL handshake to timeout if no SSL data is received from the client or if clear text data is received. This option is only valid with TTLS_Init_Connection since it only applies to a SSL handshake on a clear text connection. The AT-TLS policy must have a non-zero HandshakeTimeout value. This is required so that the handshake will not hang indefinitely. TTLS_Allow_HSTimeout can only be used when the application is acting as the server in the SSL handshake. SSL handshakes always start with the client sending a SSL hello.

A new structure is created, the TTLSHeader. The TTLSHeader contains control information about the number of requests contained in the buffer. Each request is represented by a TTLSQuadruplet, which defines the request. The TTLSHeader is pointed to be the existing TTLSi_BufferPtr. TTLSK_TTLSRule_Name, TTLSK_TTLSGroupAction_Name, TTLSK_TTLSEnvironmentAction_Name, and TTLSK_TTLSConnectionAction_Name are the **TTLSQ_Key** values used on the ioctl to **r**etrieve policy rule and action names. The policy rule and action names will return up to a 48 character buffer with a null character terminated. TTLSK_Certificate is used to retrieve the partners certificate. It is equivalent to a TTLSi_Return_Certificate request. TTLSK_Host_Status is used to validate the host name from the partner certificate against a host name supplied in the TTLSHeader buffer. A host name must be passed, pointed to by the TTLSQuadruplet. The TTLSQ_Rcode is set upon return.

The SIOCTTLSCTL IOCTL data structures has been updated for all supported languages below.

Assembler – ezbztlsp.macro

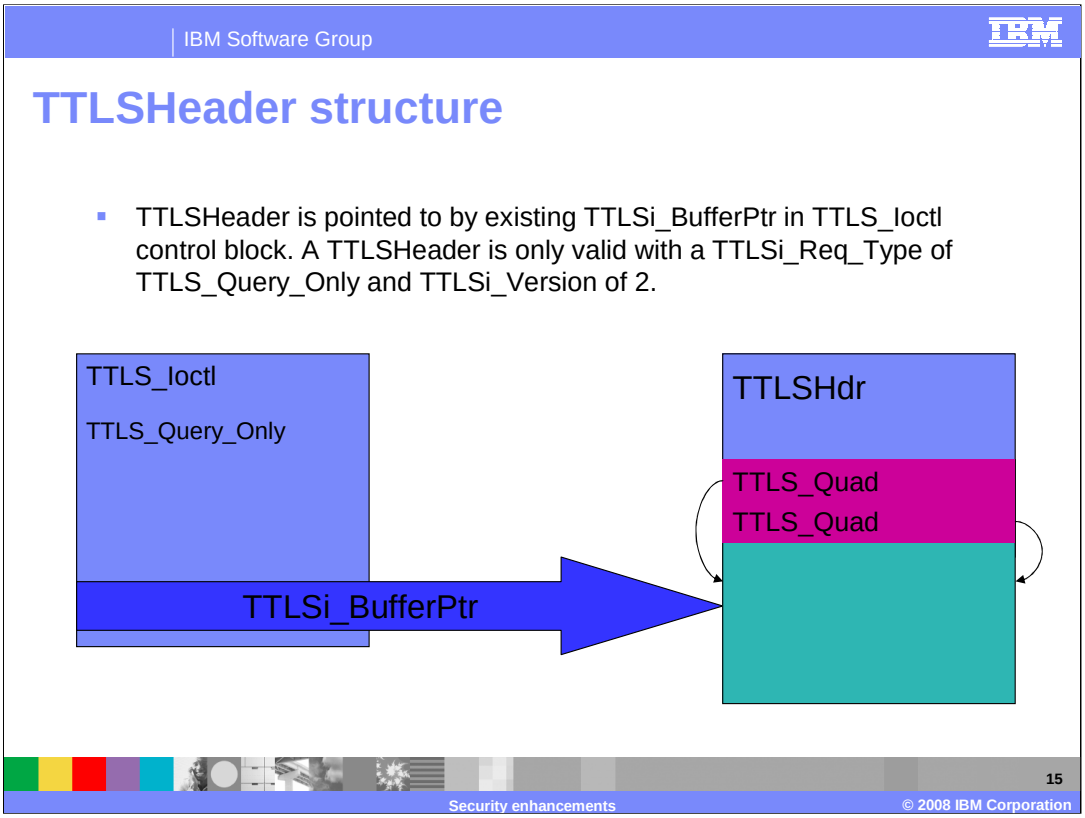
C - ezbtlsc.h

Cobol – ezbztlsb.sample

PL/1 – ezbztls1.sample

Rexx sockets – STOPCONNECTION and INITCONNHSTIMEOUT requests defined

Rexx sockets has a unique constant defined, INITCONNHSTIMEOUT, which combines the TTLS_Init_Connection and TTLS_Allow_HSTimeout options. Rexx sockets have constants for the AT-TLS Query functions. QueryRuleName, QueryGroupActionName, QueryEnvironmentActionName, and QueryConnectionActionName are used to retrieve policy rule and action names QueryHost accepts a host name as a parameter to validate against the partner certificate. Rexx sockets do not have access to the partner certificate directly.




The TTLSHeader is pointed to by the existing TTLSi_BufferPtr. The version in the SIOCTTLSCTL request must be set to 2. The TTLSHeader is only supported when the TTLSi_Req_Type is set to TTLS_Query_Only(0). The TTLSHeader must be first in storage and contains the number of requests in the buffer. The TTLS_Quadruplets immediately follow the TTLSHeader. Each TTLS_Quadruplet can point into the buffer for the request. For example, the host name to be compared against a partner certificate would be after the TTLSQuadruplet and pointed to be the TTLSQuadruplet. Upon return, the TTLSQuadruplet is updated to point to the returned information, if any.

## Diagnosis

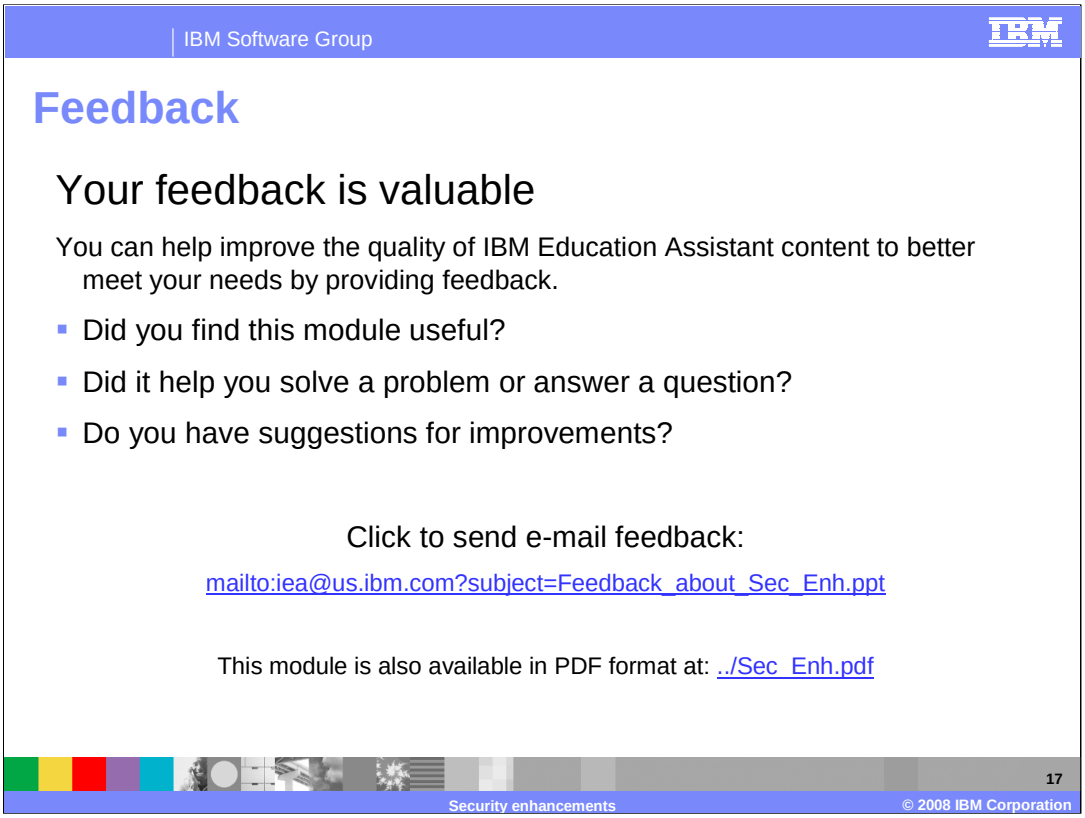
- Check return code values
  - EPerm
    - TTLS_Stop_Connection combined with other TTLSi_Req_Type
    - TTLS_Allow_HSTimeout without TTLS_Init_Connection
  - EProto with reason code JrConnDeniedPolicy
    - TTLS_Allow_HSTimeout but policy has HandshakeRole of client or HandshakeTimeout value of 0.
  - ENoBufs for Query functions
    - Use TTLSHdr_BytesNeeded to determine the size of buffer needed
- Use ctrace option IOCTL or SOCKAPI to trace TTLSHeader buffer and return values




The return code and reason code can be used to determine why the SIOCTTLSCTL ioctl request failed.

If the TTLSHeader buffer does not have enough room to hold the returned information, the return code is set to ENoBufs. The TTLSHdr_BytesNeeded field is updated with the amount of storage required to complete the request.

The IOCTL ctrace option will trace the TTLSHeader control block and the return and reason codes The SOCKAPI option can also be used to trace these requests.




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 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.