

Inspection for Mobile Networks

The following topics explain application inspection for protocols used in mobile networks such as LTE. These inspections require the Carrier license. For information on why you need to use inspection for certain protocols, and the overall methods for applying inspection, see Getting Started with Application Layer Protocol Inspection.

- Mobile Network Inspection Overview, on page 1
- Licensing for Mobile Network Protocol Inspection, on page 8
- Defaults for GTP Inspection, on page 8
- Configure Mobile Network Inspection, on page 9
- Monitoring Mobile Network Inspection, on page 39
- History for Mobile Network Inspection, on page 44

Mobile Network Inspection Overview

The following topics explain the inspections available for protocols used in mobile networks such as LTE. There are other services available for SCTP traffic in addition to inspection.

GTP Inspection Overview

GPRS Tunneling Protocol is used in GSM, UMTS and LTE networks for general packet radio service (GPRS) traffic. GTP provides a tunnel control and management protocol to provide GPRS network access for a mobile station by creating, modifying, and deleting tunnels. GTP also uses a tunneling mechanism for carrying user data packets.

Service provider networks use GTP to tunnel multi-protocol packets through the GPRS backbone between endpoints. In GTPv0-1, GTP is used for signaling between gateway GPRS support nodes (GGSN) and serving GPRS support nodes (SGSN). In GTPv2, the signaling is between Packet Data Network Gateways (PGW) and the Serving Gateway (SGW) as well as other endpoints. The GGSN/PGW is the interface between the GPRS wireless data network and other networks. The SGSN/SGW performs mobility, data session management, and data compression.

You can use the ASA to provide protection against rogue roaming partners. Place the device between the home GGSN/PGW and visited SGSN/SGW endpoints and use GTP inspection on the traffic. GTP inspection works only on traffic between these endpoints. In GTPv2, this is known as the S5/S8 interface.

GTP and associated standards are defined by 3GPP (3rd Generation Partnership Project). For detailed information, see http://www.3gpp.org.

Tracking Location Changes for Mobile Stations

You can use GTP inspection to track location changes for mobile stations. Tracking location changes might help you identify fraudulent roaming charges, for example, if you see a mobile station move from one location to another within an unlikely time window, such as moving from a cell in the United States to one in Europe within 30 minutes.

When you enable location logging, the system generates syslog messages for new or changed location for each International Mobile Subscriber Identity (IMSI):

- 324010 indicates the creation of a new PDP context, and includes the Mobile Country Code (MCC), Mobile Network Code (MNC), the information elements, and optionally the cell ID where the user currently is registered. The cell ID is extracted from the Cell Global Identification (CGI) or E-UTRAN Cell Global Identifier (ECGI).
- 324011 indicates that the IMSI has moved from the one stored during the PDP context creation. The message shows the previous and current MCC/MNC, information elements, and optionally, cell ID.

By default, syslog messages do not include timestamp information. If you plan to analyze these messages to identify improbable roaming, you must also enable timestamps. Timestamp logging is not part of the GTP inspection map. Use the **logging timestamp** command.

For information on enabling location logging, see Configure a GTP Inspection Policy Map, on page 9.

GTP Inspection Limitations

Following are some limitations on GTP inspection:

- GTPv2 piggybacking messages are not supported. They are always dropped.
- GTPv2 emergency UE attach is supported only if it contains IMSI (International Mobile Subscriber Identity).
- GTP inspection does not inspect early data. That is, data sent from a PGW or SGW right after a Create Session Request but before the Create Session Response.
- For GTPv2, inspection supports up to 3GPP 29.274 V15.5.0. For GTPv1, support is up to 3GPP 29.060 V15.2.0. For GTPv0, support is up to release 8.
- GTP inspection does not support inter-SGSN handoff to the secondary PDP context. Inspection needs to do the handoff for both primary and secondary PDP contexts.

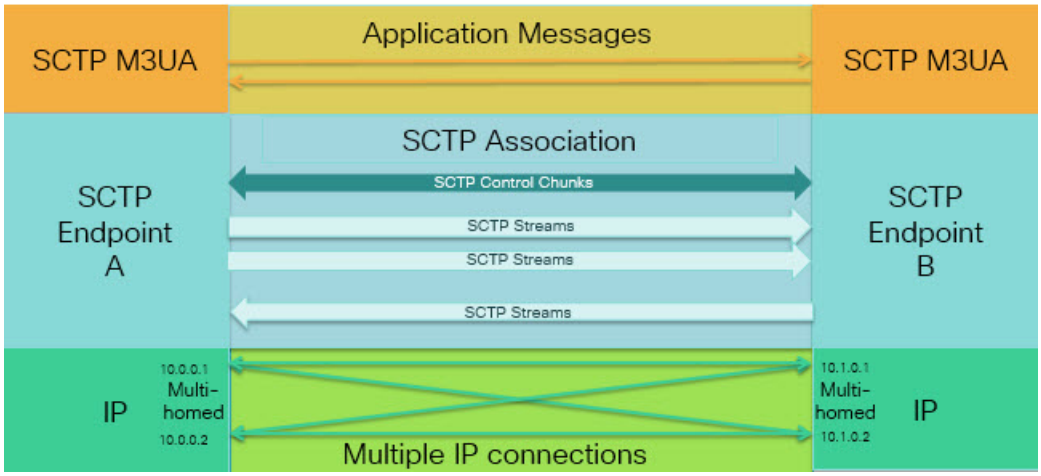
Stream Control Transmission Protocol (SCTP) Inspection and Access Control

SCTP (Stream Control Transmission Protocol) is described in RFC 4960. The protocol supports the telephony signaling protocol SS7 over IP and is also a transport protocol for several interfaces in the 4G LTE mobile network architecture.

SCTP is a transport-layer protocol operating on top of IP in the protocol stack, similar to TCP and UDP. However, SCTP creates a logical communication channel, called an association, between two end nodes over one or more source or destination IP addresses. This is called multi-homing. An association defines a set of IP addresses on each node (source and destination) and a port on each node. Any IP address in the set can be used as either a source or a destination IP address of data packets associated to this association to form multiple connections. Within each connection, multiple streams may exist to send messages. A stream in SCTP represents a logical application data channel.

The following figure illustrates the relationship between an association and its streams.

Figure 1: Relationship Between SCTP Association and Streams



If you have SCTP traffic going through the ASA, you can control access based on SCTP ports, and implement application layer inspection to enable connections and to optionally filter on payload protocol ID to selectively drop, log, or rate limit applications.

Note Each node can have up to three IP addresses. Any addresses over the limit of three are ignored and not included in the association. Pinholes for secondary IP addresses are opened automatically. You do not need to write access control rules to allow them.

Heartbeats and Association State

To maintain the association and ensure accurate reporting, the inspection engine monitors SCTP heartbeats. The system links the connection duration to the heartbeat signals. If the heartbeat is lost or the association is explicitly terminated, the device closes the connection and finalizes the event duration.

SCTP Association and Port Reporting

SCTP inspection is association-based rather than flow-based. Because the device tracks the entire association as a single unit, it does not map individual stream ports to traditional port fields in the logging output. As a result, when viewing connection details, you may observe the source and destination ports displayed as `0`. This is because the association is uniquely identified by the IP addresses and the association ID, rather than by individual source/destination port pairs.

SCTP Stateful Inspection

Similar to TCP, SCTP traffic is automatically inspected at layer 4 to ensure well-structured traffic and limited RFC 4960 enforcement. The following protocol elements are inspected and enforced:

- Chunk types, flags, and length.
- Verification tags.
- Source and destination ports, to prevent association redirect attacks.

- IP addresses.

SCTP stateful inspection accepts or rejects packets based on the association state:

- Validating the 4-way open and close sequences for initial association establishment.
- Verifying the forward progression of TSN within an association and a stream.
- Terminating an association when seeing the ABORT chunk due to heartbeat failure. SCTP endpoints might send the ABORT chunk in response to bombing attacks.

If you decide you do not want these enforcement checks, you can configure SCTP state bypass for specific traffic classes, as explained in Configure Connection Settings for Specific Traffic Classes (All Services).

SCTP Access Control

You can create access rules for SCTP traffic. These rules are similar to TCP/UDP port-based rules, where you simply use **sctp** as the protocol, and the port numbers are SCTP ports. You can create service objects or groups for SCTP, or specify the ports directly. See the following topics.

- Configure Service Objects and Service Groups
- Add an Extended ACE for Port-Based Matching

SCTP NAT

You can apply static network object NAT to the addresses in SCTP association establishment messages. Although you can configure static twice NAT, this is not recommended because the topology of the destination part of the SCTP association is unknown. You cannot use dynamic NAT/PAT.

NAT for SCTP depends upon SCTP stateful inspection rather than SCTP application-layer inspection. Thus, you cannot NAT traffic if you configure SCTP state bypass.

SCTP Application Layer Inspection

You can further refine your access rules by enabling SCTP inspection and filtering on SCTP applications. You can selectively drop, log, or rate limit SCTP traffic classes based on the payload protocol identifier (PPID).

If you decide to filter on PPID, keep the following in mind:

- PPIDs are in data chunks, and a given packet can have multiple data chunks or even a control chunk. If a packet includes a control chunk or multiple data chunks, the packet will not be dropped even if the assigned action is drop.
- If you use PPID filtering to drop or rate-limit packets, be aware that the transmitter will resend any dropped packets. Although a packet for a rate-limited PPID might make it through on the next attempt, a packet for a dropped PPID will again be dropped. You might want to evaluate the eventual consequence of these repeated drops on your network.

SCTP Limitations

SCTP support includes the following limitations.

- Each node can have up to three IP addresses. Any addresses over the limit of three are ignored and not included in the association. Pinholes for secondary IP addresses are opened automatically. You do not need to write access control rules to allow them.
- Unused pinholes time out in 5 minutes.
- Dual stack IPv4 and IPv6 addresses on multi-homed endpoints is not supported.
- Network object static NAT is the only supported type of NAT. Also, NAT46 and NAT64 are not supported.
- Fragmentation and reassembly of SCTP packets is done only for traffic handled by Diameter, M3UA, and SCTP PPID-based inspection.
- ASCONF chunks, which are used to dynamically add or delete IP addresses in SCTP, are not supported.
- The Hostname parameter in INIT and INIT-ACK SCTP messages, which is used to specify a hostname which can then be resolved to an IP address, is not supported.
- SCTP/M3UA does not support equal-cost multipath routing (ECMP), whether configured on the ASA or elsewhere in the network. With ECMP, packets can be routed to a destination over multiple best paths. However, an SCTP/M3UA packet response to a single destination has to come back on the same interface that it exited. Even though the response can come from any M3UA server, it must always come back on the same interface that it exited. The symptom for this problem is that SCTP INIT-ACK packets are dropped, which you can see in the **show asp drop flow sctp-chunk-init-timeout** counter:

  ```
  Flow drop:
  SCTP INIT timed out (not receiving INIT ACK)(sctp-chunk-init-timeout)
  ```

  If you encounter this problem, you can resolve it by configuring static routes to the M3UA servers, or by configuring policy-based routing to implement a network design that ensures that INIT-ACK packets go through the same interface as the INIT packets.

Diameter Inspection

Diameter is an Authentication, Authorization, and Accounting (AAA) protocol used in next-generation mobile and fixed telecom networks such as EPS (Evolved Packet System) for LTE (Long Term Evolution) and IMS (IP Multimedia Subsystem). It replaces RADIUS and TACACS in these networks.

Diameter uses TCP and SCTP as the transport layer, and secures communications using TCP/TLS and SCTP/DTLS. It can optionally provide data object encryption as well. For detailed information on Diameter, see RFC 6733.

Diameter applications perform service management tasks such as deciding user access, service authorization, quality of service, and rate of charging. Although Diameter applications can appear on many different control-plane interfaces in the LTE architecture, the ASA inspects Diameter command codes and attribute-value pairs (AVP) for the following interfaces only:

- S6a: Mobility Management Entity (MME) - Home Subscription Service (HSS).
- S9: PDN Gateway (PDG) - 3GPP AAA Proxy/Server.
- Rx: Policy Charging Rules Function (PCRF) - Call Session Control Function (CSCF).

Diameter inspection opens pinholes for Diameter endpoints to allow communication. The inspection supports 3GPP version 12 and is RFC 6733 compliant. You can use it for TCP/TLS (by specifying a TLS proxy when

you enable inspection) and SCTP, but not SCTP/DTLS. Use IPsec to provide security to SCTP Diameter sessions.

You can optionally use a Diameter inspection policy map to filter traffic based on application ID, command codes, and AVP, to apply special actions such as dropping packets or connections, or logging them. You can create custom AVP for newly-registered Diameter applications. Filtering lets you fine-tune the traffic you allow on your network.



Note Diameter messages for applications that run on other interfaces will be allowed and passed through by default. However, you can configure a Diameter inspection policy map to drop these applications by application ID, although you cannot specify actions based on the command codes or AVP for these unsupported applications.

M3UA Inspection

MTP3 User Adaptation (M3UA) is a client/server protocol that provides a gateway to the SS7 network for IP-based applications that interface with the SS7 Message Transfer Part 3 (MTP3) layer. M3UA makes it possible to run the SS7 User Parts (such as ISUP) over an IP network. M3UA is defined in RFC 4666.

M3UA uses SCTP as the transport layer. SCTP port 2905 is the default port.

The MTP3 layer provides networking functions such as routing and node addressing, but uses point codes to identify nodes. The M3UA layer exchanges Originating Point Codes (OPC) and Destination Point Codes (DPC). This is similar to how IP uses IP addresses to identify nodes.

M3UA inspection provides limited protocol conformance. You can optionally implement strict application server process (ASP) state checking and additional message validation for select messages. Strict ASP state checking is required if you want stateful failover or if you want to operate within a cluster. However, strict ASP state checking works in Override mode only, it does not work if you are running in Loadsharing or Broadcast mode (per RFC 4666). The inspection assumes there is one and only one ASP per endpoint.

You can optionally apply access policy based on point codes or Service Indicators (SI). You can also apply rate limiting based on message class and type.

M3UA Protocol Conformance

M3UA inspection provides the following limited protocol enforcement. Inspection drops and logs packets that do not meet requirements.

- Common message header. Inspection validates all fields in the common header.
  - Version 1 only.
  - Message length must be correct.
  - Message type class with a reserved value is not allowed.
  - Invalid message ID within the message class is not allowed.
- Payload data message.
  - Only one parameter of a given type is allowed.
  - Data messages on SCTP stream 0 are not allowed.

- The Affected Point Code field must be present in the following messages or the message is dropped: Destination Available (DAVA), Destination Unavailable (DUNA), Destination State Audit (DAUD), Signaling Congestion (SCON), Destination User Part Unavailable (DUPU), Destination Restricted (DRST).
- If you enable message tag validation for the following messages, the content of certain fields are checked and validated. Messages that fail validation are dropped.
  - Destination User Part Unavailable (DUPU)—The User/Cause field must be present, and it must contain only valid cause and user codes.
  - Error—All mandatory fields must be present and contain only allowed values. Each error message must contain the required fields for that error code.
  - Notify—The status type and status information fields must contain allowed values only.
- If you enable strict application server process (ASP) state validation, the system maintains the ASP states of M3UA sessions and allows or drops ASP messages based on the validation result. If you do not enable strict ASP state validation, all ASP messages are forwarded uninspected.

M3UA Inspection Limitations

Following are some limitations on M3UA inspection.

- NAT is not supported for IP addresses that are embedded in M3UA data.
- M3UA strict application server process (ASP) state validation depends on SCTP stateful inspection. Do not implement SCTP state bypass and M3UA strict ASP validation on the same traffic.
- Strict ASP state checking is required if you want stateful failover or if you want to operate within a cluster. However, strict ASP state checking works in Override mode only, it does not work if you are running in Loadsharing or Broadcast mode (per RFC 4666). The inspection assumes there is one and only one ASP per endpoint.

RADIUS Accounting Inspection Overview

The purpose of RADIUS accounting inspection is to prevent over-billing attacks on GPRS networks that use RADIUS servers. Although you do not need the Carrier license to implement RADIUS accounting inspection, it has no purpose unless you are implementing GTP inspection and you have a GPRS setup.

The over-billing attack in GPRS networks results in consumers being billed for services that they have not used. In this case, a malicious attacker sets up a connection to a server and obtains an IP address from the SGSN. When the attacker ends the call, the malicious server will still send packets to it, which gets dropped by the GGSN, but the connection from the server remains active. The IP address assigned to the malicious attacker gets released and reassigned to a legitimate user who will then get billed for services that the attacker will use.

RADIUS accounting inspection prevents this type of attack by ensuring the traffic seen by the GGSN is legitimate. With the RADIUS accounting feature properly configured, the ASA tears down a connection based on matching the Framed IP attribute in the Radius Accounting Request Start message with the Radius Accounting Request Stop message. When the Stop message is seen with the matching IP address in the Framed IP attribute, the ASA looks for all connections with the source matching the IP address.

You have the option to configure a secret pre-shared key with the RADIUS server so the ASA can validate the message. If the shared secret is not configured, the ASA will only check that the source IP address is one of the configured addresses allowed to send the RADIUS messages.

Note When using RADIUS accounting inspection with GPRS enabled, the ASA checks for the 3GPP-Session-Stop-Indicator in the Accounting Request STOP messages to properly handle secondary PDP contexts. Specifically, the ASA requires that the Accounting Request STOP messages include the 3GPP-SGSN-Address attribute before it will terminate the user sessions and all associated connections. Some third-party GGSNs might not send this attribute by default.

Licensing for Mobile Network Protocol Inspection

Inspection of the following protocols requires the license listed in the table below.

- GTP
- SCTP.
- Diameter
- M3UA

Model	License Requirement
ASA Virtual (all models)	Carrier license (enabled by default)
Secure Firewall 3100	Carrier license
Firepower 4100	Carrier license
Firepower 9300	Carrier license
All other models	The Carrier license is not available on other models. You cannot inspect these protocols.

Defaults for GTP Inspection

GTP inspection is not enabled by default. However, if you enable it without specifying your own inspection map, a default map is used that provides the following processing. You need to configure a map only if you want different values.

- Errors are not permitted.
- The maximum number of requests is 200.
- The maximum number of tunnels is 500. This is equivalent to the number of PDP contexts (endpoints).
- The GTP endpoint timeout is 30 minutes. Endpoints include GSNs (GTPv0,1) and SGW/PGW (GTPv2).
- The PDP context timeout is 30 minutes. In GTPv2, this is the bearer context timeout.

- The request timeout is 1 minute.
- The signaling timeout is 30 minutes.
- The tunneling timeout is 1 hour.
- The T3 response timeout is 20 seconds.
- Unknown message IDs are allowed. You can configure **match message v1/v2 id range** commands to drop and log any commands that you do not support or want to allow. Messages are considered unknown if they are either undefined or are defined in GTP releases that the system does not support.

Configure Mobile Network Inspection

Inspections for protocols used in mobile networks are not enabled by default. You must configure them if you want to support mobile networks.

Procedure

Step 1 (Optional.) Configure a GTP Inspection Policy Map, on page 9.

Step 2 (Optional.) Configure an SCTP Inspection Policy Map, on page 14.

Step 3 (Optional.) Configure a Diameter Inspection Policy Map, on page 15.

If you want to filter on attribute-value pairs (AVP) that are not yet supported in the software, you can create custom AVP for use in the Diameter inspection policy map. See Create a Custom Diameter Attribute-Value Pair (AVP), on page 19.

Step 4 (Optional.) If you want to inspect encrypted Diameter TCP/TLS traffic, create the required TLS proxy as described in Inspecting Encrypted Diameter Sessions, on page 20

Step 5 (Optional.) Configure an M3UA Inspection Policy Map, on page 31

Step 6 Configure the Mobile Network Inspection Service Policy , on page 35.

Step 7 (Optional.) Configure RADIUS Accounting Inspection, on page 36.

RADIUS accounting inspection protects against over-billing attacks.

Configure a GTP Inspection Policy Map

If you want to enforce additional parameters on GTP traffic, and the default map does not meet your needs, create and configure a GTP map.

Before you begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 Create a GTP inspection policy map: **policy-map type inspect gtp** *policy_map_name*

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) Add a description to the policy map: **description** *string*

Step 3 To apply actions to matching traffic, perform the following steps.

a) Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **apn regex** {*regex_name* | **class** *class_name*}—Matches the access point name (APN) against the specified regular expression or regular expression class.
- **match** [**not**] **message** {**v1** | **v2**} **id** {*message_id* | **range** *message_id_1 message_id_2*}—Matches the message ID, which can be 1 to 255. You can specify a single ID or a range of IDs. You must specify whether the message is for GTPv0/1 (**v1**) or GTPv2 (**v2**).
- **match** [**not**] **message length min** *bytes* **max** *bytes*—Matches messages where the length of the UDP payload (GTP header plus the rest of the message) is between the minimum and maximum values, from 1 to 65536.
- **match** [**not**] **msisdn regex** {*regex_name* | **class** *class_name*}—Matches the Mobile Station International Subscriber Directory Number (MSISDN) information element in the Create PDP Context request, Create session request, and Modify Bearer Response messages against the specified regular expression or regular expression class. The regular expression can identify a specific MSISDN, or a range of MSISDNs based on the first x number of digits. MSISDN filtering is supported for GTPv1 and GTPv2 only.
- **match** [**not**] **selection-mode** *mode_value*—Matches the Selection Mode information element in the Create PDP Context request. The selection mode specifies the origin of the Access Point Name (APN) in the message, and can be one of the following. Selection Mode filtering is supported for GTPv1 and GTPv2 only.
  - 0—Verified. The APN was provided by the mobile station or network, and the subscription is verified.
  - 1—Mobile Station. The APN was provided by the mobile station, and the subscription is not verified.
  - 2—Network. The APN was provided by the network, and the subscription is not verified.
  - 3—Reserved, not used.
- **match** [**not**] **version** {*version_id* | **range** *version_id_1 version_id_2*}—Matches the GTP version, which can be 0 to 255. You can specify a single version or a range of versions.

b) Specify the action you want to perform on the matching traffic by entering one of the following commands:

- **drop** [**log**]—Drop all packets that match. Add the **log** keyword to also send a system log message.
- **rate-limit** *message_rate*—Limit the rate of messages. This option is available with **message id** only.

You can specify multiple **match** commands in the policy map. For information about the order of **match** commands, see How Multiple Traffic Classes are Handled.

Step 4 To configure parameters that affect the inspection engine, perform the following steps:

a) Enter parameters configuration mode:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b) Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **anti-replay** [*window_size*]—Enable anti-replay by specifying a sliding window for GTP-U messages. The size of the sliding window is in number of messages and can be 128, 256, 512, or 1024. If you do not specify a size, you get the default, 512. As valid messages appear, the window moves to the new sequence numbers. Sequence numbers are in the range 0-65535, wrapping when they reach the maximum, and they are unique per PDP context. Messages are considered valid if their sequence numbers are within the window. Anti-replay helps prevent session hijacking or DoS attacks, which can occur when a hacker captures GTP data packets and replays them.
- **permit errors**—Allows all packets, including invalid GTP packets or packets that otherwise would fail parsing and be dropped. Packets can still be dropped based on the actions you define in the policy map.
- **request-queue** *max_requests*—Sets the maximum number of GTP requests that will be queued waiting for a response. The default is 200. When the limit has been reached and a new request arrives, the request that has been in the queue for the longest time is removed. The Error Indication, the Version Not Supported and the SGSN Context Acknowledge messages are not considered as requests and do not enter the request queue to wait for a response.
- **tunnel-limit** *max_tunnels*—Sets the maximum number of active GTP tunnels allowed. This is equivalent to the number of PDP contexts or endpoints. The default is 500. New requests will be dropped once the number of tunnels specified by this command is reached.
- **timeout** {**endpoint** | **pdp-context** | **request** | **signaling** | **t3-response** | **tunnel**} *time*—Sets the idle timeout for the specified service (in hh:mm:ss format). To have no timeout, specify 0 for the number. Enter the command separately for each timeout.
  - **endpoint**—The maximum period of inactivity before a GTP endpoint is removed.
  - **pdp-context**—The maximum period of inactivity before removing the PDP Context for a GTP session. In GTPv2, this is the bearer context.
  - **request**—The maximum period of inactivity after which a request is removed from the request queue. Any subsequent responses to a dropped request will also be dropped.
  - **signaling**—The maximum period of inactivity before GTP signaling is removed.
  - **t3-response**—The maximum wait time for a response before removing the connection.
  - **tunnel**—The maximum period of inactivity for the GTP tunnel before it is torn down.

Step 5 While still in parameter configuration mode, configure GTP-U checking for IP packets and anti-spoofing.

gtp-u-header-check [**anti-spoofing** [**gtpv2-dhcp-bypass** | **gtpv2-dhcp-drop**]]

Without keywords, this command checks whether the inner payload of a GTP data packet is a valid IP packet, and drops the packet if it has a non-IP header.

If you include the **anti-spoofing** keyword, the system also checks whether the mobile user IP address in the IP header of the inner payload matches the IP address assigned in GTP control messages such as Create Session Response, and drops the GTP-U message if the IP addresses do not match. This check supports IPv4, IPv6, and IPv4v6 PDN Types. If the mobile station gets its address using DHCP, the end-user IP address in GTPv2 is 0.0.0.0 (IPv4) or *prefix*::0 (IPv6), so in this case, the system updates the end-user IP address with the first IP address found in the inner packets. You can change the default behavior for DHCP-obtained addresses using the following keywords:

- **gtpv2-dhcp-bypass**—Do not update the 0.0.0.0 or *prefix*::0 address. Instead, allow packets where the end-user IP address is 0.0.0.0 or *prefix*::0. This option bypasses the anti-spoofing check when DHCP is used to obtain the IP address.
- **gtpv2-dhcp-drop**—Do not update the 0.0.0.0 or *prefix*::0 address. Instead, drop all packets where the end-user IP address is 0.0.0.0 or *prefix*::0. This option prevents access for users that use DHCP to obtain the IP address.

Step 6 While still in parameter configuration mode, configure the inspection to drop GTP-in-GTP encapsulated packets.

gtp-u-header-check gtp-in-gtp-encapsulation

This option can help prevent GTP tunneling attacks. The system examines the data payload for GTP ports (UDP/2123 or UDP/2152) as the source or destination port, and drops matching packets.

Without this option, GTP-in-GTP encapsulated packets are allowed.

Step 7 While still in parameter configuration mode, configure IMSI prefix filtering, if desired:

mcc *country_code* **mnc** *network_code*

drop mcc *country_code* **mnc** *network_code*

You can enter the command as many times as necessary to specify all targeted MCC/MNC pairs, but all commands within the policy map must be either **mcc** or **drop mcc**. You cannot combine these commands.

By default, GTP inspection does not check for valid Mobile Country Code (MCC)/Mobile Network Code (MNC) combinations. If you configure IMSI prefix filtering, the MCC and MNC in the IMSI of the received packet is compared with the configured MCC/MNC combinations. The system then takes one of the following actions based on the command:

- **mcc** command—The packet is dropped if it does not match.
- **drop mcc** command—The packet is dropped if it does match.

The Mobile Country Code is a non-zero, three-digit value; add zeros as a prefix for one- or two-digit values. The Mobile Network Code is a two- or three-digit value.

Add all MCC and MNC combinations you want to either permit or to drop. By default, the ASA does not check the validity of MNC and MCC combinations, so you must verify the validity of the combinations configured. To find more information about MCC and MNC codes, see the ITU E.212 recommendation, *Identification Plan for Land Mobile Stations*.

Step 8 While still in parameters configuration mode, enable location logging, if desired.

location-logging [**cell-id**]

Log the location of subscribers to track location changes for mobile stations. Tracking location changes can help you identify fraudulent roaming charges. When you enable location logging, the system generates syslog messages for new (message 324010) or changed (message 324011) location for each International Mobile Subscriber Identity (IMSI).

Specify the **cell-id** parameter if you want the log messages to include the the cell ID where the user currently is registered. The cell ID is extracted from the Cell Global Identification (CGI) or E-UTRAN Cell Global Identifier (ECGI).

Step 9 While still in parameter configuration mode, configure GSN or PGW pooling, if desired.

permit-response to-object-group *SGSN-SGW_name* **from-object-group** *GSN-PGW_pool*

When the ASA performs GTP inspection, by default the ASA drops GTP responses from GSNs or PGWs that were not specified in the GTP request. This situation occurs when you use load-balancing among a pool of GSNs or PGWs to provide efficiency and scalability of GPRS.

To configure GSN/PGW pooling and thus support load balancing, create a network object group that specifies the GSN/PGW endpoints and specify this on the **from-object-group** parameter. Likewise, create a network object group for the SGSN/SGW and select it on the **to-object-group** parameter. If the GSN/PGW responding belongs to the same object group as the GSN/PGW that the GTP request was sent to and if the SGSN/SGW is in an object group that the responding GSN/PGW is permitted to send a GTP response to, the ASA permits the response.

The network object group can identify the endpoints by host address or by the subnet that contains them.

Example:

The following is an example of GSN/PGW pooling. An entire Class C network is defined as the GSN/PGW pool but you can identify multiple individual IP addresses, one per **network-object** command, instead of identifying whole networks. The example then modifies a GTP inspection map to permit responses from the pool to the SGSN/SgW.

```
hostname(config)# object-group network gsnpool32
hostname(config-network)# network-object 192.168.100.0 255.255.255.0
hostname(config)# object-group network sgsn32
hostname(config-network)# network-object host 192.168.50.100

hostname(config)# policy-map type inspect gtp gtp-policy
hostname(config-pmap)# parameters
hostname(config-pmap-p)# permit-response to-object-group sgsn32
from-object-group gsnpool32
```

Example

The following example shows how to limit the number of tunnels in the network:

```
hostname(config)# policy-map type inspect gtp gmap
hostname(config-pmap)# parameters
hostname(config-pmap-p)# tunnel-limit 3000

hostname(config)# policy-map global_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect gtp gmap
```

```
hostname(config)# service-policy global_policy global
```

What to do next

You can now configure an inspection policy to use the map. See Configure the Mobile Network Inspection Service Policy , on page 35.

Configure an SCTP Inspection Policy Map

To apply alternative actions to SCTP traffic based on the application-specific payload protocol identifier (PPID), such as rate limiting, create an SCTP inspection policy map to be used by the service policy.

Note PPIDs are in data chunks, and a given packet can have multiple data chunks or even a control chunk. If a packet includes a control chunk or multiple data chunks, the packet will not be dropped even if the assigned action is drop. For example, if you configure an SCTP inspection policy map to drop PPID 26, and a PPID 26 data chunk is combined in a packet with a Diameter PPID data chunk, that packet will not be dropped.

Procedure

Step 1 Create an SCTP inspection policy map: **policy-map type inspect sctp** *policy_map_name*

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) Add a description to the policy map: **description** *string*

Step 3 Drop, rate limit, or log traffic based on the PPID in SCTP data chunks.

a) Identify the traffic based on the PPID.

match [**not**] **ppid** *ppid_1* [*ppid_2*]

Where *ppid_1* is the PPID number (0-4294967295) or name (see the CLI help for the available names). You can include a second (higher) PPID, *ppid_2*, to specify a range of PPIDs. Use **match not ppid** to identify traffic that does not match the PPID or range.

You can find the current list of SCTP PPIDs at http://www.iana.org/assignments/sctp-parameters/sctp-parameters.xhtml#sctp-parameters-25.

b) Specify the action to perform on matching packets.

- **drop**—Drop and log all packets that match.
- **log**—Send a system log message.
- **rate-limit** *rate*—Limit the rate of messages. The rate is in kilobits per second (kbps).

c) Repeat the process until you identify all PPIDs you want to selectively handle.

Example

The following example creates an inspection policy map that will drop unassigned PPIDs (unassigned at the time this example was written), rate limit PPIDs 32-40, and log the Diameter PPID. The service policy applies the inspection to the inspection_default class, which matches all SCTP traffic.

```
policy-map type inspect sctp sctp-pmap
 match ppid 58 4294967295
  drop
 match ppid 26
  drop
 match ppid 49
  drop
 match ppid 32 40
  rate-limit 1000
 match ppid diameter
  log

policy-map global_policy
 class inspection_default
  inspect sctp sctp-pmap
!
service-policy global_policy global
```

What to do next

You can now configure an inspection policy to use the map. See Configure the Mobile Network Inspection Service Policy , on page 35.

Configure a Diameter Inspection Policy Map

You can create a Diameter inspection policy map to filter on various Diameter protocol elements. You can then selectively drop or log connections.

To configure Diameter message filtering, you must have a good knowledge of these protocol elements as they are defined in RFCs and technical specifications. For example, the IETF has a list of registered applications, command codes, and attribute-value pairs at http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xhtml, although Diameter inspection does not support all listed items. See the 3GPP web site for their technical specifications.

Before you begin

Some traffic matching options use regular expressions for matching purposes. If you intend to use one of those techniques, first create the regular expression or regular expression class map.

Procedure

Step 1 (Optional) Create a Diameter inspection class map by performing the following steps.

A class map groups multiple traffic matches. You can alternatively identify **match** commands directly in the policy map. The difference between creating a class map and defining the traffic match directly in the inspection policy map is that the class map lets you create more complex match criteria, and you can reuse class maps.

To specify traffic that should not match the class map, use the **match not** command. For example, if the **match not** command specifies the string “example.com,” then any traffic that includes “example.com” does not match the class map.

For the traffic that you identify in this class map, you specify actions to take on the traffic in the inspection policy map.

If you want to perform different actions for each **match** command, you should identify the traffic directly in the policy map.

a) Create the class map: **class-map type inspect diameter** [**match-all** | **match-any**] *class_map_name*

   Where *class_map_name* is the name of the class map. The **match-all** keyword is the default, and specifies that traffic must match all criteria to match the class map. The match-any keyword specifies that the traffic matches the class map if it matches at least one **match** statement. The CLI enters class-map configuration mode, where you can enter one or more **match** commands.

b) (Optional) Add a description to the class map: **description** *string*

   Where *string* is the description of the class map (up to 200 characters).

c) Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

   - **match** [**not**] **application-id** *app_id* [*app_id_2*]—Matches the application identifier, where *app_id* is the Diameter application name or number (0-4294967295). If there is a range of consecutively-numbered applications that you want to match, you can include a second ID. You can define the range by application name or number, and it applies to all the numbers between the first and second IDs.

     These applications are registered with the IANA. Following are the core supported applications, but you can filter on other applications. Use the CLI help for a list of application names.

     - **3gpp-rx-ts29214** (16777236)
     - **3gpp-s6a** (16777251)
     - **3gpp-s9** (16777267)
     - **common-message** (0). This is the base Diameter protocol.

   - **match** [**not**] **command-code** *code* [*code_2*]—Matches the command code, where *code* is the Diameter command code name or number (0-4294967295). If there is a range of consecutively-numbered command codes that you want to match, you can include a second code. You can define the range by command code name or number, and it applies to all the numbers between the first and second codes.

     For example, the following command matches the Capability Exchange Request/Answer command code:

     ```
     match command-code cer-cea
     ```

- Match the attribute-value pair (AVP).

  To match AVP by attribute only:

  match [**not**] **avp** *code* [*code_2*] [**vendor-id** *id_number*]

  To match an AVP based on the value of the attribute:

  match [**not**] **avp** *code* [**vendor-id** *id_number*] *value*

  Where:

  - *code*—The name or number (1-4294967295) of an attribute-value pair. For the first code, you can specify the name of a custom AVP or one that is registered in RFCs or 3GPP technical specifications and is directly supported in the software. If you want to match a range of AVP, specify the second code by number only. If you want to match an AVP by its value, you cannot specify a second code. See the CLI help for a list of AVP names.
  - **vendor-id** *id_number*—(Optional.) The ID number of the vendor to also match, from 0-4294967295. For example, the 3GPP vendor ID is 10415, the IETF is 0.
  - *value*—The value portion of the AVP. You can configure this only if the data type of the AVP is supported. For example, you can specify an IP address for AVP that have the address data type. Following are the specific syntax of the value option for the supported data types:
    - Diameter Identity, Diameter URI, Octet String—Use regular expression or regular expression class objects to match these data types.

      {**regex** *regex_name* | **class** *regex_class*}

    - Address—Specify the IPv4 or IPv6 address to match. For example, 10.100.10.10 or 2001:DB8::0DB8:800:200C:417A.
    - Time—Specify the start and end dates and time. Both are required. Time is in 24-hour format.

      date *year month day* **time** *hh:mm:ss* **date** *year month day* **time** *hh:mm:ss*

      For example:

      ```
      date 2015 feb 5 time 12:00:00 date 2015 mar 9 time 12:00:00
      ```

    - Numeric—Specify a range of numbers:

      range *number_1 number_2*

      The valid number range depends on the data type:

      - Integer32: -2147483647 to 2147483647
      - Integer64: -9223372036854775807 to 9223372036854775807
      - Unsigned32: 0 to 4294967295
      - Unsigned64: 0 to 18446744073709551615
      - Float32: decimal point representation with 8 digit precision
      - Float64: decimal point representation with 16 digit precision

d) Enter **exit** to leave class map configuration mode.

Step 2 Create a Diameter inspection policy map: **policy-map type inspect diameter** *policy_map_name*

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 3 (Optional) Add a description to the policy map: **description** *string*

Step 4 To apply actions to matching traffic, perform the following steps.

a) Specify the traffic on which you want to perform actions using one of the following methods:

- If you created a Diameter class map, specify it by entering the following command: **class** *class_map_name*
- Specify traffic directly in the policy map using one of the **match** commands described for Diameter class maps.

b) Specify the action you want to perform on the matching traffic by entering one of the following commands:

- **drop**—Drop all packets that match.
- **drop-connection**—Drop the packet and close the connection.
- **log**—Send a system log message.

You can specify multiple **class** or **match** commands in the policy map. For information about the order of **class** and **match** commands, see How Multiple Traffic Classes are Handled.

Example:

```
hostname(config)# policy-map type inspect diameter diameter-map
hostname(config-pmap)# class diameter-class-map
hostname(config-pmap-c)# drop
hostname(config-pmap-c)# match command-code cer-cea
hostname(config-pmap-c)# log
```

Step 5 To configure parameters that affect the inspection engine, perform the following steps:

a) Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b) Set one or more parameters. You can set the following options; use the no form of the command to disable the option.

- **unsupported** {**application-id** | **command-code** | **avp**} **action log**—Enables logging for unsupported Diameter elements. These options specify application IDs, command codes, and AVP that are not directly supported by the software. The default is to allow the elements without logging them. You can enter the command three times to enable logging for all elements.
- **strict-diameter** {**state** | **session**}—Enables strict Diameter protocol conformance to RFC 6733. By default, inspection ensures that Diameter frames comply with the RFC. You can add **state** machine validation or **session**-related message validation, or both by entering the command twice.

Example:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)# unsupported application-id action log
hostname(config-pmap-p)# unsupported command-code action log
hostname(config-pmap-p)# unsupported avp action log
hostname(config-pmap-p)# strict-diameter state
hostname(config-pmap-p)# strict-diameter session
```

Example

The following example shows a how to log some applications and block a specific IP address.

```
class-map type inspect diameter match-any log_app
  match application-id 3gpp-s6a
  match application-id 3gpp-s13

class-map type inspect diameter match-all block_ip
  match command-code cer-cea
  match avp host-ip-address 1.1.1.1

policy-map type inspect diameter diameter_map
  parameters
    unsupported application-id log
  class log_app
    log
  class block_ip
    drop-connection

policy-map global_policy
  class inspection_default
    inspect diameter diameter_map

service-policy global_policy global
```

What to do next

You can now configure an inspection policy to use the map. See Configure the Mobile Network Inspection Service Policy , on page 35.

Create a Custom Diameter Attribute-Value Pair (AVP)

As new attribute-value pairs (AVP) are defined and registered, you can create custom Diameter AVP to define them and use them in your Diameter inspection policy map. You would get the information you need to create the AVP from the RFC or other source that defines the AVP.

Create custom AVP only if you want to use them in a Diameter inspection policy map or class map for AVP matching.

Procedure

Create a custom Diameter AVP.

diameter avp *name* **code** *value* **data-type** *type* [**vendor-id** *id_number*] [**description** *text*]

Where:

- *name*—The name of the custom AVP you are creating, up to 32 characters. You would refer to this name on the match avp command in a Diameter inspection policy map or class map.
- **code** *value*—The custom AVP code value, from 256-4294967295. You cannot enter a code and vendor-id combination that is already defined in the system.
- **data-type** *type*—The data type of the AVP. You can define AVP of the following types. If the new AVP is of a different type, you cannot create a custom AVP for it.
  - **address**—For IP addresses.
  - **diameter-identity**—Diameter identity data.
  - **diameter-uri**—Diameter uniform resource identifier (URI).
  - **float32**—32-bit floating point number.
  - **float64**—64-bit floating point number.
  - **int32**—32-bit integer.
  - **int64**—64-bit integer.
  - **octetstring**—Octet string.
  - **time**—Time value.
  - **uint32**—32-bit unsigned integer.
  - **uint64**—64-bit unsigned integer.
- **vendor-id** *id_number*—(Optional.) The ID number of the vendor who defined the AVP, from 0-4294967295. For example, the 3GPP vendor ID is 10415, the IETF is 0.
- **description** *text*—(Optional.) A description of the AVP, up to 80 characters. Enclose the description in quotation marks if you include spaces.

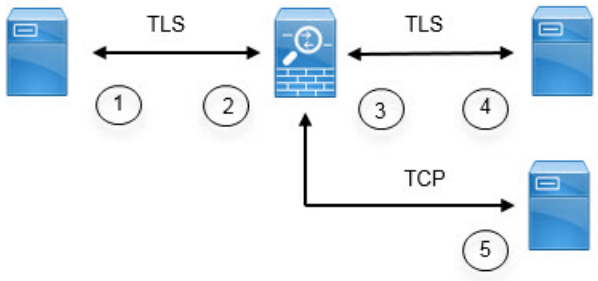
Inspecting Encrypted Diameter Sessions

If a Diameter application uses encrypted data over TCP, inspection cannot see inside the packets to implement your message filtering rules. Thus, if you create filtering rules, and you want them to also apply to encrypted TCP traffic, you must configure a TLS proxy. You also need a proxy if you want strict protocol enforcement on encrypted traffic. This configuration does not apply to SCTP/DTLS traffic.

The TLS proxy acts as a man-in-the-middle. It decrypts traffic, inspects it, then encrypts it again and sends it to the intended destination. Thus, both sides of the connection, the Diameter server and Diameter client, must trust the ASA, and all parties must have the required certificates. You must have a good understanding of digital certificates to implement TLS proxy. Please read the chapter on digital certificates in the ASA general configuration guide.

The following illustration shows the relationship among the Diameter client and server, and the ASA, and the certification requirements to establish trust. In this model, a Diameter client is an MME (Mobility Management Entity), not an end user. The CA certificate on each side of a link is the one used to sign the certificate on the other side of the link. For example, the ASA proxy TLS server CA certificate is the one used to sign the Diameter/TLS client certificate.

Figure 2: Diameter TLS Inspection



1	Diameter TLS client (MME) • Client identity certificate • CA certificate used to sign the ASA TLS proxy server's identity certificate	2	ASA proxy TLS server • Server identity certificate • CA certificate used to sign the Diameter TLS client's identity certificate
3	ASA proxy TLS client • Client identity (static or LDC) certificate • CA certificate used to sign the Diameter TLS server identity certificate	4	Diameter TLS server (full proxy) • Server identity certificate • CA certificate used to sign the ASA proxy TLS client's identity certificate
5	Diameter TCP server (TLS offload).	—	—

You have the following options for configuring TLS proxy for Diameter inspection:

- Full TLS proxy—Encrypt traffic between the ASA and Diameter clients and the ASA and Diameter server. You have the following options for establishing the trust relationship with the TLS server:
  - Use a static proxy client trustpoint. The ASA presents the same certificate for every Diameter client when communicating with the Diameter server. Because all clients look the same, the Diameter server cannot provide differential services per client. On the other hand, this option is faster than the LDC method.
  - Use local dynamic certificates (LDC). With this option, the ASA presents unique certificates per Diameter client when communicating with the Diameter server. The LDC retains all fields from the

received client identity certificate except its public key and a new signature from the ASA. This method gives the Diameter server better visibility into client traffic, which makes it possible to provide differential services based on client certificate characteristics.

- TLS offload—Encrypt traffic between the ASA and Diameter client, but use a clear-text connection between the ASA and Diameter server. This option is viable if the Diameter server is in the same data center as the ASA, where you are certain that the traffic between the devices will not leave the protected area. Using TLS offload can improve performance, because it reduces the amount of encryption processing required. It should be the fastest of the options. The Diameter server can apply differential services based on client IP address only.

All three options use the same configuration for the trust relationship between the ASA and Diameter clients.

Note TLS proxy uses TLSv1.0 - 1.2. You can configure the TLS version and the cipher suite.

The following topics explain how to configure TLS proxy for Diameter inspection.

Configure Server Trust Relationship with Diameter Clients

The ASA acts as a TLS proxy server in relation to the Diameter clients. To establish the mutual trust relationship:

- You need to import the Certificate Authority (CA) certificate used to sign the ASA's server certificate into the Diameter client. This might be in the client's CA certificate store or some other location that the client uses. Consult the client documentation for exact details on certificate usage.
- You need to import the CA certificate used to sign the Diameter TLS client's certificate so the ASA can trust the client.

The following procedure explains how to import the CA certificate used to sign the Diameter client's certificate, and import an identity certificate to use for the ASA TLS proxy server. Instead of importing an identity certificate, you could create a self-signed certificate on the ASA.

Procedure

Step 1 Import the CA certificate that is used to sign the Diameter client's certificate into an ASA trustpoint.

This step allows the ASA to trust the Diameter clients.

a) Create the trustpoint for the Diameter client.

In this example, **enrollment terminal** indicates you will paste the certificate into the CLI. The trustpoint is called **diameter-clients**.

```
ciscoasa(config)# crypto ca trustpoint diameter-clients
ciscoasa(ca-trustpoint)# revocation-check none
ciscoasa(ca-trustpoint)# enrollment terminal
```

b) Add the certificate.

```
ciscoasa(config)# crypto ca authenticate diameter-clients
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
MIIDRTCCAu+gAwIBAgIQKVcqP/KW74VP0NZzL+JbRTANBgkqhkiG9w0BAQUFADCB
[certificate data omitted]
/7QEM8izy0EOTSErKu7Nd76jwf5e4qttkQ==
quit

INFO: Certificate has the following attributes:
Fingerprint: 24b81433 409b3fd5 e5431699 8d490d34
Do you accept this certificate? [yes/no]: y
Trustpoint CA certificate accepted.

% Certificate successfully imported
```

Step 2 Import the certificate and create a trustpoint for the ASA proxy server's identity certificate and keypair.

This step allows the Diameter clients to trust the ASA.

a) Import the certificate in pkcs12 format.

In the following example, **tls-proxy-server-tp** is the trustpoint name and **"123"** is the decryption pass phrase. Use your own trustpoint name and pass phrase.

```
ciscoasa (config)# crypto ca import tls-proxy-server-tp pkcs12 "123"

Enter the base 64 encoded pkcs12.
End with a blank line or the word "quit" on a line by itself:
[PKCS12 data omitted]

quit

INFO: Import PKCS12 operation completed successfully

ciscoasa (config)#
```

b) Configure the trustpoint.

```
ciscoasa(config)# crypto ca trustpoint tls-proxy-server-tp
ciscoasa(ca-trustpoint)# revocation-check none
```

Configure Full TLS Proxy with Static Client Certificate for Diameter Inspection

If the Diameter server can accept the same certificate for all clients, you can set up a static client certificate for the ASA to use when communicating with the Diameter server.

With this configuration, you need to establish the mutual trust relationship between the ASA and clients (as explained in Configure Server Trust Relationship with Diameter Clients, on page 22), and the ASA and Diameter server. Following are the ASA and Diameter server trust requirements.

- You need to import the CA certificate used to sign the Diameter Server's identity certificate so the ASA can validate the server's identity certificate during the TLS handshake.

- You need to import the client certificate, one that the Diameter server also trusts. If the Diameter server does not already trust the certificate, import the CA certificate used to sign it into the server. Consult the Diameter server's documentation for details.

Procedure

Step 1 Import the CA certificate that is used to sign the Diameter server's certificate into an ASA trustpoint.

This step allows the ASA to trust the Diameter server.

a) Create the trustpoint for the Diameter server.

In this example, **enrollment terminal** indicates you will paste the certificate into the CLI. You could also use enrollment url to specify automatic enrollment (SCEP) with the CA. The trustpoint is called **diameter-server**.

```
ciscoasa(config)# crypto ca trustpoint diameter-server
ciscoasa(ca-trustpoint)# revocation-check none
ciscoasa(ca-trustpoint)# enrollment terminal
```

b) Add the certificate.

```
ciscoasa(config)# crypto ca authenticate diameter-server
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
MIIDRTCCAu+gAwIBAgIQKVcqP/KW74VP0NZzL+JbRTANBgkqhkiG9w0BAQUFADCB
[certificate data omitted]
/7QEM8izy0EOTSErKu7Nd76jwf5e4qttkQ==
quit

INFO: Certificate has the following attributes:
Fingerprint: 24b81433 409b3fd5 e5431699 8d490d34
Do you accept this certificate? [yes/no]: y
Trustpoint CA certificate accepted.

% Certificate successfully imported
```

Step 2 Import the certificate and create a trustpoint for the ASA proxy client's identity certificate and keypair.

This step allows the Diameter server to trust the ASA.

a) Import the certificate in pkcs12 format.

In the following example, **tls-proxy-client-tp** is the trustpoint name and **"123"** is the decryption pass phrase. Use your own trustpoint name and pass phrase.

```
ciscoasa (config)# crypto ca import tls-proxy-client-tp pkcs12 "123"

Enter the base 64 encoded pkcs12.
End with a blank line or the word "quit" on a line by itself:
[PKCS12 data omitted]

quit
```

```
INFO: Import PKCS12 operation completed successfully

ciscoasa (config)#
```

b) Configure the trustpoint.

```
ciscoasa(config)# crypto ca trustpoint tls-proxy-client-tp
ciscoasa(ca-trustpoint)# revocation-check none
```

Step 3 Configure the TLS proxy.

a) Name the TLS proxy and enter TLS proxy configuration mode.

tls-proxy *name*

b) Identify the trustpoint used when the ASA acts as the proxy server in relationship to the Diameter clients.

server trust-point *trustpoint_name*

Note
For testing purposes, or if you are certain that you can trust the Diameter clients, you can skip this step and include the **no server authenticate-client** command in the TLS proxy configuration.

c) Identify the trustpoint used when the ASA acts as the proxy client in relationship to the Diameter server.

client trust-point *name*

d) (Optional.) Define the ciphers that the client can use.

client cipher-suite *cipher-list*

Where *cipher-list* can include any combination of the following:

- **3des-sha1**
- **aes128-sha1**
- **aes256-sha1**
- **des-sha1**
- **null-sha1**
- **rc4-sha1**

Separate multiple options with spaces.

If you do not define the ciphers the TLS proxy can use, the proxy server uses the global cipher suite defined by the **ssl cipher** command. By default, the global cipher level is medium, which means all ciphers are available except for NULL-SHA, DES-CBC-SHA, and RC4-MD5. Specify the **client cipher-suite** command only if you want to use a different suite than the one generally available on the ASA.

To set the minimum TLS version for all SSL client connections on the ASA, see the **ssl client-version** command. The default is TLS v1.0.

e) (Optional.) Define the ciphers that the server can use.

server cipher-suite *cipher-list*

Where *cipher-list* can include any combination of the following:

- **3des-sha1**
- **aes128-sha1**
- **aes256-sha1**
- **des-sha1**
- **null-sha1**
- **rc4-sha1**

Separate multiple options with spaces.

If you do not define the ciphers the TLS proxy can use, the proxy server uses the global cipher suite defined by the **ssl cipher** command. By default, the global cipher level is medium, which means all ciphers are available except for NULL-SHA, DES-CBC-SHA, and RC4-MD5. Specify the **server cipher-suite** command only if you want to use a different suite than the one generally available on the ASA.

To set the minimum TLS version for all SSL server connections on the ASA, see the **ssl server-version** command. The default is TLS v1.0.

Example:

```
ciscoasa(config)# tls-proxy diameter-tls-static-proxy
ciscoasa(config-tlsp)# server trust-point tls-proxy-server-tp
ciscoasa(config-tlsp)# client trust-point tls-proxy-client-tp
```

What to do next

You can now use the TLS proxy in Diameter inspection. See Configure the Mobile Network Inspection Service Policy , on page 35.

Configure Full TLS Proxy with Local Dynamic Certificates for Diameter Inspection

If the Diameter server needs unique certificates for each client, you can configure the ASA to generate local dynamic certificates (LDC). These certificates exist for the duration of the client's connection and are then destroyed.

With this configuration, you need to establish the mutual trust relationship between the ASA and clients (as explained in Configure Server Trust Relationship with Diameter Clients, on page 22), and the ASA and Diameter server. The configuration is similar to the one described in Configure Full TLS Proxy with Static Client Certificate for Diameter Inspection, on page 23, except instead of importing a Diameter client certificate, you set up the LDC on the ASA. Following are the ASA and Diameter server trust requirements.

- You need to import the CA certificate used to sign the Diameter Server's identity certificate so the ASA can validate the server's identity certificate during the TLS handshake.
- You need to create the LDC trustpoint. You need to export the LDC server's CA certificate and import it into the Diameter server. The export step is explained below. Consult the Diameter server's documentation for information on importing certificates.

Procedure

Step 1 Import the CA certificate that is used to sign the Diameter server's certificate into an ASA trustpoint.

This step allows the ASA to trust the Diameter server.

a) Create the trustpoint for the Diameter server.

In this example, **enrollment terminal** indicates you will paste the certificate into the CLI. You could also use enrollment url to specify automatic enrollment (SCEP) with the CA. The trustpoint is called **diameter-server**.

```
ciscoasa(config)# crypto ca trustpoint diameter-server
ciscoasa(ca-trustpoint)# revocation-check none
ciscoasa(ca-trustpoint)# enrollment terminal
```

b) Add the certificate.

```
ciscoasa(config)# crypto ca authenticate diameter-server
Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself
MIIDRTCCAu+gAwIBAgIQKVcqP/KW74VP0NZzL+JbRTANBgkqhkiG9w0BAQUFADCB
[certificate data omitted]
/7QEM8izy0EOTSErKu7Nd76jwf5e4qttkQ==
quit

INFO: Certificate has the following attributes:
Fingerprint: 24b81433 409b3fd5 e5431699 8d490d34
Do you accept this certificate? [yes/no]: y
Trustpoint CA certificate accepted.

% Certificate successfully imported
```

Step 2 Create the local CA to sign local dynamic certificates (LDC).

a) Create an RSA keypair for the trustpoint.

In this example, the keypair name is ldc-signer-key.

```
ciscoasa(config)# crypto key generate rsa label ldc-signer-key
INFO: The name for the keys will be: ldc-signer-key
Keypair generation process
ciscoasa(config)#
```

b) Create the LDC issuer trustpoint.

In this example, the trustpoint name is **ldc-server**, the keypair created above is used, self-signed enrollment is specified (**enrollment self**, which is required), and the common name of the ASA is included as the subject name. Check whether the Diameter application has specific requirements for the subject name.

The **proxy-ldc-issuer** command defines the local CA role for the trustpoint to issue dynamic certificates for TLS proxy.

```
ciscoasa(config)# crypto ca trustpoint ldc-server
ciscoasa(ca-trustpoint)# keypair ldc-signer-key
ciscoasa(ca-trustpoint)# subject-name CN=asa3
ciscoasa(ca-trustpoint)# enrollment self
ciscoasa(ca-trustpoint)# proxy-ldc-issuer
ciscoasa(ca-trustpoint)# exit
```

c) Enroll the trustpoint.

```
ciscoasa(config)# crypto ca enroll ldc-server
```

Step 3 Configure the TLS proxy.

a) Name the TLS proxy and enter TLS proxy configuration mode.

tls-proxy *name*

b) Identify the trustpoint used when the ASA acts as the server in relationship to the Diameter clients.

server trust-point *trustpoint_name*

Note
For testing purposes, or if you are certain that you can trust the Diameter clients, you can skip this step and include the **no server authenticate-client** command in the TLS proxy configuration.

c) Identify the LDC trustpoint used when the ASA issues dynamic certificates and acts as the client in relationship to the Diameter server.

client ldc issuer *name*

d) Identify the LDC keypair. Specify the same key that is defined in the LDC trustpoint.

client ldc key-pair *name*

e) (Optional.) Define the ciphers that the client can use.

client cipher-suite *cipher-list*

Where *cipher-list* can include any combination of the following:

- **3des-sha1**
- **aes128-sha1**
- **aes256-sha1**
- **des-sha1**
- **null-sha1**
- **rc4-sha1**

Separate multiple options with spaces.

If you do not define the ciphers the TLS proxy can use, the proxy server uses the global cipher suite defined by the **ssl cipher** command. By default, the global cipher level is medium, which means all ciphers are available except for NULL-SHA, DES-CBC-SHA, and RC4-MD5. Specify the **client cipher-suite** command only if you want to use a different suite than the one generally available on the ASA.

To set the minimum TLS version for all SSL client connections on the ASA, see the **ssl client-version** command. The default is TLS v1.0.

f) (Optional.) Define the ciphers that the server can use.

server cipher-suite *cipher-list*

Where *cipher-list* can include any combination of the following:

- **3des-sha1**
- **aes128-sha1**
- **aes256-sha1**
- **des-sha1**
- **null-sha1**
- **rc4-sha1**

Separate multiple options with spaces.

If you do not define the ciphers the TLS proxy can use, the proxy server uses the global cipher suite defined by the **ssl cipher** command. By default, the global cipher level is medium, which means all ciphers are available except for NULL-SHA, DES-CBC-SHA, and RC4-MD5. Specify the **server cipher-suite** command only if you want to use a different suite than the one generally available on the ASA.

To set the minimum TLS version for all SSL server connections on the ASA, see the **ssl server-version** command. The default is TLS v1.0.

Example:

```
ciscoasa(config)# tls-proxy diameter-tls-ldc-proxy
ciscoasa(config-tlsp)# server trust-point tls-proxy-server-tp
ciscoasa(config-tlsp)# client ldc issuer ldc-server
ciscoasa(config-tlsp)# client ldc key-pair ldc-signer-key
```

Step 4 Export the LDC CA certificate and import it into the Diameter server.

a) Export the certificate.

In the following example, the LDC trustpoint is ldc-server; specify your own LDC trustpoint name.

```
ciscoasa(config)# crypto ca export ldc-server identity-certificate
-----BEGIN CERTIFICATE-----
MIIDbDCCAlSgAwIBAgIQfWOQvGFpj7hCCB49+kS4CjANBgkqhkiG9w0BAQUFADAT
MREwDwYDVQQDEwhIdW5ueUJlZTAeFw0xMzA2MjUwMTE5MzJaFw00ODA2MjUwMTI5
...[data omitted]...
lJZ48NoI64RqfGC/KHUsOQ==
-----END CERTIFICATE-----
```

b) Copy the certificate data and save it into a file.

You can now import it into the Diameter server. Consult the Diameter server's documentation for the procedure. Note that the data is in Base64 format. If your server requires binary or DER format, you will need to use OpenSSL tools to convert formats.

What to do next

You can now use the TLS proxy in Diameter inspection. See Configure the Mobile Network Inspection Service Policy , on page 35.

Configure TLS Proxy with TLS Offload for Diameter Inspection

If you are certain the network path between the ASA and Diameter server is secure, you can avoid the performance cost of encrypting data between the ASA and server. With TLS offload, the TLS proxy encrypts/decrypts sessions between the Diameter client and the ASA, but uses clear text with the Diameter server.

With this configuration, you need to establish the mutual trust relationship between the ASA and clients only, which simplifies the configuration. Before doing the following procedure, complete the steps in Configure Server Trust Relationship with Diameter Clients, on page 22.

Procedure

Step 1 Configure the TLS proxy with TLS offload.

a) Name the TLS proxy and enter TLS proxy configuration mode.

tls-proxy *name*

b) Identify the trustpoint used when the ASA acts as the server in relationship to the Diameter clients.

server trust-point *trustpoint_name*

Note
For testing purposes, or if you are certain that you can trust the Diameter clients, you can skip this step and include the **no server authenticate-client** command in the TLS proxy configuration.

c) (Optional.) Define the ciphers that the server can use.

server cipher-suite *cipher-list*

Where *cipher-list* can include any combination of the following:

- **3des-sha1**
- **aes128-sha1**
- **aes256-sha1**
- **des-sha1**
- **null-sha1**
- **rc4-sha1**

Separate multiple options with spaces.

If you do not define the ciphers the TLS proxy can use, the proxy server uses the global cipher suite defined by the **ssl cipher** command. By default, the global cipher level is medium, which means all ciphers are available except for NULL-SHA, DES-CBC-SHA, and RC4-MD5. Specify the **server cipher-suite** command only if you want to use a different suite than the one generally available on the ASA.

To set the minimum TLS version for all SSL server connections on the ASA, see the **ssl server-version** command. The default is TLS v1.0.

d) Specify that communication between the ASA and the Diameter server should be in clear text. In this relationship, the ASA acts as a client of the Diameter server.

client clear-text

Example:

```
ciscoasa(config)# tls-proxy diameter-tls-offload-proxy
ciscoasa(config-tlsp)# server trust-point tls-proxy-server-tp
ciscoasa(config-tlsp)# client clear-text
```

Step 2 Because the Diameter ports differ for TCP and TLS, configure a NAT rule to translate the TCP port to the TLS port for traffic going from the Diameter server to the client.

Create an object NAT rule for each Diameter server. Each rule should:

- Perform static identity NAT for the Diameter server address. That is, the IP address in the object should be the same as the translated address in the NAT rule.
- Translate the real port 3868, which is the default Diameter TCP port number, to 5868, the default Diameter TLS port number.
- The source interface should be the one that connects to the Diameter server, and the destination interface the one that connects to the Diameter client.

The following example translates TCP traffic on port 3868 coming to the outside interface from the 10.29.29.29 Diameter server to port 5868 on the inside interface.

```
ciscoasa(config)# object network diameter-client
ciscoasa(config-network-object)# host 10.29.29.29
ciscoasa(config-network-object)# nat (outside,inside) static 10.29.29.29
service tcp 3868 5868
```

What to do next

You can now use the TLS proxy in Diameter inspection. See Configure the Mobile Network Inspection Service Policy , on page 35.

Configure an M3UA Inspection Policy Map

Use an M3UA inspection policy map to configure access control based on point codes. You can also drop and rate limit messages by class and type.

The default point code format is ITU. If you use a different format, specify the required format in the policy map.

If you do not want to apply policy based on point code or message class, you do not need to configure an M3UA policy map. You can enable inspection without a map.

Procedure

Step 1 Create an M3UA inspection policy map: **policy-map type inspect m3ua** *policy_map_name*

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) Add a description to the policy map: **description** *string*

Step 3 To apply actions to matching traffic, perform the following steps.

a) Specify the traffic on which you want to perform actions using one of the following **match** commands. If you use a **match not** command, then any traffic that does not match the criterion in the **match not** command has the action applied.

- **match** [**not**] **message class** *class_id* [**id** *message_id*]—Matches the M3UA message class and type. The following table lists the possible values. Consult M3UA RFCs and documentation for detailed information about these messages.

M3UA Message Class	Message ID Type
0 (Management Messages)	0-1
1 (Transfer Messages)	1
2 (SS7 Signaling Network Management Messages)	1-6
3 (ASP State Maintenance Messages)	1-6
4 (ASP Traffic Maintenance Messages)	1-4
9 (Routing Key Management Messages)	1-4

- **match** [**not**] **opc** *code*—Matches the originating point code in the data message, that is, the traffic source. Point code is in *zone-region-sp* format, where the possible values for each element depend on the SS7 variant:
  - **ITU**—Point codes are 14 bit in 3-8-3 format. The value ranges are [0-7]-[0-255]-[0-7].
  - **ANSI**—Point codes are 24 bit in 8-8-8 format. The value ranges are [0-255]-[0-255]-[0-255].
  - **Japan**—Point codes are 16 bit in 5-4-7 format. The value ranges are [0-31]-[0-15]-[0-127].
  - **China**—Point codes are 24 bit in 8-8-8 format. The value ranges are [0-255]-[0-255]-[0-255].
- **match** [**not**] **dpc** *code*—Matches the destination point code in the data message. Point code is in *zone-region-sp* format, as explained for **match opc**.

- **match** [**not**] **service-indicator** *number*—Matches the service indicator number, 0-15. Following are the available service indicators. Consult M3UA RFCs and documentation for detailed information about these service indicators.
  - 0—Signaling Network Management Messages
  - 1—Signaling Network Testing and Maintenance Messages
  - 2—Signaling Network Testing and Maintenance Special Messages
  - 3—SCCP
  - 4—Telephone User Part
  - 5—ISDN User Part
  - 6—Data User Part (call and circuit-related messages)
  - 7—Data User Part (facility registration and cancellation messages)
  - 8—Reserved for MTP Testing User Part
  - 9—Broadband ISDN User Part
  - 10—Satellite ISDN User Part
  - 11—Reserved
  - 12—AAL type 2 Signaling
  - 13—Bearer Independent Call Control
  - 14—Gateway Control Protocol
  - 15—Reserved

b) Specify the action you want to perform on the matching traffic by entering one of the following commands:

- **drop** [**log**]—Drop all packets that match. Optionally, send a system log message.
- **rate-limit** *message_rate*—Limit the rate of messages. This option is available with **match message class** only.

You can specify multiple **match** commands in the policy map. For information about the order of match commands, see How Multiple Traffic Classes are Handled.

Step 4 To configure parameters that affect the inspection engine, perform the following steps:

a) Enter parameters configuration mode:

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

b) Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option:

- **message-tag-validation** {**dupu** | **error** | **notify**}—Ensures that the content of certain fields are checked and validated for the specified message type. Messages that fail validation are dropped. Validation differs by message type.

  - Destination User Part Unavailable (DUPU)—The User/Cause field must be present, and it must contain only valid cause and user codes.
  - Error—All mandatory fields must be present and contain only allowed values. Each error message must contain the required fields for that error code.
  - Notify—The status type and status information fields must contain allowed values only.

- **ss7 variant** {**ITU** | **ANSI** | **JAPAN** | **CHINA**}—Identifies the variant of SS7 used in your network. This option determines the valid format for point codes. After you configure the option and deploy an M3UA policy, you cannot change it unless you first remove the policy. The default variant is ITU.
- **strict-asp-state**—Performs application server process (ASP) state validation. The system maintains the ASP states of M3UA sessions and allows or drops ASP messages based on the validation result. If you do not enable strict ASP state validation, all ASP messages are forwarded uninspected. Strict ASP state checking is required if you want stateful failover or if you want to operate within a cluster. However, strict ASP state checking works in Override mode only, it does not work if you are running in Loadsharing or Broadcast mode (per RFC 4666). The inspection assumes there is one and only one ASP per endpoint.
- **timeout endpoint** *time*—Sets the idle timeout to remove statistics for an M3UA endpoint, in hh:mm:ss format. To have no timeout, specify 0. The default is 30 minutes (00:30:00).
- **timeout session** *time*—Sets the idle timeout to remove an M3UA session if you enable strict ASP state validation, in hh:mm:ss format. To have no timeout, specify 0. The default is 30 minutes (00:30:00). Disabling this timeout can prevent the system from removing stale sessions.

Example

The following is an example of an M3UA policy map and service policy.

```
hostname(config)# policy-map type inspect m3ua m3ua-map
hostname(config-pmap)# match message class 2 id 6
hostname(config-pmap-c)# drop
hostname(config-pmap-c)# match message class 9
hostname(config-pmap-c)# drop
hostname(config-pmap-c)# match dpc 1-5-1
hostname(config-pmap-c)# drop log
hostname(config-pmap-c)# parameters
hostname(config-pmap-p)# ss7 variant ITU
hostname(config-pmap-p)# timeout endpoint 00:45:00

hostname(config)# policy-map global_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect m3ua m3ua-map

hostname(config)# service-policy global_policy global
```

What to do next

You can now configure an inspection policy to use the map. See Configure the Mobile Network Inspection Service Policy , on page 35.

Configure the Mobile Network Inspection Service Policy

Inspections for the protocols used in mobile networks are not enabled in the default inspection policy, so you must enable them if you need these inspections. You can simply edit the default global inspection policy to add these inspections. You can alternatively create a new service policy as desired, for example, an interface-specific policy.

Procedure

Step 1 If necessary, create an L3/L4 class map to identify the traffic for which you want to apply the inspection.

```
class-map name
   match parameter
```

Example:

```
hostname(config)# class-map mobile_class_map
hostname(config-cmap)# match access-list mobile
```

In the default global policy, the inspection_default class map is a special class map that includes default ports for all inspection types (**match default-inspection-traffic**). If you are using this class map in either the default policy or for a new service policy, you can skip this step.

For information on matching statements, see Create a Layer 3/4 Class Map for Through Traffic.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic: **policy-map** *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 class map you are using for the inspection: **class** *name*

Example:

```
hostname(config-pmap)# class inspection_default
```

To edit the default policy, or to use the special inspection_default class map in a new policy, specify **inspection_default** for the *name*. Otherwise, you are specifying the class you created earlier in this procedure.

Step 4 Enable the inspections.

In the following commands, the inspection policy maps are optional. If you created any of these maps to customize the inspection, specify their names on the appropriate command. For Diameter, you can also specify a TLS proxy to enable inspection of encrypted messages.

- **inspect gtp** [*map_name*]—To enable GTP inspection.
- **inspect sctp** [*map_name*]—To enable SCTP inspection.

- **inspect diameter** [*map_name*] [**tls-proxy** *proxy_name*]—To enable Diameter inspection.

  Note
  If you specify a TLS proxy for Diameter inspection, and you apply NAT port redirection to Diameter server traffic (for example, redirect server traffic from port 5868 to 3868), configure inspection globally or on the ingress interface only. If you apply the inspection to the egress interface, NATed Diameter traffic bypasses inspection.

- **inspect m3ua** [*map_name*]—To enable M3UA inspection.

Example:

```
hostname(config-class)# inspect gtp
hostname(config-class)# inspect sctp
hostname(config-class)# inspect diameter
hostname(config-class)# inspect m3ua
```

Note
If you are editing the default global policy (or any in-use policy) to use a different inspection policy map, you must remove the inspection with the **no inspect** version of the command, and then re-add it with the new inspection policy map name. For example, to change the policy map for GTP:

```
hostname(config-class)# no inspect gtp
hostname(config-class)# inspect gtp gtp-map
```

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Configure RADIUS Accounting Inspection

RADIUS accounting inspection is not enabled by default. You must configure it if you want RADIUS accounting inspection.

Procedure

Step 1 Configure a RADIUS Accounting Inspection Policy Map, on page 37.

Step 2 Configure the RADIUS Accounting Inspection Service Policy, on page 38.

Configure a RADIUS Accounting Inspection Policy Map

You must create a RADIUS accounting inspection policy map to configure the attributes needed for the inspection.

Procedure

Step 1 Create a RADIUS accounting inspection policy map: **policy-map type inspect radius-accounting** *policy_map_name*

Where the *policy_map_name* is the name of the policy map. The CLI enters policy-map configuration mode.

Step 2 (Optional) Add a description to the policy map: **description** *string*

Step 3 Enter parameters configuration mode.

```
hostname(config-pmap)# parameters
hostname(config-pmap-p)#
```

Step 4 Set one or more parameters. You can set the following options; use the **no** form of the command to disable the option.

- **send response**—Instructs the ASA to send Accounting-Request Start and Stop messages to the sender of those messages (which are identified in the **host** command).
- **enable gprs**—Implement GPRS over-billing protection. The ASA checks for the 3GPP VSA 26-10415 attribute in the Accounting-Request Stop and Disconnect messages in order to properly handle secondary PDP contexts. If this attribute is present, then the ASA tears down all connections that have a source IP matching the User IP address on the configured interface.
- **validate-attribute** *number*—Additional criteria to use when building a table of user accounts when receiving Accounting-Request Start messages. These attributes help when the ASA decides whether to tear down connections.

  If you do not specify additional attributes to validate, the decision is based solely on the IP address in the Framed IP Address attribute. If you configure additional attributes, and the ASA receives a start accounting message that includes an address that is currently being tracked, but the other attributes to validate are different, then all connections started using the old attributes are torn down, on the assumption that the IP address has been reassigned to a new user.

  Values range from 1-191, and you can enter the command multiple times. For a list of attribute numbers and their descriptions, see http://www.iana.org/assignments/radius-types.
- **host** *ip_address* [**key** *secret*]—The IP address of the RADIUS server or GGSN. You can optionally include a secret key so that the ASA can validate the message. Without the key, only the IP address is checked. You can repeat this command to identify multiple RADIUS and GGSNs hosts. The ASA receives a copy of the RADIUS accounting messages from these hosts.

- **timeout users** *time*—Sets the idle timeout for users (in hh:mm:ss format). To have no timeout, specify 00:00:00. The default is one hour.

Example

```
policy-map type inspect radius-accounting radius-acct-pmap
  parameters
    send response
    enable gprs
    validate-attribute 31
    host 10.2.2.2 key 123456789
    host 10.1.1.1 key 12345
class-map type management radius-class
  match port udp eq radius-acct
policy-map global_policy
  class radius-class
    inspect radius-accounting radius-acct-pmap
```

Configure the RADIUS Accounting Inspection Service Policy

RADIUS accounting inspection is not enabled in the default inspection policy, so you must enable it if you need this inspection. Because RADIUS accounting inspection is for traffic directed to the ASA, you must configure it as a management inspection rule rather than a standard rule.

Procedure

Step 1 Create an L3/L4 management class map to identify the traffic for which you want to apply the inspection, and identify the matching traffic.

```
class-map type management name
match {port | access-list} parameter
```

Example:

```
hostname(config)# class-map type management radius-class-map
hostname(config-cmap)# match port udp eq radius-acct
```

In this example, the match is for the radius-acct UDP port, which is 1646. You can specify a different port, a range of ports (**match port udp range** *number1 number2*) or use **match access-list** *acl_name* and use an ACL.

Step 2 Add or edit a policy map that sets the actions to take with the class map traffic: **policy-map** *name*

Example:

```
hostname(config)# policy-map global_policy
```

In the default configuration, the global_policy policy map is assigned globally to all interfaces. If you want to edit the global_policy, enter global_policy as the policy name.

Step 3 Identify the L3/L4 management class map you are using for RADIUS accounting inspection: **class** *name*

Example:

```
hostname(config-pmap)# class radius-class-map
```

Step 4 Configure RADIUS accounting inspection: **inspect radius-accounting** [*radius-accounting_policy_map*]

Where *radius_accounting_policy_map* is the RADIUS accounting inspection policy map you created in Configure a RADIUS Accounting Inspection Policy Map, on page 37.

Example:

```
hostname(config-class)# no inspect radius-accounting
hostname(config-class)# inspect radius-accounting radius-class-map
```

Note
If you are editing an in-use policy to use a different inspection policy map, you must remove the RADIUS accounting inspection with the **no inspect radius-accounting** command, and then re-add it with the new inspection policy map name.

Step 5 If you are editing an existing service policy (such as the default global policy called global_policy), you are done. Otherwise, activate the policy map on one or more interfaces.

service-policy *policymap_name* {**global** | **interface** *interface_name*}

Example:

```
hostname(config)# service-policy global_policy global
```

The **global** keyword applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. You can override the global policy on an interface by applying a service policy to that interface. You can only apply one policy map to each interface.

Monitoring Mobile Network Inspection

The following topics explain how to monitor mobile network inspection.

Monitoring GTP Inspection

To display the GTP configuration, enter the **show service-policy inspect gtp** command in privileged EXEC mode.

Use the **show service-policy inspect gtp statistics** command to show the statistics for GTP inspection. The following is sample output:

```
firewall(config)# show service-policy inspect gtp statistics
```

```
GPRS GTP Statistics:
  version_not_support               0     msg_too_short             0
  unknown_msg                       0     unexpected_sig_msg        0
  unexpected_data_msg               0     ie_duplicated             0
  mandatory_ie_missing              0     mandatory_ie_incorrect    0
  optional_ie_incorrect             0     ie_unknown                0
  ie_out_of_order                   0     ie_unexpected             0
  total_forwarded                  67     total_dropped             1
  signalling_msg_dropped            1     data_msg_dropped          0
  signalling_msg_forwarded         67     data_msg_forwarded        0
  total created_pdp                33     total deleted_pdp         32
  total created_pdpmcb             31     total deleted_pdpmcb      30
  total dup_sig_mcbinfo             0     total dup_data_mcbinfo    0
  no_new_sgw_sig_mcbinfo            0     no_new_sgw_data_mcbinfo   0
  pdp_non_existent                  1
```

You can get statistics for a specific GTP endpoint by entering the IP address on the **show service-policy inspect gtp statistics** *ip_address* command.

```
firewall(config)# show service-policy inspect gtp statistics 10.9.9.9
1 in use, 1 most used, timeout 0:30:00
 GTP GSN Statistics for 10.9.9.9, Idle 0:00:34, restart counter 0
  Tunnels Active                    0
  Tunnels Created                   1
  Tunnels Destroyed                 0
  Total Messages Received           1
                         Signalling Messages        Data Messages
  total received                   1                          0
  dropped                          0                          0
  forwarded                        1                          0
```

Use the **show service-policy inspect gtp pdp-context** command to display PDP context-related information. For GTPv2, this is the bearer context. For example:

```
ciscoasa(config)# show service-policy inspect gtp pdp-context
4 in use, 5 most used

Version v1,   TID 050542012151705f,  MS Addr 2005:a00::250:56ff:fe96:eec,
SGSN Addr 10.0.203.22,      Idle 0:52:01,   Timeout 3:00:00,   APN ssenoauth146

Version v2,   TID 0505420121517056,  MS Addr 100.100.100.102,
SGW Addr 10.0.203.24,      Idle 0:00:05,   Timeout 3:00:00,   APN ssenoauth146

Version v2,   TID 0505420121517057,  MS Addr 100.100.100.103,
SGW Addr 10.0.203.25,      Idle 0:00:04,   Timeout 3:00:00,   APN ssenoauth146

Version v2,   TID 0505420121517055,  MS Addr 100.100.100.101,
SGW Addr 10.0.203.23,      Idle 0:00:06,   Timeout 3:00:00,   APN ssenoauth146

ciscoasa(config)# show service-policy inspect gtp pdp-context detail
1 in use, 1 most used

Version v1,   TID 050542012151705f,  MS Addr 2005:a00::250:56ff:fe96:eec,
SGSN Addr 10.0.203.22,      Idle 0:06:14,   Timeout 3:00:00,   APN ssenoauth146

    user_name (IMSI):  50502410121507    MS address: 2005:a00::250:56ff:fe96:eec
    nsapi: 5                linked nsapi: 5
    primary pdp: Y         sgsn is Remote
    sgsn_addr_signal: 10.0.203.22    sgsn_addr_data: 10.0.203.22
    ggsn_addr_signal: 10.0.202.22    ggsn_addr_data: 10.0.202.22
    sgsn control teid:     0x00000001    sgsn data teid:        0x000003e8
```

```
    ggsn control teid:     0x000f4240    ggsn data teid:        0x001e8480
    signal_sequence:               18     state:    Ready
...
```

The PDP or bearer context is identified by the tunnel ID (TID), which is a combination of the values for IMSI and NSAPI (GTPv0-1) or IMSI and EBI (GTPv2). A GTP tunnel is defined by two associated contexts in different GSN or SGW/PGW nodes and is identified with a Tunnel ID. A GTP tunnel is necessary to forward packets between an external packet data network and a mobile subscriber (MS) user.

Monitoring SCTP

You can use the following commands to monitor SCTP.

- **show service-policy inspect sctp**

  Displays SCTP inspection statistics. The sctp-drop-override counter increments each time a PPID is matched to a drop action, but the packet was not dropped because it contained data chunks with different PPIDs. For example:

```
ciscoasa# show service-policy inspect sctp
Global policy:
  Service-policy: global_policy
    Class-map: inspection_default
      Inspect: sctp sctp, packet 153302, lock fail 0, drop 20665, reset-drop 0,
5-min-pkt-rate 0 pkts/sec, v6-fail-close 0, sctp-drop-override 4910
        Match ppid 30 35
          rate-limit 1000 kbps, chunk 2354, dropped 10, bytes 21408, dropped-bytes 958

        Match: ppid 40
           drop, chunk 5849
        Match: ppid 55
           log, chunk 9546
```

- **show sctp** [**detail**]

  Displays current SCTP cookies and associations. Add the **detail** keyword to see detailed information about SCTP associations. The detailed view also shows information about multi-homing, multiple streams, and fragment reassembly.

```
ciscoasa# show sctp

AssocID: 71adeb15
Local:  192.168.107.12/50001 (ESTABLISHED)
Remote: 192.168.108.122/2905 (ESTABLISHED)
Secondary Conn List:
    192.168.108.12(192.168.108.12):2905 to 192.168.107.122(192.168.107.122):50001
    192.168.107.122(192.168.107.122):50001 to 192.168.108.12(192.168.108.12):2905
    192.168.108.122(192.168.108.122):2905 to 192.168.107.122(192.168.107.122):50001
    192.168.107.122(192.168.107.122):50001 to 192.168.108.122(192.168.108.122):2905
    192.168.108.12(192.168.108.12):2905 to 192.168.107.12(192.168.107.12):50001
    192.168.107.12(192.168.107.12):50001 to 192.168.108.12(192.168.108.12):2905
```

- **show conn protocol sctp**

  Displays information about current SCTP connections.

- **show local-host** [**connection sctp** *start*[*-end*]]

Displays information on hosts making SCTP connections through the ASA, per interface. Add the **connection sctp** keyword to see only those hosts with the specified number or range of SCTP connections.

- **show traffic**

  Displays SCTP connection and inspection statistics per interface if you enable the **sysopt traffic detailed-statistics** command.

Monitoring Diameter

You can use the following commands to monitor Diameter.

- **show service-policy inspect diameter**

  Displays Diameter inspection statistics. For example:

```
ciscoasa# show service-policy inspect diameter
Global policy:
  Service-policy: global_policy
    Class-map: inspection_default
      Inspect: Diameter Diameter_map, packet 0, lock fail 0, drop 0, -drop 0,
5-min-pkt-rate 0 pkts/sec, v6-fail-close 0
        Class-map: log_app
            Log: 5849
         Class-map: block_ip
             drop-connection: 2
```

- **show diameter**

  Displays state information for each Diameter connection. For example:

```
ciscoasa# show diameter
Total active diameter sessions: 5
Session 3638
       ==========
       ref_count: 1 val = .; 1096298391; 2461;
           Protocol : diameter Context id : 0
           From inside:211.1.1.10/45169 to outside:212.1.1.10/3868
...
```

- **show conn detail**

  Displays connection information. Diameter connections are marked with the Q flag.

- **show tls-proxy**

  Displays information about the TLS proxy if you use one in Diameter inspection.

Monitoring M3UA

You can use the following commands to monitor M3UA.

- **show service-policy inspect m3ua drops**

  Displays drop statistics for M3UA inspection.

- **show service-policy inspect m3ua endpoint** [*IP_address*]

  Displays statistics for M3UA endpoints. You can specify an endpoint IP address to see information for a specific endpoint. For high availability or clustered systems, the statistics are per unit, they are not synchronized across units. For example:

```
ciscoasa# sh service-policy inspect m3ua endpoint
 M3UA Endpoint Statistics for 10.0.0.100, Idle : 0:00:06 :
                   Forwarded        Dropped          Total Received
 All Messages      21                5                 26
 DATA Messages     9                 5                 14
 M3UA Endpoint Statistics for 10.0.0.110, Idle : 0:00:06 :
                   Forwarded        Dropped          Total Received
 All Messages      21                8                 29
 DATA Messages     9                 8                 17
```

- **show service-policy inspect m3ua session**

  Displays information about M3UA sessions if you enable strict application server process (ASP) state validation. Information includes source association ID, whether the session is single or double exchange, and in clustering, whether it is a cluster owner session or a backup session. In a cluster with 3 or more units, you might see stale backup sessions if a unit leaves and then returns to the cluster. These stale sessions are removed when they time out, unless you disabled session timeout.

```
Ciscoasa# show service-policy inspect m3ua session
0 in use, 0 most used
Flags: o - cluster owner session, b - cluster backup session
       d - double exchange      , s - single exchange
AssocID: cfc59fbe in Down state, idle:0:00:05, timeout:0:01:00, bd
AssocID: dac2e123 in Active state, idle:0:00:18, timeout:0:01:00, os
```

- **show service-policy inspect m3ua table**

  Displays the run-time M3UA inspection table, including classification rules.

- **show conn detail**

  Displays connection information. M3UA connections are marked with the v flag.

History for Mobile Network Inspection

Feature Name	Releases	Feature Information
GTPv2 inspection and improvements to GTPv0/1 inspection.	9.5(1)	GTP inspection can now handle GTPv2. In addition, GTP inspection for all versions now supports IPv6 addresses. We changed the **match message id** command to **match message** {**v1** | **v2**} **id** *message_id* . We replaced the **timeout gsn** command with **timeout endpoint**. We removed the **gsn** keyword from the **clear/show service-policy inspect gtp statistics** command; now, simply enter the endpoint ID to see or clear these statistics. The **clear/show service-policy inspect gtp request** and **pdpmcb** commands now include a **version** keyword, so you can display information about a specific GTP version.
SCTP inspection	9.5(2)	You can now apply application-layer inspection to Stream Control Transmission Protocol (SCTP) traffic to apply actions based on payload protocol identifier (PPID). We added or modified the following commands: **clear conn protocol sctp**, **inspect sctp**, **match ppid**, **policy-map type inspect sctp**, **show conn protocol sctp**, **show local-host connection sctp**, **show service-policy inspect sctp**.
Diameter inspection	9.5(2)	You can now apply application-layer inspection to Diameter traffic and also apply actions based on application ID, command code, and attribute-value pair (AVP) filtering. We added or modified the following commands: **class-map type inspect diameter**, **diameter**, **inspect diameter**, **match application-id**, **match avp**, **match command-code**, **policy-map type inspect diameter**, **show conn detail**, **show diameter**, **show service-policy inspect diameter**, **unsupported**.
Diameter inspection improvements	9.6(1)	You can now inspect Diameter over TCP/TLS traffic, apply strict protocol conformance checking, and inspect Diameter over SCTP in cluster mode. We added or modified the following commands: **client clear-text**, **inspect diameter**, **strict-diameter**.
SCTP stateful inspection in cluster mode	9.6(1)	SCTP stateful inspection now works in cluster mode. You can also configure SCTP stateful inspection bypass in cluster mode. We did not introduce or change any commands.

Feature Name	Releases	Feature Information
MTP3 User Adaptation (M3UA) inspection.	9.6(2)	You can now inspect M3UA traffic and also apply actions based on point code, service indicator, and message class and type. We added or modified the following commands: **clear service-policy inspect m3ua** {**drops** | **endpoint** [*IP_address*]}, **inspect m3ua**, **match dpc**, **match opc**, **match service-indicator**, **policy-map type inspect m3ua**, **show asp table classify domain inspect-m3ua**, **show conn detail**, **show service-policy inspect m3ua** {**drops** | **endpoint** [*IP_address*]}, **ss7 variant**, **timeout endpoint**.
Support for SCTP multi-streaming reordering and reassembly and fragmentation. Support for SCTP multi-homing, where the SCTP endpoints have more than one IP address.	9.7(1)	The system now fully supports SCTP multi-streaming reordering, reassembly, and fragmentation, which improves Diameter and M3UA inspection effectiveness for SCTP traffic. The system also supports SCTP multi-homing, where the endpoints have more than one IP address each. For multi-homing, the system opens pinholes for the secondary addresses so that you do not need to write access rules to allow them. SCTP endpoints must be limited to 3 IP addresses each. We modified the output of the following command: **show sctp detail**.
M3UA inspection improvements.	9.7(1)	M3UA inspection now supports stateful failover, semi-distributed clustering, and multihoming. You can also configure strict application server process (ASP) state validation and validation for various messages. Strict ASP state validation is required for stateful failover and clustering. We added or modified the following commands: **clear service-policy inspect m3ua session** [**assocID** *id*], **match port sctp**, **message-tag-validation**, **show service-policy inspect m3ua drop**, **show service-policy inspect m3ua endpoint**, **show service-policy inspect m3ua session**, **show service-policy inspect m3ua table**, **strict-asp-state**, **timeout session**.
Support for setting the TLS proxy server SSL cipher suite.	9.8(1)	You can now set the SSL cipher suite when the ASA acts as a TLS proxy server. Formerly, you could only set global settings for the ASA using the **ssl cipher** command. We introduced the following command: **server cipher-suite**
GTP inspection enhancements for MSISDN and Selection Mode filtering, anti-replay, and user spoofing protection.	9.10(1)	You can now configure GTP inspection to drop Create PDP Context messages based on Mobile Station International Subscriber Directory Number (MSISDN) or Selection Mode. You can also implement anti-replay and user spoofing protection. We added the following commands: **anti-replay**, **gtp-u-header-check**, **match msisdn**, **match selection-mode**.

Feature Name	Releases	Feature Information
GTPv1 release 10.12 support.	9.12(1)	The system now supports GTPv1 release 10.12. Previously, the system supported release 6.1. The new support includes recognition of 25 additional GTPv1 messages and 66 information elements. In addition, there is a behavior change. Now, any unknown message IDs are allowed. Previously, unknown messages were dropped and logged. We did not add or change any commands.
Location logging for mobile stations (GTP inspection).	9.13(1)	You can configure GTP inspection to log the initial location of a mobile station and subsequent changes to the location. Tracking location changes can help you identify possibly fraudulent roaming charges. We added the following command: **location-logging**.
GTPv2 and GTPv1 release 15 support.	9.13(1)	The system now supports GTPv2 3GPP 29.274 V15.5.0. For GTPv1, support is up to 3GPP 29.060 V15.2.0. The new support includes recognition of 2 additional messages and 53 information elements. We did not add or change any commands.
Ability to specify the IMSI prefixes to be dropped in GTP inspection.	9.16(1)	GTP inspection lets you configure IMSI prefix filtering, to identify the Mobile Country Code/Mobile Network Code (MCC/MNC) combinations to allow. You can now do IMSI filtering on the MCC/MNC combinations that you want to drop. This way, you can list out the unwanted combinations, and default to allowing all other combinations. We added the following command: **drop mcc**.
Secure Firewall 3100 support for the Carrier license	9.18(1)	The Carrier license enables Diameter, GTP/GPRS, SCTP inspection. New/Modified commands: **feature carrier**