ADC Administration Guide, StarOS Release 21.19

First Published: 2020-05-11

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

About this Guide

Note Control and User Plane Separation (CUPS) represents a significant architectural change in the way StarOS-based products are deployed in the 3G, 4G, and 5G networks. Unless otherwise specified, it should not be assumed that any constructs (including, but not limited to, commands, statistics, attributes, MIB objects, alarms, logs, services) referenced in this document imply functional parity with CUPS products. References to any CUPS products or features are for informational purposes only. Please contact your Cisco Account or Support representative for any questions about parity between this product and any CUPS products.

Note The documentation set for this product strives to use bias-free language. For purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

Note The HA, HSGW, PDSN, and SecGW products have reached end of life and are not supported in this release. Any references to these products (specific or implied) their components or functions including CLI commands and parameters in this document are coincidental and are not supported. Full details on the end of life for these products are available at https://www.cisco.com/c/en/us/products/collateral/wireless/asr-5000-series/eos-eol-notice-c51-740422.html.

This preface describes the *ADC Administration Guide*, how it is organized and its document conventions.

Application Detection and Control (ADC) is a StarOS™ in-line service application that runs on Cisco® ASR 5500 platform.

The ADC in-line service makes use of innovative and highly accurate protocol behavioral detection techniques to reliably detect protocols and applications in the network. ADC is mainly used to detect Peer-to-Peer protocols by analyzing traffic, and can also detect network traffic created by audio and video clients.

Conventions Used

The following tables describe the conventions used throughout this documentation.

Notice Type	Description
Information Note	Provides information about important features or instructions.
Caution	Alerts you of potential damage to a program, device, or system.
Warning	Alerts you of potential personal injury or fatality. May also alert you of potential electrical hazards.

Typeface Conventions	Description
Text represented as a `screen display`	This typeface represents displays that appear on your terminal screen, for example: `Login:`
Text represented as **commands**	This typeface represents commands that you enter, for example: **show ip access-list** This document always gives the full form of a command in lowercase letters. Commands are not case sensitive.
Text represented as a **command** *variable*	This typeface represents a variable that is part of a command, for example: **show card** *slot_number* *slot_number* is a variable representing the desired chassis slot number.
Text represented as menu or sub-menu names	This typeface represents menus and sub-menus that you access within a software application, for example: Click the **File** menu, then click **New**

Supported Documents and Resources

Related Common Documentation

The following common documents are available:

- *AAA Interface Administration and Reference*
- *Command Line Interface Reference*

- *GTPP Interface Administration and Reference*
- *Installation Guide* (platform dependant)
- *Release Change Reference*
- *SNMP MIB Reference*
- *Statistics and Counters Reference*
- *System Administration Guide* (platform dependant)
- *Thresholding Configuration Guide*

Related Product Documentation

The most up-to-date information for this product is available in the product Release Notes provided with each product release.

The following product documents are also available and work in conjunction with ADC:

- *ECS Administration Guide*
- *GGSN Administration Guide*
- *P-GW Administration Guide*

Obtaining Documentation

The most current Cisco documentation is available on the following website:

http://www.cisco.com/cisco/web/psa/default.html

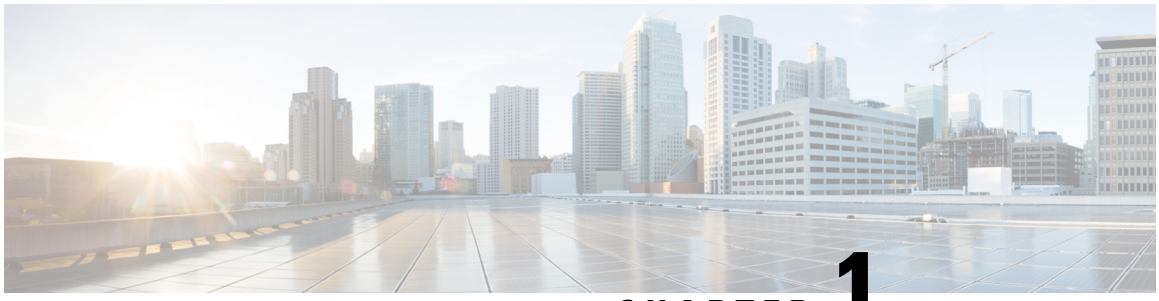
Use the following path selections to access the ADC documentation:

Products > Wireless > Mobile Internet > In-Line Services > Cisco Peer-to-Peer

Contacting Customer Support

Use the information in this section to contact customer support.

Refer to the support area of http://www.cisco.com for up-to-date product documentation or to submit a service request. A valid username and password are required to access this site. Please contact your Cisco sales or service representative for additional information.



CHAPTER 1

Application Detection and Control Overview

This chapter provides an overview of the Application Detection and Control (ADC) in-line service, formerly known as Peer-to-Peer Detection.

The System Administration Guide provides basic system configuration information, and the product administration guides provide procedures to configure basic functionality of core network service. It is recommended that you select the configuration example that best meets your service model, and configure the required elements for that model, as described in the respective product Administration Guide, before using the procedures in this chapter.

This chapter covers the following topics:

ADC Overview

The ADC in-line service is mainly used to detect Peer-to-Peer protocols by analyzing traffic. Other popular applications that generate the bulk of Internet traffic like Social Networking and Gaming applications can be detected.

The ADC in-line service works in conjunction with the following products:

- GGSN
- PDSN
- P-GW

The in-line service now known as ADC is continued to be referred as "P2P" in the configuration.

Peer to Peer (P2P) is a term used in two slightly different contexts. At a functional level, it means protocols that interact in a peering manner, in contrast to client-server manner. There is no clear differentiation between the function of one node or another. Any node can function as a client, a server, or both — a protocol may not clearly differentiate between the two. For example, peering exchanges may simultaneously include client and server functionality, sending and receiving information. P2P is a type of transient Internet network that allows a group of computer users with the same networking program to connect with each other and directly access files from one another's hard drives. A common use case of a P2P application is file sharing.

Once the P2P Client is downloaded and installed, it will log on to a central indexing server. This central server indexes all users who are currently online connected to the server. This server does not host any files for downloading. The P2P client can search for a specific file. The utility queries the index server to find other connected users with the specific file. When a match is found, the central server directs to find the requested

file. The result is chosen from the search query and the utility will then attempt to establish a connection with the computer hosting the requested file. If a successful connection is made, it will begin downloading the file. Once the file download is complete, the connection will be broken.

The stunning growth and intensive bandwidth nature of P2P applications can have a significant impact on the underlying network. As most deployments are designed with a significant bias towards downstream traffic, P2P applications stress uplink capacity resulting in increased latency, decreased responsiveness and packet loss.

To avoid detection, P2P software undergoes frequent changes and this requires service providers to upgrade the software with the latest P2P detection logic. This upgrade is time consuming, also causing disruption in services and revenue loss. The Dynamic Software Upgrade (DSU) addresses these problems by enabling operators to upgrade their detection capabilities with no downtime. The detection logic is separated out from the main code and shipped as a plugin. Whenever there is a need for software upgrade, the new plugin will be shipped and loaded into the system. For more information, refer to the *Dynamic Software Upgrade* section.

Qualified Platforms

ADC is a StarOS in-line service application that runs on Cisco ASR 5500 platforms. For additional platform information, refer to the appropriate System Administration Guide and/or contact your Cisco account representative.

License Requirements

The ADC is a licensed Cisco feature. A separate feature license may be required. Contact your Cisco account representative for detailed information on specific licensing requirements. For information on installing and verifying licenses, refer to the *Managing License Keys* section of the *Software Management Operations* chapter in the *System Administration Guide*.

Dynamic Software Upgrade

This section describes the Dynamic Software Upgrade (DSU) process that can be used to dynamically update plugins without having to update StarOS and reload the system.

Overview

Dynamic Software Upgrade is the new approach to upgrade the P2P library version that will enable operators to upgrade their detection capabilities with no downtime. This is done by providing updates in the form of software patches which the operator can apply in a live setup with minimal interference.

In this approach, P2P detection code is now delivered as a plugin within the StarOS binary. The plugin is loaded into the system at run time. Whenever there is a change in P2P detection logic of an existing application or a new P2P protocol/application needs to be added, a new version of the plugin is provided as a plugin module. The new plugin is loaded onto the system dynamically without disrupting other services. Once the plugin has been installed and configured, the new P2P rules come into effect for detection.

Important The dynamically loaded plugins are not incremental. A plugin loads protocol detection logic for all the protocols/applications. A user can update to a higher priority plugin or rollback to a lower priority plugin.

Patching is the process used to install a plugin as an update to a StarOS release. One patch can be provided to multiple compatible, concurrent product releases. A plugin patch is distributed in the form of a compressed distribution kit through the internet or by other means (USB, CD, etc.).

A plugin is a functional software entity that provides updates to a pre-existing StarOS software component. Plugins can be dynamically loaded at runtime and do not require a system restart.

A plugin module is a specific instance of a plugin version consisting of at least one file that can be added to a running, in-service system. The module contains the information or instructions for a specific component's update. Typically this will be a single plugin file.

The Version Priority List (VPL) is a linked list of module versions associated with a specific plugin. Each plugin has one VPL. The list is sorted in ascending order by the priority number that is assigned by the administrator. When updating, the lowest priority number is loaded first and if that version is not successful, the version in the VPL with the next sequentially greater priority number is loaded. This list is iterated until a successful version is found. The VPL also supports manual rollback to a previous version (higher priority number).

The basic sequence for the dynamic software upgrade process is as follows:

- Downloading the Patch Kit
- Unpacking the Patch Kit
- Configuring the Plugin
- Loading the Plugin
- Rolling Back to a Previous Plugin Version

For the detailed procedure on performing dynamic software upgrade, refer to the *Configuring Dynamic Software Upgrade* section of the *Application Detection and Control Configuration* chapter.

Note For information on the applications and protocols currently supported by the Application Detection and Control in-line service, contact your Cisco Account representative.

License Requirements

From Release 21.6 onwards, DSU is a licensed-controlled feature. A separate feature license may be required. Contact your Cisco account representative for detailed information on specific licensing requirements. For information on installing and verifying licenses, refer to the *Managing License Keys* section of the *Software Management Operations* chapter in the *System Administration Guide*.

Limitations for DSU

This section lists the limitations of Dynamic Software Upgrade.

- Support for session recovery is limited and there is no support for ICSR in this release.
- The system will allow loading two plugins at the most at any point of time. If there is a need to upgrade the system again, the oldest plugin will be unloaded.
- Detection state for a few subscribers may be lost if a plugin is unloaded from the memory.
- The newly upgraded plugin will be used for all new calls. The existing calls will continue to use the previous plugin.

ADC Protocol Group Detection

Application Detection and Control (ADC) performs traffic analysis and classifies flows into applications and its traffic type. To provide a high-level classification, the protocol grouping feature is implemented to support various application/protocol groups like gaming, file-sharing, e-mail, communicator, etc. Protocol Grouping is done based on the functionality provided by the application. For example, applications like Skype and Yahoo are used for VoIP, so these applications are grouped as Communicator group. The feature is implemented based on the Dynamic Software Upgrade plugin philosophy.

For configuration-related information, refer to the *Configuring P2P Protocol Groups* section of the *Application Detection and Control Configuration* chapter.

Behavioral Traffic

Behavioral Traffic Analysis is a method to analyze network traffic such that all the traffic is analyzed by the generic behavior of each flow. ADC supports behavioral traffic analysis for P2P (Peer-to-Peer), Video, VoIP (Voice over IP), Upload and Download. If the generic behavior of protocols is detected and traffic classified correctly using behavioral analysis, lesser amount of unknown traffic flows can be seen. These behavioral detections must not be used for charging purposes. This feature is based on the Dynamic Software Upgrade plugin philosophy.

Behavioral P2P and behavioral VoIP are meant for zero day detection of P2P/file sharing protocols and VoIP traffic respectively. Behavioral Video is meant for support of video detection. Behavioral Upload/Download must detect flows of non-standard ports that cannot be detected by ECS. This is similar to client-server upload/download using HTTP/FTP/SFTP encrypted download.

Important This feature is disabled by default and meant only for statistical purposes (not for charging purposes).

For configuration-related information, refer to the *Configuring Behavioral P2P and VoIP* section of the *Application Detection and Control Configuration* chapter.

SNI Detection

Server Name Indication (SNI) is an extension of the Transport Layer Security (TLS) protocol that provides a mechanism for the client to tell the server which hostname it is trying to connect to.

ADC detects encrypted traffic using the SNI field (signatures) of TLS/SSL (Secure Sockets Layer) traffic. Due to increased number of applications moving towards TLS/SSL, an option is provided to configure the SNI in ruledef and classify traffic based on the configured SNI with this release.

For detailed overview and configuration information, see the *Support for SNI Detection* chapter in this guide.

SSL Renegotiation Tracking

SSL Renegotiation flows can be detected by tracking the SSL Session ID and its associated protocol. This feature is disabled by default. CLI support is added to enable or disable this feature. The maximum entries of SSL Session ID tracked per Session Manager and the reduce factor can be configured.

Certain applications like Facebook, Gmail, Yahoo, Skype, Twitter, iCloud, etc. widely use the SSL Session Renegotiation feature in their mobile applications to reduce the computational intensive operations involved in a complete SSL negotiation.

Limitations: In certain cases, the SSL Renegotiation detection logic does not work if the SSL sessions involved in the negotiation is spread across more than one subscriber session.

For configuration-related information, refer to the *Configuring SSL Renegotiation* section of the *Application Detection and Control Configuration* chapter.

Analyzer Interworking

Analyzer interworking feature is implemented to analyze the various analyzers simultaneously if the flow is detected as P2P and based on ruledef priority, appropriate charging action will be taken. Currently supported analyzers are FTP, HTTP, RTSP and SIP. CLI support is added to enable or disable this feature. This feature is enabled by default if P2P detection/protocol is enabled.

For configuration-related information, refer to the *Configuring Analyzers* section of the *Application Detection and Control Configuration* chapter.

Traffic Sub-classification

ADC has the capability to detect network traffic for sub-classification of audio, file transfer, instant messaging, video, voipout or unclassified traffic. The duration of the call is a direct indication to the revenue impact of the network operator. The ADC in-line service is well poised to process the network traffic online to detect and control the presence of different network traffic, and generate records that can be used to calculate the traffic call duration.

ADC Support for TRM/FP

P2P flows now support the Transactional Rule Matching (TRM) feature. The TRM/FP feature enables the Enhanced Charging Service (ECS) to bypass per-packet rule matching on a transaction once the transaction is fully classified. This enables ECS to better utilize CPU resources and accommodate additional throughput for the system, thus improving the overall performance.

A transaction for TRM can be defined as the entire UDP flow, the ACK of the 3-way handshake to the FIN/RST of a TCP flow, or the HTTP request to the next HTTP request, or HTTP request to the FIN/RST for the final request of the flow. The TRM feature can also perform rule matching on IP L4 rules (UDP, TCP), HTTP, and HTTPS.

For more information on the TRM/FP feature, refer to the *ECS Administration Guide*.

ADC Support for FAPA

The Flow Aware Packet Acceleration (FAPA) feature improves the throughput in terms of PPS, by caching rule matching results of a flow for selected flows so as not to incur the lookup penalty for a large number of packets in that flow. This new accelerated path is capable of performing a full range of basic functions including handling charging, modification of packet headers, and incrementing various counters. The accelerated path dynamically evaluates the current flow state and reverts back to the slow path when the flow cannot be handled on the fast path.

TRM/FP support has been extended beyond rule-matching. The FAPA function identifies packets that need only a small amount of processing, and performs necessary tasks on these packets. Only those packets that do not require DPI are allowed to enter the Accelerated path. VoLTE, encrypted, HTTP, HTTPS, RTP and plain TCP/UDP traffic where L7 analysis is not enabled, and so on are all the flows that will get accelerated.

Important A Flow Aware Packet Acceleration license is required on ASR5500 and VPC platforms.

P2P flows will be optimized and accelerated using FAPA. ADC when enabled with FAPA improves P2P performance considerably.

Important FAPA accelerates P2P flows for most protocols except for some protocols/applications explicitly listed below.

FAPA accelerates some P2P flows for the following protocols and not all:

- Ares
- Bittorrent
- Didi
- DirectConnect
- Edonkey
- Iskoot
- PPlive
- Scydo
- Soulseek
- Thunder
- ThunderHS
- Tunnelvoice
- Viber
- Whatsapp
- Winny
- Zattoo

The following protocols do not support FAPA:

- ActionVoip
- BBM
- Blackdialer
- Facetime
- Gtalk
- Jabber
- Jumblo
- Kakaotalk
- Magicjack
- MyPeople
- Nateontalk
- Oscar
- Paltalk
- Plingm
- Skype
- Smartvoip
- Tango

- Voipdiscount
- Vopium
- Vtok

For more information on the FAPA feature, refer to the *ECS Administration Guide*.

ADC Support over Gx

The ADC Rule feature will support detection of application level flows as described in Release-11 of 3GPP standard. ADC Rules are certain extensions to dynamic and predefined PCC Rules in order to support specification, detection and reporting of an application flow. These rules are installed (modified/removed) by PCRF via CCA-I/CCA-U/RAR events. ADC rules can be either dynamic PCC or predefined PCC rules, and the existing attributes of dynamic and predefined rules will be applicable.

Important ADC Rule support is a licensed-controlled feature. Contact your Cisco account representative for detailed information on specific licensing requirements.

When the license is not enabled, P2P continues to function as per its original behavior, that is, it monitors the traffic at the entire rulebase level. When the license is enabled, the P2P behavior changes such as to monitor traffic at per subscriber level.

In 19.3 and later releases, this feature is extended to support non-ADC based rules (ECS protocols) in addition to existing P2P protocols, and also detection of application flows for Group of Ruledefs. The following enhancements are supported:

- ADC rules support combination of P2P and non-P2P rule lines in the same ruledef.
- Detection of application flows based on group of ruledefs.
- Application START/STOP event reporting at instance level, that is, per flow basis. This was supported per Application ID basis in previous releases.
- Support of dynamic routes to analyzers for installed ADC rules. Dynamic routes will be supported only for these protocols - HTTP, HTTPS, FTP, RTP, RTCP and SIP.
- Support multi-line AND logic for rulelines when configuring ADC ruledefs.
- Removal of all PCC rules will result in termination of Application Detection for that application. In previous releases, if more than one PCC rule with same Application ID is installed, then removal of any of the PCC rules will terminate Application Detection for that application.

Dynamic PCC rule contains either traffic flow filters or Application ID. When Application ID is present, the rule is treated as ADC Rule. Application ID is the name of the ruledef which is pre-defined in the ASR 5500. This ruledef contains application filters that define the application supported by P2P protocols and non-P2P protocols.

In releases prior to release 19.3: PCEF will process and install ADC rules that are received from PCRF interface, and will detect the specified application(s) and report detection of application traffic to the PCRF. Reporting of application traffic are controlled by PCRF and generates Application Start/Stop events along with the Application ID. Application mute status can be enabled or disabled on both dynamic and predefined ADC rules. When mute is disabled, Application Start/Stop event trigger will be generated by PCEF for that specific Application ID. Mute status can be enabled or disabled by PCRF for dynamic rules, and configured on ASR 5500 for pre-defined rules.

In 19.3 and later releases: When a subscriber attaches to the network, PCRF will install ADC rule/Group of Ruledefs towards PCEF to detect Application flow. The Install ADC rules will additionally enable default routes to HTTP, HTTPS, FTP, RTSP, RTCP or SIP analyzer based on the rule-definition. The default routes use the standard ports associated with the respective protocol. When a new flow comes, the route matching happens for dynamic routes first, then static routes and finally default routes. When a flow matches that ADC rule, an APP-START notification is sent to PCRF with Application ID, Instance ID and flow information. Instance ID is a unique identifier for a particular ADC flow. PCRF then takes necessary action for the detected application. When ADC flow terminates, an APP-STOP notification is sent to PCRF with Application ID and Instance ID.

The following types of ADC ruledefs can be configured.

- **ruledef adc_rule**
  p2p protocol = <name>
  p2p protocol-group = <name>
  p2p behavioral = <name>
  multiline-or all-lines
  end

  multiline-or all-lines is optional if rule contains only one line.

- **ruledef adc_rule_type2**
  p2p protocol = <name>
  p2p traffic-type = <sub_type_name>
  end

- **ruledef adc_rule_type3**
  p2p anymatch = TRUE
  end

  When **p2p any-match** = **TRUE** is configured, only one rule containing this rule line can be installed. This rule line must not be used with any other P2P rule line.

ADC Mute Customization

Earlier, 3GPP ADC over Gx did not support application MUTE status change. Once the application was muted, it was not possible to unmute it. From release 21.1, this feature introduces custom MUTE/UNMUTE functionality. ASR 5500 PCEF now supports customization to control reporting of the Application Detection Information CCRUs. For this, an AVP has been introduced with two possible values - custom MUTE and custom UNMUTE.

- A Gx message might contain both Standards based MUTE and the custom MUTE.
- Standards based MUTE is given preference over the custom MUTE/UNMUTE.
- A dynamic ADC rule can be installed and modified with a custom MUTE.
- Custom-Mute-Notification AVP can be sent by the PCRF in CCA-I and RAR.
- A dynamic ADC rule can be modified with a custom UNMUTE.
- On a custom MUTE for a given dynamic ADC rule, PCEF sends a single APPLICATION_START/ APPLICATION_STOP response for the entire application traffic rather the per flow APPLICATION_START /APPLICATION_STOP response.
- On a custom MUTE for a given dynamic ADC rule, if no APPLICATION_START has been sent prior to the custom MUTE then a single APPLICATION_START is sent on the next flow packet that hits the dynamic rule.

- On a custom MUTE for a given dynamic rule, the APPLICATION_START response is sent with the flow's 5-tuple information.
- On a custom MUTE for a given dynamic rule, the APPLICATION_START response is sent with TDF-Application-Instance-Identifier = 0.
- On a custom MUTE for a given dynamic rule, a single APPLICATION_STOP is sent when the last flow associated with the given dynamic rule is terminated. Such an APPLICATION_STOP will not contain 5-tuple information of the last flow and is sent with TDF-Application-Instance-Identifier = 0.
- On a custom UNMUTE for a given dynamic rule, APPLICATION STARTs response is matched with the given dynamic rule and then sent to all the forthcoming flows.
- There is no change in behavior for a custom UNMUTE, which has not been custom MUTED or standard MUTED before UNMUTING. APPLICATION_STARTs and APPLICATION_STOPs is continued to be sent per flow as before.
- On a custom UNMUTE, PCEF sends an APPLICATION_STOP each for all flows that terminate then onwards.
- A given dynamic rule is recovered in both SR and ICSR including the Custom MUTE/UNMUTE status. The APPLICATION_START status for a given dynamic rule is check-pointed and recovered. This ensures that an extra APPLICATION_START is not sent to the PCRF post recoveries.

TOS/DSCP Support

In 19.3 and later releases, the ADC functionality is extended to identify applications and distinguish bearer traffic based on TOS/DSCP. DSCP/TOS based ADC dynamic rules over Gx will be supported for default and dedicated bearers. Bearer mapping and rule matching will be done based on DSCP/TOS value. Filters can be created for PCC rules based on TOS-Traffic-Class AVP under flow information.

When a subscriber attaches to the network, PCRF will install PCC rule with TOS/DSCP filter towards PCEF. PCEF will create a dedicated bearer and send the packet filters to UE as well. When a new flow comes with first packet as Uplink, UE does bearer matching based on the TOS/DSCP value, and sends flow on the correct dedicated bearer. For downlink packet, ECS does bearer lookup and assigns correct bearer to the flow based on the TOS/DSCP value.

ADC Event reporting will contain flow template with outer IP 3 tuples (Source IP, Destination IP, Port). L4-L7 rule match will also work for PMIP service.

Limitations

The limitations for the ADC over Gx feature are:

- Registration of the duplicate application IDs are not supported.
- Readdress/Redirection for P2P flows will not be supported.
- Redirection happens only on transactions of GET/Response.
- Port based, IP Protocol based, and URL based applications are not supported.
- Pre-configured options (precedence, redirect-server-ip) for dynamic ADC Rules are not supported.
- Simultaneous instances of an application for the same subscriber are not distinguished.
- Flow recovery is not supported for application flows.

Dynamic Advertisement Server Correlation

ADC supports many streaming applications that are ad-supported and the flows corresponding to these third-party advertisements are generic. These advertisement flows could not be differentiated from specific

application flows based on the deterministic pattern. As part of this feature, a configurable option is provided to dynamically correlate advertisement flows and associate the respective applications.

Any advertisement service is associated with the corresponding application protocol. The type of ad-flow will be configured per application. Refer to the *Configuring P2P Advertisement server* section in the *Application Detection and Control Configuration* chapter for more information on configuring the P2P Advertisement Server correlation group.

Limitations

Some limitations of this feature are listed below:

- Maximum number of ads groups that can be configured is 100.
- Maximum number of ad-source lines per ads-group that can be configured is 32.
- Configuration will take effect only for new flows.
- Applications added using TLS/SNI ruledefs (custom defined protocols) will not be supported.

Bulk Statistics Support

The system can be configured to collect bulk statistics (performance data) and send them to a collection server (called a receiver). Bulk statistics are statistics that are collected in a group. The individual statistics are grouped by schema. ADC uses P2P schema for bulk statistics support.

Important The bulk statistics format previously supported by the older implementation for individual ADC protocols in ECS schema is deprecated, and the new bulk statistics format is supported in 14.0 and later releases in the new P2P schema.

The P2P schema is designed in such a way that all variables that end with numeral value "name" are used to extract all data with numeral values "value" for all the protocols supported by the currently loaded patch. With the Dynamic Software Upgrade, the operator need not change the P2P schema by adding or removing variables related to a particular protocol manually for each new patch.

The following is a sample configuration of bulk statistics in the P2P schema:

p2p schema p2p *format*

"%p2p-protocol%\n%p2p-protocol-group%\n%p2p-uplnk-bytes-name%:%p2puplnk-bytes-value

%\n%p2p-dwlnk-bytes-name%:%p2p-dwlnk-bytes-value%\n%p2p-uplnk-pktsname%:%p2p-uplnk-pkts-value%

p2p-uplnk-pkts-value%\n%p2p-dwlnk-pkts-name%:%p2p-dwlnk-pkts-value%\n%p2pduration-name%:

%p2p-duration-value%\n----------\n"

Important If detection of a specific P2P protocol is enabled, bulk statistics for that protocol will be automatically generated based on the plugin installed on the chassis. In the case of protocols that support sub-classification (audio/file transfer/instant messaging/video/voipout/unclassified), the bulk statistics will be dynamically generated for each of the supported sub-classifications per protocol and also the corresponding total count which is the sum of values of the sub-classified data.

For more information, see the *P2P Schema* chapter of the *Statistics and Counters Reference*.

How ADC Works

As part of traffic analysis, packets will be first passed through the ADC analyzers when "p2p dynamic-flow-detection" is enabled. If it is not detected as P2P by any of the ADC analyzers, then it will follow the rule matching to find an application analyzer.

ADC analyzers examine uplink and downlink traffic and use rules that define what packet content to take action on and what action to take when the rule is true. The analyzers also generate usage records for the billing system. The rules are configured/defined in the same way as ECS in-line service ruledefs and rulebases.

For a few specific protocols, packets will be sent to non-ADC analyzers even after marking the flow as P2P. If the flow is marked as P2P and also analyzed by other analyzers, the statistics for display and debug purposes reflect in both analyzers. The EDR also displays the ADC application/protocol names if configured.

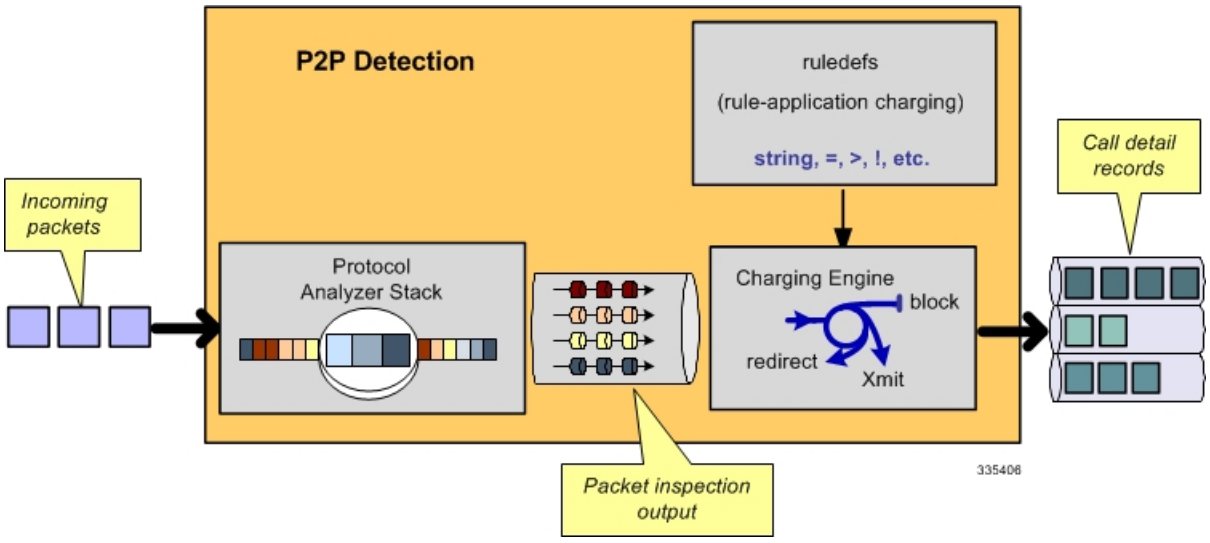
ADC also interfaces to a PCRF Diameter Gx interface to accept policy ACLs and rulebases from a PDF. ADC supports real-time dynamic policy updates during a subscriber session. This includes modifying the subscriber's policy rules during an active session by means of ACL name and Rulebase name.

In Gx interface, a Charging Rulebase will be treated as a group of ruledefs. A group of ruledefs enables grouping rules into categories, so that charging systems can base the charging policy on the category. When a request contains names of several Charging Rulebases, groups of ruledefs of the corresponding names are activated. For ADC rules to work in the group of ruledefs, P2P detection has to be enabled in the rulebase statically.

Static policy is supported initially. A default subscriber profile is assumed and can be overwritten on the gateway. Per-subscriber static policy is pulled by the gateway from the AAA service at subscriber authentication.

The following figure illustrates how packets travel through the system using ADC. The packets are investigated and then handled appropriately using ruledefs for charging.

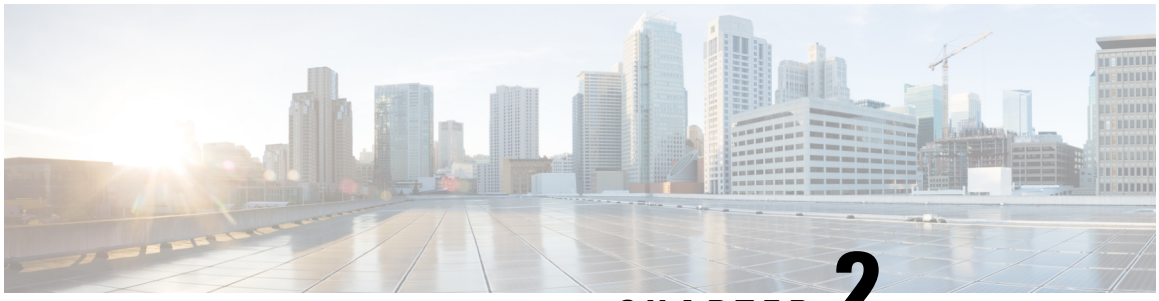
Figure 1: Overview of Packet Processing in ECSv2



Limitations

This section lists the limitations for the ADC protocols that support audio and video sub-classification.

If Audio and Video contents are in the same flow (TCP/UDP), video is considered as the predominant component and the flow is marked as "video". In this scenario, throttling video will block both audio and video. Throttling only audio or video is not possible.



CHAPTER 2

Application Detection and Control Configuration

This chapter describes how to configure the Application Detection and Control (ADC) feature.

This chapter covers the following topics:

Configuring Dynamic Software Upgrade

This section describes how to install and configure the dynamic software upgrade plugin in the ASR 5500.

How to perform Dynamic Software Upgrade

The procedure for the dynamic software upgrade is as follows:

Step 1 Obtain the patch TAR file from your designated Cisco representative.

Step 2 Copy the patch file into the */flash* directory of the ASR 5500 chassis by the TFTP or FTP method.

`patch plugin` *`plugin_name filepath`*

For example:

```
patch plugin p2p /flash/libp2p-1.97.354.so.tgz
```

Step 3 After the file has been copied, install the plugin using the following **install plugin** command from the Exec mode.

`install plugin` *`plugin_name patch_file_name`*

For example:

```
install plugin p2p patch_libp2p-1.97.354.so.tgz
patch file patch_libp2p-1.97.354.so.tgz installed successfully
```

Important The plugin will be unpacked into */flash/patch/p2p* directory. Ensure that there is enough space in the */flash* directory before installing a given patch. Verify if the file is installed correctly using the **dir /flash/patch/p2p/[patch_version_number.so]** command. For example: dir /flash/patch/p2p/libp2p-1.97.354.so

Step 4 After the patch has been successfully installed, the patch version must be configured before it can be loaded into the system. To configure the patch before any other version of patch, first check the existing plugin configuration using the following command. This command entered in the Exec mode is used to list the priorities on the configured patches.

show plugin *plugin_name*

For example:

show plugin *p2p*

Important The data is read from */flash/module.sys* and if it is not available, reads the default priorities from */etc/plugin.conf* (read only) and lists the priorities.

Step 5 Configure the plugin using the **plugin** command from the Global Configuration mode, and enters the Plugin Configuration mode.

plugin <*plugin_name*>

module priority <*number*> **version** <*module_version*>

For example:

```
configure
  plugin p2p
  module priority 1 version 1.97.354
```

The plugin name must match the name of the plugin which has been copied to and unpacked on the system or an error message is displayed.

Important The above configuration will be internally stored in */flash/module.sys* so that the current configuration survives an ASR 5500 reload. Ensure that the file is not deleted by mistake.

Step 6 Enter the following command to update the specified module running in the system. Wait 5-10 seconds for the update to occur on all the PSC cards.

update module *<plugin_name>*

For example:

```
update module p2p
Update to module p2p version 1.97.354 successful
```

When the above command is issued, the priorities configured using **module priority** command is converted into a Module priority List and then the module with least priority is loaded. If it fails to load, the module with next higher priority is attempted and so on till a successful load occurs.

Important If none of the configured modules load properly, then the system automatically tries to load the default patch that comes along with the currently loaded build.

Step 7 Enter the following command to rollback a running patch version in the system:

rollback module *<plugin_name>*

For example:

```
rollback module p2p
rollback plugin p2p module priority 10 library /var/opt/lib/libp2p-1.97.357.so load successful
```

When the above command is issued, the system automatically tries to load the patch with next highest priority in the VPL and if that fails, it tries to load the next one and so on.

Step 8 Enter the following command in Exec mode to view the status of the currently running VPL. Ensure that the patch with least priority is loaded. Others will have the "loaded" column displayed as "no".

show module *<plugin_name>* **[verbose]**

For example:

```
show module p2p
Module p2p
Priority   version  loaded     location       update/rollback time     status
     1   1.97.354     no    /var/opt/lib   FRI Feb 28 07:15:42 2014   success
     2   1.97.357    yes    /var/opt/lib   FRI Feb 28 07:15:42 2014   success
     X   1.82.118     no        /lib               (never)            N/A
```

Step 9 Delete older patch files. Unconfigure plugin from configuration and manually delete the older patch files from */flash/patch/<plugin_name>* directory to save disk space if necessary.

plugin *<plugin_name>*
no module priority *<plugin_number>*

For example:

```
configure
  plugin p2p
    no module priority 1
    end
del /flash/patch/p2p/libp2p-1.97.354.so
del /flash/patch/p2p/patch_libp2p-1.97.354.so.tgz
```

Step 10 For an ICSR setup:

If the ASR 5500 system is in ICSR mode (geographical redundancy) then the operator has to repeat the above steps for update/rollback in both the systems individually. If it is not done, then after an SRP switchover the new active ASR 5500 comes up with an outdated plugin priority which can lead to loading an older version of patch for a particular plugin. Using the **srp validate-configuration** command, check for the same plugin module priority and raise error if it is different across the active-standby pair.

Configuring System for ADC

This section lists the high-level steps to configuring the system with enhanced charging services for ADC in conjunction with ECS services.

To configure the system for ADC support with ECS:

Step 1 Set initial configuration parameters such as modifying the local context as described in the Initial Configuration section.

Step 2 Enable the Enhanced Charging service with ADC and set basic ECS parameters such as service configuration, Ruledefs, charging actions, and EDRs as described in the ADC Configuration section.

Step 3 Save your configuration to flash memory, an external memory device, and/or a network location using the Exec mode command save configuration. For additional information on how to verify and save configuration files, refer to the *System Administration Guide* and the *Command Line Interface Reference*.

Important Commands used in the configuration examples in this section provide base functionality to the extent that the most common or likely commands and/or keyword options are presented. In many cases, other optional commands and/or keyword options are available. Refer to the *Command Line Interface Reference* for complete information regarding all commands.

Initial Configuration

To perform initial system configuration for ADC support with ECS:

Step 1 Enable ACS as described in the Enabling Enhanced Charging, on page 16 section.

Step 2 Set local system management parameters as described in the Modifying the Local Context, on page 16 section.

Enabling Enhanced Charging

Use the following configuration example to enable enhanced charging on the system:

```
configure
   require active-charging
   end
```

Important After you configure the **require active-charging** command, you must save the configuration and then reload the chassis for the command to take effect. For information on saving the configuration file and reloading the chassis, refer to the *System Administration Guide* for your deployment.

Modifying the Local Context

Use the following configuration example to set the default subscriber and AAA group in the local context:

```
configure
   context local
      interface <interface>
      ip address <ipv4/ipv6_address/mask>
      ip arp timeout <timeout>
      exit
   server ftpd
   exit
   server sshd
      subsystem sftp
      exit
   server telnetd
      exit
   subscriber default
      exit
   administrator <security_admin> encrypted password <password> ftp
   aaa group default
```

```
    exit
  gtpp group default
    exit
  ip route <route> SPIO1
  exit
  port ethernet <slot/port>
    no shutdown
    bind interface <interface> local
  exit
snmp engine-id local <id_number>
end
```

ADC Configuration

To configure ADC with ACS:

Step 1 Create the ACS service as described in the Creating the Active Charging Service, on page 17 section.

Step 2 Configure ADC rules as described in the Configuring ADC Rules section.

Step 3 Configure P2P protocol groups as described in the Configuring P2P Protocol Groups, on page 19 section.

Step 4 Configure behavioral traffic as described in the Configuring Behavioral Detection, on page 20 section.

Step 5 Configure the P2P Advertisement server correlation group as described in the Configuring P2P Advertisement server, on page 21 section.

Step 6 Configure SSL renegotiation as described in the Configuring SSL Renegotiation section.

Step 7 Configure analyzers as described in the Configuring Analyzers section.

Step 8 Configure the charging action as described in the Configuring the Charging Action, on page 22 section.

Step 9 Configure the rulebase as described in the Configuring the Rulebase section.

Step 10 *Optional:* Set EDR formats as described in the Setting EDR Formats section.

Step 11 Configure the IP protocol and server port mapping for EDRs as described in the Configuring IP Protocol and Server port mapping, on page 24 section.

Step 12 Configure the P2P events for generation of EDRs as described in the Configuring EDR for P2P Events, on page 25 section.

Important Commands used in the configuration examples in this section provide base functionality to the extent that the most common or likely commands and/or keyword options are presented. In many cases, other optional commands and/or keyword options are available. Refer to the *Command Line Interface Reference* for complete information regarding all commands.

Creating the Active Charging Service

Use the following configuration example to create the ACS service:

```
configure
 active-charging service <acs_service_name> [ -noconfirm ]
 end
```

Important The **p2p-dynamic-rules protocol all** CLI command under Active Charging service is deprecated, and not supported in 12.0 and later releases.

Configuring ADC Rules

Use the following configuration example to set the P2P detection protocols in the ACS and the rule definitions for each P2P protocol.

Important The list of P2P protocols will be populated based on the currently installed plugin.

```
configure
 active-charging service <acs_service_name>
 p2p-detection protocol all
 ruledef <charging_ruledef_actionvoip>
 p2p protocol = actionvoip
 exit
 ruledef <charging_ruledef_facebook>
 p2p protocol = facebook
 exit
 ruledef <charging_ruledef_jabber>
 p2p protocol = jabber
 exit
 ruledef <charging_ruledef_skype>
 p2p protocol = skype
 exit
# Configuration example to report audio, file transfer, instant messaging, video, voipout
and unclassified components:
 ruledef <charging_ruledef_<protocol>_audio>
 p2p protocol = <protocol>
 p2p traffic-type = audio
 exit
 ruledef <charging_ruledef_<protocol>_ft>
 p2p protocol = <protocol>
 p2p traffic-type = file-transfer
 exit
 ruledef <charging_ruledef_<protocol>_im>
 p2p protocol = <protocol>
 p2p traffic-type = im
 exit
 ruledef <charging_ruledef_<protocol>_video>
 p2p protocol = <protocol>
 p2p traffic-type = video
 exit
 ruledef <charging_ruledef_<protocol>_voipout>
 p2p protocol = <protocol>
 p2p traffic-type = voipout
 exit
```

```
ruledef <charging_ruledef_<protocol>_unclassified>
p2p protocol = <protocol>
p2p traffic-type = unclassified
exit
end
```

Configuring ADC Ruledef Types

Use the following configuration to configure ADC ruledefs in support of the ADC over Gx feature:

```
configure
    active-charging service service_name
         ruledef adc_rule_type1
              p2p protocol = protocol_name
              p2p protocol-group = protocol_group
              p2p behavioral = behavioral_list
              multi-line-or all-lines
              exit
         ruledef adc_rule_type2
              p2p protocol = protocol_name
              p2p traffic-type = traffic-type
              exit
         ruledef adc_rule_type2
              p2p any-match = TRUE
              exit
```

Sample Ruledef Configuration for Windows Updates

Use the following ruledef CDP configuration for detecting Windows Updates:

```
Ruledef Name: windowsupdate
tls sni ends-with windowsupdate.com
tls sni ends-with update.microsoft.com
http host ends-with windowsupdate.com
http host ends-with update.microsoft.com
tls set-app-proto windowsupdate
Rule Application Type: Charging
Copy Packet to Log: Disabled
Tethered Flow Check: Disabled
Multi-line OR: All Lines
```

Configuring P2P Protocol Groups

Use the following configuration example to configure P2P protocol groups:

```
configure
      active-charging service <acs_service_name>
            ruledef <ruledef_name>
                  [ no ] p2p protocol-group <operator> <group_list>
                  end
```

Notes:

<*group_list*> must be one of the following:

- **anonymous-access**
- **business**
- **communicator**
- **cloud**
- **e-mail**
- **e-news**
- **e-store**
- **internet-privacy**
- **filesharing**
- **gaming**
- **p2p-anon-filesharing**
- **p2p-filesharing**
- **remote-control**
- **social-nw-gaming**
- **social-nw-generic**
- **social-nw-videoconf**
- **standard**
- **streaming**
- **untagged**

For more information, refer to the *ACS Ruledef Configuration Mode Commands* chapter of the *Command Line Interface Reference*.

Configuring Behavioral Detection

Use the following configuration example to configure behavioral detection of unidentified traffic:

```
configure
      active-charging service <acs_service_name>
            [ no ] p2p-detection behavioral <behavioral_list>
            end
```

Use the following to define rule expressions to match behavioral detection type — P2P, Video, VoIP, Behavioral Upload or Behavioral Download.

```
configure
      active-charging service <acs_service_name>
            ruledef <ruledef_name>
                  [ no ] p2p behavioral <operator> <behavioral_list>
                  end
```

Notes:

- Here the <*behavioral_list*> is the list of supported behavioral detection logic populated from the currently loaded P2P plugin. The supported behavioral list is:
  - **download**: Detects unknown flows which are data download using behavioral analysis
  - **p2p**: Detects P2P and file sharing protocols using behavioral analysis
  - **upload**: Detects unknown flows which are data upload using behavioral analysis
  - **video**: Detects video flows using behavioral analysis

- **voip**: Detects VoIP (voice and video) protocols using behavioral analysis

Behavioral P2P, behavioral video and behavioral VoIP are meant for zero day detection of P2P/file sharing protocols, video traffic and VoIP traffic respectively. Behavioral Upload/Download must detect flows of non-standard ports that cannot be detected by ECS. This feature is meant only for statistical purposes (not for charging purposes). For more information, refer to the *ACS Configuration Mode Commands* and *ACS Ruledef Configuration Mode Commands* chapters of the *Command Line Interface Reference*.

Configuring P2P Advertisement server

Use the following configuration to configure the P2P Advertisement server correlation group:

```
configure
   active-charging service acs_service_name
      [ no ] p2p ads-group ads_group_name
         [ no ] ad-source operator http_host_name/ssl_server_name
         [ no ] map-to-application { p2p_list } +
         end
```

Notes:

- On entering this command, the CLI prompt changes to the P2P Advertisement Server Group Configuration Mode:

  [context_name]hostname(config-acs-p2p-ads)#

- *ads_group_name* must be an alphanumeric string of 1 through 63 characters.
- The following two commands are supported in the new P2P Advertisement Server Group Configuration Mode.
  - **ad-source** *operator http_host_name/ssl_server_name*: Configures the P2P Advertisement source that can be a HTTP host or SSL server. SSL supports the Server Name indication (SNI) field.

    operator can be “=”, “contains”, “ends-with” or “starts-with”.

    http_host_name/ssl_server_name must be an alphanumeric string of 1 through 127 characters.

  - **map-to-application** *p2p_list*: Configures the P2P advertisement application that will map the advertisement group to the corresponding application/protocol. *p2p_list* is the list of protocols/applications supported in the P2P plugin. The maximum number of map-to-application rule lines that can be configured is equal to the number of the applications present in *p2p_list*.
- The existing analyzer statistics and EDR will accumulate P2P related statistics based on the ads-group configuration. Bulk statistics will accumulate "ads" subtype statistics for the configured protocol in p2p-ads-group.
- The existing ruledef configuration will be used to configure any charging action.

For more information, refer to the *ACS Configuration Mode Commands* and *P2P Advertisement Server Group Configuration Mode Commands* chapters of the Command Line Interface Reference.

Configuring SSL Renegotiation

Use the following configuration example to configure SSL renegotiation:

```
configure
      active-charging service <acsservice_name>
            [ no ] p2p-detection attribute { <attribute_list> [
<sub_attribute_name> <sub_attribute_value> ] }
            end
```

Notes:

- Here the <*attribute_list*> is the list of configurable P2P detection attributes populated from the currently loaded P2P plugin.

  Supported attribute: **ssl-renegotiation**

- <*sub_attribute_name*> is the list of configurable P2P detection sub-attributes related to the attribute selected from the attribute list. This list is populated from the currently loaded P2P plugin.

  Supported sub-attributes if selected attribute is **ssl-renegotiation**:

  - **max-entry-per-sessmgr**
  - **id-reduce-factor**

- <*sub_attribute_value*> is the value of the selected sub-attribute. If sub-attribute is not specified, the default value set in the P2P plugin will be used.

For more information, refer to the *ACS Configuration Mode Commands* chapter of the *Command Line Interface Reference*.

Configuring Analyzers

Use the following configuration example to configure analyzers for ECS analysis:

```
configure
      require active-charging
      active-charging service <acs_service_name>
            [ no ] p2p-detection ecs-analysis { all | ftp | http | rtsp
| sip }
            end
```

Important After you configure **require active-charging** and **active-charging service**<*acs_service_name*> commands, you must save the configuration and then reload the chassis for the command to take effect. For information on saving the configuration file and reloading the chassis, refer to the *System Administration Guide* for your deployment.

For more information, refer to the *ACS Configuration Mode Commands* chapter of the *Command Line Interface Reference*.

Configuring the Charging Action

Use the following configuration example to configure the charging actions:

```
configure
  active-charging service <acs_service_name>
    charging-action <charging_action_name1>
```

```
      flow limit-for-bandwidth direction downlink peak-data-rate 4000
peak-burst-size 1024 violate-action discard committed-data-rate 3200
committed-burst-size 512 exceed-action discard
      exit
    charging-action <charging_action_name2>
      content-id 1
      exit
    charging-action <charging_action_name3>
      flow action terminate-flow
      end
```

Configuring the Rulebase

Use the following configuration example to configure the rulebases for P2P.

```
configure
 active-charging service <acs_service_name>
 rulebase <rulebase_name>
 action priority action_priority { [ dynamic-only { adc [ mute ] } |
static-and-dynamic | timedef timedef_name ] { group-of-ruledefs
ruledefs_group_name | ruledef ruledef_name } charging-action charging_action_name [
 monitoring-key monitoring_key ] [ description description ] }
# Configuration
example to detect P2P applications configured for the Active Charging
Service:
 action priority <priority> ruledef <charging_ruledef_actionvoip> charging-action
  <charging_action_name>
 action priority <priority> ruledef <charging_ruledef_facebook> charging-action
<charging_action_name>
 action priority <priority> ruledef <charging_ruledef_jabber> charging-action
<charging_action_name>
 action priority <priority> ruledef <charging_ruledef_skype> charging-action
<charging_action_name>
# Configuration example to report audio, file transfer, instant messaging, video, voipout
and unclassified components:
 action priority <priority> ruledef <charging_ruledef_protocol_audio>
charging-action <charging_action_name>
 action priority <priority> ruledef <charging_ruledef_protocol_ft> charging-action
  <charging_action_name>
 action priority <priority> ruledef <charging_ruledef_protocol_im> charging-action
  <charging_action_name>
 action priority <priority> ruledef <charging_ruledef_protocol_video>
charging-action <charging_action_name>
 action priority <priority> ruledef <charging_ruledef_protocol_voipout>
charging-action <charging_action_name>
 action priority <priority> ruledef <charging_ruledef_protocol_unclassified>
charging-action <charging_action_name>
 end
 end
```

Notes:

- The **adc** keyword option specifies the ruledef to be given as ADC rule. This predefined rule can be activated from PCRF/Gx. This can be configured only with the **dynamic-only** keyword and optional along with ruledef. Group-of-ruledefs is not supported in this release.

- The **mute** keyword is optional and can be configured only with the **adc** keyword. This keyword option will disable ADC application reporting to PCRF, that is, will mute the Application Start and Application Stop notifications to PCRF/Gx. Detection of protocols in the rule will still happen. Whenever the application traffic matches the specified ruledef for the first time in that flow, it is considered as Application Start. At the end of flow, it is considered as Application Stop.

Setting EDR Formats

ECS generates postpaid charging data files which can be retrieved from the system periodically and used as input to a billing mediation system for post-processing. Event Detail Records (EDRs) are generated according to action statements in rule commands.

Up to 32 different EDR schema types may be specified, each composed of up to 32 fields or analyzer parameter names. The records are written at the time of each rule event in a comma-separated (CSV) format. This configuration aids in capturing the detected P2P protocol data in the EDR.

Use the following example to set the EDR configuration:

```
configure
  active-charging service <ecs_service>
    edr-format <edr_flow_format>
      rule-variable traffic-type priority <priority>

      rule-variable p2p duration priority <priority>
      attribute sn-start-time format seconds priority <priority>
      attribute sn-end-time format seconds priority <priority>
      attribute radius-calling-station-id priority <priority>
      rule-variable ip server-ip-address priority <priority>
      attribute sn-server-port priority <priority>
      attribute sn-port-service-name priority <priority>
      attribute sn-app-protocol priority <priority>
      attribute sn-parent-protocol priority <priority>
      rule-variable ip protocol priority <priority>
      attribute sn-ip-protocol-name priority <priority>
      rule-variable p2p protocol priority <priority>
      rule-variable p2p protocol-group priority <priority>
      attribute sn-volume-amt ip bytes uplink priority <priority>
      attribute sn-volume-amt ip bytes downlink priority <priority>
      attribute sn-volume-amt ip pkts uplink priority <priority>
      attribute sn-volume-amt ip pkts downlink priority <priority>
      rule-variable bearer 3gpp charging-id priority <priority>
      rule-variable bearer 3gpp imei priority <priority>
      rule-variable bearer 3gpp rat-type priority <priority>
      rule-variable bearer 3gpp user-location-information priority
<priority>
      end
```

For information on EDR format configuration and rule variables, refer to the *EDR Format Configuration Mode Commands* chapter of the *Command Line Interface Reference Guide*.

Configuring IP Protocol and Server port mapping

Use the following configuration to enable IP protocol and server port mapping for EDRs:

```
configure
 active-charging service <acs_service_name>
 [ default | no ] edr-ipproto-port-map
 end
```

For information, refer to the *ACS Configuration Mode Commands* chapter of the *Command Line Interface Reference*.

Configuring EDR for P2P Events

ADC supports sub-protocol detection for certain applications like Skype, Yahoo, MSN, GTalk, etc. Along with sub-protocol tracking, the application detection logic can track start and end duration of audio/video flow. This audio-end/video-end tracking heavily depends on the audio/video patterns used by the application detection logic. For example, in the case of audio flow start for a sub-protocol tracking, application detection logic begins tracking the start time and if that flow toggles to another sub-protocol that is video or unclassified, the end time for that sub-protocol is set and the EDR for that flow is dumped. Since certain applications use simultaneous flows for audio/video, parallel flow is tracked for all such flows instead of tracking separated audio/video flows.

Use the following configuration to generate EDRs for P2P events. This command is associated with the Dynamic Software Upgrade process.

```
configure
   active-charging service <acs_service_name>
      rulebase <rulebase_name>
         edr p2p <p2p_event_list> [ charging-edr <charging_edr_format_name> |
edr-format <edr_format_name> | reporting-edr <reporting_edr_format_name> ] +
         end
```

Important When the 1.97.357 ADC Plugin is used, Voice Duration details in EDR logs are missing. This has been fixed in plugins later than 1.97.357.

Notes:

- The plugin supports only the "audio-end" and "video-end" events. The P2P event list can be any P2P event that is supported by the plugin.
- The EDR generated for audio-end/video-end must not be used for flow analysis, and must be used only for audio/video duration analysis.
- For audio/video duration analysis (for example, VCD reports), the "sn-closure-reason" field must be checked to identify the reason for closure and the appropriate EDR chosen to generate the reports. The "sn-closure-reason" field will be set to **14** for capturing audio-end and video-end generated EDRs. For example, in case of audio-end events, all VoIP call related statistics such as VoIP duration and bytes/packets are captured along with traffic-type field set as "audio" and sn-closure-reason set as "14".

For information, refer to the *ACS Rulebase Configuration Mode Commands* chapter of the *Command Line Interface Reference*.

Gathering ADC Statistics

In the following table, the first column lists what statistics to gather, the second column lists an action to perform, and the third column describes what information is displayed or what information to look for in the resulting output.

Table 1: Gathering Statistics

Statistics Wanted	Action to Perform	Information to Look For
Analyzer statistics	**show active-charging analyzer statistics name p2p verbose**	The output of this command displays the analyzer statistics if a P2P analyzer is used. Since the analyzer statistics are not bound to any service, the traffic information per gateway can be obtained.
Ruledef statistics	**show active-charging ruledef statistics name** <*name*>	The output of this command displays the Ruledef statistics including the packet count, byte count and hits.
P2P flow statistics	**show active-charging flows type p2p traffic-type audio** **show active-charging flows type p2p traffic-type file-transfer** **show active-charging flows type p2p traffic-type im** **show active-charging flows type p2p traffic-type video** **show active-charging flows type p2p traffic-type voipout** **show active-charging flows type p2p traffic-type unclassified**	The output of this command displays the number of P2P audio, file transfer, instant messaging, video, voipout and unclassified flows.
Charging Action information	**show active-charging charging-action statistics**	The output of this command displays the charging action information and corresponding statistics configured in the active charging service.
Transmit and Receive data	**show active-charging sessions tx-data** <*operator*> <*bytes*> **show active-charging sessions rx-data** <*operator*> <*bytes*>	The output of the command displays the information for sessions that have received or transmitted data which matches the criteria.

Statistics Wanted	Action to Perform	Information to Look For
Sessions using specific protocol	**show active-charging sessions type p2p application** <*protocol*> **show active-charging sessions full all**	The output of this command displays information for the sessions using the specified protocol.
Total and current P2P flows and P2P audio, file-transfer, im, video, voipout, or unclassified flows	**show active-charging subsystem all**	The output of this command displays total and current P2P flows and P2P audio/file-transfer/instant messaging/video/voipout/unclassified flow statistics, and total number of subscribers.
Voice Statistics	**show active-charging analyzer statistics name p2p application [actionvoip | facetime | gtalk | icall | jumblo | magicjack | msn | oscar | plingm | rynga | skype | smartvoip | talkatone | voipdiscount | vopium | yahoo] verbose**	The output of this command displays the voice and non-voice analyzer statistics for voice supported protocols.
P2P Protocol Group Statistics	**show active-charging analyzer statistics name p2p protocol-group wide all verbose**	The output of this command displays the P2P protocol group statistics if a P2P analyzer is used.
Subscriber Readdress Statistics	**show subscribers callid <callid> adc readdress statistics**	In support of the ADC over Gx feature, the output of this command displays readdress statistics at subscriber level for a given call ID.

Supported Bulk Statistics

ADC bulk statisticsa are available as part of the P2P schema. If detection of a specific P2P protocol is enabled, bulk statistics for that protocol will be automatically generated using the Dynamic Software Upgrade plugin installed on the chassis. In the case of protocols that support sub-classification (audio/video/unclassified), the bulk statistics will be generated for each of the supported sub-classifications per protocol and also the corresponding cumulative count.

For information on ADC bulk statistics and bulk statistics configuration and collection, refer to the *Bulk Statistics Configuration Mode Commands* chapter of the *Command Line Interface Reference*, and the *Statistics and Counters Reference*.

P2P Reports

The P2P reports provide details of the bandwidth consumption of P2P traffic over time. These reports are used to analyze network performance, identify the customer trends, network usage patterns, and network categorization.

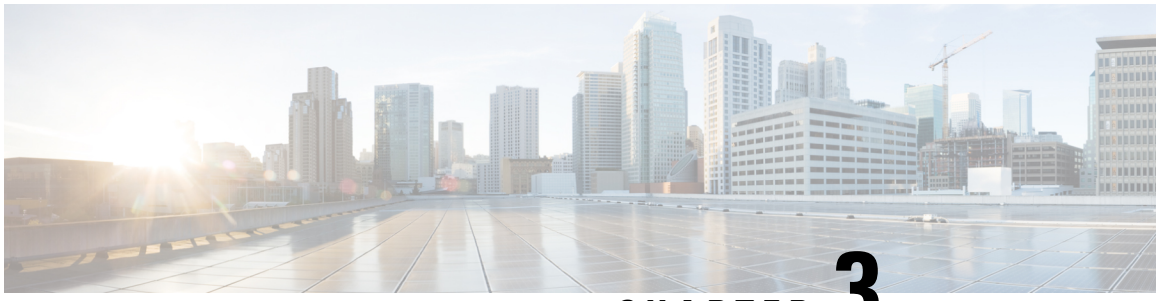
Important In 9.0 and earlier releases, the P2P reporting functionality was available in the Web Element Manager software. For more information, refer to the *Web Element Manager Online Help* documentation.

Important In 10.0 and later releases, the P2P reporting functionality is supported in Mobility Unified Reporting (MUR) / Mobility Unified Reporting and Analytics (MURAL) system. For more information on MUR, refer to the *Mobility Unified Reporting Online Help* documentation. For more information on MURAL support, refer to the *MURAL Installation and Administration Guide* and *MURAL User Guide*.

The following bandwidth usage reports are supported:

- Cumulative analyzer count - representing the total bandwidth consumed by the P2P traffic in bits/sec. Daily, monthly or yearly reports are supported.
- Total bandwidth consumed P2P traffic against other protocols like HTTP, RTSP, etc. Daily or monthly reports are supported.
- Per protocol type - total bandwidth consumed by the individual P2P protocol traffic in packets/sec or bytes/sec plotted against time range or date range. Daily reports are supported. The graph uses separate colors to differentiate among the multiple protocol types.
- The number of active users per application for specified date/time range. Daily reports are supported.
- Analysis of the percentage of total bandwidth consumed by P2P traffic from the total subscriber traffic. Weekly reports are supported.

Important For additional information about viewing reports, refer to the *Web Element Manager Online Help* documentation.



CHAPTER 3

App-based Tethering Detection

Feature Information

Summary Data

Status	New Feature
Introduced-In Release	21.2
Modified-In Release(s)	Not Applicable
Applicable Product(s)	P-GW
Applicable Platform(s)	ASR 5500
Default Setting	Disabled
Related CDETS ID(s)	CSCvd65410
Related Changes in This Release	Not Applicable
Related Documentation	ADC Administration Guide Command Line Interface Reference

Revision History

Important Revision history details are not provided for features introduced before release 21.2.

Revision Details	Release	Release Date
New in this release.	21.2	April 27, 2017

Feature Description

The tethering of IPv4 and IPv6 traffic has increased, and the native device recognition techniques are less viable with the advent of proxy tethering Apps on smartphones.

A new App-based Tethering Detection is introduced with this release. This solution interwork with other existing Tethering Detection techniques.

How It Works

This solution is built around the existing ADC plugins for App identifications. Tethering specific patterns are added on top of recognized App plugins. These plugins successively return if the App flow is tethered or not.

Important With this release, App-based Tethering Detection is introduced only for Netflix and YouTube.

Licensing

This feature requires both ADC, and Tethering Detection License. Contact your Cisco account representative for detailed information on specific licensing requirements. For information on installing and verifying licenses, refer to the *Managing License Keys* section of the *Software Management Operations* chapter in the *System Administration Guide*.

Configuring App-based Tethering Detection

This section describes how to enable support for App-based Tethering Detection.

Enabling App-based Tethering Detection at Rulebase Level

Use the following commands to enable App-based Tethering Detection for ADC traffic under ACS Rulebase Configuration Mode:

```
configure
  active-charging service service_name
    rulebase rulebase_name
      tethering-detection application
      exit
    exit
  exit
```

Notes:

- The **default tethering-detection** command configures its default setting.

  Default: By default, the Tethering Detection feature is disabled. When enabled, unless a specific database is specified to be used, tethering detection will make use of both the databases by default.

- If previously configured, use the **no tethering-detection** command to remove the tethering detection configuration from the rulebase.

Enabling App-based Tethering Detection at Ruledef Level

Use the following commands to enable App-based Tethering Detection for ADC traffic under ACS Ruledef Configuration Mode:

```
configure
  active-charging service service_name
    ruledef ruledef_name
      tethering-detection application flow-tethered
      exit
    exit
  exit
```

Notes:

- If previously configured, use the **no tethering-detection** command to remove the tethering detection configuration from the ruledef.

Enabling App-based Tethering Detection at Rule-variable

Use the following commands to enable App-based Tethering Detection field in EDRs under EDR Format Configuration Mode:

```
configure
  active-charging service service_name
    edr-format format_name
      rule-variable flow tethered-application priority priority
      exit
    exit
  exit
```

Notes:

- *flow tethered-application*: The *flow* specifies Flow related fields. *tethered-application* specifies application based tethering detected on flow.
- **priority** *priority*: Specifies the CSV position of the field (protocol rule) in the EDR. *priority* must be an integer from 1 through 65535.
- If previously configured, use the **no rule-variable** *rule_variable* **[priority** *priority* **]** command to remove the specified rule variable configuration.

Monitoring and Troubleshooting the App-based Tethering Detection

This section provides information regarding commands available to monitor and troubleshoot the App-based Tethering Detection.

Show Commands and Outputs

This section provides information regarding show commands and their outputs in support of this enhancement.

show active-charging tethering-detection statistics

The following fields are available in the output of this show command in support of this enhancement:

```
Current Tethered Subscribers:  0
Total flows scanned:           0
Total Tethered flows detected: 0
Total Tethered flows recovered:     0
Total flows bypassed for scanning : 0

Tethering Detection Statistics (os-ua):
  TAC ID lookups:              0
  TAC ID matches:              0
  OS signature lookups:        0
  OS signature matches:        0
  IPv6 OS signature lookups:   0
  IPv6 OS signature matches:   0
  UA signature lookups:        0
  UA signature matches:        0
  Total flows scanned:         0
  Tethered flows detected:     0
  Non-tethered flows detected: 0
  Tethered Uplink Packets:     0
  Tethered Downlink Packets:   0
  Current tethering-detected indications sent:   0
  Total tethering-detected indications sent:     0

Tethering Detection Statistics (ip-ttl):
  Total flows scanned:                 0
  Tethered flows detected:             0
  Tethered uplink packets:             0
  Tethered downlink packets:           0

Tethering Detection Statistics (DNS Based):
  Total flows scanned:                 0
  Tethered flows detected:             0
  Tethered uplink packets:             0
  Tethered downlink packets:           0

Tethering Detection Statistics (Application):
  Total flows scanned:                 0
  Tethered flows detected:             0
  Tethered uplink packets:             0
  Tethered downlink packets:           0

Change Statistics for Multiple SYN in Flow:
```

```
  Tethered to Non-Tethered:                          0
  Non-Tethered to Tethered:                          0
  Tethered to Tethered:                              0
  Non-Tethered to Non-Tethered:                      0
```

show active-charging rulebase statistics

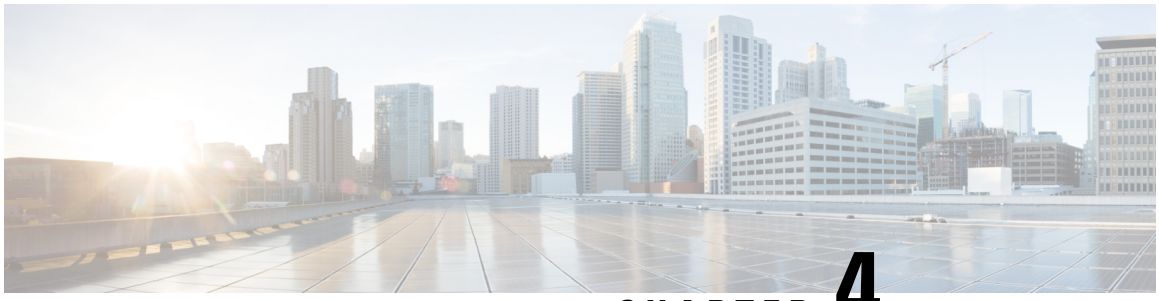
The following fields are available in the output of this show command in support of this enhancement:

```
Tethering Detection:
     TAC ID lookups:                                 0
     TAC ID matches:                                 0
     OS signature lookups:                           0
     OS signature matches:                           0
     IPv6 OS signature lookups:                      0
     IPv6 OS signature matches:                      0
     UA signature lookups:                           0
     UA signature matches:                           0
     Total flows scanned:                            0
     Tethered flows detected:                        0
     Tethered uplink packets:                        0
     Tethered downlink packets:                      0

     Tethering Detection (ip-ttl):
     Total flows scanned:                            0
     Tethered flows detected:                        0
     Tethered uplink packets:                        0
     Tethered downlink packets:                      0

     Tethering Detection (DNS Based):
     Total flows scanned:                            0
     Tethered flows detected:                        0
     Tethered uplink packets:                        0
     Tethered downlink packets:                      0

     Tethering Detection (Application):
     Total flows scanned:                            0
     Tethered flows detected:                        0
     Tethered uplink packets:                        0
     Tethered downlink packets:                      0
```



CHAPTER 4

Reporting SSL Parameters in EDR

This chapter describes the following topics:

Feature Summary and Revision History

Summary Data

Applicable Product(s) or Functional Area	All Products supporting ADC
Applicable Platform(s)	• ASR 5500 • VPC-DI • VPC-SI
Feature Default	Enabled - Always-on
Related Changes in This Release	Not Applicable
Related Documentation	*ADC Administration Guide*

Revision History

Revision Details	Release
First introduced.	21.5

Feature Description

The ADC (P2P) engine detects encrypted traffic using Server Name Indication (SNI)/ Canonical Name (CNAME) field (signatures) of SSL flow in Client and Server Hello packets. To mark the flow as SSL, both Client and Hello packets must be parsed without any error. There could be several error conditions due to which ADC SSL decoder could fail, such as failure to receive Server Hello, invalid Type Value Length (TLV), invalid header length, absence of a certificate, and so on. When any of these error conditions are seen, the flow is marked as "p2p-unknown" instead of SSL although the packets are exchanged on TCP Port 443. As a result, the flow is matched to the "ip any-match" rule instead of an SSL rule.

With this feature, these failure reasons are reported in EDR for better debugging purposes and to know why a flow is not marked as SSL. Subsequently, marking a flow as an application traffic based on SNI has a limitation of users spoofing the SNI field to gain advantage of the various service provider monetary plans. SSL makes use of certificates to authenticate the endpoints. As such, to overcome SNI spoofing, ADC parses the various information in the SSL Server Hello packet and reports the same in EDR for debugging.

How It Works

This section provides a brief overview of how this feature works.

- ADC stores the reason for SSL decode failure.
- ADC stores the validity of SSL certificate.
  - Has the certificate expired (> not-after).
  - Is the certificate still valid (between not-before and not-after).
  - Can the certificate be used (< not-before).

    Currently, ADC decodes ‘not-after’ and ‘not-before’ and stores them internally. As part of this feature, the ‘not-after’ and ‘not-before’ is parsed with current system time and report values 0, 1, 2 in EDR.

- Support is added to configure the required SSL certificate parameters in EDR.
- Support is added to report the certificate information in charging EDR.
- Support is added to generate charging EDR for that flow and report the parsed information.
- Boxer is compatible with a plugin with and without changes.
- If the value of attributes; SSL ISSUER CNAME, SSL ISSUER ORG or SSL SUBJECT ORG contains “,”, the Boxer converts the same to “_” before printing in EDR.

EDR

The SSL parameters issued from plugin can be configured as EDR fields. The particular values corresponding to different EDR fields is populated, if present in the flow.

Following SSL parameters are identified to be reported in EDR:

- **SSL decode failure**: A flow can fail to be detected as SSL due to one of the following reasons:

1. IPOQUE_SSL_CLIENT_HELLO_DECODE_SUCCESS,
2. IPOQUE_SSL_SERVER_HELLO_DECODE_SUCCESS,
3. IPOQUE_SSL_CLIENT_HELLO_INVALID_EXT_LEN,
4. IPOQUE_SSL_SERVER_HELLO_INVALID_EXT_LEN,
5. IPOQUE_SSL_SERVER_HELLO_NO_CERTIFICATE,
6. IPOQUE_SSL_SERVER_HELLO_INVALID_SNO_LEN,
7. IPOQUE_SSL_SERVER_HELLO_INVALID_SIGNATURE_LEN,
8. IPOQUE_SSL_SERVER_HELLO_ISSUER_NOT_FOUND,
9. IPOQUE_SSL_SERVER_HELLO_INVALID_VALIDITY,
10. IPOQUE_SSL_SERVER_HELLO_INVALID_VALIDITY_NOT_BEFORE,
11. IPOQUE_SSL_SERVER_HELLO_INVALID_VALIDITY_NOT_AFTER,
12. IPOQUE_SSL_SERVER_HELLO_INVALID_SUBJECT_LEN

EDR will contain a value of 1-12 corresponding to above reasons in the same order with IPOQUE_SSL_CLIENT_HELLO_DECODE_SUCCESS being 1 and IPOQUE_SSL_SERVER_HELLO_INVALID_SUBJECT_LEN being 12.

Following are the conditions when above failure reasons will be reported:

- Client Hello Success – Client Hello decoded successfully but further packets are not received in flow (absence of Server Hello).
- Server Hello Success – SSL flow decoded successfully without any issues, both Client and Server Hello packets.
- Invalid Client/Server Hello Extension Length – When length of any of the TLVs in Client/Server Hello is 0 (zero) or more than packet length respectively.
- Server Hello No certificate – Absence of any certificates in Server Hello.
- Server Hello Invalid Subject/SNO/Signature Len – When length of Subject/Serial Number/Signature TLVs in Server Hello is 0 (zero) or more than packet length respectively.
- Server Hello Issuer Not Found – Absence of Issuer certificate in Server Hello.
- Server Hello Invalid Validity/Validity Not Before/Validity Not After – Absence of these fields in Server Hello.
- Subject Organization Name – Name of the Organization/Company to which the SSL certificate is issued.
- Issuer CNAME – Canonical Name of the Organization/Company issuing the certificate
- Issuer Organization Name – Name of the Organization/Company issuing the SSL certificate
- SSL Certificate Validity:
  - EDR value of 1 – Current date is less than Certificate not before date
  - EDR value of 2 – Current date is greater than Certificate not after date

- EDR value of 3 – Current date is within Certificate not before and not after dates

  Issuer provides a certificate subjected for a particular duration of time, after which the same must be renewed or the certificate is deemed invalid. This field states whether the certificate exchanged in Server Hello is currently valid or expired.

Configuring SSL Parameters in EDR

This section provides information on CLI commands available in support of this feature.

Enabling SSL Parameters

Use the following configuration to enable SSL Parameters under EDR Format Configuration Mode:

```
active-charging service service_name
  edr-format format_name
    rule-variable p2p ssl-params { cert-issuer-cname | cert-subject-oname
 | cert-issuer-oname | cert-validity | ssl-decode-failure } priority
priority
    end
```

NOTES:

- **ssl-params**: Specifies the SSL flow parameters.
- **cert-issuer-cname**: Specifies the SSL Certificate Issuer CName.
- **cert-subject-oname**: Specifies the SSL Certificate Subject Organization Name.
- **cert-issuer-oname**: Specifies the SSL Certificate Issuer Organization Name.
- **cert-validity**: Specifies the validity of SSL Certificate.
- **ssl-decode-failure**: Specifies the reason for SSL Decode failure.

Monitoring and Troubleshooting

This section provides information about show CLI commands and/or their outputs in support of this feature.

Show Commands and/or Outputs

show active-charging edr-format all

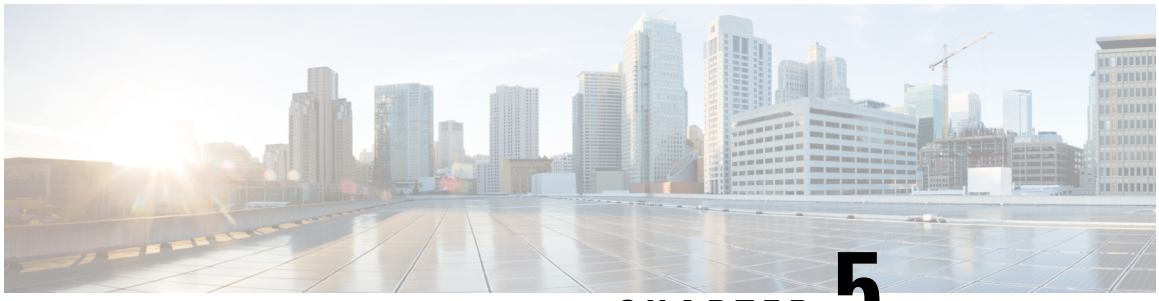
The output of this CLI command has been enhanced to display the new SSL parameters. Following is a sample output:

```
show active-charging edr-format all
 Service Name: service_1
  Edr Format  Name: edr_1
      rule-variable p2p ssl-params cert-subject-oname priority 1
```

show configuration active-charging service all

The output of this CLI command has been enhanced to display the new SSL parameters. Following is a sample output:

```
show configuration active-charging service all
config
  active-charging service service_1
    no ng-ecs-enabled
    edr-format edr_1
      rule-variable p2p ssl-params cert-subject-oname priority 1
    exit
    rulebase default
      no tcp check-window-size
    exit
    policy-control burst-size auto-readjust duration 5
  exit
end
```



CHAPTER 5

Support for SNI Detection

This chapter describes the Server Name Indication (SNI) Detection feature in ADC, and provides detailed information on the following topics:

Feature Description

Server Name Indication (SNI) is an extension of the Transport Layer Security (TLS) protocol that allows multiple secure (HTTPS) websites (or any other service over TLS) to be served from the same IP address without requiring all those sites to use the same certificate. SNI provides a mechanism for the client to tell the server which hostname it is trying to connect to.

ADC detects encrypted traffic using the SNI field (signatures) of TLS/SSL (Secure Sockets Layer) traffic. These signatures are added along with other detection mechanisms and delivered as a plugin. If there are new SNI fields either in the already detected applications or new applications, then these new fields are added to the plugin and a new version of the plugin is released. This results in frequent releases of plugin versions causing delay in upgrading the new plugin in the network and leading to revenue leak to the operator. Due to increased number of applications moving towards TLS/SSL, an option is provided to configure the SNI in ruledef and classify traffic based on the configured SNI with this release.

Important The SNI Detection feature requires a valid Application Detection and Control license. Contact your Cisco Account representative for more information.

The SNI field in the TLS/SSL handshake is used to determine the type of TLS/SSL flow being setup. SNI rule variable is added as an optional field in EDRs for detection of TLS/SSL flows. When the “tls sni” rule variable is configured and a valid SNI name is detected in the flow, the SNI field is populated in the EDR.

QUIC SNI Detection

The SNI Detection feature is enhanced to support QUIC SNI configuration and classify traffic based on the configured SNI. The CLI commands added to configure the SNI are generic such that other application-identifiers added in the future are taken care of with the Plugin changes only.

Important This feature requires the latest ADC Plugin to be loaded from the adc_v2.x stream along with StarOS changes. The default plugin does not support this feature. Contact your Cisco account representative for more information.

When a QUIC flow is analyzed, the P2P rule match takes place on this flow with the configured SNI. If the flow/packet matches the rule, the custom-defined-protocol (CDP) name specified in the ruledef is taken and the flow is marked as CDP. If no CDP is configured in the rule, then the flow is treated as QUIC flow.

Within the P2P plugin, when a QUIC flow arrives with SNI that is already hardcoded and the same SNI is also configured in ruledef, the ruledef with higher priority takes precedence. Based on rule priority, rule matching is done and CDP statistics get incremented. When the new QUIC flow comes with SNI that is not hardcoded, the flow will be marked as a QUIC flow.

- The **p2p app-identifier** and **p2p set-app-proto** commands are added in the ACS Ruledef Configuration mode to configure QUIC-SNI and TLS-SNI that are dynamically populated from the P2P plugin and match the traffic against it.
- EDR attributes are added to support app-identifiers supplied from plugin. EDR logs the matched CDP in the "p2p-protocol" field for the flow. If QUIC-SNI is configured in the EDR field, then QUIC-SNI string will be populated.
- Analyzer statistics, ruledef statistics, and bulk statistics are updated for newly identified protocols
- Backward compatibility for the TLS-SNI feature is maintained such that this feature works seamlessly with new CLI commands added in this release.

Limitations

The limitations with this feature are listed in this section:

- The **quic-sni** identifier is configured in the EDR with the new plugin. When rolled back to old plugins, the EDR headers will print **p2p-unknown** since the old plugin does not support the configured app-identifier. When upgraded to a new plugin that supports QUIC-SNI identifier, **p2p-unknown** will be updated to display **p2p-quic-sni** in the EDR.
- The help strings related to the new CLI keywords will be updated in a later release.

Relationships to Other Features

This section describes how the SNI Detection feature relates to other ADC features.

- **Analyzer Interworking**: In support of SNI Detection, HTTPS protocol support is added for ECS analysis of all SSL flows as part of the Analyzer Interworking feature. This feature is enabled by default for all analyzers including HTTPS if P2P detection/protocol is enabled.

  The different behaviors when Analyzer Interworking is enabled or disabled is listed in the table below.

Condition	HTTPS Routing Enabled	HTTPS Routing Disabled
With SNI feature in 17.5 and later releases:		

Condition	HTTPS Routing Enabled	HTTPS Routing Disabled
If SSL protocol is enabled, SNI/EDR features will work if any routing rule is configured	• Analyzer Interworking enabled for HTTPS: App-proto = HTTPS(6) and p2p protocol = SSL • Analyzer Interworking disabled for HTTPS: App-proto = P2P(29) and p2p protocol = SSL	App-proto = P2P(29) and p2p protocol = SSL
If SSL protocol is disabled, SNI/EDR features will not work.	App-proto = HTTPS(6) and p2p protocol = Unknown	App-proto = Unknown(0) and p2p protocol = Unknown
Without SNI feature in releases prior to 17.5:		
SNI/EDR features will not work and SSL will not be exposed to ASR 5500.	App-proto = HTTPS(6) and p2p protocol = Unknown	App-proto = Unknown(0) and p2p protocol = Unknown

- **SSL Renegotiation Tracking**: With the SNI Detection feature, the ADC plugin must be able to store Session ID in SSL renegotiated table, map the renegotiated flow to stored Session ID, and map the corresponding CDP name to the flow in the same way as it is done for SSL Renegotiation feature.

For more information on these features, refer to the *ADC Administration Guide*.

Configuring SNI Detection

This section describes how to configure the SNI Detection feature.

Configuring SNI in Ruledef

Use the following configuration to configure the TLS/SSL and QUIC Server Name Indication (SNI) and the corresponding custom defined protocol (CDP).

Important The QUIC SNI Detection feature requires the latest ADC Plugin to be loaded from the adc_v2.x stream along with StarOS changes. The default plugin does not support this feature. Contact your Cisco account representative for more information.

```
configure
  active-charging service service_name
    ruledef ruledef_name
      [ no ] tls { set-app-proto cdp_name_string | sni operator server_name_string
}
      [ no ] p2p app-identifier { quic-sni operator quic_sni_string | tls-cname
operator tls_cname_string | tls-sni operator tls_sni_string }
```

```
[ no ] p2p set-app-proto cdp_name_string
end
```

Notes:

- The **tls set-app-proto** command specifies the name of the custom defined protocol for TLS/SSL flows matching the ruledef.
- The **tls sni** command specifies the TLS/SSL Server Name Indication (SNI) field value in the Client Hello packet.
- In release 20.2, the **p2p app-identifier** command configures "quic-sni", "tls-sni", and "tls-cname" app-identifiers supported by the P2P dynamic library.
- In release 20.2, the **p2p set-app-proto** command configures the custom-defined protocol (CDP) name.
- The following commands must be configured for SNI rules to work:
  - Enable SSL protocol in the Active Charging Service configuration:

    ```
    [local]P2P_SSl(config-acs)# p2p-detection protocol ssl
    ```

    If the **p2p-detection protocol all** CLI command is enabled in the Active Charging Service configuration, then the **ssl** keyword need not be enabled again as it will be already enabled with the **all** keyword.

    The **ssl** protocol is available only in Plugin releases 1.142.526 and later.
  - Enable P2P in the ACS Rulebase configuration:

    ```
    [local]P2P_SSl(config-rule-base)# p2p dynamic-flow-detection
    ```
  - The action priority for SNI ruledef must be configured in the rulebase similar to other ruledefs.

For more information, refer to the *ACS Ruledef Configuration Mode* chapter of the *Command Line Interface Reference*.

Configuring SNI rule variable

Use the following configuration to configure the SNI rule variable for TLS/SSL and QUIC flows in EDR.

```
configure
  active-charging service acs_service_name
    edr-format format_name
      rule-variable tls sni priority priority
      rule-variable p2p app-identifier { quic-sni | tls-cname | tls-sni
} priority priority
      end
```

Notes:

- The **tls sni** command specifies the TLS/SSL SNI rule variable configured for TLS/SSL flows in EDR.
- In release 20.2, the **p2p app-identifier** command specifies the QUIC-SNI, TLS-SNI, and TLS-CNAME application identifiers populated from the plugin.
- **priority** *priority*: Specifies the CSV position of the field (protocol rule) in the EDR. *priority* must be an integer from 1 through 65535.

For more information, refer to the *EDR Format Configuration Mode* chapter of the *Command Line Interface Reference*.

Enabling HTTPS Analyzer Interworking

Use the following configuration to enable or disable ECS analysis for HTTPS analyzer interworking.

```
configure
    active-charging service service_name
         [ no ] p2p-detection ecs-analysis { https }
         end
```

Notes:

- The Active Charging flows will have the app-proto marked as "HTTPS" instead of P2P for all SSL flows if analyzer interworking for HTTPS is enabled.
- The Active Charging flows will have the app-proto marked as "P2P" for all SSL flows if analyzer interworking for HTTPS is disabled.
- By default, analyzer interworking for all analyzers including HTTPS is enabled when P2P detection is enabled.

For more information on the commands listed above, refer to the *Command Line Interface Reference*.

Verifying the SNI Configuration

Executing the following command displays the application/protocol configured in the "set app-proto" string of TLS ruledef:

show active-charging analyzer statistics name cdp [application *app_name* **| instance** *instance_number* **| summary | verbose | wide]**

Executing the following command displays the fields for TLS/SSL SNI and CDP as configured in the TLS ruledef:

show active-charging ruledef name *ruledef_name*

Monitoring and Troubleshooting the SNI Detection

This section provides information on the show commands available to support this feature.

SNI Detection Show Command(s) and/or Outputs

show active-charging analyzer statistics name cdp

The following fields display the analyzer statistics for the custom defined protocol.

- CDP Summary:
  - Total Uplink Bytes
  - Total Downlink Bytes

- Total Uplink Pkts
- Total Downlink Pkts

For description of the fields listed above see, *Statistics and Counters Reference*.

show active-charging flows type cdp

The following fields display the flow-level statistics for the custom defined protocol.

- Session ID
- Flow-ID
- Application Protocol
- Transport Protocol
- Tethered Flow
- Bytes-Up
- Bytes-Down
- Pkts-Up
- Pkts-Down

Bulk Statistics

In support of the SNI detection feature, the "p2p-protocol" field in the P2P schema will display the application protocol configured in the "set app-proto" string of TLS ruledef.

For more information on bulk statistics, see the *P2P Schema Statistics* chapter in the *Statistics and Counters Reference*.