Enhanced Charging Service Overview

This chapter provides an overview of the Enhanced Charging Service (ECS) in-line service, also known as Active Charging Service (ACS).

This chapter covers the following topics:

Introduction

The Enhanced Charging Service (ECS) is an in-line service feature that enables operators to reduce billing-related costs and gives the ability to offer tiered, detailed, and itemized billing to their subscribers. Using shallow and deep packet inspection (DPI), ECS allows operators to charge subscribers based on actual usage, number of bytes, premium services, location, and so on. ECS also generates charging records for postpaid and prepaid billing systems.

The ECS is an enhanced or extended premium service. The *System Administration Guide* provides basic system configuration information, and the product administration guides provide information to configure the core network service functionality. It is recommended that you select the configuration example that best meets your service model, and configure the required elements for that model before using the procedures in this document.

Qualified Platforms

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

License Requirements

The ECS in-line service is a licensed Cisco feature. Separate session and feature licenses 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*.

Basic Features and Functionality

This section describes basic features of the ECS in-line service.

Shallow Packet Inspection

Shallow packet inspection is the examination of the layer 3 (IP header) and layer 4 (for example, UDP or TCP header) information in the user plane packet flow. Shallow packet analyzers typically determine the destination IP address or port number of a terminating proxy.

Deep Packet Inspection

Deep-packet inspection is the examination of layer 7, which contains Uniform Resource Identifier (URI) information. In some cases, layer 3 and 4 analyzers that identify a trigger condition are insufficient for billing purposes, so layer 7 examination is used. Whereas, deep-packet analyzers typically identify the destination of a terminating proxy.

For example, if the Web site "www.companyname.com" corresponds to the IP address 1.1.1.1, and the stock quote page (www.companyname.com/quotes) and the company page (www.companyname.com/business) are chargeable services, while all other pages on this site are free. Because all parts of this Web site correspond to the destination address of 1.1.1.1 and port number 80 (http), determination of chargeable user traffic is possible only through the actual URL (layer 7).

DPI performs packet inspection beyond layer 4 inspection and is typically deployed for:

- Detection of URI information at level 7 (for example, HTTP, WTP, RTSP URLs)
- Identification of true destination in the case of terminating proxies, where shallow packet inspection would only reveal the destination IP address/port number of a terminating proxy such as the OpCo's WAP gateway
- De-encapsulation of nested traffic encapsulation, for example MMS-over-WTP/WSP-over-UDP/IP
- Verification that traffic actually conforms to the protocol the layer 4 port number suggests

Charging Subsystem

ECS has protocol analyzers that examine uplink and downlink traffic. Incoming traffic goes into a protocol analyzer for packet inspection. Routing rules definitions (ruledefs) are applied to determine which packets to inspect. This traffic is then sent to the charging engine where charging rules definitions are applied to perform actions such as block, redirect, or transmit. These analyzers also generate usage records for the billing system.

Traffic Analyzers

Traffic analyzers in ECS are based on configured ruledefs. Ruledefs used for traffic analysis analyze packet flows and create usage records. The usage records are created per content type and forwarded to a prepaid server or to a billing system.

The Traffic Analyzer function can perform shallow (layer 3 and layer 4) and deep (above layer 4) packet inspection of IP packet flows. It is able to correlate all layer 3 packets (and bytes) with higher layer trigger criteria (for example, URL detected in an HTTP header). It also performs stateful packet inspection for complex protocols like FTP, RTSP, and SIP that dynamically open ports for the data path and this way, user plane payload is differentiated into "categories". Traffic analyzers can also detect video streaming over RTSP, and image downloads and MMS over HTTP and differential treatment can be given to the Vcast traffic.

Traffic analyzers work at the application level as well, and perform event-based charging without the interference of the service platforms.

The ECS content analyzers can inspect and maintain state across various protocols at all layers of the OSI stack. ECS supports the following protocols:

- Domain Name System (DNS)
- File Transfer Protocol (FTP)
- Hyper Text Transfer Protocol (HTTP)
- Internet Control Message Protocol (ICMP)
- Internet Control Message Protocol version 6 (ICMPv6)
- Internet Message Access Protocol (IMAP)
- Internet Protocol version 4 (IPv4)
- Internet Protocol version 6 (IPv6)
- Multimedia Messaging Service (MMS)
- Mobile IPv6 (MIPv6)
- Post Office Protocol version 3 (POP3)
- Remote Authentication Dial In User Service (RADIUS)
- RTP Control Protocol/Real-time Transport Control Protocol (RTCP)
- Real-time Transport Protocol (RTP)
- Real Time Streaming Protocol (RTSP)
- Session Description Protocol (SDP)
- Secure-HTTP (S-HTTP)
- Session Initiation Protocol (SIP)
- Simple Mail Transfer Protocol (SMTP)
- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

- WebSocket Protocol
- Wireless Session Protocol (WSP)
- Wireless Transaction Protocol (WTP)

Notes:

- Apart from the above protocols, ECS also supports analysis of downloaded file characteristics (for example, file size, chunks transferred, and so on) from file transfer protocols such as HTTP and FTP.
- Mobile IPv6 (MIPv6) protocol analyzer provides network-based IP mobility management support to a mobile node, without requiring the participation of the mobile node in any IP mobility related signaling. The mobile node may be an IPv4-only node or IPv6-only node.

How ECS Works

This section describes the base components of the ECS solution, and the roles they play.

Content Service Steering

Content Service Steering (CSS) enables directing selective subscriber traffic into the ECS subsystem (in-line services internal to the system) based on the content of the data presented by mobile subscribers.

CSS uses Access Control Lists (ACLs) to redirect selective subscriber traffic flows. ACLs control the flow of packets into and out of the system. ACLs consist of "rules" (ACL rules) or filters that control the action taken on packets matching the filter criteria.

ACLs are configurable on a per-context basis and applies to a subscriber through either a subscriber profile (for PDSN) or an APN profile (for GGSN) in the destination context.

Important For more information on CSS, refer to the *Content Service Steering* chapter of the *System Administration Guide*. For more information on ACLs, refer to the *IP Access Control Lists* chapter of the *System Administration Guide*.

Protocol Analyzer

The Protocol Analyzer is the software stack responsible for analyzing the individual protocol fields and states during packet inspection.

The Protocol Analyzer performs two types of packet inspection:

- **Shallow Packet Inspection**—Inspection of the layer 3 (IP header) and layer 4 (for example, UDP or TCP header) information.
- **Deep Packet Inspection**—Inspection of layer 7 and 7+ information. DPI functionality includes:
  - Detection of Uniform Resource Identifier (URI) information at level 7 (for example, HTTP, WTP, and RTSP URLs)
  - Identification of true destination in the case of terminating proxies, where shallow packet inspection would only reveal the destination IP address/port number of a terminating proxy
  - De-encapsulation of upper layer protocol headers, such as MMS-over-WTP, WSP-over-UDP, and IP-over GPRS

- Verification that traffic actually conforms to the protocol the layer 4 port number suggests

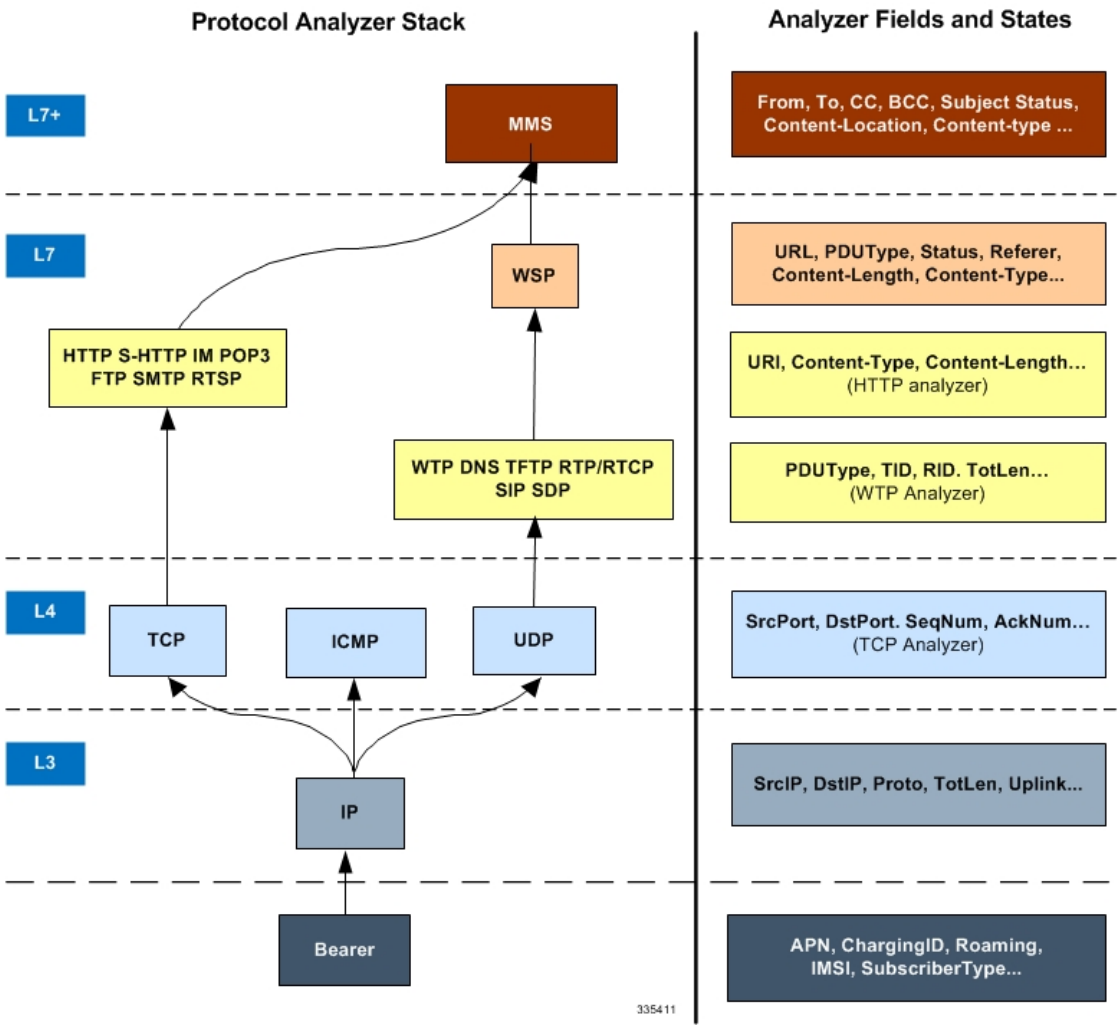
The Protocol Analyzer performs a stateful packet inspection of complex protocols, such as FTP, RTSP, and SIP, which dynamically open ports for the data path, so the payload can be classified according to content.

The Protocol Analyzer is also capable of determining which layer 3 packets belong (either directly or indirectly) to a trigger condition (for example, URL). In cases where the trigger condition cannot be uniquely defined at layers 3 and 4, then the trigger condition must be defined at layer 7 (that is, a specific URL must be matched).

Protocol Analyzer Software Stack

Every packet that enters the ECS subsystem must first go through the Protocol Analyzer software stack, which comprises of individual protocol analyzers for each of the supported protocols.

Figure 1: ECS Protocol Analyzer Stack



Note that protocol names are used to represent the individual protocol analyzers.

Each analyzer consists of fields and states that are compared to the protocol-fields and protocol-states in the incoming packets to determine packet content.

Important In 14.0 and later releases, the ECS HTTP analyzer supports both CRLF and LF as valid terminators for HTTP header fields.

Rule Definitions

Rule definitions (ruledefs) are user-defined expressions based on protocol fields and protocol states, which define what actions to take on packets when specified field values match.

Rule expressions may contain a number of operator types (string, =, >, and so on) based on the data type of the operand. For example, "string" type expressions like URLs and host names can be used with comparison operators like "contains", "!contains", "=", "!=", "starts-with", "ends-with", "!starts-with" and "!ends-with". In 19.2 and later releases, "!present" and "present" operators are added to enhance rule detection on the basis of absence/presence of Accept, Referer, X-header, User-agent, Cookies and Version fields in HTTP header request.

In 14.0 and later releases, ECS also supports regular expression based rule matching. For more information, refer to the *Regular Expression Support for Rule Matching* section.

Integer type expressions like "packet size" and "sequence number" can be used with comparison operators like "=", "!=", ">=", "<=". Each ruledef configuration consists of multiple expressions applicable to any of the fields or states supported by the respective analyzers.

Ruledefs are of the following types:

- **Routing Ruledefs** — Routing ruledefs are used to route packets to content analyzers. Routing ruledefs determine which content analyzer to route the packet to when the protocol fields and/or protocol-states in ruledef expression are true. Up to 256 ruledefs can be configured for routing.
- **Charging Ruledefs** — Charging ruledefs are used to specify what action to take based on the analysis done by the content analyzers. Actions can include redirection, charge value, and billing record emission.

  In releases prior to 21.1: Up to 2048 ruledefs can be configured in the system.

  In 21.1 and later releases: Up to 2500 ruledefs can be configured in the system.
- **Post-processing Ruledefs** — Used for post-processing purposes. Enables processing of packets even if the rule matching for them has been disabled.

Important When a ruledef is created, if the rule-application is not specified for the ruledef, by default the system considers the ruledef as a charging ruledef.

Ruledefs support a priority configuration to specify the order in which the ruledefs are examined and applied to packets. The names of the ruledefs must be unique across the service or globally. A ruledef can be used across multiple rulebases.

Important Ruledef priorities control the flow of the packets through the analyzers and control the order in which the charging actions are applied. The ruledef with the lowest priority number invokes first. For routing ruledefs, it is important that lower level analyzers (such as the TCP analyzer) be invoked prior to the related analyzers in the next level (such as HTTP analyzer and S-HTTP analyzers), as the next level of analyzers may require access to resources or information from the lower level. Priorities are also important for charging ruledefs as the action defined in the first matched charging rule apply to the packet and ECS subsystem disregards the rest of the charging ruledefs.

Each ruledef can be used across multiple rulebases, and up to 2048 ruledefs can be defined in a charging service in releases prior to 21.1. In 21.1 and later releases, up to 2500 ruledefs can be configured in a charging service.

In 15.0 and later releases, a maximum of 32 rule expressions (rule-lines) can be added in one ruledef.

Ruledefs have an expression part, which matches specific packets based upon analyzer field variables. This is a boolean (analyzer_field operator value) expression that tests for analyzer field values.

The following is an example of a ruledef to match packets:

```
http url contains cnn.com
```

–or–

```
http any-match = TRUE
```

In the following example the ruledef named "rule-for-http" routes packets to the HTTP analyzer:

```
route priority 50 ruledef rule-for-http analyzer http
```

Where, **rule-for-http** has been defined with the expressions: **tcp either-port = 80**

The following example applies actions where:

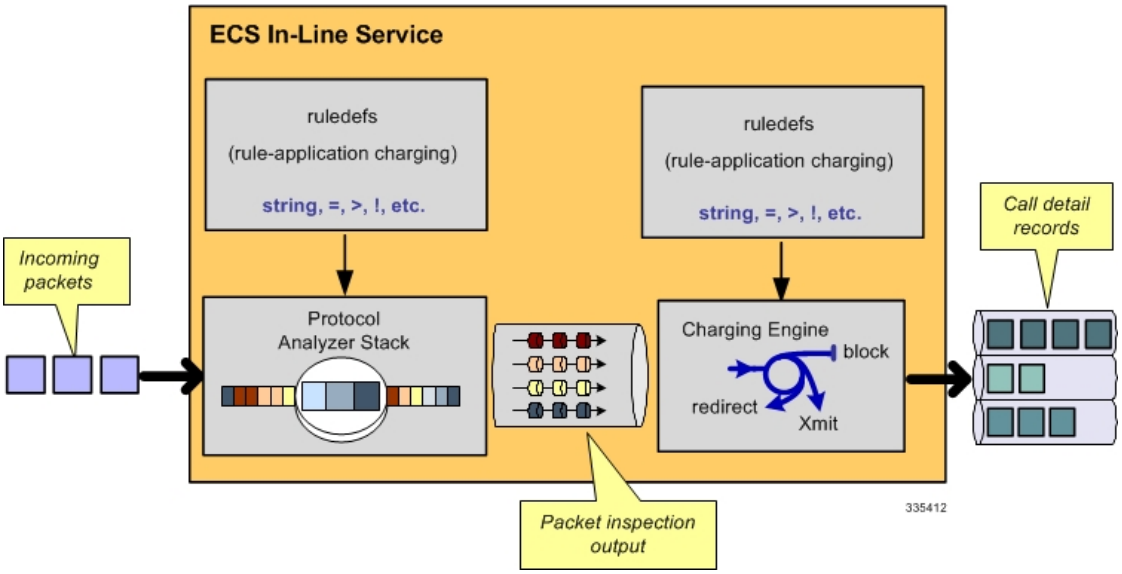
- Subscribers whose packets contain the expression "bbc-news" are not charged for the service.
- All other subscribers are charged according to the duration of use of the service.

```
ruledef port-80
     tcp either-port = 80
     rule-application routing
     exit
ruledef bbc-news
     http url starts-with http://news.bbc.co.uk
     rule-application charging
     exit
ruledef catch-all
     ip any-match = TRUE
     rule-application charging
     exit
charging-action free-site
     content-id 100
     [ ... ]
     exit
charging-action charge-by-duration
     content-id 101
     [ ... ]
     exit
rulebase standard
     [ ... ]
     route priority 1 ruledef port-80 analyzer http
```

```
action priority 101 ruledef bbc-news charging-action free-site
action priority 1000 ruledef catch-all charging-action charge-by-duration
[ ... ]
exit
```

The following figure illustrates how ruledefs interact with the Protocol Analyzer Stack and Action Engine to produce charging records.

Figure 2: ECS In-line Service Processing

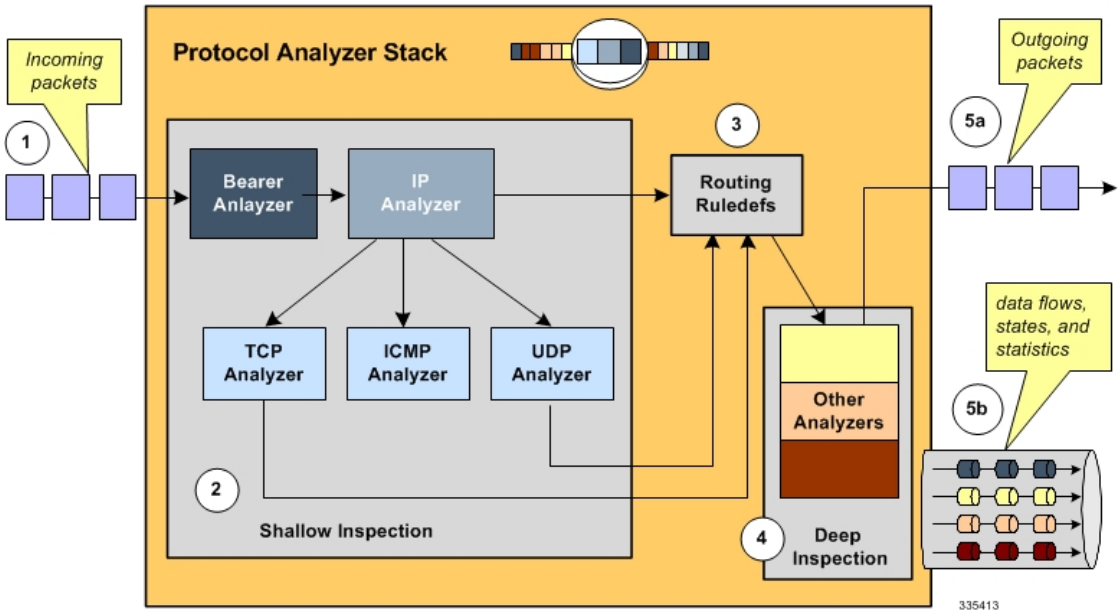


Packets entering the ECS subsystem must first pass through the Protocol Analyzer Stack where routing ruledefs apply to determine which packets to inspect. Then output from this inspection is passed to the charging engine, where charging ruledefs apply to perform actions on the output.

Routing Ruledefs and Packet Inspection

The following figure and the steps describe the details of routing ruledef application during packet inspection.

Figure 3: Routing Ruledefs and Packet Inspection



Step 1 The packet is redirected to ECS based on the ACLs in the subscriber's template /APN and packets enter ECS through the Protocol Analyzer Stack.

Step 2 Packets entering Protocol Analyzer Stack first go through a shallow inspection by passing through the following analyzers in the listed order:

a) Bearer Analyzer
b) IP Analyzer
c) ICMP, TCP, or UDP Analyzer as appropriate

Important In the current release traffic routes to the ICMP, TCP, and UDP analyzers by default. Therefore, defining routing ruledefs for these analyzers is not required.

Step 3 The fields and states found in the shallow inspection are compared to the fields and states defined in the routing ruledefs in the subscriber's rulebase.

The ruledefs' priority determines the order in which the ruledefs are compared against packets.

Step 4 When the protocol fields and states found during the shallow inspection match those defined in a routing ruledef, the packet is routed to the appropriate layer 7 or 7+ analyzer for deep-packet inspection.

Step 5 After the packet has been inspected and analyzed by the Protocol Analyzer Stack:

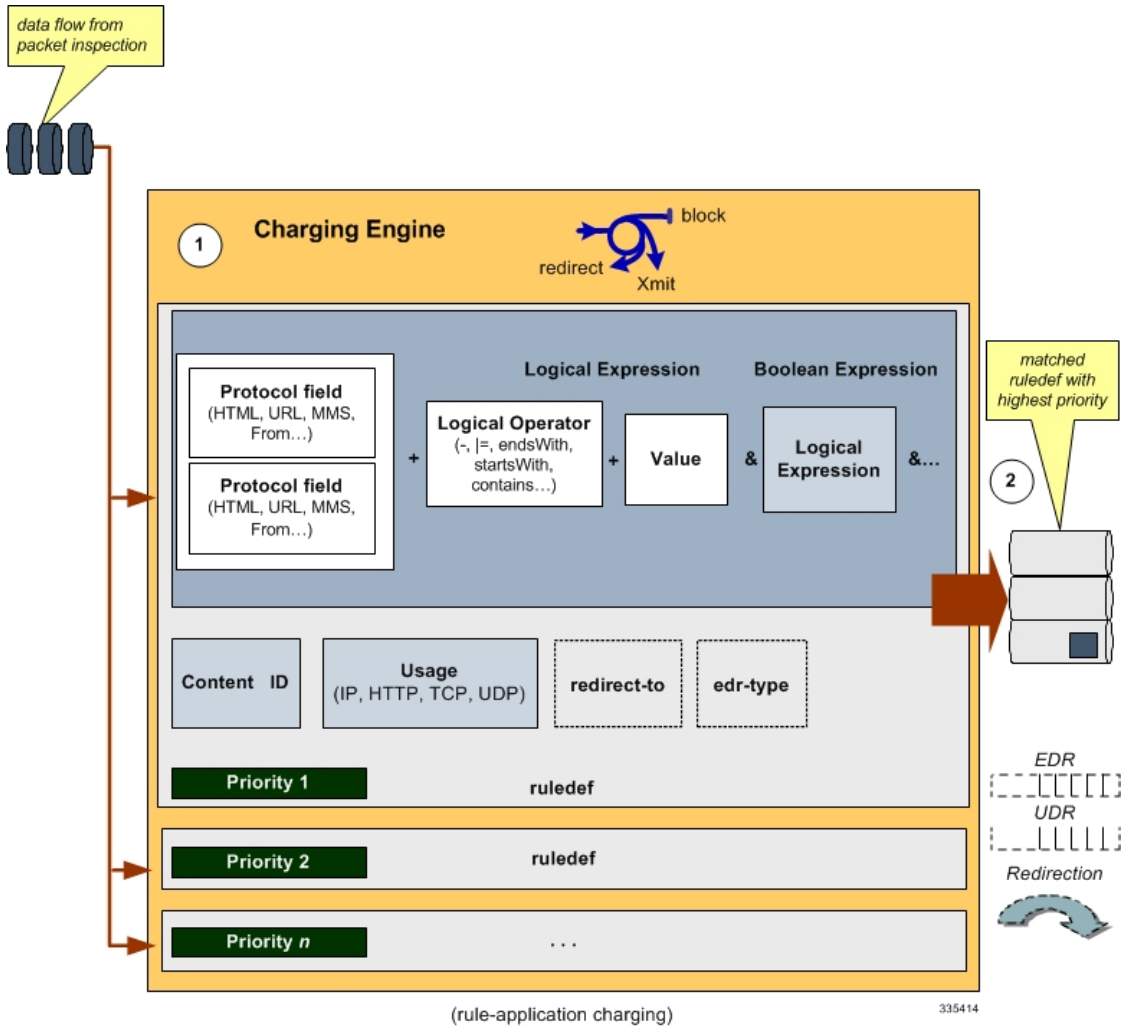
a) The packet resumes normal flow and through the rest of the ECS subsystem.
b) The output of that analysis flows into the charging engine, where an action can be applied. Applied actions include redirection, charge value, and billing record emission.

Charging Ruledefs and the Charging Engine

This section describes details of how charging ruledefs are applied to the output from the Protocol Analyzer Stack.

The following figure and the steps that follow describe the process of charging ruledefs and charging engines.

Figure 4: Charging Ruledefs and Charging Engine



Step 1 In the Classification Engine, the output from the deep-packet inspection is compared to the charging ruledefs. The priority configured in each charging ruledef specifies the order in which the ruledefs are compared against the packet inspection output.

Step 2 When a field or state from the output of the deep-packet inspection matches a field or state defined in a charging ruledef, the ruledef action is applied to the packet. Actions can include redirection, charge value, or billing record emission. It is also possible that a match does not occur and no action will be applied to the packet at all.

Regular Expression Support for Rule Matching

This section describes ECS support for regular expression (regex) rule matching.

In this release, ECS supports regex rule matching only for the following string-based rules:

- http host

- http referer
- http uri
- http url
- rtsp uri
- wsp url
- www url

When rule lines are added or modified, the entire trie is recreated and it mallocs memory for every URL present in the configuration. This leads to huge memory allocation that gets freed once the trie is created.

The following table lists the special characters that you can use in regex rule expressions.

Table 1: Special Characters Supported in Regex Rule Expressions

Regex Character	Description
*	Zero or more characters
+	Zero or more repeated instances of the token preceding the +
?	Match zero or one character **Important** The CLI does not support configuring "?" directly, you must instead use "077". For example, if you want to match the string "xyz<any one character>pqr", you must configure it as: **http host regex** *"xyz077pqr"* In another example, if you want to exactly match the string "url?resource=abc", you must configure it as: **http uri regex** *"url077resource=abc"* Where, the first "\" (backslash) is for the escaping of "?", and then "\077" for specifying "?" to the CLI.
\character	Escaped character
\?	Match the question mark (\<ctrl-v>?) character
\+	Match the plus character
*	Match the asterisk character
\a	Match the alert (ASCII 7) character
\b	Match the backspace (ASCII 8) character
\f	Match the form-feed (ASCII 12) character
\n	Match the new line (ASCII 10) character

Regex Character	Description
\r	Match the carriage return (ASCII 13) character
\t	Match the tab (ASCII 9) character
\v	Match the vertical tab (ASCII 11) character
\0	Match the null (ASCII 0) character
\\	Match the backslash character
Bracketed range [0-9]	Match any single character from the range
A leading ^ in a range	Do not match any in the range. All other characters represent themselves.
.\x##	Any ASCII character as specified in two-digit hex notation. For example, \x5A yields a "Z".
|	Specify OR regular expression operator **Important** When using the regex operator "|" in regex expressions, always wrap the string in double quotes. For example, if you want to match the string pqr OR xyz, you must configure it as: **http host regex** *"pqr|xyz'"*.

The following are some examples of the use of regex characters in rule expressions:

- The following command specifies a regex rule expression using the regex character * (asterisk) to match any of the following or similar values in the HTTP Host request-header field: host1, host101, host23w01.

  http host regex *"host*1"*

- The following command specifies a regex rule expression using the regex character + (plus) to match any of the following or similar values in the HTTP Host request-header field: host1, host101, host23w01.

  http host regex *"host+"*

- The following command specifies a regex rule expression using the regex character \077 (?) to match any of the following or similar values in the HTTP Host request-header field: host101.

  http host regex *"hos\077t101"*

- The following command specifies a regex rule expression using the regex character (escaped character) to match the following value in the HTTP Host request-header field: host?example.

  http host regex *"host\\\077example"*

  The first two \form an escape sequence and \077 is converted to ?. The \? is converted to ? as a character and not a place-holder.

- The following command specifies a regex rule expression using the regex character (escaped backslash character) to match the following value in the HTTP Host request-header field: host*01.

  http host regex *"host*01"*

  The first \ is used as an escape sequence for the second .

- The following command specifies a regex rule expression using the regex character \+ (escaped + character) to match the following value in the HTTP Host request-header field: host+01.

  http host regex *"host\\+01"*

- The following command specifies a regex rule expression using the regex character \\ (escaped backslash character) to match the following value in the HTTP Host request-header field: host\01.

  http host regex *"host\\01"*

- The following command specifies regex rule expression using the regex [0-9] to match any of the following or similar values in the HTTP Host request-header field: hostaBc, hostXyZ, hosthost. Values starting with the word "host" and not containing numbers.

  http host regex *"host[^0-9]"*

- The following command specifies regex rule expression using the regex [a-z] to match any of the following or similar values in the HTTP Host request-header field: hostabc, hostxyz, hosthost. Values starting with the word "host" and containing only lowercase letters.

  http host regex *"host[a-z]"*

- The following command specifies a regex rule expression using the regex | (or) to match either of the following values in the HTTP Host request-header field: host1, host23w01.

  http host regex *"host1|host23w01"*

- The following command defines a regex rule expression to match any of the following or similar values in the RTSP URI string: rtsp://pvs29p.cvf.fr:554/t1/live/Oui17, rtsp://pvs00p.cvf.fr:554/t1/live/Nrj12, rtsp://pvs90p.cvf.fr:554/t1/live/France24_fr.

  rtsp uri regex *"rtsp://pvs([0-9][0-9])p.cvf.fr:554/t1/live/(Gulli|Tf1|Tmc|Nrj12|France24_fr|Oui17)*"*

- The following command defines a regex rule expression to match either of the following values in the WWW URL string: http://tp2.site.com/httppvc_clnsssite.com.wap.symphonieserver.musicwaver.com/, http://134.210.11.13/httppvc_clnsssite.com.wap.symphonieserver.musicwaver.com/.

  www url regex
  "http://(tp2.site.com|134.210.11.3)/httppvc_clnsssite.com.wap.symphonieserver.musicwaver.com/"

- The following command defines a regex rule expression to match any of the following or similar values in the WSP URL string: wsp://home.opera.yahoo.com, wsp://dwld.yahoo.com, wsp://dwld2.yahoo.com.

  wsp url regex *"wsp://(dwld|opera|home.opera|dwld[1-3]).yahoo.com"*

- The following command defines a regex rule expression to match any of the following or similar values in the HTTP URL string: http://yahoo.com, http://www.yahoo.co.in, http://yahoo.com/news.

  http url regex *"(http://|http://www).yahoo.(co.in|com)*"*

- The following command defines a regex rule expression to match any of the following or similar values in the HTTP URI string: http://server19.com/search?form=zip, http://server20.com/search?form=pdf.

  http uri regex *"(http://|http://www).server[0-2][0-9].com/search?form=(pdf|zip)"*

How it Works

This section describes how regex rule matching works.

The following steps describe how regex rule matching works:

1. Regex ruledefs/group-of-ruledefs are configured in the CLI.

   Regex ruledefs are ruledefs that contain regex rule expressions. A ruledef can contain both regex and regular rule expressions.

   Regex group-of-ruledefs are group-of-ruledefs that contain regex ruledefs. A group-of-ruledefs can contain both regex and regular ruledefs.

2. After the regex ruledefs are configured, on the expiry of an internal 30 second timer, building of the regex engines is triggered.

   Note that one regex engine is built per each regex rule expression type.

   Just as with first-time or incremental configurations, SessCtrl/SessMgr recovery/reconciliation also triggers the building of regex engines.

3. The regex engine matches the regex string (specified in the regex expression) against live traffic, and returns all matching ruledefs.

4. The rule matches are then verified with those configured in the rulebase to determine the best matching rule.

Limitations and Dependencies

This section lists known limitations and restrictions to regex rule matching.

- Changes to ruledefs cause the optimization engines to get updated, hence any changes to ruledefs must be done with care. Preferably during low load times.
- Addition, modification, and deletion of regex ruledefs will result in rebuilding of regex engines, which is time consuming and resource intensive. While the engines are being rebuilt, rule-matching based on the old engines and old configurations may yield inconsistent results.

  Addition, modification, and deletion of action priority lines inside the rulebase has no impact on the regex engines. The regex engines remain intact and the removed action priorities from the rulebase are ignored during rule matching. Similarly, addition, modification (adding or removing ruledefs from it), or deletion of a group-of-ruledefs has no impact on regex engines.

- When adding regex ruledefs, use the following guidelines:
  - As per the current implementation, a maximum of 12 ruledefs is supported which contains rule lines as "xyz*" or "*xyz" or "*xyz*" as they are known to consume large memory. Instead, configure Aho-Corasick rules using "starts-with xyz" or "contains xyz" or "ends-with xyz" constructs, which comparatively consume less memory. The "starts-with", "ends-with" and "contains" operators are specially tailored for these types of operations, and work much faster (with lot less memory) than the corresponding "regex xyz*" or "regex *xyz*" operators. Hence, it is recommended that the "starts-with", "ends-with" and "contains" approach be preferred. Every regex rule line which contains "*" increases the memory/performance impact and its use must be avoided as much as possible.
  - Do not configure rules frequently. Push as much configuration as possible simultaneously so that all the regex rules are available for engine building at the same time. Frequent configuration changes may result in infinite loops with wasted memory and CPU cycles.

- Do not configure large number of regex rules as memory utilization will be high depending on the type of regex rules.
- Frequently monitor status of the engine using the **show active-charging regex status { all | instance** <*instance*> **}** CLI command in the Exec Mode. Where <*instance*> is the SessMgr instance number.

- When deleting ruledefs use the following guidelines:
  - Avoid deleting ruledefs at heavy loads, instead remove them from the required rulebases using the **no action priority** <*action_priority*> CLI command in the ACS Rulebase Configuration Mode. Doing so has no impact on regex building, although it uses additional memory there is no impact on traffic processing.
  - Deletion of ruledefs must be done during low load times. As described earlier, it is highly recommended that ruledefs be added, modified, or deleted in bulk, as it results in optimization engine updates.

Group of Ruledefs

Group-of-Ruledefs enable grouping ruledefs into categories. When a group-of-ruledefs is configured in a rulebase and any of the ruledefs within the group matches, the specified charging-action is applied and action instances are not processed further.

A group-of-ruledefs may contain optimizable ruledefs. Whether a group is optimized or not is decided on whether all the ruledefs in the group-of-ruledefs can be optimized, and if the group is included in a rulebase that has optimization turned on.

When a new ruledef is added, it is checked if it is included in any group-of-ruledefs, and whether it requires optimization.

The group-of-ruledefs configuration enables setting the application for the group (group-of-ruledefs-application parameter). When set to gx-alias, the group-of-ruledefs is expanded only to extract the rule names out of it (with their original priority and charging actions) ignoring the field priority set within the group. This is just an optimization over the PCRF to PCEF interface where a need to install/remove a large set of predefined rules at the same time exists. Though this is possible over the Gx interface (with a limit of 256), it requires a large amount of PCRF resources to encode each name. This also increases the message size.

This aliasing function enables to group a set of ruledef names and provides a simple one-name alias that when passed over Gx, as a Charging-Rule-Base-Name AVP, is expanded to the list of names with each rule being handled individually. From the PCEF point of view, it is transparent, as if the PCRF had activated (or deactivated) those rules by naming each one.

In 14.1 and earlier releases, a maximum of 128 ruledefs can be added to a group-of-ruledefs, and a maximum of 64 group-of-ruledefs can be configured.

In 15.0 and later releases, a maximum of 128 ruledefs can be added to a group-of-ruledefs, and a maximum of 128 group-of-ruledefs can be configured.

In 20.1 and later releases, a maximum of 512 ruledefs can be added to a group-of-ruledefs, and a maximum of 384 group-of-ruledefs can be configured.

Important The total number of ruledefs supported for all GoRs must be used with caution due to the high memory impact. Any modifications to the ruledef or GoR configurations beyond the WARN state of the SCT Task memory may have adverse impact on the system.

Rulebase

A rulebase allows grouping one or more rule definitions together to define the billing policies for individual subscribers or groups of subscribers.

A rulebase is a collection of ruledefs and their associated billing policy. The rulebase determines the action to be taken when a rule is matched. A maximum of 512 rulebases can be specified in the ECS service.

It is possible to define a ruledef with different actions. For example, a Web site might be free for postpaid users and charge based on volume for prepaid users. Rulebases can also be used to apply the same ruledefs for several subscribers, which eliminate the need to have unique ruledefs for each subscriber.

Rulebase List

A rulebase list allows grouping one or more rulebases together, enabling the Online Charging System (OCS) to choose the rulebase for a subscriber from the rulebase list.

A rulebase list enables a list of rulebases to be sent to the OCS over the Gy interface using a buffer. The OCS can then select a specific rulebase from the rulebase list, and apply the ruledefs and billing policies associated with that rulebase to subscribers.

Rulebase lists are created and configured in the ACS Configuration Mode. The maximum length of an individual rulebase-list name is 64 bytes. The buffer that stores space-separated rulebase names within a rulebase-list is of 256 bytes.

In 12.3 and earlier releases, a maximum of 20 rulebase lists can be configured per active charging service.

In 14.0 and later releases, a maximum of 128 rulebase lists can be configured per active charging service.

When a subscriber call is connected, the Session Manager provides the list of rulebase names to the OCS, which chooses the rulebase to be used for the subscriber session from the list.

In case the OCS is not reachable, the rulebase configured as the default will be used.

Bulk Statistics Support

The system's support for bulk statistics allows operators to choose which statistics to view and to configure the format in which the statistics is presented. This simplifies the post-processing of statistical data since it can be formatted to be parsed by external, back-end processors.

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. The following schemas are supported by ECS:

- **ECS**: Provides Enhanced Charging Service statistics
- **ECS Rulebase**: Provides Enhanced Charging Service Rulebase statistics

The system supports the configuration of up to 4 sets (primary/secondary) of receivers. Each set can be configured with to collect specific sets of statistics from the various schemas. Statistics can be pulled manually from the chassis or sent at configured intervals. The bulk statistics are stored on the receiver(s) in files.

The format of the bulk statistic data files can be configured by the user. Users can specify the format of the file name, file headers, and/or footers to include information such as the date, chassis host name, chassis uptime, the IP address of the system generating the statistics (available for only for headers and footers), and/or the time that the file was generated.

When the Web Element Manager is used as the receiver, it is capable of further processing the statistics data through XML parsing, archiving, and graphing.

The Bulk Statistics Server component of the Web Element Manager parses collected statistics and stores the information in the PostgreSQL database. If XML file generation and transfer is required, this element generates the XML output and can send it to a Northbound NMS or an alternate bulk statistics server for further processing.

Additionally, if archiving of the collected statistics is desired, the Bulk Statistics server writes the files to an alternative directory on the server. A specific directory can be configured by the administrative user or the default directory can be used. Regardless, the directory can be on a local file system or on an NFS-mounted file system on the Web Element Manager server.

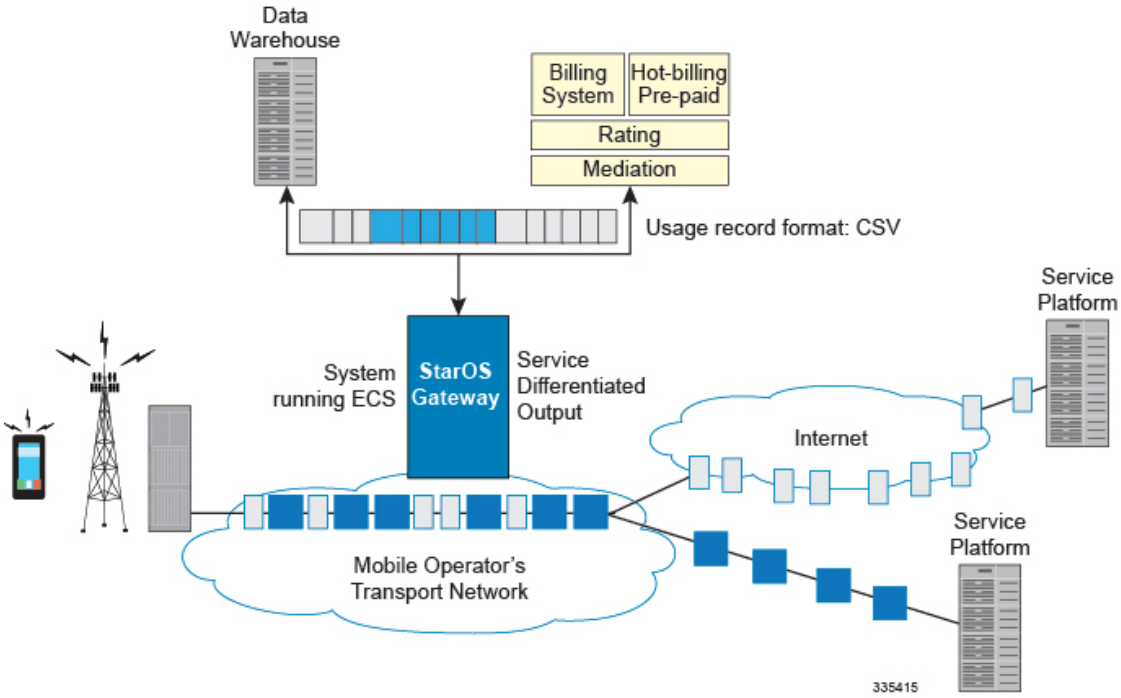
For more information on bulk statistic configuration, refer to the *Configuring and Maintaining Bulk Statistics* chapter in the *System Administration Guide*.

For more information on bulk statistic variables, see the *ECS Schema Statistics* and *ECS Rulebase Schema Statistics* chapter of the *Statistics and Counters Reference*.

ECS Deployment and Architecture

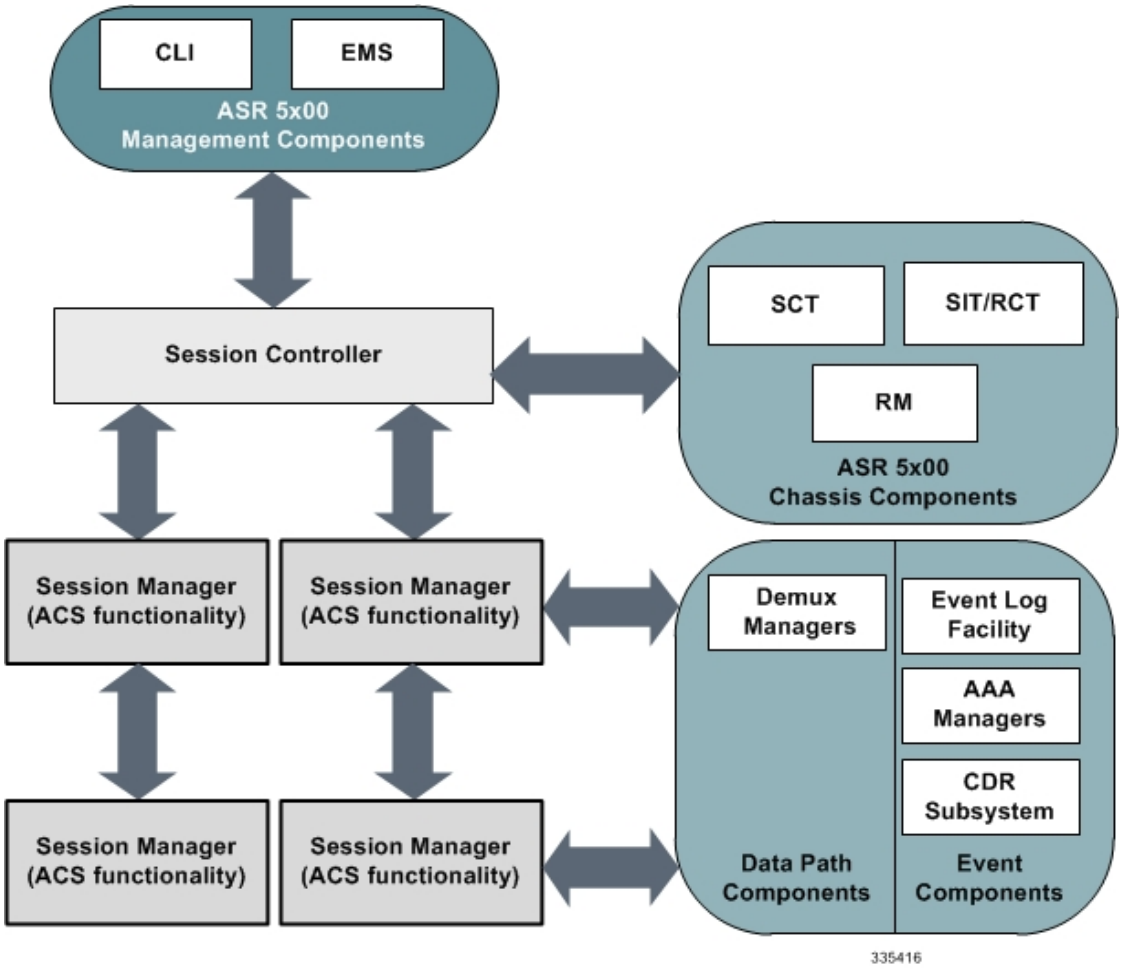
The following figure shows a typical example of ECS deployment in a mobile data environment.

Figure 5: Deployment of ECS in a Mobile Data Network



The following figure depicts the ECS architecture managed by the Session Controller (SessCtrl) and Session Manager (SessMgr) subsystems.

Figure 6: ECS Architecture



Service-Scheme Framework

The Service-scheme framework is introduced to disassociate the dependency with rulebase/PCRF and associate them with policies on the basis of rulebase name, APN name, IMSI range, and so on. This feature offers reduced PCRF dependency that can in turn reduce Gx signaling and integration of any third-party PCRF. The service-scheme framework configuration introduces several CLI commands in support of this feature.

With the previous implementation, the features/policies are always associated with the rulebase and any new proprietary feature for individual subscriber or certain set of subscribers requires support from PCRF. The service-scheme framework helps in overriding this feature behavior for subscribers without involving PCRF. The user can also update the policies specific to subscribers based on pre-configured events.

The subscribers will be classified on the basis of rulebase name, APN name, v-APN name, and so on. These conditions can also be used in combination. If multiple set of conditions are defined for a set of subscribers then conditions with higher priority will be applied for subscriber's selection.

The two main constructs for the new policy framework are listed below:

- Subscriber-base: This helps in associating subscribers with service-scheme based on rule-base, APN name, v-APN name, and so on.
- Service-scheme: This helps in associating trigger actions based on trigger conditions that can be applied on different events at call-setup time, location-update time, flow creation time or any other event triggered through control or data path.

Notes:

- Conflicting actions between PCRF and service-scheme framework against the same trigger events must not be configured.
- If service-scheme is deleted while the call is active, then no new triggers will be processed but existing trigger actions will be applicable for the call duration.
- Any change in the classification of the subscriber will not modify the existing trigger actions for the current active call.
- Any configuration change under subscriber-class condition will be evaluated only for new calls.
- After SR/ICR, the framework will re-evaluate trigger condition configured only under "sess-setup" trigger event.
- Any change under trigger events related to trigger condition and trigger action will depend on the type of trigger event.
  - For session-setup trigger event, the change will be reflected on new calls.
  - For location-update trigger event, the change will be reflected whenever the subscriber changes location.

Refer to the *Configuring Service-scheme Framework* section in the *Enhanced Charging Service Configuration* chapter for more details.

Enhanced Features and Functionality

This section describes enhanced features supported in ECS.

Important The features described in this section may be licensed Cisco features. 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*.

Content Filtering Support

ECS provides offline content filtering support and in-line static and dynamic content filtering support to control static and dynamic data flow and content requests.

Content Filtering Server Group Support

ECS supports external Content Filtering servers through Internet Content Adaptation Protocol (ICAP) implementation between ICAP client and Active Content Filter (ACF) server (ICAP server).

ICAP is a protocol designed to support dynamic content filtering and/or content insertion and/or modification of Web pages. Designed for flexibility, ICAP allows bearer plane nodes such as firewalls, routers, or systems running ECS to interface with external content servers such as parental control (content filtering) servers to provide content filtering service support.

In-line Content Filtering Support

Content Filtering is a fully integrated, subscriber-aware in-line service available for 3GPP and 3GPP2 networks to filter HTTP and WAP requests from mobile subscribers based on the URLs in the requests. This enables operators to filter and control the content that an individual subscriber can access, so that subscribers are inadvertently not exposed to universally unacceptable content and/or content inappropriate as per the subscribers' preferences. Content Filtering uses Deep Packet Inspection (DPI) capabilities of ECS to discern HTTP and WAP requests.

Important For more information on Content Filtering support, refer to the *Content Filtering Services Administration Guide*.

Implementation of AES Encryption

URL redirection is used for user equipment (UE) self-activation, along with pre-paid mobile broadband and other projects.

In the current implementation, when a URL redirection occurs, additional dynamic fields such as MSISDN, IMEI, and username can be appended to the redirection URL for use by the IT portal during the account activation process. StarOS currently supports URL encryption of attributes within the redirection by using Blowfish (64 and 128 bit keys) encryption. It also provides the ability to encrypt either single or multiple concatenated plain text fields. However, Blowfish is no longer considered robust and thus operator now has the option to augment the security of these redirection parameters with a more robust encryption based on AES Encryption.

For URL encryption, AES is an additional option along with Blowfish. The operator has flexibility of choosing the encryption mechanism— Blowfish or AES. This is achieved using CLI and there are no changes done to the dynamic fields. The operator can have different encryption for different rules configurable using CLI.

AES encryption is available for 128 and 256 bit keys. For AES encryption with CBC mode of operation, a key-phrase is taken as configurable field from the operator. This key phrase is internally converted to a 128/256 bit key. An additional field value ("salt") is also allowed as a configurable field. This configurable field is optional.

Security of the subscriber sensitive attributes is enhanced with a more robust encryption algorithm. This helps protect subscriber specific information sent to different servers, thus helping operators to adhere to regulatory policies.

For more information on these commands, see the *ACS Charging Action Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

IP Readdressing

The IP Readdressing feature enables redirecting unknown gateway traffic based on the destination IP address of the packets to known/trusted gateways.

IP Readdressing is configured in the flow action defined in a charging action. IP readdressing works for traffic that matches particular ruledef, and hence the charging action. IP readdressing is applicable to both uplink and downlink traffic. In the Enhanced Charging Subsystem, uplink packets are modified after packet inspection, rule matching, and so on, where the destination IP/port is determined, and replaced with the readdress IP/port just before they are sent out. Downlink packets (containing the readdressed IP/port) are modified as soon as they are received, before the packet inspection, where the source IP/port is replaced with the original server IP/port number.

For one flow from an MS, if one packet is re-addressed, then all the packets in that flow will be re-addressed to the same server. Features like DPI and rule-matching remain unaffected. Each IP address + port combination will be defined as a ruledef.

In case of IP fragmentation, packets with successful IP re-assembly will be re-addressed. However, IP fragmentation failure packets will not be re-addressed.

New hierachy approach has also been provided for selecting the server in case of server list configured under charging-action. This helps the operator to specify list of DNS servers in the order of preference. In hierarchy based approach, queries are redirected as per primary, secondary, and tertiary selection. Both round-robin and hierarchy based server selection approaches would be applicable for both IPv4 and IPv6 based servers. An additional CLI is provided that enables you to select from hierarchy or round-robin approach for server selection. See the *Configuring IP Readdressing* for more information.

Next-hop Address Configuration

ECS supports the ability to set the next-hop default gateway IP address as a charging action associated with any ruledef in a rulebase. This functionality provides more flexibility for service based routing allowing the next-hop default gateway to be set after initial ACL processing. This removes need for AAA to send the next-hop default gateway IP address for CC opted in subscribers.

In 15.0 and later releases, ECS behaves such that rule matching is not done for partial HTTP request if HTTP analysis is enabled.

Assume ECS has received partial HTTP GET packet where URL is not complete, and there are a few URL based rules configured. At this point of time, ECS will not be in a position to match proper rule as complete URL information is not available. When packet where request is completed, is received by ECS, proper rule matching is possible. Earlier partial packets and bytes of this request will be charged accordingly.

Also, this does not apply to post-processing rules. Post-processing rules are matched for all the packets, irrespective of the packet is partial or not. If the customer wants to configure actions like next-hop forwarding or ip-readdressing, then that can be configured in post-processing rules.

In releases prior to 15.0, partial packets do not go for post processing rule match. Whereas in 15.0 and later releases, the partial packets go for required rule match. This behavior change is introduced to obtain the correct statistics about the packets.

How it works:

Step 1 The next-hop address is configured in the charging action.

Step 2 Uplink packet sent to ECS is sent for analysis.

Step 3 When the packet matches a rule and the appropriate charging action is applied, the next-hop address is picked from the charging action and is copied to the packet before sending the packet to Session Manager.

Step 4 Session Manager receives the packet with the next-hop address, and uses it accordingly.

Post Processing

The Post Processing feature enables processing of packets even if the rule matching for them has been disabled. This enables all the IP/TCP packets including TCP handshaking to be accounted and charged for in the same bucket as the application flow. For example, delay-charged packets for IP Readdressing and Next-hop features.

- Readdressing of delay-charged initial hand-shaking packets.
- Sending the delay-charged initial packets to the correct next-hop address.
- DCCA—Taking appropriate action on retransmitted packets in case the quota was exhausted for the previous packet and a redirect request was sent.
  - DCCA with buffering enabled—Match CCA rules, charging-action will decide action—terminate flow/redirect
  - DCCA with buffering disabled—Match post-processing rules, and take action
- Content ID based ruledefs—On rule match, if content ID based ruledef and charging action are present, the rule is matched, and the new charging action will decide the action

A ruledef can be configured as a post-processing rule in the ruledef itself using rule-application of the ruledef. A rule can be charging, routing, or a post-processing rule. If the same ruledef is required to be a charging rule in one rulebase and a post-processing rule in another one, then two separate identical ruledefs must be defined.

How the Post-processing Feature Works

The following steps describe how the Post-processing feature works:

Step 1 Charging rule-matching is done on packets and the associated charging-action is obtained.

Step 2 Using this charging-action the disposition-action is obtained.

Step 3 If the disposition action is to either buffer or discard the packets, or if it is set by the ACF, or if there are no post-processing rules, the packets are not post processed. The disposition action is applied directly on the packets. Only if none of the above conditions is true, post processing is initiated.

Step 4 Post-processing rules are matched and the associated charging-action and then the disposition-action obtained through control-charge.

Step 5 If both match-rule and control-charge for post processing succeed, the disposition-action obtained from post-processing is applied. Otherwise, the disposition-action obtained from charging rule-matching is used.

If no disposition action is obtained by matching post-processing rules, the one obtained by matching charging-rules will be applied.

Irrespective of whether post processing is required or not, even if a single post-processing rule is configured in the rulebase, post processing will be done.

The following points should be considered while configuring post-processing rules for next-hop/readdressing.

- The rules will be L3/L4 based.
- They should be configured in post-processing rules' charging actions.

For x-header insertion, there should either be a post-processing rule whose charging-action gives no disposition-action or the packet should not match any of the post-processing rules so that the disposition action obtained from charging-rule matching is applied.

Pre-defined Rule Retention for Rulebase Change Trigger from Charging Action

Rulebase change is triggered from the Gx, Gy and RADIUS CoA external interfaces, and also from charging action by configuring the rulebase change in the charging action definition. With the old implementation, the rulebase change trigger does not retain predefined rules that are common between the current rulebase and destination rulebase. The predefined rules in all triggers could be deactivated and activated even if they existed in the destination rulebase.

With this release, when rulebase change is triggered through charging action, the predefined rules common between the current rulebase and destination rulebase are retained. This change applies only to the charging-action trigger and no rule retention is done for external triggers - Gx, Gy, and RADIUS CoA interfaces.

The following behavior and limitations are applicable with the rulebase change trigger from charging action:

- Rules will be retained only for rulebase change triggered through charging action.
- Rules with matching rule name and charging action name in the destination rulebase will be retained.
- Rule retention will be applied for ADC rules.

  Only the APP-START event notification will be seen for ADC rules as rule is retained on rulebase change.
- Rule retention will be supported for static-and-dynamic rules only from Gx R7 onwards.
- Rules will not be retained for rulebase change triggered through external interfaces - Gx, Gy, and RADIUS CoA.
- Rules will not be retained for SFW and NAT rules.
- Rules will not be retained for UDR and CDR functionality.
- This is a configuration restriction. In a scenario for rulebase change where current rulebase has a rule with configuration in its charging action to change the current rulebase to new rulebase. If there is the same rulename configured with new rulebase in the same charging action, then this scenario could lead to loop of rulebase change. This is the existing behavior and will not be fixed. Hence, this configuration will not be valid.

RADIUS Based Dual Factor Authentication For Mobile Private Network

Dual Factor Authentication has been implemented for Mobile Private Network's (MPN's) mobile devices, most typically for terminals like lottery machine devices, ATMs, and so on. For security reasons, this DFA procedure is followed before traffic can flow normally. The first level authentication happens as part of call setup using RADIUS. While the call is established, the pre-DFA-rulebase that has the configuration to allow only RADIUS and ICMP traffic is used; rest of the traffic is dropped. Until then all the normal traffic is denied and is resumed only after the additional RADIUS based authentication is successful.

The success of RADIUS authentication is determined by a RADIUS analyzer. This analyzer understands the authentication requests and responses especially 'Access-Request' and 'Access-Accept'. Whenever the RADIUS 'Access-Request' message is matched with 'Access-Accept' message, the rulebase is changed to new rulebase

called Post-DFA-rulebase and the existing dedicated bearers are deleted and the same is informed to PCRF. The RADIUS analyzer does not analyze any other message but only the 'Access-Request', 'Access-Accept', and the 'Access-Reject'.

For the Dual Factor Authentication feature to function, the config pre-DFA-rule-base, the RADIUS analyzer, and the post-DFA-rule-base.

For more information on configuring the Radius Analyzer, see *Configuring RADIUS Analyzer* section in the *Enhanced Charging Service Configuration* chapter.

RAN Bandwidth Optimization

When the rule is installed and active, P-GW uses the GBR/MBR assigned in the rule for calculating the GBR / MBR values towards the bearers created. When more than one rule is installed, P-GW adds the GBR / MBR values from all the active and installed rules even if the flow of a certain rule is marked as disabled. This current behavior is in accordance with 3GPP TS standard specification 29.212, and this might result in RAN bandwidth wastage. To avoid this wastage, some optimization is done while calculating MBR and GBR for GBR bearer.

The RAN bandwidth optimization feature provides the ability to configure a list of APNs, for which the optimized calculation of MBR, GBR can be enabled. By default, this optimized calculation should be enabled only for the IMS APN.

This feature further helps optimize the logic of aggregating MBR and GBR values, based on "Flow-Status" AVP value received in the rule definition through RAR. This operation is controlled through a CLI command **ran bandwidth optimize** added in the ACS Rulebase configuration mode.

For more information on this command, see the *ACS Rulebase Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

Selective TFT Suppression for Default Bearer

With this feature, the selected TFT updates can be controlled and sent to the UE. A new CLI command "**tft-notify-ue**" is introduced, which suppresses the selected TFT updates to the UE. This is provided by specific charging-action level option to identify if the appropriate TFT defined in the charging action needs to be sent to the UE or not. This CLI is supported for both default and dedicated bearer.

One more new CLI "**tft-notify-ue-def-bearer**" to suppress TFTs on default bearer has been added, so the operator has the flexibility to configure this per Rulebase and also configure to suppress TFT updates only. This CLI allows sending other QoS updates to the UE and is only controlling TFT related updates. This CLI is supported only for default bearer.

For more information on these commands, see the *ACS Charging Action Configuration Mode Commands* and *ACS Rulebase Configuration Mode Commands* chapters in the *Command Line Interface Reference*.

Service Group QoS Feature

The Service Group QoS feature enables the chassis/PCEF to define and enforce Fair-Usage-Policy (FUP) per subscriber. This enables changing certain charging-action parameters and all QoS-group-of-ruledefs parameters over the Gx interface per individual subscriber session.

In the chassis/PCEF, the Service Group QOS feature enables to:

- Define service-groups that may include unrelated services defined on the chassis/PCEF.

- Dynamically install pre-defined service-groups for a subscriber over Gx.
- Dynamically remove pre-defined service-groups for a subscriber over Gx.
- Dynamically set and change QoS parameters of a service-group for a subscriber over Gx using CCA and RAR messages. QoS parameters of a service-group (FUP-QoS) are:
  - Flow-Rate
  - Flow-Status
  - Volume Threshold
- Apply Flow-Status to a packet-flow progressively at service and service-group levels. The rules for hierarchical enforcement of Flow-Status rule are:
  - Flow-Status Gating at Service Level: If rule indicates "block" or "redirect", then that action is taken. If rule indicates "allow", then next level's gating rule is applied.
  - Flow-Status Gating at Service-Group Level: If rule indicates "block" or "redirect", then that action is taken. If rule indicates "allow", then next level's gating rule is applied.
- Apply Flow-Rate to a packet-flow progressively at service and service-group levels. Maximum Bit-rate and Burst size is defined by Flow-Rate. Meter the traffic to the configured Flow-Rate and based on the output, apply DSCP marking to the packet.

Important The output action of Flow-Rate can be forward, drop, or mark DSCP. Flow-Rate may allow the packet without DSCP marking.

The rules of hierarchical QOS enforcement are:

- Metering at Service Level: Initially, traffic is metered against service-level QOS rule. If the result of metering marks or drops the packet, then the next level metering is not performed.
- Metering at Service-Group Level: If the packet is allowed at service level, then service-group level QOS metering is done. If the result of metering marks or drops the packet, then the next level metering is not performed.

Important The packet is first subjected to Flow-Status enforcement and if allowed by Flow-Status only then Flow-Rate is enforced. Flow-Status enforcement includes applying Flow-Status progressively at service and service-group Levels. If the flow-status at both levels allows the packet to pass only then it is given for flow-rate enforcement, which applies Flow-Rate progressively at service and service-group levels.

- Monitor volume usage of a group-of-services. Multiple group-of-services can share a volume-quota.
- Provide a mechanism to share configured volume threshold of a service-group across all services in that group. This sharing would be dynamic, that is no predefined quota is allocated per service in a service group.

- Generate a notification to PCRF in a CCR-U message, when volume threshold for a group-of-services is crossed. Once a notification is generated, the trigger is disarmed to generate notification. Continue to monitor usage, but do not report further breaches until PCRF explicitly enables threshold-breach notification trigger in a CCA-U message.
- Report volume usage to PCRF in a CCR-U message when the service-group removed is the last using the shared volume-quota.

Configuration Overview

QoS-group-of-ruledefs are statically configured in the CLI, in the Active Charging Service Configuration Mode. The CLI allows addition and removal of charging and dynamic ruledefs to a named QoS-group-of-ruledefs. A single ruledef can belong to multiple QoS-groups. A maximum of 64 QoS-group-of-ruledefs can be configured in the ACS service. Each QoS-group-of-ruledefs can contain up to 128 ruledefs.

PCRF will be aware of all QoS-group-of-ruledefs names and their constituent ruledefs configured on the chassis/PCEF. The PCRF can activate and remove QoS-group-of-ruledefs for a subscriber session over Gx using a proprietary AVP in CCA and RAR messages. This AVP specifies the name of the QoS-group-of-ruledefs to activate or to remove. Individual ruledefs cannot be dynamically added or removed from a predefined QoS-group-of-ruledefs over the Gx interface. Attributes of QoS-group-of-ruledefs (FUP-QoS parameters) cannot be defined in the CLI. These parameters can only be set and changed over the Gx interface. This feature allows setting different QoS parameters for different subscribers for a named QoS-group-of-ruledefs.

The following attributes of QoS-group-of-ruledefs are supported:

- Precedence or Priority: Priority of a QoS-group-of-ruledefs implies priority of applying QoS-parameters of a QoS-group-of-ruledefs to an incoming data packet. If a packet matches a charging rule which is part of multiple QoS-groups activated for the session, then QoS-parameters of the QoS-group-of-ruledefs with highest priority is applied to the packet. A lower priority number indicates higher priority of application of QoS-parameters of that group. Priority of a QoS-group-of-ruledefs is set by PCRF over Gx for each subscriber session.
- Flow-Status: Can be set to Forward, Block, or Redirect.

Important The Append-Redirect option is not supported.

Important Block can be for uplink, downlink, or both uplink and downlink traffic.

- Flow-Rate: Specifies max rate, max burst-size, conform action, and exceed action; individually for uplink and downlink traffic.
- Usage Monitoring Key: A monitoring key, which has an integer value, is set by PCRF over Gx. Volume threshold values are set for this key by PCRF, to perform usage monitoring. Usage is tracked against a monitoring key.
- Volume Thresholds: The PCRF can set volume threshold values for a monitoring key over Gx. An event is reported when thresholds are crossed, and usage is reported at predefined events — such as session termination and when the QoS-group-of-ruledefs removed is the last using the shared volume-quota.

Important In this release, time thresholds are not supported.

- Attributes of QoS-group-of-ruledefs cannot be defined using CLI. These attributes can only be set and changed over Gx. This allows setting different QoS parameters for different subscribers for a named QoS-group-of-ruledefs.
- When a QoS-group-of-ruledefs is activated, its QoS parameters can be set and changed over Gx. This is achieved using a combination of standard and proprietary AVPs.
- The following attributes of charging-action can be set and changed by PCRF over Gx.
  - Flow-Status: Can be set to Forward, Block, or Redirect.

Important ECS does not support the Append-Redirect option.

Important Block can be for uplink, downlink, or both uplink and downlink traffic.

  - Flow-Rate: Specifies max rate, max burst-size, conform action, and exceed action; individually for uplink and downlink traffic.
  - Volume Threshold: Thresholds are set for usage monitoring. PCRF can set threshold for a monitoring key, which is statically defined for a charging-action using CLI. Usage is reported when thresholds are crossed, and at predefined events such as session termination and removal of QoS-group-of-ruledefs.

Important Monitoring-key is not received over Gx for static charging-action. Triggers for threshold breached are same as the usage-reporting for static charging-action.

- Flow-Status and Flow-Rate can be statically defined for a charging action, and thus applied to a ruledef. These parameters may be overridden by PCRF over Gx. Volume-Threshold-Key (monitoring key) can be statically defined for a ruledef in a rulebase. However, its value — the volume quota — can only be set over Gx. Parameters set over Gx will always take precedence over any static configuration.

Important Time-Monitoring over Gx is not supported in this release.

Support for Service-based QoS

As explained earlier, a service can be mapped in ECS to a set-of-ruledefs with the same charging-action applied to them. This section explains the support for QoS control at the charging-action level:

- Flow-Status: In ECS, you can configure a flow-action in a charging-action. If flow-action is not configured for a charging-action, it implies "Forward" action.

Important Flow-Status value of "Append-Redirect" is not supported by ECS.

- Flow-Rate: ECS charging-action supports configuration of bandwidth limits for a flow. Flow limits can be separately configured for uplink and downlink. ECS supports configuration of peak data-rate and burst-size as well as committed data-rate and burst-size, along with corresponding exceed actions. Specification of committed rate and burst-size is optional.

  ECS does not support specifying conform-action (i.e. conform-action is always "Allow"). For exceeding traffic it supports only "Drop" and "Set IP-TOS to 0" as actions. In ECS, traffic matching a flow — both conforming and exceeding, cannot be marked with a specific DSCP mark.

  In ECS, charging-action also contains a Content-Id. Multiple charging-actions can contain the same Content-Id. ECS supports a bandwidth-limiting meter per charging action per subscriber session. This metering is separate from traffic meters that are keyed on Content-Id.

- Volume Thresholds: ECS supports setting and monitoring Volume Threshold per flow using the "monitoring-key" mechanism. Monitoring-key is specified in a rulebase configuration. Monitoring-key is associated with a volume-threshold, which is set over Gx. A single monitoring-key can be specified for multiple ruledefs. This allows sharing of assigned volume quota across all the ruledefs with the same Monitoring-Key ID. To configure service-level volume quota, you can configure the same monitoring-key for all ruledefs that share the same charging action. Monitoring-Key mechanism enables setting and changing Volume-threshold over Gx.

  In ECS, changing QoS parameters at a service level means changing parameters of a charging-action.

  ECS supports three different kinds of ruledefs:

  - Static rules that are defined in the CLI, and are active immediately after they are defined.
  - Pre-defined rules that are defined in the CLI and activated/deactivated over Gx.
  - Dynamic rules which are defined, activated and deactivated over Gx.

  For static and predefined rules, ECS supports updating per-subscriber FUP parameters of a charging-action over Gx. This is achieved using the Charging-Action-Install AVP. Changes to FUP-parameters of dynamic rules are done using the 3GPP-standard Charging-Rule-Definition AVP.

Hierarchical Enforcement of QoS Parameters

When a packet arrives, ECS performs Deep Packet Inspection and rule matching. If the packet matches a rule, Control-Charge processing is performed as defined by the matched rule. Ruledef-level and QoS-group-of-ruledefs level QoS enforcement are performed as part of Control-Charge processing.

It is not mandatory to set QoS parameters for a ruledef over Gx. If QoS parameters are not set over Gx, then static definition, if any, is enforced. Similarly, for a subscriber session it is not mandatory to group ruledefs in one or more QoS-group-of-ruledefs. A subscriber may not have any QoS-group-of-ruledefs configured. Incoming traffic may match a ruledef, which has no associated QoS-group-of-ruledef for that subscriber session. In that case, action is taken based only on the configuration for that ruledef.

Applying Flow-Status

Flow-Status is applied in a hierarchical manner with the following precedence:

1. Flow Gating at charging-action Level: If flow-action in charging rule indicates "block" or "redirect", then that action is taken. If rule indicates "allow", then next level's gating rule is applied.
2. Flow Gating at QoS-Group-of-Ruledefs Level: Flow-Status specified for the matched QoS-group-of-ruledefs is applied.

Applying Flow-Rate

Hierarchy of metering and marking packet follows the precedence:

1. Metering at Charging-Action Level: Flow-Rate at ruledef level is specified in the charging-action associated with the ruledef. Bandwidth metering specified for the charging-action is first applied to every packet. If the packet conforms to specified bandwidth limits, then QoS-group-of-ruledefs level metering will be performed. If the packet exceeds bandwidth limit at charging-action, then specified exceed action will be taken and bandwidth metering at QoS-group-of-ruledefs and subscriber level will not be performed.
2. Metering at QoS-Group-of-Ruledefs Level: If a packet conforms to charging-action bandwidth limits, then QoS-group-of-ruledefs level bandwidth metering will be done. If the packet conforms to specified bandwidth limits, then subscriber-level metering will be performed. If the packet exceeds bandwidth limit at QoS-group, then specified exceed action will be taken.

Monitoring Usage and Reporting Threshold Breaches

Volume usage is tracked at the charging-action level and at QoS-group-of-ruledefs level. If a received packet causes volume threshold to exceed, then a trigger ECS sends a CCR-U message to PCRF with Service-Group-Event AVP indicating the relevant threshold that was crossed. ECS will then disarm the trigger. If the trigger needs to be rearmed, PCRF will explicitly enable it in the CCA-U message.

In 14 and later releases, Time Reporting over Gx is supported. The time usage is tracked at session/flow level and will be reported to PCRF on meeting certain conditions.

FUP Enforcement for Dynamic Rules

The chassis/PCEF supports dynamic rule installation using 3GPP-standards-based AVPs. The Charging-Rule-Definition AVP is used to install dynamic rules and configure charging behavior and QoS parameters. For dynamic rules, charging-action is part of the rule definition, and not a separate named entity. QoS parameters of a dynamic-rule are changed using the same Charging-Rule-Definition AVP. For dynamic rules, per-service QoS control maps to per-dynamic-rule QoS-control.

- For dynamic rules, service-level QoS control is supported using 3GPP-Standard AVPs. For hierarchical enforcement of FUP parameters for a packet matching a dynamic rule, charging-action level parameters are read from the dynamic rule itself. Hierarchical FUP enforcement will otherwise be similar to that for predefined rules.
- Dynamic rule has a name associated with it. This name can be added to statically (CLI) defined QoS-group-of-ruledefs. So, a dynamic rule can be configured to be part of a QoS-group-of-ruledefs. Multiple dynamic rules can be part of a QoS-group-of-ruledefs. QoS control for a QoS-group-of-ruledefs is transparently enforced, irrespective of whether constituent ruledefs are static, predefined, or dynamically installed.

Reporting Statistics and Usage to PCRF

The PCEF reports volume usage to the PCRF in CCR-U and RAR messages at the following events:

- Volume threshold for a charging-action is crossed, and an event trigger for that threshold breach is set by the PCRF.
- Volume threshold for a QoS-group-of-ruledefs is crossed, and an event trigger for that threshold breach is set by the PCRF.
- A QoS-group-of-ruledefs removed is the last using the shared volume-quota.

Monitoring and reporting of time-usage is not supported in this release. Also, packet drops due to enforcement of FUP-QoS parameters is not reported in CDR.

Statistics pertaining to FUP enforcement are available through Show CLI commands for all active sessions.

Delayed enforcement of bandwidth limiting

As per standards, the gateway enforces the bandwidth limiting based on the configured values. A configurable charging action is provided to allow the carrier to not enforce bandwidth limiting on a flow for a certain duration based on a configurable timer. The charging-action for a packet becomes known after rule-match. The **throttle-suppress** CLI command in the ACS Charging Action Configuration Mode can be configured to suppress bandwidth limiting. for the specified "timeout" period.

When the bandwidth limiting feature is turned on for a flow, the following types of bandwidth limiting will be suppressed:

- ITC bandwidth limiting
- Bearer level bandwidth limiting
- QoS-Group level bandwidth limiting
- APN-AMBR bandwidth limiting for downlink packets only

This section describes the bandwidth limiting behavior with various ECS functionalities:

- **Static and Predefined Ruledef/GoR/QGR:** Static and predefined rules / Group-of-ruledefs / QoS group-of-ruledefs are associated with a charging-action. Hence, all types of bandwidth limiting will be suppressed for the configured "timeout" period.
- **Dynamic Rules:** Dynamic-rules will not trigger throttle-suppress on a flow. However, on an ongoing throttle-suppress flow, dynamic-rule if hit, will suppress the dynamic-rule level bandwidth limiting.
- **TCP OOO Packets:** For TCP OOO packets, bearer bandwidth limiting is applied for packets that are transmitted without reordering. For a given flow, when the TCP OOO packets are processed within the time window of Suppress Start Time and Suppress End Time, the bearer bandwidth limiting will be suppressed.
- **ADC Rules:** For ADC Rules, all the rules across all bearers are matched. When an ADC rule (present on the same or different bearer) is matched and suppress bandwidth limiting is configured, the bandwidth limiting will be suppressed. For non-ADC Rules, the rule matching mechanism considered the rules present only on the bearer on which the flow is attached.
- **TRM:** Throttle-suppress is supported in TRM path.
- **Fast Path / Accelerated-ECS:** For Fast Path (FP) / Accelerated-ECS (A-ECS), bandwidth limiting will be suppressed in the same way as is for normal path.
- **DCCA Buffering:** DCCA buffering remains unaffected with the bandwidth limiting feature.

Session Control in ECS

In conjunction with the Cisco ASR 5500 chassis, the ECS provides a high-level network flow and bandwidth control mechanism in conjunction with the Session Control subsystem. ECS Session Control feature uses the interaction between SessMgr subsystem and Static Traffic Policy Infrastructure support of the chassis to provide an effective method to maximize network resource usage and enhancement of overall user experience.

This feature provides the following functionality:

- **Flow Control Functionality**—Provides the ability to define and manage the number of simultaneous IP-based sessions and/or the number of simultaneous instances of a particular application permitted for the subscriber.

  If a subscriber begins a packet data session and system is either pre-configured or receives a subscriber profile from the AAA server indicating the maximum amount of simultaneous flow for a subscriber or an application is allowed to initiate. If subscriber exceeds the limit of allowed number of flows for subscriber or type of application system blocks/redirect/discard/terminate the traffic.

  The following type of flow quotas are available for Flow Control Functionality:

  - **Subscriber-Level Session Quota**—Configurable on a per-rulebase basis
  - **Application-Level Session Quota**—Configurable on a per-charging-action basis

- **Bandwidth Control Functionality**—Allows the operator to apply rate limit to potentially bandwidth intensive and service disruptive applications.

  Using this feature the operator can police and prioritize subscribers' traffic to ensure that no single or group of subscribers' traffic negatively impacts another subscribers' traffic.

  For example, if a subscriber is running a peer-to-peer (P2P) file sharing program and the system is pre-configured to detect and limit the amount of bandwidth to the subscriber for P2P application. The system gets the quota limit for bandwidth from PDP context parameter or individual subscriber. If the subscriber's P2P traffic usage exceeds the pre-configured limit, the Session Control discards the traffic for this subscriber session.

  Session Control feature in ECS also provides the controls to police any traffic to/from a subscriber/application with the chassis.

Support for Splash Pages

The Splash Page support feature helps to distinguish HTTP traffic coming from mobile browsers and redirect the very first flow to a splash page whenever a subscriber attaches to the network. Splash page is the page of a website that the user sees first before being given the option to continue to the main content of the site.

When a subscriber attaches to the network, PCRF installs predefined rule/group of ruledefs towards PCEF to match mobile browser specific flows. On the first match of this rule/group of ruledef, redirect packet containing information of the welcome page where the flow needs to be redirected, is sent to UE and the first request gets terminated. Subsequently the predefined rule/group of ruledef from the list is removed and sends CCR-U with Charging Rule Report (CRR) AVP to PCRF for rule status. The existing **deactivate-predefined-rule** CLI command in the ACS Charging Action configuration mode is used to remove the matched predefined rule/group of ruledef.

In releases prior to 19.2, the redirection functionality was supported in the case when 80% threshold usage is reached for the subscriber and the same rule gets deactivated to ensure one time redirection for the subscriber.

In this release, the functionality is extended to redirect the first mobile browser flow to the splash page whenever subscriber attaches to the network. This feature now supports predefined group of ruledefs in addition to previously supported predefined rules. CLI and Statistics are enhanced to support HTTP-based rule matching in HTTP header.

Support for WebSocket Protocol Identification

This feature extends support for WebSocket Protocol identification.

The Websocket protocol is an independent TCP based protocol. A connection is identified as Websocket through the first HTTP Get Request header after the three way handshake. This packet includes an upgrade header (Upgrade: websocket) and other Websocket headers (Sec-Websocket-*) to upgrade HTTP to Websocket protocol. This helps operators to categorize Websocket traffic and apply different policies for such traffic.

A new CLI **websocket flow-detection** has been implemented at rulebase level to detect the Websocket protocol. The Websocket protocol identification can be enabled or disabled with the new CLI Websocket protocol.

For more information on these commands, see the *ACS Rulebase Configuration Mode Commands* chapter in the *Command Line Interface Reference*.

How it Works

This section describes how Websocket Protocol Identification feature works.

If the Websocket detection is enabled in the rulebase, the ECS parser looks for the following fields in the HTTP Get header fields **Host**, **Upgrade**, **Connection**, **Sec-Websocket-Key**, **Origin**, and **Sec-Websocket-Version**. If these headers are present, the TCP connection is upgraded to a Websocket connection. A ruledef can be defined to identify the HTTP GET request for the websocket and rate it in a certain way. The subsequent data that is transferred through the websocket is also billed the same way as the first packet.

Important You need to enable HTTP analysis to detect Websockets, and Websocket connections cannot be detected on secure-HTTP connections.

TCP Proxy

The TCP Proxy feature enables the ASR 5500 to function as a TCP proxy. TCP Proxy along with other capabilities enables the ASR 5500 to transparently split every TCP connection passing through it between sender and receiver hosts into two separate TCP connections, and relay data packets from the sender host to the receiver host via the split connections. Any application that needs to modify the TCP payload or manage TCP connection establishment and tear down uses the TCP Proxy feature.

TCP Proxy is enabled dynamically based on specified conditions. When TCP proxy is started dynamically on a flow, the original client (MS) first starts the TCP connection with the final server. ECS keeps on monitoring the connection. Based on any rule-match/charging-action, it may happen that the connection will be proxied automatically. This activity is transparent to original client and original server. The functional/charging behavior of ECS for that particular connection before the dynamic proxy is started is exactly same as when there is no proxy.

TCP Proxy impacts post-recovery behavior and the charging model. With TCP Proxy, whatever packets are received from either side is charged completely. The packets that are sent out from the ECS are not considered for charging. This approach is similar to the behavior of ECS without proxy.

The following packets will be charged at ECS:

- Uplink packets received at Gn interface
- Downlink packets received at Gi interface

The following packets will not be considered for charging:

- Uplink packets forwarded/sent out by ECS/Stack on the Gi interface
- Downlink packets forwarded/sent out by ECS/Stack on the Gn interface

ECS supports bulk statistics for the TCP Proxy feature. For details see the *ECS Schema Statistics* chapter of the *Statistics and Counters Reference*.

Flow Admission Control

The Flow Admission Control feature controls the number of flows required to be proxied. It restricts admission of new calls based on the current resource usage, thus preventing system hog and service degradation to existing subscribers.

The number of flows required to be proxied will greatly depend on the deployment scenario. Operators have the provision to configure an upper bound on the memory used by proxy flows. This is specified as a percentage of the Session Manager memory that may be used for proxy flows. When memory utilization by existing proxy flows reaches this value, no further flows will be proxied.

Operators can also set a limit on the number of flows that can be proxied per subscriber. This would exercise Fair Usage policy to a certain extent. No credit usage information by proxy is communicated to the Session Manager.

TCP Proxy Behavior and Limitations

The following are behavioral changes applicable to various ECS features and on other applications after enabling TCP Proxy.

- TCP Proxy Model: Without TCP Proxy, for a particular flow, there is only a single TCP connection between subscriber and server. ECS is a passive entity with respect to flows and the packets received on ingress were sent out on egress side (except in case where some specific actions like drop are configured through CLI) transparently.

  With TCP Proxy, a flow is split into two TCP connections — one between subscriber and proxy and another between chassis and server.

- Ingress Data Flow to Proxy: For all uplink packets, ingress flow involves completing the following steps and then enters the Gn side TCP IP Stack of proxy:

  1. IP Analysis (support for IP reassembly)
  2. Shallow/Deep Packet TCP Analysis (support for TCP OOO)
  3. Stateful Firewall Processing
  4. Application Detection and Control Processing

5. DPI Analysis
6. Charging Function (including rule-matching, generation of various records, and applying various configured actions)

For all downlink packets, ingress flow would involve completing the following steps, and then enters the Gi side TCP IP Stack of proxy:

1. IP Analysis (support for IP reassembly)
2. Network Address Translation Processing
3. Shallow/Deep Packet TCP Analysis (support for TCP OOO)
4. Stateful Firewall Processing
5. Application Detection and Control Processing
6. DPI Analysis
7. Charging Function (including rule-matching, generation of various records, and applying various configured actions)

- Egress Data Flow from Proxy: All egress data flow is generated at proxy stack. For uplink packets, egress data flow would involve the following and then are sent out of the chassis:

  1. IP Analysis
  2. Shallow/Deep Packet TCP Analysis
  3. Stateful Firewall processing
  4. Network Address Translation processing

  For downlink packets, egress data flow would involve the following and then are sent out of the chassis:

  1. IP Analysis
  2. Shallow/Deep Packet TCP Analysis
  3. Stateful Firewall processing

On enabling TCP Proxy the behavior of some ECS features will get affected. For flows on which TCP Proxy is enabled it is not necessary that all the packets going out of the Gn (or Gi) interface are the same (in terms of number, size, and order) as were on Gi (or Gn).

- IP Reassembly: If the fragments are successfully reassembled then DPI analysis is done on the reassembled packet.

  Without TCP Proxy, fragmented packets will go out on the other side. With TCP proxy, normal (non-fragmented) IP packets will go out on the other side (which will not be similar to the incoming fragmented packets).

  With or without TCP Proxy, if fragment reassembly was not successful, then all the fragments will be dropped except under the case where received fragments were sufficient enough to identify the 5-tuple TCP flow and the flow had TCP Proxy disabled on it.

- TCP OOO Processing: Without TCP Proxy if it is configured to send the TCP OOO packets out (as they come), without TCP proxy such packets were sent out. With TCP Proxy, OOO packets coming from one

side will go in-order on the other side. For proxied flows TCP OOO expiry timer will not be started and hence there will be no specific handling based on any such timeouts. Also, TCP OOO packets will not be sent to other side unless the packets are re-ordered.

In releases prior to 14.0, when TCP Out-of-Order (OOO) packets were received and when there was any error in buffering those packets at ECS due to memory allocation failure, these packets were marked as TCP error packets and the rule matching was done accordingly. These packets were also marked as TCP error packets when the reordering packet was not received before the OOO timeout.

In 14.0 and later releases, in the above mentioned scenarios, the packets are not considered as TCP error and the TCP error flag is not set for OOO packets. So, these packets will not match TCP error related ruledef but match other appropriate ruledefs.

If the customer has configured TCP error related rules, then OOO timeout failure packets and memory allocation failure packets will not match these rules now. It will match normal TCP rules.

- TCP Checksum Validation: Without TCP Proxy TCP Checksum validation is optional (configurable through "transport-layer-checksum verify-during-packet-inspection tcp" CLI command). With TCP Proxy TCP checksum is automatically done irrespective of whether the CLI command is configured or not. If the checksum validation fails, the packet is not processed further and so it does not go for application layer analysis.
- TCP Reset Packet Validation: Without TCP Proxy TCP reset packet is not validated for Seq and ACK number present in the segment and the flow is cleared immediately.

  With TCP Proxy TCP Reset packet validation is done. The flow will be cleared only if a valid TCP Reset segment is arrived. This validation is not configurable.
- TCP Timestamp (PAWS) Validation: Without TCP Proxy timestamp verification is not performed and even if there is any timestamp error, the packet is processed normally and goes for further analysis and rule-matching.

  With TCP Proxy if the connection is in established state, timestamp validation for packets is performed. If TCP timestamp is less than the previous timestamp, the packet is marked TCP error packet and is dropped. The packet is not analyzed further and not forwarded ahead. This packet should match TCP error rule (if configured). This validation is not configurable.
- TCP Error Packets: Without TCP Proxy ECS being a passive entity, most of the errors (unless configured otherwise) were ignored while parsing packets at TCP analyzer and were allowed to pass through. With TCP Proxy TCP error packets are dropped by Gi and Gn side TCP IP stack. However, since the ECS processing is already done before giving the packet to the stack, these packets are charged but not sent out by proxy on the other end.
- Policy Server Interaction (Gx): With TCP Proxy, application of policy function occurs on two separate TCP connections (non-proxy processed packets on Gn/Gi). Only external packets (the ones received from Radio and Internet) will be subject to policy enforcement at the box. This does not have any functional impact.
- Credit Control Interaction (Gy): With TCP Proxy, application of Credit Control function occur on two separate TCP connections (non-proxy processed packets on Gn/Gi). Only external packets (the ones received from Radio and Internet) will be subject to credit control at the box. This does not have any functional impact.
- DPI Analyzer: With TCP Proxy, application of DPI analyzer occurs on two separate TCP connections (non-proxy processed packets on Gn/Gi). Only external packets (the ones received from Radio and

Internet) will be subject to DPI analyzer at the chassis. Any passive analyzer in the path would be buffering packet using the existing ECS infrastructure.

- ITC/BW Control: With TCP Proxy, only incoming traffic is dropped based on bandwidth calculation on ingress side packets. The BW calculation and dropping of packet is be done before sending packet to ingress TCP IP Stack. ToS and DSCP marking will be on flow level. The ToS and DSCP marking can be done only once for whole flow and once the ToS is marked for any packet either due to "ip tos" CLI command configured in the charging action or due to ITC/BW control, it will remain same for the whole flow.
- Next Hop and VLAN-ID: Without TCP Proxy nexthop feature is supported per packet, that is nexthop address can be changed for each and every packet of the flow depending on the configuration in the charging action. With TCP Proxy only flow-level next-hop will be supported. So, once the nexthop address is changed for any packet of the flow, it will remain same for the complete flow. The same is the case for VLAN-ID.
- TCP state based rules: Without TCP Proxy there is only one TCP connection for a flow and the TCP state based rules match to state of subscriber stack. With TCP Proxy there are two separate connections when TCP proxy is enabled. TCP state ("tcp state" and "tcp previous-state") based rules will match to MS state on egress side. Two new rules (tcp proxy-state and tcp proxy-prev-state) have been added to support the existing cases (of TCP state based rules). "tcp proxy-state" and "tcp proxy-prev-state" are the state of the embedded proxy server, that is the proxy ingress-side. These rules will not be applicable if proxy is not enabled.

  Using both "tcp state" and "tcp proxy-state" in the same ruledef is allowed. If proxy is enabled, they would map to Gi-side and Gn-side, respectively. If TCP Proxy is not enabled, the "tcp proxy-state" and "tcp proxy-prev-state" rules will not be matched because proxy-state will not be applicable.

  Since TCP state and previous-state rules are now matched based on state on Gi side connection, ECS will not be able to support all the existing use-cases with the existing configuration. New ruledefs based on the new rules (tcp proxy-state and tcp proxy-prev-state) need to be configured to support existing use cases. Note that even by configuring using new rules; all use-cases may not be supported. For example, detection of transition from TIME-WAIT to CLOSED state is not possible now.
- TCP MSS: TCP IP Stack always inserts MSS Field in the header. This causes difference in MSS insertion behavior with and without TCP Proxy.
  - TCP CFG MSS limit-if-present: If incoming SYN has MSS option present, in outgoing SYN and SYN-ACK MSS value is limited to configured MSS value (CFG MSS)
  - TCP CFG MSS add-if-not-present: If incoming SYN does not have MSS option present, in outgoing SYN and SYN-ACK MSS configured MSS value is inserted (CFG MSS)
  - TCP CFG MSS limit-if-present add-if-not-present: If incoming SYN has MSS option present, in outgoing SYN and SYN-ACK MSS value is limited to configured MSS value (CFG MSS), OR if incoming SYN does not have MSS option present, in outgoing SYN and SYN-ACK MSS configured MSS value is inserted (CFG MSS).
- Flow Discard: Flow discard occurring on ingress/egress path of TCP Proxy would be relying on TCP-based retransmissions. Any discard by payload domain applications would result in data integrity issues as this might be charged already and it may not be possible to exclude packet. So it is recommended that applications in payload domain (like dynamic CF, CAE readdressing) should not be configured to drop packets. For example, dynamic content filtering should not be configured with drop action. If drop is absolutely necessary, it is better to use terminate action.

- DSCP/IP TOS Marking: Without TCP Proxy DSCP/IP TOS marking is supported per packet, that is IP TOS can be changed for each and every packet of the flow separately based on the configuration. With TCP Proxy flow-level DSCP/IP TOS marking is supported. So, once the IP TOS value is changed for any packet of the flow, it will remain same for the complete flow.
- Redundancy Support (Session Recovery and ICSR): Without TCP Proxy after recovery, non-syn flows are not reset. With TCP Proxy session recovery checkpointing is bypassing any proxied flows (currently on NAT flows support recovery of flows). If any flow is proxied for a subscriber, after recovery (session recovery or ICSR), if any non-syn packet is received for that subscriber, ECS sends a RESET to the sender. So, all the old flows will be RESET after recovery.
- Charging Function: Application of charging function would occur on two separate TCP connections (non proxy processed packets on Gn/Gi). Only external packets (the ones received from Radio and Internet) shall be subject to Policy enforcement at the box. Offline charging records generated at charging function would pertain to different connections hence.

Dynamic Disabling of TCP Proxy

TCP proxy can be dynamically disabled to reduce the performance overhead on CPU and memory resources. This enables applications to use proxy only when required.

Dynamic disabling is achieved by merging the TCP connections. Before dynamic disabling occurs, the packets are added to a TCP stack with a full proxy connection. Once proxy is disabled dynamically, the TCP stack and proxy are removed from the data processing path and the packets are forwarded without buffering.

Disabling of TCP proxy dynamically occurs only after the following conditions are met:

- There is no data to be delivered by ECS to the peer.
- The flow control buffers do no contain any data.
- There is no data to be read by ECS.

Limitations for Dynamically Disabling TCP Proxy

This section lists known limitations to disabling TCP proxy dynamically:

- TCP proxy cannot be disabled when one end of the TCP supports time stamp and other does not.
- Dynamic disabling does not work when both sides of the TCP have different MSS negotiated.
- Toggling the proxy on the same connection might reduce TCP performance.
- TCP proxy can only be disabled when both ends of TCP are in connected states.
- Multiple connections (1:n) connections cannot be joined together.
- TCP proxy can only be disabled when the conditions outlined for dynamic disabling is achieved (when there is no unAcked data in the network).

Time and Flow-based Bearer Charging in ECS

ECS supports Time-based Charging (TBC) to charge customers on either actual consumed time or total session time usage during a subscriber session. TBC generates charging records based on the actual time difference between receiving the two packets, or by adding idle time when no packet flow occurs.

ECS also supports Flow-based Charging (FBC) based on flow category and type.

PDP context charging allows the system to collect charging information related to data volumes sent to and received by the MS. This collected information is categorized by the QoS applied to the PDP context. FBC

integrates a Tariff Plane Function (TPF) to the charging capabilities that categorize the PDP context data volume for specific service data flows.

Service data flows are defined by charging rules. The charging rules use protocol characteristics such as:

- IP address
- TCP port
- Direction of flow
- Number of flows across system
- Number of flows of a particular type

FBC provides multiple service data flow counts, one each per defined service data flow. When FBC is configured in the ECS, PDP context online charging is achieved by FBC online charging using only the wildcard service data flow.

When further service data flows are specified, traffic is categorized, and counted, according to the service data flow specification. You can apply wildcard to service data flow that do not match any of the specific service data flows.

The following are the chargeable events for FBC:

- **Start of PDP context**—Upon encountering this event, a Credit Control Request (CCR) starts, indicating the start of the PDP context, is sent towards the Online Charging Service. The data volume is captured per service data flow for the PDP context.
- **Start of service data flow**—An interim CCR is generated for the PDP context, indicating the start of a new service data flow, and a new volume count for this service data flow is started.
- **Termination of service data flow**—The service data flow volume counter is closed, and an interim CCR is generated towards the Online Charging Service, indicating the end of the service data flow and the final volume count for this service data flow.
- **End of PDP context**—Upon encountering this event, a CCR stop, indicating the end of the PDP context, is sent towards the Online Charging Service together with the final volume counts for the PDP context and all service data flows.
- **Expiration of an operator configured time limit per PDP context**—This event triggers the emission of an interim CCR, indicating the elapsed time and the accrued data volume for the PDP context since the last report.
- **Expiration of an operator configured time limit per service data flow**—The service data flow volume counter is closed and an interim CCR is sent to the Online Charging Service, indicating the elapsed time and the accrued data volume since the last report for that service data flow. A new service data flow container is opened if the service data flow is still active.
- **Expiration of an operator configured data volume limit per PDP context**—This event triggers the emission of an interim CCR, indicating the elapsed time and the accrued data volume for the PDP context since the last report.
- **Expiration of an operator configured data volume limit per service data flow**—The service data flow volume counter is closed and an interim CCR is sent to the Online Charging Service, indicating the elapsed time and the accrued data volume since the last report for that service data flow. A new service data flow container is opened if the service data flow is still active.

- **Change of charging condition**—When QoS change, tariff time change are encountered, all current volume counts are captured and sent towards the Online Charging Service with an interim CCR. New volume counts for all active service data flows are started.
- **Administrative intervention** by user/service also force trigger a chargeable event.

The file naming convention for created xDRs (EDR/UDR/FDRs) are described in Impact on xDR File Naming, on page 54.

Time-of-Day Activation/Deactivation of Rules

Within a rulebase, ruledefs/groups-of-ruledefs are assigned priorities. When packets start arriving, as per the priority order, every ruledef/group-of-ruledefs in the rulebase is eligible for matching regardless of the packet arrival time. By default, the ruledefs/groups-of-ruledefs are active all the time.

The Time-of-Day Activation/Deactivation of Rules feature uses time definitions (timedefs) to activate/deactivate static ruledefs/groups-of-ruledefs such that they are available for rule matching only when they are active.

Important The time considered for timedef matching is the system's local time.

How the Time-of-Day Activation/Deactivation of Rules Feature Works

The following steps describe how the Time-of-Day Activation/Deactivation of Rules feature enables charging according to the time of the day/time:

Step 1 Timedefs are created/deleted in the ACS Configuration Mode.

A maximum of 10 timedefs can be created in an ECS service.

Step 2 Timedefs are configured in the ACS Timedef Configuration Mode. Within a timedef, timeslots specifying the day/time for activation/deactivation of rules are configured.

A maximum of 24 timeslots can be configured in a timedef.

Step 3 In the ACS Rulebase Configuration Mode, timedefs are associated with ruledefs /groups-of-ruledefs along with the charging action.

One timedef can be used with several ruledefs/group-of-ruledefs. If a ruledef/group-of-ruledefs does not have a timedef associated with it, it will always be considered as active.

Step 4 When a packet is received, and a ruledef/group-of-ruledefs is eligible for rule matching, if a timedef is associated with the ruledef/group-of-ruledefs, before rule matching, the packet-arrival time is compared with the timeslots configured in the timedef. If the packet arrived in any of the timeslots configured in the associated timedef, rule matching is undertaken, else the next ruledef/group-of-ruledefs is considered.

This release does not support configuring a timeslot for a specific date.

If, in a timeslot, only the time is specified, that timeslot will be applicable for all days.

If for a timeslot, "start time" > "end time", that rule will span the midnight. That is, that rule is considered to be active from the current day until the next day.

If for a timeslot, "start day" > "end day", that rule will span over the current week till the end day in the next week.

In the following cases a rule will be active all the time:

- A timedef is not configured in an action priority
- A timedef is configured in an action priority, but the named timedef is not defined
- A timedef is defined but with no timeslots

URL Filtering

The URL Filtering feature simplifies using rule definitions for URL detection.

The following configuration is currently used for hundreds of URLs:

```
ruledef HTTP://AB-WAP.YZ
      www url starts-with HTTP://CDAB-SUBS.OPERA-MINI.NET/HTTP://AB-WAP.YZ
      www url starts-with HTTP://AB-WAP.YZ
      multi-line-or all-lines
      exit
```

In the above ruledef:

- The HTTP request for the URL "http://ab-wap.yz" is first sent to a proxy "http://cdab-subs.opera-mini.net/".
- The URL "http://cdab-subs.opera-mini.net/" will be configured as a prefixed URL.

Prefixed URLs are URLs of the proxies. A packet can have a URL of the proxy and the actual URL contiguously. First a packet is searched for the presence of proxy URL. If the proxy URL is found, it is truncated from the parsed information and only the actual URL (that immediately follows it) is used for rule matching and EDR generation.

The group-of-ruledefs can have rules for URLs that need to be actually searched (URLs that immediately follow the proxy URLs). That is, the group-of-prefixed-URLs will have URLs that need to be truncated from the packet information for further ECS processing, whereas, the group-of-ruledefs will have rules that need to be actually searched for in the packet.

URLs that you expect to be prefixed to the actual URL can be grouped together in a group-of-prefixed-URLs. A maximum of 64 such groups can be configured. In each such group, URLs that need to be truncated from the URL contained in the packet are specified. Each group can have a maximum of 10 such prefixed URLs. By default, all group-of-prefixed-URLs are disabled.

In the ECS rulebase, you can enable/disable the group-of-prefixed-URLs to filter for prefixed URLs.

Important A prefixed URL can be detected and stripped if it is of the type "http://www.xyz.com/http://www.abc.com". Here, "http://www.xyz.com" will be stripped off. But in "http://www.xyz.com/www.abc.com", it cannot detect and strip off "http://www.xyz.com" as it looks for occurrence of "http" or "https" within the URL.

Accounting and Charging Interfaces

ECS supports different accounting and charging interfaces for prepaid and postpaid charging and record generation.

Important Some feature described in this section are licensed Cisco features. 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*.

Accounting Interfaces for Postpaid Service: ECS supports the following accounting interfaces for postpaid subscribers

- Remote Authentication Dial-In User Service (RADIUS) Interface
- GTPP Accounting Interface

Accounting and Charging Interface for Prepaid Service: ECS supports the following Credit Control Interfaces for prepaid subscribers

- RADIUS Prepaid Credit Control interface
- Diameter Prepaid Credit Control Application (DCCA) Gy Interface
- Diameter Gx interface

Charging Records in ECS: ECS provides the following charging records for postpaid and prepaid charging

- GGSN-Call Detail Records (G-CDRs)
- Enhanced GGSN-Call Detail Records (eG-CDRs)
- Event Detail Records (EDRs)
- Usage Detail Records (UDRs)

GTPP Accounting

ECS enables the collection of counters for different types of data traffic, and including that data in CDRs that is sent to a Charging Gateway Function (CGF).

For more information on GTPP accounting, if you are using StarOS 12.3 or an earlier release, refer to the *AAA and GTPP Interface Administration and Reference*. If you are using StarOS 14.0 or a later release, refer to the *GTPP Interface Administration and Reference*.

RADIUS Accounting and Credit Control

The Remote Authentication Dial-In User Service (RADIUS) interface in ECS is used for the following purposes:

- **Subscriber Category Request**—ECS obtains the subscriber category from the AAA server (either prepaid or postpaid) when a new data session is detected. The AAA server used for the subscriber category request can be different from the AAA server used for service authorization and accounting.
- **Service Access Authorization**—ECS requests access authorization for a specific subscriber and a newly detected data session. The AAA server is the access Policy Decision Point and the ECS the Policy Enforcement Point.

- **On-line Service Accounting (Prepaid)**—ECS reports service usage to the AAA server. The AAA server acts as a prepaid control point and the ECS as the client. Accounting can be applied to a full prepaid implementation or just to keep ECS updated of the balance level and trigger a redirection if the subscriber balance reaches a low level.

Diameter Accounting and Credit Control

The Diameter Credit Control Application (DCCA) is used to implement real-time online or offline charging and credit control for a variety of services, such as network access, messaging services, and download services.

In addition to being a general solution for real-time cost and credit control, DCCA includes these features:

- **Real-time Rate Service Information:** DCCA can verify when end subscribers' accounts are exhausted or expired; or deny additional chargeable events.
- **Support for Multiple Services:** DCCA supports the usage of multiple services within one subscriber session. Multiple service support includes:
  - The ability to identify and process the service or group of services that are subject to different cost structures.
  - Independent credit control of multiple services in a single credit control sub-session.

Gx Interface Support

The Gx interface is used in IMS deployment in GPRS/UMTS networks. Gx interface support on the system enables wireless operators to intelligently charge the services accessed depending on the service type and parameters with rules. It also provides support for IP Multimedia Subsystem (IMS) authorization in a GGSN service. The goal of the Gx interface is to provide network-based QoS control as well as dynamic charging rules on a per bearer basis for an individual subscriber. The Gx interface is in particular needed to control and charge multimedia applications.

Important For more information on Gx interface support, see the *Gx Interface Support* appendix in the administration guide for the product that you are deploying.

Gy Interface Support

The Gy interface provides a standardized Diameter interface for real-time content-based charging of data services. It is based on the 3GPP standards and relies on quota allocation.

It provides an online charging interface that works with the ECS Deep Packet Inspection feature. With Gy, customer traffic can be gated and billed in an "online" or "prepaid" style. Both time- and volume-based charging models are supported. In all these models, differentiated rates can be applied to different services based on shallow or deep-packet inspection.

Gy is a Diameter interface. As such, it is implemented atop, and inherits features from, the Diameter Base Protocol. The system supports the applicable base network and application features, including directly connected, relayed or proxied DCCA servers using TLS or plain text TCP.

In the simplest possible installation, the system will exchange Gy Diameter messages over Diameter TCP links between itself and one "prepay" server. For a more robust installation, multiple servers would be used.

These servers may optionally share or mirror a single quota database so as to support Gy session failover from one server to the other. For a more scalable installation, a layer of proxies or other Diameter agents can be introduced to provide features such as multi-path message routing or message and session redirection features.

The Diameter Credit Control Application (DCCA) which resides as part of the ECS manages the credit and quota for a subscriber.

Important For more information on Gy interface support, see the *Gy Interface Support* appendix in the administration guide for the product that you are deploying.

Event Detail Records (EDRs)

Event Detail Records (EDRs) are usage records with support to configure content information, format, and generation triggers by the system administrative user.

EDRs are generated according to explicit action statements in rule commands. Several different EDR schema types, each composed of a series of analyzer parameter names, are specified in EDR. EDRs are written at the time of each event in CSV format. EDRs are stored in timestamped files that can be downloaded via SFTP from the configured context.

EDRs are generated on per flow basis, and as such they catch whatever bytes get transmitted over that flow including retransmitted.

EDR format

The EDRs can be generated in comma separated values (CSV) format as defined in the traffic analysis rules.

Important In EDRs, the maximum field length for normal and escaped strings is 127 characters. If a field's value is greater than 127 characters, in the EDR it is truncated to 127 characters. In 15 and later releases, an optional filter "length" is supported for HTTP URL and User-Agent fields which when added will allow the user to configure length from 1 to 255 for these fields in EDRs. For more information, see the **rule-variable** command in the *Command Line Interface Reference*. In 17 and later releases, the allowed length for HTTP URL is 1 through 4095. For more information, see the **rule-variable** command in the *Command Line Interface Reference*.

In 21.1 and later releases, a maximum of 75 EDR attribute fields can be configured in an EDR record. The limit is expanded from 50 fields up to 75 fields.

Flow-overflow EDR

Flow-overflow EDR or Summary FDR is a feature to count the data bytes from the subscriber that are missed due to various reasons in ECS.

In case any condition that affects the callline (FLOW end-condition like hagr, handoff) occurs, flow-overflow EDR generation is enabled, an extra EDR is generated. Based on how many bytes/packets were transferred from/to the subscriber for which ECS did not allocate data session. This byte/packet count is reflected in that extra EDR. This extra EDR is nothing but "flow-overflow" EDR or Summary FDR.

The extra EDR is generated if all of the following is true:

- Subscriber affecting condition occurs (session-end, hand-off, hagr)
- Flow-overflow EDR generation is enabled

- EDR generation on session-end, hand-off or hagr is enabled
- Number of bytes/packets for flow-overflow EDR is non-zero.

The bytes/packet count will be printed as a part of "sn-volume-amt" attribute in the EDR. Hence, this attribute must be configured in the EDR format.

EDR Generation in Flow-end and Transaction Complete Scenarios with sn-volume Fields

"sn-volume-amt" counters will be re-initialized only when the fields are populated in EDRs. For example, consider the following two EDR formats:

```
edr-format edr1
     rule-variable http url priority 10
     attribute sn-volume-amt ip bytes uplink priority 500
     attribute sn-volume-amt ip bytes downlink priority 510
     attribute sn-volume-amt ip pkts uplink priority 520
     attribute sn-volume-amt ip pkts downlink priority 530
     attribute sn-app-protocol priority 1000
     exit
edr-format edr2
     rule-variable http url priority 10
     attribute sn-app-protocol priority 1000
     exit
```

"sn-volume-amt counters" will be re-initialized only if these fields are populated in the EDRs. Now if edr2 is generated, these counters will not be re-initialized. These will be re-initialized only when edr1 is generated. Also, note that only those counters will be re-initialized which are populated in EDR. For example, in the following EDR format:

```
edr-format edr3
     rule-variable http url priority 10
     attribute sn-volume-amt ip bytes uplink priority 500
     attribute sn-volume-amt ip bytes downlink priority 510
     attribute sn-app-protocol priority 1000
     exit
```

If edr3 is generated, only uplink bytes and downlink bytes counter will be re-initialized and uplink packets and downlink packets will contain the previous values till these fields are populated (say when edr1 is generated).

For the voice call duration for SIP reporting requirements, ECS SIP analyzer keeps timestamp of the first INVITE that it sees. It also keeps a timestamp when it sees a 200 OK for a BYE. When this 200 OK for a BYE is seen, SIP analyzer triggers creation of an EDR of type ACS_EDR_VOIP_CALL_END_EVENT. This will also be triggered at the time of SIP flow termination if no 200 OK for BYE is seen. In that case, the last packet time will be used in place of the 200 OK BYE timestamp. The EDR generation logic calculates the call duration based on the INVITE and end timestamps, it also accesses the child RTP/RTCP flows to calculate the combined uplink/downlink bytes/packets counts and sets them in the appropriate fields.

The HTTP URL and HTTP User Agent can be configured in the EDR in two methods:

- Default, where no length is defined in the EDR configuration. The following is an example EDR configuration where the corresponding variable name for HTTP URL/HTTP User Agent will be "http-url" and "http-user-agent" respectively.

  ```
  rule-variable http url priority 2
  ```

  ```
  rule-variable http user-agent priority 2
  ```

- Length for HTTP URL/HTTP User Agent is defined in the EDR configuration. The following is an example EDR configuration where the corresponding variable name for HTTP URL/HTTP User Agent

will be "http-url-2000" and "http-user-agent-100" respectively. The length for HTTP URL is any number between 1 and 4095 (xyz) that translates into a EDR attribute "http-url-xyz", where xyz is the number entered by the user. The length for HTTP User Agent is any number between 1 and 255 (xyz) that translates into a EDR attribute "http-user-agent-xyz", where xyz is the number entered by the user.

```
rule-variable http url length 2000 priority 4
rule-variable http user-agent length 100 priority 4
```

Usage Detail Records (UDRs)

Usage Detail Records (UDRs) contain accounting information based on usage of service by a specific mobile subscriber. UDRs are generated based on the content-id for the subscriber, which is part of charging action. The fields required as part of usage data records are configurable and stored in the System Configuration Task (SCT).

UDRs are generated on any trigger of time threshold, volume threshold, handoffs, and call termination. If any of the events occur then the UDR subsystem generates UDRs for each content ID and sends to the CDR module for storage.

UDR format

The UDRs are generated in Comma Separated Values (CSV) format as defined in the traffic analysis rules.

Charging Methods and Interfaces

This section provides an overview of the Charging methods and interfaces.

Prepaid Credit Control

Prepaid billing operates on a per content-type basis. Individual content-types are marked for prepaid treatment. A match on a traffic analysis rule that has a prepaid-type content triggers prepaid charging management.

In prepaid charging, ECS performs the metering function. Credits are deducted in real time from an account balance or quota. A fixed quota is reserved from the account balance and given to the system by a prepaid rating and charging server, which interfaces with an external billing system platform. The system deducts volume from the quota according to the traffic analysis rules. When the subscriber's quota gets to the threshold level specified by the prepaid rating and charging server, system sends a new access request message to the server and server updates the subscriber's quota. The charging server is also updated at the end of the call.

ECS supports the following credit control applications for prepaid charging:

- **RADIUS Credit Control Application**—RADIUS is used as the interface between ECS and the prepaid charging server. The RADIUS Prepaid feature of ECS is separate to the system-level Prepaid Billing Support and that is covered under a different license key.
- **Diameter Credit Control Application**—The Diameter Credit Control Application (DCCA) is used to implement real-time credit control for a variety of services, such as networks access, messaging services, and download services.

In addition to being a general solution for real-time cost and credit control, DCCA includes the following features:

- **Real-time Rate Service Information**—DCCA can verify when end subscribers' accounts are exhausted or expired; or deny additional chargeable events.

- **Support for Multiple Services**—DCCA supports the usage of multiple services within one subscriber session. Multiple service support includes:
  - The ability to identify and process the service or group of services that are subject to different cost structures.
  - Independent credit control of multiple services in a single credit control sub-session.

Postpaid

In a postpaid environment, the subscribers pay after use of the service. AAA/RADIUS server is responsible for authorizing network nodes to grant access to the user, and the CDR system generates G-CDRs/eG-CDRs/EDRs/UDRs for billing information on pre-defined intervals of volume or per time.

Important G-CDRs and eG-CDRs are only available in UMTS networks.

ECS also supports FBC and TBC methods for postpaid billing. For more information on FBC and TBC in ECS, see Time and Flow-based Bearer Charging in ECS, on page 37.

Prepaid Billing in ECS

In a prepaid environment, the subscribers pay for service prior to use. While the subscriber is using the service, credit is deducted from subscriber's account until it is exhausted or call ends. The prepaid charging server is responsible for authorizing network nodes to grant access to the user, as well as grant quotas for either time connected or volume used. It is up to the network node to track the quota use, and when these use quotas run low, the network node sends a request to the prepaid server for more quota.

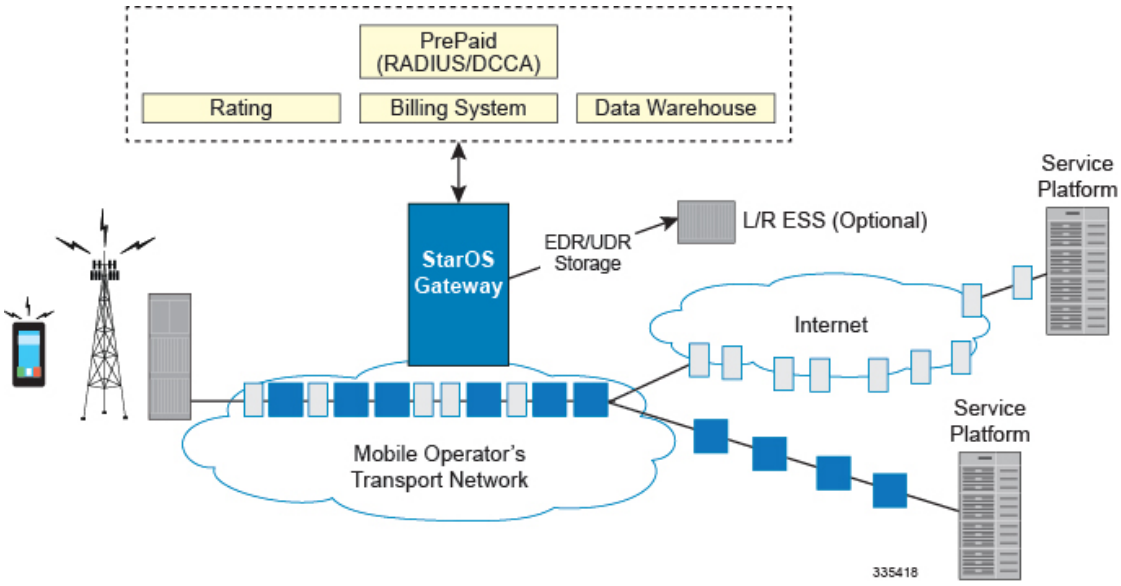
If the user has not used up the purchased credit, the server grants quota and if no credit is available to the subscriber the call will be disconnected. ECS and DCCA manage this functionality by providing the ability to set up quotas for different services.

Prepaid quota in ECS is implemented using RADIUS and DCCA as shown in the following figure.

How ECS Prepaid Billing Works

The following figure illustrates a typical prepaid billing environment with system running ECS.

Figure 7: Prepaid Billing Scenario with ECS



Credit Control Application (CCA) in ECS

This section describes the credit control application that is used to implement real-time credit-control for a variety of end user services such as network access, Session Initiation Protocol (SIP) services, messaging services, download services, and so on. It provides a general solution to the real-time cost and credit control.

CCA with RADIUS or Diameter interface uses a mechanism to allow the user to be informed of the charges to be levied for a requested service. In addition, there are services such as gaming and advertising that may debit from a user account.

How Credit Control Application (CCA) Works for Prepaid Billing

The following figure and steps describe how CCA works with in a GPRS/UMTS or CDMA-2000 network for prepaid billing.

Figure 8: Prepaid Charging in GPRS/UMTS/CDMA-2000 Networks

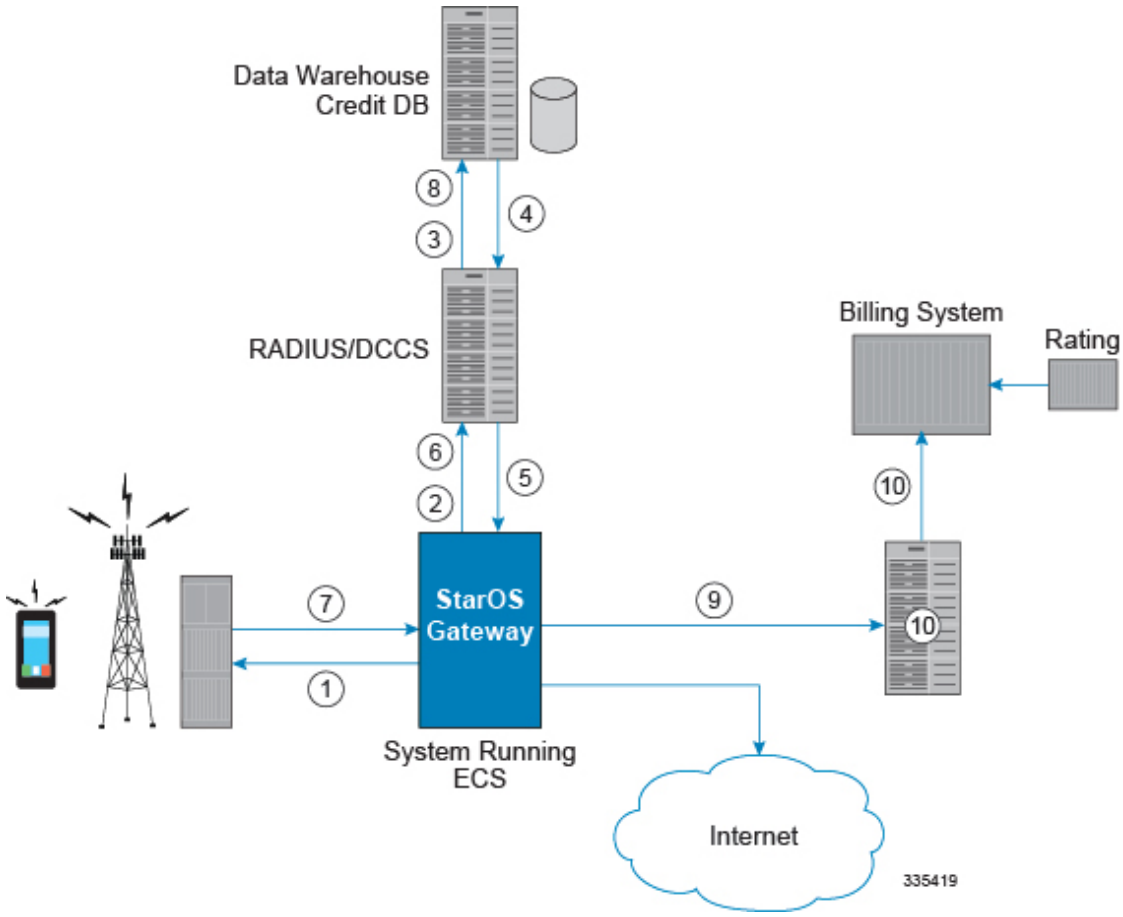


Table 2: Prepaid Charging in GPRS/UMTS/CDMA-2000 Networks

Step No.	Description
1	Subscriber session starts.
2	System sends request to CCA for subscriber's quota.
3	CCA sends request to Data Warehouse (DW) credit quota for subscriber.
4	Credit Database in DW sends pre-configured amount of usage limit from subscriber's quota to CCA. To reduce the need for multiple requests during subscriber's session configured amount of usage limit a major part of available credit quota for subscriber is set.
5	CCA sends the amount of quota required to fulfill the subscriber's initial requirement to the system.

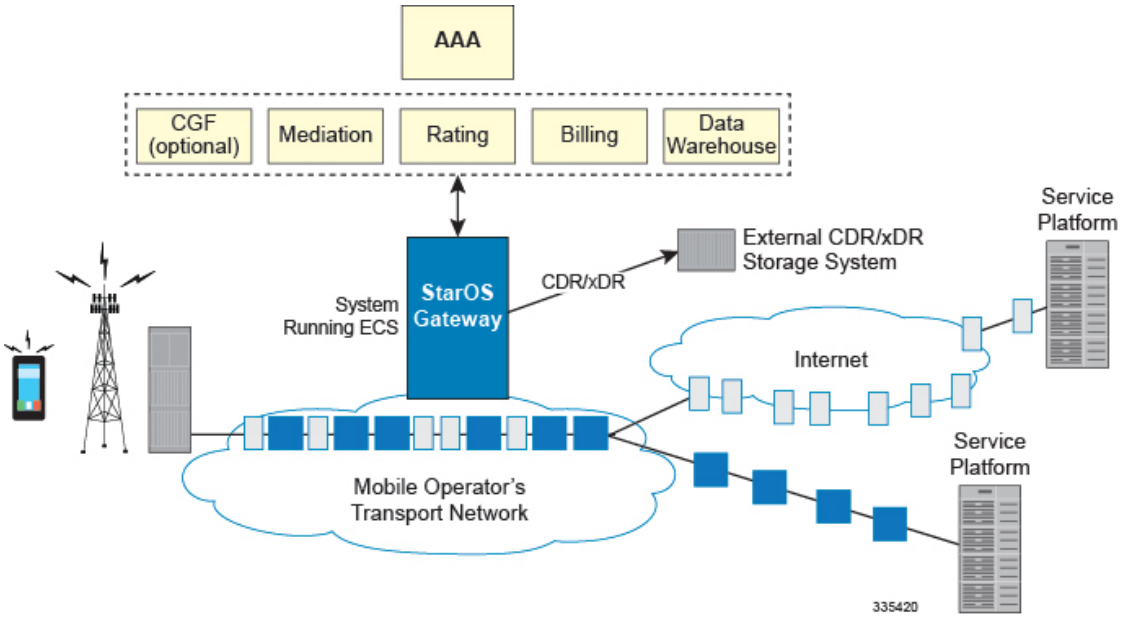
Step No.	Description
6	When the initial amount of quota runs out, system sends another request to the CCA and the CCA sends another portion of available credit quota.
7	Subscriber session ends after either quota exhausts for subscriber or subscriber terminates the session.
8	CCA returns unused quota to DW for update to subscribers Credit DB.
9	EDRs and UDRs are periodically SFTPd from system memory to the external storage, if deployed or to billing system directly as they are generated. Or, if configured, pushed to the external storage at user-configurable intervals.
10	The external storage periodically sends records to the billing system or charging reporting and analysis system.

Postpaid Billing in ECS

This section describes the postpaid billing that is used to implement offline billing processing for a variety of end user services.

The following figure shows a typical deployment of ECS for postpaid billing system.

Figure 9: Postpaid Billing System Scenario with ECS



How ECS Postpaid Billing Works

This section describes how the ECS postpaid billing works in the GPRS/UMTS and CDMA-2000 Networks.

ECS Postpaid Billing in GPRS/UMTS Networks

The following figure and steps describe how ECS works in a GPRS/UMTS network for postpaid billing.

Figure 10: Postpaid Billing with ECS in GPRS/UMTS Network

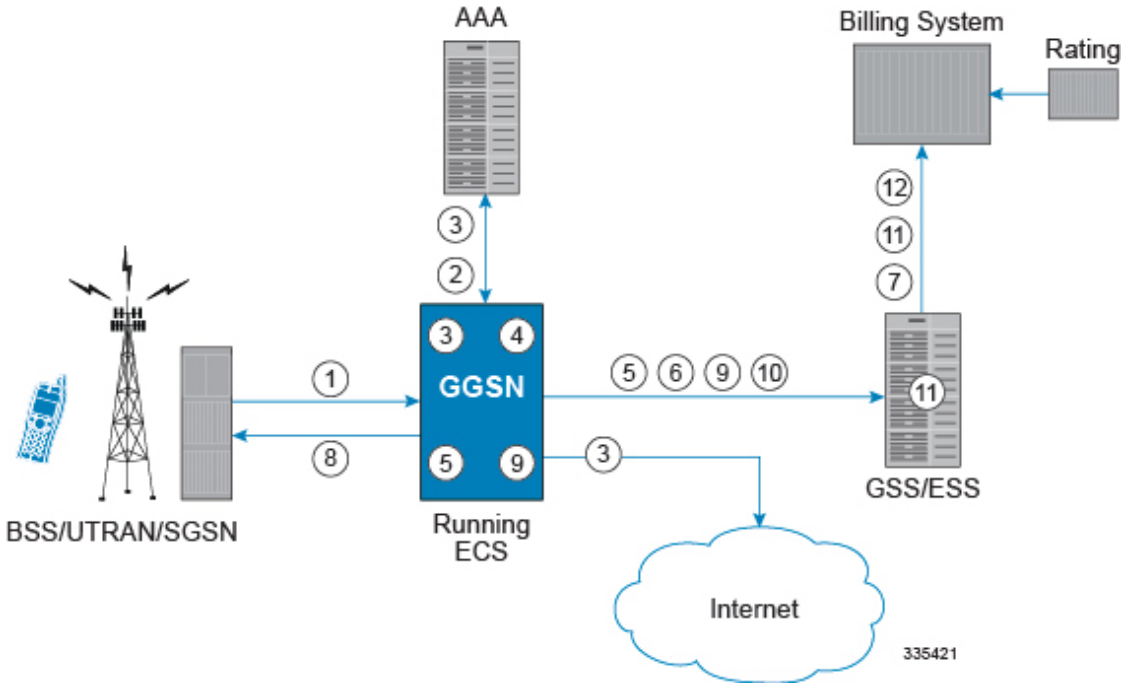


Table 3: Postpaid Billing with ECS in GPRS/UMTS Network

Step No.	Description
1	The subscriber initiates the session.
2	After subscriber authentication and authorization, the system starts the session.
3	Data packet flow and accounting starts.
4	System periodically generates xDRs and stores them to the system memory.
5	System generates G-CDRs/eG-CDRs and sends them to billing system as they are generated.
6	The billing system picks up the CDR files periodically.
7	Subscriber session ends after subscriber terminates the session.

Step No.	Description
8	The system stores the last of the xDRs to the system memory and final xDRs are SFTPd from system memory to external storage, if deployed or to billing system directly.
9	System sends the last of the G-CDRs/eG-CDRs to the billing system.
10	File Generation Utility, FileGen in external storage periodically runs to generate G-CDRs/eG-CDRs files for billing system and send them to the billing system.
11	The billing system picks up the xDR files from the external storage periodically.

ECS Postpaid Billing in CDMA-2000 Networks

The following figure and steps describe how ECS works within a CDMA-2000 network for postpaid billing.

Figure 11: Postpaid Billing with ECS in CDMA-2000 Network

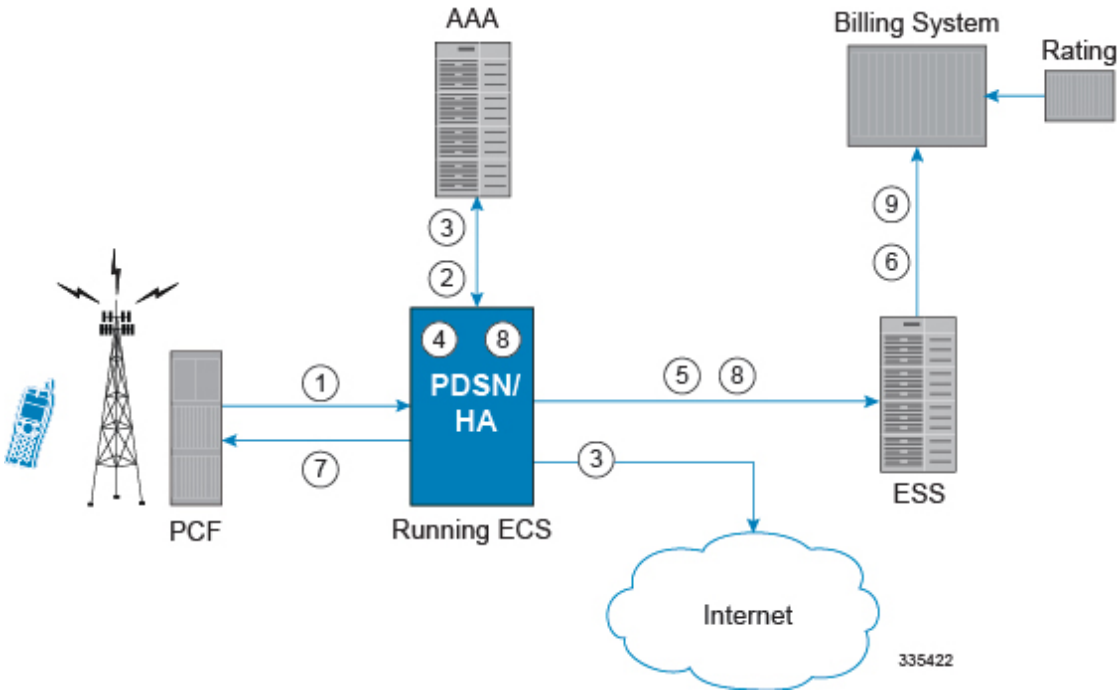


Table 4: Postpaid Billing with ECS in CDMA-2000 Network

Step No.	Description
1	The subscriber initiates the session.
2	After subscriber authentication and authorization, the system starts the session.

Step No.	Description
3	Data packet flow and accounting starts.
4	System periodically generates xDRs and stores them to the system memory.
5	EDRs/UDRs are periodically SFTPd from system memory to external storage, if deployed or to billing system directly as they are generated.
6	The billing system picks up the xDR files from the external storage periodically.
7	Subscriber session ends after subscriber terminates the session.
8	The system stores the last of the xDRs to the system memory and final xDRs are SFTPd from system memory to the external storage, if deployed or to billing system directly.
9	The external storage finally sends xDRs to the billing system.

External Storage

Important For information on availability/support for external storage, contact your Cisco account representative.

The external storage is a high availability, fault tolerant, redundant solution for short-term storage of files containing detail records (UDRs/EDRs/FDRs (xDRs)). To avoid loss of xDRs on the chassis due to overwriting, deletion, or unforeseen events such as power or network failure or unplanned chassis switchover, xDRs are off-loaded to external storage for storage and analysis to avoid loss of charging and network analysis information contained in the xDRs.

The xDR files can be pulled by the external storage from the chassis, or the chassis can push the xDR files to the external storage using SFTP protocol. In the Push mode, the external storage URL to which the CDR files need to be transferred to is specified. The configuration allows a primary and a secondary server to be configured. Configuring the secondary server is optional. Whenever a file transfer to the primary server fails for four consecutive times, the files will be transferred to the secondary server. The transfer will switch back to the original primary server when:

- Four consecutive transfer failures to the secondary server occur
- After switching from the primary server, 30 minutes elapses

In the push transfer mode, the following can be configured:

- Transfer interval—A time interval, in seconds, after which the CDRs are pushed to the configured IP periodically. All the files that are completed before the PUSH timer expires are pushed.

- Remove file after transfer—An option to keep or remove the CDR files on the hard disk after they are transferred to the external storage successfully.

The system running with ECS stores xDRs on an external storage, and the billing system collects the xDRs from the external storage and correlates them with the AAA accounting messages using 3GPP2-Correlation-IDs (for PDSN) or Charging IDs (for GGSN).

System Resource Allocation

ECS does not require manual resource allocation. The ECS subsystem automatically allocates the resources when ECS is enabled on the chassis. ECS must be enabled on the chassis before configuring services.

Redundancy Support in ECS

This section describes the redundancy support available in ECS to recover user sessions and charging records in the event of software/hardware failure.

Caution Persistent data flows are NOT recoverable during session recovery.

Important Redundancy is not available in the current version of the Cisco® XT2 platform.

Intra-chassis Session Recovery Interoperability

Intra-chassis session recovery is coupled with SessMgr recovery procedures.

Intra-chassis session recovery support is achieved by mirroring the SessMgr and AAAMgr processes. The SessMgrs are paired one-to-one with the AAAMgrs. The SessMgr sends checkpointed session information to the AAAMgr. ECS recovery is accomplished using this checkpointed information.

Important In order for session recovery to work there should be at least four packet processing cards, one standby and three active. Per active CPU with active SessMgrs, there is one standby SessMgr, and on the standby CPU, the same number of standby SessMgrs as the active SessMgrs in the active CPU.

There are two modes of session recovery, one from task failure and another on failure of CPU or packet processing card.

Recovery from Task Failure

When a SessMgr failure occurs, recovery is performed using the mirrored "standby-mode" SessMgr task running on the active packet processing card. The "standby-mode" task is renamed, made active, and is then populated using checkpointed session information from the AAAMgr task. A new "standby-mode" SessMgr is created.

Recovery from CPU or Packet Processing Card Failure

When a PSC, PSC2, or PPC hardware failure occurs, or when a planned packet processing card migration fails, the standby packet processing card is made active and the "standby-mode" SessMgr and AAAMgr tasks on the newly activated packet processing card perform session recovery.

Inter-chassis Session Recovery Interoperability

The system supports the simultaneous use of ECS and the Inter-chassis Session Recovery feature. When both features are enabled, ECS session information is regularly checkpointed from the active chassis to the standby as part of normal Service Redundancy Protocol processes. For more information on the Inter-chassis Session Recovery feature, refer to the *System Administration Guide*.

In the event of a manual switchover, there is no loss of accounting information. All xDR data from the active chassis is moved to a customer-configured external storage before switching over to the standby. This data can be retrieved at a later time. Upon completion of the switchover, the ECS sessions are maintained and the "now-active" chassis recreates all of the session state information including the generation of new xDRs.

In the event of an unplanned switchover, all accounting data that has not been written to the external storage is lost. Note that either the external storage can pull the xDR data from the chassis, or the chassis can push the xDR files to a configured external storage at user-configured intervals. For more information, see External Storage, on page 52. Upon completion of switchover, the ECS sessions are maintained and the "now-active" chassis recreates all of the session state information including the generation of new xDRs.

Regardless of the type of switchover that occurred, the names of the new xDR files will be different from those stored in the /records directory of packet processing card RAM on the "now-standby" chassis. Also, in addition to the file name, the content of many of the fields within the xDR files created by the "now-active" chassis will be different. ECS manages this impact with recovery mechanism. For more information on the differences and how to correlate the two files and other recovery information, see Impact on xDR File Naming, on page 54.

Inter-chassis Session Recovery Architecture

Inter-chassis redundancy in ECS uses Flow Detail Records (FDRs) and UDRs to manage the switchover between Active-Standby system. xDRs are moved between redundant external storage server and Active-Standby systems.

Session Recovery Improvements

In StarOS releases prior to 14.0, there were only 10 PCC rules that were recovered per bearer in the event of a session manager crash. In 14.0 and later releases, this limit has been increased to 24. That is, up to 24 PCC rules can be recovered post ICSR.

With the increase in the limit of PCC rules that can be recovered, the rules are not lost and hence the charging applied to the end users are not impacted.

Impact on xDR File Naming

The xDR file name is limited to 256 characters with the following syntax:

basename_ChargSvcName_ timestamp_SeqNumResetIndicator_FileSeqNumber

where:

- *basename*—A global configurable text string that is unique per system that uniquely identifies the global location of the system running ECS.
- *ChargSvcName*—A system context-based configurable text string that uniquely identifies a specific context-based charging service.
- *timestamp*—Date and time at the instance of file creation. Date and time in the form of "MMDDYYYYHHmmSS" where HH is a 24-hour value from 00-23.
- *SeqNumResetIndicator*—A one-byte counter used to discern the potential for duplicated FileSeqNumber with a range of 0 through 255, which is incremented by a value of 1 for the following conditions:
  - Failure of an ECS software process on an individual packet processing card
  - Failure of a system such that a second system takes over according to the Inter-chassis Session Recovery feature
  - File Sequence Number (FileSeqNumber) rollover from 999999999 to 0
- *FileSeqNumber*—Unique file sequence number for the file with nine-digit integer having range from 000000000 to 999999999. It is unique on each system.

With inter-chassis session recovery, only the first two fields in the xDR file names remain consistent between the active and standby chassis as these are parameters that are configured locally on the chassis. Per inter-chassis session recovery implementation requirements, the two chassis systems must be configured identically for all parameters not associated with physical connectivity to the distribution node.

The fields "timestamp", "SeqNumResetIndicator", and "FileSeqNumber" are all locally generated by the specific system through CDR subsystem, regardless of whether they are in an Inter-chassis Session Recovery arrangement or not.

- The "timestamp" value is unique to the system generating the actual xDRs and generated at the time the file is opened on the system.
- The SeqNumResetIndicator is a unique counter to determine the number of resets applied to FileSeqNumber. This counter is generated by CDR subsystem and increment the counter in event of resets in FileSeqNumber. This is required as "timestamp" field is not sufficient to distinguish between a unique and a duplicate xDR.

As such, the "SeqNumResetIndicator" field is used to distinguish between xDR files which have the same "FileSeqNumber" as a previously generated xDR as a result of:

- Normal operation, for example a rollover of the "FileSeqNumber" from maximum limit to 0.
- Due to a failure of one of the ECS processes running on a packet processing card card.
- Failure of the system (that is, Inter-chassis Session Recovery switchover).

In any scenario where the "FileSeqNumber" is reset to 0, the value of the "SeqNumResetIndicator" field is incremented by 1.

- The value of the "FileSeqNumber" is directly linked to the ECS process that is generating the specific xDRs. Any failure of this specific ECS process results in resetting of this field to 0.

Impact on xDR File Content

The following scenarios impact the xDR file content:

- On failure of an active chassis:

  On system startup, xDR files are generated in accordance with the standard processes and formats. If the system fails at any time it results in an inter-chassis session recovery switchover from active to standby and the following occurs depending on the state of the call/flow records and xDR file at the time of failure:

  - Call/flow records that were being generated and collected in system memory prior to being written out to /records directory on packet processing card RAM are not recoverable and therefore are lost.
  - Closed xDRs that have been written out to records directory on packet processing card RAM but that have yet to be retrieved by the external storage are recoverable.
  - Closed xDRs that have been retrieved and processed by the external storage have no impact.

- On the activation of a Standby chassis:

  Upon detection of a failure of the original active chassis, the standby chassis transits to the active state and begins serving the subscriber sessions that were being served by the now failed chassis. Any subsequent new subscriber session will be processed by this active chassis and will generate xDRs per the standard processes and procedures.

  However, this transition impacts the xDRs for those subscribers that are in-progress at the time of the transition. For in progress subscribers, a subset of the xDR fields and their contents are carried over to the newly active chassis via the SRP link. These fields and their contents, which are carried over after an Inter-chassis Session Recovery switchover, are as follows:

  - HA-CORRELATION-ID
  - PDSN-CORRELATION-ID (PDSN only)
  - PDSN-NAS-IP-ADDRESS (PDSN only)
  - PDSN-NAS-ID (PDSN only)
  - USERNAME
  - MSID
  - RADIUS-NAS-IP-ADDRESS

  All remaining fields are populated in accordance with the procedures associated with any new flow with the exceptions that, the field "First Packet Direction" is set to "Unknown" for all in-progress flows that were interrupted by the switchover and the field "FDR Reason" is marked as a PDSN Handoff and therefore is set to a value of "1" and corresponding actions are taken by the billing system to assure a proper and correct accounting of subscriber activities.