Oracle® Communications Convergent Charging Controller

Radius Control Agent Technical Guide

Release 6.0.1

April 2017

ORACLE®

Copyright

Contents

About This Document

Scope

The scope of this document includes all the information required to install, configure and administer the Radius Control Agent application.

Audience

This guide was written primarily for system administrators and persons installing, configuring and administering the Radius Control Agent application. However, sections of the document may be useful to anyone requiring an introduction to the application.

Prerequisites

A solid understanding of UNIX and a familiarity with IN concepts are essential prerequisites for safely using the information contained in this technical guide. Attempting to install, remove, configure or otherwise alter the described system without the appropriate background skills could cause damage to the system; including temporary or permanent incorrect operation, loss of service, and may render your system beyond recovery.

Although it is not a prerequisite to using this guide, familiarity with the target platform would be an advantage.

This manual describes system tasks that should only be carried out by suitably trained operators.

Related Documents

The following documents are related to this document:

- *Advanced Control Services Technical Guide*
- *Charging Control Services Technical Guide*
- *Charging Control Services User's Guide*
- *Radius Control Agent User's Guide*
- *Roaming Technical Guide*
- *Service Logic Execution Environment Technical Guide*
- *Service Management System Technical Guide*
- *Service Management System User's Guide*

Document Conventions

Typographical Conventions

The following terms and typographical conventions are used in the Oracle Communications Convergent Charging Controller documentation.

Formatting Convention	Type of Information
Special Bold	Items you must select, such as names of tabs. Names of database tables and fields.
Italics	Name of a document, chapter, topic or other publication. Emphasis within text.
Button	The name of a button to click or a key to press. **Example:** To close the window, either click **Close**, or press **Esc**.
Key+Key	Key combinations for which the user must press and hold down one key and then press another. Example: **Ctrl+P** or **Alt+F4**.
`Monospace`	Examples of code or standard output.
`Monospace Bold`	Text that you must enter.
variable	Used to indicate variables or text that should be replaced with an actual value.
menu option > menu option >	Used to indicate the cascading menu option to be selected. Example: **Operator Functions > Report Functions**
hypertext link	Used to indicate a hypertext link.

Specialized terms and acronyms are defined in the glossary at the end of this guide.

Chapter 1

System Overview

Overview

Introduction

This chapter provides a high-level overview of the application. It explains the basic functionality of the system and lists the main components.

It is not intended to advise on any specific Oracle Communications Convergent Charging Controller network or service implications of the product.

In this Chapter

This chapter contains the following topics.

What is the Radius Control Agent?

Introduction

The Radius Control Agent (RCA) is a SLEE interface that acts as the gateway between Radius and INAP, providing a seamless integration between prepaid PSDNs and applications such as Advanced Control Services (ACS) or Prepaid Charging. The connection to the Oracle SLC is through a CAMEL 2 interface.

RCA can also be configured to work between Cisco GGSN and Prepaid Charging by editing the **eserv.config** file and **sms.jnlp** files.

The following functionality is provided:

- Listening on specific ports for incoming Radius messages
- Sending Radius messages to predetermined ports
- Sending SLEE events to another SLEE application using a service key
- Constructing outgoing INAP messages
- Processing incoming INAP messages

RCA Profile Block list

Here are the profile blocks available when RCA is installed.

Name	App block	Description
Outgoing Extensions	19	This profile block, if not empty, will be copied into a special extension profiles value in the outgoing Connect message.

Name	App block	Description
Incoming Extensions	20	This profile block contains the profiles value from the special extension obtained from the InitialDP which enables applications, such as RCA, to communicate with slee_acs and the GGSN network node.

RCA primary tags

Here is a list of the profile primary tags used in the RCA service.

Description	Decimal	Vendor ID	Type
3GPP Radius 3GPP_CAMEL_CHARGING_INFO	6760024	10415	24
3GPP Radius 3GPP_CG_ADDRESS	6760004	10415	4
3GPP Radius 3GPP_CG_IPV6_ADDRESS	6760014	10415	14
3GPP Radius 3GPP_CHARGING_CHARACTERISTICS	6760013	10415	13
3GPP Radius 3GPP_CHARGING_ID	6760002	10415	2
3GPP Radius 3GPP_GGSN_ADDRESS	6760007	10415	7
3GPP Radius 3GPP_GGSN_IPV6_ADDRESS	6760016	10415	16
3GPP Radius 3GPP_GGSN_MCC_MNC	6760009	10415	9
3GPP Radius 3GPP_GPRS_QOS_PROFILE	6760005	10415	5
3GPP Radius 3GPP_IMEISV	6760020	10415	20
3GPP Radius 3GPP_IMSI	6760001	10415	1
3GPP Radius 3GPP_IMSI_MCC_MNC	6760008	10415	8
3GPP Radius 3GPP_IPV6_DNS_SERVER	6760017	10415	17
3GPP Radius 3GPP_MS_TIMEZONE	6760023	10415	23
3GPP Radius 3GPP_NSAPI	6760010	10415	10
3GPP Radius 3GPP_PDP_TYPE	6760003	10415	3
3GPP Radius 3GPP_RAT_TYPE	6760021	10415	21
3GPP Radius 3GPP_SELECTION_MODE	6760012	10415	12
3GPP Radius 3GPP_SESSION_STOP_INDICATOR	6760011	10415	11
3GPP Radius 3GPP_SGSN_ADDRESS	6760006	10415	6
3GPP Radius 3GPP_SGSN_IPV6_ADDRESS	6760015	10415	15
3GPP Radius 3GPP_SGSN_MCC_MNC	6760018	10415	18
3GPP Radius 3GPP_USER_LOCATION_INFO	6760022	10415	22
Cisco Radius Connect-Progress	6820004	9	1
Cisco Radius Downlink-Nexthop	6820003	9	1
Cisco Radius Quota-Server-Address	6820002	9	1
Cisco Radius Rulebase-ID	6820001	9	1
Cisco Radius Teardown-Indicator	6820005	9	1
Core Radius acct_authentic	6690045	-	45
Core Radius acct_delay_time	6690041	-	41
Core Radius acct_input_gigawords	6690052	-	52
Core Radius acct_input_octets	6690042	-	42
Core Radius acct_input_packets	6690047	-	47

Description	Decimal	Vendor ID	Type
3GPP Radius 3GPP_CAMEL_CHARGING_INFO	6760024	10415	24
Core Radius acct_interim_interval	6690085	-	85
Core Radius acct_link_count	6690051	-	51
Core Radius acct_multi_session_id	6690050	-	50
Core Radius acct_output_gigawords	6690053	-	53
Core Radius acct_output_octets	6690043	-	43
Core Radius acct_output_packets	6690048	-	48
Core Radius acct_session_id	6690044	-	44
Core Radius acct_session_time	6690046	-	46
Core Radius acct_status_type	6690040	-	40
Core Radius acct_terminate_cause	6690049	-	49
Core Radius arap_challenge_response	6690084	-	84
Core Radius arap_features	6690071	-	71
Core Radius arap_password	6690070	-	70
Core Radius arap_security	6690073	-	73
Core Radius arap_security_data	6690074	-	74
Core Radius arap_zone_access	6690072	-	72
Core Radius callback_id	6690020	-	20
Core Radius callback_number	6690019	-	19
Core Radius called_station_id	6690030	-	30
Core Radius calling_station_id	6690031	-	31
Core Radius chap_challenge	6690060	-	60
Core Radius chap_password	6690003	-	3
Core Radius class	6690025	-	25
Core Radius configuration_token	6690078	-	78
Core Radius connect_info	6690077	-	77
Core Radius eap_message	6690079	-	79
Core Radius error_cause	6690101	-	101
Core Radius event_timestamp	6690055	-	55
Core Radius filter_id	6690011	-	11
Core Radius framed_appletalk_link	6690037	-	37
Core Radius framed_appletalk_network	6690038	-	38
Core Radius framed_appletalk_zone	6690039	-	39
Core Radius framed_compression	6690013	-	13
Core Radius framed_interface_id	6690096	-	96
Core Radius framed_ip_address	6690008	-	8
Core Radius framed_ip_netmask	6690009	-	9
Core Radius framed_ipv6_pool	6690100	-	100
Core Radius framed_ipv6_prefix	6690097	-	97
Core Radius framed_ipv6_route	6690099	-	99

Description	Decimal	Vendor ID	Type
3GPP Radius 3GPP_CAMEL_CHARGING_INFO	6760024	10415	24
Core Radius framed_ipx_network	6690023	-	23
Core Radius framed_mtu	6690012	-	12
Core Radius framed_pool	6690088	-	88
Core Radius framed_protocol	6690007	-	7
Core Radius framed_route	6690022	-	22
Core Radius framed_routing	6690010	-	10
Core Radius idle_timeout	6690028	-	28
Core Radius login_ip_host	6690014	-	14
Core Radius login_ipv6_host	6690098	-	98
Core Radius login_lat_group	6690036	-	36
Core Radius login_lat_node	6690035	-	35
Core Radius login_lat_port	6690063	-	63
Core Radius login_lat_service	6690034	-	34
Core Radius login_service	6690015	-	15
Core Radius login_tcp_port	6690016	-	16
Core Radius message_authenticator	6690080	-	80
Core Radius nas_identifier	6690032	-	32
Core Radius nas_ip_address	6690004	-	4
Core Radius nas_ipv6_address	6690095	-	95
Core Radius nas_port	6690005	-	5
Core Radius nas_port_id	6690087	-	87
Core Radius nas_port_type	6690061	-	61
Core Radius password_retry	6690075	-	75
Core Radius port_limit	6690062	-	62
Core Radius prompt	6690076	-	76
Core Radius proxy_state	6690033	-	33
Core Radius reply_message	6690018	-	18
Core Radius service_type	6690006	-	6
Core Radius session_timeout	6690027	-	27
Core Radius state	6690024	-	24
Core Radius termination_action	6690029	-	29
Core Radius user_name	6690001	-	1
Core Radius user_password	6690002	-	2
MS- primary-DNS-server	6950028	311	28
MS-Primary-NBNS-Server	6950030	311	30
MS-Secondary-DNS-Server	6950029	311	29
MS-Secondary-NBNS-Server	6950031	311	31

Note: These tags are only present for RCA and are displayed on the ACS Configuration screen, **Profile Tag Details** tab.

Chapter 2

Configuration

Overview

Introduction

This chapter explains how to configure the Oracle Communications Convergent Charging Controller application.

In this chapter

This chapter contains the following topics.

Configuration Overview

Introduction

This topic provides a high level overview of how the RCA interface is configured.

There are configuration options which are added to the configuration files that are not explained in this chapter. These configuration options are required by the application and should not be changed.

Configuration components

The Radius Control Agent is configured by the following components:

Component	Locations	Description	Further Information
eserv.config	all SLC machines	RCA is configured by the RADIUSControlAgent section of **eserv.config**.	**eserv.config** *Configuration* (on page 6)
acs.conf	all SLC machines	The acs.conf configures the acsChassis for RADIUS.	*acs.conf Configuration* (on page 8) and *ACS Technical Guide*.
SLEE.cfg	all SLC machines	The SLEE interface is configured to include the RCA service.	*SLEE.cfg Configuration* (on page 7) and *SLEE Technical Guide.*

eserv.config Configuration

Introduction

The **eserv.config** file is a shared configuration file, from which many Oracle Communications Convergent Charging Controller applications read their configuration. Each Convergent Charging Controller machine (SMS, SLC, and VWS) has its own version of this configuration file, containing configuration relevant to that machine. The **eserv.config** file contains different sections; each application reads the sections of the file that contains data relevant to it.

The **eserv.config** file is located in the **/IN/service_packages/** directory.

The **eserv.config** file format uses hierarchical groupings, and most applications make use of this to divide the options into logical groupings.

Configuration File Format

To organize the configuration data within the **eserv.config** file, some sections are nested within other sections. Configuration details are opened and closed using either { } or [ ].

- Groups of parameters are enclosed with curly brackets – { }
- An array of parameters is enclosed in square brackets – [ ]
- Comments are prefaced with a # at the beginning of the line

To list things within a group or an array, elements must be separated by at least one comma or at least one line break. Any of the following formats can be used, as in this example:

```
    { name="route6", id = 3, prefixes = [ "00000148", "0000473"] }
    { name="route7", id = 4, prefixes = [ "000001049" ] }
```

or

```
    { name="route6"
        id = 3
        prefixes = [
            "00000148"
            "0000473"
        ]
    }
    { name="route7"
        id = 4
        prefixes = [
            "000001049"
        ]
    }
```

or

```
    { name="route6"
        id = 3
        prefixes = [ "00000148", "0000473" ]
    }
    { name="route7", id = 4
        prefixes = [ "000001049" ]
    }
```

eserv.config Files Delivered

Most applications come with an example **eserv.config** configuration in a file called **eserv.config.example** in the root of the application directory, for example, **/IN/service_packages/eserv.config.example**.

Editing the File

Open the configuration file on your system using a standard text editor. Do not use text editors, such as Microsoft Word, that attach control characters. These can be, for example, Microsoft DOS or Windows line termination characters (for example, ^M), which are not visible to the user, at the end of each row. This causes file errors when the application tries to read the configuration file.

Always keep a backup of your file before making any changes to it. This ensures you have a working copy to which you can return.

Loading eserv.config Changes

If you change the configuration file, you must restart the appropriate parts of the service to enable the new options to take effect.

RCA eserv.config configuration

The **eserv.config** file must be configured to enable the RCA to work. All necessary configuration in **eserv.config** is done at installation time by the configuration script.

For any type of Radius client other than 3GPP2 (CDMA), it is necessary to edit the **eserv.config** and define a ClientType section. For example, this is necessary for a CISCO GGSN.

Note: The RCA configuration options in **eserv.config** are explained in the section on the *radiusControlAgent* (on page 9) background process.

SLEE.cfg Configuration

Introduction

The SLEE.cfg file must be configured to enable the RCA to work. All necessary SLEE configuration is done at installation time by the configuration script, this section is for information only.

The SLEE configuration file is located in:

/IN/service_packages/SLEE/etc/SLEE.cfg

Refer to *SLEE Technical Guide* for details on SLEE configuration.

RCA SLEE configuration

On install the following lines are added to the SLEE.cfg

```
SERVICEKEY=INTEGER 14 RADIUS
SERVICE=RADIUS 1 slee_acs CCS_DATA
INTERFACE=rca rca.sh /IN/service_packages/RCA/bin EVENT
```

Note: It is essential for the correct operation of this application that the SLEE Interface type is always set to EVENT.

SERVICEKEY

The SERVICEKEY entry specifies the SLEE service key for the RADIUS service.

During rcaScp installation the value of the service key (default: 14) can be specified, or modified manually after install, if necessary.

SERVICE

The SERVICE entry specifies that the RADIUS service (that is, Service Key 14) corresponds to the CCS_DATA service entry in acs.conf.

acs.conf Configuration

Introduction

The acsChassis section in the acs.conf file on the SLC must be configured to enable the RCA to work. All necessary ACS configuration is done at installation time by the configuration script, this section is for information only. However, to enable RCA to work in the parameterised mode for Radius clients other than 3GPP2, it may be necessary to manually edit some of the parameters here as indicated.

Refer to the *ACS Technical Guide* for details on ACS configuration.

acsChassis configuration

On install the following lines are added to the acsChassis section of acs.conf.

```
ServiceEntry (CCS_DATA,cC,cC,ccsSvcLibrary.so)
minimumSizeOfConnectSleeEvent 1500
```

ServiceEntry

The ServiceEntry name (“CCS_DATA”) matches the SERVICE entry from SLEE.cfg.

minimumSizeOfConnectSleeEvent

This value is used to increase the parameters in the outgoing extensions. It is added manually to increase the default value, if required.

Default: 1024

Allowed:

Chapter 3

Background Processes

Overview

Introduction

This chapter explains the processes that are started automatically by Service Logic Execution Environment (SLEE).

Note: This chapter also includes some plug-ins to background processes which do not run independently.

In this chapter

This chapter contains the following topics.

radiusControlAgent

Purpose

The radiusControlAgent process manages and processes the incoming Radius messages.

Startup

This process is started automatically by the SLEE. For more information see *SLEE.cfg Configuration* (on page 7).

Command line parameters

The radiusControlAgent process accepts the following command line parameters.

Usage:

```
radiusControlAgent -d -c config_section
```

The available parameters are:

Parameter	Description
-d	Displays the values assigned to the configuration options for the RCA at startup, as defined in **eserv.config**.
-c *config_section*	Specifies which section in **eserv.config** to load. If this parameter is not defined, the default used is RADIUSControlAgent.

Example eserv.config

The following provides an example of the lines added to the **eserv.config** file on the SLC when you install Radius Control Agent.

```
RADIUSControlAgent = {

    ConfigName = "default"
    TimerInterface = "Timer"
    DestinationSK = 14
    OracleUserAndPassword = "/"
    WorkingDirectory = "/IN/service_packages/RCA"
    CorePort = 1812
    AccountingPort = 1813
    DynamicAuthorisationPort = 3799
    Window = 300
    NumberOfRetries = 10
    ServiceKey = 14
    CalledStationExtensionId = 1
    SfAccessPointExtensionId = 2
    CorrelationIdExtensionId = 3
    NoA = 5
    DurationThresholdQuotaGap = 10
    VolumeThresholdQuotaGap = 10000

    NumberRules = [
                  { fromNoa=4,    remove=0, prepend="00" }
                  { fromNoa=3,    remove=0, prepend="0064" }
                  { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
                  { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
                  { prefix="0", min=9, remove=1, resultNoa=3 }
                  { prefix="", min=9, remove=0, resultNoa=3 }
          ]

    PollSleeCount = 100
    SelectTimeout = 10000
    MinimumSLEEEventSize = 1024

    BearerCapability_CodingStandard = 0
    BearerCapability_ITC = 0
    BearerCapability_TransferMode = 0
    BearerCapability_ITR = 0
    BearerCapability_uiProto1 = 1
    VolumeConversionRatio = 1
    UseCAP2ApplicationContext = false

    IgnoredInapOpsList = [34,35,56]
    CallingPartyCategory = 0
    EventTypeBCSM = "ETBanalyzedInformation"
    CalledPartyBCDNumber = "1111"

    RADIUSUserNameAsCLI = true
    RADIUSUserNameAsLocationNumber = true
    MaxUdrOpenTime = 600
    MaxUdrSize = 1024

    defaultClientType = "Cisco"
      ClientTypes = [
      { clientTypeName = "3GPP2"
        parameterisedRCA = false

        IPAddresses = [
          { address = "192.168.1.100" }
        ]
      }

      { clientTypeName = "Cisco"
        parameterisedRCA = true
```

```
    IPAddresses = [
      { address = "192.168.14.100" }
      { address = "192.168.1.100" }
      { address = "192.168.1.101" }
    ]

ParameterisedRCAFields = {
    QualifyingAttributeRules = [
        { AttributeValuePairs = [
            {attr = 81, attrType = "string", value = "abc" }
            {vendorID = 10415, attr = 21, attrType = "integer", value = 4 }
            ]
        }
        { AttributeValuePairs = [
            { attr = 181, attrType = "string", value = "2" }
            { attr = 182, attrType = "integer", value = 4 }
            { vendorID = 10415, attr = 121, attrType = "integer", anyValue =
            true}
            ]
        }
    ]
   sendAccessRejectMessages = false
   sendAccountingResponses = false
   expectResponsesFromDisconnectRequest = true
   useQuotainAccessAccept = false
   useThresholdinAccessAccept = false
   QuotaAttributeinAccessAccept = { vendorID = 10415, attr = 10, type =
"QVdigits" }
   ThresholdAttributeinAccessAccept = { vendorID = 10415, attr = 10, type =
"QVdigits" }
   sendZeroQuotaAccessAcceptOnFundsExpiry = false
   noFundsDisconnectMethod = "DisconnectRequest"
   messageAfterFinalAccessRequest = "emptyAccessAccept"

    ExtendedAttributeTypes = [
        {
        type = "QVdigits"
        pattern = "QV*"
        selectNth = 1
        firstFactor = "1"
        secondFactor = "0"
        }

        {
        type = "CiscoOverflow"
        pattern = "I*;*"
        selectNth = 1
        firstFactor = "4292967296"
        secondFactor = "1"
        }
    ]
    IncomingMapping = {
           Attributes = [
            { attr = 1, attrName = "User-Name",          attrType = "string",
    storeToTag = 6690001 }
           { attr = 2, attrName = "User-Password",      attrType = "string",
    storeToTag = 6690002 }
           { attr = 3, attrName = "CHAP-Password",      attrType = "string",
    storeToTag = 6690003 }
            { attr = 4, attrName = "NAS-IP-Address",     attrType = "IPv4",
    storeToTag = 6690004 }
            { attr = 5, attrName = "NAS-Port",           attrType = "integer",
    storeToTag = 6690005 }
```

```
        { attr = 6, attrName = "Service-Type",       attrType = "integer",
storeToTag = 6690006 }
        { attr = 7, attrName = "Framed-Protocol",    attrType = "integer",
storeToTag = 6690007 }
        { attr = 8, attrName = "Framed-IP-Address",  attrType = "IPv4",
storeToTag = 6690008 }
        { attr = 9, attrName = "Framed-IP-Netmask",  attrType = "IPv4",
storeToTag = 6690009 }
       { attr =22, attrName = "Framed-Route",       attrType = "string",
storeToTag = 6690022 }
        { attr =25, attrName = "Class",              attrType = "string",
storeToTag = 6690025 }
       { attr =27, attrName = "Session-Timeout",    attrType = "integer",
storeToTag = 6690027 }
       { attr =28, attrName = "Idle-Timeout",       attrType = "string",
storeToTag = 6690028 }
        { attr =30, attrName = "Called-Station-ID",  attrType = "string",
storeToTag = 6690030 }
        { attr =31, attrName ="Calling-Station-ID",  attrType = "string",
storeToTag = 6690031 }
       { attr =32, attrName = "NAS-Identifier",     attrType = "string",
storeToTag = 6690032 }
        { attr =40, attrName = "Acct-Status-Type",   attrType = "integer",
storeToTag = 6690040 }
        { attr =41, attrName = "Acct-Delay-Time",    attrType = "integer",
storeToTag = 6690041 }
        { attr =42, attrName = "Acct-Input-Octets",  attrType = "integer",
storeToTag = 6690042 }
        { attr =43, attrName = "Acct-Output-Octets", attrType = "integer",
storeToTag = 6690043 }
        { attr =44, attrName = "Acct-Session-Id",    attrType = "string",
storeToTag = 6690044 }
        { attr =45, attrName = "Acct-Authentic",     attrType = "integer",
storeToTag = 6690045 }
       { attr =46, attrName = "Acct-Session-Time",  attrType = "integer",
storeToTag = 6690046 }
        { attr =47, attrName = "Acct-Input-Packets", attrType = "integer",
storeToTag = 6690047 }
        { attr =48, attrName = "Acct-Output-Packets",attrType = "integer",
storeToTag = 6690048 }
       { attr =60, attrName = "CHAP-Challenge",     attrType = "string",
storeToTag = 6690060 }
        { attr =61, attrName = "NAS-Port-Type",      attrType = "string",
storeToTag = 6690061 }
       { attr =96, attrName = "Framed-IPv6-Prefix", attrType = "string",
storeToTag = 6690096 }

     # Cisco Vendor Specific

        { vendorID=9, attr=1, keystring="csg:billing_plan=",
attrName="Rulebase-ID", attrType="string", storeToTag = 6820001 }
        { vendorID=9, attr=1, keystring="csg:quota_server=", attrName="Quota-
Server-Address", attrType="string", storeToTag = 6820002 }
        { vendorID=9, attr=1, Keystring="csg:downlink_nexthop=", attrName =
"Downlink-Nexthop", attrType="string", storeToTag = 6820003 }

      # 3GPP Vendor Specific

        { vendorID = 10415, attr = 1, attrName = "3GPP-IMSI",        attrType
= "string", storeToTag = 6760001 }
        { vendorID = 10415, attr = 2, attrName = "3GPP-Charging_Id", attrType
= "string", storeToTag = 6760002 }
        { vendorID = 10415, attr = 3, attrName = "3GPP-PDPType",     attrType
= "integer", storeToTag = 6760003 }
```

```
        { vendorID = 10415, attr = 4, attrName = "3GPP-CG-Address",  attrType
= "IPv4", storeToTag = 6760004 }
       { vendorID = 10415, attr = 5, attrName = "3GPP-GPRS-QoS-Profile",
attrType = "string", storeToTag = 6760005 }
        { vendorID = 10415, attr = 6, attrName = "3GPP-SGSN-Address",attrType
= "IPv4", storeToTag = 6760006 }
        { vendorID = 10415, attr = 7, attrName = "3GPP-GGSN-Address",attrType
= "IPv4", storeToTag = 6760007 }
        { vendorID = 10415, attr = 8, attrName = "3GPP-IMSI-MCC-MNC",attrType
= "string", storeToTag = 6760008 }
        { vendorID = 10415, attr = 9, attrName = "3GPP-GGSN-MCC-MNC",attrType
= "string", storeToTag = 6760009 }
        { vendorID = 10415, attr =10, attrName = "3GPP-NSAPI",
attrType= "string", storeToTag = 6760010 }
        { vendorID = 10415, attr =12, attrName = "3GPP-Selection-
Mode",attrType = "string", storeToTag = 6760012 }
        { vendorID = 10415, attr =13, attrName = "3GPP-Charging-
Characteristics",attrType = "string", storeToTag = 6760013 }
        { vendorID = 10415, attr =18, attrName = "3GPP-SGSN-MCC-MNC",attrType
= "string", storeToTag = 6760018 }
        { vendorID = 10415, attr =20, attrName = "3GPP-IMEISV",      attrType
= "octets", storeToTag = 6760020 }
        { vendorID = 10415, attr =21, attrName = "3GPP-RAT-Type",    attrType
= "integer", storeToTag = 6760021 }
        { vendorID = 10415, attr =22, attrName = "3GPP-User-Loc-
Info",attrType = "octets", storeToTag = 6760022 }
        { vendorID = 10415, attr =23, attrName = "3GPP-MS-TimeZone",attrType
= "octets", storeToTag = 6760023 }
      ]
}
ValueEditingRules = [
    {
    attrName = "User-Name"
    ifStartsWith = "ADMIN"
    removeFront = 5
    ifEndsWith = ".com"
    removeBack = 4
    }

]
DecryptTags = [
    { fromTag = 6690002, decryptToTag = 6760099 }
]
HardcodedValues = [
    {tag=6820005, tagType="string", value = "gprs:teardown-ind=yes" }
    {tag=6820006, tagType="string", value = "PrimaryDNSserver" }
    {tag =327689, tagType="string", value = "1111" }
    {tag =327690, tagType="string", value = "027774444" }
    {tag =327691, tagType="string", value = "2008" }
    {tag =327713, tagType="string", value = "49393404" }
    {tag =327801, tagType="string", value = "88888" }
    {tag =327703, tagType="string", value = "777777" }
    {tag =327868, tagType="string", value = "666666" }
]

ReleaseCall = [
    {code=41, vendorID=9, attr=1, tagType="string", value = "gprs:teardown-
    ind=yes" }
    {code=31, vendorID=9, attr=1, tagType="string", value = "gprs:teardown-
    ind=no" }
]
InitialDPMapping = {
    callingPartyNumber = {
```

```
            tag = 6690031

            NumberRules = [
                { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
                { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
                { prefix="0", min=9, remove=1, resultNoa=3 }
                { prefix="", remove=0, resultNoa=3 }
            ]
        }
        calledPartyBCDNumber = {
            tag = 327689
            NumberRules = [
                { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
                { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
                { prefix="0", min=9, remove=1, resultNoa=3 }
                { prefix="", remove=0, resultNoa=3 }
            ]
        }
        locationNumberTag = {
            tag = 0
            NumberRules = [
                { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
                { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
                { prefix="0", min=9, remove=1, resultNoa=3 }

            ]
        }
        callReferenceNumberTag = 0
    }

    InitialDPExtensions = [6760003,6760006,6690004]

    ConnectMapping = {
        destinationRoutingAddress = {
            tag = 327689

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
        originalCalledPartyID = {
            tag = 327713

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
        callingPartysCategoryTag = 327801
        redirectingPartyId = {
            tag = 327703

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
        genericNumbersAdditionalCallingPartyNumber = {
            tag = 327868

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
```

```
            ]
        }
    }

    ConnectExtensions = [
        { vendorID = 311, attr = 30, attrName = "PrimaryNBNSserver",  attrType =
        "IPv4", profileBlockTag = 6950030 }
    ]

    OutgoingMappingtoSwitch = {
        AccessAcceptMapping = [
            {attr = 6, attrName = "Service-Type",     attrType = "integer",
            fromTag = 6690006 }
            {attr = 7, attrName = "Framed-Protocol",  attrType = "integer",
            fromTag = 6690007 }
            {attr = 8, attrName = "Framed-IP-Address",attrType = "IPv4",
            fromTag = 6690008 }
            {attr = 9, attrName = "Framed-IP-Netmask",attrType = "IPv4",
            fromTag = 6690009 }
            {attr =22, attrName = "Framed-Route",     attrType = "integer",
            fromTag = 6690022 }
            {attr =25, attrName = "Class",            attrType = "string",
            fromTag = 6690025 }
            {attr =27, attrName = "Session-Timeout",  attrType = "integer",
            fromTag = 6690027 }
            {attr =28, attrName = "Idle-Timeout",     attrType = "integer",
            fromTag = 6690028 }
            {vendorID = 311, attr = 30, attrName = "PrimaryNBNSserver",attrType =
            "IPv4", fromTag = 6950030 }
            {vendorID = 9, attr = 1, attrName = "Tear-down-indicator",attrType =
            "string", fromTag = 6820005 }
        ]
        DisconnectRequestMapping = [
            {vendorID = 10415, attr = 1, attrName = "3GPP-IMSI",   attrType =
            "string", fromTag = 6760001 }
            {vendorID = 10415, attr = 10,attrName = "3GPP-NSAPI",  attrType =
            "string", fromTag = 6760010 }
            {vendorID = 9,     attr = 1, attrName = "TeardownInd", attrType =
            "string", fromTag = 6820005 }
            {attr = 44,attrName = "Acct-Session-Id", attrType = "string", fromTag
            = 6690044 }
            {attr = 6,attrName = "Service-Type", attrType = "integer", fromTag =
            6690006 }
            {attr = 4,attrName = "NAS-IP-Address", attrType = "IPv4", fromTag =
            6690004 }
            {attr = 8,attrName = "Framed-IP-Address", attrType = "IPv4", fromTag
            = 6690008 }
            {attr = 18,attrName = "originalCalledPartyID", attrType = "string",
            fromTag = 327713 }
        ]
    }

# Charging

    UseUnitsfromAccessRequest = false
    UseFinalUnitsFromAccountingRequest = true
    CumulativeConversionRules = false
    AccountingRequestConversionRules = [
        { ruleName = "rule1",
                attributeList = [
                { attributeTag = 6690042, multiplier = 0.00001 } # Deciseconds in
                ACR = 0.00001 * input-octets +
```

```
            { attributeTag = 6690043, multiplier = 0.00001 } #
            0.00001 * output-octets
        ]
    }

    { ruleName = "rule2",
        attributeList = [
            { attributeTag = 6690046, multiplier = 10 } # Deciseconds in ACR
            = 10 * session-time
        ]
    }
]
conversionFactor = 100000.0
clientThresholdQuotaGap = 100

SessionInactiveRules = [
    {AttributeValuePairs = [
            { attr = 81, attrType = "string", value = "2" } # and
            { attr = 82, attrType = "integer", value = 4 } # and
            { vendorID = 10415, attr = 21, attrType = "integer", value = 4 }
        ]
    } # or
    {AttributeValuePairs = [
            { attr = 81, attrType = "string", value = "2" } # and
            { attr = 82, attrType = "integer", value = 4 } # and
            { vendorID = 10415, attr = 21, attrType = "integer", value = 3 }
        ]
    }
    ]
ServiceKeyRules = [
    {AttributeValuePairs = [
            { attr = 61, attrType = "string", value = "2" }
            { attr = 62, attrType = "integer", value = 4 }
            { vendorID = 10415, attr = 21, attrType = "integer", value = 4 }
        ]
    sleeServiceKey=231
    inapServiceKey=231
    }
]

CorrelationAttributes = {
    AccountingRequest = [
        { vendorID = 10415, attr = 2 }  # 3GPP-Charging-Id
        {                 , attr = 1 }
        {                 , attr = 6 }
    ]
    AccessRequest = [
        { vendorID = 10415, attr = 2 }  # 3GPP-Charging-Id
        {                 , attr = 1 }
        {                 , attr = 6 }
    ]
}

systemErrorAttribute = {
    attr = 0 # attr = 0 means do not add any such attribute
    attrType="string"
    value = "System Error"
}

invalidMessageSequenceAttribute = {
    attr = 0 # attr = 0 means do not add any such attribute
    attrType="string"
    value = "Invalid Message Sequence"
}
```

```
        insufficientFundsAttribute = {
            attr = 0 # attr = 0 means do not add any such attribute
            attrType="string"
            value = "Insufficient Funds"
        }

        timeoutAttribute = {
            attr = 0 # attr = 0 means do not add any such attribute
            attrType="string"
            value = "Session Timed Out"
        }

        replyMessagestoAttribute = 0     # 18 would seem sensible but Cisco do not
        support it

        sessionTimeout = 3600
        idleTimeout = 600
        eventTimestampInAccessAccept = false

    } #end ParameterisedRCAFields
    }
    }

    tracing = {
        enabled = false
        outputFile = "/IN/service_packages/RCA/tmp/rcaTrace.log"
        outputFileCycle = 512
        maxFileSizeKB = 0
        maxNumFiles = 4
        callsPerMinute = 2
        origAddress = [
        ""
        ]
        destAddress = [
        ""
        ]
        useTONNPI = true
    }
}
```

RADIUSControlAgent parameters

The configuration parameters for the radiusControlAgent are automatically added to the RCA section of **eserv.config** at installation.

The available parameters are:

AccountingPort

Syntax: `AccountingPort = value`

Description: Defines the port number used by the RCA to receive incoming Radius accounting messages.

Type: Integer

Optionality: Optional

Allowed: Any valid port number

Default: 1813

Notes:

Example: `AccountingPort = 1813`

BearerCapability_CodingStandard

Syntax: `BearerCapability_CodingStandard = value`

Description: Defines the bearer capability coding standard.

Type: Integer

Optionality: Optional

Allowed:
- 0 = ITU_T
- 1 = ISO IEC
- 2 = National
- 3 = Network

Default: No default

Notes: By default, this parameter is not used.

Example: `BearerCapability_CodingStandard = 0`

BearerCapability_ITC

Syntax: `BearerCapability_ITC = value`

Description: Defines the ITC bearer capability.

Type: Integer

Optionality: Optional

Allowed:
- 0 = Speech
- 8 = Unrestricted Digital Information (UDI)
- 9 = Restricted Digital Information (RDI)
- 16 = 3.1 KHz Audio
- 17 = UDI with tones / announcements
- 24 = Video

Default: No default

Notes: By default, this parameter is not used.

Set the ITC (Information Transfer Capability) to 8, "Unrestricted Digital Information" (that is, matching config item `CCS.ccsActions.configuredVolumeITC`) if using RADIUS Volume-based billing with Oracle Prepaid Charging billing engines, to prevent volume-based charging sessions timing out prematurely on the BE.

Example: `BearerCapability_ITC = 8`

BearerCapability_ITR

Syntax: `BearerCapability_ITR = value`

Description: Defines the bearer capability information transfer rate.

Type: Integer

Optionality: Optional

Allowed:
- 0 = Packet
- 16 = 64 kb per second
- 17 = 2 * 64 kb per second
- 19 = 384 kb per second
- 21 = 1536 kb per second
- 23 = 1920 kb per second
- 24 = Multirate

Default: No default
Notes: By default this parameter is not used.
Example: `BearerCapability_ITR = 0`

BearerCapability_TransferMode

Syntax: `BearerCapability_TransferMode = value`
Description: Defines the bearer capability transfer mode.
Type: Integer
Optionality: Optional
Allowed:
- 0 = Circuit
- 2 = Packet

Default: No default
Notes: By default, this parameter is not used.
Example: `BearerCapability_TransferMode = 0`

BearerCapability_uiProto1

Syntax: `BearerCapability_uiProto1 = value`
Description: Defines the bearer capability user information layer protocol.
Type: Integer
Optionality: Optional
Allowed:
- 1 = ITU V110 I460 X30
- 2 = G.711 U LAW
- 3 = G.711 A LAW
- 4 = G.721 32 KBits
- 5 = H.221/ H.242
- 6 = H.223 / H.245
- 7 = Non-ITU SRA
- 8 = ITU V.120
- 9 = X31 HDLC
- 255 = Not present

Default: No default
Notes: By default, this parameter is not used.
Example: `BearerCapability_uiProto1 = 1`

CalledPartyBCDNumber

Syntax: `CalledPartyBCDNumber = "str"`
Description: A fixed number to send in the InitialDP, used as a dummy value in calculating tariffs (in the case of WiFi there is no real called party).
Type: String
Optionality: Optional
Allowed:
Default: No default
Notes: Contains a dummy value taken from **eserv.config**. If not specified, will not be sent in the InitialDP.
Example: `CalledPartyBCDNumber = "1111"`

CalledStationExtensionId

Syntax: `CalledStationExtensionId = value`

Description: Defines the ACS extension field where the called party URL is stored.

Type: Integer

Optionality: Required

Allowed:

Default: No default

Notes: Contains the RADIUS User-Name.

Example:
```
CalledStationExtensionId = 1
```

CallingPartyCategory

Syntax: `CallingPartyCategory = value`

Description: Defines the calling party category to use in all outgoing IDP messages.

Type: Integer

Optionality: Optional

Allowed: Any valid calling party category number

Default: 0

Notes:

Example:
```
CallingPartyCategory = 0
```

ClientTypes

Syntax:
```
ClientTypes = [
    clientTypeName = "str"
    parameterisedRCA = true|false
    IPAddresses = ["str"
        ]
]
```

Description: Lists the attributes of the Radius type client switches that will be supported.

Type: Array

Optionality: Optional (default used if not set).

Allowed:

Default:

Notes:

Example:
```
ClientTypes = [
    clientTypeName = "Cisco"
    parameterisedRCA = true
    IPAddresses = [
    { address = "123.456.78.900" }
        ]
]
```

clientTypeName

Syntax: `clientTypeName = "str"`

Description: Name of the Radius type client switch that will be supported by RCA.

Type: String

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example: `clientTypeName = "Cisco"`

parameterisedRCA

Syntax: `parameterisedRCA = true|false`

Description: Indicates if the default RCA processing logic for 3GPP2 or the one for Cisco GGSN should be used.

Type: Boolean

Optionality: Optional (default used if not set).

Allowed:
- true – Parameterised for Cisco GGSN logic
- false – Default RCA logic

Default:

Notes:

Example: `parameterisedRCA = true`

IPAddresses

Syntax: `IPAddresses = ["addr1","addr2", "addrn"]`

Description: The Internet Protocol (IP) addresses of the SLCs handling calls for this client type.

Type: Numeric string

Optionality: Optional (default used if not set)

Allowed: IP version 4 (IPv4) addresses, IP version 6 (IPv6) addresses

Default:

Notes: IPv6 addresses have the following format: `2001:db8:n:n:n:n:n:n` where *n* is a group of 4 hexadecimal digits. The industry standard for omitting zeros is also allowed when specifying IP addresses.

Example:
```
IPAddresses = [
    {address = "192.0.2.1"
    address = "2001:db8:0000:1050:0005:0600:300c:326b"
    address = "2001:db8:0:0:0:500:300a:326f"
    address = "2001:db8::c3"}
    ]
```

ConfigName

Syntax: `ConfigName = "str"`

Description: The Name, defined in the Global Config screen, that is assigned to the database configuration (configured through the RCA screens in the SMS) to be loaded.

Type: String

Optionality: Optional

Allowed: Any defined configuration name

Default: "default"

Notes:

Example: `ConfigName = "default"`

CorePort

Syntax: `CorePort =value`

Description: Defines the port number used to receive incoming Radius core messages.

Type: Integer
Optionality: Optional
Allowed: Any valid port number
Default: 1812
Notes:
Example: `CorePort = 1812`

CorrelationIdExtensionId

Syntax: `CorrelationIdExtensionId = value`
Description: Specifies which IDP extension to put the correlation ID into, as an ASN.1 octet string.
Type: Integer
Optionality: Optional
Allowed:
Default: 3
Notes: If you want to print the RADIUS Correlation-ID in the EDR, there must be an `extensionNumber` parameter in the acs.conf with an extension type matching the value of this parameter. This number must not be one used by any other extension.

Refer to *ACS Technical Guide* for details.
Example: `CorrelationIdExtensionId = 3`

DestinationSK

Syntax: `DestinationSK = value`
Description: An integer that defines the SLEE service key for all INAP SLEE outward messages sent by the RCA.
Type: Integer
Optionality: Optional
Allowed: Any valid service key integer
Default: 14
Notes:
Example: `DestinationSK = 14`

DurationThresholdQuotaGap

Syntax: `DurationThresholdQuotaGap = value`
Description: Gap between duration-threshold and duration-quota (seconds).
Type: Integer
Optionality: Optional
Allowed:
Default: 10
Notes:
Example: `DurationThresholdQuotaGap = 10`

DynamicAuthorisationPort

Syntax: `DynamicAuthorisationPort = value`
Description: Defines the port number that RCA sends Radius dynamic authorisation messages to.

Type: Integer
Optionality: Optional
Allowed: Any valid port number
Default: 3799
Notes:
Example: `DynamicAuthorisationPort = 3799`

EventTypeBCSM

Syntax: `EventTypeBCSM = "str"`
Description: Defines the BCSM event type to use in all outgoing IDP messages.
Type: String
Optionality: Optional
Allowed: ETBcollectedInfo
ETBanalyzedInformation
Default: ETBanalyzedInformation
Notes:
Example: `EventTypeBCSM = "ETBanalyzedInformation"`

IgnoredInapOpsList

Syntax: `IgnoredInapOpsList = [val1,val2,valn]`
Description: Lists the INAP operations that cannot be handled by the RCA, and hence ignored.
Type: Integer
Optionality: Optional
Allowed: Any INAP operation number
Default: None
Notes:
Example: `IgnoredInapOpsList = [34,35,56]`

MaxUdrOpenTime

Syntax: `MaxUdrOpenTime = value`
Description: Defines the time (in seconds) that UDRs can remain open.
Type: Integer
Optionality: Optional
Allowed:
Default: 600
Notes:
Example: `MaxUdrOpenTime = 600`

MaxUdrSize

Syntax: `MaxUdrSize = value`
Description: Defines the maximum size (in kilobytes) for a UDR file.
Type: Integer
Optionality: Optional
Allowed:

Default: 1024

Notes: If the size limit is exceeded by the current UDR record, the RCA completes the record in the current file and then creates a new file for the next record.

Example: `MaxUdrSize = 1024`

MinimumSLEEEventSize

Syntax: `MinimumSLEEEventSize = value`

Description: The minimum size of the slee event containing the InitialDP.

Type: Integer

Optionality: Optional (default used if not set)

Allowed:

Default: 1024

Notes: If the profile extensions are too large, that is, if there are a lot of items in InitialDPExtensions, increase this parameter.

Example: `MinimumSLEEEventSize = 1024`

NoA

Syntax: `NoA = value`

Description: Defines the Nature of Address value to use in the outgoing IDPs.

Type: Integer

Optionality: Optional

Allowed:

- 1 = international
- 2
- 3
- 4
- 5 = unknown

Default: 5

Notes:

Example: `NoA = 5`

NumberOfRetries

Syntax: `NumberOfRetries = value`

Description: Defines the maximum number of retries allowed before the interaction is terminated.

Type: Integer

Optionality: Optional

Allowed:

Default: 10

Notes:

Example: `NumberOfRetries = 10`

OracleUserAndPassword

Syntax: `OracleUserAndPassword = "str/str"`

Description: The Oracle username and password for the RCA.

Type: String

Optionality: Optional

Allowed:
Default: /
Notes:
Example: `OracleUserAndPassword = "/"`

PollSleeCount

Syntax: `PollSleeCount = value`
Description: Poll the SLEE this many times for each UDP poll.
Type: Integer
Optionality: Optional
Allowed:
Default: 100
Notes:
Example: `PollSleeCount = 100`

RADIUSUserNameAsCLI

Syntax: `RADIUSUserNameAsCLI = true|false`
Description: Defines the CLI.
Type: Boolean
Optionality: Optional
Allowed: Allowable values:
- true – Use the User-Name attribute from the AccessRequest message
- false – Use the Calling-Station-ID

Default: true
Notes:
Example: `RADIUSUserNameAsCLI = true`

RADIUSUserNameAsLocationNumber

Syntax: `RADIUSUserNameAsLocationNumber = true|false`
Description: Defines the location number.
Type: Boolean
Optionality: Optional
Allowed: Allowable values:
- true – Use the User-Name attribute from the AccessRequest message
- false – Use the Calling-Station-ID

Default: true
Notes:
Example: `RADIUSUserNameAsLocationNumber = true`

SelectTimeout

Syntax: `SelectTimeout = value`
Description: Timeout of the UDP select call (microseconds).
Type: Integer
Optionality: Optional

Allowed:
Default: 10000
Notes:
Example: `SelectTimeout = 10000`

ServiceKey

Syntax: `ServiceKey = value`
Description: Defines the service key to use in the outgoing IDPs.
Type: Integer
Optionality: Optional
Allowed:
Default: 14
Notes:
Example: `ServiceKey = 14`

SfAccessPointExtensionId

Syntax: `SfAccessPointExtensionId = value`
Description: The InitialDP extension slot number to use for the RADIUS SF-Access-Point-Id, that is, the name of the WiFi hotspot the subscriber is establishing the connection through.
Type: Integer
Optionality: Optional
Allowed:
Default: 2
Notes: If you want to print the RADIUS SF-Access-Point-Id in the EDR, there must be an `extensionNumber` parameter in the acs.conf with an extension type matching the value of this parameter. This number must not be one used by any other extension.

See *ACS Technical Guide* for details.
Example: `SfAccessPointExtensionId = 2`

TimerInterface

Syntax: `TimerInterface = "str"`
Description: The timer interface used by the RCA to set its timers.
Type: String
Optionality: Optional
Allowed:
Default: "Timer"
Notes:
Example: `TimerInterface = "Timer"`

UseCAP2ApplicationContext

Syntax: `UseCAP2ApplicationContext = true|false`
Description: If set to true, enables the use of CAP2 application context for backwards compatibility.
Type: Boolean
Optionality: Optional (default used if not set)

Allowed: true, false
Default: false
Notes:
Example: `UseCAP2ApplicationContext = false`

VolumeConversionRatio

Syntax: `VolumeConversionRatio = value`
Description: How many octets are considered equal to one second for charging.
Type: Integer
Optionality: Optional (default used if not set).
Allowed: Minimum allowed value is 1, maximum is 1048576.
Default: 1 (that is, 1 octet is reported as 1 second)
Notes: This only affects volume charges.
All reservations will be rounded up to be whole chunks, so this is also the billing granularity for volume reservations.
Example: `VolumeConversionRatio = 1`

VolumeThresholdQuotaGap

Syntax: `VolumeThresholdQuotaGap = value`
Description: Gap between duration-threshold and duration-quota (octets).
Type: Integer
Optionality: Optional
Allowed:
Default: 10000
Notes:
Example: `VolumeThresholdQuotaGap = 10000`

Window

Syntax: `Window = value`
Description: Defines the accept time delay window (seconds) for an incoming message. It has been provided to prevent replay attack.
Type: Integer
Optionality: Optional
Allowed:
Default: 300 seconds
Notes:
Example: `Window = 300`

WorkingDirectory

Syntax: `WorkingDirectory = "str"`
Description: The name for the base directory used by the RCA to store UDR and core files.
Type: String
Optionality: Optional
Allowed:
Default: /IN/service_packages/RCA

Notes:

Example: `WorkingDirectory = "/IN/service_packages/RCA"`

NumberRules parameters

Description

This section defines the global number normalisation rules for RCA. They are found within `NumberRules = [ ]`.

This section is optional.

NumberRules parameters

The following parameters define the number normalisation rules for RCA:

prefix

Syntax: `prefix = "pref"`

Description: Contains a digit or digits. Used to attempt to match the first digit or digits of a prefix number with the specified value. If the digit or digits match, the prefix part of the number rule is met.

Type: String

Optionality: Optional

Allowed: One or more decimal digits

Notes: This parameter is an element of the `NumberRules` parameter array.

Example: `prefix = "25"`

fromNoa

Syntax: `fromNoa = int`

Description: Used when attempting to match the nature of address (NoA) number contained in a message. If there is a match, the `fromNoa` part of the number rule is evaluated.

Type: Integer

Optionality: Required

Allowed:

- 2 – For unknown NoAs
- 3 – For national NoAs
- 4 – For international NoAs

Notes: If you omit `fromNoa` from the `NumberRules` parameter section, then no matching rule will be found.

Example: `fromNoa = 3`

min

Syntax: `min = num`

Description: Specifies the minimum number of digits a number may contain. To meet the min part of the number rule, the number of digits in the number must be equal to or greater than the value of `min`.

Type: Integer

Optionality: Optional (default used if not set)

Default: 0

Notes: The value of the `min` parameter must be greater than or equal to the value of the `remove` (on page 29) parameter.

Example: `min = 5`

max

Syntax: `max = num`

Description: Specifies the maximum number of digits a number may contain. To meet the max part of the number rule, the number of digits in the number must be equal to or less than the value of `max`.

Type: Integer

Optionality: Optional (default used if not set)

Default: 999

Example: `max = 9`

remove

Syntax: `remove = num`

Description: The number of digits stripped from the beginning of a number.

Type: Integer

Optionality: Required

Notes: The value of the `remove` parameter must be less than or equal to the value of the *min* (on page 28) parameter.

Example: `remove = 2`

prepend

Syntax: `prepend = "digits"`

Description: Defines digits added to the beginning of a number.

Type: String

Optionality: Optional

Allowed: Any combination of decimal digits, or a null string (`""`)

Notes:

- If the `remove` and `prepend` parameters are both used in the same number rule, "*prepend*" is added to the beginning of the number after the number has been modified by the `remove` parameter.
- The `prepend` parameter is an element of the `NumberRules` parameter array.

Example: `prepend = "0"`

resultNoa

Syntax: `resultNoa = noa`

Description: A nature of address (NOA) sent to the network.

Type: Integer

Optionality: Optional

Notes:

- A value is typically specified in demoralization rules
- This parameter is an element of the `NumberRules` parameter array

Example: `resultNoa = 4`

ParameterisedRCAFields section

Description

The parameters in this section are used to configure RCA for Cisco GGSN call handling. They are found within `ParameterisedRCAFields = { }`.

This section needs to be configured for all client types other than 3GPP2 (CDMA). When the client type is 3GPP2, the default RCA configuration is used.

ParameterisedRCAFields

```
ParameterisedRCAFields = {
    QualifyingAttributeRules = [
        { AttributeValuePairs = [
            {attr = 81, attrType = "string", value = "abc" }
            {vendorID = 10415, attr = 21, attrType = "integer", value = 4 }
            ]
        }
        { AttributeValuePairs = [
            { attr = 181, attrType = "string", value = "2" }
            { attr = 182, attrType = "integer", value = 4 }
            { vendorID = 10415, attr = 121, attrType = "integer", anyValue =
            true}
            ]
        }
    ]
   sendAccessRejectMessages = false
   sendAccountingResponses = false
   expectResponsesFromDisconnectRequest = true
   useQuotainAccessAccept = false
   useThresholdinAccessAccept = false
   QuotaAttributeinAccessAccept = { vendorID = 10415, attr = 10, type =
"QVdigits" }
   ThresholdAttributeinAccessAccept = { vendorID = 10415, attr = 10, type =
"QVdigits" }
   sendZeroQuotaAccessAcceptOnFundsExpiry = false
   noFundsDisconnectMethod = "DisconnectRequest"
   messageAfterFinalAccessRequest = "emptyAccessAccept"

    ExtendedAttributeTypes = [
        {
        type = "QVdigits"
        pattern = "QV*"
        selectNth = 1
        firstFactor = "1"
        secondFactor = "0"
        }

        {
        type = "CiscoOverflow"
        pattern = "I*;*"
        selectNth = 1
        firstFactor = "4292967296"
        secondFactor = "1"
        }
    ]
    IncomingMapping = {
           Attributes = [
            { attr = 1, attrName = "User-Name",          attrType = "string",
    storeToTag = 6690001 }
           { attr = 2, attrName = "User-Password",      attrType = "string",
    storeToTag = 6690002 }
```

```
       { attr = 3, attrName = "CHAP-Password",      attrType = "string",
storeToTag = 6690003 }
        { attr = 4, attrName = "NAS-IP-Address",     attrType = "IPv4",
storeToTag = 6690004 }
        { attr = 5, attrName = "NAS-Port",           attrType = "integer",
storeToTag = 6690005 }
        { attr = 6, attrName = "Service-Type",       attrType = "integer",
storeToTag = 6690006 }
        { attr = 7, attrName = "Framed-Protocol",    attrType = "integer",
storeToTag = 6690007 }
        { attr = 8, attrName = "Framed-IP-Address",  attrType = "IPv4",
storeToTag = 6690008 }
        { attr = 9, attrName = "Framed-IP-Netmask",  attrType = "IPv4",
storeToTag = 6690009 }
       { attr =22, attrName = "Framed-Route",       attrType = "string",
storeToTag = 6690022 }
        { attr =25, attrName = "Class",              attrType = "string",
storeToTag = 6690025 }
       { attr =27, attrName = "Session-Timeout",    attrType = "integer",
storeToTag = 6690027 }
       { attr =28, attrName = "Idle-Timeout",       attrType = "string",
storeToTag = 6690028 }
        { attr =30, attrName = "Called-Station-ID",  attrType = "string",
storeToTag = 6690030 }
        { attr =31, attrName ="Calling-Station-ID",  attrType = "string",
storeToTag = 6690031 }
       { attr =32, attrName = "NAS-Identifier",     attrType = "string",
storeToTag = 6690032 }
        { attr =40, attrName = "Acct-Status-Type",   attrType = "integer",
storeToTag = 6690040 }
        { attr =41, attrName = "Acct-Delay-Time",    attrType = "integer",
storeToTag = 6690041 }
        { attr =42, attrName = "Acct-Input-Octets",  attrType = "integer",
storeToTag = 6690042 }
        { attr =43, attrName = "Acct-Output-Octets", attrType = "integer",
storeToTag = 6690043 }
        { attr =44, attrName = "Acct-Session-Id",    attrType = "string",
storeToTag = 6690044 }
        { attr =45, attrName = "Acct-Authentic",     attrType = "integer",
storeToTag = 6690045 }
       { attr =46, attrName = "Acct-Session-Time",  attrType = "integer",
storeToTag = 6690046 }
        { attr =47, attrName = "Acct-Input-Packets", attrType = "integer",
storeToTag = 6690047 }
        { attr =48, attrName = "Acct-Output-Packets",attrType = "integer",
storeToTag = 6690048 }
       { attr =60, attrName = "CHAP-Challenge",     attrType = "string",
storeToTag = 6690060 }
        { attr =61, attrName = "NAS-Port-Type",      attrType = "string",
storeToTag = 6690061 }
       { attr =96, attrName = "Framed-IPv6-Prefix", attrType = "string",
storeToTag = 6690096 }

     # Cisco Vendor Specific

        { vendorID=9, attr=1, keystring="csg:billing_plan=",
attrName="Rulebase-ID", attrType="string", storeToTag = 6820001 }
        { vendorID=9, attr=1, keystring="csg:quota_server=", attrName="Quota-
Server-Address", attrType="string", storeToTag = 6820002 }
        { vendorID=9, attr=1, Keystring="csg:downlink_nexthop=", attrName =
"Downlink-Nexthop", attrType="string", storeToTag = 6820003 }

      # 3GPP Vendor Specific
```

```
        { vendorID = 10415, attr = 1, attrName = "3GPP-IMSI",        attrType
= "string", storeToTag = 6760001 }
        { vendorID = 10415, attr = 2, attrName = "3GPP-Charging_Id", attrType
= "string", storeToTag = 6760002 }
        { vendorID = 10415, attr = 3, attrName = "3GPP-PDPType",     attrType
= "integer", storeToTag = 6760003 }
        { vendorID = 10415, attr = 4, attrName = "3GPP-CG-Address",  attrType
= "IPv4", storeToTag = 6760004 }
       { vendorID = 10415, attr = 5, attrName = "3GPP-GPRS-QoS-Profile",
attrType = "string", storeToTag = 6760005 }
        { vendorID = 10415, attr = 6, attrName = "3GPP-SGSN-Address",attrType
= "IPv4", storeToTag = 6760006 }
        { vendorID = 10415, attr = 7, attrName = "3GPP-GGSN-Address",attrType
= "IPv4", storeToTag = 6760007 }
        { vendorID = 10415, attr = 8, attrName = "3GPP-IMSI-MCC-MNC",attrType
= "string", storeToTag = 6760008 }
        { vendorID = 10415, attr = 9, attrName = "3GPP-GGSN-MCC-MNC",attrType
= "string", storeToTag = 6760009 }
        { vendorID = 10415, attr =10, attrName = "3GPP-NSAPI",
attrType= "string", storeToTag = 6760010 }
        { vendorID = 10415, attr =12, attrName = "3GPP-Selection-
Mode",attrType = "string", storeToTag = 6760012 }
        { vendorID = 10415, attr =13, attrName = "3GPP-Charging-
Characteristics",attrType = "string", storeToTag = 6760013 }
        { vendorID = 10415, attr =18, attrName = "3GPP-SGSN-MCC-MNC",attrType
= "string", storeToTag = 6760018 }
        { vendorID = 10415, attr =20, attrName = "3GPP-IMEISV",      attrType
= "octets", storeToTag = 6760020 }
        { vendorID = 10415, attr =21, attrName = "3GPP-RAT-Type",    attrType
= "integer", storeToTag = 6760021 }
        { vendorID = 10415, attr =22, attrName = "3GPP-User-Loc-
Info",attrType = "octets", storeToTag = 6760022 }
        { vendorID = 10415, attr =23, attrName = "3GPP-MS-TimeZone",attrType
= "octets", storeToTag = 6760023 }
      ]
}
ValueEditingRules = [
    {
    attrName = "User-Name"
    ifStartsWith = "ADMIN"
    removeFront = 5
    ifEndsWith = ".com"
    removeBack = 4
    }

]
DecryptTags = [
    { fromTag = 6690002, decryptToTag = 6760099 }
]
HardcodedValues = [
    {tag=6820005, tagType="string", value = "gprs:teardown-ind=yes" }
    {tag=6820006, tagType="string", value = "PrimaryDNSserver" }
    {tag =327689, tagType="string", value = "1111" }
    {tag =327690, tagType="string", value = "027774444" }
    {tag =327691, tagType="string", value = "2008" }
    {tag =327713, tagType="string", value = "49393404" }
    {tag =327801, tagType="string", value = "88888" }
    {tag =327703, tagType="string", value = "777777" }
    {tag =327868, tagType="string", value = "666666" }
]

ReleaseCall = [
```

```
    {code=41, vendorID=9, attr=1, tagType="string", value = "gprs:teardown-
    ind=yes" }
    {code=31, vendorID=9, attr=1, tagType="string", value = "gprs:teardown-
    ind=no" }
]
InitialDPMapping = {
    callingPartyNumber = {
        tag = 6690031

        NumberRules = [
            { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
            { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
            { prefix="0", min=9, remove=1, resultNoa=3 }
            { prefix="", remove=0, resultNoa=3 }
        ]
    }
    calledPartyBCDNumber = {
        tag = 327689
        NumberRules = [
            { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
            { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
            { prefix="0", min=9, remove=1, resultNoa=3 }
            { prefix="", remove=0, resultNoa=3 }
        ]
    }
    locationNumberTag = {
        tag = 0
        NumberRules = [
            { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
            { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
            { prefix="0", min=9, remove=1, resultNoa=3 }

        ]
    }
    callReferenceNumberTag = 0
}

InitialDPExtensions = [6760003,6760006,6690004]

ConnectMapping = {
    destinationRoutingAddress = {
        tag = 327689

        NumberRules = [
            { fromNoa=4,    remove=0, prepend="00" }
            { fromNoa=3,    remove=0, prepend="0064" }
        ]
    }
    originalCalledPartyID = {
        tag = 327713

        NumberRules = [
            { fromNoa=4,    remove=0, prepend="00" }
            { fromNoa=3,    remove=0, prepend="0064" }
        ]
    }
    callingPartysCategoryTag = 327801
    redirectingPartyId = {
        tag = 327703

        NumberRules = [
            { fromNoa=4,    remove=0, prepend="00" }
            { fromNoa=3,    remove=0, prepend="0064" }
```

```
            ]
        }
        genericNumbersAdditionalCallingPartyNumber = {
            tag = 327868

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
    }

    ConnectExtensions = [
        { vendorID = 311, attr = 30, attrName = "PrimaryNBNSserver",  attrType =
        "IPv4", profileBlockTag = 6950030 }
    ]

    OutgoingMappingtoSwitch = {
        AccessAcceptMapping = [
            {attr = 6, attrName = "Service-Type",     attrType = "integer",
            fromTag = 6690006 }
            {attr = 7, attrName = "Framed-Protocol",  attrType = "integer",
            fromTag = 6690007 }
            {attr = 8, attrName = "Framed-IP-Address",attrType = "IPv4",
            fromTag = 6690008 }
            {attr = 9, attrName = "Framed-IP-Netmask",attrType = "IPv4",
            fromTag = 6690009 }
            {attr =22, attrName = "Framed-Route",     attrType = "integer",
            fromTag = 6690022 }
            {attr =25, attrName = "Class",            attrType = "string",
            fromTag = 6690025 }
            {attr =27, attrName = "Session-Timeout",  attrType = "integer",
            fromTag = 6690027 }
            {attr =28, attrName = "Idle-Timeout",     attrType = "integer",
            fromTag = 6690028 }
            {vendorID = 311, attr = 30, attrName = "PrimaryNBNSserver",attrType =
            "IPv4", fromTag = 6950030 }
            {vendorID = 9, attr = 1, attrName = "Tear-down-indicator",attrType =
            "string", fromTag = 6820005 }
        ]
        DisconnectRequestMapping = [
            {vendorID = 10415, attr = 1, attrName = "3GPP-IMSI",   attrType =
            "string", fromTag = 6760001 }
            {vendorID = 10415, attr = 10,attrName = "3GPP-NSAPI",  attrType =
            "string", fromTag = 6760010 }
            {vendorID = 9,     attr = 1, attrName = "TeardownInd", attrType =
            "string", fromTag = 6820005 }
            {attr = 44,attrName = "Acct-Session-Id", attrType = "string", fromTag
            = 6690044 }
            {attr = 6,attrName = "Service-Type", attrType = "integer", fromTag =
            6690006 }
            {attr = 4,attrName = "NAS-IP-Address", attrType = "IPv4", fromTag =
            6690004 }
            {attr = 8,attrName = "Framed-IP-Address", attrType = "IPv4", fromTag
            = 6690008 }
            {attr = 18,attrName = "originalCalledPartyID", attrType = "string",
            fromTag = 327713 }
        ]
    }

# Charging

    UseUnitsfromAccessRequest = false
    UseFinalUnitsFromAccountingRequest = true
```

```
CumulativeConversionRules = false
AccountingRequestConversionRules = [
    { ruleName = "rule1",
            attributeList = [
            { attributeTag = 6690042, multiplier = 0.00001 } # Deciseconds in
            ACR = 0.00001 * input-octets +
            { attributeTag = 6690043, multiplier = 0.00001 } #
            0.00001 * output-octets
        ]
    }

    { ruleName = "rule2",
        attributeList = [
            { attributeTag = 6690046, multiplier = 10 } # Deciseconds in ACR
            = 10 * session-time
        ]
    }
]
conversionFactor = 100000.0
clientThresholdQuotaGap = 100

SessionInactiveRules = [
    {AttributeValuePairs = [
            { attr = 81, attrType = "string", value = "2" } # and
            { attr = 82, attrType = "integer", value = 4 } # and
            { vendorID = 10415, attr = 21, attrType = "integer", value = 4 }
        ]
    } # or
    {AttributeValuePairs = [
            { attr = 81, attrType = "string", value = "2" } # and
            { attr = 82, attrType = "integer", value = 4 } # and
            { vendorID = 10415, attr = 21, attrType = "integer", value = 3 }
        ]
    }
    ]
ServiceKeyRules = [
    {AttributeValuePairs = [
            { attr = 61, attrType = "string", value = "2" }
            { attr = 62, attrType = "integer", value = 4 }
            { vendorID = 10415, attr = 21, attrType = "integer", value = 4 }
        ]
    sleeServiceKey=231
    inapServiceKey=231
    }
]

CorrelationAttributes = {
    AccountingRequest = [
        { vendorID = 10415, attr = 2 }  # 3GPP-Charging-Id
        {                 , attr = 1 }
        {                 , attr = 6 }
    ]
    AccessRequest = [
        { vendorID = 10415, attr = 2 }  # 3GPP-Charging-Id
        {                 , attr = 1 }
        {                 , attr = 6 }
    ]
}

systemErrorAttribute = {
    attr = 0 # attr = 0 means do not add any such attribute
    attrType="string"
    value = "System Error"
```

```
        }

        invalidMessageSequenceAttribute = {
            attr = 0 # attr = 0 means do not add any such attribute
            attrType="string"
            value = "Invalid Message Sequence"
        }

        insufficientFundsAttribute = {
            attr = 0 # attr = 0 means do not add any such attribute
            attrType="string"
            value = "Insufficient Funds"
        }

        timeoutAttribute = {
            attr = 0 # attr = 0 means do not add any such attribute
            attrType="string"
            value = "Session Timed Out"
        }

        replyMessagestoAttribute = 0     # 18 would seem sensible but Cisco do not
        support it

        sessionTimeout = 3600
        idleTimeout = 600
        eventTimestampInAccessAccept = false

    } #end ParameterisedRCAFields
```

ParameterisedRCAFields parameters

The parameters available in this section are:

sendAccessRejectMessages

Syntax: `sendAccessRejectMessages = true|false`

Description: Determines if RCA can send Access-Rejects messages to Cisco.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true – Access-Reject messages can be sent
- false – Access-Reject messages cannot be sent

Default:

Notes: If set to false, then Disconnect-Request messages are used instead.

Example: `sendAccessRejectMessages = false`

sendAccountingResponses

Syntax: `sendAccountingResponses = true|false`

Description: Determines if Accounting-Response messages are sent in response to Accounting-Request.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true
- false

Default:

Notes:

Example: `sendAccountingResponses = false`

expectResponsesFromDisconnectRequest

Syntax: `expectResponsesFromDisconnectRequest = true|false`

Description: Determines if RCA expects the Cisco client to send Disconnect-ACK and Disconnect-NAK messages.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true
- false

Default:

Notes:

Example: `expectResponsesFromDisconnectRequest = true`

useQuotainAccessAccept

Syntax: `useQuotainAccessAccept = true|false`

Description: Determines if RCA indicates the total usage quota for a user, when sending an Access-Accept.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true
- false

Default:

Notes:

Example: `useQuotainAccessAccept = false`

useThresholdinAccessAccept

Syntax: `useThresholdinAccessAccept = true|false`

Description: When RCA sends an Access-Accept message, this parameter determines if a quota threshold is added to show how much the user is allowed to use before sending another Access-Request.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true
- false

Default:

Notes:

Example: `useThresholdinAccessAccept = false`

QuotaAttributeinAccessAccept

Syntax:
```
QuotaAttributeinAccessAccept = {
    vendorID = int, attr = int, type = "str"
}
```

Description: Indicates the attribute in the Access-Accept messages which stores the usage quota.

Type: Structure

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: When the optional `type` parameter is set for `QuotaAttributeinAccessAccept`, then these AVPs are interpreted according to the extended types rules.

Example:
```
QuotaAttributeinAccessAccept = { vendorID = 12345, attr =
10, type = "QVdigits" }
```

ThresholdAttributeinAccessAccept

Syntax:
```
ThresholdAttributeinAccessAccept = {
    vendorID = int, attr = int, type = "str"
}
```

Description: Indicates the attribute in the Access-Accept messages which stores the quota threshold.

Type: Structure

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: When the optional `type` parameter is set for `ThresholdAttributeinAccessAccept`, then these AVPs are interpreted according to the extended types rules.

Example:
```
ThresholdAttributeinAccessAccept = { vendorID = 12345, attr
= 10 , type = "QVdigits" }
```

sendZeroQuotaAccessAcceptOnFundsExpiry

Syntax:
```
sendZeroQuotaAccessAcceptOnFundsExpiry = true|false
```

Description: Sets whether to allow termination on fund expiry by sending an Access-Accept and then waiting for an Accounting Request.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true – Allow termination on fund expiry by sending an Access- Accept
- false – Do not allow termination on fund expiry

Default: false

Notes:

Example:
```
sendZeroQuotaAccessAcceptOnFundsExpiry = false
```

noFundsDisconnectMethod

Syntax:
```
noFundsDisconnectMethod = "str"
```

Description: The method (message) used to terminate connections if no funds are available.

Type: String

Optionality: Optional (default used if not set)

Allowed: DisconnectRequest
AccountingResponseSTOP
AccessReject

Default:

Notes:

Example:
```
noFundsDisconnectMethod = "DisconnectRequest"
```

messageAfterFinalAccessRequest

Syntax: `messageAfterFinalAccessRequest = "str"`

Description: Indicates which message to send when receiving Access-Request for an inactive session.

Type: String

Optionality: Optional (default used if not set)

Allowed: emptyAccessAccept
disconnectRequest
noMessage

Default:

Notes:

Example: `messageAfterFinalAccessRequest = "emptyAccessAccept"`

UseUnitsfromAccessRequest

Syntax: `UseUnitsfromAccessRequest = true|false`

Description: Determines if the units used by the session are derived from subsequent Access-Request messages or Accounting-Request messages.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true – Use Access-Request messages
- false – Use Accounting-Request messages

Default:

Notes:

Example: `UseUnitsfromAccessRequest = false`

UseFinalUnitsFromAccountingRequest

Syntax: `UseFinalUnitsFromAccountingRequest = true|false`

Description: Set to override UseUnitsfromAccessRequest.

Type: Boolean

Optionality: Optional (default used if not set).

Allowed:
- true – Get all except final units from access request, get final units from accounting request.
- false – Do not override UseUnitsfromAccessRequest.

Default: false

Notes:

Example: `UseFinalUnitsFromAccountingRequest = true`

CumulativeConversionRules

Syntax: `CumulativeConversionRules = true|false`

Description: If this is set to true then the result of AccountingRequestConversionRules is added on to the amount already used.

If this is set to false then the result is taken as the total amount used for the session.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:
- true - add result to the amount
- false - use result as total amount for session

Default: false

Notes:

Example:
```
CumulativeConversionRules = true
```

AccountingRequestConversionRules

Syntax:
```
AccountingRequestConversionRules = [
          {ruleName = "str",
          attributeList = [
                { attributeTag = int, multiplier = int }
                { attributeTag = int, multiplier = int }
            ]
          }
          { ruleName = "str",
            attributeList = [
                { attributeTag = int, multiplier = int }
            ]
          }
      ]
```

Description: The given attributes are multiplied by the multipliers and then summed to give a deciseconds equivalent.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:
```
AccountingRequestConversionRules = [
          {ruleName = "rule1",
          attributeList = [
                { attributeTag = 6690042, multiplier =
        0.00001 }
                { attributeTag = 6690043, multiplier =
        0.00001 }
            ]
          }
          { ruleName = "rule2",
            attributeList = [
                { attributeTag = 6690046, multiplier = 10 }
            ]
          }
      ]
```

conversionFactor

Syntax: `conversionFactor = int`

Description: The multiplication factor that is applied to the Apply charging deciseconds.

Type: Integer

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:
```
conversionFactor = 100000.0
```

clientThresholdQuotaGap

Syntax: `clientThresholdQuotaGap = int`

Description: The difference between the quota and the threshold specified in the Access-Accept message, where:

`thresold = quota - clientThresholdQuotaGap`

Type: Integer

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: For parameterised clients, the units of clientThresholdQuotaGap are decisecond equivalents.

Example:
```
clientThresholdQuotaGap = 100
```

SessionInactiveRules

Syntax:
```
SessionInactiveRules = [
    {AttributeValuePairs = [
        { attr = int, attrType = "str", value = int }
        { attr = int, attrType = "str", value = int }
        { vendorID = int, attr = int, attrType = str, value = int
        }
    ]}
]
```

Description: If the Access-request matches any of the specified rules, then the session is not active.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:
```
SessionInactiveRules = [
    {AttributeValuePairs = [
        { attr = 81, attrType = "string", value = "2" }
        { attr = 82, attrType = "integer", value = 4 }
        { vendorID = 10415, attr = 21, attrType = "integer", value
        = 4 }
    ]}
]
```

ServiceKeyRules

Syntax:
```
ServiceKeyRules = [
    {AttributeValuePairs = [
        { attr = int, attrType = "str", value = int }
        { attr = int, attrType = "str", value = int }
        { vendorID = int, attr =" int, attrType = "str", value =
        int }
    ]
    sleeServiceKey=int
    inapServiceKey=int
    }
]
```

Description: The rules for setting the service keys, if different from the defaults defined in the global RCA section.

Type: Array

Optionality: Optional
Allowed:
Default:
Notes:
Example:

```
ServiceKeyRules = [
        {AttributeValuePairs = [
        { attr = 61, attrType = "string", value = "2" }
        { attr = 62, attrType = "integer", value = 4 }
        { vendorID = 10415, attr = 21, attrType = "integer", value
        = 4 }
    ]
    sleeServiceKey=231
    inapServiceKey=231
    }
]
```

SessionInactiveRules

Syntax:

```
ReleaseCall = [
    { code = int, vendorID = int, attr = int, tagType = "str",
    value = tagType }

]
```

Description: Maps release call causes into specific attributes, before being put into outgoing messages such as outgoing Disconnect-Request or Access-Reject messages.
Type: Array
Optionality: Optional
Allowed:
Default: None
Notes:
Example:

```
ReleaseCall = [
    { code = 41, vendorID = 9, attr = 1, tagType = "string" value
    = "gprs:teardown-ind=no" }
    { code = 31, vendorID = 9, attr = 1, tagType = "string" value
    = "gprs:teardown-ind=yes" }
    ]
```

CorrelationAttributes

Syntax: See example
Description: Attributes specified here are used to correlate incoming Access-Request or Account-Request messages with existing sessions.
Type: Array
Optionality: Optional (default used if not set)
Allowed:
Default:
Notes:
Example:

```
CorrelationAttributes = {
    AccountingRequest = [
        { vendorID = 10415, attr = 2,
        attrType = "string",
        stripPrefix = "parent-session-id=",
        stripSuffix = ""
        }
       ]
    AccessRequest = [
        { vendorID = 10415, attr = 2,
```

```
        attrType = "string",
        stripPrefix = "",
        stripSuffix = ""
        }
        ]
    }
```

systemErrorAttribute

Syntax:

```
systemErrorAttribute = {
    attr = int
    attrType = "str"
    value = "str"
    }
```

Description: The attribute to be added to the following when the session is terminated due to a system error that is not caused by the radius client:

- Disconnect-Request
- Access-Reject or
- Accounting-Request (STOP)

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:

```
systemErrorAttribute = {
    attr = 1
    attrType = "string"
    value = "System Error"
    }
```

invalidMessageSequenceAttribute

Syntax:

```
invalidMessageSequenceAttribute = {
    attr = int
    attrType = "str"
    value = "str"
    }
```

Description: The attribute to be added to the following when the session is terminated due to an invalid Radius message sequence, i.e, the error is caused by the radius client:

- Disconnect-Request
- Access-Reject or
- Accounting-Request (STOP)

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:

```
invalidMessageSequenceAttribute = {
    attr = 0
    attrType = "string"
    value = "Invalid Message Sequence"
}
```

insufficientFundsAttribute

Syntax:
```
insufficientFundsAttribute = {
    attr = int
    attrType = "str"
    value = "str"
}
```

Description: The attribute to be added to the following messages when the session is terminated due to insufficient funds:

- Disconnect-Request
- Access-Reject or
- Accounting-Request (STOP)

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:
```
insufficientFundsAttribute = {
    attr = 0
    attrType = "string"
    value = "Insufficient Funds"
}
```

timeoutAttribute

Syntax:
```
timeoutAttribute = {
    attr = 0
    attrType="string"
    value = "Session Timed Out"
}
```

Description: The attribute to be added to the following when the session is terminated due to timeout:

- Disconnect-Request
- Access-Reject or
- Accounting-Request (STOP)

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:
```
timeoutAttribute = {
    attr = 0
    attrType="string"
    value = "Session Timed Out"
}
```

replyMessagestoAttribute

Syntax: `replyMessagestoAttribute = int`

Description: The Attribute number to send reply messages in.

Type: Integer

Optionality: Optional (default used if not set).

Allowed:

Default:

Notes: A value of `0` ensures the Reply Messages is not sent at all.

Example: `replyMessagestoAttribute = 0`

`sessionTimeout`

Syntax: sessionTimeout `= seconds`

Description: The number of seconds to wait for a subsequent Access-Accept message before terminating the session.

Type: Integer

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example: sessionTimeout = 3600

`idleTimeout`

Syntax: `idleTimeout = seconds`

Description: The number of seconds to maintain state of inactivity before terminating the session.

Type: Integer

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example: `idleTimeout = 600`

`eventTimestampInAccessAccept`

Syntax: eventTimestampInAccessAccept `= true|false`

Description: Determines whether to add the event timestamp core radius field in the Access-Accept message.

Type: Boolean

Optionality: Optional (default used if not set)

Allowed:

- true
- false

Default:

Notes:

Example: eventTimestampInAccessAccept `= false`

QualifyingAttributeRules

Introduction

This section specifies the rules which must be matched before this client configuration will be selected for a given Radius message.

Each rule set is a list of AVPs, all of which must be matched to the contents of the incoming message. If any rule set matches then the message can be processed by this client type, otherwise the RCA will look for another client associated with the same IP address.

`QualifyingAttributeRules` is a part of the `ParameterisedRCAFields` section.

Example

Here is an example of the QualifyingAttributeRules configuration section.

```
        QualifyingAttributeRules = [
            { AttributeValuePairs = [
                {attr = 81, attrType = "string", value = "abc" }
                {vendorID = 10415, attr = 21, attrType = "integer", value = 4 }
                ]
            }
            { AttributeValuePairs = [
                { attr = 181, attrType = "string", value = "2" }
                { attr = 182, attrType = "integer", value = 4 }
                { vendorID = 10415, attr = 121, attrType = "integer", anyValue =
                true}
                ]
            }
    ]
```

QualifyingAttributeRules parameters

The parameters available in this section are:

`AttributeValuePairs`

Syntax:

```
AttributeValuePairs = [
    { attr = int , attrType = "type" , value = "str"|int}
    { vendorID = int , attr = int, attrType = "type", anyValue =
    true|false }
    ]
```

Description: Defines an attribute rule for matching AVPs. An AVP will match if its value in the incoming message agrees with the value configured here. If the rule includes the parameter `anyValue=true` then any value will be accepted.

Type: Array

Optionality: Required

Allowed:

Default:

Notes: For more information about the attr, attrType, and vendorID parameters, see *IncomingMapping section* (on page 49).

Example:

```
AttributeValuePairs = [
    { attr = 181 , attrType = "string" , value = "2"}
    { vendorID = 10415 , attr = 121, attrType = "integer",
    anyValue = true }
]
```

`value`

Syntax: `value = attrType_value`

Description: The value to match against the AVP value.

Type: Dependent on the value of `attrType`.

Optionality: Optional

Allowed:

Default:

Notes:

Example: `value = "abc"`
`value = 4`

anyValue

Syntax: `anyValue = true|false`
Description: When set to true, any value for the AVP will be matched.
Type: Boolean
Optionality: Optional
Allowed:
Default:
Notes:
Example: `anyValue = true`

ExtendedAttributeTypes section

Introduction

This section defines extended AVP types which can be referred to in `IncomingMapping` and `OutgoingMapping` through the `attrType` parameter.

`ExtendedAttributeTypes` is a part of the `ParameterisedRCAFields` section.

Example usage

ExtenedAttributeTypes configuration allows you to configure an IncomingMapping.Attributes item where, for example, attrType = "QVdigits". The RCA will handle this configuration by transforming the incoming string to a 32-bit integer. It will ignore the "QV" and pick out the digits that follow it. The decodeAttributesFromRadiusMessage code will detect where the type is extended (i.e. where the type is not one of the defined types) and hand off to a helper function to do the processing and return the TypeAndValue object. For example:

```
ExtenedAttributeTypes = [
    {
        type = "QVdigits"
        pattern = "QV*"
        selectNth = 1
        firstFactor = "0"
        secondFactor = "1"
    }
]
```

You can also use ExtendedAttributesType configuration to define patterns with more than one group of fixed digits, and instruct the code which group to extract. For example:

```
ExtendedAttributeTypes = [
        {
        type = "QUDdigits"
        pattern = "QU;*; D;*;*"
        selectNth = 2
        firstFactor = "0"
        secondFactor = "1"
    }
]
```

This configuration will process the input value: `QU;512000;256000; D;512000;25600` as follows and pick out the last number:

- extract the second group of numbers
- zeroise the first number by multiplying it by the firstFactor value (0), and
- extract the second number.

Note: Numbers are always multiplied by the applicable factor and then added together.

Example ExtendedAttributeTypes configuration

```
ExtendedAttributeTypes = [
    {
    type = "QVdigits"
    pattern = "QV*"
    selectNth = 1
    firstFactor = "1"
    secondFactor = "0"
    }

    {
    type = "CiscoOverflow"
    pattern = "I*;*"
    selectNth = 1
    firstFactor = "4292967296"
    secondFactor = "1"
    }
]
```

ExtendedAttributeTypes parameters

The parameters available in this section are:

type

Syntax: `type = "str"`

Description: Defines the name of the extended AVP type.

Type: String

Optionality: Optional

Allowed:

Default: None

Notes:

Example: `type = "QVdigits"`

pattern

Syntax: `pattern = "str"`

Description: Defines the pattern to use to help extract the extended AVP type.

Type: String

Optionality: Optional

Allowed:

Default: None

Notes:

Example: `pattern = "QV;*;*"`

selectNth

Syntax: `selectNth = int`
Description: Where fields repeat, defines which fields to use.
Type: Integer
Optionality: Optional
Allowed:
Default:
Notes:
Example: `selectNth = 1`

firstFactor

Syntax: `firstFactor = "int"`
Description: Where there are two numbers this is the multiplication factor to apply to the first number.
Type: String
Optionality: Optional
Allowed: A numeric value
Default:
Notes:
Example: `firstFactor = "1"`

secondFactor

Syntax: `secondFactor = "int"`
Description: Where there are two numbers this is the multiplication factor to apply to the second number.
Type: String
Optionality: Optional
Allowed: A numeric value
Default:
Notes:
Example: `secondFactor = "4378"`

IncomingMapping section

Introduction

This section specifies how the attributes are copied from each Radius message received into the attribute map for each session in RCA.

IncomingMapping is a part of the ParameterisedRCAFields section.

Notes:

- All attributes used by RCA, whether to send them to ACS, or put them in an outgoing Radius message must be specified in this section.

- Although arbitrary tags may be specified for this section, it is strongly suggested that the convention followed in the eserv.config be used as it matches the pre-provisioned tags in the ACS_PROFILE_DETAILS table. For more information on ACS_PROFILE_DETAILS table, see *ACS Technical Guide*.

Parameters

The parameters available in this section are:

Attributes

Syntax:
```
Attributes = [
            { attr = 1, attrName = "User-Name",          attrType
    = "string",  storeToTag = 6690001 }
]
```

Description: Contains the list of attributes obtained from the special extension incoming on the InitialDP.

Type: Array

Optionality: Optional (default used if not set).

Allowed:

Default:

Notes:

Example:
```
Attributes = [
            { attr = 1, attrName = "User-Name",          attrType
    = "string",  storeToTag = 6690001 }
]
```

attr

Syntax: `attr = int`

Description: Attribute number.

Type: Integer

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:
```
attr = 1
```

attrName

Syntax: `attrName = "str"`

Description: Name or description of the attribute supported by radius.

Type: String

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example:
```
attrName = "User-Name"
```

attrType

Syntax: `attrType = "str"`

Description: The data type of the attribute.

Type: String

Optionality: Optional (default used if not set)

Allowed: Specify one of:

- string – If the attribute value is an UTF-8 encoded string
- octets – If the attribute value is an octet string
- integer – If the attribute value is an integer, an enum, or a framed integer
- IPv4 – If the attribute value is an IP version 4 (IPv4) address in the range of 00.00.00.00 to FF.FF.FF.FF, encoded in 4 octets
- IPv6 – If the attribute value is an IP version 6 address (IPv6) in the range of 0000.0000.0000.0000 to FFFF.FFFF.FFFF.FFFF, encoded in 16 octets

Default:

Notes:

Example: `attrType = "integer"`

storeToTag

Syntax: `storeToTag = tag`

Description: Decimal value of the RCA tag to which the attribute is stored.

Type: Numeric string

Optionality: Optional

Allowed:

Default:

Notes:

Example: `storeToTag = 6690003`

vendorID

Syntax: `vendorID = int`

Description: The ID assigned to vendor specific tags as per industrial standards.

Type: Numeric string

Optionality: Optional

Allowed: Any valid vendor ID, for example:

- Cisco - 9
- 3GPP - 10415
- Microsoft - 311

Default:

Notes:

Example: `vendorID = 10415`

ValueEditingRules section

Introduction

This section defines rules for editing incoming values before they are placed in the map. Checks can be made on both prefix and suffix values within the same rule.

ValueEditingRules is a part of the ParameterisedRCAFields section.

Example

```
ValueEditingRules = [
    {
    attrName = "User-Name"
    ifStartsWith = "ADMIN"
    removeFront = 5
    ifEndsWith = ".com"
    removeBack = 4
    }

]
```

ValueEditingRules parameters

The parameters available in this section are:

attrName

Syntax: `attrName = "str"`

Description: The name of the attribute to apply the rule to.

Type: String

Optionality: Optional

Allowed:

Default: None

Notes:

Example: `attrName = "User-Name"`

ifStartsWith

Syntax: `ifStartsWith = "str"`

Description: Defines the prefix to look for. If defined, then the `removeFront` parameter must also be defined.

Type: String

Optionality: Optional

Allowed:

Default: None

Notes:

Example: `ifStartsWith = "ADMIN"`

removeFront

Syntax: `removeFront = int`

Description: Defines the number of prefix characters to remove. May only be used in conjunction with the `ifStartsWith` parameter.

Type: Integer

Optionality: Optional.

Allowed:

Default: None

Notes:

Example: `removeFront = 5`

ifEndsWith

Syntax: `ifEndsWith = "str"`

Description: The suffix to look for. If specified, then the `removeBack` parameter must also be specified.

Type: String

Optionality: Optional

Allowed:

Default: None.

Notes:

Example: `ifEndsWith = ".com"`

removeBack

Syntax: `removeBack = int`

Description: The number suffix characters to remove. Must be used in conjunction with the `ifEndsWith` parameter.

Type: Integer

Optionality: Optional

Allowed:

Default:

Notes:

Example: `removeBack = 4`

DecryptTags

Introduction

`DecryptTags` takes an attribute item such as a password, from the incoming mapping and uses it to create a new decrypted attribute item.

`DecryptTags` is a part of the `ParameterisedRCAFields` section.

Example

```
DecryptTags = [
    { fromTag = 6690002, decryptToTag = 6760099 }
]
```

DecryptTags parameters

The parameters available in this section are:

fromTag

Syntax: `fromTag = int`

Description: Defines the tag where the attribute item to be decrypted is stored.

Type: Integer

Optionality: Optional

Allowed:

Default: None
Notes:
Example: `fromTag = 6690002`

decryptToTag

Syntax: `decryptToTag = int`
Description: Defines the tag where the new decrypted attribute item will be stored.
Type: Integer
Optionality: Optional
Allowed:
Default:
Notes:
Example: `decryptToTag = 6760099`

HardcodedValues

Description

This section specifies the hard-coded values which are always put into the attribute map, at the start of each session. This is a part of the ParameterisedRCAFields section.

Hard coded values are specified for the following parameters:

Example

Here is an example of the HardcodedValues configuration section.

```
HardcodedValues = [
    {tag=6820005, tagType="string", value = "gprs:teardown-ind=yes" }
    {tag=6820006, tagType="string", value = "PrimaryDNSserver" }
    {tag =327689, tagType="string", value = "1111" }
    {tag =327690, tagType="string", value = "027774444" }
    {tag =327691, tagType="string", value = "2008" }
    {tag =327713, tagType="string", value = "49393404" }
    {tag =327801, tagType="string", value = "88888" }
    {tag =327703, tagType="string", value = "777777" }
    {tag =327868, tagType="string", value = "666666" }
]
```

Parameters

The parameters available in this section are:

tag

Syntax: `tag = tag`
Description: Decimal value of the RCA tag in the attribute map to which the hard-coded value is stored.
Type: Numeric string
Optionality: Optional (default used if not set)
Allowed:
Default:

Notes:

Example: `tag = 6780007`

tagType

Syntax: `tagType = "str"`

Description: The data type of the RCA tag in the attribute map. You specify the tag value in the *value* parameter.

Type: String

Optionality: Optional (default used if not set)

Allowed:

- string – The value specified in the *value* parameter is an UTF-8 encoded string
- octets – The value specified in the *value* parameter is an octet string
- integer – The value specified in the *value* parameter is an integer, an enum or a "framed" integer
- IPv4 – The value specified in the *value* parameter is an IP version 4 address in the range of 00.00.00.00 to FF.FF.FF.FF, encoded in 4 octets
- IPv6 – The value specified in the *value* parameter is an IP version 6 address in the range of 0000.0000.0000.0000 to FFFF.FFFF.FFFF.FFFF, encoded in 16 octets

Default:

Notes:

Example: `tagtype = "string"`

value

Syntax: `value = "str"`

Description: The hard-coded value stored in the attribute map.

Type: String

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example: `value = "1111"`

InitialDPMapping

Introduction

The InitialDPMapping section specifies how values are obtained from the attribute map to the InitialDP which RCA sends to ACS. This is a part of the ParameterisedRCAFields section.

Here is an example of the InitialDPMapping configuration section:

```
ParameterisedRCAFields = {
    InitialDPMapping = {
        callingPartyNumber = {
            tag = 6690031

            NumberRules = [
                { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
```

```
                { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
                { prefix="0", min=9, remove=1, resultNoa=3 }
                { prefix="", remove=0, resultNoa=3 }
            ]
        }
        calledPartyBCDNumber = {
            tag = 327689
            NumberRules = [
                { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
                { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
                { prefix="0", min=9, remove=1, resultNoa=3 }
                { prefix="", remove=0, resultNoa=3 }
            ]
        }
        locationNumberTag = {
            tag = 0
            NumberRules = [
                { prefix="0064",  min=11, remove=4, prepend="", resultNoa=3 }
                { prefix="00",  min=5, remove=2, prepend="", resultNoa=4 }
                { prefix="0", min=9, remove=1, resultNoa=3 }

            ]
        }
        callReferenceNumberTag = 0
    }

    InitialDPExtensions = [6760003,6760006,6690004]

    }
```

Parameters

The parameters available in this section are:

callingPartyNumber

Syntax:

```
callingPartyNumber = {
                tag = int
                  NumberRules = []
            }
```

Description: Maps the calling party number values from the attribute map to RCA tag.

- tag maps to the RCA specific ACS primary tag.
- NumberRules define denormalisation rules of the calling party number.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable.

Example:

```
callingPartyNumber = {
                tag = 123456
                NumberRules = [
                {fromNoa=4,remove=0, prepend="00" }
                { fromNoa=3,remove=0, prepend="0064" }
                ]
            }
```

calledPartyBCDNumber

Syntax:

```
calledPartyBCDNumber = {
                tag = int
                  NumberRules = []
        }
```

Description: Maps the calledParty BCDNumber values from the attribute map to RCA tag:

- tag maps to the RCA specific ACS primary tag.
- NumberRules define denormalisation rules of the calledPartyBCDNumber.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable. For more information, see *CalledPartyBCDNumber* (on page 19).

Example:

```
calledPartyBCDNumber = {
                tag = 103456
                NumberRules = [
                {fromNoa=4,remove=0, prepend="00" }
                { fromNoa=3,remove=0, prepend="0064" }
                ]
            }
```

locationNumberTag

Syntax:

```
locationNumberTag  = {
    tag = int
    NumberRules = []
        }
```

Description: Maps the locationNumberTag values from the attribute map to RCA tag:

- tag maps to the RCA specific ACS primary tag. A value of 0 (zero) means do not put a location number into the IDP.
- NumberRules define denormalisation rules of the locationNumberTag.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable.

A tag value of 0 (zero) means do not put a location number in the IDP.

Example:

```
locationNumberTag  = {
    tag = 123455
}
```

callReferenceNumberTag

Syntax:

```
callReferenceNumberTag  = int
```

Description: Maps the callReferenceNumberTag values from the attribute map to RCA tag.

Type: Numeric string

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: A value of 0 (zero) means do not put a call reference number in the IDP.

Example: `callReferenceNumberTag  = 789654`

InitialDPExtensions

Syntax: `InitialDPExtensions = [tag1, tag2, tag3]`

Description: Specifies the tags to copy from the attribute map into the extension 701 profile block in the IDP.

The extension type 701 is used to encode profile blocs in the InitialDP and Connects operations.

Type: Numeric String

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes:

Example: `InitialDPExtensions = [6890003,6450006,6660006]`

ConnectMapping

Introduction

ConnectMapping specifies the values to copy into the attribute map from the Connect message sent by ACS to RCA. This is a part of the ParameterisedRCAFields section.

Here is an example of the ConnectMapping section:

```
ParameterisedRCAFields = {
    ConnectMapping = {
        destinationRoutingAddress = {
            tag = 327689

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
        originalCalledPartyID = {
            tag = 327713

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
        callingPartysCategoryTag = 327801
        redirectingPartyId = {
            tag = 327703

            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
        genericNumbersAdditionalCallingPartyNumber = {
            tag = 327868
```

```
            NumberRules = [
                { fromNoa=4,    remove=0, prepend="00" }
                { fromNoa=3,    remove=0, prepend="0064" }
            ]
        }
    }

    ConnectExtensions = [
        { vendorID = 311, attr = 30, attrName = "PrimaryNBNSserver",  attrType =
        "IPv4", profileBlockTag = 6950030 }
    ]

}
```

Parameters

The parameters available in this section are:

destinationRoutingAddress

Syntax:

```
destinationRoutingAddress  = {
                tag = int
                  NumberRules = []
            }
```

Description: Maps the destinationRoutingAddress values from the INAP Connect message to the RCA tag in attribute map.

- tag specifies the RCA tag to use for mapping.
- NumberRules define normalisation rules of the destinationRoutingAddress.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable.

Example:

```
destinationRoutingAddress  = {
    tag = 327689
    NumberRules = [
                {fromNoa=4,remove=0, prepend="00" }
                { fromNoa=3,remove=0, prepend="0064" }
                ]
            }
```

originalCalledPartyID

Syntax:

```
originalCalledPartyID  = {
                tag = int
                  NumberRules = []
            }
```

Description: Maps the originalCalledPartyID values from the INAP Connect message to the RCA tag in attribute map.

- `tag` specifies the RCA tag to use for mapping.
- NumberRules define normalisation rules of the destinationRoutingAddress.

Type: Array

Optionality: Optional (default used if not set).

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable.

Example:

```
originalCalledPartyID = {
    tag = 355689
    NumberRules = [
                {fromNoa=4,remove=0,
    prepend="00" }
                { fromNoa=3,remove=0,
    prepend="0064" }
                ]
            }
```

callingPartysCategoryTag

Syntax:

```
callingPartysCategoryTag  = {
                tag = int
                  NumberRules = []

            }
```

Description: Maps the callingPartysCategoryTag values from the INAP Connect message to the RCA tag in attribute map.

- tag specifies the RCA tag to use for mapping.
- NumberRules define normalisation rules of the destinationRoutingAddress.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable.

Example:

```
callingPartysCategoryTag  = {
    tag = 327889
    NumberRules = [
                {fromNoa=4,remove=0, prepend="00" }
                { fromNoa=3,remove=0, prepend="0064" }
                ]
            }
```

redirectingPartyId

Syntax:

```
redirectingPartyId  = {
                tag = int
                  NumberRules = []

            }
```

Description: Maps the value for the redirecting party id from the INAP Connect message to the RCA tag in attribute map.

- tag specifies the RCA tag to use for mapping.
- NumberRules define normalisation rules of the destinationRoutingAddress.

Type: Array

Optionality: Optional (default used if not set).

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable.

Example:

```
redirectingPartyId  = {
    tag = 327703
    NumberRules = [
                {fromNoa=4,remove=0, prepend="00" }
                { fromNoa=3,remove=0, prepend="0064" }
                ]
            }
```

genericNumbersAdditionalCallingPartyNumber

Syntax:

```
genericNumbersAdditionalCallingPartyNumber  = {
                tag = int
                  NumberRules = []
        }
```

Description: Maps the genericNumbersAdditionalCallingPartyNumber values from the INAP Connect message to the RCA tag in attribute map.

- tag specifies the RCA tag to use for mapping.
- NumberRules define normalisation rules of the destinationRoutingAddress.

Type: Array

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: If NumberRules are not defined, the denormalisation rules defined globally in *NumberRules* (on page 28) are applicable.

Example:

```
genericNumbersAdditionalCallingPartyNumber = {
    tag = 366689
    NumberRules = [
                {fromNoa=4,remove=0, prepend="00" }
                { fromNoa=3,remove=0, prepend="0064" }
                ]
            }
```

ConnectExtensions

Syntax: See example

Description: Specifies the values to copy from the extension 701 profile block in the Connect message (sent by ACS to RCA) into the attribute map.

The extension type 701 is used to encode profile blocs in the InitialDP and Connect operations.

Type: Numeric String

Optionality: Optional (default used if not set)

Allowed:

Default:

Notes: For more information about the ConnectExtensions parameters, see *IncomingMapping section* (on page 49).

Example:

```
ConnectExtensions = [
     { vendorID = 007, attr = 30, attrName =
    "PrimaryNBNSserver",  attrType = "IPv4", profileBlockTag =
    6950030 }
    ]
```

profileBlockTag

Syntax: `profileBlockTag = int`
Description: The tag number of the connect extensions profile block.
Type: Integer
Optionality: Optional
Allowed:
Default:
Notes:
Example: `profileBlockTag = 6950030`

OutgoingMappingtoSwitch

Introduction

This section specifies which tags to copy from the attribute map into the attributes in the Radius messages sent by RCA. OutgoingMapping is a part of the ParameterisedRCAFields section.

The AccessAcceptMapping section describes the attribute map for handling Access-Accept messages, while the configuration in DisconnectRequestMapping section is used for Disconnect-Request messages.

Parameters

The following parameters are common to the IncomingMapping section:

- *attr* (on page 50)
- *attrName* (on page 50)
- *attrType* (on page 50)
- *vendorID* (on page 51)

The rest of the parameters are described below:

AccessAcceptMapping

Syntax: See example
Description: Defines the attribute mapping for Access-Accept messages.
Type: Array
Optionality: Optional
Allowed:
Default:
Notes: For more information about the attr, attrName, and attrType parameters, see *IncomingMapping section* (on page 49)
Example:

```
AccessAcceptMapping = [
    { attr = 6, attrName = "Service-Type", attrType =
    "integer", fromTag = 6690006}
    { attr = 7, attrName = "Framed-Protocol", attrType =
    "integer", fromTag = 6690007}
]
```

DisconnectRequestMapping

Syntax: See example
Description: Defines the attribute mapping for Disconnect-Request messages.
Type: Array

Optionality: Optional
Allowed:
Default:
Notes: For more information about the vendorID, attr, attrName, and attrType parameters, see *IncomingMapping section* (on page 49)
Example:

```
DisconnectRequestMapping = [
    { vendorID = 10415, attr = 1, attrName = "3GPP-IMSI",
    attrType = "string", fromTag = 6760001}
    { vendorID = 10415, attr = 10, , attrName = "£GPP-NSAPI",
    attrType = "string", fromTag = 6760010 }
]
```

fromTag

Syntax: `fromTag = tag`
Description: Decimal value of the RCA tag from which the attribute value is copied.
Type: Numeric string
Optionality: Optional (default used if not set)
Allowed:
Default:
Notes:
Example: `fromTag = 6660003`

Tracing parameters

Introduction

The parameters in this section are used for tracing activities. They are all found within the `tracing = { }` statement.

Tracing parameters

The available parameters are:

callsPerMinute

Syntax: `callsPerMinute = value`
Description: Maximum number of initiated calls per minute for which tracing will be activated.
Type: Integer
Optionality: Optional
Allowed: Any integer
Default: 2
Notes:
Example: `callsPerMinute = 2`

destAddress

Syntax: `destAddress = [addr1,addr2,addrn]`
Description: List of destination addresses that are to be traced.
Type: Numeric string

Optionality: Optional

Allowed:
- Any valid addresses
- ""

Default: ""

Notes: "" = trace all known destination addresses.

Example:
```
destAddress = [
""
]
```

enabled

Syntax: `enabled = true|false`

Description: Switches tracing on or off.

Type: Boolean

Optionality: Optional

Allowed:
- true
- false

Default: false

Notes: If false, the rest of the parameters can be ignored.

Example: `enabled = false`

maxFileSizeKB

Syntax: `maxFileSizeKB = value`

Description: Maximum file size in KB for the trace file(s).

Type: Integer

Optionality: Optional

Allowed: Any integer

Default: 0

Notes: 0 = unlimited file size.

If `maxFileSizeKB` is > 0 and if `maxNumFiles` is > 0 then every `outputFileCycle` trace will check to see if the `outputFile` size is > `maxFileSizeKB`.

If so, the file is renamed to `outputFile.`*`N`* where *`N`* is from 1 to `maxNumFiles`.

If all *`N`* files exist, then the oldest file is overwritten.

Example: `maxFileSizeKB = 0`

maxNumFiles

Syntax: `maxNumFile = value`

Description: Maximum number of trace files to use

Type: Integer

Optionality: Optional

Allowed: Any integer

Default: 4

Notes: If `maxFileSizeKB` is > 0 and if `maxNumFiles` is > 0 then every `outputFileCycle` trace will check to see if the `outputFile` size is > `maxFileSizeKB`.

If so, the file is renamed to `outputFile.`*`N`* where *`N`* is from 1 to `maxNumFiles`.

If all *N* files exist, then the oldest file is overwritten.

Example: `maxNumFiles = 4`

origAddress

Syntax: `origAddress = [addr1,addr2,addrn]`

Description: List of originating addresses that are to be traced.

Type: Numeric string

Optionality: Optional

Allowed:
- Any valid addresses
- ""

Default: ""

Notes: "" = trace all known originating addresses.

Example:
```
origAddress = [
""
]
```

outputFile

Syntax: outputFile `= "path_and_filename"`

Description: Primary trace output file

Type: String

Optionality: Mandatory

Allowed: Any existing file path and name

Default: no default

Notes: See also notes for `maxNumFiles`.

Example: `outputFile = "/IN/service_packages/RCA/tmp/rcaTrace.log"`

outputFileCycle

Syntax: outputFileCycle `= value`

Description: Close and re-open the trace file every *N* calls

Type: Integer

Optionality: Optional

Allowed: Any integer

Default: 512

Notes: The `maxFleSizeKB` exceeded is checked for at this time also, and a new trace file started if required.

Example: `outputFileCycle = 512`

useTONNPI

Syntax: `useTONNPI = true|false`

Description: Indicates whether to include TON and NPI in the address for trace prefix matching or not.

Type: Boolean

Optionality: Optional

Allowed:
- true
- false

Default: true
Notes:
Example: `useTONNPI = true`

Failure

If the radiusControlAgent fails, no RCA messages will be processed.

Log output

The radiusControlAgent process writes output to:

/IN/service_packages/RCA/tmp/rca.log

Notes:

- As with other Convergent Charging Controller software, the contents of this log depend on the value of the DEBUG environment variable.
  To find out the location of this file, check **/IN/service_packages/RCA/bin/rca.sh** and see where it redirects the output from radiusControlAgent.
- If rca.sh calls radiusControlAgent with the `-d` option, RCA will log a dump of its config on startup or config re-read.

Trace output

The radiusControlAgent process writes any trace output to:

/IN/service_packages/RCA/tmp/rcaTrace.log

Notes:

- This is specified in **eserv.config** RADIUSControlAgent.tracing.outputFile.
- If the cmnTraceOutput debug section is enabled (in the DEBUG environment variable) the trace output will also be present in the main RCA log file, which is more convenient as any alarms or debug output will appear along with the trace output.

Statistics

Introduction

Statistics from the RCA are automatically collected from startup. There is no need or requirement for any configuration.

Viewing statistics

All the RCA statistics collected can be viewed through the SMS reporting mechanism. See *SMS User's Guide*.

Statistics collected

This table describes the statistics collected when the RCA is active.

Statistic	Description
ACCESS_ACCEPTS_SENT	All Access-Accept messages sent.
ACCESS_REJECTS_SENT	All Access-Reject messages sent.

Statistic	Description
ACCESS_REQUESTS_RECEIVED	All Access-Request messages received.
ACCOUNTING_REQUESTS_RECEIVED	All Accounting-Request messages received.
ACCOUNTING_RESPONSES_SENT	All Accounting-Response messages sent.
DISCONNECT_ACKS_RECEIVED	
DISCONNECT_NAKS_RECEIVED	
DISCONNECT_REQUESTS_SENT	All Disconnect-Request messages sent.
SESSIONS_ALLOWED	All sessions allowed by Prepaid Charging.
SESSIONS_DENIED	All sessions denied by Prepaid Charging. **Note:** For example, when the subscriber does not have sufficient credit, or is not allowed to use the WiFi service.
SESSIONS_DENIED_IN_ERR	All sessions denied due to errors of communication between RCA and Prepaid Charging.
SESSIONS_DENIED_RADIUS_ERR	All sessions denied due to errors in the initial RADIUS Access-Request message. **Note:** For example, if the Access-Request message is missing a parameter required by RCA.
SESSIONS_ENDED_DISCONNECTED	All sessions ended due to the subscriber's funds running out.
SESSIONS_ENDED_IN_ERR	All sessions ended due to errors of communication between RCA and Prepaid Charging. **Note:** This count does not include RADIUS sessions *denied* due to errors of communication between RCAand Prepaid Charging – those are counted in SESSIONS_DENIED_IN_ERR.
SESSIONS_ENDED_RADIUS_ERR	All sessions ended due to RADIUS protocol errors.
SESSIONS_ENDED_TIMEOUT	All sessions ended due to timeouts enforced by RCA.
SESSIONS_ENDED_USER_HUNGUP	All sessions ended due to the subscriber hanging up.
SESSIONS_INITIATED	All sessions initiated by the Orbyte. **Note:** This is different to the number of RADIUS Access-Request messages received, as a single session may consist of many Access-Requests (one for the initial quota reservation, followed by on-line quota updates).
SESSIONS_TOTAL_DURATION	Sum total duration of all RADIUS sessions ended due to the subscriber's funds running out or the subscriber hanging up. **Note:** This is a session duration count of the

Statistic	Description
	“normal cases”. The average session duration can be calculated by dividing this statistic by the sum of statistics SESSIONS_ENDED_DISCONNECTED and SESSIONS_ENDED_USER_HUNGUP.

Chapter 4

About Installation and Removal

Overview

Introduction

This chapter provides information about the installed components for the Convergent Charging Controller application described in this guide. It also lists the files installed by the application that you can check for, to ensure that the application installed successfully.

In this Chapter

This chapter contains the following topics.

Installation and Removal Overview

Introduction

For information about the following requirements and tasks, see *Installation Guide*:

- Convergent Charging Controller system requirements
- Pre-installation tasks
- Installing and removing Convergent Charging Controller packages

Radius Control Agent packages

An installation of Radius Control Agent includes the following packages, on the:

- SMS:
  - rcaSms
- SLC:
  - rcaScp

Removing RCA from the crontab

When removing rcaScp, follow these steps to remove the rca_get_read_count.sh entry from the crontab on the SLC.

Step	Action
1	Log in as acs_oper.
2	At the command line prompt, type `crontab -e`
	Note: To edit the crontab, use the vi editor.
3	Delete the entry for rca_get_read_count.sh

Step	Action
4	Save the file and restart the SLEE.

RCA Table Replication

Introduction

Tables and statistics must be replicated to the SLC to complete the installation of the rcaScp package.

The following RCA tables must be replicated to the SLC in the given order:

- RCA_IP_ADDRESS
- RCA_ADDRESS_LIST_ENTRY
- RCA_BILL_TYPE_PREFIX
- RCA_CLI_ADDR_LIST
- RCA_CONFIG_READ_SCHEDULER
- RCA_RELEASE_MESSAGE
- RCA_GLOBAL_CONFIG
- RCA_SVC_OPT_LIST-ENTRY

The following RCA statistics table must be replicated to the SLC:

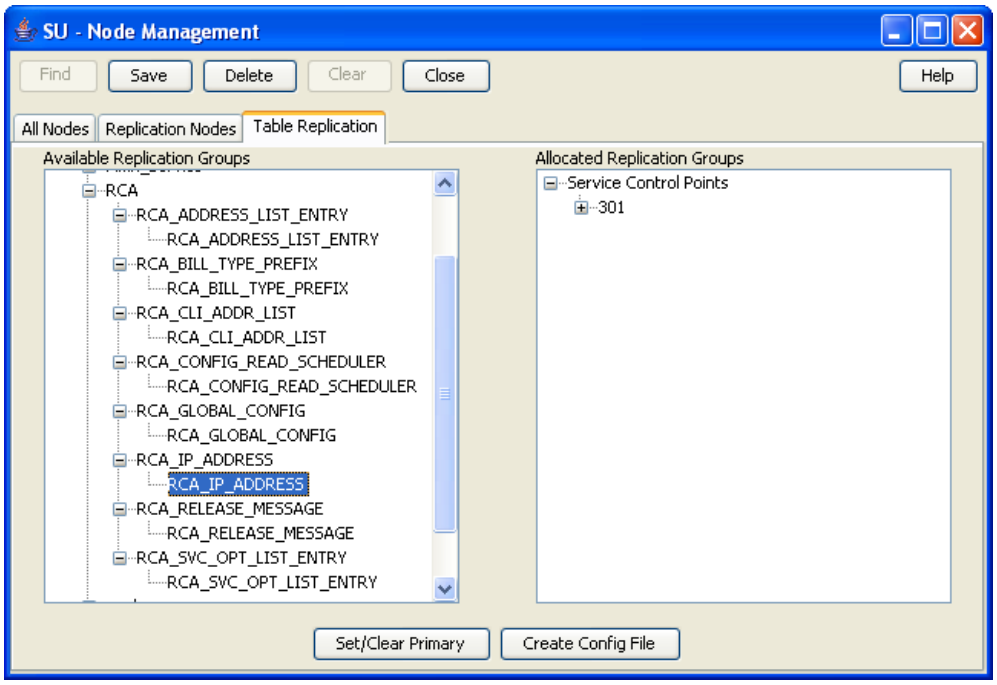
- SMF_STDEF_RCA

Replicating the RCA tables

Follow these steps to replicate the RCA tables to the SLC.

Step	Action
1	From the main SMS screen, select the **Operator Functions > Node Management** menu options.
2	Select the **Table Replication** tab. **Result:** You see the tab, showing the available and allocated replication groups.

Step	Action



3 Expand the RCA group in the **Available Replication Groups** list.

4 Click on the table to be replicated, and drag it to the appropriate SLC node under Service Control Points in the **Allocated Replication Groups** list.

Note: To check which nodes are SLC nodes, you can use Find and Search on the **Replication Nodes** tab.

5 Repeat step 4 for all the tables to be replicated.

6 Click **Save**.

7 Click **Create Config File**.

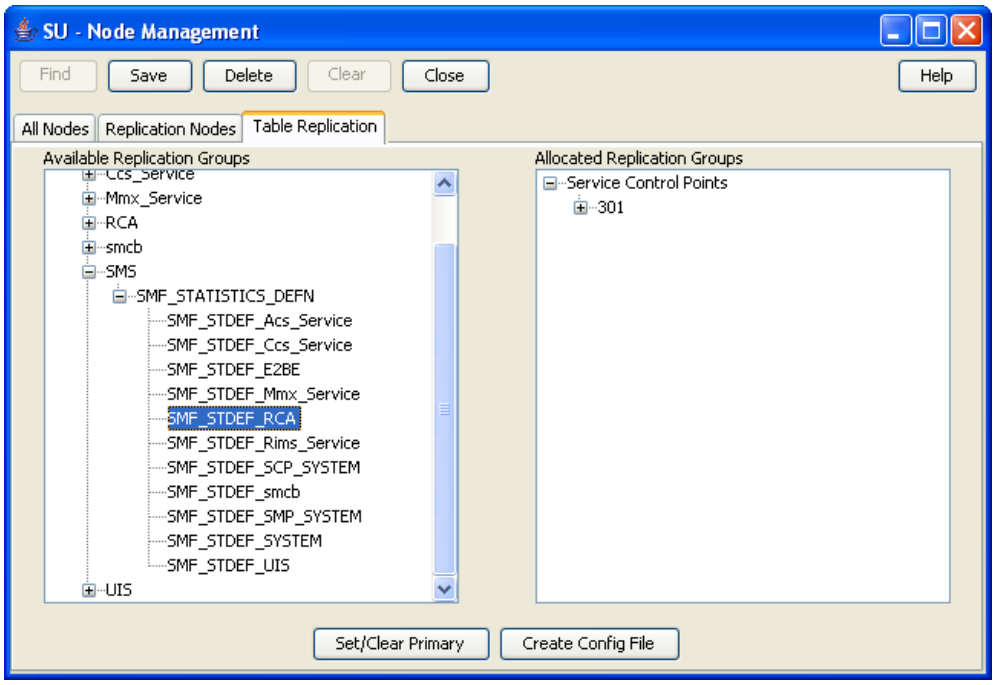
8 To ensure the data has been replicated, check the database on the SLC.

Replicating the RCA statistics

Follow these steps to replicate the RCA statistics to the SLC.

Step	Action
1	From the main SMS screen, select the **Operator Functions > Node Management** menu options.
2	Select the **Table Replication** tab. **Result:** You see the tab, showing the available and allocated replication groups.

Step	Action



3 Expand the SMS group in the **Available Replication Groups** list.

4 Expand the SMF_STATISTICS_DEFN table.

5 Click on the SMF_STDEF_RCA replication group, and drag it to the appropriate SLC node under Service Control Points in the **Allocated Replication Groups** list.

Note: Which nodes are SLC nodes can be checked by using Find and Search on the **Replication Nodes** tab.

6 Click **Save**.

7 Click **Create Config File**.

8 Check the database on the SLC to ensure the data has been replicated.

Checking the Installation

Introduction

Refer to these check lists to ensure the Radius Control Agent has been installed correctly.

RCA database tables - SMS

The following tables should exist in the SMF database on the SMS:

- RCA_ADDRESS_LIST
- RCA_ADDRESS_LIST_ENTRY
- RCA_BILL_TYPE_PREFIX
- RCA_CLI_ADDR_LIST
- RCA_CONFIG_READ_SCHEDULER
- RCA_GLOBAL_CONFIG
- RCA_IP_ADDRESS

- RCA_RELEASE_MESSAGE
- RCA_SERVICE_OPTION
- RCA_SVC_OPTION_LIST
- RCA_SVC_OPT_LIST_ENTRY
- SMS_STDEF_RCA

RCA database tables - SLC

The following tables should exist on the SCP database on the SLC.

- RCA_ADDRESS_LIST_ENTRY
- RCA_BILL_TYPE_PREFIX
- RCA_CLI_ADDR_LIST
- RCA_CONFIG_READ_SCHEDULER
- RCA_GLOBAL_CONFIG
- RCA_IP_ADDRESS
- RCA_RELEASE_MESSAGE
- RCA_SVC_OPT_LIST_ENTRY
- SMS_STDEF_RCA

RCA directories and files

The RCA installation creates the following directories:

- **/IN/service_packages/RCA/bin**
- **/IN/service_packages/RCA/cdr**
- **/IN/service_packages/RCA/core**
- **/IN/service_packages/RCA/db**
- **/IN/service_packages/RCA/etc**
- **/IN/service_packages/RCA/lib**
- **/IN/service_packages/RCA/tmp**

The RCA installation installs the following binaries and interfaces:

- **/IN/services_packages/RCA/bin/addLine.sh**
- **/IN/services_packages/RCA/bin/addLineRevese.sh**
- **/IN/services_packages/RCA/bin/radiusControlAgent**
- **/IN/services_packages/RCA/bin/rca.sh**
- **/IN/services_packages/RCA/bin/rca_get_read_count.sh**

The RCA installation installs the following example configuration file:

- **/IN/services_packages/RCA/eserv.config**

The RCA installation installs the following shared library:

- **/IN/services_packages/RCA/lib/librcaalarm.so**

Glossary of Terms

AAA

Authentication, Authorization, and Accounting. Specified in Diameter RFC 3588.

ACS

Advanced Control Services configuration platform.

ANI

Automatic Number Identification - Term used in the USA by long-distance carriers for CLI.

ASN.1

Abstract Syntax Notation One - a formal notation used for describing data transmitted by telecommunications protocols. ASN.1 is a joint ISO/IEC and ITU-T standard.

AVP

Attribute Value Pair, used in Diameter to represent properties of a particular request or answer.

BCSM

Basic Call State Model - describes the basic processing steps that must be performed by a switch in order to establish and tear down a call.

CAMEL

Customized Applications for Mobile network Enhanced Logic

This is a 3GPP (Third Generation Partnership Project) initiative to extend traditional IN services found in fixed networks into mobile networks. The architecture is similar to that of traditional IN, in that the control functions and switching functions are remote. Unlike the fixed IN environment, in mobile networks the subscriber may roam into another PLMN (Public Land Mobile Network), consequently the controlling function must interact with a switching function in a foreign network. CAMEL specifies the agreed information flows that may be passed between these networks.

CCS

1) Charging Control Services component.

2) Common Channel Signalling. A signalling system used in telephone networks that separates signalling information from user data.

CDMA

Code Division Multiple Access is a method for describing physical radio channels. Data intended for a specific channel is modulated with that channel's code. These are typically pseudo-random in nature, and possess favourable correlation properties to ensure physical channels are not confused with one another.

CLI

Calling Line Identification - the telephone number of the caller. Also referred to as ANI.

Convergent

Also “convergent billing”. Describes the scenario where post-paid and pre-paid calls are handed by the same service platform and the same billing system. Under strict converged billing, post-paid subscribers are essentially treated as “limited credit pre-paid”.

cron

Unix utility for scheduling tasks.

crontab

File used by cron.

Diameter

A feature rich AAA protocol. Utilises SCTP and TCP transports.

DP

Detection Point

DTMF

Dual Tone Multi-Frequency - system used by touch tone telephones where one high and one low frequency, or tone, is assigned to each touch tone button on the phone.

FDA

First Delivery Attempt - the delivery of a short message directly to the SME rather than relaying it through the MC.

GPRS

General Packet Radio Service - employed to connect mobile cellular users to PDN (Public Data Network- for example the Internet).

GSM

Global System for Mobile communication.

It is a second generation cellular telecommunication system. Unlike first generation systems, GSM is digital and thus introduced greater enhancements such as security, capacity, quality and the ability to support integrated services.

HLR

The Home Location Register is a database within the HPLMN (Home Public Land Mobile Network). It provides routing information for MT calls and SMS. It is also responsible for the maintenance of user subscription information. This is distributed to the relevant VLR, or SGSN (Serving GPRS Support Node) through the attach process and mobility management procedures such as Location Area and Routing Area updates.

HPLMN

Home PLMN

IDP

INAP message: Initial DP (Initial Detection Point)

IMSI

International Mobile Subscriber Identifier. A unique identifier allocated to each mobile subscriber in a GSM and UMTS network. It consists of a MCC (Mobile Country Code), a MNC (Mobile Network Code) and a MSIN (Mobile Station Identification Number).

The IMSI is returned by the HLR query (SRI-SM) when doing FDA. This tells the MSC exactly who the subscriber is that the message is to be sent to.

IN

Intelligent Network

INAP

Intelligent Network Application Part - a protocol offering real time communication between IN elements.

Initial DP

Initial Detection Point - INAP Operation. This is the operation that is sent when the switch reaches a trigger detection point.

IP

1) Internet Protocol

2) Intelligent Peripheral - This is a node in an Intelligent Network containing a Specialized Resource Function (SRF).

IP address

Internet Protocol Address - network address of a card on a computer.

ITU

International Telecommunication Union

MC

Message Centre. Also known as SMSC.

MCC

Mobile Country Code. In the location information context, this is padded to three digits with leading zeros. Refer to ITU E.212 ("Land Mobile Numbering Plan") documentation for a list of codes.

MNC

Mobile Network Code. The part of an international address following the mobile country code (MCC), or at the start of a national format address. This specifies the mobile network code, that is, the operator owning the address. In the location information context, this is padded to two digits with a leading zero. Refer to ITU E.212 ("Land Mobile Numbering Plan") documentation for a list of codes.

MS

Mobile Station

MSC

Mobile Switching Centre. Also known as a switch.

MSIN

Mobile Station Identification Number.

MT

Mobile Terminated

NOA

Nature Of Address - a classification to determine in what realm (Local, National or International) a given phone number resides, for the purposes of routing and billing.

NPI

Number Plan Indicator

PLMN

Public Land Mobile Network

RADIUS

Remote Authentication Dial-In User Service - a system of distributed security that secures remote access to networks and network services against unauthorised access.

SCP

Service Control Point. Also known as SLC.

SCTP

Stream Control Transmission Protocol. A transport-layer protocol analogous to the TCP or User Datagram Protocol (UDP). SCTP provides some similar services as TCP (reliable, in-sequence transport of messages with congestion control) but adds high availability.

SGSN

Serving GPRS Support Node

SLC

Service Logic Controller (formerly UAS).

SLEE

Service Logic Execution Environment

SME

Short Message Entity - This is an entity which may send or receive short messages. It may be located in a fixed network, a mobile, or an SMSC.

SMS

Depending on context, can be:

- Service Management System hardware platform
- Short Message Service
- Service Management System platform
- Convergent Charging Controller Service Management System application

SN

Service Number

SRF

Specialized Resource Function – This is a node on an IN which can connect to both the SSP and the SLC and delivers additional special resources into the call, mostly related to voice data, for example play voice announcements or collect DTMF tones from the user. Can be present on an SSP or an Intelligent Peripheral (IP).

SRI

Send Routing Information - This process is used on a GSM network to interrogate the HLR for subscriber routing information.

SSP

Service Switching Point

TCP

Transmission Control Protocol. This is a reliable octet streaming protocol used by the majority of applications on the Internet. It provides a connection-oriented, full-duplex, point to point service between hosts.

URL

Uniform Resource Locator. A standard way of specifying the location of an object, typically a web page, on the Internet.

VLR

Visitor Location Register - contains all subscriber data required for call handling and mobility management for mobile subscribers currently located in the area controlled by the VLR.

VWS

Oracle Voucher and Wallet Server (formerly UBE).

Index

S

T

U

V

W