

# Oracle® Communications

## Cloud Native Policy and Charging Rules Function User's Guide



Release 1.6.0

F31348-01

May 2020

The Oracle logo, consisting of a solid red square with the word "ORACLE" in white, uppercase, sans-serif font centered within it.

ORACLE®

Copyright © 2019, 2020, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

# Contents

<b>1</b>	<b>Introduction</b>	
	Overview	1-1
	Acronyms	1-1
	References	1-2
<b>2</b>	<b>About Cloud Native Policy and Charging Rule Function Services</b>	
	How the Services Talk to Each Other	2-1
<b>3</b>	<b>Cloud Native Policy and Charging Rules Function Architecture</b>	
	About Policy Design Experience	3-1
<b>4</b>	<b>Configuring Cloud Native Policy and Charging Rules Function Using Cloud Native Core Console</b>	
	Configuring Services and Manageable Objects	4-1
	Managing a Charging Server	4-2
	Managing Custom AVPs	4-3
	About Custom AVPs	4-3
	Creating a Custom AVP	4-4
	Modifying an AVP	4-8
	Deleting an AVP	4-8
	Managing Custom Vendors	4-9
	About Custom Vendors	4-9
	Creating a Custom Vendor	4-9
	Modifying a Custom Vendor	4-9
	Deleting a Custom Vendor	4-10
	Creating a Media Profile	4-10
	Session Viewer	4-12
	Configuring Core Service	4-12
	Managing Policy	4-12
	Settings	4-13

Creating a Policy Project	4-13
Managing State Variables	4-14
Data Model	4-15
Configuring Policy Common Configurations	4-17
Connecting to LDAP Data Source	4-17
Managing Match Lists	4-20
Importing Configurable Objects	4-22
Exporting Configurable Objects	4-22

## 5 Writing Policy Conditions

---

Policy Condition Categories	5-1
Mobility Conditions	5-1
where the mobile session supports sponsored connectivity	5-1
where the Cell Identifier matches one of specified CI value(s)	5-2
where the IP address of the Serving Gateway matches one of specified value(s) (es)	5-3
where the IP-CAN type specified	5-3
where network initiated requests are supported	5-4
where the E-UTRAN Cell Identifier matches one of specified ECI value(s)	5-4
where the subscribed PRA area matches one of PRA area(s)	5-5
where the UE is inside/outside/inactive for any one of PRA Area	5-6
where the APN matches one of specified APN value(s)	5-7
Network Device Conditions	5-8
where the device type is specified type	5-8
where the User Equipment IMEISV matches one of specified IMEISV value(s)	5-9
Policy SDP Properties Conditions	5-9
where the local codec data is an offer	5-9
where the local specified SDP property exists	5-10
where the local specified SDP property is numerically equal to value	5-12
where the local specified SDP property matches one of value(s)	5-14
Request Conditions	5-16
where the requested QCI is specified QCI	5-16
where the request supports feature name	5-17
where the session is an enforcement session	5-18
where the request AVP name exists	5-18
where the request AVP namevalue matches one of value(s)	5-19
where the request AVP name value is numerically equal to value	5-20
where the request AVP name value is contained in Match Lists select lists	5-21
where the request AVP name value contains one of value(s)	5-22
where the request AVP Media-Component-Description exists	5-23
where at least one flow has media type that matches specified type(s)	5-23

where the AF-Application-ID matches one of specified value(s)	5-24
where the corresponding enforcement session supports feature name	5-24
where the flow media type is one of specified type(s)	5-25
where the flow usage is one of specified usage(s)	5-26
where the request is creating a new flow	5-26
where the Service-URN is one of specified value(s)	5-27
where the specific action is one of specified action(s)	5-27
where the rule report contains one of specified rule name(s) and the rule status is active	5-28
where the request MPS Identifier matches one of value(s)	5-28
where the requested media component description reservation priority is one of specified	5-29
where the requested session reservation priority is one of specified	5-30
where the flow media type matches one of user defined media type(s)	5-31
where the Sponsor-Identity matches one of specified Sponsor Identity(s)	5-31
where the AF-Application-ID is available	5-32
where the requested APN aggregate maximum bitrate upstream is greater than # bps	5-32
Time of Day Conditions	5-33
where the current time is between start time and end time using configured local time	5-33
where the current time is within the specified time period(s)	5-34
where today is a week day using configured local time	5-35
where today is a weekend day using configured local time	5-35
where today is day using configured local time	5-36
where today is the specified number(s) th day(s) of Any Month in natural order using configured local time	5-37

## 6 Actions for Writing Policy Rules

---

add custom grouped AVP name and send always	6-1
set value to Existing or New custom AVP name and send always	6-2
set custom AVP name value to the policy context property name	6-2
remove custom AVP name from reply always	6-3
mark request AVP name as failed if exists and send always	6-3
set the user property name to Existing or New custom AVP name and send always	6-4

## 7 Cloud Native Policy and Charging Rule Function Alerts

---

PCRF Alert Configuration	7-3
--------------------------	-----

---

# What's New in This Guide

This section introduces the new features for Release 1.6.0 in Oracle Communications Cloud Native Policy and Charging Rules Function (CNPCRF) User's Guide.

## **New Features for Release 1.6.0**

For CNPCRF Release 1.6.0, this guide has been updated to include the following new development features:

- Added [Session Viewer](#) support
- Added [Custom AVP](#) support
- Added [Custom Vendor](#) support
- Added [State Variables](#) support
- Updated the [Connecting to LDAP Data Source](#) feature documentation to align with the new GUI
- Added the supported conditions and actions for Custom AVP in the Policy [Conditions](#) and [Actions](#) chapter

# My Oracle Support

My Oracle Support (<https://support.oracle.com>) is your initial point of contact for all product support and training needs. A representative at Customer Access Support can assist you with My Oracle Support registration.

Call the Customer Access Support main number at 1-800-223-1711 (toll-free in the US), or call the Oracle Support hotline for your local country from the list at <http://www.oracle.com/us/support/contact/index.html>. When calling, make the selections in the sequence shown below on the Support telephone menu:

- For Technical issues such as creating a new Service Request (SR), select **1**.
- For Non-technical issues such as registration or assistance with My Oracle Support, select **2**.
- For Hardware, Networking and Solaris Operating System Support, select **3**.

You are connected to a live agent who can assist you with My Oracle Support registration and opening a support ticket.

My Oracle Support is available 24 hours a day, 7 days a week, 365 days a year.

# 1

## Introduction

This document provides information on how to use the Cloud Native Policy and Charging Rule Function (CNPCRF) and configure the services.

## Overview

The Oracle Communications Cloud Native Core Policy and Charging Rules Function (cnPCRF) is a micro-services based solution for managing policy and charging functionality in a 3G and 4G network. Oracle cnPCRF's offers operators a highly scalable PCRF that is cloud ready. It helps service providers to create and manage complex network policies in a telecom network.

Oracle Communications cnPCRF has the Oracle's PCRF functionalities with product architecture designed for the cloud. The cnPCRF provides a new policy designer/ configuration & troubleshooting GUI, besides a set of new functionalities and architectural changes, the prominent features are listed below:

- 3-Tier Architecture with N+M Geo-Redundancy model
- Compliant with 3GPP Release 15
- Leverages a common Oracle Communications Cloud Native Environment (CNE)
- Packaged to support both VM-based and container-based cloud infrastructure
- Policy solution handling 4G Policy and Charging Control (PCC) use cases with support to legacy diameter based interfaces
- Supports CI/CD
- Integrated with Kubernetes and 5G/CNE common services
- Integrated with DevOps workflows
- Supports all legacy diameter interfaces

## Acronyms

The following table provides information about the acronyms used in the document.

**Table 1-1 Acronyms**

Acronym	Definition
CNPCRF	Cloud Native Policy Charging and Rules Function
DNAI	DN Access Identifier
FQDN	Fully Qualified Domain Name
MDBV	Maximum Data Burst Volume
MFBR	Maximum Flow Bit Rate
NAI	Network Access Identifier



**Table 1-1 (Cont.) Acronyms**

<b>Acronym</b>	<b>Definition</b>
PSA	PDU Session Anchor
PDS	Policy Data Source
QFI	QoS Flow Identifier
QoE	Quality of Experience
RQA	Reflective QoS Attribute
RQI	Reflective QoS Indication
S-NSSAI	Single Network Slice Selection Assistance Information
SUPI	Subscription Permanent Identifier

## References

Refer to the following documents for more information.

- Oracle Communications Cloud Native Policy and Charging Rule Function (CNPCRF) Installation and Upgrade Guide.

# 2

## About Cloud Native Policy and Charging Rule Function Services

This section provides information about the CNPCRF services which includes:

- PCRF Core Service

### PCRF Core Service

PCRF core service includes all the features of 4G PCRF except those provided by MRA, including:

- Protocol implementation including the support of various call flows, validity check, etc.
- Session correlation
- Retrieval and storage of user information
- Invoker of policy service and process of policy actions

## How the Services Talk to Each Other

In general, most services under Cloud Native Policy and Charging Rules Function (CNPCRF) would use ClusterIP as deployment type under Kubernetes cluster environment. However, the following two services need LoadBalancer deployment type which require external access:

- diam-gateway service
- cm service

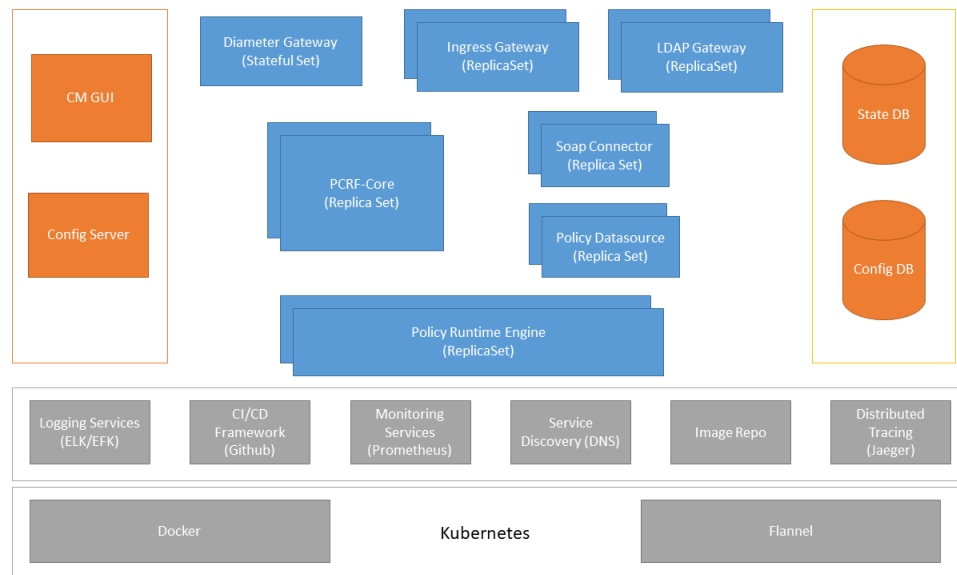
Given above deployment structure, a public IP would allocate to the above two services which accept external request, the inner communication rely on cluster IP to find each other.

# 3

## Cloud Native Policy and Charging Rules Function Architecture

The Oracle Communications Cloud Native Policy and Charging Rule Function (CNPCRF) is built as a cloud-native application composed of a collection of microservices running in a cloud-native environment. It separates processing/business logic and state concerns following the corresponding logical grouping of microservices/components:

**Figure 3-1 PCRF Architecture**



### About Policy Design Experience

Policy design experience allows you to craft and deploy, from scratch, operator policies in production in very less time. 5G brings the policy design experience to the next level by providing flexibility, extensibility, modularization, and assurance to the operator to rapidly, yet confidently deploy new operator policies and enable use cases more faster.

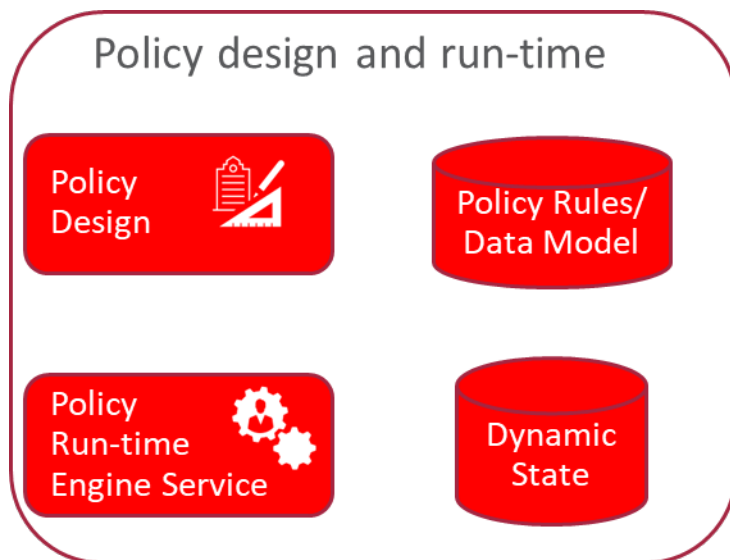
The Cloud Native Policy and Charging Rules Function (CNPCRF) packages its microservices into containers and leverages Kubernetes' constructs and abstractions such as Pods, ReplicaSets, and services so it can enable Kubernetes to manage and orchestrate the CNPCRF. It also leverages Istio as a service mesh (including Envoy proxies as sidecars) for the internal communication amongst the various microservices. The Oracle Communications CNPCRF integrates with a variety of common services for data collection, analysis, and visualization services for operational aspects like logs, metrics, and traces. The Oracle Communications CNPCRF comprises

artifacts like Helm charts that encapsulate lifecycle instructions and resource dependencies for all member components.

The Oracle Communications CNPCRF is flexible to run in various cloud-native environments. The CNPCRF can be configured to leverage common services provided by the cloud-native environment and/or provide its own set if certain common services aren't provided by the underlying environment.

The following figure highlights the various components used by the policy design and run-time:

**Figure 3-2 Policy Design Experience**



### Design

- Modular and flexible domain driven policy design
- Modules encompasses data model, triggers, conditions and actions
- Modules can be designed via a GUI (very intuitive, can be used by anyone) and allows any language supported by JVM for advanced cases if needed (e.g. Java, Groovy, etc)
- Pre-packaged modules provided by Oracle
- Modules can be extended or built by operators

### Run-time

- Run-time engine service to expose APIs
- Run-time engine service to be stateless and independently scalable
- Newly designed policies or policy updates can be rolled out in an incremental fashion (e.g. to a specific set of policy run-time engines) to enable canary releases and ensure updates are working as expected before being rolled out globally

### **Debugging and testing**

- Debugging policy logic capability as a complementary tool to the design experience
- Automated testing framework to enable regression and validation of policy logic and modules before deployment

# 4

## Configuring Cloud Native Policy and Charging Rules Function Using Cloud Native Core Console

This chapter describes how to configure different services in Oracle Communications Cloud Native Policy and Charging Rules Function (CNPCRF) and how to create policies and manageable objects in CNPCRF using Oracle Communications Cloud Native Core Console.

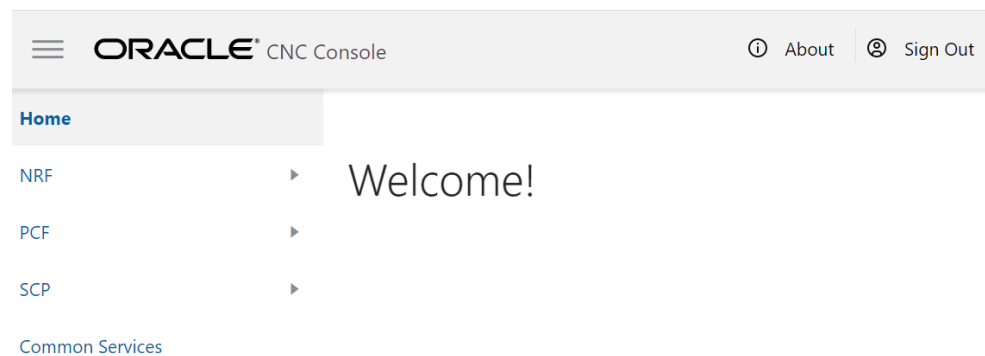
### Cloud Native Core Console Interface

This section provides an overview of the Oracle Communications Cloud Native Core (CNC) Console, which includes a interface to aid in creating policies and manageable objects in CNPCRF.

To Log in:

1. Open a web browser and enter the IP address of the CNC Console system. The login page opens.
2. Enter your **Username**.
3. Enter your **Password**.
4. Click **Login**.  
The main page opens.

**Figure 4-1 CNC Console Interface**



You are logged in. All the PCF related configurations are available in the left navigation menu under **PCRF**.

## Configuring Services and Manageable Objects

This section describes how to create and manage the services and manageable objects that are available to be used in policies.

## Managing a Charging Server

This chapter describes how to define and manage charging servers within the CNPCRF GUI. A charging server is an application that calculates billing charges.

To define a charging server:

1. From the navigation pane, under **PCRF**, then under **Configurations**, select **Charging Servers**.

The **Charging Server Management** screen appears.

2. Click **Add**.

The Create Charging Server page opens.

3. (Required) Enter the **Name** for the charging server.

The name can only contain the characters A through Z, a through z, 0 through 9, period (.), hyphen (-), and underline (\_).

4. Enter the **Description/Location**.

Free-form text that identifies the charging server within the network. Enter up to 250 characters.

5. (Required) Enter the **Host Name**.

The FQDN (fully qualified domain name assigned) to the charging server.

6. Enter the **Port** number on which the charging server is listening for messages.

If left blank, port 3868 is used.

7. Select the **Transport** protocol used to communicate with the charging server:

Available options include:

- **tcp**  
Transmission Control Protocol (used with TACACS+)
- **udp**  
User Datagram Protocol (used with RADIUS)

 **Note:**

If you configure the Transport protocol as **udp**, you cannot configure the AAA Protocol as **diameter**.

- **sctp**  
Stream Control Transmission Protocol
8. Select the Authentication, Authorization, and Accounting (AAA) **Protocol** used to communicate with the charging server.

Available options include:

- **diameter**
- **radius**
- **tacacs+**

 **Note:**

If you configure the Transport protocol as **udp**, you cannot configure the AAA Protocol as **diameter**.

9. Select if transport **Security** is used to communicate with the charging server.
10. Click **Save**.

The charging server is displayed on the Charging Server Management page.

 **Note:**

Use pencil icon or trash bin icon available in the next column to edit or update the created charging server.

## Managing Custom AVPs

This chapter describes how to create, modify, and delete custom third-party attribute-value pairs (AVPs) in the CNPCRF User Interface (UI).

In a wireless network, custom AVPs are used to encapsulate protocol-specific data for routing, authentication, authorization, and accounting information.

### About Custom AVPs

An attribute-value pair (AVP) is used to encapsulate protocol-specific information with usage monitoring supported by the MPE device. Diameter messages such as RAA, CCA, CCR, and RAR are supported by third-party AVP policy conditions. The supported outgoing Diameter messages set or remove third-party AVPs.

 **Note:**

The Diameter messages listed are examples only. There are many messages associated with Diameter.

You can create policy conditions to evaluate the presence of both standard (base) and third-party AVPs in Diameter messages or group AVPs during policy execution. A policy condition can check for the presence of both standard and third-party AVPs in incoming Diameter messages and evaluate their values. A policy action can use standard and third-party AVPs for routing, authentication, authorization, and accounting.

Standard AVPs can be included in third-party AVP conditions and actions. To include a standard (base) AVP in a nonstandard application message, or to use a pre-standard AVP as a standard AVP, define it as a custom AVP.

When defined, custom AVPs are located at the end of a parent Diameter message or group AVP. If the parent AVP is null, the custom AVP is inserted at the root level of the



message. For example, a custom AVP definition appears at the end of this Charging-Rule-Install message:

```
Charging-Rule-Install ::= < AVP Header: 1001 >
*[ Charging-Rule-Definition ]
*[ Charging-Rule-Name ]
*[ Charging-Rule-Base-Name ]
[ Bearer-Identifier ]
[ Rule-Activation-Time ]
[ Rule-Deactivation-Time ]
[ Resource-Allocation-Notification ]
[ Charging-Correlation-Indicator ]
*[ customAVP ]
```

A Set or Get SPR user attribute value can be set to the defined third-party AVP in Diameter messages. You can also set or remove defined third-party AVPs during the execution point.

A third-party AVP is identified by a unique identifier in the following format:

*name:vendorId*

For example:

#### **Condition**

where the request AVP *NEW\_AVP3:555* value is numerically equal to 2012

#### **Parameters**

The AVP name and vendor ID. In the example, the vendor ID is 555.

#### **Description**

A well-defined AVP custom name is referred to if the vendor ID is not specified.

When entering and sending a new third-party AVP definition to an MPE or MRA device, the definition must include the AVP name, code, vendor ID, data type, and an optional AVP flag.

Validation of the AVP code, Name, and vendor ID prohibits a user from overwriting the existing base AVPs.

These AVP actions include the ability to perform the following:

- Routing
- Authentication
- Authorization
- Accounting

## Creating a Custom AVP

To create a Custom AVP:

1. From the **PCRF** section of the navigation pane, select **Custom AVP** under **Configurations** .

The **Custom AVP Management** screen appears.

**2. Click Add.**

The Create Custom AVP page opens.

**3. Enter information as appropriate:****a. AVP Name (required)** — The name you assign to the AVP.

The name can only contain the characters A–Z, a–z, 0–9, period (.), hyphen (-), and underline (\_). The maximum length is 255 characters.

**b. Description** — Free-form text that identifies the AVP.

Enter up to 250 characters.

**c. AVP Code (required)** — A unique numeric value assigned to the new AVP.**d. Vendor** — Select a vendor from the vendor list.

To add a vendor to the list, see [Managing Custom Vendors](#).

**e. Mandatory Flag (optional)** —**f. Protect Flag (optional)** — When checked, specifies the protected AVP values.**g. May Encrypt Flag** — The AVP is encrypted if the checkbox is specified.**h. Vendor Specific Flag** — The AVP is vendor specific if the checkbox is specified. **Note:**

This box is checked automatically if the value of the vendor ID is not 0.

**i. AVP Type (required)** — Select the data type from the list:

- **address**
- **enumerated**
- **float32**
- **float64**
- **grouped**
- **id**
- **int32**
- **int64**
- **ipFilterRule**
- **octetString**
- **time**
- **uint32**
- **uint64**
- **uri**
- **utf8String**

- j. **Parent AVP** — If the AVP is a member of a grouped AVP, then the parent AVP must be specified. Select one of the following from the list:
- **ADC-Rule-Definition:10415**
  - **ADC-Rule-Install:10415**
  - **ADC-Rule-Remove:10415**
  - **ADC-Rule-Report:10415**
  - **AF-Correlation-Information:10415**
  - **Acceptable-Service-Info:10415**
  - **Access-Network-Charging-Identifier-Gx:10415**
  - **Access-Network-Charging-Identifier:10415**
  - **Access-Network-Physical-Access-ID:10415**
  - **Allocation-Retention-Priority:10415**
  - **Application-Detection-Information:10415**
  - **CC-Money**
  - **Charging-Information:10415**
  - **Charging-Rule-Definition-3GPP2:5535**
  - **Charging-Rule-Definition:10415**
  - **Charging-Rule-Event-Cisco:9**
  - **Charging-Rule-Event-Trigger-Cisco:9**
  - **Charging-Rule-Install-3GPP2:5535**
  - **Charging-Rule-Install:10415**
  - **Charging-Rule-Remove:10415**
  - **Charging-Rule-Report-3GPP2:5535**
  - **Charging-Rule-Report:10415**
  - **Codec-Data-Tmp:10415**
  - **Codec-Data:10415**
  - **Cost-Information**
  - **Default-EPS-Bearer-Qos:10415**
  - **E2E-Sequence**
  - **Envelope:10415**
  - **Event-Report-Indication:10415**
  - **Explicit-Route-Record:21274**
  - **Explicit-Route:21274**
  - **Failed-AVP**
  - **Final-Unit-Indication**
  - **Flow-Description-Info:5535**
  - **Flow-Description:10415**

- **Flow-Grouping:10415**
- **Flow-Info:5535**
- **Flow-Information:10415**
- **Flow:10415**
- **G-S-U-Pool-Reference**
- **Granted-Qos:5535**
- **Granted-Service-Unit**
- **Juniper-Discovery-Descriptor:2636**
- **Juniper-Provisioning-Descriptor:2636**
- **LI-Indicator-Gx:12951**
- **LI-TargetMFAddr:12951**
- **Media-Component-Description:10415**
- **Media-Sub-Component:10415**
- **Multiple-Services-Credit-Control**
- **Offline-Charging:10415**
- **PCEF-Forwarding-Info:971**
- **PCEF-Info:971**
- **PS-Furnish-Charging-Information:10415**
- **PS-information:10415**
- **Packet-Filter-Information:10415**
- **Qos-Information-3GPP2:5535**
- **Qos-Information:10415**
- **Qos-Rule-Install:10415**
- **Qos-Rule-Definition:10415**
- **Qos-Rule-Remove:10415**
- **Qos-Rule-Report:10415**
- **Reachable-Peer:21274**
- **Redirect-Information:10415**
- **Redirect-Server**
- **Requested-Qos:5535**
- **Requested-Service-Unit**
- **Service-Information:10415**
- **Service-Parameter-Info**
- **Siemens-DL-SDP-Data:4329**
- **Siemens-UL-SDP-Data:4329**
- **Subscription Id**
- **Subscription-Id-3GPP:10415**

- **Supported-Features:10415**
  - **TDF-Information:10415**
  - **TFT-Packet-Filter-Information:10415**
  - **TMO-Redirect-Server-29168**
  - **Time-Quota-Mechanism:10415**
  - **Trigger:10415**
  - **Tunnel-Header-Filter:10415**
  - **Unit-Value**
  - **Usage-Monitoring-Control:21274**
  - **Usage-Monitoring-Information:10415**
  - **Used-Service-Unit**
  - **User-CSG-Information:10415**
  - **User-Equipment-Info**
  - **User-Location-Info-3GPP:10415**
  - **VZW-Access-Network-Physical-Access-ID:12951**
  - **Vendor-Specific-Application-Id**
  - **Vzw-Trigger:12951**
4. Click **Save**.
  5. If the AVP name matches the name of a standard AVP, a confirmation message displays. Click **OK** to overwrite the existing AVP.

The AVP is created.


## Modifying an AVP

To modify an AVP:

1. From the **PCRF** section of the navigation pane, select **Custom AVP** under **Configurations** .

The **Custom AVP Management** page opens in the work area, listing the defined AVPs.




2. From the work area, click  (pencil icon), located to the right of the AVP.  
The Edit Custom AVP page opens.
3. Modify AVP information as required.  
For a description of the fields contained on this page, see [Creating a Custom AVP](#).
4. Click **Save**.

The AVP is modified.

## Deleting an AVP

To delete an AVP:

1. From the **PCRF** section of the navigation pane, select **Custom AVP** under **Configurations** .  
The **Custom AVP Management** page opens in the work area, listing the defined AVPs.
2. From the work area, click  (trash can icon), located to the right of the AVP.  
A confirmation message displays.
3. Click **OK**.  
The AVP is deleted.

## Managing Custom Vendors

This chapter describes how to create, modify, and delete custom vendor definitions in the CNPCRF User Interface (UI).

Custom vendors are used in RADIUS Change of Authorization (CoA) messages.

### About Custom Vendors

A custom vendor is used to define a vendor in the CNPCRF system. This dictionary includes vendor IDs and text descriptions. You can define custom vendors and add them to the dictionary.

### Creating a Custom Vendor

To create a custom vendor:

1. From the **PCRF** section of the navigation pane, select **Custom Vendor** under **Configurations** .  
The **Custom Vendor Management** screen appears.
2. Click **Add**.  
The Create Custom Vendor page opens.
3. Enter information as appropriate:
  - a. **Name** (required) — The name you assign to the vendor.  
The name can only contain the characters A–Z, a–z, 0–9, period (.), hyphen (-), and underline (\_).
  - b. **Description** — Free-form text that identifies the vendor.  
Enter up to 250 characters.
  - c. **Vendor Id** — Enter the vendor ID.  
Enter a positive integer.
4. Click **Save**.  
The vendor is created.

### Modifying a Custom Vendor

To modify a custom vendor definition:

1. From the **PCRF** section of the navigation pane, select **Custom Vendor** under **Configurations** .

The **Custom Vendor Management** page opens in the work area, listing the defined vendors.



2. From the work area, click (pencil icon), located to the right of the vendor.

The Edit Custom Vendor page opens.

3. Modify Vendor information as required.

For a description of the fields contained on this page, see [Creating a Custom Vendor](#) .

4. Click **Save**.

The Vendor is modified.

## Deleting a Custom Vendor

You cannot delete a custom vendor definition that is used in a CoA template.

To delete a custom vendor definition:

1. From the **PCRF** section of the navigation pane, select **Custom Vendor** under **Configurations** .

The **Custom Vendor Management** page opens in the work area, listing the defined vendors.

2. From the work area, click  (trash can icon), located to the right of the vendor.

A confirmation message displays.

3. Click **OK**.

The custom vendor definition is deleted.

## Creating a Media Profile

This section defines how to manage media profiles in the CNPCRF GUI. In a cable network, a media profile describes a CODEC supported for Rx-to-PCMM translation.



### Note:

Media Profiles is a function that is applicable to Cable mode only.

To create a media profile:

1. From the navigation pane, under **PCRF**, then under **Configurations**, select **Media Profiles**.

The **Media Profile Management** screen appears.

2. Click **Add**.

The Create Media Profile page opens.

3. Enter the following information:
  - a. **ID** — Unique ID assigned to the media profile.
  - b. **Name** — Unique name assigned to the media profile.
  - c. **Description** — specifies the description of the media profile.
  - d. **Codec Name** — Unique media subtype assigned to the media profile.  
This is defined in the IANA MIME registration for the CODEC. Enter a string of up to 255 characters.
  - e. **Transport Type** — Select from the following:
    - **RTP/AVP** (default) — RTP audio-video profile.
    - **RTP/SAVP** — RTP secure audio-video profile.
    - **RTP/AVPF** — RTP extended audio-video profile with feedback.
  - f. **Payload Number** — The payload number.  
Valid payload numbers range from 0 through 127. Enter -1 to indicate an unknown payload number.

 **Note:**

You cannot add a CODEC that is predefined with a payload number in the range of 0 to 96.

- g. **Sample Rate (kHz)** — The sampling rate of the CODEC in KHz.  
The valid range is an integer from 1 through 100 KHz.
  - h. **Frame Size in Milliseconds** — The size of one audio frame in milliseconds.  
This is the length of time represented by one audio frame. A single RTP packet may contain multiple audio frames. The bitrate is calculated using the frame size in milliseconds, the frame size in bytes, and the packetization time. The valid range is 0 through 100 ms.
  - i. **Frame Size in Bytes** — The size of one audio frame size in bytes.  
This is the size represented by one audio frame. A single RTP packet may contain multiple audio frames. The bitrate is calculated using the frame size in milliseconds, the frame size in bytes, and the packetization time. The valid range is 1 through 1,500 bytes.
  - j. **Packetization Time** — The length of time, in milliseconds, represented by the media in a packet.  
The bitrate is calculated using the frame size in milliseconds, the frame size in bytes, and the packetization time. The valid range is 1 through 100.
  - k. **Always Use Default Ptime** — Select to always use the default packetization time, ignoring the value received in the SDP message.  
The default is unchecked.
4. Click **Save**.  
The media profile is created.



**Note:**

Use pencil icon or trash bin icon available in the next column to edit or update the created media profile.

## Session Viewer

The Session Viewer displays detailed session information for a specific subscriber. Within the session viewer, you can enter query parameters to render session data for a specific subscriber. This section provides information about viewing the sessions.

To view the sessions:

1. From the navigation menu, under **PCRF**, click **Session Viewer**. The Session Viewer page appears.
2. From the **Identifier Type** drop-down menu, select the identifier type for the selected session type. Possible values are:
  - DIAMETER SESSION ID
  - IMSI
  - MSISDN
  - IPV4
3. Enter the value in the **Identifier Value** field for the selected identifier type.
4. Click **Query**. Information about the subscriber session(s) is displayed.

If session data is not available, the error is displayed along with No session found.

## Configuring Core Service

You can configure the CNPCRF core service from this page.

To configure the Core Service:

1. From the navigation menu, under **PCRF**, then under **Services**, click **Core Service**.  
The Core Service screen appears.
2. Click **Edit** to edit the core service configurations. This enables the **Add** button in **Advance Settings** group.
3. Click **Add**. The **Add Advanced Settings** window opens.
4. Enter the values in **Key** and **Value** fields.
5. Click **Save**.

## Managing Policy

Cloud Native Policy and Charging Rules Function (CNPCRF) offers a Policy Design editor based on Blockly interface. You can create and manage a policy project for PCRF core service.

## Settings

You can manage and view the CNPCRF supported services from this page.

To edit the Settings:

1. From the navigation menu, under **Policy Management**, click **Settings**. The Policy Runtime Environment screen appears.
2. Click **Edit** to edit the settings.
3. Enter the value in **Log Level** field. The default value is WARN.
4. Click **Add** in the **Supported Services** group. The Add Supported Services screen appears.
5. Enter the following information to create service:
  - **Service Name**: Enter the service name.
  - **Service Label**: Enter the service label.
  - **Relative URL**: Enter the relative URL.
6. Click **Save**. The services get listed in the Supported Services list. The supported services are pcrf-core and pds.

 **Note:**

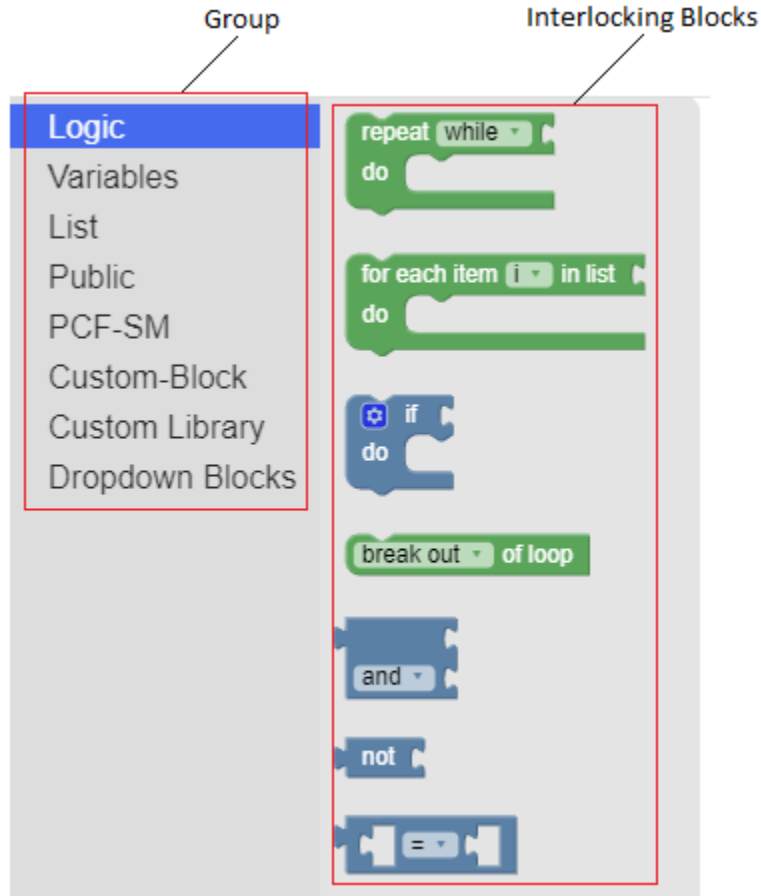
Use **Edit** or **Delete** buttons available in the next column to update or delete the services.

## Creating a Policy Project

To create a policy project:

1. From the **Policy Management** section of the navigation pane, select **Policy Projects**.
2. Click **Create**. The Create Project window opens.
3. In the **Name** field, enter the name for the project.
4. In the **Description** field, enter the description for the project.
5. In the **Service Type**, select the service from list of services already created in **Settings** section.
6. Click **Save**. The policy project is created.
7. Select the policy project created and click **Open**. This opens a Blockly editor. You can construct one or more policies as required using the building blocks provided in the Left Side Panel of the editor construct one or more Policies as required.

The following screen capture shows an example of how the policies can be created using the building blocks.



8. Click **Save**.  
The policy for the selected policy project is created.

## Managing State Variables

State Variables are set within a policy action to be used at a later time during policy rule execution (in either conditions or actions). The names of these variables are not predefined and are determined at the time of creation. State variables have a scope which determines how long the value persists after it is set. The scopes are:

- **Subscriber Policy Evaluation State Variables** — This variable exists locally and has a value as long as the associated subscriber has at least one session. After the last session is terminated these variables no longer have value and will no longer be available for use in policies.
- **Session State Variables** — This variable has a value that is saved as long as the session the variable is associated with is still valid. After the session is terminated, this variable no longer has value and will no longer be available for use in policies.
- **Policy Evaluation State Variables** — This variable are available for the lifetime of a policy evaluation cycle (the process of evaluating all the policies for a single request or context)
- **Data Source State Variables** —

 **Note:**

State Variables are only supported for Session Management service.

### Creating State Variables Condition

You can use the default blocks provided in **Variables** section to create state variables condition. However, the following blocks in the **State Variables** section under the **Public** section can also be used to create these conditions:

#### Syntax

*operation variable-name* in *scope* context

#### Parameters

##### *operation*

One of the following:

- **Save**- To save the state variable in a specific context.
- **Load**- To load the state variable from the selected context into policy evaluation.
- **Remove**- To remove the state variable from the selected context.
- **Remove All**- To remove all the state variable from the selected context

##### *variable-name*

String. Specifies name of the state variable.

##### **Scope**

One of the following:

- **Policy**- Policy evaluation variables that last only for the duration of policy evaluation cycle.
- **Session**- Session variables that have a value as long as the session they are associated with is open.
- **Subscriber**- Subscriber variables that are associated with a subscriber that has at least one session.
- **Data source**

## Data Model

You can create and manage sample attributes for policy. This is used for testing the policies.

To create the Data Model from this page:

1. From the navigation menu, under **System Administration**, click **Data Model**. The **Data Model Management** screen appears with the listing of all the attributes created. You can create or import new attributes from this page.

 **Note:**

Click the **Export** button to download the available listings to your system.

2. Click **Add**.  
The **Create Data Model** screen appears.
3. On the **Create Data Model** screen, enter values for the input fields.  
The following table describes the fields:

Field Name	Description
ID	ID of the attribute, not displayed on the GUI.
Name	Name of the attribute, not displayed on the GUI.
Label Name	Name of the attribute, displayed on the GUI.
Description	Description of the attribute
Type	Select one of the values: enum or object

4. In the **Fields** group, click **Add** to add the field details:
  - a. Enter the applicable values in the input fields available on the window.  
The following table describes the fields:

Field Name	Description
Name	Name of the field, not displayed on the GUI.
Description	Description of the field
Label Name	Name of the field, displayed on the GUI.
Type	Select either of the values from drop-down (primitive, object, array)
Primitive Type	Defines the primitive type
Units	Specifies the units
Object Type	Defines the object type
<b>Item Type</b>	
Type	Select either of the values from drop-down (primitive, object)
Primitive Type	Defines the primitive type
Object Type	Defines the object type

 **Note:**

Click **Remove** to cancel the changes.

- b. Click **Save**.
5. In the **Enum Items** group, click **Add** to add the field details:
  - a. Enter the applicable values in the input fields available on the window.  
The following table describes the fields:

Field Name	Description
Name	Name of the field, not displayed on the GUI.
Value	Specify the value.

- b. Click **Save**.
6. Click **Save**.  
The value gets listed on the **Data Model Management** screen.



**Note:**

Use **Edit** or **Delete** buttons available in the next column to update or delete the listing.

### Importing the Data Model

To import the session rules:

1. Click **Import**.  
The **File Upload** window appears on the screen.
2. Upload the files in required format by clicking **Drop Files here or click to upload**.

## Configuring Policy Common Configurations

This chapter describes how to configure the managed objects which are common to Policy Control Function and Cloud Native Policy Charging and Rules Function.

### Connecting to LDAP Data Source

PCRF-core establishes connections with data sources to retrieve information about subscribers from the database. The PCRF-core queries a data source using a key attribute that uniquely identifies a subscriber and stores the results in its cache. A data source uses this key attribute (for example, the phone or account number of the subscriber) to index the information contained in the database.

Oracle Communications Cloud Native Core Policy and Charging Rule Function (PCRF) supports Lightweight Directory Access Protocol (LDAP) data source. Based on the conditions implemented in CNPCRF system, Policy Data Source (PDS) would retrieve all the relevant information from LDAP data source based on the rules configured in the system through LDAP gateway.

LDAP credentials are stored as kubernetes secret along with Authentication DN and LDAP name. You must create a kubernetes secret to store LDAP credentials before setting a PDS as LDAP data source.

To create a kubernetes secret for storing LDAP credentials:

1. Create a yaml file with the following syntax:

```
apiVersion: v1
kind: Secret
metadata:
```

```

name: ldapsecret
labels:
  type: ocpm.secret.ldap
type: Opaque
stringData:
  name: "ldap1"
  password: "camiant"
  authDn: "uid=PolicyServer,ou=vodafone,c=hu,o=vodafone"

```

where, *name* is the configured LDAP server name.

*password* is the LDAP credential for that data source.

*authDN* is the authentication DN for that LDAP datasource.

 **Note:**

For different LDAP data sources more entries can be added in above format only the key of the entry should be the ldap name specified in CNPCRF Graphical User Interface (GUI).

2. Create the secret by executing the following command:

```
kubectl apply -f yaml_file_name -n pcrf-namespace
```

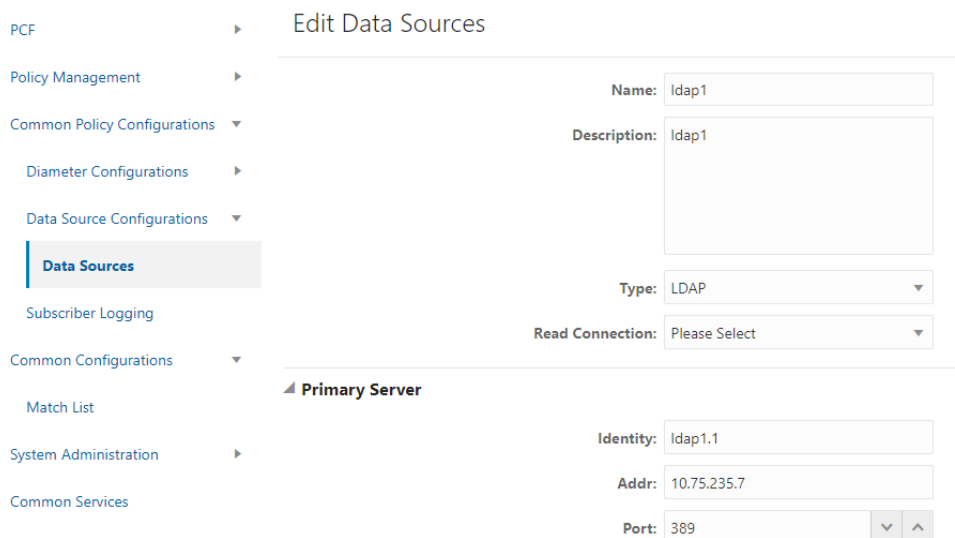
where:

*yaml\_file\_name* is a name of the yaml file that is created in step 1.

*pcrf-namespace* is the deployment namespace used by the helm command.

To set Policy Data Source as LDAP Data Source using CNPCRF GUI:

1. Add LDAP data source. To add LDAP source, From the navigation menu, under **Common Policy Configuration**, then under **Data Source Configurations**, click **Data Sources**. In the **Type** drop-down list, select **LDAP**. The following screen capture shows the example of adding LDAP data source in GUI:



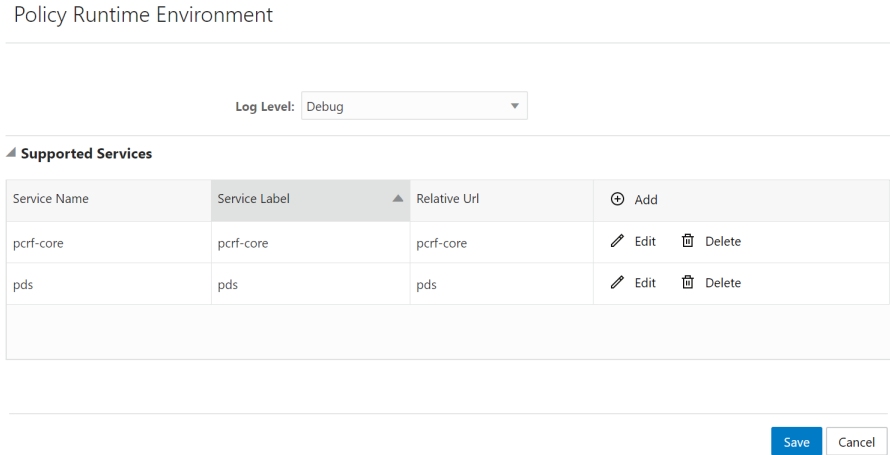
The screenshot displays the 'Edit Data Sources' interface in the CNPCRF GUI. On the left, a navigation menu lists various configuration sections, with 'Data Sources' currently selected. The main panel contains a form for configuring a new data source. The 'Name' field is set to 'ldap1', and the 'Description' field also contains 'ldap1'. The 'Type' dropdown menu is set to 'LDAP', and the 'Read Connection' dropdown is set to 'Please Select'. Below this, the 'Primary Server' section is expanded, showing fields for 'Identity' (ldap1.1), 'Addr' (10.75.235.7), and 'Port' (389).

In the above example, LDAP datasource with name **LDAP1** is created.

2. Edit the **pcrf-core** deployment file in vi editor to enable policy data source:

```
- name: SH_ENABLED
  value: "false"
- name: SY_ENABLED
  value: "false"
- name: USERSERVICE_ENABLED
  value: "true"
```

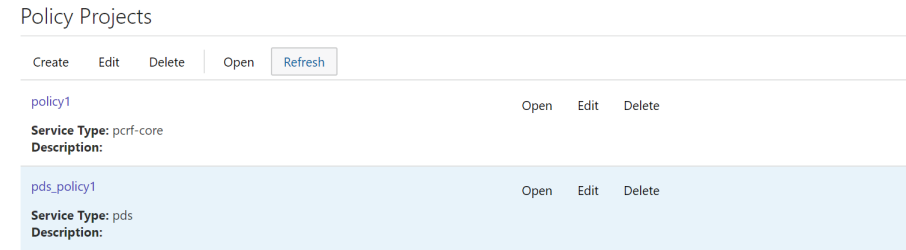
3. Create **pds** service type in CNPCRF system. To create **pds** service type, From the navigation menu, under **Policy Management**, click **Settings** . On Settings page, click **Add** to create **pds** service type. The following screen capture shows the example of creating **pds** service type in GUI:



In the above example, **pds** service type is created.

**Note:**  
The service name should be entered as **pds**.

4. Create Policy Project with **pds** Service Type. From the navigation menu, under **Policy Management**, click **Policy Projects**. On Policy Projects page, click **Create** to create policy project. While creating a policy project select **pds** as a service type. The following screen capture shows the example of creating policy project with **pds** service type in GUI:

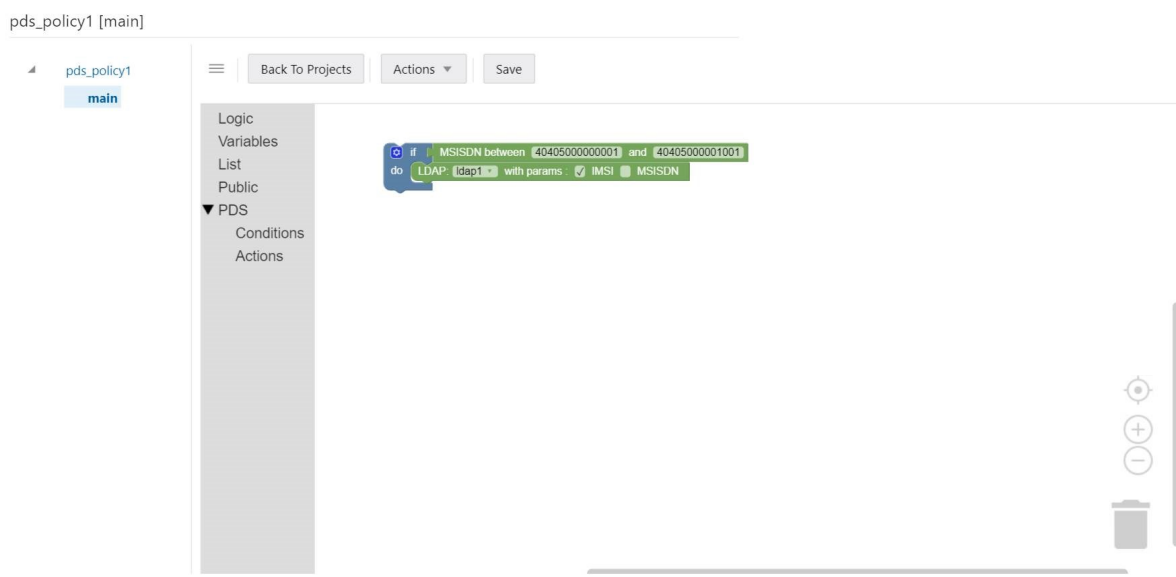


In the above example, **s** policy project is created with **pds** service type.



5. Create policy action and condition in previously created policy project. Click **Open** for the selected policy project and you can see the project is a file. You can create the policy action and condition by using the different blocks available under **Conditions** and **Actions** under **PDS**.

The following screen capture shows the example of creating policy action and condition in GUI:



In the above example, if request received for configured IMSI ranges between 40405000000001 and 40405000000001, then PCF will forward request to PDS and PDS will forward the request to LDAP gateway to lookup user information in LDAP1.

## Managing Match Lists

In a wireless network, a match list is a set of defined values that can represent, for example, IDs or Internet addresses. Match lists provide whitelist and blacklist functions in policy rules. Match lists support wildcard matching.

A match list is a set of values in various categories, including access point names (APNs), subscriber IMSIs, location area codes (LACs), service area codes (SACs), Internet addresses, and user equipment identities. A match list can function as a whitelist (listing items to be included) or a blacklist (listing items to be excluded). By using a match list, you can, for example, apply a policy to all subscribers in a set of LACs, or block access to a list of Internet addresses known to be high risk. Match lists support wildcards. Using wildcards, a range of values can be specified compactly.

### Creating a Match List

To create a match list:

1. From the navigation pane, under **Policy Common Configurations**, select **Match List**.  
The **Match List Management** page opens in the work area.
2. Click **Create**.  
The Create Match List page opens.

3. Enter the following information:
  - **ID:** The ID assigned to the match list.
  - **Name:** The name assigned to the match list.  
The name can only contain the characters A-Z, a-z, 0-9, period (.), hyphen (-), and underline (\_). The maximum length is 40 characters.
  - **Description:** Free-form text
  - **Type:** Select from the following:
    - **string** (default) - The list consists of strings.
    - **wildcard string** - The list consists of wildcard match patterns that use an asterisk (\*) to match zero or more characters or a question mark (?) to match exactly one character.
  - **Items:**
4. Click **Save**.

The match list is defined in the database and can now be used in a policy.

### Modifying a Match List

To modify a match list:

1. From the navigation pane, under **Policy Common Configurations**, select **Match List**.  
The **Match List Management** page opens in the work area, displaying the list of defined match lists.
2. Select the match list you want to modify.
3. Click **Edit**.  
The Edit Match List page opens.
4. Modify match list information as required.
5. Click **Save**.  
The match list is modified.

### Deleting a Match List

To delete a match list:

1. From the navigation pane, under **Policy Common Configurations**, select **Match List**.  
The **Match List Management** page opens in the work area, displaying the list of defined match lists.
2. Select the match list you want to delete.
3. Click **Delete**.  
A confirmation message displays.
4. Click **OK**.  
The match list is deleted.

### Importing the Match Lists

To import the match lists:

1. Click **Import**.

The **File Upload** window appears on the screen.

2. Upload the files in required format by clicking **Drop Files here or click to upload**.

### Exporting the Match Lists

You can export the match lists by clicking **Export All**. The Match Lists will be downloaded in a local machine.

## Importing Configurable Objects

This section describes how to perform a bulk import of configurable objects into the system.

### Importing Configuration Object Files

To import json or ZIP files:

1. From the navigation pane, under **Policy Common Configurations**, click **Bulk Import**.  
The **Upload** option appears on the screen.
2. Click **Upload**.  
Locate the file to be imported.
3. Select a processing option to use to **Handle collisions between imported items and existing items**:
  - **Delete all before importing** The system deletes all objects for each object type matching the import file before importing the object type json file.  
**Attention:** This import strategy can result in object inconsistency. For example, if you import a ZIP file that only contains traffic profiles, all the traffic profiles are deleted first. However, if existing policies depend on the existing traffic profiles, and the import file does not contain them, the policies can become invalid.
  - **Overwrite with imported version** For each object in the import file, if the object exists in the system, the import updates the object with the configuration contained in the import file. If an object does not exist, the system adds the object to the system.
4. Click **Import**.

The configuration objects and their configuration settings are imported into the database. After the import is complete, the window reports the results for each json file contained in the ZIP file.

## Exporting Configurable Objects

This section describes how to perform a bulk export of configurable objects.

### Exporting All Configuration Object Files

To export all configuration objects:

1. From the navigation pane, under **Policy Common Configurations**, click **Bulk Export**.  
The **Export All** option appears on the screen.
2. Click **Export All**.

A ZIP file is downloaded to your local computer.

# 5

## Writing Policy Conditions

The policy wizard supports a large number of conditions that can be used for constructing policy rules. To help you find the conditions you want, the conditions are organized into different categories, which are summarized in

The conditions that are included within each of these categories are described in the sections that follow. Within each category, conditions are listed in alphabetical order. The parameters that can be modified within each condition are also detailed.

### Policy Condition Categories

#### **Mobility**

Conditions that are based on information associated with wireless networks that include mobile subscribers. See [Mobility Conditions](#).

#### **Network Devices**

Conditions related to the specific network device for which the policy rule is being evaluated. This includes conditions based on the network device type, as well as those that refer to specific unique identifiers for network devices. See [Network Device Conditions](#).

#### **Policy SDP Properties**

Conditions related to SDP properties that are used to check the codec type (offer/answer) for the device (remote/local). See [Policy SDP Properties Conditions](#).

#### **Request**

Conditions that are based on information that is explicitly contained within or related to the protocol message (request) that triggered the policy rule execution. See [Request Conditions](#).

#### **Time of Day**

Conditions related to the time at which the policy rules are being executed. See [Time of Day Conditions](#).

### Mobility Conditions

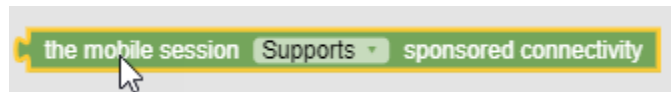
Mobility conditions are based on information associated with networks that include mobile subscribers (such as a wireless network).

where the mobile session *supports* sponsored connectivity

#### **Syntax**

where the mobile session *support* sponsored connectivity

The following screen capture shows a sample policy condition:



### Parameters

#### **support**

One of the following:

- **does not support**
- **supports** (default)

### Description

Triggers a policy that evaluates whether or not the mobile session supports sponsored data connectivity. This condition supports sponsored data connectivity for both Gx and Rx requests.

### Example 5-1 Example

The following condition evaluates as true if the mobile session supports sponsored data connectivity:

```
where the mobile session supports sponsored connectivity
```

where the Cell Identifier *matches one of specified CI value(s)*

### Syntax

where the Cell Identifier *matches-op match-list*

The following screen capture shows a sample policy condition:



### Parameters

#### **matches-op**

One of the following:

- **matches one of** (default)
- **does not match any of**

#### **match-list**

A comma-separated list of values, where each value is a wildcard match pattern that uses the \* (asterisk) character to match zero or more characters and the ? (question mark) character to match exactly one character.

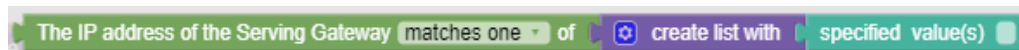
### Description

Triggers a policy that is only evaluated for one or more specific Cell Identifier values (based on matching wildcard patterns). A valid Cell Identifier is an integer between 0 and 65535.

where the IP address of the Serving Gateway *matches one of specified value(s)(es)*

### Syntax

where the IP address of the Serving Gateway *matches-op match-list*  
The following screen capture shows a sample policy condition:



### Parameters

#### *matches-op*

One of the following:

- **matches one of** (default)
- **does not match any of**

#### *match-list*

A comma-separated list of values, where each value is a wildcard match pattern that uses the \* (asterisk) character to match zero or more characters and the ? (question mark) character to match exactly one character.

### Description

Triggers a policy that is only evaluated for one or more specific Serving Gateway addresses (based on matching wildcard patterns).

where the IP-CAN type *specified*

### Syntax

where the IP-CAN type *ip-can-type*

The following screen capture shows a sample policy condition:



### Parameters

#### *ip-can-type*

One or more of the following:

- **THREEGPP\_GPRS**
- **THREEGPP\_EPS**
- **NON\_THREEGPP\_EPS**
- **THREEGPP2**
- **WIMAX**
- **DOCSIS**

- XDSL
- WIRELINE\_ACCESS

#### Description

Triggers a policy that is only evaluated for a protocol message with a specific IP-CAN type.

where network initiated requests are *supported*

#### Syntax

where network initiated requests are *support*

The following screen capture shows a sample policy condition:

A screenshot of a policy condition in a configuration tool. The text 'network initiated requests are supported' is displayed in a green box with a yellow border. The word 'are' is in a smaller font and has a small downward arrow next to it.

#### Parameters

##### *support*

One of the following:

- **not supported**
- **supported** (default)

#### Description

Triggers a policy that is only evaluated when network initiated requests are or are not supported.

where the E-UTRAN Cell Identifier *matches one of specified ECI value(s)*

#### Syntax

where the E-UTRAN Cell Identifier *matches-op match-list*

The following screen capture shows a sample policy condition:

A screenshot of a policy condition in a configuration tool. The text 'The E-UTRAN Cell Identifier matches one of create list with specified ECI value(s)' is displayed in a green box with a yellow border. The words 'matches one' and 'of' are in a smaller font. There is a small icon of a gear next to 'create list with' and a small square icon next to 'specified ECI value(s)'.

#### Parameters

##### *matches-op*

One of the following:

- **matches one of** (default)
- **does not match any of**

##### *match-list*

A comma-separated list of values, where each value is a wildcard match pattern that uses the \* (asterisk) character to match zero or more characters and the ? (question mark) character to match exactly one character.



## Description

Triggers a policy that is only evaluated for one or more specific E-UTRAN Cell Identifier values (based on matching wildcard patterns).

where the subscribed PRA area *matches one of PRA area(s)*

## Syntax

where the subscribed PRA area *matches-op pra-areas*

The following screen capture shows a sample policy condition:



## Parameters

### *matches-op*

One of the following:

- **matches one of** (default)
- **does not match any of**

### *pra-areas*

A single area or multiple specific PRA areas selected from the defined PRA areas, manually input, or Default.

- **CMP defined PRA lists**  
Select one or more defined PRA lists
- **Manual Input**  
Enter the identifier for the PRA in hexadecimal format or a custom PRA from a subscriber profile in the format *{User.CustomField}*.

The manual input format for multiple PRAs is:

```
PRA
identifier1 [;PRA element list1],PRA identifier2 [; PRA element
list2],...
```

The format of PRA identifier and PRA Element List is according to section 8.108 of TS 29.274[9]. It is specified in Hexadecimal format. If only has a PRA identifier then the PRA area is a predefined PRA area. If both PRA identifier and PRA Element List exists then it is a UE-dedicated PRA Area. The manual input is typically used to input a temporarily PRA area. The manual input can also be used to get a PRA area from Custom field of subscriber. For example, *{User.Custom4}*. If the operator wants to manually input a PRA area they need to get the Hexadecimal value for each. Different vendors/operators should interact or exchange PRA info using the Hexadecimal representation as defined in section 8.108 of TS 29.274[9].

The manual input format for a single PRA is:

```
PRA identifier [, PRA element list]
```

- **Default**

The PRA to which the user equipment is already subscribed, if any.

The Default option specifies using the default PRA area. The default PRA is the PRA area subscribed or provisioned by the PCRF for the UE during IP-CAN session life cycle. It is either a UE- dedicated or predefined PRA area. It can be a PRA area that is retrieved from the subscriber profile (normally UE-dedicated) or a PRA area defined in the CMP (normally predefined). When used , this Default option means to check whether the UE has subscribed to a PRA area but does not care what the PRA area is.

### Description

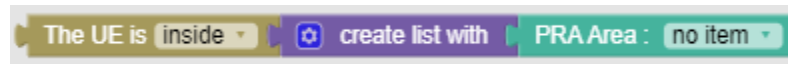
Triggers a policy that is only evaluated for one or more specific PRA values. If **default area** is selected as the definition for the parameter *pra-areas*, the policy is only evaluated if the user equipment is already subscribed to a PRA.

where the UE is *inside/outside/inactive* for any one of *PRA Area*

### Syntax

where the UE is *location* subscribed PRA area *PRA areas*

The following screen capture shows a sample policy condition:



### Parameters

#### *location*

One of the following:

- **inside**
- **outside**
- **inactive**

#### *pra-areas*

A single area or multiple specific PRA areas selected from the defined PRA areas, manually input, or Default.

- **CMP defined PRA lists**  
Select one or more defined PRA lists
- **Manual Input**  
Enter the identifier for the PRA in hexadecimal format or a custom PRA from a subscriber profile in the format *{User.CustomField}*.

The manual input format for multiple PRAs is:

```
PRA
identifier1 [;PRA element list1],PRA identifier2 [; PRA element
list2],...
```

The format of PRA identifier and PRA Element List is according to section 8.108 of TS 29.274[9]. It is specified in Hexadecimal format. If only has a PRA identifier then the PRA area is a predefined PRA area. If both PRA identifier and PRA Element List exists then it is a UE-dedicated PRA Area. The manual input is typically used to input a temporarily PRA area. The manual input can also be used to get a PRA area from Custom field of subscriber. For example, {User.Custom4}. If the operator wants to manually input a PRA area they need to get the Hexadecimal value for each. Different vendors/operators should interact or exchange PRA info using the Hexadecimal representation as defined in section 8.108 of TS 29.274[9].

The manual input format for a single PRA is:

```
PRA identifier [, PRA element list]
```

- **Default**

The PRA to which the user equipment is already subscribed, if any.

The Default option specifies using the default PRA area. The default PRA is the PRA area subscribed or provisioned by the PCRF for the UE during IP-CAN session life cycle. It is either a UE- dedicated or predefined PRA area. It can be a PRA area that is retrieved from the subscriber profile (normally UE-dedicated) or a PRA area defined in the CMP (normally predefined). When used , this Default option means to check whether the UE has subscribed to a PRA area but does not care what the PRA area is.

### Description

Triggers a policy that is only evaluated when the user equipment is or is not inside the subscribed PRA.

where the APN *matches one of specified APN value(s)*

### Syntax

where the APN *matches-op match-list*

The following screen capture shows a sample policy condition:



### Parameters

#### *matches-op*

One of the following:

- **matches one of** (default)

- **does not match any of**

***match-list***

A comma-separated list of values, where each value is a wildcard match pattern that uses the \* (asterisk) character to match zero or more characters and the ? (question mark) character to match exactly one character.

**Description**

Triggers a policy that is only evaluated for one or more specific access point name (APN) values (based on matching wildcard patterns). A valid APN value is any domain name; for example: `network.operator.com`.

## Network Device Conditions

Network Device conditions are related to the specific network device for which the policy rule is being evaluated. This includes conditions based on the network device type, as well as those that refer to specific unique identifiers for network devices.

### where the device type is specified type

**Syntax**

where the device type `operator-binary device-type`

**Parameters**

***operator-binary***

One of the following:

- **is** (default)
- **is not**

***device-type***

One or more of the following:

- **PDSN**
- **GGSN**
- **HomeAgent**
- **HSGW**
- **PGW**
- **SGW**
- **DPI**

**Description**

Triggers a policy based on the device type for which it is evaluated.

where the User Equipment IMEISV matches one of specified IMEISV value(s)

### Syntax

where the User Equipment IMEISV *matches-op match-list*  
The following screen capture shows a sample policy condition:



### Parameters

#### ***matches-op***

One of the following:

- **matches one of** (default)
- **does not match any of**

#### ***matches-op***

One of the following:

- **matches one of** (default)
- **does not match any of**

### Description

Triggers a policy that is only evaluated for one or more specific IMEISV values (based on matching wildcard patterns). A valid IMEISV value has 16 decimal digits. If the IMEISV value matches with IMEISV value present in event, ensession and then session then the condition returns true, otherwise false.

## Policy SDP Properties Conditions

Session Description Protocol (SDP) properties conditions identify any specific SDP attributes and evaluate their value. This includes setting proper bandwidth values on related PCC rules. The following conditions are available.

where the *local* codec data is an *offer*

### Syntax

where the *sdp\_capabilities* codec data is an *codec-type*

### Parameters

#### ***sdp\_capabilities***

Specifies where to search for the SDP property:

- **Local**—The capabilities of the device for the subscriber.
- **Remote**—The capabilities of the device for the remote party.
- **Common**—The capabilities that the local and remote devices have in common.

**codec-type**

Specifies the Codec type. The options are:

- **offer** (default)
- **answer**

**Description**

Checks the Codec type (offer or answer) for a subscribers device (remote, local or both).

## where the local specified SDP property exists

**Syntax**

where the *sdp\_capabilities* SDP property accessibility

**Parameters****sdp\_capabilities**

Specifies where to search for the SDP property:

- **Local**—The capabilities of the device for the subscriber.
- **Remote**—The capabilities of the device for the remote party.
- **Common**—The capabilities that the local and remote devices have in common.

**SDP property**

A comma-delimited list of SDP properties. Specify the SDP properties using one of the following methods:

- Generic descriptor

`sdp.[option]`

Where:

**option**

Is any name (for example, i) or any keyword (for example, a=ptime)

Examples using an SDP generic descriptor:

- where the common `sdp.[a]` exists
- where the remote `sdp.[a=ptime]` exists
- where the common `sdp.[gd]` exists

- Media descriptor

`sdp.[m.option]`

Where:

**option**

- fmt
- port
- numberofports
- media
- proto

Examples using an SDP media descriptor:

- where the local `sdp.[m]` exists

- **rtpmap**

`sdp.[codec-name(codec-name).rtpmap.OPTION]`

Where:

**codec-name**

Specifies a codec name.

**option**

- payloadtype
- clockrate
- encodingparameters

- **fntp**

`sdp.[codec-name(codec-name).fntp.OPTIONS]`

Where:

**codec-name**

Specifies a codec name.

**option**

- fmt
- profile-level-id
- mode-set
- packetization-mode
- Any other parameter to be conveyed

Examples using fntp:

- where the common `sdp.[codec-name(AMR-WB).fntp.fmt]` exists

**accessibility**

One of the following:

- **exists** (default)
- **does not exist**

**Description**

Checks for the existence or non-existence of any SDP property.

where the local specified SDP property is numerically equal to value

### Syntax

where the *sdp\_capabilities sdp\_capabilities* is numerically *operator value*

### Parameters

#### **sdp\_capabilities**

Specifies where to search for the SDP property:

- **Local**—The capabilities of the device for the subscriber.
- **Remote**—The capabilities of the device for the remote party.
- **Common**—The capabilities that the local and remote devices have in common.

#### **SDP property**

A comma-delimited list of SDP properties. Specify the SDP properties using one of the following methods:

- **Generic descriptor**

**Syntax:** `sdp.[option]`

Where:

#### **option**

Is any name (for example, `i`) or any keyword (for example, `a=ptime`)

Examples using an SDP generic descriptor:

- where the common `sdp.[a=ptime]` is numerically equal to 20
- where the common `sdp.[f=hello]` is numerically equal to 20

- **Media descriptor**

**Syntax:** `sdp.[m.option]`

Where:

#### **option**

- `fmt`
- `port`
- `numberofports`
- `media`
- `proto`

Example using an SDP media descriptor:

- where the local `sdp.[m.numberofports]` is numerically equal to 2

- **rtpmap**

**Syntax:** `sdp.[codec-name(codec-name).rtpmap.OPTION]`



Where:

**codec-name**

Specifies a codec name.

**option**

- payloadtype
- clockrate
- encodingparameters

Examples using rtpmap:

- where the local `sdp.[codec-name(AMR-WB).rtpmap.clockrate]` is numerically less than or equal to 16000

- **fntp**

**Syntax:** `sdp.[codec-name(codec-name).fntp.OPTIONS]`

Where:

**codec-name**

Specifies a codec name.

**option**

- fnt
- profile-level-id
- mode-set
- packetization-mode
- Any other parameter to be conveyed

Example using fntp:

- where the local `sdp.[codec-name(AMR-WB).fntp.mode-set]` is numerically less than or equal to 4

**operator**

One of the following:

- **greater than or equal to**
- **greater than**
- **less than or equal to**
- **less than**
- **equal to**
- **not equal to**

For this condition the default is **equal to**.

**value**

String.

**Description**

Compares a numerical SDP property value against a specified number.

where the local specified SDP property matches one of value(s)

**Syntax**

where the *sdp\_capabilities* SDP property matches-op value-list

**Parameters*****sdp\_capabilities***

Specifies where to search for the SDP property:

- **Local**—The capabilities of the device for the subscriber.
- **Remote**—The capabilities of the device for the remote party.
- **Common**—The capabilities that the local and remote devices have in common.

***SDP property***

A comma-delimited list of SDP properties. Specify the SDP properties using one of the following methods:

- **Generic descriptor**

**Syntax:** *sdp.[option]*

Where:

***option***

Is any name (for example, i) or any keyword (for example, a=ptime)

Examples using an SDP generic descriptor:

- where the local *sdp.[i]* matches one of *\*recvonly\**
- where the common *sdp.[a=ptime]* matches one of 20
- where the common *sdp.[a]* matches one of *ptime: 20*
- where the common *sdp.[u]* matches one of *http://www.oracle.com:8080/hr/one.htm*
- where the common *sdp.[u=http://www.oracle.com]* matches one of *8080/hr/one.htm*
- where the common *sdp.[u=http]* matches one of *//www.oracle.com:8080/hr/one.htm*
- where the remote *sdp.[xy]* matches one of *z*
- where the remote *sdp.[xy=z]* matches one of 80

- **Media descriptor**

**Syntax:** *sdp.[m.option]*

Where:

**option**

- fmt
- port
- numberofports
- media
- proto

Examples using an SDP media descriptor:

- where the common `sdp.[m.fmt]` matches one of 102
- where the common `sdp.[m.port]` does not match any of 41000,41002
- where the remote `sdp.[m.media]` matches one of audio,video
- where the local `sdp.[m.proto]` matches one of RTP/AVP

- **rtpmap**

**Syntax:** `sdp.[codec-name(codec-name).rtpmap.OPTION]`

Where:

**codec-name**

Specifies a codec name.

**option**

- payloadtype
- clockrate
- encodingparameters

Examples using rtpmap:

- where the common `sdp.[ codec-name(AMR-WB).rtpmap]` matches one of 104 AMR-WB/160000
- where the common `sdp.[ codec-name(AMR-WB).rtpmap.encodingparameters]` matches one of 2
- where the common `sdp.[ codec-name(AMR-WB).rtpmap.payloadtype]` matches one of 104,102

- **fntp**

**Syntax:** `sdp.[codec-name(codec-name).fntp.OPTIONS]`

Where:

**codec-name**

Specifies a codec name.

**option**

- fmt
- profile-level-id

- mode-set
- packetization-mode
- Any other parameter to be conveyed

Examples using fmp:

- where the common `sdp.[codec-name(AMR-WB).fmp.fmt]` matches one of 104,102
- where the common `sdp.[codec-name(AMR-WB).fmp.mode-set]` matches one of 2,4
- where the common `sdp.[codec-name(H264).fmp.profile-level-id]` matches one of 42e00c

#### **matches-op**

One of the following:

- **matches one of** (default)
- **does not match any of**

#### **value-list**

A comma-delimited list of values to compare against.

#### **Description**

Checks the Codec type (offer or answer) for a subscribers device (remote, local or both) for specific values.

## Request Conditions

Request conditions are based on information that is explicitly contained within, or related to, the protocol message (request) that triggered the policy rule execution.

where the requested QCI is *specified QCI*

#### **Mode**

Cable, Wireless

#### **Syntax**

where the requested QCI is create list with *specified QCI*  
The following screen capture displays the sample policy condition:



#### **Parameters**

##### ***specified QCI***

One or more of the following:

- **1** (Conversational speech)
- **2** (Conversational)
- **3** (Streaming speech)
- **4** (Streaming)
- **5** (Interactive with priority 1 signalling)
- **6** (Interactive with priority 1)
- **7** (Interactive with priority 2)
- **8** (Interactive with priority 3)
- **9** (Background)
- **65** (Mission critical push-to-talk voice)
- **66** (Push-to-talk voice)
- **69** (Mission critical push-to-talk signalling)
- **70** (Mission critical data)

#### Description

Selects protocol messages based on the QoS class identifier (QCI).

where the request *supports* feature *name*

#### Syntax

where the request *supports* feature *value-list*

The following screen capture shows a sample policy condition:



#### Parameters

##### ***supports***

One of the following:

- **supports** (default)
- **does not support**

##### ***value-list***

A comma-delimited list of values to compare against.

#### Description

Determines whether the request supports a specified feature.

## where the session is *an enforcement session*

### Syntax

where the session is *session-type*

The following screen capture shows a sample policy condition:



### Parameters

#### ***session-type***

One of the following:

- **an enforcement session** (default)
- **an application session**
- **a credit control session**
- **a radius authorization session**

### Description

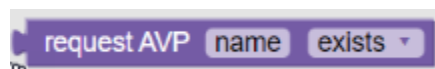
Distinguishes between protocol messages that are operating on different sessions.

## where the request AVP *name exists*

### Syntax

where the request AVP *avp accessibility*

The following screen capture shows a sample policy condition:



### Parameters

#### ***avp***

AVP in one of the following formats:

*name:vendorID*

or a full path

*[avp\_name1]:vendorID.[avp\_name2]:vendorID...*

for the members of the grouped AVPs

**accessibility**

One of the following:

- **exists** (default)
- **does not exist**

**Description**

Checks for the presence or absence of the third-party AVP in an incoming Diameter message.

 **Note:**

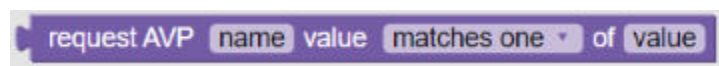
The condition supports both loaded base Diameter AVPs and third-party AVPs.

where the request AVP *name* value matches one of value(s)

**Syntax**

where the request AVP *avp matches-op value-list*

The following screen capture shows a sample policy condition:

**Parameters****avp**

AVP in one of the following formats:

*name*:*vendorID*

or a full path

[*avp\_name1*]:*vendorID*. [*avp\_name2*]:*vendorID*...

for the members of the grouped AVPs

**matches-op**

One of the following:

- **matches one of** (default)
- **does not match any of**

**value-list**

A comma-delimited list of values to compare against.

**Description**

Compares the specified AVP value with the values or variables from the specified list. The condition is where the request AVP name value matches one of the values. The

values can be evaluated for equality as well as inequality. To evaluate an AVP value for inequality, the variable **matches one of** must be changed to **does not match any of**.

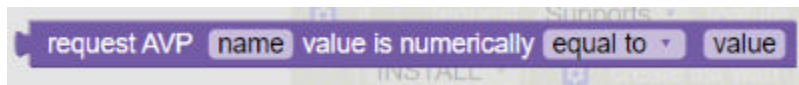
**Note:**

The condition supports both loaded base Diameter AVPs and third-party AVPs.

where the request AVP *name* value is numerically *equal to* value

**Syntax**

where the request AVP *avp* value is numerically *operator value*  
The following screen capture shows a sample policy condition:

**Parameters****avp**

AVP in one of the following formats:

*name*:*vendorID*

or a full path

[*avp\_name1*]:*vendorID*. [*avp\_name2*]:*vendorID*...

for the members of the grouped AVPs

**operator**

One of the following:

- **greater than or equal to**
- **greater than**
- **less than or equal to**
- **less than**
- **equal to**
- **not equal to**

The default for this condition is **equal to**.

**value**

String.



### Description

Compares a numerical AVP value against a specified number or policy context number variable value.

#### Note:

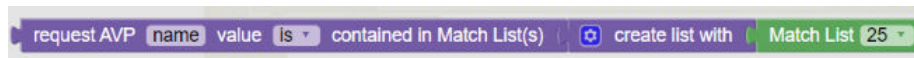
The condition supports both loaded base Diameter AVPs and third-party AVPs.

where the request AVP *name* value *is* contained in Match Lists *select lists*

### Syntax

where the request AVP *avp* value *operator-binary* contained in Match Lists *match-list*

The following screen capture shows a sample policy condition:



### Parameters

#### **avp**

AVP in one of the following formats:

*name*:*vendorID*

or a full path

[*avp\_name1*]:*vendorID*. [*avp\_name2*]:*vendorID*..

for the members of the grouped AVPs

#### **operator-binary**

One of the following:

- **is** (default)
- **is not**

#### **match-list**

A comma-separated list of values, where each value is a wildcard match pattern that uses the \* (asterisk) character to match zero or more characters and the ? (question mark) character to match exactly one character.

### Description

Compares the specified AVP value with the values or variables from the specified match list. The condition is where the request AVP name value matches one of the values. The values can be evaluated for equality as well as inequality. To evaluate an

AVP value for inequality, the condition **matches one of** must be changed to **does not match any of**.



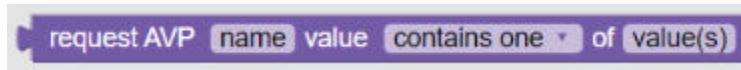
#### Note:

The condition supports both loaded base Diameter AVPs and third-party AVPs.

where the request AVP *name* value *contains one of value(s)*

#### Syntax

where the request AVP *avp* value *containment* *value-list*  
The following screen capture shows a sample policy condition:



#### Parameters

##### **avp**

AVP in the format:

*name*:*vendorID*

or a full path

[*avp\_name1*]:*vendorID*. [*avp\_name2*]:*vendorID*..

for the members of the grouped AVPs

##### **containment**

One of the following:

- **contains one of** (default)
- **does not contain any of**

##### **value-list**

A comma-delimited list of values to compare against.

#### Description

Performs a lookup of the sub-strings in the AVP value. It is possible to check multiple sub-string entries at on time. If the operation type is changed, you can check the opposite scenario, which would not include any of the provided sub-strings.



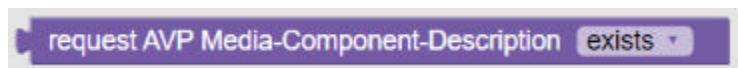
#### Note:

The condition supports both loaded base Diameter AVPs and third-party AVPs.

## where the request AVP Media-Component-Description exists

### Syntax

where the request AVP Media-Component-Description *accessibility*  
The following screen capture shows a sample policy condition:



### Parameters

#### *accessibility*

One of the following:

- **exists** (default)
- **does not exist**

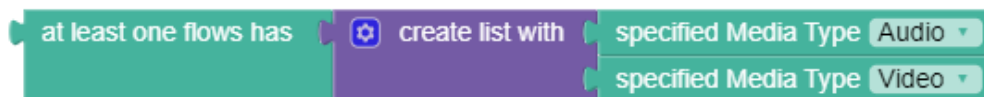
### Description

Determines whether the AVP Media-Component-Description is accessible.

## where at least one flow has media type that matches *specified type(s)*

### Syntax

where at least one flow has media type that matches *media-type*  
The following screen capture shows a sample policy condition:



### Parameters

#### *media-type*

One or more of the following, used to determine the type of media:

- **Audio**
- **Video**
- **Data**
- **Application**
- **Control**
- **Text**
- **Message**
- **Other**

**Description**

Triggers a policy based on whether at least one flow matches one or more of the specified media types.

**Example 5-2 Example**

where at least one flow has media type that matches *Video,Application*

where the AF-Application-ID matches one of *specified value(s)*

**Syntax**

where the AF-Application-ID matches one of *value-list*  
The following screen capture shows a sample policy condition:

A screenshot of a policy condition in a configuration interface. The text reads "AF-Application-Id matches one of afaudio". The text is highlighted in a green box.

**Parameters*****value-list***

A comma-delimited list of values to compare against.

**Description**

Selects protocol messages based on the Diameter AF Application Identifier field. A valid AF Application identifier is any string describing the application, for example VoIP or streaming.

**Example 5-3 Example**

where the AF-Application-ID matches one of *ptt-application-id*  
apply *PTT* to all flows in the request  
continue processing message

where the corresponding enforcement session *supports* feature *name*

**Syntax**

where the corresponding enforcement session *action* feature *value-list*  
The following screen capture shows a sample policy condition:

A screenshot of a policy condition in a configuration interface. The text reads "the corresponding enforcement session Supports feature name". The text is highlighted in a green box.

**Parameters*****action***

Select one of the following :

- **supports** (default)
- **does not support**

**value-list**

A comma-delimited list of values to compare against.

This list can contain one or more supported feature. To use a wildcard match pattern, select **Evaluate as Expression**. Wildcard match patterns use the following characters:

- \* (asterisk) character to match zero or more characters
- ? (question mark) character to match exactly one character

**Description**

Evaluates the feature name in the enforcement session that correlates to the corresponding application (Rx) request.

**Example 5-4 Example**

where the corresponding enforcement session *supports* feature *GroupComService*.

where the flow media type is one of *specified type(s)*

**Syntax**

where the flow(s) media type is one of *media-type*  
The following screen capture shows a sample policy condition:

**Parameters****media-type**

One or more of the following, used to determine the type of media:

- **Audio**
- **Video**
- **Data**
- **Application**
- **Control**
- **Text**
- **Message**
- **Other**

**Description**

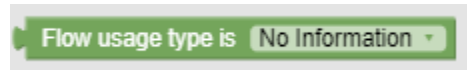
Selects protocol messages based on the media type of the flow or flows.

where the flow usage is one of *specified usage(s)*

#### Syntax

where the flow usage is *flow-usage-type*

The following screen capture shows a sample policy condition:



#### Parameters

##### ***flow-usage-type***

One or more of the following:

- **No Information**
- **RTCP**
- **AF Signaling**

#### Description

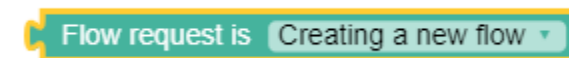
Selects protocol messages based on the flow usage.

where the request is *creating a new flow*

#### Syntax

where the request is *change-type*

The following screen capture shows a sample policy condition:



#### Parameters

##### ***change-type***

One or more of the following:

- **creating a new flow** (default)
- **modifying an existing flow**
- **provisioning a default flow**
- **terminating an existing flow**

#### Description

Distinguishes between protocol messages based on the type of operation being performed on the flow.

where the Service-URN is one of *specified value(s)*

### Syntax

where the Service-URN is one of *value-list*

The following screen capture shows a sample policy condition:

A screenshot of a policy condition. The text "the Service-URN is one of specified value(s)" is highlighted in a green box with a yellow border.

### Parameters

#### *value-list*

A comma-delimited list of values to compare against.

#### Description

Selects Rx protocol messages based on the value of the Service-URN field.

where the specific action is one of *specified action(s)*

### Syntax

where the specific action is one of *action*

The following screen capture shows a sample policy condition:

A screenshot of a policy condition. The text "The specific action is" is highlighted in a green box. To its right, there is a purple box containing "create list with" and two blue boxes, each containing "specified action" followed by a dropdown menu. The first dropdown menu shows "ACCESS\_NETWORK\_INFO\_REPORT" and the second shows "CHARGING\_CORRELATION\_EXCHANGE".

### Parameters

#### *action*

One or more of the following actions:

- SERVICE\_INFORMATION\_REQUEST
- CHARGING\_CORRELATION\_EXCHANGE
- INDICATION\_OF\_LOSS\_OF\_BEARER
- INDICATION\_OF\_RECOVERY\_OF\_BEARER
- INDICATION\_OF\_RELEASE\_OF\_BEARER
- INDICATION\_OF\_ESTABLISHMENT\_OF\_BEARER
- INDICATION\_OF\_IP\_CAN\_CHANGE
- INDICATION\_OF\_OUT\_OF\_CREDIT
- INDICATION\_OF\_SUCCESSFUL\_RESOURCES\_ALLOCATION
- INDICATION\_OF\_FAILED\_RESOURCES\_ALLOCATION
- USAGE\_REPORT

- **ACCESS\_NETWORK\_INFO\_REPORT**
- **INDICATION\_OF\_RECOVERY\_FROM\_LIMITED\_PCC\_DEPLOYMENT**
- **INDICATION\_OF\_ACCESS\_NETWORK\_INFO\_REPORTING\_FAILURE**

#### Description

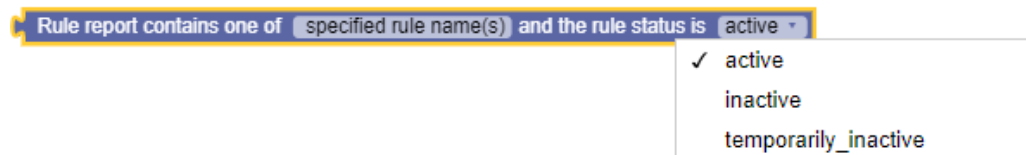
This condition lets you take action based on the value of the Specific-Action AVP field within an Rx RAA message.

where the rule report contains one of *specified rule name(s)* and the rule status is *active*

#### Syntax

where the rule report contains one of *value-list* and the rule status is *field*

The following screen capture shows a sample policy condition:



#### Parameters

##### ***value-list***

A comma-delimited list of values to compare against.

##### ***field***

One of the following:

- **active** (default)
- **inactive**
- **temporarily\_inactive**

#### Description

Selects protocol messages based on whether a rule name and a status was received in a rule report.

where the request MPS Identifier *matches one of value(s)*

#### Syntax

where the MPS Identifier *matches-op value-list*

The following screen capture shows a sample policy condition:





### Parameters

#### ***matches-op***

One of the following:

- **matches one of** (default)
- **does not match any of**

#### ***value-list***

A comma-delimited list of values to compare against.

### Description

Determines whether the MPS Identifier matches a specified value(s).

where the requested media component description reservation priority is one of *specified*

### Syntax

where the requested media component description reservation priority is one of *priority*

The following screen capture shows a sample policy condition:



### Parameters

#### ***priority***

One or more of the following:

- **DEFAULT**
- **PRIORITY\_ONE**
- **PRIORITY\_TWO**
- **PRIORITY\_THREE**
- **PRIORITY\_FOUR**
- **PRIORITY\_FIVE**
- **PRIORITY\_SIX**
- **PRIORITY\_SEVEN**
- **PRIORITY\_EIGHT**
- **PRIORITY\_NINE**
- **PRIORITY\_TEN**
- **PRIORITY\_ELEVEN**
- **PRIORITY\_TWELVE**
- **PRIORITY\_THIRTEEN**

- PRIORITY\_FOURTEEN
- PRIORITY\_FIFTEEN

#### Description

Selects Rx protocol messages based on the requested media component description reservation priority.

where the requested session reservation priority is one of *specified*

#### Syntax

where the requested session reservation priority is one of *priority*  
The following screen capture shows a sample policy condition:



#### Parameters

##### ***priority***

One or more of the following:

- DEFAULT
- PRIORITY\_ONE
- PRIORITY\_TWO
- PRIORITY\_THREE
- PRIORITY\_FOUR
- PRIORITY\_FIVE
- PRIORITY\_SIX
- PRIORITY\_SEVEN
- PRIORITY\_EIGHT
- PRIORITY\_NINE
- PRIORITY\_TEN
- PRIORITY\_ELEVEN
- PRIORITY\_TWELVE
- PRIORITY\_THIRTEEN
- PRIORITY\_FOURTEEN
- PRIORITY\_FIFTEEN

#### Description

Selects Rx protocol messages based on the requested session reservation priority.

where the flow media type *matches one of user defined media type(s)*

### Syntax

where the flow media type *matches-op value-list*

The following screen capture shows a sample policy condition:



### Parameters

#### *matches-op*

One of the following:

- **matches one of** (default)
- **does not match any of**

#### *value-list*

A comma-delimited list of values to compare against.

### Description

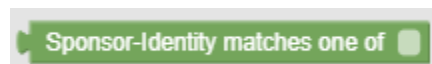
Selects one or more protocol messages that match one or more user-defined media types.

where the Sponsor-Identity matches one of *specified Sponsor Identity(s)*

### Syntax

where the Sponsor-Identity matches one of *value-list*

The following screen capture shows a sample policy condition:



### Parameters

#### *value-list*

A comma-delimited list of values to compare against.

### Description

Selects protocol messages based on whether the Sponsored-Identity AVP matches a list of sponsors. This condition supports sponsored data connectivity.

### Example 5-5 Example

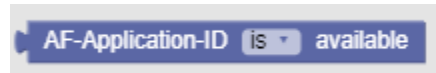
The following condition is true if the Sponsored-Identity AVP matches either ESPN or FIFA:

where the Sponsor-Identity matches one of *ESPN,FIFA*

## where the AF-Application-ID is available

### Syntax

where the AF-Application-ID *operator-binary* available  
The following screen capture shows a sample policy condition:



### Parameters

#### ***operator-binary***

One of the following:

- **is** (default)
- **is not**

### Description

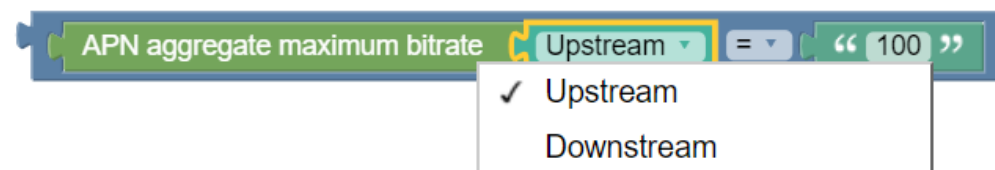
Checks for the presence or absence of the AF Application Identifier field. A valid AF Application identifier is any string describing the application, for example VoIP or streaming.

## where the requested APN aggregate maximum bitrate *upstream* is *greater than* # bps

### Syntax

where the requested APN aggregate maximum bitrate *flow-direction* is *operator bandwidth*

The following screen capture displays the sample policy condition:



### Parameters

#### ***flow-direction***

One of the following:

- **upstream**
- **downstream**
- **upstream or downstream** (default)

#### ***operator***

One of the following:

- **greater than or equal to**
- **greater than**
- **less than or equal to**
- **less than**
- **equal to**
- **not equal to**

The default for this condition is **greater than**.

*bandwidth*

A numeric value that specifies the bandwidth. The unit of bandwidth is compatible with the Credit Control Request (CCR) message.

#### Description

Selects protocol messages based on the maximum bitrate being requested for an access point name (APN) in a specific direction relative to a numeric value.

## Time of Day Conditions

Time-of-Day conditions are related to the time at which the policy rules are being executed.

#### Configuring Local Time

To Configure the local time:

1. From the navigation menu, under **PCRF**, then under **Services**, click **Core Service**.  
The Core Service screen appears.
2. Click **Edit** to edit the core service configurations.
3. In **Advance Settings**, click **Create**.  
The Create page appears.
4. Enter **DB.User.DefaultLocalTimeMode** in the **Key** field.
5. Enter **True** in the **Value** field.
6. Click **Save**.

If no configuration is provided, the **SYSTEM\_LOCAL\_TIME** is considered as default local time.

where the current time *is* between *start time* and *end time* using *configured local time*

#### Syntax

where the current time *operator-binary* between *time-of-day* and *time-of-day* using *time-zone*

The following screen capture shows a sample policy condition:

the current time **is** between start time **hh:mm** and end time **hh:mm** using **CONFIGURED\_LOCAL\_TIME**

### Parameters

#### **operator-binary**

One of the following:

- **is** (default)
- **is not**

#### **time-of-day**

A time, in the format of *hh:mm*, where *hh* is a number in the range from 0 to 23.

#### **time-zone**

One of the following:

- **CONFIGURED LOCAL TIME** (default)—Calculate the time from the location configured for this MPE device
- **SYSTEM LOCAL TIME**—Calculate the time from the location of this MPE device
- **USER LOCAL TIME**—Calculate the time from the location configured for the user equipment's location

### Description

Triggers a policy based on time. If the present time is between start time and end time then the condition returns true, otherwise false. If start time is greater than end time then the condition is evaluated, where the end time is considered as the next day.

where the current time *is* within the *specified* time period(s)

### Syntax

where the current time *operator-binary* within the *time-period* time periods  
The following screen capture shows a sample policy condition

the current time **is** within the **create list with** Time Period **one** period(s)

### Parameters

#### **operator-binary**

One of the following:

- **is** (default)
- **is not**

#### **time-period**

Names of one or more time periods that are defined in the cnPCRF GUI.

You can configure Time Periods using GUI by Navigating **PCRF** → **Configuration** → **Time Periods**.

### Description

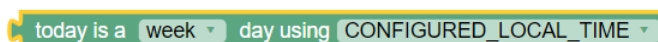
Triggers a policy based on the time period. This condition gets time slots of all the time periods, and compares current time with these time slots. If the current time falls within the range of time slots configured in these time periods then the condition returns true, otherwise false.

where today is a week day using *configured local time*

### Syntax

where today is a week day using *time-zone*

The following screen capture shows a sample policy condition:

A screenshot of a policy condition configuration. The text 'today is a week day using CONFIGURED\_LOCAL\_TIME' is displayed in a light blue box with a yellow border. The words 'week' and 'CONFIGURED\_LOCAL\_TIME' are each followed by a small downward-pointing triangle, indicating they are dropdown menus.

### Parameters

#### *time-zone*

One of the following:

- **CONFIGURED LOCAL TIME** (default)—Calculate the time from the location configured for this MPE device
- **SYSTEM LOCAL TIME**—Calculate the time from the location of this MPE device
- **USER LOCAL TIME**—Calculate the time from the location configured for the user equipment's location

### Description

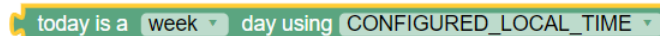
Triggers a policy based on the day of the week. If today is week day using the system time then the condition returns true, otherwise false.

where today is a weekend day using *configured local time*

### Syntax

where today is a weekend day using *time-zone*

The following screen capture shows a sample policy condition:

A screenshot of a policy condition configuration. The text 'today is a week day using CONFIGURED\_LOCAL\_TIME' is displayed in a light blue box with a yellow border. The words 'week' and 'CONFIGURED\_LOCAL\_TIME' are each followed by a small downward-pointing triangle, indicating they are dropdown menus.

### Parameters

#### *time-zone*

One of the following:

- **CONFIGURED LOCAL TIME** (default)—Calculate the time from the location configured for this MPE device
- **SYSTEM LOCAL TIME**—Calculate the time from the location of this MPE device
- **USER LOCAL TIME**—Calculate the time from the location configured for the user equipment's location

### Description

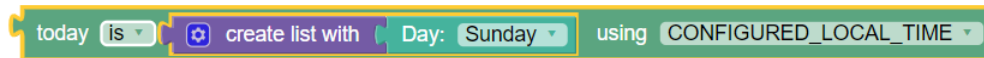
Triggers a policy based on the day of the week. If today is weekend according to configured time then the condition returns true, otherwise false.

where today *is* day using *configured local time*

### Syntax

where today *operator-binary* *day-of-week* using *time-zone*

The following screen capture shows a sample policy condition:



### Parameters

#### ***operator-binary***

One of the following:

- **is** (default)
- **is not**

#### ***day-of-week***

One of the following:

- **Sunday**
- **Monday**
- **Tuesday**
- **Wednesday**
- **Thursday**
- **Friday**
- **Saturday**

#### ***time-zone***

One of the following:

- **CONFIGURED LOCAL TIME** (default)—Calculate the time from the location configured for this MPE device
- **SYSTEM LOCAL TIME**—Calculate the time from the location of this MPE device
- **USER LOCAL TIME**—Calculate the time from the location configured for the user equipment's location



## Description

Triggers a policy based on the day of the week. If today is day within specified days then the condition returns true, otherwise false.

where *today* is the *specified number(s)* th day(s) of *Any Month* in *natural order* using *configured local time*

## Syntax

where *today operator-binary* the *value-list* th days of *month* in *order* using *time-zone*

The following screen capture shows a sample policy condition:



```
today is the delimited days day(s) of create list with Month January in natural order using CONFIGURED_LOCAL_TIME
```

## Parameters

### ***operator-binary***

One of the following:

- **is** (default)
- **is not**

### ***value-list***

A comma-delimited list of values to compare against.

### ***month***

One or more of the following:

- **January**
- **February**
- **March**
- **April**
- **May**
- **June**
- **July**
- **August**
- **September**
- **October**
- **November**
- **December**

### ***order***

Specifies the order to evaluate the value list. The options are:

- **natural order**

- **reverse order**

***time-zone***

One of the following:

- **CONFIGURED LOCAL TIME** (default)—Calculate the time from the location configured for this MPE device
- **SYSTEM LOCAL TIME**—Calculate the time from the location of this MPE device
- **USER LOCAL TIME**—Calculate the time from the location configured for the user equipment's location

**Description**

Triggers a policy based on a day in a month. If current date matches specified number th day of specified months in natural/reverse order as per the configured time then the condition returns true, otherwise false.

**Example 5-6 Example**

The following conditions, if evaluated as true, will trigger a policy:

*where today is the 1,2,3,4 th days of March, April, May in natural order using USER LOCAL TIME*

# 6

## Actions for Writing Policy Rules

The policy wizard supports a large number of actions that can be used for constructing policy rules. There are two types of policy-processing actions:

### Mandatory actions

This action defines what happens when the current policy is through executing. When you are creating a policy rule in the policy wizard, these actions are displayed at the top of the list of available actions with a radio button that forces you to select only one of these actions.

### Optional actions

These are actions executed when the conditions in the policy rule have been met. When you are creating a policy rule in the policy wizard, this is a list of actions that you can add to your policy rule. You can select none, one, several, or up to 40 of these optional actions per rule. However, each action is limited, so that it can be executed only once per policy rule.

In the same way that you can customize conditions by editing parameter values, many of these actions can be customized by specifying parameter values as well.

Actions are listed in alphabetical order. Actions also are affected by the current mode. Therefore, some of the actions documented may not be available in your policy wizard.

## add custom grouped AVP *name* and send *always*

### Syntax

```
add custom grouped AVP name and send send-mode
```

### Parameters

#### ***name***

An existing AVP name and Vendor ID.

#### ***send-mode***

One of the following:

- **always** (default)
- **unless rejected**
- **if rejected**

### Description

Add or send new custom grouped AVP to the current reply. A condition can be set specifying that the AVP is always set to send mode. If you are defining a new grouped third party AVP with members, the grouped AVP has to appear first in the policy. If you are adding a new member AVP that does not have its parent AVP added yet, the policy attempts to locate this grouped AVP in the rest of the policy. If you are including

a grouped AVP multiple times in the same message, you have to follow the order in which it appears in the message.

## set value to Existing or New custom AVP name and send always

### Syntax

```
set value to avp-type custom AVP avp-name and send send-mode
```

### Parameters

#### **value**

String.

This string represents a third-party non-grouped AVP. Check **Evaluate as expression** to evaluate this value as an expression.

#### **avp-type**

Select type of AVP name:

- **Existing** (default)
- **New**

#### **avp-name**

An existing AVP Name and Vender ID.

#### **send-mode**

One of the following:

- **always** (default)
- **unless rejected**
- **if rejected**

### Description

Adds the third-party non-grouped AVP to the current Diameter session with the specified value. If a third-party AVP value is set in the current Diameter session, it will be sent with the corresponding outgoing message. The value parameter must corresponds to the AVP data type, otherwise this AVP will not be set. If New is selected as the type of AVP name, every time this action is called a new AVP is added to the message, even if the AVP with the same name is present in the message.

## set custom AVP name value to the policy context property name

### Syntax

```
set custom AVP avp-name value to the policy context property property-name
```

### Parameters

***avp-name***

An existing AVP name and Vendor ID.

***property-name***

String that represents the policy context property.

### Description

Makes the AVP value accessible throughout the policy context so other policies can access this AVP value as a context property. The context property variable will be set only if this AVP exists in the request and its value is not null.

## remove custom AVP *name* from reply *always*

### Syntax

```
remove custom AVP name from reply send-mode
```

### Parameters

***name***

An existing AVP name and Vendor ID.

***send-mode***

One of the following:

- **always** (default)
- **unless rejected**
- **if rejected**

### Description

Removes the custom AVP name previously set from the reply message.

## mark request AVP *name* as failed if exists and send *always*

### Syntax

```
mark request AVP name as failed if exists and send send-mode
```

### Parameters

***name***

String representing existing AVP name, entered in the format *AVPname:VendorID* or, for nested AVP names in an AVP group, entered in the format [*AVPname1*]:*VendorID*. [*AVPname2*]:*VendorID* ... for the members of the grouped AVPs. There is also the option to evaluate as an expression (click to select check box).

***send-mode***

One of the following:

- **always** (default)
- **unless rejected**
- **if rejected**

#### Description

Marks a request AVP as failed in the reply message, and notifies the opposite peer of the failed AVP validation. This action supports both loaded base Diameter AVPs and third-party AVPs.

## set the user property *name* to *Existing* or *New* custom AVP *name* and send *always*

#### Syntax

set the user property *property-name* to *property-type* custom AVP *avp-name* and send *send-mode*

#### Parameters

##### ***property-name***

String.

##### ***property-type***

One of the following:

- **Existing or New** (default)
- **New**

##### ***avp-name***

Select an existing AVP Name and Vender ID.

##### ***send-mode***

One of the following:

- **always** (default)
- **unless rejected**
- **if rejected**

#### Description

Sets the user property value for an outgoing AVP. If a user property with the corresponding name exists, the AVP will be sent in the reply message.

## 7

# Cloud Native Policy and Charging Rule Function Alerts

This section includes information about alerts for CNPCRF.

Alarm Name	Alarm Description	Severity	App/Metrics
PRE_UNREACHABLE	PRE is unreachable	CRITICAL	Metrics
PDS_DOWN	PDS is down	CRITICAL	Metrics
PDS_UP	PDS is up	INFO	Metrics
DB_UNREACHABLE	Connectivity to DB lost	CRITICAL	Metrics
DB_REACHABLE	Connectivity to DB available	INFO	Metrics
SH_UNREACHABLE	Remote Sh connection is unreachable	CRITICAL	App
SY_UNREACHABLE	Remote Sy connection is unreachable	CRITICAL	App
SOAP_CONNECTOR_DOWN	SOAP Connector is down	CRITICAL	Metrics
SOAP_CONNECTOR_UP	SOAP Connector is up	INFO	Metrics
CONFIG_SERVER_DOWN	Config server is down	CRITICAL	Metrics
CONFIG_SERVER_UP	Config server is up	INFO	Metrics
DIAM_GATEWAY_DOWN	Diameter Gateway is down	CRITICAL	Metrics
DIAM_GATEWAY_UP	Diameter Gateway is up	INFO	Metrics
LDAP_GATEWAY_DOWN	LDAP Gateway is down	CRITICAL	Metrics
LDAP_GATEWAY_UP	LDAP Gateway is up	INFO	Metrics
LDAP_DATASOURCE_UNREACHABLE	LDAP Datasource is unreachable	CRITICAL	App
CM_SERVICE_DOWN	CM Service is down	CRITICAL	Metrics
CM_SERVICE_UP	CM Service is up	INFO	Metrics
CCA_SEND_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA Send Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCAI_SEND_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA-I Send Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCAT_SEND_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA-T Send Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCAU_SEND_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA-U Send Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics

Alarm Name	Alarm Description	Severity	App/Metrics
ASA_SEND_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of ASA Send Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
RAA_SEND_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of RAA Send Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
STA_SEND_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of STA Send Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCA_RECV_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA Receive Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCAI_RECV_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA-I Receive Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCAT_RECV_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA-T Receive Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCAU_RECV_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of CCA-U Receive Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
ASA_RECV_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of ASA Receive Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
RAA_RECV_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of RAA Receive Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
STA_RECV_FAIL_COUNT_EXCEEDS_THRESHOLD	Rate of STA Receive Failure has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCR_TIMEOUT_COUNT_EXCEEDS_THRESHOLD	Rate of CCR Timeout count has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCRI_TIMEOUT_COUNT_EXCEEDS_THRESHOLD	Rate of CCR-I Timeout count has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCRT_TIMEOUT_COUNT_EXCEEDS_THRESHOLD	Rate of CCR-T Timeout count has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
CCRU_TIMEOUT_COUNT_EXCEEDS_THRESHOLD	Rate of CCR-U Timeout count has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics



Alarm Name	Alarm Description	Severity	App/Metrics
ASR_TIMEOUT_COUNT_EXCEEDS_THRESHOLD	Rate of ASR Timeout count has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
RAR_TIMEOUT_COUNT_EXCEEDS_THRESHOLD	Rate of RAR Timeout count has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics
STR_TIMEOUT_COUNT_EXCEEDS_THRESHOLD	Rate of STR Timeout count has exceeded threshold limit(1000 times) in 1 min	CRITICAL	Metrics

## PCRF Alert Configuration

This section describes the Measurement based Alert rules configuration for CNPCRF. The Alert Manager uses the Prometheus measurements values as reported by microservices in conditions under alert rules to trigger alerts.

### PCRF Alert Configuration

To configure cnPCRF alerts in Prometheus:

#### Note:

1. The alert manager and prometheus tools should run in the default namespace.
2. The PCRF Templates.zip file can be downloaded from [OHC](#). Unzip the package after downloading to get cnpcrfalertrule.yaml and mib files.

1. Find the config map to configure alerts in prometheus server by executing the following command:

```
kubectl get configmap -n Namespace
```

where, *Namespace* is the namespace used in helm install command.

2. Take Backup of current config map of prometheus server by executing the following command:

```
kubectl get configmaps NAME -o yaml -n Namespace /tmp/t_mapConfig.yaml
```

where, *Name* is the release name used in helm install command.

3. Delete the entry **alertsncprf** under rule\_files, if present, in the Alert Manager config map by executing the following command:

```
sed -i '/etc\/config\/alertsncprf\/d' /tmp/t_mapConfig.yaml
```

 **Note:**

This command should be executed only once.

4. Add entry **alertscnprf** under `rule_files` in the prometheus server config map by executing the following command:

```
sed -i '/rule_files:/a\ \- /etc/config/alertscnprf' /tmp/  
t_mapConfig.yaml
```

 **Note:**

This command should be executed only once.

5. Reload the modified config map by executing the following command:

```
kubectl replace configmap <_NAME_> -f /tmp/t_mapConfig.yaml
```

 **Note:**

This step is not required for AlertRules.

6. Add `cnprfAlertrules` in config map by executing the following command :

```
kubectl patch configmap _NAME_-server -n _Namespace_--type merge --patch  
"$(cat ~/cnprfAlertrules.yaml)"
```