Oracle® Communications

InterWorking and Mediation Function (IWF) Cloud Native User's Guide

Release 1.5

F33290-01

July 2020

Oracle Communications InterWorking and Mediation Function (IWF) Cloud Native User's Guide, Release 1.5

F33290-01

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

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:

1. Select **2** for New Service Request.
2. Select **3** for Hardware, Networking and Solaris Operating System Support.
3. Select one of the following options:
   - 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**.

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.

What's New in This Guide

New and Updated Features in Release 1.5:

This section introduces the documentation updates for Release 1.5 in InterWorking and Mediation Function (IWF) Cloud Native User's Guide.

- Added new Metrics and Alerts
- Updated the following rules:
  - Mediation Rules
  - NF Mediation Rules
  - PT Mediation Rule

List of Figures

List of Tables

1 Introduction

This document provides information for using InterWorking and Mediation Function in the 5G Core Network.

The Oracle Communications 5G InterWorking and Mediation Function is a Cloud-Native solution based on micro-services architecture enabling 5G Core Network Functions (NF) to communicate with EPC network elements.

The InterWorking and Mediation Function is deployed as an independent network function in the 5G core network or as a part of Oracle 5G core NFs, which include Network Repository Function, Security Edge Protection Proxy, and Service Communication Proxy, as independent micro services within the 5G core NF.

Acronyms and Terminologies

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

Table 1-1 Acronyms and Terminologies

Field	Description
3GPP	3rd Generation Partnership Project
5GC	5G Core Network
5GS	5G System
AF	Application Function
AUSF	Authentication Server Function
BSF	Binding Support Function
CHF	Charging Function
CNE	Cloud Native Environment
EFK stack	Elasticsearch, Fluentd, and Kibana stack
FQDN	Fully Qualified Domain Name
GPSI	Generic Public Subscription Identifier
IWF	InterWorking and Mediation Function
K8s	Kubernetes
NEF	Network Exposure Function
NF	Network Function
NRF	Network Repository Function
NSSF	Network Slice Selection Function
PCF	Policy Control Function
PFD	Packet Flow Description
QFI	QoS Flow Identifier
QoE	Quality of Experience
SBA	Service Based Architecture

Table 1-1   (Cont.) Acronyms and Terminologies

Field	Description
SBI	Service Based Interface
SCP	Service Communication Proxy
SEPP	Security Edge Protection Proxy
SMF	Session Management Function
SUPI	Subscription Permanent Identifier
UDR	Unified Data Repository
UDSF	Unstructured Data Storage Function

2
About InterWorking and Mediation Function

The key capabilities of InterWorking and Mediation Function include:

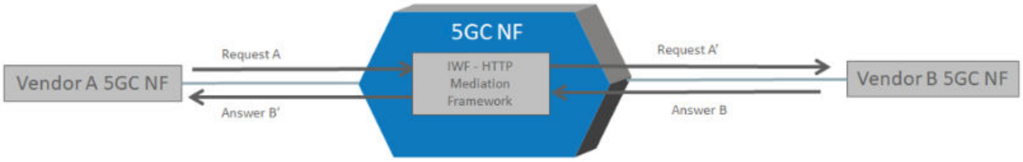
- **Protocol Translation:**
  - Allows 5GC NF to interwork the EPC network elements or vice versa
  - Supports diameter to HTTP/2 and vice versa protocol conversion capabilities
- **Message Mediation:**
  - Allows API transformation to resolve inter-NF inter-operational issues
  - Allows users to create policy rules to execute mediation transformation

About Mediation Capability

Mediation capability:

- Allows user to perform HTTP/2 signaling API message mediation with vendor-specific implementations and resolve interoperability issues for 5GC inter-NF communication.
- Supports dynamic message transformation based on the configured policy rules.

Figure 2-1 Mediation Capability



The IWF mediation feature allows the user to configure the mediation policies that are executed on incoming and outgoing NF messages.

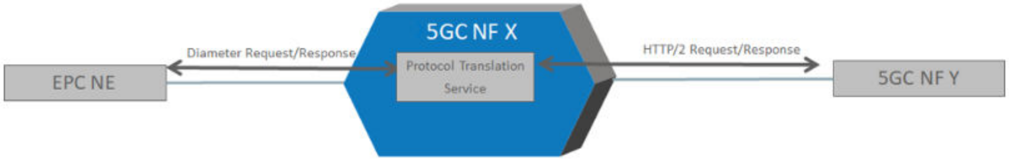
The IWF Mediation supports the following capabilities:

- **Header:** Used to make custom changes to the request and/or response headers (add, remove, and modify).
- **Body IEs:** Used to make custom changes to HTTP message bodies.
- **Method/Status Code:** Used to update the method name or status code in messages.
- **URL Rewriting:** Modify any URLs in the API messages.
- **Transformation:** Used to convert between JSON and XML.
- **Message Actions:** Forward/Reject/Drop/Message Copy the messages based on configured rules.

About Protocol Translation

The Protocol Translation enables inter-working between EPC network elements and 5GC NFs. Perform mapping of HTTP/2 to Diameter messages and vice versa.

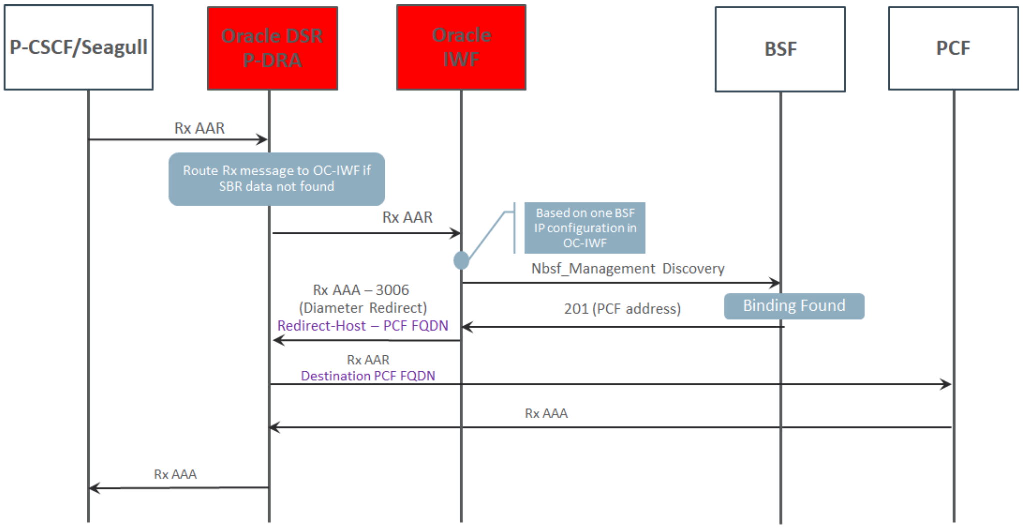
Figure 2-2   Protocol Translation



Protocol translation enables migration and inter-working strategy for an LTE network by performing protocol translation.

- Diameter-HTTP/2 inter-working is based on generic framework using IE mapping configuration and includes built-in support for the following Diameter-HTTP/2 inter-working use cases.
  - Inter-working between AF(CSCF) & PCF for IMS, QoS Control, SDC, and so on, use cases - Rx to N5
  - Inter-working between NF & PCRF for IMS, QoS Control, SDC, and so on, use cases - N7 to Gx
- Supports for HTTP/2 to HTTP/1.1
- Supports payload transformation - JSON to XML
- Supports customization of inter-working IEs of HTTP/2 and diameter messages

<u>PCF Session Binding Discovery success case</u>


P-CSCF/Seagull
Oracle DSR P-DRA
Oracle IWF
BSF
PCF
Rx AAR
Route Rx message to OC-IWF if SBR data not found
Rx AAR
Based on one BSF IP configuration in OC-IWF
Nbsf_Management Discovery
Rx AAA – 3006 (Diameter Redirect) Redirect-Host – PCF FQDN
201 (PCF address)
Binding Found
Rx AAR Destination PCF FQDN
Rx AAA
Rx AAA

3
User Configurations

The user can perform the following configurations in the InterWorking and Mediation Function:

Configuring Mediation Rules

User can prepare their own Mediation rules to apply on HTTP Request and Response. Refer to the following sample rules to create new Mediation Rules.

HTTP Request Header Manipulation Rule

Following is a sample HTTP request header manipulation rule.

```
rule "Http-Req-Header-Manipulation-Rule"
when
    req : Request(headers.has("Header Name"))
then
    req.headers.add("Header Name","Value")
    req.headers.put("Header Name","Value")
    req.headers.set("Header Name","Value")
    req.headers.del("Header Name")
end
```

HTTP Request Body Manipulation Rule

Following is a sample HTTP request body manipulation rule.

```
rule "Http-Req-Body-Manipulation_Rule"
    salience 20
when
    req : Request(body.get("JSONPath") == "value")
then
    req.body.add("JSONPATH","Value")
    req.body.put("JSONPATH","Key","Value")
    req.body.set("JSONPATH","Value")
    req.body.del("JSONPath")
end
```

HTTP Response Header Manipulation Rule

Following is a sample HTTP response header manipulation rule.

```
rule "Http-Rsp-Header-Manipulation-Rule"
when
    rsp : Response(headers.has("Header Name"))
then
    rsp.headers.add("Header Name","Value")
    rsp.headers.put("Header Name","Value")
    rsp.headers.set("Header Name","Value")
    rsp.headers.del("Header Name")
end
```

HTTP Response Body Manipulation Rule

Following is a sample HTTP response body manipulation rule.

```
rule "Http-Rsp-Body-Manipulation_Rule"
    salience 20
when
    rsp : Response(body.get("JSONPath") == "value")
then
    rsp.body.add("JSONPATH","Value")
    rsp.body.put("JSONPATH","Key","Value")
    rsp.body.set("JSONPATH","Value")
    rsp.body.del("JSONPath")
end
```

Basic Rule API for HTTP Message Manipulation

Following are the APIs for HTTP Message Manipulation:

SL.No	API	Return Type	Description	Example
1	getUri()	String	returns the URI of the Request	req.getUri()
2	setUri(String URI)	void	Sets the URI in the body of request	req.setUri("http://oracle.com")
3	getHttpStatusCode()	Integer	returns the StatusCode from the body of Response	rsp.getHttpStatusCode()
4	setHttpStatusCode(Integer httpStatusCode)	void	sets the StatusCode in the body of Response	rsp.setHttpStatusCode(201)

SL.No	API	Return Type	Description	Example
5	getRejectMessage()	String	returns the Reject Message from the body of Response	rsp.getRejectMessage()
6	setRejectMessage(String rejectMessage)	void	sets the Reject Message in the body of Response	rsp.setRejectMessage("Message")

Header Rule APIs and Body Rule APIs

The following are the new header APIs which are being supported in mediation from release 1.5.0:

Header Rule APIs

Table 3-1 Header Rule APIs

SL.No	API	Return Type	Description	Example
1	get(String key)	String	Returns a header value with that key.	req.headers.get("header")
2	has(String key)	Boolean	Returns true/false depending on whether the header is present or not.	req.headers.has("header")
3	has(String key, String value)	Boolean	Returns true/false depending on whether the header is present with that value or not.	req.headers.has("header","value")
4	put(String key, String value)	Boolean	Add the header with given key and value if does not find or updates the header with value if the given key found.	req.headers.put("header","value")
5	add(String key, String value);	Boolean	Add the header with given key and value if does not find	req.headers.add("header","value")
6	set(String key, String value);	Boolean	Updates the header with value if the given key found	req.headers.set("header","value")
7	set(String key, String oldValue, String newValue);	Boolean	Update the header with new value if the existing value matches with oldValue	req.headers.set("header"," old value","new value")
8	del(String key)	Boolean	Delete the header	req.headers.del("header")
9	del(String key, String value)	Boolean	Delete the header if it matches with given key and value	req.headers.del("header","value")
10	count();	Integer	Returns the count of total numbers of all headers.	req.headers.count()

Table 3-1 (Cont.) Header Rule APIs

SL.No	API	Return Type	Description	Example
11	count(String key);	Integer	Returns the count of total numbers of headers with the key.	req.headers.count("header")
12	count(String key, String value)	Integer	Returns the count of total numbers of headers with the key and value	req.headers.count("header","value")

Body Rule APIs

Table 3-2 Body Rule APIs

SL.No	API	Return Type	Description	Example
1	get(String path)	Object	Returns the value present at that path.	body.get("$.service Name")
2	getAll(String path)	List<Object>	Returns the list of objects presnet at the path given.	body.getAll("$.serviceName")
3	has(String path);	boolean	Checks if the key is present at that path or not.	body.has("$.service Name")
4	has(String path, Object value)	boolean	Checks if the key has the value present at that path or not.	body.get("$.service Name","value")
5	absPath(String arrayPath, String elem, Object value)	Object	Returns the whole path from head to the elem if the value is present.	req.body.absPath("$.payload","iePath", "/a")
6	indexOf(String arrayPath, String elem, Object value)	Integer	Returns the index of the value in that JSONArray	req.body.indexOf("$.dataToIntegrityProtectBlock.payload","value", 100)
7	put(String path, String key, T value)	boolean	Add the body with given key and value at path if does not find or updates the body with value at the path if the given key found.	req.body.put("$","key","value")
8	add(String path, T value)	boolean	Adds the key embedded in the path with value given if element is not present.	req.body.add("$.ipEndPoints","value")
9	set(String path, T value)	boolean	Updates the key embedded in the path with value given if element is present.	rsp.body.add("$.ipEndPoints","value")
10	del(String path, T value)	boolean	Deletes the element at that path with the value present.	req.body.del("$.path","value")

Table 3-2 (Cont.) Body Rule APIs

SL.No	API	Return Type	Description	Example
11	del(String path)	boolean	Deletes the element at that path	req.body.del("$.path ")

Regex Support and Function Support

The following are the other support for rules:

Regex Support

```
rule "regex rule"
when
   req : Request(headers.get("x-number")  matches "[0-9]")
then
   req.headers.put( "Header name", "value")
end


rule "regex rule2"
when
    req : Request(body.get("$.serviceName").toString() matches
"suci-.*0-0-0.*")
then
    req.body.put("$","new key","value")
end


Example:
suci-0-123-45-0-0-0-0-1-17-
e9b9916c911f448d8792e6b2f387f85d3ecab9040049427d9edbb5431b0bc711023be6a0
57f34956ba21b45d936238aebeb7
```

In operator

```
rule "New Rule5"
when
   req : Request(headers.get("x-number")  in (1,2,3,4,5 ))
then
   req.headers.put( "Header name", "value")
end
```

Function Support

```
function String update(String str){
   int a = Integer.parseInt(str);
   a = a+1;
   str = ""+a;
   return str;
}
```

```
rule "Function"
when
   req : Request(headers.get("ADD")==1)
then
   req.headers.set("ADD",update(req.headers.get("ADD")))
end
```

Configuring Mediation Rule for PT

Add the following rules to the mediation rules.

> **Note:**
>
> In the following rules, replace **PCF_loadBalancerIP_and_Port** and **IWF_loadBalancerIP_and_Port** with appropriate values.

```
rule "pt_d2h_to_nf_rule"
salience 1
when
    req : Request(headers.has("pt_dest_uri"))
then

req.setUri(req.headers.get("pt_dest_uri").replace("pcf.com","10.75.203.7
4:1000/simulation"))
    req.headers.del("pt_dest_uri")
end

rule "pt_rar_to_pcf_rule"
salience 1
when
  req : Request(req.getUri().toString() matches ".*(npcf-
policyauthorization)*(v1)*(notification)*(notify).*")
then
 req.getHttpMessageFactImpl().forwardPath = IWFConsts.FORWARD_TO_H2D
 req.headers.add("diameterApplicationId","16777236")
 req.headers.add("diameterCommandCode","258")
 req.headers.add("original-req-uri",req.getUri())
end

rule "pt_aar_to_pcf_rule"
salience 1
when
  req : Request(req.body.has("$.ascReqData.notifUri"))
then

req.body.add("$.ascReqData.notifUri",req.body.get("$.ascReqData.notifUri
").toString().replace("iwf.com","10.75.203.74:30079"))

req.body.add("$.ascReqData.evSubsc.notifUri",req.body.get("$.ascReqData.
evSubsc.notifUri").toString().replace("iwf.com","10.75.203.74:30079"))
```

```
end

rule "pt_asr_to_pcf_rule"
salience 1
when
  req : Request(getUri().toString() matches ".*(npcf-
policyauthorization)*(v1)*(notification)*(terminate).*")
then
  req.getHttpMessageFactImpl().forwardPath=IWFConsts.FORWARD_TO_H2D
  req.headers.add("diameterApplicationId","16777236")
  req.headers.add("diameterCommandCode","274")
  req.headers.add("original-req-uri",req.getUri())
end


rule "pt_ccri_to_h2d_rule"
salience 1
when
req : Request(req.getUri() matches ".*(npcf-smpolicycontrol/v1/sm-
policies)(/$|$)")
then
req.getHttpMessageFactImpl().forwardPath=IWFConsts.FORWARD_TO_H2D
req.headers.put("diameterApplicationId","16777238")
req.headers.put("diameterCommandCode","272")
req.headers.put("requestType","CREATE")
end

rule "pt_ccru_to_h2d_rule"
salience 1
when
req : Request(req.getUri() == ".*(npcf-smpolicycontrol/v1/sm-policies/)
(.*)(/update)(/$|$)")
then
req.getHttpMessageFactImpl().forwardPath=IWFConsts.FORWARD_TO_H2D
req.headers.put("diameterApplicationId","16777238")
req.headers.put("diameterCommandCode","272")
req.headers.put("requestType","UPDATE")
req.headers.put("original-req-uri",req.getUri())
end

rule "pt_ccrt_to_h2d_rule"
salience 1
when
req : Request(req.getUri() matches ".*(npcf-smpolicycontrol/v1/sm-
policies/)(.*)(/delete)(/$|$)")
then
req.getHttpMessageFactImpl().forwardPath=IWFConsts.FORWARD_TO_H2D
req.headers.put("diameterApplicationId","16777238")
req.headers.put("diameterCommandCode","272")
req.headers.put("requestType","DELETE")
req.headers.put("original-req-uri",req.getUri())
end
```

Configuring Mediation Rule for NF-Mediation

Extracting the group name and the trigger point from the URLs provided below:

> **Note:**
>
> In rules, the "agenda-group" is used to categories the rule group using the group name and the trigger point that is provided in the URL.

URL Type 1: `[http://{apiRoot}/nmediation-http/v1]`

- The above URL does not have any group name or trigger point. So the rules that are without agenda-group are applicable on this.
- From the below reference rules, the rule `default` will be applied.

URL Type 2: `[http://{apiRoot}/nmediation-http/v1/scp]`

- The above URL has only the group name. So the rules that are under the agenda-group `scp` is applicable.
- From the below reference rules, the rule `scp-rule-header-1` will be applied.

URL Type 3: `[http://{apiRoot}/nmediation-http/v1/scp/triggerpoint2]`

- The above URL has both the group name `scp` and the trigger point `triggerpoint2`. So the rules that are under the agenda-group `scp-triggerpoint2` are applicable.
- From the below reference rules, the rule `scp-rule-header-2` and `scp-rule-body-1` will be applied.

Add the below rules to the mediation rules:

```
function Map<Object, Object> addObject() {
    return new HashMap<Object, Object>();
}

function ArrayList<Object> addArray() {
    return new  ArrayList<Object>();
}

function Map<String,Object> ipEndPoints(String ipv4Address, String
ipv6Address, String ipv6Prefix){
        Map< String,Object> ipEndPointObj = new HashMap<
String,Object>();
        ipEndPointObj.put("ipv4Address", ipv4Address);
        ipEndPointObj.put("ipv6Address", ipv6Address);
        ipEndPointObj.put("ipv6Prefix",ipv6Prefix );
        return ipEndPointObj;
}

rule "New Rule1"
    agenda-group "scp"
when
```

```
    req : Request(body.get("$.apiPrefix")=="/mediation")
then
    req.headers.put("x-dest-
path",req.body.get("$.apiPrefix").toString())
end

rule "New Rule2"
when
    req :
Request(body.has("$.defaultNotificationSubscriptions[*].callbackUri","pc
f.oracle.com/pcs/v1.0/nrf/notifycallback"))
then

req.body.add("$.ipEndPoints",ipEndPoints("10.75.243.193","2001:0db8:85a3
:0000:0000:8a2e:0370:7335","2001:0db8:85a3"))

end

rule "New Rule3"
when
    req : Request(headers.has("x-forwarded-NF","PCF") &&
body.has("$.fqdn","pcf.oracle.com"))
then
    req.headers.put("x-forwarded-NF","AMF")
    req.body.put("$","fqdn","amf.oracle.com")
end

rule "New Rule4"
when
    req : Request(body.get("$.serviceName")=="svc1")
then
    req.headers.del("x-service-name")
end

rule "New Rule5"
when
   req : Request(headers.get("x-number")  in (1,2,3,4,5 ))
then
   req.headers.put( "x-original-authority", "10.172.19.110:8080")
end
```

Configuring Diameter Peer

The peer nodes are configured in the `configmap-pcf-diam-gateway-service-diameter.yaml` file present in the location `chart pcf/templates`.

> **Note:**
>
> The IWF does not listen for incoming connections, as the connection with diameter peer is always initiated by IWF.

Following is a sample yaml file.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: pcf-diam-gateway-config-peers
data:
  diameter-config-peers: |
    version: '0.3'
    kind: 'diameter-config'
    metadata:
      label: 'diameter-config-peers'
    setting:
      reconnectDelay: 3
      responseTimeout: 5
      connectionTimeOut: 3
      watchdogInterval: 6
      transport: 'TCP'
    # type: [af, dra]
    nodes:
      - name: 'P-CSCF'
        type: 'pcrf'
        responseOnly: true
        host: '10.75.215.205'
        port: 3880
        realm: 'ociwf.oracle.com'
        identity: 'pcrfsim.ociwf.oracle.com'
```

Table 3-3 provides information about the yaml file parameters.

Table 3-3   Config Yaml File Parameter

Parameters		Definitions
reconnectDelay		Time delay in seconds between successive peer connection establishment attempts
responseTimeout		Response timer value in seconds
connectionTimeOut		Connection timer value in seconds
watchdogInterval		Inactivity time in seconds after which DWR will be triggered
transport		Transport protocol type "TCP"
Nodes (list)	name	Name of the peer node
	responseOnly	Indicates the Diameter GW proxy client or server
	host	IP address of the peer node
	port	Port on which peer node listens for connections
	realm	Realm of the peer node
	identity	FQDN of the peer node

PCF Discovery Configuration

Update the BSF details in the pcfDiscovery section in `ociwf-custom-values-1.5.0.yaml` file

- **BSF (Binding Support Function) Configuration**
  - bsfSvc: FQDN or IP of BSF service
  - bsfPort: Node Port of BSF service,
- **NRF Configuration**
  - requesterNfType: e.g. CUSTOM_IWF
  - targetNfType: e.g. BSF
- **NRF Client configuration**
  - Update primaryNrfApiRoot with fqdn of the deployed nrf's ingress gateway
  - Update nrfClientType in profile (It must match with requesterNfType value provided in pcfDiscovery)

Rules in Configmap

Reloading the IWF-Mediation ConfigMap, if in case any updates were made in the Rules.

The tool `ociwf-rule-download_tool.sh` can be used to download the rule config map in a folder(folder name must be configmap name). It needs namespace as well as required configmap name. These rules then can be changed accordingly.

The tool `ociwf-rule-upload_tool.sh` can be used to upload the rule config map from the existing config map folder. It needs namespace as well as required configmap name to be uploaded.

•If rules changed on iwf active mediation then use `ociwf-iwf-mediation-config-active`"as the name of configmap.

•If rules changed on iwf test mediation then use `ociwf-iwf-mediation-config-test` as the name of the configmap.

Reloading the NF-Mediation ConfigMap, if in case any updates were made in the Rules.

The tool `ociwf-rule-download_tool.sh` can be used to download the rule config map in a folder(folder name will be configmap name). It needs namespace as well as required configmap name. These rules then can be changed accordingly.

The tool `ociwf-rule-upload_tool.sh` can be used to upload the rule config map from the existing config map folder. It needs namespace as well as required configmap name to be uploaded.

•If rules changed on nf active mediation then use `ociwf-nf-mediation-config-active` as the name of the configmap.

•If rules changed on nf test mediation then use `ociwf-nf-mediation-config-test` as the name of the configmap.

4 Metrics, Alerts and KPIs

This section provides the details of the Metrics, Alerts and KPIs applicable for InterWorking and Mediation Function.

- IWF Metrics
- IWF Alerts
- IWF KPIs

IWF Metrics

The following are IWF Metrics:

Table 4-1 New Metriation Metrics for 1.5 Release

SL.No	Prometheus state Metric Name	Metric Description	Dimensions	Example	Metric Type
1	ociwf_med_total_rule_count	Total Number of Rules Configured Condition: As soon as mediation service comes up, total number of rules configured will be counted.	1. app (nf-mediation,nf-mediation-test, iwf-mediation, iwf-mediation-test) 2. nfInstanceId 3. vendor 4. kubernetes_namespace	ociwf_med_total_rule_count{app= "nf-mediation",nfInstanceId="IWF1",vendor="oracle",kubernetes_namespace="medsvc"}	Gauge
2	ociwf_med_active_rule_count	Total Number of Rules which will be invoked for a particular message Condition: When rules get executed.	1. app (nf-mediation,nf-mediation-test, iwf-mediation, iwf-mediation-test) 2. nfInstanceId 3. vendor 4. kubernetes_namespace	ociwf_med_active_rule_count_total{app ="nf-mediation",nfInstanceId="IWF1",vendor="oracle",kubernetes_namespace="medsvc"}	Gauge

Table 4-1 (Cont.) New Metriation Metrics for 1.5 Release

SL.No	Prometheus state Metric Name	Metric Description	Dimensions	Example	Metric Type
3	ociwf_med_individual_rule_exec_count_total	Total Number of times a particular rule gets invoked. Condition: When rules get executed	1. app (nf-mediation,nf-mediation-test, iwf-mediation, iwf-mediation-test) 2. nfInstanceId 3. vendor 4. kubernetes_namespace 5. ruleName 6. ruleGroupName	sum(ociwf_med_individual_rule_exec_count_total{app="nf-mediation"}) by (ruleName,ruleGroupName)	Counter
4	ociwf_med_http_req_total	Total Number of ingress messages to NF Condition: Whenever a msg lands on nf-mediaton.	1. app (nf-mediation, iwf-mediation) 2. nfInstanceId 3. vendor 4. kubernetes_namespace 5. ruleGroupName 6. msgType (consumerRequest, producerResponse)	ociwf_med_req_total{msgType="consumerRequest",app="nf-mediation",ruleGroupName="scp-agenda"}	Counter

Table 4-1 (Cont.) New Metriation Metrics for 1.5 Release

SL.No	Prometheus state Metric Name	Metric Description	Dimensions	Example	Metric Type
5	ociwf_med_http_rsp_total	Total Number of egress messages Condition: When nf sends response	1. app (nf-mediation,nf-mediation-test, iwf-mediation, iwf-mediation-test) 2. nfInstanceId 3. vendor 4. kubernetes_namespace 5. ruleGroupName 6. statusCode (supports specific codes:: 200, 500, 503) 7. msgType (consumerRequest, producerResponse)	ociwf_med_rsp_total{msgType="consumerRequest",ruleGroupName="scp-agenda",statusCode~="2.*"}	Counter
6	ociwf_med_test_req_total	Total Number of Incoming messages to NF Test mode Condition: whenever test mode is enabled	1. app (nf-mediation-test, iwf-mediation-test) 2. nfInstanceId 3. vendor 4. kubernetes_namespace 5. ruleGroupName 6. msgType (consumerRequest, producerResponse)	ociwf_med_test_req_total{msgType="consumerRequest",app="nf-mediation-test",ruleGroupName="scp-agenda"}	Counter

Table 4-1 (Cont.) New Metriation Metrics for 1.5 Release

SL.No	Prometheus state Metric Name	Metric Description	Dimensions	Example	Metric Type
7	ociwf_med_test_rsp_total	Total Number of response by test mode* Condition: when response is sent by nf *Although test mode won't send any response but here it means proper execution of message by test mode.	1. app (nf-mediation,nf-mediation-test, iwf-mediation, iwf-mediation-test) 2. nfInstanceId 3. vendor 4. kubernetes_namespace 5. ruleGroupName 6. statusCode (supports specific codes:: 200, 500, 503) 7. msgType (consumerRequest, producerResponse)	ociwf_med_test_rsp_total{msgType="consumerRequest"}	Counter
8	ociwf_med_msg_forwarded_to_test_mode_total	Total Number of Requests forwarded to Test mode	1. app(nf-mediation, iwf-mediation)	sum(ociwf_med_msg_forwarded_to_test_mode_total)	Counter

9	ociwf_med_rule_update_status	If Rules Reloading failed or successful
		Value = 0 {failed}
		Value = 1 {successful}
			1. app (nf-mediation, nf-mediation-test, iwf-mediation, iwf-mediation-test)
			2. vendor
			3. nfInstanceId
			4. kubernetes_namespace
				ociwf_med_rule_update_status{app="nf-mediation"}	Gauge

Table 4-1 (Cont.) New Metriation Metrics for 1.5 Release

SL.No	Prometheus state Metric Name	Metric Description	Dimensions	Example	Metric Type
10	ociwf_med_individual_rule_processing_time	Processing time of Every rule invoked	1. app (nf-mediation, nf-mediation-test, iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace 5. ruleName 6. ruleGroupName		Histogram
11	ociwf_med_msg_processing_time	Processing time of message which lands on mediation	1. app (nf-mediation, nf-mediation-test, iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace		Histogram
12	ociwf_med_forward_to_ext_total	Total Number of messages forwarded to external. Condition: When mediation works in proxy mode.	1. app (iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace		Counter
13	ociwf_med_incoming_rsp_from_ext_total	Total Number of responses from external. Condition: When mediation works in proxy mode	1. app (iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace		Counter

Table 4-1 (Cont.) New Metriation Metrics for 1.5 Release

SL.No	Prometheus state Metric Name	Metric Description	Dimensions	Example	Metric Type
14	ociwf_med_incoming_d2h_req_total	Total Number of requests from D2H service as a part of protocol translation mode.	1. app (iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace		Counter
15	ociwf_med_outgoing_rsp_to_d2h_total	Total Number of responses to D2H service as a part of protocol translation mode.	1. app (iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace		Counter
16	ociwf_med_forward_to_h2d_total	Total Number of messages forwarded to H2D as a part of protocol translation mode.	1. app (iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace		Counter
17	ociwf_med_incoming_rsp_from_h2d_total	Total Number of responses received from H2D as a part of protocol translation as a service.	1. app (iwf-mediation, iwf-mediation-test) 2. vendor 3. nfInstanceId 4. kubernetes_namespace		Counter

Table 4-2 IWF Metric Reference

Metric Details	IWF Micro Service	Metric
Number of the incoming request to DP from PDRA	DP	pdraIngressCounter
Number of successfully converted messages by D2H	D2H	iwfd2h_conversionSuccess_messages_total
Number of successful conversion from http to diameter	H2D	iwfh2d_h2dConversion_Success_total

Table 4-2 (Cont.) IWF Metric Reference

Metric Details	IWF Micro Service	Metric
Number of successful conversion from diameter to http	H2D	iwfh2d_d2hConversion_Success_total
Number of success Outgoing responses from PcfDiscovery	PcfDiscovery	pcfDiscSuccessResponseCounter
Number of success Outgoing responses from BSF	PcfDiscovery	bsfSucessResponseCounter
Number of responses received from mediation	D2H	iwfd2h_mediation_response_total
Number of responses from mediation to D2H	Mediation	iwfd2h_mediation_response_total
Number of Response	All	IWF Egress Request rate
Number of requests sent to mediation from D2H	D2H	iwfd2h_mediation_outgoing_total
Number of requests sent to D2H from DP	DP	iwfdiameterproxy1_d2h_outgoing_total
Number of Request	All	IWF Ingress Request rate
Number of outgoing responses from H2D to mediation	Mediation	iwfmediation_h2d_response_total
Number of outgoing responses from mediation	Mediation	iwfmediation_outgoing_response_total
Number of outgoing requests from mediation	Mediation	iwfmediation_outgoing_request_total
Number of messages going from PDRA to PCF	DP	pdraPcfEgressCounter
Number of Incoming responses to PcfDiscovery	PcfDiscovery	pcfDiscRequestCounter
Number of incoming responses to mediation	Mediation	iwfmediation_incoming_response_total
Number of incoming responses to BSF	PcfDiscovery	bsfRequestCounter
Number of incoming requests to mediation	Mediation	iwfmediation_incoming_request_total
Number of incoming requests from mediation to H2D	Mediation	iwfmediation_h2d_outgoing_total
Number of incoming requests from D2H to mediation	Mediation	iwfd2h_mediation_incoming_total
Number of incoming messages to D2H from diameter Proxy Service	D2H	iwfd2h_incoming_messages_total
Number of failures while sending message to mediation from D2H	D2H	iwfd2h_mediation_error_total
Number of failure Outgoing responses from PcfDiscovery	PcfDiscovery	pcfDiscFailureResponseCounter
Number of failure Outgoing responses from BSF	PcfDiscovery	bsfFailureResponseCounter
Number of failed response (responses other than 200OK)from PCF Discovery to DP	DP	failedResponseToPdraCounter

Table 4-2 (Cont.) IWF Metric Reference

Metric Details	IWF Micro Service	Metric
Number of diameter requests sent to diameter proxy	H2D	iwfh2d_diameter_request_outgoing_total
Number of Diameter requests sent to Diameter peer	DP	iwfdiameterproxy1_diameter_outgoing_total
Number of Diameter requests received from H2D	DP	iwfdiameterproxy1_h2d_incoming_total
Number of Diameter requests received from Diameter peer	DP	iwfdiameterproxy1_diameter_incoming_total
Number of diameter request send error occurred	H2D	iwfh2d_diameter_error_total
Number of diameter answers received from diameter proxy	H2D	iwfh2d_diameter_response_incoming_total
Number of 200OK responses from PCF Discovery to DP	DP	successResponseToPdraCounter

The table *Mediation Metric Reference* provides the information about Mediation Metrics.

Table 4-3 Mediation Metric Reference

Metric	Metric Description	Mediation Micro Service
Mediation Ingress Request rate	Number of Request	All
Mediation Egress Request rate	Number of Response	All
iwfmediation_incoming_request_total	Number of incoming requests to iwf mediation	iwf-mediation
iwfmediation_outgoing_response_total	Number of outgoing responses from iwf mediation	iwf-mediation
iwfd2h_mediation_incoming_total	Number of incoming requests from D2H to iwf mediation	iwf-mediation
iwfd2h_mediation_response_total	Number of responses from iwf mediation to D2H	iwf-mediation
iwfmediation_h2d_outgoing_total	Number of incoming requests from iwf mediation to H2D	iwf-mediation
iwfmediation_h2d_response_total	Number of outgoing responses from H2D to iwf mediation	iwf-mediation
iwfmediation_outgoing_request_total	Number of outgoing requests from iwf mediation	iwf-mediation
iwfmediation_incoming_response_total	Number of incoming responses to iwf mediation	iwf-mediation
nfmediation_incoming_request_total	Number of incoming requests to nf mediation	nf-mediation
nfmediation_outgoing_response_total	Number of outgoing responses from nf mediation	nf-mediation

Alerts

The following are the alerts of IWF:

IWF Alerts

The following are IWF Alerts:

Table 4-4 IWF Alerts

SLNo	Alert Name	Severity	OID used for SNMP Traps	Metric Applicable	Threshold	Description
1	NFMediationIngressTrafficRateAboveMinorThreshold	Info	1.3.6.1.4.1.323.5.3.47.1.2.1001	rate(ociwf_nf_med_http_req_total{app="nf-mediation"}[2m])	70% of max MPS	Notify user after a certain threshold traffic rate is reached.
2	NFMediationIngressTrafficRateAboveMajorThreshold	Warning	1.3.6.1.4.1.323.5.3.47.1.2.1001	rate(ociwf_nf_med_http_req_total{app="nf-mediation"}[2m])	80% of max. MPS	Notify user after a certain threshold traffic rate is reached.
3	NFMediationIngressTrafficRateAboveCriticalThreshold	Critical	1.3.6.1.4.1.323.5.3.47.1.2.1001	rate(ociwf_nf_med_http_req_total{app="nf-mediation"}[2m])	95% of max. MPS	Notify user after a certain threshold traffic rate is reached.
4	NFMediation Response Failure	Info	1.3.6.1.4.1.323.5.3.47.1.2.2001	rate(ociwf_nf_med_http_rsp_total{statusCode!="200",app="nf-mediation"}[2m])	100	Notify user that there is a failure in the response execution.
5	NFMediation Test Response Failure	Info	1.3.6.1.4.1.323.5.3.47.1.2.2002	rate(ociwf_nf_med_test_rsp_total{statusCode!="200",app="nf-mediation-test"}[2m])	100	Notify user that there is a failure in the response execution.

Table 4-4 (Cont.) IWF Alerts

SLNo	Alert Name	Severity	OID used for SNMP Traps	Metric Applicable	Threshold	Description
6	NFMediationPodMemoryUsage	Warning	`1.3.6.1.4.1.323.5.3.47.1.2.3001`	sum(container_memory_usage_bytes{namespace="medsvc",container_name="nf-mediation"}) by (pod_name,namespace)	70%	Notify user that NFMediation Memory usage per pod threshold value is reached.
7	NFMediationPodTestMemoryUsage	Warning	`1.3.6.1.4.1.323.5.3.47.1.2.3002`	sum(container_memory_usage_bytes{namespace="medsvc",container_name="nf-mediation-test"}) by (pod_name,namespace)	70%	Notify user that NFMediation Test Memory usage per pod threshold value is reached.
8	NFMediationPodCPUUsage	Warning	`1.3.6.1.4.1.323.5.3.47.1.2.4001`	sum(container_cpu_usage_seconds_total{namespace="medsvc",container_name="nf-mediation"}) by (pod_name,namespace)	70%	Notify user that NFMediation CPU usage per pod threshold value is reached.
9	NFMediationPodTestCPUUsage	Warning	`1.3.6.1.4.1.323.5.3.47.1.2.4002`	sum(container_cpu_usage_seconds_total{namespace="medsvc",container_name="nf-mediation-test"}) by (pod_name,namespace)	70%	Notify user that NFMediation Test CPU usage per pod threshold value is reached.
10	NFMediationRuleUpdateFailure	Critical	`1.3.6.1.4.1.323.5.3.47.1.2.5001`	ociwf_med_rule_update_status{app="nf-mediation"}	0	Notify user that rule updation into the configmap failed.
11	NFMediationTestRuleUpdateFailure	Critical	`1.3.6.1.4.1.323.5.3.47.1.2.5002`	ociwf_med_rule_update_status{app="nf-mediation-test"} < 1	0	Notify user that rule updation into the configmap failed for test mode.

Table 4-4 (Cont.) IWF Alerts

SLNo	Alert Name	Severity	OID used for SNMP Traps	Metric Applicable	Threshold	Description
12	IWFMediationPodCPUUsage	Warning	`1.3.6.1.4.1.323.5.3.47.1.2.6001`	sum(container_cpu_usage_seconds_total{container="iwf-mediation"}) by (pod_name,namespace)	70%	Notify user that IWFMediation CPU usage per pod threshold value is reached.
13	IWFMediationPodMemoryUsage	Warning	`1.3.6.1.4.1.323.5.3.47.1.2.7001`	sum(container_memory_usage_bytes{container="iwf-mediation"}) by (pod_name,namespace)	70%	Notify user that IWFMediation Memory usage per pod threshold value is reached.

IWF Alert Configuration

Follow the steps below for IWF Alert configuration in Prometheus:

> **Note:**
>
> 1. By default Namespace for OCIWF is `ociwf` that must be updated as per the deployment.
> 2. The `OCIWF-config-1.5.0.0.0.zip` file can be downloaded from OHC. Unzip the `OCIWF-config-1.5.0.0.0.zip` package after downloading to get `IWFAlertrules-1.5.0.yaml`file.

Procedure

1. Take a backup of current configuration map of Prometheus:

```
kubectl get configmaps _NAME_-server -o yaml -n _Namespace_ > /tmp/
tempConfig.yaml
```

2. Check and add OCIWF Alert file name inside Prometheus configuration map:

```
sed -i '/etc\/config\/alertsiwf/d' /tmp/tempConfig.yaml
sed -i '/rule_files:/a\ \- /etc/config/alertsiwf' /tmp/
tempConfig.yaml
```

3. Update configuration map with updated file name of OCIWF alert file:

```
kubectl replace configmap _NAME_-server -f /tmp/tempConfig.yaml
```

4. Add OCIWF Alert rules in configuration map under file name of OCIWF alert file:

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

> **Note:**
>
> The Prometheus server takes an updated configuration map that is automatically reloaded after approximately 20 seconds. Refresh the Prometheus GUI to confirm that the OCIWF Alerts have been reloaded.

IWF KPIs

The following are IWF KPIs:

SL.NO	KPI Name	KPI Details	Metric User
1	OCIWF Ingress Request	Rate of HTTP requests received at OCIWF Ingress Gateway	oc_ingressgateway_http_requests
2	OCIWF Incoming Request per Agenda Group	Rate of HTTP requests received at OCIWF service per Agenda Group	ociwf_med_http_req_total
3	OCIWF 2xx Response per Agenda Group	Rate of 2xx HTTP response from OCIWF per Agenda Group	ociwf_med_http_rsp_total
4	OCIWF 4xx Response per Agenda Group	Rate of 4xx HTTP response from OCIWF per Agenda Group	ociwf_med_http_rsp_total
5	OCIWF 5xx Response per Agenda Group	Rate of 5xx HTTP response from OCIWF per Agenda Group	ociwf_med_http_rsp_total
6	OCIWF CPU Usage per service	CPU utilization per service	container_cpu_usage_seconds_total
7	OCIWF Memory consumed per service	Memory Consumed per service	container_memory_usage_bytes
8	OCIWF Processing time	OCIWF Processing Time	ociwf_med_msg_processing_time