Oracle® Communications

Cloud Native Security Edge Protection Proxy (SEPP) User's Guide

Release 1.4.0

F35438-01

September 2020

Oracle Communications Cloud Native Security Edge Protection Proxy (SEPP) User's Guide, Release 1.4.0

F35438-01

Contents

List of Tables

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.

Whats New In This Guide

New and Updated Features in Release 1.4:

- TLS Protection Mode: This enables inter PLMN communication over N32 interface using TLS mode
- Added Metrics, Alerts and KPIs

1
Overview

The Security Edge Protection Proxy (SEPP) is a non-transparent proxy that sits at the perimeter of the PLMN network and enables secured communication between inter-PLMN network messages. The OCSEPP supports the following functions:

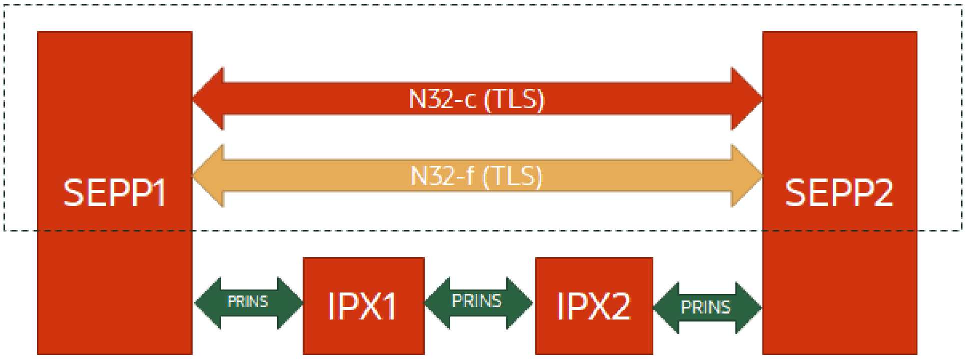
- **Implements separate security negotiation interface (N32-c) and an end-to-end application interface (N32-f)**
  - N32-C: Uses for context management for security capability negotiation.
  - N32-F: Uses for forwarding of inter-NF signaling across PLMN.
- **Secure communication**
  - Secure communication of Inter PLMN messages from Consumer NF to Producer NF using TLS protection mode (HTTP over TLS).
  - HTTPS and 3gpp-Sbi-Target-Apiroot header for secure Intra-PLMN communication to Network Functions.
- **Rest API based configurations**
  - APIs for configuration of roaming partner profiles.
  - APIs retrieval of Handshake status for each roaming partner.
- **Resiliency**
  - Implements retry mechanism for N32-C handshake failure.
- **Observability**
  - PLMN based Metrics and KPI.
  - Alarms for failure and threshold events.
  - Logging and tracing capabilities.

Security Edge Protection Proxy (SEPP) Architecture

This section explains the Security Edge Protection Proxy (SEPP) architecture.

Oracle Communications Security Edge Protection Proxy (OCSEPP) is a proxy network function (NF) which is used for secured communication between inter-Public Land Mobile Network (PLMN) messages.

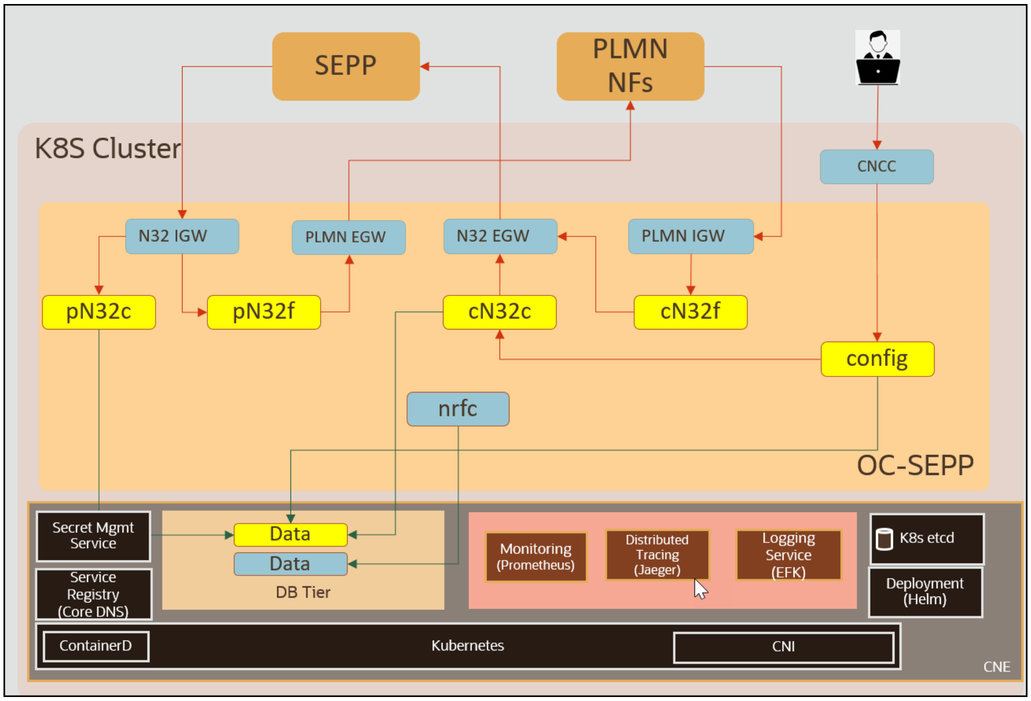
Security Edge Protection Proxy Interfaces



SEPP provides two interfaces, N32-c and N32-f .

- **N32-c:** A control plane interface between the SEPPs for performing initial handshake and negotiating the parameters to be applied for the actual N32 message forwarding.
- **N32-f:** A forwarding interface between the SEPPs which is used for forwarding the communication between the NF service consumer and the NF service producer after applying application level security protection.

Security Edge Protection Proxy Architecture



SEPP Architecture and the Microservices Overview

ocsepp-config-mgr Microservice

This microservice is used for configuration of roaming partner profile and retrieving handshake status. It also initiates a notification towards cN32c service to initiate handshake as soon as profile is enabled and notification to delete n32 context as soon as profile is disabled. Audit to re-initiate handshake notification in case of handshake failure is also handled by config-mgr.

ocsepp-cn32c Microservice
This microservice is used for context management and initiation of security capability negotiation. It create an n32 context and maintain its state and negotiated security capability based on handshake procedure.

ocsepp-pn32c Microservice
This microservice is used for context management and respond to security capability negotiation. It create an n32 context and maintain its state and negotiated security capability based on handshake procedure.

ocsepp-cn32f Microservice
This microservice receives messages from consumer NF and determine remote SEPP address to forward message over n32f interface to SEPP of producer NF network.

ocsepp-pn32f Microservice
This microservice is receives inter-PLMN messages from remote SEPP of consumer network over TLS connection and forward to Producer NF.

ocsepp-plmn-Ingress-gateway
This is the entry point for comsumer NF to SEPP messages. PLMN Ingress Gateway exposes http and https interfaces towards consumer NF network to receives messages required to be forwarded over inter-PLMN n32f interface.

ocsepp-plmn-egress-gateway
This is the exit point for SEPP to producer NF messages. PLMN Egress gateway exposes http and https interface towards producer NF network to forward inter-PLMN messages received over inter-PLMN n32f interface.

ocsepp-n32-ingress-gateway
Entry point for N32 Ingress messages. N32 Ingress gateway exposes HTTP over TLS interface on prouder SEPP to receive n32c and n32f inter-PLMN messages.

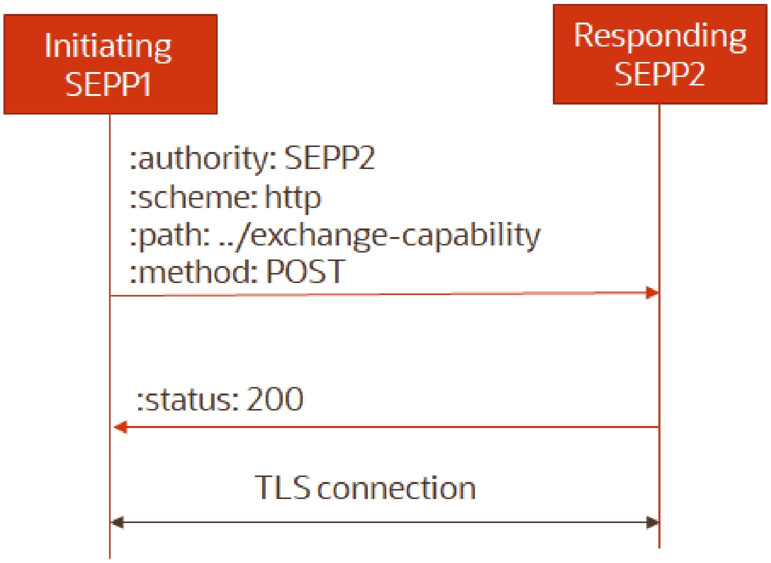
ocsepp-n32-egress-gateway
Exit point for N32 Egress messages. N32 Egress gateway exposes HTTP over TLS interface on consumer SEPP to forward n32c and n32f inter-PLMN messages.

nrf-client
NRF client (nrfc) is used for NRF registration. NRF client supporting services are not shown.

N32c and N32f Call flows

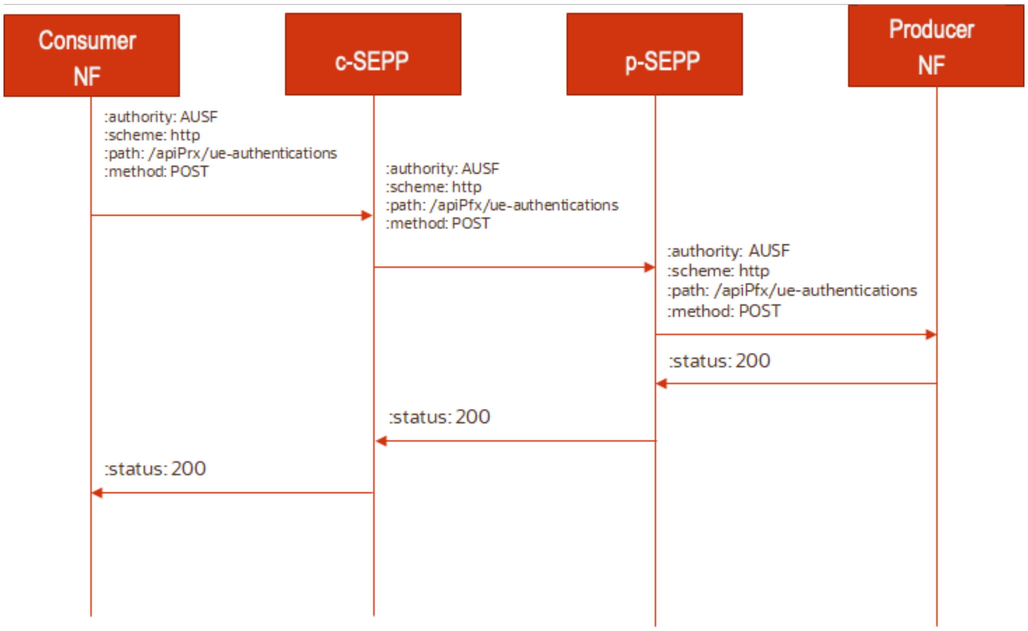
The following are the N32c and N32f call flows of SEPP.

N32c Call flow



- The initiating SEPP issues a HTTP POST request towards the responding SEPP with the request body containing the "SecNegotiateReqData" with supporting security capabilities i.e. PRINS and/or TLS.
- On successful processing of the request, the responding SEPP shall respond to the initiating SEPP1 with “200 OK” status code and a POST response body with selected security capability i.e. PRINS or TLS.

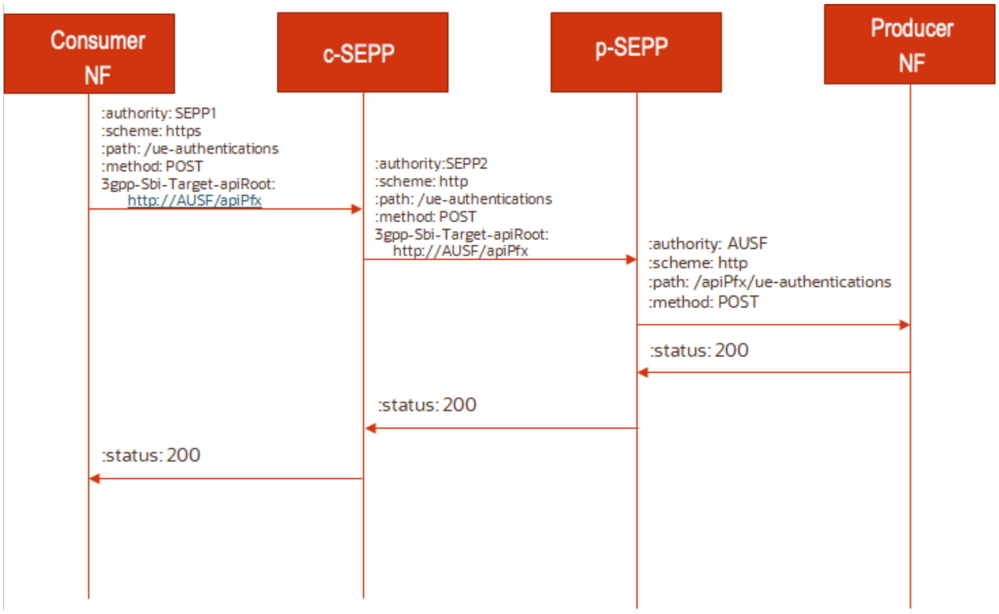
N32f Call flow: HTTP scheme is used by NF to communicate to SEPP



- The SEPP on the NF service consumer side (c-SEPP) and the SEPP on the NF service producer side (p-SEPP) negotiate the security capabilities using the procedure defined in Handshake call flow. The SEPPs mutually negotiate to use TLS as the security policy.

- The NF service consumer is configured to route all HTTP messages with inter PLMN FQDN as the "authority" part of the URI via the c-SEPP. The c-SEPP acts as a HTTP proxy.
- The c-SEPP forwards the HTTP service request within the N32-f TLS tunnel established.
- The p-SEPP forwards the HTTP service request to the NF service producer.
- The NF service producer sends the HTTP service response
- The p-SEPP forwards the HTTP service response within TLS tunnel to the c-SEPP.
- The c-SEPP forwards the HTTP service response to the NF service consumer

N32f Call flow: HTTPS with 3gpp-Sbi-Target-apiRoot header is used by NF to communicate with SEPP



- The SEPP on the NF service consumer side (c-SEPP) and the SEPP on the NF service producer side (p-SEPP) negotiate the security capabilities using the procedure defined in Handshake call flow. The SEPPs mutually negotiate to use TLS as the security policy.
- NF service consumer is configured with c-SEPP FQDN and sets up a TLS connection with c-SEPP.
- The NF service consumer sends the HTTP service request within the TLS connection to the c-SEPP, including a 3pp-Sbi-Target-apiRoot header set to the apiRoot of the p-NF.
- The c-SEPP forwards the HTTP service request within the N32-f TLS tunnel established.
- The p-SEPP extracts the HTTP message received on the TLS connection, and then seeing that the URI scheme of the NF service producer is "https" in 3gpp-sbi-target-apiRoot header, the p-SEPP sets up a TLS connection with the NF service producer.
- The p-SEPP forwards the HTTP service request to the NF service producer.

- The NF service producer sends the HTTP service response
- The p-SEPP forwards the HTTP service response within TLS tunnel to the c-SEPP.
- The c-SEPP upon receiving the HTTP response message within the TLS tunnel , forwards the response to the NF service consumer.

SEPP Microservices

The following are the microservices of SEPP

Name	Description
cN32c	Initiates N32-c handshake to remote PLMNs
pN32c	Responds to N32-c handshake from remote PLMNs
cN32f	Initiates inter PLMN messages to remote PLMNs
pN32f	Responds to inter PLMN messages from remote PLMNs
config	SEPP configuration micro service
PLMN IGW, PLMN EGW	PLMN Ingress and Egress GW micro services
N32 IGW, N32 EGW	N32 Ingress and Egress GW micro services
nrfc,config-server, app-info	NRF Client and supporting services used for SEPP registration

Acronyms

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

Table 1-1 Acronyms

Fields	Description
CNE	Cloud Native Environment
DNS	Domain Name System
FQDN	Fully Qualified Domain Name
NF	Network Function
OHC	Oracle Help Center
OSDC	Oracle Software Delivery Cloud
PLMN	Public Land Mobile Network
SEPP	Security Edge Protection Proxy
SVC	Services
TLS	Transport Layer Security

References

- Cloud Native Environment (OCCNE) Installation Guide
- Security Edge Protection Proxy (SEPP) Cloud Native Installation Guide

2

Configuring SEPP Using REST APIs

The following SEPP managed objects can be configured using REST APIs:

Resource information

RoamingPartnerProfile

Method	URL	Description	Input	Return	Example
GET	/sepp-configuration/v1/roamingpartnerprofile	Returns all the configured roaming partner profiles Empty List [], if no roaming partner profiles are configured		List of RoamingPartnerProfile	`curl --location --request GET 'http://:<SEPP-Configuration IP>:<PORT>//sepp-configuration/v1/roamingpartnerprofile/'`
GET	/sepp-configuration/v1/roamingpartnerprofile/{name}	Returns configured RoamingPartnerProfile if not present returns 404 Exception: Roaming partner profile not found for the name	Name of RoamingPartnerProfile as request parameter	RoamingPartnerProfile	`curl --location --request GET 'http://<SEPP-Configuration IP>:<PORT>//sepp-configuration/v1/roamingpartnerprofile/SEPP-3'`

Method	URL	Description	Input	Return	Example
POST	/sepp-configuration/v1/roamingpartnerprofile/	Returns configured RoamingPartnerProfile	RoamingPartnerProfile Object	RoamingPartnerProfile	`curl --location --request POST 'http://<SEPP-Configuration IP>:<PORT>/sepp-configuration/v1/roamingpartnerprofile/' \ --header 'Content-Type: application/json' \ --data-raw '{ "plmn": { "mcc":"111", "mnc":"123" }, "domain":"ocsepp-nf33", "seppId": "sepp33", "seppFqdn":"10.75.237.33", "securityCapabilityList": ["PRINS"], "connectionType":"INITIATOR", "isEnabled":"TRUE", "apiPrefix":"n32c-handshake", "apiVersion":"v1", "port":30043 }'`

Method	URL	Description	Input	Return	Example
DELETE	/sepp-configuration/v1/roamingpartnerprofile/{name}	Delete configured RoamingPartner Profile	Name of RoamingPartnerProfile as request param		`curl --location --request DELETE 'http://<SEPP-Configuration IP>:<PORT>//sepp-configuration/v1/roamingpartnerprofile/SEPP-3'`

Method	URL	Description	Input	Return	Example
PUT	/sepp-configuration/v1/roamingpartnerprofile/{name}	Update configured RoamingPartnerProfile 404 Error if not found Exception: Roaming partner profile not found for the name	1. Name of RoamingPartnerProfile as request param 2. Updated RoamingPartnerProfile object	RoamingPartnerProfile	`curl --location --request PUT 'http://<SEPP-Configuration IP>:<PORT>//sepp-configuration/v1/roamingpartnerprofile/SEPP-3' \ --header 'Content-Type: application/json' \ --data-raw '{ "name": "SEPP-3", "plmn": { "mcc": "311", "mnc": "282" }, "domain": "ocsepp-nf3", "seppId": "sepp3", "seppFqdn": "10.75.237.9", "connectionType": "INITIATOR", "securityCapabilityList": ["TLS"], "isEnabled": "FALSE", "port": 30043, "apiPrefix": "n32c-handshake", "apiVersion": "v1", "priority": 0 }'`

Method	URL	Description	Input	Return	Example
PATCH	/sepp-configuration/v1/roamingpartnerprofile/{name}	Update only provided fields of configured RoamingPartner Profile 404 Error if not found Exception: Roaming partner profile not found for the name	1. Name of RoamingPartnerProfile as request param 2. Fields of RoamingPartnerProfile to be updated	RoamingPartnerProfile	`curl --location --request PATCH 'http://<SEPP-Configuration IP>:<PORT>/sepp-configuration/v1/roamingpartnerprofile/SEPP-2' \ --header 'Content-Type: application/json' \ --data-raw '{ "isEnabled": "FALSE" }'`

Handshake Status

Attribute	Datatype	Compliance	Default Value	Pattern	Description
GET	/sepp-configuration/v1/handshakestatus	Returns all the handshake status Empty list [], if no handshake status		List of HandshakeStatus	`curl --location --request GET 'http://<SEPP-Configuration IP>:<PORT>/sepp-configuration/v1/handshakestatus/'`

Attribute	Datatype	Compliance	Default Value	Pattern	Description
GET	/sepp-configuration/v1/handshakestatus/{fqdn}	Returns handshake status of the SEPP FQDN 404 Error if not found Exception: Handshake Status not found for the fqdn	SEPP FQDN of the handshake status	HandshakeStatus	`curl --location --request GET 'http://<SEPP-Configuration IP>:<PORT>/sepp-configuration/v1/handshakestatus/10.75.191.11'`

3

Metrics, KPIs and Alerts

SEPP Metrics

This section provides information about the SEPP metrics.

Microservice Name : Consumer N32c

SL.No.	Prometheus Stat Metrics Name	Metrics Description(including pegging condition)	Dimensions
1.	sepp_n32c_handshake_failure	If N32c Handshake Procedure fails, this metrics will be pegged and corresponding alarm will be raised.	1. app(consumer, producer) 2. peer_domain 3. peer_fqdn 4. n32c_procedure 5. vendor 6. nfInstanceId 7. kubernetes_namespace
2.	sepp_cn32c_handshake_requests_total	Total number of requests send over n32c for handshake procedure. Condition:: When sepp initiates any handshake procedure requests towards peer sepp.	1. peer_domain 2. peer_fqdn 3. n32c_procedure 4. vendor 5. nfInstanceId 6. kubernetes_namespace 7. direction=egress

SL.No.	Prometheus Stat Metrics Name	Metrics Description(including pegging condition)	Dimensions
3.	sepp_cn32c_handshake_response_total	Total number of responses received over n32c for handshake procedure. Condition:: When sepp receives any handshake procedure response from peer sepp. It can be successful or failure based on responseCode	1. peer_domain 2. peer_fqdn 3. n32c_procedure 4. responseCode(2xx,4xx,5xx) 5. vendor 6. nfInstanceId 7. kubernetes_namespace 8. direction=ingress
4.	sepp_cn32c_handshake_initiation_req_total	Total number of Handshake Initiation requests received from config-mgr. Condition:: When handshake initiation requests received from config-mgr.	1. peer_domain 2. peer_fqdn 3. vendor 4. nfInstanceId 5. kubernetes_namespace
5.	sepp_cn32c_handshake_delete_req_total	Total number of Handshake context delete requests received from config-mgr. Condition:: When handshake context delete requests received from config-mgr.	1. peer-domain 2. peer_fqdn 3. vendor 4. nfInstanceId 5. kubernetes_namespace

Microservice Name : Producer N32c

S.No.	Prometheus Stat Metrics Name	Metrics Description(including pegging condition)	Dimensions
1.	sepp_pn32c_handshake_requests_total	Total number of requests received over n32c for handshake procedure. Condition:: When any handshake procedure request received from peer sepp.	1. peer_domain 2. peer_fqdn 3. n32c_procedure 4. vendor 5. nfInstanceId 6. kubernetes_namespace 7. direction=ingress

S.No.	Prometheus Stat Metrics Name	Metrics Description(including pegging condition)	Dimensions
2.	sepp_pn32c_handshake_response_total	Total number of responses sent over n32c for handshake procedure. Condition:: When sepp sends response to handshake procedure received. It can be a success response or failure response based on success code.	1. peer_domain 2. peer_fqdn 3. n32c_procedure 4. statusCode(2xx,4xx,5xx) 5. vendor 6. nfInstanceId 7. kubernetes_namespace 8. direction=egress

Microservice Name : Consumer N32f

S.No.	Prometheus Stat Metrics Name	Metrics Description(including pegging condition)	Dimensions
1.	sepp_cn32f_requests_failure	Total number of requests failed to be sent from cn32f to remote SEPP. Condition:: When any error or exception occurs on cn32f side because of which request is not sent to pn32f.	1. peer_domain 2. peer_id 3. peer_fqdn 4. vendor 5. nfInstanceId 6. kubernetes_namespace 7. statuscode 8. error_msg
2.	sepp_cn32f_response_failure	Total number of response failed to be sent from cn32f pod to NF. Condition:: When any error or exception occurs on cn32f side because of which request is not sent to NF.	1. peer_domain 2. peer_id 3. peer_fqdn 4. vendor 5. nfInstanceId 6. kubernetes_namespace 7. statuscode 8. error_msg

S.No.	Prometheus Stat Metrics Name	Metrics Description(including pegging condition)	Dimensions
3.	sepp_cn32f_requests	Total number of requests received from NF. Condition:: When a request is received on InboundInterface of cn32f.	1. peer_domain 2. peer_id 3. peer_fqdn 4. vendor 5. nfInstanceId 6. kubernetes_namespace
4.	sepp_cn32f_response	Total number of response received from remote SEPP. Condition:: When a response is received on OutboundInterface of cn32f.	1. peer_domain 2. peer_id 3. peer_fqdn 4. vendor 5. nfInstanceId 6. kubernetes_namespace

Microservice Name: pN32f

S.No	Prometheus Stat Metric Name	Metric Description(including pegging condition)	Dimensions
1	sepp_pn32f_requests_rx	Number of requests received from Peer Sepp Condition: When a request reaches pn32f from peer Sepp.	1. vendor 2. nfInstanceId 3. kubernetes_namespace 4. kubernetes_podname 5. peer_fqdn
2	sepp_pn32f_requests_tx	Number of requests transmitted to NF.Condition: When a request transmits a message to NF	1. vendor 2. nfInstanceId 3. kubernetes_namespace 4. kubernetes_podname 5. peer_fqdn

S.No	Prometheus Stat Metric Name	Metric Description(including pegging condition)	Dimensions
3	sepp_pn32f_responses_rx	Number of responses received from EGW Condition: When a response reaches pn32f from EGW.	1. vendor 2. nfInstanceId 3. kubernetes_namespace 4. kubernetes_podname 5. statusCode(2xx,4xx,5xx) 6. error-reason(in case of EGW failure) 7. peer_fqdn
4	sepp_pn32f_responses_tx	Number of responses transmitted to cSepp.Condition: When a responses transmits a message to cSepp	1. vendor 2. nfInstanceId 3. kubernetes_namespace 4. kubernetes_podname 5. statusCode(2xx,4xx,5xx) 6. peer_fqdn

Note:

The templates for dashboard creations not shared in the deliverables. So dashboard needs to be created manually based on needs

SEPP KPIs

This section provides information about the SEPP KPIs.

KPI Details	SEPP Microservice	Metrics Used	Service Operation	Response Code
cn32c Handshake Success rate	cn32c	(sum(ocsepp_cn32c_handshake_response_total)/ sum(ocsepp_cn32c_handshake_requests_total))*100	n32c Handshake	200
cn32f Routing Success Rate	cn32f	(sum(ocsepp_cn32f_response_total)/ sum(ocsepp_cn32f_requests_total))*100	n32f message forward	All

KPI Details	SEPP Microservice	Metrics Used	Service Operation	Response Code
cn32f Requests Rate Per Remote SEPP	cn32f	sum(irate(ocsepp_cn32f_requests_total[2m]))by(PEER_DOMAIN, PEER_FQDN, PLMN_ID)	n32f message forward	All
cn32f Response Rate Per Remote SEPP	cn32f	sum(irate(ocsepp_cn32f_response_total[2m])) by(PEER_DOMAIN, PEER_FQDN, PLMN_ID)	n32f message forward	All
cn32c Handshake Failure Per Remote SEPP	cn32c	sum(ocsepp_n32c_handshake_failure)by(peer_domain, peer_fqdn, peer_plmn_id)	n32c Handshake	4xx and 5xx
pn32c Handshake Success rate	pn32c	(sum(ocsepp_pn32c_handshake_response_total)/ sum(ocsepp_pn32c_handshake_requests_total))*100	n32c Handshake	200
pn32f Routing Success Rate	pn32f	(sum(ocsepp_pn32f_response_total)/ sum(ocsepp_pn32f_requests_total))*100	n32f message forward	All
pn32f Requests Rate	pn32f	sum(irate(ocsepp_pn32f_requests_rx_total[2m]))	n32f message forward	All
pn32f Response Rate	pn32f	sum(irate(ocsepp_pn32f_response_rx_total[2m]))	n32f message forward	All
pn32c Handshake Failure Per Remote SEPP	pn32c	sum(ocsepp_n32c_handshake_failure)by(peer_domain, peer_fqdn, peer_plmn_id)	n32c Handshake	4xx and 5xx

SEPP Alerts

This section provides information about the SEPP alerts and their configuration.

Table 3-1 SEPP Alerts

SL.no.	Alert	Severity	oid	Dimensions	Description	Remarks
1	SEPPCn32cHandshakeFailureAlert	Major	1.3.6.1.4.1.323.5.3.46.1.2.2001	reason	Handshake procedure has failed on Consumer sepp	SEPP-1.4
2	SEPPPn32cHandshakeFailureAlert	Major	1.3.6.1.4.1.323.5.3.46.1.2.3001	reason	Handshake procedure has failed on Producer sepp	SEPP-1.4
3	SEPPN32fRoutingFailure	warning	1.3.6.1.4.1.323.5.3.46.1.2.4001	error_msg	N32f service not able to forward message	SEPP-1.4

Table 3-1 (Cont.) SEPP Alerts

SL.no.	Alert	Severity	oid	Dimensions	Description	Remarks
4	SEPPPodMemoryUsageAlert	warning	1.3.6.1.4.1.323.5.3.46.1.2.4003		Pod memory usage is above threshold (70%)	SEPP-1.4
5	SEPPPodCpuUsageAlert	warning	1.3.6.1.4.1.323.5.3.46.1.2.4002		Pod CPU usage is above threshold (70%)	SEPP-1.4

SEPP Alert Configuration

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

Steps for SEPP Alert Configuration in Prometheus

*Namespace* - Prometheus NameSpace

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 prometheus server 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_-server -o yaml -n _Namespace_ > /tmp/
   tempConfig.yaml
   ```

   where, <Namespace> is the prometheus server namespace used in helm install command.
   For Example, assuming chart name is "prometheus-alert", so "_NAME_-server" becomes "prometheus-alert-server", execute the following command to find the config map:

   ```
   kubectl get configmaps prometheus-alert-server -o yaml -n
   prometheus-alert2 > /tmp/tempConfig.yaml
   ```

3. Check if alertssepp is present in the t_mapConfig.yaml file by executing the following command:

   ```
   cat /tmp/t_mapConfig.yaml  | grep alertssepp
   ```

4. If alertssepp is present, delete the alertssepp entry from the t_mapConfig.yaml file, by executing the following command:

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

5. If alertssepp is not present, add the alertssepp entry in the t_mapConfig.yaml file by executing the following command:

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

6. Reload the config map with the modifed file by executing the following command:

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

7. Add seppAlertRules.yaml file into prometheus config map under filename of SEPP alert file by executing the following command :

   ```
   kubectl patch configmap <Name> -n <Namespace> --type merge --patch
   "$(cat <PATH>/seppAlertRules.yaml)"
   ```

8. Restart prometheus-server pod.
9. Verify the alerts in prometheus GUI.

> **Note:**
>
> ```
> Prometheus server takes updated configmap reloaded after sometime
> automatically (~20 sec)
> ```

4

SEPP Response Codes

Following are the SEPP Response Codes:

- **N32c Handshake Response Codes**
- **N32f Plane Message Forwarding Response Codes**

N32c Handshake Response Codes

Request	Response Code	ResponseBody	Reference	Use Case	Comment/ Remarks	Applicable	Scope(PI-E)
Capability Exchange	200 OK	SecNegotiateRspBody	TS29573_N32_Handshake.yaml	Handshake success	Handshake Success		Yes
	400 Bad Request	Problem Details	TS29573_N32_Handshake.yaml	1. Invalid Request Body 2. Mandatory IE Missing	Handshake fail		Yes
	404 Not Found	Problem Details	3gpp Spec 29500	1. Incorrect API			Yes
	405 Method Not Allowed	Problem Details	3gpp Spec 29500	When request method other than POST is used			Yes
	411 Length Required	Problem Details	TS29573_N32_Handshake.yaml	When Content Length header is not present in request.			No
	413 Payload Too Large	Problem Details	TS29573_N32_Handshake.yaml	When request entity is larger than limits defined by server.			No

Request	Response Code	ResponseBody	Reference	Use Case	Comment/ Remarks	Applicable	Scope(PI-E)
	415 Unsupported Media Type	Problem Details	TS29573_N32_Handshake.yaml	When payload format is not in json. When "Content-Type:application/json" is not present.			Yes
	429 Too Many Requests	Problem Details	TS29573_N32_Handshake.yaml	Too many requests in a given amount of time ("rate limiting").			No
	500 Internal Server Error	Problem Details	TS29573_N32_Handshake.yaml	1. DB Access Fail during execution of handshake request in server side. 2. Any other internal error in server during execution of handshake request.			Yes
	503 Service Unavailable	null	TS29573_N32_Handshake.yaml	When sepp backend service is down.			Yes
	501 Not Implemented	Problem Details		When capability mismatch happens.			Yes
	401 Unauthorized	Problem Details		When request received for unknown peer sepp.			Yes

Request	Response Code	ResponseBody	Reference	Use Case	Comment/ Remarks	Applicable	Scope(PI-E)
	403 Forbidden	Problem Details		When request received for disabled peer sepp.			Yes

N32f Plane Message Forwarding Response Codes

Pod Name	Response Code	ResponseBody	Use Case	Reference	Comment	Applicable	Scope
cN32f-Sepp	200 OK	Success Body	Success	TS29573_JOSEProtectedMessageForwarding.yaml			PI-E
	404 Not Found	Problem Details	N32f Context Not Found	3gpp Spec 29500			PI-E
	401 Unauthorised	Problem Details	Context Not Established	3gpp Spec 29500			PI-E
	401 Unauthorised	Problem Details	Security Capability apart from TLS	3gpp Spec 29500			PI-E
	400 Bad Request	Problem Details	3GPP-target-sbi-apiroot/ destination producer NF is absent	TS29573_JOSEProtectedMessageForwarding.yaml			PI-E
	503 Service Unavailable	Problem Details	Service Unavailable	TS29573_JOSEProtectedMessageForwarding.yaml			PI-E
pN32f-Sepp	400 Bad Request	Problem Details	3GPP-target-sbi-apiroot/ destination producer NF is absent	TS29573_JOSEProtectedMessageForwarding.yaml			PI-E
	503 Service Unavailable	Problem Details	Service Unavailable	TS29573_JOSEProtectedMessageForwarding.yaml			PI-E

Note:

All the Exceptions/Error response codes sent by the N32-EGW at consumer side and PLMN-EGW at Producer side will be same and we will be adding a Problem statement which contains the reason why it occured.