

# Alarm Support


- Feature Summary and Revision History, on page 1
- Feature Description, on page 1
- Configuring Alarm Support, on page 9


# Feature Summary and Revision History

## Summary Data

***Table 1: Summary Data***

| | |
|---|---|
| Applicable Product(s) or Functional Area | cnBNG |
| Applicable Platform(s) | SMI |
| Feature Default Setting | Disabled - Configuration Required |
| Related Documentation | Not Applicable |

## Revision History

***Table 2: Revision History***

| Revision Details | Release |
|---|---|
| First introduced. | 2022.02.0 |

# Feature Description

When an anomaly is detected, the system generates a notification called an alarm or alert. The system triggers an alarm or alert when the statistics crosses the specified threshold. The Cloud Native BNG Control Plane uses the Common Execution Environment (CEE) infrastructure to generate alarms and SNMP traps.

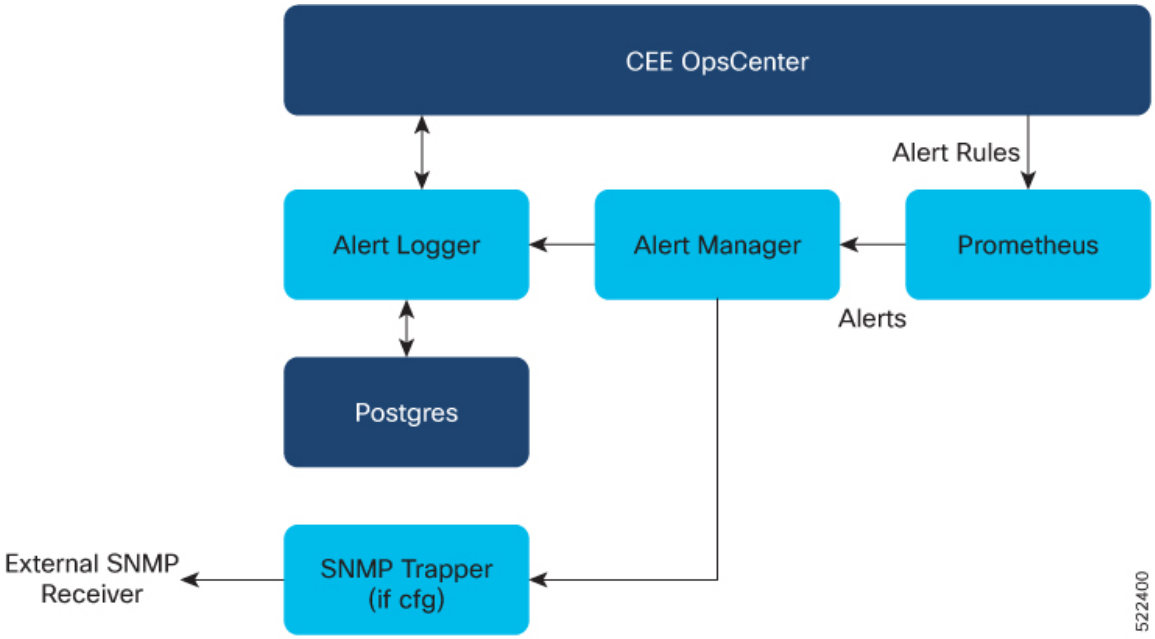
The Common Execution Environment (CEE) uses the Prometheus Alert Manager for alerting operations. The CEE YANG model - either through CLI or API - allows users to view the active alerts, silenced alerts, and alert history. Also, the applications can call the alert API directly to add or clear alerts. The Prometheus Alert Manager API (v2) is the standard API used.

The Prometheus Alerts Manager includes the following options:

- **Defining Alert Rules**: This option defines the types of alerts that the Alert Manager should trigger. Use the Prometheus Query Language (PromQL) to define the alerts.
- **Defining Alert Routing**: This option defines the action the Alert Manager should take after receiving the alerts. At present, the SNMP Trapper is supported as the outbound alerting. Also, the CEE provides an Alert Logger for storing the generated alerts.

The CNEE provides a set of predefined alerting rules regarding system health and Ops Center monitoring. For more details, see the "Alerts Reference" chapter in the *Ultra Cloud Core Subscriber Microservices Infrastructure Operations Guide*

The following figure depicts the components involved in the alerting mechanism.




# Supported Alarm Categories

The cnBNG CP supports the following alarm categories.

- **CP-UP Connectivity**—Alarms are generated if Control Plane (CP) - User Plane (UP) association fails, active, or inactive. Expressions can be formulated on CP defined UPF and Packet Forwarding Control Protocol (PFCP) metrics.
- **cnBNG Session**—Alarms can be generated if session bring-up or bring-down success rate is less than the specificied threshold, drop rate, and if subscriber limit is crossed. Expressions can be formulated on Session Manager (SM) and First Sign of Life (FSOL) metrics.

- **Accounting**—Alarms can be generated if accounting start, interim, and stop success rate is less than the specified certain threshold and so on. Expressions can be formulated on accounting metrics.
- **Radius** —Alarms can be generated for RADIUS server for the following events:
  - Active or Inactive server state
  - Statistics success, failure, or reject rates for Authorization and Accounting based on threshold
  - Change of Authorization (CoA) success or failure rates.

  Expressions can be formulated on RADIUS metrics.
- **IP Pool** —Alarms can be generated if IP pool allocation reaches the specified threshold. Expressions can be formulated on IPAM metrics.

# Alert Configuration Recommendations

Based on alarm categories, the following alert configurations are recommended.



**Note**
- The Threshold field is configurable as per requirement.
- The *interval-seconds* and *duration* can vary based on requirements.

## Application-based Alerts

Configure the following alerts to detect an application anomaly and trigger the alert or alarm.



**Note** These alerts are critical, therefore, it is recommended that these alerts are configured.

### RADIUS Authorization or Accounting Status

Use the following commands to configure an alert when the RADIUS authorization or accounting server is down.

```
alerts rules group RadiusEP
       interval-seconds 300
       rule Auth_Radius_Server_Down
       expression “sum by (namespace,
rad_svr_ip,rad_svr_port)(radius_server_status{rad_svr_port_type=\"Auth\"} < 1)”
       duration   5m
       severity   major
       type       “Processing Error Alarm”
       annotation summary
       value “Auth Radius Server: {{ $labels.rad_svr_ip}}, Port: {{ $labels.rad_svr_port
}} in namespace: {{ $labels.namespace }} is DOWN for more than 5min.”
       exit
       rule Acct_Radius_Server_Down
       expression “sum by (namespace,
radSvrIP,rad_svr_port)(radius_server_status{rad_svr_portType=\"Acct\"} < 1)”
       duration   5m
```

```
       severity   major
       type       "Processing Error Alarm"
       annotation summary
       value "Acct Radius Server: {{ $labels.rad_svr_ip}}, Port: {{ $labels.rad_svr_port
}} in namespace: {{ $labels.namespace }} is DOWN for more than 5min."
       exit
 exit
```

### User Plane Function Status

Use the following commands to configure an alert related to User Plane (UP) to Control Plane (CP) connectivity.

```
alerts rules group CpUpAssociation
       interval-seconds 300
       rule CpUpConnectionStatus
       expression "sum by (namespace, upf)(upf_status{status=~\"InActive\"}) > 0 )"
       duration   1m
       severity   major
       type       "Processing Error Alarm"
       annotation summary
       value "Upf {{ $labels.namespace }}/{{ $labels.upf }} is inactive for 1m"
       exit
 exit
```

### Subscriber Limit Threshold

Use the following commands to configure an alert when the session count crosses the specified threshold.

```
alerts rules group BngSession
       interval-seconds 300
       rule BngSubscriberLimit
       expression "sum by (namespace) ((avg(db_records_total{session_type=\"SM:PPPOE\"})
OR on() vector(0)) + (avg(db_records_total{session_type=\"SM:DHCP\"})  OR on() vector(0))
+ (avg(db_records_total{session_type=\"SM:LNS\"})  OR on() vector(0)) +
(avg(db_records_total{session_type=\"SM:LAC\"})  OR on() vector(0))) > THRESHOLD"
       severity   critical
       type       "Communications Alarm"
       annotation summary
       value "This alert is fired when session count rises above threshold."
exit
 exit
```

### System Overload Status

Use following commands to configure an alert if or when the system overloads.

```
alerts rules group BngSystemStatus
   interval-seconds 300
   rule BngOverload
   expression "sum by(component,level) (system_overload_status{level=~\"Critical|Crash\"})
 == true"
   duration   5m
   severity critical
   type "Communications Alarm"
   annotation summary
  value "This alert is fired when there is system overload as component {{ $labels.component
 }} has health level is {{ $labels.level }}."
   exit
exit
```

### IP Pool Consumption

Use the following commands to configure IP pool consumption alerts.

```
alerts rules group IPPool
   interval-seconds 300
   rule IPPoolConsumption
   expression "sum by (namespace,pool,addressType)(ipam_address_allocations_current)/sum
by (namespace,pool,address_type)(IPAM_address_pool_total) > THRESHOLD"
   duration 1m
   severity major
   type "Processing Error Alarm"
   annotation summary
   value "Pool: {{ $labels.pool }} AddressType: {{$labels.address_type}} in Namespace: {{
$labels.namespace }} has reached  THRESHOLD % of utilization"
   exit
exit
```

### PPPoE Session Limit Threshold

Use the following commands to configure an alert when the PPPoE session limit crosses the specified threshold.

```
alerts rules group PPPoESessionLimit
 rule PPPoESessionLimit
  expression
"((pppoe_session_limit_total{session_limit_count=\"SessionRejected\",session_limit_type=\"SessionMaxLimit\"}
 unless
pppoe_session_limit_total{session_limit_count=\"SessionRejected\",session_limit_type=\"SessionMaxLimit\"}
 offset 1m) OR
(increase(pppoe_session_limit_total{session_limit_count=\"SessionRejected\",session_limit_type=\"SessionMaxLimit\"}[1m])
 )) > 0"
  severity   critical
  type       "Communications Alarm"
  annotation summary
   value "PPPoE session limit crossed in last 1min."
  exit
 exit
exit
```

### L2TP Session Limit Threshold

Use the following commands to configure an alert when the L2TP session limit crosses the specified threshold.

```
alerts rules group L2TPSession
 rule SessionLimit
  expression "sum by (remote_hostname,upf_name)((l2tp_session_limit_total unless
l2tp_session_limit_total offset 1m) OR (increase(l2tp_session_limit_total[1m]) )) > 0"
  severity   critical
  type       "Communications Alarm"
  annotation summary
   value "Session Limit crossed for Tunnel: Routername {{$labels.upf_name}} Remote
{{$labels.remote_hostname}} !!!"
  exit
 exit
exit
```

## Use-Case Based Alerts

Configure the following alerts based on requirements.

### RADIUS Authorization Success Rate

Use the following commands to configure RADIUS authorization success rate alerts.

```
alerts rules group RadiusEP
       interval-seconds 300
       rule RadiusAuthSuccessRate
```

```
       expression "sum by (namespace)(increase(radius_requests_statistics{
rad_msg_code=\"AaaAuthReq\",rad_pkt_type=\"Rx\",rad_result=\"Success\"}[5m]))/sum by
(namespace)(increase(radius_requests_statistics{
rad_msg_code=\"AaaAuthReq\",rad_pkt_type=\"Tx\"}[5m])) < THRESHOLD"
       severity   major
       type       "Communications Alarm"
       annotation summary
       value "This alert is fired when the percentage of successful Radius Authentication
responses received is lesser than threshold"
       exit
 exit
```

### RADIUS Accounting Success Rate

Use the following commands to configure RADIUS accounting response success rate alerts.

```
alerts rules group RadiusEP
       interval-seconds 300
       rule RadiusAcctSuccessRate
       expression "sum by (namespace)(increase(radius_requests_statistics{
rad_msg_code=\"AaaAcctReq\",rad_pkt_type=\"Rx\",rad_result=\"Success\"}[5m]))/sum by
(namespace)(increase(radius_requests_statistics{
rad_msg_code=\"AaaAcctReq\",rad_pkt_type=\"Tx\"}[5m])) < THRESHOLD"
       severity   major
       type       "Communications Alarm"
       annotation summary
       value "This alert is fired when the percentage of successful Radius Accounting
responses received is lesser than threshold"
       exit
 exit
```

### Radius CoA Success Rate

Use the following commands to configure RADIUS Change of Authorization (CoA) success rate alerts.

```
alerts rules group RadiusEP
       interval-seconds 300
       rule RadiusCoaSuccessRate
       expression "sum by (namespace)(increase(radius_coa_dm_requests_statistics{
rad_msg_code=\"CoAACK\",rad_pkt_type=\"Tx\",rad_result=\"Success\"}[5m]))/sum by
(namespace)(increase(radius_coa_dm_requests_statistics{
rad_msg_code=\"CoARequest\",rad_pkt_type=\"Rx\"}[5m])) < THRESHOLD"
       severity   major
       type       "Communications Alarm"
       annotation summary
       value "This alert is fired when the percentage of successful Coa Ack received is
lesser than threshold"
       exit
 exit
```

### Accounting Start Success Rate

Use the following commands to configure accounting start success rate alerts.

```
alerts rules group Accounting
       interval-seconds 300
       rule AcctStartSuccessRate
       expression "sum by (namespace)(increase(accounting_message_total{
acct_type=\"Start\",status=\"Success\"}[5m]))/sum by
(namespace)(increase(accounting_message_total{ acct_type=\"Start\",status=\"Attempt\"}[5m]))
 < THRESHOLD"
       severity   major
       type       "Processing Error Alarm"
```

```
       annotation summary
      value "This alert is fired when the percentage of successful Accounting Start Responses
 received is lesser than threshold"
       exit
 exit
```

### Accounting Interim Success Rate

Use the following commands to configure accounting interim success rate alerts.

```
alerts rules group Accounting
       interval-seconds 300
       rule AcctInterimSuccessRate
       expression "sum by (namespace)(increase(accounting_message_total{
acct_type=\"Interim\",status=\"Success\"}[5m]))/sum by
(namespace)(increase(accounting_message_total{ acct_type=\"Interim\",status=\"Attempt\"}[5m]))
 < THRESHOLD"
       severity   major
       type       "Processing Error Alarm"
       annotation summary
       value "This alert is fired when the percentage of successful Accounting Interim
Responses received is lesser than threshold"
       exit
 exit
```

### Accounting Stop Success Rate

Use the following commands to configure accounting stop success rate alerts.

```
alerts rules group Accounting
       interval-seconds 300
       rule AcctStopSuccessRate
       expression "sum by (namespace)(increase(accounting_message_total{
acct_type=\"Stop\",status=\"Success\"}[5m]))/sum by
(namespace)(increase(accounting_message_total{ acct_type=\"Stop\",status=\"Attempt\"}[5m]))
 < THRESHOLD"
       severity   major
       type       "Processing Error Alarm"
       annotation summary
      value "This alert is fired when the percentage of successful Accounting Stop Responses
 received is lesser than threshold"
       exit
       exit
```

### N4 Session Creation Success Rate

Use the following commands to configure N4 session creation success rate alerts.

```
alerts rules group BngSession
       interval-seconds 300
       rule SessionCreateSuccessRate
       expression "sum by(namespace,upf_name)
(increase(bng_proto_udp_total{message_name=\"n4_session_establishment_res\",message_direction=\"inbound\",
 status=\"accepted\"}[5m]))/sum by (namespace,upf_name) (increase(bng_proto_udp_total{
message_name=\"n4_session_establishment_req\", message_direction=\"outbound\",
transport_type=\"origin\",status=\"accepted\"}[5m])) < THRESHOLD"
       severity   major
       type       "Communications Alarm"
       annotation summary
      value "This alert is fired when the percentage of successful Session Creation Responses
 received is lesser than expected threshold for upf: {{$labels.upf}}"
        exit
exit
```

### N4 Session Update Success Rate

Use the following commands to configure N4 session update success rate alerts.

```
alerts rules group BngSession
       interval-seconds 300
       rule SessionUpdateSuccessRate
       expression "sum by(namespace,upf)
(increase(bng_proto_udp_total{message_name=\"n4_session_modification_res\",message_direction=\"inbound\",
 status=\"accepted\"}[5m]))/sum by (namespace,upf) (increase(bng_proto_udp_total{
message_name=\"n4_session_modification_req\",  message_direction=\"outbound\",
transport_type=\"origin\",status=\"accepted\"}[5m])) < THRESHOLD"
       severity   major
       type       "Communications Alarm"
       annotation summary
      value "This alert is fired when the percentage of successful Session Update Responses
 received is lesser than expected threshold for upf: {{$labels.upf_name}}"
        exit
exit
```

### N4 Session Release Success Rate

Use the following commands to configure N4 session release success rate alerts.

```
alerts rules group BngSession
       interval-seconds 300
       rule SessionReleaseSuccessRate
       expression "sum by (namespace,upf_name)
(increase(bng_proto_udp_total{message_name=\"n4_session_deletion_res\",message_direction=\"inbound\",
 status=\"accepted\"}[5m]))/sum by (namespace,upf_name)(increase(bng_proto_udp_total{
message_name=\"n4_session_deletion_req\",  message_direction=\"outbound\",
transport_type=\"origin\",status=\"accepted\"}[5m])) < THRESHOLD"
       severity   major
       type       "Communications Alarm"
       annotation summary
      value "This alert is fired when the percentage of successful Session Release Responses
 received is lesser than expected threshold for upf: {{$labels.upf_name}}"
        exit
exit
```

### N4 Session Request Timeouts

Use the following commands to configure alerts to calculate the rate of N4 session requests that timeout awaiting response from the UP.

```
alerts rules group BngSession
       interval-seconds 300
       rule N4SessionReqTimeouts
       expression "sum by (namespace,
upf_name)(increase(bng_proto_udp_total{message_name=~\"n4_session_establishment_req|n4_session_modification_req|n4_session_deletion_req\",message_direction=\"outbound\",
 status=\"Timeout\"}[15m]))/sum by (namespace, upf_name)(increase(bng_proto_udp_total{
message_name=~\"n4_session_establishment_req|n4_session_modification_req|n4_session_deletion_req\",
  message_direction=\"outbound\", transport_type=\"origin\"}[15m])) > THRESHOLD"
       severity   major
       type       "Communications Alarm"
       annotation summary
       value "This alert is fired for upf {{$labels.upf_name}} as n4 session requests are
getting timeout for last 15mins"
        exit
exit
```

# Alert Routing to SNMP Trapper

The CNEE SNMP Trapper supports alert or alarm routing. Login to the CNEE Ops center to enable the SNMP Trapper because it is disabled by default. To enable SNMP traps, see Configuring SNMP Traps, on page 11.

The SNMP agent uses the Management Information Base (MIB) to handle SNMP trap notifications in the CISCO-CNEE-MIB.my. The CNEE generates two types of notifications with the following trap object identifiers (OID):

- **cneeFaultClearNotif**—The CNEE generates this notification when fault or alert gets cleared.
- **cneeFaultActiveNotif**—The CNEE generates this notification when fault or alert gets triggered.

For more details, see the "SMI MIB Reference" chapter in the *Ultra Cloud Core Subscriber Microservices Infrastructure Operations Guide*.

# Alert Routing to Alert Logger

The Alert Logger allows to view active and silenced alerts or history of alerts triggered from CNEE Ops Center using show commands. Alert routing is enabled by default.

For more details, see the "Viewing Alert Logger" section in the "Common Execution Environment" chapter of the *Ultra Cloud Core Common Execution Environment Configuration and Administration Guide*.

# Alarm Severity Levels

The alert or alarm severity levels are as follows:

- Critical
- Major
- Minor
- Warning

All severity level alerts are routed to the SNMP Trapper. The CNEE does not have a mechanism to route only critical or major alerts or alarms to the SNMP Trapper while configuring alerts rules. To address this requirement, configure the following CLI command to avoid routing minor or warning alerts or alarms to the SNMP Trapper.

```
cee# alerts silence add matchers { name severity isRegex true value
\"warning\|minor\" }
```

Use the following command to view silenced alerts or alarms.

```
show alerts silenced { summary | detail }
```

# Configuring Alarm Support

This section describes how to configure Alarm Support on cnBNG CP.

Configuring Alarm Support involves the following procedures:

# Configuring Alert Rules

Use the following commands to configure alert rules:

```
config
   alerts rules group alert_group_name
   interval-seconds seconds
   rule rule_name
      expression promql_expression
      duration duration
      severity severity_level
      type alert-type
      annotation annotation_name
      value annotation_value
      exit
   exit
```

**NOTES**:

- **alerts rules**: Specifies the Prometheus alerting rules.
- **group** *alert_group_name*: Specifies the Prometheus alerting rule group. One alert group can have multiple lists of rules. *alert-group-name* is the name of the alert group. The alert-group-name must be a string in the range of 0 to 64 characters.
- **interval-seconds** *seconds*: Specifies the evaluation interval of the rule group in seconds.
- **rule** *rule_name*: Specifies the alerting rule definition. *rule_name* is the name of the rule.
- **expression** *promql_expression*: Specifies the PromQL alerting rule expression. *promql_expression* is the alert rule query expressed in PromQL syntax.
- **duration** *duration*: Specifies the duration of a true condition before it is considered true. *duration* is the time interval before the alert is triggered.
- **severity** *severity_level*: Specifies the severity of the alert. *severity-level* is the severity level of the alert. The severity levels are critical, major, minor, and warning.
- **type** *alert_type*: Specifies the type of the alert. *alert_type* is the user-defined alert type. For example, Communications Alarm, Environmental Alarm, Equipment Alarm, Indeterminate Integrity Violation Alarm, Operational Violation Alarm, Physical Violation Alarm, Processing Error Alarm, Quality of Service Alarm, Security Service Alarm, Mechanism Violation Alarm, or Time Domain Violation Alarm.
- **annotation** *annotation_name*: Specifies the annotation to attach to the alerts. *annotation_name* is the name of the annotation.
- **value** *annotation_value*: Specifies the annotation value. *annotation_value* is the value of the annotation.

The following example configures an alert, which is triggered when the percentage of Unified Data Management (UDM) responses is less than the specified threshold limit.

**Example:**

```
config terminal
 alerts rules group BNGUDMchk_incr
 interval-seconds 300
 rule BNGUDMchk_incr
 expression "sum(increase(bng_restep_http_msg_total{nf_type=\"udm\",
```

```
message_direction=\"outbound\", response_status=~\"2..\"}[3m])) /
sum(increase(smf_restep_http_msg_total{nf_type=\"udm\", message_direction=\"outbound\"}[3m]))
 < 0.95"
  severity major
  type "Communications Alarm"
  annotation summary
  value "This alert is fired when the percentage of UDM responses is less than threshold"
  exit
 exit
exit
```

You can view the configured alert using the **show running-config alerts** command.

**Example:**

The following example displays the alerts configured in the running configuration:

```
 show running-config alerts
  interval-seconds 300
  rule SMFUDMchk_incr
  expression "sum(increase(smf_restep_http_msg_total{nf_type=\"udm\",
message_direction=\"outbound\", response_status=~\"2..\"}[3m])) /
sum(increase(smf_restep_http_msg_total{nf_type=\"udm\", message_direction=\"outbound\"}[3m]))
 < 0.95"
  severity major
  type "Communications Alarm"
  annotation summary
  value "This alert is fired when the percentage of UDM responses is less than threshold"

  exit
 exit
exit
```

# Configuring SNMP Traps

Use the following commands to configure or enable SNMP Traps.

```
config
   snmp-trapper enable true
   snmp-trapper { v2c-target target | v3-target target |
   v3-engine-id source_engine_id }
      community [ community_string ]
      port [ port ]
      exit
   snmp-trapper source-ip-routes [ vip_options ]
   exit
```

```
config
  no snmp-trapper enable
  exit
```

**NOTES**:

- **snmp-trapper enable true**: Enables the SNMP trapper parameters.
- **v2c-target|v3-target [** *target* **]**: Specifies the list of SNMP v2c and v3 trap receivers.
- **community [** *community_string* **]**: Specifies the SNMP trap receiver community.
- **v3-engine-id** *source_engine_id*: Specifies the source engine ID for the v3 traps. *source_engine_id* must be an hexagonal string. For instance, 80004f.

- **port [** *port* **]**: Specifies the SNMP trap receiver port. *port* must be an integer in the range of 0 through 65535. The default value is 162.
- **source-ip-routes [** *vip_options* **]**: Enables binding to source IP for SNMP routing. *vip_options* specifies the virtual IP (VIP) address. The different options for virtual IP addresses include:
  - **default-external-vip**: Specifies the default external VIP for source IP routing.
  - **internal-vip**: Specifies the internal VIP for source IP routing.
  - **source-external-vips**: Specifies the external VIP per namespace.
- **no snmp-trapper enable**: Disables SNMP traps.