Alarms and Notifications for ETSI LCM Operations

ETSI Alarms

ESC provides alarms and notifications to the NFVO. The NFVO has to subscribe to these alarms and notifications and send requests to ESC.

The NFVO can receive information about the alarms in the following ways:

Query All Alarms

The NFVO can get a list of all the alarms from the alarms resource.

Method Type:

```
GET
```

VNFM Endpoint:

```
/vnffm/v1/alarms
```

HTTP Request Header:

```
Accept:application/json
```

For example, to query all alarms with the event type as ENVIRONMENTAL_ALARM

Method Type:

```
GET
```

VNFM Endpoint:

```
http://localhost:8250/vnffm/v1/alarms?eventType="ENVIRONMENTAL_ALARM"
```

HTTP Request Headers:

```
Accept:application/json
```

While querying for multiple alarms, the NFVO can use the URI query parameters to filter the results. The following attribute names are supported for the URI query of the alarms:

- id
- managedObjectId
- rootCauseFaultyResource.faultyResourceType
- eventType
- perceivedSeverity
- probableCause

Note The URI query parameters are for querying multiple alarms only.

Query an Individual Alarm

The NFVO can query a particular alarm from the *alarmId* resource.

Method Type:

```
GET
```

VNFM Endpoint

```
/vnffm/v1/alarms/{alarmId}
```

HTTP Request Header:

```
Accept:application/json
```

Modify an Individual Alarm

To modify an alarm, the NFVO must send a PATCH request to the *AlarmModifications* resource.

Method Type:

```
PATCH
```

VNFM Endpoint:

HTTP Request Header:

```
Content-Type: application/merge-patch+json
If-Match: ETag value
```

Note **If-Match:** is optional. If specified, its value is validated against the ETag value stored against the VNF (and returned from a single VNF query).

The supported attribute is ackState, and the supported attribute values are ACKNOWLEDGED and UNACKNOWLEDGED. All other modification payloads are rejected.

VNF Failure and Load Alarms

The following alarms are created for ETSI VNF failure and load notifications.

- Failure Alarm—ESC generates the failure alarms when one of the compute resources within the VNF becomes unreachable based upon the VM_ALIVE KPI configuration of the VFND. For more information, see ETSI Failure and Load Notifications for VNFs.

  Example:

  Method Type

  ```
  POST
  ```

  VNFM Endpoint

  ```
  /vnffm/v1/extension/alarms
  ```

  HTTP Request Header

  ```
  Content-Type:application/json
  ```

  Request Payload:

  ```
  {
    "externalAlarmId" : "26bf1e3d-cefa-4f59-88ea-210a29358a5c", #generated value
    "alarmSource" : "MONA", #hard-coded
    "managedObjectId" : "08733ef2-319b-46ce-9d8d-95730306bd1a", #external_deployment_id
   "rootCauseFaultyResource" : "chrimann-dep_g1_0_212da327-0573-421b-ae37-057f6b1a6aef",
  #vm_name
    "alarmRaisedTime" : "$timestamp", #generated value
    "ackState" : "UNACKNOWLEDGED", #hard-coded
    "perceivedSeverity" : "CRITICAL", #hard-coded
    "eventTime" : "2018-05-08T00:59:32.571+00:00", #do we have the eventTime?
    "eventType" : "EQUIPMENT_ALARM", #hard-coded
    "faultType" : "COMPUTE", #hard-coded
    "probableCause" : "VM_MANUAL_RECOVERY_NEEDED", #event_name
    "isRootCause" : "TRUE", #hard-coded
    "links" : {
      "objectInstance" :
  "{http_scheme}://{api_root}/vnflcm/v2/vnf_instances/08733ef2-319b-46ce-9d8d-95730306bd1a"

    }
  }
  ```

- Load Alarm—ESC generates the load alarms when one of the compute resources within the VNF becomes over or under loaded based upon the related KPI configurations of the VFND. ESC creates these alarms after receiving notifications from the NFVO. For more information, see ETSI Failure and Load Notifications for VNFs.

  Example:

  Method Type

  ```
  POST
  ```

  VNFM Endpoint

  ```
  /vnffm/v1/extension/alarms
  ```

  HTTP Request Header

  ```
  Content-Type:application/json
  ```

  Request Payload

Alarm Extensions

ETSI provides an extension for the alarms to interact with the third party tools. You must send a POST request to create the alarms.

Method Type

```
POST
```

VNFM Endpoint

```
/vnffm/v1/extension/alarms
```

HTTP Request Header

```
Content-Type:application/json
```

Request Payload

```
[admin@davwebst-esc-4-2-0-49-keep ETSI]$ cat CreateAlarm.json
{
    "id": "alm87032",
    "externalAlarmId": "ext-id-xx11214",
    "managedObjectId": "930fb087-c1b9-4660-bec8-2a8d97dc1df5",
    "rootCauseFaultyResource": {
        "id": "fres7629",
        "faultyResource": {
            "resourceId": "res7727"
        },
        "faultyResourceType": "NETWORK"
    },
    "alarmRaisedTime": "2018-05-30T13:55:15.645000+00",
    "ackState": "UNACKNOWLEDGED",
    "perceivedSeverity": "MAJOR",
    "eventTime": "2018-05-30T13:55:15.645000+00",
    "eventType": "ENVIRONMENTAL_ALARM",
    "probableCause": "Server room overheating",
    "isRootCause": "false",
    "vnfInstanceIds": [
        "res-a3023a03-fc73-430a-a983-5e9439011e45"

}
```

Subscribing to Notifications

The NFVO can subscribe to the ETSI notifications related to fault management from ESC.

Create a Subscription

The NFVO sends a POST request to subscribe to the notifications.

Method Type:

```
POST
```

VNFM Endpoint:

```
/vnffm/v1
```

Response Payload:

```
{
  "filter" : {
```

```
    "notificationTypes" : [
      "AlarmNotification",
      "AlarmClearedNotification",
      "AlarmListRebuiltNotification"
    ],
    "perceivedSeverities" : [
      "CRITICAL",
      "MAJOR"
    ]
  },
  "callbackUri" : "https://nfvo.endpoint.listener",
  "authentication" : {
    "authType" : "BASIC",
    "paramsBasic" : {
      "userName" : "admin",
      "password" : "pass123"
    }
  }
}
```

This creates a new subscription resource and a new identifier. The callbackUri is the only mandatory parameter. The others are all optional. You can verify if the callbackuri is valid and reachable by sending a GET request.

Query all Subscriptions

The NFVO can query information about its subscriptions by sending a GET request to the *subscriptions* resource.

Method Type:

```
GET
```

VNFM Endpoint:

```
/vnffm/v1/subscriptions
```

HTTP Request Header:

```
Accept:application/json
```

For example, to query all alert subscriptions, when the callbackUri is

```
http://10.10.1.44:9202/alerts/subscriptions/callback
```

```
GET
```

VNFM Endpoint

```
http://localhost:8250/vnffm/v1/subscriptions?callbackUri="http://10.10.1.44:9202/alerts/subscriptions/callback"
```

HTTP Request Header

```
Accept:application/json
```

The NFVO can use the URI query parameters to filter the results. The following attribute names are supported for the URI query of the subscriptions:

- id
- filter
- callbackUri

Note The URI query parameters are for querying multiple subscriptions only.

Query an Individual Subscription

You must know the subscription ID to query an individual subscription.

Method Type:

```
GET
```

VNFM Endpoint:

```
/vnffm/v1/subscriptions/{subscriptionId}
```

HTTP Request Header:

```
Accept:application/json
```

Delete a Subscription

You can delete a subscription if the NFVO does not need it. Send a delete request to the individual subscription.

Method Type:

```
DELETE
```

VNFM Endpoint:

```
/vnffm/v1/subscriptions/{subscriptionId}
```

HTTP Request Header:

```
http://localhost:8250/vnffm/v1/subscriptions/682791f8-34ad-487e-811a-553036bf49b2
```

ETSI Failure and Load Notifications for VNFs

ESC generates notifications for the following:

- **VM Failure**

  The NFVO receives failure notifications from ESC, when the VMs within the deployed VNFs fail. After receiving the notifications, alarms are generated. For more information on alarms, see ETSI Alarms, on page 1.

  The NFVO must subscribe to the ESC for notifications.

  Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<esc_event xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <deployment_name>sample-dep</deployment_name>
  <event_name>MY_VM_UNDERLOADED</event_name>
  <event_type>VM_UNDERLOADED</event_type>
  <external_deployment_id>e911eecf-5f3f-456c-9c80-d99aca2416da</external_deployment_id>

  <external_tenant_id>etsi_tenant</external_tenant_id>
  <internal_deployment_id>99f7629f-98d3-40f5-ad68-7addcfe07006</internal_deployment_id>

  <internal_tenant_id>etsi_tenant</internal_tenant_id>
  <vm_source>
```

```
<generated_vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</generated_vm_name>

    <interfaces>
      <addresses>
        <address>
          <address_id>0</address_id>
          <gateway>172.16.0.1</gateway>
          <ip_address>172.16.0.0</ip_address>
          <dhcp_enabled>true</dhcp_enabled>
          <prefix>20</prefix>
          <subnet>365a0884-fdb3-424c-afe9-2deb3b39baae</subnet>
        </address>
      </addresses>
      <network_uuid>c7fafeca-aa53-4349-9b60-1f4b92605420</network_uuid>
      <mac_address>fa:16:3e:38:1d:6c</mac_address>
      <nic_id>0</nic_id>
      <port_forwarding/>
      <port_uuid>0aeb9585-5190-4f3b-b1aa-495e09c56b7d</port_uuid>
      <security_groups/>
      <subnet_uuid>none</subnet_uuid>
      <type>virtual</type>

<vim_interface_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vim_interface_name>

    </interfaces>
    <vim_id>default_openstack_vim</vim_id>
    <vim_project>admin</vim_project>
    <vim_project_id>c12f013306d849e5b1bbf257c54d5891</vim_project_id>
    <host_uuid>6b8cf361c5ff08a5a886e26f591b8087dadcf2d2b34fb3b5d2772a8d</host_uuid>
    <host_name>my-server</host_name>
    <vm_uuid>9fea3fe7-9417-4734-b962-b24340941ef3</vm_uuid>
    <vm_group_name>vm1</vm_group_name>
    <vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vm_name>
  </vm_source>
</esc_event>
```

- **VM Overload and Underload**

  Similarly, the NFVO receives an overload or underload notification for a VM.

  If scaling is not enabled automatically, ESC generates a notification depending on the state of the VM.

  Examples:

```
<?xml version="1.0" encoding="UTF-8"?>
<esc_event xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <deployment_name>sample-dep</deployment_name>
  <event_name>MY_VM_UNDERLOADED</event_name>
  <event_type>VM_UNDERLOADED</event_type>
  <external_deployment_id>e911eecf-5f3f-456c-9c80-d99aca2416da</external_deployment_id>

  <external_tenant_id>etsi_tenant</external_tenant_id>
  <internal_deployment_id>99f7629f-98d3-40f5-ad68-7addcfe07006</internal_deployment_id>

  <internal_tenant_id>etsi_tenant</internal_tenant_id>
  <vm_source>

<generated_vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</generated_vm_name>

    <interfaces>
      <addresses>
        <address>
          <address_id>0</address_id>
          <gateway>172.16.0.1</gateway>
```

```
            <ip_address>172.16.0.0</ip_address>
            <dhcp_enabled>true</dhcp_enabled>
            <prefix>20</prefix>
            <subnet>365a0884-fdb3-424c-afe9-2deb3b39baae</subnet>
          </address>
        </addresses>
        <network_uuid>c7fafeca-aa53-4349-9b60-1f4b92605420</network_uuid>
        <mac_address>fa:16:3e:38:1d:6c</mac_address>
        <nic_id>0</nic_id>
        <port_forwarding/>
        <port_uuid>0aeb9585-5190-4f3b-b1aa-495e09c56b7d</port_uuid>
        <security_groups/>
        <subnet_uuid>none</subnet_uuid>
        <type>virtual</type>

<vim_interface_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vim_interface_name>

    </interfaces>
    <vim_id>default_openstack_vim</vim_id>
    <vim_project>admin</vim_project>
    <vim_project_id>c12f013306d849e5b1bbf257c54d5891</vim_project_id>
    <host_uuid>6b8cf361c5ff08a5a886e26f591b8087dadcf2d2b34fb3b5d2772a8d</host_uuid>
    <host_name>my-server</host_name>
    <vm_uuid>9fea3fe7-9417-4734-b962-b24340941ef3</vm_uuid>
    <vm_group_name>vm1</vm_group_name>
    <vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vm_name>
  </vm_source>
</esc_event>
```

VM underload example:

```
<?xml version="1.0" encoding="UTF-8"?>
<esc_event xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <deployment_name>sample-dep</deployment_name>
  <event_name>MY_VM_OVERLOADED</event_name>
  <event_type>VM_OVERLOADED</event_type>
  <external_deployment_id>e911eecf-5f3f-456c-9c80-d99aca2416da</external_deployment_id>

  <external_tenant_id>etsi_tenant</external_tenant_id>
  <internal_deployment_id>99f7629f-98d3-40f5-ad68-7addcfe07006</internal_deployment_id>

  <internal_tenant_id>etsi_tenant</internal_tenant_id>
  <vm_source>

<generated_vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</generated_vm_name>

    <interfaces>
      <addresses>
        <address>
          <address_id>0</address_id>
          <gateway>172.16.0.1</gateway>
          <ip_address>172.16.0.0</ip_address>
          <dhcp_enabled>true</dhcp_enabled>
          <prefix>20</prefix>
          <subnet>365a0884-fdb3-424c-afe9-2deb3b39baae</subnet>
        </address>
      </addresses>
      <network_uuid>c7fafeca-aa53-4349-9b60-1f4b92605420</network_uuid>
      <mac_address>fa:16:3e:38:1d:6c</mac_address>
      <nic_id>0</nic_id>
      <port_forwarding/>
      <port_uuid>0aeb9585-5190-4f3b-b1aa-495e09c56b7d</port_uuid>
      <security_groups/>
      <subnet_uuid>none</subnet_uuid>
      <type>virtual</type>
```

```
<vim_interface_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vim_interface_name>

    </interfaces>
    <vim_id>default_openstack_vim</vim_id>
    <vim_project>admin</vim_project>
    <vim_project_id>c12f013306d849e5b1bbf257c54d5891</vim_project_id>
    <host_uuid>6b8cf361c5ff08a5a886e26f591b8087dadcf2d2b34fb3b5d2772a8d</host_uuid>
    <host_name>my-server</host_name>
    <vm_uuid>9fea3fe7-9417-4734-b962-b24340941ef3</vm_uuid>
    <vm_group_name>vm1</vm_group_name>
    <vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vm_name>
  </vm_source>
</esc_event>
```

Note ETSI only generates an alarm for a VNFC that exists in instantiatedVnfInfo.vnfcResourceInfo when the notification from ESC is received.

Auto-Scaling VNFs Using KPI Instructions

ESC can auto-scale VMs using the KPI instructions. The scaling workflow begins when the VNF instance is in the instantiated state.The NFVO enables and disables the auto-scaling while modifying *isAutoscaleEnabled* configurable property of the VNF.

Following are the events that trigger an ETSI-compliant auto-scale, which requires an instigation of a *ScaleVnfToLevelRequest:* functionality.

- **Overload and Underload**

  If the state of a VM changes and it is under or overloaded, ESC gets a notification to determine if the scaling is automatically enabled. If it is not, ESC generates a notification towards the ETSI-VNFM component to check the VNF's state.

  The following example shows underloaded notification from ESC:

```
Headers:
  esc-status-code = 200
  esc-status-message = VM [sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de]
underloaded.
Body:
<?xml version="1.0" encoding="UTF-8"?>
<esc_event xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <deployment_name>sample-dep</deployment_name>
  <event_name>MY_VM_UNDERLOADED</event_name>
  <event_type>VM_UNDERLOADED</event_type>
  <external_deployment_id>e911eecf-5f3f-456c-9c80-d99aca2416da</external_deployment_id>

  <external_tenant_id>etsi_tenant</external_tenant_id>
  <internal_deployment_id>99f7629f-98d3-40f5-ad68-7addcfe07006</internal_deployment_id>

  <internal_tenant_id>etsi_tenant</internal_tenant_id>
  <vm_source>

<generated_vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</generated_vm_name>

    <interfaces>
      <addresses>
        <address>
```

```
            <address_id>0</address_id>
            <gateway>172.24.0.1</gateway>
            <ip_address>172.24.0.37</ip_address>
            <dhcp_enabled>true</dhcp_enabled>
            <prefix>20</prefix>
            <subnet>365a0884-fdb3-424c-afe9-2deb3b39baae</subnet>
          </address>
        </addresses>
        <network_uuid>c7fafeca-aa53-4349-9b60-1f4b92605420</network_uuid>
        <mac_address>fa:16:3e:38:1d:6c</mac_address>
        <nic_id>0</nic_id>
        <port_forwarding/>
        <port_uuid>0aeb9585-5190-4f3b-b1aa-495e09c56b7d</port_uuid>
        <security_groups/>
        <subnet_uuid>none</subnet_uuid>
        <type>virtual</type>

<vim_interface_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vim_interface_name>

    </interfaces>
    <vim_id>default_openstack_vim</vim_id>
    <vim_project>admin</vim_project>
    <vim_project_id>c12f013306d849e5b1bbf257c54d5891</vim_project_id>
    <host_uuid>6b8cf361c5ff08a5a886e26f591b8087dadcf2d2b34fb3b5d2772a8d</host_uuid>
    <host_name>my-server-65</host_name>
    <vm_uuid>9fea3fe7-9417-4734-b962-b24340941ef3</vm_uuid>
    <vm_group_name>vm1</vm_group_name>
    <vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vm_name>
  </vm_source>
</esc_event>
```

The following example shows overloaded notification from ESC:

```
Headers:
  esc-status-code = 200
  esc-status-message = VM [sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de]
overloaded.
Body:
<?xml version="1.0" encoding="UTF-8"?>
<esc_event xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <deployment_name>sample-dep</deployment_name>
  <event_name>MY_VM_OVERLOADED</event_name>
  <event_type>VM_OVERLOADED</event_type>
  <external_deployment_id>e911eecf-5f3f-456c-9c80-d99aca2416da</external_deployment_id>

  <external_tenant_id>etsi_tenant</external_tenant_id>
  <internal_deployment_id>99f7629f-98d3-40f5-ad68-7addcfe07006</internal_deployment_id>

  <internal_tenant_id>etsi_tenant</internal_tenant_id>
  <vm_source>

<generated_vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</generated_vm_name>

    <interfaces>
      <addresses>
        <address>
          <address_id>0</address_id>
          <gateway>172.24.0.1</gateway>
          <ip_address>172.24.0.37</ip_address>
          <dhcp_enabled>true</dhcp_enabled>
          <prefix>20</prefix>
          <subnet>365a0884-fdb3-424c-afe9-2deb3b39baae</subnet>
        </address>
      </addresses>
      <network_uuid>c7fafeca-aa53-4349-9b60-1f4b92605420</network_uuid>
```

```
      <mac_address>fa:16:3e:38:1d:6c</mac_address>
      <nic_id>0</nic_id>
      <port_forwarding/>
      <port_uuid>0aeb9585-5190-4f3b-b1aa-495e09c56b7d</port_uuid>
      <security_groups/>
      <subnet_uuid>none</subnet_uuid>
      <type>virtual</type>

<vim_interface_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vim_interface_name>

    </interfaces>
    <vim_id>default_openstack_vim</vim_id>
    <vim_project>admin</vim_project>
    <vim_project_id>c12f013306d849e5b1bbf257c54d5891</vim_project_id>
    <host_uuid>6b8cf361c5ff08a5a886e26f591b8087dadcf2d2b34fb3b5d2772a8d</host_uuid>
    <host_name>my-server-65</host_name>
    <vm_uuid>9fea3fe7-9417-4734-b962-b24340941ef3</vm_uuid>
    <vm_group_name>vm1</vm_group_name>
    <vm_name>sample-dep_vm1_0_fbc3da46-e0c6-40dc-91c8-70b1a88857de</vm_name>
  </vm_source>
</esc_event>
```

- **VNFD**

  The VNFD notification contains the instructions for the scale action required for *isAutoscaleEnabled* configurable property of the VNF operation flow.

  If the scaling is not enabled automatically, you can instigate the manual LCM operations using the KPI instructions. It is instigated by processing the ESC notification stream. You must validate the notification once you receive the KPI events.

  You must take the following actions:

  - Find the matching VNF instance
  - Validate that the appropriate configuration property is set to enable the automated operation

  If the validation passes then you can request to instigate the operation flow to generate the appropriate operation occurrence and associated notifications. For scaling, any specified KPI data determines the scaling parameters. The properties file includes the following new attributes:

```
 external.scaling.decision = 1
   #external.scaling.window = 120
   external.healing.decision = 1
   #external.healing.window = 120
```

- **VnfInstance resource**

  The VNFD determines the scale level using the current scaleStatus. The processing of the request determines the number of VMs to request from ESCManager. The request only supplies a relative number of increments (SCALE_IN or SCALE_OUT).

  You can call the *ScaleVnfToLevel* endpoint with the following payload, using *vnfInstanceId* from the vnfInstance resource of the VNF to be scaled.

  Ensure that the VnfLcmOpOcc.isAutomaticInvocation is set to true.

  The following eexample shows JSON payload:

```
{
  /* "instantiationLevelId":"id111", */
  "scaleInfo": [
    { "aspectId":"processing", "scaleLevel":"3" },
```

```
      { "aspectId":"database",   "scaleLevel":"2" }
    ]
    "additionalParams": {
      "password": "pass1234",
      "username": "admin"
    },
    "action": "scale_to_level"
  }
```

Healing VNFs Using KPI Instructions

ESC can auto-heal VMs using the KPI instructions. The NFVO enables and disables the auto-healing while modifying *isAutohealEnabled* configurable property of the VNF.

The *isAutohealEnabled* property permits to enable (TRUE)/disable (FALSE) the auto-healing functionality.

-