Monitoring Virtual Network Functions

Monitoring the VNFs

After deploying VNFs, they are monitored periodically to check their health and workload. Monitoring is based on the definition of metrics within the KPI section of the deployment data model. As described in the KPIs section the metric type determined not only the variable to monitor, but also the collector action to be executed. ESC allows you to define the metrics to be monitored and the actions that needs to be executed when the conditions are met. These metrics and actions are defined in the *deployment datamodel*. Several monitoring methods are used to monitor the VNFs. You can monitor the following:

- VM aliveness
- VM variables for Disk usage, Memory, CPU, Network throughput
- ICMP message on the VM monitoring interface.

Pre-requisites for Monitoring

The following pre-requisites must be met for the VMs to be monitored by ESC:

- Monitoring is enabled for VMs that are successfully deployed. The deployed VMs must be alive.
- KPI must be configured in the data model with the monitoring parameters.

Monitoring and Action Execution Engine

Monitoring is based on the definition of metrics within the KPI section of the deployment datamodel. As described in the KPIs section the metric type determines not only the variable to monitor, but also the collector action to be executed. The monitoring engine comprises of metrics and actions.

1. Metrics
2. Actions

The metrics and actions <metadata> section describes the properties or entries controlling the programmable aspect of the engine.

Metrics Section

The metrics section is as follows:

```
<metrics>
     <metric>
     <name>{metric name}name>
           <type>{metric type}type>
           <metaData>
           <type>{monitoring engine action type}</type>
         <properties>
         <property>
         <name></name>
        <value></value>
       </property>
       : : : : : : : :
      <properties/>
     </metaData>
    </metric or action>
     : : : : : : : : :
<metrics>
```

Table 1: Metric Section Description

Tag name	Description	Values
name	A user defined metric name. The metric name must be unique.	
type	Dynamic mapping supported type.	MONITOR_SUCCESS_FAILURE MONITOR_THRESHOLD MONITOR_COMPUTE_THRESHOLD

Metric Metadata Section

The purpose of the metadata section is to provide information specific to the monitoring solution.

Table 2: Metric Metadata Section

Tag Name	Description	Values
type	The action type, values are a one to one mapping with MONA supported actions.	custom_script custom_script_threshold snmp_get_threshold
properties	A container for a list of properties (name/value) that will be passed to selected action. The properties are defined by the list of expected monitoring and actions attributes.	Properties are based on the selected action type.

Actions Section

The actions section is as follows:

```
<actions>
           <action>
           <name>{action name}name>
           <type>{action type}type>
           <metaData>
           <type>{monitoring engine action type}</type>
           <properties>
           <property>
           <name></name>
           <value></value>
          </property>
           : : : : : : : :
           <properties/>
          </metaData>
         </action>
           : : : : : : : : :
          <actions>
```

Table 3: Actions

Tag Name	Description	Values
name	A user defined action name. The action name must be unique.	One of the main requirements is also to have the chosen name prefixed with TRUE or FALSE to allow mapping between ESC data model rule and dynamic actions, just for MONITOR_SUCCESS_FAILURE.
type	Supported type.	ESC_POST_EVENT SCRIPT CUSTOM_SCRIPT

Actions Metadata Section

The purpose of the metadata section is to provide information specific to the monitoring solution.

Table 4: Action Metadata section

Tag Name	Description	Values
type	The action type, values are a one to one mapping with monitoring and actions engine supported actions.	icmp_ping icmp4_ping icmp6_ping esc_post_event script custom_script snmp_get snmp_get_threshold
properties	A container for a list of properties (name/value) that will be passed to selected action. The properties are defined by the list of expected monitoring and action attributes.	Properties are based on the selected action type.

For more details see the KPIs, Rules and Dynamic Mapping APIs section.

Table 5: Supported Action Types

Type	Properties and their description
icmp_ping	• ip_address • enable_events_after_success: Boolean controlling when MONA will start forwarding events notifications. If set to true, notification will be forwarded only after the first transition to success. • timeOut: Default set to 5 seconds
icmpv4_ping	• ip_address • enable_events_after_success: Boolean controlling when MONA will start forwarding events notifications. If set to true, notification will be forwarded only after the first transition to success. • timeOut: Default set to 5 seconds

Type	Properties and their description
icmpv6_ping	• ip_address • enable_events_after_success: Boolean controlling when MONA will start forwarding events notifications. If set to true, notification will be forwarded only after the first transition to success. • timeOut: Default set to 5 seconds
script	• script_filename: Full path to the script to be executed (The script has to be located on the ESC VM). • wait_for_script: Boolean controlling if the action is waiting for the completion of the script. (Not actually exercised)
custom_script	script_filename: Full path to the script to be executed (The script has to be located on the ESC Manager VM).
custom_script_threshold	• script_filename: Full path to the script to be executed (The script has to be located on the ESC Manager VM). • threshold
post_esc_event	• esc_url • vm_external_id • vm_name • esc_event • event_name
snmp_get	• target_oid , agent_address, IP Address of the SNMP agent (IPV4/IPV6 is supported) • agent_port: Port used by the SNMP Agent. • agent_protocol: Protocol used by the SNMP Agent (tcp/udp). • Community: SNMP v2c community string used by the SNMP agent

Type	Properties and their description
snmp_get_threshold	• target_oid: Object Identifier that will be used for the threshold comparison. • agent_address: IP Address of the SNMP agent (IPV4/IPV6 is supported). • agent_port: Port used by the SNMP Agent. • agent_protocol: Protocol used by the SNMP Agent (tcp/udp). • community: SNMP v2c community string used by the SNMP agent
snmp_get_threshold_ratio	• oid_total_value: Object Identifier that will be used to represent the current for the ratio/percentage computation. • oid_current_value: Object Identifier that will be used to represent the current for the ratio/percentage computation. Algorithm to be used for the computation of the percentage/ratio. We are currently supporting two algorithms: COMPUTE_TOTAL_CURRENT_BASED , COMPUTE_TOTAL_AVAILABILITY_BASED. • agent_address: IP Address of the SNMP agent (IPV4/IPV6 is supported). • agent_port: Port used by the SNMP Agent. • agent_protocol: Protocol used by the SNMP Agent (tcp/udp). • community: SNMP v2c community string used by the SNMP agent.

Properties and Runtime Parameter Injection

The properties list passed to the selected action type supports the capabilities to automatically inject runtime value for some selected parameters. For example, runtime value of the virtual machine ip_address or its name can be passed automatically as arguments to the selected action.

Following are some of the parameters that can be passed to the scripts at the time of execution. Parameter value is populated at runtime only if :

- the parameter is a supported one, and
- its value is empty within the dynamic-mappings.xml file.

Otherwise, the value defined within the script is passed as is.

Table below shows the parameters passed during runtime.

esc_url	The URL of the Elastic Services Controller.
vm_external_id	The external id of the managed VM.
vm_name	The name of the managed VM.
vm_mac_address	The mac address of the managed VM.
vm_external_host_id	The VM external host Identifier.
vm_external_host_name	The VM external host name.
vm_group_name	The VM group name.
ip_address	The VM IP Address.
event_name	The ESC event name.

Note The properties list passed to the selected action, is not bound by the parameters in the action type. A script designer can define its own parameters. However, the values have to be provided.

Monitoring Methods

ESC uses several monitoring methods to monitor the VNFs. You must configure the KPI data model for the monitoring methods.

ICMP Ping Monitoring

Ping monitoring assess the liveliness or reachability of a VNF.

If a VM is unreachable the healing of the VM is triggered. At every defined interval, ESC polls the metric value and sends alarms whenever needed. The number of polls, metric value, and other configuration are set in the KPI datamodel.

SNMP Monitoring

In SNMP Monitoring, load of the VM such as memory usage and CPU in a given period is monitored. The SNMP Get operation is used to assess the liveliness or reachability of a VNF. In this monitoring method, only the success or failure is monitored.

SNMP Threshold Monitoring

In SNMP threshold monitoring, you can set the upper and lower threshold levels in the kpi section of the data model. Actions are performed based on the upper and lower threshold levels.

Custom Monitoring

In ESC 2.1 or earlier, the Dynamic Mapping XML is required to map the actions and metrics defined in the datamodel to the valid actions and metrics available in the monitoring agent. The file is stored on the ESC VM and is modified using a text editor. This method is error prone and modification for an HA pair requires

to take place on both the primary and secondary VMs. ESC 2.2 or later does not have an *esc-dynamic-mapping* directory and *dynamic_mappings.xml* file. The CRUD operations for mapping the actions and the metrics is now available through REST API in ESC. For more information, see KPIs, Rules and Metrics.

Monitoring a VM

Cisco Elastic Services Controller monitors the VM to detect any erroneous condition. ESC uses one of its monitoring methods to detect actions on a VM, and passes this information to the rules service for processing. The monitoring request comes from the northbound client along with VNF deployment requests.

There are two sections in the datamodel xml file which define the events and rules: KPI and Rule.

Based on the monitors and actions, rules are triggered.

```
<kpi>
   <event_name>VM_ALIVE</event_name>
   <metric_value>50</metric_value>
   <metric_cond>GT</metric_cond>
   <metric_type>UINT32</metric_type>
   <metric_occurrences_true>3</metric_occurrences_true>
   <metric_occurrences_false>3</metric_occurrences_false>
   <metric_collector>
      <type>ICMPPing</type>
      <nicid>0</nicid>
      <poll_frequency>15</poll_frequency>
      <polling_unit>seconds</polling_unit>
      <continuous_alarm>false</continuous_alarm>
   </metric_collector>
</kpi>
```

In the example above, an event is sent to check whether the VM is alive. The VM is pinged at regular intervals, and based on the result VM_ALIVE event is sent to the rules engine along with the details of the VM.

The rules engine receives events from the monitoring engine. The rules engine can handle simple to complex events. Based on the event received an action is triggered.

If the VM is not alive, based on the event the actions defined in the <rule> section are triggered. This can be found in the dep.xml datamodel.

```
<rules>
   <admin_rules>
      <rule>
         <event_name>VM_ALIVE</event_name>
         <action>ALWAYS log</action>
         <action>FALSE recover autohealing</action>
         <action>TRUE servicebooted.sh</action>
      </rule>
   </admin_rules>
</rules>
```

The rules section describes the actions to be executed once a monitoring event has been detected. The dynamic mapping API drives the rules based on keywords.

In the above example, the following actions are performed based on the given condition:

- ALWAYS log: Whether the event is pingable or not, the details are logged.

- TRUE servicebooted.sh: The action identified by this keyword in the dynamic mapping API is triggered when the VM moves from a non-pingable to a pingable state. The serviceboot script informs ESC that the VM is alive allowing it to transition the VMs state.
- FALSE recover autohealing: The action identified by this keyword will be triggered and the VM will be recovered without the administrator's intervention.

Monitoring log files for troubleshooting are available at `/var/log/mona`.

Monitoring the VM Network Status

When using ICMP ping monitoring, if ESC receives a VM Down event, the healing workflow attempts to recover the VM with the recovery policy. If there is an issue with the network interface or IP route from ESC to the VNF. For example, if the gateway is down, it might trigger the VM Down event incorrectly, which leads to an unnecessary recovery.

The check interface function does further scan to the network route by checking the health status of all the network interfaces and the operation state of the gateway. If there is any problem in the network environment, it assumes the VNF is alive.

The VM_NETWORK_STATE event is sent to northbound if ESC detects a network issue or if any existing issue is fixed (autohealing).

The following failure notification is sent to northbound:

```
16:13:15,567 14-Mar-2018 WARN  ===== SEND NOTIFICATION STARTS =====
16:13:15,567 14-Mar-2018 WARN  Type: VM_NETWORK_STATE
16:13:15,567 14-Mar-2018 WARN  Status: FAILURE
16:13:15,567 14-Mar-2018 WARN  Status Code: 500
16:13:15,567 14-Mar-2018 WARN  Status Msg: Warning: VM
[NG_G1_0_46fdcf70-f4ea-4289-ae79-08674e7d6f42] has a network problem: Network interface not
 healthy, please check.
16:13:15,567 14-Mar-2018 WARN  Tenant: tenant2
16:13:15,567 14-Mar-2018 WARN  Deployment ID: 455d2407-9dda-4203-95b0-724c4a651720
16:13:15,567 14-Mar-2018 WARN  Deployment name: NG
16:13:15,567 14-Mar-2018 WARN  VM group name: G1
16:13:15,567 14-Mar-2018 WARN  VM Source:
16:13:15,567 14-Mar-2018 WARN      VM ID: 4bee016a-6b30-43ff-a249-157a07d9b4db
16:13:15,567 14-Mar-2018 WARN      VM Name: NG_G1_0_46fdcf70-f4ea-4289-ae79-08674e7d6f42
16:13:15,568 14-Mar-2018 WARN      VM Name (Generated):
NG_G1_0_46fdcf70-f4ea-4289-ae79-08674e7d6f42
16:13:15,568 14-Mar-2018 WARN      VIM ID: default_openstack_vim
16:13:15,568 14-Mar-2018 WARN      VIM Project: tenant2
16:13:15,568 14-Mar-2018 WARN      VIM Project ID: 62afb63cd28647a7b526123cac1ba605
16:13:15,568 14-Mar-2018 WARN      Host ID:
b83004159a46c20bc8383927c2231067bb0c1905b4b4c28475653190
16:13:15,568 14-Mar-2018 WARN      Host Name: my-server-50
16:13:15,568 14-Mar-2018 WARN  =====  SEND NOTIFICATION ENDS  =====
```

The following success notification is sent to northbound when the network problem is fixed.

```
16:13:19,141 14-Mar-2018 INFO  ===== SEND NOTIFICATION STARTS =====
16:13:19,141 14-Mar-2018 INFO  Type: VM_NETWORK_STATE
16:13:19,142 14-Mar-2018 INFO  Status: SUCCESS
16:13:19,142 14-Mar-2018 INFO  Status Code: 200
16:13:19,142 14-Mar-2018 INFO  Status Msg: Network of VM
[NG_G1_0_46fdcf70-f4ea-4289-ae79-08674e7d6f42] has been restored.
16:13:19,142 14-Mar-2018 INFO  Tenant: tenant2
16:13:19,142 14-Mar-2018 INFO  Deployment ID: 455d2407-9dda-4203-95b0-724c4a651720
16:13:19,142 14-Mar-2018 INFO  Deployment name: NG
```

```
16:13:19,142 14-Mar-2018 INFO  VM group name: G1
16:13:19,142 14-Mar-2018 INFO  VM Source:
16:13:19,142 14-Mar-2018 INFO      VM ID: 4bee016a-6b30-43ff-a249-157a07d9b4db
16:13:19,142 14-Mar-2018 INFO      VM Name: NG_G1_0_46fdcf70-f4ea-4289-ae79-08674e7d6f42
16:13:19,142 14-Mar-2018 INFO      VM Name (Generated):
NG_G1_0_46fdcf70-f4ea-4289-ae79-08674e7d6f42
16:13:19,142 14-Mar-2018 INFO      VIM ID: default_openstack_vim
16:13:19,143 14-Mar-2018 INFO      VIM Project: tenant2
16:13:19,143 14-Mar-2018 INFO      VIM Project ID: 62afb63cd28647a7b526123cac1ba605
16:13:19,143 14-Mar-2018 INFO      Host ID:
b83004159a46c20bc8383927c2231067bb0c1905b4b4c28475653190
16:13:19,143 14-Mar-2018 INFO      Host Name: my-server-50
16:13:19,143 14-Mar-2018 INFO  =====  SEND NOTIFICATION ENDS  =====
```

For information on monitoring VNFs using ETSI API, see the *Cisco Elastic Services Controller ETSI NFV MANO Guide*.

Notification for VM Monitoring Status

ESC sends the VM_MONITORING_STATUS notification, if the missing monitoring script causes timer expiration when monitors are reset after ESC switchover. VM_MONITOR_STATUS notification is sent to NB. ESC doesn't monitor the VM, it fails to enter the recovery process. To enable monitoring after the failure, you must disable, and then enable the monitoring.

Notifications

```
WARN ===== SEND NOTIFICATION STARTS =====
WARN Type: VM_MONITORING_STATUS
WARN Status: FAILURE
WARN Status Code: 500
WARN Status Msg: No response from the monitor
WARN Tenant: tenant
WARN Deployment ID: 02cc4018-e4e3-4974-884a-f9fee17d7040
WARN Deployment name: dep
WARN VM group name: g1
WARN VM Source:
WARN VM ID: 6aa98b79-9d35-442a-9abb-f611e6316083
WARN VM Name: dep_g1_0_7fdae2a6-5095-4071-9c50-fb80c0e6b80e
WARN VM Name (Generated): dep_g1_0_7fdae2a6-5095-4071-9c50-fb80c0e6b80e
WARN VIM ID: default_openstack_vim
WARN VIM Project: tenant
WARN VIM Project ID: 33bf6768e45445da87feed838b248849
WARN Host ID: 79e4104d1d33de80aab13205b1e3c61d64aa4b61230c8b7b064b2891
WARN Host Name: my-ucs-62
WARN ===== SEND NOTIFICATION ENDS =====
```

Monitoring Operations

You can set and unset monitoring of VMs using RESTful interface.

A payload is required to monitoring VMs:

```
POST ESCManager/v0/{internal_tenant_id}/deployments/vm/{vm_name}
```

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<vm_operation xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
   <operation>enable_monitoring</operation>
   <force>false</force>
</vm_operation>
```

You must mention enable_monitoring to set VM monitoring, and disable_monitoring to unset VM monitoring in the operation field.

Note When a user reboots the VM from the ESC portal, the monitoring is automatically enabled.