Monitoring Virtual Network Functions

- Monitoring Virtual Network Functions Using ETSI API, on page 1
- VM Monitoring Operations, on page 4

Monitoring Virtual Network Functions Using ETSI API

During the deployment of a VNF, metrics must be defined to instruct the ESC monitoring agent component (MONA) how to determine if the VNF is healthy. The definition of metrics is within the Key Performance Indicator (KPI) section of the VNFD and allow MONA to periodically monitor the VNF to check its health and workload, defined on a per-VNFC basis. Actions are then associated with these KPIs and executed when the appropriate conditions are met.

There are several built-in monitoring methods such as ICMP Ping and SNMP. Some of the metrics to monitor on the constituent VNFCs include:

- reachability
- resource usage (such as CPU, memory, disk and network throughput)

The following pre-requisites must be met for the deployed VNFCs to be monitored:

- The deployed VNFCs must be alive
- Monitoring is enabled
- KPIs must be configured

Example:

```
vdu1:
      type: tosca.nodes.nfv.Vdu.Compute
      properties:
        name: Example VDU1
        description: Example VDU
        boot_order:
          - boot1-volume
        configurable_properties:
          additional_vnfc_configurable_properties:
            vim_flavor: Automation-Cirros-Flavor
            bootup_time: 1800
            vm_name_override: my-vdu-1
            recovery_action: REBOOT_THEN_REDEPLOY
            recovery_wait_time: 100
```

```
      monitor_on_error: false
      max_retries: 2
      kpi_data:
        VM_ALIVE-1:
          event_name: 'VM_ALIVE-1'
            metric_value: 1
            metric_cond: 'GT'
            metric_type: 'UINT32'
            metric_occurrences_true: 1
            metric_occurrences_false: 30
            metric_collector:
              type: 'ICMPPing'
              nicid: 1
              poll_frequency: 10
              polling_unit: 'seconds'
              continuous_alarm: false
      admin_rules:
        VM_ALIVE-1:
          event_name: 'VM_ALIVE-1'
          action:
            - 'ALWAYS log'
            - 'FALSE recover autohealing'
            - 'TRUE esc_vm_alive_notification'
      placement_type: zone
      placement_target: nova
      placement_enforcement: strict
      vendor_section:
        cisco_esc:
          config_data:
            example.txt:
              file: ../Files/Scripts/example.txt
                variables:
                  DOMAIN_NAME: { get_input: DOMAIN_NAME }
                  NAME_SERVER: { get_input: NAME_SERVER }
                  VIP_ADDR: { get_input: VIP_ADDR }
                  VIP_PREFIX: { get_input: VIP_PREFIX }
    vdu_profile:
      min_number_of_instances: 1
      max_number_of_instances: 1
           capabilities:
    virtual_compute:
      properties:
        virtual_cpu:
          num_virtual_cpu: 8
        virtual_memory:
          virtual_mem_size: 16
  requirements:
    - virtual_storage:  cdr1-volume
    - virtual_storage:  boot1-volume
```

The kpi_data shown above is the default KPI required that is required in all deployments at a minimum so that the VM_ALIVE message is generated to tell ESC Manager that the VNFC has been deployed successfully; it consists of the KPI, how it is collected and the actions to be executed when the KPI is met.

Cisco data structure properties

Data Type	Property Name	Description	Values
cisco.datatypes.nfv.data.Kpi	KPI label	Unique user-defined KPI name	Any

Data Type	Property Name	Description	Values
cisco.datatypes.nfv.data.Kpi	monitoring_agent	Specifies the monitoring agent for a VNF, for example, local or distributed MONA	URI for the agent
cisco.datatypes.nfv.data.Kpi	event_name		
cisco.datatypes.nfv.data.Kpi	metric_value		
cisco.datatypes.nfv.data.Kpi	metric_cond		
cisco.datatypes.nfv.data.Kpi	metric_type		
cisco.datatypes.nfv.data.Kpi	metric_occurrences_true		
cisco.datatypes.nfv.data.Kpi	metric_occurrences_false		
cisco.datatypes.nfv.metric.Collector	type	See the NETCONF API Guide	See the NETCONF API Guide
cisco.datatypes.nfv.metric.Collector	nicid		
cisco.datatypes.nfv.metric.Collector	poll_frequency		
cisco.datatypes.nfv.metric.Collector	polling_unit		
cisco.datatypes.nfv.metric.Collector	continuous_alarm		
cisco.datatypes.nfv.metric.Collector	property_list		
cisco.datatypes.nfv.data.Admin_rules	Rule label	Unique user-defined name	Any
cisco.datatypes.nfv.data.Admin_rules	event_name	This value must match a Kpi event_name	
cisco.datatypes.nfv.data.Admin_rules	action		
cisco.datatypes.nfv.data.Admin_rules	property_list		

The following extract is from the ETSI properties file, which allows the subscription to an extension notification type:

```
# For notificationType "InfrastructureOperationOccurrenceNotification"
subscription.notifications.infra.filter.operationTypes=MONITORING_MIGRATION
subscription.notifications.infra.filter.operationStates=COMPLETED,FAILED_TEMP,FAILED,ROLLED_BACK
subscription.notifications.infra.callbackUri=http://<nfvoHost>:<nfvoPort>/monitoring/migration/notification
 # Full URL where the notification will be sent
subscription.notifications.infra.authentication.authType=BASIC # or OAUTH2_CLIENT_CREDENTIALS

# Basic Auth credentials (based on authType)
subscription.notifications.infra.authentication.paramsBasic.userName=nfvo
subscription.notifications.infra.authentication.paramsBasic.password=mypw
```

```
# Alternatively, OAUTH 2.0 credentials (based on authType)
#subscription.notifications.infra.authentication.paramsOauth2ClientCredentials.clientId=
#subscription.notifications.infra.authentication.paramsOauth2ClientCredentials.clientPassword=
#subscription.notifications.infra.authentication.paramsOauth2ClientCredentials.tokenEndpoint=
```

Note If the previous properties are not set, then these notifications are sent to the subscribers where all notifications types are applicable.

For more information on KPIs and Rules, see the *Cisco Elastic Services Controller User Guide*.

VM Monitoring Operations

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

The operation is defaults to asynchronous, you must set `sync.supported=true` to use this functionaity in a synchronous way.

A payload is required to monitor VMs:

Method type

`POST`

VNFM Endpoint

Example for SOL003:

`{apiRoot}/or_vnfm/vnflcm/v2/ext/vnf_instances/{vnfInstanceId}/monitoring/operations`

Example for SOL002:

`{apiRoot}/ve_vnfm/vnflcm/v2/ext/vnf_instances/{vnfInstanceId}/monitoring/operations`

To start and stop monitoring operation on a specified VM, set the `vnfcInstanceIds`

with payload:

```
{
    "vnfcInstanceIds": ["vnfcInstanceId1","vnfcInstanceId2",...,"vnfcInstaceIdN"],    ##
optional
    "operation": "ENABLE_MONITOR",                                                    ##
mandatory ENABLE_MONITOR, DISABLE_MONITOR, REBOOT
    "additionalParams": []                                                            ##
optional - for future use :-)
}
```

To start and stop monitoring operation on the entire VNF, do not set the `vnfcInstanceIds`.

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 ETSI interface, the monitoring is automatically enabled.

Notification for VM Monitoring Status

ETSI NFV MANO provides status notifications for VM Monitoring. You can enable, disable, and reboot the VMs on a particular VNF or on a particular VM of a VNF using payload.

ETSI NFV MANO sends the following `[notifications-per-operation]` when setting, unsetting, or rebooting the VMs:

```
[notifications-per-operation]
-------------------------------------
VM_MONITOR_SET notification when enabling a monitor
VM_MONITOR_UNSET notification when disabling a monitor
VM_REBOOTED notification when rebooting
```