Monitoring, Metrics and Events Requirements High Level List

This wiki is a WIP. Please feel free to modify this page with relevant information

The intention of this wiki is to list the requirements for the monitoring agent as well as the requirements for the metrics and events that should be monitored or collected within the NFVI. In addition to the metrics/events collected about the NFVI, some information about the monitoring process (the process which collects the information and metrics) itself is also required.

This list should be developed in conjunction with the Doctor (Faults) and VES Projects in OPNFV.

This wiki heavily references The ETSI NFV draft specification titled “Network Functions Virtualisation (NFV); Testing; NFVI Compute and Network Metrics Specification” which can be found here: **TST008** (**please consult the latest version, and leave a comment if this link is broken, ETSI seems to move it frequently).**

Distinction between metrics and events

For the purposes of Platform Service Assurance, it's important to distinguish between metrics and events as well as how they are measured (from a timing perspective).

A Metric is a (standard) definition of a quantity describing the performance and/or reliability of a monitored function, which has an intended utility and is carefully specified to convey the exact meaning of the measured value. A measured value of a metric is produced in an assessment of a monitored function according to a method of measurement. For example the number of dropped packets for a networking interface is a metric.

An Event is defined as an important state change in a monitored function. The monitor system is notified that an event has occurred using a message with a standard format. The Event notification describes the significant aspects of the event, such as the name and ID of the monitored function, the type of event, and the time the event occurred. For example, an event notification would take place if the link status of a networking device on a compute node suddenly changes from up to down on a node hosting VNFs in an NFV deployment.

Collector requirements:

Polling vs Event capture for the monitoring agent

In the context of the monitoring agent polling the subsystem it's querying. Both polling and event driven updates should be supported with events driven updates being the preferred model to use. This depends on the subsystem you are monitoring, default would be to leverage event based systems where they exist, but polling should be supported as a configuration option that can be selected by the end user.

In the context of the VIM polling the monitoring Agent:

- Fault events should always use a push model, and the mechanism over which events are sent needs to be reliable.
- Telemetry, can be polled or pushed (could be polled to spread the load on the collection side).

- Network (over)load should be taken into consideration as regards which model to use (push vs pull), you don't want to destabilize the network. push is more scalable overall and preferred for fault management.

Collector configuration

Should be able to dynamically:

- Enable/disable/or restart resource monitoring
- Get values/notifications
- Get capabilities
- Get the list of metrics being collected
- flush the list of metrics
- Set thresholds for resources
- blacklist resources
- support some sort of buffering mechanism, and should be able to configure
- get the timing information for the agent and do aTiming sync if required.

Collector Time stamping support

The Time sent with a sample should be: time stamp at which the value was collected.

Currently there are 2 scenarios as regards time stamps with samples:

1. Where the subsystem we are reading from CAN provide us with the “incident” time (time at which an event occurred) and the collector can provide us with the collection time (time at which a sample was collected): In this case we have the “incident” time for the sample/event and the time when a collector retrieves the sample...

2. where the subsystem we are reading from CANNOT provide us with the “incident” time only the collection time: In this case we only have the time for when the collector retrieves the sample.

The recommendation for collectors where possible is to collect both incident time and collection time and send them with a sample.

For collectd there is only 1 time stamp field. The recommendation is to send the collection time in the collectd time stamp field for values and notifications- BUT where detection time is available to send it in the metadata.

Events Requirements

Timing

Events must be detected within a 10ms interval in order to allow for a 40ms failover time.

Metrics/Events Format

It's important to define a common format that can be used for the list of identified metrics and events that should be monitored/collected in the NFVI.

- + Name
- + Where the Metric/Event is collected (e.g., the measurement point, such as Host/Guest/Both)
- + Parameters (input factors or variables)
- + Scope of measurement coverage
- + Unit(s) of measure or associated severities
- Definition
- Method of Measurement
- Sources of Error
- Comments

In addition to the measurement result, items marked "+" should either be available for collection, or reported with the measurement result.

Information to be collected in conjunction with NFVI Metrics/Events

It's essential to collect some information about the environment that is being monitored as well as the monitoring process(es) themselves in order to associate the mertrics/events with the relevant host.

Host information:

Each host in a deployment should have a Unique identifier that distinguishes it from all other hosts. A UUID can be used in this case.

Monitoring Process information:

Each monitoring process in a deployment should have a Unique Process identifier.

Each monitoring process in a deployment should support the following events:

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments
Heartbeat/ping	Host/Guest (where the monitoring process is running)	ping frequency and size of packet	liveliness check	N/A	Heartbeat/ping to check liveliness of monitoring process	external ping	false alarm for host due to network interruption	

Each monitoring process in a deployment should support the following Metrics:

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments
`write_queue/queue_length`	Host/Guest (where the monitoring process is running)	measurement frequency	The monitoring application being used		The number of metrics currently in the write queue.			
`write_dropped`	Host/Guest (where the monitoring process is running)	measurement frequency	The monitoring application being used		The number of metrics dropped due to a queue length limitation.			
`cache_size`	Host/Guest (where the monitoring process is running)	measurement frequency	The monitoring application being used		The number of elements in the metric cache			
CPU utilization	Host/Guest (where the monitoring process is running)	measurement frequency, interrupt frequency, set of execution contexts, time of measurement	The CPUs that are being used by the monitoring application	Nanoseconds or percentage of total CPU utilization	The CPU utilization of the monitoring process	kernel interrupt to read current execution context	short-lived contexts may come and go between interrupts	see section 6 of **TST008**
Memory Utilization	Host/Guest (where the monitoring process is running)	Time of measurement, total memory available, swap space configured	The Memory that is being used by the monitoring application	Kibibytes	The amount of physical RAM, in kibibytes, used by the monitoring application	memory management reports current values at time of measurement		see section 8 of **TST008**

NFVI Other/Additional Information

BIOS information

NFVI Events

What about entire node and switch failures? In terms of service affecting priority, host and switch failures are at the top as they can affect the most VMs / Containers / VNFs...

While the status of switches and hosts might be the domain of services that have a system-wide view, a host-resident component might be part of the monitoring functionality.

Compute

At a **minimum** the following **events** should be monitored:

- Machine check exceptions (System, Processor, Memory...) [**TODO**: Break this down further]
    - DIMM corrected and uncorrected Errors

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments
MCEs	Host		Memory, CPU, IO		Machine Check Exception	using mcelog		

PCIe Errors	Host							

Networking

At a **minimum** the following **events** should be monitored for a Networking interface:

- Link Status
- Dropped Receive Packets – An increasing count could indicate the failure or service interruption of an upstream processes.

vSwitch liveliness

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments
Link Status								
vSwitch Status (liveliness)								
Packet Processing Core Status								

Storage

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments

NFVI Metrics

Compute

At a **minimum** the following **metrics** should be collected:

- CPU utilization **TODO**: Break this down further]
- vCPU utilization **TODO**: Break this down further]
- Memory utilization **TODO**: Break this down further]
- vMemory utilization **TODO**: Break this down further]
- Cache utilization
  - Hits
  - Misses
  - Instructions per clock (IPC)
  - Last level cache utilization
  - Memory Bandwidth utilization
- Platform Metrics (thermals, fan-speed) [**TODO**: Break this down further]

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments
cpu_idle	Host		The host CPUs, individually or total usage summed across all CPUs	nanoseconds or percentage	Time the host CPU spends idle.			see CPU Utilization above, and section 6 of **TST008**
cpu_nice	Host		The host CPUs, individually or total usage summed across all CPUs	nanoseconds or percentage	Time the host CPU spent running user space processes that have been niced. The priority level a user space process can be tweaked by adjusting its *niceness*.			see CPU Utilization above, and section 6 of **TST008**
cpu_interrupt	Host		The host CPUs, individually or total usage summed across all CPUs	nanoseconds or percentage	Time the CPU has spent servicing (hardware) interrupts.			see CPU Utilization above, and section 6 of **TST008**
cpu_softirq	Host		The host	nanosecond	Time spent handling interrupts that are synthesized, and			see CPU

			CPUs, individually or total usage summed across all CPUs	s or percentage	almost as important as Hardware interrupts (above). "In current kernels there are ten softirq vectors defined; two for tasklet processing, two for networking, two for the block layer, two for timers, and one each for the scheduler and read-copy-update processing. The kernel maintains a per-CPU bitmask indicating which softirqs need processing at any given time." [Ref]			Utilization above, and section 6 of **TST008**
cpu_steal	Host		The host CPUs, individually or total usage summed across all CPUs	nanoseconds or percentage	CPU steal is a measure of the fraction of time that a machine is in a state of "involuntary wait." It is time for which the kernel cannot otherwise account in one of the traditional classifications like user, system, or idle. It is time that went missing, from the perspective of the kernel.			see CPU Utilization above, and section 6 of **TST008**
cpu_system	Host		The host CPUs, individually or total usage summed across all CPUs	nanoseconds or percentage	Time that the CPU spent running the kernel.			see CPU Utilization above, and section 6 of **TST008**
cpu_user	Host		The host CPUs, individually or total usage summed across all CPUs	nanoseconds or percentage	Time CPU spends running un-niced user space processes.			see CPU Utilization above, and section 6 of **TST008**
cpu_wait	Host		The host CPUs, individually or total usage summed across all CPUs	nanoseconds or percentage	The time the CPU spends idle while waiting for an I/O operation to complete			see CPU Utilization above, and section 6 of **TST008**
total_vcpu_utilization	Host		The host CPUs used by a guest, total usage summed across all CPUs	nanoseconds or percentage	The total utilization summed across all execution contexts (except Idle) and all CPUs in Scope.			see CPU Utilization above, and section 6 of **TST008**

Networking

At a **minimum** the following **metrics** should be collected for a Networking interface:

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments
Total Packets received								see section 7 of **TST008**
Total Packets transmitted								see section 7 of **TST008**
Total Octets received								see section 7 of **TST008**
Total Octets transmitted								see section 7 of **TST008**
Total Error frames received								see section 7 of **TST008**
Total Errors when attempting to transmit a frame								see section 7 of **TST008**
Broadcast Packets								
Multicast Packet								
Average bitrate								
Average latency								
RX Packets dropped								
TX packets dropped								

Networking MIBs

Where possible the metrics, events and information should be supported for the following Networking MIBs:

MIB Name	RFC	Description
IF-MIB	RFC2863	Network interface sub-layers
EtherLike-MIB	RFC3635	Ethernet like network interfaces
IP-MIB	RFC4293	IP and ICMP without routing info
IP-FORWARD_MIB	RFC4292	CIDR multipath IP routes
TCP-MIB	RFC4022	TCP stack counters and info
UDP-MIB	RFC4133	UDP counters and info
IPV6 MIBs	RFC2465 RFC2466 RFC2452 RFC2454	IPv6 equivalents
SCTP-MIB	RFC3873	SCTP protocol
UCD-IPFWACC-MIB		IP firewall accounting firewall rules

Virtual Switch Reporting

- Per interface (stats and info mentioned in the tables above) from Open vSwitch/Open vSwitch on DPDK/ VPP should be collected and exposed.
- sflow, Netflow/IPFIX flow telemetry should be supported, collected and exposed.

Storage

Note: collectd plugins **df** and **disk** can help here.

Name	Collection location	Parameters	Scope of coverage	Unit(s) of measure	Definition	Method of Measurement	Sources of Error	Comments
Latency (read and write)	host	host name, storage resource name, data retaining policy (persistent or ephemeral), type (block, object, file), size in GB	specified storage resource	milliseconds	The average amount of time to perform a R/W operation, in milliseconds	Comparison of counter readings over the observation interval	simultaneity of readings	Need update of **TST008**
IOPS (read and write)	host	host name, storage resource name, data retaining policy (persistent or ephemeral), type (block, object, file), size in GB	specified storage resource	operations per second	The average rate of performing R/W in IO operations per second	Comparison of counter readings over the observation interval	simultaneity of readings	Need update of **TST008**
Throughput (read and write)	host	host name, storage resource name, data retaining policy (persistent or ephemeral), type (block, object, file), size in GB	specified storage resource	bytes per second	The average rate of performing R/W operations in Bytes per second	Comparison of counter readings over the observation interval	simultaneity of readings	Need update of **TST008**
Disk Utilization	host	host name, storage resource name, data retaining policy (persistent or ephemeral), type (block, object, file), size in GB	specified storage resource	bytes	The amount of free, used, and reserved disk space at the time of the reading, in bytes (reported individually).	readings in all three categories	simultaneity of readings, disk space usage can be very dynamic.	Need update of **TST008**

The host CPUs, individually or total usage summed across all CPUs