ETSI GS NFV-TST 008 V3.3.1 (2020-06)

GROUP SPECIFICATION

Network Functions Virtualisation (NFV) Release 3; Testing; NFVI Compute and Network Metrics Specification

Reference

RGS/NFV-TST008ed331

Keywords

metrics, network, NFV, NFVI, testing

Contents

Intellectual Property Rights

Foreword

This Group Specification (GS) has been produced by ETSI Industry Specification Group (ISG) Network Functions Virtualisation (NFV).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

Although many metrics for the performance and utilization of the Network Function Virtualisation Infrastructure (NFVI) components have been in wide use for many years, there were no independent specifications to support consistent metric development and interpretation. The present document provides the needed specifications for key NFVI metrics.

1 Scope

The present document specifies detailed and vendor-agnostic key operational performance metrics at different layers of the NFVI, especially processor usage and network interface usage metrics. These metrics are expected to serve as references for processed and time-aggregated measurement values for performance management information that traverses the Or-Vi and/or Vi-Vnfm reference points of the NFV architectural framework. The present document contains normative provisions.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

Not applicable.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI GS NFV-INF 003 (V1.1.1) (12-2014): "Network Functions Virtualisation (NFV); Infrastructure; Compute Domain".

[i.2] Linux™/UNIX system programming training, Linux man-pages: "TOP(1)".

NOTE: Available at http://man7.org/linux/man-pages/man1/top.1.html#2._SUMMARY%C2%A0Display.

[i.3] O'Reilly Linux™ Dev Center: "Exploring the /proc/net/ Directory".

NOTE: Available at https://www.linuxtoday.com/infrastructure/2000112300806NWHLSW.

[i.4] RHEL™ 6.8 Deployment Guide: "E.2.18. /proc/meminfo".

NOTE: Available at https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Deployment_Guide/s2-proc-meminfo.html.

[i.5] ETSI GS NFV 003: "Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV".

[i.6] IETF RFC 7348: "Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks".

[i.7] "free(1) Linux User Commands", published by man7.org.

NOTE: Available at http://man7.org/linux/man-pages/man1/free.1.html.

[i.8] collectd codebase, file: "/src/memory.c", published by GitHub, Inc.

NOTE: Available at https://github.com/collectd/collectd/blob/collectd-5.7/src/memory.c#L325.

[i.9] ETSI GS NFV-IFA 003 (V2.4.1): "Network Functions Virtualisation (NFV) Release 2; Acceleration Technologies; vSwitch Benchmarking and Acceleration Specification".

[i.10] Debian Wiki: "Hugepages".

NOTE: Available at https://wiki.debian.org/Hugepages.

[i.11] ETSI GS NFV-INF 010 (V.1.1.1) (12-2014): "Network Functions Virtualisation (NFV); Service Quality Metrics".

[i.12] "Understanding CPU Steal Time - when should you be worried?", Scout application performance monitoring blog, Derek Haynes, July 25th, 2013.

NOTE: Available at https://scoutapm.com/blog/understanding-cpu-steal-time-when-should-you-be-worried.

[i.13] "Understanding Linux Container Scheduling", Squarespace Engineering blog, June 27th, 2017.

NOTE: Available at https://engineering.squarespace.com/blog/2017/understanding-linux-container-scheduling.

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI GS NFV 003 [i.5] apply.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI GS NFV 003 [i.5] and the following apply:

CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
HZ	Hertz of the system clock, an operating system parameter
KiB	Kibibytes
NFVI	Network Functions Virtualisation Infrastructure
OS	Operating System
OSC	Operating System Container
RAM	Random Access Memory
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VXLAN	Virtual eXtensible Local Area Network

NOTE: IETF RFC 7348 [i.6].

4 Time and Time Intervals for Metrics

Coherent compute domains [i.1] usually need access to a clock with accurate time-of-day (or simply time) and sources of periodic interrupts. Time sources are accessed to provide timestamps for events and log entries that document the recent history of the compute environment. Periodic interrupts provide a trigger to increment counters and read current conditions in the compute and networking environments. The compute domain may contain a very large number of NFV compute nodes [i.1], and each node needs to execute a process to synchronize its hardware and system clocks to a source of accurate time-of-day, preferably traceable to an international time standard.

With the foundation of time, date, and periodic interrupts, a measurement system can determine the beginning and end of time intervals, which is a fundamental aspect of metrics that involve counting and collecting events.

Table 4-1 specifies requirements applicable to time, date, and periodic interrupts.

Table 4-1: Requirements applicable to time, date and periodic interrupts

General-Time-01	Each node in the compute domain shall be able to take readings from (or access) a clock with accurate time-of-day and calendar date.
General-Time-02	Each node in the compute domain shall have a source of periodic interrupts available which are derived from the time-of-day clock, with configurable period (a parameter of metrics that use this feature).

When the results from measurement systems are collected and reported by management systems, the management systems may provide an additional time and date reading associated with the operation to collect the results, using their own time source.

5 Framework for Metric Definitions

The metric definitions in the present document are primarily based on the fact that the resources of the NFVI have utilization and performance characteristics that can be assessed by measurement processes. The resources may be implemented in hardware, software (such as virtual resources) or a combination of both. The measurement processes are primarily implemented in software (such as in the kernel or user space), but may be assisted by features of the hardware.

The measured NFVI resources and the measurement processes shall be completely specified in the dimensions of model numbers, firmware versions, software versions and any other aspects that influence the results (such as physical location of the components within a datacentre's racks and shelves). For example, the fixed frequency of the physical CPU clock in Hz, which governs the rate that the CPU executes instructions, is one important descriptor of the NFVI. Clock Speed may depend on other CPU settings, such as energy-saving power control. For one list of NFVI platform descriptors, see clause 5.1 of ETSI GS NFV-IFA 003 [i.9].

For each metric it specifies, the present document provides the following elements:

- Background
- Name
- Parameters (input factors)
- Scope of coverage
- Unit(s) of measure
- Definition
- Method of Measurement
- Sources of Error
- Discussion

NOTE: The present document specifies well-known metrics, and assumes that Virtual Infrastructure Managers (VIMs) will control and expose the metrics as specified here, or will be enhanced to collect and convey the metrics with the required framework elements, which are Name, Parameters, Scope, Units of measure and the source of the measurement (where the metric was measured, which may be synonymous with the scope).

6 Compute Metrics

6.1 Background

The Central Processing Unit (CPU) is an essential component of every coherent compute domain. Each CPU is a limited resource in terms of the instructions per second it can execute. It is valuable to monitor the utilization of the CPU resource to fulfil the goals of maintaining continued and efficient operations, and for troubleshooting abnormal behaviour to find root causes. For many uses, it is helpful to categorize the CPU's execution time into multiple execution contexts, such as system and user contexts. A compute node may include additional processors beyond the main CPU; the metrics specified in this clause can also be used to measure and report the usage of such processors.

VNFs also have a view of CPU resources in terms of execution time they have used during a measurement interval. However, the configured instantiation of the VNF determines how to map its view of virtual processor resource usage to actual hardware CPU resources available and used. For example, a VNF's processes may be pinned to one or more CPU cores or the VNF may be sharing the resources of many CPU cores with other VNFs.

6.2 Name

There are two variants of this metric:

- Processor Usage
- Processor Utilization

The two variants allow reporting this metric as a percentage. The metric is a function of the scope, set of reported contexts, measurement interval and other factors.

6.3 Parameters

The following parameters shall be supported for this metric:

- **Tick interval:** the period of timed interrupts when the processor's execution context can be recorded. Note that this parameter is an integral part of the method of measurement. The tick interval is sometimes called a "jiffy". The tick interval is controlled by a system parameter, "HZ", whose default value shall be 250 Hz for measurements complying with the present document.
- **Set of execution contexts:** the desired set of processor states with reported utilization. For example, the simplest set includes two states: active and idle. Time in the active context can be calculated as a sum of states with more specific definitions, such as the active states user and system. A commonly chosen set of four states is user, system, wait, and idle. See [i.2] for a list of eight states available in Linux OS.
- **End time:** the termination of the measurement interval (time and date).
- **Measurement interval:** the duration of the observation by the measurement system to assess the metric.

6.4 Scope

The list of one or more compute resources which shall be included in the values reported, and whether the resource is physical or virtual. Annex A gives examples of the scope usage for physical and virtual processor metrics.

6.5 Units of Measure

Processor usage results shall be reported as time in nanoseconds, and utilization shall be reported as the ratio of total time in one or more execution contexts to the total time in the measurement interval, expressed as a percentage.

6.6 Definition

The compute resource usage at time T (for a given scope, context, and measurement interval) shall be assessed as indicated in table 6.6-1.

Table 6.6-1: Requirements for Processor Metric variants

Requirement number	Metric Variant Name	Requirement
Compute-01	Processor Usage	Reported values of this metric variant shall quantify the total time that one or more compute resources (according to the defined scope) execute instructions in the specified execution context during the measurement interval.
Compute-02	Processor Utilization	Reported values of this metric variant shall quantify the ratio of time of Processor Usage to the time in the measurement interval, expressed as a percentage.

6.7 Method of Measurement

The measurement process shall interrupt the processor periodically, determine the execution context which was present when the timed interrupt occurred, and allocate the entire time interval since the previous interrupt to the usage total of that execution context. The total times for each desired context shall be accumulated throughout the specified measurement interval. On the completion of a measurement interval, the measured times shall be summed and the usage and/or the utilization shall be reported.

6.8 Sources of Error

The sources of error for this metric are listed below:

1) The method only reads the execution context when the timed interrupt occurs. Therefore, a context which is present only briefly between successive timed interrupts will not be observed and accumulate no measured usage.

2) The interval between successive time interrupts (between ticks, or the length of a jiffy) is a configurable system control setting in most operation systems. Use of a long interval with respect to processor context swapping will result in usage assessment with large quantization and to frequently miss contexts that have short duration.

3) The measurement process itself uses processor resources, and the resources consumed increase with the number of timed interrupts, or ticks per measurement interval. In Linux systems, the system parameter that controls the tick interval or duration of a jiffy is called HZ.

4) The accurate generation of timed interrupts on the intended interval boundaries is a function of many factors, including the activity of other interrupts and virtualisation of the process that generates the tick interval. There can be slightly less of slightly more than the configured number of ticks in each second, and this causes error in both the processor usage and utilization measurements.

5) Synchronization of the time-of-day clock with an external reference usually ensures sufficiently accurate timestamps and measurement intervals, but loss of synchronization for an extended period will cause time accuracy to suffer.

6.9 Discussion

The processor usage measurement for virtual processors is a topic of ongoing study and advanced development. Methods which produce more accurate usage measurements are expected in the future.

The main purpose of NFVI employing Hypervisors and Virtual Machines (VMs) (or other forms of virtualisation layers, such as Operating System Containers, OSCs) is to provide a predictable computing environment for VNFs. ETSI GS NFV-INF 010 [i.11] describes a set of Service Quality Metrics on the NFVI from the VNF point-of-view. One particular metric that was intended to constrain the predictability of computing environments is the VM Stall metric, defined in clause 5.1 of ETSI GS NFV-INF 010 [i.11]. Essentially, this metric is intended to measure both the frequency and duration of events where the VM was unable to execute its processes as planned, and execution is deferred to a later time.

Execution context is one of the Parameters for the Processor Usage or Utilization metric defined in clause 6.6 above, and [i.2] lists a set of states available in Linux OS. The key execution context of interest here is CPU steal time, or "st: time stolen from this vm by the hypervisor" [i.2]. Thus, for the Scope of a specific VM and the parameter CPU steal time, the Usage and Utilization metrics report the total time and percentage of time during the Measurement Interval that the VM Stall condition was present. While these results are not in the form of event frequency and duration originally envisioned in ETSI GS NFV-INF 010 [i.11], they should still be useful for trouble-shooting performance-related issues.

Ideally, CPU steal time Utilization should be zero percent. If a measurement indicates CPU steal time Utilization much greater than zero, there are several possible causes, including that the VM's resource allocation is too low or that the host is overbooked and the VM cannot access the configured resources [i.12].

Different tools and statistics are applicable to the OSC environment. There is no way to guarantee a minimal number of time shares to a particular container group of processes or cgroup. However, the upper limit on CPU time allocated to a cgroup can be set with a cgroup quota in the Linux Completely Fair Scheduler. When processes in a cgroup exceed their quota, the processes will be throttled by the quota. The number of periods and the accumulated amount of time in nanoseconds a cgroup has been throttled is reported in the `cpu.stat` file as the `nr_throttled` and `throttled_time` statistics [i.13]. The key execution context to reflect this phenonenon is the `throttled_time`.

7 Network Metrics

7.1 Background

Network Interfaces exist at many points on the communication paths through the NFVI. It is valuable to monitor the traffic load of arbitrary interfaces, both the physical interfaces with hardware resources and their virtual counterparts. Additional information can assist troubleshooting and resolution, so interfaces also indicate problems with attempted transmission or reception of packets.

7.2 Name

There are four fundamental metrics of traffic load and communication issues:

1) Packet count.

2) Octet count.

3) Dropped Packet count.

4) Errored Packet count.

These four metrics apply to a single direction of transmission. Therefore, transmit and receive versions of the four metrics make a total of eight for each interface.

7.3 Parameters

The following parameters shall be supported for the four metrics:

- **Interface:** the name (or other identifier) of a single interface where communication metrics are monitored and shall be unique within the Scope of measurement.
- **Measurement time:** the point in time when the counter(s) was (were) read (time and date).
- **Interface Speed:** the nominal frequency of the physical interface bit clock in bits per second, which governs the rate that the interface operates, and may be reported using a prefix multiplier. Virtual interfaces may not have a meaningful value for this parameter.
- **Interface Status:** the operational state of the interface indicating readiness for use, usually expressed as "up" or "down".

7.4 Scope

The list of one or more interface resources which are included in the values reported and whether the resource is physical or virtual.

7.5 Units of Measure

The Packet interface metrics shall be reported in units of packets as defined by the layer at which the interface is operating, such as the link layer. Examples include the Ethernet layer and VXLAN.

The Octet interface metric shall be reported in units of 8-bit bytes contained in successfully communicated packets as defined by the layer at which the interface is operating, such as link layer. Examples include the Ethernet layer and VXLAN.

7.6 Definition

The values of each metric at time T (for a given scope and measurement time) shall be assessed as the total counts meeting the criteria as specified in table 7.6-1, since the interface was registered with the overall management process.

Table 7.6-1: Requirements for Network Metrics

Requirement number	Metric Name	Requirement
Network-01	Packet Count	Reported values of this metric shall quantify the number of successfully transmitted or received packets from the individual interface's perspective.
Network-02	Octet Count	Reported values of this metric shall quantify the number of 8-bit bytes that constitute all successfully transmitted or received packets from the individual interface's perspective.
Network-03	Dropped Packet Count	Reported values of this metric shall quantify the number of discarded packets due to lack of available processing resources, such as memory buffer space.
Network-04	Errored Packet Count	For received packets, reported values of this metric shall quantify the number of packets detected with various forms of corruption during transmission, including errors detected with checks on length (too long, too short), and errors detected with integrity checks (e.g. based on cyclic redundancy checks, CRC). For transmitted packets, reported values of this metric shall quantify the number of packet transmission attempts with errors, including loss of link carrier during transmission, aborted transmission (due to multi-access collisions), and other detectable forms of transmission failure.

7.7 Method of Measurement

The interface driver shall interrupt its management process when one or more transmission or reception events take place, and to increment the relevant counters. When a measurement is performed at time T, the present communications counter values of each of the desired interfaces shall be recorded for both the transmit and receive directions.

7.8 Sources of Error

The sources of error for these metrics are listed below:

1) Counter roll-over: All counters have a finite range, and end-of-range roll-over will likely be encountered during the life of a busy interface. This is especially true for the Octets count and Gbit/s interfaces. Mitigations include reading the counters often enough to detect roll-over algorithmically and provide a wider range for the counts.

2) Hardware discard: Some received packets will be discarded at the physical layer and may not be included in the counts of Dropped packets. In other words, not all lost packets are reflected in error counts.

3) Resource limitations: Since interface counts are managed according to the priority of their interrupts, circumstance may occur when all interrupts cannot be served in a sufficiently timely fashion and information (including packets) may be lost.

4) Synchronization of the time-of-day clock with an external reference usually ensures sufficiently accurate timestamps, but loss of synchronization for an extended period will cause time accuracy to suffer.

7.9 Discussion

When interfaces are activated or created, they are registered with the management processes (in the Linux kernel, for example). Each registered interface effectively defines the packet type and header format and contents. For example, an Ethernet definition would reference the Ethernet header and trailer.

For the Linux OS, [i.3] describes the information available directly from the kernel and how to access it from the command line.

8 Memory Metrics

8.1 Background

Memory is a key resource for the NFVI. It is valuable to monitor the utilization and management of system and user memory and their virtual counterparts. This information can assist troubleshooting and resolution.

8.2 Name

There are six fundamental metrics of memory utilization:

1) Memory buffered

2) Memory Cached

3) Memory free

4) Memory Slab

5) Memory Total

6) Memory Used

The first four metrics comprise the memory unoccupied by user processes. Therefore, the sum of these metrics can be subtracted from the Memory Total to obtain the current value of Memory Used.

There are additional metrics of memory utilization that are dependent on the memory page size. Most operating systems have the ability to define memory pages that are larger than the typical size (4 Kibibytes) and establish a pool of these pages, called Hugepages in the Linux operating system, or similar on other operating systems [i.10].

7) Hugepage Pool Total

8) Hugepage Bytes Used

9) Hugepage Bytes Free

10) Hugepages Used

11) Hugepages Free

12) Hugepages Percent Used

13) Hugepages Percent Free

8.3 Parameters

The following parameters shall be supported for these four metrics:

- **Measurement time:** the point in time when the values were read (time and date).
- **RAM:** the Random Access Memory, RAM, available to the compute node measured.
- **Swap space:** the configured memory available for processes to share through swapping (reducing the RAM available for user processes).
- **Hugepage size:** the configured number of memory bytes contained in each Hugepage (specified in Kibibytes).

8.4 Scope

The list of one or more compute node resources which are included in the values reported, and whether the resource is physical or virtual.

8.5 Units of Measure

The memory values shall be reported in units of Kibibytes, or KiB, where 1 KiB equals 1 024 Bytes.

NOTE: Some systems will report values and display the units of kB (equal to 1 000 Bytes), such as the Linux /proc/meminfo. However, the values are actually in units of KiB.

The Hugepage memory values shall be reported in units of Kibibytes, KiB, Pages or Percent of Pages in the Pool Total, consistent with the metric name.

8.6 Definition

The values of each metric at time T (for a given scope and measurement time) shall be assessed as the total KiB meeting the criteria as specified in table 8.6-1.

Table 8.6-1: Requirements for Memory Metrics

Requirement number	Metric Name	Requirement
Memory-01	Memory Buffered	Reported values of this metric variant shall quantify the amount of temporary storage for raw disk blocks.
Memory-02	Memory Cached	Reported values of this metric variant shall quantify the amount of RAM used as cache memory.
Memory-03	Memory Free	Reported values of this metric variant shall quantify the amount of RAM unused.
Memory-04	Memory Slab	Reported values of this metric variant shall quantify the amount of memory used as a data structure cache by the kernel.
Memory-05	Memory Total	Reported values of this metric variant shall quantify the amount of usable RAM, which excludes a quantity of reserved bits and the kernel binary code.
Memory-06	Memory Used	Reported values of this metric variant shall quantify the amount of memory used by user processes, which is derived from other metric variants as follows: Memory Used = Memory_Total - (Memory Free + Memory Buffered + Memory_Cached + Memory_Slab)

The values of each Hugepage metric at time T (for a given scope and measurement time) shall be assessed as the total units meeting the criteria as specified in table 8.6-2.

Table 8.6-2: Requirements for Hugepage Metrics

Requirement number	Metric Name	Requirement
Hugepage-01	Hugepage Pool Total	Reported values of this metric variant shall quantify the number of Hugepages currently configured in the pool, which is the total of pages available.
Hugepage-02	Hugepage Bytes Used	Reported values of this metric variant shall quantify the number of used Bytes in the Hugepage Pool.
Hugepage-03	Hugepage Bytes Free	Reported values of this metric variant shall quantify the number of free Bytes in the Hugepage Pool.
Hugepage-04	Hugepages Used	Reported values of this metric variant shall quantify the number of used pages in the Hugepage Pool.
Hugepage-05	Hugepages Free	Reported values of this metric variant shall quantify the number of free pages in the Hugepage Pool.
Hugepage-06	Hugepage Percent Used	Reported values of this metric variant shall quantify the number of used pages in the Hugepage Pool as a percentage of the total pool.
Hugepage-07	Hugepage Percent Free	Reported values of this metric variant shall quantify the number of free pages in the Hugepage Pool as a percentage of the total pool.

8.7 Method of Measurement

When a measurement is performed at time T, each of the desired metrics and their present values shall be recorded for reporting.

8.8 Sources of Error

The sources of error for these metrics are listed below:

1) Synchronization of the time-of-day clock with an external reference usually ensures sufficiently accurate timestamps, but loss of synchronization for an extended period will cause time accuracy to suffer.

8.9 Discussion

For the Linux OS, [i.3] describes the information available directly from the kernel and how to access it from the command line. [i.4] describes the information available in /proc/meminfo. Note that the methods of measurement for storage systems vary widely and depend on the implementation, therefore storage metrics are not considered for standardization at this time.

9 Follow-on Activities

The present document focuses on specifications of metrics reported by the NFVI and virtualised versions of platform resources. Additional metrics could be provided in the future, such as Storage Metrics.

Normative metric specifications are also needed for externally observable characteristics of the NFVI, for example, in terms of the communications speed and capacity provided.

Annex A (informative): Examples of Scope Specification for Metrics

A.1 Description

As described in clause 5 of the present document, each metric specification is composed of a framework with nine elements. One framework element is the Scope applicable to each measurement, which provides the list of physical or virtual resources intended to be measured. An example of each type of metric (physical and virtual) is illustrated in figure A.1-1.

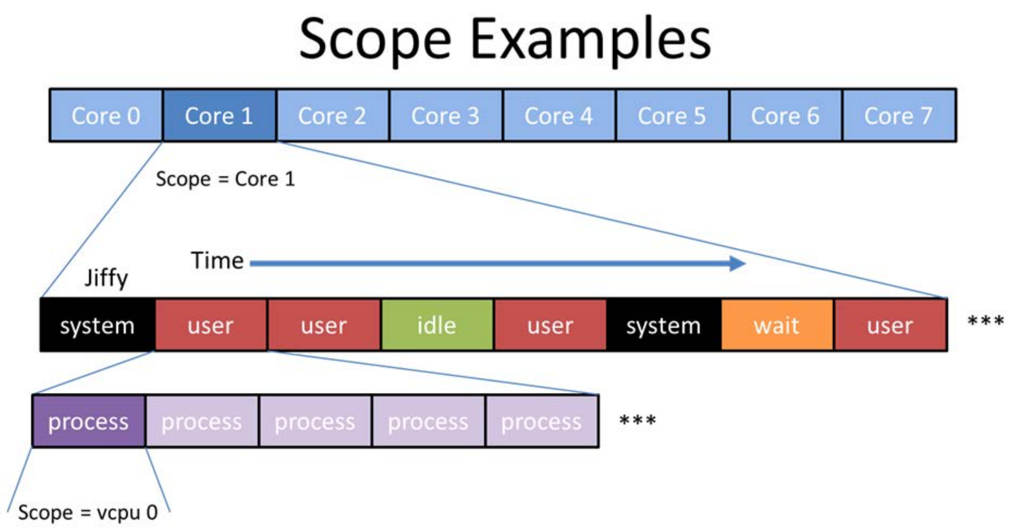

Figure A.1-1: Examples of Scope coverage for Physical and Virtual Processor Usage Metrics and Measurements

The top of figure A.1-1 depicts a compute resource with eight physical processor cores. As an example of a physical processor measurement Scope, figure A.1-1 indicates coverage of a single processor core (Core 1).

Within the single processor core, processing time is divided among different processor contexts [i.2]. The context is measured at each tick interval (called a Jiffy). The time spent in each context is accumulated and reported at the end of each measurement interval.

The user context is further divided among the processes spawned by different applications. VNFC (synonymous with Virtual Machines (VMs) or OS Containers (OSCs), or a Libvirt domain) will occupy one or more processes.

Last, figure A.1-2 illustrates a Libvirt domain specification with a single virtual CPU (vcpu) that has been statically placed on Core 1 of the physical processor. Therefore, the scope of process identification(s) associated with the VNFC of this domain describes its compute resource usage within the user context (sometimes called the guest context in some virtualisation methods) and comprises the coverage of the processor usage measurement for a VM or OSC.

```
Libvirt domain spec for CPU allocation
<domain>
  …
  <vcpu placement='static' cpuset="1" current="1"></vcpu>
  <vcpus>
    <vcpu id='0' enabled='yes' hotpluggable='no' order='1'/>
</vcpus>
  ...
</domain
```

Figure A.1-2: Example of Scope coverage partly based on Libvirt domain specification

In summary, the scope element provides a way to narrow-down the resources to which a metric will be applied, and the coverage of measurements using the metric definition.

A.2 Memory Scope

The described Scope of Memory Metrics is supported by the Metric Parameter definitions, in terms of the configured values and the host operating system itself (some memory is reserved for kernel operations, etc.).

RAM is a configured parameter, representing a limit on physical RAM or a limit on RAM allocation to a guest host (VM or OSC). RAM is different from "Memory Total" as assessed by some management processes, as described below. Swap Space is another configured parameter (which reduces the RAM available).

Table A.2-1 organizes and compares the various specifications and their definitions, as made available through different forms of different system assessments.

Table A.2-1: Comparison of Memory Metric Definitions

Metric Name	Assessment	Metric Name	Definition	Reference
Memory Total	/proc/meminfo	MemTotal	Total amount of usable RAM, in kibibytes, which is physical RAM minus a number of reserved bits and the kernel binary code	[i.4]
	free(1)	total	Total installed memory (**MemTotal** and **SwapTotal** in /proc/meminfo)	[i.7]
Memory Buffered	/proc/meminfo	Buffers	The amount, in kibibytes, of *temporary storage for raw disk blocks*	[i.4]
	free(1)	buffers	Memory used by kernel buffers (Buffers in /proc/meminfo)	[i.7]
Memory Cached	/proc/meminfo	Cached	The amount of physical RAM, in kibibytes, used as cache memory	[i.4]
	free(1)	cache	Memory used by the page cache and slabs (**Cached** and **Sreclaimable** in /proc/meminfo)	[i.7]
Memory Free	/proc/meminfo	MemFree	The amount of physical *RAM, in kibibytes, left unused* by the system	[i.4]
	free(1)	free	Unused memory (**MemFree** and **SwapFree** in /proc/meminfo)	[i.7]
Memory Slab	/proc/meminfo	Slab	The total amount of *memory, in kibibytes, used by the kernel to cache data structures* for its own use	[i.4]
	/proc/meminfo	Sreclaimable	The part of Slab that can be reclaimed, such as caches	[i.4]
	/proc/meminfo	Sunreclaim	The part of Slab that cannot be reclaimed, even when lacking memory	[i.4]
"memory used"	/proc/meminfo	none		[i.4]
	free(1)	used	Used memory (calculated as **total – free – buffers – cache**) Equiv: **(MemTotal + SwapTotal)** **- (MemFree + SwapFree)** **- (Buffers)** **- (Cached + Sreclaimable)**	[i.7]
Memory Used	collectd memory.c	mem_used	mem_used = mem_total – (mem_free + mem_buffered + mem_cached + mem_slab_total); or **MemTotal – (MemFree + Buffers + Cached + Slab)**	[i.8]

There are differences between the definitions of "memory used" reported by the free(1) user command and collectd.

User command free(1) for memory "used" includes the memory allocated to Swap space as the basis for total, from which the free memory, free swap space, buffer memory, cache memory, and only the reclaimable memory Slabs are subtracted in the calculation.

The value reported by collectd for "mem_used" excludes the memory allocated to Swap space as the basis for total, from which the memory left unused, the memory used for buffers, memory used as cache, and **all memory Slabs** (both the reclaimable and unreclaimable memory Slabs) are subtracted in the calculation. The present document follows the collected calculation.

Neither calculation of used memory uses the RAM as a basis, which means that the memory occupied by "a number of reserved bits and the kernel binary code" are not reported as used (and therefore, the value leans toward the description: user memory used).

History

Document history		
V3.1.1	August 2018	Publication
V3.2.1	March 2019	Publication
V3.3.1	June 2020	Publication