Tech Note:

CPPM Graphite Reporting
TechNote

Copyright
Copyright © 2014 Aruba Networks, Inc.

Aruba Networks trademarks include AirWave, Aruba Networks®, Aruba Wireless Networks®, the registered Aruba the Mobile Edge Company logo, Aruba Mobility Management System®, Mobile Edge Architecture®, People Move. Networks Must Follow®, RFProtect®, Green Island®. All rights reserved. All other trademarks are the property of their respective owners.

Open Source Code
Certain Aruba products include Open Source software code developed by third parties, including software code subject to the GNU General Public License (GPL), GNU Lesser General Public License (LGPL), or other Open Source Licenses. The Open Source code used can be found at this site:

http://www.arubanetworks.com/open_source

<table>
<thead>
<tr>
<th>Date</th>
<th>Modified By</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>March 19th 2014</td>
<td>Danny Jump</td>
<td>Initial Published Version</td>
</tr>
</tbody>
</table>
Table of Contents
Overview ................................................................................................................................. 4
Background............................................................................................................................... 4
High-level................................................................................................................................. 4
Setup Details........................................................................................................................... 4
Granting/Restricting Access to Graphite in CPPM 6.3.1 ....................................................... 6
How to Access the Graphite Interface .................................................................................. 7
  Graphite ‘Graph’ Usage......................................................................................................... 7
  Breaking out basic_perf options......................................................................................... 9
  Breaking out stats options................................................................................................. 16
  Redundancy ....................................................................................................................... 18
Appendix A - Process Counters .......................................................................................... 19
Appendix B - System Counters ............................................................................................ 21
Appendix B - Network Counters .......................................................................................... 23

Table of Figures
Figure 1 - Banner message indicating ‘Performance Monitoring’ is disabled ....................... 5
Figure 2 - Enabling ‘Performance Monitor Display’ on a CPPM node .................................. 5
Figure 3 - Viewing-running daemons – notice two new Stats daemons processes ............... 5
Figure 4 - Allowing stats collection to be started ................................................................ 6
Figure 5 - Restricting access to Graphite from Network ACL .............................................. 6
Figure 6 - Graphite Navigation Interface ............................................................................. 7
Figure 7 - Graph with ‘No Data’ metrics selected ................................................................ 8
Figure 8 - Two unrelated metric that fit on a graph............................................................... 8
Figure 9 - Two unrelated metrics that appear to fit but are flat-lined to top and bottom ...... 9
Figure 10 - Top level node view ................................................................................................. 9
Figure 11 – basic_perf – node level stats .................................................................................. 10
Figure 12 - basic_perf - disks ..................................................................................................... 10
Figure 13 - basic_perf - cpu iostat ............................................................................................. 11
Figure 14 – basic_perf - CPU load options 1min/5min/15min .................................................... 11
Figure 15 - Example of a 5min CPU load graph ........................................................................ 12
Figure 16 - basic_perf - memory usage ....................................................................................... 12
Figure 17 - basic_perf - RAW Ethernet metrics ........................................................................ 13
Figure 18 - basic_perf - CPPM applications ............................................................................... 14
Figure 19 - basic_perf - TCP connections on the system ............................................................ 15
Figure 20 - basic_perf - UDP packets Bytes/Kb/MB .................................................................. 15
Figure 21 - stats default-metrics ................................................................................................. 16
Figure 22 - stats timers-metrics .................................................................................................. 17
Overview

The following guide has been produced to help introduce the Graphite reporting feature released in CPPM 6.3 release. Within the 6.3 release Graphite can be thought of as ‘sunrise’ feature. This guide will be updated and republished to reflect new and improved functionality we deliver in subsequent software releases. Please check back regularly for updates.

Note: Where you see a red-chili this is to signify a ‘hot’ important point and highlights that this point is to be taken as a best-practice recommendation.

Background

As customer deployments become more complex we realize there is a need to expand and deliver additional reporting capabilities to ensure our customers are able to fully understand how CPPM is performing in their environment. Graphite is a new reporting tool to COMPLIMENT Insight. There are some minor reporting overlaps within Graphite when compared to InsightDB. Think of Graphite more of a resource and system-monitoring product than a L7 application specific reporting tool.

i.e Graphite can report on Memory Usage, Disk IOPs, where Insight can better report on number of RADIUS Auths or number if OnGuard posture updates from a particular NAD.

High-level

Graphite runs on every CPPM instance, whether it’s a standalone-node or a clustered multi-node deployment. When systems are clustered statistics can be viewed on any single node within the cluster, this can be any node independent of Insight or Publisher/Subscriber.

Setup Details

On every system that runs there is a process called StatsD. This is responsible for collecting statistics locally and storing the data in its own whisper database. It takes feeds/subscriptions from multiple places with in the CPPM node. Carbon sends data to its local CPPM instance. All the metrics are stored locally within the node, finally there is another process, Graphite that is responsible for producing the graphs and reports.

To ensure that the collection (statsd) and the aggregation (carbon) processes are running review the System level processes as shown below in Figure 1. Notice the two new services highlighted below.

Note: The Performance Monitoring Display is disabled by default on a CPPM node.
Whether you have a standalone node or you enable Performance Monitoring on just one or all the nodes in a cluster you need to ensure that is enabled some-where. If its not, then the below message will be displayed in the banner on the Dashboard at sign-in.

![ClearPass Policy Manager](image)

**Figure 1 - Banner message indicating 'Performance Monitoring' is disabled**

To fix this issue ‘Enable Performance Monitoring Display’ as shown below under Administration-> Server Manager->Server Configuration->{CPPM Node}->System

![Server Configuration - qa83 (10.100.8.83)](image)

**Figure 2 - Enabling 'Performance Monitor Display' on a CPPM node**

Specific to the other running daemon on the system required fro Graphite, they can be monitored under Administration-> Server Manager->Server Configuration->{CPPM Node}->Services Control as shown below in Figure 3.

![Server Configuration - qa86.amigopod.arubanetworks.com (10.100.9.86)](image)

**Figure 3 - Viewing-running daemons – notice two new Stats daemons processes**
If the above daemon's are not running please check under **Administration->Server Manager->Server Configuration->{CPPM Node}->Service Parameters** to ensure Enable Stats Collection is set to **TRUE** (shown below in Figure 4). This does not actually stop/start the collections daemons, it merely allows an administrator to stop/start them (Figure 3).

![Figure 4 - Allowing stats collection to be started](image)

If you toggle this setting from **TRUE/FALSE** and save the change you will receive an informational message indicating you have to manually stop start the daemons. If this is set to **FALSE**, the collection services will not start if the system reboots.

**Granting/Restricting Access to Graphite in CPPM 6.3.1**

Starting in CPPM 6.3.1 we have provided the ability to restrict access to the Graphite UI by utilizing the same ACL based system we implemented previously for Policy Manager, Guest Operator, OnGuard and Insight, under **Administration->Server Manager->{CPPM Node}->Network->Restrict Access->Resource Name**. By default access is **EXPLICITLY DENIED** to Graphite and you need to either add an ‘Allow’ ACL based upon the source IP-host/IP-subnet/FQDN to match the management users who need access, or add a ‘DENY’ ACL for a ghost network, then all other subnets/hosts get an explicit ALLOW to the Graphite UI.

![Figure 5 - Restricting access to Graphite from Network ACL](image)

If you inadvertently lock yourself out of the Policy Manager UI then the following CLI command removes the Policy Manager ACL’s - **system apps-access-reset**
How to Access the Graphite Interface

To access the Graph data available within a node or cluster use the following URL format
https://CPPM_IP_ADDRESS/graphite

In CPPM 6.3.0 this will display without any authentication the graphing interface. In CPPM 6.3.1 by default access is **EXPLICITLY DENIED**, you need to configure an allow network access ACL (see - Restricting Access to Graphite in CPPM 6.3.1 on Page 6).

After you connect, in the Left-Hand-Side navigation bar, you are presented with the following.

![Graphite Navigation Interface](image)

**Figure 6 - Graphite Navigation Interface**

From here you can expand out for more detail for the following `subject` areas.

- basic_perf
- carbon
- stats
- stats_count
- statsd

**Graphite ‘Graph’ Usage**

When you initially select a metric to Graph, it will initially show ‘No Data’, as shown below in Figure 6. However, as you select additional metrics it remembers the previous metrics you selected and keeps all the metric on the same graph. This is both useful if you want to graph say CPU and memory together, however less important and confusing if you want to graph disk IO writes against free memory. Its also very important to realize that when graphing multiple items, on the X or Y axis one of the metrics may be many orders of magnitude different than the other metric, so it appears as though it graphs as a flat line on the Y axis, see the example below in Figure 8. So it’s very important to remember that as you select graphing metrics you also **deselect** them as you move to a new graph/resource type.
Figure 7 - Graph with 'No Data' metrics selected

The below graphed metrics fit to the same X scale. However its pure coincidence and they have no relation to each-other. The graph in Figure 8, shows two flat line metrics as there scale is so far apart both appear as flat-lines at 0 and 100 on the X-axis.

Figure 8 - Two unrelated metric that fit on a graph
So ensure that you deselect metrics as you move to your next graph.

We will explain the following graphing areas, **basic_perf**, **carbon and stats** and what can be reported. Some of the stats/reports are available for all the nodes in cluster, whilst some stats are only available for the node you are specifically connected to. For the purpose of this TechNote we have four nodes, qa74, qa78, qa79 and qa83, we will connect to qa83.

**Note:** If you select a different node from the navigation list as shown below, then what your actually signed into, Graphite will have to connect to that nodes whisper DB to read the data prior to rendering/displaying the graph. Therefore be conscious of this when nodes are separated over a WAN with restricted bandwidth and/or high latency that will limit data throughput because of the data path BDP.

**Breaking out basic_perf options**
If you expand out the options for any individual node (e.g. qa83) you see the following list of items. We will go through most of the details available below.

**disks** – we can view all the disks in the node and report the performance of individual disks metrics. The below screen shows multiple disk in this system, this system is actually a cp-hw-25k appliance as we have six scsi disks shown. The newer Dell R620 appliance utilizes six scsi disks, where the older (pre May 2013) 25k appliance only had four.

For each of the disks we are able to report the following metrics.

- **io** – Shows current/ms/weighted_ms
- **read** – Shows issued/merged/ms/sectors
- **write** – Shows complete/merged/ms/sectors

**Note:** We do not currently provide guidelines in relation to what is normal/poor/average performance due to the multiple personalities CPPM can undertake. A deployment where a system is performing mainly AAA authentication is a very different performance profile from a deployment where significant amounts of OnBoarding or Guest creation is happening. The later generates a large amount of DB writes while the former is 90%+ read.
**iostat** – In Linux the iostat command can generate multiple reports, here we are using iostat to generate CPU Utilization statistics.

![iostat](image1)

**Figure 13 - basic_perf - cpu iostat**

**idle** - Shows the percentage of time that the CPU or CPUs were idle and the system did not have an outstanding disk I/O request.

**iowait** - Shows the percentage of time that the CPU or CPUs were idle during which the system had an outstanding disk I/O request.

**nice** - Shows the percentage of CPU utilization that occurred while executing at the user level with nice priority.

**steal** - Shows the percentage of time spent in involuntary wait by the virtual CPU or CPUs while the hypervisor was servicing another virtual processor.

**system** - Shows the percentage of CPU utilization that occurred while executing at the system level (kernel).

**user** - Shows the percentage of CPU utilization that occurred while executing at the user level (application).

**load** – this relates to processor (CPU) load on the system. We have the option of showing either 1min, 5min or 15minute graphs. This is inline with what you'd see when looking at the Linux command ‘top’ which also display averages in 1,5 and 15minutes periods.

![load](image2)

**Figure 14 – basic_perf - CPU load options 1min/5min/15min**
Figure 15 - Example of a 5min CPU load graph

On multi-processor system, the load is relative to the number of processor cores available. The "100% utilization" mark is 1.00 on a single-core system, 2.00, on a dual-core, 4.00 on a quad-core, etc. so it’s very important to understand the numbers of cores on the system you are monitoring.

memory – we can show statistics about core memory or statistics related to how we are swapping between memory and disk.

Figure 16 - basic_perf - memory usage
**buffers** - Shows the amount of memory being used by the kernel buffer cache. The buffer cache is used to speed up disk operations, by allowing disk reads and writes to be serviced directly from memory.

**cache** - Shows how many memory pages the kernel has cached for faster access later.

**free** - Shows memory not in use.

**shared** - Shows the amount of physical memory shared between multiple processes.

**total** – Shows the total (physical) RAM (excluding a small bit that the kernel permanently reserves for itself at startup).

**used** – Shows the memory in use by the OS.

**swap** - Shows information about swap space usage (i.e. memory contents that have been temporarily moved to disk).

**nic** – We can view ‘raw’ metrics relating to the Ethernet cards.

**Figure 17 - basic_perf - RAW Ethernet metrics**

For each of the Ethernet adapters on the system (eth0 – Management Interface, eth1 – Data Interface) we can graph the following Receive (rx) or Transmit (tx) metrics or look at the collisions (collisions should only occur if there is a mis-match between the interface on CPPM and the upstream switch, i.e. one side configured for auto-negotiation and the other hard-coded) for the interface.

**bytes / dropped / errors / frame / overruns / packets**
**nw** – CPPM Networking applications

![Figure 18 - basic_perf - CPPM applications](image)

We have the ability to graph metrics across multiple CPPM applications. The metrics are **Total's** for each of the applications below, except for RADIUS where we provide statistics broken down for the common RADIUS ports 1812/1813/1645/1646.

- **agent_controller** – This relates to traffic on port 6658
- **db** – This relates to traffic on port 5432
- **http** – This is relates to traffic on port 80
- **https** – This is related to traffic on port 443
- **ntp** – This relates to the number of bytes of Network Time traffic on port 123
- **radius** – This relates to traffic on ports 1812,1813,1645,1646
- **ssh** – This related to the amount of ssh (appadmin/arubasupport) traffic on port 22
- **tacacs** – This relates to the amount of tacacs traffic processed by CPPM on port 49
tcp – We can monitor and graph metrics relating to all tcp connection on the system. Very useful to understand the network connection and how this potentially translates to load on the system.

![tcp connections](image)

**Figure 19 - basic_perf - TCP connections on the system**

We are able to graph the number of current TCP connections active/failed/passive and the number of TCP resets received/sent/sockets on the node. This can be utilized with other metric to understand the load on the system. High levels of TCP resets could be an indication to a system overload and our inability to process new connections/users.

udp - We can monitor and graph metrics relating to all udp packets send or received.

![udp packets](image)

**Figure 20 - basic_perf - UDP packets Bytes/Kb/MB**

In relation to UDP metrics across the system, this can be viewed as data relating to DNS, NTP, SNMP and off course RADIUS authentications and RADIUS authorizations.
Breaking out stats options

In this section are three sections, default, statsd and timers, we will work through default and timers at a high level. Statsd refers to the internal processing of statsd itself and is not covered in this document.

**Note:** Reporting metrics in these sections (default/statsd/timers) are only available to the node you are logged into. Metrics for this section are not replicated/retrieved across the network.

**stats - default**

![Figure 21 - stats default-metrics](image)

Within this section we have the ability to report on a multitude of metrics. Specifically we can report the following.

- **auth** – count and success
- **autz** – count and success
- **rules** – enforcement count and service count
- **webauth** – applogin count and success
**stats - timers**

![Diagram of stats - timers](image)

**Figure 22 - stats timers-metrics**

Within this section of the stats metrics, timers provides an insight to a comprehensive list of metrics that can be reported. At a high level we can see the following application/processes.

Expanding on any of the application display a plethora of metric that is beyond this documents scope. But the following are available.....

**ps - autz / enforcement / policy / rolemapping / rules / service / sessionlog**

**webauth – auth / policy / service time**
Redundancy

If a CPPM node goes down it loses the ability to send information to Graphite. No performance data can be collected from this node.

If a CPPM node hosting Graphite goes down then we can do any of the following

- No monitoring data will be available until Graphite is enabled on the new node. The system will lose the historical information.
- Enable Graphite on the secondary node.
Appendix A - Process Counters

**Note:** Process counters are prefixed by 'stats', CPPM zone and hostname.

- `radius.auth.{authsource}.success` -> Total num of successful authentications against the auth source.
- `radius.auth.{authsource}.failure` -> Total num of failed authentications against the auth source.
- `radius.auth.{authsource}.count` -> Total authentications against the auth source.
- `radius.auth.{authsource}.lookup-time` -> Time taken to perform user lookup against the auth source.
- `radius.auth.{authsource}.time` -> Time taken for user lookup against the authentication source.
- `radius.policyeval.time` -> Total time taken for policy evaluation from RADIUS perspective.
- `radius.request.time` -> Total time taken for end to end RADIUS request (Access Request to Access Accept/Reject).
- `ps.service.count` -> Total num of service policies evaluated.
- `ps.rolemapping.count` -> Total num of role mapping policies evaluated.
- `ps.posture.count` -> Total num of posture policies evaluated.
- `ps.audit.count` -> Total num of audit policies evaluated.
- `ps.restriction.count` -> Total num of authsource restrictions evaluated.
- `ps.enforcement.count` -> Total num of enforcement policies evaluated.
- `ps.auth.success` -> Total num of auths.
- `ps.auth.failure` -> Total num of failed auths.
- `ps.auth.total` -> Total num of successful auths.
- `ps.service.time` -> Time taken for service classification rules.
- `ps.rolemapping.time` -> Time taken for role mapping rules.
- `ps.posture.time` -> Time taken for posture rules.
- `ps.audit.time` -> Time taken for audit rules.
- `ps.restriction.time` -> Time taken for auth source restriction rules.
- `ps.enforcement.time` -> Time taken for enforcement policy rules.
- `ps.policy.{connproto}.time` -> Total time taken to process policy-request.
- `ps.policy.rm.time` -> Total time taken to perform role mapping.
- `ps.policy.sessionlog.time` -> Total time taken to perform session logging.
- `autz.{autzsource}.success` -> Total num of successful authorizations.
- `autz.{autzsource}.failure` -> Total num of failed authorizations.
- `autz.{autzsource}.count` -> Total num of authorizations.
- `autz.{autzsource}.time` -> Time taken for authorization for an auth source.
- `autz.time` -> Total time taken for auth as a part of policy evaluation.
- `webauth.weblogin.time` -> Time taken for weblogin.
- `webauth.weblogin.count` -> Total num of weblogins.
- `webauth.weblogin.success` -> Total num of successful weblogins.
ClearPass 6.3.x

- \texttt{webauth.weblogin.failure} - Total num of failed weblogins.
- \texttt{webauth.weblogin.service.time} - Time taken for weblogin service classification.
- \texttt{webauth.weblogin.policy.time} - Time taken for weblogin policy evaluation.
- \texttt{webauth.weblogin.auth.time} - Time taken for weblogin PAP/TokenService authentication.
- \texttt{webauth.applogin.time} - Time taken for appllogins.
- \texttt{webauth.applogin.count} - Total num of appllogins.
- \texttt{webauth.applogin.success} - Total num of successful appllogins.
- \texttt{webauth.applogin.failure} - Total num of failed appllogins.
- \texttt{webauth.applogin.service.time} - Time taken for appllogin service classification.
- \texttt{webauth.applogin.policy.time} - Time taken for appllogin policy evaluation.
- \texttt{webauth.applogin.auth.time} - Time taken for appllogin PAP/TokenService authentication.
- \texttt{webauth.saml.idp.time} - Time taken for SAML IdP requests.
- \texttt{webauth.saml.idp.count} - Total num of SAML IdP requests.
- \texttt{webauth.saml.idp.success} - Total num of successful SAML IdP requests.
- \texttt{webauth.saml.idp.failure} - Total num of failed SAML IdP requests.
- \texttt{webauth.saml.idp.service.time} - Time taken for SAML IdP service classification.
- \texttt{webauth.saml.idp.policy.time} - Time taken for SAML IdP policy evaluation.
- \texttt{webauth.saml.idp.auth.time} - Time taken for SAML IdP PAP/TokenService authentication.
- \texttt{webauth.saml.sp.time} - Time taken for SAML SP requests.
- \texttt{webauth.saml.sp.count} - Total num of SAML SP requests.
- \texttt{webauth.saml.sp.success} - Total num of successful SAML SP requests.
- \texttt{webauth.saml.sp.failure} - Total num of failed SAML SP requests.
- \texttt{webauth.saml.sp.service.time} - Time taken for SAML SP service classification.
- \texttt{webauth.saml.sp.policy.time} - Time taken for SAML SP policy evaluation.
- \texttt{webauth.saml.sp.auth.time} - Time taken for SAML SP PAP/TokenService authentication.
- \texttt{tacacs.auth.time} - Total time taken for tacacs authentication.
- \texttt{tacacs.authn.time} - Time taken for LDAP/DB PAP authentication
- \texttt{tacacs.auth.service.time} - Time taken for service classification.
- \texttt{tacacs.auth.policy.time} - Time taken for policy evaluation.
- \texttt{tacacs.auth.count} - Total num of authn requests.
- \texttt{tacacs.auth.success} - Num of successful authn.
- \texttt{tacacs.auth.failure} - Num of failure authn.
- \texttt{tacacs.autz.time} - Time taken for authorization.
- \texttt{tacacs.autz.count} - Total num of authn requests.
- \texttt{tacacs.autz.success} - Num of successful autz.
- \texttt{tacacs.autz.failure} - Num of failure autz.
Appendix B - System Counters

**Note:** System counters are prefixed by 'basic_perf', CPPM zone and hostname.

- \texttt{disks.{sda#}.io.current} -> Current IOs in progress
- \texttt{disks.{sda#}.io.ms} -> Milli seconds spent doing IOs
- \texttt{disks.{sda#}.io.weighted_ms} -> Weighted number of milli seconds spent doing IOs
- \texttt{disks.{sda#}.read.issued} -> Number of reads issued
- \texttt{disks.{sda#}.read.merged} -> Number of reads merged
- \texttt{disks.{sda#}.read.ms} -> Milli seconds spent reading
- \texttt{disks.{sda#}.read.sectors} -> Number of sectors read
- \texttt{disks.{sda#}.write.complete} -> Number of writes complete
- \texttt{disks.{sda#}.write.merged} -> Number of writes merged
- \texttt{disks.{sda#}.write.ms} -> Milli seconds spent writing
- \texttt{disks.{sda#}.write.sectors} -> Number of sectors wrote
- \texttt{iostat.idle} -> % of time that CPU was idle and the system didn’t have an outstanding disk IO request
- \texttt{iostat.iowait} -> % of time that CPU was idle during which system had an outstanding disk IO request
- \texttt{iostat.nice} -> % of CPU utilization that occurred while executing at the user level with nice priority
- \texttt{iostat.steal} -> % of time spent in involuntary wait by the CPU while system was servicing another vCPU
- \texttt{iostat.system} -> % of CPU utilization that occurred while executing at the kernel level
- \texttt{iostat.user} -> % of CPU utilization that occurred while executing at the user level
- \texttt{load.15min} -> CPU load average over the last 15 mins
- \texttt{load.10min} -> CPU load average over the last 10 mins
- \texttt{load.5min} -> CPU load average over the last 5 mins
- \texttt{memory.mem.buffers} -> Amount of memory reserved by the OS for future allocations
- \texttt{memory.mem.cached} -> Amount of memory cached
- \texttt{memory.mem.free} -> Amount of memory free
- \texttt{memory.mem.shared} -> Amount of memory shared by different processes
- \texttt{memory.mem.total} -> Total memory available on the system
- \texttt{memory.mem.used} -> Amount of memory in use
- \texttt{memory.swap.free} -> Amount of swap free
- \texttt{memory.swap.used} -> Amount of swap used
- \texttt{memory.swap.total} -> Amount of swap total
- \texttt{nic.{eth#}.rx.bytes} -> Bytes received
- \texttt{nic.{eth#}.rx.dropped} -> Bytes dropped by receive buffer
- \texttt{nic.{eth#}.rx.errors} -> Number of error packets in the receive buffer
- \texttt{nic.{eth#}.rx.frame} -> Number of receive frames
- \texttt{nic.{eth#}.rx.overruns} -> Receive packets resulted in overruns
**Tech Note:** ClearPass Graphite Reporting TechNote

- **nic.{eth#}.rx.packets** -> Number of packets received
- **nic.{eth#}.tx.bytes** -> Bytes transmitted
- **nic.{eth#}.tx.dropped** -> Bytes dropped by transmission buffer
- **nic.{eth#}.tx.errors** -> Number of error packets in the transmission buffer
- **nic.{eth#}.tx.carrier** -> Number of transmission carriers
- **nic.{eth#}.tx.overruns** -> Transmission packets resulted in overruns
- **nic.{eth#}.tx.packets** -> Number of packets transmitted
- **nic.{eth#}.collisions** -> Field to indicate tx/rx collisions
- **tcp.connections.active** -> Active TCP connections
- **tcp.connections.failed** -> Failed TCP connections
- **tcp.connections.passive** -> Passive TCP connections
- **tcp.resets.received** -> Received TCP resets
- **tcp.resets.sent** -> Sent TCP resets
- **udp.packets.received** -> Received UDP packets
- **udp.packets.sent** -> Sent UDP packets
Appendix B - Network Counters

**Note:** Network traffic counters are prefixed by 'basic_perf', CPPM zone and hostname. The values are indicated in bytes.

- **nw.agent_controller.total** -> Total Agent controller traffic on the port 6658
- **nw.db.total** -> Total PostgreSQL database traffic on the port 5432
- **nw.http.total** -> Total HTTP traffic on the port 80
- **nw.https.total** -> Total HTTPS traffic on the port 443
- **nw.ntp.total** -> Total NTP traffic on the port 123
- **nw.radius.1645.total** -> Total RADIUS auth traffic on port 1645
- **nw.radius.1646.total** -> Total RADIUS accounting traffic on port 1646
- **nw.radius.1812.total** -> Total RADIUS auth traffic on port 1812
- **nw.radius.1813.total** -> Total RADIUS accounting traffic on port 1813
- **nw.ssh.total** -> Total SSH traffic on port 22
- **nw.tacacs.total** -> Total TACACS+ traffic on port 4949