Runtime analysis of the monitoring agents methodology

Suggestion

Metrics monitored by the agents:	Publishing Mode:	Frequency of Reading the values:	Other metrics that may apply only to few agents:	System Configuration
CPU, Processes, Memory, Interfaces, Libvirt, IPMI, Disk Status	Writing to the file	1 Sec.	OVS, DPDK, PCM, RAS (MceLog)	1. Intel Xeon Server with at least 3 ethernet interfaces. 2. KVM/Qemu 3. At least 2 VMs Running.

Q. Why These metrics?

A. The choice of metrics first started with this link - https://wiki.opnfv.org/display/fastpath/Collectd+Metrics+and+Events . From this list, these metrics where chosen as they are supported by all the agents.

Q. Why publishing mode as 'writing to the file'?

A. This mode is, again, supported by all (well, almost). And, it makes comparison fairer!

Q. Why Frequency of 1sec?

A. Frankly, I wasn't sure. This is based on the input I received during the previous Barometer weekly call.

Q. What about other metrics from the link?

A. These will be considered and studies only for those that support – as they are relevant to NFV

Recommendation

To make the comparison fair I would really advise the following:

- The publishing Mode – should really be writing somewhere else off the system – ideally some sort of time series DB… you want to minimize the impact of noise on the system
- You isolate and pin cores appropriately
- Footprints measurement process:
    - Measure Idle System resources usage
    - Run plugin/plugins combination - Measure System resources usage
    - Repeat tests on a busy System – or one running a workload.
    - Report results
    - Repeat with a busy system.
    - Metrics to collect:
        - Sysstat metrics
            - CPU   %user   %nice %system %iowait   %steal
            - Memory usage: kbmemfree kbmemused %memused kbbuffers kbcached kbswpfree
            - Cache thrashing if any
            - IO
                - tps – Transactions per second (this includes both read and write)
                - rtps – Read transactions per second
                - wtps – Write transactions per second
                - bread/s – Bytes read per second
                - bwrtn/s – Bytes written per second
- collectd/any other collector specific process stats if possible.
- Application stats for the application you are running – to determine the impact of collectd/other collectors on the workload.
    - You might pick a usecase with some network traffic – to see the impact on this if any.
    - Intervals: you might want to try 1 second, 10 seconds and 60 seconds… if possible you might drop below a second.

Process to be followed:

1. Isolate the CPUs on the monitoring node. [Added isolcpus option in the grub]
2. Run collectd on the isolated CPU. [Used taskset command to run collectd with appropriate CPU-mask]
3. Plugins: Make collectd to monitor following metrics [CPU, Memory, Disk, Interface, IPMI, processes, libvirt, Caches, OVS, hugepages]
4. Output: Make collectd to send metrics to influxdb running on separate node.
5. Workload: stress-ng + iperf.
6. Monitoring duration: 5 minutes.
7. Frequency: 1sec, 10 seconds, 60 seconds.
8. Collected Metrics to analyze collectd's runtime performance [Used Snap to collect 'collectd-process' metrics and CPU and memory data]
9. Note the iperf performance (to study any effect on it due to collectd]
10. Currently seeing if I can get more information from LTT-NG.

*** Repeat the above process for other monitoring agents ***

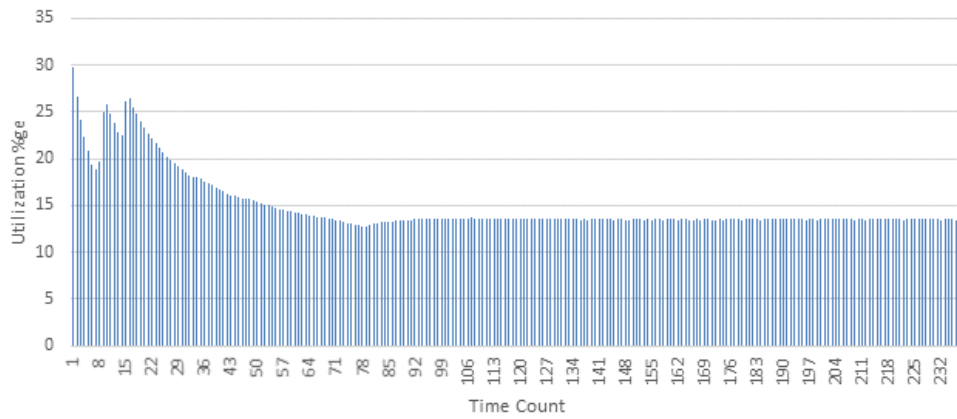## Steps:

Barometer- runti...lysis-Steps.docx
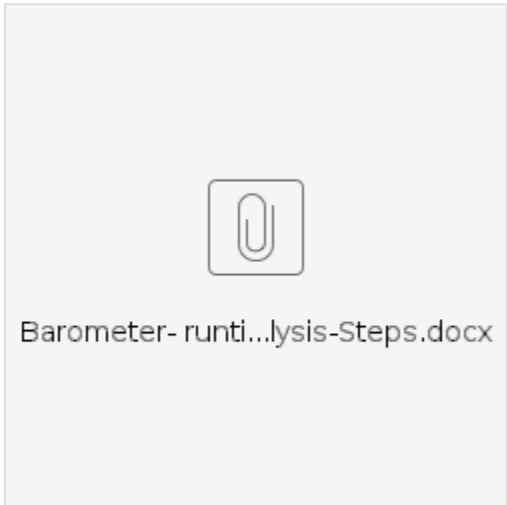
Initial findings

Below are plots for CPU Utilization Collectd and Telegraf. There isn't much difference between these agents. Snap was used to collect the process stats for both collectd and telegraf

Apart from the CPU data which is in the graphs below – the other comparative data are as follows:

	Telegraf	Collectd
Virtual Memory	345899008	958205952
RSS	7880	2066
Code	15036416	14442496
Data	321084800	831705088
StackSize	416	2288
iperf (550Mbps)	542	522

Collectd - CPU Utilization percentage

Utilization%ge (y-axis: 0, 5, 10, 15, 20, 25, 30, 35)

Time Count (x-axis: 1, 8, 15, 22, 29, 36, 43, 50, 57, 64, 71, 78, 85, 92, 99, 106, 113, 120, 127, 134, 141, 148, 155, 162, 169, 176, 183, 190, 197, 204, 211, 218, 225, 232)

Telegraf - CPU Utilization Percentage

Utilization % (y-axis: 0, 5, 10, 15, 20, 25, 30, 35)

Time Count (x-axis: 1, 8, 15, 22, 29, 36, 43, 50, 57, 64, 71, 78, 85, 92, 99, 106, 113, 120, 127, 134, 141, 148, 155, 162, 169, 176, 183, 190, 197, 204, 211, 218, 225, 232, 239, 246)