Monitoring IO on HPC Systems

23th June 2020

OVERVIEW

High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation, in order to solve large problems in science, engineering, or business. HPC Systems consists of many compute nodes that work together to complete one or more tasks. HPC Solutions have three main components

1. Compute

2. Network

3. Storage

To operate an HPC at maximum performance, each component must keep pace with others. For example, the storage component must be able to feed and ingest data to and from the compute servers as quickly as it is processed. If one component cannot keep up with the rest, the performance of the entire HPC infrastructure suffers thus we get to observe “bottleneck” in our system. HPC IO systems are built around a parallel file system that organizes storage and manages access. Parallel file systems are distributed systems that provide file data model to users. When storage performance/bottleneck issues are present, it impacts several resources attached to the storage system. Disk capacity, as well as storage I/O performance is critical to many users working in different environments. Applications that are I/O heavy often cause bottlenecks. I/O intensive applications are often more sensitive to a storage latency issue. Application bottlenecks can occur if you have multiple busy applications using the same datastore. Neither application is going to have optimum performance causing storage and application bottlenecks.

GOALS

1. Logging appropriate metrics for storage systems (eg. IOPS).
2. Using the logged data to draw conclusions about the performance achieved.
3. Utilizing various tools and techniques to design and make a cohesive tool to monitor I/O and other storage metrics.
4. Use these tools to identify the possible bottlenecks impacting performance of HPC.

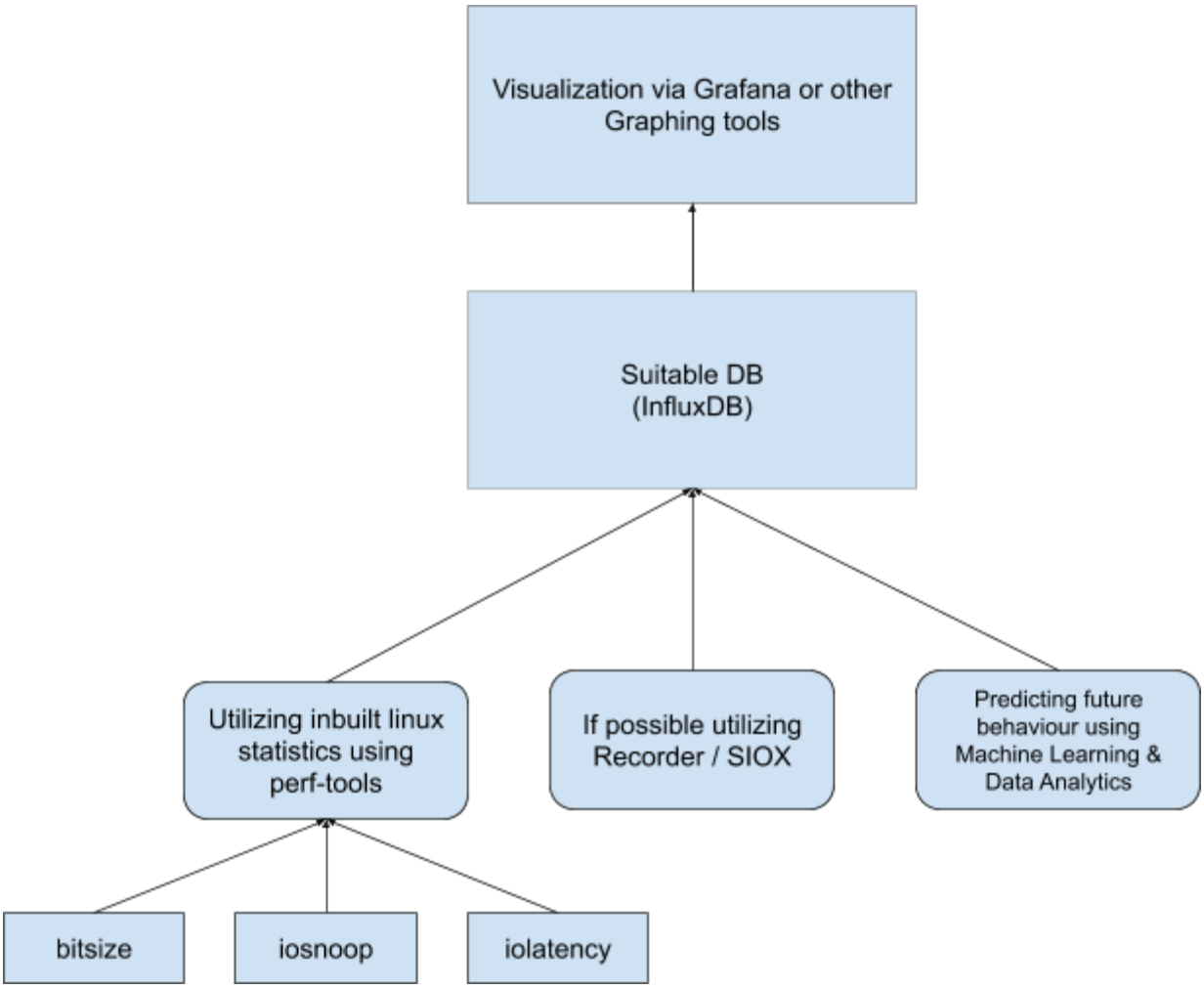
APPROACH

Essentially we can combine a variety of tools to form a cohesive suite which will provide you the bigger picture of the system in question. The I/O operations and other storage parameters will be logged and we can draw additional insights from the available data.

Open source tools which could be utilized:

1. Sysstat : https://github.com/sysstat/sysstat
2. Darshan : https://xgitlab.cels.anl.gov/darshan/darshan
3. Grafana : https://github.com/grafana/grafana
4. IOsnoop : https://github.com/brendangregg/perf-tools/blob/master/iosnoop
5. IOlatency : https://github.com/brendangregg/perf-tools/blob/master/iolatency
6. execSnoop : https://github.com/brendangregg/perf-tools/blob/master/execsnoop
7. Syscount : https://github.com/brendangregg/perf-tools/blob/master/syscount
8. Bitesize : https://github.com/brendangregg/perf-tools/blob/master/disk/bitesize
9. uiuc-Recorder : https://github.com/uiuc-hpc/Recorder
10. Iotop : http://guichaz.free.fr/iotop/
11. Collectl : http://collectl.sourceforge.net/
12. Collectd : https://collectd.org/index.shtml
13. Hpcmd : https://gitlab.mpcdf.mpg.de/mpcdf/hpcmd

PROPOSED METHODOLOGY



We can run our cohesive tool in the background on each compute node, perform measurements at regular intervals, and compute derived metrics. Measured values can be written to a log file and log files of each node can be combined in a central log system. After collecting the performance data we can have good insight into the system performance and possibly identify the bottlenecks.

Measurements (per node)

Disk Utilization	Latency (Request Response Time)
I/O Wait	Reads kBps
Writes / Sec	Writes kBps
Reads / Sec	Cache Hit - Miss statistics per node
Memory - Swap page statistics	Block I/O issued

General Architecture