Performance Benchmarking

FortiSOAR 7.3.1

FORTINET DOCUMENT LIBRARY
https://docs.fortinet.com

FORTINET VIDEO GUIDE
https://video.fortinet.com

FORTINET BLOG
https://blog.fortinet.com

CUSTOMER SERVICE & SUPPORT
https://support.fortinet.com

FORTINET TRAINING & CERTIFICATION PROGRAM
https://www.fortinet.com/training-certification

NSE INSTITUTE
https://training.fortinet.com

FORTIGUARD CENTER
https://www.fortiguard.com

END USER LICENSE AGREEMENT
https://www.fortinet.com/doc/legal/EULA.pdf

FEEDBACK
Email: techdoc@fortinet.com

February, 2023
FortiSOAR 7.3.1 Performance Benchmarking
00-400-000000-20210112

TABLE OF CONTENTS

Change Log

Date	Change Description
2023-02-16	Initial release of 7.3.1

FortiSOAR Performance Benchmarking for v7.3.1

This document details the performance benchmark tests conducted in Fortinet labs. The objective of this performance test is to measure the time taken to create alerts in FortiSOAR, and complete the execution of corresponding playbooks on the created alerts in the following cases:

- Single-node FortiSOAR appliance
- Cluster setup of FortiSOAR

The data from this benchmark test can help you in determining your scaling requirements for a FortiSOAR instance to handle the expected workload in your environment.

Summary of the tests performed

The following tests are performed on both a single FortiSOAR node, and a cluster of two active-active FortiSOAR nodes:

1. Ingestion of alerts into FortiSOAR.
2. Ingestion of alerts followed by extraction of associated indicators.
3. Ingestion of alerts, then extraction of associated indicators, followed by enrichment of the extracted indications using one source.
4. Sustain the processes of ingestion of alerts and extraction of indicators for a 12-hour period.

Environment

FortiSOAR Virtual Appliance Specifications

Component	Specifications
FortiSOAR Build Version	7.3.1-2105
CPU	16 CPUs
Memory	32 GB
Storage	300 GB virtual disk, HDD with IOPS 3000, attached to a VMware ESXI instance

Operating System Specifications

Operating System	Kernel Version
Rocky Linux 8.7	4.18.0-425.3.1.el8.x86_64

External Tools Used

Tool Name	Version
FortiMonitor	Cloud
Internal Script to gather data	

Pre-test conditions on both the standalone FortiSOAR machine and the FortiSOAR High Availability (HA) cluster

At the start of each test run -

- The test environment contained zero alerts.
- The test environment contained only the FortiSOAR built-in connectors such as IMAP, Utilities, etc.
- The system playbooks were deactivated such as, alert assignment notification, SLA calculation, etc., and there were no running playbooks.
- The playbook execution logs were purged.
- Configured tunables as follows:
  - Changed celery workers to 16
  - Elastic heaps size to 8GB
  - Changes related to PostgresSQL:
    - max_connections = 1200
    - shared_buffers = 5GB
    - effective_cache_size = 15GB
    - maintenance_work_mem = 2GB
    - default_statistics_target = 500
    - effective_io_concurrency = 2
    - work_mem = 546kB
    - min_wal_size = 4GB
    - max_wal_size = 16GB
    - max_parallel_workers_per_gather = 4
    - max_parallel_workers = 8
    - max_parallel_maintenance_workers = 4
  - Changes related to NGINX:
    - worker_processes auto;
    - worker_connections 1024;
    - access_log off;
    - keepalive_timeout 15;

In a production environment the network bandwidth, especially for outbound connections, to applications such as VirusTotal might vary, which could affect the observations.

Test setups

1. For a single-node FortiSOAR instance - Configure the standalone FortiSOAR appliance according to the specifications mentioned in the Environment topic.
2. For the FortiSOAR HA cluster - Create an HA cluster of two FortiSOAR instances that are joined in the active-active state. Configure both the FortiSOAR appliances according to the specifications mentioned in the Environment topic.

Tests performed

Test 1: Ingest alerts into FortiSOAR using the ingestion playbook

This test is executed by manually triggering ingestion playbook that creates alerts in FortiSOAR.

Steps followed

1. Created the alerts using the ingestion playbook.
   You can download and use the sample playbooks contained in "PerfBenchmarking_Test01_PB_Collection_7_3.zip" file if you want to run the tests in your environment. All the sample playbook collections are included in Appendix: JSONs for sample playbooks.
2. Once the alerts are created, measure the total time taken to create all the alerts in FortiSOAR.

Observations

The data in the following table outlines the number of alerts ingested and the total time taken to ingest those alerts:

Number of alerts created in FortiSOAR	Total number of playbooks executed in FortiSOAR	Total time (in seconds) taken to create all alerts in a standalone FortiSOAR appliance	Total time (in seconds) taken to create all alerts in an HA active-active FortiSOAR cluster **
1	1	0.35	0.38
5	1	0.56	0.61
10	1	0.86	0.90
25	1	1.71	1.92
50	1	3.22	3.64
100	1	6.82	7.27

** The slight overhead of workload distribution across the HA cluster outweighs the scaling benefits under very low load. That is why the total execution time for a single alert creation is more in the case of a two-node cluster.

Note: Once this test is completed, refer to the pre-test conditions before starting a new test.

Test 2: Ingest alerts into FortiSOAR followed by automated indicator extraction

This test is executed by manually triggering the ingestion playbook that creates alerts in FortiSOAR. Once the alerts are created in FortiSOAR, an "Extraction" playbook is triggered and the total time taken for all the extraction playbooks to complete their execution is calculated.

Steps followed

1. Created the alerts using the ingestion playbook.
2. Post alert creation, the "Extraction" playbooks are triggered. You can download and use the sample playbooks contained in the "PerfBenchmarking_Test02_PB_Collection_7_3.zip" file if you want to run the tests in your environment. All the sample playbook collections are included in Appendix: JSONs for sample playbooks.

Observations

The data in the following table outlines the number of alerts ingested, the total time taken to ingest those alerts, and the total time taken for all the triggered playbooks to complete their execution:

Number of alerts created in FortiSOAR	Total number of playbooks executed in FortiSOAR	Total time (in seconds) taken to create all alerts in a standalone FortiSOAR appliance	Total time (in seconds) taken to create all alerts in an HA active-active FortiSOAR cluster
1	2	1.43	1.53 **
5	6	1.94	1.81
10	11	2.82	2.51
25	26	7.17	5.55
50	51	9.49	8.19
100	101	14.61	11.62

** The slight overhead of workload distribution across the HA cluster outweighs the scaling benefits under very low load. That is why the total execution time for a single alert creation is more in the case of a two-node cluster.

Note: Once this test is completed, refer to the pre-test conditions before starting a new test.

Test 3: Ingest alerts into FortiSOAR, automated indicator extraction, followed by enrichment of the extracted indicators

The test was executed using an automated testbed that starts the ingestion which in turn creates alerts in FortiSOAR. Once the alerts are created in FortiSOAR, "Extraction" and "Enrichment" playbooks are triggered and the total time taken for all the extraction and enrichment playbooks to complete their execution is calculated.

The setup for this test is exactly the same as Test 2, however this test additionally requires the "VirusTotal" connector to be configured.

Steps followed

1. Created the alerts using the ingestion playbook.
2. Post alert creation, the "Extraction" and "Enrichment" playbooks are triggered. You can download and use the sample playbooks contained in the "PerfBenchmarking_Test03_PB_Collection_7_3.zip" file if you want to run the tests in your environment. All the sample playbook collections are included in Appendix: JSONs for sample playbooks.

Observations

The data in the following table outlines the number of alerts ingested, the total time taken to ingest those alerts, and the total time taken for all the triggered playbooks to complete their execution.

Number of alerts created in FortiSOAR	Total number of playbooks executed in FortiSOAR	Total time (in seconds) taken to create all alerts in a standalone FortiSOAR appliance	Total time (in seconds) taken to create all alerts in an HA active-active FortiSOAR cluster
1	8	5.46	5.48
5	36	5.38	6.12
10	71	9.74	7.35
25	176	20.3	15.19
50	351	34.87	27.56
100	700	1 minute 4 seconds	48.35

Note: Once this test is completed, refer to the pre-test conditions before starting a new test. Also, note that enrichment of playbooks makes API calls over the Internet, and the times mentioned in this table to execute playbooks is inclusive of this time.

Sustained Data Ingestion Test

A sustenance test was also conducted on a standalone FortiSOAR appliance with the configuration as defined in "Test 2", i.e., the test is executed by manually triggering the ingestion playbook, “FortiSIEM -> Ingest 100 Alerts”, which creates alerts in FortiSOAR. Once the alerts are created in FortiSOAR, an "Extraction" playbook is triggered and the total time taken for all the extraction playbooks to complete their execution is calculated.

Number of alerts: **100/min**

Duration: **12 hours**

Playbooks configured: **As defined in Test 2 comprising of "Ingestion" and "Indicator Extraction" playbooks**.

Total number of playbooks executed: **72720**

Results

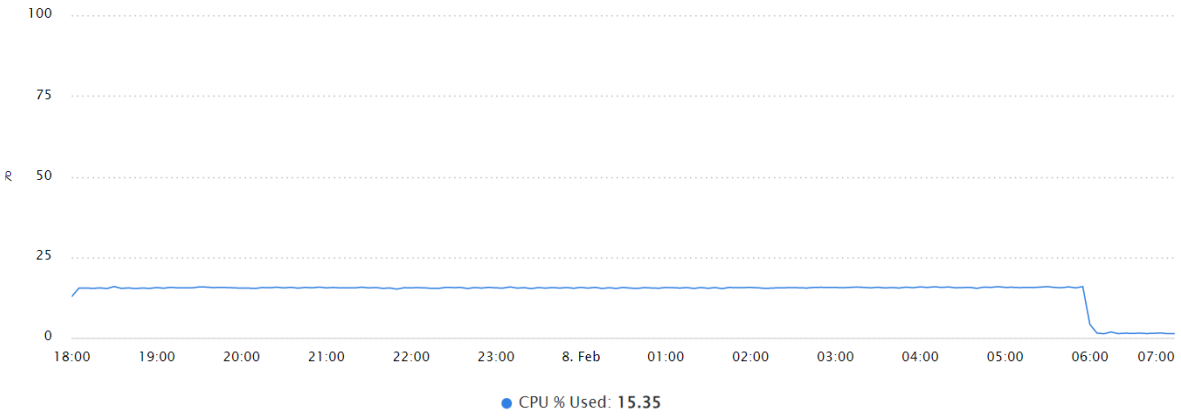
The system performed well under the sustained load. All 72100 alerts were successfully ingested and all the extraction playbooks were successfully completed without any queuing.

Graphs

The following graphs are plotted for the vital statistics for the system that was under test during the period of the test run.

CPU Percentage Usage Graph

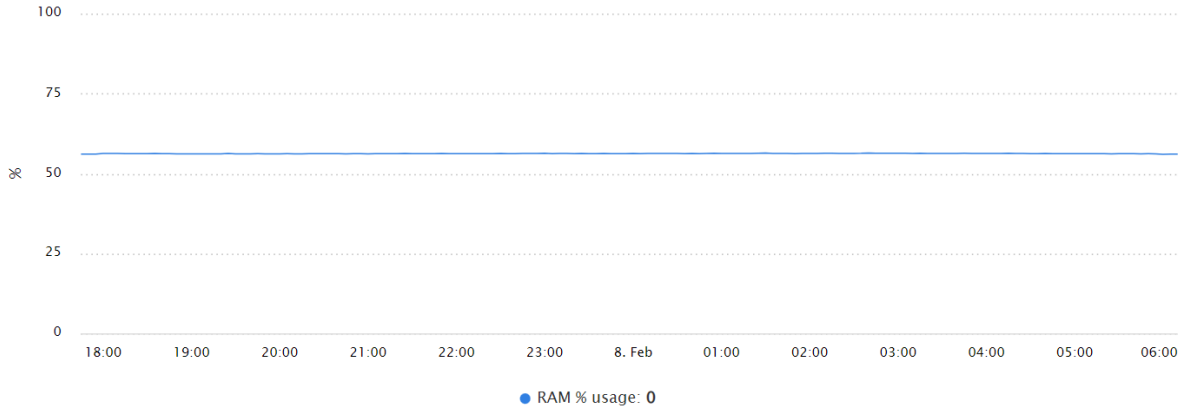
Analysis of CPU Percentage Usage when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test" was running the CPU utilization was normal and the performance of the system did not get impacted.

Memory Utilization Graph

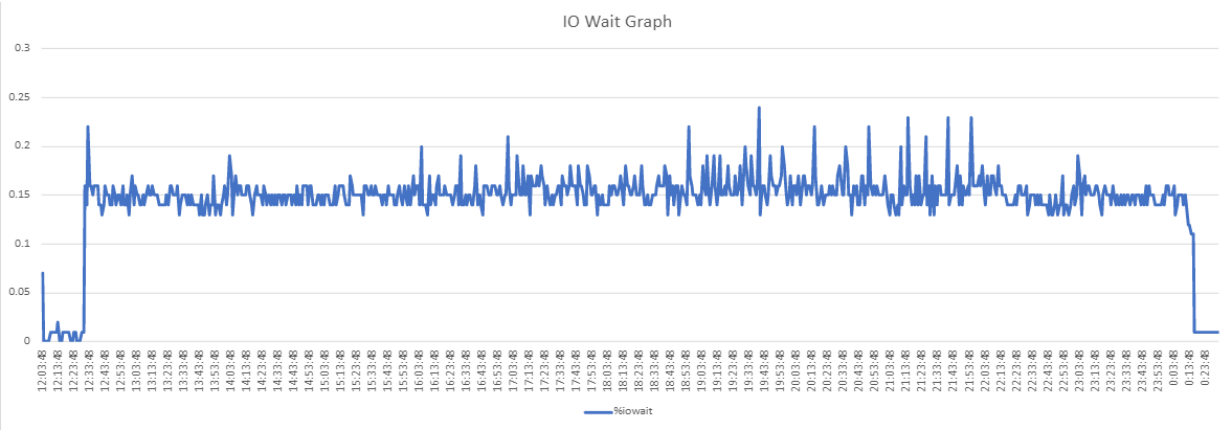
Analysis of memory utilization when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test" was running the Memory utilization was around 55%.

IO Wait Graph

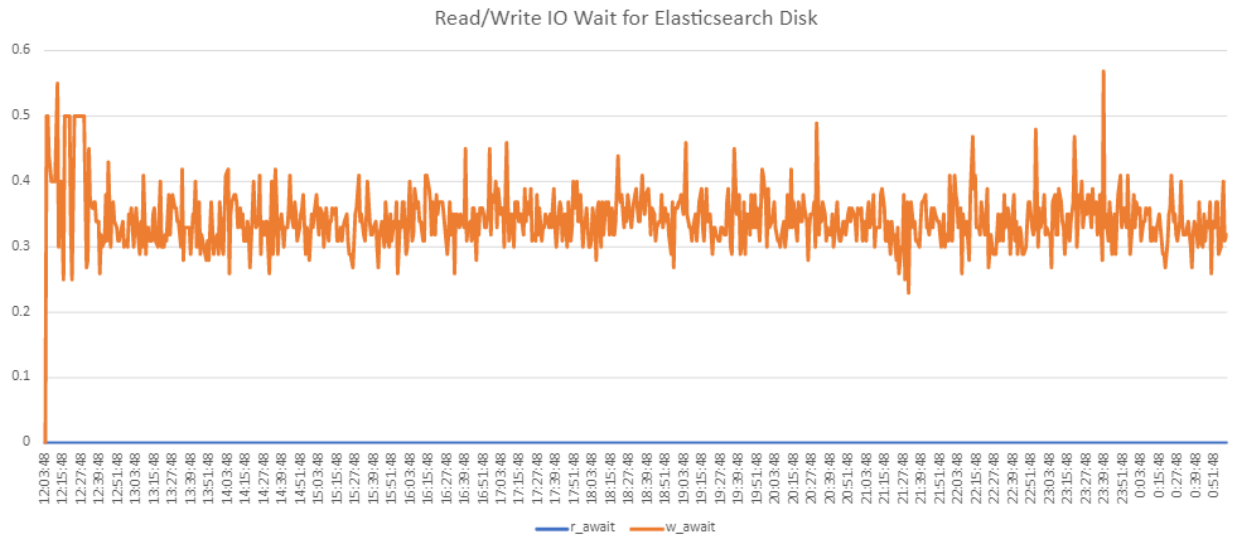
Analysis of IO Wait when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test" was running the IO Wait time was normal, with the average IO Wait being around 1% of the CPU idle time.

Read/Write IO Wait Graph for ElasticSearch

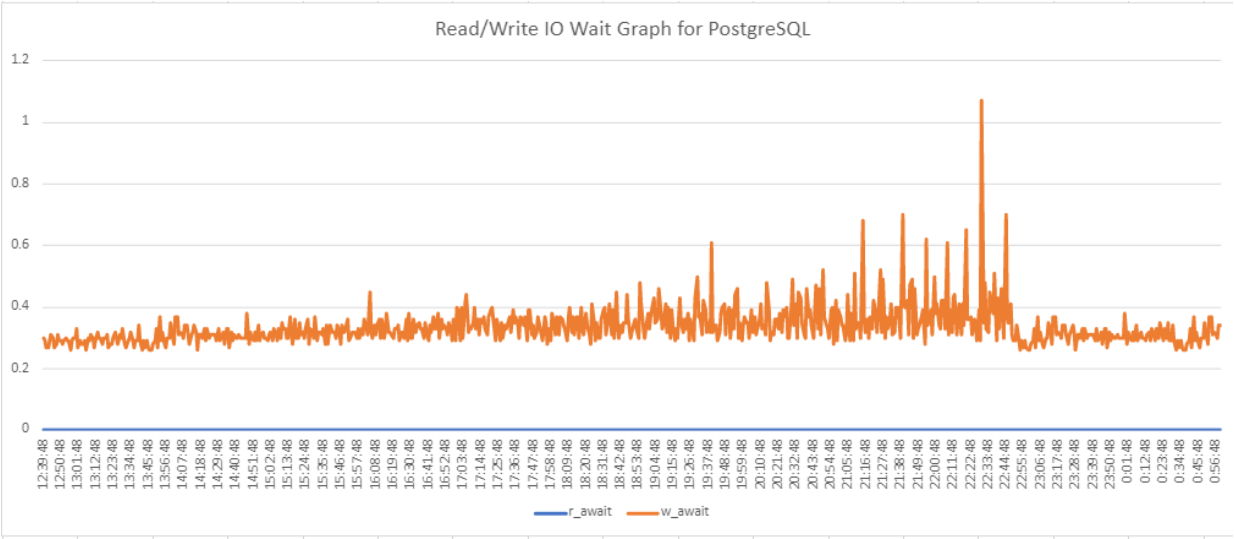
Analysis of Read/Write IO Wait for ElasticSearch when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test" was running the "Read" Wait for the ElasticSearch disk was almost 0 milliseconds. The "Write" Wait for the ElasticSearch disk averaged around 0.4 milliseconds, with the maximum wait of 0.55 milliseconds and the minimum wait of 0.25 milliseconds.

Read/Write IO Wait Graph for PostgreSQL

Analysis of Read/Write IO Wait for PostgreSQL when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test" was running, the "Read" Wait for the PostgreSQL disk averaged around 0 milliseconds. The "Write" Wait for the PostgreSQL disk averaged around 0.3 milliseconds, with the maximum wait of 1.1 milliseconds and the minimum wait of 0.21 milliseconds.

Sustained Data Ingestion Test for the cluster of two active-active FortiSOAR nodes

A sustenance test was also conducted on the FortiSOAR active-active two node cluster with the configuration as defined in "Test 2", i.e., the test is executed by manually triggering the ingestion playbook, “FortiSIEM -> Ingest 100 Alerts”, which creates alerts in FortiSOAR. Once the alerts are created in FortiSOAR, an "Extraction" playbook is triggered and the total time taken for all the extraction playbooks to complete their execution is calculated.

Number of alerts: **100/min**

Duration: **12 hours**

Playbooks configured: **As defined in Test 2 comprising of "Ingestion" and "Indicator Extraction" playbooks**.

Total number of playbooks executed: **72720**

Results

The system performed well under the sustained load. All 72100 alerts were successfully ingested and all the extraction playbooks were successfully completed without any queuing.

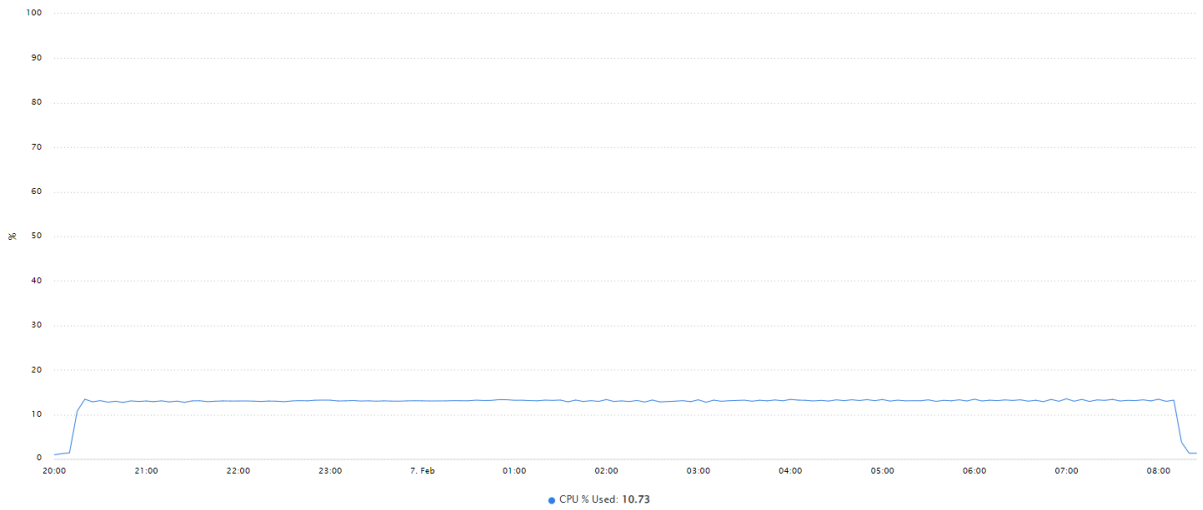
Graphs

The following graphs are plotted for the vital statistics for the HA cluster that was under test during the period of the test run.

All the graphs included in this section are from the Primary/Active Node.

CPU Percentage Usage Graph

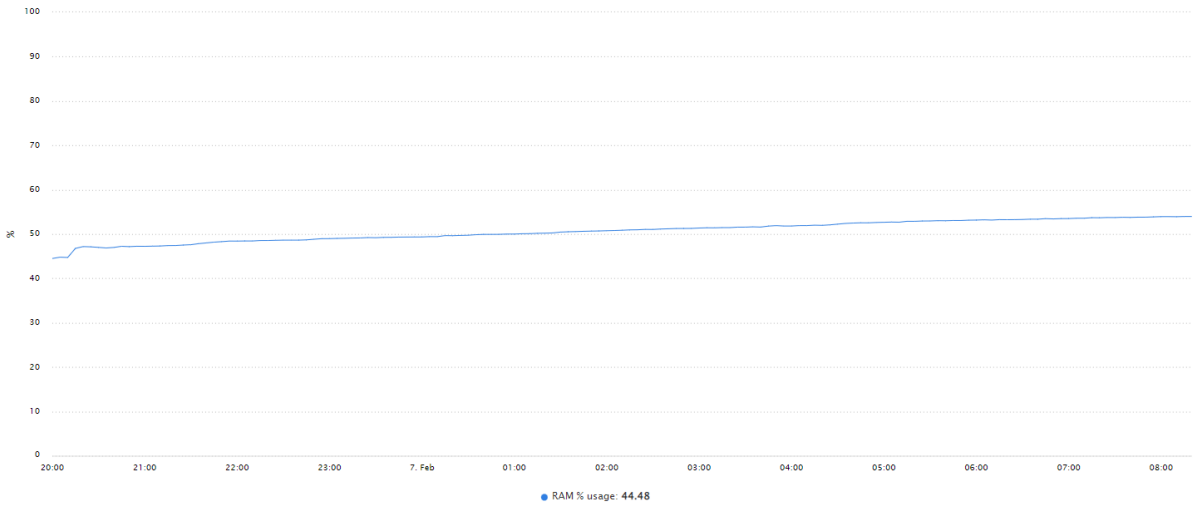
Analysis of CPU Percentage Usage when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test for an HA cluster" was running the CPU utilization was normal and the performance of the system did not get impacted.

Memory Utilization Graph

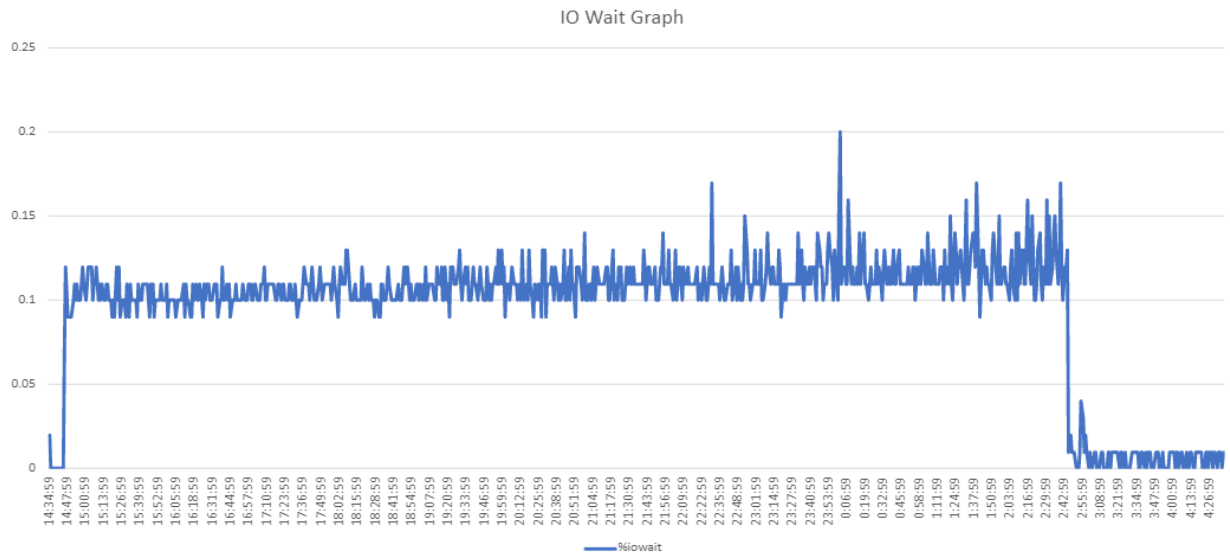
Analysis of memory utilization when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test for an HA cluster" was running the Memory utilization was around 55%.

IO Wait Graph

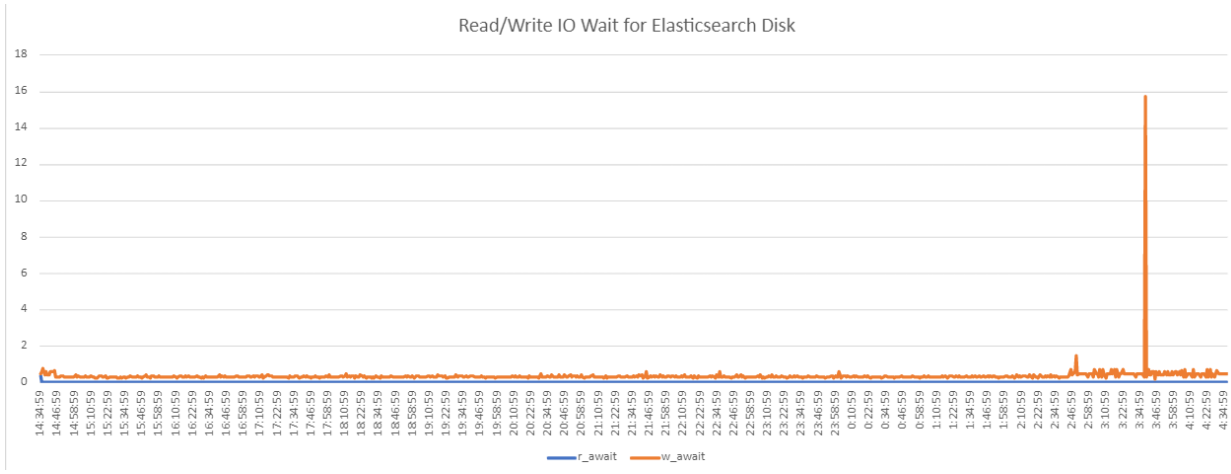
Analysis of IO Wait when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test for an HA cluster" was running the IO Wait time was normal.

Read/Write IO Wait Graph for ElasticSearch

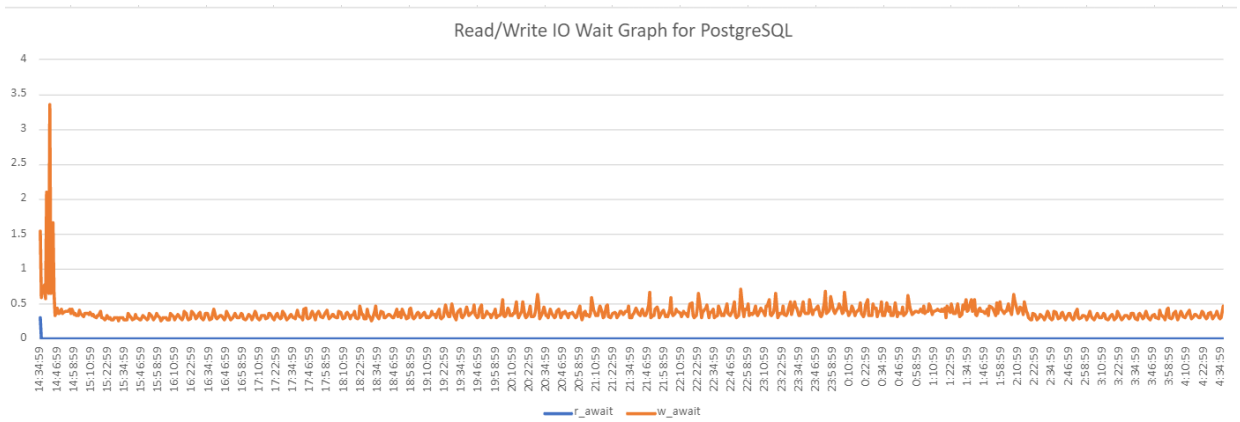
Analysis of Read/Write IO Wait for ElasticSearch when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test for an HA cluster" was running the "Read" Wait for the ElasticSearch disk was almost 0 milliseconds. The "Write" Wait for the ElasticSearch disk averaged around 1 millisecond, with the maximum wait of 16 milliseconds and the minimum wait of 0.3 milliseconds.

Read/Write IO Wait Graph for PostgreSQL

Analysis of Read/Write IO Wait for PostgreSQL when the test run was in progress on the appliance:



Using the system resources specified in the "Environment" and "Pre-Test Conditions" topics, it was observed that when the "Sustained Data Ingestion Test for an HA cluster" was running, the "Read" Wait for the PostgreSQL disk averaged around 0 milliseconds. The "Write" Wait for the PostgreSQL disk averaged around 0.5 millisecond, with the maximum wait of 3.4 milliseconds and the minimum wait of 0.2 milliseconds.

Appendix: JSONs for sample playbooks

You can download the JSON for the following sample playbook collections so that you can run the same tests in your environment to see the performance in your version/hardware platforms. Or, if you want to make some additions that are specific to your environment, you can also tweak the existing playbooks.

- PerfBenchmarking_Test01_PB_Collection_7_3.zip
- PerfBenchmarking_Test02_PB_Collection_7_3.zip
- PerfBenchmarking_Test03_PB_Collection_7_3.zip

www.fortinet.com