Performance Testing Report

Objective

The purpose of load testing and performance validation is to evaluate the system's ability to handle high log ingestion rates while maintaining stability and efficiency. This test aims to:

- Validate capacity limits by determining the maximum events per second (EPS) the system can process before performance starts to degrade.
- Ensure system stability under normal and peak workloads.
- Identify bottlenecks in log processing, query response time, and resource usage.
- Verify Graylog’s alerting and indexing performance under stress conditions.

KPIs & Baselines

The following Key Performance Indicators (KPIs) will be used to measure system performance:

Log Ingestion Rate (EPS - Events Per Second)

System should be able to sustain the expected EPS without performance degradation.

- Production Target: 10,000 EPS
- Prototype Target: 500–1,000 EPS

Query Response Time

We’ll run queries at various load levels to check response time.

- Target: Queries should return results within 5 seconds.

System Resource Usage

System metrics will be monitored during testing.

- Target:
  - CPU Usage: Graylog and Elasticsearch should not exceed 80% CPU usage.
  - Memory Usage: Should remain below 70%.
  - Disk I/O Utilisation: Should not exceed 85% to avoid slowdowns.

Log Processing Latency

We'll inject logs and measure the delay before they appear in search results.

- Target: Logs should be indexed and searchable within 3-5 seconds of ingestion.

Alert Generation Speed

We'll simulate security events and measure the delay before alerts appear.

- Target: Security alerts should trigger within 5 seconds of event detection.

System Stability & Failure Rate

We'll gradually increase EPS beyond expected loads until failures appear.

- Target: No crashes, missed logs, or significant slowdowns.

Test Scenarios

To evaluate the system's performance, logs will be generated using Logstash and injected into Graylog. Different test batches will be used to simulate varying log loads and analyse the system's response.

1. Baseline Performance Test

- **Objective:** Establish normal system behavior under expected production loads.
- **Method:**
    - Use Logstash to generate 500 EPS for 10 minutes.
    - Measure CPU, memory, disk usage, and query response time.
    - Check if logs are indexed within 5 seconds.
    - Monitor alert generation speed.

2. Stress Test - Increasing Log Volume

- **Objective:** Determine the maximum log ingestion capacity before performance degradation.
- **Method:**
    - Start at 600 EPS and gradually increase by 100 EPS every 3 minutes.
    - Continue until system bottlenecks appear (e.g., delays, dropped logs, resource exhaustion).

    - Record the highest sustainable EPS before failures occur.

3. Query Performance Under Load

- **Objective:** Test query response time under high system load.
- **Method:**
    - Inject 1,000 EPS for 10 minutes.
    - Simultaneously execute complex search queries in Graylog.
    - Measure query response time and compare against baseline.

Results & Findings

Test Scenario 1: Baseline Performance Test

Steps Taken:

- I configured Logstash to generate 500 Events Per Second (EPS) for 10 minutes.
- Then monitored CPU, memory, and disk usage in real-time using *htop* command.
- I collected system performance data using the *sar* command over a 10-minute period, which generated logs for CPU, memory, and disk usage.

Findings:

- **CPU and Memory Usage:** Both remained consistently above 80%, indicating high resource utilisation even under normal/minimal load conditions.
- **Disk I/O Performance:** Disk usage showed moderate write operations, with no observed bottlenecks.
- **Log Ingestion Time:** Logs were indexed within 5 seconds, meeting the expected benchmark.

Although the system functioned within expected parameters, it performed inadequately for a production environment. I initially estimated that the system would handle between 500-1000 EPS, but resource utilisation was already high at 500 EPS, which was my minimum benchmark. The consistently high CPU and memory usage suggests that we will need to optimise our SIEM environment further if we plan to scale beyond 500 EPS.

Test Scenario 2: Stress Test - Increasing Log Volume

Steps Taken:

- Since I had already observed how 500 EPS impacted the system, in the first test, I started this test at 600 EPS.
- I planned to increase log ingestion by 100 EPS every 3 minutes until the system showed signs of degradation.
- Then I monitored CPU, memory, and disk usage throughout the test.
- I identified the highest stable EPS before performance degradation.

Findings:

- **System Degradation at 600 EPS:**
  - Initially, resource usage spiked, but logs were still indexed within 5 seconds.
  - However, shortly after increasing the load, the host system's disk became full, causing VirtualBox to pause the VM.
- **Error Messages Encountered:**
  - BLKCACHE_IOERR: The I/O cache failed due to insufficient disk space. I attempted to resolve this by:
    - Enabling *"Use Host I/O Cache"*
    - Increasing VM storage from **50GB to 60GB**
  - DrvVD_DISKFULL: The host machine itself ran out of space, suspending VM execution. Additional actions I took:
    - Increased storage from **60GB → 70GB**
    - Increased RAM from **6000MB → 7000MB**
    - Increased CPU cores from **3 → 4**
- After researching the error messages, I realised the VM failure was due to insufficient disk space on the host machine, not the VM itself. The higher log ingestion rate caused an increase disk I/O operations, then the host machine which already had limited storage, was exhausted. As a result, VirtualBox paused the VM, preventing it from running.
- My initial attempts, such as increasing the VM's RAM, CPU cores, and storage, did not address the root cause, as the bottleneck was on the host system, not the VM. After checking the host system's disk usage, I confirmed that it was completely full. The final solution was to free up space on the host machine before resuming testing.

Test Scenario 3: Query Performance Under Load

Steps Taken:

- I injected 500 EPS and 600 EPS into Graylog.
- I executed simple queries under both loads to measure response times.
- I compared the query response times against the baseline requirement.

Findings:

- **Baseline Query Response:** Before increasing the load, query responses were under 3 seconds.
- **Query Performance at 500 and 600 EPS:**
    - Response time slowed but remained within 5 seconds, which is at the borderline of acceptable performance.
    - I was unable to extensively test complex queries under high loads due to system failures encountered in Scenario 2. However, from my limited testing, the query performance did not show significant degradation under moderate load.

Recommendations

1. Optimise System Resources

- CPU and memory usage remained high even under normal loads. To improve performance, we should:
    - Increase available system resources (e.g., allocate more CPU and memory).
    - Optimise Logstash configurations to reduce unnecessary processing overhead.

2. Ensure Sufficient Disk Space

- The host machine's lack of available storage became a bottleneck that we did not anticipate.
- For future tests, we recommend:
    - Allocating at least 30% more storage than the expected log ingestion demands.

    - Implementing automated log rotation and archiving will prevent disk exhaustion by regularly removing or compressing old logs. Additionally, creating dedicated streams for different log sources allows for custom retention policies, ensuring critical logs are stored longer while less important logs are deleted sooner. This optimizes storage usage and query performance, reducing unnecessary resource consumption.

3. Improve Query Efficiency

- Test index optimisations such as:
    - Reducing retention periods for older logs.
    - Optimising Elasticsearch settings to enhance search performance.
- Conduct further testing at higher EPS rates with more complex queries.

4. Implement Continuous Performance Monitoring

- To detect early signs of performance degradation, we should:
    - Use real-time monitoring tools for Graylog and system resources.
    - Set up alerting thresholds for CPU, memory, disk usage, and log ingestion delays.

I wasn’t able to test the alerting system because it hadn’t been fully set up at the time of testing.

Conclusion

This performance test revealed key insights into how the system handles high log volumes and stress conditions. While it meets basic needs at moderate loads (500 EPS), it struggles to scale efficiently—CPU and memory usage often spike beyond acceptable levels, even under minimal strain. Unexpected storage limits on the host machine also caused reliability issues.

To improve performance at higher EPS rates, we’ll need to optimise resources and manage disk space more proactively. Simple fixes like log rotation, tuning Logstash, and adjusting Elasticsearch settings should boost efficiency. Next, we should test query performance under heavy loads and ensure alerts work reliably. With these tweaks, the system will be far better suited to production demands.