Link to GitHub Page:
https://github.com/amosproj/amos2025ws01-opensearch-load-tester/wiki/User-Documentation

Environment Variables

This page documents all environment variables used in the OpenSearch Load Tester project. Variables are grouped by service:

- TestDataGenerator
- LoadGenerator
- MetricsReporter
- OpenSearch Test Target

Each variable includes its default value, description, and when it should be changed.

TestDataGenerator – Environment Variables

These variables configure how test data is generated and preloaded into OpenSearch.

Table: TestDataGenerator Variables

Variable	Default	Description	Usage
SPRING_APPLICATION_NAME	testdata-generator	Name of the Spring Boot application.	Mainly for logs; normally not changed.
OPENSEARCH_URL	http://test-target-opensearch:9200/	URL of the OpenSearch instance to write test data into.	Change if OpenSearch runs under a different host or port.
DATA_GENERATION_MODE	dynamic	Defines how documents are created.	`dynamic` generates data at runtime; extendable for future modes.
DATA_GENERATION_DOCUMENT_TYPE	ano	Type of document to generate.	`Use ano or duo depending on which index type you want.`
DATA_GENERATION_COUNT	10	Total number of documents to create.	`Increase to preload larger datasets.`
LOGGING_LEVEL_	ERROR	Global logging level.	`Change to INFO`

ROOT			`or DEBUG when troubleshooting.`
LOGGING_LEVEL_OR_ELASTICSEARCH	ERROR	Logging level for OpenSearch client.	`Normally unchanged.`
LOGGING_LEVEL_COM_OPENSEARCHLOADTESTER_TESTDATAGENERATOR	INFO	Logging for TestDataGenerator package.	`Increase to DEBUG for deep debugging.`

LoadGenerator – Environment Variables

These variables define how queries are executed and how performance metrics are sent to the MetricsReporter.

Table: LoadGenerator Variables

Variable	Default	Description	Usage
SPRING_APPLICATION_NAME	load-generator	Name of the Spring Boot application.	Normally not changed.
OPENSEARCH_URL	http://test-target-opensearch:9200/	URL of the OpenSearch instance the LoadGenerator queries.	Change if OpenSearch runs elsewhere.
METRICS_REPORTER_URL	http://metrics-reporter:8080/api/	Base URL for metrics submission.	Modify if MetricsReporter runs on another host/port.
LOGGING_LEVEL_ROOT	ERROR	Global logging level.	Change to `INFO` or `DEBUG` when analyzing runs.
LOGGING_LEVEL_ORG_ELASTICSEARCH	ERROR	Logging for OpenSearch client.	Rarely changed.
LOGGING_LEVEL_COM_OPENSEARCHLOADTESTER_LOADGENERATOR	INFO	Logging for LoadGenerator package.	Increase to `DEBUG` during troubleshooting.

SCENARIO_CONFIG_PATH${SCENARIO_CONFIG_PATH:-scenarios/default-scenario.yaml}	Path to the scenario YAML file used by the LoadGenerator.	Override to change which scenario is executed.	

MetricsReporter – Environment Variables

These variables control how metrics are aggregated and exposed.

Table: MetricsReporter Variables

Variable	Default	Description	Usage
SPRING_APPLICATION_NAME	metrics-reporter	Application name.	For logs/dashboards.
LOAD_GENERATOR_REPLICAS	1	Expected number of load-generator replicas.	Bump when scaling out generators.
REPORT_OUTPUT_DIRECTORY	./reports	Directory where reports are written.	Point to persistent storage if needed.
REPORT_JSON_FILENAME	report.json	JSON report filename.	Change to avoid collisions per run.
REPORT_CSV_FILENAME	report.csv	CSV report filename.	Change to avoid collisions per run.
REPORT_EXPORT_JSON_ENABLED	true	Toggle JSON export.	Set false to skip JSON.
REPORT_EXPORT_CSV_ENABLED	true	Toggle CSV export.	Set false to skip CSV.
LOGGING_LEVEL_ROOT	ERROR	Global logging level.	Set to INFO/DEBUG when debugging.
LOGGING_LEVEL_ORG_ELASTICSEARCH	ERROR	Logging for Elasticsearch/OpenSearch client.	Usually unchanged.
LOGGING_LEVEL_COM_OPENSEARCHLOADTESTER_METRICSREPORTER	INFO	Logging for Metrics Reporter package.	Increase to DEBUG for troubleshooting.

OpenSearch Test Target – Environment Variables

These variables configure the embedded OpenSearch test cluster.

Table: OpenSearch Target Variables

Variable	Default	Description	Usage
DISABLE_SECURITY_PLUGIN	true	Disables authentication and TLS.	Keep true for local development/testing.
discovery.type	single-node	Starts OpenSearch as a single-node cluster.	Change only for multi-node setups.
OPENSEARCH_JAVA_OPTS	-Xms512m -Xmx512m	JVM heap size.	Increase for heavy loads or large datasets.

Common Logging Variables

The following variables may be used across multiple services:

Variable	Description
LOGGING_LVL_ROOT	Overrides global logging level.
LOGGING_LVL_ELASTIC	Controls all OpenSearch client logging.
LOGGING_LVL_TESTDATA_GEN	Logging level for TestDataGenerator.
LOGGING_LVL_LOAD_GEN	Logging level for LoadGenerator.
LOGGING_LVL_METRICS_REPORTER	Logging level for MetricsReporter.

These allow centralized configuration via a `.env` file.

How to Override Environment Variables

Using a `.env` file

```
OPENSEARCH_URL=http://localhost:9200
DATA_GENERATION_COUNT=5000
LOGGING_LVL_ROOT=INFO
```

Overriding inline

```
docker compose run -e DATA_GENERATION_COUNT=20000 testdata-generator
```

Overriding inside docker-compose.yml

```
environment:
  - DATA_GENERATION_COUNT=20000
```

YAML Scenario Configuration

The **Load Generator** supports scenario-based configuration using YAML files.
Each scenario YAML file defines how a load test behaves: duration, QPS, concurrency, and the type of queries executed against OpenSearch.

Scenarios are stored in:

```
./load-generator/scenarios/
```

The scenario used at runtime is selected via:

```
SCENARIO_CONFIG_PATH
```

Default:

```
scenarios/default-scenario.yaml
```

---

1. Purpose of Scenario YAML Files

A Scenario YAML file allows users to:

- Define **how long** a test should run
- Control **queries per second (QPS)**
- Configure **warm-up** and **ramp-up** phases
- Adjust **concurrency parameters** (clients, threads)
- Specify **which query type** the LoadGenerator should execute
- Pass **query parameters** (e.g., year ranges)

---

2. Scenario File Structure

Below is the default scenario configuration included in the project:

```yaml
name: default-scenario
document_type: ANO
duration: PT5M
queries_per_second: 50
warm_up_duration: PT30S
ramp_up_duration: PT1M
concurrency:
  client_size: 10
  thread_pool_size: 5
query:
  type: YEAR_RANGE
  parameters:
    from: 2010
    to: 2025
```

---

3. Field-by-Field Explanation

This table explains each component of the scenario YAML.

Scenario Metadata

Field	Example	Allowed Values / Options	Explanation
name	`default-scenario`	Any text (string)	A human-readable scenario name used to identify test configurations. No functional effect.
document_type	`ANO`	`ANO`, `DUO`	Specifies which dataset/index the scenario targets. Must match what the TestDataGenerator created.
duration	`PT5M`	`ISO-8601 durations:• PT10S (10s)• PT30S (30s)• PT1M (1 min)•`	Total duration of the load test.

		`PT5M (5 min)• PT30M, PT1H`	
queries_per_second	`50`	`Any positive integer:• Light: 1–10• Normal: 10–50• Heavy: 100–500`	Target QPS the load generator attempts to maintain. Higher values = more stress on OpenSearch.
warm_up_duration	`PT30S`	`ISO-8601 durations`	Time period before the main test where load increases gradually.
ramp_up_duration	`PT1M`	`ISO-8601 durations`	Time over which the system slowly ramps to the full QPS.
concurrency.client_size	`10`	`Integer 1–200`	Number of logical “clients”. Represents user sessions simulated concurrently.
concurrency.thread_pool_size	`5`	`Integer 1–100`	Number of threads sending requests in parallel. Higher value increases CPU use and request throughput.
query.type	`YEAR_RANGE`	`Currently supported:• YEAR_RANGE`	Defines the query pattern the LoadGenerator executes.
query.parameters	`2010`	`Type needed by parameter.`	Any paramenter needed by query.

The `parameters` section depends on the selected query type.

---

4. Changing the Scenario File

You can create multiple scenario files, for example:

`scenarios/heavy-load.yaml`

```yaml
name: heavy-load
document_type: ANO
duration: PT5M
queries_per_second: 200
warm_up_duration: PT1M
ramp_up_duration: PT2M
concurrency:
  client_size: 50
  thread_pool_size: 20
query:
  type: YEAR_RANGE
  parameters:
    from: 2024
    to: 2025
```

Then select it using:

Option A — `.env` file

```
SCENARIO_CONFIG_PATH=scenarios/heavy-load.yaml
```

Option B — Inline override

```
SCENARIO_CONFIG_PATH=scenarios/heavy-load.yaml docker compose up
load-generator
```

Option C — Modify docker-compose

```yaml
environment:
  - SCENARIO_CONFIG_PATH=scenarios/heavy-load.yaml
```

---

5. Creating Your Own Scenario (Step-by-Step)

1. Create a new `.yaml` file in:

```
load-generator/scenarios/
```

1. Add your scenario fields.
2. Adjust:
    a. duration
    b. QPS
    c. query type
    d. concurrency

3. Select it using:

```
SCENARIO_CONFIG_PATH=scenarios/my-scenario.yaml
```

1. Start the system:

```
make build && make run
```
