Performance and Scalability Test Plans

The following tests are designed to demonstrate how "performance" of the Fedora repository changes as the size of the repository grows under various, but discrete uses.

Note: "Performance" will be measured by requesting CRUD operations after every x-number of ingest events.

Setup

Along with the following tests, the following items will also be documented:

- Version of Fedora
- Fedora configuration details
- System details (OS, memory, processors, hardware specs or virtualization, JVM, etc)
- Initial state of the repository - likely empty
- Number of client processes/threads (ideally clients are on a separate machine)

Tests

The following measurements should be recorded:

1. Number of resources in the repository
2. Average response times of the 1000-request blocks
3. Individual response times of the *performance requests* (see below)
4. System resource usage on an on-going basis or at the 1000-request boundaries, as appropriate
    a. memory
    b. inode count
    c. disk usage (including swap space)
    d. garbage collection
    e. CPU usage

Each of the tests below should be run with different numbers of clients

1. First run: 1 client
2. Second run: 2 clients
3. Third run: 4 clients
4. Fourth run: 8 clients

All tests should be scripted using the JMeter framework. The additional advantage is the potential use of those same JMeter scripts to outsource large-scale testing to https://blazemeter.com/.

- The current test plans are available in https://github.com/fcrepo4-labs/fcrepo4-jmeter
- For informational purposes, outdated Fedora4/JMeter scripts exist as a starting point: https://github.com/fcrepo4-archive/ff-jmeter-madness

Test Completion

All of the tests below should continue to run until one of the following events take place:

1. A pre-defined number of resources have been created
2. The collective response time of the *performance requests* exceeds 1 minute
3. Garbage collection is running non-stop

Test 1 - Size of files - large

1. Load different "large" files with POST requests at the top of the repository ranging between 10KB and 10GB
2. After every 1000 requests, *performance requests* should be made

Note: Fedora stores files based on their SHA1. In order for this test to be effective, each file should have a different SHA1.

Test 2 - Size of files - small

1. Load different "small" files with POST requests at the top of the repository all below the default 4096 Byte threshold
2. After every 1000 requests, *performance requests* should be made

Note: Fedora stores files based on their SHA1. In order for this test to be effective, each file should have a different SHA1.

Test 3 - Number of files

1. Load different files with POST requests at the top of the repository ranging between 10KB and 100KB
2. After every 1000 requests, *performance requests* should be made

Test 4 - Number of containers - default

1. Create containers with POST requests at the top of the repository
2. After every 1000 requests, *performance requests* should be made
3. Results

Test 5 - Number of containers - RDF bodies

1. Create containers with POST requests with RDF bodies at the top of the repository
2. After every 1000 requests, *performance requests* should be made

Test 6 - Number of mixed resources - files and containers

1. Create a mix of resources representative of a typical LDP model including containers and files
2. After every 1000 requests, *performance requests* should be made

Performance requests

1. PUT a new resource
2. OPTIONS the resource
3. GET the resource
4. PATCH a property to the resource
5. DELETE the resource

Logs

- JMeter
  - {test-name}-summary.log
  - apache-jmeter-2.13/bin/log/perf.log
  - apache-jmeter-2.13/bin/log/jmeter-{test-name}-threads.csv
- Tomcat
  - catalina.out
  - local_access_log.txt
  - java-gc.log
- MySQL
  - error.log
- PostgreSQL
  - postgresql-ver-main.log