**Exercise 7c**

*Simple Benchmarking with Siege*

**Prior Knowledge**

Previous exercises

**Objectives**

Benchmarking runtimes

**Software Requirements**

* Java Development Kit 8
* Redis
* siege

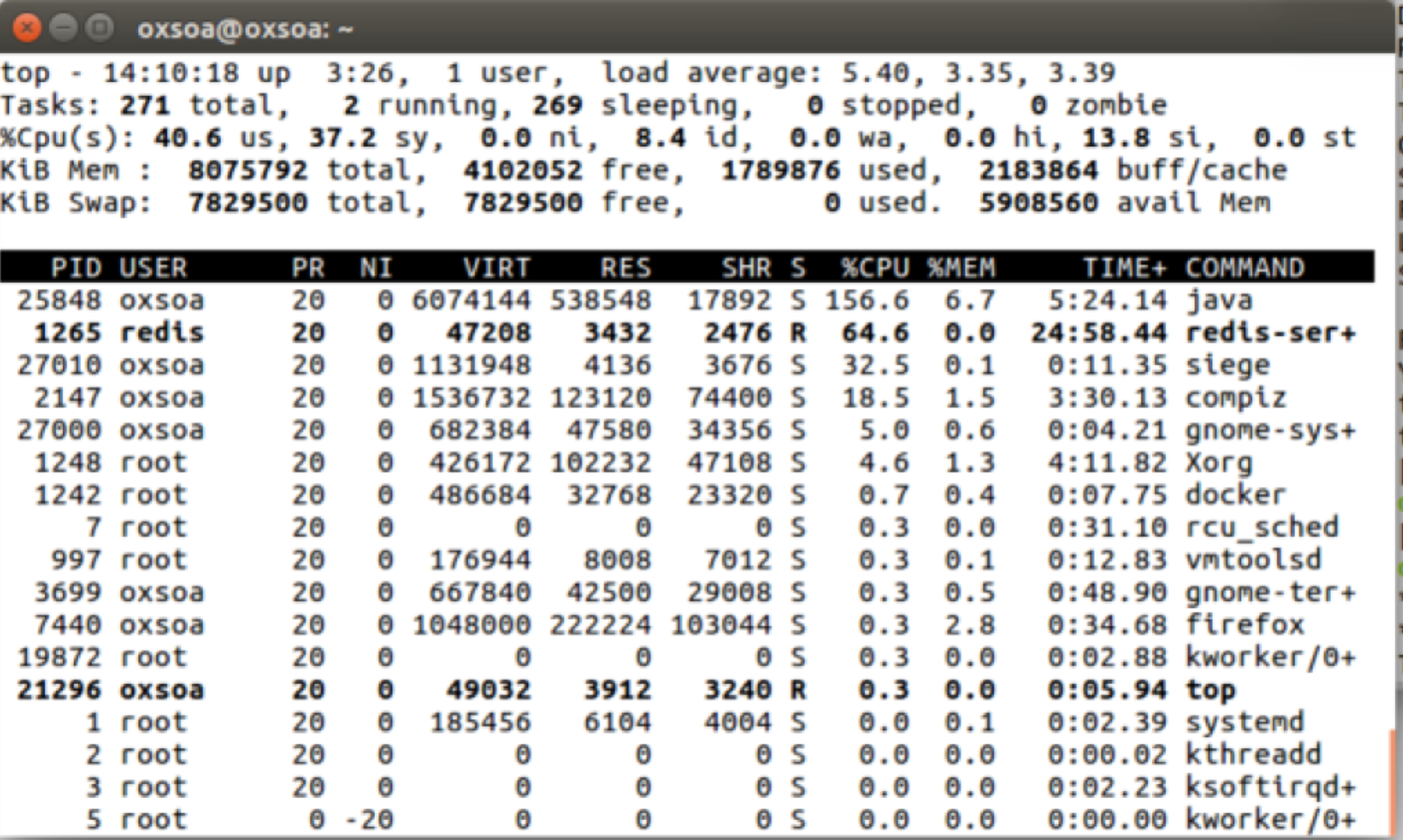
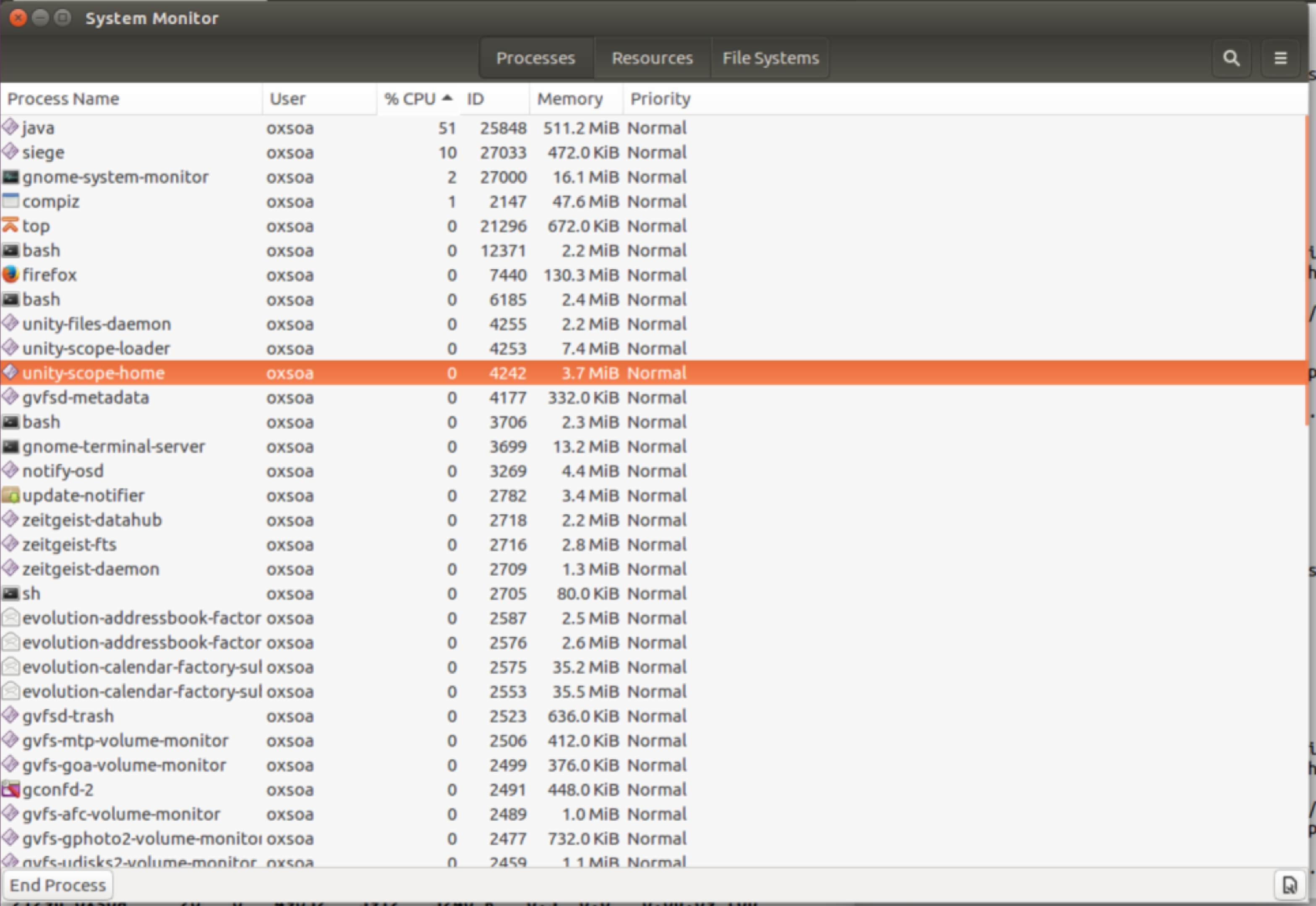
**Overview**

We will look at using a benchmarking tool to call our APIs very fast and see how they react.

Steps

1. Create a directory for this work and clone the git repository:  
   mkdir ~/ex7c  
   cd ~/ex7c  
   git clone <https://github.com/pzfreo/POResourceComplete.git>
2. Make sure redis is running locally:  
   sudo service redis-server start
3. Build the code

cd POResourceComplete  
gradle shadowJar

1. Run the shadow JAR as before
2. We can performance test our app. First lets install siege, a simple HTTP performance test app:  
     
   sudo apt install -y siege
3. Now we can run a test:  
   siege -b -c 15 -r 10000 http://localhost:8080/purchase
4. This will constantly hit our server with 15 concurrent clients each calling 10k times, in benchmark mode (i.e. each request hits immediately after the one before rather than being random).
5. While it is running you can monitor the CPU.  
   Open up a new terminal window and type:  
   top
6. You will see a memory/cpu/process monitor.   
   
7. Alternatively you can use the Ubuntu System Monitor:  
   
8. Back in your Siege terminal window you should see it complete:

\*\* SIEGE 3.0.8

\*\* Preparing 15 concurrent users for battle.

The server is now under siege.. done.

Transactions: 150000 hits

Availability: 100.00 %

Elapsed time: 63.76 secs

Data transferred: 32.33 MB

Response time: 0.01 secs

Transaction rate: 2352.57 trans/sec

Throughput: 0.51 MB/sec

Concurrency: 14.80

Successful transactions: 150000

Failed transactions: 0

Longest transaction: 0.12

Shortest transaction: 0.00

FILE: /var/log/siege.log

You can disable this annoying message by editing

the .siegerc file in your home directory; change

the directive 'show-logfile' to false.

[error] unable to create log file: /var/log/siege.log: Permission denied

1. Note that this is not a real performance analysis. Ideally the servers would be on a separate machine from the client load drivers (siege engines!). Also, microservices are designed to be run in parallel in multiple containers with load balancing across them, so this model is not the recommended way of running either deployment.
2. That’s all