Performance Tips for container applications

How to test and check performance issues

Schedule

- Infrastructure and Application
- What look?
- How to look?
- Optimization tradeoffs

- Isolated
- One container application doesn't affect other
- You don't need to worry about where your container will run
- You can reproduce the same behavior locally

- Your application will share resources with other containers?
 - Pros
 - Better use of your physical resources (CPU, Memory, etc)
 - Your application it is not using CPU ALL the time
 - Make sense have more than one container in the same host
 - Save money
 - Cons
 - The physical resources available changes over time
 - Application impacts each other
 - Spent more money

- All containers in the same host are yours?
 - If yes
 - The control about sharing resources belongs to the team responsible by the applications
 - The development team needs to know about infrastructure
 - The development team could tune the applications and hardware
 - Could have waste of resources (Ex: batch operations)
 - If not
 - The control about sharing resources belongs to the infrastructure team
 - The development team has fewer options to tune application
 - Possible no waste of resources

- The host will share containers with other applications?
 - Data base
 - Infrastructure resources
 - Load Balancers
 - HAProxy
 - Nginx / Apache
 - Cache Layer
 - Logger App (ex: FluentD)

- The environment will put some limits?
 - Memory
 - CPU usage
 - File Descriptor
 - Network Bandwidth
 - Disk I/O operations

Application - Concerns

- Your application will run inside a Virtual Machine / Server ?
 - Java with JVM (Application Server: Tomcat, Jetty, etc)
 - .NET Kestrel Web Server
- Your container runs more than one application?

What look?

- Memory
- CPU usage
- File Descriptor
 - Sockets
 - Files
- Network Bandwidth
- Disk I/O operations

How to look? - Docker run command

```
docker run -it --ulimit nofile=<file_descriptor_limit>
--memory <memory_limit> --cpus <cpu_limit> --name <app_name>
<app_image>
```

Hardware/Infrastructure Limits:

- <file_descriptor_limit>
- < <memory_limit>
- cpu_limit>

How to look? - Minimum requirements

- Run your application without limits
- Performance Test with status URL (do nothing)
- Performance Test with service URL (do the thing)

- Java Application
- Endpoints
 - GET /status : do nothing, return HTTP 200
 - POST /sort : Sort N lists of integer numbers internally using async for each list and with thread
 sleep with random milliseconds

```
curl -v -X POST http://localhost:8080/sort 'content-type: application/json' \
  -d '{
    "intList": [{
        "list": [8, 9, 4, 12, 56, 78, 90, 56, 34, 74]
        "list": [8, 9, 4, 12, 56, 78, 90, 56, 34, 74]
        "list": [8, 9, 4, 12, 56, 78, 90, 56, 34, 74]
    } , {
        "list": [8, 9, 4, 12, 56, 78, 90, 56, 34, 74]
        "list": [8, 9, 4, 12, 56, 78, 90, 56, 34, 74]
    } ]
```

Run Application

docker run -it -p 8080:8080 --name demo demo:1.0.0

Docker Status

docker stats demo

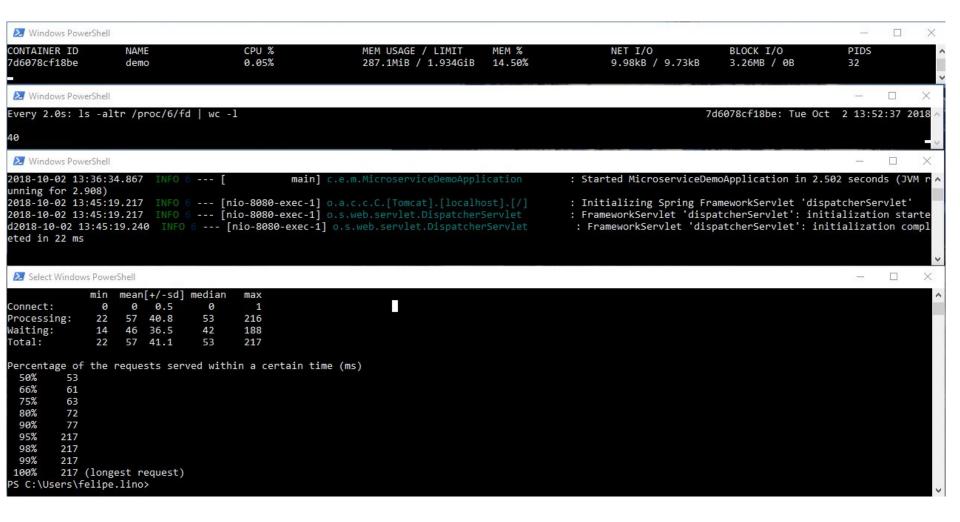
Open File Descriptors

```
docker exec -it demo /bin/bash
apt-get update
apt-get install procps -y
ps -efl
watch 'ls -altr /proc/6/fd | wc -l'
```

```
Performance Test (Ex: Apache Benchmark Tool)

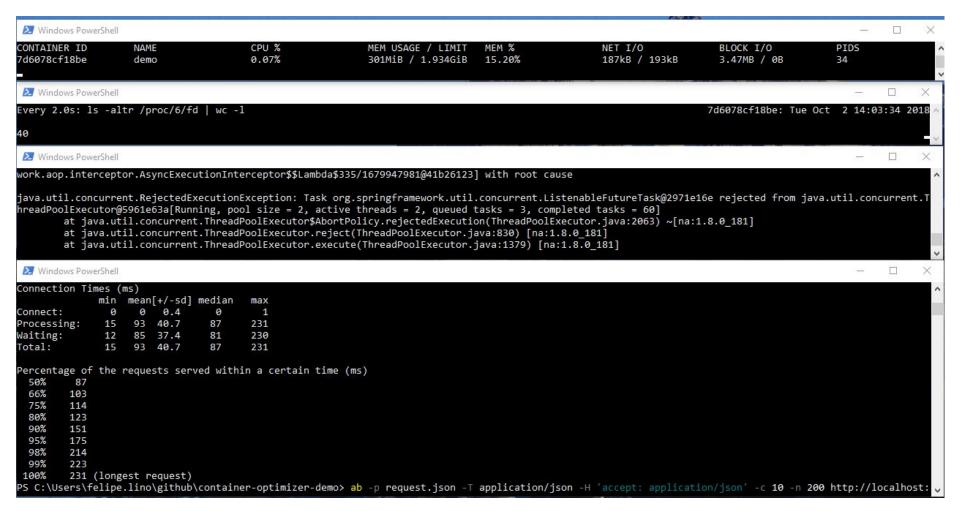
ab -s 60 -c 10 -n 20 <a href="http://localhost:8080/status">http://localhost:8080/status</a>
```

- -c : Concurrent Users
- -n : Number of iterations
- -s: timeout (seconds)



Performance Test (Ex: Apache Benchmark Tool)

```
ab -p request.json -T application/json -H 'accept: application/json' -c 10 -n 200 http://localhost:8080/sort
```



How to look? - Maximum requirements

- Run your application applying infrastructure limits
- Run your application applying server (JVM) parameters
- Performance Test with service URL (do the thing)

Run Application

```
docker run -it --ulimit nofile=128 --memory 1000MB --cpus 0.5 -e "JAVA_OPTS=-Xms312m -Xmx750m -DcorePoolSize=100 -DmaxPoolSize=120 -DqueueCapacity=300" -p 8080:8080 --name demo demo:1.0.0
```

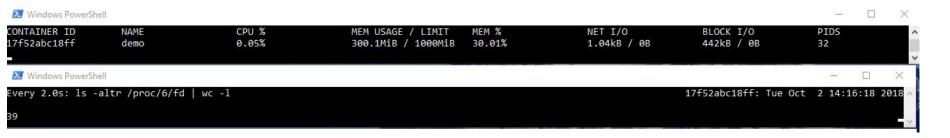
Infrastructure limits:

- --cpus
- --memory
- --ulimit

Applications config:

- Threads: corePoolSize (2), maxPoolSize (2), queueCapacity(3)
- Xms / Xmx
- etc.

How to look? Before start again



- Memory Limit is 1GB
- You start to usage around 300MB because of JAVA_OPTS with -Xms312m

```
Windows PowerShell
CONTAINER ID
                    NAME
                                         CPU %
                                                             MEM USAGE / LIMIT
                                                                                   MEM %
                                                                                                       NET I/O
                                                                                                                           BLOCK I/O
                                                                                                                                                PIDS
c4a0020e315e
                                         0.36%
                                                             288.6MiB / 1000MiB
                                                                                  28.86%
                                                                                                       162kB / 132kB
                                                                                                                            303kB / 0B
                    demo
                                                                                                                                                132
 Windows PowerShell
Every 2.0s: ls -altr /proc/6/fd | wc -l
                                                                                                                        c4a0020e315e: Tue Oct 2 14:
40
 Mindows PowerShell
2018-10-02 14:21:22.554
                                    [CustomThread-43] c.e.m.service.SortServiceImpl
                                                                                                 : Sleeping for 98 ms
2018-10-02 14:21:22.554
                                --- [CustomThread-86] c.e.m.service.SortServiceImpl
                                                                                                 : Sleeping for 82 ms
2018-10-02 14:21:22.554
                                --- [CustomThread-87] c.e.m.service.SortServiceImpl
                                                                                                 : Sleeping for 36 ms
2018-10-02 14:21:22.558
                                --- [CustomThread-94] c.e.m.service.SortServiceImpl
                                                                                                 : InputList: [8, 9, 4, 12, 56, 78, 90, 56, 34, 74]
2018-10-02 14:21:22.558
                                --- [CustomThread-94] c.e.m.service.SortServiceImpl
                                                                                                 : Sleeping for 31 ms
2018-10-02 14:21:22.568
                                --- [CustomThread-33] c.e.m.service.SortServiceImpl
                                                                                                 : OutputList: [4, 8, 9, 12, 34, 56, 56, 74, 78, 90]
                                                                                                 : OutputList: [4, 8, 9, 12, 34, 56, 56, 74, 78, 90]
2018-10-02 14:21:22.589
                                --- [CustomThread-40] c.e.m.service.SortServiceImpl
 Windows PowerShell
Connection Times (ms)
              min mean[+/-sd] median
                                         max
Connect:
                         0.4
                                           1
                     0
                                  0
               92 191 105.3
                                       1245
Processing:
                                184
Waiting:
               80 183 95.0
                                178
                                       1038
Total:
               92 192 105.3
                                       1245
                                184
Percentage of the requests served within a certain time (ms)
  50%
         184
  66%
         196
  75%
         205
  80%
         216
  90%
         285
  95%
         322
  98%
         504
  99%
         524
 100%
        1245 (longest request)
```

PS C:\Users\felipe.lino\github\container-optimizer-demo> _

Optimizations tradeoffs

- Change memory limits
- Change CPU limits
- File Descriptors and Sockets
- Number of containers x Threads (workers) inside application
- Specific configurations for your Server/JVM
 - server.undertow.io-threads
 - KestrelServerOptions.ThreadCount

Optimizations tradeoffs

- IO Usage
- Network Usage
- Related dependencies
 - APIs
 - Database
 - File System

Any doubts?



Source:

https://github.com/felipelino/container-optimizer-demo

Contacts:

Felipe Lino (felipelino44@gmail.com)