



Linux DevOps Lab

JAVA PLATFORM

02_Stack_Heap

Home task

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

CONFIDENTIAL | Effective Date: 25-Jan-19

PREREQUISITES

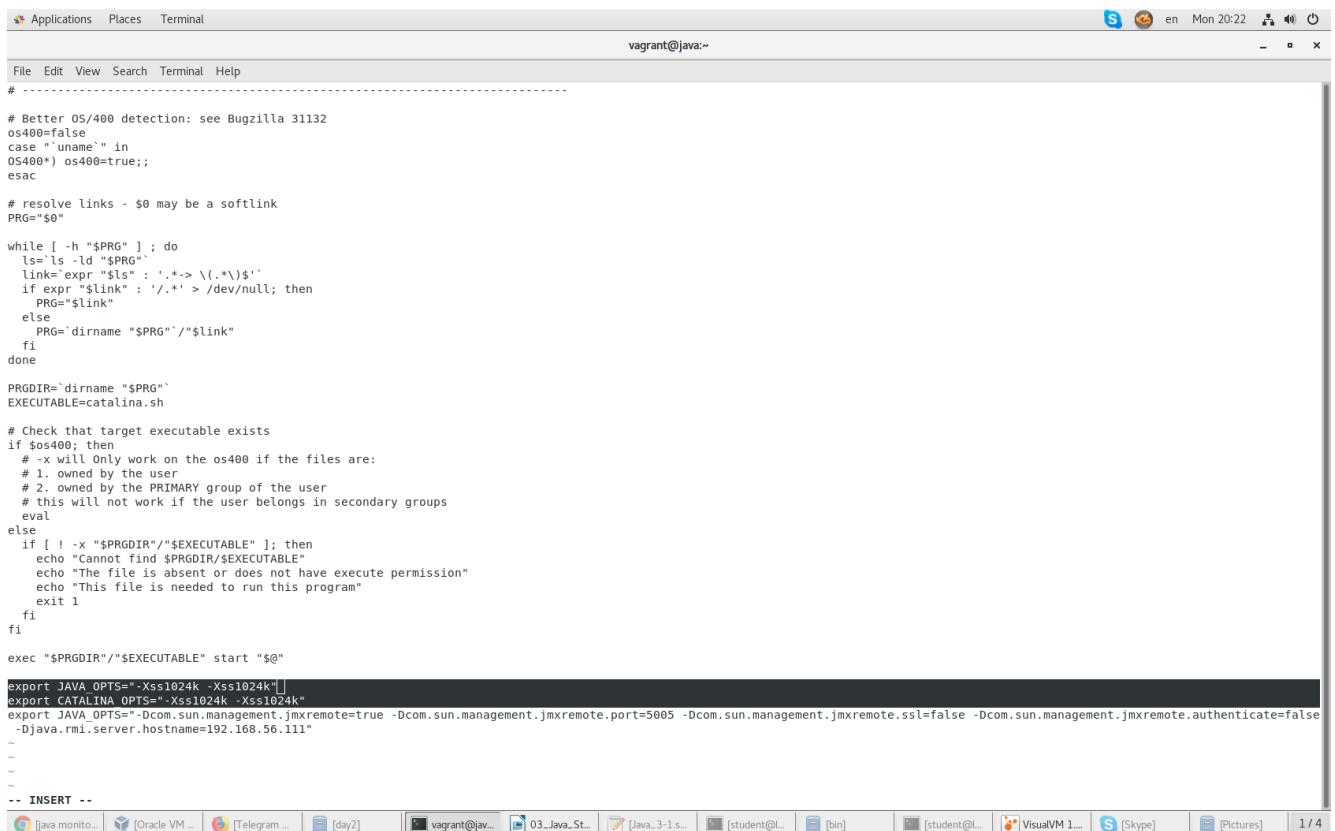
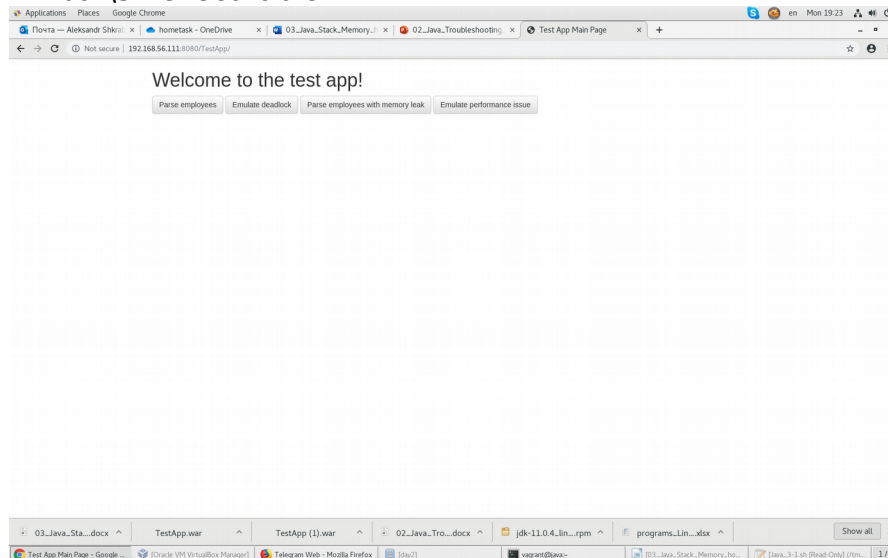
Oracle JDK 1.8 installed

GOAL

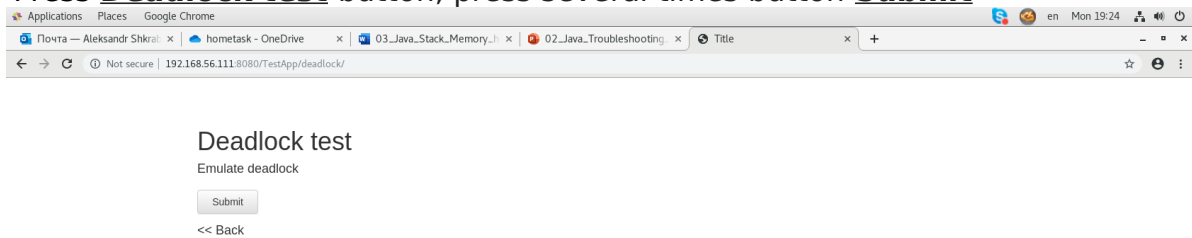
Understand principle of slow threads and threads deadlock detection

TASK

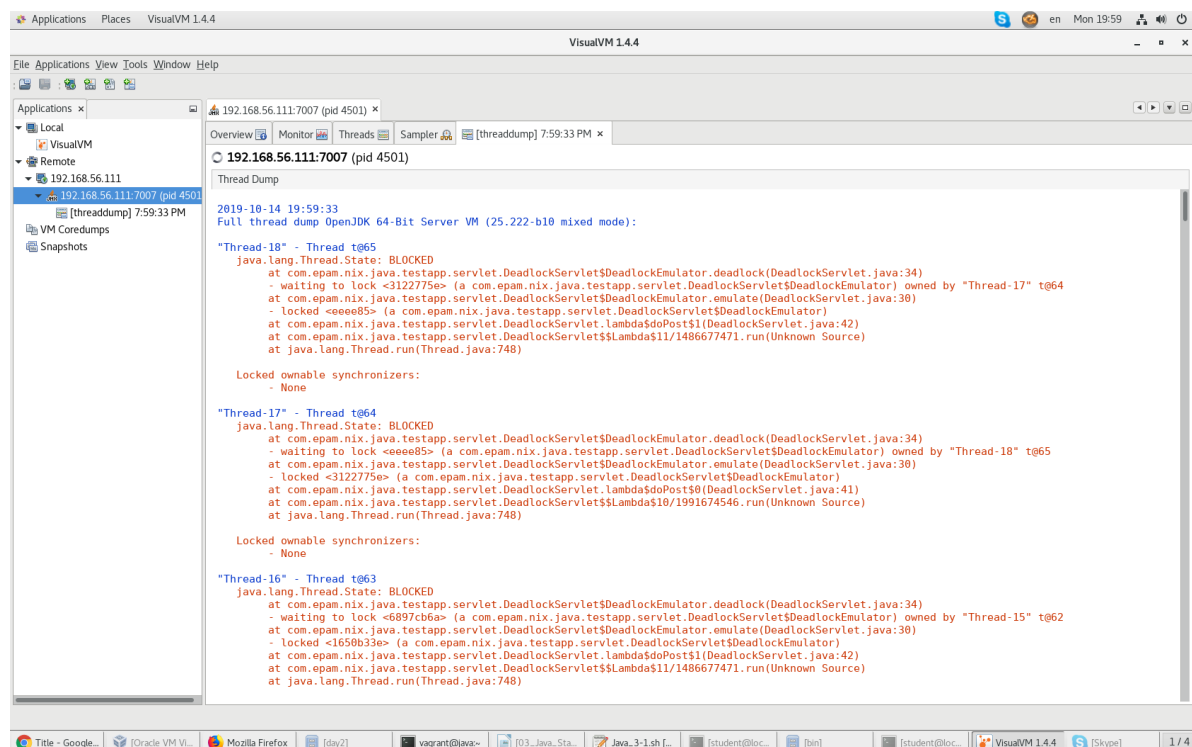
- 1) Deploy **TestApp.war** and start Tomcat with max stack size 1024k using single bat/sh executable



- 2) Press **Deadlock test** button, press several times button **Submit**



- 3) Run **VisualVM**, select tomcat pid, open tab threads. What does VisualVM tell us about threads state?
Сообщает о том, что был обнаружен deadlock. При просмотре treaddumpa видим какие треды вызвали deadlock и почему, а именно com.epam.nix.java.testapp.servlet.DeadlockServlet\$DeadlockEmulator.d eadlock каждый последующий тред ожидает пока освободится объект, который занят предидущим тредом, и при этом сам блокирует объект который будет необходим следующему треду.



- 4) Obtain thread dump with **jstack**. Analyze thread dump using **IBM Thread and Monitor Dump Analyzer** and answer a question:
 - o What threads produce dead lock

com.epam.nix.java.testapp.servlet.DeadlockServlet\$DeadlockEmulator.deadlock содержит идентичные треды каждый из которых ожидает объект, который занят предыдущим. Поэтому они накапливаются и не завершаются.

The screenshot shows the fastThread web application interface. The left sidebar has a menu with 'Comparative summary', 'THREAD DUMP 1', and 'THREAD DUMP 2' (which is selected). The main content area displays 'Most used methods' and 'CPU consuming threads'.

Most used methods
(Frequently executed methods are reported. If lot of threads executes same method, it may be a concern. Learn [all roads lead to Rome pattern](#))

Thread Count	Method	Percentage
106 threads	com.epam.nix.java.testapp.servlet.DeadlockServlet\$DeadlockEmulator.deadlock(DeadlockServlet.java:34) <small>To see stack trace click here</small>	71% <div></div>
22 threads	sun.misc.Unsafe.park(Native Method) <small>To see stack trace click here</small>	15% <div></div>
6 threads	sun.nio.ch.EPollArrayWrapper.epollWait(Native Method) <small>To see stack trace click here</small>	4% <div></div>
4 threads	java.lang.Object.wait(Native Method) <small>To see stack trace click here</small>	3% <div></div>
4 threads	java.net.PlainSocketImpl.socketAccept(Native Method) <small>To see stack trace click here</small>	3% <div></div>

[Show all methods >>](#)

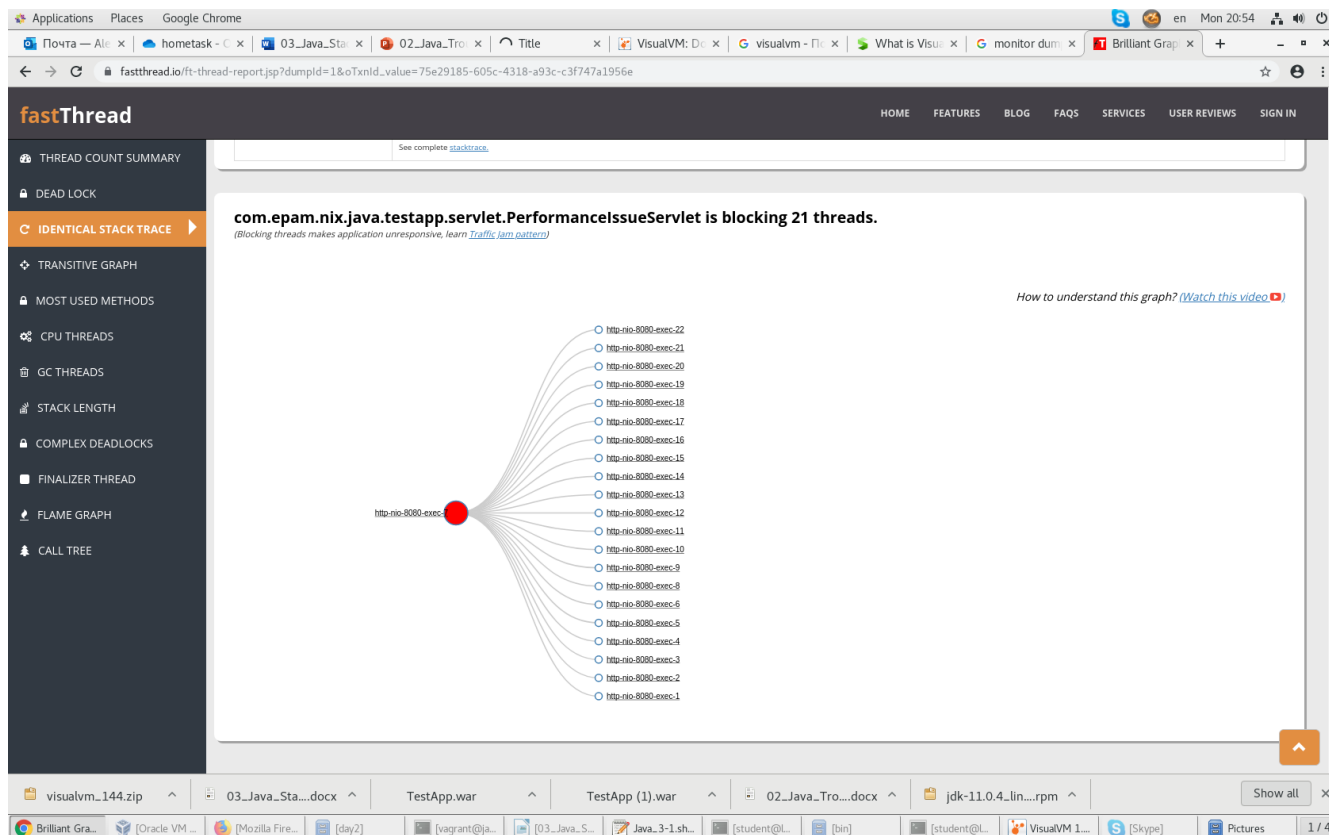
CPU consuming threads
(If application is consuming high CPU, investigate below threads. Learn [Athlete pattern](#))

Thread	CPU consuming thread's stacktrace
RMI TCP Connection(5)-192.168.56.1	<pre> java.lang.Thread.State: RUNNABLE at sun.management.ThreadImpl.dumpThreads0(Native Method) at sun.management.ThreadImpl.dumpAllThreads(ThreadImpl.java:454) at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62) ... </pre> <small>See complete stacktrace</small>

The bottom of the image shows a Windows taskbar with various open applications including VisualVM, Java IDEs, and web browsers.

- 5) Press **Emulate performance issue** button, press several times button **Submit**

- 6) Obtain thread dump with **jstack**. Analyze thread dump using and answer a question:
 - o What thread has performance issue?



У треда http-nio-8080-еке-7 статус стоит RUNNING и его код написан так, что он не освобождает необходимый для других тредов объект в итоге все они находятся в статусе waiting. Если мы откроем информацию по остальным тредам, то там увидим, что они ожидают пока освободится определенный объект и по id этого объекта можем обнаружить, что он занят тредом http-nio-8080-еке-7. И все это в com.epam.nix.java.testapp.servlet.PerformanceIssueServlet

