Linux DevOps Lab

JAVA PLATFORM

02_Stack_Heap

Home task

PREREQUISITES

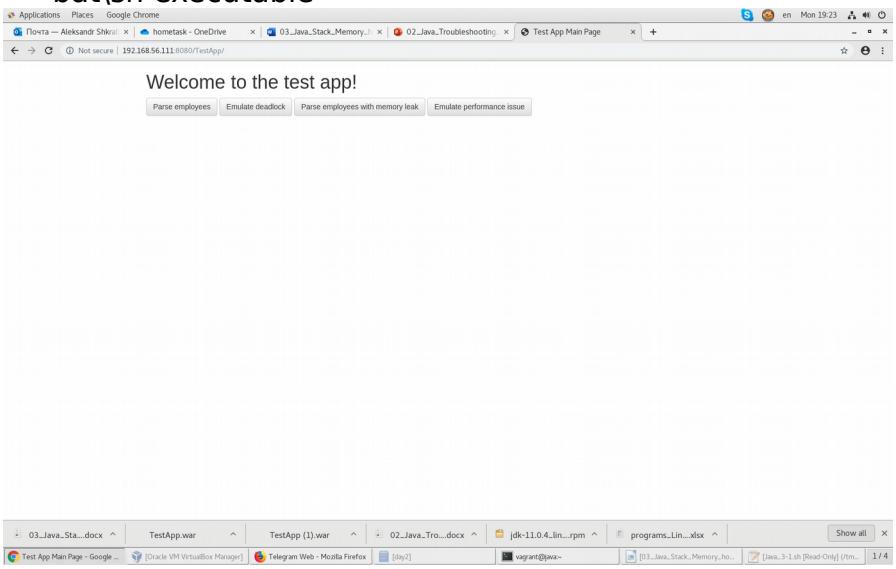
Oracle JDK 1.8 installed

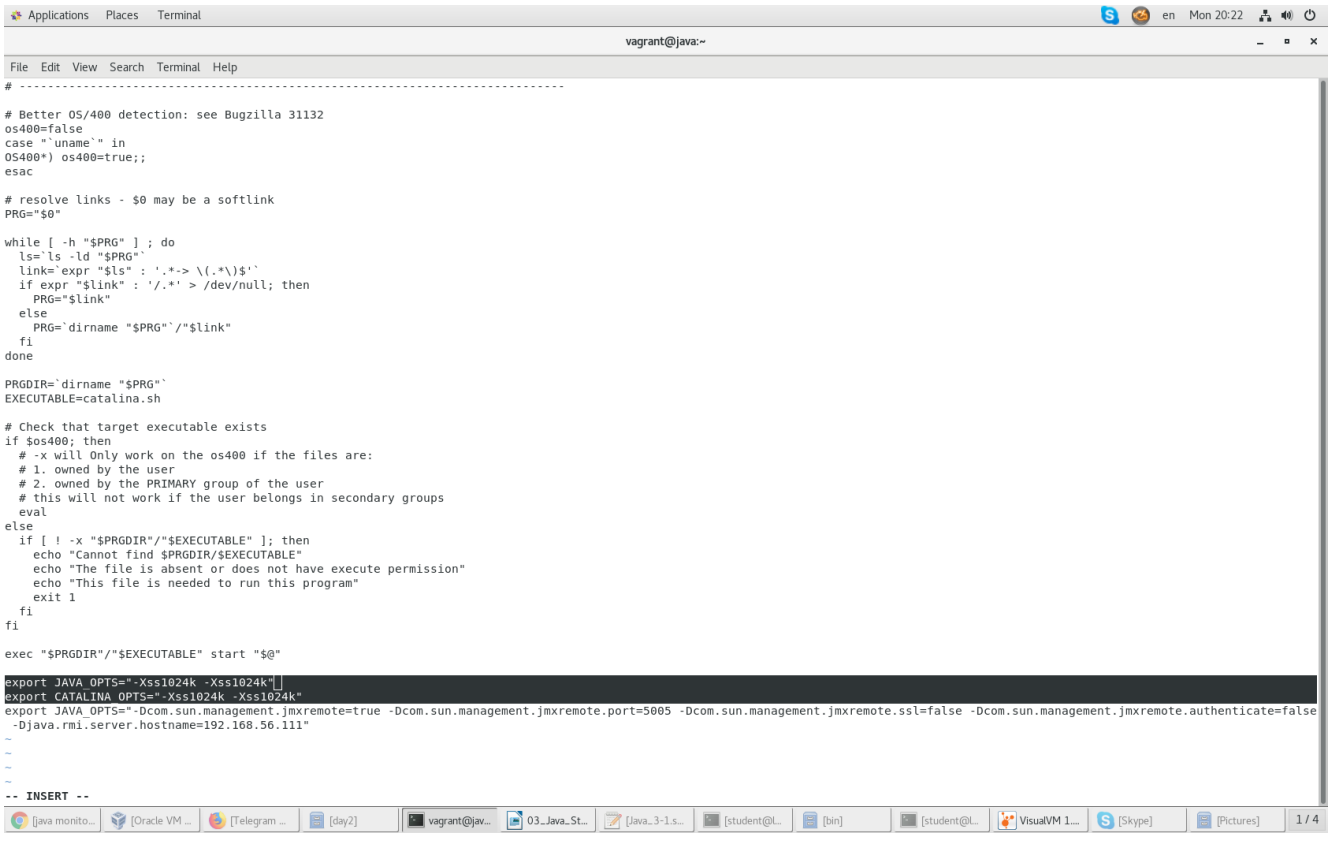
GOAL

Understand principle of slow threads and threads deadlock detection

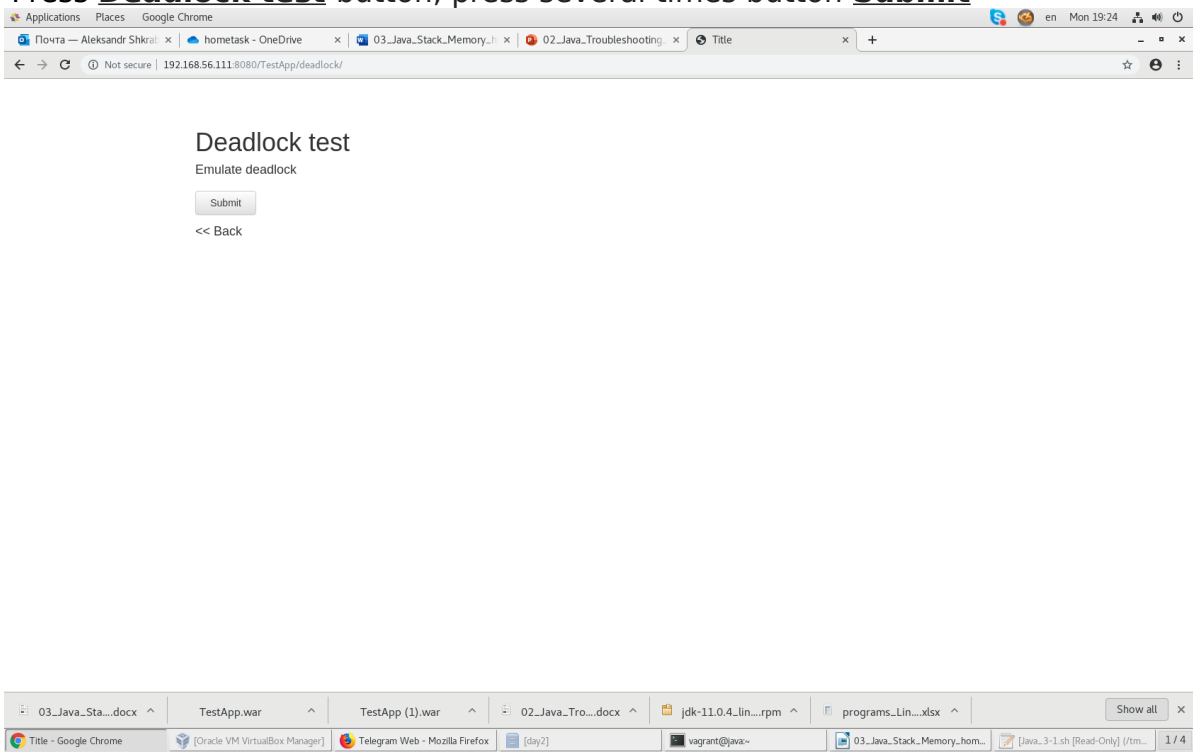
TASK

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



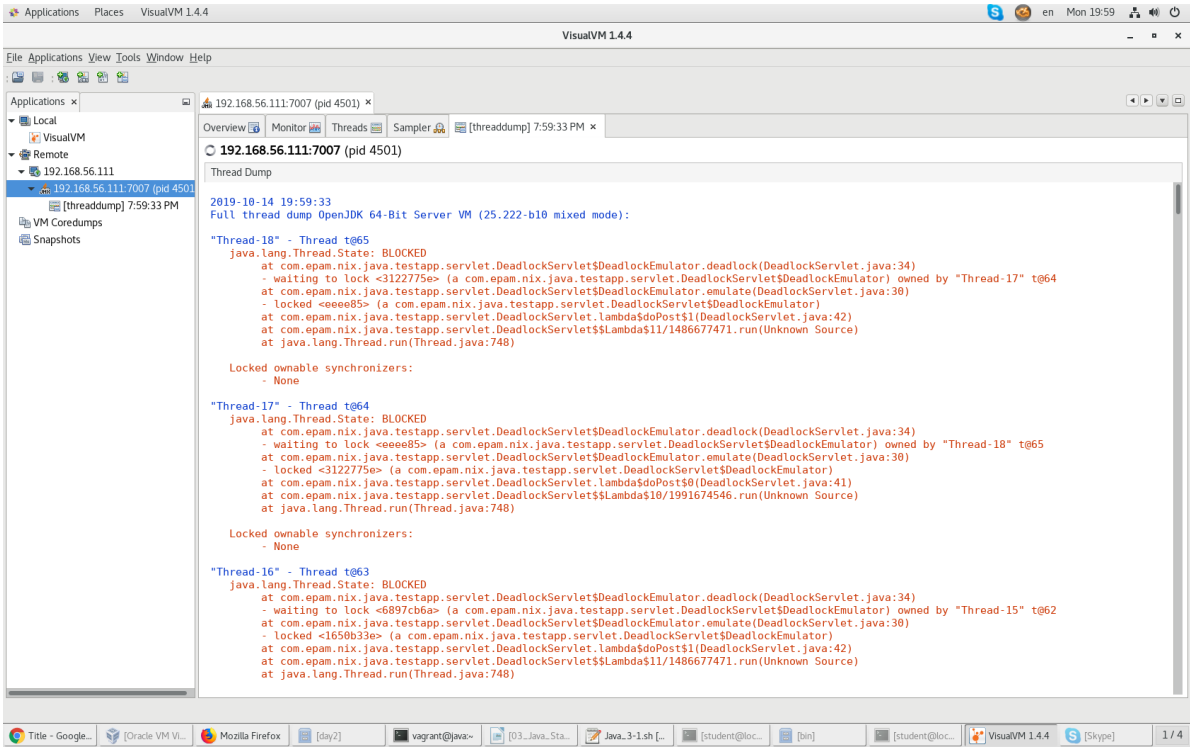


2) Press **Deadlock test** button, press several times button **Submit**



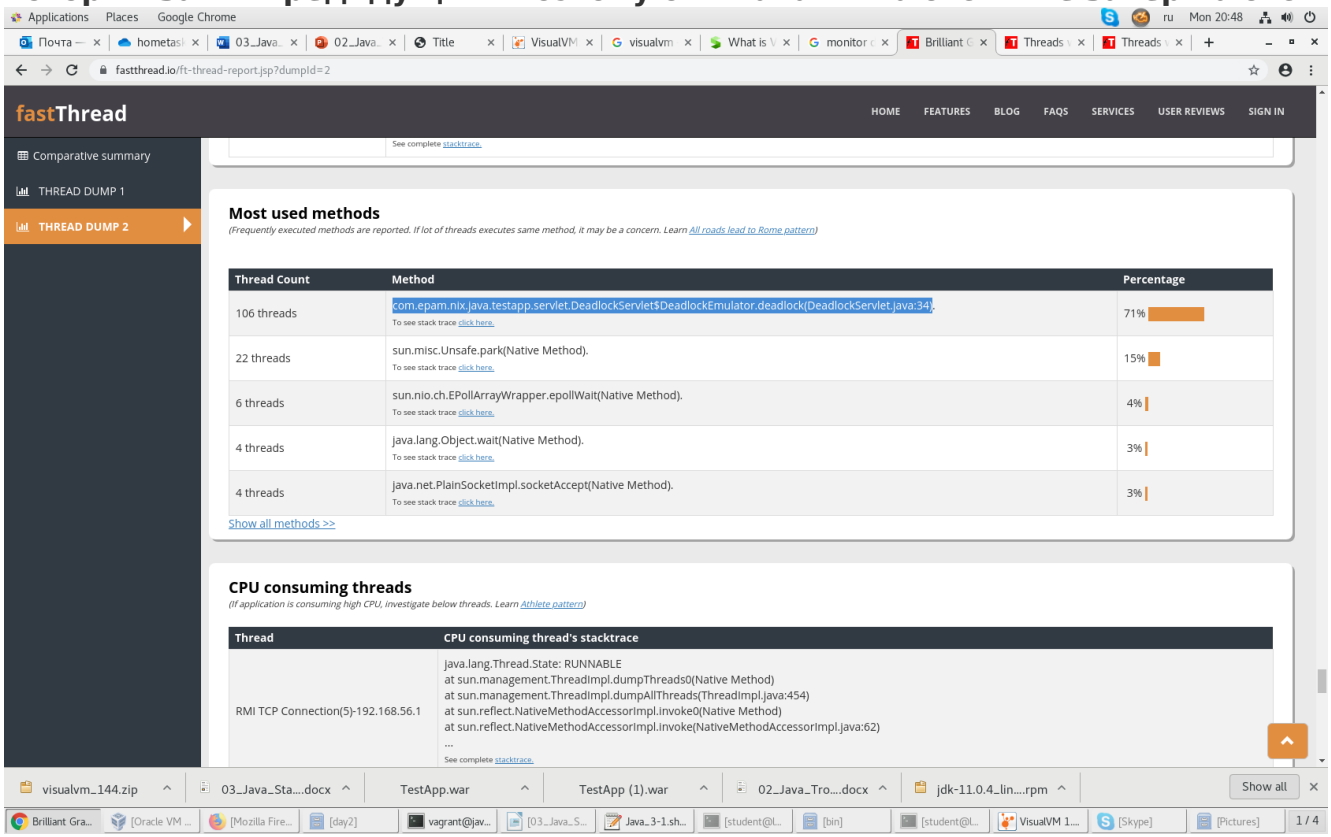
3) Run **VisualVM**, select tomcat pid, open tab threads. What does VisualVM tell us about threads state?

Сообщает о том, что был обнаружен deadlock. При просмотре treaddumpa видим какие треды вызвали deadlock и почему, а именно в com.epam.nix.java.testapp.servlet.DeadlockServlet$DeadlockEmulator.deadlock каждый последующий тред ожидает пока освободится объект, который занят предидущим тредом, и при этом сам блокирует объект который будет необходим следующему треду.



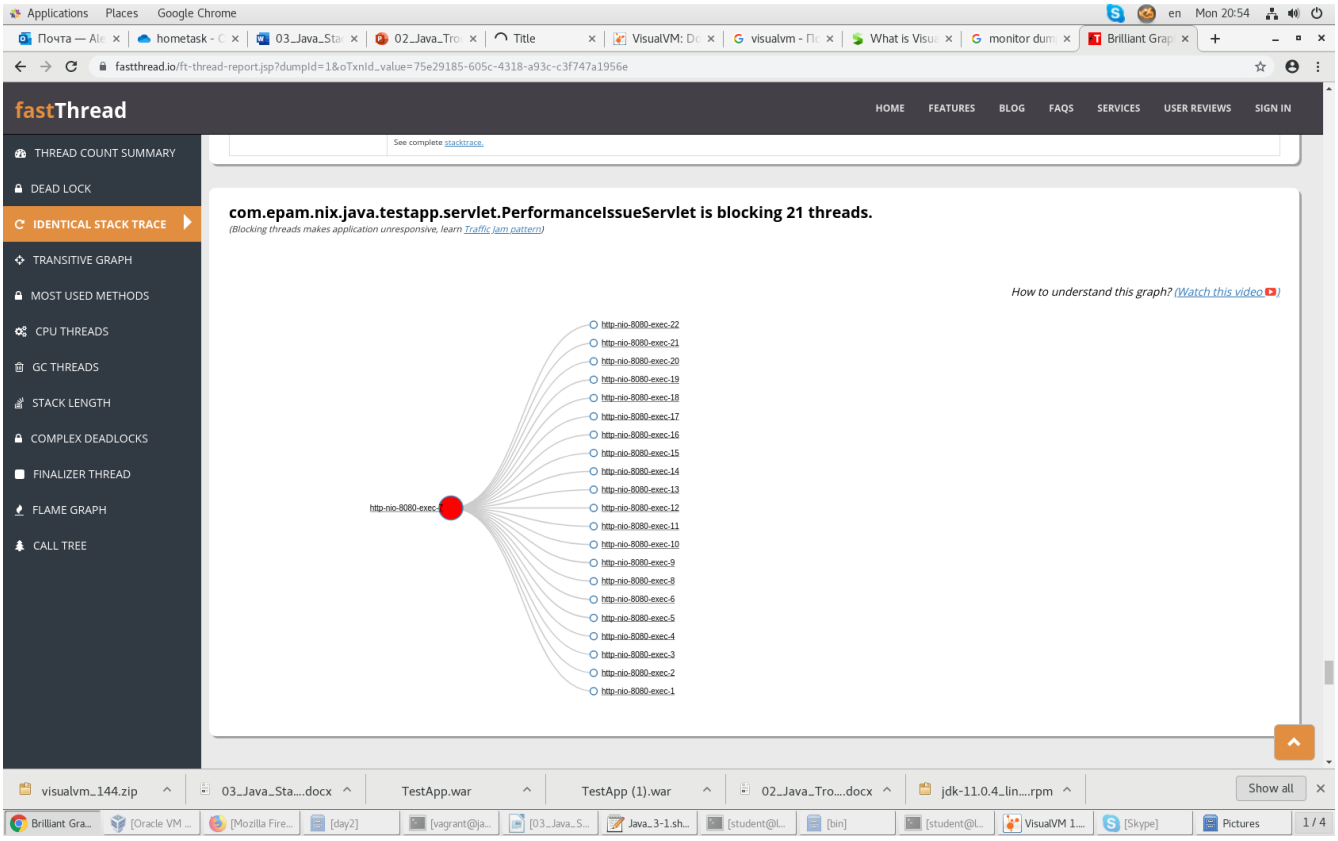
4) Obtain thread dump with **jstack**. Analyze thread dump using **IBM Thread and Monitor Dump Analyzer** and answer a question:
   - What threads produce dead lock

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



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:
   - What thread has performance issue?



У треда http-nio-8080-exec-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-exec-7. И все это в com.epam.nix.java.testapp.servlet.PerformanceIssueServlet