Background

In the class, we have covered architectural analysis, including architecture recovery techniques, and architectural implementation. In this homework assignment you will use and apply concepts from these topics.

In the second homework, you analyzed the architectures of two major versions of Tomcat and identified architectural vulnerabilities in one of them, and then inspected whether they were present in the second version. In this homework, you will further analyze the architectures of those two versions and their implementations.

Tasks

In the previous homework, you have identified components and connectors of Tomcat’s architecture for two of Tomcat’s major versions. You also identified 5 vulnerabilities for the first version and the components and connectors involved in those vulnerabilities. You then checked if those 5 vulnerabilities still exist in another major version of Tomcat, referred to as the second version. In this homework, you will identify the involved components and connectors for all the vulnerabilities that according to your HW2 would still exist in the second version. If no vulnerabilities from the first version would exist in your second version, then identify the components and connectors in the second version that “prevented” the vulnerability from existing. You will map the identified components and connectors from HW2 to Tomcat’s implementation for both versions.

1. For the first version:
   a. Provide a table with the CVE ID, the corresponding CWE category, and all the components and connectors involved in that CVE in different rows of one column, and all files in Tomcat’s source code that are mapped to that component and connector in the subsequent column of the corresponding row. For instance, if you have a CVE with ID XXX-XXXX, whose category is Y, and one of the relevant components is Engine, with the involved files being StandardEngine.java and StandardEngineValve.java, one of the rows in your table would look like Table 1.
Table 1. Example on mapping of vulnerable components and connectors to source files

<table>
<thead>
<tr>
<th>CVE ID</th>
<th>CWE</th>
<th>Component &amp; Connector</th>
<th>Corresponding files</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXX-XXXX</td>
<td>Y</td>
<td>Engine</td>
<td>StandardEngine.java</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>StandardEngineValve.java</td>
</tr>
</tbody>
</table>

b. What percentage of your vulnerable components and connectors are implemented across multiple classes vs. within a single class? If components do connectors are implemented within a class, do vulnerabilities tend to involve more than one procedure? Please limit your answer to two paragraphs.

c. If in the 5 vulnerabilities you have picked, you have both vulnerabilities that happen due to a wrong architectural decision and vulnerabilities that happen due to incorrect implementation of a correct decision:
   i. Do you see a difference in how components involved in vulnerabilities that happen due to the wrong architectural decisions vs. correct decisions implemented incorrectly are mapped to the implementation? Are “wrong architectural decision” vulnerabilities more concentrated or spread out across classes than the “wrong implementation of the right architectural decision” vulnerabilities? Please limit your answer to one paragraph.

If all the 5 vulnerabilities you have picked are either vulnerabilities that happen due to a wrong architectural decision or vulnerabilities that happen due to incorrect implementation of a correct decision:
   i. Are there any commonalities in how the different vulnerabilities are manifested in the implementation? Please limit your answer to one paragraph.

d. What differences do you notice between (i) the architecture depicted in Tomcat’s documentation for the identified vulnerable components and connectors and (ii) what is implemented? Please limit your answer to two paragraphs.

2. Compare the two versions:
   a. To what extent can the differences between the two Tomcat versions' architectures be attributed to the removal of the first version's vulnerabilities in the later version? Did this affect the mappings of architectural elements to implementation elements? If so, how and why? Please limit your answer to one paragraph.
3. For this question, please refer to the Appendix on ACDC and ARC. Run both ACDC and ARC to cluster Tomcat’s architecture for the two versions. *For each technique*
   a. Provide a table with all your clusters and their respective elements. For instance if you have a cluster `org.apache.catalina.startup.ss` that contains the class `org.apache.catalina.startup.Tool` and `org.apache.catalina.startup.ClassLoaderFactory$2`, your table would look like Table 2.

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>org.apache.catalina.startup.ss</td>
<td>org.apache.catalina.startup.Tool</td>
</tr>
<tr>
<td></td>
<td>org.apache.catalina.startup.ClassLoaderFactory$2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

*Table 2: Example on how to represent clusters and their elements*

b. What clusters include the vulnerable components and connectors from task 1? Highlight those clusters in the first table you created. Are the implementation classes that make up a vulnerable component or connector within a cluster or spread out across clusters? If the classes that make up your vulnerable component are spread out across clusters, why do you think that happened? Please limit your answer to one paragraph.

c. Do the clusters change from one version to another? If yes, can you speculate why based on your understanding of how the respective architecture recovery techniques work? You should refer to the documentation on ACDC and ARC, provided below. Please limit your answer to one paragraph.
Appendix

ACDC and ARC are well-known architecture recovery techniques. An implementation of them has been included in a larger tool, Architecture Recovery, Change, And Decay Evaluator (ARCADE). The tool has a detailed manual, which you will consult on how to run ACDC and ARC:

1. [https://softarch.usc.edu/~lemduc/Recovered_files/ArchitectureEvolutionAnalysiswithARCade.pdf](https://softarch.usc.edu/~lemduc/Recovered_files/ArchitectureEvolutionAnalysiswithARCade.pdf)

You are welcome to read the entire manual, but the relevant parts for you are in pages 11-20.

The link to the repository in the manual will not be accessible, but the following repository contains a clone of that code:

1. [https://github.com/asejfia/CS578-arcade](https://github.com/asejfia/CS578-arcade)

You will have all the jar files and the code in this repository.

In your case, you do not need to worry or report on the architecture smells, which are also discussed in the manual; your focus is solely on clustering. Your versions are the two versions you have selected in the second homework. Bear in mind that you will need to compile those versions. The following list contains the pages in tomcat’s documentation that help with how to build different versions of Tomcat:

2. [https://tomcat.apache.org/tomcat-8.0-doc/building.html](https://tomcat.apache.org/tomcat-8.0-doc/building.html) (these instructions are valid for both versions 8.0 and 8.5)
5. [https://tomcat.apache.org/tomcat-5.5-doc/building.html](https://tomcat.apache.org/tomcat-5.5-doc/building.html)