POLITECHNIKA POZNAŃSKA

Wydział Informatyki i Telekomunikacji

Application Security (MIKE) -> (LIMA)

Patryk Miedziaszczyk 135653
patryk.miedziaszczyk@student.put.poznan.pl

25 listopad 2022

Introduction

The results presented in this document were made on an application project that contained a server application and a client application. Therefore, I performed the analysis using all the tools for both the server and client code. For all of the eight tools, the results obtained from the vulnerability analyzes carried out were appropriately added to individual catalogs, broken down into the results obtained for the server code and the results obtained for the client code.

Tools

1. **automated-security-helper**

As far as this tool is concerned, you can find information that it does not support Java, due to the fact that the application I tested was made in Java, this tool did not return any interesting results.

2. **betterscan-ce**

In this case, the tool for the client application found 26 vulnerabilities and 28 vulnerabilities for the server application. One of the more interesting information returned may be as follows

“Cryptographic keys should not be kept in the source code. The source code can be widely shared in an enterprise environment, and is certainly shared in open source. To be managed safely, passwords and secret keys should be stored in separate configuration files or keystores.SeverityWarningLine49Fileapp/src/main/java/com/example/insecurecommunicationserverclient/EncryptData.java”

3. **report SpotBugs**

Here, the tool, like the above, returned, among others, the following information and identified this problem as a high risk

H I Dm: Found reliance on default encoding in com.example.insecurecommunicationserverclient.EncryptData.getSha256HashInBytes(String): String.getBytes() At EncryptData.java

4. Fluid Attack’s Scanner

In the case of this tool, no data was returned, but the tool itself, as can be seen in the attached screenshots, gives the impression that it has carried out the tests correctly. Due to the fact that the tool did not return any error during launch, I decided that this particular tool did not find any vulnerabilities.

5. Horusec

This tool seems to me to be the most interesting, or at least it provides us with the most important information and verifies the code very carefully, below I present one of the vulnerabilities returned, which turned out to be a critical vulnerability in this application.

Language: Java
Severity: CRITICAL
Line: 15
Column: 23
SecurityTool: HorusecEngine
Confidence: MEDIUM
File: /home/patryk/Desktop/AppSecurity/InsecureCommunicationServerClient/app/src/main/res/layout/activity_main.xml
Code: android:text="10.0.2.2" />
RuleID: HS-JAVA-152
Type: Vulnerability
ReferenceHash: 9589106879c63897095de32644d6a47029fdfd864be97df81ecb9cdf6bf77d95

Details: (1/2) * Possible vulnerability detected: Spring Framework Remote Code Execution
It has been identified that versions prior to < 5.3.18 or < 5.2.20 of the spring framework are vulnerable to remote code execution. Please upgrade to version >= 5.3.18 or >= 5.2.20. For more information checkout the CVE-2022-22965 (https://tanzu.vmware.com/security/cve-2022-22965) advisory.

(2/2) * Possible vulnerability detected: Remote code injection Apache Log4j
Log4j versions prior to 2.17.1 are subject to a remote code execution vulnerability via the ldap JNDI parser, uncontrolled recursion from self-referential lookups and some other vulnerabilities. For more information checkout the CVE-2021-44228 (https://nvd.nist.gov/vuln/detail/CVE-2021-44228), CVE-2021-45046 (https://nvd.nist.gov/vuln/detail/CVE-2021-45046), CVE-2021-45105 (https://nvd.nist.gov/vuln/detail/CVE-2021-45105) and CVE-2021-44832 (https://nvd.nist.gov/vuln/detail/CVE-2021-44832) advisories.

6. Mobile Security Framework

Due to the fact that this tool was launched as the sixth one, I noticed that most of the information returned coincided with the reports from previous applications. We can, inter alia, again find information about line 49 in which a vulnerability defined by the tool as “ A hardcoded Key is identified.”.

7. gitleaks

As for this tool, I had trouble running it on Linux / Ubuntu 20.4.

8. SonarQube SCA

Compared to the previous tools, this tool returned much less vulnerabilities, but nonetheless, the ones it did return are very interesting.

Summary

Although the application is not very extensive and theoretically it should not contain major or minor vulnerabilities, as shown by the results obtained during application testing, using the above tools, the application does not contain any vulnerabilities, and it can even be said that for this type of the application managed to find a lot of vulnerabilities. What drew my attention the most during the analysis of the results was the fact that the application has critical vulnerabilities.