Vulnerability Assessment

By Vincent Faragalli

# Goals:

* Run a complete SCA scan with SYNYK and list all CVE IDs
  + Identify vulnerabilities that are weaponized, what that CVE ID is
* Perform a risk assessment for the SNYK vulnerabilities
  + Find the top vulnerabilities
  + Use own risk matrix
  + Identify the mitigations/remediations for the threats

# Services/ applications used:

* GitHub
* SNYK

# Pre-Requisites

* GitHub account
* Knowledge of git

Contents

[Goals: 1](#_Toc172621812)

[Services/ applications used: 1](#_Toc172621813)

[Pre-Requisites 1](#_Toc172621814)

[Set up 2](#_Toc172621815)

[SCA Scan With SNYK 2](#_Toc172621816)

[Risk assessment 5](#_Toc172621817)

[VID – 001 6](#_Toc172621818)

[VID – 002 7](#_Toc172621819)

[VID – 004 8](#_Toc172621820)

[Summary 9](#_Toc172621821)

# Set up

A computer screen shot of white text

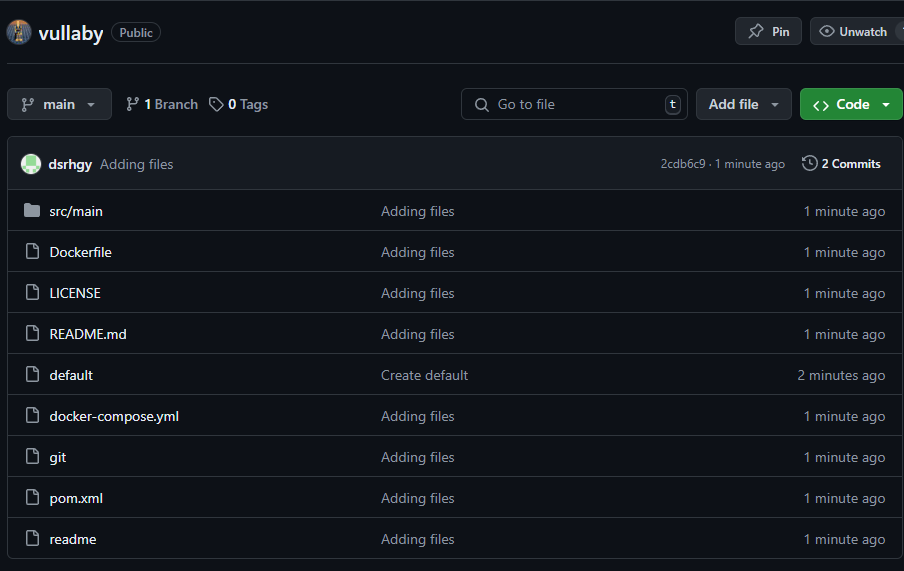
Description automatically generated

In the command prompt clone the lab GitHub

git clone <https://github.com/vulnassessment101/JavaVulnerableLab>

Create a new git repository in the newly created JavaVulnerableLab folder

Push it to your own GitHub



You should now have the lab in a personal GitHub that you will be able to change and update.

Now that that the OWASP lab is now in GitHub the setup is complete

# SCA Scan With SNYK

What is Snyk?

Snyk is a security platform to help find, fix, and prevent vulnerabilities in open-source dependencies and containerized applications. Snyk scans a project's dependencies, including libraries and frameworks, to identify known vulnerabilities. It provides actionable insights into the severity of these vulnerabilities and suggests fixes. Snyk helps organizations enhance their application security posture by addressing vulnerabilities in both their code dependencies and containerized environments, thereby reducing the risk of security breaches and ensuring compliance with security best practices.

Log in or create an account at [SNYK](https://app.snyk.io/).

Authorize snyk to your github

A screenshot of a computer

Description automatically generated

Select the github that has the lab files in it

A screenshot of a computer

Description automatically generated

After selecting Scan then the project files will be uploaded and scanned ready for the next part

A screenshot of a computer

Description automatically generated

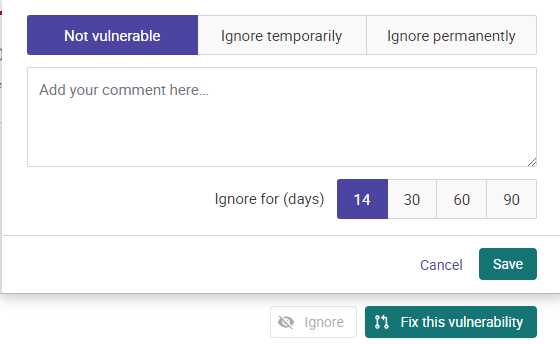
Select pom.xml to see all of the vulnerabilities

Each vulnerability shows its CWE (common weakness enumeration), its CVE details (Common Vulnerabilities and Exposures) and CVSS which is a scoring system for vulnerabilities. The scores are placed in categories based on the threat, these categories are low, medium, high and critical. This gives insight into which threats are a higher priority. Snyk provides links to each of the sites with information about the vulnerability and also provides its own summary of the vulnerability with details on why it was given such a score, what it exploits and why this vulnerability came to be.

A close up of a screen

Description automatically generated

Synk provides an ignore option if the vulnerability is a false positive and is of no worry.



The first two vulnerabilities in the list are level critical meaning that they need to be addressed or a major vulnerability will be left in the files.

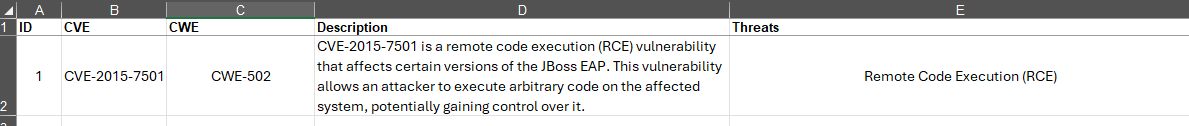
There are a total of 19 vulnerabilities found and each has their own threat and mitigation strategy. Each of these vulnerabilities will be going into the risk assessment form.

For this project, this documentation will have an in-depth explanation on how the first, second and fourth vulnerability were assessed, why they got the scores they do and the translation of that information onto the spreadsheet. The rest of the vulnerabilities will be assessed without documentation as it would be much to long to document each assessment.

# Risk assessment

Since the SNYK scan already orders the threats by their own metrics the start of the assessment will be the CVE-2015-7501: Deserialization of Untrusted Data.

The CVE and CWE ID are provided on the SNYK scan which can be copied to the ID slot on the spreadsheet.

The Description is created from looking at the vulnerability on the CVE and CWE websites and summarizing what it is doing. The first part of the vulnerability assessment is putting the data into the sheet to give a summery of what the vulnerability actually is

The next part of the vulnerability assessment is assessing the vulnerabilities likelihood, impact, and risk.

A screenshot of a computer

Description automatically generatedThe tab named Risk Matrix in the spreadsheet has my own custom-made Risk Matrix that is built for SNYK. I made this after researching different matrixes and specified it for the snyk scan

## VID – 001

First to calculate likelihood a score needs to be determined from the three sections. For Exploitability it would be Likely due to the vulnerability involving deserialization of untrusted data, exploitability can be considered likely since it requires sending malicious serialized objects. This attack does require user credentials and cannot be done without them solidifying its score of 4. Next is Discoverability which I have identified as Very likely (5) since this vulnerability can be discovered by scanning publicly facing instances of JBoss EAP. Given that JBoss EAP instances can be publicly identified, and that many attackers automatically scan everything in a company’s network it makes it very likely an attacker will exploit this vulnerability. Finally, the Reproducibility would also be Very likely (5) since the environment needed to exploit the vulnerability can be recreated with minimal effort.

With the average being 4.67 it makes the Likelihood of this vulnerability a 5 (Very Likely)



Now for the Impact of this vulnerability. Once again it is split into three parts which have their own set definitions in the assessment. First for type of impact it would be critical (5) since this vulnerability allows for remote code execution, it can result in complete system compromise, which will significantly impact privacy, reputation and legality. Next, the assets that would be impacted would also be critical (5) as JBoss EAP is often used to manage mission-critical applications and services. Affected assets may include databases, application servers, and other critical infrastructure. Finally, Coverage would be critical (5) as if the vulnerability is not patched, all instances of JBoss EAP that are publicly accessible are potentially vulnerable.

After assessing the impact of the vulnerability, it is determined that the vulnerability is critical



The Vulnerability Risk score is 25 (High risk) as the likelihood and Impact scores both being 5

A screenshot of a computer

Description automatically generated

The next step to the assessment is determining the mitigation strategy and how it was mitigated and when the fixes where applied. This is a known vulnerability so there are known fixes which apply a patch that can be found from redhat[.]com and ensure JBoss EAP instances are securely configured, with restricted access and limited exposure. This was accomplished by me using SNYK’s known solutions and having it create a pull request to apply the patch to my GitHub directly.

Now that the mitigation has been applied new scores can be applied to the vulnerability for future documentation. The score for likelihood I decided to give it a 2 (unlikely) 2-2-1 as the vulnerability has been patched and cannot be accessed easily and could only be done with admin privileges without replication. The new impact score would be moderate (10) because even if exploited, the effects would be lessened by the security measures. The 2 and 10 would now give it a risk score of 6 (medium) and wrap up this vulnerability for the assessment.

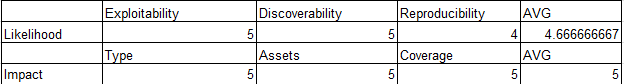
## VID – 002

CVE-2015-4852 is a vulnerability related to Oracle WebLogic Server that allows remote attackers to execute arbitrary code via a crafted XML document. This vulnerability is significant due to its potential for remote code execution (RCE), which can lead to full system compromise.

Just like with VID1 the vulnerability is a remote code execution. This vulnerability involves sending a crafted XML document to the WebLogic server, which can be exploited remotely and without any need for credentials making the exploitability Very Likely (5). Next, the vulnerability can be discovered by scanning publicly accessible WebLogic instances and checking for the specific XML handling flaw making its discoverability also Very Likely (5). Finally, once the exploit method is known, it can be repeated with minor customizations deeming it Likely (4) finalizing the Likelihood score as Very Likely (5).

Now for the impact score if the vulnerability is exploited it would allow sensitive data to be exposed potentially having serios privacy implications, violations with protection laws, and damage to reputation making the impact type critical (5). Next for the assets that would be impacted, WebLogic Server is often used to manage mission-critical applications and services making it highly valuable and the impacted assets Critical (5). Finally, for the coverage if the vulnerability is not patched, all instances of WebLogic Server that are publicly accessible are potentially vulnerable deeming it Critical (5) that it is taken care of. Finalizing the impact score as 5.

With these two scores it gives this vulnerability a Risk score of 25 (High risk)



A screenshot of a computer

Description automatically generated

Now for mitigating the vulnerability, as it is a known vulnerability the first step is to apply the security patch provided by Oracle for WebLogic Server. The next step is to ensure that XML parsing in WebLogic Server is secure to prevent XML-based attacks, such as XML External Entity (XXE) attacks. Once again, the mitigation was accomplished by using SNYK’s patch application services for the files. Now that the vulnerability has been patched the revised risk score will be 6 (Unlikely) for the reason that it now requires admin privileges to exploit the vulnerability and by checking that the XML parsing is secure will reduce the effectiveness of the vulnerability if its exploited.

## VID – 004

CVE-2018-1000632 refers to a vulnerability in the XML parser of a specific software where it incorrectly processes XML External Entity (XXE) data. This can allow attackers to access sensitive data on the system or execute arbitrary code, depending on how the software is configured.

To start off the Likelihood calculation the exploitability is not the easiest but achievable without needing full administrative privileges. It requires a user with certain permissions to perform actions that trigger the vulnerability which means that the exploitability is likely (4) to happen. Next, discovering this specific vulnerability requires a good understanding of XML parsing and the software's implementation details, making it not immediately obvious to a general user. This means that the discoverability of this vulnerability is likely (4) to happen but not guaranteed. Finally, once the vulnerability is known, it can be exploited repeatedly with some adjustments. Due to the changes only being minor the reproducibility is Very Likely (5) to happen. This gives likelihood a score of 4 (likely) for not being the hardest vulnerability to exploit but does need some specific knowledge.

Now for impact, XXE vulnerabilities can lead to severe consequences like unauthorized data exposure, denial of service, or remote code execution, all of which can disrupt operations and compromise privacy. This makes the impact type to be critical (5) due to how dangerous the results are if the vulnerability is exploited. Next for impacted assets the software containing the vulnerability is considered a critical asset because it performs essential functions and handles sensitive data. The only affected asset is a single software making the impact high (4) as it doesn’t affect multiple assets. Finally, XML parsers are widely used across various applications and services. The vulnerability can potentially impact a significant portion of the installed base if the same XML parser is used across multiple systems. Due to the broad potential exploitation the coverage impact would be critical (5) as there are many possibilities besides exploiting the same vulnerability. Due to those factors, it makes the vulnerability impact critical (5).

With the likelihood score being 4 and the impact score being 5 the Risk score given is 20 (High risk)

A white rectangular object with black text

Description automatically generated

A colorful squares with black text

Description automatically generated with medium confidence

Mitigating this vulnerability takes form in applying the patch that is provided from the maintainers of the Jackson Databind library. Also, implementing secure configuration practices helps to reduce the attack surface and prevent exploitation of vulnerabilities. The mitigation was accomplished by using SNYK’s patch application services for the files. To end the assessment on this vulnerability the revised scores need to be calculated which both scores are now 2 since the vulnerability isn’t much of a threat after the patches are applied. This gives a risk score of 4 (low) which is acceptable and should not be of future concern.

# Summary

This project provided valuable insights into conducting a comprehensive vulnerability assessment on a network. By leveraging SNYK to scan files and identify known vulnerabilities, I was able to pinpoint potential exploits. Researching these exploits allowed me to evaluate their threat levels and develop appropriate mitigation strategies. Crafting my own risk matrix was particularly enlightening, as it helped me accurately gauge the severity of each vulnerability.

The process of identifying and analyzing vulnerabilities not only increased my understanding of potential threats but also deepened my knowledge of the affected systems and the mechanics of various exploits. This hands-on experience has been invaluable in enhancing my ability to assess and address network security risks effectively.

Future implementation would take form in another risk assessment on a larger scale file system or my own files themselves. Another possibility would be to use a different vulnerability scanner such as the “Coverity scan” which also scans files for known vulnerabilities.