Vulnerability Checklist

1. Outdated software

a. Scan all ports for enumerating all present services on the targeted machine. Services can be hosted on "unsual" for nmap ports.

b. Manually search the enumerated software version for active publicly available vulnerabilities, either from exploit-db or searchsploit (they user the same engine)

2. Information disclosure

a. Often organizations are leaving (by mistake or not) artifacts to the world wide web. This can contain emails, usernames, passwords, ssh keys, open endpoint, log files and more. Always make sure to scan code repositories, pastebins and query search engines as much as possible.

b. Directory-listing is a nice way to enumerate the web application, often sysadmins are storing sensitive endpoints that are not normally queried from search engines, nor people. By using directory busters such as dirb / gobuster, you can find web endpoints that can lead to more information disclosure or other vulnerabilities.

c. Often sysadmins disable "interesting" endpoints to be queried from search engines, by adding them to /robots.txt. Peeking at the file, the attacker can gain better understanding of the application's context.

3. SQL Injection

a. SQL injection is a vulnerability, allowing the attacker to intervene with the connection between the web server and its database server. It is always a good idea to fuzz all the available inputs for sqlinjection payloads (such as '). Depending on the output, you can enumerate if the application is vulnerable.

4. Insecure sudo permissions

a. Sometimes sudo can be misconfigured to allow users to run commands, which can be abused for privilege escalation. Always check this with sudo -l and use *https://gtfobins.github.io/* for performing the privilege escalation vector.

5. Directory traversal

a. If the web application you are testing is requiring a "file" or "f" parameter in any of its requests, it is a good practice to fuzz it for directory traversal vulnerability. If such vulnerability is present, it can allow the attacker to escape the web context and read local system files from any directory (if of course the permissions are present).

6. Cross-site Scripting

a. Cross-site Scripting (XSS) is a vulnerability that allows the attacker to inject custom JavaScript code, mainly attacking other users. It is always a good practice to fuzz all potential parameters and fields with simple js payloads, such as **"<script>alert('1');</script>"**. You can find more payloads at: *https://github.com/payloadbox/xss-payload-list*

SoftUni

© SoftUni – about.softuni.bg. Copyrighted document. Unauthorized copy, reproduction or use is not permitted.

Follow us:

7. Arbitrary FTP file upload

a. Always test FTP servers if you can:

   i. Login anonymously.

   ii. Upload files to the FTP server.

b. After successful upload, make sure to check if you can access the files using other potentially chained application (such as web applications).

c. Always make sure to test if you can also download files, sometimes sensitive data can be stored on FTP servers.

d. If you found credentials throughout the information gathering process or later, always make sure to test them towards as much services as possible, including FTP servers.

8. OS Command injection

a. Always make sure to properly analyze the application you are testing. If you can enumerate some of its functionalities are dependent on OS native commands, always try OS Command injection vulnerability, by injecting simple command. You can see sample payloads at: *https://github.com/payloadbox/command-injection-payload-list*.

9. Access control

a. Always make sure to test for access control vulnerabilities. If you found endpoint that should require privileges, always try to navigate to it by analyzing the request and messing around with its payloads.

10. Arbitrary File Upload

a. Always make sure to test for arbitrary file upload whenever you found file upload function. To test you need to answer:

   i. Where the files are being uploaded?

   ii. What programming language your target is using?

   iii. Can you upload files of that extension?

   iv. Can you access and execute that file?

SoftUni

© SoftUni – about.softuni.bg. Copyrighted document. Unauthorized copy, reproduction or use is not permitted.

Follow us: