Using ssh port forwarding

Zachary Zebrowski

zakz@zakz.info

This talk is related to the Web Identity talk on opensecuritytraining.info

This talk goes into depth about how to use ssh port forwarding (and other tools) to obfuscate where a server is that is serving content to the internet, through multiple techniques. Thus it falls under the similar share alike license that is required by the Creative Commons.

About me

- Novalug member since (at least) 2013.
- Know enough to be dangerous.
- I play with tor related technologies... and kites and drones.
- I have public presentations on web identity and forensic database analysis.

Why bother

- You can host data locally and share globally using (mostly) your own resources.
- You may be able to share other ports for remote administration of a server.
- You can share a demo app that you created locally.
- Be aware of how certain ransomware or similar is distributed.

External Tools

- There are various websites that allow you to (with various levels of authentication due to spam etc), to be able to do port forwarding as a service. The more expensive the service, the more resistant to (spam / other bad things), as well as general updtime. Some have custom clients, that use ssh under the hood. Some do not.

- Custom Clients Required ** ngrok (freemium) ** localtonet.com (freemium)
- Pure ssh ** localhost.run (free) ** ssi.sh (free but requires go client)

SSH Syntax

- -L , -R and -D.
- -L specifys that you are opening a local port as if it was on the remote server. ** Syntax: ssh -L 8080:localhost:80 root@remote2.zakz.biz ** First port (8080) is the listening port on local server. ** Second port is the port that the server is connecting to on localhost. ** localhost can be changed to an arbitrary host on the internet if desired.

- -R is the reverse. The remote server is listening and forwarding traffic to your local server. ** Syntax: ssh -R 8080:localhost:80 root@remote2.zakz.biz ** First port is the remote port you want to have exposed to the internet. (8080) ** Second port is the local port you want exposed to the internet. (80) ** localhost can be changed to an arbitrary host in your local network or internet host.

- -D opens a Dynamic port. On the port specified on the local server, that port becomes a SOCKS proxy as if you were using the remote host for all ports. ** Syntax: ssh -D 9999 zaz@remote2.zakz.biz # Opens a local socks port 9999 for the user zaz on remote2.zakz.biz

---

Demo

1. ensure you are running a webserver.
   In this case, we will use remote2.zakz.biz (a random host I have access too).
2. elinks **http://localhost** # Note content
3. ssh -R 80:localhost:80 nokey@localhost.run # Opening port 80 to the server on localhost.run

---

Gotchas

1. On services like localhost.run, they do get scanned by adversaries, so caveat emptor.
2. If the ip address is not stable for remote host, you will need to reconnect. (See autossh)
3. It can be a way to exfiltrate data.
4. It can host files from another host without being obvious.

Questions?