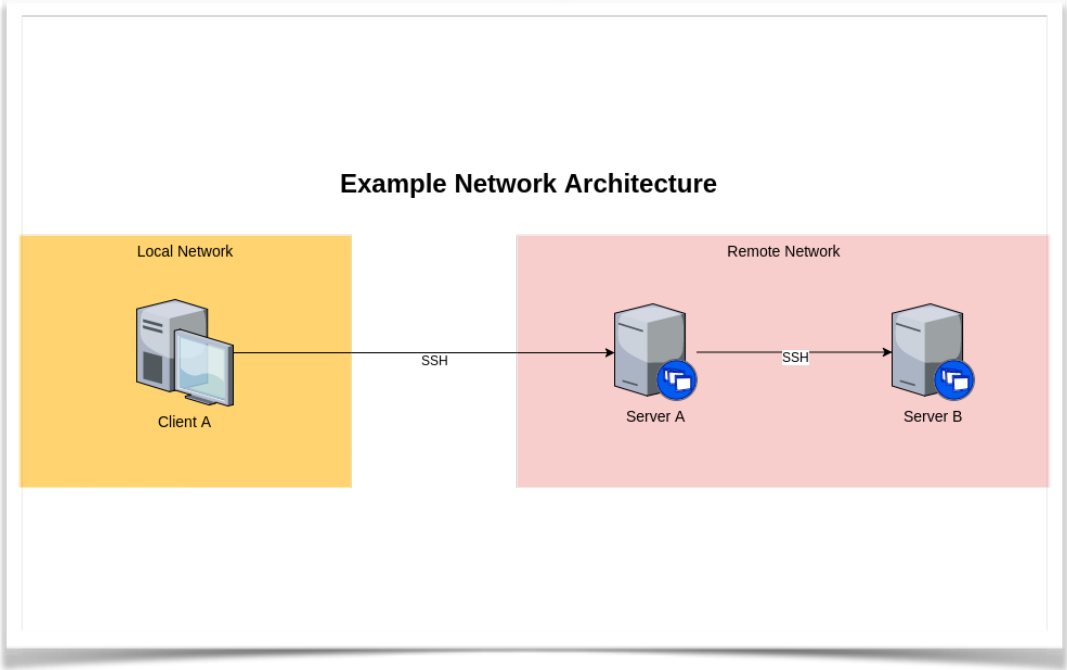
HUGO BARBOSA

MARCH 29, 2019

Intelie Test

DevOps Challenge



Network Architecture (test)

The Challenge

Propose a way to access a service running on a remotely blocked port. The only allowed connection between Client A and Server A is via SSH. The only allowed connection between Server A and Server B is via SSH. We need to access, from Client A and using HTTP, a service running on port 8000 of Server B. There is a service running on port 8000 of Server A. Both client and servers run CentOS 7 without X.

SIMPLES STEPS

1

SERVER B

- Login as root
- Edit /etc/ssh/sshd_config
- Set AllowTcpForwarding to Yes
- systemctl restart sshd

2

SERVER A

- Login as root
- Edit /etc/ssh/sshd_config
- Set AllowTcpForwarding to Yes
- systemctl restart sshd

3

CLIENT A

- ssh -g -L 8001:<ip server A>:8001 <user serverA>@<ip server A>
- ssh -g -L 8001:<ip server B>:8000 <user serverB>@<ip server B>
- .wget localhost:8001

Hugo Barbosa
36 years old, Brazilian, Married, 1 year old son, love games and IT.

Test:
Access Server B on port 8000 (HTTP) through Server A without direct access to the port mentioned.

Method
TCP Port Forward (ssh tunneling)

"*All that is necessary for the triumph of evil is that good men do nothing*" - Edmund Burke

Why SSH forward

Used mainly for testing, the TCP Port forward configuration from ssh daemon can be used in case of TCP port restrictions on networks, however it is not fit for production environment due to security issues. It is a workaround for urgent issues, the proper way would be set up SGs (Security Groups) and VPC (Virtual Private Could) configuration if the services are virtualized (AWS). If not, change the network configuration to allow the port over a VPN and/or endpoint with restricted access.

Possible failures:

Firewall restrictions, Previous configuration from sshd_config, Network Configuration

PRE-REQUIREMENTS

Have login account to access all server with “sudo su -“ permission

PROCEDURE

SERVER B

1 - Open a SSH terminal (PuTTY or unix terminal)

2 - Login as **root** (or other user and sudo su - privileges)

3 - Edit the config file “/etc/ssh/sshd_config”

```
# vi /etc/ssh/sshd_config
```

4 - Set the configuration parameter AllowTcpForwarding to “Yes”, remove any comments on the line (#)

```
AllowTcpForwarding Yes
```

5 - Restart the service sshd

```
# systemctl restart sshd
```

SERVER A

6 - Repeat the steps executed on Server B

CLIENT A

7 - Repeat the steps executed on Server B

8 - Login as **any** user on Client A

9 - Execute the command below (*This command will forward the requests from <ip Client A>:8001 To <ip Server A>:8001*), use the login able to access server A.

Simulated prompt: userCA@hostClientA => username@hostname

```
userCA@hostClientA # ssh -g -L 8001:<ip serverA>:8001 <login server A>@<ip serverA>

Password: <password here>

userSA@hostServerA # ssh -g -L 8001:<ip serverB>:8000 <login server B>@<ip serverB>

Password: <password here>

userSB@hostServerB #
```

IMPORTANT

Don't close the terminal so you can keep the connection alive!

TESTING

10 - Now you can access from any server on Client A:8001 but the content will be actually from Server B: 8000:

Information: The "-g" option allows you to remotely access the Client A on the port 8001. Without it, you can only access the content from Client A (localhost).

```
userCA@hostClientA # wget localhost:8001
```

Or

```
userCA@hostClientA # curl <ip ClientA>:8001
```