**Bypassing IGS Fortinet Firewall port 443 :D**

**# To set up a lab environment at ease using AWS:**

Login => EC2 => Launch Instance => Enter Linux name => Create security group => Network Settings allow HTTPS, HTTP => create key pair =>

RSA + .ppk =>

EC2 => Instances (running)

A screenshot of a computer

Description automatically generated

34.252.139.254

PuTTY

A screenshot of a computer

Description automatically generated

Connection => SSH => Auth

Browse... => Select earlier downloaded .pem from aws

A screenshot of a computer

Description automatically generated

AWS => EC2 => Instances (running)

A screenshot of a computer

Description automatically generated

Select Linux3 => Connect

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Copy command:

ssh -i "linux3.pem" ec2-user@ec2-34-243-197-243.eu-west-1.compute.amazonaws.com;

A screenshot of a computer

Description automatically generated

.pem is in EC2 => Key pairs

# Upon connecting to your ec2 as ec2-user

# Change /etc/ssh/sshd\_config to allow SSH tunneling

<https://dev.to/tastaslim/how-to-enable-password-authentication-in-aws-ec2-instances-40m6>

sudo vi /etc/ssh/sshd\_config;

Port22

A black screen with blue text

Description automatically generated

PasswordAuthentication yes

A computer screen with blue text

Description automatically generated

AllowAgentForwarding yes

AllowTcpForwarding yes

GatewayPorts yes

X11Forwarding yes

A computer screen shot of text

Description automatically generated

# Restart sshd service

sudo service sshd restart;

On you closed port 22 Linux waiting to Tunnel through Firewall

ssh ec2-user@**34.243.197.243**;

Our Kali Linux

A screenshot of a computer

Description automatically generated

My laptop windows

A computer screen shot of a computer program

Description automatically generated

**Now, let's do SSH tunnelling Local Forwarding :D !!**

A blackboard with a diagram of a remote tunnel

Description automatically generated

A blackboard with colorful text

Description automatically generated

**On our Kali attempting to connect to Remote AWS EC2 Linux with Public facing Gateway ports open as ec2-user**

**ssh -L localhost:localPort:remoteSide(AWS):remotePort**

A screenshot of a computer

Description automatically generated

**(Kali) On our side, we wanna listen on 127.0.0.1 on port 8000**

**Our Kali will act as a Web Server on 127.0.0.1 on port 8000**

**ssh -L localhost:8082:localhost:8082 root@192.168.2.240**

**(AWS) On the other end, we wanna hook up to localhost:80 on AWS**

**Kali will connect to our AWS Web Server localhost on port 80**

A computer screen shot of a computer

Description automatically generated

**We're now on AWS EC2 Linux as ec2-user**

**Now, the tunnel has been set up! :D**

**On AWS EC2 => install web server**

**sudo yum install httpd -y;**

A screenshot of a computer

Description automatically generated

**# Start Web Server on AWS EC2**

**sudo service httpd start;**

A black background with white text

Description automatically generated

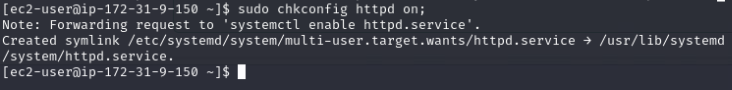
**sudo service httpd status;**

A screenshot of a computer program

Description automatically generated

**#**

**sudo chkconfig httpd on;**



**Any browser either Kali or laptop to browse internet**

[**http://**34.243.197.243:80/](http://34.243.197.243:80/)

Thus, AWS EC2's Web Server has go online in the internet

A screenshot of a computer

Description automatically generated

**if we hosted a Web Site on AWS EC2 Linux (Public IP: 34.243.197.243**)

**\*\*\* On Kali, we can browse it using browser \*\*\***

[**http://localhost:8000/**](http://localhost:8000/)

**On our Kali linux**

A screenshot of a computer

Description automatically generated

**As we can browse the Web Server hosted globally on AWS EC2 on**

[**http://34.243.197.243:80**](http://34.243.197.243:80)

**Now, we confirm that the SSH tunnel local forwarding has been set up well ! :D**

**Combined with DNS spoofing, we can man-in-the-middle anyone in this private network, pretending some websites to be legitimate, yet do NOT do this kind of stuff without permissions :D**

**Next, we'll use our Kali to do Remote Forwarding SSH Tunneling to our AWS EC2 Linux**

**First, we exit AWS EC2 as ec2-user;**

**exit;**

A black background with white numbers

Description automatically generated

**[Our AWS EC2 is NOT currently listening on & forwarding port 2222]**

**ssh -R remoteHost:remotePort:localhost:localPort ec2-user@34.243.197.243;**

**ssh -R 127.0.0.1:2222:127.0.0.1:22 ec2-user@34.243.197.243;**

**We're connected using a Remote Forwarding SSH Tunnel**

A computer screen with white text

Description automatically generated

**ec2-user ifconfig**

**IP: 172.31.9.150**

**netmask 255.255.240.0**

**Before we proceed Remote SSH Tunneling :D**

**we'll need to confirm that:**

**i. Our Kali is NOT opening port 2222!**

A computer screen with white text

Description automatically generated

**ii. Our AWS EC2 is NOT publicly opening port 2222!**

A computer screen shot of a computer

Description automatically generated

**We're now on AWS EC2 Linux**

**Yet, if we do:**

**sudo ssh -p 2222 localhost;**

A computer screen shot of a computer code

Description automatically generated

**yes**

**Enter password**

A screen shot of a computer

Description automatically generated

**After we go back to our Kali from AWS EC2 using Remote SSH Tunnel**

**logout**

A screenshot of a computer

Description automatically generated

**Now, we're back to AWS EC2 Linux**

A computer screen shot of a number

Description automatically generated

**on AWS EC2 Linux, install netcat to telnet to 127.0.0.1:2222**

**sudo yum install nc -y;**

A screenshot of a computer

Description automatically generated

**Now, on AWS EC2 Linux, see whether 127.0.0.1:2222 is now open :D**

A black background with white text

Description automatically generated

**This Remote SSH Tunneling has succeeded! :D**

**Why?**

**1. On our Kali, we did NOT open port 2222!**

A computer screen with white text

Description automatically generated

**2. On our AWS EC2 Linux, we did NOT publicly open port 2222!**

A computer screen shot of a black screen

Description automatically generated

**2. From our Kali, we established a Remote SSH Tunnel by:**

**ssh -R remoteHost:remotePort:localhost:localPort <user@publicIP>;**

**ssh -R 127.0.0.1:2222:127.0.0.1:22 ec2-user@34.243.197.243;**

**3. Upon connecting to AWS EC2, we go back to our Kali via port 2222**

**ssh -p 2222 localhost;**

**4. We log out from Kali, it goes back to AWS EC2, we nc -v 127.0.0.1:2222**

A black background with white text

Description automatically generated

**This Remote SSH Tunneling does NOT require either of our Kali to open port 2222 locally NOR our AWS EC2 Linux to publicly open port 2222,**

**this has completely bypassed Firewall rules!**

**All traffic in this deep net has been encapsulated in SSH Tunnels go through IGS Fortinet Firewall port 443 :D  
Cuz this entire connection was undertaken completely within our Firewall**

**which does NOT need Port Forwarding for 2222**