Universität Passau || Security Insider Lab I: Infrastructure Security || WS15

Part 4: Virtual Private Networks

Ahmed Azri | Emna Maâtoug

February 11, 2016

Advisor: Oussama Mahjoub, Bouthayna Belgacem

Contents

1 Exercise 1: OpenVPN : GATEWAY <–> Road-warrior

In order to complete this lab exercise, we need one additional virtual machine, called "VPN-gateway". The network will then look as the one depicted in the figure below which overviews all IP configurations of the different virtual machines in our network. We will also install and use the software package OpenVPN as the basis for our VPN gateway. OpenVPN can be used as a server as well as a client.

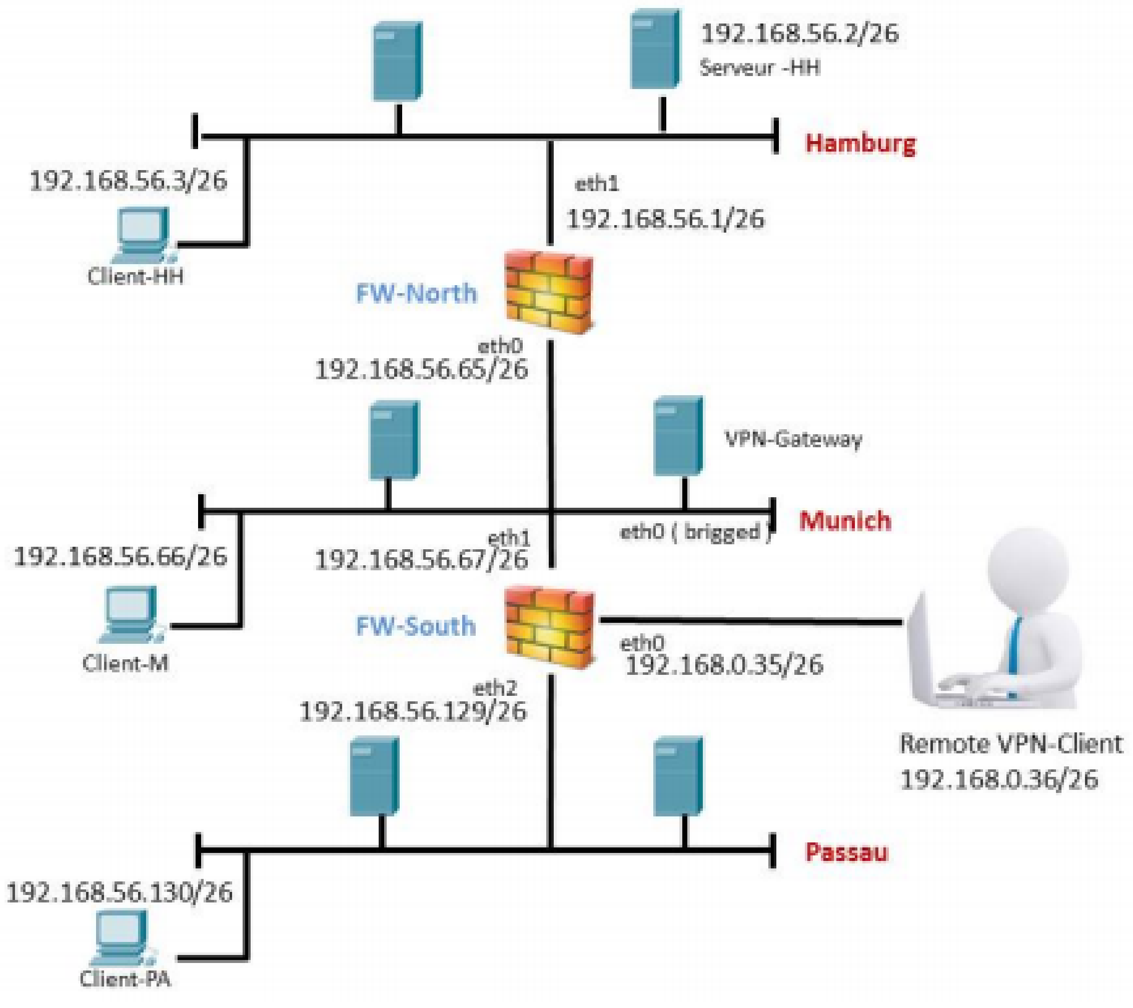

Figure 1: Desired 'VPN Extension' of the existing network setup

In this exercise, we will establish a VPN connection between a gateway (VPN Gateway) and a mobile "road warrior" (Remote Client). To achieve this, we will be using another VM that will be playing the role of the remote client.

In order to configure our VPN connection correctly, we'll be proceeding as follows. First, on both VPN-Gateway and remote client virtual machines, we start by installing OpenVPN using the following command: apt-get install openvpn

Then, in order to be able to create and manage bridging of network interfaces on the VPN-Gateway for later use, the bridge-utils package needs to be installed:

- sudo apt-get install bridge-utils

Now before configuring OpenVPN, we need to allow remote UDP traffic from the outside world to the VPN-Gateway via FW-south's eth0 interface on port 443 using the udp protocol:

- iptables -I FORWARD -p udp –dport 443 -i eth0 -j ACCEPT

Then, and in order to make sure that all incoming VPN traffic on FW-south's eth0 interface will be routed to the VPN-Gateway, the following rule can be set up to translate every packet's destination to the IP address of the VPN-Gateway:

- iptables -t nat -A PREROUTING -i eth0 -p udp –dport 443 -j DNAT –to-destination 192.168.56.68:443

Also, packet forwarding on FW-south needs to be enabled. We need to make this permanent so that FW-south still forwards traffic after rebooting. So first, we open the *sysctl* setting file (/etc/sysctl.conf) and then we uncomment the next line to enable packet forwarding for IPv4: #net.ipv4.ip_forward=1

Afterwards, on the VPN-Gateway, we need to create a PreShared Symmetric Key as follows:

- openvpn –genkey –secret static.key

Then, the server configuration file is the starting point for the OpenVPN server configuration. This file will create a VPN using the TAP network interface (tap0 interface) and will listen for client connections on UDP port 443. It also includes external scripts that will be executed when starting up or shutting down the OpenVPN service.

/etc/openvpn/server.conf

```
port 443
proto udp
dev tap0
```

```
verb 5
secret static.key
log openvpn.log
script-security 3 system
up ./start.sh
down ./stop.sh
```

The following describes the content of both start and stop external scripts that were mentioned in the server configuration file. The start script will set up the bridge and tap interfaces and configure the bridging when OpenVPN service starts, whereas the stop script will remove these interfaces at the end of the OpenVPN connection.

/etc/openvpn/start.sh

```
#!/bin/bash
br="br0"
tap="tap0"
eth="eth0"
eth_ip="192.168.56.68"
eth_netmask="255.255.255.192"
eth_broadcast="192.168.56.127"
for t in $tap; do
openvpn --mktun --dev $t
done
brctl addbr $br brctl
addif $br $eth
for t in $tap; do
brctl addif $br $t
done
for t in $tap; do
ifconfig $t 0.0.0.0 promisc up
done
ifconfig $eth 0.0.0.0 promisc up
ifconfig $br $eth_ip netmask $eth_netmask
broadcast $eth_broadcast
```

/etc/openvpn/bridge_stop.sh

```
#!/bin/bash
br="br0"
tap="tap0"
```

```
ifconfig $br down
brctl delbr $br
for t in $tap; do
openvpn --rmtun --dev $t
done
```

Now that the VPN-Gateway server is ready, let's move on to the OpenVPN client configuration. Like for the server, a client configuration file needs to be set up on the remote client machine to enable connection to the VPN-Gateway. Similarly, the client will be using the udp protocol on port 443 over the tap interface. It also contains the IP address of the client itself and will configure openVPN to adjust the client's routes to use the VPN connection to access our three internal subnets.
/etc/openvpn/client.conf

```
dev tap
proto udp
remote 192.168.12.1 443
secret static.key
ifconfig 192.168.56.71 255.255.255.0
route 192.168.56.0 255.255.255.0
route 192.168.56.64 255.255.255.0
route 192.168.56.128 255.255.255.0
```

Once done with all client and server configurations, all we need to do is to start the openVPN service on the VPN-Gateway: service openvpn start
Afterwards, we can establish a connection to the VPN-Gateway server from our remote client using the following command: openvpn client.conf
What's left is to add a default route on the remote client with the VPN-Gateway server as a gateway for our VPN connections: route add default gw 192.168.56.68

1.1 Using own words, explain the difference between an IPsec and an SSL based VPN? Which one is implemented by OpenVPN?

The difference between IPsec and SSL-based VPN is the network layer at which they function. IPsec is a Layer 3 VPN. For both network-to-network and remote-access deployments, an encrypted Layer 3 tunnel is established between the peers. An SSL VPN, in contrast, is typically a remote-access technology that provides Layer 6 encryption services for Layer 7 applications and, through local redirection on the

client, tunnels other TCP protocols.
OpenVPN implements SSL-based VPN.

1.2 Try to find reasons why we chose port 443?

The reason is that port 443 is, possibly, the last one to be blocked by a firewall. Often on locked-down networks, only ports like 80 and 443 will be allowed out for security reasons, and running OpenVPN on these allowed ports can help to get out in situations where access may otherwise be restricted.

1.3 What is the difference between the TUN and the TAP interfaces?

The essential difference between TUN and TAP is the OSI layer at which they function. TAP functions as a physical extension to the ethernet cable our machine is connected to. This means it can pass any frame which exists on that wire (IPv4/6, Netware IPX and Appletalk). TUN functions as an endpoint to a TUNnel and only passes routable IPv4 packets. It also requires routing to be correctly set up so that those packets can be correctly routed to the next hop.
Super simplistically, a TAP interface is a virtual ethernet adapter, while a TUN interface is a virtual point-to-point IP link.

1.4 Why they are (TUN and TAP) called virtual (Virtual from the virtual machine's point of view)?

Being network devices supported entirely in software, TUN and TAP are called virtual network kernel devices. They differ from ordinary network devices which are backed up by hardware network adapters.

1.5 What does the term 'VPN-Endpoint' mean ? What are the endpoints in your setting? what are the implication in terms of encrypted traffic?

VPN goes between a machine and a network (road-warrior), or a LAN and a network (site-to-site). Each end of the connection is a VPN "endpoint".
Endpoints in our settings are two virtual machines. The first one is acting as the VPN server which is the VPN-Gateway in our case, and the second one is acting as the VPN client which is the remote client machine.

1.6 What is the state of the physical network interface? why is this state necessary for VPN to work?

The state of the physical interface of the gateway needs to be set to "promiscuous mode". This tells the interface to forward all packets arriving at the NIC to the operating system, even those who are not addressed to the interface itself, so that packets that arrive can be sent through the bridge to the TAP device.

2 Exercise 2: OpenVPN : Site to Site

2.1 What are the pro/cons for using PKI or static keys

A PKI allows computers to authenticate to each other or encrypt and decrypt data without prior contact. So no password exchange on forehand needed, no exchange of passwords over any medium.
As for the disadvantages, a thorough understanding of PKI and asymmetric encryption principles is needed to set this up. It's not the simplest thing to do. Also, asymmetric encryption is slow. That's why it is only used on short messages. For large volume encryption, a PKI can be used to exchange the keys of a (fast) symmetric algorithm.

On the other hand, Pre-shared key authentication does not require the hardware and configuration investment of a public key infrastructure. Pre-shared keys are simple to configure on a remote access server, and they are relatively simple to configure on a remote access client.

However, the biggest problem with static key encryption is that we need to have a way to get the key to the party with whom we are sharing data. The key must be transmitted (either manually or through a communication channel) since the same key is used for encryption and decryption. A serious concern is that there may be a chance that an enemy can discover the secret key during transmission. When someone gets their hands on a symmetric key, they can decrypt everything encrypted with that key.

2.2 What Do you suggest to use TAP or TUN? and why?

The TAP interface should be used and not the TUN one. In fact, the need of routing the packets from the source to the destination requires the use of that kind of interfaces.

List of Figures