Report on VPN (Task 4)

Name: Aravindh JAWAHAR

StrongSwan Ipsec VPN

This lab will be composed of 4 CentOS 8 VM with 1GB of ram and 2vCPUS each. They will need to in same subnet and be able to reach each other.

0. Firewall for the VM(1,2) has been disabled and we delete the IP tables by flushing it if any values

```
[root@centos8 centos]# systemctl stop firewalld
[root@centos8 centos]# systemctl disable firewalld
Removed /etc/systemd/system/multi-user.target.wants/firewalld.service.
Removed /etc/systemd/system/dbus-org.fedoraproject.FirewallD1.service.
[root@centos8 centos]# iptables -F
[root@centos8 centos]# iptables -X
[root@centos8 centos]# iptables -t nat -F
[root@centos8 centos]# iptables -t nat -X
```

   SELinux has been disabled in the VM 1 and 2

1. Enabling routing on both VM's(VM-1 and VM-2) as follows
   Enabling the kernel IP forwarding functionality in /etc/sysctl.conf

```
# sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).
net.ipv4.ip_forward = 1
net.ipv6.conf.all.forwarding = 1
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
```

2. Adding routes to the new remote local network by enabling the scripts in /etc/sysconfig/network-scripts/route-ens34

```
[root@centos8 ~]# cd /etc/sysconfig
[root@centos8 sysconfig]# cd network-scripts/
[root@centos8 network-scripts]# ls
ifcfg-ens33  route-ens34
```

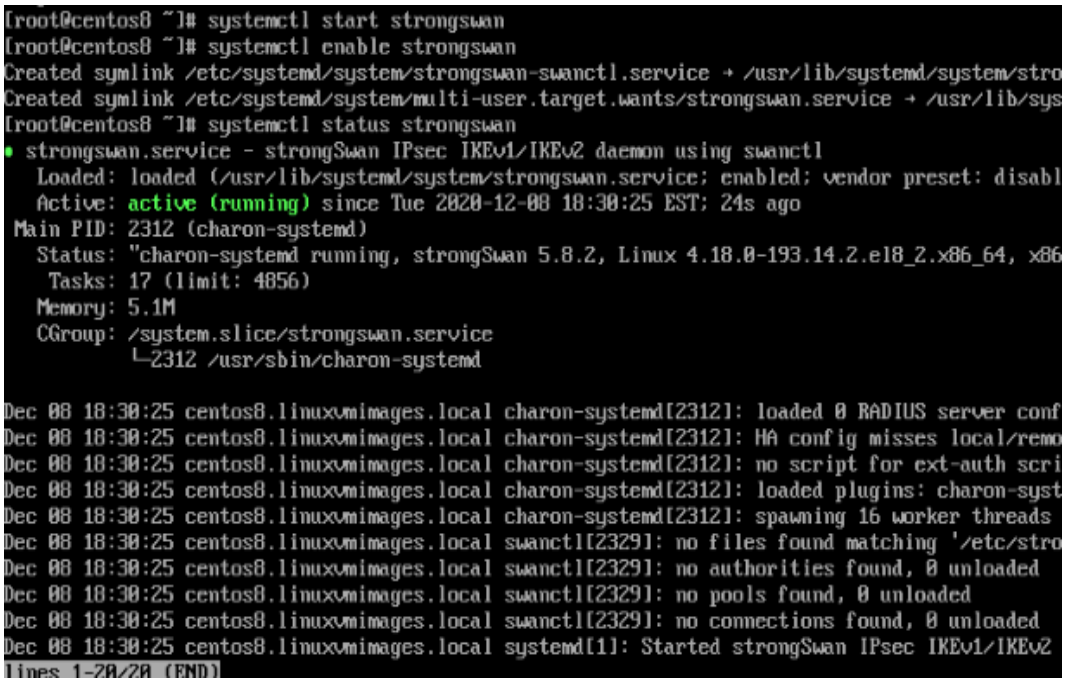
3. Installing **strongswan** on VM-1 and VM-2

```
[root@centos8 ~]# yum makecache
CentOS-8 - AppStream
CentOS-8 - Base
CentOS-8 - Extras
Extra Packages for Enterprise Linux Modular 8 - x86_64
Extra Packages for Enterprise Linux 8 - x86_64
Metadata cache created.
[root@centos8 ~]# yum install strongswan -y
Last metadata expiration check: 0:00:36 ago on Tue 08 Dec 2020 06:29:27 PM EST.
Dependencies resolved.
================================================================================
 Package                    Architecture                Version
================================================================================
Installing:
 strongswan                 x86_64                      5.8.2-5.el8

Transaction Summary
================================================================================
Install  1 Package

Total download size: 1.5 M
Installed size: 4.2 M
Downloading Packages:
strongswan-5.8.2-5.el8.x86_64.rpm
```

4. Enabling the service (strongswan) on VM-1 and VM-2



5. On both VMS we copy the contents of ipsec.conf to ipsec.conf.orig
6. Configure the connection profiles on each security gateways for each site using the **/etc/strongswan/ipsec.conf** strongswan configuration file

   Config setup - the general configuration information for IPSec which applies to all connections

   charondebug specifies the amount of charon debug output to be logged

   uniqueids - participant ID

   conn gateway1-to-gateway2 - setting connection name

   type - connection type

   auto - handle connection when ipsec started or restarted

   keyexchange - version of IKE protocol

   authby -specifies how peers authenticate each other

   left - left participant public IP

   leftsubnet - private subnet of left

   right - right participant public IP

   rightsubnet - private subnet behind left

   ike - declare list of IKE authentication algorithm

   esp - specify list pf ESP algorithm for connection

   aggressive - declares to use aggressive or main mode

   keyingtries - defines attempts t negotiate a connection

   ikelifetime - pecifies how long the keying channel of a connection should last before being renegotiated

   lifetime - length of instance of connection

   dpddelat - declares the time interval of message exchanges sent to the peer

   dpdtimeout - timeout interval incase the connections are deleted after inactivity

   dpdaction - uses dead peer connection protocol to manage the connection

VM-1

```
conn gateway1-to-gateway2
        type=tunnel
        auto=start
        keyexchange=ikev2
        authby=secret
        left=192.168.182.129
        leftsubnet=10.10.1.1/24
        right=192.168.182.130
        rightsubnet=10.20.1.1/24
        iike=aes255-sha1-modp1024!
        esp=aes256-sha1!
        aggressive=no
        keyingtries=%forever
        ikelifetime=28800s
        lifetime=3600s
        dpddelay=30s
        dpdtimeout=120s
        dpdaction=restart
# Sample VPN connections

#conn sample-self-signed
#       leftsubnet=10.1.0.0/16
#       leftcert=selfCert.der
#       leftsendcert=never
#       right=192.168.0.2
#       rightsubnet=10.2.0.0/16
#       rightcert=peerCert.der
#       auto=start

#conn sample-with-ca-cert
#       leftsubnet=10.1.0.0/16
#       leftcert=myCert.pem
#       right=192.168.0.2
#       rightsubnet=10.2.0.0/16
#       rightid="C=CH, O=Linux strongSwan CN=peer name"
#       auto=start
[root@centos8 ~]# _
```

7. VM-2

```
[root@centos8 strongswan]# cat ipsec.conf
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
        # strictcrlpolicy=yes
        uniqueids = yes
        charondebug="all"

# Add connections here.
conn gateway2-to-gateway1
        type=tunnel
        auto=start
        keyexchange=ikev2
        authby=secret
        left=192.168.182.130
        leftsubnet=10.20.1.1/24
        right=192.168.182.129
        rightsubnet=10.10.1.1/24
        iike=aes255-sha1-modp1024!
        esp=aes256-sha1!
        aggressive=no
        keyingtries=%forever
        ikelifetime=28800s
        lifetime=3600s
        dpddelay=30s
        dpdtimeout=120s
        dpdaction=restart
# Sample VPN connections

#conn sample-self-signed
#       leftsubnet=10.1.0.0/16
#       leftcert=selfCert.der
#       leftsendcert=never
#       right=192.168.0.2
#       rightsubnet=10.2.0.0/16
#       rightcert=peerCert.der
#       auto=start

#conn sample-with-ca-cert
#       leftsubnet=10.1.0.0/16
#       leftcert=myCert.pem
#       right=192.168.0.2
#       rightsubnet=10.2.0.0/16
#       rightid="C=CH, O=Linux strongSwan CN=peer name"
#       auto=start
[root@centos8 strongswan]# _
```

8. Pre-Shared Key (**PSK**) is a client **authentication** method that uses a string of 64 hexadecimal digits, or as a passphrase of 8 to 63 printable ASCII characters, to generate unique encryption keys for each wireless client.
   We generate the random key and copy it to /etc/strongswan/ipsec.secrets

```
[root@centos8 strongswan]# ls
ipsec.conf  ipsec.conf.orig  ipsec.d  ipsec.secrets  strongswan.conf  strongswan.d  swanctl
[root@centos8 strongswan]# head -c 24 /dev/urandom | base64 > /etc/strongswan/ipsec.secrets
[root@centos8 strongswan]# _
```

   On VM-1

```
[root@centos8 strongswan]# cat ipsec.secrets
BQiYdLgSVgmZeRpFLiruJllgzhsJQUrj
192.168.182.129 192.168.182.130 : PSK "BQiYdLgSVgmZeRpFLiruJllgzhsJQUrj"
[root@centos8 strongswan]# _
```

   On VM-2

```
[root@centos8 strongswan]# cat ipsec.secrets
# ipsec.secrets - strongSwan IPsec secrets file
192.168.182.130 192.168.182.129 : PSK "BQiYdLgSVgmZeRpFLiruJllgzhsJQUrj"
[root@centos8 strongswan]# _
```

9. We restart the strongswan by rebooting the system and the service
10. Testing the network access via ping for both VM's(1 and 2)
    Ping to tunnel interface from VM-1

```
[root@centos8 strongswan]# ping 10.20.1.1
PING 10.20.1.1 (10.20.1.1) 56(84) bytes of data.
64 bytes from 10.20.1.1: icmp_seq=1 ttl=64 time=0.022 ms
64 bytes from 10.20.1.1: icmp_seq=2 ttl=64 time=0.030 ms
64 bytes from 10.20.1.1: icmp_seq=3 ttl=64 time=0.034 ms
64 bytes from 10.20.1.1: icmp_seq=4 ttl=64 time=0.033 ms
64 bytes from 10.20.1.1: icmp_seq=5 ttl=64 time=0.041 ms
^C
--- 10.20.1.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 113ms
rtt min/avg/max/mdev = 0.022/0.032/0.041/0.006 ms
[root@centos8 strongswan]# ping 10.10.1.1
PING 10.10.1.1 (10.10.1.1) 56(84) bytes of data.
64 bytes from 10.10.1.1: icmp_seq=1 ttl=64 time=0.029 ms
64 bytes from 10.10.1.1: icmp_seq=2 ttl=64 time=0.035 ms
64 bytes from 10.10.1.1: icmp_seq=3 ttl=64 time=0.076 ms
64 bytes from 10.10.1.1: icmp_seq=4 ttl=64 time=0.078 ms
64 bytes from 10.10.1.1: icmp_seq=5 ttl=64 time=0.065 ms
64 bytes from 10.10.1.1: icmp_seq=6 ttl=64 time=0.043 ms
64 bytes from 10.10.1.1: icmp_seq=7 ttl=64 time=0.036 ms
64 bytes from 10.10.1.1: icmp_seq=8 ttl=64 time=0.040 ms
64 bytes from 10.10.1.1: icmp_seq=9 ttl=64 time=0.035 ms
64 bytes from 10.10.1.1: icmp_seq=10 ttl=64 time=0.076 ms
^C
--- 10.10.1.1 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 152ms
rtt min/avg/max/mdev = 0.029/0.051/0.078/0.019 ms
```

Ping to tunnel interface from VM-2

```
[root@centos8 ~]# ping 10.20.1.1
PING 10.20.1.1 (10.20.1.1) 56(84) bytes of data.
64 bytes from 10.20.1.1: icmp_seq=1 ttl=64 time=0.038 ms
64 bytes from 10.20.1.1: icmp_seq=2 ttl=64 time=0.035 ms
64 bytes from 10.20.1.1: icmp_seq=3 ttl=64 time=0.093 ms
^C
--- 10.20.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 52ms
rtt min/avg/max/mdev = 0.035/0.055/0.093/0.027 ms
```

```
       valid_lft forever preferred_lft forever
[root@centos8 ~]# ping 10.10.1.1
PING 10.10.1.1 (10.10.1.1) 56(84) bytes of data.
64 bytes from 10.10.1.1: icmp_seq=1 ttl=64 time=0.033 ms
64 bytes from 10.10.1.1: icmp_seq=2 ttl=64 time=0.034 ms
64 bytes from 10.10.1.1: icmp_seq=3 ttl=64 time=0.041 ms
64 bytes from 10.10.1.1: icmp_seq=4 ttl=64 time=0.029 ms
64 bytes from 10.10.1.1: icmp_seq=5 ttl=64 time=0.039 ms
64 bytes from 10.10.1.1: icmp_seq=6 ttl=64 time=0.040 ms
64 bytes from 10.10.1.1: icmp_seq=7 ttl=64 time=0.035 ms
64 bytes from 10.10.1.1: icmp_seq=8 ttl=64 time=0.036 ms
64 bytes from 10.10.1.1: icmp_seq=9 ttl=64 time=0.032 ms
^C
--- 10.10.1.1 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 189ms
rtt min/avg/max/mdev = 0.029/0.035/0.041/0.006 ms
```

11. Ping to tunnel interface of VM-1 from VM-4 and to VM-2 tunnel interface from VM-3

From VM-3

```
[root@centos8 strongswan]# ping 10.20.1.1
PING 10.20.1.1 (10.20.1.1) 56(84) bytes of data.
64 bytes from 10.20.1.1: icmp_seq=1 ttl=64 time=0.029 ms
64 bytes from 10.20.1.1: icmp_seq=2 ttl=64 time=0.037 ms
64 bytes from 10.20.1.1: icmp_seq=3 ttl=64 time=0.046 ms
64 bytes from 10.20.1.1: icmp_seq=4 ttl=64 time=0.072 ms
64 bytes from 10.20.1.1: icmp_seq=5 ttl=64 time=0.033 ms
64 bytes from 10.20.1.1: icmp_seq=6 ttl=64 time=0.037 ms
64 bytes from 10.20.1.1: icmp_seq=7 ttl=64 time=0.043 ms
^C
--- 10.20.1.1 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 153ms
rtt min/avg/max/mdev = 0.029/0.042/0.072/0.014 ms
```

From VM-4

```
[root@centos8 ~]# ping 10.10.1.1
PING 10.10.1.1 (10.10.1.1) 56(84) bytes of data.
64 bytes from 10.10.1.1: icmp_seq=1 ttl=64 time=0.184 ms
64 bytes from 10.10.1.1: icmp_seq=2 ttl=64 time=0.040 ms
64 bytes from 10.10.1.1: icmp_seq=3 ttl=64 time=0.042 ms
64 bytes from 10.10.1.1: icmp_seq=4 ttl=64 time=0.046 ms
64 bytes from 10.10.1.1: icmp_seq=5 ttl=64 time=0.077 ms
64 bytes from 10.10.1.1: icmp_seq=6 ttl=64 time=0.121 ms
64 bytes from 10.10.1.1: icmp_seq=7 ttl=64 time=0.030 ms
^C
--- 10.10.1.1 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 26ms
rtt min/avg/max/mdev = 0.030/0.077/0.184/0.052 ms
```

12. IPsec, also known as the Internet Protocol Security or IP Security protocol, defines the architecture for security services for IP network traffic. It defines the cryptographic algorithms used to encrypt, decrypt, and authenticate packets, as well as the protocols needed for secure key exchange and key management.
    It has mechanisms for IP security

    A. Encapsulation security payload (ESP)

        Method for encrypting the IP packets

    B. Internet key exchange (IKE)
        Used to manage cryptographic keys used by hosts for Ipsec

    C. The IP Authentication Header (AH)

        for digital signing the IP packets

    Internet key exchange protocol (IEKV2) is defined to allow hosts to specify which services are incorporated in the packets in which algorithms are used to provide services and mechanisms for sharing the keys

    IPsec protocol:

    Defined for both version of IP's ipv4 and ipv6

    Working of Ipsec protocol:

    - When the host recognizes packets should be transmitted using ipsec
    - IKE phase 1, allows two hosts using ipsec to negotiate policy to authenticate themselves and initiate channel between hosts
    - IKE Phase 2, which itself is conducted over the secure channel setup in IKE Phase 1. It requires the two hosts to negotiate and initiate the security association for the IPsec circuit carrying actual network data
    - actual exchange of data across the newly created IPsec encrypted tunnel
    - Termination of Ipsec tunnel when the communication is complete or during session time out. Then the hosts discard the keys over security association