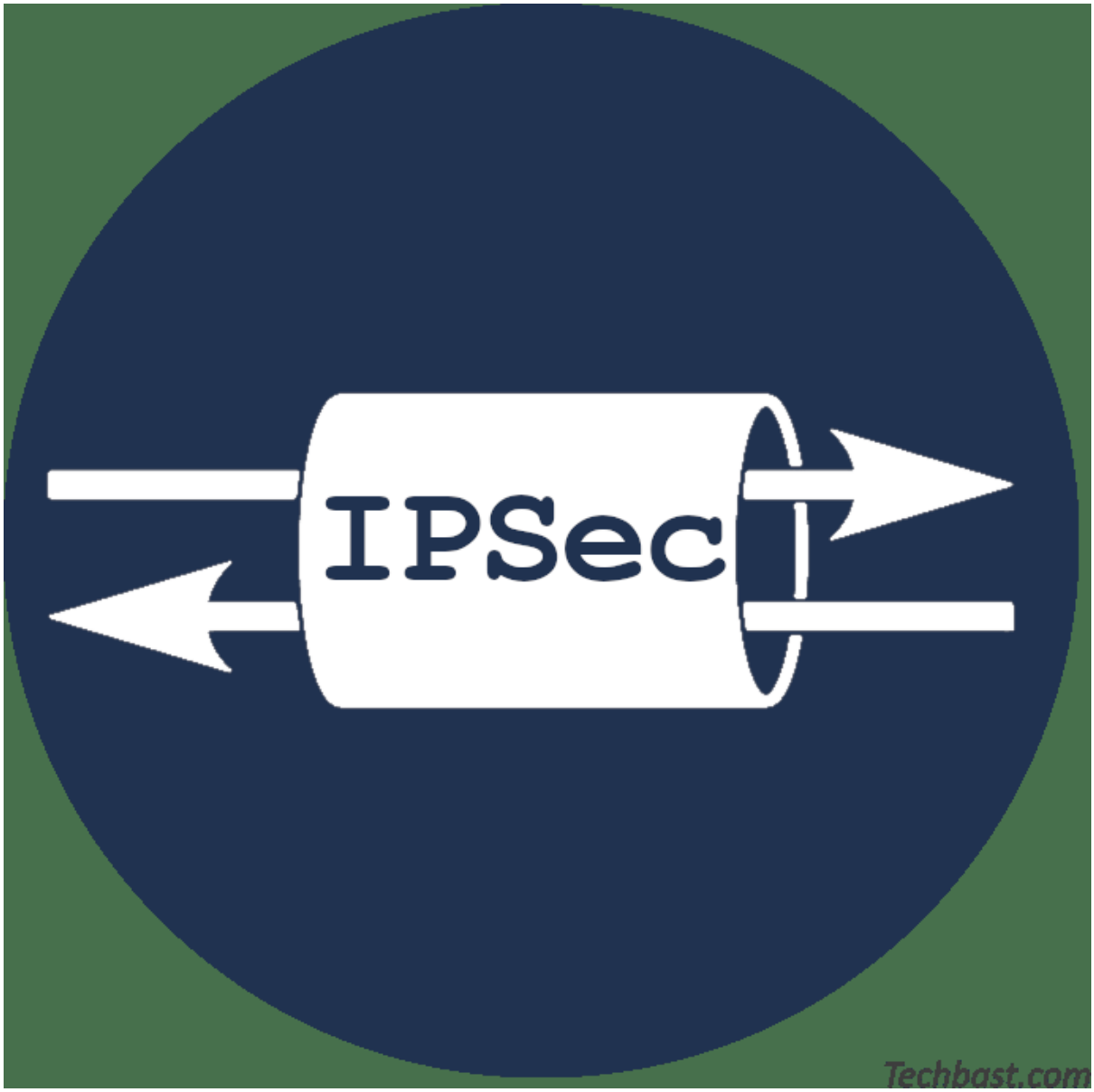

IPSec
Techbast.com

IPSec

ljm

1. Intro

IPSec is a way to create an encrypted VPN over another network (often the Internet). IPSec is sometimes described as easy to implement. But there are many pitfalls. Also, most explanations of what IPSec does are either high level (It is an encrypted VPN) or recaps for people that already know how it works and they just list the terminology.

So what are the actual steps in IPSec?

1. Recognizing traffic that needs to go through the tunnel. This type of traffic is called *interesting traffic* and the packets trigger the set-up of the tunnel.
2. IKE phase 1. Both sides iuse Internet Key Exchange to set-up a set of policies that will be used to create a secure channel.
3. IKE phase 2. This step sets-up an IPSec circuit over the tunnel that was established in phase 1. Both sides also exchange the encryption keys that are used for encrypting the data.
4. IPSec transmission. The actual transmission of data, using the previously constructed tunnels.
5. Termination.

This is in short the IPSec process.

In IKE phase 1, a Security Association is created. A security association (SA) is a logical connection between the two endpoints. It is what we colloquially called the secure channel. The SA is created with the Internet Security Association and Key Management Protocol (ISAKMP). The authenticated keying is offered by the protocol Internet Key Exchange (IKE). in precise terminology, ISAKMP is part of IKE. However, Cisco uses only ISAKMP to implement IKE, and the Cisco does not make a distinction between the two. This may be confusing when you connect to non-Cisco equipment. In phase 1, peers exchange information, including:

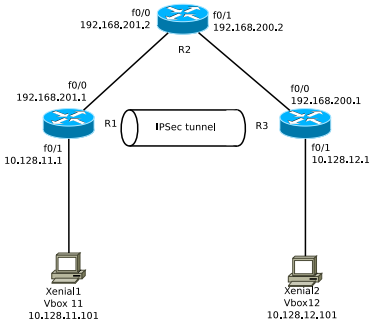
— algoritms that are used: hashing, encryption, Diffie-Hellman group

— the actual Diffie-Hellman exchange to establish a shared secret

— authentication, typically pre-shared keys

2. The basic network

In our first test, we'll create an IPSec tunnel over an intermediate network. The tunnel will be created between router R1 and router R3, both Cisco routers. The intermediate network is represented by router R2.



We make sure that R1, R2 and R3 can see eachother. The IP addressing table is:

Device	Interface	IP address	Subnet mask	Default Gateway
R1	F0/0	192.168.201.1	255.255.255.0	N/A
	F0/1	10.128.11.1	255.255.255.0	N/A
R2	F0/0	192.168.201.2	255.255.255.0	N/A
	F0/1	192.168.200.2	255.255.255.0	N/A
R3	F0/0	192.168.200.1	255.255.255.0	N/A
	F0/1	10.128.12.1	255.255.255.0	N/A
xenial1	vbox11	10.128.11.101	255.255.255.0	10.128.11.1
xenial2	vbox12	10.128.12.101	255.255.255.0	10.128.12.1

We'll configure OSPF to provide us with the complete routing.

2.1 Routers

2.1.1 R1

The basic configuration for R1 is:

```
enable
config t
hostname R1
no ip domain-lookup
int f0/0
ip address 192.168.201.1  255.255.255.0
no shut
int f0/1
ip address 10.128.11.1 255.255.255.0
no shut
router ospf 101
network 192.168.201.0 0.0.0.255 area 0
network 10.128.11.0 0.0.0.255 area 0
```

2.1.2 R2

The basic configuration for R2 is:

```
enable
config t
hostname R2
no ip domain-lookup
int f0/0
ip address 192.168.201.2  255.255.255.0
no shut
int f0/1
ip address 192.168.200.2 255.255.255.0
no shut
router ospf 101
network 192.168.200.0 0.0.0.255 area 0
network 192.168.201.0 0.0.0.255 area 0
```

2.1.3 R3

The basic configuration for R3 is:

```
enable
config t
hostname R3
no ip domain-lookup
int f0/0
ip address 192.168.200.1  255.255.255.0
no shut
int f0/1
ip address 10.128.12.1 255.255.255.0
no shut
router ospf 101
network 192.168.200.0 0.0.0.255 area 0
network 10.128.12.0 0.0.0.255 area 0
```

2.2 Hosts

2.2.1 Vagrantfile

For ease, the hosts are created with Vagrant.

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# Vagrantfile API/syntax version. Don't touch unless you know what you're doing!
VAGRANTFILE_API_VERSION = "2"

Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|

          config.vm.define :xenial1 do |t|
                    t.vm.box = "ubuntu/xenial64"
                    t.vm.box_url = "file:////links/virt_comp/vagrant/boxes/xenial64.box"
                    t.vm.provider "virtualbox" do |prov|
                              prov.customize ["modifyvm", :id, "--nic2", "hostonly", "--hostonlyadapter2", "vboxnet11" ]
                    end
                    t.vm.provision "shell", path: "./setup.xenial1.sh"
          end

          config.vm.define :xenial2 do |t|
                    t.vm.box = "ubuntu/xenial64"
                    t.vm.box_url = "file:////links/virt_comp/vagrant/boxes/xenial64.box"
                    t.vm.provider "virtualbox" do |prov|
                              prov.customize ["modifyvm", :id, "--nic2", "hostonly", "--hostonlyadapter2", "vboxnet12" ]
                    end
                    t.vm.provision "shell", path: "./setup.xenial2.sh"
          end
end
```

2.2.2 Setup for xenial1

```
#!/bin/bash
ETH1=$(dmesg | grep -i 'renamed from eth1' | sed -n 's/: renamed from eth1//;s/.* //p')
ifconfig $ETH1  10.128.11.101 netmask 255.255.255.0 up

route add -net 192.168.192.0 netmask 255.255.192.0 gw 10.128.11.1
route add -net 10.128.0.0  netmask 255.255.0.0 gw 10.128.11.1

apt-get update
apt-get install traceroute
```

Some remarks are useful here. We use `$ETH1` as the network interface, because Ubuntu has chosen to make the name of the interface unpredictable.

We also route only the networks that we use in the lab to R1. In this way, we can ensure that we can reach the host with `vagrant ssh` and that the host can reach the Internet via its NAT adapter.

2.2.3 Setup for xenial2

The same remarks as for xenial1 are also applicable here.

```
#!/bin/bash
ETH1=$(dmesg | grep -i 'renamed from eth1' | sed -n 's/: renamed from eth1//;s/.* //p')


ifconfig $ETH1  10.128.12.101 netmask 255.255.255.0 up
route add -net  192.168.192.0 netmask 255.255.192.0 gw 10.128.12.1
route add -net  10.128.0.0    netmask 255.255.0.0   gw 10.128.12.1


apt-get update
apt-get install traceroute
```

3. IPSec tunnel

3.1 Configuration

We'll create an IPSec tunnel between R1 and R3. This means that we will no longer see the 10.128.x.x traffic on R2, if everything goes as planned.

The Internet Key Exchange must be enabled first on the routers R1 and R3. Remember that with Cisco, there is no real distinction between ISAKMP and IKE.

```
(config)#crypto isakmp enable
```

So, next we'll take the following steps:

- Establish an ISAKMP policy
- Configure pre-shared keys
- Configure the IPsec transform set
- Define interesting traffic
- Apply the crypto map to interfaces

For both routers, we use the following ISAKMP policy:

```
enable
 config t
  crypto isakmp policy 10
  hash sha
  authentication pre-share
  group 14
  lifetime 3600
  encryption aes 256
 end
```

`crypto isakmp policy 10` creates a policy with a priority of 10. 1 is the highest priority, and 10 seems to be a more or less standard priority. As a hashing agorithm, we'll use SHA. That means SHA-1, which is not considered fit for use in the outside world. But this is only a demo environment. We'll use pre-shared keys for the initial set-up. `group 14` means 2048-bit Diffie-Hellman group. And the rest is easy to understand.

The set-up is easy to verify:

```
R1# show crypto isakmp policy
Global IKE policy
Protection suite of priority 10
        encryption algorithm:  AES - Advanced Encryption Standard (256 bit keys).
        hash algorithm:        Secure Hash Standard
        authentication method: Pre-Shared Key
        Diffie-Hellman group:  #14 (2048 bit)
        lifetime:              3600 seconds, no volume limit
```

The next step is to configure the pre-shared keys. IRL, you would not use `test123` as key, ofcourse. Both sides must have the same key. For R1, that would be:

```
 crypto isakmp key test123 address 192.168.200.1
```

and for R3:

```
crypto isakmp key test123 address 192.168.201.1
```

Next is the transform set. A *transform* is a single IPSec security protocol (either AH or ESP) with its corresponding security algorithms and mode.

The AH (Authentication Header) is mainly used for authentication. In addition, AH provides data integrity, data origin authentication, and an optional replay protection service. Data integrity is ensured by using a message digest that is generated by an algorithm such as HMAC-SHA that we use here. Data origin authentication is ensured by using a shared secret key to create the message digest. Replay protection is provided by using a sequence number field with the AH header. AH authenticates IP headers and their payloads, with the exception of certain header fields that can be legitimately changed in transit, such as the Time To Live (TTL) field.

The ESP (Encapsulating Security Payload) protocol provides data confidentiality (encryption) and authentication (data integrity, data origin authentication, and replay protection). ESP can be used with confidentiality only, authentication only, or both confidentiality and authentication. When ESP provides authentication functions, it uses the same algorithms as AH, but the coverage is different. AH-style authentication authenticates the entire IP packet, including the outer IP header, while the ESP authentication mechanism authenticates only the IP datagram portion of the IP packet.

Either protocol can be used alone to protect an IP packet, or both protocols can be applied together to the same IP packet

We'll use ESP in this case. We define the transform set on both routers with:

```
crypto ipsec transform-set 50 esp-aes 256 esp-sha-hmac
```

Next step: define interesting traffic, that is, what traffic should go through the tunnel. Whenever Cisco defines something about traffic, it is always done with an ACL. So here too, we'll create an ACL. For R1:

```
access-list 101 permit ip 10.128.11.0 0.0.0.255 10.128.12.0 0.0.0.255
```

and for R3:

```
access-list 101 permit ip 10.128.12.0 0.0.0.255 10.128.11.0 0.0.0.255
```

A crypto map is a configuration entity that combines:

— Selection data flows that need security processing.

— Definition the policy for these flows and the crypto peer to which that traffic needs to go.

We define it as follows (for R1):

```
crypto map CMAP 10 ipsec-isakmp
match address 101
set peer 192.168.200.1
set pfs group14
set transform-set 50
set security-association lifetime seconds 900
```

The `CMAP` is the name of the crypto map and `10` its sequence number. The `ipsec-isakmp` is the type for this crypto map. The selection of the addresses is done with the ACL that we defined (101). The peer is R3.

PFS, or Perfect Forward Security, ensures will force a new key every set-up, and a Diffie-Hellman key exchange.

group	bits	algortithm	Remarks
1	768-bit	DH	No longer recommended
2	1024-bit	DH	No longer recommended
5	1536-bit	DH	No longer recommended
14	2048-bit	DH	
15	3072-bit	DH	
16	4096-bit	DH	
19	256-bit	elliptic curve DH (ECDH)	
20	384-bit	ECDH	
24	2048-bit	DH/DSA	

The transform-set is defined previously. The lifetime is, at this point quite arbitrary.

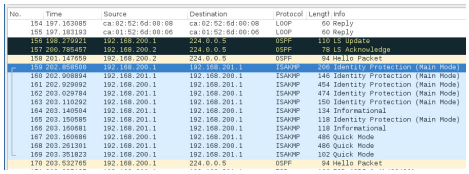
And finaly, we apply the crypto map to the outgoing interface of the router:

```
int f0/0
crypto map CMAP
```

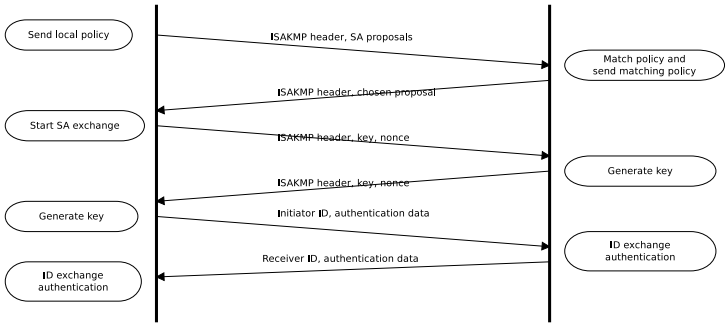
For R3, the only difference is the `set peer 192.168.201.1` (R1 is the peer here).

3.2 Analysis

And after a ping fron xenial1 to xenial2, the tunnel is up before we can fire-up wireshark. Which is good, but does not allow us to examine what is happening. So we just reload R3 (be sure to `wr` to save the configuration!). After some OSPF packets, we see the ISAKMP:



We'll try to match that with the protocol as we know it from text books:



Our first packet contains ISAKMP headers and the SA proposal. It is an UDPs 500 packet from R3 (the initiator) to R2.

```
Frame 159: 206 bytes on wire (1648 bits), 206 bytes captured (1648 bits) on interface 0
Ethernet II, Src: ca:02:52:6d:00:08 (ca:02:52:6d:00:08), Dst: ca:01:52:6d:00:06 (ca:01:52:6d:00:06)
Internet Protocol Version 4, Src: 192.168.200.1, Dst: 192.168.201.1
User Datagram Protocol, Src Port: 500 (500), Dst Port: 500 (500)
    Source Port: 500
    Destination Port: 500
    Length: 172
    Checksum: 0x8da5 [validation disabled]
    [Stream index: 0]
```

The protocol is ISAKMP. To focus on specific lines, I've removed some lines that are less interesting at this point.

```
Internet Security Association and Key Management Protocol
    Initiator SPI: 9e8a87000c6619ff
    Responder SPI: 0000000000000000
    Next payload: Security Association (1)
          <...>
    Type Payload: Proposal (2) # 1
          <...>
        Proposal transforms: 1
            Type Payload: Transform (3) # 1
            Next payload: NONE / No Next Payload  (0)
            Payload length: 36
            Transform number: 1
            Transform ID: KEY_IKE (1)
            Transform IKE Attribute Type (t=1,l=2) Encryption-Algorithm : AES-CBC
            Transform IKE Attribute Type (t=14,l=2) Key-Length : 256
            Transform IKE Attribute Type (t=2,l=2) Hash-Algorithm : SHA
            Transform IKE Attribute Type (t=4,l=2) Group-Description : 2048 bit MODP group
            Transform IKE Attribute Type (t=3,l=2) Authentication-Method : PSK
            Transform IKE Attribute Type (t=11,l=2) Life-Type : Seconds
            Transform IKE Attribute Type (t=12,l=2) Life-Duration : 3600
            Type Payload: Vendor ID (13) : RFC 3947 Negotiation of NAT-Traversal in the IKE
    Next payload: Vendor ID (13)
    <...>
```

Here we see the result of our IOS configs:

crypto isakmp policy 10	
hash sha	Hash-Algorithm : SHA
authentication pre-share	Authentication-Method : PSK
group 14	Group-Description : 2048 bit MODP group
lifetime 3600	Life-Duration : 3600
encryption aes 256	Encryption-Algorithm : AES-CBC Key-Length : 256

Frame 160, the reply to this, has basically the same information.

Next, in packets 161 and 162, we see a nonce being exchanged, and the remainder of the packets just contain encrypted data.

After the ISAKMP, there are a number of ESP packets. These are all encrypted, so it is impossible to see from the network waht is in those packets.

APPENDIX A

Used litterature

Salman, Fatima. (2017). Implementation of IPsec-VPN tunneling using GNS3. Indonesian Journal of Electrical Engineering and Computer Science. 7. 855-860. 10.11591/ijeecs.v7.i3.pp855-860.

https://www.gns3network.com/how-to-configure-ipsec-tunnel-between-cisco-routers/

https://searchsecurity.techtarget.com/definition/IPsec-Internet-Protocol-Security

https://ccnasec.com/8-4-1-3-lab-configuring-a-site-to-site-vpn-using-cisco-ios-instructor-version.html

https://www.ibm.com/docs/en/zos/2.2.0?topic=ipsec-ah-esp-protocols

https://getlabsdone.com/how-to-install-pfsense-on-virtualbox/

CONTENTS