LPIC-2 TRAINING COURSE

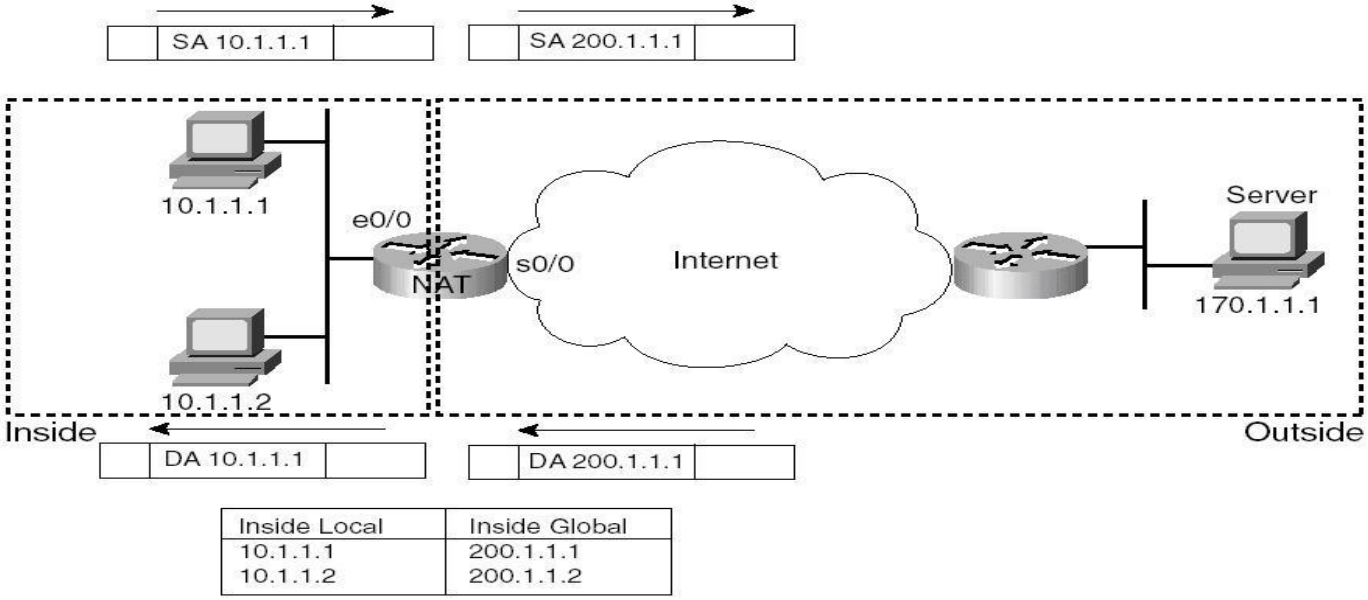
Topic 205: Advanced Networking

Linux Networking Summary

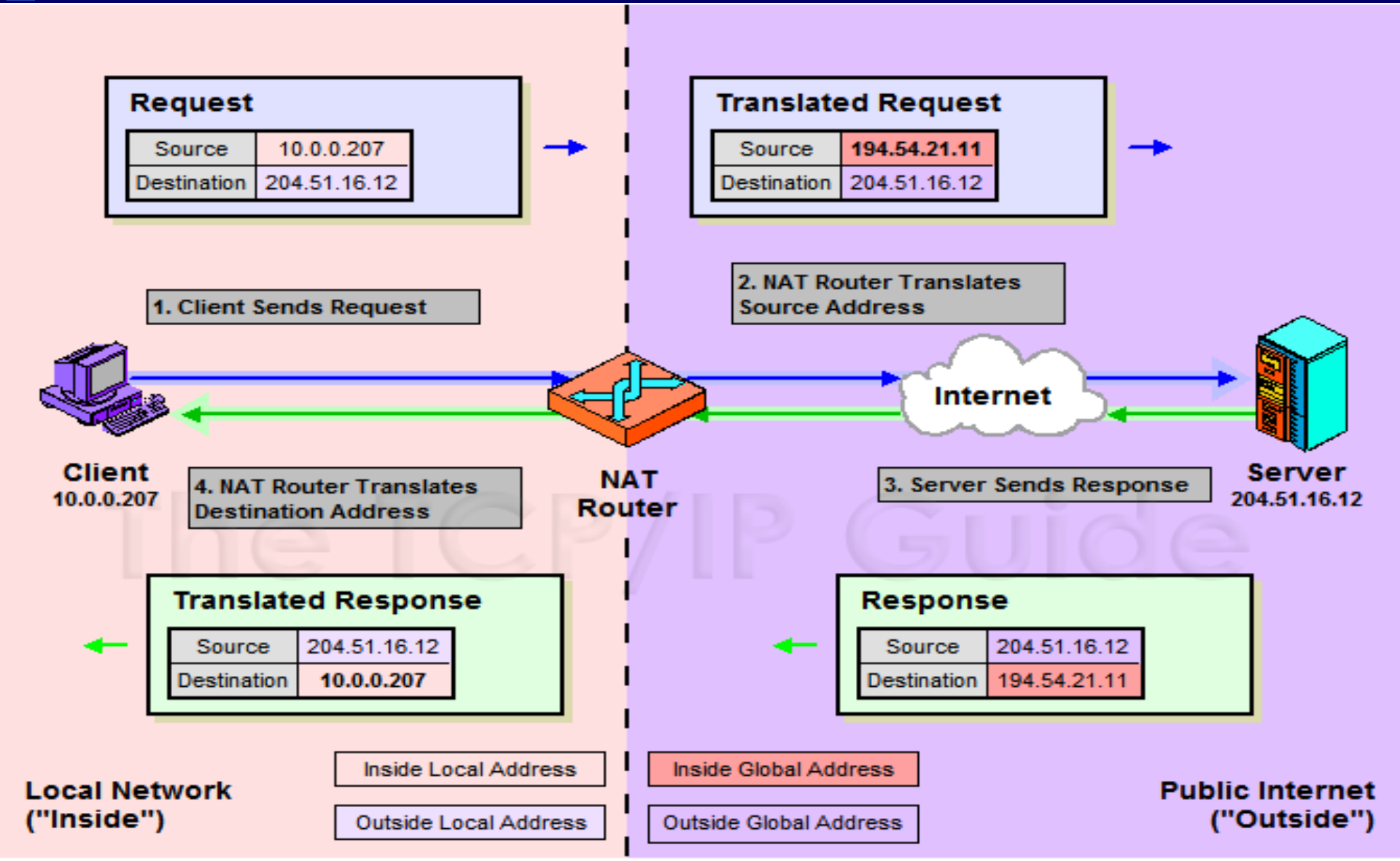
Parameter	Dynamic Configuration	Static Configuration	Verify/Monitor
Host Name	**# hostname *hostname***	**# vi /etc/sysconfig/network** HOSTNAME=*hostname* **# vi /etc/hosts** 127.0.0.1 localhost *hostname*	**hostname**
IP Address & Netmask	**# ifconfig eth*X* *IP_Address* netmask *Netmask***	**# vi /etc/sysconfig/network-scripts/ifcfg-eth*X*** IPADDR=*IP_Adress* NETMASK=*Netmask*	**ifconfig** **ping**
Virtual IP Address	**# ifconfig eth*X*:*Y* *IP_Address* netmask *Netmask***	**# vi /etc/sysconfig/network-scripts/ifcfg-eth*X*:*Y***	
MAC Address	**# ifconfig eth*X* down** **# ifconfig eth*X* hw eth *MACADDR*** **# ifconfig eth*X* up**	**# vi /etc/sysconfig/network-scripts/ifcfg-eth*X*** HWADDR=*MAC_ADDRESS*	**ifconfig** **arp** **arpwatch**
Default Gateway	**# route add default gw *GATEWAY***	**# vi /etc/sysconfig/network** GATEWAY=*Gateway_IP*	**netstat -r** **route** **traceroute**
Routing	**# route add [-net|-host] *Target_IP* gw *Gateway_IP***	Add **route** command to ***/etc/rc.local***	
DNS Server	**# vi /etc/resolv.conf** nameserver *DNS_SERVER_IP*		**nslookup** **host** **dig**
Local resolve	**# vi /etc/hosts** *IP_Address* *Hostname1* *Hostname2* …		

Network Address Translation - NAT

- **Allows connect from a private network to a public network by using address translation**
- **NAT gateway maintain a *translation table***
- **3 types of translation table**
  - Static Mappings (Static NAT)
  - Dynamic Mappings (Dynamic NAT)
  - Masquerading (NPAT - Network Port Address Translation)

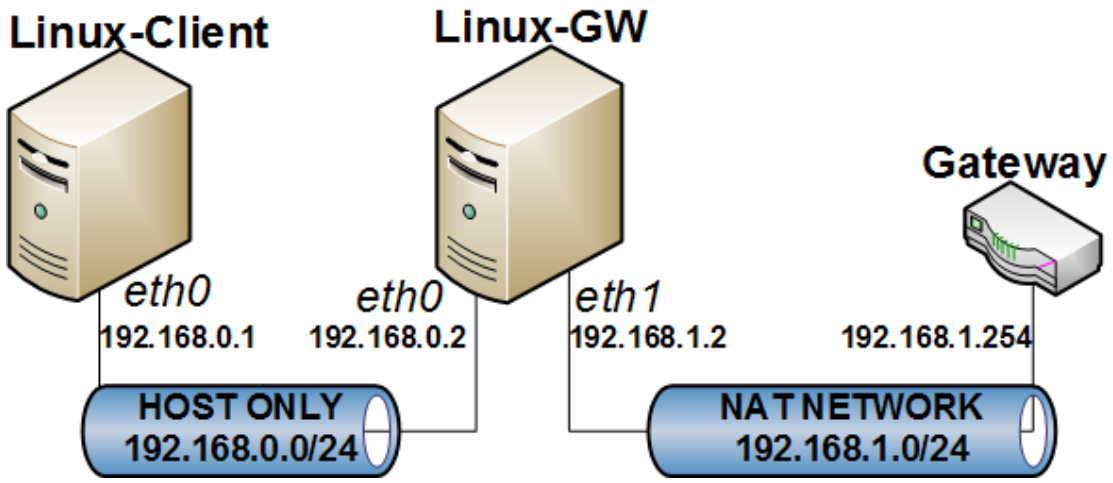


NAT Example


Request
Source 10.0.0.207
Destination 204.51.16.12
Translated Request
Source 194.54.21.11
Destination 204.51.16.12
1. Client Sends Request
2. NAT Router Translates Source Address
Internet
Client
10.0.0.207
4. NAT Router Translates Destination Address
NAT Router
3. Server Sends Response
Server
204.51.16.12
Translated Response
Source 204.51.16.12
Destination 10.0.0.207
Response
Source 204.51.16.12
Destination 194.54.21.11
Inside Local Address
Outside Local Address
Inside Global Address
Outside Global Address
Local Network ("Inside")
Public Internet ("Outside")

Exercise 1: Configuring Linux as a NAT Gateway

Set up a network topology as below with your Virtual Machines



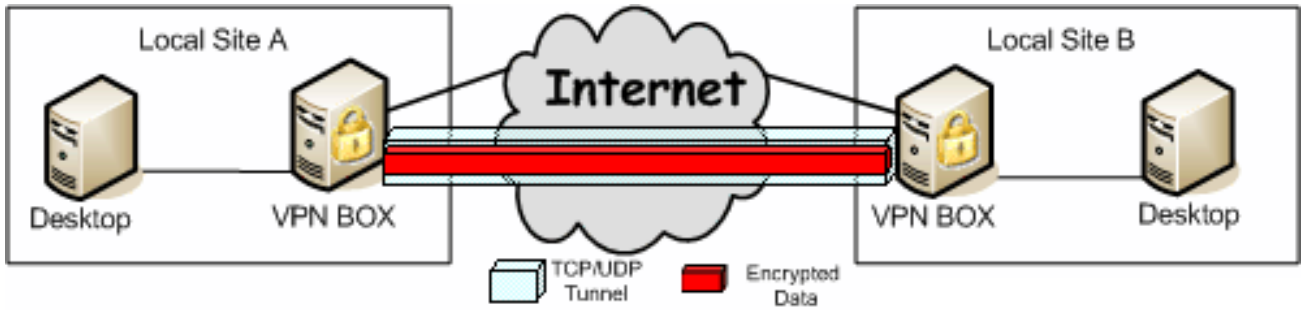
*In this exercise, we will configure Linux-GW as a NAT Gateway using Linux's **iptables** to provide outside access for Linux-Client*

Exercise 1: Configuring Linux as a NAT Gateway (cont')

1. Verify the IP Addresses from all interfaces of **Linux-GW** and **Linux-Client**
   Hint: `[Linux-Client]# ifconfig eth0`
   `[Linux-GW]# ifconfig eth0`
   `[Linux-GW]# ifconfig eth1`
   `[Linux-GW]# route -n`
2. Verify that **Linux-Client** can only ping addresses in the same subnet (192.168.0.0) and **Linux-GW** can ping addresses in both subnet (192.168.0.0 and 192.168.1.0)
3. Turn on IP forwarding on **Linux-GW**. Restart network service and verify
   Hint: `[Linux-GW]# vi /etc/sysctl.conf`
   `net.ipv4.ip_forward = 1`
   `[Linux-GW]# sysctl -p`
   `[Linux-GW]# service network restart`
   `[Linux-GW]# sysctl net.ipv4.ip_forward`
   `net.ipv4.ip_forward = 1`
4. Setup IP forwarding and Masquerading using ***iptables*** on **Linux-GW**
   Hint: `[Linux-GW]# service iptables start`
   `[Linux-GW]# iptables -A POSTROUTING -t nat -o eth1 -s 192.168.0.0/24 -d 0/0 -j MASQUERADE`
5. In **Linux-Client**, point the gateway to **Linux-GW**'s eth0 Address (192.168.0.2)
   Hint: `[Linux-Client]# route add default gw 192.168.0.2`
6. Verify that **Linux-Client** can now ping addresses in both subnet (192.168.0.0 and 192.168.0.1)

Virtual Private Network - VPN

- A network that uses Internet or other network service to transmit data
- Includes authentication and encryption to protect data integrity & confidentiality
- Types of VPNs
  - **Remote Access VPN**: Provides access to internal coporate network over Internet
  - **Site-to-Site VPN**: Connects multiple offices over Internet

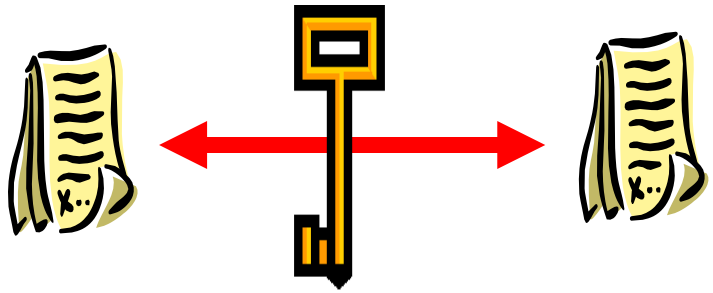


VPN Protocols

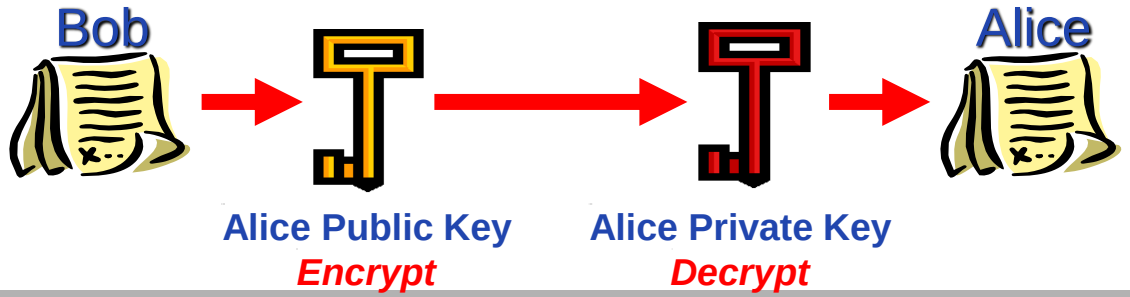
- **PPTP (Point-to-Point Tunneling Protocol): Layer 2 remote access VPN distributed with Windows product family**
  - Uses proprietary authentication and encryption
  - Limited user management and scalability
- **L2TP (Layer 2 Tunneling Protocol): Layer 2 remote access VPN protocol**
  - Combines and extends PPTP and L2F (Cisco supported protocol)
  - Weak authentication and encryption
  - Must be combined with IPSec for enterprise-level security
- **IPSec (Internet Protocol Security): Layer 3 protocol for remote access and site-to-site VPNs**
  - Internet Standard for VPNs
  - Provides flexible encryption and message authentication/integrity
- **SSL (Secure Socket Layer): Layer 4 protocol for remote access and site-to-site VPNs**
  - Authenticate the server and client using PKI
  - Provide an encrypted connection for the client and server to exchange data

VPN Encryption

- **Used to convert data to a secret code for transmission over an trusted network**
- **Symmetric Encryption: DES, 3DES, RC5**
  - Same key used to encrypt and decrypt message
  - Faster than asymmetric encryption
  - Used by IPSec to encrypt actual message data

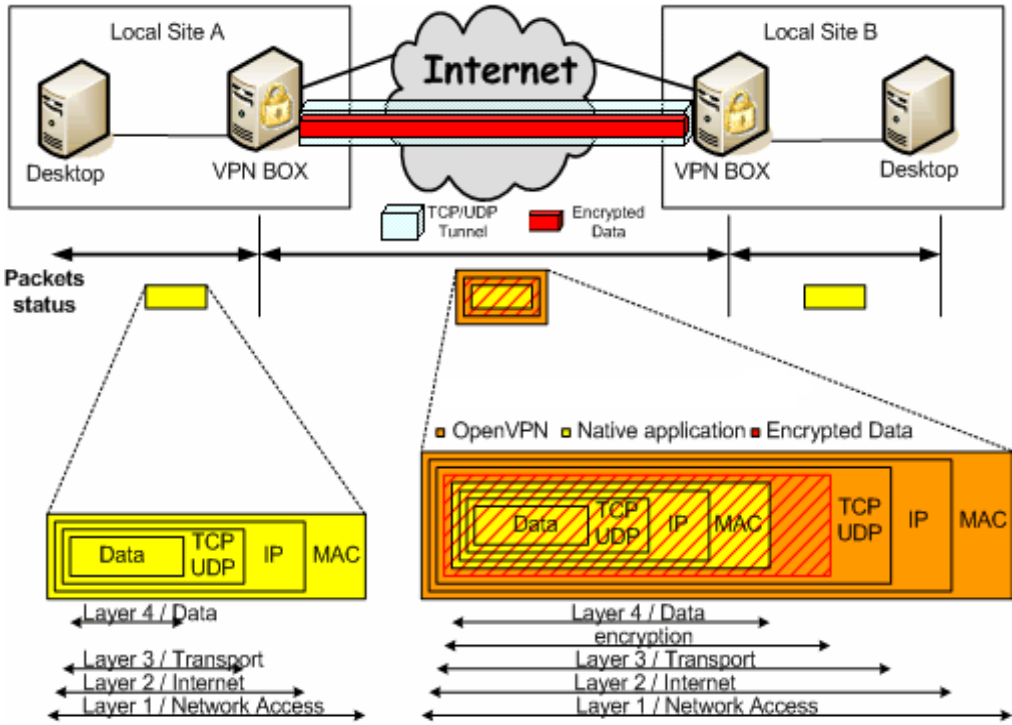


- **Asymetric Encryption: RSA, DSA, SHA-1, MD5**
  - Different keys used to encrypt and decrypt message (public & private)
  - Provides non-repudation of message or message integrity



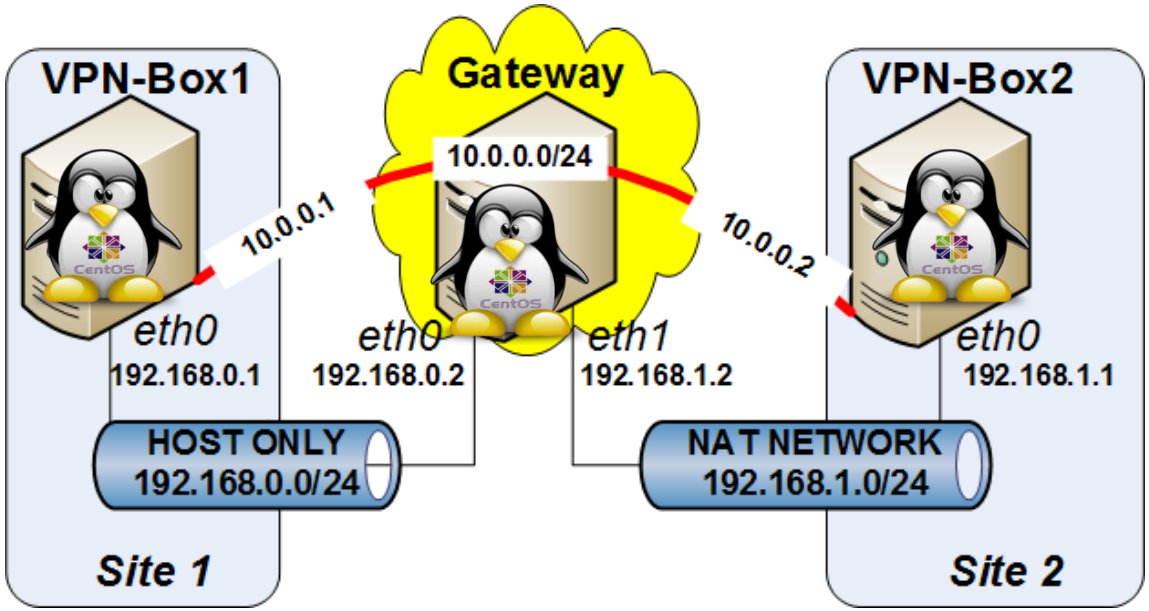
OpenVPN

- Open source tool used to build site-to-site and remote-access VPN
- Easy to install and configure
- Support many platform (Linux, Windows, BSD, MacOS X, Solaris)
- Tunnelling via TCP or UDP (default: UDP port 1194)
- Security: Pre-shared (Symmetric) key or SSL (Asymmetric/Symmetric)
- Bridging/Routing: TAP or TUN network driver



Exercise 2: Configuring Site-to-Site VPN

Set up a network topology as below with your Virtual Machines



In this exercise, we will create an VPN tunnel in route mode between two VPN boxes using OpenVPN

Exercise 2: Configuring Site-to-Site VPN (cont')

1. Configure **Gateway** as a NAT Gateway for **VPN-Box1** and **VPN-Box2** to access the other (review Exercise 1)
2. Find and download the latest **openvpn** and **lzo** *rpm* package from ***rpmfind.net*** and install these packages on **VPN-Box1** and **VPN-Box2**

   Hint: `[VPN-Box1]# rpm -ivh lzo-….rpm`
   `[VPN-Box1]# rpm -ivh openvpn-….rpm`
   `[VPN-Box2]# rpm -ivh lzo-….rpm`
   `[VPN-Box2]# rpm -ivh openvpn-….rpm`
3. Create a shared key on **VPN-Box1**

   Hint: `[VPN-Box1]# cd /etc/openvpn`
   `[VPN-Box1]# openvpn --genkey --secret /etc/openpvn/static.key`
4. Create a **openvpn**'s configuration file on **VPN-Box1**

   *(Refer to next slide: **Ref: /etc/openvpn/vpn-box1.net.conf**)*

   Hint: `[VPN-Box1]# vi /etc/openvpn/vpn-box1.net.conf`
   `[VPN-Box1]# mkdir /var/log/openvpn`
   `[VPN-Box1]# chown nobody.nogroup /var/log/openvpn`
5. Create a **openvpn**'s configuration file on **VPN-Box2**

   *(Refer to next slide: **Ref: /etc/openvpn/vpn-box2.net.conf**)*

   Hint: `[VPN-Box2]# vi /etc/openvpn/vpn-box2.net.conf`
   `[VPN-Box2]# mkdir /var/log/openvpn`
   `[VPN-Box2]# chown nobody.nogroup /var/log/openvpn`

Exercise 2: Configuring Site-to-Site VPN (cont')

6. Copy the static.key from **VPN-Box1** to **VPN-Box2** via **scp**
   Hint: `[VPN-Box1]# scp /etc/openvpn/static.key root@VPN-BOX2:/etc/openvpn/`
7. Configure **iptables** on **VPN-Box1** and **VPN-Box2** to allow VPN connection from port 15000 (udp) on **VPN-Box1** to port 1194 (udp) on **VPN-Box2**
   Hint:
   ```
   [VPN-Box1]# iptables -A INPUT -p udp --sport 1194 --dport 15000 -j ACCEPT
   [VPN-Box1]# iptables -A OUTPUT -p udp --sport 15000 --dport 1194 -j ACCEPT
   [VPN-Box2]# iptables -A INPUT -p udp --sport 15000 --dport 1194 -j ACCEPT
   [VPN-Box2]# iptables -A OUTPUT -p udp --sport 1194 --dport 15000 -j ACCEPT
   ```
8. Firing up the connection on both hosts and verify **tun0** device on both hosts are created
   Hint:
   ```
   [VPN-Box1]# /etc/init.d/openvpn start
   [VPN-Box2]# /etc/init.d/openvpn start
   [VPN-Box1]# ifconfig tun0
   [VPN-Box2]# ifconfig tun0
   ```
9. Verify that the VPN tunnel is working by pinging from **VPN-Box1** to **VPN-Box2** and vice-versa

Ref: /etc/openvpn/vpn-box1.net.conf

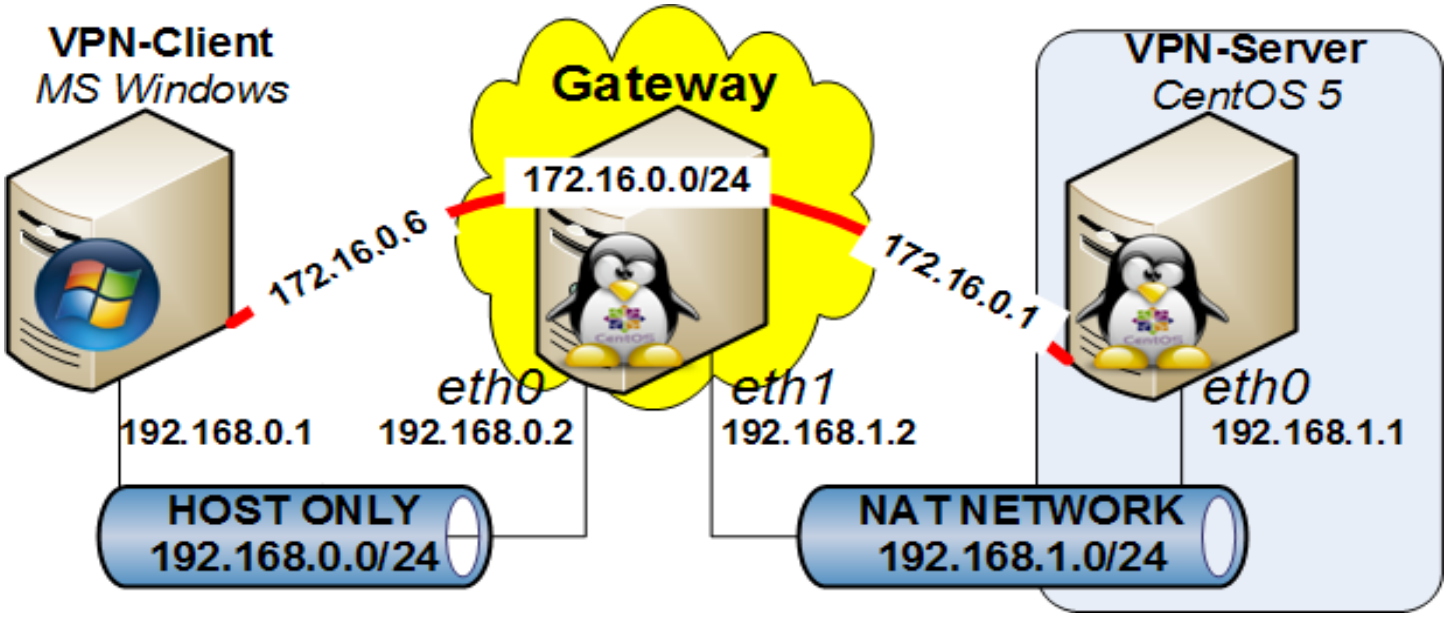
```
dev tun0
remote VPN-Box2
ifconfig 10.0.0.1 10.0.0.2
route 192.168.1.0 255.255.255.0
secret /etc/openvpn/static.key
daemon

lport 15000
rport 1194

user nobody
group nogroup
persist-key
persist-tun

status /var/log/openvpn/rA.example.net-status.log
log-append /var/log/openvpn/rA.example.net.log

ping-restart 60
ping 20
```

Ref: /etc/openvpn/vpn-box2.net.conf

```
dev tun0
remote VPN-Box1
ifconfig 10.0.0.2 10.0.0.1
route 192.168.0.0 255.255.255.0
secret /etc/openvpn/static.key
daemon

lport 1194
rport 15000

user nobody
group nogroup
persist-key
persist-tun

status /var/log/openvpn/rA.example.net-status.log
log-append /var/log/openvpn/rA.example.net.log

ping-restart 60
ping 20
```

Exercise 3: Configuring Remote-Access VPN

Set up a network topology as below with your Virtual Machines



In this exercise, we will configure a VPN Server on CentOS and a VPN Client on Windows machine for VPN Remote Access

Exercise 3: Configuring Remote-Access VPN (cont')

1. Configure **Gateway** as a NAT Gateway for **VPN-Client** and **VPN-Server** to access the other (review Exercise 1)
2. Find and download the latest **openvpn** and **lzo** *rpm* package from ***rpmfind.net*** and install these packages on **VPN-Server**
   Hint: `[VPN-Server]# rpm -ivh lzo-….rpm`
   `[VPN-Server]# rpm -ivh openvpn-….rpm`
3. Copy the script templates used for generating certification to ***/etc/openvpn/***
   Hint: `[VPN-Server]# cp -R /usr/share/doc/openvpn-2.0.9/easy-rsa/ /etc/openvpn/`
4. Make these scripts executable with **`chmod`** and initialize the PKI
   Hint: `[VPN-Server]# cd /etc/openvpn/easy-rsa/2.0/`
   `[VPN-Server]# chmod a+x *`
   `[VPN-Server]# . ./vars`
   `[VPN-Server]# ./clean-all`
   `[VPN-Server]# ./build-ca`
5. Generate a certificate and private key for the OpenVPN Server
   Hint: `[VPN-Server]# ./build-key-server server`
6. Generate OpenVPN Client certificate
   Hint: `[VPN-Server]# ./build-key client`
7. Generate Diffie-Hellman parameters for the OpenVPN Server
   Hint: `[VPN-Server]# ./build-dh`

Exercise 3: Configuring Remote-Access VPN (cont')

8. Copy OpenVPN Server configuration template to ***/etc/openvpn***
   Hint: `[VPN-Server]# cp /usr/share/doc/openvpn-2.1.4/sample-config-files/server.conf /etc/openvpn/openvpn.conf`
9. Edit the OpenVPN Server configuration file to reflex your topology
   *(Refer to the next slide: **Ref: /etc/openvpn/openvpn.conf**)*
10. Copy the generated server certificates and keys to ***/etc/openvpn/***
    Hint: `[VPN-Server]# cd /etc/openvpn/easy-rsa/2.0/keys/`
    `[VPN-Server]# cp ./{ca.crt,ca.key,server.crt,server.key,dh1024.pem} /etc/openvpn`
11. Start OpenVPN Server
    Hint: `[VPN-Server]# service openvpn start`
    *If the service can not start, check the syslog for errors, normally due to mistakes in **openvpn.conf***
12. Verify that the OpenVPN service is running
    Hint: `[VPN-Server]# ifconfig tun0`
    `[VPN-Server]# netstat -an | grep 2000 #`*we config **openvpn** to use port 2000 in this lab*
13. **[On VPN Client]** Download and install OpenVPN for Windows from *http://openvpn.net/index.php/open-source/downloads.html*
14. **[On VPN Client]** Copy ***ca.crt***, ***client.crt***, ***client.key*** from ***/etc/openvpn/easy-rsa/2.0/keys/*** on VPN Server to ***C:\Program Files\OpenVPN\config*** on VPN Client
15. **[On VPN Client]** Copy ***C:\Program Files\OpenVPN\sample-config\client.ovpn*** to ***C:\Program Files\OpenVPN\config*** and edit this file to reflex your topology
    *(Refer to the next slide: **Ref: C:\Program Files\OpenVPN\config\client.ovpn**)*
16. **[On VPN Client]** Run the **OpenVPN GUI**, right click on its taskbar icon and select **Connect** initialize the connection to VPN Server.
17. Verify the VPN tunnel between Client and Server with **ping**

Ref: /etc/openvpn/openvpn.conf

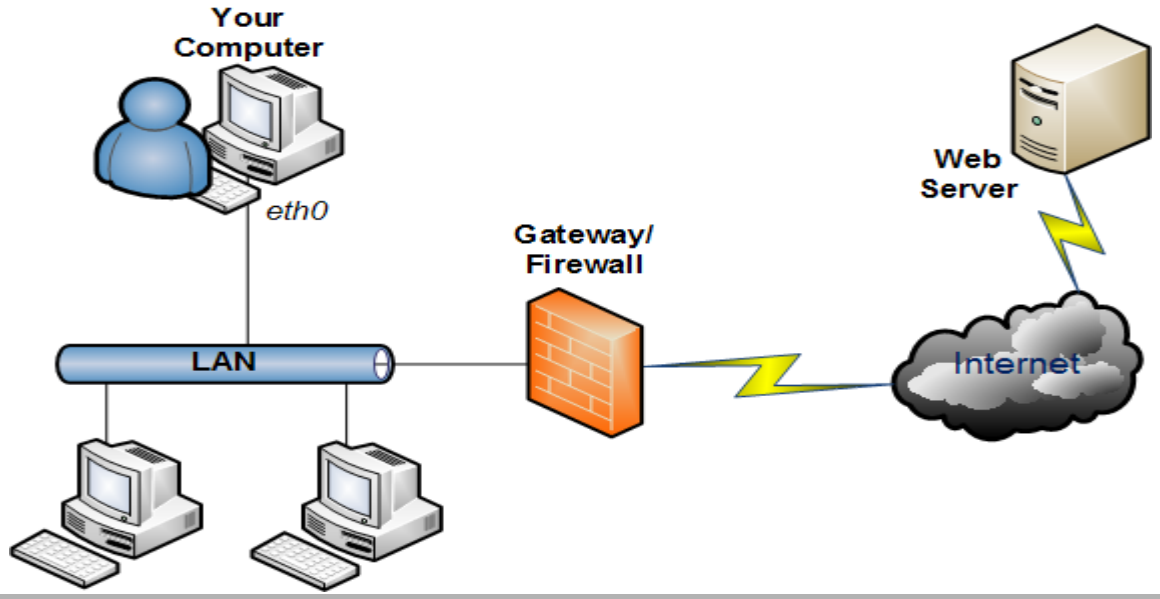
```
port 2000                    #Port on which OpenVPN server listen on
proto tcp                    #Use tcp for VPN Connection
dev tun                      #Use routed IP tunnel (tun driver)
ca ca.crt                    #Root Certificate file
cert server.crt              #Server certificate file
key server.key               #Server private key
dh dh1024.pem                #Diffie-Hellman parameter file
server 172.16.0.0 255.255.255.0      #Configure the server mode and supply a VPN subnet
ifconfig-pool-persist ipp.txt    #Maintain a record of client<->IP Address in this file
push "route 192.168.1.0 255.255.255.0" #Push routes to the client to reach other subnet
push "dhcp-option DNS 8.8.8.8"       #Push DNS information to the client
client-to-client             #Allow different client to be able to see each other
duplicate-cn                 #Multiple clients might connect with the same certificate/key files
keepalive 10 120             #Ping every 10 seconds, assume the peers is down if no ping recived in 120 seconds
comp-lzo                     #Enable compression on the VPN link
user nobody                  #Reduce the OpenVPN daemon's priviledges after initialization
group nobody                 #Reduce the OpenVPN daemon's priviledges after initialization
persist-key                  #Avoid accessing certain resources on restart
persist-tun                  #Avoid accessing certain resources on restart
status openvpn-status.log    #Output a short status file showing current connection
verb 3                       #Set the appropriate level of log file verbosity to 3
```

Ref: C:\Program Files\OpenVPN\config\client.ovpn

```
client                      #Specify that we are client
dev tun                     #Use the same setting as using on the server
proto tcp                   #Use the same setting as using on the server
remote 192.168.1.1 2000     #The hostname/IP and port of the server
persist-key                 #Preserve some state across restarts
persist-tun                 #Preserve some state across restarts
ca ca.crt                   #Root Certificate file
cert client1.crt            #Client Certificate file
key client1.key             #Client Private key
ns-cert-type server #Verify server certificate by checking the nsCertType field
comp-lzo                    #Enable compression on the VPN link as on the server
verb 3                      #Log file verbosity
```

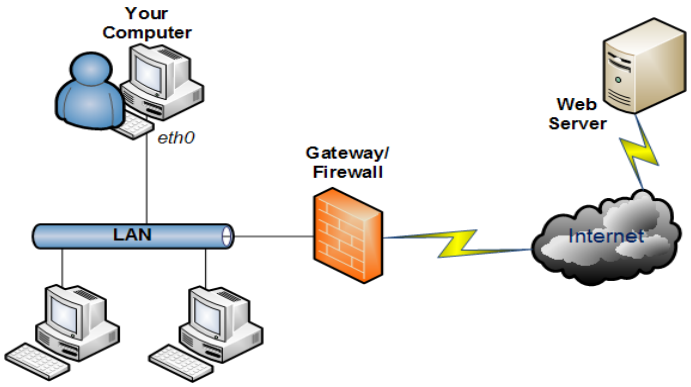
Troubleshooting Networking Issue

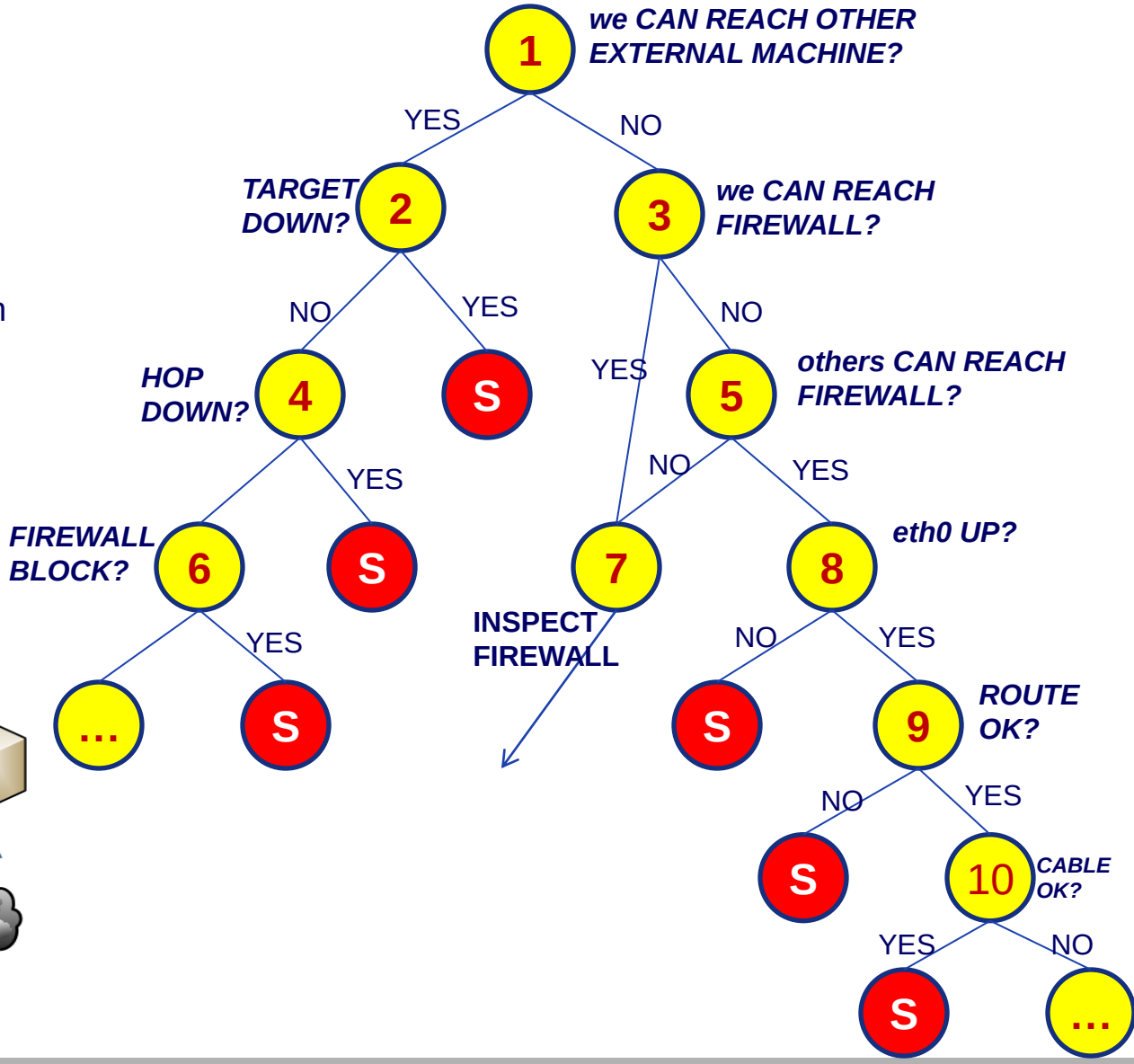
- **Example of Problem:**
  Your machine is connected to the Internet. Suddenly you can't view a web page anymore
- **First thing first:** Draw out your connection, list the components that are involved



Troubleshooting Networking Issue (cont')

1. **ping** some IP Address on the internet
   Note: *Firewall could be blocking ICMP echo request and replies*
2. **ping** the target from another network
3. **ping** the firewall
4. **traceroute** to find out hops between you and the target
5. **ping** the firewall from another machine in the same LAN
6. Does the firewall block your traffic?
7. Inspect the firewall rules, test the interface
8. Is our eth0 up? Test by issuing **ifconfig eth0**
9. Verify route table by **route -n**





Thank You !

BACKUP SLIDES