SAPIENZA
UNIVERSITÀ DI ROMA

Practical Network Defense

Master's degree in Cybersecurity 2024-25

OpenVPN activity

Angelo Spognardi
spognardi@di.uniroma1.it
Dipartimento di Informatica
Sapienza Università di Roma

Aim of the lab

1) Highlight the tunneling concept
2) Use openvpn to make the boundary to be the VPN tunnel end-point
   - The kali/host will have the access to the **internal** subnet passing through an encrypted channel
   - Two scenarios
     - Static key configuration
     - Dynamic key configuration
       - Using certificates

To do the activities

- We will use Kathará (formerly known as netkit)
  - A container-based framework for experimenting computer networking: http://www.kathara.org/
- A virtual machine is made ready for you
  - https://drive.google.com/file/d/1W6JQzWVyH5_LKLD20R6XH1ugPDP5LWP5/view?usp=sharing
- For not-Cybersecurity students, please have a look at the Network Infrastructure Lab material
  - http://stud.netgroup.uniroma2.it/~marcos/network_infrastructures/current/cyber/
    - Instructions are for netkit, we will use kathara

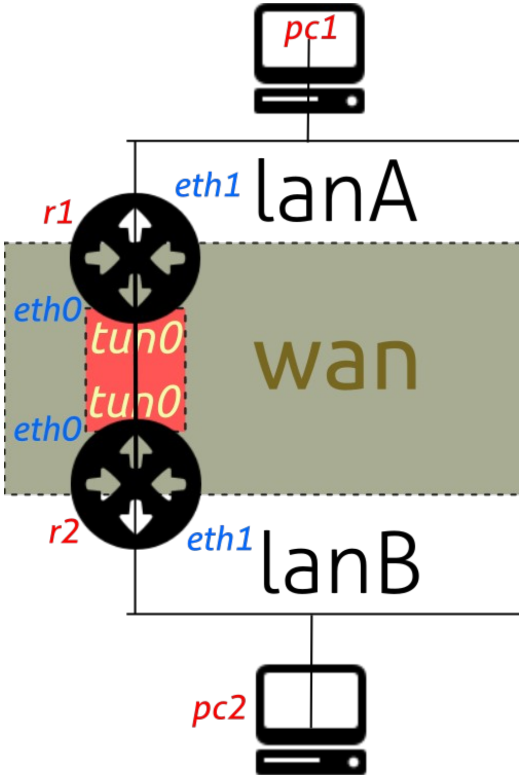
The kathara VM

- It <u>should</u> work in both Virtualbox and VMware
- It <u>should</u> work in Linux, Windows and MacOS
- There are some alias (shortcuts) prepared for you
  - Check with **`alias`**
- All the exercises can be found in the git repository:
  - https://github.com/vitome/pnd-labs.git
- You can move in the directory and run lstart
  - **NOTE**: launch docker first or the first lstart attempt can (…will…) fail

Lab activity: ex1

pnd-labs/lab5/ex1

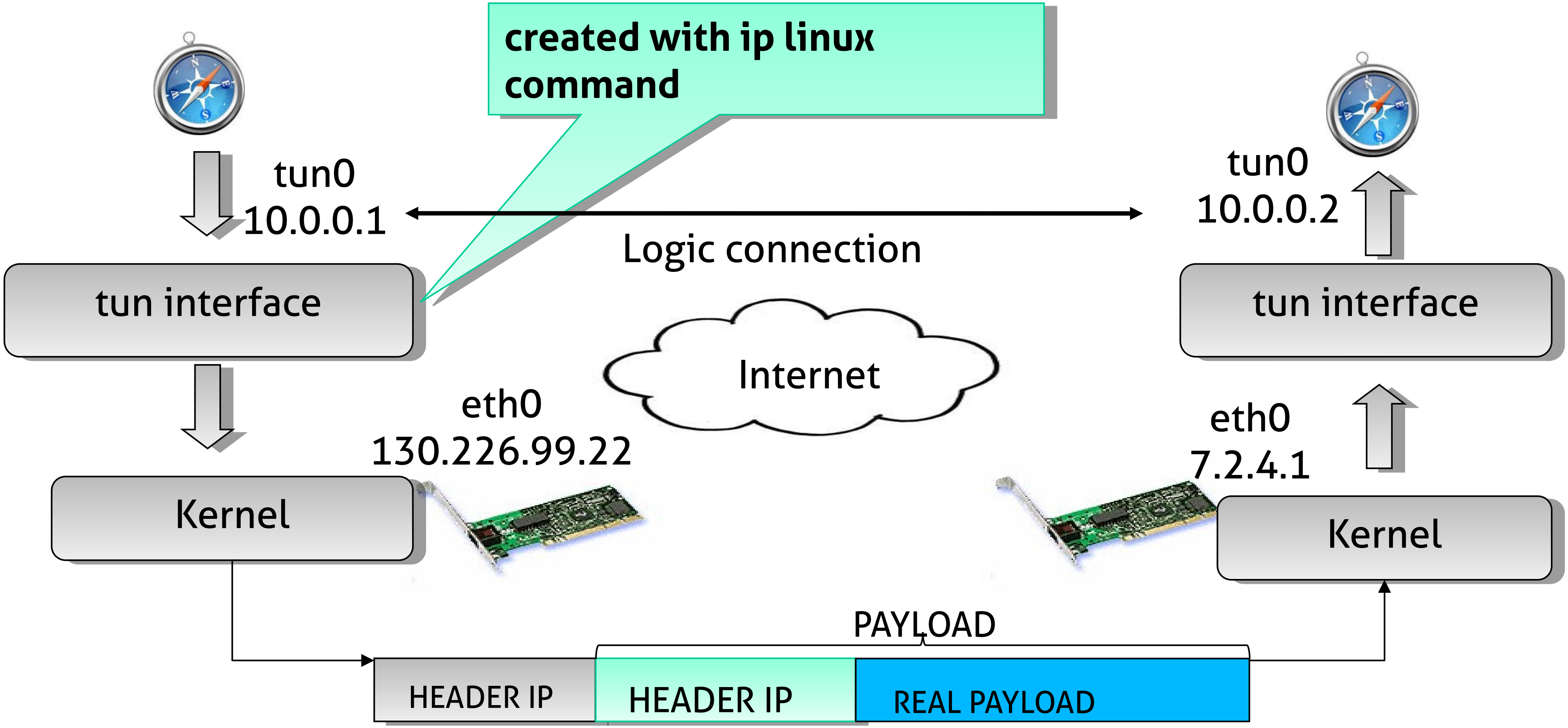
- You have to setup the IPv4 addressing
  - Follow README instructions



Fundamentals: simple tunneling

- Universal tun/tap drive
  - Creates a virtual interface that encapsulates network traffic
- Any application can use that interface without any need to change its code
- Usually identified with names tun* or tap*
- tun* encapsulate IP layer
- tap* encapsulate Ethernet layer
- Reference:
  - http://man7.org/linux/man-pages/man8/ip-tunnel.8.html
  - https://developers.redhat.com/blog/2019/05/17/an-introduction-to-linux-virtual-interfaces-tunnels#ipip_tunnel

Universal tun driver (L3)



Create a tunnel (r1-r2)

- Let's check our local interfaces

```
ip link
```

- Create a new tun virtual interface (use root user)

```
ip tunnel add tun0 mode ipip remote <ipaddressR> local <ipaddressL>
```

  - ipaddressR is the IP address of the remote machine
  - ipaddressL is the IP address of our local machine
    - alternatively use the `ip link add name tun0 type ipip …` command

- Let's check again our local interfaces

```
ip link
```

- Let's activate the new interface

```
ip addr add 10.0.0.1/30 dev tun0
ip link set tun0 up
```

  - the IP address of the remote machine **MUST be different!!**

Analyze the traffic

- Open wireshark to sniff the traffic
  - Use tcpdump to save the traffic from different observation points
- Generate some traffic in the tunnel
  - `ping 10.0.0.2`
- What do you notice?
  - See the difference between tun0 and the eth0
- Can you see the basic principle of a VPN?

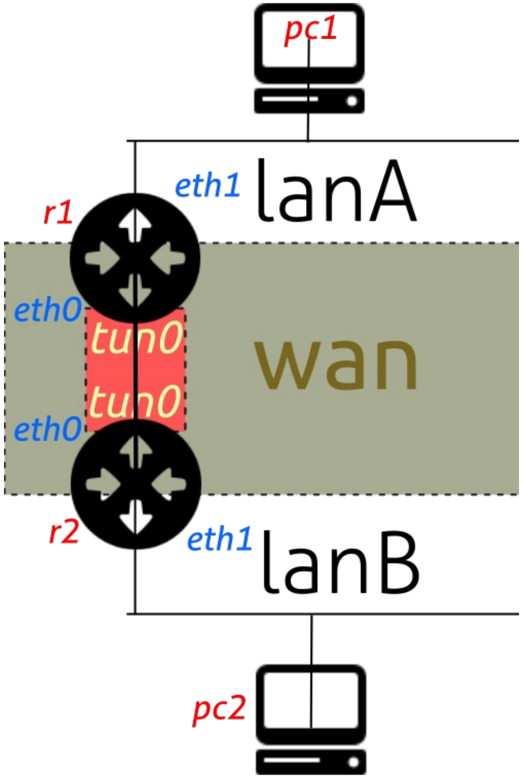
Use the tunnel to connect lanA and lanB

- You have to setup the routes
- Who will r1 forward the packets for lanB?
  - In order to use the tunnel?
- Properly set up the routing tables
- Check what happens when pc1 tries to reach pc2
  - Use wireshark in r1 or r2
  - You can join the wan network

Lab activity: ex2

pnd-labs/lab5/ex2

- Same topology than ex1
- Must run with the option –**`privileged`**
  - Use **`lstart.sh`** script
- You have to setup the IPv4 addressing
  - Follow README instructions
- Check that openvpn is installed in both r1 and r2



OpenVPN

- Open-source software to realize VPN, namely encrypted tunnels
- Usually uses UDP with one single port
  - Can also use TCP
- Can be used also through firewalls or NAT
- OpenSSL based
- Multiple modes
  - Static: symmetric shared key
  - Dynamic: Public Key Infrastructure

https://community.openvpn.net/openvpn/wiki/SecurityOverview

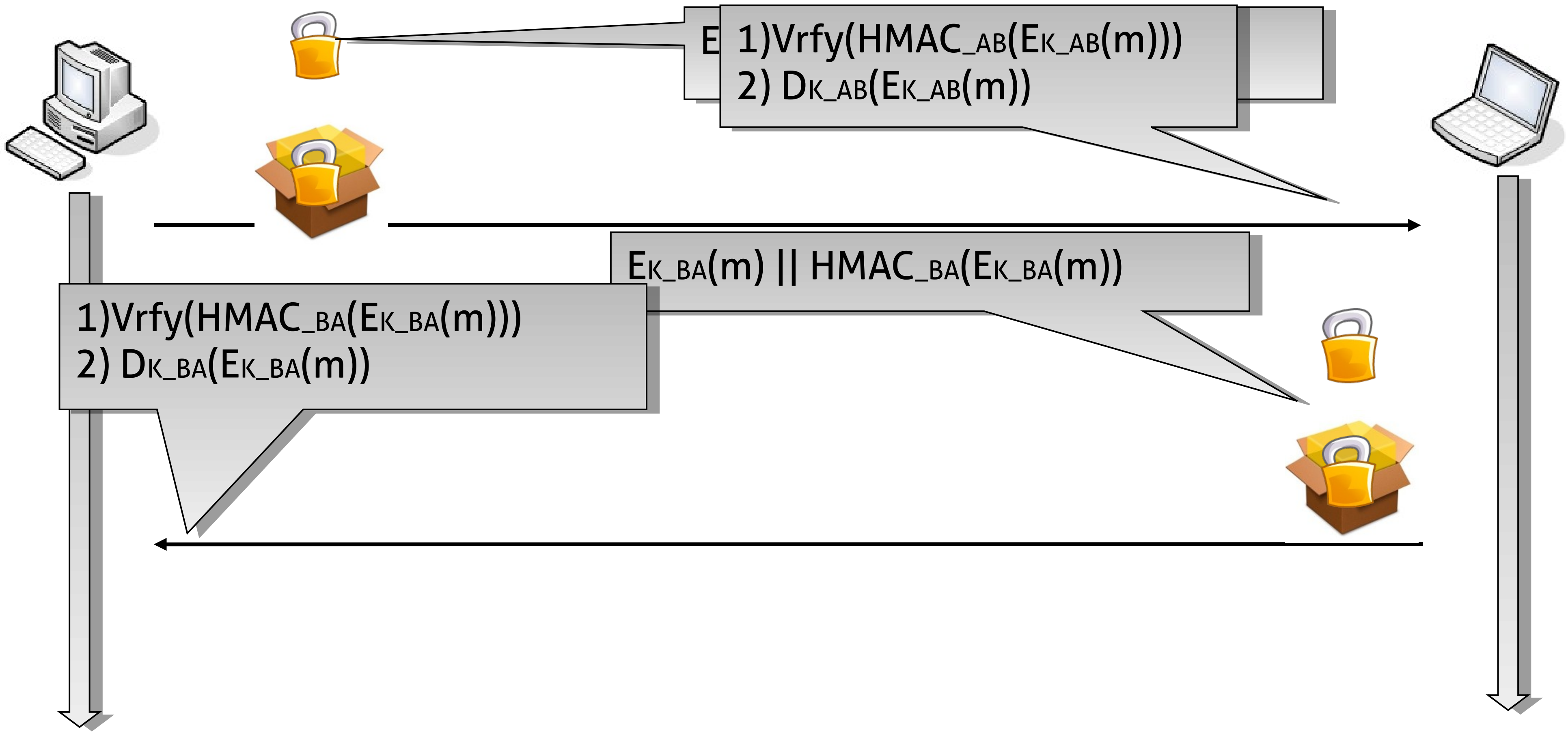
OpenVPN static key

- The endpoints share a key generated with openvpn command
- Very easy to configure
- No CA or certificates
  - No TLS!
- NOTE: requires a secure channel to exchange the keys
- The key never changes: no forward secrecy

OpenVPN static key: keys

- Uses 4 independent keys:
  - K_AB (to encrypt A -> B)
  - HMAC_AB (to authenticate A -> B)
  - K_BA (to encrypt B -> A)
  - HMAC_BA (to authenticate B -> A)
- This is required to reduce the risks of Replay and DoS attacks

OpenVPN static key: traffic exchange



Standard crypto

Blowfish
Block cipher: **64-bit block**
Key length: **32 bits to 448 bits**
Designed by **Bruce Schneier**
Much **faster** than DES and IDEA
Unpatented and royalty-free
No license required
Free **source code available**

- OpenVPN standard
- 128 bit keys
- CBC (Cipher-block Chaining)
- You can choose the default with the cipher option in the configuration file
- Many others available
  - `openvpn --show-ciphers`

SHA-1
Published: **1995**
Designed by: **NSA**
Output length: **160 bit**

- OpenVPN standard
- Uses a different key than the encryption one

OpenVPN static key: key generation

- Generate the shared key on one side of the tunnel (say r1)
  - `openvpn --genkey secret secret.key`
- Exchange the secret.key file with scp
- OR, if you don't send it with scp:
  - Encrypt the key (because we'll use an insecure channel)
    - `openssl enc -aes-128-cbc -e -a -in secret.key -out secret.key.enc`
  - Exchange the shared key
    - Prepare to receive the shared key on the other side of the tunnel (say r2):
      - `r2# nc -l -p 9000 > secret.key.enc`
    - Send the shared key
      - `r1# nc <r2IPaddress> 9000 < secret.key.enc`
  - Decrypt the key on the other side of the channel
    - `openssl enc -aes-128-cbc -d -a -in secret.key.enc -out secret.key`

OpenVPN static key: file conf

r1

```
port 1194
proto udp
dev tun
secret secret.key
cipher AES-256-CBC
ifconfig 10.10.10.1 10.10.10.2
```

r2

```
remote <r1IPaddress>
port 1194
proto udp
dev tun
secret secret.key
cipher AES-256-CBC
ifconfig 10.10.10.2 10.10.10.1
```

- r1 plays the role of the passive actor → waits for connections
- Create a new file r1.conf/r2.conf and use the above conf
  - You can check the path `/usr/share/doc/openvpn/examples/`

OpenVPN static key: file conf

- Start openvpn
  - `r1# openvpn --config r1.conf`
  - `r2# openvpn --config r2.conf`
- Check the connectivity on the new interfaces and analyze the traffic with wireshark
- To give visibility of another subnet (i.e., the **lanA** network), you can use the **`route`** option
  - `route <network> <netmask> <host>`
    - Network and netmask in dotted decimal
    - Host is the next-hop for reaching the network

OpenVPN dynamic key

- Uses SSL/TLS and certificates for authentication and key exchange
- Certificates for both endpoints
- If the certificates are valid
  - HMAC and encryption keys are dynamically generated with OpenSSL
  - This assures Forward Secrecy
- Both parties contribute to key generation

OpenVPN dynamic key: cert generation

- We should have a certification authority issuing the certs
  - This should be done using the → **easy-rsa scripts**, that can be found in the `/usr/share/easy-rsa` directory
    - Alternatively, we can use the ones provided by openvpn for test purposes in `/usr/share/doc/openvpn/examples/sample-keys`
- Needed ingredients/files
  - {client,server}.crt : CA signed public key
  - {client,server}.key: CA signed private key
    - Certificates are issued after signing the requests (client!=server)
  - dh.pem: Diffie-Hellman key exchange parameters
  - ta.key: for TLS HMAC authentication (optional)

OpenVPN dynamic key: file conf

- Start from the sample configuration files in the directory

  `/usr/share/doc/openvpn/examples/sample-config-files/`
- Server (VPN gateway):
  - `.../sample-config-files/server.conf`
- client:
  - `.../sample-config-files/client.conf`

OpenVPN dynamic key: run

- Start openvpn
- Server:
  - **openvpn --config server.conf**
- Client:
  - **openvpn --config client.conf**
- Check the connectivity on the new interfaces and analyze the traffic with wireshark

That’s all for today

- **Questions?**
- Resources:
  - https://openvpn.net/community-resources/how-to/
  - https://wiki.wireshark.org/OpenVPN
  - Chapter 24 textbook
  - Virtual private networking, Gilbert Held, Wiley ed.
  - Guide to IPsec VPNs, NIST800-77
  - Guide to SSL VPNs, NIST-SP800-113
  - http://www.tcpipguide.com/free/t_IPSecurityIPSecProtocols.htm