SAPIENZA
UNIVERSITÀ DI ROMA

Practical Network Defense

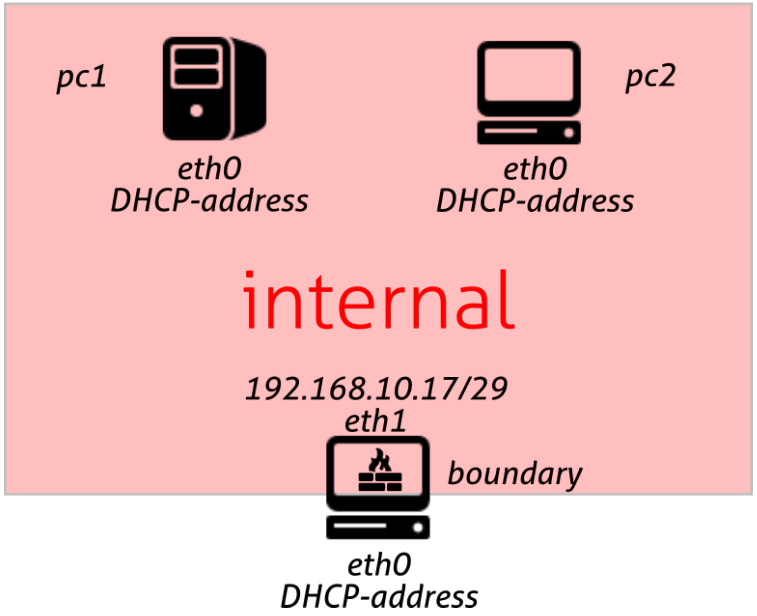
Master's degree in Cybersecurity 2018-19

OpenVPN activity

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

Lab setting

- You can use pnd-lab2-es1 topology
  - Remember to start udhcp



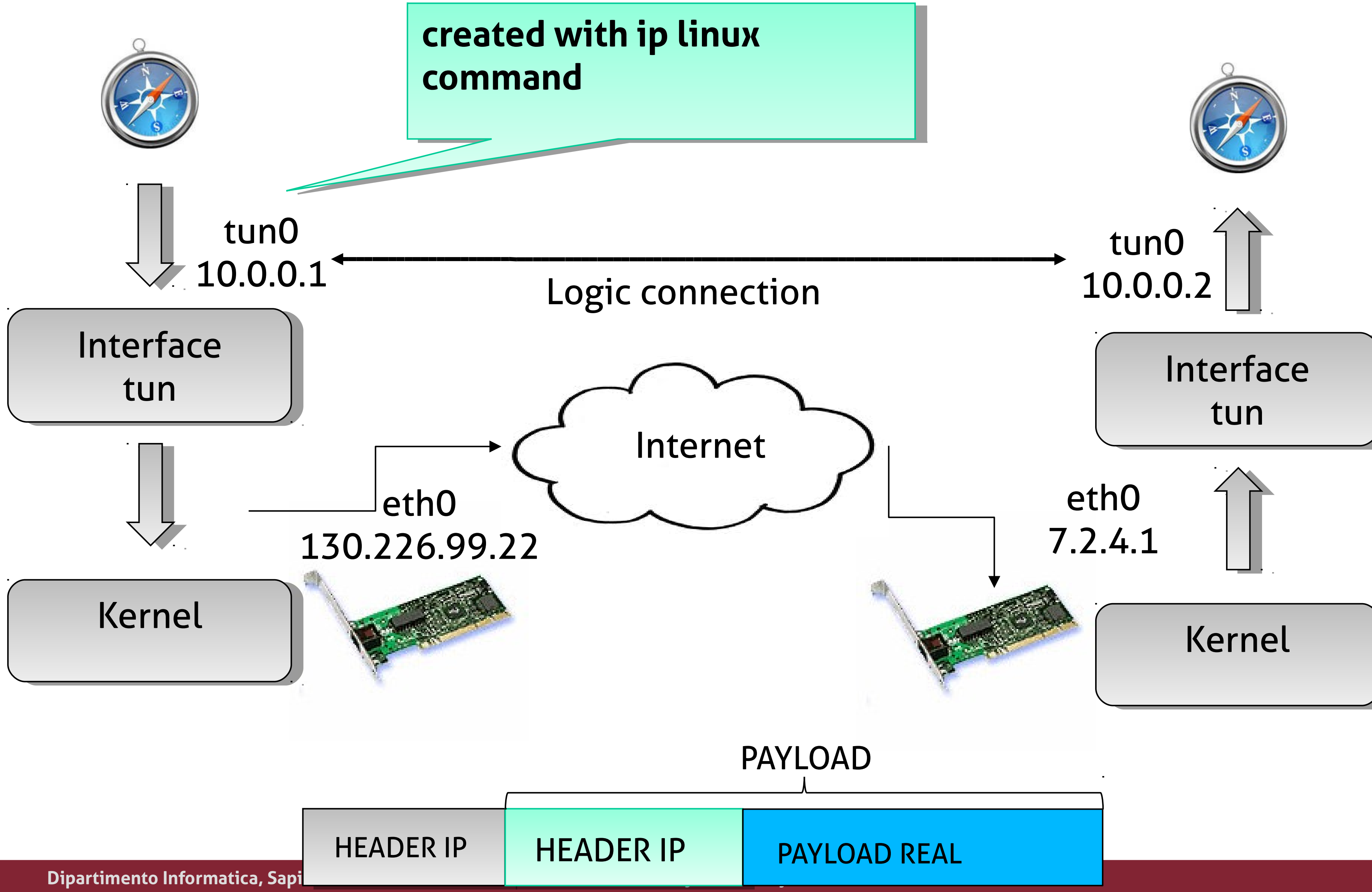
Aim of the lab

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

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

Universal tun driver (L3)



created with ip linux command

tun0
10.0.0.1

Logic connection

tun0
10.0.0.2

Interface
tun

Interface
tun

Internet

eth0
130.226.99.22

eth0
7.2.4.1

Kernel

Kernel

PAYLOAD

HEADER IP | HEADER IP | PAYLOAD REAL

Create a tunnel (host-boundary or pc1-boundary)

- Let's check our local interfaces

```
ifconfig -a
```

- 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

- Let's check again our local interfaces

```
ifconfig -a
```

- Let's activate the new interface

```
ifconfig tun0 up 10.0.0.1/30 mtu 1500
```

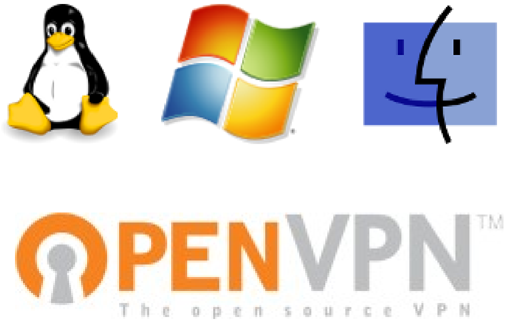
  - the IP address of the remote machine **MUST be different!!**

Analyze the traffic

- Open wireshark to sniff the traffic
  - If pc1: use tcpdump
- 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?

OpenVPN

- `apt install 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



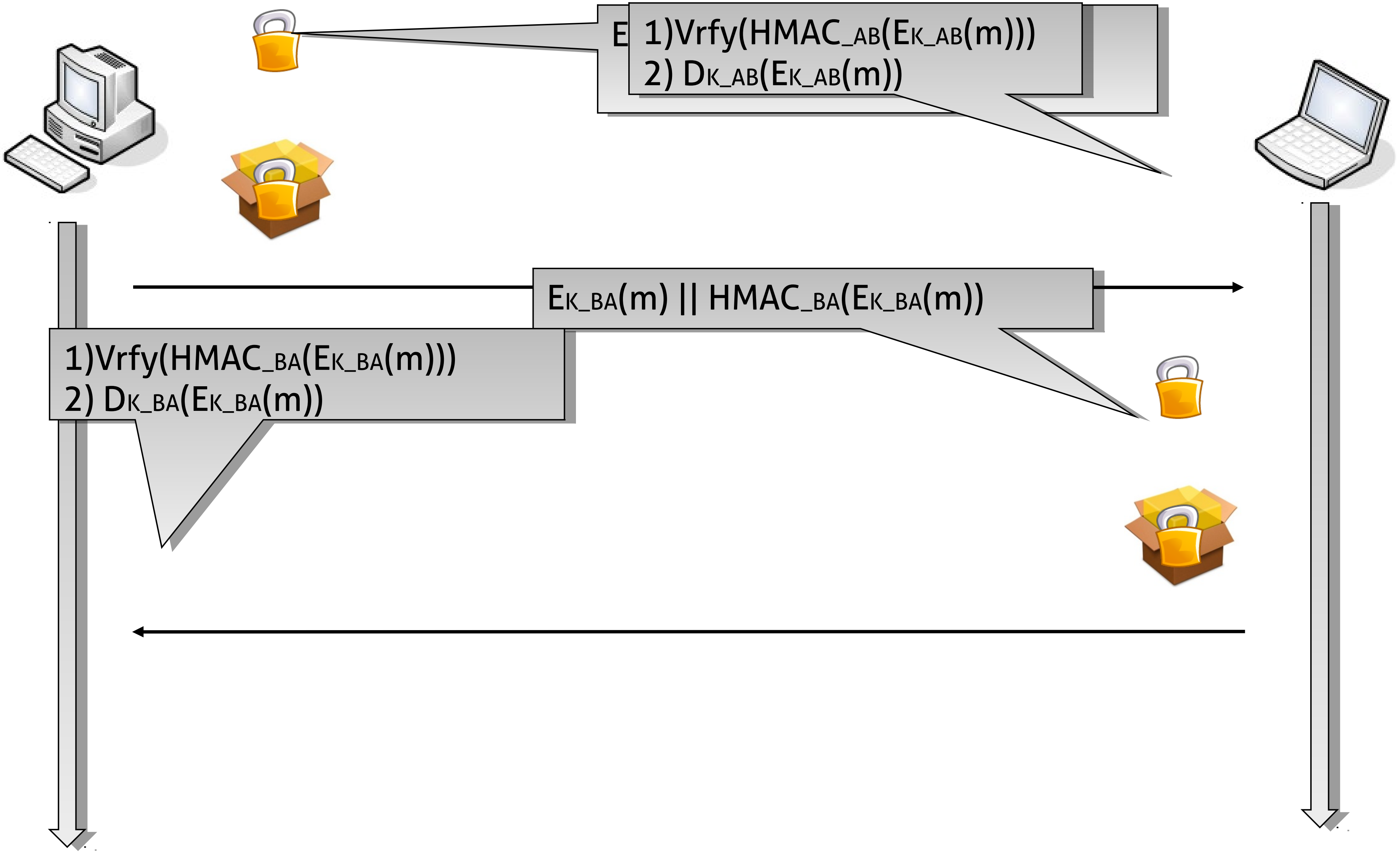
OpenVPN static key

- The endpoints share a key generated with openvpn command
- Very easy to configure
- No CA or certificates
- 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



E

1)Vrfy(HMAC$_{\_AB}$(E$_{K\_AB}$(m)))
2) D$_{K\_AB}$(E$_{K\_AB}$(m))

E$_{K\_BA}$(m) || HMAC$_{\_BA}$(E$_{K\_BA}$(m))

1)Vrfy(HMAC$_{\_BA}$(E$_{K\_BA}$(m)))
2) D$_{K\_BA}$(E$_{K\_BA}$(m))

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 Alice)
  - `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 Bob):
      - **nc -l -p 9000 > secret.key.enc**
    - Send the shared key
      - **nc <BobIPaddress> -p 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

ALICE

```
port 1194
proto udp
dev tun
secret secret.key 1
ifconfig 10.10.10.1 10.10.10.2
```

BOB

```
remote <AliceIPaddress>
port 1194
proto udp
dev tun
secret secret.key 0
ifconfig 10.10.10.2 10.10.10.1
```

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

OpenVPN static key: file conf

- Start openvpn
  - `boundary# openvpn --config alice.conf`
  - `host# openvpn --config bob.conf`
- Check the connectivity on the new interfaces and analyze the traffic with wireshark
- To give visibility to bob of the alice subnet (namely the **internal** network), add to bob.conf
  - `route 192.168.10.16 255.255.255.248`

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 dynamicly 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 in kali with the easy-rsa scripts
- BUT, we'll use the ones provided by openvpn for test purposes, instead…
- We should have
  - {client,server}.crt : CA signed public key
  - {client,server}.key: CA signed private key
  - dh2048.pem: Diffie-Hellman key exchange parameters

OpenVPN dynamic key: file conf

- Use the sample-conf
- boundary:
  - sample-conf/server.conf
  - sample-key with the needed crypto material
- client:
  - sample-conf/client.conf
  - sample-key with the needed crypto material
  - change the "**remote**" option with the ipaddress of the server

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?**
- See you tomorrow
- Resources:
  - https://openvpn.net/community-resources/how-to/
  - 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