PALEA

Know the unknown

NETWORK / DATAPROTECTION

- Deny hackers entry
- Keep sensitive data inside
- Not through front door (known corporate gateway)
- Not through back door (the unknown)

WHY SCAN?

- A hacker break in can lead to several forms of damage.
  - Stealing customer records, financial forecasts
  - Undisclosed business plans and other property
- Your systems might be used in an attack
- Legal obligations to report a hacker break-in
- Compliancy regulations or consequences

BACKDOORS TO THE INTERNET

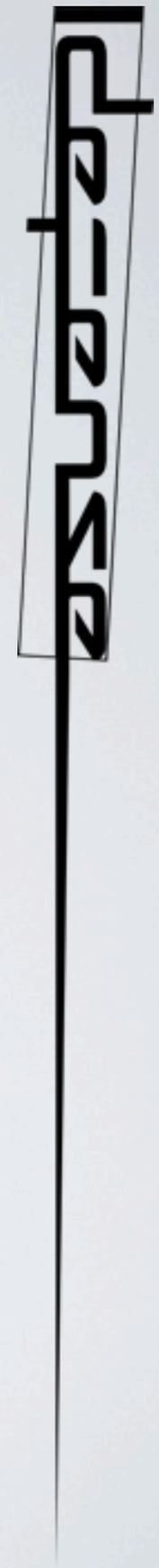
- Unauthorized routes to the internet might enable compromised machines to talk to their hacker (botnets)

- Stealing computer resources

- Unauthorized routes to the internet might be used as covert channel for stealing data

- Unauthorized routes might be used as secret channel to enter the company

HOW TO FIND THESE CONNECTIONS



- We might do our very best to find external connections from the outsite. (a needle in a haystack)

- We better ask all the machines if they are a router or know a router to outside.

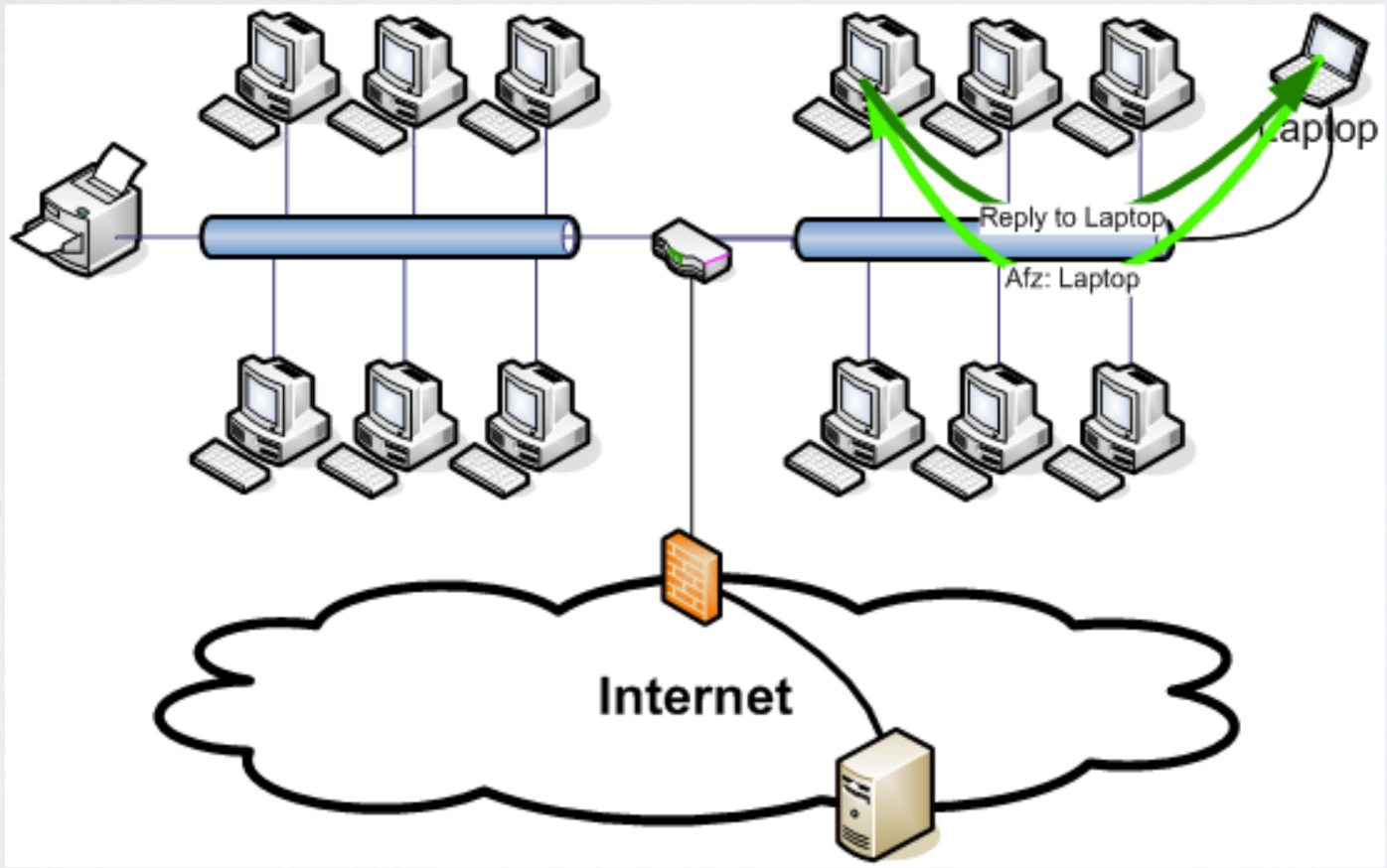
- Let the sum of the machines do the work

HOW TO FIND THESE CONNECTIONS

- We simply need a system to try and deliver a package to the internet, something we can catch, identify and trace back to the specific system.

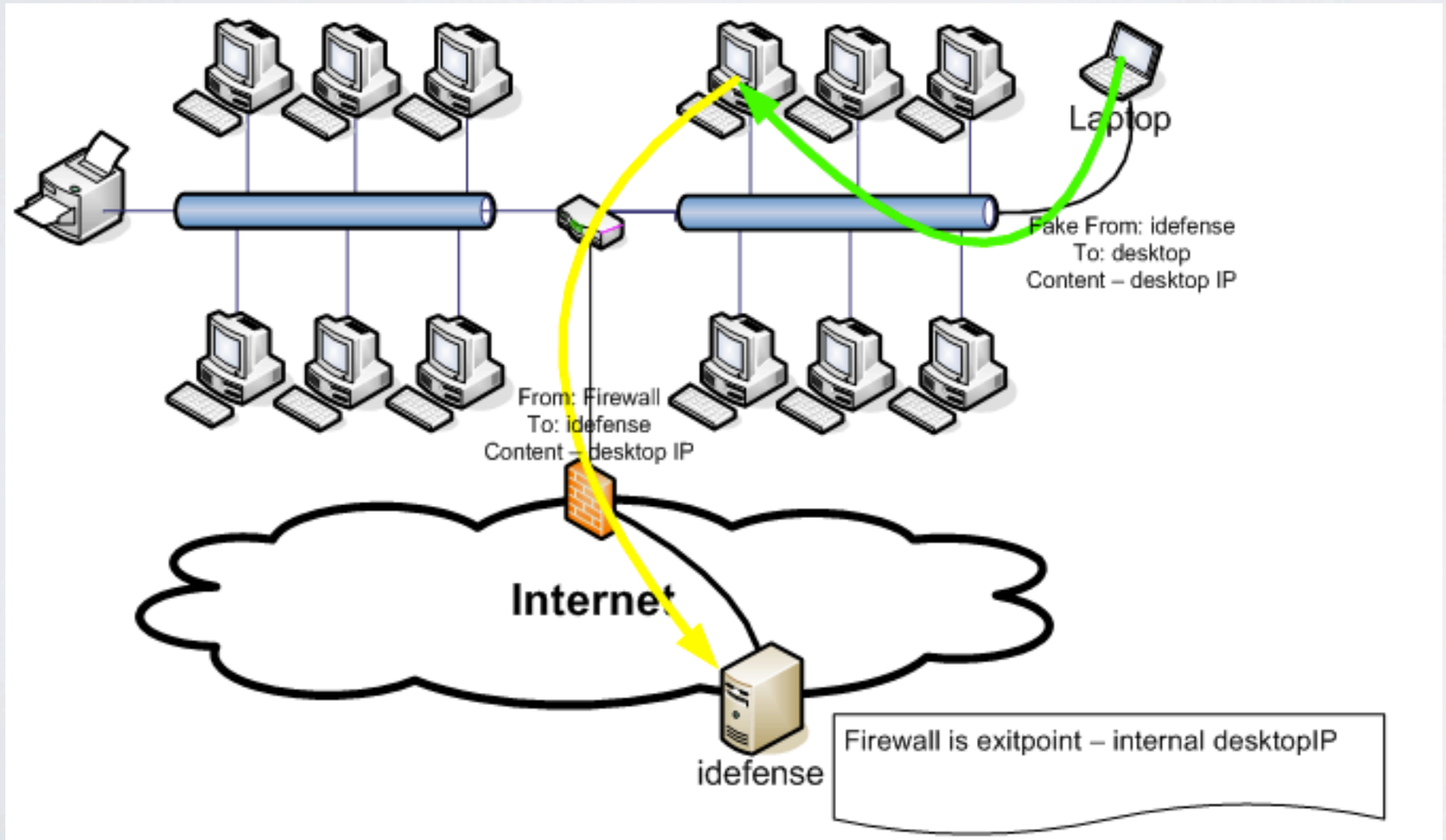
INTERNET INTERFACE AS WE KNOW IT

- Laptop sends a ‘HELLO’ to a system, the system replies



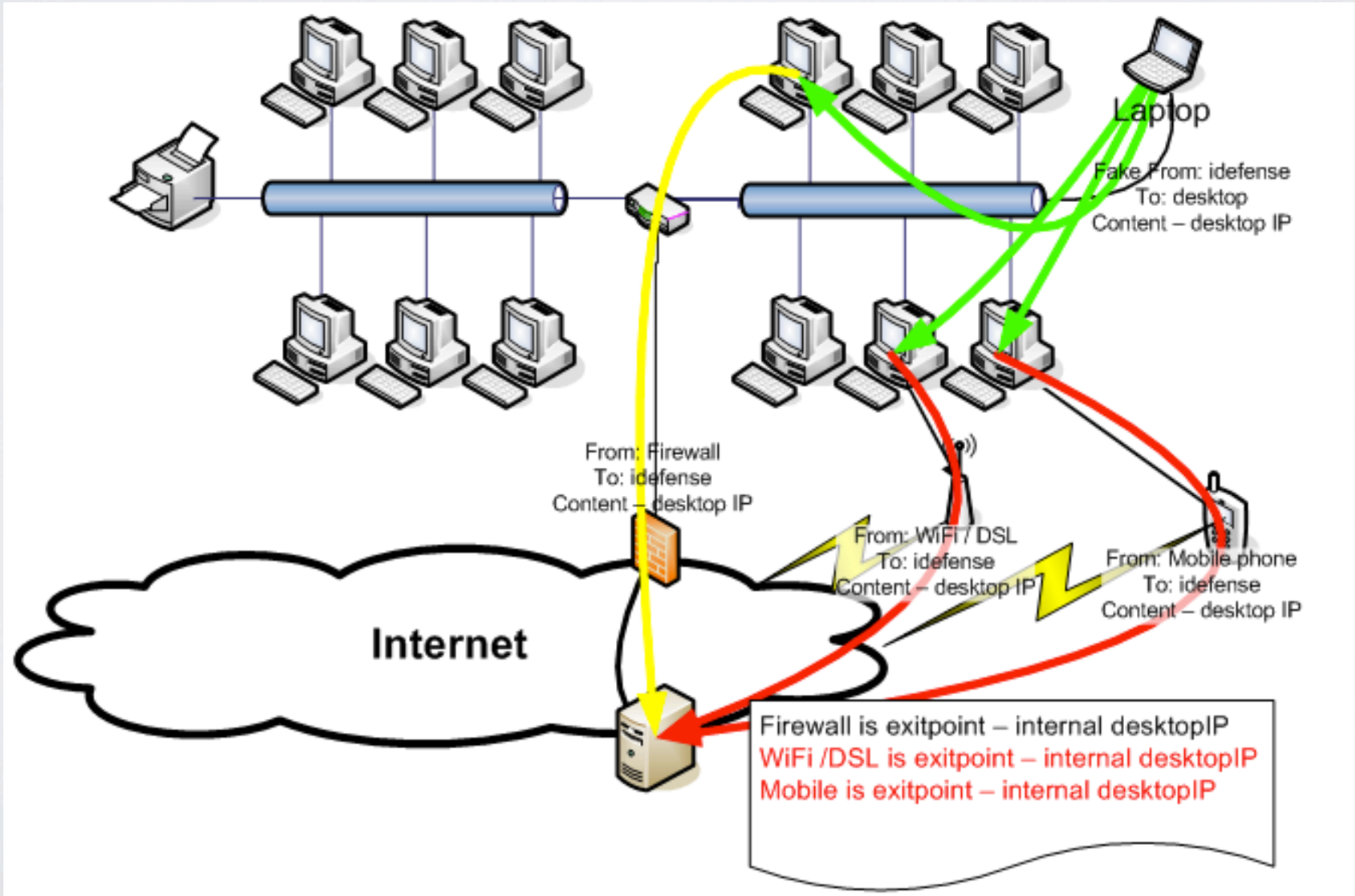
INTERNET INTERFACE AS WE KNOW IT

- Laptop sends a 'HELLO' to with a fake FROM address to a system, the reply gets sent to a server on the internet. Firewall MIGHT stop this.



INTERNET INTERFACES AS THEY CAN BE

- Laptop sends packets to several systems with fake FROM address, some desktops might know alternatives ways to the outside.

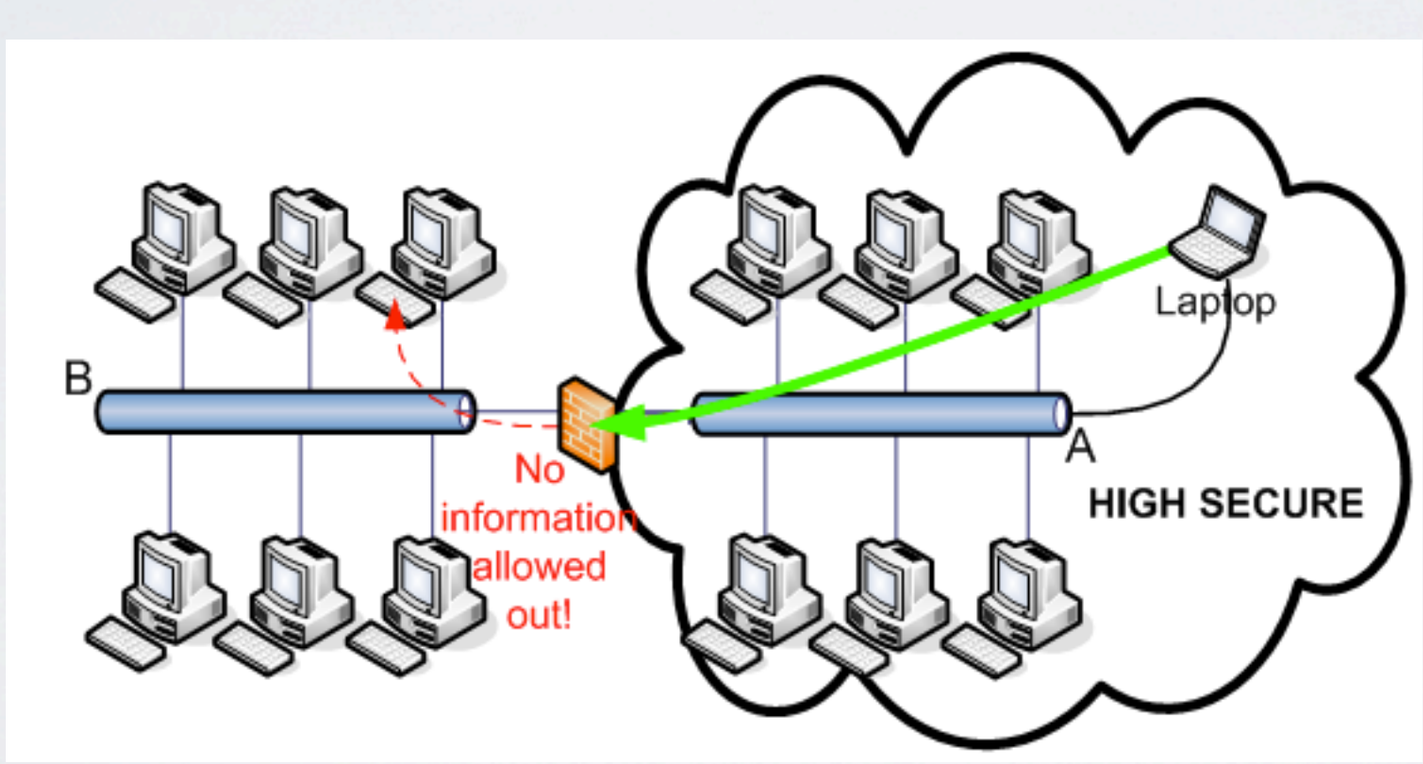


HIGHLY SECURE ENVIRONMENT

- Information in highly secure environments is not supposed to leak out to the rest of the network

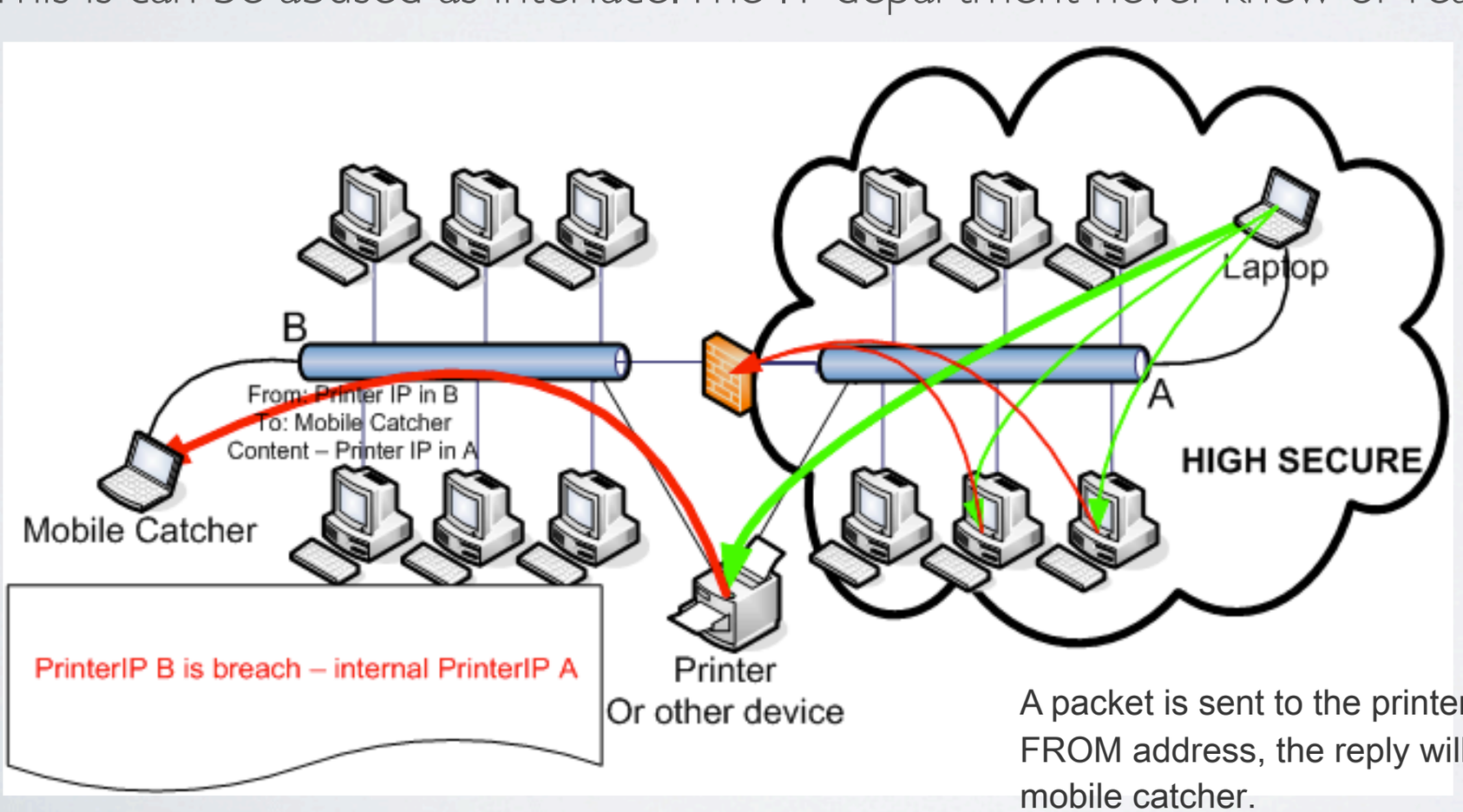
INTERNAL INTERFACING AS WE KNOW IT

- Highly secure network is not allowed to leak data, a firewall is in place. This is the situation as documented



INTERNAL INTERFACING AS THEY MIGHT BE

- A printer is used to print to from the high secure environment but also from a user network.
- This is can be abused as interface. The IT department never knew or realized.



A packet is sent to the printer with a fake FROM address, the reply will go to our mobile catcher.

BEING IN CONTROL OF THE ENVIRONMENT

- These scans should be repeated 24/7 several times a day for full coverage
  - GSM / Modem connections aren't permanent, you need it to be active to find it
  - New interfaces might be created
  - IT is changing all the time
- Security scans should be repeated as new vulnerabilities might develop

TIME TO GO TECH!

- Spoof IP to trick every system in an attempt to speak to the internet

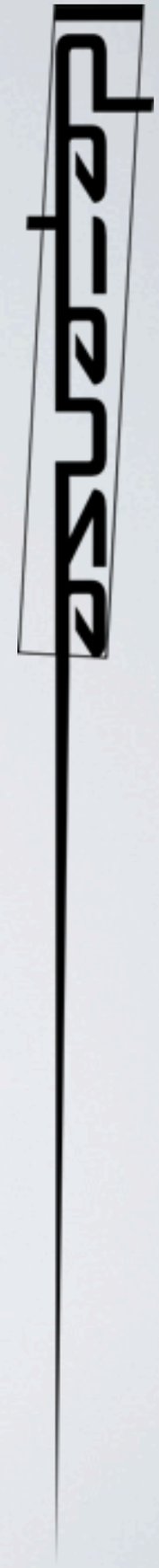
- Tag all packets to recognize them once/if they arrive

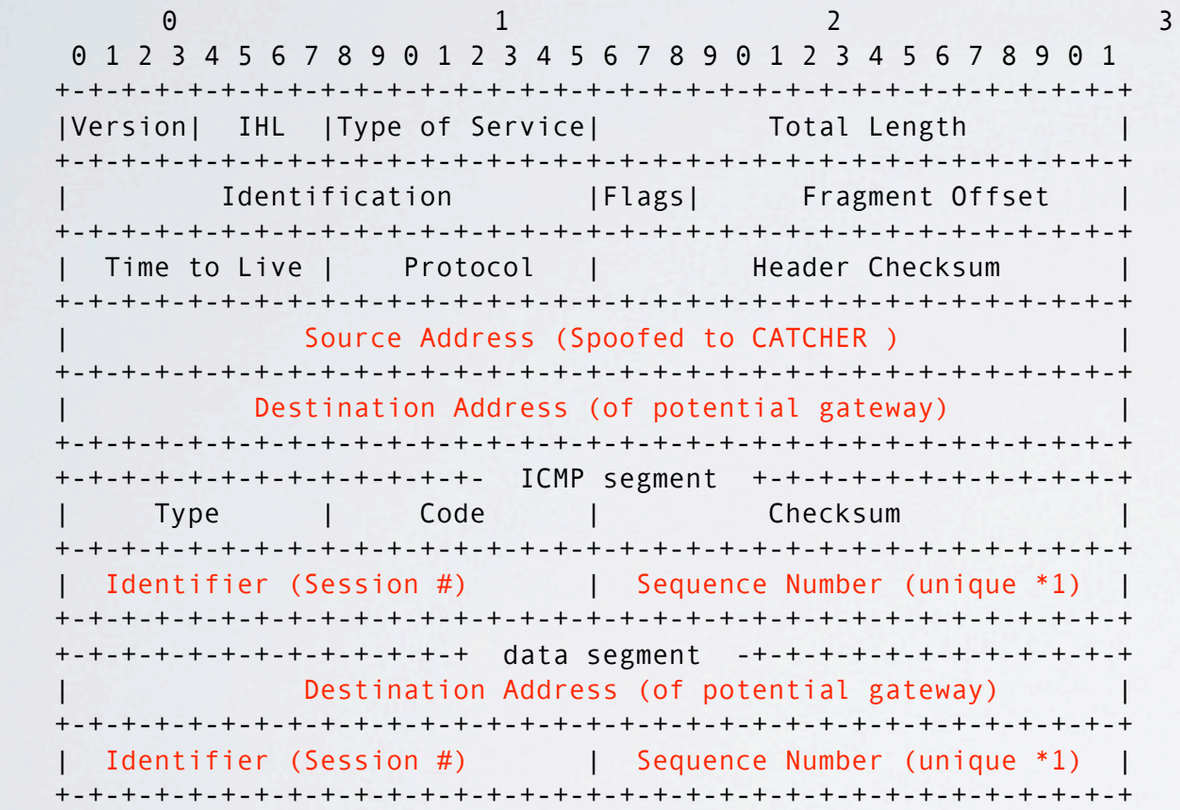
THE PACKETS

- We will send out specially crafted ICMP and UDP packets. The structure of these packets are explained in the following slides.

- Some information is stored redundant to ensure it will be available after receiving the reply. This redundancy can also be used as sanity check for filtering out rogue packets floating on the net.

(THROW 1/3) ICMP ECHO REQUEST

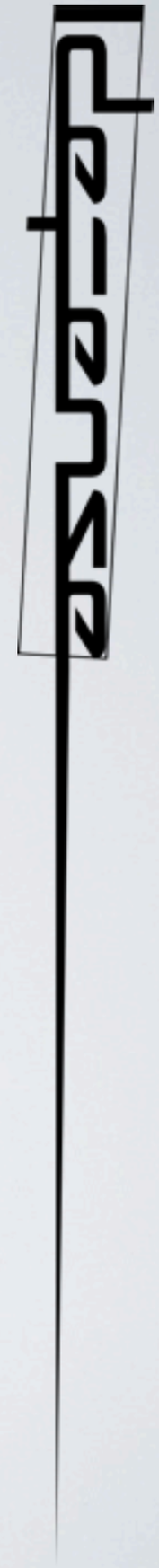




As data we place the Destination Address (the system we send the packet to) we also place it in the data in case another machine is the gateway or replying machine. We want to use a maximum of 64 bits of data since, if we get a DU of PU ICMP reply we will receive Internet Header + 64 bits of Original Data Datagram.

[*1] Unique is a unique number (counter) within a session, we also store locally what we send out, If we can ID the session and retrieve this unique number we can find the packet that was placed on the wire initially.

(THROW 2/3) UDP PACKET

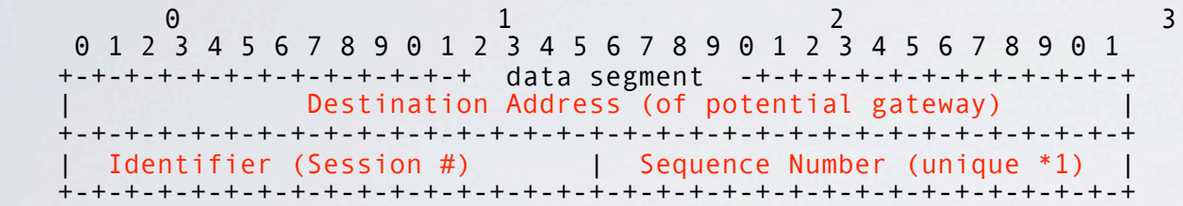


```
      0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Identification (NR in session)|Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              Source Address (Spoofed to CATCHER )             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Destination Address (of potential gateway)          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
+-+-+-+-+-+-+-+-+-+-+-+-  UDP segment  -+-+-+-+-+-+-+-+-+-+-+-+-+
|    Source Port (session #)    |      Destination port (53)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Lenth              |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
+-+-+-+-+-+-+-+-+-+-+-+-+-+  data segment  -+-+-+-+-+-+-+-+-+-+-+-+-+
|              Destination Address (of potential gateway)       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Identifier (Session #)       |  Sequence Number (unique *1)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

As data we place the Destination Address (the system we send the packet to) we also place it in the data in case another machine is the gateway or replying machine. We want to use a maximum of 64 bits of data since, if we get a DU of PU ICMP reply we will receive Internet Header + 64 bits of Original Data Datagram.

[1] (page 3, RFC 792) The address of the gateway or host that composes the ICMP message. Unless otherwise noted, this can be any of a gateway's addresses

(THROW 3/3) DATA IN PACKETS



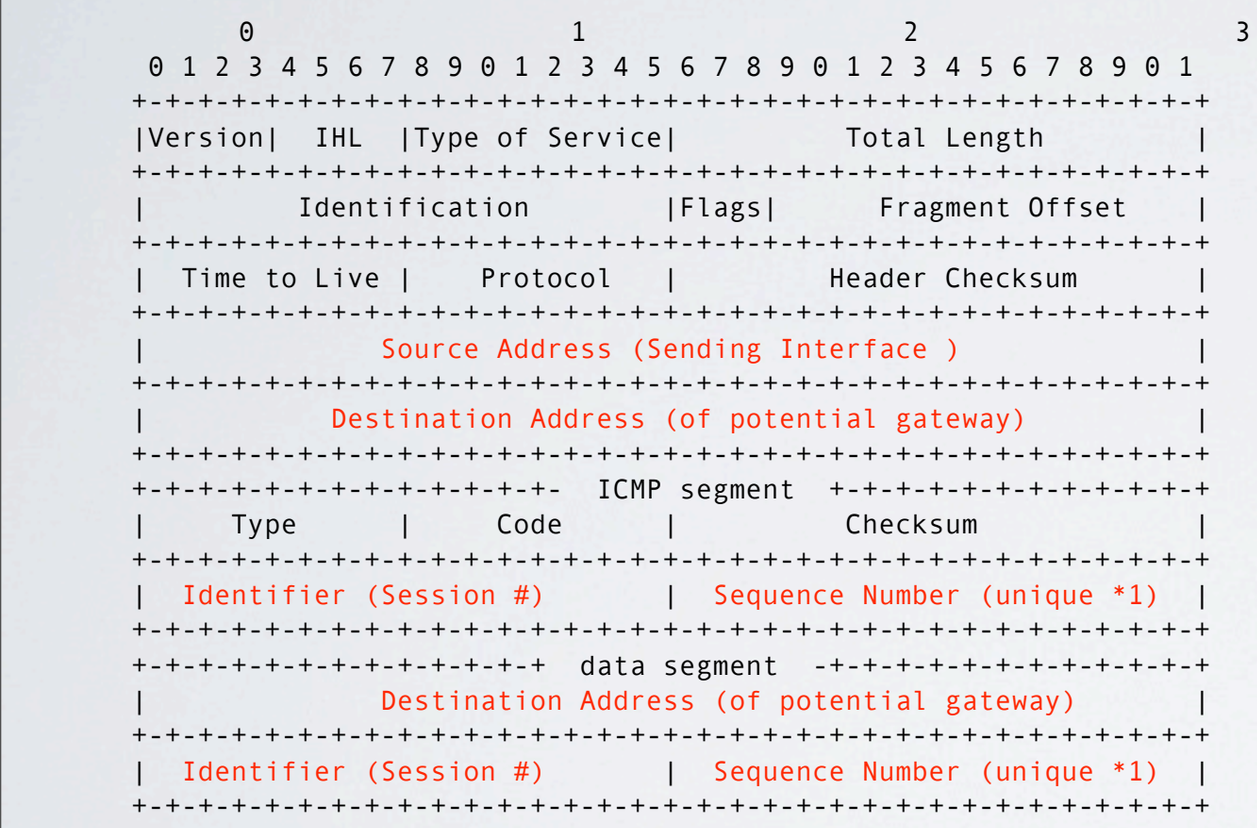
In ICMP and UDP data fields we can stuff at most 8 bytes that we can expect back as stated in RFC792 on page 3. Stuffing the IP address of the system we shoot the initial packet to will take up 4 bytes. This leaves us with 4 bytes, just enough to store session # and unique #

It is necessary to store the system we shoot the packet to in case the packet leaves that machine from another interface or the reply is generated by another machine than the targeted machine (i.e. in case of Destination Unreachable reply). The from address will then contain the IP address of the replying interface. The data should therefore contain the system we targeted. Off course using session # and unique # we can recall the full initial packet from the logs in the ‘thrower’. We include it anyway in order to generate live results without correlation phase.

(CATCH 1/4) WHAT TO EXPECT

- We've send out ICMP Echo Request and UPD packets. We might get back:
  - ICMP reply
  - DU as reply to ICMP
  - DU as reply to UDP
  - PU as reply to UDP

(CATCH 2/4) ICMP REPLY



As data we place the Destination Address (the system we send the packet to) we also place it in the data in case another machine is the gateway or replying machine. We want to use a maximum of 64 bits of data since, if we get a DU of PU ICMP reply we will receive Internet Header + 64 bits of Original Data Datagram.

[*1] Unique is a unique number (counter) within a session, we also store locally what we send out, If we can ID the session and retrieve this unique number we can find the packet that was placed on the wire initially.

(CATCH 3/4) ICMP DU

(CATCH 4/4) ICMP DU

PROOF OF CONCEPT

The setup, vmware fusion with

- 2 Linux VM's, sender and catcher
- Windows VM, our dual homed victim

PROOF OF CONCEPT

The setup, vmware fusion with

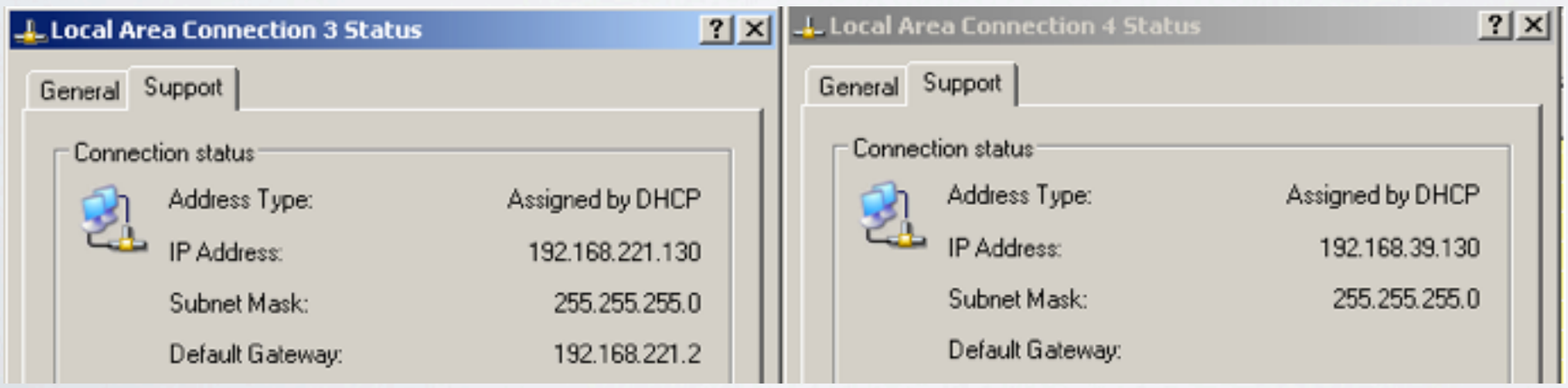
- Linux, Is thrower this POC.

```
eth1      Link encap:Ethernet  HWaddr 00:0c:29:08:56:8d
          inet addr:192.168.39.131  Bcast:192.168.39.255  Mask:255.255.255.0
--
```

- Linux, Is cacher in this POC

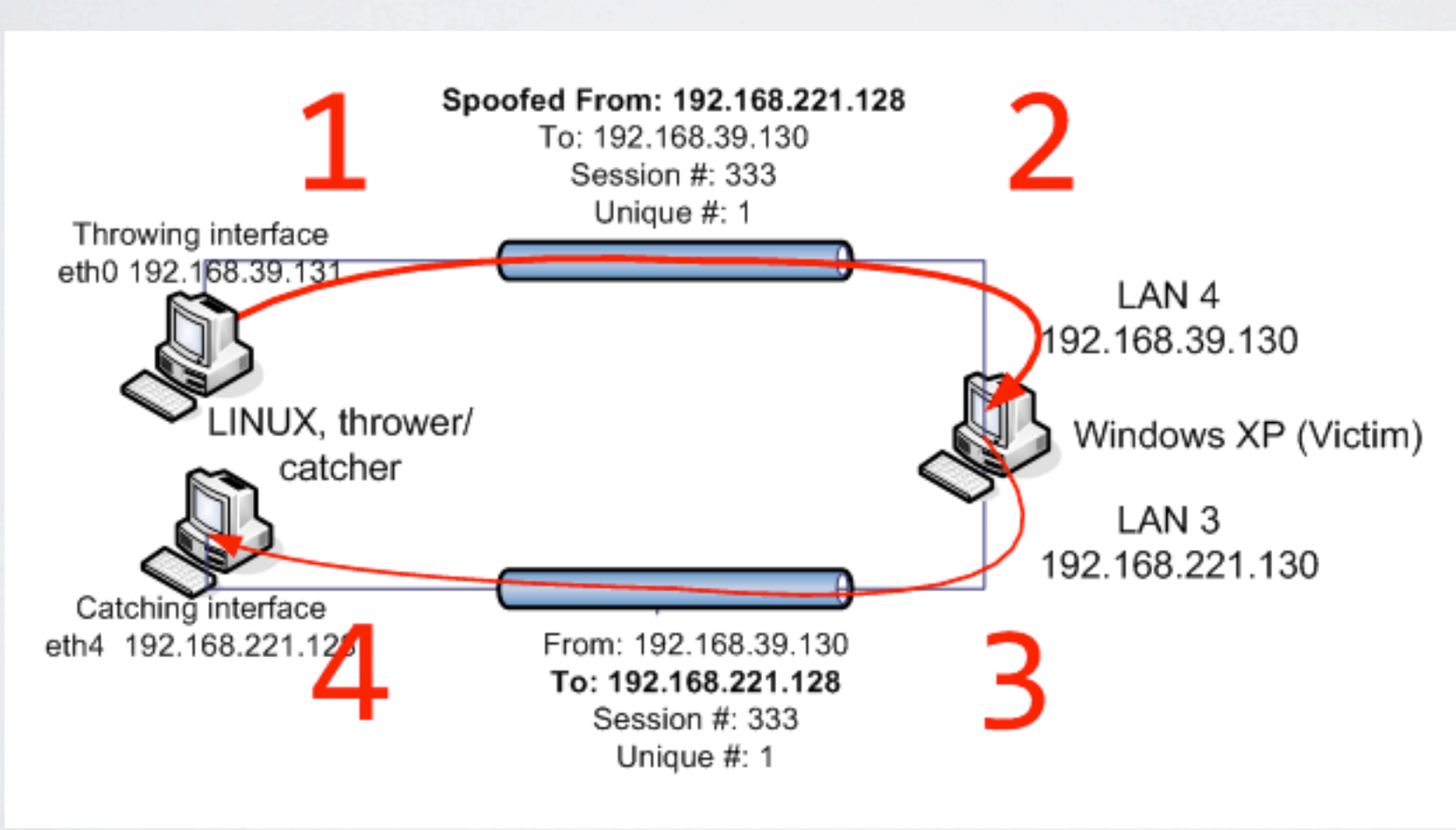
```
eth4      Link encap:Ethernet  HWaddr 00:0c:29:08:56:83
          inet addr:192.168.221.128  Bcast:192.168.221.255  Mask:255.255.255.0
```

- Windows, dual homed



PROOF OF CONCEPT

This setup gives the following network diagram. The numbers indicate the path and steps a specially crafted packet will take.

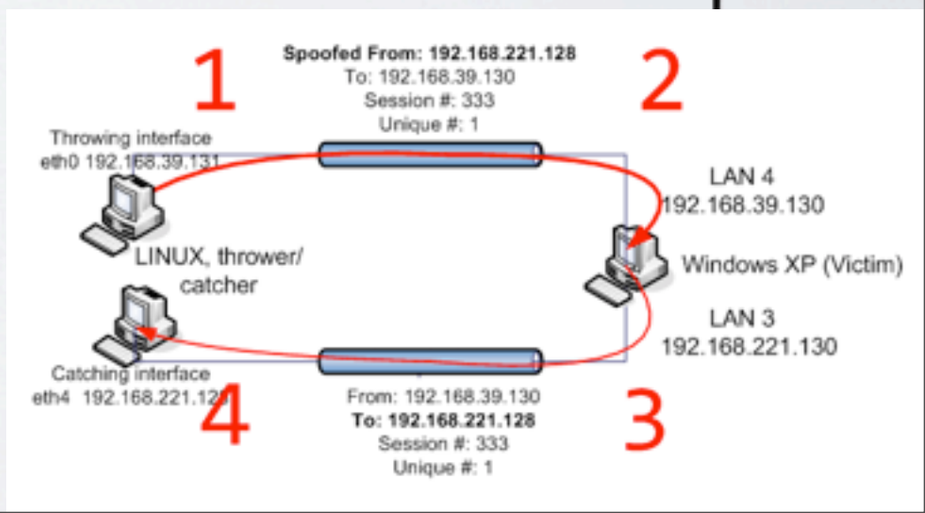


PROOF OF CONCEPT

Step 1: We send a packet from the Linux Thrower machine to the dual homed XP. We want to test if it has an interface on the 192.168.221.0/24 network.

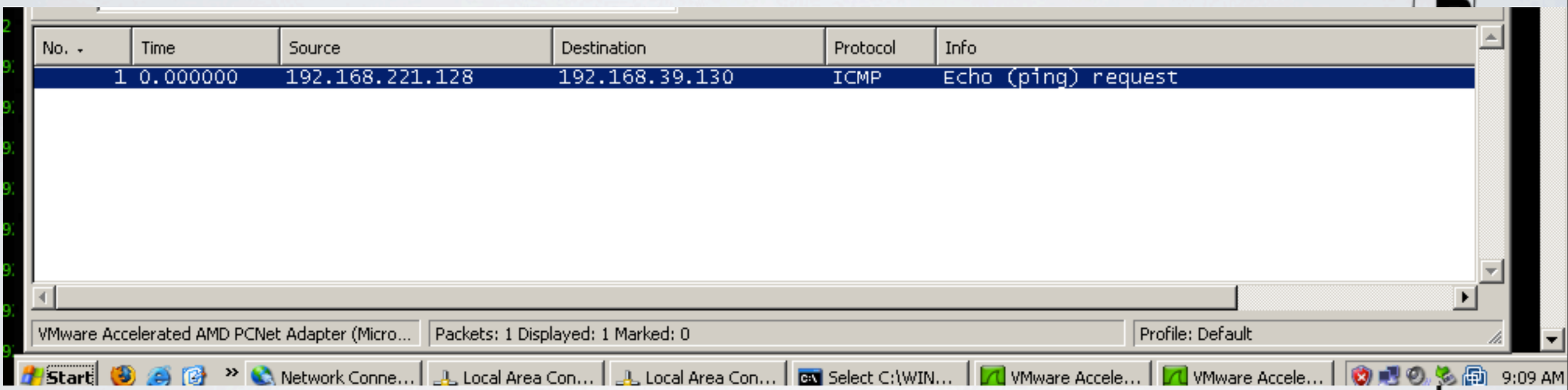
```
>>> from poc import SendUDP, SendICMP
>>> SendICMP("192.168.221.128","192.168.39.130",333,1)
1265786621.7597001
>>>
```

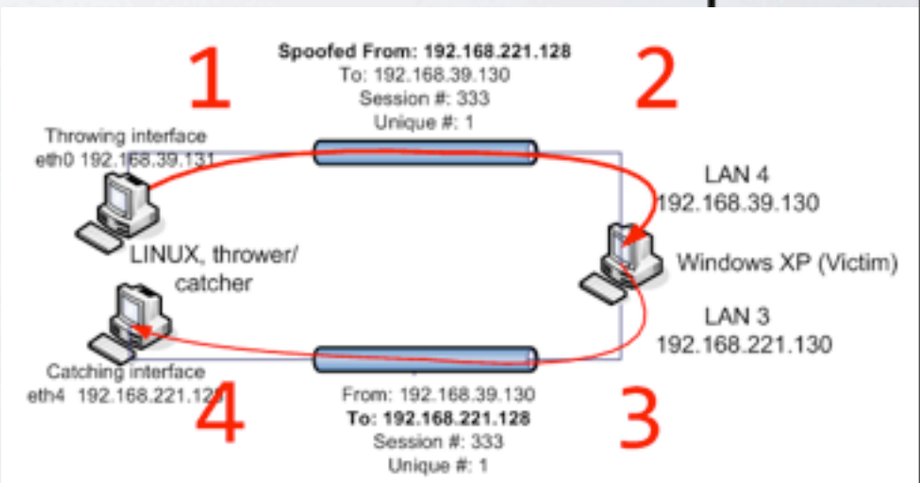
Note: in order to be able to use this code you need python and impacket installed (http://www.python.org/) (http://oss.coresecurity.com/projects/impacket.html)



PROOF OF CONCEPT

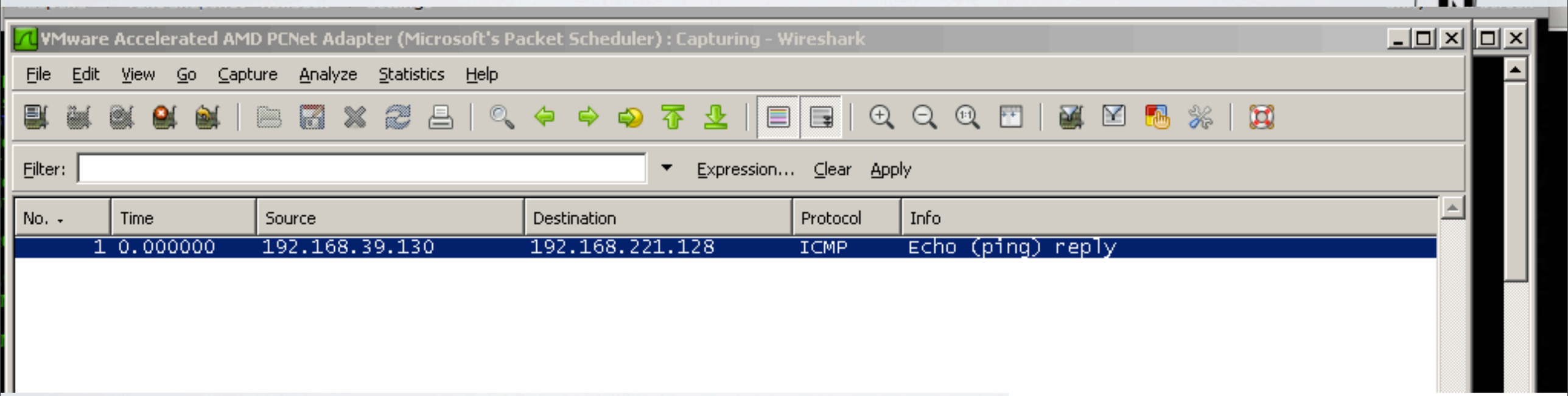
Step 2: The spoofed packet is received on interface LAN 4.

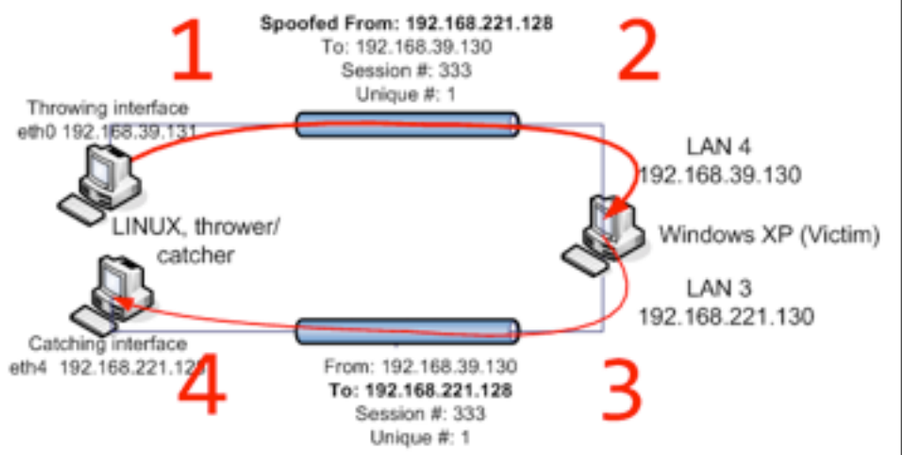




PROOF OF CONCEPT

Step 3: The reply is put on interface LAN 3 determined by the route the packet needs to be delivered to 192.168.221.128



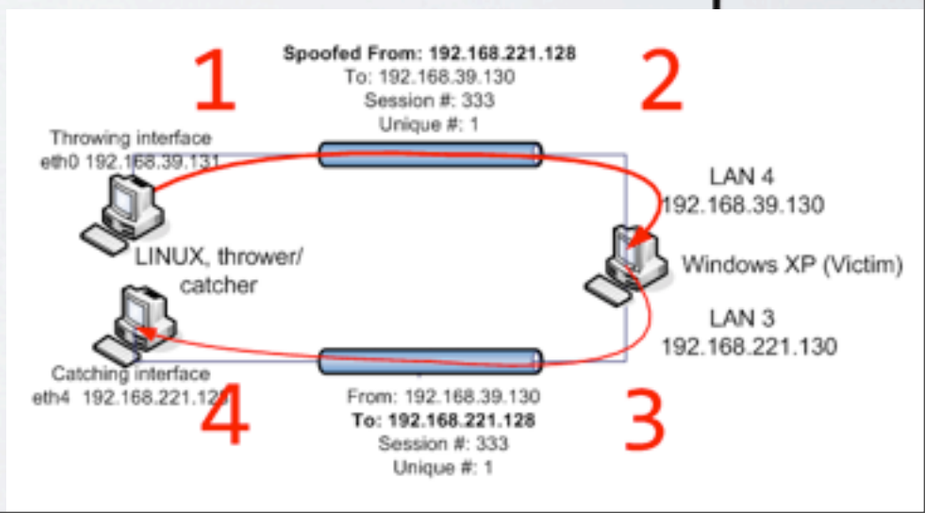


PROOF OF CONCEPT

Step 4: The reply received by the Linux machine on eth4

```
xychix@pentest:~/cathro-dev/WIP/cathro-bin$ sudo tcpdump  -i eth4
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth4, link-type EN10MB (Ethernet), capture size 96 bytes
08:22:30.472170 IP 192.168.39.130 > pentest.local: ICMP echo reply, id 333, seq 1, length 16
```

```
eth4      Link encap:Ethernet  HWaddr 00:0c:29:08:56:83
          inet addr:192.168.221.128  Bcast:192.168.221.255  Mask:255.255.255.0
```



QUESTIONS?

ing. Mark Bergman RE CISA CISSP
mark@bergman.nl
+31 6 18113618