

Albert-Ludwigs-Universität Freiburg

Institute for Computer Science

Seminar Internet-working

Topic:

Ip Address Spoofing

**Leila Fatmasari Rahman
&
Rui Zhou**

Abstract

The paper focuses on IP address spoofing and its application. In section one we introduce what is IP address spoofing. Section two is about IP routing mechanism and its problems. The section three is about forms of IP address spoofing and its applications, we concentrate on splitting routing (asymmetric routing), sat dsl, nat and IP masquerading. In section four we talk about some attacks based on IP address spoofing. Section five is about how to stop IP address spoofing. In the last section, we describe the experiment we did, a splitting routing IP spoofing scenario.

Acknowledgement

Many thanks for the help of Dirk and Nils in the experiments.

Content

1	What is IP address spoofing.....	4
2	IP routing mechanism and problems.....	5
3	IP address spoofing and Applications.....	5
3.1	Asymmetric routing (Splitting routing)	5
3.2	Implementation of asymmetric routing.....	5
3.3	SAT DSL	6
3.4	Probable problem with AOLs DSL connection setup.....	7
3.5	NAT	8
3.6	IP masquerade:.....	9
4	IP address spoofing attack.....	10
4.1	Blind IP spoofing	10
4.2	Man-in-the-middle attacks	10
4.3	Attacks concerning the routing protocols	11
4.4	IP address spoofing attack with ICMP.....	12
4.4.1	ICMP Echo attacks.....	12
4.4.2	ICMP Redirect attacks	12
4.4.3	ICMP destination unreachable attacks.....	13
4.5	UDP attacks.....	14
4.6	TCP attacks	14
5	Stopping IP address spoofing attack	15
5.1	Packet filtering	15
5.2	Limits of packet filtering	15
6	Experiment.....	17
6.1	Scenario description.....	17
6.2	Configuration	17
6.3	Experiment procedure	17
6.4	Experiment result	18
7	Reference	20

1 What is IP address spoofing

IP address spoofing is the creation of IP packets using somebody else's IP source addresses. This technique is used for obvious reasons and is employed in several of the attacks discussed later. Examining the IP header, we can see that the first 12 bytes contain various information about the packet. The next 8 bytes, however, contains the source and destination IP addresses. Using one of several tools, an attacker can easily modify these addresses – specifically the “source address” field.

A common misconception is that "IP spoofing" can be used to hide our IP address while surfing the Internet, chatting on-line, sending e-mail, and so forth. This is generally not true. Forging the source IP address causes the responses to be misdirected, meaning you cannot create a normal network connection.

Figure 1: Valid source IP address, illustrates a typical interaction between a workstation with a valid source IP address requesting web pages and the web server executing the requests. When the workstation requests a page from the web server the request contains both the workstation's IP address (i.e. source IP address 192.168.0.5) and the address of the web server executing the request (i.e. destination IP address 10.0.0.23). The web server returns the web page using the source IP address specified in the request as the destination IP address, 192.168.0.5 and its own IP address as the source IP address, 10.0.0.23.

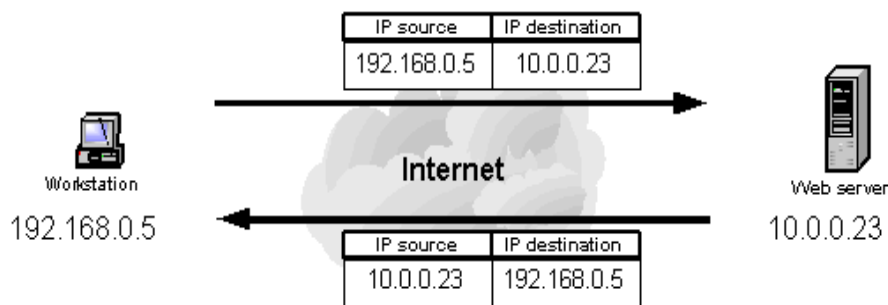


Figure 1: Valid source IP address

Figure 2: Spoofed source IP address, illustrates the interaction between a workstation requesting web pages using a spoofed source IP address and the web server executing the requests. If a spoofed source IP address (i.e. 172.16.0.6) is used by the workstation, the web server executing the web page request will attempt to execute the request by sending information to the IP address of what it believes to be the originating system (i.e. the workstation at 172.16.0.6). The system at the spoofed IP address will receive unsolicited connection attempts from the web server that it will simply discard.

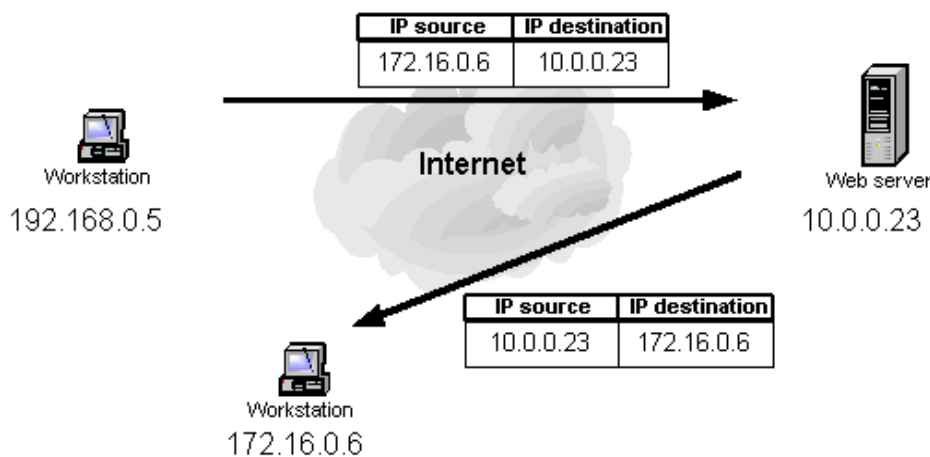


Figure 2: Spoofed source IP address

2 IP routing mechanism and problems

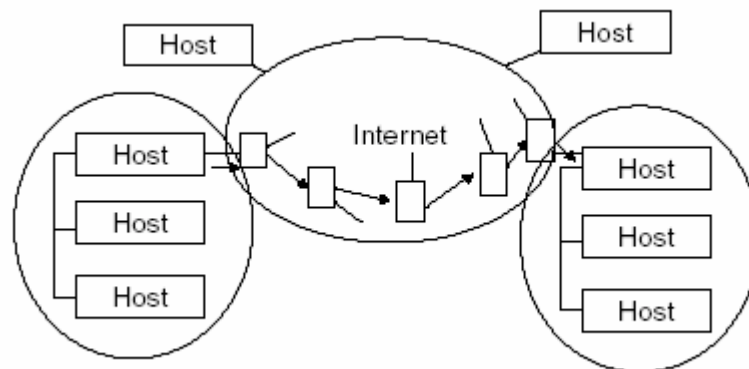


Figure 3: IP Routing mechanism

IP routing is hop by hop. Every IP packet is routed separately. The route of a IP packet is decided by all the routers the packet goes through.

IP address spoofing is possible because routers only require inspection of the destination IP address in the packet to make routing decisions. The source IP address is not required by routers and an invalid source IP address will not affect the delivery of packets.

That address is only used by the destination machine when it responds back to the source.

3 IP address spoofing and Applications

3.1 Asymmetric routing (Splitting routing)

Asymmetric routing means traffic goes over different interfaces for directions in and out. In other words, asymmetric routing is when the response to a packet follows a different path from one host to another than the original packet did. The more correct and more general answer is, for any source IP address 'A' and destination 'B', the path followed by any packet (request or response) from 'A' to 'B' is different than the path taken by a packet from 'B' to 'A'.

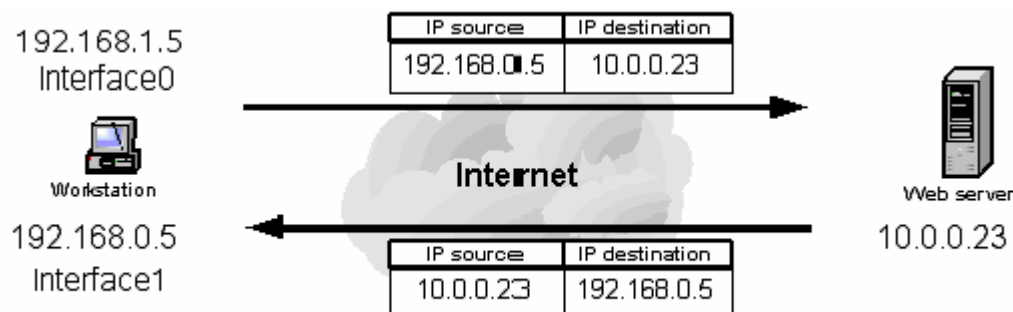


Figure 4: Asymmetric routing

3.2 Implementation of asymmetric routing

Modern O.S. allows us to receive packets from an input interface, different from the output interface.

In Linux, we can implement asymmetric routing using iptables (linux 2.4):

```
iptables -A POSTROUTING -t nat -j SNAT --to 192.168.0.5 --o eth0
```

This means, for all the packets going out via eth0, their source IP address will be changed to 192.168.0.5.

We also have to "disable" reverse path filtering

```
Echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter
```

3.3 SAT DSL

Satellite DSL (SAT DSL) makes use of asymmetric routing.

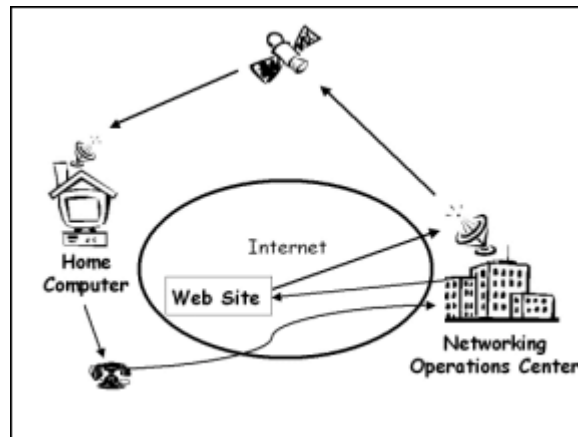


Figure 5. Satellite DSL

The advantage of a satellite network is to provide high bandwidth services independent of the users location over a wide geographical area. A satellite network consists of two types of stations: feeds and receivers. Every receiver has a satellite dish connected to a user station. The user station has an extra interface, DSL modem connected to the ISP, this is called return channel. All requests to Internet are sent via DSL connection, and responses from Internet should be routed by a feed on the satellite network. After the information is sent from the feed to a satellite, it will be broadcast to all the receivers that belong to the satellite coverage. Installing feeds in strategic positions over the Internet will create shorter paths and higher bandwidth provided by the satellite network.

The user host has therefore two IP addresses, one for the satellite subnetwork and the other for the regular connection subnetwork (return channel).

The traffic path of satellite dsl is:

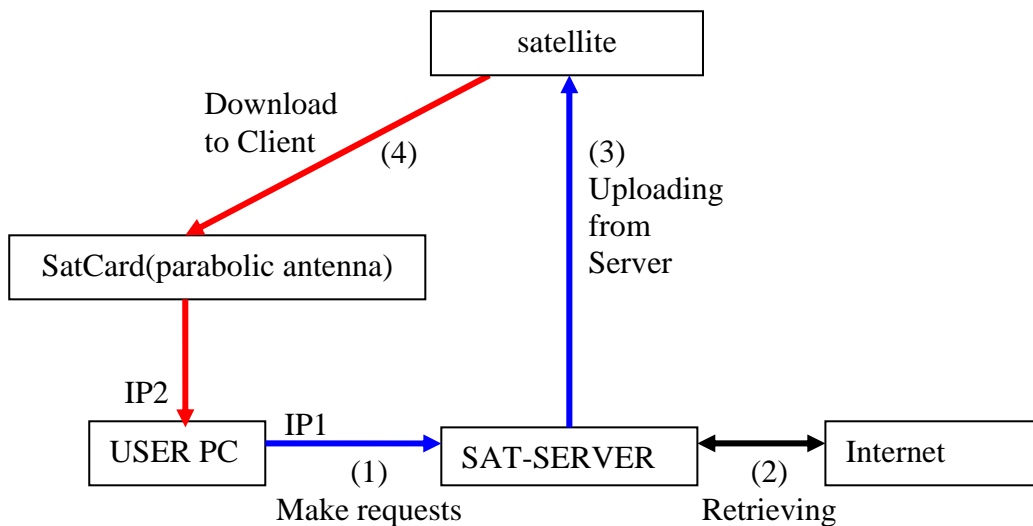


Figure 6: Traffic Path of Satellite DSL

First we make the request (1) (using our Internet connection) to the Sat-Server, after it retrieves out info from Internet (2) it will send it to Satellite (3); in the end we would receive data from the satellite(4) to our home using a parabolic antenna and a Sat Card.

3.4 Probable problem with AOLs DSL connection setup

AOL DSL service implements a certain connection setup procedure in order to apply VPN (Virtual Private Network) for its users. When a user dials in to the AOL DSL ISP, these procedures are taken place:

1. User is connected to the ISP using a public account and so a network connection between user and the ISP is established. But user can only receive data using this connection, thus is not able to send any internet request.
2. On top of this connection, A VPN is established using user's private account. After the authentication succeeds, a user can send and receive data through this VPN connection.

This certain procedures are AOL's attempt to create secure internet traffic over DNS connection. But as it usually is, one solution to a security problem may lead to another problem. And this applies also to AOL's DSL connection setup. With certain setup and an **IP address spoofing** technique, a user can connect to AOL DSL ISP, and download as much data as he wants using this connection without paying any cent. This picture depicts such setup and how the attack works.

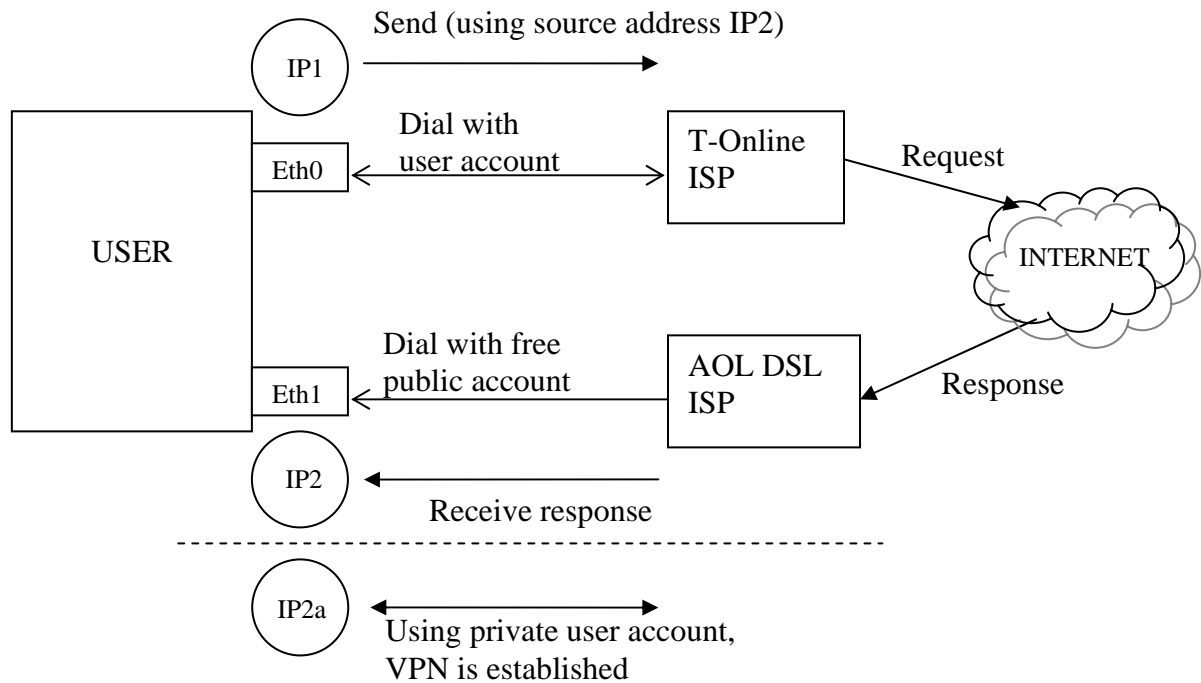


Figure 7: Problem in AOL DSL

1. On first network interface, the user dials for a DSL connection to T-Online or other ISPs using his account. The user can send and receive data with this connection.
2. On second network interface, the user dials to AOL DSL ISP using a free public account to establish a DSL connection that goes one way from ISP to user.
3. Before the user sends packet through T-Online connection, he spoofs the source IP address of the packet into the IP address of the second network interface (which is connected to AOL DSL)
4. And so he sends requests through T-Online connection, and receives response through AOL DSL connection. This way the user only needs to pay for every bits he sends to T-Online, and get for free every bits he receives from AOL DSL, which would have cost a lot more than the cost for sending bits, because people usually spend more time downloading from the internet instead of sending data to the internet.

3.5 NAT

NAT is network address translation.

Normally, packets on a network travel from their source to their destination through many different links. None of these links really alter your packet, they just send it onward.

If one of these links were to do NAT, then they would alter the source or destinations of the packet as it passes through. Usually the link doing NAT will remember how it mangled a packet, and when a reply packet passes through the other way, it will do the reverse mangling on that reply packet, so everything works.

NAT have several applications:

- **Modem Connections To The Internet**

Most ISPs give you a single IP address when you dial up to them. You can send out packets with any source address you want, but only replies to packets with this source IP address will

return to you. If you want to use multiple different machines (such as a home network) to connect to the Internet through this one link, you'll need NAT.

- **Multiple Servers**

Sometimes you want to change where packets heading into your network will go. Frequently this is because (as above) you have only one IP address, but you want people to be able to get into the boxes behind the one with the 'real' IP address. If you rewrite the destination of incoming packets, you can manage this. This type of NAT was called port-forwarding. A common variation of this is load-sharing, where the mapping ranges over a set of machines, fanning packets out to them.

- **Transparent Proxying**

Sometimes you want to pretend that each packet which passes through your Linux box is destined for a program on the Linux box itself. This is used to make transparent proxies: a proxy is a program which stands between your network and the outside world, shuffling communication between the two. The transparent part is because your network won't even know it's talking to a proxy, unless of course, the proxy doesn't work.

NAT has two different types: **Source NAT (SNAT)** and **Destination NAT (DNAT)**.

Source NAT is when you alter the source address of the first packet: i.e. you are changing where the connection is coming from. Source NAT is always done post-routing, just before the packet goes out onto the wire. Masquerading is a specialized form of SNAT.

Destination NAT is when you alter the destination address of the first packet: i.e. you are changing where the connection is going to. Destination NAT is always done before routing, when the packet first comes off the wire. Port forwarding, load sharing, and transparent proxying are all forms of DNAT.

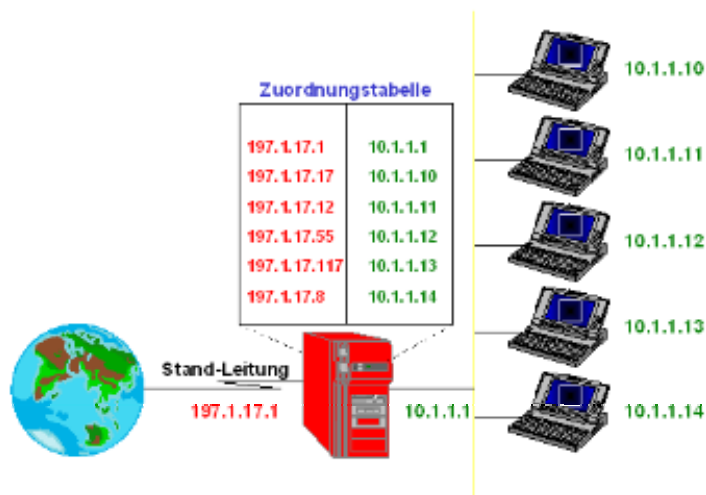


Figure 8: NAT

3.6 IP masquerade:

IP Masquerade, is a specific form of Network Address Translation (NAT) which allows internally connected computers that do not have registered Internet IP addresses to communicate to the Internet via the Linux server's Internet IP address. IP masquerading lets you use a single Internet-connected computer running Linux with a real IP address as a gateway for non-connected machines with "fake" IP addresses. The Linux box with a real address handles mapping packets from your intranet out to the Internet, and when responses come back, it maps them back to your intranet. This lets you browse

the web and use other Internet functions from multiple machines without having a special network setup from your ISP.

IP Masquerade is a networking function in Linux similar to the one-to-many (1:Many) NAT (Network Address Translation) servers found in many commercial firewalls and network routers. For example, if a Linux host is connected to the Internet via PPP, Ethernet, etc., the IP Masquerade feature allows other "internal" computers connected to this Linux box (via PPP, Ethernet, etc.) to also reach the Internet as well. Linux IP Masquerading allows for this functionality even though these internal machines don't have an officially assigned IP address.

IP masquerading is different from NAT. While IP masquerading implements a specific many-to-one NAT, IP NAT allows complex many-to-many translations. For static real IP address we use NAT, while for dynamic real IP address (via PPP) we use IP masquerading.

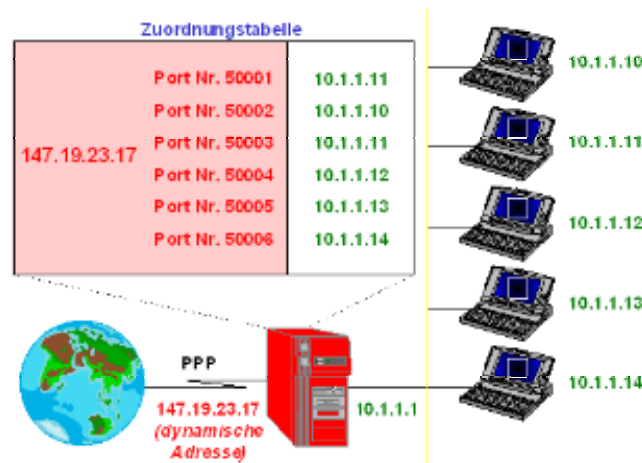


Figure 9: IP Masquerading

4 IP address spoofing attack

4.1 Blind IP spoofing

Usually the attacker does not have access to the reply, abuse trust relationship between hosts.

For example:

Host C sends an IP datagram with the address of some other host (Host A) as the source address to Host B. Attacked host (B) replies to the legitimate host (A)

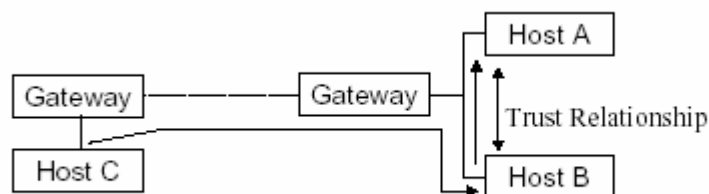


Figure 10: Blind IP Spoofing

4.2 Man-in-the-middle attacks

If an attacker controls a gateway that is in the delivery route, he can

- sniff the traffic
- intercept / block / delay traffic
- modify traffic

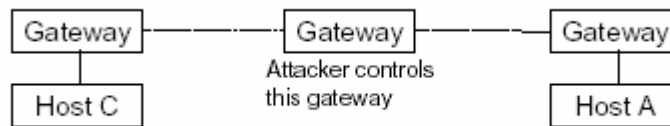


Figure 11: Man-in-the-middle attacks

This is not easy in the Internet because of hop-by-hop routing, unless you control one of the backbone hosts or source routing is used.

This can also be done combined with IP source routing option. IP source routing is used to specify the route in the delivery of a packet, which is independent of the normal delivery mechanisms. If the traffic can be forced through specific routes (=specific hosts), and if the reverse route is used to reply traffic, a host on the route can easily impersonate another host.

The attack procedure could be:

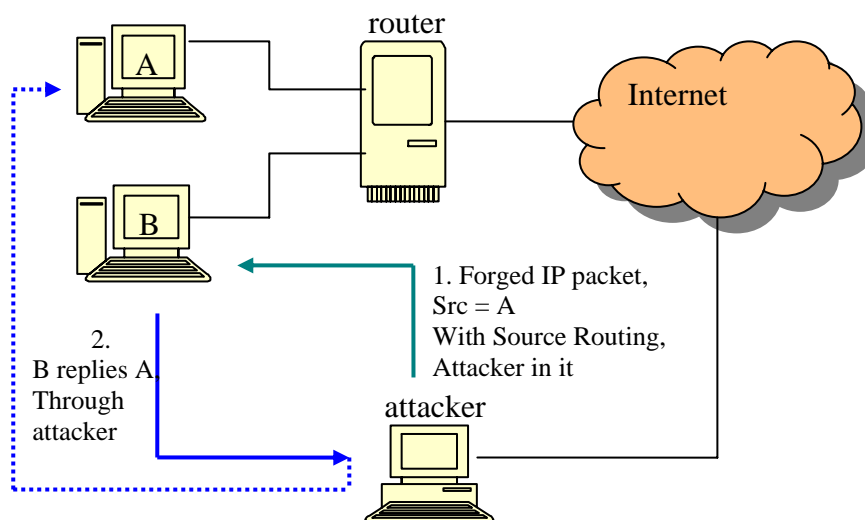


Figure 12: Source Routing attacks

4.3 Attacks concerning the routing protocols

A host can send spoofed RIP packets in order to “inject” routes into a host. This is easy to implement, it only requires IP/UDP spoofing. On a LAN with RIPv2 passwords have to be used for updating routes, but plaintext passwords are used. The plaintext passwords can be sniffed.

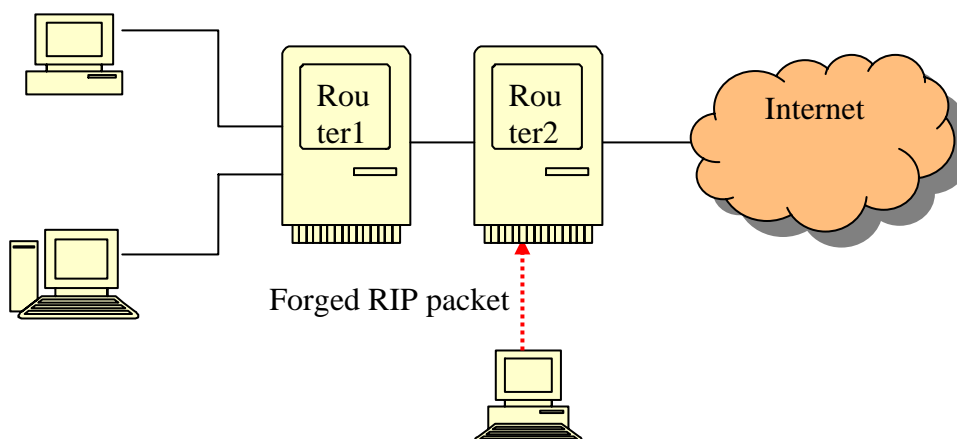


Figure 13: Link state before RIP attack

Attacker sends a forged RIP packet router 2 and says it has the shortest path to the network that router1 connects. Then all the packets to that network will be routed to attacker. The attacker can sniff the traffic.

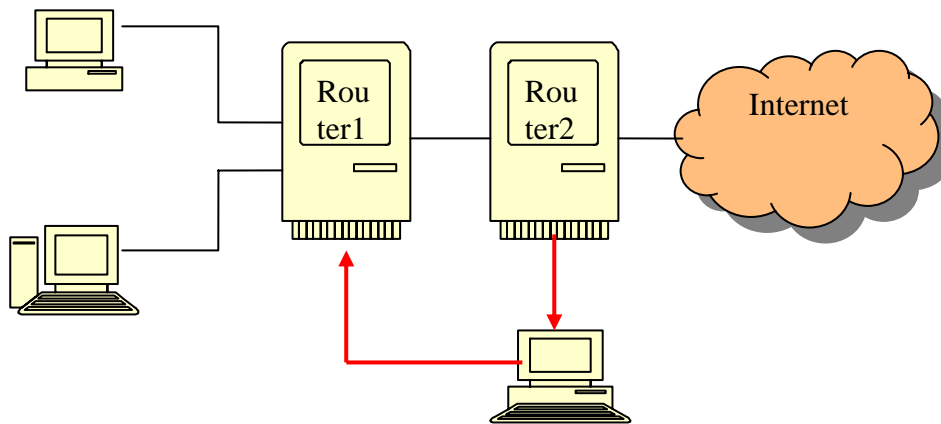


Figure 14: Link State after RIP attack

4.4 IP address spoofing attack with ICMP

4.4.1 ICMP Echo attacks

- Map the hosts of a network
The attack sends ICMP echo datagram to all the hosts in a subnet, then he collects the replies and determines which hosts are alive.
- Denial of service attack (SMURF attack)
The attack sends spoofed (with victim's IP address) ICMP Echo Requests to subnets, the victim will get ICMP Echo Replies from every machine.

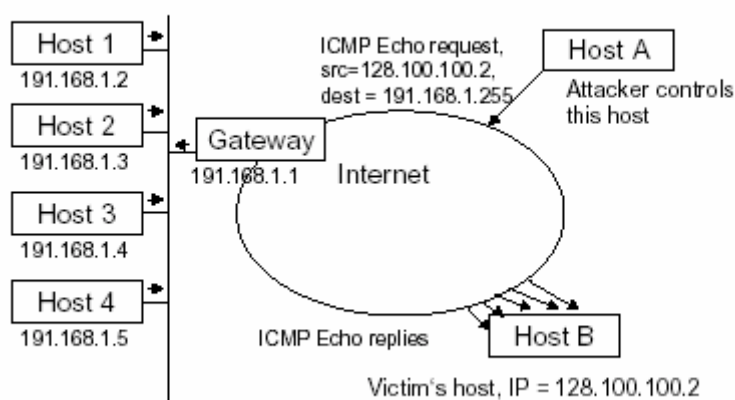


Figure 15: Smurf attack

4.4.2 ICMP Redirect attacks

ICMP redirect messages can be used to re-route traffic on specific routes or to a specific host that is not a router at all.

The ICMP redirect attack is very simple: just send a spoofed ICMP redirect message that appears to come from the host's default gateway.

For example: Host 192.168.1.4 sends a forged ICMP packet to host 192.168.1.3, saying the route through 192.168.1.4 is a better way to internet. The source IP address of this forged ICMP packet is the gateway's IP address 192.168.1.1. Then all the traffic from 192.168.1.3 to internet will go through 192.168.1.4.

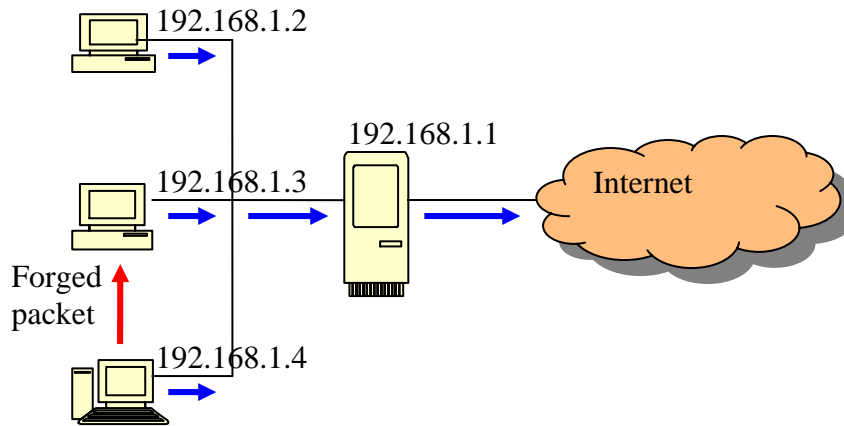


Figure 16: Before ICMP redirect attack

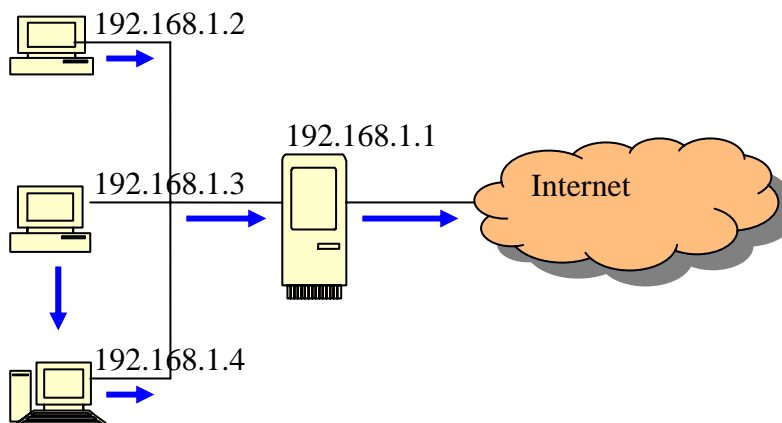


Figure 17: After ICMP redirect attack

4.4.3 ICMP destination unreachable attacks

ICMP destination unreachable message is used by gateways to state that the datagram cannot be delivered. It can be used to “cut” out nodes from the network. It is a denial of service attack (DOS)

Example:

An attacker injects many forged destination unreachable messages stating that 100.100.100.100 is unreachable) into a subnet (e.g. 128.100.100.*). If someone from the 128.100.100.* net tries to contact 100.100.100.100, he will immediately get an ICMP Time Exceeded from the attacker's host. For 128.100.100.* this means that there is no way to contact 100.100.100.100, and therefore communication fails.

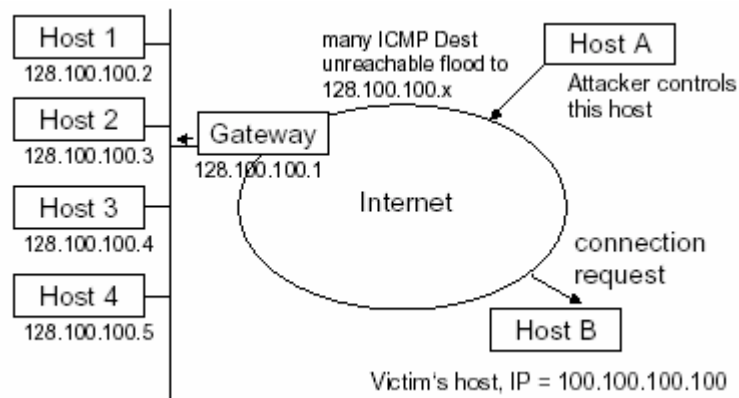


Figure 18: ICMP destination unreachable attacks

4.5 UDP attacks

UDP is an unreliable transport layer protocol. It relies on IP, it is connectionless, and its checksum is optional. Therefore, the delivery, integrity, non-duplication and ordering are not guaranteed. It is easy to send a forged packet to the target. Compared with this, TCP is connection oriented and the TCP connection setup sequence number is hard to predicated, so it is hard to insert forged packet into the TCP connection. Therefore UDP traffic is more vulnerable for IP spoofing than TCP.

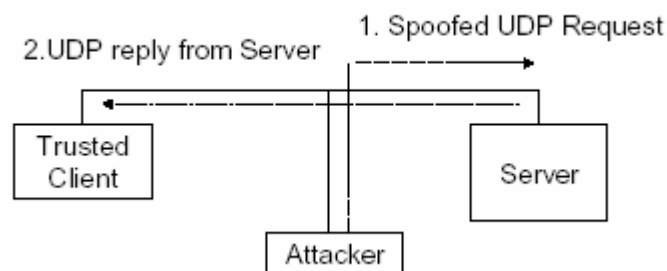


Figure 19: UDP spoofing

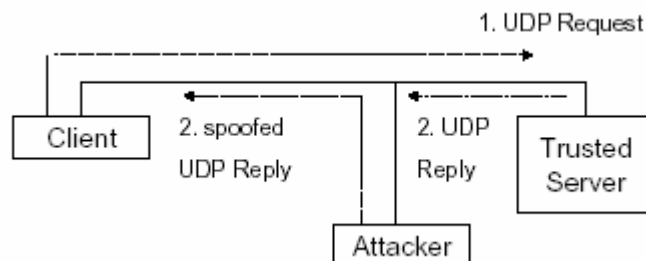


Figure 20: UDP hijacking

4.6 TCP attacks

Although it is hard to do IP spoofing on TCP, it is still can be realized on the specific OS. The attack aims at impersonating another host mostly during the TCP connection establishment phase.

For example:

- 1) Node A trusts node B (e.g. login with no password)
- 2) Node C wants to impersonate B with respect to A in opening a TCP connection
- 3) C kills B (flooding, redirecting or crashing) firstly
- 4) C sends A an TCP segment in a spoofed IP packet with B's address as the source IP and 11000 as the sequence number.
- 5) A replies with a TCP SYN/ACK segment to B with 54002 as the sequence number
- 6) C does not receive the segment from A to B, but in order to finish the handshake it has to send an ACK segment with 54002+1 as the acknowledge number to A. C has to guess or predicate the value of 54002.

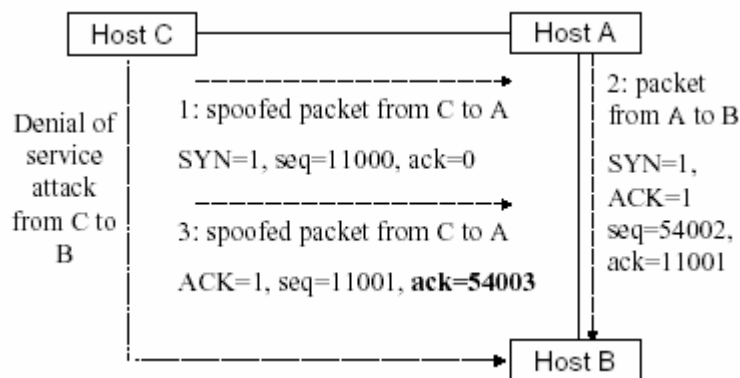


Figure 21: TCP spoofing

5 Stopping IP address spoofing attack

5.1 Packet filtering

The router that connects a network to another network is known as a border router. One way to mitigate the threat of IP spoofing is by inspecting packets when they leave and enter a network looking for invalid source IP addresses. If this type of filtering were performed on all border routers, IP address spoofing would be greatly reduced.

Egress filtering checks the source IP address of packets to ensure they come from a valid IP address range within the internal network. When the router receives a packet that contains an invalid source address, the packet is simply discarded and does not leave the network boundary.

Ingress filtering checks the source IP address of packets that enter the network to ensure they do not come from sources that are not permitted to access the network. At a minimum, all private, reserved, and internal IP addresses should be discarded by the router and not allowed to enter the network.

In Linux, packet filtering can be enabled using:

```
echo 2 > /proc/sys/net/ipv4/conf/*/rp_filter
```

5.2 Limits of packet filtering

Packet filtering normally may not prevent a system from participating in an attack if the spoofed IP address used could fall within the valid internal address range. However it will simplify the process of tracing the packets, since the systems will have to use a source IP address within the valid IP range of the network.

We take the campus network as example:

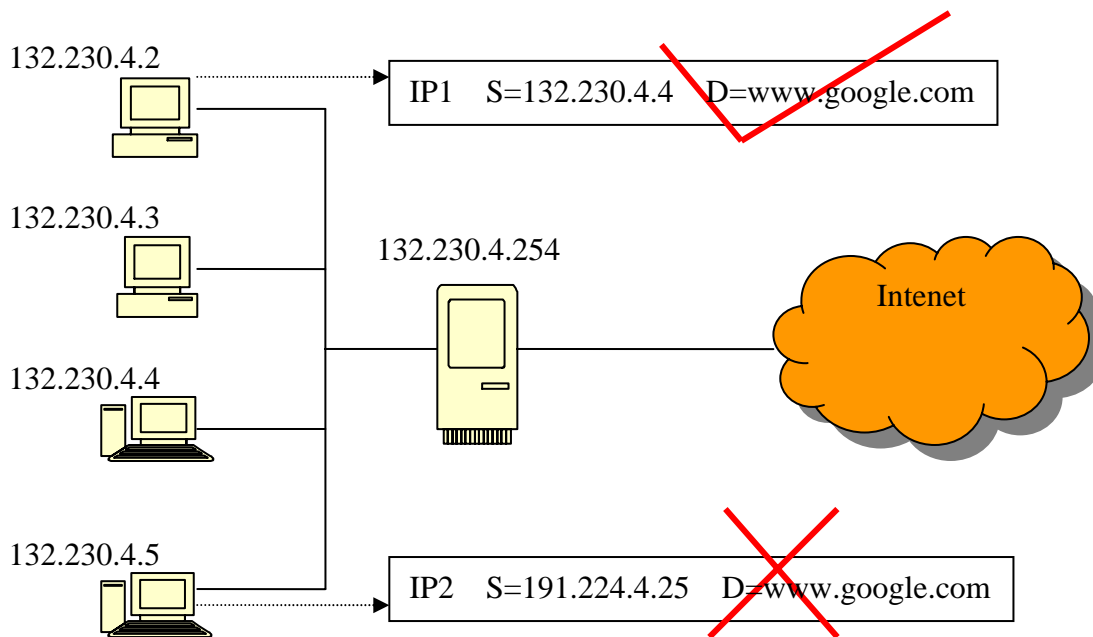


Figure 22: Campus network

The network number is 132.230.0.0/16. The packet filtering of the router is enabled.

For IP packet 1, host 132.230.4.1 forges a packet from 132.230.4.4, the source IP address is in the valid IP range, the router thinks it is valid packet and sends it out to internet.

For IP packet 2, host 132.230.4.4 forges a packet from 191.224.4.25, the source IP address is not in the valid IP range, the router thinks it is invalid and discards it.

Packet filtering can pose problems if you use splitting routing (packets from you to a host take a different path than packets from that host to you). If splitting routing is in use, enabling packet filtering facility will block all packets with spoofed source addresses.

To turn `rp_filter` off, use:

```
echo 0 > /proc/sys/net/ipv4/conf/<device>/rp_filter
```

or

```
echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter
```

Instances where you *might* need to disable packet filtering include:

- If you want to do asymmetric routing (accepting returning packets inbound an interface other than the outbound interface).
- If the box has multiple interfaces up on the same network.
- If you are using special VPN interfaces to tunnel traffic (e.g. FreeS/WAN)

Another problem is that many ISPs do not have the technical ability to arrange packet filtering to block packets with spoofed source addresses. Also, packet filtering reduces equipment performance.

6 Experiment

Goal: Implement an example environment for splitting routing, IP spoofing scenario.

6.1 Scenario description

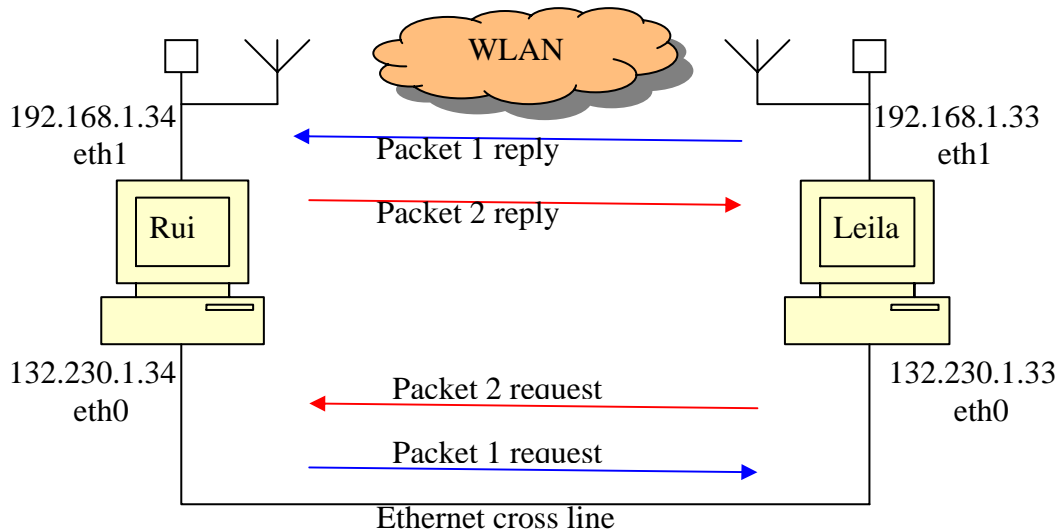


Figure 23: experiment scenario

6.2 Configuration

We do the experiment under Linux Suse 8.0. The tools needed are:

- Iptables
- Ethereal

The configuration is:

Leila:

```
ifconfig eth0 132.230.1.33
ifconfig eth1 192.168.1.33
iptables -A POSTROUTING -t nat -j SNAT -to 192.168.1.33 -o eth0
echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter
```

Rui:

```
ifconfig eth0 132.230.1.34
ifconfig eth1 192.168.1.34
iptables -A POSTROUTING -t nat -j SNAT -to 192.168.1.34 -o eth0
echo "0" > /proc/sys/net/ipv4/conf/all/rp_filter
```

6.3 Experiment procedure

Packet 1: Rui→Leila:

```
ping 132.230.1.33
```

The request packet is sent from interface eth0/Rui, using the IP address of interface eth1/Rui, i.e. 192.168.1.34.

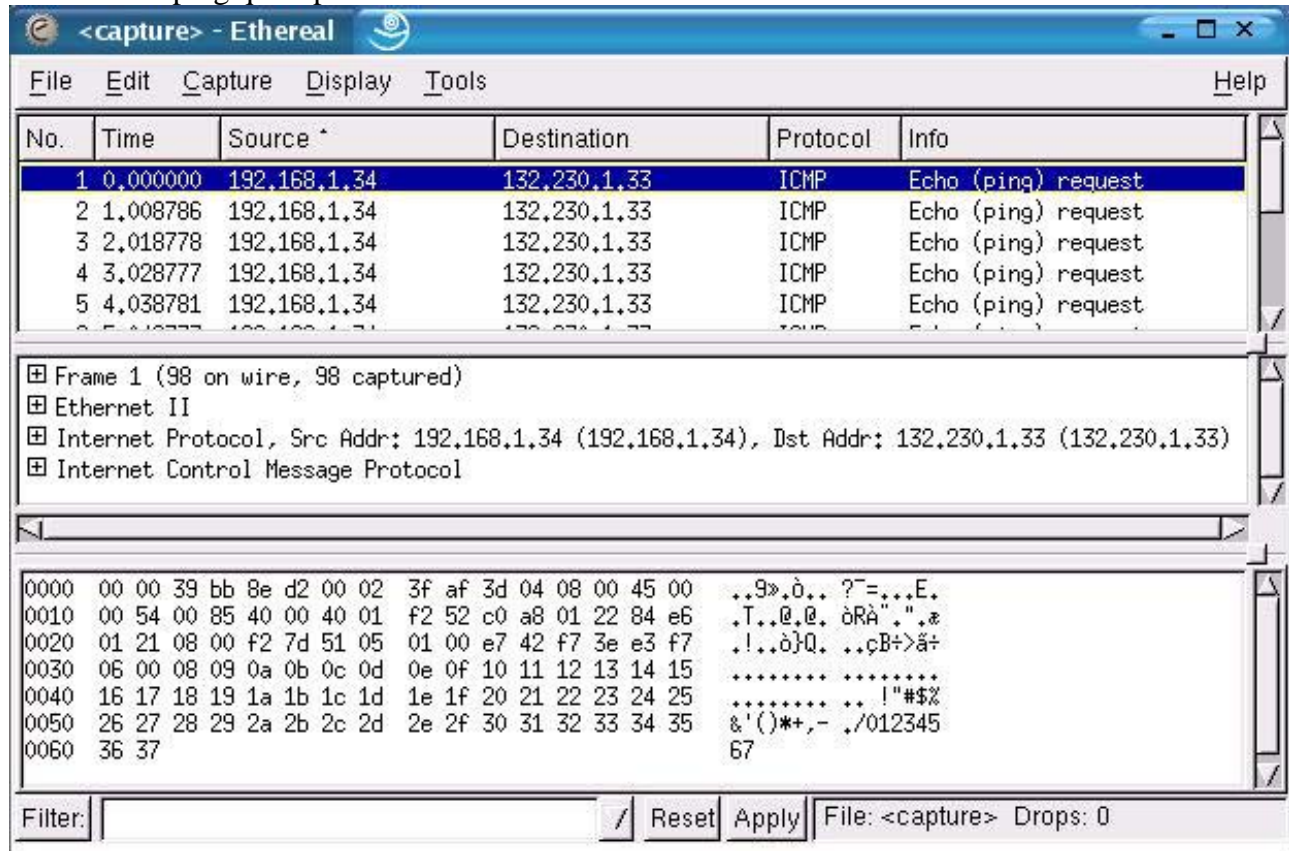
Packet 2: Leila→Rui:

```
ping 132.230.1.34
```

The request packet is sent from interface eth0/Leila, using the IP address of interface eth1/Leila, i.e. 192.168.1.33.

6.4 Experiment result

Eth0: ICMP ping quest packet from Rui to Leila



The screenshot shows the Wireshark interface with a packet capture on the 'Ethereal' interface. The packet list shows five ICMP Echo (ping) requests from source 192.168.1.34 to destination 132.230.1.33. The packet details pane shows the structure of the first packet: Ethernet II, Internet Protocol (Src Addr: 192.168.1.34, Dst Addr: 132.230.1.33), and Internet Control Message Protocol. The packet bytes pane displays the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.34	132.230.1.33	ICMP	Echo (ping) request
2	1.008786	192.168.1.34	132.230.1.33	ICMP	Echo (ping) request
3	2.018778	192.168.1.34	132.230.1.33	ICMP	Echo (ping) request
4	3.028777	192.168.1.34	132.230.1.33	ICMP	Echo (ping) request
5	4.038781	192.168.1.34	132.230.1.33	ICMP	Echo (ping) request

Frame 1 (98 on wire, 98 captured)

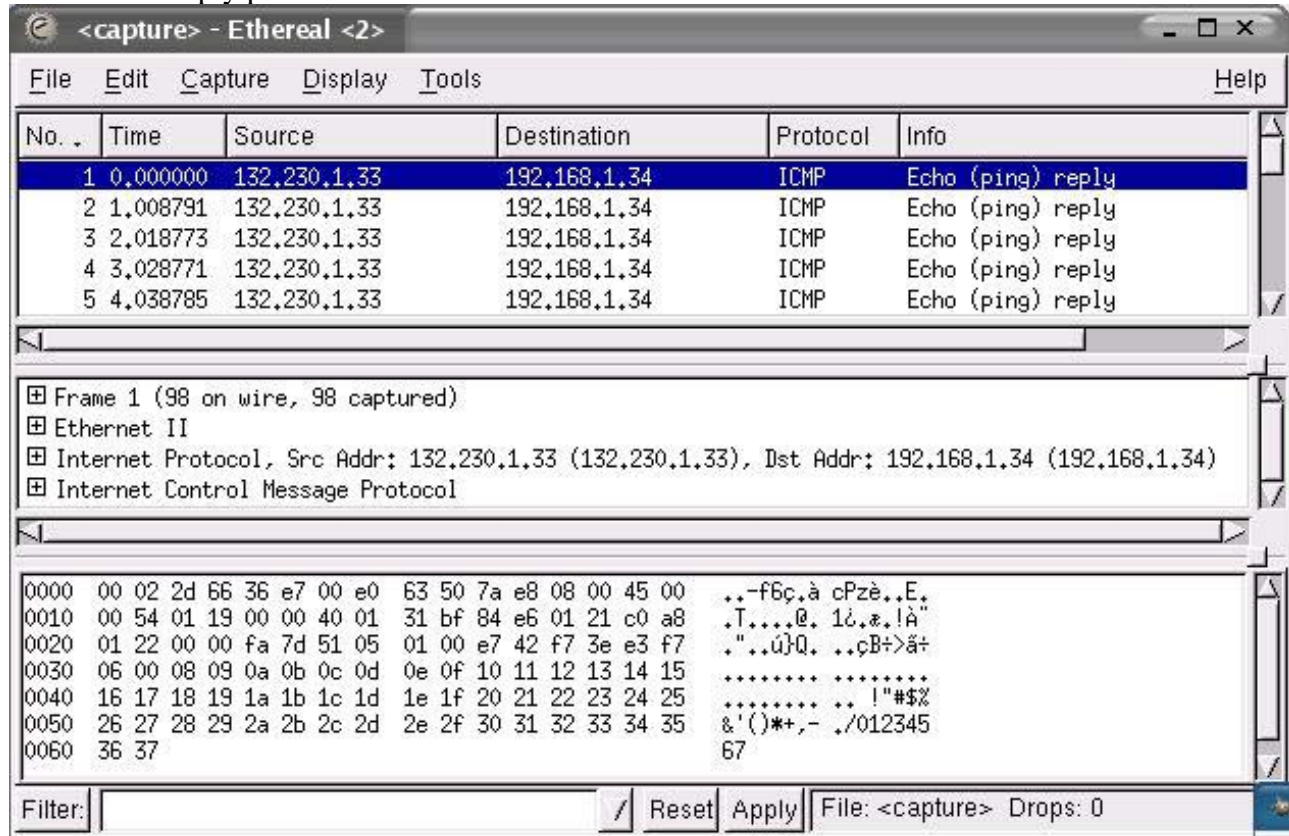
- Ethernet II
- Internet Protocol, Src Addr: 192.168.1.34 (192.168.1.34), Dst Addr: 132.230.1.33 (132.230.1.33)
- Internet Control Message Protocol

```

0000  00 00 39 bb 8e d2 00 02 3f af 3d 04 08 00 45 00  ..9».ò.. ?=-...E.
0010  00 54 00 85 40 00 40 01 f2 52 c0 a8 01 22 84 e6  .T..@.@. òRÀ".",.
0020  01 21 08 00 f2 7d 51 05 01 00 e7 42 f7 3e e3 f7  .!..ò}Q. ..çB+>ã+
0030  06 00 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15  .....
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  ..... !"#$$%
0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0060  36 37 67
  
```

Filter: / Reset Apply File: <capture> Drops: 0

Eth1: ICMP reply packet from Leila to Rui



The screenshot shows the Wireshark interface with a packet capture on the 'Ethereal <2>' interface. The packet list shows five ICMP Echo (ping) replies from source 132.230.1.33 to destination 192.168.1.34. The packet details pane shows the structure of the first packet: Ethernet II, Internet Protocol (Src Addr: 132.230.1.33, Dst Addr: 192.168.1.34), and Internet Control Message Protocol. The packet bytes pane displays the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	132.230.1.33	192.168.1.34	ICMP	Echo (ping) reply
2	1.008791	132.230.1.33	192.168.1.34	ICMP	Echo (ping) reply
3	2.018773	132.230.1.33	192.168.1.34	ICMP	Echo (ping) reply
4	3.028771	132.230.1.33	192.168.1.34	ICMP	Echo (ping) reply
5	4.038785	132.230.1.33	192.168.1.34	ICMP	Echo (ping) reply

Frame 1 (98 on wire, 98 captured)

- Ethernet II
- Internet Protocol, Src Addr: 132.230.1.33 (132.230.1.33), Dst Addr: 192.168.1.34 (192.168.1.34)
- Internet Control Message Protocol

```

0000  00 02 2d 66 36 e7 00 e0 63 50 7a e8 08 00 45 00  ..-f6ç.à cPzè...E.
0010  00 54 01 19 00 00 40 01 31 bf 84 e6 01 21 c0 a8  .T....@. 1ç.ç.!"
0020  01 22 00 00 fa 7d 51 05 01 00 e7 42 f7 3e e3 f7  ."..ú}Q. ..çB+>ã+
0030  06 00 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15  .....
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  ..... !"#$$%
0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0060  36 37 67
  
```

Filter: / Reset Apply File: <capture> Drops: 0

Eth0: ICMP ping request packet from Leila to Rui

The screenshot shows a Wireshark capture on interface 'capture'. The packet list contains six entries. Packet 3, at time 0.000169, is an ICMP Echo (ping) request from source 192.168.1.33 to destination 132.230.1.34. The packet details pane shows the Ethernet II header, Internet Protocol header (Src Addr: 192.168.1.33, Dst Addr: 132.230.1.34), and Internet Control Message Protocol header. The packet bytes pane shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	Intel_bb:8e:d2	ff:ff:ff:ff:ff:ff	ARP	Who has 132.230.1.34? Tell 192.168.1.33
2	0.000142	Compal_af:3d:04	Intel_bb:8e:d2	ARP	132.230.1.34 is at 00:02:3f:af:3d:04
3	0.000169	192.168.1.33	132.230.1.34	ICMP	Echo (ping) request
4	1.008144	192.168.1.33	132.230.1.34	ICMP	Echo (ping) request
5	2.018142	192.168.1.33	132.230.1.34	ICMP	Echo (ping) request
6	3.028141	192.168.1.33	132.230.1.34	ICMP	Echo (ping) request

Frame 3 (98 on wire, 98 captured)

- Ethernet II
- Internet Protocol, Src Addr: 192.168.1.33 (192.168.1.33), Dst Addr: 132.230.1.34 (132.230.1.34)
- Internet Control Message Protocol

```

0000  00 02 3f af 3d 04 00 00 39 bb 8e d2 08 00 45 00  ..?-=... 9».ò..E.
0010  00 54 01 41 40 00 00 01 f1 96 c0 a8 01 21 84 e6  .T.A@.0. ñ.À".!.*
0020  01 22 08 00 b9 a4 33 06 01 00 1a 46 f7 3e 04 cd  ."..¹x3. ...F+>.í
0030  09 00 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15  .....
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  ..... ..!"#$%
0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0060  36 37 67
  
```

Filter: / Reset Apply File: <capture> Drops: 0

Eth1: ICMP reply packet from Rui to Leila

The screenshot shows a Wireshark capture on interface 'capture'. The packet list contains six entries. Packet 3, at time 0.001725, is an ICMP Echo (ping) reply from source 132.230.1.34 to destination 192.168.1.33. The packet details pane shows the Ethernet II header, Internet Protocol header (Src Addr: 132.230.1.34, Dst Addr: 192.168.1.33), and Internet Control Message Protocol header. The packet bytes pane shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	Agere_66:36:e7	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.1.33? Tell 132.230.1.34
2	0.000020	CABLETRO_50:7a:e8	Agere_66:36:e7	ARP	192.168.1.33 is at 00:e0:63:50:7a:e8
3	0.001725	132.230.1.34	192.168.1.33	ICMP	Echo (ping) reply
4	1.007839	132.230.1.34	192.168.1.33	ICMP	Echo (ping) reply
5	2.017829	132.230.1.34	192.168.1.33	ICMP	Echo (ping) reply
6	3.027833	132.230.1.34	192.168.1.33	ICMP	Echo (ping) reply

Frame 3 (98 on wire, 98 captured)

- Ethernet II
- Internet Protocol, Src Addr: 132.230.1.34 (132.230.1.34), Dst Addr: 192.168.1.33 (192.168.1.33)
- Internet Control Message Protocol

```

0000  00 e0 63 50 7a e8 00 02 2d 66 36 e7 08 00 45 00  .àcPzè.. -f6ç..E.
0010  00 54 00 99 00 00 40 01 32 3f 84 e6 01 22 c0 a8  .T....0. 2?.&."À"
0020  01 21 00 00 c1 a4 33 06 01 00 1a 46 f7 3e 04 cd  .!..Áx3. ...F+>.í
0030  09 00 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15  .....
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  ..... ..!"#$%
0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0060  36 37 67
  
```

Filter: / Reset Apply File: <capture> Drops: 0

7 Reference

- [1] Following the Journey of a Spoofed Packet
<http://www.scs.carleton.ca/~dlwhyte/whytepapers/ipspoof.htm>
- [2] NAT and Networks
<http://www.suse.de/~mha/linux-ip-nat/diplom/node4.html>
- [3] Asymmetric routing
Jani Lakkakorpi
<http://keskus.hut.fi/tutkimus/ipana/paperit/QoS/S130-QoS-asymmetric.pdf>
- [4] TCP/IP protocol suite
Thomas Toth
<http://www.infosys.tuwien.ac.at/Teaching/Courses/InetSec/slides/slides2.pdf>
- [5] Security problems in the TCP/IP protocol suite, S.M. Bellovin, AT&T Bell Laboratories, Murray Hill, New Jersey 07974
<http://www.research.att.com/~smb/papers/ipext.pdf>
- [6] Linux 2.4 NAT HOWTO
<http://www.netfilter.org/unreliable-guides/NAT-HOWTO/>
- [7] Linux IP Masquerade HOWTO
<http://www.tldp.org/HOWTO/IP-Masquerade-HOWTO/index.html>
- [8] Linux 2.4 Advanced Routing HOWTO
<http://www.linuxdocs.org/HOWTOs/Adv-Routing-HOWTO.html>
- [9] Introduction To Network Address Translation (NAT)
<http://www.firewall.cx/nat-intro.php>
- [10] Network Address Translation (NAT/ PAT/ IP Masquerading)
http://home.t-online.de/home/TschiTschi/ip_masquerading.htm
- [11] Attacks over the internet
<http://zork.net/~phil/Cracking/Internet.html>
- [12] IP spoofing
<http://bear.cba.ufl.edu/teets/projects/ISM6222F102/perryna/index.html>