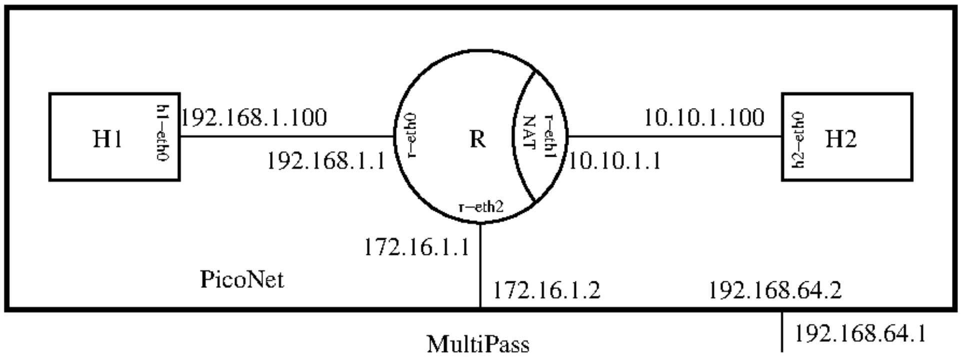
8 IP ADDRESSING AND TRANSLATION

Objective: Students to understand IP addressing and Network Address Translation

To understand IP addressing and Network Address Translation (NAT) on piconet and in other scenarios



On H1

```
root@h1 piconet> ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
4: h1-eth0@if3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
        link/ether 4a:0a:42:c0:37:86 brd ff:ff:ff:ff:ff:ff link-netnsid 1
        inet 192.168.1.100/24 scope global h1-eth0
        valid_lft forever preferred_lft forever
```

On R

```
root@r piconet> ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
```

```
3: r-eth0@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 52:d7:c7:00:b2:b1 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.1.1/24 scope global r-eth0
    valid_lft forever preferred_lft forever
5: r-eth1@if6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 26:1f:09:e7:c2:18 brd ff:ff:ff:ff:ff:ff link-netnsid 1
    inet 10.10.1.1/24 scope global r-eth1
    valid_lft forever preferred_lft forever
8: r-eth2@if7: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 36:1f:c7:6a:1b:a8 brd ff:ff:ff:ff:ff:ff link-netnsid 2
    inet 172.16.1.1/24 scope global r-eth2
    valid_lft forever preferred_lft forever
```

On H2

```
root@h2 piconet> ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
6: h2-eth0@if5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 32:2b:d8:6a:b6:35 brd ff:ff:ff:ff:ff:ff link-netnsid 1
    inet 10.10.1.100/24 scope global h2-eth0
    valid_lft forever preferred_lft forever
```

On H1

```
root@h1 piconet> ping -c 5 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=0.043 ms
64 bytes from h2 (10.10.1.100): icmp_seq=2 ttl=63 time=0.081 ms
64 bytes from h2 (10.10.1.100): icmp_seq=3 ttl=63 time=0.073 ms
64 bytes from h2 (10.10.1.100): icmp_seq=4 ttl=63 time=0.087 ms
64 bytes from h2 (10.10.1.100): icmp_seq=5 ttl=63 time=0.066 ms

--- h2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4203ms
rtt min/avg/max/mdev = 0.043/0.070/0.087/0.015 ms
```

On H2

```
root@h2 piconet> tcpdump -l -n -i h2-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h2-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
13:52:35.269914 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 23339, seq 1, length 64
13:52:35.269930 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 23339, seq 1, length 64
13:52:36.312517 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 23339, seq 2, length 64
13:52:36.312542 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 23339, seq 2, length 64
13:52:37.377290 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 23339, seq 3, length 64
13:52:37.377315 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 23339, seq 3, length 64
13:52:38.427394 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 23339, seq 4, length 64
13:52:38.427416 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 23339, seq 4, length 64
13:52:39.473276 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 23339, seq 5, length 64
13:52:39.473292 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 23339, seq 5, length 64
```

On R, enable NAT between R and H2

```
root@r piconet> iptables -t nat -A POSTROUTING -o r-eth1 -j MASQUERADE
```

iptables is used to set up, maintain, and inspect the tables of IP packet filter rules in the Linux kernel. Several different tables may be defined. Each table contains a number of built-in chains and may also contain user-defined chains.
Each chain is a list of rules which can match a set of packets. Each rule specifies what to do with a packet that matches. This is called a 'target', which may be a jump to a user-defined chain in the same table.

-t, --table table
This option specifies the packet matching table which the command should operate on. If the kernel is configured with automatic module loading, an attempt will be made to load the appropriate module for that table if it is not already there.

nat:This table is consulted when a packet that creates a new connection is encountered. It consists of three built-ins: PREROUTING (for altering packets as soon as they come in), OUTPUT (for altering locally-generated packets before routing), and POSTROUTING (for altering packets as they are about to go out).

-A, --append chain rule-specification
Append one or more rules to the end of the selected chain. When the source and/or destination names resolve to more than one address, a rule will be added for each possible address combination.
-o, --out-interface [!] name
Name of an interface via which a packet is going to be sent (for packets entering the FORWARD, OUTPUT and POSTROUTING chains). When the "!" argument is used before the interface name, the sense is inverted. If the interface name ends in a "+", then any interface which begins with this name will match. If this option is omitted, any interface name will match.
-j, --jump target

This specifies the target of the rule; i.e., what to do if the packet matches it. The target can be a user-defined chain (other than the one this rule is in), one of the special builtin targets which decide the fate of the packet immediately, or an extension (see EXTENSIONS below). If this option is omitted in a rule (and -g is not used), then matching the rule will have no effect on the packet's fate, but the counters on the rule will be incremented.
MASQUERADE
This target is only valid in the nat table, in the POSTROUTING chain. It should only be used with dynamically assigned IP (dialup) connections: if you have a static IP address, you should use the SNAT target. Masquerading is equivalent to specifying a mapping to the IP address of the interface the packet is going out, but also has the effect that connections are forgotten when the interface goes down. This is the correct behavior when the next dialup is unlikely to have the same interface address (and hence any established connections are lost anyway). It takes one option:
--to-ports port[-port]
This specifies a range of source ports to use, overriding the default SNAT source port-selection heuristics (see above). This is only valid if the rule also specifies -p tcp or -p udp.

On H1, ping again

```
root@h1 piconet> ping -c 5 h2
PING h2 (10.10.1.100) 56(84) bytes of data.
64 bytes from h2 (10.10.1.100): icmp_seq=1 ttl=63 time=0.053 ms
64 bytes from h2 (10.10.1.100): icmp_seq=2 ttl=63 time=0.065 ms
64 bytes from h2 (10.10.1.100): icmp_seq=3 ttl=63 time=0.074 ms
64 bytes from h2 (10.10.1.100): icmp_seq=4 ttl=63 time=0.061 ms
64 bytes from h2 (10.10.1.100): icmp_seq=5 ttl=63 time=0.066 ms

--- h2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4189ms
rtt min/avg/max/mdev = 0.053/0.063/0.074/0.012 ms
```

On H2, tcpdump now shows ping from 10.10.1.1, instead of 192.168.1.100

```
14:03:49.019338 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 1, length 64
14:03:49.019361 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 1, length 64
14:03:50.082522 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 2, length 64
14:03:50.082551 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 2, length 64
14:03:51.115850 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 3, length 64
14:03:51.115877 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 3, length 64
14:03:52.171858 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 4, length 64
14:03:52.171901 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 4, length 64
14:03:53.246001 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 5, length 64
14:03:53.246029 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 5, length 64
```

On R, tcpdump shows the translation between H1 (h1-eth0)’s IP address and R (r-eth1)’s IP address

```
root@r piconet> tcpdump -l -n -i any
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes
14:03:49.019311 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 24256, seq 1, length 64
14:03:49.019337 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 1, length 64
14:03:49.019362 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 1, length 64
14:03:49.019375 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 24256, seq 1, length 64
14:03:50.082480 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 24256, seq 2, length 64
14:03:50.082521 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 2, length 64
14:03:50.082553 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 2, length 64
14:03:50.082557 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 24256, seq 2, length 64
14:03:51.115823 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 24256, seq 3, length 64
14:03:51.115848 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 3, length 64
14:03:51.115878 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 3, length 64
14:03:51.115883 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 24256, seq 3, length 64
14:03:52.171832 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 24256, seq 4, length 64
14:03:52.171856 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 4, length 64
14:03:52.171902 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 4, length 64
14:03:52.171907 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 24256, seq 4, length 64
14:03:53.245965 IP 192.168.1.100 > 10.10.1.100: ICMP echo request, id 24256, seq 5, length 64
14:03:53.245999 IP 10.10.1.1 > 10.10.1.100: ICMP echo request, id 24256, seq 5, length 64
14:03:53.246030 IP 10.10.1.100 > 10.10.1.1: ICMP echo reply, id 24256, seq 5, length 64
14:03:53.246035 IP 10.10.1.100 > 192.168.1.100: ICMP echo reply, id 24256, seq 5, length 64
```