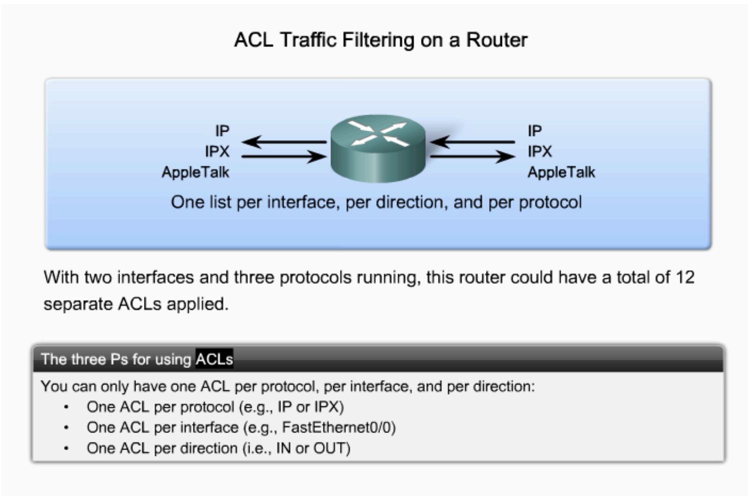
The Network Layer

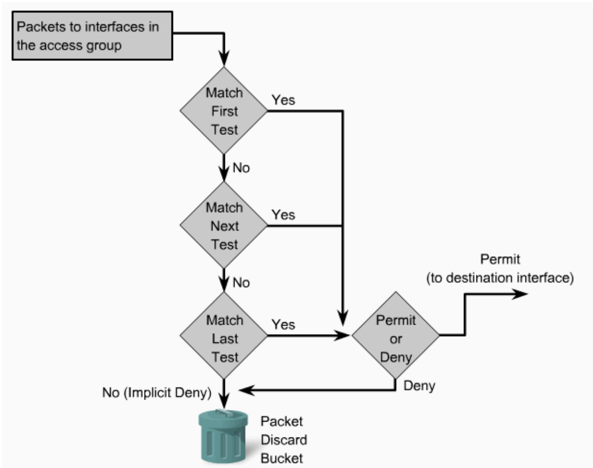
ACLs, DHCP and NAT

What is an ACL



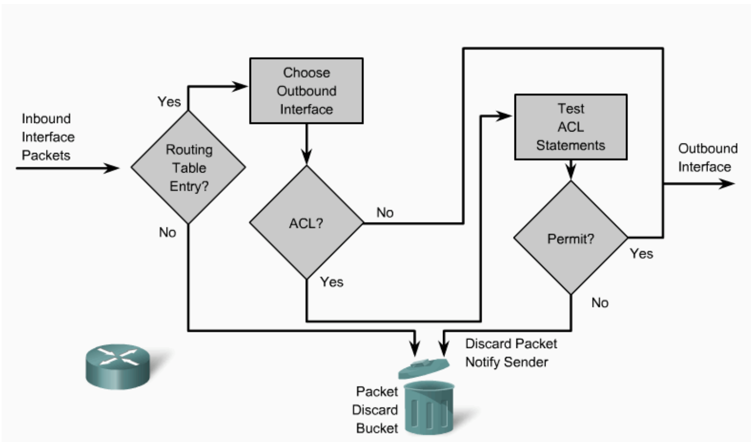
An Access Control List (ACL) is a router configuration script that controls whether a router permits or denies packets to pass based on criteria found in the packet header.

How ACLs Work



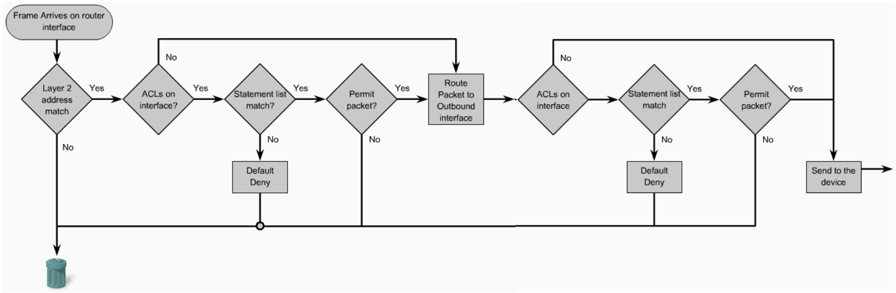
Inbound ACLs

How ACLs Work



Outbound ACLs

ACL Operation



The Implied "Deny All Traffic" Criteria Statement

•At the end of every access list is an implied "deny all traffic" criteria statement.
•If a packet does not match any of the ACL entries, it is automatically blocked.
•The implied "deny all traffic" is the default behavior of ACLs and cannot be changed.

Types of ACLs

1. **Standard ACLs**
    - Allow you to permit or deny traffic from source IP addresses.
    - The destination of the packet and the ports involved do not matter.
    - The example allows all traffic from network 192.168.30.0/24 network.
    - Because of the implied "deny any" at the end, all other traffic is blocked with this ACL.
2. **Extended ACLs**
    - Extended ACLs filter IP packets based on several attributes, for example, protocol type, source and destination IP address, source and destination TCP or UDP ports. In the figure, ACL 103 permits traffic originating from any address on the 192.168.30.0/24 network to any destination host port 80 (HTTP).

Standard ACLs filter IP packets based on the source address only.

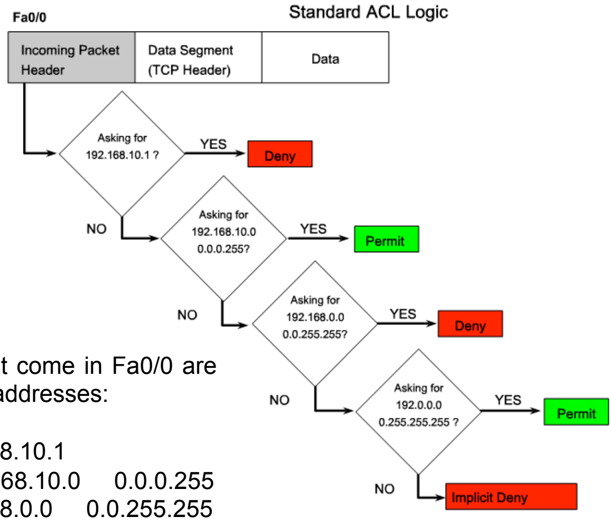
```
access-list 10 permit 192.168.30.0 0.0.0.255
```

Extended ACLs filter IP packets based on several attributes, including the following:

- Source and destination IP addresses
- Source and destination TCP and UDP ports
- Protocol type (IP, ICMP, UDP, TCP, or protocol number)

```
access-list 103 permit tcp 192.168.30.0 0.0.0.255 any eq 80
```

Standard ACL example



In the figure, packets that come in Fa0/0 are checked for their source addresses:

access-list 2 deny 192.168.10.1
access-list 2 permit 192.168.10.0 0.0.0.255
access-list 2 deny 192.168.0.0 0.0.255.255
access-list 2 permit 192.0.0.0 0.255.255.255

The `any` and `host` keywords

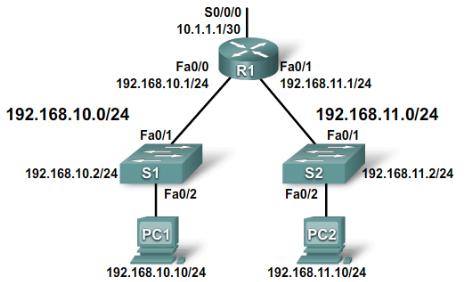
Example 1:

```
R1(config)#access-list 1 permit 0.0.0.0 255.255.255.255
R1(config)#access-list 1 permit any
```

Example 2:

```
R1(config)#access-list 1 permit 192.168.10.10 0.0.0.0
R1(config)#access-list 1 permit host 192.168.10.10
```

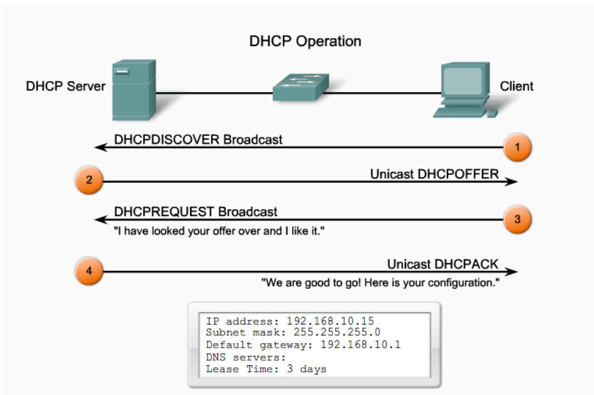
Standard ACL Example




```
R1(config)#no access-list 1
R1(config)#access-list 1 deny 192.168.10.10 0.0.0.0
R1(config)#access-list 1 permit 192.168.10.0 0.0.0.255
R1(config)#interface S0/0/0
R1(config-if)#ip access-group 1 out
```

The first command deletes the previous version of ACL 1.
The next ACL statement, denies the PC1 host located at 192.168.10.10.
Every other host on the 192.168.10.0 /24 network is permitted.
The implicit deny statement matches every other network.
The ACL is applied to interface S0/0/0 in an outbound direction.

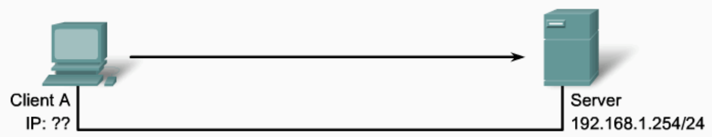
DHCP



- DHCP assigns IP addresses and other important network configuration information dynamically
- RFC 2131 describes DHCP

DHCP

DHCP Discover and Offer



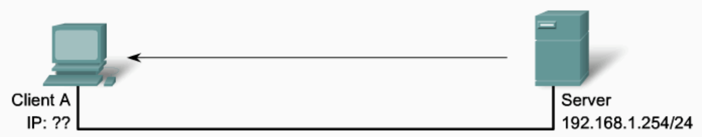
Ethernet Frame	IP	UDP	DHCPDISCOVER
SRC MAC: MAC A DST MAC: FF:FF:FF:FF:FF:FF	IP SRC: ? IP DST: 255.255.255.255	UDP 67	CIADDR: ? GIADDR: ? Mask:? CHADDR: MAC A

MAC: Media Access Control Address
CIADDR: Client IP Address
GIADDR: Gateway IP Address
CHADDR: Client Hardware Address

The DHCP Client sends a directed IP broadcast, with a DHCP discover packet. In the simplest case, there is a DHCP server on the same segment, which will pick up this request. The server notes the GIADDR field is blank, so the client is on the same segment. The server also notes the hardware address of the client in the request packet.

DHCP

How Does DHCP Work?

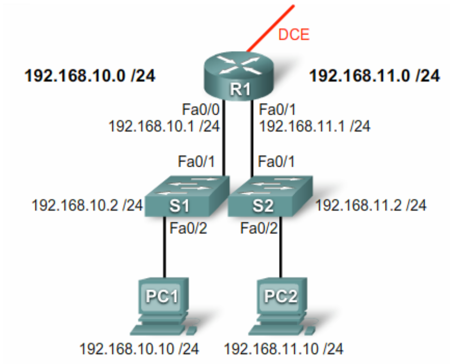


Ethernet Frame	IP	UDP	DHCP Reply
SRC MAC: MAC Serv DST MAC: MAC A	IP SRC: 192.168.1.254 IP DST: 192.168.1.10	UDP 68	CIADDR: 192.168.1.10 GIADDR: ? Mask: 255.255.255.0 CHADDR: MAC A

MAC: Media Access Control Address
CIADDR: Client IP Address
GIADDR: Gateway IP Address
CHADDR: Client Hardware Address

The DHCP server picks an IP address from the available pool for that segment, as well as the other segment and global parameters. It puts them into the appropriate fields of the DHCP packet. It then uses the hardware address of A (in CHADDR) to construct an appropriate frame to send back to the client.

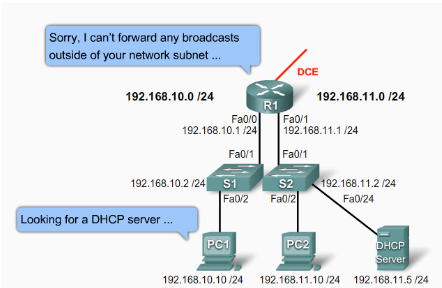
DHCP Configuration



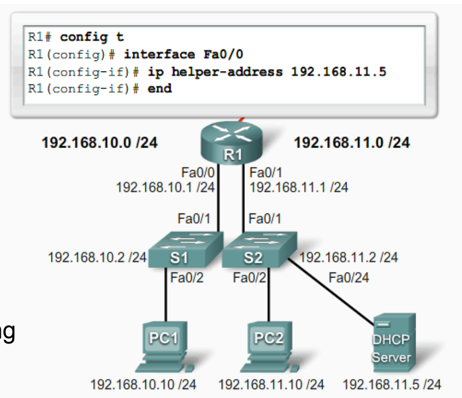
ip dhcp excluded-address 192.168.10.1 192.168.10.9
ip dhcp excluded-address 192.168.10.254
ip dhcp pool LAN-POOL-1
network 192.168.10.0 255.255.255.0
default-router 192.168.10.1
domain-name span.com

ip dhcp excluded-address 192.168.11.1 192.168.11.9
ip dhcp excluded-address 192.168.11.254
ip dhcp pool LAN-POOL-2
network 192.168.11.0 255.255.255.0
default-router 192.168.11.1
domain-name span.com

DHCP Relay Agent



This solution enables routers to forward DHCP broadcasts to the DHCP servers. When a router forwards address assignment/parameter requests, it is acting as a DHCP relay agent.



Network Address Translation

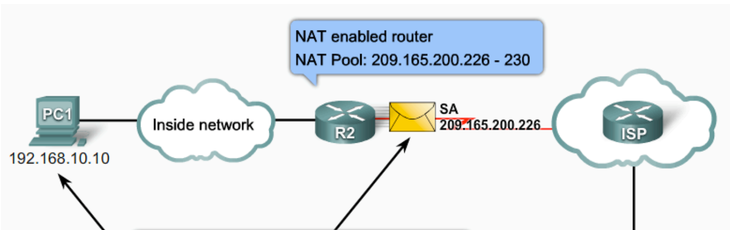
Public Internet addresses are regulated by five Regional Internet Registries (RIRs):

- ARIN
- RIPE
- APNIC
- LACNIC
- AfriNIC

Private Internet addresses are defined in RFC 1918:

Class	RFC 1918 Internal Address Range	CIDR Prefix
A	10.0.0.0 - 10.255.255.255	10.0.0.0/8
B	172.16.0.0 - 172.31.255.255	172.16.0.0/12
C	192.168.0.0 - 192.168.255.255	192.168.0.0/16

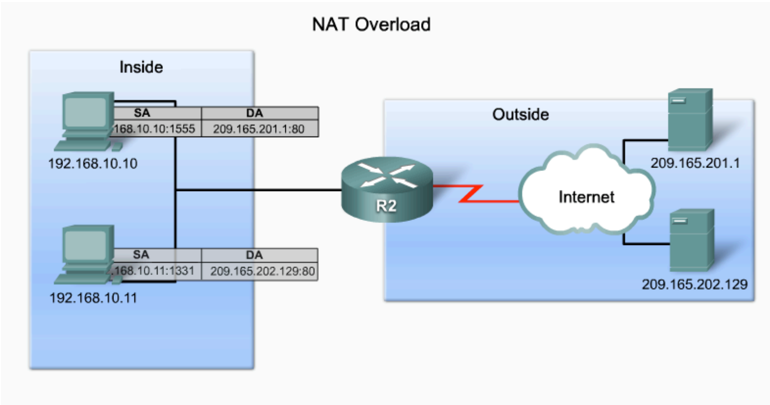
NAT



NAT Table		
Inside Local Address	Inside Global Address	Outside Global Address
192.168.10.10	209.165.200.226	209.165.201.1

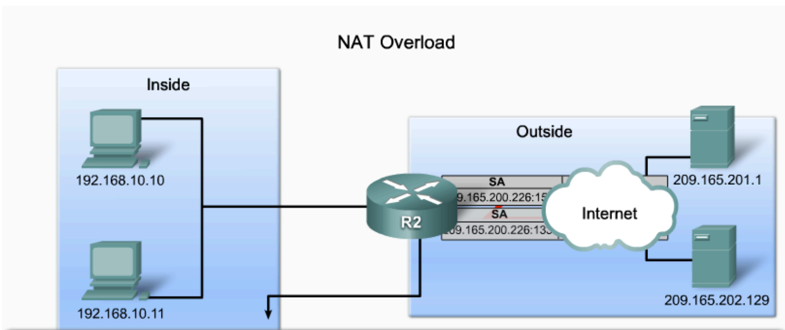
NAT: Translating inside local into global addresses from an address pool

NAT Overload



NAT overloading (sometimes called Port Address Translation or PAT) maps multiple private IP addresses to a single public IP address or a few addresses

NAT Overload

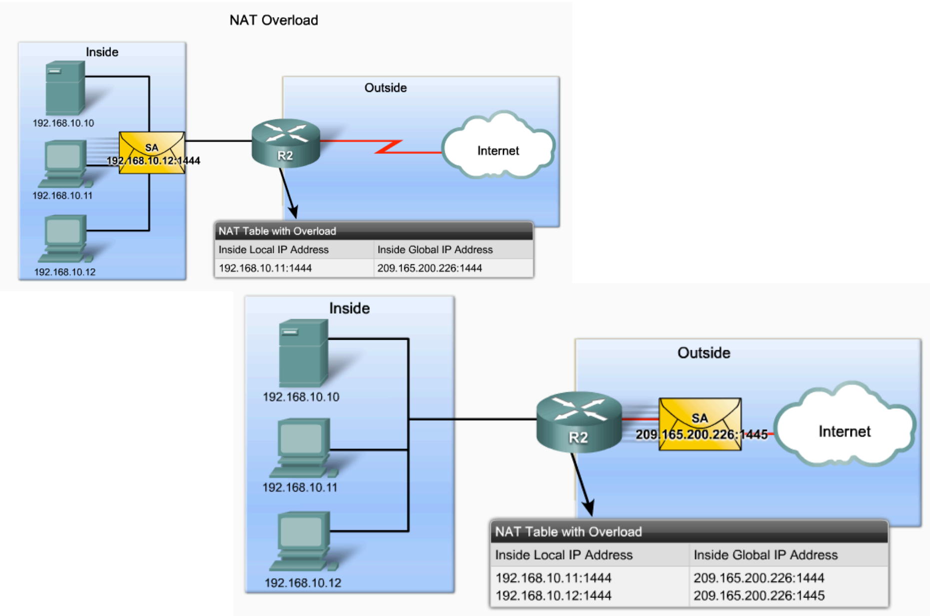


NAT Table with Overload

Inside Local IP Address	Inside Global IP Address	Outside Global IP Address	Outside Local IP Address
192.168.10.10:1555	209.165.200.226:1555	209.165.201.1:80	209.165.201.1:80
192.168.10.11:1331	209.165.200.226:1331	209.165.202.129:80	209.165.202.129:80

NAT overloading (sometimes called Port Address Translation or PAT) maps multiple private IP addresses to a single public IP address or a few addresses

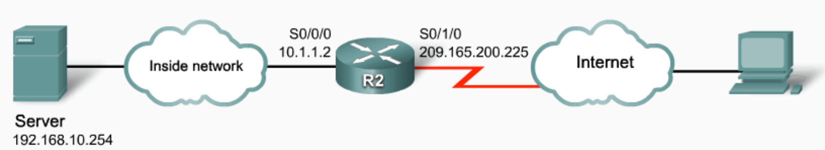
Next Available Port



Network Address Translation

- The NAT box translates the internal IP address into its own.
- What happens when the reply comes back addressed to the NAT box?
  - Almost all IP packets carry TCP or UDP segments and every segment carry source and destination port numbers (16 bit).
  - The NAT box overwrites the 16 bits of the source port in the packet with an index to an internal table (2^{16} entries) where the real source port and internal IP address are stored.
  - When the reply came back the internal table is checked, the source port and destination address is changed, the checksum is recomputed and the packet is forwarded to the destination.

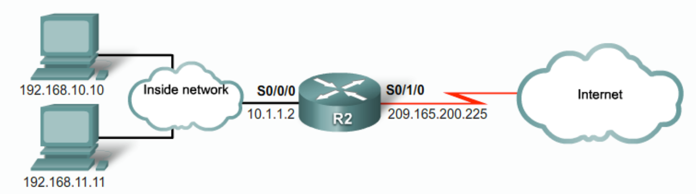
Static NAT Example




```
ip nat inside source static 192.168.10.254 209.165.200.254
!–Establishes static translation between an inside local address and an inside global address.
interface serial 0/0/0
ip nat inside
!—Identifies Serial 0/0/0 as an inside NAT interface.
interface serial 0/1/0
ip nat outside
!—Identifies Serial 0/1/0 as an outside NAT interface.
```

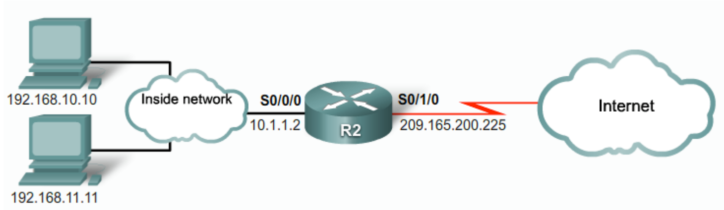
With this configuration, 192.168.10.254 will always translate to 209.165.200.254

Dynamic NAT Example




```
ip nat pool NAT-POOL1 209.165.200.226 209.165.200.240 netmask 255.255.255.224
!—Defines a pool of public IP addresses under the pool name NAT-POOL1
access-list 1 permit 192.168.0.0 0.0.255.255
!—Defines which addresses are eligible to be translated
ip nat inside source list 1 pool NAT-POOL1
!—Binds the NAT pool with ACL 1
interface serial 0/0/0
  ip nat inside
!—Identifies interface Serial 0/0/0 as an inside NAT interface
interface serial 0/1/0
  ip nat outside
!—Identifies interface Serial 0/1/0 as the outside NAT interface
```

Dynamic NAT (PAT) Example




```
access-list 1 permit 192.168.0.0 0.0.255.255
!—Defines which addresses are eligible to be translated
ip nat inside source list 1 interface serial 0/1/0 overload
!—Identifies the outside interface Serial 0/1/0 as the inside global address to be overloaded
interface serial 0/0/0
  ip nat inside
!—Identifies interface Serial 0/0/0 as an inside NAT interface
interface serial 0/1/0
  ip nat outside
!—Identifies interface Serial 0/1/0 as the outside NAT interface
```

NAT Statistics

```
R2#show ip nat translations
Pro Inside global          Inside local          Outside local         Outside global
tcp 209.165.200.225:16642  192.168.10.10:16642   209.165.200.254:80    209.165.200.254:80
tcp 209.165.200.225:62452  192.168.11.10:62452   209.165.200.254:80    209.165.200.254:80

R2#show ip nat translations verbose
Pro Inside global          Inside local          Outside local         Outside global
tcp 209.165.200.225:16642  192.168.10.10:16642   209.165.200.254:80    209.165.200.254:80
    create 00:01:45, use 00:01:43 timeout:86400000, left 23:58:16, Map-Id(In): 1,
    flags:
extended, use_count: 0, entry-id: 4, lc_entries: 0
tcp 209.165.200.225:62452  192.168.11.10:62452   209.165.200.254:80    209.165.200.254:80
    create 00:00:37, use 00:00:35 timeout:86400000, left 23:59:24, Map-Id(In): 1,
    flags:
extended, use_count: 0, entry-id: 5, lc_entries: 0
R2#
```

NAT debugging

```
R2# debug ip nat
IP NAT debugging is on
R2#
*Oct  6 19:55:31.579: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14434]
*Oct  6 19:55:31.595: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6334]
*Oct  6 19:55:31.611: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14435]
*Oct  6 19:55:31.619: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14436]
*Oct  6 19:55:31.627: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14437]
*Oct  6 19:55:31.631: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6335]
*Oct  6 19:55:31.643: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6336]
*Oct  6 19:55:31.647: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14438]
*Oct  6 19:55:31.651: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6337]
*Oct  6 19:55:31.655: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14439]
*Oct  6 19:55:31.659: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6338]

<Output omitted>
```

Network Address Translation

- Problems with NAT
  - Violates the holly grail of IP which states that every IP address uniquely identifies a single machine worldwide.
  - Changes Internet in a kind of connection oriented network.
  - If the NAT box crashes all connections are lost; this breaks a fundamental property of the Internet.
  - NAT violates the layer independence.
  - Hosts in the Internet are not required to use TCP or UDP; if a new transport protocol is used in a network with NAT the application will fail.
  - Some applications provide IP addresses in the payload of packets that are used for communication (FTP, H.323 Internet telephony) The NAT box knows nothing about this... and the applications may fail.