Internet Ideal: Simple Network Model

- Globally unique identifiers
  - Each node has a unique, fixed IP address
  - … reachable from everyone and everywhere
- Simple packet forwarding
  - Network nodes simply forward packets
  - … rather than modifying or filtering them
Internet Reality

- **Host mobility**
  - Changes in IP addresses as hosts move
- **IP address depletion**
  - Dynamic assignment of IP addresses
  - Private addresses (10.0.0.0/8, 192.168.0.0/16, …)
- **Security concerns**
  - Discarding suspicious or unwanted packets
  - Detecting suspicious traffic
- **Performance concerns**
  - Controlling how link bandwidth is allocated
  - Storing popular content near the clients

Middleboxes

- **Middleboxes are intermediaries**
  - Interposed in-between the communicating hosts
  - Often without knowledge of one or both parties
- **Myriad uses**
  - Address translators
  - Firewalls
  - Traffic shapers
  - Intrusion detectors
  - Transparent proxies
  - Application accelerators

“An abomination!”
- Violation of layering
- Hard to reason about
- Responsible for subtle bugs

“A practical necessity!”
- Solve real/pressing problems
- Needs not likely to go away
Network Address Translation

History of NATs

• IP address space depletion
  – Clear in early 90s that $2^{32}$ addresses not enough
• Key Ideas
  – Share addresses among numerous devices
  – … without requiring changes to existing hosts
• Meant to provide temporary relief
  – Intended as a short-term remedy
  – Now, NAT are very widely deployed
  – … much more so than IPv6 😊
**Active Component in the Data Path**

Problem: Local address not globally addressable

NAT rewrites the IP addresses
- Make “inside” look like single IP addr
- Change header checksums accordingly

**Private IP Network**

- Not directly connected to the Internet
- Uses *private (non-routable) IP addresses*:
  - Not registered and not guaranteed to be globally unique
  - Private IP address ranges:
    - 10.0.0.0 – 10.255.255.255 (/8)
    - 172.16.0.0 – 172.31.255.255 (/12)
    - 192.168.0.0 – 192.168.255.255 (/16)
Private Addresses

Basic Operation of NAT
What if Two Hosts Contact a Same Site?

• Suppose two hosts contact a same destination
  – e.g., both hosts open a socket with local port 3345 to
destination 128.119.40.186 on port 80

• NAT gives packets same source address
  – All packets have source address 138.76.29.7

• Problems
  – Can destination differentiate between senders?
  – Can return traffic get back to the correct hosts?

Port-Translating NAT

• Map outgoing packets
  – Replace source address with NAT address
  – Replace source port number with new port number
  – Remote hosts respond using (NAT address, new port #)

• Maintain a translation table
  – Map (source address, port #) to (NAT address, new port #)

• Map incoming packets
  – Consult the translation table
  – Map the destination address and port number
  – Local host receives the incoming packet
Network Address Translation Example

Maintaining the Mapping Table

- Create an entry upon seeing a packet
  - Packet with new (source addr, source port) pair
- Eventually, need to delete the map entry
  - But when to remove the binding?
- If no packets arrive within a time window
  - … then delete the mapping to free up the port #s
  - At risk of disrupting a temporarily idle connection
- Yet another example of “soft state”
  - i.e., removing state if not refreshed for a while
Where is NAT Implemented?

- **Home router (e.g., Linksys box)**
  - Integrates router, DHCP server, NAT, etc.
  - Use single IP address from the service provider
  - ... and have a bunch of hosts hiding behind it
- **Campus or corporate network**
  - NAT at the connection to the Internet
  - Share a collection of public IP addresses
  - Avoid complexity of renumbering end hosts and local routers when changing service providers

Practical Objections Against NAT

- **Port #s are meant to identify sockets**
  - Yet, NAT uses them to identify *end hosts*
  - Makes it hard to run a server behind a NAT
Running Servers Behind NATs

- Running servers is still possible
  - Admittedly with a bit more difficulty

- By explicit configuration of the NAT box
  - E.g., internal service at <dst 138.76.29.7, dst-port 80>
  - ... mapped to <dst 10.0.0.1, dst-port 80>

NAT can Do Load Balancing of Servers

Split load over server replicas
(At the connection level)

Apply load balancing policies
Load Balancers

Replicated Servers

- One site, many servers
  - www.youtube.com
Firewalls

Firewall filters packet-by-packet, based on:

- Source and destination IP addresses and port numbers
- TCP SYN and ACK bits; ICMP message type
- Deep packet inspection of packet contents (DPI)
Packet Filtering Examples

• Block all packets with IP protocol field = 17 and with either source or dest port = 23.
  – All incoming and outgoing UDP flows blocked
  – All Telnet connections are blocked

• Block inbound TCP packets with SYN but no ACK
  – Prevents external clients from making TCP connections with internal clients
  – But allows internal clients to connect to outside

• Block all packets with TCP port of Quake

Firewall Configuration

• Firewall applies a set of rules to each packet
  – To decide whether to permit or deny the packet

• Each rule is a test on the packet
  – Comparing IP and TCP/UDP header fields
  – … and deciding whether to permit or deny

• Order matters
  – Once the packet matches a rule, the decision is done
Firewall Configuration Example

- Alice runs the network 222.22.0.0/16
  - Wants to let Bob’s school access certain hosts
    - Bob is on 111.11.0.0/16
    - Alice’s special hosts on 222.22.22.0/24
  - Alice doesn’t trust Trudy, a guy inside Bob’s network
    - Trudy is on 111.11.0.0/24
  - Alice wants no other traffic from Internet
- Rules
  - #1: Don’t let Trudy’s machines in
    - Deny (src = 111.11.11.0/24, dst = 222.22.0.0/16)
  - #2: Let rest of Bob’s network in to special dsts
    - Permit (src=111.11.0.0/16, dst = 222.22.22.0/24)
  - #3: Block the rest of the world
    - Deny (src = 0.0.0.0/0, dst = 0.0.0.0/0)

Stateful Firewall

- Stateless firewall:
  - Treats each packet independently
- Stateful firewall
  - Remembers connection-level information
  - E.g., client initiating connection with a server
  - … allows the server to send return traffic
Firewall Implementation Challenges

- **Per-packet handling**
  - Must inspect every packet
  - Challenging on very high-speed links
- **Complex filtering rules**
  - May have large # of rules
  - May have very complicated rules
- **Location of firewalls**
  - Complex firewalls near the edge, at low speed
  - Simpler firewalls in the core, at higher speed

Clever Users Subvert Firewalls

- **Example: filtering dorm access to a server**
  - Firewall rule based on IP addresses of dorms
  - … and the server IP address and port number
  - Problem: users may log in to another machine
  - E.g., connect from the dorms to another host
  - … and then onward to the blocked server
LAN Appliances
aka WAN Accelerators
aka Application Accelerators

At Connection Point to the Internet

• Improve performance between edge networks
  – E.g., multiple sites of the same company
  – Through buffering, compression, caching, …

• Incrementally deployable
  – No changes to the end hosts or the rest of the Internet
  – Inspects the packets as they go by, and takes action

40
Example: Improve TCP Throughput

- Appliance with a lot of local memory
- Sends ACK packets quickly to the sender
- Overwrites the receive window with a large value
- Or, even run a new and improved version of TCP

Example: Compression

- Compress the packet
- Send the compressed packet
- Un-compress at the other end
Example: Caching

- Server sends object and pointer referring to object
- Client caches copy of object and pointer
- On new request of past object, server checks for changes to the data
- If no change, just send a pointer to the past object

Example: Encryption

- Two sites share keys for encrypting traffic
- Sending appliance encrypts the data
- Receiving appliance decrypts the data
- Protects the sites from snoopers on the Internet
Tunneling

IP Tunneling

• IP tunnel is a virtual point-to-point link
  – Illusion of a direct link between two nodes

  Logical view: A ——— Tunnel ——— E

  Physical view: A ——— B ——— E ——— F

• Encapsulation of the packet inside IP datagram
  – Node B sends a packet to node E
  – ... containing another packet as the payload
6Bone: Deploying IPv6 over IP4

Remote Access Virtual Private Network

- Tunnel from user machine to VPN server
  - A “link” across the Internet to the local network
- Encapsulates packets to/from the user
  - Packet from 12.1.1.73 to 12.1.1.100
  - Inside a packet from 1.2.3.4 to 12.1.1.1
Conclusions

• **Middleboxes address important problems**
  – Getting by with fewer IP addresses
  – Blocking unwanted traffic
  – Making fair use of network resources
  – Improving end-to-end performance

• **Middleboxes cause problems of their own**
  – No longer globally unique IP addresses
  – No longer can assume network simply delivers packets