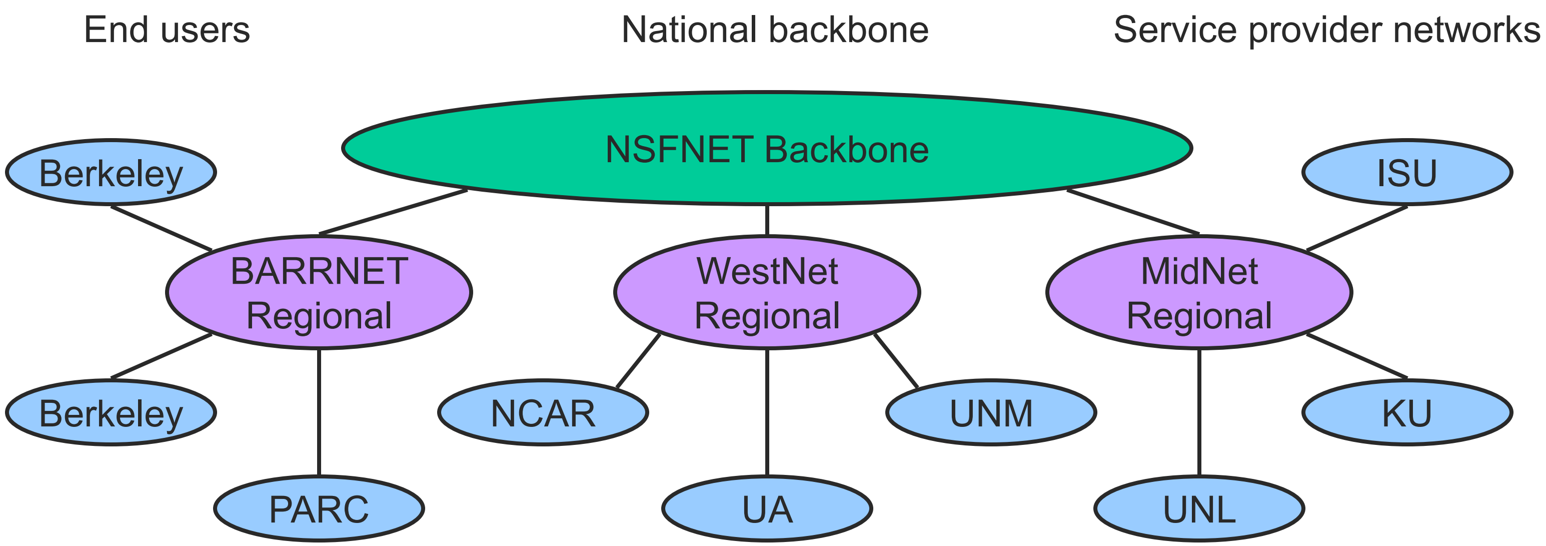
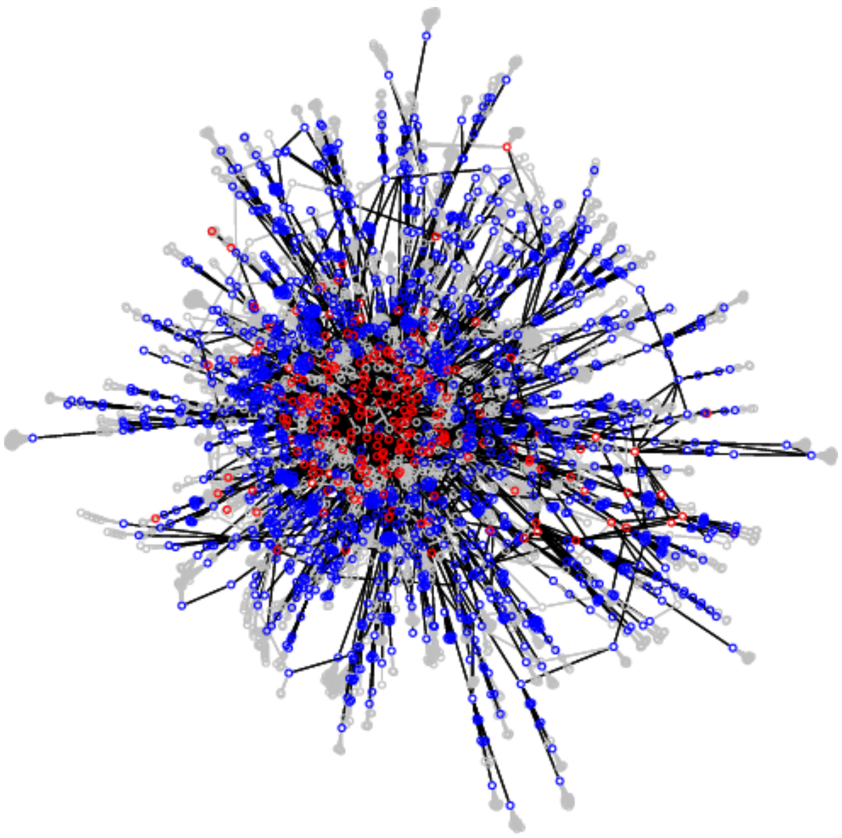
IP Addressing

Evolution of Internet Structure

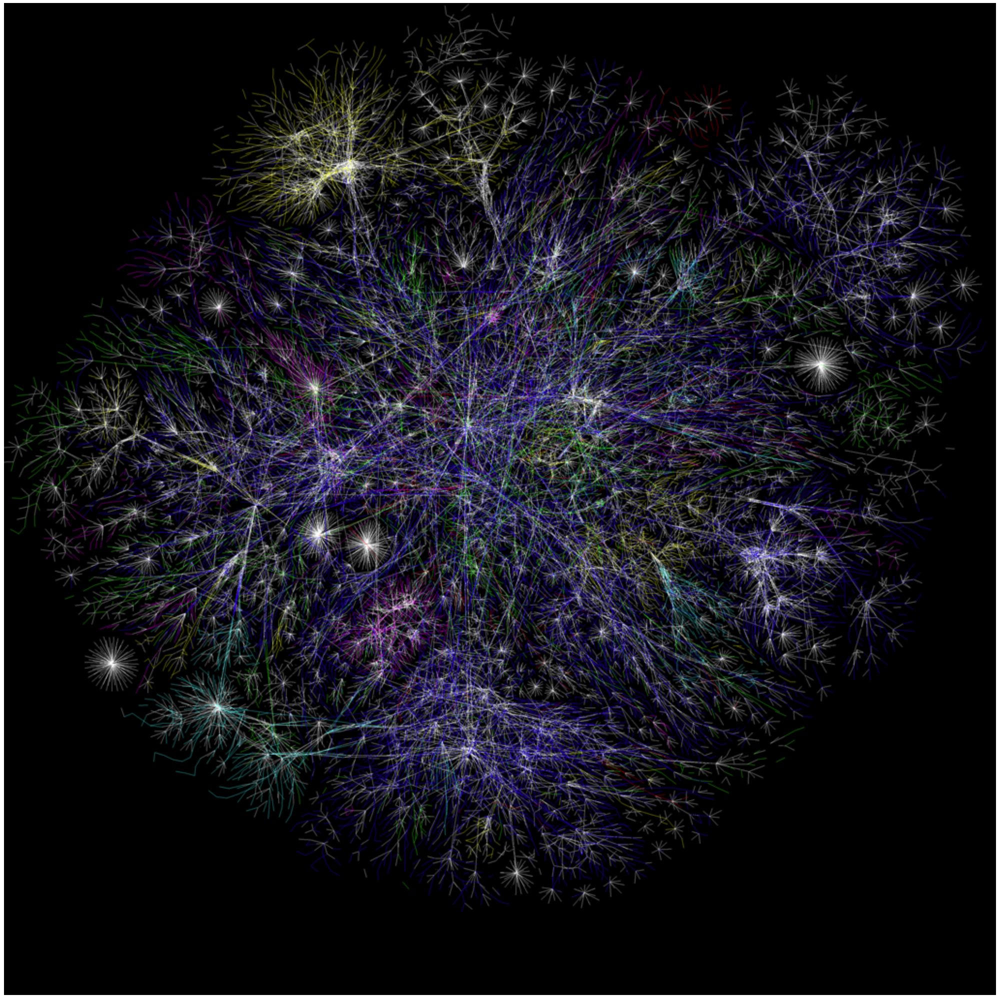
- Internet c. 1990
  - Tree structure, centered around one backbone
  - National Science Foundation (NSF) funded



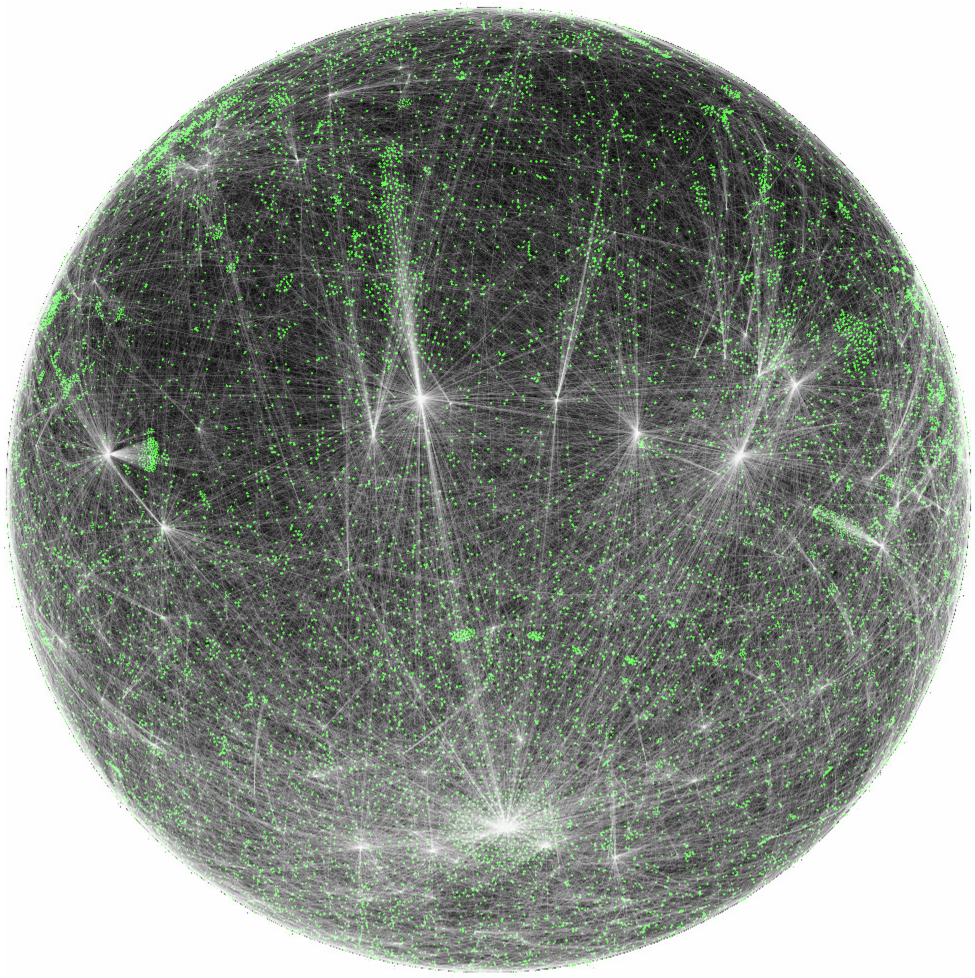
An Old Internet ISP Map



A New Internet Map

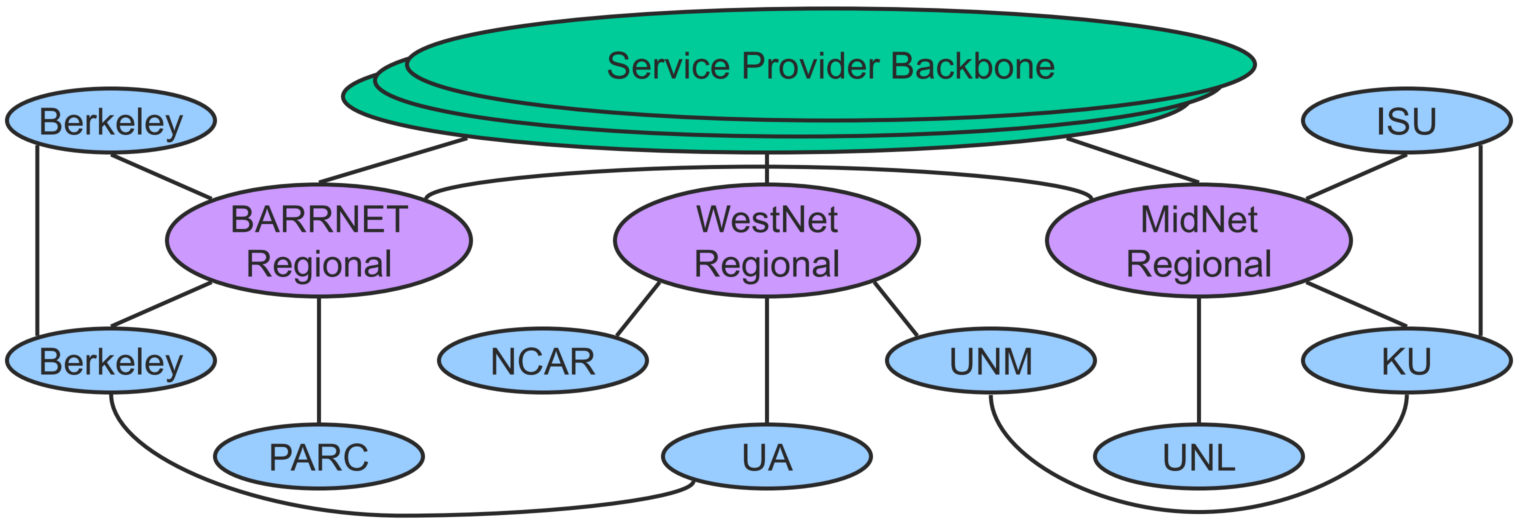


Another Internet Map



Evolution of Internet Structure

- Today
  - Multiple backbone service providers
  - Arbitrary graph structure



Problems of Scale

- Main problems
  - Inefficient address allocation
  - Too many networks for routing

IPv4 Address Model

- Properties
  - 32-bit address
  - Hierarchical
    - Network, subnet, host hierarchy
  - Maps to logically unique network adaptor
    - Exceptions: service request splitting for large web servers
- Three Class Model

Class A:	0	Network (7 bits)	Host (24 bits)		
Class B:	1	0	Network (14 bits)	Host (16 bits)	
Class C:	1	1	0	Network (21 bits)	Host (8 bits)

IPv4 Address Model

Class	Network ID	Host ID	# of Addresses	# of Networks
A	0 + 7 bit	24 bit	2^{24}-2	126
B	10 + 14 bit	16 bit	65,536 - 2	2^{14}
C	110 + 21 bit	8 bit	256 - 2	2^{21}
D	1110 + Multicast Address		IP Multicast	
E	Future Use			

Basic Datagram Forwarding with IP

- Hosts and routers maintain forwarding tables
  - List of <prefix, next hop> pairs
    - IP = 69.2.1.2 = 01000101 00000010 00000001 00000010
    - 24-bit prefix = 69.2.1.0/24
      = 01000101 00000010 00000001 ********
  - Often contains a default route
    - Pass unknown destination to provider ISP
  - Simple and static on hosts, edge routers
    - Complex and dynamic on core routers

Basic Datagram Forwarding with IP

- Packet forwarding
  - Compare network portion of address with <network/host, next hop> pairs in table
    - Send directly to host on same network
    - Send to indirectly (via router on same network) to host on different network
  - Use ARP to get hardware address of host/router

IPv4 Address Model

- IP addresses
  - Host in class A network
    - 56.0.78.100 www.usps.gov
  - Host in class B network
    - 128.174.252.1 www.cs.uiuc.edu
  - Host in class C network
    - 198.182.196.56 www.linux.org
- Questions
  - What networks should be allocated to a company with 1000 machines?
  - What about a company with 100 machines?
  - What about a company with 2 machines that plans to grow rapidly?

Problems of Scale

- Pressure mostly on class B networks
  - Most companies plan to grow beyond 255 machines
  - Renumbering is time consuming and can interrupt service
  - Approximately 16,000 class B networks available
- Class B networks aren't very efficient
  - Few organizations have O(10,000) machines
  - More likely use O(1,000) of the 65,000 addresses
- Scaling problems with alternatives
  - Multiple class C networks
    - Routing tables don't scale
  - Protocols do not scale beyond O(10,000) networks

IP Address Hierarchy Evolution

- Began with class based system
  - Subnetting within an organization
    - Network can be broken into smaller networks
    - Recognized only within the organization
    - Implemented by packet switching
    - Smaller networks called subnets

Class A:

0	**Network (7 bits)**	**Host (24 bits)**

Class B:

1	**0**	**Network (14 bits)**	**Host (16 bits)**

Class C:

1	**1**	**0**	**Network (21 bits)**	**Host (8 bits)**

Subnetting

- Simple IP
  - All hosts on the same network must have the same *network* number
- Assumptions
  - Subnets are close together
    - Look like one network to distant routers
- Idea
  - Take a single IP network number
  - Allocate the IP addresses to several physical networks (subnets)
- Subnetting
  - All hosts on the same network must have the same *subnet* number

Subnetting

- Enables a domain to further partition address space into smaller networks
  - Subdivide host id into subnet ID + host ID
  - Subnet mask
- Only routers in the domain interpret subnet mask
  - Other routers treat IP address as normal class A, B or C address

Subnet Example

- Consider
  - A domain with a class B address
  - 135.104.*
- Without subnetting
  - Every router in the domain needs to know how to route to every host
- However
  - the domain itself is likely organized as a hierarchy of physical networks

Subnet Example

- Solution
  - Partition the 65,536 address in the class B network
    - 256 subnets each with 256 addresses
    - Subnet mask: 255.255.255.0
  - If 135.104.5.{1,2,3} are all on the same physical network reachable from router 135.105.4.1
    - There only needs to be one routing entry for 135.104.5.* pointing to 135.105.4.1 as next hop

Subnetting

- Normal IP

Class B:

1	0	**Network (14 bits)**	**Host (16 bits)**

- Typical subnetting example
  - Use first byte of host as subnet number

Class B:

1	0	**Network (14 bits)**	**Subnet (8 bits)**	**Host (8 bits)**

- Atypical example
  - Non-contiguous 6-bit subnet number

Class B:

1	0	**Network (14 bits)**								

Subnetting

- The subnet mask specifies the bits of network and subnet addresses
- Routing table entries carry both addresses and subnet masks

Class B:

1	**0**	**Network (14 bits)**	**Host (16 bits)**

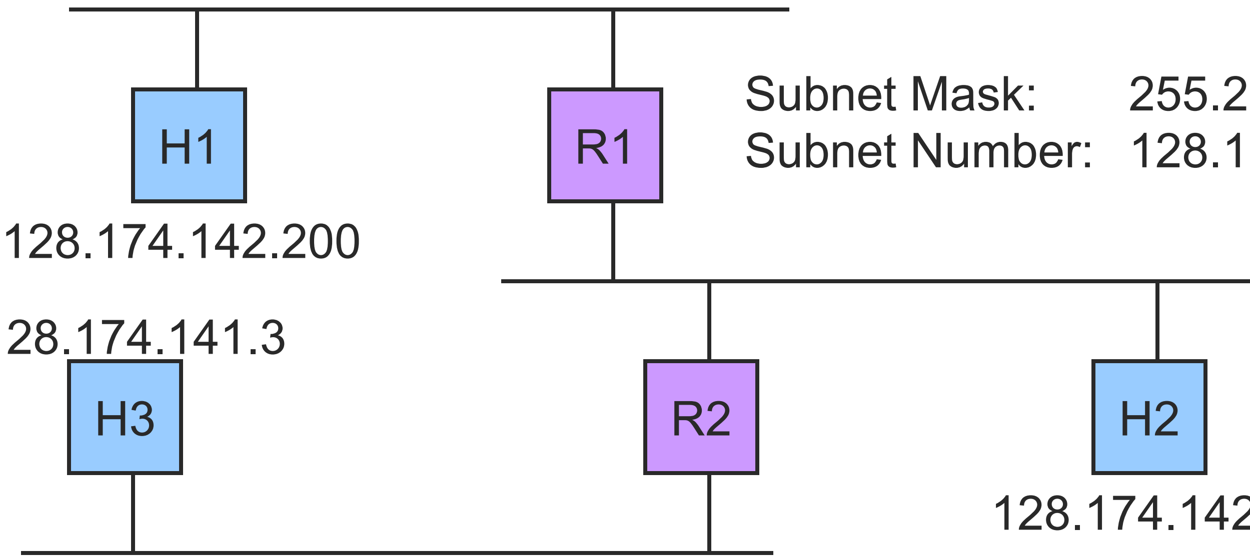
Class B:

1	**0**	**Network (14 bits)**	**Subnet (8 bits)**	**Host (8 bits)**

Subnet Mask:

1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Subnetting – Host 1



Subnet Mask: 255.255.255.128
Subnet Number: 128.174.142.128

Host 1: 128.174.142.200

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	1	1	0	0	1	0	0	0

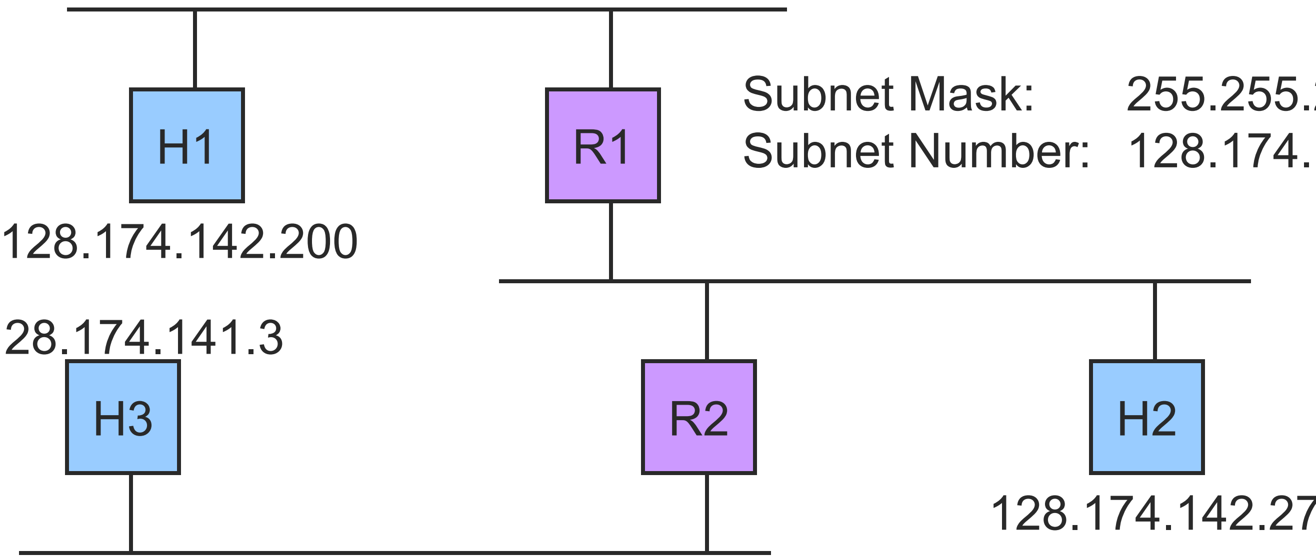
Subnet Mask 255.255.255.128

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0

Subnet # 128.174.142.128

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0

Subnetting – Host 3



Subnet Mask: 255.255.255.0
Subnet Number: 128.174.141.0

Host 3: 128.174.141.3

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	1	1

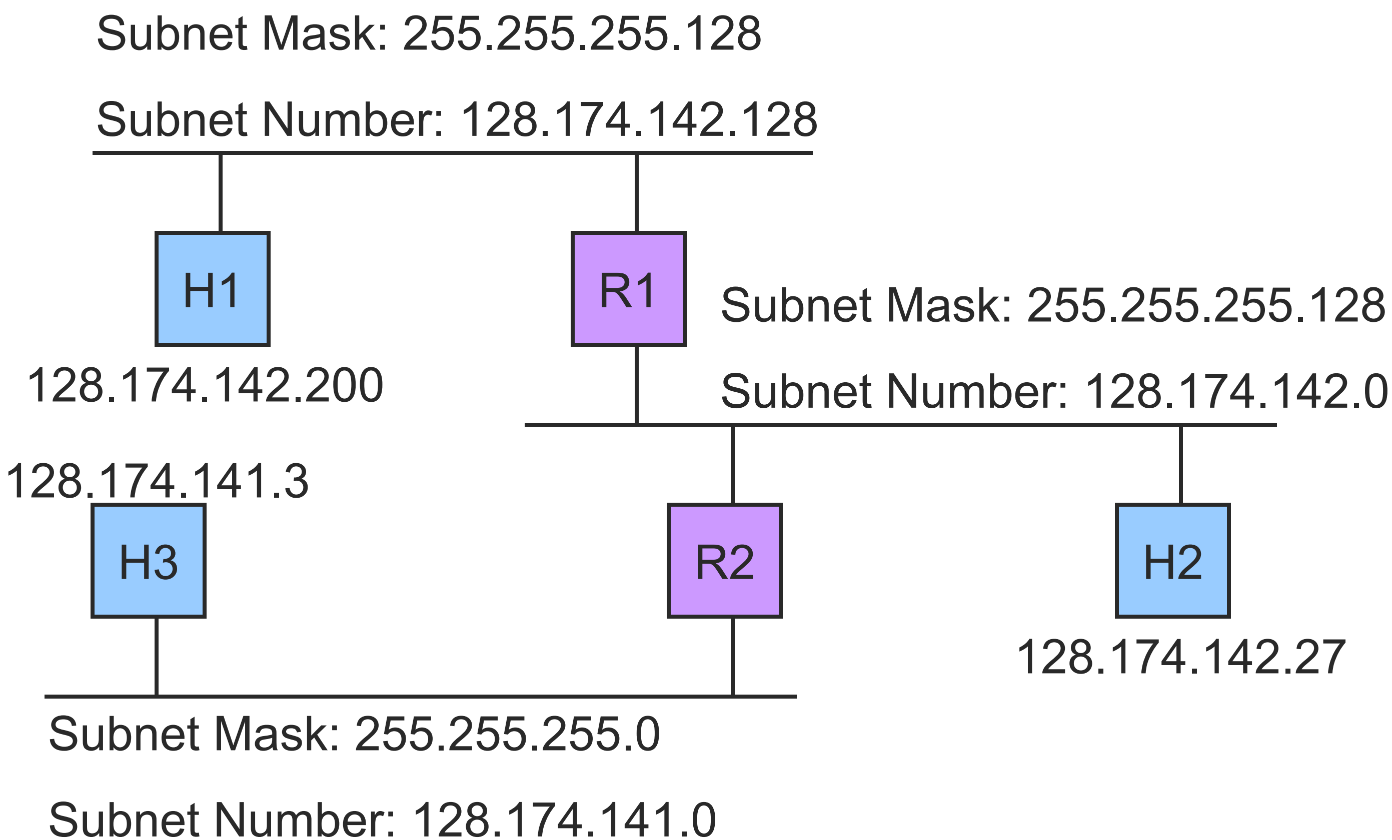
Subnet Mask 255.255.255.0

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Subnet # 128.174.141.0

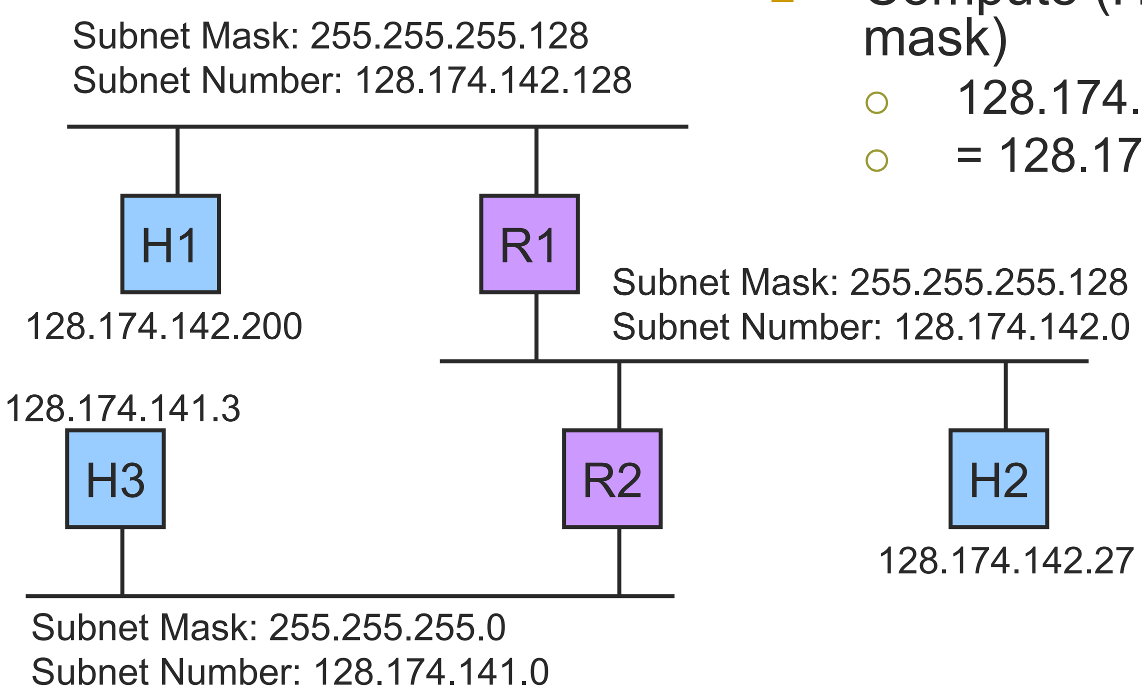
1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0

Subnetting - Example



Subnetting

Send from H1 to H3



- At H1:
- Compute (H3 AND H1's subnet mask)
  - 128.174.141.3 **AND** 255.255.255.128
  - = 128.174.141.0 (≠ 128.174.142.128)
- If result == H1's subnet number
  - H3 and H1 are on the same subnet
- else
  - route through appropriate router

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

```
196 = 1100 0100

128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196 to R1
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

`196 = 1100 0100`

```
128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

```
95 = 0101 1111
128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

```
95 = 0101 1111
```

```
128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

to Interface 1

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

```
137 = 1000 1001
128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

`137 = 1000 1001`

to Interface 0

```
128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

```
18 = 0001 0010
128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

`18 = 0001 0010`

to R3

`128 = 1000 0000`
`141 = 1000 1101`
`142 = 1000 1110`
`145 = 1001 0001`
`196 = 1100 0100`

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15

```
15 = 0000 1111
```

```
128 = 1000 0000
141 = 1000 1101
142 = 1000 1110
145 = 1001 0001
196 = 1100 0100
```

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196
    - 128.174.142.95
    - 128.174.141.137
    - 128.174.145.18
    - 131.126.244.15 to R3

`15 = 0000 1111`

`128 = 1000 0000`
`141 = 1000 1101`
`142 = 1000 1110`
`145 = 1001 0001`
`196 = 1100 0100`

Routing with Subnetting

Subnet #	Subnet Mask	Next Hop
128.174.141.0	255.255.255.0	Interface 0
128.174.142.0	255.255.255.128	Interface 1
128.174.142.128	255.255.255.128	R1
128.174.0.0	255.255.0.0	R3
Default	0.0.0.0	R3

- Example Table from R2
  - Next hop
    - 128.174.142.196 to R1
    - 128.174.142.95 to Interface 1
    - 128.174.141.137 to Interface 0
    - 128.174.145.18 to R3
    - 131.126.244.15 to R3

Subnetting

- Notes
  - Non-contiguous subnets are difficult to administer
  - Multiple subnets on one physical network
    - Must be routed through router
- Pros
  - Helps address consumption
  - Helps reduce routing table size

The Crisis

- Fixed 32-bit address space for IPv4
- Network allocation based on Classic A, B, C Model
- Central allocation authority
  - Randomly assigning addresses
- Problems
  - Router table explosion
  - Address space exhaustion

Classless Interdomain Routing (CIDR)

- CIDR/Supernetting
  - Problem with subnetting
    - Allows hierarchy within organizations
    - Does not reduce class B address space pressure
  - Solution
    - Aggregate routes in routing tables
    - Eliminate class notation
    - Generalize subnet notion
    - Allow only contiguous subnet masks
    - Specify network by <network #, # of bits in subnet mask>
    - Equivalent to <network #, # of hosts>
    - Blocks of class C networks can now be treated as one network

CIDR

- Route aggregation
  - Use contiguous blocks of Class C addresses
    - Example:
      - 192.4.16 – 192.4.31
      - 20 bit subnet mask
    - Block size must be a power of 2
  - Network number may be any length

192.4.16.0

1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

192.4.31.0

1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0

Subnet Mask

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

CIDR

Subnet # / length	Next Hop
128.174.141.0 / 24	Interface 0
128.174.142.192 / 27	Interface 1
128.174.142.128 / 25	R1
128.174.0.0 / 16	R3
Default	R3

- CIDR is similar to subnetting
  - Trend is for increasing amounts of overlap in routing table entries
  - Example: 128.174.142.200
    - Matches second, third and fourth lines
    - Route to entry with longest match

CIDR

Subnet: 128.174.141.0

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0

Subnet Mask length = 24 (255.255.255.0)

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Host: 128.174.142.200

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	1	1	0	0	1	0	0	0

Resulting Subnet Number: 128.174.142.0 (≠ 128.174.141.0)

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0

Subnet: 128.174.142.192

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	1	1	0	0	0	0	0	0

Subnet Mask length = 27 (255.255.255.224)

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0

Host: 128.174.142.200

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	1	1	0	0	1	0	0	0

Resulting Subnet Number: 128.174.142.192 (= 128.174.142.192)

1	0	0	0	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	0	1	1	1	0	1	1	0	0	0	0	0	0

CIDR

Subnet: 128.174.142.128

1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 1 0 0 0 1 1 1 0 1 0 0 0 0 0 0 0

Subnet Mask length = 25 255.255.255.192

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0

Host: 128.174.142.200

1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 1 0 0 0 1 1 1 0 1 1 0 0 1 0 0 0

Resulting Subnet Number: 128.174.142.128 (= 128.174.142.128)

1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 1 0 0 0 1 1 1 0 1 0 0 0 0 0 0 0

Subnet: 128.174.0.0

1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Subnet Mask length = 16 255.255.0.0

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Host: 128.174.142.200

1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 1 0 0 0 1 1 1 0 1 1 0 0 1 0 0 0

Resulting Subnet Number: 128.174.0.0 (= 128.174.0.0)

1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

CIDR

- Subnetting
  - Share one address (network number) across multiple physical networks
- Supernetting
  - Aggregate multiple addresses (network numbers) for one physical network

CIDR

- Allows hierarchical development
  - Assign a block of addresses to a regional provider
    - Ex: 128.0.0.0/9 to BARRNET
  - Regional provider subdivides address and hands out block to sub-regional providers
    - Ex: 128.132.0.0/16 to Berkeley
  - Sub-regional providers can divide further for smaller organizations
    - Ex: 128.132.32.0/1 to Berkeley Computer Science Department

Pros and Cons

- Provides a fast easy solution
- Was not intended to be permanent
- Multihomed sites cannot benefit from aggregation
- Not backward compatible

IPv6

- History
  - Next generation IP (AKA IPng)
  - Intended to extend address space and routing limitations of IPv4
    - Requires header change
    - Attempted to include everything new in one change
  - IETF moderated
    - Based on Simple Internet Protocol Plus (SIPP)

IPv6

- Wish list
  - 128-bit addresses
  - Multicast traffic
  - Mobility
  - Real-time traffic/quality of service guarantees
  - Authentication and security
  - Autoconfiguration for local IP addresses
  - End-to-end fragmentation
  - Protocol extensions
- Smooth transition!
- Note
  - Many of these functionalities have been retrofit into IPv4

IPv6 Addresses

- 128-bit
  - 3.4 x 1038 addresses (as compared to 4 x 109
- Classless addressing/routing (similar to CIDR)
- Address notation
  - String of eight 16-bit hex values separated by colons
    - 5CFA:0002:0000:0000:CF07:1234:5678:FFCD
  - Set of contiguous 0’s can be elided
    - 5CFA:0002::0000:CF07:1234:5678:FFCD
- Address assignment
  - Provider-based
  - geographic

3	m	n	o	p	125-m-n-o-p
010	**Region ID**	**Provider ID**	**Subscriber ID**	**Subnet**	**Host**

IPv6

Prefix	Address type
0000 0000	Reserved (includes transition addresses)
0000 0001	ISO NSAP (Network Service Point) Allocation
0000 010	Novell IPX allocation
010	Provider-based unicast
100	Geographic multicast
1111 1110 10	Link local address
1111 1110 11	Site local address
1111 1111	Multicast address
Other	unassigned

IPv4 Packet Format

- 20 Byte minimum
- Mandatory fields are not always used
  - e.g. fragmentation
- Options are an unordered list of (name, value) pairs

<table>
<tr><th colspan="2">0</th><th colspan="2">8</th><th colspan="3">16</th><th>31</th></tr>
<tr><td>version</td><td>hdr len</td><td colspan="2">TOS</td><td colspan="4">length</td></tr>
<tr><td colspan="4">ident</td><td>flags</td><td colspan="3">offset</td></tr>
<tr><td colspan="2">TTL</td><td colspan="2">protocol</td><td colspan="4">checksum</td></tr>
<tr><td colspan="8">source address</td></tr>
<tr><td colspan="8">destination address</td></tr>
<tr><td colspan="6">options (variable)</td><td colspan="2">pad (variable)</td></tr>
</table>

IPv6 Packet Format

0	8	16	31
version	priority	flow label	
payload length		next header	hop limit
source address word 1			
source address word 2			
source address word 3			
source address word 4			
destination address word 1			
destination address word 2			
destination address word 3			
destination address word 4			
options (variable number, usually fixed length)			

IPv6 Packet Format

- 40 Byte minimum
- Mandatory fields (almost) always used
- Strict order on options reduces processing time
  - No need to parse irrelevant options

<table>
<tr><th>0</th><th></th><th>8</th><th>16</th><th>31</th></tr>
<tr><td>version</td><td>priority</td><td colspan="3">flow label</td></tr>
<tr><td colspan="3">payload length</td><td>next header</td><td>hop limit</td></tr>
<tr><td colspan="5">source address 4 words</td></tr>
<tr><td colspan="5">destination address 4 words</td></tr>
<tr><td colspan="5">options (variable number, usually fixed length)</td></tr>
</table>

IPv6 Packet Format

- Version
  - 6
- Priority and Flow Label
  - Support service guarantees
  - Allow “fair” bandwidth allocation
- Payload Length
  - Header not included
- Next Header
  - Combines options and protocol
  - Linked list of options
  - Ends with higher-level protocol header (e.g. TCP)
- Hop Limit
  - TTL renamed to match usage

IPv6 Extension Headers

- Must appear in order
  - Hop-by-hop options
    - Miscellaneous information for routers
  - Routing
    - Full/partial route to follow
  - Fragmentation
    - IP fragmentation info
  - Authentication
    - Sender identification
  - Encrypted security payload
    - Information about contents
  - Destination options
    - Information for destination

IPv6 Extension Headers

- Hop-by-Hop extension
  - Length is in bytes beyond mandatory 8

0	8	16	31
next header	length	type	
value			

- Jumbogram option (packet longer than 65,535 bytes)
  - Payload length in main header set to 0

0	8	16	31
next header	0	194	0
Payload length in bytes			

IPv6 Extension Headers

0	8	16	31
next header	0	# of addresses	next address
	strict/loose routing bitmap		
1 – 24 addresses			

- Routing extension
  - Up to 24 “anycast” addresses target AS’ s/providers
  - Next address tracks current target
  - Strict routing requires direct link
  - Loose routing allows intermediate nodes

IPv6 Extension Headers

0	8	16		31
next header	reserved	offset	reserved	M
ident				

- Fragmentation extension
  - Similar to IPv4 fragmentation
    - 13-bit offset
    - Last fragment mark (M)
  - Larger fragment identification field

IPv6 Extension Headers

- Authentication extension
  - Designed to be very flexible
  - Includes
    - Security parameters index (SPI)
    - Authentication data
- Encryption Extension
  - Called encapsulating security payload (ESP)
  - Includes an SPI
  - All headers and data after ESP are encrypted

IPv6 Design Controversies

- Address length
  - 8 byte
    - Might run out in a few decades
    - Less header overhead
  - 16 byte
    - More overhead
    - Good for foreseeable future
  - 20 byte
    - Even more overhead
    - Compatible with OSI
  - Variable length

IPv6 Design Controversies

- Hop limit
  - 65,535
    - 32 hop paths are common now
    - In a decade, we may see much longer paths
  - 255
    - Objective is to limit lost packet lifetime
    - Good network design makes long paths unlikely
      - Source to backbone
      - Across backbone
      - Backbone to destination

IPv6 Design Controversies

- Greater than 64KB data
  - Good for supercomputer/high bandwidth applications
  - Too much overhead to fragment large data packets
- 64 KB data
  - More compatible with low-bandwidth lines
  - 1 MB packet ties up a 1.5MBps line for more than 5 seconds
  - Inconveniences interactive users

IPv6 Design Controversies

- Keep checksum
  - Removing checksum from IP is analogous to removing brakes from a car
    - Light and faster
    - Unprepared for the unexpected
- Remove checksum
  - Typically duplicated in data link and transport layers
  - Very expensive in IPv4

IPv6 Design Controversies

- Mobile hosts
  - Direct or indirect connectivity
    - Reconnect directly using canonical address
    - Use home and foreign agents to forward traffic
  - Mobility introduces asymmetry
    - Base station signal is strong, heard by mobile units
    - Mobile unit signal is weak and susceptible to interference, may not be heard by base station

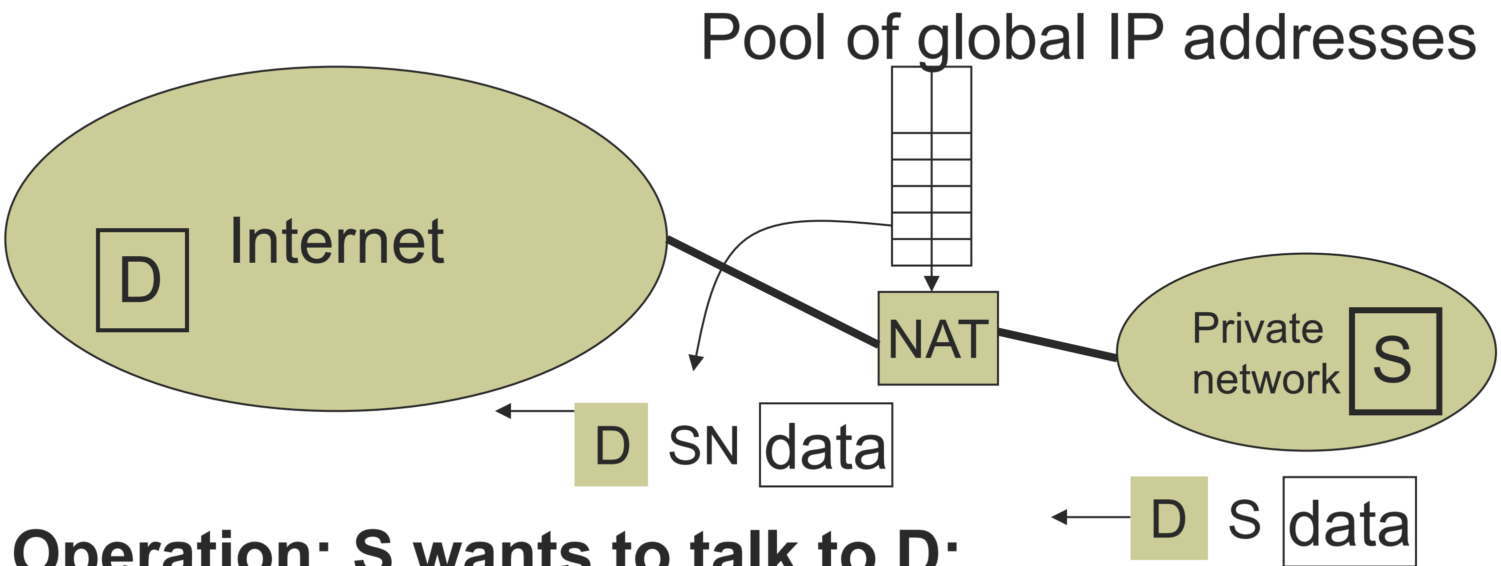
IPv6 Design Controversies

- Security
  - Where?
    - Network layer
      - A standard service
    - Application layer
      - No viable standard
      - Application susceptible to errors in network implementation
      - Expensive to turn on and off
  - How?
    - Political import/export issues
    - Cryptographic strength issues

Network Address Translation (NAT)

- Kludge (but useful)
- Sits between your network and the Internet
- Translates local network layer addresses to global IP addresses
- Has a pool of global IP addresses (less than number of hosts on your network)

NAT Illustration



Operation: S wants to talk to D:

- Create S-SN mapping
- Replace S with SN for outgoing packets
- Replace SN with S for incoming packets

What if we only have few (or just one) IP address?

- Use NAPT (Network Address Port Translator)
- NAPT translates:
  - <Paddr1, portA> to <Gaddr, portB>
  - potentially thousands of simultaneous connections with one global IP address

Problems with NAT

- Hides the internal network structure
  - some consider this an advantage
- Multiple NAT hops must ensure consistent mappings
- Some protocols carry addresses
  - e.g., FTP carries addresses in text
  - what is the problem?
- Encryption

NAT: Network Address Translation

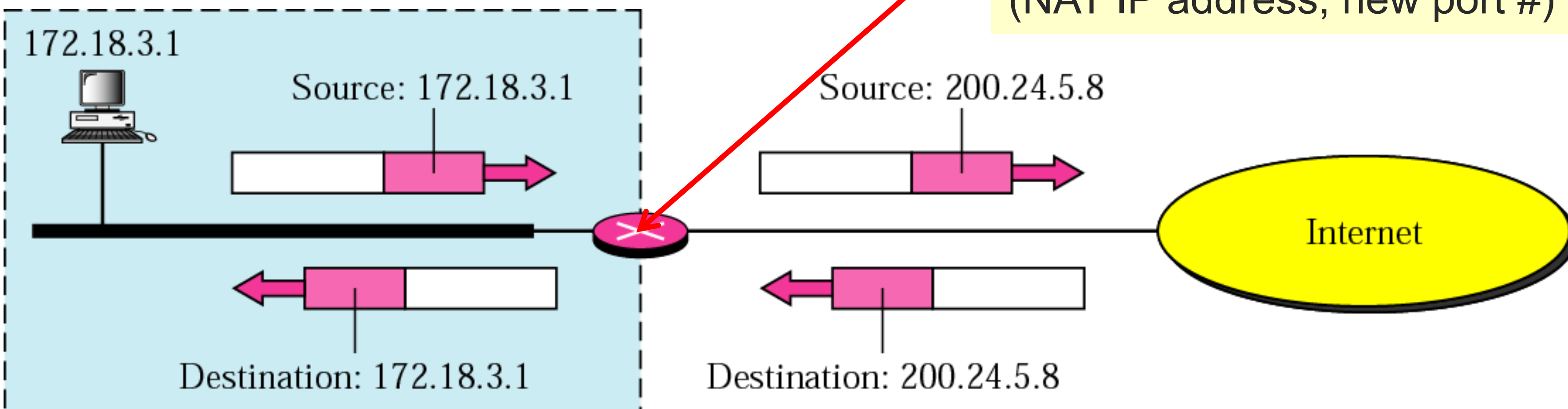
- Approach
  - Assign one router a global IP address
  - Assign internal hosts local IP addresses
- Change IP Headers
  - IP addresses (and possibly port numbers) of IP datagrams are replaced at the boundary of a private network
  - Enables hosts on private networks to communicate with hosts on the Internet
  - Run on routers that connect private networks to the public Internet

NAT: Network Address Translation

What address do the remote hosts respond to?

- Outgoing packet
  - Source IP address (private IP) replaced by global IP address maintained by NAT router
- Incoming packet
  - Destination IP address (global IP of NAT router) replaced by appropriate private IP address

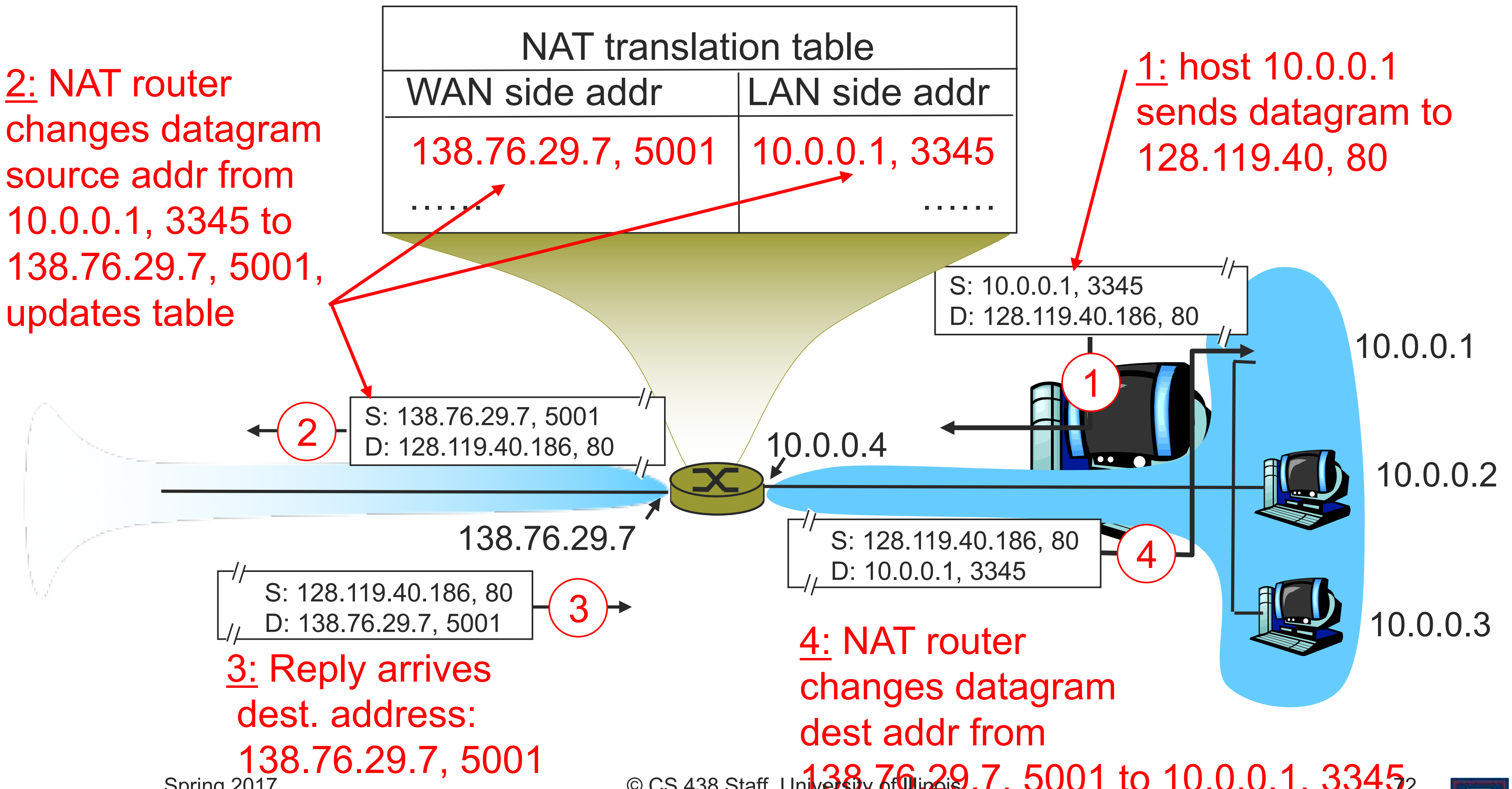
NAT router caches translation table:
(source IP address, port #) ➔ (NAT IP address, new port #)



NAT: Network Address Translation

- Benefits: local network uses just one (or a few) IP address as far as outside word is concerned
  - No need to be allocated range of addresses from ISP
    - Just one IP address is used for all devices
  - Can change addresses of devices in local network without notifying outside world
  - Can change ISP without changing addresses of devices in local network
  - Devices inside local net not explicitly addressable, visible by outside world (a security plus)

NAT: Network Address Translation



NAT: Network Address Translation

- Address Pooling
  - Corporate network has many hosts
  - Only a small number of public IP addresses
- NAT solution
  - Manage corporate network with a private address space
  - NAT, at boundary between corporate network and public Internet, manages a pool of public IP addresses
  - When a host from corporate network sends an IP datagram to a host in public Internet, NAT picks a public IP address from the address pool, and binds this address to the private address of the host

NAT: Network Address Translation

- Load balancing
  - Balance the load on a set of identical servers, which are accessible from a single IP address
- NAT solution
  - Servers are assigned private addresses
  - NAT acts as a proxy for requests to the server from the public network
  - NAT changes the destination IP address of arriving packets to one of the private addresses for a server
  - Balances load on the servers by assigning addresses in a round-robin fashion

NAT: Consequences

- 16-bit port-number field
  - 60,000 simultaneous connections with a single LAN-side address!
- End-to-end connectivity
  - NAT destroys universal end-to-end reachability of hosts on the Internet
  - A host in the public Internet often cannot initiate communication to a host in a private network
  - The problem is worse, when two hosts that are in different private networks need to communicate with each other

NAT: Consequences

- Performance
  - Modifying the IP header by changing the IP address requires that NAT boxes recalculate the IP header checksum
  - Modifying port number requires that NAT boxes recalculate TCP checksum
- Fragmentation
  - Datagrams fragmented before NAT device must not be assigned different IP addresses or different port numbers

NAT: Consequences

- IP address in application data
  - Applications often carry IP addresses in the payload of the application data
  - No longer work across a private-public network boundary
  - Hack: Some NAT devices inspect the payload of widely used application layer protocols and, if an IP address is detected in the application-layer header or the application payload, translate the address according to the address translation table