IP Subnetting, IPv6, RIP

1 January 2025
Lecture 8

Topics for Today

- IP Subnetting
- IPv6
- Routing Introduction
- RIP

- Sources in PD:
  - Subnetting: 4.3.1
  - RIP: 3.3.2
- Sources in Dordal:
  - IPv6: 11

Scaling Problems

Not enough network numbers.

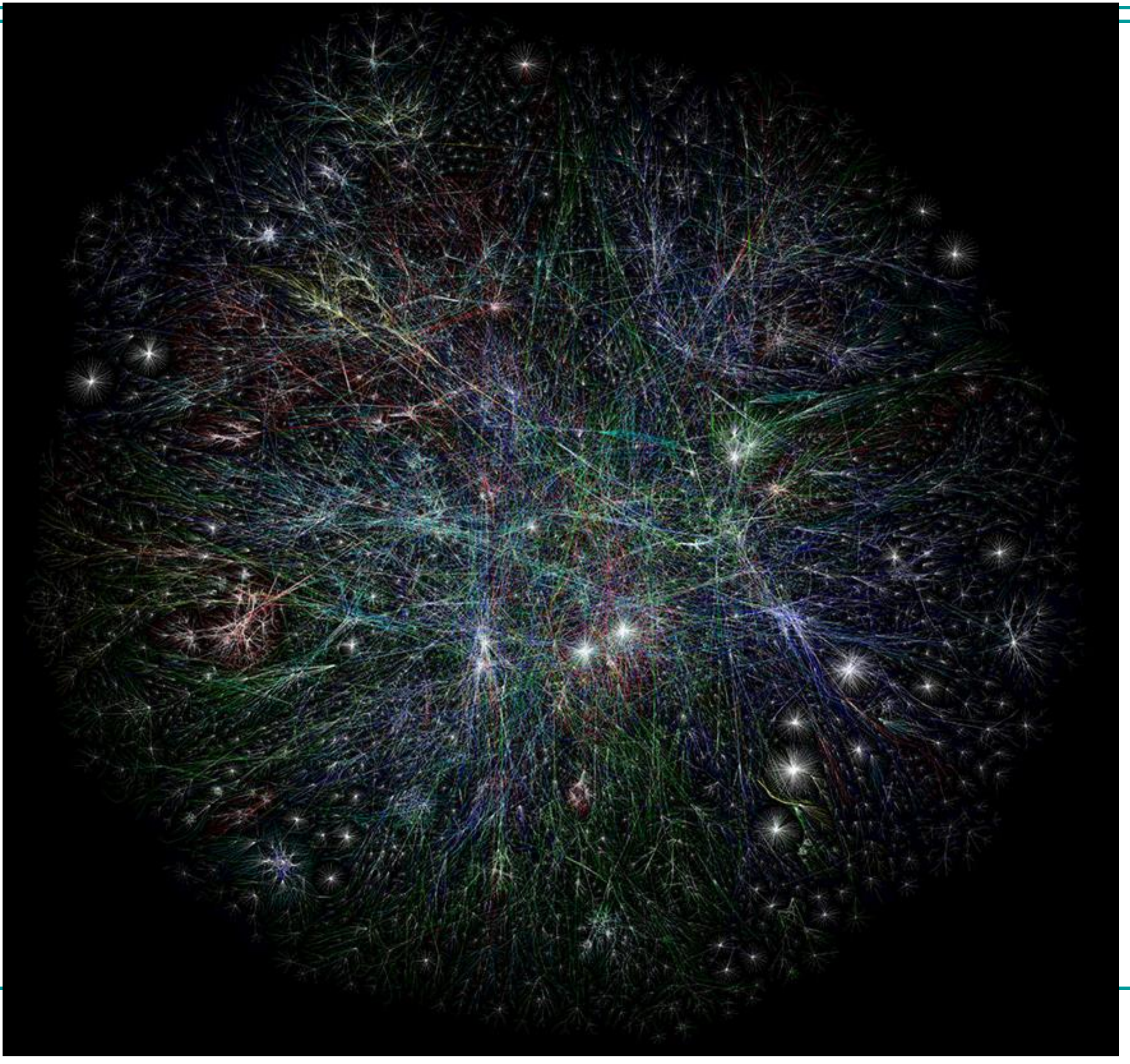
- Class C network with 2 nodes wastes 253 IP addresses
- Class B network with ~ 300 nodes wastes $64{,}000$ IP addresses

Only $2^{14} \sim 16{,}500$ Class B networks

Routing information too cumbersome.

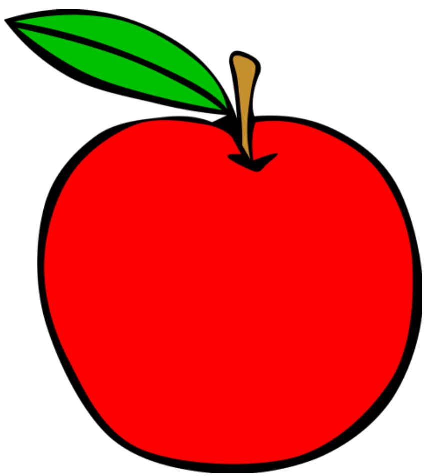
- More networks means larger routing tables

Copyright: Opte project via Kaspersky.com



Subnetting

- Idea: One IP network number allocated to several physical networks.
  - The multiple physical networks are called *subnets*
  - Should be close together (why?)
  - Useful when a large company (or university!) has many physical networks.



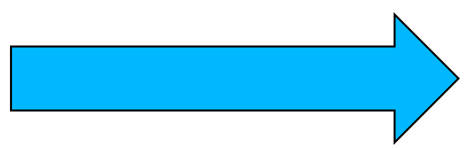
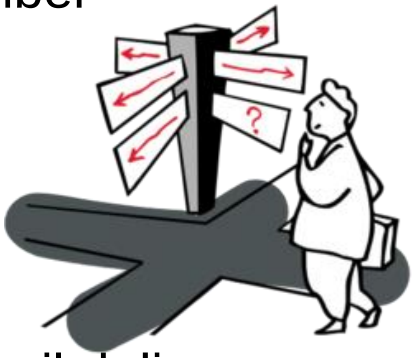


Image source: https://images-na.ssl-images-amazon.com/images/I/61wI3JN2G2L._SY355_.jpg

Subnetting Houses

- A small yishuv has 200 houses
  - Every house has a number - not necessarily organized in order
  - To send Moshe a letter:
    - Moshe, House 121, Moshav Yula
  - The mail deliverer memorizes where each house is
  - A visitor asks for directions based on the house number

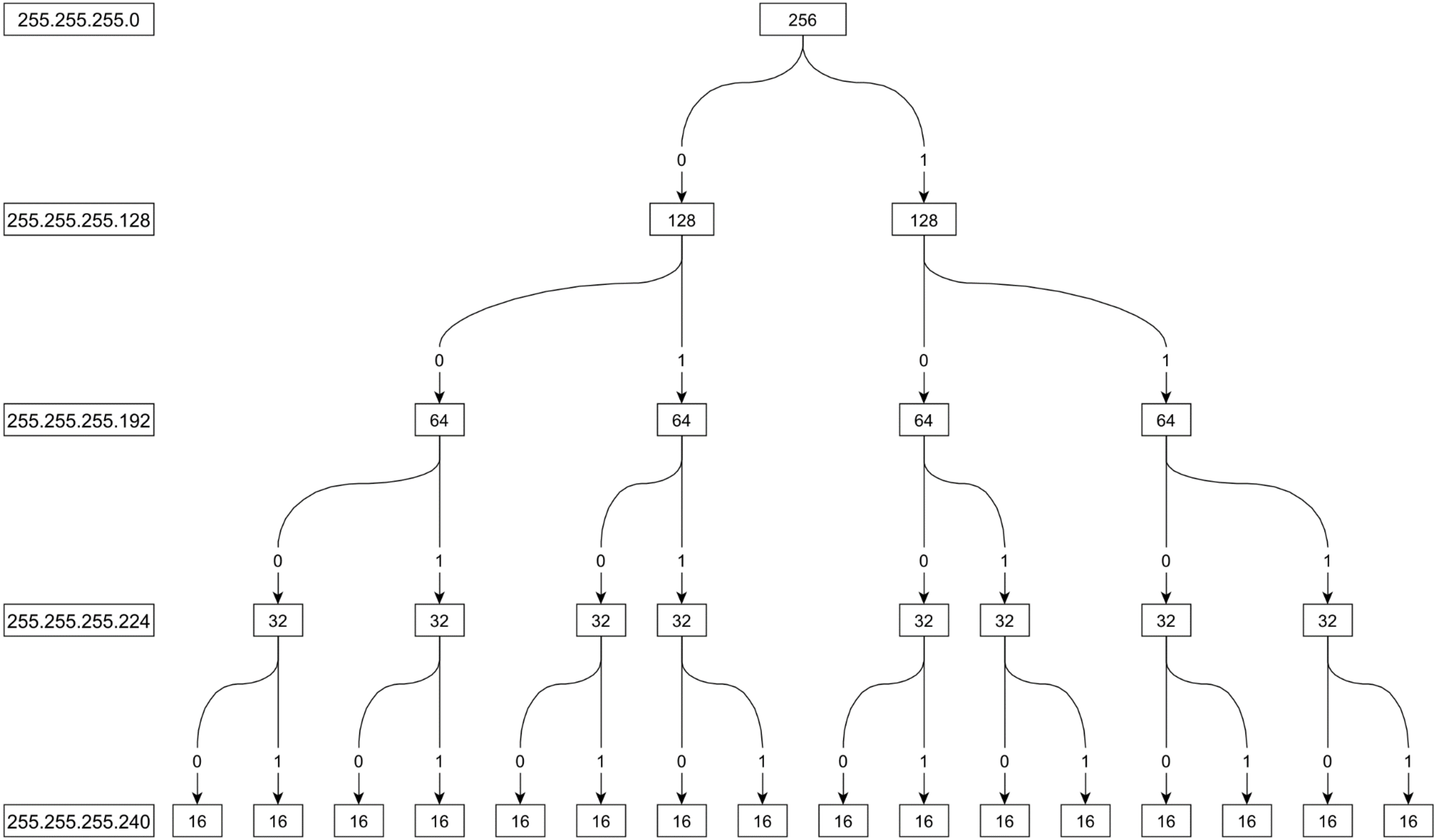


- What if there are 10,000 houses?
  - We can still give numbers to each house
    - Moshe, House 3093, Moshav Yula
  - Something is not quite right, though – how will the mail deliverer know where each house is?
  - There are too many houses to keep track of this way

Subnetting Houses

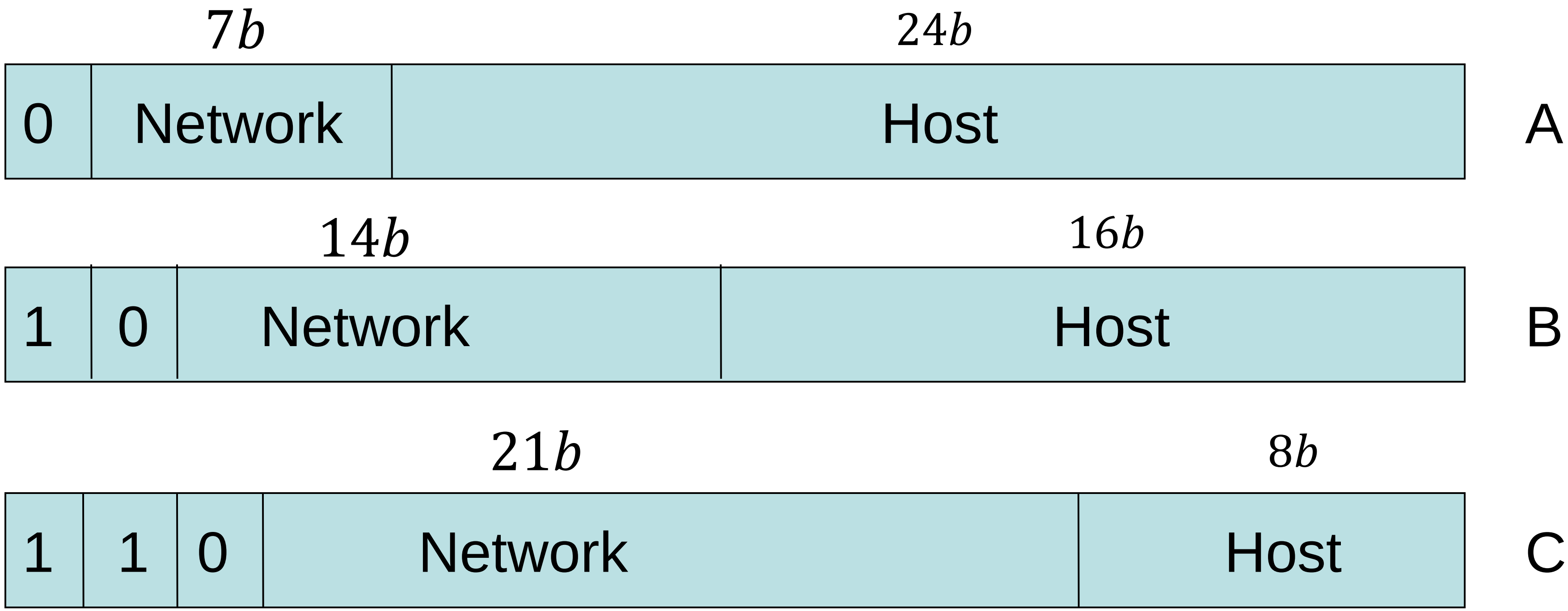
- One option: Divide the city into streets
  - Organize the houses onto streets with names
    - May require us to renumber some houses if they weren't done right
  - Make the address – Street + number
  - To send a letter to Moshe:
    - All houses 3000-3099 are on Rechov Hula
    - Moshe, House 3093, Yula → Moshe, Rechov Hula 93, Yula
  - We make a mapping from the house number to a street
    - Here 3000-3099 are all on Rechov Hula
    - Say 4000-4099 are on Rechov Goober
    - We can write that as 30XX are on Rechov Hula, 40XX are on Rechov Goober
    - Given a house number we can figure out the street based on it

Prefix Hierarchy



IP addresses

- Hierarchical, not flat as in Ethernet



- Written as four decimal numbers separated by dots: 158.130.14.2

Subnet Numbers

- *Subnetting*
  - All nodes are configured with *subnet mask*
  - Allows definition of a *subnet number*
- All hosts on a physical subnetwork share the same *subnet number*

Determines how many computers are on the subnet

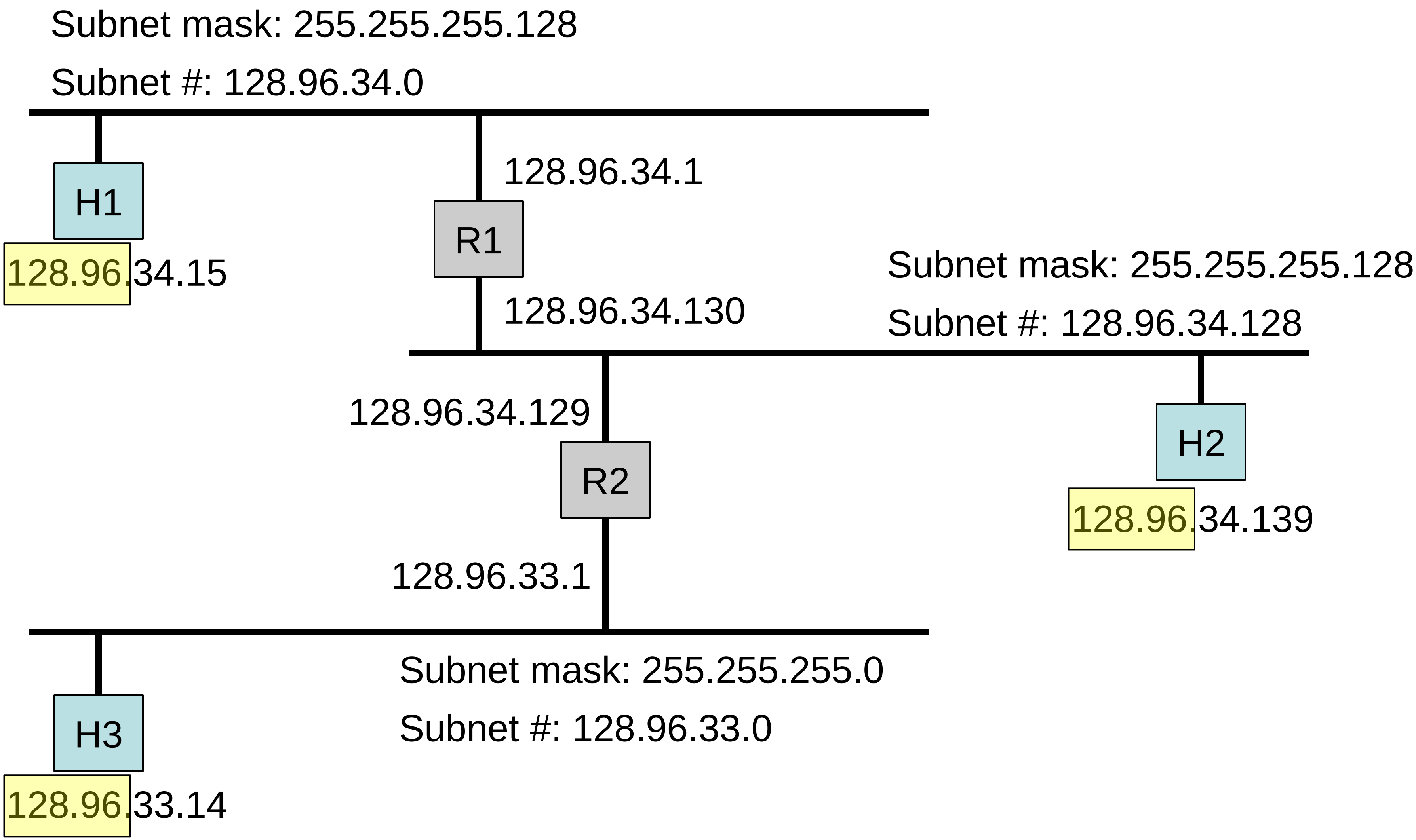
Subnet Mask (255.255.255.0)

1111 1111	1111 1111	1111 1111	0000 0000

Subnetted Address:

Network Number	Subnet ID	Host ID

Example of Subnetting



Subnets, continued

- Mask is bitwise-ANDed with address
- This is done at routers
- Router tables in this model:
  - $\langle Subnet\ \#, Subnet\ Mask, NextHop \rangle$
  - Routing table sorted by the length of the Subnet Mask (longest first).

- Subnetting allows a set of physical networks to look like a single logical network from elsewhere

So Far

- IP Subnetting
- IPv6
- Routing Introduction
- RIP

IPv6 Addresses

- 16 bytes = 128 bits
  - 3.4×10^{38} addresses in theory (16, 000 per m^2 of the Earth)
  - Written in 8 four letter hex blocks
  - Omit leading 0s, compress runs `::`

Examples:

- `2a00:1450:4028:804::200e`
- `2a03:2880:f142:182:face:b00c:0:25de`
- `2a02:26f0:7000::211:71f0`

https://m.facebook.com/whatismyip/photos/a.10150650334802180/10154092465582180/

IPv6 Address Types

Localhost

Interface identifiers

Link Local

Anycast

Multicast

IPv6 Address Types: Localhost

“this” for the computer

Like `127.0.0.1` in IPv4

`::1`

- `0:0:0:0:0:0:0:1`

IPv6 Address Types: Link Local

Unique only for local LAN

- Auto-configuration
- Site local addresses also

`fe80::/64`

- `fe80:0:0:0:XXXX:XXXX:XXXX:XXXX`
- Can embed the physical address in last 64 bits

Valid only on a single network

- Like 169.254.0.0/16
- Advertised to other hosts by routers
- Not routable on the internet

Scoped to a particular interface

- E.g. %eth0

EUI-64 format

- Take Ethernet 48 bits
- Add 1111 1110 between 3rd and 4th byte
- Flip 7th bit

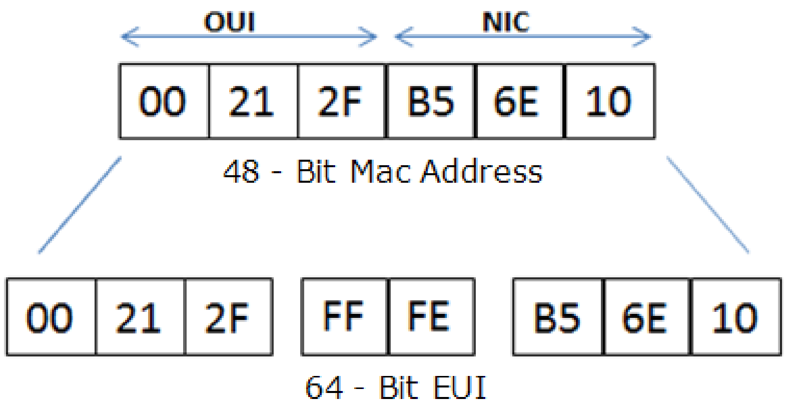
EUI-64 Example

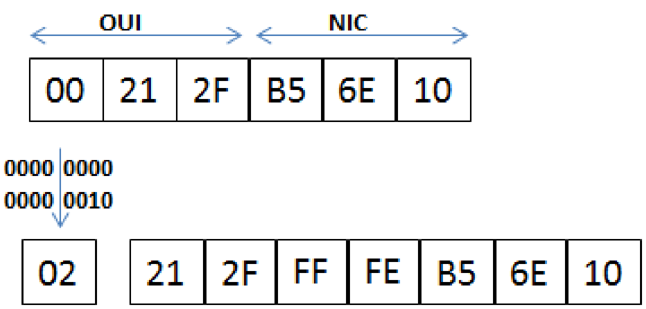
- Ethernet 00:a0:cc:24:b0:e4
- Result: fe80::2a0:ccff:fe24:b0e4

IPv6 Address Types: Link Local

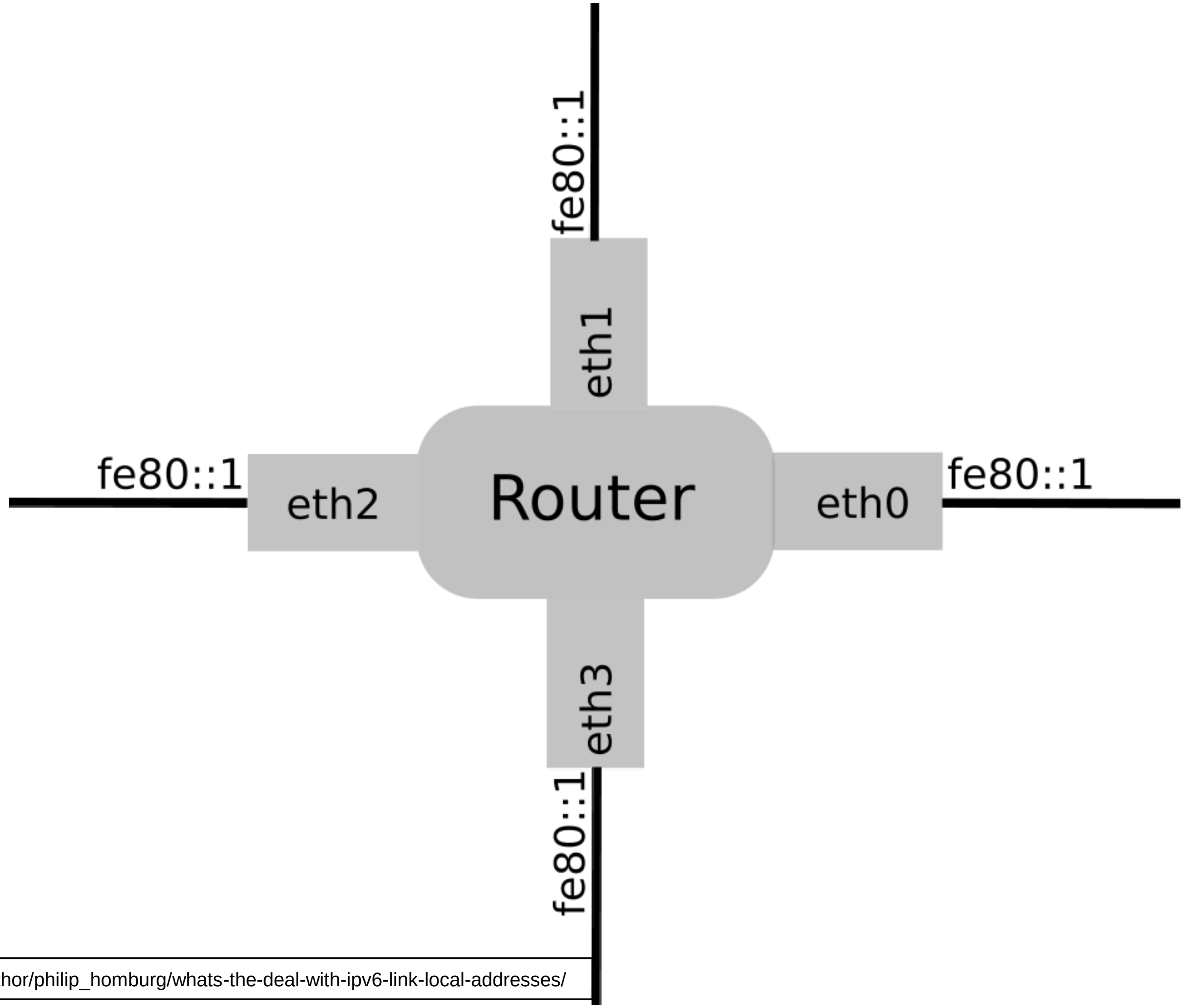
SPUI, Public domain, via Wikimedia Commons

EUI-64 Illustrated





Why you need to scope



https://labs.ripe.net/author/philip_homburg/whats-the-deal-with-ipv6-link-local-addresses/

IPv6 Address Types: Interface identifiers

Typical division

- 64 bits network prefix
- 64 bits host id

Started with `2000::/3` block

- `0200:0:0:0:0:0:0:0`

Used EUI-64 format for host identifier

- Privacy problems
- Can track your computer wherever it goes

IPvSeeYou vulnerability

- Home router might have fixed EUI-64 address
- Can track your public IPv6 address back to home router
- Easy to map MAC wireless router to map
- wigle.net

More secure idea using hashes and secret key for host id

IPv6 Address Types: Anycast

One address, many potential servers

Route to nearest one normally

One of the set receives the message and may respond

No special prefix

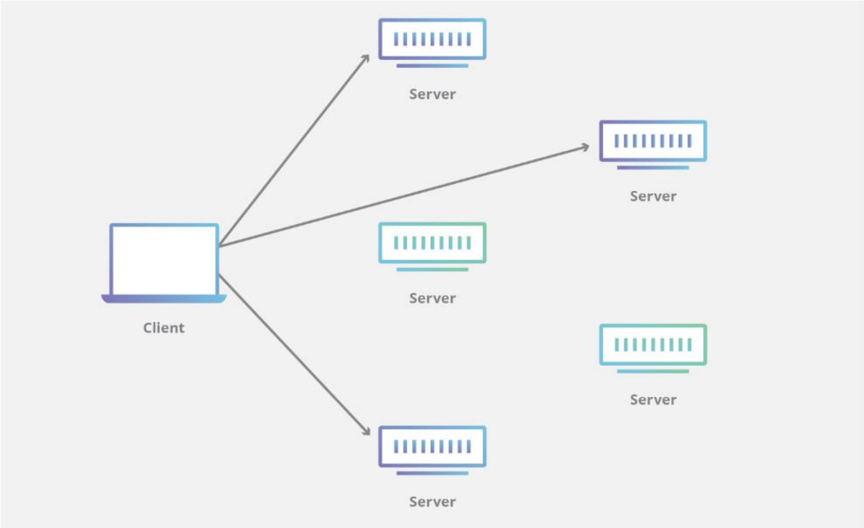


Image © Cloudflare, source: https://www.cloudflare.com/learning/cdn/glossary/anycast-network/

IPv6 Address Types: Multicast

Group of receivers

`ff00::/8`

- `ff00:0:0:0:0:0:0:0`

Lots of groups

- No more need for IP LAN broadcast
- Subscribe using Multicast Listener Discovery (MLD) messages
- Switches forward to routers to ensure only interested nodes receive

Ethernet multicast for LAN distribution

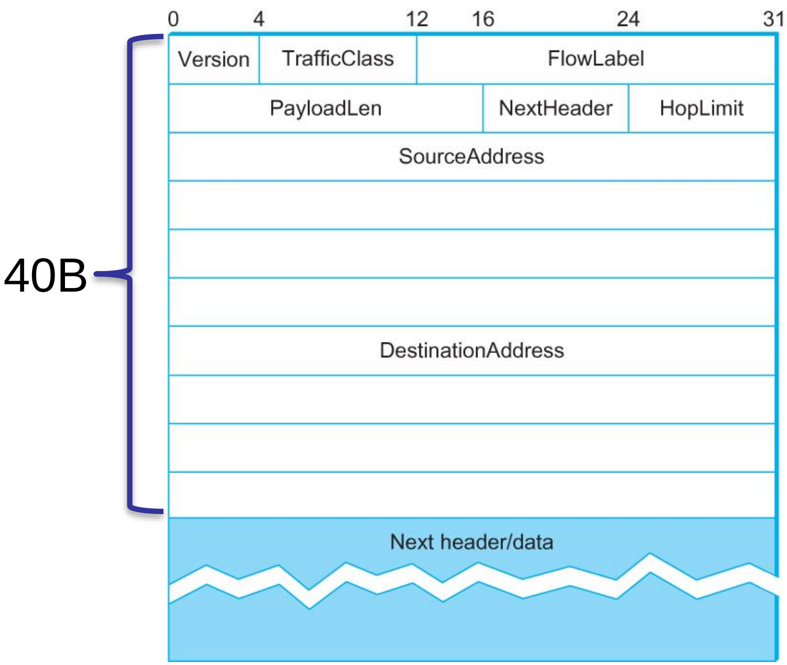
- Last 4 bytes of IPv6 multicast group embedded in Ethernet with 3333 prefix
- w, x, y, z last bytes of group, embedded as 33:33:w:x:y:z

Example:

- All nodes: `ff02::1` → `33:33:00:00:00:01`
- All routers: `ff02::2` → `33:33:00:00:00:02`

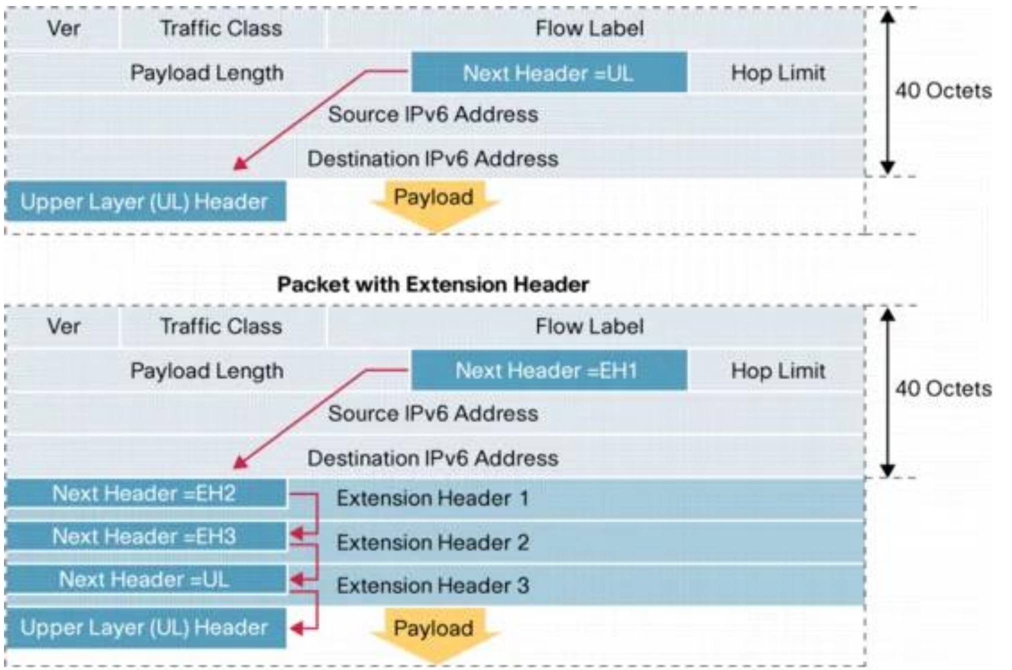
IPv6 Packet Format

- Version (=6)
- Traffic class: for DiffServ QoS
- Flow label: grouping real time packets into flows for QoS and prioritizing
  - Non-real time traffic has no flow label
- Payload Length: Body length (bytes)
- Next Header: Higher level protocol the packet belongs to (Protocol in IPv4)
- Hop Limit: Same as TTL field of IPv4
  - No checksum!
- Source address (16B)
- Destination address (16B)

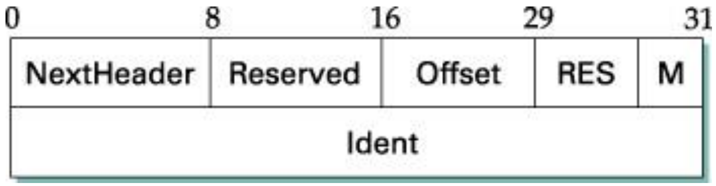


IPv6 Next Header

- Additional options and extensions are in headers after the initial 40B
  - Fixed order, multiple of 8 bytes
- If there are extensions → Next Header has the next extension header type
  - 44 for fragmentation
  - 43 mobile routing
  - 51 authentication
  - 50 encrypted data
- If no extensions → it's the inner protocol number
  - 59 if there is nothing inside



Source: https://www.cisco.com/en/US/technologies/tk648/tk872/technologies_white_paper0900aecd8054d37d.html

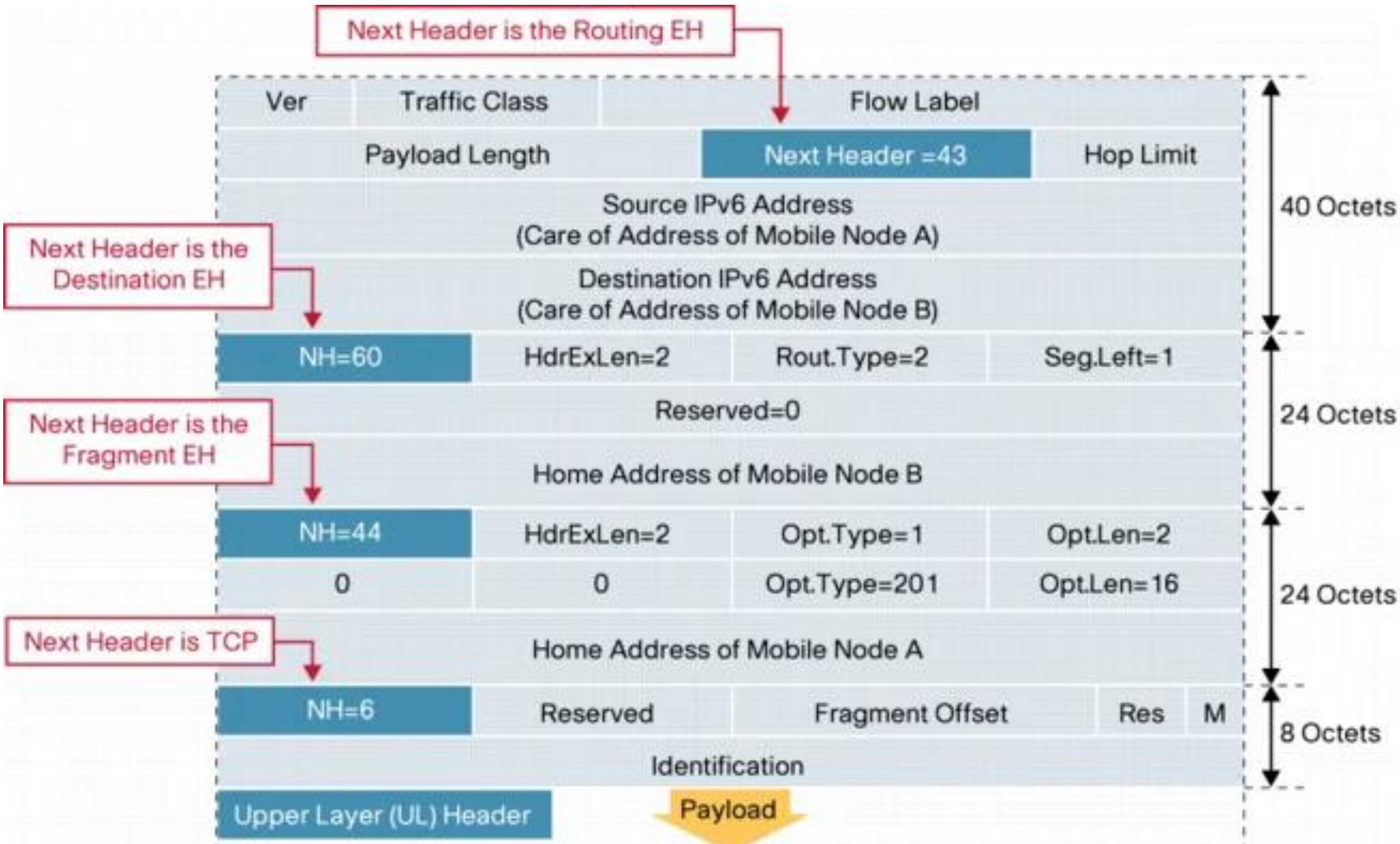


44 fragmentation header

IPv6 Extension Headers List (RFC 2460)

Order	Header Type	Next Header Code
1	**Basic IPv6 Header**	-
2	**Hop-by-Hop Options**	0
3	**Destination Options (with Routing Options)**	60
4	**Routing Header**	43
5	**Fragment Header**	44
6	**Authentication Header**	51
7	**Encapsulation Security Payload Header**	50
8	**Destination Options**	60
9	**Mobility Header**	135
	No next header	59
Upper Layer	TCP	6
Upper Layer	UDP	17
Upper Layer	ICMPv6	58

How Extension Chaining Looks



Header details: Hop-by-Hop Options

Type, Value pairs that each router on the path should examine

Only options defined now are:

- Padding
- For jumbograms (>64KB)

Use is discouraged since it may slow down the internet

Header details: Destination Options

Type, Value pairs that the destination should examine

Only options defined now are:

- Padding
- Limit nesting of tunnels
- Mobile IPv6
- Routing protocol for low power and lossy networks (RPL)

Header details: Routing Header

Original version

- A type of source routing
- List of routers $\langle R_1, R_2, \dots, R_n, D\rangle$ to go through on the way to D
- Pointer to where up to on the list – next hop put in the IPv6 Destination field

Deprecated due to security problem

- What if you write $\langle R_1, R_2, R_1, R_2, R_1, \dots, D\rangle$?

More limited header defined later

- Original one still use for the RPL low power setting

Header details: Fragmentation Header

Only the sender can fragment

- Routers and middle nodes can't fragment or reassemble
- Avoid if possible

Fragmentation fields

- 32 bit Unique ID
- 13 bit Offset

Each fragment must have a copy of the IPv6 header and critical extension headers

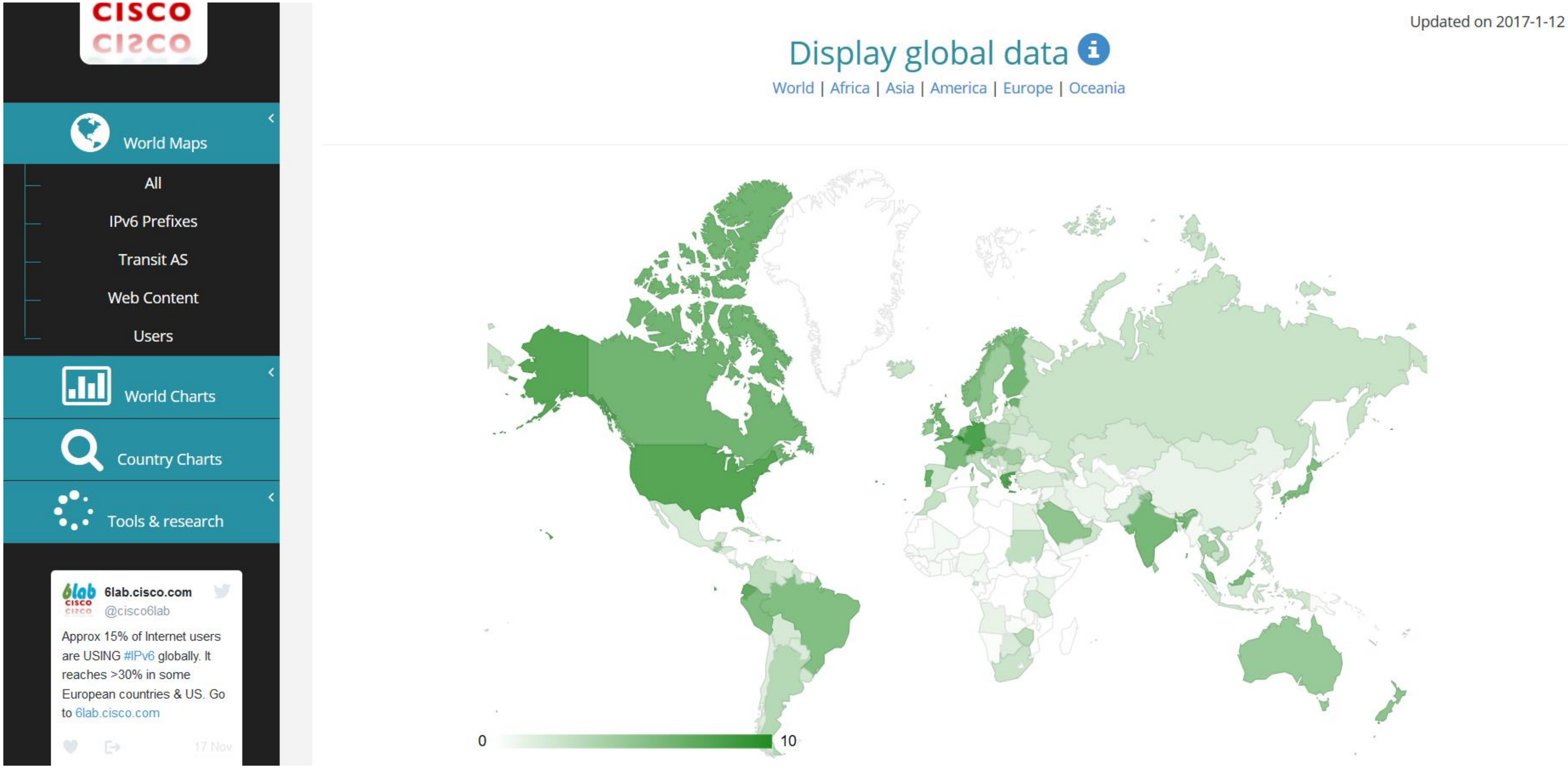
IPv6 requires path MTU of 1280 bytes after link layer headers

- If smaller, link layer must fragment and reassemble on its own

IPv6 Deployment Map

http://6lab.cisco.com/stats/

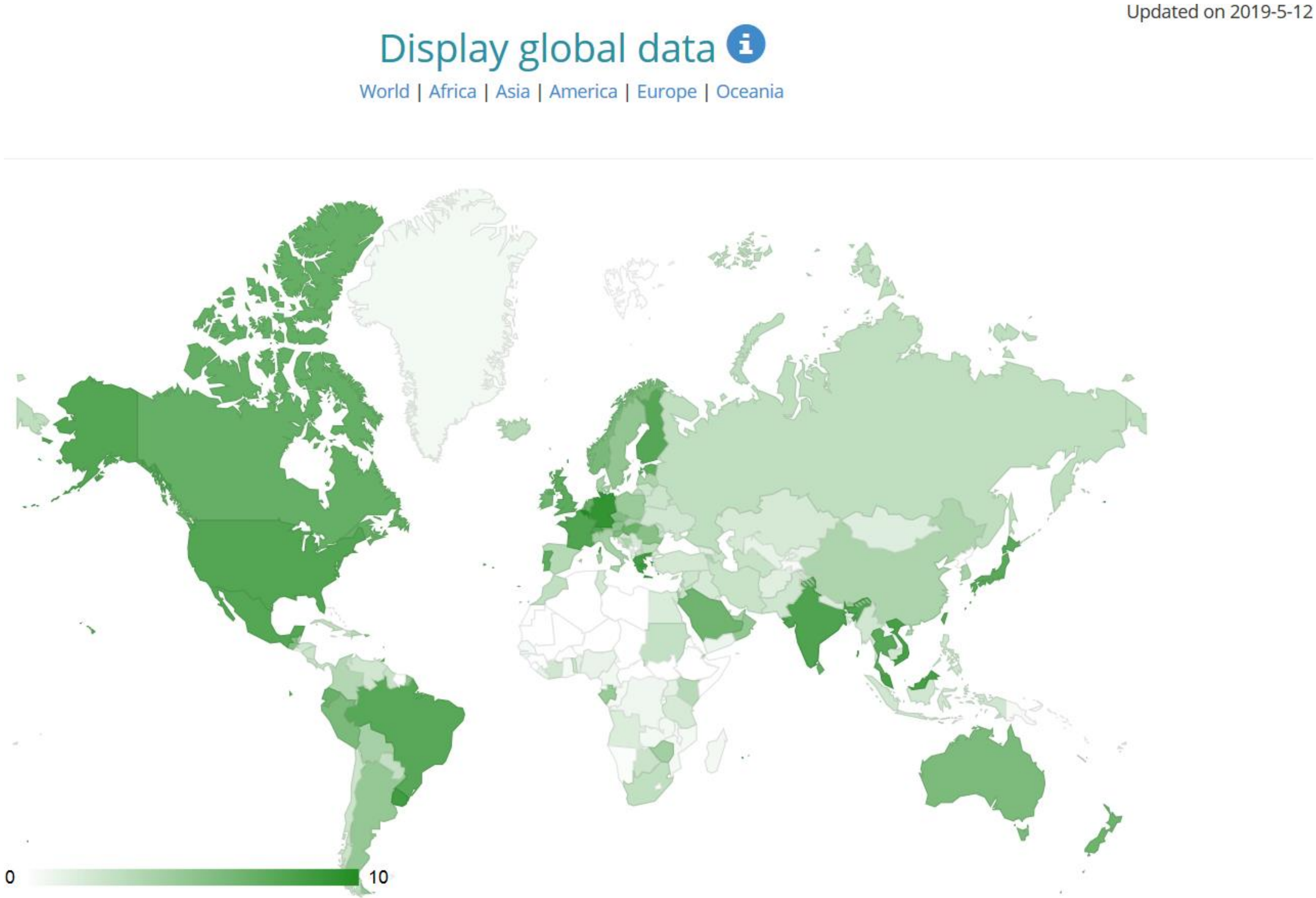
http://6lab.cisco.com/stats/cible.php?country=IL&option=all



IPv6 Deployment Map

http://6lab.cisco.com/stats/

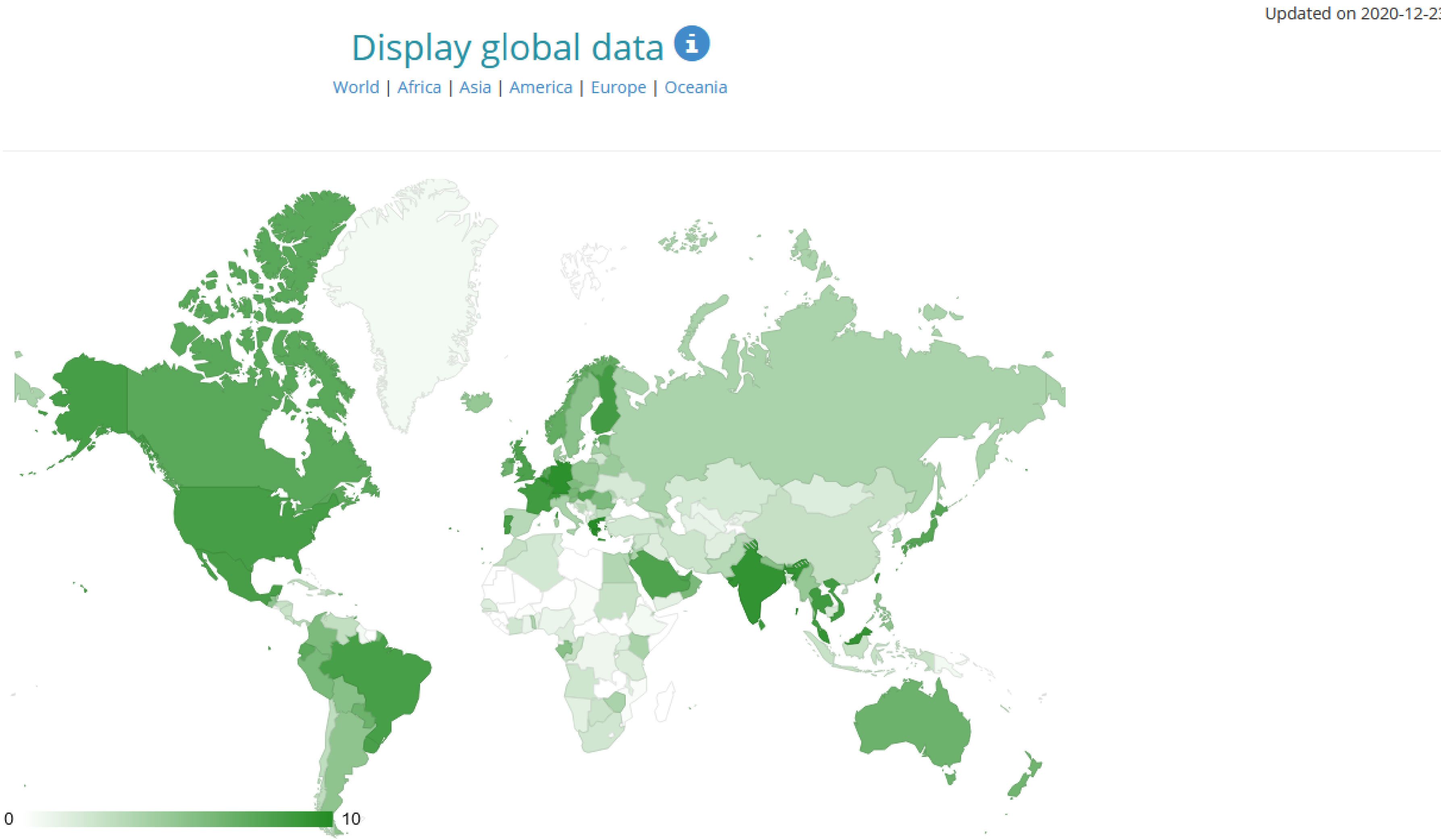
https://6lab.cisco.com/stats/index.php?option=all



IPv6 Deployment Map

http://6lab.cisco.com/stats/

https://6lab.cisco.com/stats/index.php?option=all



IPv6 Deployment Map

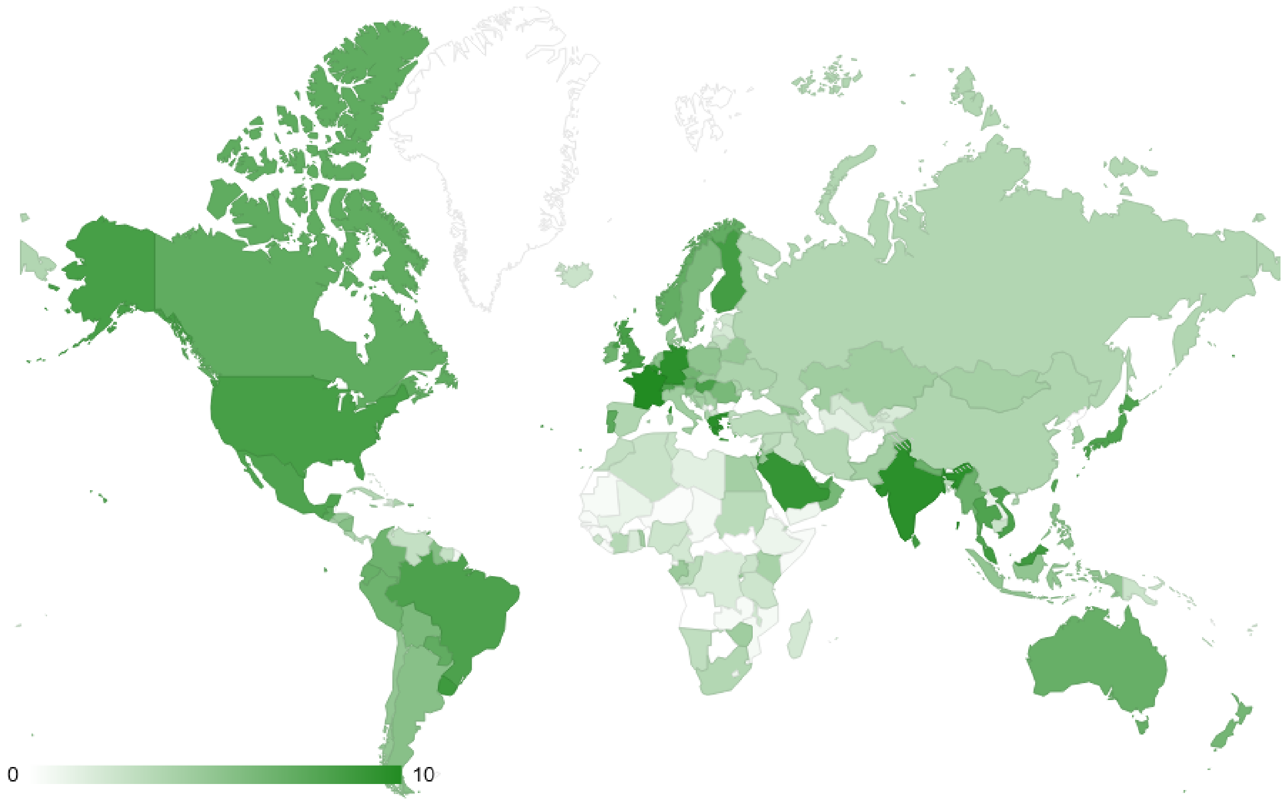
http://6lab.cisco.com/stats/

https://6lab.cisco.com/stats/index.php?option=all

Updated on 2022-12-15

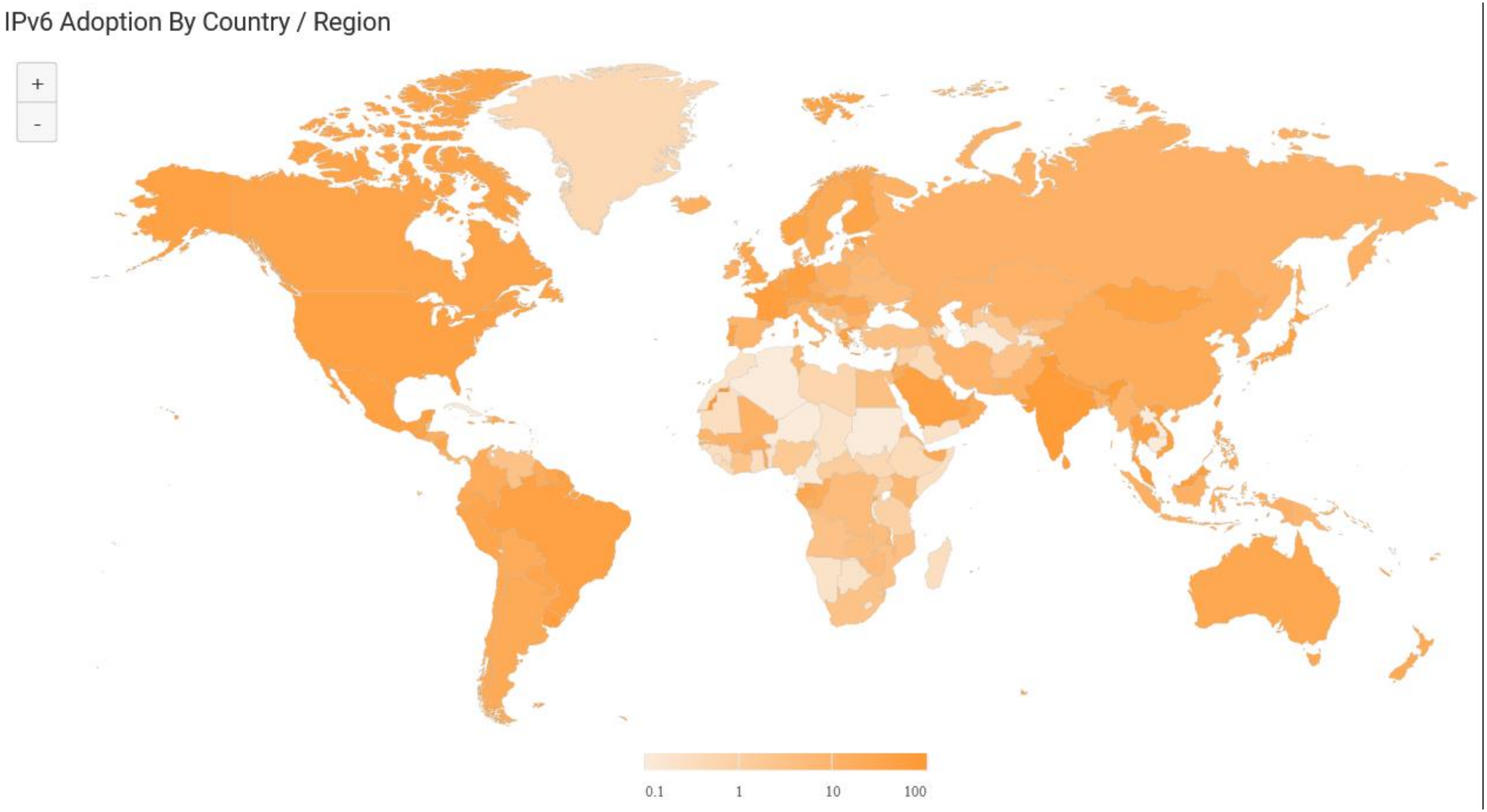
Display global data

World | Africa | Asia | America | Europe | Oceania



IPv6 Deployment Map

Updated June 2024



Top 10 Countries

https://www.akamai.com/visualizations/state-of-the-internet-report/ipv6-adoption-visualization

*Country data ranked by % of IPv6 connections from that country.

Search:

RANK	IPV6%	COUNTRY / REGION
1	68.5%	Montserrat
2	67.4%	Pitcairn
3	65.7%	Malaysia
4	65.3%	India
5	61.7%	Tokelau
6	61.6%	Germany
7	61.4%	France
8	59.1%	Belgium
9	56%	Uruguay
10	53.2%	Saudi Arabia

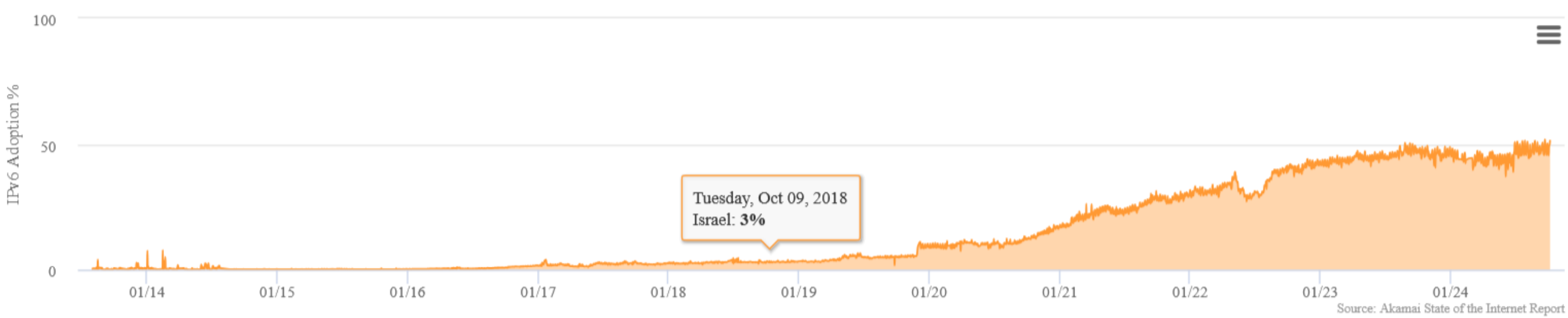
Top 10 Countries

https://www.akamai.com/visualizations/state-of-the-internet-report/ipv6-adoption-visualization

**Country data ranked by % of IPv6 connections from that country.*

Search: isr

RANK	IPV6%	COUNTRY / REGION
14	51.3%	Israel



So Far

- IP Subnetting
- IPv6
- Routing Introduction
- RIP

Routing

Images: https://www.aaroads.com/blog/south-carolinas-new-highway-signs/#post/0
By de:User:Jutta234, CC BY-SA 2.5, https://commons.wikimedia.org/w/index.php?curid=16078428

Routing protocols on the internet

Routing information protocol (RIP)

- Distance vector routing
- Uses Bellman-Ford algorithm
- Outdated, suffers from count-to-infinity problem

Open shortest path first (OSPF)

- Link state routing
- Runs over Layer 3 (over IP)
- Uses Dijkstra algorithm to determine shortest paths

Intermediate System to Intermediate System IS-IS

- Link state routing – similar to OSPF
- Runs over Layer 2 (under IP)
- Uses Dijkstra's algorithm

Border gateway protocol (BGP)

- Between networks (administrative domains, Autonomous Systems)
- Path-vector routing
- Consideration of business agreements

Routing criteria

Correctness

Every packet is delivered to its destination

Efficiency

Choose paths with small delay and high throughput (network wide)

Complexity

Setting up routing tables

Making routing decisions

Robustness

Cope with topology changes

No network reboot

Adaptiveness

Load balancing and traffic control

Fairness

All users get the same degree of service

Path costs, Routing metrics

Minimum hop

Number of channels traversed

Shortest path

Channels have statically assigned weights

Cost of a path is the sum of costs of the edges

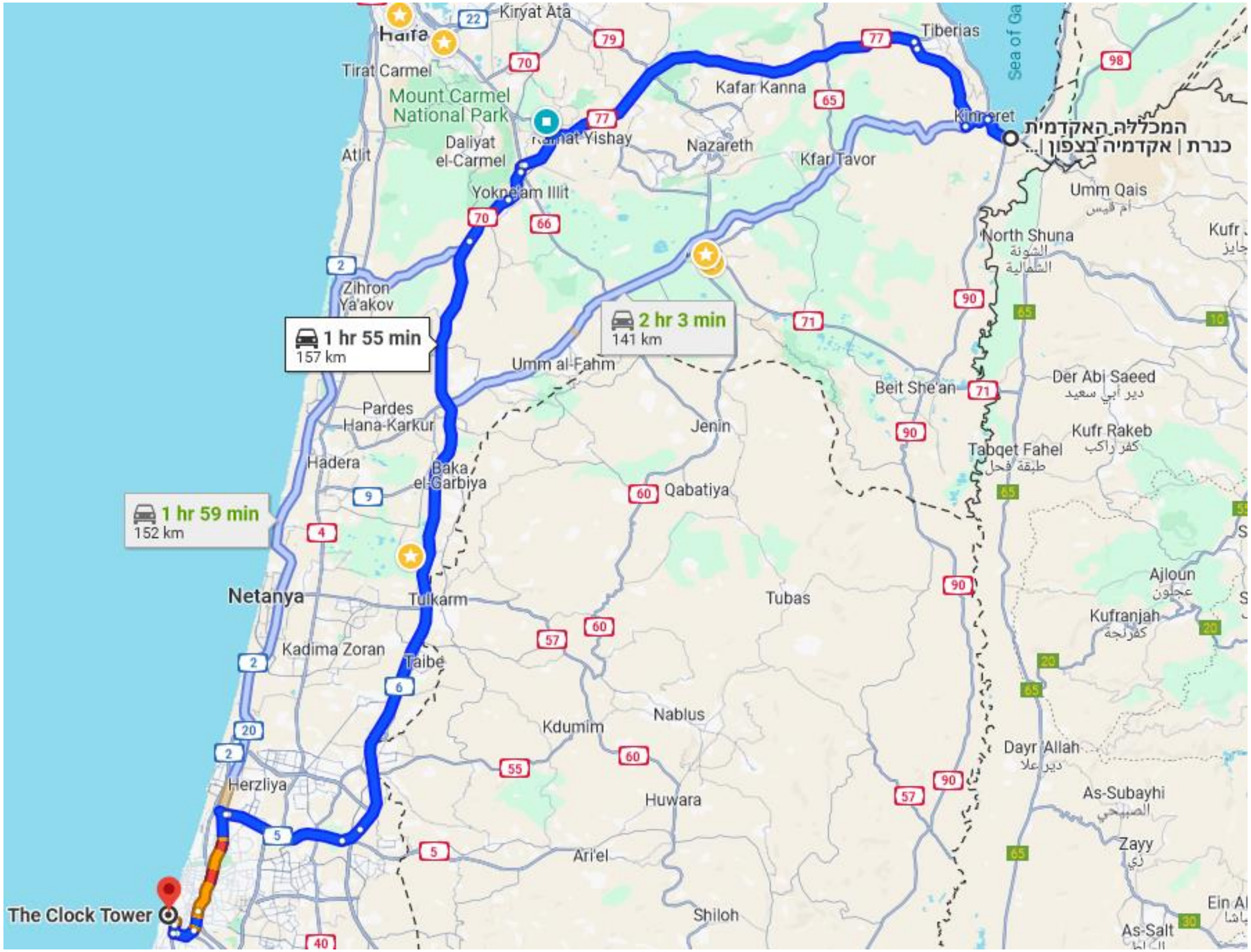
Presumes no negative-cost cycles

Minimum delay

Channels have dynamically assigned weights based on the traffic on the channel

Routing tables are repeatedly revised such that paths with close to minimal delays are chosen

Shortest vs Fastest

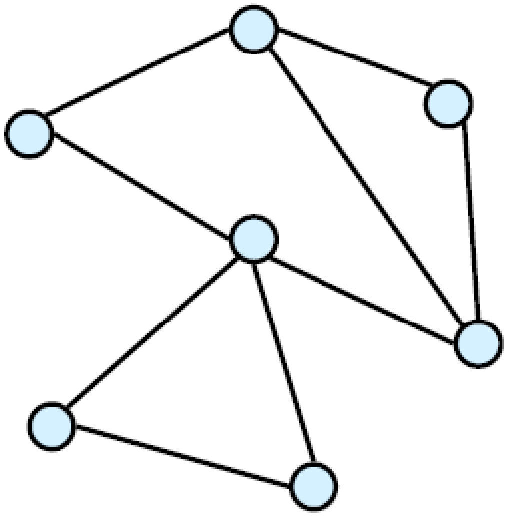


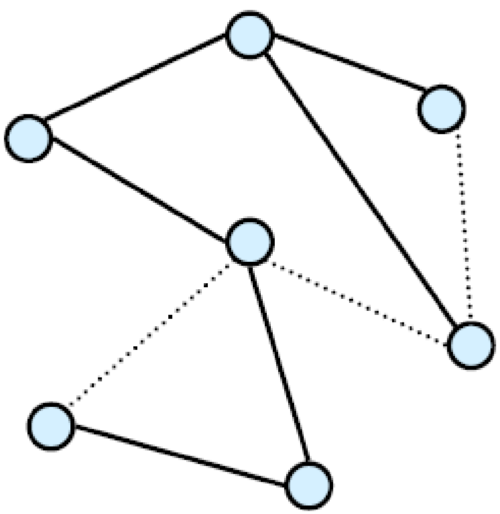
So Far

- IP Subnetting
- IPv6
- Routing Introduction
- RIP

Spanning Trees (Abstractly)

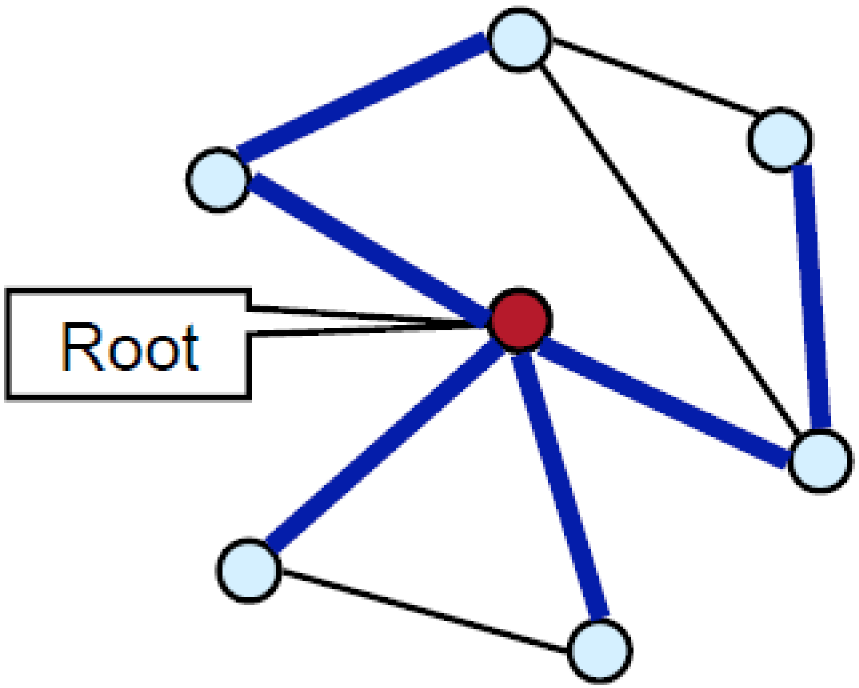
- Given a connected graph G
- A *spanning tree* is an acyclic, connected subgraph of G that contains all the nodes.





Spanning Tree Algorithm (Abstractly)

- Pick a root node
  - Compute shortest paths to root
  - Need to break ties

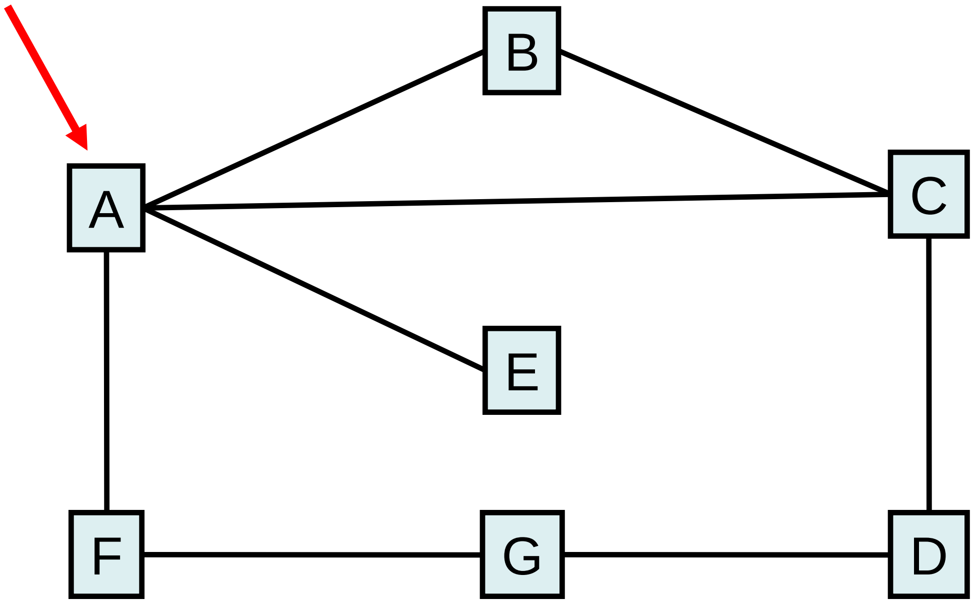


Distance Vector Algorithm (RIP)

- Similar to the Spanning Tree Algorithm
  - Except that information about distance to ALL nodes is forwarded (not just info. about root.)
  - Sometimes called Bellman-Ford algorithm

- Each node constructs a *Distance Vector*
  - Contains distances (costs) to reach all other node
  - Initially:
    - Distance to neighbors (a simplification for now) = 1
    - Distance to others = ∞
  - Routing table reflects node’s beliefs

Example Network Graph



A's initial information:

Dest	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

Iteration Steps

Each host sends its DV (cost) to its neighbors

Neighbors update distance vectors and routing information accordingly.

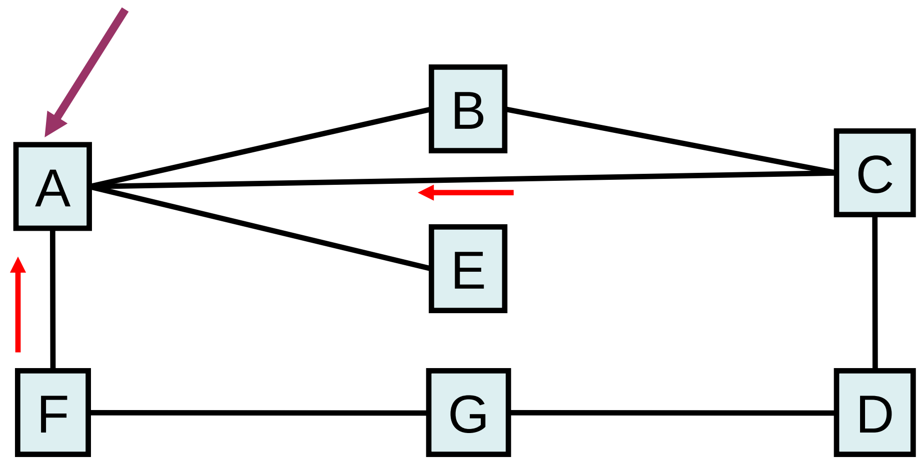
- Ignore worse information
- Update any better routes

If host changed its tables, send new DV to neighbors

After a few iterations, routing information *converges*

Example Iteration Steps

A	1
B	∞
C	∞
D	∞
E	∞
F	0
G	1



A	1
B	1
C	0
D	1
E	∞
F	∞
G	∞

F sends A its DV
- A discovers that G can be reached in to two hops via F

C send A its DV
- A discovers that D can be reached in two hops via C

Dest	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

Dest	Cost	NextHop
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	2	F

Dest	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Details

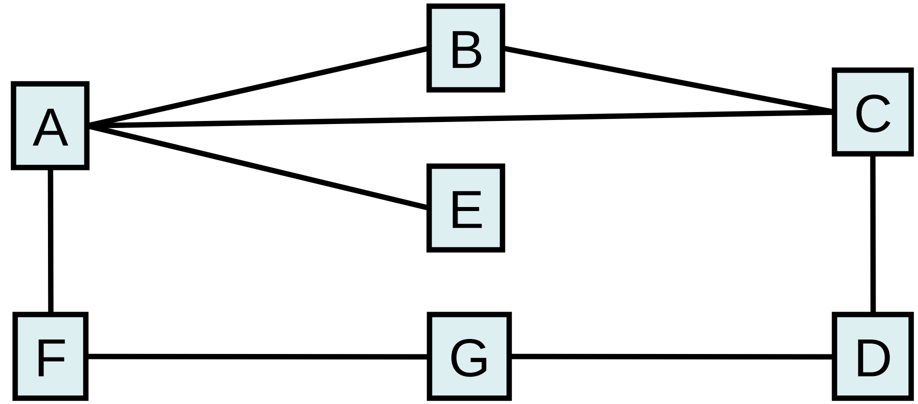
- Note: No single host has all routing information.
- When to send update vectors?
  - When your routing table changes (triggered)
  - Periodically ("I'm alive!")

- Detecting link/node failure
  - (1) Periodically exchange "I'm alive!" messages.
  - (2) Timeout mechanism

- In a static network, if all weights are 1, once A discovers a path to B (cost $< \infty$), no subsequent round will ever discover a better path to B
  - All weights are 1 is called "Hop Count" (Default RIP action)

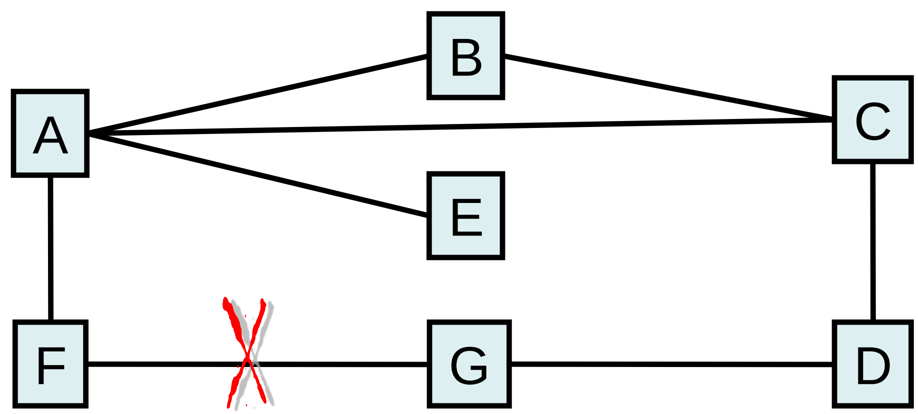
Dynamic Networks

What about a dynamic network?



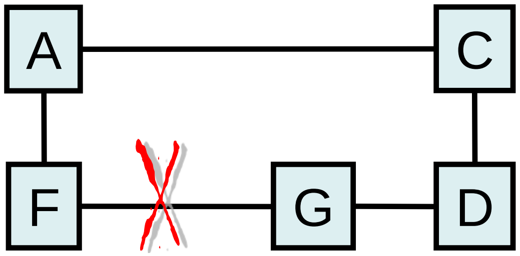
- A nodes enters
  - The new neighbors will eventually inform everyone else (easy)

- A node/link fails
  - Is the network partitioned?
  - How do nodes discover that a node/link is gone?



Link Failure Example 1

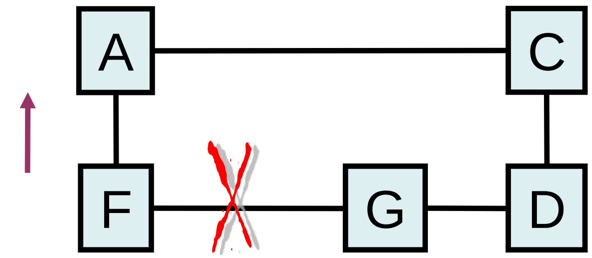
A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	2	F	G	2	D	G	1	G	G	1	G



A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	2	F	G	2	D	G	1	G	G	∞	-

Link Failure Example 1

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	2	F	G	2	D	G	1	G	G	∞	-



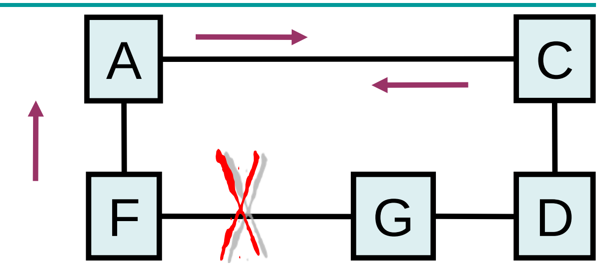
Case 1:

1. F sends to A

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	∞	-	G	2	D	G	1	G	G	∞	-

Link Failure Example 1

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	∞	-	G	2	D	G	1	G	G	∞	-

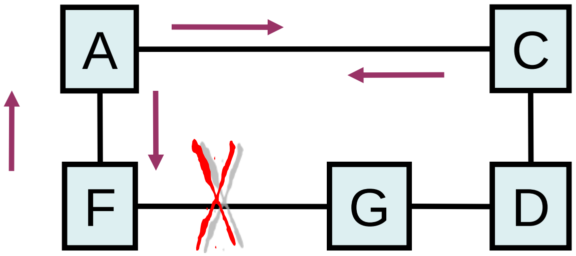


A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	3	C	G	2	D	G	1	G	G	∞	-

Case 1:
1. F sends to A
2. A sends to C
   - (nothing)
3. C sends to A

Link Failure Example 1

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	3	C	G	2	D	G	1	G	G	∞	-



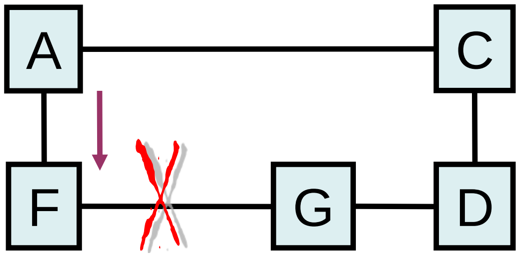
A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	3	C	G	2	D	G	1	G	G	4	A

Case 1:

1. F sends to A
2. A sends to C
   - (nothing)
3. C sends to A
4. A sends to F

Link Failure Example 2

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	2	F	G	2	D	G	1	G	G	∞	-



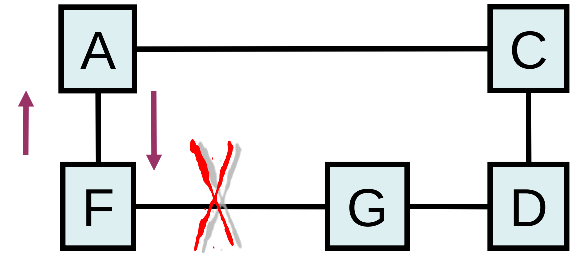
Case 2:

1. A sends to F

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	2	F	G	2	D	G	1	G	G	3	A

Link Failure Example 2

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	2	F	G	2	D	G	1	G	G	3	A



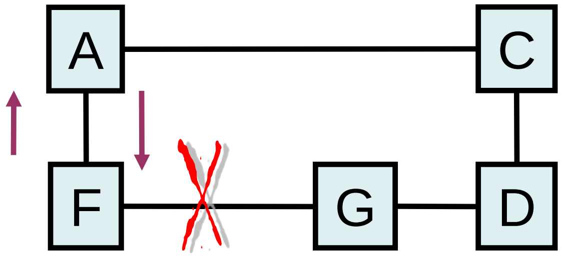
Case 2:

1. A sends to F
2. F sends to A

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	4	F	G	2	D	G	1	G	G	3	A

Link Failure Example 2

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	4	F	G	2	D	G	1	G	G	3	A



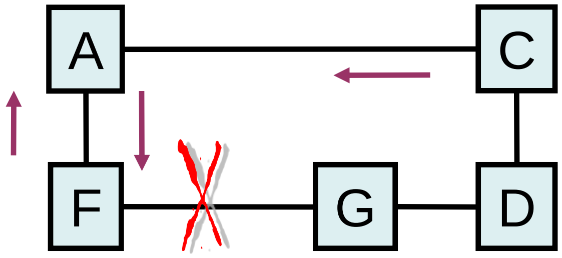
Case 2:

1. A sends to F
2. F sends to A
3. A sends to F

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	4	F	G	2	D	G	1	G	G	5	A

Link Failure Example 2

A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	4	F	G	2	D	G	1	G	G	5	A



A			C			D			F		
Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop	Dest	C	Next hop
A	-	-	A	1	A	A	2	C	A	1	A
C	1	C	C	-	-	C	1	C	C	2	A
D	2	C	D	1	D	D	-	-	D	2	G
F	1	F	F	2	A	F	2	G	F	-	-
G	3	C	G	2	D	G	1	G	G	4	A

Case 2:

1. A sends to F
2. F sends to A
3. A sends to F
4. …
5. C sends to A
6. A sends to F

Network Partition

- The loop between A and F is broken only by C's update
  - C has **real information** about another path to G

- What if C's update never comes? What if C's information is false?
  - The network is partitioned –OR–
  - G is completely offline

- The other nodes will continue counting until infinity (or the distance field reaches its max)

Three RIP Solutions

Option 1: Infinity is small

- Don’t let the distance fields go above a predefined maximum
  - Say 15, 20, etc. (15 is the actual value)
  - The max must be greater than the diameter of the network

Option 2: Don’t send routes to the one who sent it to you

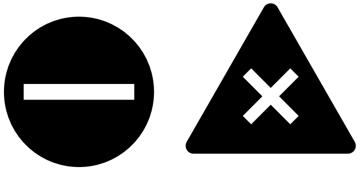
Example: F receives an update from A about a path to G.

- In the example before, F didn’t know that’s A’s “path” went via F
- A doesn’t send F an advertisement to get to G since it goes via F
- Called “split horizon”
- “Split horizon with poison reverse”- A sends route for G to F, but with infinite distance

Three RIP Solutions

Option 3: Route Poisoning and Holddowns

- Instead of just stopping to announce a lost destination, a router sends a poison message for the destination
  - E.g. the router announces distance 16 ($\equiv \infty$) for the destination
- For a fixed period of time (say 3 minutes) any new offers for the destination via the sender will be rejected
  - E.g. Wait until everyone has heard the poison and then accept new offers
- Gives a chance for the route to be deleted by everyone.
- Cisco specific



RIP v1 (1986) Details

Classful routing only

- No subnet masks
- Assumes everyone has same mask

Supports multiple address families

Commands

- Request: Ask for someone to send their info
- Response: Respond with routing table

Updates sent

- Timer about 30 seconds between updates
- Due to changes

Split Horizon, Poison Reverse, Small Infinity

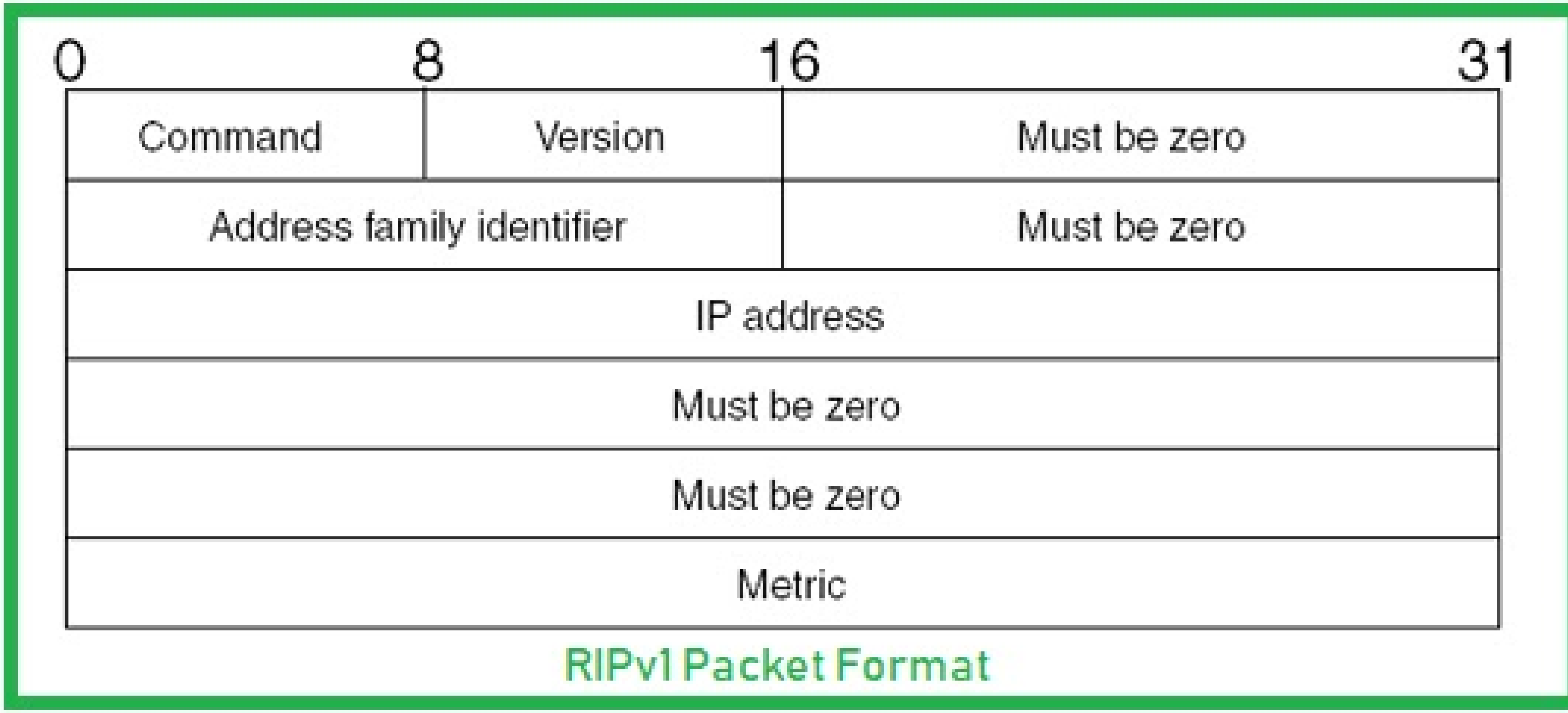
Takes shortest path

- If two are shortest, splits traffic on them equally

Route deletion

- If no update in 180 seconds or reaches infinity
- Marked as deleted
- Lives 120 seconds longer to send to others

RIP v1 (1986) Details



RIPv1 Packet Format

https://www.networkurge.com/2020/05/ripv1.html

RIP v2 (1998) Details

Added authentication

- Simple password
- Address family ID 0xFFFF

Route tag for exterior routing

```
 0                   1                   2                   3 3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Command (1)   | Version (1)   |            unused             |
+---------------+---------------+-------------------------------+
| Address Family Identifier (2) |         Route Tag (2)         |
+-------------------------------+-------------------------------+
|                         IP Address (4)                        |
+---------------------------------------------------------------+
|                         Subnet Mask (4)                       |
+---------------------------------------------------------------+
|                         Next Hop (4)                          |
+---------------------------------------------------------------+
|                         Metric (4)                            |
+---------------------------------------------------------------+
```

Next hop

- 0.0.0.0 for the sender
- Another address must be on the subnet, allows just some routers to run RIP

Message sent via Multicast 224.0.0.9

```
 0                   1                   2                   3 3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Command (1)   | Version (1)   |            unused             |
+---------------+---------------+-------------------------------+
|             0xFFFF            |    Authentication Type (2)    |
+-------------------------------+-------------------------------+
~                       Authentication (16)                     ~
+---------------------------------------------------------------+
```

Conclusion

- IP Subnetting
- IPv6
- Routing Introduction
- RIP