Network Layer and Link State Routing
Network Layer Outline

- IP Issues
  - Fragmentation, addressing, subnets
- DHCP
- Network Address Translation (NAT)
- Link State Routing
  - Reliable Flooding
  - Dijkstra’s Algorithm
- Hierarchical Routing
- RIP, OSPF, BGP
Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What’s inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing
### IP Datagram Format

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP protocol version</td>
<td>IP protocol version number.</td>
</tr>
<tr>
<td>header length</td>
<td>32 bits</td>
</tr>
<tr>
<td>“type” of data</td>
<td>16-bit identifier</td>
</tr>
<tr>
<td>max number remaining hops</td>
<td>Time to live (decremented at each router)</td>
</tr>
<tr>
<td>upper layer protocol</td>
<td>32 bit source IP address</td>
</tr>
<tr>
<td>to deliver payload</td>
<td>32 bit destination IP address</td>
</tr>
<tr>
<td>how much overhead with TCP?</td>
<td>Options (if any)</td>
</tr>
<tr>
<td>20 bytes of TCP</td>
<td>Data (variable length, typically a TCP or UDP segment)</td>
</tr>
<tr>
<td>20 bytes of IP</td>
<td></td>
</tr>
<tr>
<td>= 40 bytes + app layer overhead</td>
<td></td>
</tr>
</tbody>
</table>

- **Total datagram length**: (bytes) for fragmentation/reassembly
- **E.g. timestamp, record route taken, specify list of routers to visit.**

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

**Notes:**
- For fragmentation/reassembly purposes, the header contains the following fields:
  - **Identification**: 16-bit identifier for tracking fragments.
  - **Flags**: Flags indicating whether fragmentation is required.
  - **Fragment Offset**: Indicates the offset of the fragment within the original datagram.
- The upper layer protocol is responsible for delivering the payload to the intended service.
- Network links have MTU (max. transfer size) - largest possible link-level frame.
  - Different link types, different MTUs
- Large IP datagram divided ("fragmented") within net
  - One datagram becomes several datagrams
  - "Reassembled" only at final destination
  - IP header bits used to identify, order related fragments
IP Fragmentation and Reassembly

Example

- 4000 byte datagram
- MTU = 1500 bytes

One large datagram becomes several smaller datagrams

1480 bytes in data field

Offset = 1480/8
Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What’s inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing
IP Addressing: Introduction

- **IP address**: 32-bit identifier for host, router **interface**
- **interface**: connection between host/router and physical link
  - router’s typically have multiple interfaces
  - host typically has one interface
  - IP addresses associated with each interface

![Diagram of IP addressing](image)
Subnets

- **IP address:**
  - subnet part (high order bits)
  - host part (low order bits)

- **What's a subnet?**
  - device interfaces with same subnet part of IP address.
  - can physically reach each other without intervening router.

network consisting of 3 subnets
Recipe

- To determine the subnets, detach each interface from its host or router, creating islands of isolated networks. Each isolated network is called a subnet.

Subnet mask: /24 :: defined by the leftmost 24 bits.
Subnets

How many?

Computer Networks  Network Layer
CIDR: Classless InterDomain Routing

- subnet portion of address of arbitrary length
- address format: \texttt{a.b.c.d/x}, where \texttt{x} is \# bits in subnet portion of address.

\begin{itemize}
  \item [\texttt{11001000}] \text{subnet part}
  \item [\texttt{00010111}] \text{host part}
  \item [\texttt{00010000}] \text{subnet part}
  \item [\texttt{00000000}] \text{host part}
\end{itemize}

\texttt{200.23.16.0/23}
Q: How does a host get IP address?

- hard-coded by system admin in a file
  - Windows: control-panel->network-configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- **DHCP:** Dynamic Host Configuration Protocol: dynamically get address from a server
  - A “plug-and-play” protocol
**Goal:** Allow a host to *dynamically* obtain its IP address from network server when it joins the network.

- Can renew its lease on address in use.
- Allows reuse of addresses (only hold address while connected an “on”).
- Support for mobile users who want to join network (more shortly).

**DHCP overview:**

1. host broadcasts “DHCP discover” msg [optional]
2. DHCP server responds with “DHCP offer” msg [optional]
3. host requests IP address: “DHCP request” msg
4. DHCP server sends address: “DHCP ack” msg
DHCP Client-Server Scenario

DHCP server

arriving DHCP client needs address in this network
DHCP Client-Server Scenario

DHCP server: 223.1.2.5

1. DHCP discover
   - src: 0.0.0.0, 68
   - dest.: 255.255.255.255, 67
   - yiaddr: 0.0.0.0
   - transaction ID: 654

2. DHCP offer
   - src: 223.1.2.5, 67
   - dest: 255.255.255.255, 68
   - yiaddr: 223.1.2.4
   - transaction ID: 654
   - Lifetime: 3600 secs

3. DHCP request
   - src: 0.0.0.0, 68
   - dest: 255.255.255.255, 67
   - yiaddr: 223.1.2.4
   - transaction ID: 655
   - Lifetime: 3600 secs

4. DHCP ACK
   - src: 223.1.2.5, 67
   - dest: 255.255.255.255, 68
   - yiaddr: 223.1.2.4
   - transaction ID: 655
   - Lifetime: 3600 secs
DHCP can return more than just allocated IP address on subnet:

- address of first-hop router for client
- name and IP address of DNS server
- network mask (indicating network versus host portion of address).
DHCP: Example

- connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP
  - DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.1 Ethernet
  - Ethernet frame broadcast (dest: FFFFFFFF0000) on LAN, received at router running DHCP server
  - Ethernet demux'ed to IP demux'ed, UDP demux'ed to DHCP
DHCP: Example

- DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server

- encapsulation of DHCP server, frame forwarded to client, demux'ing up to DHCP at client.

- client now knows its IP address, name and IP address of DNS server, IP address of its first-hop router.
DHCP: Wireshark Output (home LAN)

Message type: Boot Request (1)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP Request
Option: (61) Client identifier
  Length: 7; Value: 010016D323688A;
  Hardware type: Ethernet
  Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Option: (t=50,l=4) Requested IP Address = 192.168.1.101
Option: (t=12,l=5) Host Name = "nomad"
Option: (55) Parameter Request List
  Length: 11; Value: 010F03062C2E2F1F21F92B
  1 = Subnet Mask; 15 = Domain Name
  3 = Router; 6 = Domain Name Server
  44 = NetBIOS over TCP/IP Name Server

......

Message type: Boot Reply (2)
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 192.168.1.101 (192.168.1.101)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 192.168.1.1 (192.168.1.1)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP ACK
Option: (t=54,l=4) Server Identifier = 192.168.1.1
Option: (t=1,l=4) Subnet Mask = 255.255.255.0
Option: (t=3,l=4) Router = 192.168.1.1
Option: (6) Domain Name Server
  Length: 12; Value: 445747E2445749F244574092;
  IP Address: 68.87.71.226;
  IP Address: 68.87.73.242;
  IP Address: 68.87.64.146
Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."
NAT: Network Address Translation

All datagrams leaving local network have same single source NAT IP address: 138.76.29.7, different source port numbers

Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)
**Motivation:** local network uses just one IP address as far as outside world is concerned:

- range of addresses not needed from ISP: just one IP address 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).
Implementation: NAT router must:

- **outgoing datagrams:** replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  
  . . . remote clients/servers will respond using (NAT IP address, new port #) as destination address.

- **remember (in NAT translation table)** every (source IP address, port #) to (NAT IP address, new port #) translation pair

- **incoming datagrams:** replace (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table.
1: host 10.0.0.1 sends datagram to 128.119.40.186, 80

2: NAT router changes datagram source addr from 10.0.0.1, 3345 to 138.76.29.7, 5001, updates table

WAN side addr | LAN side addr
--- | ---
138.76.29.7, 5001 | 10.0.0.1, 3345

3: Reply arrives dest. address: 138.76.29.7, 5001

4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 10.0.0.1, 3345
NAT Traversal Problem

- client wants to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can’t use it as destination addr)
  - only one externally visible NATted address: 138.76.29.7
- Solution 1: statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (138.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000
NAT Traversal Problem

- Solution 2: Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATted host to:
  - learn public IP address (138.76.29.7)
  - add/remove port mappings (with lease times)

i.e., automate static NAT port map configuration
**NAT Traversal Problem**

- Solution 3: relaying (used in Skype)
  - NATed client establishes connection to relay
  - External client connects to relay
  - relay bridges packets between to connections

![Diagram showing NAT traversal](image-url)
Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What’s inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing
1. Each router is responsible for meeting its neighbors and learning their names.

2. Each router constructs a link state packet (LSP) which consists of a list of names and cost to reach each of its neighbors.

3. The LSP is transmitted to ALL other routers. Each router stores the most recently generated LSP from each other router.

4. Each router uses complete information on the network topology to compute the shortest path route to each destination node.
Figure 4.18 Reliable LSP Flooding

(a) X A C B D
(b) X A C B D
(c) X A C B D
(d) X A C B D

P&D slide
Reliable Flooding

• The process of making sure all the nodes participating in the routing protocol get a copy of the link-state information from all the other nodes.

• **LSP** contains:
  - Sending router’s node ID
  - List of connected neighbors with the associated link cost to each neighbor
  - Sequence number
  - Time-to-live (TTL) \{an aging mechanism\}
Reliable Flooding

- First two items enable route calculation.
- Last two items make process reliable
  - ACKs and checking for duplicates is needed.
- Periodic **Hello** packets used to determine the demise of a neighbor.
- The sequence numbers are not expected to wrap around.
  - ➔ this field needs to be large (64 bits)!!
Dijkstra’s algorithm

- net topology, link costs known to all nodes
  - accomplished via “link state broadcast”.
  - all nodes have same info.
- computes least cost paths from one node (‘source”) to all other nodes
  - gives forwarding table for that node.
- iterative: after k iterations, know least cost path to k destinations.

Notation:

- \( c(x,y) \): link cost from node \( x \) to \( y \); \( = \infty \) if not direct neighbors.
- \( D(v) \): current value of cost of path from source to destination \( v \)
- \( p(v) \): predecessor node along path from source to \( v \)
- \( N' \): set of nodes whose least cost path is definitively known.
Dijsktra's Algorithm [K&R]

1 Initialization:
2 N' = \{u\}
3 for all nodes v
4    if v adjacent to u
5       then D(v) = c(u,v)
6    else D(v) = \infty
7
8 Loop
9    find w not in N' such that D(w) is a minimum
10   add w to N'
11   update D(v) for all v adjacent to w and not in N':
12      D(v) = \text{min}( D(v), D(w) + c(w,v) )
13      /* new cost to v is either old cost to v or known
14      shortest path cost to w plus cost from w to v */
15 until all nodes in N'

Computer Networks  Network Layer
Dijkstra's Shortest Path Algorithm

Initially mark all nodes (except source) with infinite distance.
working node = source node
Sink node = destination node
While the working node is not equal to the sink
  1. Mark the working node as permanent.
  2. Examine all adjacent nodes in turn
     If the sum of label on working node plus distance from
     working node to adjacent node is less than current labeled
     distance on the adjacent node, this implies a shorter path.
     Relabel the distance on the adjacent node and label it with
     the node from which the probe was made.
  3. Examine all tentative nodes (not just adjacent nodes) and
     mark the node with the smallest labeled value as
     permanent. This node becomes the new working node.
Reconstruct the path backwards from sink to source.
## Dijkstra's Algorithm: Example

<table>
<thead>
<tr>
<th>Step</th>
<th>N'</th>
<th>D(v),p(v)</th>
<th>D(w),p(w)</th>
<th>D(x),p(x)</th>
<th>D(y),p(y)</th>
<th>D(z),p(z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>u</td>
<td>2,u</td>
<td>5,u</td>
<td>1,u</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>ux</td>
<td>2,u</td>
<td>4,x</td>
<td>2,x</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>2</td>
<td>uxy</td>
<td>2,u</td>
<td>3,y</td>
<td></td>
<td></td>
<td>4,y</td>
</tr>
<tr>
<td>3</td>
<td>uxyv</td>
<td>2,u</td>
<td>3,y</td>
<td></td>
<td></td>
<td>4,y</td>
</tr>
<tr>
<td>4</td>
<td>uxyvw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>uxyvwz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The diagram shows a network with nodes u, v, w, x, y, and z, and edges with weights as specified in the table. The algorithm is applied to find the shortest path from u to z.
Resulting shortest-path tree from u:

Resulting forwarding table in u:

<table>
<thead>
<tr>
<th>destination</th>
<th>link</th>
</tr>
</thead>
<tbody>
<tr>
<td>v</td>
<td>(u,v)</td>
</tr>
<tr>
<td>x</td>
<td>(u,x)</td>
</tr>
<tr>
<td>y</td>
<td>(u,x)</td>
</tr>
<tr>
<td>w</td>
<td>(u,x)</td>
</tr>
<tr>
<td>z</td>
<td>(u,x)</td>
</tr>
</tbody>
</table>
Algorithm complexity: \( n \) nodes
- each iteration: need to check all nodes, \( w \), not in \( N \)
- \( n(n+1)/2 \) comparisons: \( O(n^2) \)
- more efficient implementations possible: \( O(n \log n) \)

Oscillations possible:
- e.g., link cost = amount of carried traffic
Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What’s inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing
Hierarchical Routing

- Our routing study thus far - an idealization
- all routers identical
- network “flat”
- ... not true in practice

**scale:** with 200 million destinations:
- can’t store all destinations in routing tables!
- routing table exchange would swamp links!

**administrative autonomy**
- internet = network of networks
- each network admin may want to control routing in its own network
Hierarchical Routing

- aggregate routers into regions, “autonomous systems” (AS)
- routers in same AS run same routing protocol
  - “intra-AS” routing protocol
  - routers in different AS can run different intra-AS routing protocol

Gateway router
- Direct link to router in another AS
Interconnected AS's

- Forwarding table configured by both intra- and inter-AS routing algorithm
  - intra-AS sets entries for internal dests
  - inter-AS & intra-AS set entries for external dests
suppose router in AS1 receives datagram destined outside of AS1:

- router should forward packet to gateway router, but which one?

**AS1 must:**

1. learn which dests are reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1

Job of inter-AS routing!
Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What’s inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing
Intra-AS Routing

- also known as **Interior Gateway Protocols (IGP)**
- most common Intra-AS routing protocols:
  - **RIP**: Routing Information Protocol
  - **OSPF**: Open Shortest Path First
  - **IGRP**: Interior Gateway Routing Protocol (Cisco proprietary)
Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing
RIP had widespread use because it was distributed with BSD Unix in "routed", a router management daemon in 1982.

- **RIP** - most used Distance Vector protocol.
- RFC1058 in June 1988
- Runs over UDP.
- Metric = hop count
- BIG problem is max. hop count =16
  - RIP limited to running on small networks (or AS's that have a small diameter)!!
Routing Information Protocol (RIP)

- Sends DV packets every 30 seconds (or faster) as Response Messages (also called advertisements).
- each advertisement: list of up to 25 destination subnets within AS.
- Upgraded to RIPv2

From router A to subnets:

<table>
<thead>
<tr>
<th>destination</th>
<th>hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>u</td>
<td>1</td>
</tr>
<tr>
<td>v</td>
<td>2</td>
</tr>
<tr>
<td>w</td>
<td>2</td>
</tr>
<tr>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>y</td>
<td>3</td>
</tr>
<tr>
<td>z</td>
<td>2</td>
</tr>
</tbody>
</table>

Network Layer
## Figure 4.17 RIP Packet Format

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>Must be zero</td>
</tr>
<tr>
<td>Version</td>
<td></td>
</tr>
<tr>
<td>Family of net 1</td>
<td>Address of net 1</td>
</tr>
<tr>
<td>Address of net 1</td>
<td></td>
</tr>
<tr>
<td>Distance to net 1</td>
<td></td>
</tr>
<tr>
<td>Family of net 2</td>
<td>Address of net 2</td>
</tr>
<tr>
<td>Address of net 2</td>
<td></td>
</tr>
<tr>
<td>Distance to net 2</td>
<td></td>
</tr>
</tbody>
</table>

- **(network_address, distance)** pairs
OSPF (Open Shortest Path First)

- “open”: publicly available
- uses Link State algorithm
  - LS packet dissemination
  - topology map at each node
  - route computation using Dijkstra’s algorithm.

- OSPF advertisement carries one entry per neighbor router.
- advertisements disseminated to entire AS (via flooding)
  - carried in OSPF messages directly over IP (rather than TCP or UDP).
OSPF “Advanced” Features (not in RIP)

- **security**: all OSPF messages authenticated (to prevent malicious intrusion).

- **multiple same-cost paths** allowed (only one path in RIP).

- For each link, multiple cost metrics for different **TOS** (e.g., satellite link cost set “low” for best effort; high for real time).

- **integrated uni- and multicast** support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF.

- **hierarchical OSPF** in large domains.
Hierarchical OSPF
Hierarchical OSPF

- **two-level hierarchy**: local area, backbone.
  - Link-State Advertisements (LSAs) only in area
  - each node has detailed area topology; only knows direction (shortest path) to nets in other areas.
- **area border routers**: “summarize” distances to nets in own area, advertise to other Area Border routers.
- **backbone routers**: run OSPF routing limited to backbone.
- **boundary routers**: connect to other AS’s.
OSPF LSA Types

1. Router link advertisement [Hello message]
2. Network link advertisement
3. Network summary link advertisement
4. AS border router's summary link advertisement
5. AS external link advertisement
Chapter 4: Network Layer

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing
Internet Inter-AS routing: BGP

- BGP (Border Gateway Protocol): the de facto standard

- BGP provides each AS a means to:
  1. Obtain subnet reachability information from neighboring AS’s.
  2. Propagate the reachability information to all AS-internal routers.
  3. Determine “good” routes to subnets based on reachability information and policy.

- allows subnet to advertise its existence to rest of Internet: “I am here!”
Network Layer Summary

- **IP Issues**
  - Fragmentation, addressing, subnets
- **DHCP**
- **Network Address Translation (NAT)**
- **Link State Routing**
  - Reliable Flooding
  - Dijkstra’s Algorithm
- **Hierarchical Routing**
- **RIP, OSPF, BGP**