Week 4
Network Layer

These slides are modified from the slides made available by Kurose and Ross.

Network Layer functions
- transport packet from sending to receiving hosts
- network layer protocols in every host, router

Three important functions:
- path determination: route taken by packets from source to dest. Routing algorithms
- forwarding: move packets from router's input to appropriate router output
- call setup: some network architectures require router call setup along path before data flows

Network layer functions

Virtual circuits: signaling protocols
- used to setup, maintain, teardown VC
- used in ATM, frame-relay, X.25
- not used in today's Internet

Datagram networks: the Internet model
- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets forwarded using destination host address
  - packets between same source-dest pair may take different paths

Datagram or VC network: why?
Internet
- data exchange among computers
  - "elastic" service, no strict timing required
- "smart" end systems (computers)
  - can adapt, perform control, error recovery
  - simple inside network, complexity at "edge"

ATM
- evolved from telephony human conversation:
  - strict timing, reliability requirements
  - need for guaranteed service
- "dumb" end systems
- telephones
- complexity inside network

Routing
- Graph abstraction for routing algorithms:
  - graph nodes are routers
  - graph edges are physical links
  - link cost: delay, $ cost, or congestion level

"good" path:
- typically means minimum cost path
- other definitions possible
Routing Algorithm classification

Global or decentralized information?
- Global:
  - all routers have complete topology, link cost info
  - “link state” algorithms

Decentralized:
- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- “distance vector” algorithms

Static or dynamic?
- Static:
  - routes change slowly over time
- Dynamic:
  - routes change more quickly
  - periodic update
  - in response to link cost changes

A Link-State Routing Algorithm

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 routing table for that node
- iterative: after k iterations, know least cost path to k dest.

Notation:
- \( c(i,j) \): link cost from node i to j; cost infinite if not direct neighbors
- \( D(v) \): current value of cost of path from source to dest v
- \( p(v) \): predecessor node along path from source to v, that is next v

Dijsktra’s Algorithm

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

Distance Table Routing Algorithm

iterative:
- continues until no nodes exchange info
- self-terminating: no “signal” to stop
asynchronous:
- nodes need not exchange info/iterate in lock step
distributed:
- each node communicates only with directly-attached neighbors

Distance Table data structure:
- each node has its own row for each possible destination
- column for each directly-attached neighbor to node
- example: in node X, for dest. Y via neighbor Z:

\[
D(X,Z) = \text{distance from X to Y, via Z as next hop} = c(X,Z) + \min_{w} \{D^W(Y,w)\}
\]

Distance Vector Routing: overview

Iterative, asynchronous:
- each local iteration caused by:
  - local link cost change
  - message from neighbor: its least cost path change from neighbor
Distributed:
- each node notifies neighbors only when its least cost path to any destination changes
- neighbors then notify their neighbors if necessary

Each node:
- wait for (change in local link cost of msg from neighbor)
- recompute distance table
- if least cost path to any dest has changed, notify neighbors
**Distance Vector Algorithm:**

At all nodes, X:
1. Initialization:
   2. for all adjacent nodes v:
      3. \( D^X(v) = \infty \)  
      4. \( D^X(v,v) = c(X,v) \)
   5. for all destinations, y
      6. send \( \min \{D^X(y,v)\} \) to each neighbor  
     7. \( ^* \) w over all X’s neighbors 

**Distance Vector Algorithm (cont.):**

8. loop
9. wait (until I see a link cost change to neighbor V
10. or until I receive update from neighbor V)
11. if (c(X,V) changes by d)
12. change cost to all dest’s via neighbor v by d
13. note: d could be positive or negative
14. for all destinations y: \( D^X(y,V) = \min \{D^X(y,v)\} + d \)
15. else if (update received from V wrt destination Y)
16. shortest path from V to some Y has changed
17. call this received new value is “newval”
18. for the single destination y: \( D^X(Y,V) = c(X,V) + newval \)
19. if we have a new \( \min \{D^X(Y,v)\} \) for any destination Y
20. send new value of \( \min \{D^X(Y,v)\} \) to all neighbors
21. forever

### Comparison of LS and DV algorithms

**Message complexity**
- **LS:** with \( n \) nodes, \( E \) links, \( O(nE) \) msgs sent each
- **DV:** exchange between neighbors only
  - may converge time varies

**Speed of Convergence**
- **LS:** \( O(n^2) \) algorithm requires \( O(nE) \) msgs
  - may have oscillations
- **DV:** convergence time varies
  - may be routing loops
  - count-to-infinity problem

**Robustness:** what happens if router malfunctions?
- **LS:**
  - node can advertise incorrect link cost
  - each node computes only its own table
- **DV:**
  - DV node can advertise incorrect path cost
  - each node’s table used by others
    - error propagate thru network

### Hierarchical Routing

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

**scale:** with 200 million destinations:
- can’t store all dest’s 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

---

### Intra-AS and Inter-AS routing

**Gateway routers**
- special routers in AS
- run intra-AS routing protocol with all other routers in AS
- also responsible for routing to destinations outside AS
- run inter-AS routing protocol with other gateway routers
Network Layer 4-19

Intra-AS and Inter-AS routing

We’ll examine specific inter-AS and intra-AS Internet routing protocols shortly.

Network Layer 4-20

IP Addressing: introduction

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

Network Layer 4-21

IP Addressing

- IP address:
  - network part (high order bits)
  - host part (low order bits)
- What’s a network? (from IP address perspective)
  - device interfaces with same network part of IP address
  - can physically reach each other without intervening router

Network Layer 4-22

IP Addressing

- How to find the networks?
  - Detach each interface from router, host
  - create “islands of isolated networks”

Network Layer 4-23

IP Addresses

given notion of “network”, let’s re-examine IP addresses: “class-full” addressing:

- class A: network 1.0.0 to 127.255.255.255
- class B: network 128.0.0 to 191.255.255.255
- class C: network 192.0.0 to 223.255.255.255
- class D: multicast address 224.0.0 to 239.255.255.255

Network Layer 4-24

IP addressing: CIDR

- Classful addressing:
  - inefficient use of address space, address space exhaustion
  - e.g., class B net allocated enough addresses for 65K hosts, even if only 2K hosts in that network
- CIDR: Classless InterDomain Routing
  - network portion of address of arbitrary length
  - address format: a.b.c.d/x, where x is # bits in network portion of address
  - network part: 11001000 00010111 00010000 00000000
  - host part: 200.23.16.0/23
**IP addresses: how to get one?**

**Q:** How does host get IP address?

- hard-coded by system admin in a file
  - Wintel: control-panel->network->configuration->TCP/IP->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from server

---

**Network Layer 4-25**

**IP addresses: how to get one?**

**Q:** How does network get network part of IP addr?

**A:** gets allocated portion of its provider ISP's address space

<table>
<thead>
<tr>
<th>ISP's block</th>
<th>11001000 00010111 00010000 00000000</th>
<th>200.23.16.0/20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization 0</td>
<td>200.23.16.0/20</td>
<td>200.23.16.023</td>
</tr>
<tr>
<td>Organization 1</td>
<td>200.23.18.0/23</td>
<td>200.23.18.023</td>
</tr>
<tr>
<td>Organization 2</td>
<td>200.23.20.0/23</td>
<td>200.23.20.023</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Organization 7</td>
<td>200.23.30.0/23</td>
<td>200.23.30.023</td>
</tr>
</tbody>
</table>

---

**Network Layer 4-26**

**Hierarchical addressing: route aggregation**

Hierarchical addressing allows efficient advertisement of routing information:

- Organization 0
  - 200.23.16.0/23
  - 200.23.18.0/23
  - 200.23.20.0/23
  - 200.23.22.0/23

- Organization 1
  - 200.23.16.0/23
  - 200.23.18.0/23
  - 200.23.20.0/23

- Organization 7
  - 200.23.30.0/23

---

**Network Layer 4-27**

**Hierarchical addressing: more specific routes**

ISPs-R-Us has a more specific route to Organization 1

- 200.23.16.0/23
- 200.23.18.0/23
- 200.23.19.0/23

Fly-By-Night-ISP

- Organization 0
  - 200.23.16.0/23
  - 200.23.18.0/23
  - 200.23.20.0/23

- Organization 1
  - 200.23.21.0/23

- Organization 7
  - 200.23.30.0/23

Internet

---

**Network Layer 4-28**

**IP addressing: the last word...**

**Q:** How does an ISP get block of addresses?

**A:** ICANN: Internet Corporation for Assigned Names and Numbers

- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

---

**Network Layer 4-29**

**Getting a datagram from source to dest.**

<table>
<thead>
<tr>
<th>Min. fields</th>
<th>Source IP addr</th>
<th>Dest. IP addr</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>223.1.1</td>
<td>223.1.2</td>
<td>223.1.3</td>
<td>1</td>
</tr>
<tr>
<td>223.1.1</td>
<td>223.1.2</td>
<td>223.1.3</td>
<td>2</td>
</tr>
<tr>
<td>223.1.1</td>
<td>223.1.2</td>
<td>223.1.3</td>
<td>2</td>
</tr>
<tr>
<td>223.1.1</td>
<td>223.1.2</td>
<td>223.1.3</td>
<td>2</td>
</tr>
</tbody>
</table>

Forwarding table in A

- Dest. Net. | next router | N hops |
- 223.1.1.1  |             | 1      |
- 223.1.1.2  |             | 2      |
- 223.1.1.3  |             | 2      |

- datagram remains unchanged, as it travels source to destination
- addr fields of interest here

---

**Network Layer 4-30**
Getting a datagram from source to dest.

Getting a datagram from source to dest.

Getting a datagram from source to dest.

Getting a datagram from source to dest.

Getting a datagram from source to dest.

Getting a datagram from source to dest.

Getting a datagram from source to dest.

Getting a datagram from source to dest.

IP fragmentation & reassembly

IP Fragmentation and Reassembly

IP datagram format
**DHCP: Dynamic Host Configuration Protocol**

**Goal:** allow host to dynamically obtain its IP address from network server when it joins network
Can renew its lease on address in use
Allows reuse of addresses (only hold address while connected on “on”)
Support for mobile users who want to join network (more shortly)

DHCP overview:
- host broadcasts "DHCP discover" msg
- DHCP server responds with "DHCP offer" msg
- host requests IP address: "DHCP request" msg
- DHCP server sends address: "DHCP ack" msg

**DHCP client-server scenario**

**NAT: Network Address Translation**

- **Motivation:** local network uses just one IP address as far as outside world 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).

**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 addr.
- **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.
Routing in the Internet

- The Global Internet consists of Autonomous Systems (AS) interconnected with each other:
  - Stub AS: small corporation, one connection to other AS’s
  - Multihomed AS: large corporation (no transit), multiple connections to other AS’s
  - Transit AS: provider, hooking many AS’s together

- Two-level routing:
  - Intra-AS: administrator responsible for choice of routing algorithm within network
  - Inter-AS: unique standard for inter-AS routing: BGP

Internet AS Hierarchy

- Intra-AS border (exterior gateway) routers
- Inter-AS interior (gateway) routers

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)

RIP (Routing Information Protocol)

- Distance vector algorithm
- Included in BSD-UNIX Distribution in 1982
- Distance metric: # of hops (max = 15 hops)
- Distance vectors: exchanged among neighbors every 30 sec via Response Message (also called advertisement)
- Each advertisement: list of up to 25 destination nets within AS

NAT: Network Address Translation

- 16-bit port-number field:
  - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - routers should only process up to layer 3
  - violates end-to-end argument
    - NAT possibility must be taken into account by app designers, eg, P2P applications
  - address shortage should instead be solved by IPv6
RIP: Link Failure and Recovery

If no advertisement heard after 180 sec -->
neighbor/link declared dead
  ▸ routes via neighbor invalidated
  ▸ new advertisements sent to neighbors
  ▸ neighbors in turn send out new advertisements (if
tables changed)
  ▸ link failure info quickly propagates to entire net
  ▸ poison reverse used to prevent ping-pong loops
    (infinite distance = 16 hops)

RIP Table processing

- RIP routing tables managed by application-level
  process called route-d (daemon)
- advertisements sent in UDP packets, periodically
  repeated

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)

Inter-AS routing in the Internet: BGP

- BGP (Border Gateway Protocol): the de facto
  standard
- Path Vector protocol:
  ▸ similar to Distance Vector protocol
  ▸ each Border Gateway broadcast to neighbors
    (peers) entire path (i.e., sequence of AS's) to
    destination
  ▸ BGP routes to networks (ASs), not individual
    hosts
  ▸ E.g., Gateway X may send its path to dest. Z:
    Path (X,Z) = X,Y1,Y2,Y3,...,Z
### Internet inter-AS routing: BGP

**Suppose**: gateway X send its path to peer gateway W
- W may or may not select path offered by X
  - cost, policy (don’t route via competitors AS), loop prevention reasons.
- If W selects path advertised by X, then:
  - Path \((W, Z) = w, \) Path \((X, Z)\)
- Note: X can control incoming traffic by controlling it route advertisements to peers:
  - e.g., don’t want to route traffic to Z → don’t advertise any routes to Z

### Why different Intra- and Inter-AS routing?

**Policy:**
- Inter-AS: admin wants control over how its traffic routed, who routes through its net.
- Intra-AS: single admin, so no policy decisions needed

**Scale:**
- hierarchical routing saves table size, reduced update traffic

**Performance:**
- Intra-AS: can focus on performance
- Inter-AS: policy may dominate over performance

### Router Architecture Overview

Two key router functions:
- run routing algorithms/protocol (RIP, OSPF, BGP)
- switching datagrams from incoming to outgoing link

### IPv6

**Initial motivation**: 32-bit address space completely allocated by 2008.

**Additional motivation**:
- header format helps speed processing/forwarding
- header changes to facilitate QoS
- new “anycast” address: route to “best” of several replicated servers

**IPv6 datagram format**:
- fixed-length 40 byte header
- no fragmentation allowed

### IPv6 Header (Cont)

**Priority**: identify priority among datagrams in flow
**Flow Label**: identify datagrams in same "flow" (concept of "flow" not well defined).
**Next header**: identify upper layer protocol for data

### Other Changes from IPv4

**Checksum**: removed entirely to reduce processing time at each hop

**Options**: allowed, but outside of header, indicated by "Next Header" field

**ICMPv6**: new version of ICMP
- additional message types, e.g. "Packet Too Big"
- multicast group management functions
Transition From IPv4 To IPv6

- Not all routers can be upgraded simultaneously
  - no "flag days"
  - How will the network operate with mixed IPv4 and IPv6 routers?
- Two proposed approaches:
  - Dual Stack: some routers with dual stack (v6, v4) can "translate" between formats
  - Tunneling: IPv6 carried as payload in IPv4 datagram among IPv4 routers

Dual Stack Approach

- scenarios:
  - Flow: X
    - Src: A
    - Dest: F
    - data
  - B-to-C:
    - IPv4
    - B-to-C:
      - IPv6
      - B-to-C:
        - IPv4

Network Layer 4-61

Network Layer 4-62