Network Layer


Reference: Computer Networks: A Systems Approach. Larry Peterson, Bruce Davie, Morgan Kaufmann
Network Layer

- Network Layer
  - Routing (Control Plane)
  - Forwarding (Data Plane)
  - Router Architecture Overview

✓ Data Plane
  - Forwarding
  ✓ Internet Protocol (IP)
    - Generalized Forwarding and SDN

- Control Plane
  - Routing (Per-Router Control)
    - Algorithms
    - Protocols
    - Policies
  - Software Defined Networking (Logically Centralized Control)
IPv6

• **Initial motivation:** 32-bit address space soon to be completely allocated.

• **Additional motivation**
  • Header format helps speed processing/forwarding
  • Header changes to facilitate QoS

**IPv6 datagram format**
  • Fixed-length 40 byte header
  • No fragmentation allowed
IPv6 Datagram Format

**Priority:** Identify priority among datagrams in flow

**Flow Label:** Identify datagrams in same flow

**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
  • How will network operate with mixed IPv4 and IPv6 routers?

• Tunneling: IPv6 datagram carried as payload in IPv4 datagram among IPv4 routers
Tunneling

logical view:

physical view:

IPv4 tunnel connecting IPv6 routers
Tunneling

logical view:

physical view:

flow: X
src: A
dest: F
data

A-to-B: IPv6

B-to-C: IPv6 inside IPv4

src:B
dest: E
Flow: X
Src: A
Dest: F
data

B-to-C: IPv6 inside IPv4

e-to-F: IPv6

IPv4 tunnel connecting IPv6 routers
IPv6: Adoption

• Google: 8% of clients access services via IPv6

• NIST (National Institute of Standards and Technology): 1/3 of all US government domains are IPv6 capable

• Long time for deployment and use
  • 20 years and counting!
  • Think of application-level changes in last 20 years: WWW, Facebook, streaming media, Skype, …
  • Why?
Network Layer

• Network Layer
  • Routing (Control Plane)
  • Forwarding (Data Plane)
  • Router Architecture Overview

✓ Data Plane
  • Forwarding
  • Internet Protocol (IP)
  ✓ Generalized Forwarding and SDN

• Control Plane
  • Routing (Per-Router Control)
    • Algorithms
    • Protocols
    • Policies
  • Software Defined Networking (Logically Centralized Control)
Generalized Forwarding & SDN

• Each router contains a **flow table** that is computed and distributed by a **logically centralized** routing controller.
OpenFlow Data Plane Abstraction

- **Flow**: defined by header fields
- **Generalized forwarding**: simple packet-handling rules
  - **Pattern**: match values in packet header fields
  - **Actions** (for matched packet): drop, forward, modify, matched packet or send matched packet to controller
  - **Priority**: disambiguate overlapping patterns
  - **Counters**: Number of bytes & number of packets

- Flow table in a router (computed and distributed by controller) define router’s match+action rules
OpenFlow Data Plane Abstraction

1. src=1.2.3.*, dest=3.4.5.* → drop
2. src = *.*.*, dest=3.4.*.* → forward(2)
3. src=10.1.2.3, dest=*.*.*.* → send to controller

*: wildcard
OpenFlow: Flow Table Entries

1. Forward packet to port(s)
2. Encapsulate and forward to controller
3. Drop packet
4. Send to normal processing pipeline
5. Modify Fields
Example

Destination-based forwarding:

<table>
<thead>
<tr>
<th>Switch Port</th>
<th>MAC Src</th>
<th>MAC Dst</th>
<th>Eth Type</th>
<th>VLAN ID</th>
<th>IP Src</th>
<th>IP Dst</th>
<th>IP Prot</th>
<th>TCP Sport</th>
<th>TCP Dport</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51.6.0.8</td>
<td></td>
<td></td>
<td></td>
<td>port6</td>
</tr>
</tbody>
</table>

IP datagrams destined to IP address 51.6.0.8 should be forwarded to router output port 6

Firewall:

<table>
<thead>
<tr>
<th>Switch Port</th>
<th>MAC Src</th>
<th>MAC Dst</th>
<th>Eth Type</th>
<th>VLAN ID</th>
<th>IP Src</th>
<th>IP Dst</th>
<th>IP Prot</th>
<th>TCP Sport</th>
<th>TCP Dport</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>drop</td>
</tr>
</tbody>
</table>

do not forward (block) all datagrams destined to TCP port 22

<table>
<thead>
<tr>
<th>Switch Port</th>
<th>MAC Src</th>
<th>MAC Dst</th>
<th>Eth Type</th>
<th>VLAN ID</th>
<th>IP Src</th>
<th>IP Dst</th>
<th>IP Prot</th>
<th>TCP Sport</th>
<th>TCP Dport</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>128.119.1.1</td>
<td></td>
<td></td>
<td></td>
<td>drop</td>
</tr>
</tbody>
</table>

do not forward (block) all datagrams sent by host 128.119.1.1
Example

Destination-based layer 2 (switch) forwarding:

<table>
<thead>
<tr>
<th>Switch Port</th>
<th>MAC src</th>
<th>MAC dst</th>
<th>Eth type</th>
<th>VLAN ID</th>
<th>IP Src</th>
<th>IP Dst</th>
<th>IP Prot</th>
<th>TCP sport</th>
<th>TCP dport</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>22:A7:23:11:E1:02</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>port3</td>
</tr>
</tbody>
</table>

Layer 2 frames from MAC address 22:A7:23:11:E1:02 should be forwarded to output port 6
Open Flow Abstraction

- **Match+action**: Unifies different kinds of devices

- Router
  - **Match**: Longest destination IP prefix
  - **Action**: Forward out a link

- Switch
  - **Match**: Destination MAC address
  - **Action**: Forward or flood
Open Flow Abstraction

• Firewall
  • **Match**: IP addresses and TCP/UDP port numbers
  • **Action**: Permit or deny

• NAT
  • **Match**: IP address and port
  • **Action**: Rewrite address and port
**OpenFlow Example**

- **Example:** Datagrams from hosts h5 and h6 should be sent to h3 or h4, via s1 and from there to s2
Network Layer

• Network Layer
  • Routing (Control Plane)
  • Forwarding (Data Plane)
  • Router Architecture Overview

• Data Plane
  • Forwarding
  • Internet Protocol (IP)
  • Generalized Forwarding and SDN

✓ Control Plane
  ✓ Routing (Per-Router Control)
    • Algorithms
    • Protocols
    • Policies
  • Software Defined Networking (Logically Centralized Control)
Network Layer

• Two main functionalities
  • Routing
    • Determine the route taken by packets from source to destination
    • Happens in Control Plane
    • Network-wide logic
      • Per-router control
      • Centralized control
  
  • Forwarding
    • Moving packet from input to the appropriate output at each device (router) along the path
    • Happens in Data Plane
    • Local function at each device
“The process of determining systematically how to forward messages toward the destination node based on its address is called **Routing**.”
Control Plane: Routing

• **Routers:** “A device that is connected to two or more networks is called a router”

• **Routing**
  • **Traditional Routing (Per-Router Control)**
    • Static
    • Dynamic
    • **Routing Algorithms**
      • Distance Vector
      • Link State
    • Routing Protocols
    • Routing Policies
  • **Software Defined Networking (Logically Centralized Control)**
Routing Protocols

Routing protocol goal: Determine good paths (equivalently, routes), from sending hosts to receiving host, through network of routers

- **Path**: sequence of routers packets will traverse in going from given initial source host to given final destination host

- **Good**: least cost, fastest, least congested

- **Routing**: a top-10 networking challenge!
Graph Abstraction of the Networks

Graph: $G = (N, E)$

$N =$ set of routers = \{ u, v, w, x, y, z \}

$E =$ set of links \{ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) \}

**aside:** graph abstraction is useful in other network contexts, e.g., P2P, where $N$ is set of peers and $E$ is set of TCP connections
Graph Abstraction: Costs

Cost could always be 1, or inversely related to bandwidth, or inversely related to congestion.

\[ c(x,x') = \text{cost of link } (x,x') \text{ e.g., } c(w,z) = 5 \]

Key Question: what is the least-cost path between u and z?
Routing Algorithm: algorithm that finds that least cost path
Routing Algorithms: Distance Vector

• Each router only knows directly connected neighbors
• Algorithm: Bellman-Ford

\[ D_x(y) : \text{Cost of least cost path from } x \text{ to } y \]
\[ \text{Cost}(x,y) : \text{Cost of a direct link from } x \text{ to } y \]

Least cost path from \( x \) to \( y \) is the minimum of cost path to \( y \) through all direct neighbors of \( x \)

\[ D_x(y) = \min_v \{ \text{Cost}(x,v) + D_v(y) \} \]
Routing Algorithms: Distance Vector

\[ D_v(z) = 5 \]
\[ D_x(z) = 3 \]
\[ D_w(z) = 3 \]

\[ D_u(z) = \min \{ \text{Cost}(u,v) + D_v(z), \text{Cost}(u,x) + D_x(z), \text{Cost}(u,w) + D_w(z) \} \]
\[ = \min \{ 2+5, 1+3, 5+3 \} \]
\[ = 4 \]

The **next hop** to send packets to \( z \) is \( x \) with cost 4.
Does not keep the knowledge of the path after the next hop.
Routing Algorithms: Distance Vector

Bellman-Ford

Resulting forwarding table at 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>

Resulting path information at u:

- v: 2
- x: 1
- y: 2
- w: 3
- z: 4
Routing Algorithms: Distance Vector Analysis

• Iterative
  • Each local iteration cause by
    • Local link change
    • Update message from a neighbor

• Asynchronous
  • Iterations not in sync among the nodes

• Distributed
  • Each node notify neighbors only if changes
Routing Algorithms: Distance Vector Analysis

\[
D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} \\
= \min\{2+0, 7+1\} = 2
\]

\[
D_y(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(z)\} \\
= \min\{2+1, 7+0\} = 3
\]
Routing Algorithms: Distance Vector Analysis

\[ D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} \]
\[ = \min\{2+0 , 7+1\} = 2 \]

\[ D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} \]
\[ = \min\{2+1 , 7+0\} = 3 \]
Routing Algorithms: Distance Vector Analysis

• Link cost changes and link failures
  • Good news traverses fast
  • Bad news traverses slow
    • Count to infinity
  • Poisoned Reverse
    • If Z routes through Y to get to X
      • Z tells Y that Z’s distance to X is infinite
    • Will this completely solve count to infinity problem? No. Will not solve the loops involving three or more nodes
Routing Algorithms: Link State

• All routers have complete topology of the network & link cost info
• Algorithm: **Dijkstra’s shortest path**

{Initialization}

\[ M = \{s\} \]
\[
\text{for each } n \text{ in } N - \{s\} \\
\quad \text{if } n \text{ adjacent to } s, D(n) = \text{Cost}(s,n) \text{ otherwise } D(n) = \infty
\]

{Computation Loop}

\[
\text{while } (N \neq M) \\
\quad M = M \cup \{w\} \text{ such that } D(w) \text{ is the minimum for all } w \text{ in } (N - M) \\
\quad \text{for each } n \text{ in } (N - M) \\
\quad \quad D(n) = \text{MIN}(D(n), D(w) + \text{Cost}(w,n))
\]
Routing Algorithms: Link State

- All routers have complete topology of the network & link cost info
- Algorithm: Dijkstra’s shortest path

\[ M = \{s\} \]

\[ s \text{ is the source (starting) node} \]

\[ M \text{ is the set of examined nodes.} \]

\[ \text{for each } n \text{ in } N - \{s\} \]
\[ \text{if } n \text{ adjacent to } s, \ D(n) = \text{Cost}(s,n) \text{ otherwise } D(n) = \infty \]

\[ \text{while } (N \neq M) \]
\[ M = M \cup \{w\} \text{ such that } D(w) \text{ is the minimum for all } w \text{ in } (N - M) \]
\[ \text{for each } n \text{ in } (N - M) \]
\[ D(n) = \text{MIN}(D(n), D(w) + \text{Cost}(w,n)) \]
Routing Algorithms: Link State

- All routers have complete topology of the network & link cost info
- Algorithm: Dijkstra’s shortest path

Path Cost, is equal to Link Cost if there is a direct link between source $s$ and node $n$. The path cost is $\infty$ if there is no direct link between $s$ and $n$.

$N$ is set of all nodes in the graph

$M = \{s\}$

for each $n$ in $N \setminus \{s\}$ if $n$ adjacent to $s$, $D(n) = \text{Cost}(s,n)$ otherwise $D(n) = \infty$

while $(N \neq M)$

$M = M \cup \{w\}$ such that $D(w)$ is the minimum for all $w$ in $(N - M)$

for each $n$ in $(N - M)$

$D(n) = \text{MIN}(D(n), D(w) + \text{Cost}(w,n))$

Loop over every node $n$ in the graph set of nodes $N$ other than the source node $s$.
Routing Algorithms: Link State

- All routers have complete topology of the network & link cost info
- Algorithm: **Dijkstra’s shortest path**

\[
\begin{align*}
M &= \{s\} \\
\text{for each } n \text{ in } N - \{s\} \\
&\quad \text{if } n \text{ adjacent to } s, D(n) = \text{Cost}(s, n) \text{ otherwise } D(n) = \infty \\
\text{while (} N \neq M ) \\
M &= M \cup \{ w \} \text{ such that } D(w) \text{ is the minimum for all } w \text{ in } (N - M) \\
\text{for each } n \text{ in } (N - M) \\
D(n) &= \text{MIN}(D(n), D(w) + \text{Cost}(w, n))
\end{align*}
\]
Routing Algorithms: Link State

- All routers have complete topology of the network & link cost info
- Algorithm: **Dijkstra’s shortest path**

\[
M = \{s\} \\
\text{for each } n \text{ in } N - \{s\} \\
\quad \text{if } n \text{ adjacent to } s, \quad D(n) = \text{Cost}(s, n) \quad \text{otherwise } D(n) = \infty
\]

For each node \( w \), add \( w \) with minimum cost from \( s \) among all remaining nodes to the set \( M \)

while \( (N \neq M) \)
\[
M = M \cup \{w\} \text{ such that } D(w) \text{ is the minimum for all } w \text{ in } (N - M) \\
\text{for each } n \text{ in } (N - M) \\
\quad D(n) = \text{MIN}(D(n), D(w) + \text{Cost}(w, n))
\]
Routing Algorithms: Link State

• All routers have complete topology of the network & link cost info
• Algorithm: Dijkstra’s shortest path

For each node \( n \), if the cost of reaching \( n \) from \( s \) (current source node) through \( w \) is less than currently stored cost, update the cost to the path cost going through node \( w \)

For each node \( n \) in \( N - \{s\} \)
- if \( n \) adjacent to \( s \), \( D(n) = \text{Cost}(s,n) \) otherwise \( D(n) = \infty \)

while \( (N \neq M) \)
- \( M = M \cup \{w\} \) such that \( D(w) \) is the minimum for all \( w \) in \( (N - M) \)

for each \( n \) in \( (N - M) \)
- \( D(n) = \text{MIN}(D(n), D(w) + \text{Cost}(w,n)) \)
Routing Algorithms: Link State

Dijkstra’s shortest path
Min cost calculation. Cost updates from *every other node in the network*, not just the neighbors.

\[
\begin{align*}
D(v) &= 2 \text{ with path } \{u,v\} \\
D(x) &= 1 \text{ with path } \{u,x\} \\
D(w) &= 3 \text{ with path } \{u,x,w\} \\
D(y) &= 2 \text{ with path } \{u,x,y\} \\
D(z) &= \min \{D(v) + \text{Cost}(v,z), \\
& \quad D(x) + \text{Cost}(x,z), \\
& \quad D(w) + \text{Cost}(w,z), \\
& \quad D(y) + \text{Cost}(y,z)\} \\
&= \min \{2 + \infty, \\
& \quad 1 + \infty, \\
& \quad 3 + 5, \\
& \quad 2 + 2\} \\
&= 4
\end{align*}
\]

The *path* from u to z goes through \(\{u,x,y,z\}\) with cost 4.
Routing Algorithms: Link State Analysis

Dijkstra’s shortest path

Resulting forwarding table at 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>

Resulting shortest-path tree from u:
Routing Algorithms: Distance Vector or Link State?

• Message Exchange
  • LS: Exchange with all nodes in the network $O(|N||E|)$
  • DV: Exchange with neighbors only

• Convergence
  • LS: Compute path from each node to every other node ($O(N^2)$)
  • DV: Decide next directly connected node from each node to every other node
    • Routing Loops & Count to Infinity Problem

• Robustness
  • LS: Error on the incorrect advertised link
  • DV: Error propagates through the network

• Store
  • LS: All paths
  • DV: Next hop
Acknowledgements

• The following materials have been used in preparation of this slide set:

   7th Edition
   James Kurose, Keith Ross
   Pearson
   2016

   5th Edition
   Larry Peterson, Bruce Davie
   Morgan Kaufmann
   2011