Communication Networks
Prof. Laurent Vanbever

Please register your group for the projects
https://comm-net.ethz.ch by Friday

Last week on
Communication Networks

We started looking at the two fundamental challenges underlying networking

Routing is the control-plane process that computes and populates the forwarding tables

Forwarding vs Routing summary

<table>
<thead>
<tr>
<th></th>
<th>forwarding</th>
<th>routing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>goal</strong></td>
<td>directing packet to an outgoing link</td>
<td>computing the paths packets will follow</td>
</tr>
<tr>
<td><strong>scope</strong></td>
<td>local</td>
<td>network-wide</td>
</tr>
<tr>
<td><strong>implem.</strong></td>
<td>hardware</td>
<td>software</td>
</tr>
<tr>
<td><strong>timescale</strong></td>
<td>nanoseconds</td>
<td>10s of ms</td>
</tr>
</tbody>
</table>
The goal of routing is to compute valid global forwarding state.

**Definition**

A global forwarding state is valid if it always delivers packets to the correct destination.

**Theorem**

A global forwarding state is valid if and only if:
- There are no dead ends
- No outgoing port defined in the table
- There are no loops
- Packets going around the same set of nodes

Producing valid routing state is harder but doable.

How do we verify that a forwarding state is valid?

**Question 2**

How do we compute valid forwarding state?

You acted as IP routers and run a flavor of distributed Breadth-First-Search (BFS).

We then looked at the three ways to compute valid routing state:

1. **Intuition**
   - Use tree-like topologies
   - Spanning-tree

2. **Intuition**
   - Rely on a global network view
   - Link-State SDN

3. **Intuition**
   - Rely on distributed computation
   - Distance-Vector BGP

The easiest way to avoid loops is to route traffic on a loop-free topology.

**Intuition**

- Simple algorithm:
  - Take an arbitrary topology
  - Build a spanning tree and ignore all other links
  - Done!

- Why does it work?
  - Spanning trees have only one path between any two nodes
If each router knows the entire graph, it can locally compute paths to all other nodes.

### Use tree-like topologies
- Spanning-tree

### Rely on a global network view
- Link-State
- SDN

### Rely on distributed computation
- Distance-Vector
- BGP

**Initialization**

\[
S = \{u\} \\
for \text{all nodes } v: \\
\text{if } (v \text{ is adjacent to } u): \\
D(v) = c(u,v) \\
\text{else: } \\
D(v) = \infty
\]

**Loop**

\[
\text{while not all nodes in } S: \\
\text{add } w \text{ with the smallest } D(w) \text{ to } S \\
\text{update } D(v) \text{ for all adjacent } v \text{ not in } S: \\
D(v) = \min\{D(v), D(w) + c(w,v)\}
\]

**This week**

on Communication Networks

**We’ll do that layer-by-layer, bottom-up, starting with the Link layer**
Communication Networks
Part 2: The Link Layer

How do local computers communicate?

eth0 --|-- eth1

How do local computers communicate?
eth0
eth1

Network adapters communicate together through the medium

The Link Layer provides a best-effort delivery service to the Network layer

encoding represents the 0s and the 1s
framing encapsulate packet into a frame
adding header and trailer
error detection detects errors with checksum
error correction optionally corrects errors
flow control pace sending and receiving node

The Link Layer provides a best-effort delivery service to the Network layer, composed of 5 sub-services

<table>
<thead>
<tr>
<th>Level</th>
<th>Layer</th>
<th>Delivery</th>
</tr>
</thead>
<tbody>
<tr>
<td>L3</td>
<td>Network</td>
<td>global best-effort delivery</td>
</tr>
<tr>
<td>L2</td>
<td>Link</td>
<td>local best-effort delivery</td>
</tr>
<tr>
<td>L1</td>
<td>Physical</td>
<td>physical transfer of bits</td>
</tr>
</tbody>
</table>
As of March 2020, State-of-the-art Ethernet adapters clock at **200 Gbps**

*source: [Mellanox ConnectX-6]*

- 215 million pkt/sec
- sub 0.8 usec latency
- PCIe Gen 4.0

---

**Communication Networks**

**Part 2: The Link Layer**

- What is a link?
- #2 How do we identify link adapters?
- How do we share a network medium?
- What is Ethernet?
- How do we interconnect segments at the link layer?

---

**Medium Access Control addresses**

- MAC addresses are hierarchically allocated
  - 34:36:3b:d2:8a:86

- The first 24 bits blocks are assigned to network adapter vendor by the IEEE
  - 34:36:3b:d2:8a:86
  - assigned by Apple to my adapter

- The second 24 bits block is assigned by the vendor to each network adapter
  - 34:36:3b:22:8a:86

---

- MAC addresses are uniquely assigned
- hard-coded into the adapter when built
- use a flat space of 48 bits
- allocated hierarchically

---

*Medium Access Control addresses*
Communication Networks | Mon 9 March 2020

The address with all bits set to 1 identifies the broadcast address

```
ff:ff:ff:ff:ff:ff
```

enables to send a frame to all adapters on the link.

By default, adapters only decapsulate frames addressed to the local MAC or the broadcast address.

The promiscuous mode enables to decapsulate everything, independently of the destination MAC.

Why don’t we simply use IP addresses?

- Links can support any protocol (not just IP)
- Different addresses on different kinds of links
- Adapters may move to different locations
- Cannot assign static IP address; it has to change
- Adapters must be identified during bootstrap
- Need to talk to an adapter to give it an IP address

You need to solve two problems when you bootstrap an adapter:

- **Who am I?**
  - MAC-to-IP binding

- **Who are you?**
  - IP-to-MAC binding

Adapters must be identified during bootstrap.

- Need to talk to an adapter to give it an IP address

Network adapters traditionally acquire an IP address using the Dynamic Host Configuration Protocol (DHCP):

- **Who am I?**
  - MAC-to-IP binding
  - Dynamic Host Configuration Protocol

- **Who are you?**
  - IP-to-MAC binding
  - Address Resolution Protocol

Given an IP address reachable on a link, how do I find out what MAC to use?
Every connected device needs an IP address.

Host sends an "IP request" to everyone on the link using the broadcast address.

DHCP server (if any) answers with an IP address.

The Address Resolution Protocol (ARP) enables a host to discover the MAC associated to an IP.
I want to send an IP packet to 192.168.1.10? What destination MAC do I use?!

Who has 192.168.1.10? Tell 192.168.1.9

ff:ff:ff:ff:ff:ff

payload

ARP request

Who has 192.168.1.10? Tell 192.168.1.9

ff:ff:ff:ff:ff:ff

payload

ARP reply

192.168.1.10 is at 34:36:3b:d2:8a:86

34:36:3b:8a:10

192.168.1.9

Some medium are multi-access:
>1 host can communicate at the same time
Some medium are multi-access: >1 host can communicate at the same time

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>collisions lead to garbled data</td>
<td>distributed algorithm for sharing the channel</td>
</tr>
<tr>
<td>When can each node transmit?</td>
<td></td>
</tr>
</tbody>
</table>

Essentially, there are three techniques to deal with Multiple Access Control (MAC)

- Divide the channel into pieces either in time or in frequency
- Take turns pass a token for the right to transmit
- Random access allow collisions, detect them and then recover

Communication Networks
Part 2: The Link Layer

What is a link?
How do we identify link adapters?
How do we share a network medium?

#4 What is Ethernet?
How do we interconnect segments at the link layer?

Ethernet offers an unreliable, connectionless service

- unreliable Receiving adapter does not acknowledge anything
- Packets passed to the network layer can have gaps which can be filled by the transport protocol (TCP)
- connectionless No handshaking between the send and receive adapter

“Traditional” Ethernet relies on CSMA/CD

CSMA/CD imposes limits on the network length

Suppose A sends a packet at time t
B sees an idle line just before t+d and sends a packet

Effect
B would detect a collision and sends a jamming signal
A can detect the collision only after t+2d

For this reason, Ethernet imposes a minimum packet size (512 bits)

This imposes restriction on the length of the network

\[
\text{Network length} \leq \frac{\text{min\_frame\_size} \times \text{speed of light}}{2 \times \text{bandwidth}}
\]

= 768 meters for 100 Mbps

What about for 1 Gbps, 10 Gbps, 100 Gbps?
Modern Ethernet links interconnect exactly two hosts, in full-duplex, rendering collisions impossible!

CSMA/CD is only needed for half-duplex communications. 10 Gbps Ethernet does not even allow half-duplex anymore.

This means the 64 bytes restriction is not strictly needed but IEEE chose to keep it.

Multiple Access Protocols are still important for Wireless important concepts to know in practice.

The Ethernet header is simple, composed of 6 fields only:

<table>
<thead>
<tr>
<th>preamble</th>
<th>dest address</th>
<th>src address</th>
<th>type</th>
<th>data</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>used for synchronization</td>
<td>usually, IPv4 (0x0800)</td>
<td>Cyclic Redundant Check</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ethernet efficiency (payload/tot. frame size): ~97.5%

Maximum throughput for 100 Mbps: ~97.5 Mbps

Communication Networks
Part 2: The Link Layer

What is a link?

How do we identify link adapters?

How do we share a network medium?

What is Ethernet?

How do we interconnect segments at the link layer?

#5

Historically, people connected Ethernet segments together at the physical level using Ethernet hubs

Hubs work by repeating bits from one port to all the other ones

Local Area Networks are now almost exclusively composed of Ethernet switches

Hubs are now OBSOLETE

advantages
disadvantages

simple, cheap
inefficient, each bit is sent everywhere
limits the aggregates throughput
limited to one LAN technology
can’t interconnect different rates/formats
limited number of nodes and distances
cannot go beyond 2500m on Ethernet
Switches connect two or more LANs together at the Link layer, acting as L2 gateways.

Switches are "store-and-forward" devices, they:
- extract the destination MAC from the frame
- look up the MAC in a table (using exact match)
- forward the frame on the appropriate interface

Switches are similar to IP routers, except that they operate one layer below.

Unlike with hubs, switches enable each LAN segment to carry its own traffic.

Switches are plug-and-play devices, they build their forwarding table on their own.

Switches are plug-and-play devices, they build their forwarding table on their own.

In cases of misses, switches simply floods the frames.

When a frame arrives:
- inspect the source MAC address
- associate the address with the port
- store the mapping in the switch table
- launch a timer to eventually forget the mapping

When a frame arrives with an unknown destination:
- forward the frame out of all interfaces except for the one where the frame arrived

Hopefully, this is an unlikely event.
While flooding enables automatic discovery of hosts, it also creates problems when the networks has loops.

Each frame leads to the creation of at least two new frames! exponential increase, with no TTL to remove looping frames...

While loops create major problems, networks need redundancy for tolerating failures!

solution
Reduce the network to one logical spanning tree
Upon failure, automatically rebuild a spanning tree.

In practice, switches run a distributed Spanning-Tree Protocol (STP)

Algorhyme
I think that I shall never see
A graph more lovely than a tree.
A tree whose crucial property
Is loop-free connectivity.
A tree that must be sure to span
So packets can reach every LAN.
First, the root must be selected.
By ID, it is elected.
Least-cost paths from root are traced.
In the tree, these paths are placed.
A mesh is made by folks like me,
Then bridges find a spanning tree.
— Radia Perlman

Constructing a Spanning Tree in a nutshell

Switches...

elect a root switch
the one with the smallest identifier

determine if each interface is
on the shortest-path from the root
and disable it if not

For this switches exchange
Bridge Protocol Data Unit (BDPU) messages

initially
Each switch proposes itself as root
sends (X,0,X) on all its interfaces

Upon receiving (Y, d, X), checks if Y is a better root
if so, considers Y as the new root, flood updated message
Switches compute their distance to the root, for each port
simply add 1 to the distance received, if shorter, flood
Switches disable interfaces not on shortest-path

A tree that must be sure to span
So packets can reach every LAN.
First, the root must be selected.
By ID, it is elected.
Least-cost paths from root are traced.
In the tree, these paths are placed.
A mesh is made by folks like me,
Then bridges find a spanning tree.

Least-cost paths from root are traced. In the tree, these paths are placed. A mesh is made by folks like me, Then bridges find a spanning tree.

Switches compute their distance to the root, for each port simply add 1 to the distance received, if shorter, flood
Switches disable interfaces not on shortest-path
Upon receiving BPDUs from a neighboring switch:
Pick the BDU with the lowest port ID (e.g., port 2 < port 3).

Upon receiving BPDUs from switches with the same cost:
Pick the BDU with the lower switch sender ID.

Tie-breaking:

Apply the algorithm starting with switch 4:

To be robust, STP must react to failures:

- Any switch, link or port can fail, including the root switch.
- Root switch continuously sends messages announcing itself as the root (1,0,1); others forward it.
- Failures are detected through timeout (soft state).
  - If no word from root in \( X \), times out and claims to be the root.