Outline today

- **IP Anycast**
  - N destinations, 1 should receive the message
  - Providing a service from multiple network locations
  - Using routing protocols for automated failover

- **Multicast protocols**
  - N destinations, N should receive the message
  - Examples
    - IP Multicast
    - SRM (Scalable Reliable Multicast)
    - PGM (Pragmatic General Multicast)

Limitations of DNS-based failover

- Failover/load balancing via multiple A records
  ```
  ;; ANSWER SECTION:
  www.cnn.com. 300 IN A 157.166.255.19
  www.cnn.com. 300 IN A 157.166.224.25
  www.cnn.com. 300 IN A 157.166.226.26
  www.cnn.com. 300 IN A 157.166.255.18
  ```
- If server fails, service unavailable for TTL
  - Very low TTL: Extra load on DNS
  - Anyway, browsers cache DNS mappings 😊
- What if root NS fails? All DNS queries take > 3s?

Motivation for IP anycast

- Failure problem: client has resolved IP address
  - What if IP address can represent many servers?
- Load-balancing/failover via IP addr, rather than DNS
- IP anycast is simple reuse of existing protocols
  - Multiple instances of a service share same IP address
  - Each instance announces IP address / prefix in BGP / IGP
  - Routing infrastructure directs packets to nearest instance of the service
  - Can use same selection criteria as installing routes in the FIB
  - No special capabilities in servers, clients, or network

IP anycast in action
From client/router perspective, topology could as well be:

### Routing Table from Router 1:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Mask</th>
<th>Next-Hop</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.0</td>
<td>/29</td>
<td>127.0.0.1</td>
<td>0</td>
</tr>
<tr>
<td>10.0.0.0</td>
<td>/28</td>
<td>192.168.0.1</td>
<td>1</td>
</tr>
<tr>
<td>10.0.0.0</td>
<td>/28</td>
<td>192.168.0.2</td>
<td>2</td>
</tr>
</tbody>
</table>

DNS lookup for http://www.server.com/
produces a single answer:

www.server.com. IN A 10.0.0.1
Downsides of IP anycast

• Many Tier-1 ISPs ingress filter prefixes > /24
  – Publish a /24 to get a “single” anycasted address: Poor utilization

• Scales poorly with the # anycast groups
  – Each group needs entry in global routing table

• Not trivial to deploy
  – Obtain an IP prefix and AS number; speak BGP

Downsides of IP anycast

• Subject to the limitations of IP routing
  – No notion of load or other application-layer metrics
  – Convergence time can be slow (as BGP or IGP converge)

• Failover doesn’t really work with TCP
  – TCP is stateful: if switch destination replicas, other server instances will just respond with RSTs
  – May react to network changes, even if server online

• Root nameservers (UDP) are anycasted, little else

Multicast

• Many receivers
  – Receiving the same content

• Applications
  – Video conferencing
  – Online gaming
  – IP television (IPTV)
  – Financial data feeds

Iterated Unicast

• Unicast message to each recipient

• Advantages
  – Simple to implement
  – No modifications to network

• Disadvantages
  – High overhead on sender
  – Redundant packets on links
  – Sender must maintain list of receivers

IP Multicast

• Embed receiver-driven tree in network layer
  – Sender sends a single packet to the group
  – Receivers “join” and “leave” the tree

• Advantages
  – Low overhead on the sender
  – Avoids redundant network traffic

• Disadvantages
  – Control-plane protocols for multicast groups
  – Overhead of duplicating packets in the routers
Multicasting messages

- **Simple application multicast**: Iterated unicast
  - Client simply unicasts message to every recipient
  - **Pros**: simple to implement, no network modifications
  - **Cons**: \(O(n)\) work on sender, network

- **Advanced overlay multicast** ("peer-to-peer")
  - Build receiver-driven tree
  - **Pros**: Scalable, no network modifications
  - **Cons**: \(O(\log n)\) work on sender, network; complex to implement

- **IP multicast**
  - Embed receiver-driven tree in network layer
  - **Pros**: \(O(1)\) work on client, \(O(#\ of\ receivers)\) on network
  - **Cons**: requires network modifications; scalability concerns?

**Multicast Tree**

IP multicast in action

- **Single vs. Multiple Senders**
  - **Source-based tree**
    - Separate tree for each sender
    - Tree is optimized for that sender
    - But, requires multiple trees for multiple senders
  - **Shared tree**
    - One common tree
    - Spanning tree that reaches all participants
    - Single tree may be inefficient
    - But, avoids having many different trees

**Multicast Addresses**

- **Multicast “group”** defined by IP address
  - Multicast addresses look like unicast addresses
  - 224.0.0.0 to 239.255.255.255

- **Using multicast IP addresses**
  - Sender sends to the IP address
  - Receivers join the group based on IP address
  - Network sends packets along the tree

Example Multicast Protocol

- Receiver sends a “join” messages to the sender
  - And grafts to the tree at the nearest point
IGMP v1

- Two types of IGMP msgs (both have IP TTL of 1)
  - Host membership query: Routers query local networks to discover which groups have members
  - Host membership report: Hosts report each group (e.g., multicast addr) to which they belong, by broadcast on net interface from which query was received

- Routers maintain group membership
  - Host senders an IGMP “report” to join a group
  - Multicast routers periodically issue host membership query to determine liveness of group members
  - Note: No explicit “leave” message from clients

IGMP: Improvements

- IGMP v2 added:
  - If multiple routers, one with lowest IP elected querier
  - Explicit leave messages for faster pruning
  - Group-specific query messages

- IGMP v3 added:
  - Source filtering: Join specifies multicast “only from” or “all but from” specific source addresses

IGMP: Parameters and Design

- Parameters
  - Maximum report delay: 10 sec
  - Membership query internal default: 125 sec
  - Time-out interval: 270 sec = 2 * (query interval + max delay)

- Router tracks each attached network, not each peer

- Should clients respond immediately to queries?
  - Random delay (from 0..D) to minimize responses to queries
  - Only one response from single broadcast domain needed

- What if local networks are layer-2 switched?
  - L2 switches typically broadcast multicast traffic out all ports
  - Or, IGMP snooping (sneak peek into layer-3 contents), Cisco’s proprietary protocols, or static forwarding tables

IP Multicast is Best Effort

- Sender sends packet to IP multicast address
  - Loss may affect multiple receivers

Challenges for Reliable Multicast

- Send an ACK, much like TCP?
  - ACK-implosion if all destinations ACK at once
  - Source does not know # of destinations

- How to retransmit?
  - To all? One bad link effects entire group
  - Only where losses? Loss near sender makes retransmission as inefficient as replicated unicast

- Negative acknowledgments more common

Scalable Reliable Multicast

- Data packets sent via IP multicast
  - Data includes sequence numbers

- Upon packet failure
  - If failures relatively rare, use Negative ACKs (NAKs) instead: “Did not receive expected packet”
  - Sender issues heartbeats if no real traffic. Receiver knows when to expect (and thus NAK)
Handling Failure in SRM

- Receiver multicasts a NAK
  - Or send NAK to sender, who multicasts confirmation

- Scale through NAK suppression
  - If received a NAK or NCF, don’t NAK yourself
  - Add random delays before NAK’ing

- Repair through packet retransmission
  - From initial sender
  - From designated local repairer

Outline today

- **IP Anycast**
  - N destinations, 1 should receive the message
  - Providing a service from multiple network locations
  - Using routing protocols for automated failover

- **Multicast protocols**
  - N destinations, N should receive the message
  - Examples
    - IP Multicast and IGMP
    - SRM (Scalable Reliable Multicast)
    - PGM (Pragmatic General Multicast)

Pragmatic General Multicast (RFC 3208)

- Similar approach as SRM: IP multicast + NAKs
  - ... but more techniques for scalability

- Hierarchy of PGM-aware network elements
  - NAK suppression: Similar to SRM
  - NAK elimination: Send at most one NAK upstream
    - Or completely handle with local repair!
  - **Constrained forwarding**: Repair data can be suppressed downstream if no NAK seen on that port
  - **Forward-error correction**: Reduce need to NAK

- Works when only sender is multicast-able