Internet Indirection Infrastructure

Ion Stoica
UC Berkeley

Motivations

• Today’s Internet is built around a unicast point-to-point communication abstraction:
  – Send packet “p” from host “A” to host “B”
• This abstraction allows Internet to be highly scalable and efficient, but…
• … not appropriate for applications that require other communications primitives:
  – Multicast
  – Anycast
  – Mobility
  – …

Why?

• Point-to-point communication → implicitly assumes there is one sender and one receiver, and that they are placed at fixed and well-known locations
  – E.g., a host identified by the IP address 128.32.xxx.xxx is located in Berkeley

Key Observation

• Virtually all previous proposals use indirection, e.g.,
  – Physical indirection point → mobile IP
  – Logical indirection point → IP multicast

“Any problem in computer science can be solved by adding a layer of indirection”

Our Solution

Build an efficient indirection layer on top of IP

• Use an overlay network to implement this layer
  – Incrementally deployable; don’t need to change IP

Internet Indirection Infrastructure (i3)

• Each packet is associated an identifier id
• To receive a packet with identifier id, receiver R maintains a trigger (id, R) in the overlay network
Service Model

- API
  - sendPacket(p);
  - insertTrigger(t);
  - removeTrigger(t) // optional
- Best-effort service model (like IP)
- Triggers periodically refreshed by end-hosts
- ID length: 256 bits

Anycast

- Use longest prefix matching instead of exact matching
- Prefix p: anycast group identifier
- Suffix s: encode application semantics, e.g., location

Mobility

- Host just needs to update its trigger as it moves from one subnet to another

Service Composition: Sender Initiated

- Use a stack of IDs to encode sequence of operations to be performed on data path
- Advantages
  - Don’t need to configure path
  - Load balancing and robustness easy to achieve

Multicast

- Receivers insert triggers with same identifier
- Can dynamically switch between multicast and unicast

Service Composition: Receiver Initiated

- Receiver can also specify the operations to be performed on data
Quick Implementation Overview

- ID space is partitioned across infrastructure nodes
  - Each node responsible for a region of ID space
- Each trigger \((id, R)\) is stored at the node responsible for \(id\)
- Use Chord to route triggers and packets to nodes responsible for their IDs
  - \(O(\log N)\) hops

Optimization: Triangular Routing

- Use well-known trigger for initial rendezvous
- Exchange a pair of (private) triggers well-located
- Use private triggers to send data traffic

Example

- ID space \([0..63]\) partitioned across five i3 nodes
- Each host knows one i3 node
- \(R\) inserts trigger \((37, R)\); \(S\) sends packet \((37, data)\)

Outline

- Overview
  - Security
- Discussion

Optimization: Path Length

- Sender/receiver caches i3 node mapping a specific ID
- Subsequent packets are sent via one i3 node

Some Attacks

- Eavesdropping
- Loop
  - Victim (V)
  - Attacker
- Confluence
- Dead-End
Constrained Triggers

- $h_i()$: well-known one-way hash functions
- Use $h_i()$ to constrain trigger $(x, y)$

<table>
<thead>
<tr>
<th>ID</th>
<th>Prefix</th>
<th>Key</th>
<th>Suffix</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>128</td>
<td>64</td>
<td></td>
</tr>
</tbody>
</table>

Right constrained: $y_{key} = h_i(x)$

Left constrained: $x_{key} = h_i(y)$

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>x, y</th>
<th>end-host address</th>
</tr>
</thead>
</table>

Design Principles

1) Give hosts control on routing
   - A trigger is like an entry in a routing table!
   - Flexibility, customization
   - End-hosts can
     - Source route
     - Set-up acyclic communication graphs
     - Route packets through desired service points
     - Stop flows in infrastructure
     - ...

2) Implement data forwarding in infrastructure
   - Efficiency, scalability

Design Principles (cont’d)

<table>
<thead>
<tr>
<th>Internet &amp; Infrastructure overlays</th>
<th>Host</th>
<th>Infrastructure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Data plane</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Control plane</td>
</tr>
<tr>
<td>p2p &amp; End-host overlays</td>
<td></td>
<td>Data plane</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Control plane</td>
</tr>
<tr>
<td>i3</td>
<td></td>
<td>Control plane</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data plane</td>
</tr>
</tbody>
</table>

Example: Application Specific Routing

- Network measurements
- Query-reply routing info.
- Setup routes

Outline

- Overview
- Security
- Detailed Discussion
Conclusions

• Indirection – key technique to implement basic communication abstractions
  – Multicast, Anycast, Mobility, ...

• This research
  – Advocates for building an efficient Indirection Layer on top of IP
  – Explore the implications of changing the communication abstraction; already done in other fields
    • Direct addressable vs. associative memories
    • Point-to-point communication vs. Tuple space (in Distributed systems)