On-Demand Routing for Scalable Name-Based Forwarding

Onur Ascigil, Sergi Rene, Ioannis Psaras, and George Pavlou
Department of Electronic and Electrical Engineering,
University College London, UK
Outline

• **Background**
  • On-demand Routing
  • Routing Information Discovery
• Evaluation
• Conclusions
Background – Routing Scalability Problem

• As part of its original design, CCN/NDN overloads Interest names with the functionality of network location and content identifiers.
  • Route-by-name.
• Route-by-name involves resolving a location from a content name, i.e., name resolution, in a hop-by-hop manner.

PROBLEM:
Conventional wisdom dictates that routing and forwarding information are pre-computed and stored for \( \sim O(10^9) \) name prefixes!
Background – A well-known solution: Location-identity split!

• Map content identifiers to network location names, i.e., locators.
  • Route-by-locator as opposed to route-by-name.

• Location-identity split in NDN:
  • Interests contain **a content identifier and (optionally) a locator**.
  • Locator is used as a **fallback**, only in case of a FIB miss during route-by-name.

• NDN terminology: **forwarding hints, i.e., Link objects** in interest packets.
  • Obtained out-of-band from a resolution service.
  • a la NDNS.
Background – Location-identity split in NDN

• **Problem:** Malicious hosts can place a victim’s locator along with non-existent content names in the Interests to launch targeted attacks.

• **NDN’s solution:** Link objects carry a secure binding (i.e., signature) between content identifiers and the corresponding locators.

---

A possible show stopper for NDN’s use of Link objects:

• Verification of Link object binding is very difficult to perform in the middle of the network!
Background – Secure binding of locators and content identifiers

• Signature verification is not sufficient!
  • Need to check the legitimacy of the signing key for a given prefix-to-locator binding!
  • Must execute trust policies in the middle of the network.
    • Possibly verify a chain of certificates.

**PROBLEM:** Forwarding hints are obtained out-of-band and placed in the Interests by untrusted end-users!
• Background

• **On-demand Routing**

• Routing Information Discovery

• Evaluation

• Conclusions
On-Demand Routing

• Trust Domains (TDs) perform name resolution individually.
  • In-band solution.
  • On-demand routing mechanism.
  • Compute forwarding hints with TD-specific scope.
On-Demand Routing (cont’d)

• A way for routers to obtain and scale the storage of routing information in the form of:
  • TD-specific “instructions on how to route packets”.
  • We store these routing instructions as **Routing information Objects (RIOs)**.

• Routing information is shareable across nodes in the same TD.

• **A Routing Strategy** component at each forwarding node performs on-demand routing using an RIO.

**Main idea:** Treat Routing Information Objects (RIOs) similar to content: use caching and content discovery mechanisms to scale name-based forwarding.
Trust Domain

Routing Service

/node/c

Content Advertisement
Name Prefixes
/ubuntukernel
/foo/bar

/node/b

Content Advertisement
Name Prefixes
/ndn/arch

/node/a

Content Advertisement
Name Prefixes
/ubuntukernel
/foo/bar
Routing Information Objects:

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Locator(s)</th>
<th>Routing Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>/foo/bar</td>
<td>/node/c</td>
<td>Hot-potato</td>
</tr>
<tr>
<td></td>
<td>/node/a</td>
<td></td>
</tr>
<tr>
<td>/ndn/arch</td>
<td>/node/b</td>
<td>N/A</td>
</tr>
<tr>
<td>/ubuntu/kernel</td>
<td>/node/a</td>
<td>Load-Balance</td>
</tr>
<tr>
<td></td>
<td>/node/c</td>
<td></td>
</tr>
</tbody>
</table>

Content Advertisement

- Name Prefixes
  - /ubuntu/kernel
  - /foo/bar

Content Advertisement

- Name Prefixes
  - /ndn/arch
FIB Table contains only (intra-TD) topological information!

Routing Information Objects:

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Locator(s)</th>
<th>Routing Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>/foo/bar</td>
<td>/node/c</td>
<td>Hot-potato</td>
</tr>
<tr>
<td></td>
<td>/node/a</td>
<td></td>
</tr>
<tr>
<td>/ndn/arch</td>
<td>/node/b</td>
<td>N/A</td>
</tr>
<tr>
<td>/ubuntu/kernel</td>
<td>/node/a /node/c</td>
<td>Load-Balance</td>
</tr>
</tbody>
</table>

FIB Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Next-hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>/node/c</td>
<td>Face:2</td>
</tr>
<tr>
<td>/node/b</td>
<td>Face:1</td>
</tr>
<tr>
<td>/node/a</td>
<td>Face:1</td>
</tr>
</tbody>
</table>
**FIB Table**

<table>
<thead>
<tr>
<th>Name</th>
<th>Next-hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>/node/c</td>
<td>Face:2</td>
</tr>
<tr>
<td>/node/b</td>
<td>Face:1</td>
</tr>
<tr>
<td>/node/a</td>
<td>Face:1</td>
</tr>
</tbody>
</table>

**Routing Information Objects:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Locator(s)</th>
<th>Routing Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>/foo/bar</td>
<td>/node/c</td>
<td>Hot-potato</td>
</tr>
<tr>
<td></td>
<td>/node/a</td>
<td></td>
</tr>
<tr>
<td>/ndn/arch</td>
<td>/node/b</td>
<td>N/A</td>
</tr>
<tr>
<td>/ubuntu/kernel</td>
<td>/node/a</td>
<td>Load-Balance</td>
</tr>
<tr>
<td></td>
<td>/node/c</td>
<td></td>
</tr>
</tbody>
</table>

**Interest Packet**

Prefix: /foo/bar  Fwd Hint: /node/i
Routing Service

Routing Information Store (RIS)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Locator(s)</th>
<th>Routing Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>/foo/bar</td>
<td>/node/c</td>
<td>Hot-potato</td>
</tr>
</tbody>
</table>

FIB Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Next-hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>/node/c</td>
<td>Face:2</td>
</tr>
<tr>
<td>/node/b</td>
<td>Face:1</td>
</tr>
<tr>
<td>/node/a</td>
<td>Face:1</td>
</tr>
</tbody>
</table>

Interest Packet

Prefix: /foo/bar
Fwd Hint: /node/a

/node/c
/node/b
/node/a
FIB Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Next-hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>/node/c</td>
<td>Face:2</td>
</tr>
<tr>
<td>/node/b</td>
<td>Face:1</td>
</tr>
<tr>
<td>/node/a</td>
<td>Face:1</td>
</tr>
</tbody>
</table>

Routing Information Store (RIS)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Locator(s)</th>
<th>Routing Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>/foo/bar</td>
<td>/node/c</td>
<td>Hot-potato</td>
</tr>
<tr>
<td></td>
<td>/node/a</td>
<td></td>
</tr>
</tbody>
</table>

Interest Packet

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Fwd Hint</th>
</tr>
</thead>
<tbody>
<tr>
<td>/foo/bar</td>
<td>/node/c</td>
</tr>
</tbody>
</table>

ISP 1

| node/c |

ISP 2

| node/b |

ISP 3

| node/a |
• Background
• On-demand Routing

**Routing Information Discovery**
• Evaluation
• Conclusions
Routing Information Discovery and Caching

• **Passive Discovery:**
  • Simply observing passing-by Interests carrying Forwarding Hints.

• **Active (on-demand) Discovery – in case of a RIO miss:**
  • Search nearby or nodes along a path
  • Forward the Interest towards a neighbor with higher likelihood of RIO hit.
    • Any node on the path with the RIO can perform routing and divert the packet along the policy-compliant route..

• Discovered information is cached locally at the forwarders.
  • Different caching strategies are possible (probabilistic, LCE, etc.)
Routing Information Discovery: Search on-path

**Routing Service**

**Data**
- **Name**: /foo/bar
- **locator**: /node/a/face1

**Routing Information**:
- **Search on-path**
  - Interest /RoutingInfo/foo/bar
  - Hint: /RoutingService

**Nodes**:
- **P**: Route Points
- **Q**: Route Points
- **R**: Route Points
- **S**: Route Points

**Faces**:
- **/node/a**: face1
- **/node/b**: face2

**Interest**:
- /foo/bar
- Hint: /node/a...
Routing Information Discovery: Search nearby

Routing Information Discovery:
- Search nearby

Routing Service
- Data
  - Name: /foo/bar
  - locator: /node/a/face1

Storage overhead for “waiting Interests” + processing of RIO request/response
Routing Information Discovery: Forward to RS

Routing Service

<table>
<thead>
<tr>
<th>Interest</th>
<th>/foo/bar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hint:</td>
<td>/node/a...</td>
</tr>
</tbody>
</table>

RIS Cache of P
Name locator
/foo/bar /node/a/face1

No storage overhead or request/response traffic for RIOs.
We illustrate the processing steps in Fig. 4, where we also highlight (the new Incoming face is added to the face list), and the packet is ing strategy and is added as a forwarding hint, the forwarder makes Forwarding functionality.

Intra-domain routing policies (e.g., hot-potato routing), as discussed found in RIOs. Di a single "best" destination locator is selected among possibly many in the Fig. 4. In case, an RIO is retrieved, it is added to the RIS, and the PIT, the processing continues with checking if the packet carries Prefix in RIS?

If there is no entry in the RIS for the routable name pre, the main Interest packet goes to the RS (i.e. the node or RS) or ii) forward the interest itself towards the RS. In the former case, a search for routing information is initiated as shown in the latter case, the locator of RS is attached to the interest, and in the future. In case, a hint has not been added to the packet (by a forwarding hint (i.e., a single locator). In case there is a hint, the best Locator is selected among the possible next-hop faces. The result of the FIB lookup is then sent to the router, which uses this information to forward the packet to the nexthop. If there is no entry in the FIB table for the locator and extracts a set of possible destinations.

Routing Information Discovery

Using caching and content discovery mechanisms, we shift the rout-destination. If the forwarder is unable to explicitly indicate the routable prefix in RIS, it can divert its lookup to its FIB table for the locator and extracts a set of possible destinations. The main Interest packet arrives without containing a forwarding hint, the forwarder looks up (in its local RIS) for routing information associated with the routable prefix in RIS. Again, the main Interest packet waits while the best Locator is selected among the possible next-hop faces.

In this case, the forwarder relays the interest directly to the authoritative Route Service node to retrieve the routing information from the authoritative RS Service can respond with the routing information. Nearby nodes do not return any routing information (i.e., a negative acknowledgement (NACK)), then the forwarder sends a request to the authoritative RS. Because any node may have the routing information, we shift the routing decision in time and space as we describe in this section. We demonstrate three strategies with an example in Fig. 5, where an interest for the prefix in RIS arrives without containing a forwarding hint. The main Interest packet waits while the best Locator is selected among the possible next-hop faces.
• Background
• On-demand Routing
• Routing Information Discovery

• **Evaluation**

• Conclusions
Evaluation – Experiment Setup

- **Platform**: Icarus – a Python based simulator for ICN.
- **Topology**: ISP topology (TISCALI) from Rocketfuel dataset.
  - Prefix categories: Internal vs. External. 90% have *external* producers.
    - Randomly picked *three* locators per external name prefix.
    - One locator per internal prefixes.
Evaluation – Experiment Setup

- **Workload**: 100 flows per msec – Scalability limit of the simulator
  - 90% of flows are transit – as in a carrier ISP.
Evaluation – Experiment Setup & Metrics

• **Scalability Parameters:**
  • $|\text{RIS}| / |\text{Prefixes}| = 0.0075$ – Based on BGP router fast memory size.
  • Routing Service Replicas $= 10$ – RIOs are sharded onto RS instances.
  • Name Prefix Popularity Zipf Exponent $= 1.0$ – based on web

• **Performance Metrics:**
  • **Discovery Rate:** Percentage of Interests whose routing information is obtained from the RIS cache of a forwarder (as opposed to an RS node).
    • Measure of the load on RS and impacts latency.
  • **Latency:** This metric measures the average round-trip time (RTT) delay in retrieving content.
    • Caching of regular content is disabled.
  • **Overhead:** Average number of hops that routing information and Interests for routing information travel in the network per issued interest.
Evaluation – Benchmarks for comparison

• **FIB-as-a-cache**: Store forwarding information (as opposed to routing information) in the FIB that is used as a cache.
  • Based on the work by Detti et al. [1]
  • A centralised controller (similar to SDN controller) pushes forwarding information to nodes in case of a FIB miss.
  • **Forwarding information has local significance as opposed to RIOs.**

• **FIB-Cache with Forwarding Hints**: Store forwarding information along with forwarding hint in the FIB.
  • Hints are specific to a node.

Evaluation – Results: Impact of Prefix Popularity

Performance of Forward-to-RS is close to Search-Nearby.
Evaluation – Results: Impact of Prefix Popularity

Additional latency drops to 5msec for realistic popularity distributions.
Evaluation – Results: Impact of Prefix Popularity

Retrieving node-specific FIB information is much more costly!
Evaluation – Results: Impact of RIS Size

Less than 30% of traffic requires involvement of Routing Service for realistic RIS sizes
Conclusions

• Scale name-based forwarding through caching and information discovery mechanisms.
  • Allow per-prefix, AS-specific “routing instructions” (RIOs) to be treated as data objects.

• RIOs are maintained by resourceful servers – i.e., Routing Service.
• Routing is performed on-demand.
  • Forwarding Hints are inserted in packets and used within Trust Domains.

• Acceptable performance in comparison to pre-computed FIB approach.
  • Routing Stretch.
  • Additional Traffic to fetch routing information.

• Secure name resolution within Trust domains.