Stateless ICN Forwarding with P4 towards Netronome NFP-based Implementation

Aytac Azgin, Ravishankar Ravindran, Guo-Qiang Wang
aytac.azgin, ravi.ravindran, gq.wang@huawei.com

Huawei Research Center, Santa Clara, CA
ICN Summary

- ICN enables content-centric communication
  - Replaces current IP’s host-centric design
  - Content can be fetched from anywhere, irrespective of location

- Various ICN solutions exist
  - Content-centric Networking (CCN) and Named-data Networking (NDN) represent the leading approaches

- CCN/NDN architecture
  - Pull-based solution (i.e., request/response) with Interest/Data primitives
  - Provides additional features:
    - in-network caching using Content Store (CS),
    - stateful forwarding with Pending Interest Table (PIT)
    - integrated security within packets (through signatures)
    - Also supports multicasting, multi-homing, and mobility
Packet Forwarding in CCN/NDN

Receive Interest: 
/ContentName/ChunkID

ICN Router

ICN Network Controller

Routing Agent

Forwarding Logic

FIB

Contents exist

Drop

Incoming Faces

3 Check PIT (Perform exact name match)

Matching PIT entry

Update PIT entry

Type (I) Content Name Optional TLV Headers Interest Message

Type (D) Content Name Optional TLV Headers Signature TLV Content Message

1 Check CS (Perform exact name match)

No CS match

No PIT match

No FIB entry

Matching FIB entry

2 Respond with Data

Insert new PIT entry and Check FIB

Forward Interest

3 Check PIT (Perform exact name match)

Matching PIT entry

Update PIT entry

4 Drop Interest

5 Forward Interest

6 (Perform LPM)

Matching FIB entry

Receive Interest

Receive Data: 
/ContentName/ChunkID

Outgoing Faces

Incoming Faces

Packet Forwarding in CCN/NDN
Stateful Forwarding

- **CCN/NDN by default uses** stateful forwarding
  - In PIT, routers keep information on received requests:
    - content name, incoming/outgoing interfaces, nonces (if implemented), timeout
  - **Stateful forwarding has multiple purposes**
    - Aggregate incoming requests (e.g., same name, different incoming interface and nonce values)
    - Prevent attacks targeting a content name (as requests targeting the same name are suppressed at the edge)
    - Create breadcrumbs for the Data packets (received Data packets are checked with PIT entries for a match)
Problems with Stateful Forwarding*

- Still, we see concerns with stateful forwarding
  - Aggregation is limited to edges
  - Shown to not fully prevent attacks
  - And introduces additional overhead, in storage and processing

What remains is the breadcrumb advantage
  - which can be replicated using stateless forwarding with in-packet filters

*“pit/LESS: Stateless Forwarding in Content Centric Networks”, A. Azgin, et al.
Stateless Forwarding Design - Choices

- Use an **in-packet filter**, which carries reverse path information
  - Optional hop-by-hop header, updated at each supported hop along the path

- Different alternatives for the in-packet filter
  - **Bloom filter** $\rightarrow$ **Static field**
    - **Constant size**, bits are set until received by content source
    - On reverse path, **no modification is possible**
    - Only requires **look-up and forward** operation
  - **Counting bloom filter** $\rightarrow$ **Dynamic field**
    - Consists of 2 hop-by-hop optional headers
      - a constant size Bloom filter component, a variable-sized counter field (encoded counter to reduce overhead)
    - On reverse path, **update is possible** (removing checked-entries)
    - Requires, **look-up, update, and forward** operation
  - **Dynamic in-packet filter**
    - Non-bloom filter based filter
Which Stateless Forwarding Approach?

Observation: Bloom filter introduces significant overhead, triggered by false positives.

Regular bloom filter performance:

Number of hops vs. Overhead ratio (%)

Observation: Proposed counting bloom filter (CBF) avoids false positives significantly and reduces the overall overhead.

(a) $N = 1000$ nodes. (b) $N = 3000$ nodes.
Packet Flow in Stateless Forwarding

Incoming Interest

Check CS
(Perform exact name match)

Content exists

Respond with Data

No CS match

Check FIB
(Perform LPM and extract interface)

Matching FIB entry

Update in-packet filter

Forward Interest

Drop Interest

Incoming Data

Check CS
(Perform exact name match)

Content exists

Drop Data

No CS match

Check IMT

Matching entry

Update in-packet filter

Forward Data

No match

Update in-packet filter

Forward Data

Drop Data
Stateful ICN Forwarding with P4*

- Parse nested TLVs
  - Encoding dependent fields of packet type, content name, name components, nonce, etc.

Stateless ICN Forwarding with P4

INPUT \[\rightarrow\] PARSER

ICN-P4 Forwarding Engine

Lower layer headers \(\rightarrow\) TLV-Type \(\rightarrow\) Content name \(\rightarrow\) Component array \(\rightarrow\) In-packet filter

Recursively extracted/parsed packet headers

Match Action Tables

1. Count Table
   - Counting name components
   - Use component_array and #components to create component hashes (if Data, component hashes are skipped)

2. Hash Map Table
   - Perform longest prefix matching on component hashes to determine interface (if Data, bypassed to Filter Table)

3. FIB Table
   - Includes name hash
   - Component hash array

4. Filter Table
   - Outgoing Interface
   - In-packet filter

5. Routing Table
   - Use interface metrics, exported from FIB Table (for Interest) or Filter Table (for Data) to create egress_out

Metadata processed/created during Match-Action tables

#components (1) \(\rightarrow\) Component hash array (2) \(\rightarrow\) Outgoing Interface (3) \(\rightarrow\) In-packet filter (4)
Basic Metrics of Interest

- Storage/processing overheads
  - For stateful forwarding, PIT requirements
  - Processing overhead for each scenario

- Forwarding performance
  - Typical forwarding latency for received requests, depending on forwarding operation

- Combined analysis
  - Stateful and stateless traffic at different ratios, impact of one on the other, etc.
Integrating to Netronome NFP

- Netronome’s NFP (used on Agilio ISA) allows for more realistic implementations with better features
  - High parallelized processing capabilities, flexible storage options, and the integration of P4 and C

- As our main purpose is to demonstrate ICN capabilities with improved features, Agilio ISA offers a good design option for us

- We have other testing scenarios to get a better sense on the impact of ICN
  - Label based forwarding in ICN
    - Require a Forwarding Label Table (FLT) to use in conjunction with FIB
    - Additional variable sized packet headers to support the use of forwarding label
  - Flow-driven ICN forwarding
    - Require Flow Tables to store active flow information and to perform lookup
    - Additional packet headers to represent Flow Identifiers