ENDN: An Enhanced NDN Architecture with a P4-programmable Data Plane

Ouassim Karrakchou, Nancy Samaan, Ahmed Karmouch

School of Electrical Engineering and Computer Science

PRESENTED BY:
Ouassim Karrakchou
Agenda

➢ Motivation
➢ Proposed Architecture
  ✓ EProcessing Module
  ✓ Forwarding Logic Module
➢ Proof of Concept Experiments
➢ Conclusion and Future Work
Agenda

➢ Motivation
➢ Proposed Architecture
  ✓ EProcessing Module
  ✓ Forwarding Logic Module
➢ Proof of Concept Experiments
➢ Conclusion and Future Work
Motivation

- Current Internet:
Motivation

• Current Internet:

• Content Generation and Hosting
Motivation

• Current Internet:
  • Content Generation and Hosting
  • Congestion Control
  • Firewall
  • Bitrate Adjustment
  • Geo-Fencing

• Best-Effort Forwarding
Motivation

• Current Internet:

Intelligence in the application

• Best-Effort Forwarding

Content Generation and Hosting
- Congestion Control
- Firewall
- Bitrate Adjustment
- Geo-Fencing
Motivation

• Congestion Control
• Firewall
• Bitrate Adjustment
• Geo-Fencing

• Content Generation and Hosting
Motivation

- Congestion Control
- Firewall
- Bitrate Adjustment
- Geo-Fencing

More intelligence in the network?

Content Generation and Hosting

YouTube
Motivation

- Transition to an application-centric network:

  Applications
  
  Adapt to Current Network

  Configure

  Provider
Transition to an application-centric network:
Motivation

Transition to an application-centric network:

Advantages:
• Optimization of network resources
• Better network manageability
• Faster resolution of network issues
Motivation

• Transition to an application-centric network:

**Advantages:**
- Optimization of network resources
- Better network manageability
- Faster resolution of network issues
Existing Solutions

• **NDN**, a first step towards an *application-centric* network:
  - The network is aware of contents shared by applications.
  - Optimize the delivery of contents to requesters.

• **Network Programmability**:
  - Allow providers to completely specify their custom forwarding behaviors.
  - **SDN**: programmability in the control plane.
  - **P4**: a language to program the data plane.
Existing Challenges

• **NDN:**
  - NDN suitable for pull traffic, but challenges in other content delivery patterns (e.g., pub/sub or push).

• **P4:**
  - Current P4 data planes allow a single large P4 program to run on a switch. Hence, *downtime* is needed for updates.
  - P4 has limitations when processing *string-based* protocols as in the case of NDN.

=> Application-centric network architecture still missing
Contributions

- Design ENDN, a novel NDN architecture with the following features:
  - Extensible Catalog of Network Services
  - New Content Delivery Patterns
  - Custom Forwarding Functions
  - Zero Downtime Function Configuration

Control Plane

Data Plane
Overview of ENDN

Main Focus of the Paper

ENDN

Namespaces and selected Services

Applications

Control Plane

Manage and Configure

Programmable Data Plane

Extensible Catalog of Network Services

Examples:
- Pub/Sub
- Congestion Control
- Adaptive Forwarding
- Monitoring

ENDN

Main Focus of the Paper

NAMED DATA NETWORKING++

P4

uOttawa
Agenda

➢ Motivation
➢ Proposed Architecture
  ✓ EProcessing Module
  ✓ Forwarding Logic Module
➢ Proof of Concept Experiments
➢ Conclusion and Future Work
Proposed Data Plane Architecture

Interest -> Parser -> EPIT -> EFIB -> EPacket

Data -> EPacket

forwarding Logic:

Functions

Deparser

CS

Strategy

Internal

Face

Loopback

Faces

Face
Add support for new content delivery patterns
Proposed Data Plane Architecture

CS outside of the fast path for easier control of caching decisions using next-hops
Configure complex stateful forwarding behaviors per namespace. Examples: telemetering, geo-fencing, and stateful firewall.
1. Associate namespaces to P4 functions
2. Prepare packets for P4
Agenda

- Motivation
- Proposed Architecture
  - EProcessing Module
  - Forwarding Logic Module
- Proof of Concept Experiments
- Conclusion and Future Work
**EPIT and EFIB entries**

<table>
<thead>
<tr>
<th>Name Prefix</th>
<th>Next-Hop List</th>
<th>Type</th>
<th>Timer</th>
<th>Timer Expiry Function</th>
<th>Forwarding Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EPIT entry:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>EFIB entry:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EPIT and EFIB entries

PIT promoted as a routing table for Data packets

EPIT entry:

<table>
<thead>
<tr>
<th>Name Prefix</th>
<th>Next-Hop List</th>
<th>Type</th>
<th>Timer</th>
<th>Timer Expiry</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

EFIB entry:

<table>
<thead>
<tr>
<th>Name Prefix</th>
<th>Next-Hop List</th>
<th>EPIT Parameters</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

PIT promoted as a routing table for Data packets
Persistent or Non-Persistent

EPIT entry:

<table>
<thead>
<tr>
<th>Name Prefix</th>
<th>Next-Hop List</th>
<th>Type</th>
<th>Timer</th>
<th>Timer Expiry Function</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

EFIB entry:

<table>
<thead>
<tr>
<th>Name Prefix</th>
<th>Next-Hop List</th>
<th>EPIT Parameters</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

- **Persistent**: EPIT entry not deleted after a Data packet is forwarded
EPIT and EFIB entries

EPIT entry:

<table>
<thead>
<tr>
<th>Name Prefix</th>
<th>Next-Hop List</th>
<th>Type</th>
<th>Timer</th>
<th>Timer Expiry Function</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

EFIB entry:

<table>
<thead>
<tr>
<th>Name Prefix</th>
<th>Next-Hop List</th>
<th>EPIT Parameters</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

Garbage Collection
### EPIT and EFIB entries

**EPIT entry:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Prefix</th>
<th>Next-Hop List</th>
<th>Type</th>
<th>Timer</th>
<th>Timer Expiry Function</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

**EFIB entry:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Prefix</th>
<th>Next-Hop List</th>
<th>EPIT Parameters</th>
<th>Forwarding Function</th>
</tr>
</thead>
</table>

[Diagram showing EPIT and EFIB entries with Name, Prefix, Next-Hop List, Type, Timer, Timer Expiry Function, Forwarding Function, EPIT Parameters, and Forwarding Function]
New Content Delivery Patterns

- The modifications to the NDN PIT and FIB tables allow ENDN to support new content delivery patterns:

  - **Publish/Subscribe:**
    - The EFIB route is configured to create persistent EPIT entries when Interest packets are forwarded.

  - **Push:**
    - Persistent EPIT routes are proactively configured by the control plane along the desired push route.
EProcessing Module

- Construction of the EPacket:

Name: /com/Netflix/video/1

- Parse
- Query

EFIB/EPIT

EPacket

Parsed Packet
EPIT/EFIB entry

Construct EPacket

Extracted Route
P4 Function Name
EProcessing Module

• Construction of the EPacket:

String Processing: done outside of P4
Agenda

➢ Motivation
➢ Proposed Architecture
  ✓ EProcessing Module
  ✓ Forwarding Logic Module
➢ Proof of Concept Experiments
➢ Conclusion and Future Work
Forwarding Logic Module

EPacket

Retrieve P4 Function

P4 Function Target 1
(e.g., stateful firewall)

P4 Function Target 2
(e.g., geo-fencing)

P4 Function Target N
(e.g., adaptive forwarding)

... (Optionally) Modify NDN Packet

Output Actions

EPacket

Deparser
Forwarding Logic Module

EPacket

Retrieve P4 Function

P4 Function Target 1
(e.g., stateful firewall)

P4 Function Target 2
(e.g., geo-fencing)

P4 Function Target N
(e.g., adaptive forwarding)

Isolated P4 Code

Output Actions

EPacket

(Optionally) Modify NDN Packet

Deparser

uOttawa
Examples of output actions: forward to next-hop, drop, notify the controller, modify header field, execute metering functionalities.
P4 Function Target

Lightweight P4 code Execution Environment

- Standard Metadata
  - Input (e.g., incoming face)
  - Output (e.g., drop packet)
- EPacket
- Access and Modify
- P4 Code
  - Match-Action Pipeline
- Extern Functions (e.g., process name)
- Extern Objects
  - Registers
  - Counters
  - Meters
String processing is done through extern functions outside of P4.
P4 Function Target

- **Standard Metadata**
  - Input (e.g., incoming face)
  - Output (e.g., drop packet)
- **EPacket**
- **P4 Code**
  - Match-Action Pipeline
- **Extern Functions** (e.g., process name)
- **Extern Objects**
  - Registers
  - Counters
  - Meters

Stateful Objects
P4 Function Target

No P4 Parser or Deparser

Standard Metadata
- Input (e.g., incoming face)
- Output (e.g., drop packet)

EPacket

Access and Modify

P4 Code
- Match-Action Pipeline

Extern Functions
(e.g., process name)

Extern Objects
- Registers
- Counters
- Meters
Agenda

➢ Motivation
➢ Proposed Architecture
  ✓ EProcessing Module
  ✓ Forwarding Logic Module
➢ Proof of Concept Experiments
➢ Conclusion and Future Work
Implementation Feasibility

- **Software Feasibility**: we implemented ENDN by modifying the NDN Forwarding Daemon (NFD) to use libraries from the P4 Behavioral Model software target (BMv2). Our experiments were then simulated using ndnSIM.

- **Hardware Feasibility**: FPGA-based P4 hardware like the NetFPGA-SUME card can be used where string-processing is done in HDL (e.g., EProcessing and extern functions).
Experiment 1

- Geo-Fencing:

![Diagram showing the network topology with nodes and connections labeled for Experiment 1. The diagram includes nodes S1, SR1, SC, SR2, S2, C11, C12, C13, C14, C21, and C22, with connections and delays marked (20 ms and 30 ms).]
Experiment 1

- Geo-Fencing:

  - Regional Producer 1
  - Regional Producer 2
  - Consumers 1
  - Consumers 2
  - Central Producers
  - 20 ms
  - 30 ms
  - Region-specific content
Experiment 1

- **Geo-Fencing:**

- Tag Interests from Region 1

- Tag Interests from Region 2
Experiment 1

• Geo-Fencing:

Forward tagged Interests to Regional Producer until traffic exceeds a threshold
Experiment 1

- **Geo-Fencing:**

  ![Diagram showing the network structure with load-balancer](/assets/diagram.png)

- Load-Balancer

  - **Central Producers**
  - **Regional Producers 1**
  - **Regional Producers 2**

  Connections between producers and consumers are indicated with delays:
  - 20 ms
  - 30 ms

  Consumers 1 and Consumers 2 are connected to the regional producers through the central producer.
Experiment 1

- **Geo-Fencing:**

![Graph showing RTT (ms) vs Simulation Time (s) for different conditions including C1 NDN, C2 NDN, C1 ENDN, C2 ENDN. Key markers indicate when C12 starts, C13 and C22 start, and C14 starts.]
Experiment 1

- **Geo-Fencing:**

  NDN: Geo-Fencing managed at the central producers
Experiment 1

- Geofencing:

NDN: Some consumers can access the regional producers by querying the correct namespace.
Experiment 1

- **Geo-Fencing:**

  NDN: Some consumers can access the regional producers by querying the correct namespace.

  NDN: Geo-Fencing using available routes.
Experiment 1

- Geo-Fencing:

ENDN: Geo-Fencing managed by the network
Experiment 1

- **Geo-Fencing:**

  ENDN: Excess traffic offloaded to central producers
Experiment 1

- Geo-Fencing: Delay Reduction
Experiment 2

• Application-aware congestion avoidance:
Experiment 2

- Application-aware congestion avoidance:

Diagram:
- Congested Link
- Consumer 1
- Switch 1
- Consumer 2
- 1.5 Mbps
- Switch 2
- Producer 1
- Producer 2
Experiment 2

- Application-aware congestion avoidance:
Experiment 2

- Application-aware congestion avoidance:

![Graph showing received throughput over simulation time](image-url)

- Congestion
Experiment 2

- Application-aware congestion avoidance:

![Graph showing received throughput over simulation time with congestion detected]
Experiment 2

• **Application-aware congestion avoidance:**

![Graph showing received throughput over simulation time with various traffic types and a notation of congestion and network notification to reduce bitrate.](image-url)
Experiment 2

• Application-aware congestion avoidance:

Congestion Avoided

Network notifies producer to reduce bitrate
Agenda

➢ Motivation
➢ Proposed Architecture
  ✓ EProcessing Module
  ✓ Forwarding Logic Module
➢ Proof of Concept Experiments
➢ Conclusion and Future Work
Conclusion and Future Work

• Contributions:
  – A new Enhanced NDN (ENDN) architecture.
  – Extensible catalog of network services to applications.
  – New delivery patterns with modifications to the PIT and FIB tables.
  – Programmability in the NDN data plane using isolated P4 functions.

• Future Work:
  – Control Plane: Northbound Interface, Consistency and Scalability.
Thank you