Enabling Information-Centric Networking in IP Networks Using SDN

M. Vahlenkamp\textsuperscript{1,2}, F. Schneider\textsuperscript{1}, D. Kutscher\textsuperscript{1}, J. Seedorf\textsuperscript{1}
\textsuperscript{1} NEC Laboratories Europe, NEC Europe Ltd.
\textsuperscript{2} Hamburg University of Applied Sciences
fabian.schneider@neclab.eu
Software-Defined Networking

- Allows programming the network by separating control plane and data plane

- **OpenFlow** is most prominent implementation
  - Interface for provisioning forwarding rules of datapath elements
  - Allows centralized control plane
Information-Centric Networking

ICN Concept

- Network is aware of content
- Content naming is location independent
- Consumer requests content from the network
- Network elements perform request forwarding decisions
- Content is inherently cached throughout the network

Host A sends requests to Host B

Legacy

Host A sends requests directly to network

ICN
Problem and Motivation

Ideal ICN deployment – all network elements are ICN-aware
- Unexpected to happen soon, needs fork lift of all network equipment
- Typical ICN deployments as overlays retain several drawbacks
- SDN can be enabler of non-overlay ICN deployments
- Our approach enables eased ICN island deployment

Blefari-Mellazi et al. proposed SDN for ICN before, but require:
- New transport protocol for ICN
  - Might be blocked & lack of SDN match support
- Or use of IP options to encode content names
  - Slow path processing
- Both of these options need kernel support
  - No support for existing ICN implementations using default UDP/TCP
- Proxy ICN nodes at the edge of the network
- Knowledge of name resolution service on all ICN nodes
Goals of our approach

- Initial and partial ICN deployment over IP
- Take advantage of centralized view and control
  - Simplified ICN Routing / Content location learning
  - Efficient response path pre-provisioning
  - Support for cache pre-population and synchronization
  - Allow for request aggregation
  - Enables traffic-engineering

- No network stack changes on end nodes and ICN caches
- No need for proxy nodes at SDN network edge
- Require only the SDN Controller to be ICN aware
- Require OpenFlow 1.0 features only
Introduce ICN-SDN IP prefix
- Per ICN-SDN domain
- Announced via routing
- Used to direct traffic

ICN protocol identifier
- Dedicated port number

Separate SDN from edge addressing and forwarding
- Header rewriting

Encode path tag in regular packet header
- Depends on name of requested content
- Per-content fwd’ing

Request for NDO
1. Request Detection
2. Request Indication
3. Fwd’ing/Routing Decision
4. Install Fwd’ing Rules
5. Request Forwarding
6. Response Forwarding
7. Response Indication
8. State Removal
9. Final Response Delivery
Implementation Components

**Trema:** OpenFlow Controller Framework

**CCNx:** ICN Implementation

**mininet:** OpenFlow network emulation toolset
Evaluation Environment

- 5 Mbit file transfer (5000 chunks) between H1 to H3
  - Host-to-switch latency: 10ms
  - Switch chain latency (S1 to S5): 10ms
  - Switch-to-controller latency: 5ms
  - Link bandwidth: 10 Mbit/s
Evaluation Results

- **Bridged 1-Hop**
  - Common bridging between content requester and origin

- **Bridged 2-Hop**
  - Common bridging with additional hop
  - Pass through H4

- **CCNx-SDN Controlled**
  - ICN-aware SDN controller

Our approach outperforms the typical CCN overlay approach (Bridged 2-hop)
Lessons Learned

- CCNx is not OpenFlow friendly

- Issues with chunk sizes > 1500 Bytes
  - Option 1: Use TCP
    Hard to make forwarding decisions based on the SYN only
  - Option 2: Use UDP + IP fragmentation
    OpenFlow does not support matching on IP ID

- Issues with one port number per ICN node
  - All requests are sent from same port number
    - Good when all content comes from same source
    - Hard to implement per-requests optimizations

- OpenFlow should support arbitrary defined header graphs
Conclusions

Our approach enabling ICN communication in existing IP networks
- Enabling ICN hosts to send requests to an ICN network without any knowledge about cache locations and addresses
- Easy deployment – only controller needs to be ICN-aware
- Requiring no changes on hosts (ICN requestor and origin server/cache).

Preliminary evaluation shows promising results for our approach

Future Work
- Enhance controller knowledge through cache awareness
- Introduce cache pre-population/synchronisation functionality
- Move evaluation to real testbed
- Implement the general approach also for NetInf