KITE: Producer Mobility Support in Named Data Networking

Yu Zhang¹, Zhongda Xia¹, Spyridon Mastorakis², Lixia Zhang²

¹ Harbin Institute of Technology ² UCLA
NDN Mobility Support

- **Consumer mobility** is natively supported ✓
  - pull-based communication model + stateful forwarding plane
- **Producer mobility** is still an open issue ✗
  - existing solutions are not “simple” enough
- **Motivation**: can we further exploit native NDN features to support producer mobility in a relatively simple way?
NDN Stateful Forwarding Plane

Interest /alice/selfie.png

A

FIB

/alice  B

PIT

/alice/selfie.png  A

...
NDN Stateful Forwarding Plane

- Data is forwarded back along a hop-by-hop “breadcrumb trail”
- KITE saves the “breadcrumb trail” for Interest forwarding
Overview

Consumer

Mobile Producer (MP)

Rendezvous Server (RV)

namespace design

trace setup

trace maintenance (trace is soft-state)

trace Interest (TI)

trace Data (TD)

consumer Interest
As simple as kite flying
Namespace Design

TI/TD prefix = routing prefix + tag (“trace”) + tracing segment

/trace

Routing prefix

MP
/foo/bar

/tracing prefix

Routing prefix

trace Interest (TI)

trace Data (TD)

consumer Interest
Namespace Design: Explained

- Tracing prefix is longer than routing prefix, so trace setup is separated from the routing plane.

- Tracing prefix and TI/TD prefix share a common prefix: routing prefix, so consumer Interest and trace can meet at the RV.

- Tracing prefix can be derived from TI/TD prefix by removing tag.

- Tag is a reserved keyword ("trace") that separates trace setup from data retrieval.
Consumer Interest Forwarding

Consumer

MP

//foo/bar

Consumer

RV

path shortcut

/foo

consumer Interest
Trace Setup

1. **TI** carries verification information in name

2. Send **TD** only for **TI** with valid verification information

3. Triggered by **TD** sent back by **RV**
Trace Setup: Prevent Prefix Hijacking

• We make the following assumptions:
  (1) signature cannot be spoofed
  (2) RV is not compromised
  (3) the infrastructure is trusted

• Prefix hijacking is prevented under the assumptions above
  • attacker cannot push TD out: not possible with NDN
  • attacker cannot generate valid TI: (1)
  • attacker cannot pull TD back by sending fake TI: (2)
  • attacker cannot receive TI and collude with a partner: (3)
Trace Setup: Summary

• Forwarder forwards TI and TD as regular packets
• Forwarder searches for the “trace” tag to identify a TD
• Prefix hijacking is prevented by doing verification at the RV
• The trace setup process forms a closed feedback loop
  • receipt of TD implies two-way connectivity
  • MP can recover from packet loss during trace setup by retransmitting TI
Trace Maintenance

• Why soft-state?
  • Simple, no inter-forwarder protocol

• How?
  • MP decides the lifetime, and put in TI as “Parameter”
  • Forwarder obeys MP’s instruction and manages the lifecycle of trace accordingly
Soft-state Trace and Relocation

issues a new TI

expires on itself
Issues with Soft-state: Signaling Overhead

• Extra signaling overhead is incurred if trace expires before relocation
  • more than one TI needs to be sent per relocation
  • MP may adjust lifetime setting according to estimated time of stay
  • lifetime should be set short enough to recover from failures promptly

![Diagram showing TD, TI, and relocation over time with 30s and 15s TIs and 30s gap]
Issues with Soft-state: Stale Trace

• Stale trace emerges if previous trace is still alive after relocation
  • leads consumer Interests into a dead-end
• In certain cases, consumer can’t reach the MP until stale trace expires
  • path shortcut + stale trace + no fresh trace on the forwarding path
Mitigate the Impact of Stale Trace with Forwarding Strategy

- fresh trace
- stale trace
- consumer Interest

try alternative paths
- other next-hops
- shorter matching FIB entries
Proof of Reachability

• Upon receipt of TD, forwarding paths to the MP exists for any consumer
  • as long as the routing prefix is globally reachable

• Consumer Interests can always reach the MP as long as valid forwarding paths exist
  • assume that last-hop forwarder will send NACK back for failed recovery
  • the introduced forwarding strategy does a depth-first search on the forwarding tree consisting all alive traces (fresh and stale)
In-network Interest Retransmission

- times out and considered lost
- retransmit along new trace ASAP
- issues a new TI
Summary

Consumer

Mobile Producer (MP)

Rendezvous Server (RV)

Forwarder

Consumer

issues TI to set up and refresh trace

verify TI and send TD for valid TI

process TD: set up and maintain traces

trace Interest (TI)

trace Data (TD)

consumer Interest

regular NDN consumers
KITE and Applications
Pull

Alice

RV

Bob

trace is set up proactively

/alice/photos

/alice/trace/photos/…

/alice/photos/selfie.png

Alice’s “home”
Upload

Alice

RV

trace is set up on-demand, just like in data retrieval

/dropbox/trace/alice/…

/dropbox/alice/selfie.png

RV is also the consumer

no stale trace issue

safe to set lifetime longer

less signaling overhead
Identifies a sharing group

Enable multicast for this prefix

/share/trace/memes/…

Alice

RV

Bob

/share/memes/

/share/memes/

/share/trace/memes/…

/share/trace/memes/…
Push

Alice

/notify/alice

trace is set up proactively

RV

/notify/trace/alice/…

Bob

identifies a notification service

/notify/alice/dropbox/bob/selfie.png

/dropbox/bob/selfie.png

/notify

/notify/alice
KITE vs. Other Solutions
MP-Chasing: locate the MP

- Mapping-based: data prefix is mapped to a topology-dependent locator
- Routing-based: updates the forwarding tree built with routing
  - scalability concerns
- Tracing-based: trace the MP with the stateful forwarding plane
  - ensure reachability with a routable prefix
- KITE: a tracing-based approach
  - locator-free
  - transparent to routing and data retrieval
  - abuse-proof
Implementation and Evaluation
Implementation

• Proof-of-concept prototype
  • open source: https://github.com/KITE-2018
  • based on “real” NDN code (NDN Forwarding Daemon, NFD)
  • two application scenarios
  • directly usable for simulations with ndnSIM 2.x
Evaluation

• KITE vs mapping-based solutions
  • simulations on a toy topology
  • Pull and Upload scenario
  • different mobility patterns
• Preliminary results, not for showcasing performance
Simulation Results

• KITE is insensitive to mobility pattern

![Graph showing signaling overhead vs. mobile producer speed](image1)

**Signaling overhead - Pull**

![Graph showing packet loss vs. mobile producer speed](image2)

**Packet loss - Pull**
Simulation Results

- KITE only slightly suffers from triangle routing

![Hop count - Pull](image1)

![Hop count - Pull - distributed RV](image2)
Conclusion
Conclusion

• Transparency
  • to data retrieval process: no locator, no change to names
  • to routing: operates on non-routable prefixes

• Security
  • provides usable security with RV doing the verification

• Scalability
  • orthogonal to routing scalability
  • the RV functionality can be distributed across cooperative RV instances to scale with increasing number of mobile devices/prefixes/events

• Architectural impact
  • only extends the stateful forwarding plane
  • no change to packet format
Future Work
Future Work

• Efficient soft-state management
• Extensive evaluation to quantify performance in more, finer tuned simulation setups
• Integrate KITE into NDN code release
  • NFD release
  • API library release (ndn-cxx)
• Conduct experiments on NDN testbed
• KITE with distributed RV for better scalability and robustness
Thank you!

Q&A

xiazhongda@hit.edu.cn