NDN Tutorial

Beichuan Zhang
The University of Arizona
Outline

• How NDN works
  • Data, Interests, Security, Network

• Open research problems
  • Applications, Security, Network

• What have been been doing
  • Applications and protocols, Codebase, Community
An example scenario: smart homes

• Read temperature, get camera feed, turn on a light, etc.

• IP solution
  • Figure out where (address) to get the information
    • thermostat, camera, home controller
  • Send request to that particular address.

• NDN solution
  • Send request explicitly asking for the data without specifying destination.
Data Packet

• The most essential component of NDN architecture

| Name | Uniquely identifies the content, e.g., /UCLA/BH234/temperature/timestamp |
| MetaInfo | Info to help common data consumption e.g., FreshnessPeriod. |
| Content | Can be anything |
| Signature | Signed by the data producer |

• Data is an immutable object.
  • When content changes, name as well as signature should change too.
## Interest Packets

- Interests are sent to retrieve Data

<table>
<thead>
<tr>
<th>Name</th>
<th>What data are of interest, e.g., <code>/UCLA/BH234/temperature</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>Selectors (optional)</td>
<td>Help narrow down data selection</td>
</tr>
<tr>
<td>Nonce</td>
<td>A random number to differentiate interests that have the same name.</td>
</tr>
<tr>
<td>InterestLifeTime, etc. (optional)</td>
<td></td>
</tr>
</tbody>
</table>

```css
Name
Selectors (optional)
Nonce
InterestLifeTime, etc. (optional)
```
Basic communication

- **Consumer pulls Data**
  - one interest for one data packet.
  - Interest and data names must match
  - Rate control, data verification, loss detection and recovery, explore different network interfaces, etc.

- **Data production**
  - Naming
  - Signing
  - Segmentation if needed.
It’s all about names

• **Data and Interests carry names, no address or port.**
  • That’s what makes all the differences: benefits and challenges.

• Names are assigned to each packet by applications, e.g.,
  • `/UCLA/RoyceHall/ARFeed/FrontView/mp4/_frame=12/_chunk=20`

• **Names are hierarchical**
  • Facilitate name aggregation
  • Preserve application context for data consumption, e.g., security.

• **Naming conventions to avoid conflicts and facilitate communication.**
The role of names

• Names are used as the de-multiplexer across layers.
  • No need for each layer to have its own identifier (e.g., address, port), the management of them, and translation between them.
  • In cases of multiple interfaces and mobility, not bound to a particular address or port.

• Auto-configuration and auto-discovery
  • `/_ThisRoom/Projector/command/TurnOn/…`

• Naming is the major part of an application protocol design.
  • Once the naming convention is known, any application can use it to access the projector.
Name Discovery

• How do consumer apps learn about data names?
  • Usually know the name prefix, but not the complete name.

• In-network name discovery
  • May also use selectors to narrow the selection.

  /UCLA/BH234/temp

  /UCLA/BH234/temp/2017082109/_chunk=0

  /UCLA/BH234/temp/2017082109/_chunk=1

• Other discovery mechanisms for specific scenarios.
  • E.g., a Manifest that lists known names of relevant data.
Data-centric security

- In the Internet you secure the connection...
- ...but the server may still be hacked!
- In NDN the producer signs the data ...
- ... so that consumers know when they get bad data!
- even when the producer is offline.
Authentication of NDN Data

- Keys are named data, retrieved and secured as any other Data.

KeyLocator: /UCLA/RoyceHall/Manager/KEY

KeyLocator: /UCLA/Campus/Manager/KEY

Signed by
Name-based trust relationship and policy

- For Data to be valid, they must be signed by certain keys.
- The chain goes all the way to the trust anchor of the application.
Trust Schema: Name-Based Definition of Trust Model

- Formal description of the trust model by schematizing the relationship between the name of the data and the name of signing key.

\[
\begin{align*}
&\text{Local trust anchor(s)} \\
&\text{Key1 Rule} \\
&\text{Key2 Rule} \\
&\text{Key3 Rule} \\
&\text{Interest Rule} \\
&\text{Data Rule}
\end{align*}
\]
Trust Schema as a Bag of Bits

• Can be distributed using basic NDN mechanisms
• Secured as any other data packet

• Power of trust schema data
  • My phone can reliably validate the received video feed data
  • Camera can properly sign video feed data
  • Camera can validate commands from my phone
  • Routers can validate data and authorize requests
How Packet Forwarding Works
From a single node’s point of view

Traditional Networks

- Packets
  -Dest? → FIB
  - Accept

NDN Networks

- Interests
  - Got Data? → FIB
    - Return Data
  - Pending Interest?
    - Data
Forwarding Table (FIB)

• It stores name-prefixes and corresponding next-hops.
  • E.g., /UCLA

• Perform longest prefix match with incoming Interest’s name.
  • E.g., /UCLA/RoyceHall/…

• Multiple next-hops, which may lead to different data sources
  • since we’re forwarding towards data, not a particular destination.
  • NDN enjoys more forwarding choices since it doesn’t need to worry about loops.
Building FIB

- Applications register their data’s nameprefix with local node.
- Conventional Routing Algorithms
  - Announcing nameprefixes to the network
  - E.g., link state
- “unconventional” routing algorithms
  - Take advantage of underlying NDN networks
  - E.g., Hyperbolic Routing
- Flooding-based learning
  - Flood initial interest, observe where data comes from and add a FIB entry for it.
  - Suitable to local, ad-hoc environments
Content Store (CS)

- The data cache enabled by NDN at every node.
  - Transparent
  - In-network
  - On-path
- Many benefits
  - Reduce redundant traffic for ISPs
  - Reduce server load for producers, especially during attacks.
  - Reduce response time for consumers.
  - Even at time scale of RTT, it helps loss recovery and mobility.
- Decouples data production and data consumption
  - Naturally support DTN type of communication.
Pending Interest Table (PIT)

- New states introduced by NDN to get Data back to the consumer.
  - Each entry records (name, nonce, incoming faces)
  - Created when a new Interest is forwarded
  - Updated when more interests carrying the same name arrive
  - Deleted when a matching Data returns.
Benefits of PIT

- **Native multicast**
  - No difference between multicast and unicast operations

- **Suppress duplicate packets (e.g., caused by forwarding loops)**
  - Important to mobile, ad hoc communication

- **Provide closed-loop feedback for the success/performance of data retrieval, at every hop.**
Forwarding Strategy

- FIB provides multiple forwarding options.
- PIT enables fault detection and performance feedback.
- Based on the above, Forwarding Strategies make the best forwarding decisions for data retrieval.
  - E.g., a strategy may minimize path changes, or look for the shortest delay, or look for higher throughput, or always multicast/broadcast, etc.
Summary of how NDN works

- Data are identified by names, and signed by producer.

- Interests carry names to retrieve matching Data.

- Data-centric security and name-based trust schema.

- Stateful data-plane that supports in-network caching, native multicast, and versatile forwarding strategies.
Open Research Problems
NDN Research

• NDN is a new network architecture
  • from traditional *point-to-point conversation* to *distributed data production, retrieval, and consumption*.

• It requires rethinking of applications and networks together with built-in security.
  • How does NDN work in a particular network environment?
  • How to fully take advantage of NDN’s features?
  • How to optimize NDN’s performance?

• Selected research problems in applications, security, and networks.
Applications

• NDN brings network semantics closer to application semantics.

• Data-centric app designs take the most advantage of NDN.

• We drive network architecture research by building applications. (And, vice versa.)

• Notably, enables *cloud-assisted* alternatives to the current *cloud-centric* paradigms for distributed computing, which fail to support highly dynamic, low-latency, mobile apps.
Namespace Design

• Data namespace design becomes a key part of app design.
  • Tussle between data access, forwarding, security, and other requirements placed on names.

• Home IoT Example
  • Name data, actuation points, devices, keys / certificates, access control policies.
  • Often, four-part name: /A/B/C/D
    • A: How to reach the data. (e.g., localhop, home-<guid>, /edu/ucla, etc.)
    • B: High-level identifiers. (e.g., living_room/temperature)
    • C: Derived or related data identifiers. (e.g., KEY, _mimetype)
    • D: Type-specific suffixes (e.g., segment or sequence number, version, etc.)
  • Keylocator in the data packet: another name related to trust.
Application APIs

• APIs and app conceptualization changes:
  • Consumer-driven: pulling rather than pushing data
  • Asynchronous multi-node data dissemination rather than client/server
  • Local and global communication can use same mechanisms

• *Home IoT Example*
  • Retrieval and actuation possible with basic primitives.
  • Discovery & bootstrapping can also be implemented with basic primitives + name conventions (afternoon demonstration).
  • Hierarchical names common in other layers.
  • Less obvious are how to achieve rare but critical notifications.
  • Important to remember NDN at Layer 3.
Usable Security

• App data security can be built in from the ground-up, but the approaches and tools are new and need work to be easy-to-use.

• Home IoT Example
  • Schematized trust: we’ve found easy to use, just need conventions and more examples.
  • Name-based access control: many options, harder to conceptualize just in terms of names.
  • Basic demonstrations this afternoon.
Research problems & approaches

• Open questions of what network provides to app
  • Network storage (e.g., repo)
  • Indirection (e.g., NDNS)
  • Handling mobile publishers

• *IoT Example*
  • ”Memory content cache” easily extended to persistent tuple store.
  • Certificate storage, name redirection, could come in infrastructure
  • From home networking to vehicular networking.
    • Multihomed, mobile.
    • Local, neighborhood, global data.
Data Access Control

• Separate data retrieval from data access control
  • Encrypt the data, which can be retrieved by anyone.
  • Control the access to decryption keys.

• The exact mechanism design has many options, also depending on the network environments, e.g., resource constrained devices.
DDoS and Content Poisoning

• NDN architecture is more resilient to DDoS attacks than IP.
  • Cannot flood victim with Data.
  • Not effective if flood victim with Interests for existent Data.
  • Only way to attack is to flood with Interests for routable but non-existent Data, but this can be detected from PIT behavior.
  • Design the mechanism to detect and mitigate this DDoS with minimal collateral damage.

• Content Poisoning
  • Routers by default don’t verify data for performance reason.
  • What if consumers received forged data and want the routers to retrieve a different one.
Infrastructure-less environments

- Network environments that have no reliable fixed infrastructure
  - Mobile, ad hoc, wireless device-to-device, delay-tolerant network, disaster recovery, etc.

- NDN thrives in these environments
  - Fetching data vs. chasing a mobile node, or establishing a connection between two nodes not online at the same time, or going through cloud for local communication, etc.

- Research issues
  - Auto-configuration, auto-discovery, device-to-device communication
  - Security models and mechanisms
  - Routing and forwarding strategies under mobility, and use multiple interfaces at the same time.
Forwarding Strategy

- A powerful mechanism that makes data-plane smart
- Can employ different strategies for different types of data and in different networks.

- For examples:
  - Strategies for large scientific data movement, VR/AR data, IoT data.
  - Strategies for vehicular networks, smart homes, sensor networks, delay-tolerant networks, data center networks, etc.
  - Supporting flexible strategy composition
Sync

- Multi-party synchronization of a shared dataset.
  - Each party may start with a different subset
  - The dataset may change over time.
  - The abstraction for NDN transport.
  - TCP-like reliable transfer is a special case.
    - Two parties, sender has the full set, and receiver has none.

- Basic approach
  - Efficient representation and exchange of the subsets, utilize multicast/broadcast to share data and remove redundancy.

- A number of solutions have been proposed.
  - They different in data naming, state representation, change notification, and update retrieval.
Congestion Control

• It is a different story for NDN
  • No longer a point-to-point session with a single base RTT.
  • Now multipath, multi-source data transfer.
  • Need to regulate Interest rate in order to control congestion

• Need hop-by-hop solutions where
  • Every node on the path participates
  • Be able to use multiple paths and multiple sources
  • Impacts on overall network behavior
Routing, Forwarding, and Caching

• In NDN, these are all related.
  • For example, forwarding decision affects cache availability, which in turns affects future forwarding decisions.

• Joint optimization of them in different network environments
  • Data centers, ISP networks, mobile edge networks, etc.

• Explore new routing protocols
  • With smart data plane, the requirements on control plane have been relaxed.

• Routing Scalability
  • Table size
  • Routing churns
Scalable forwarding engine

• Table lookup and update (FIB, PIT, CS)
  • Names are of variable-length
  • Table size can be large, content can be dynamic.
  • Matching rules are also different for each table.

• A large body of work has explored different data structures
  • Hash tables, tries, bloom filter.
  • Mostly focused on FIB.
  • Need better designs for CS and PIT.
What we have been doing
Application-driven architecture development

• Current focus areas:
  • Mobile edge computing
  • Internet of Things
  • Navigable media

• Other applications
  • Open mHealth (mobile health)
  • Building automation and management
  • Scientific “big data” (e.g., climate change)
  • Real-time conferencing
  • Neighborhood solar
  • File sharing, chat, etc.
Protocol and Mechanism Design

• Articulated the design principles, developed the packet format and protocol specs.
• Routing protocols
• Forwarding Strategies
• Table structures and algorithms
• Name-based authentication, trust, and access control
• Sync protocols
• Congestion control
• ….
Running Code and Evaluation Platforms

- Network forwarder, libraries, tools.
- On conventional platforms and IoT devices
- Simulator, emulator, and global testbed
- All code is open sourced.
Research Community

- NDNComm
  - 2017 at Memphis, 73 people from 36 institutions
  - 2015 at UCLA, 116 people from 49 institutions
  - 2014 at UCLA, 87 people from 31 institutions

- ACM ICN conference and ICN-related workshops
- IRTF ICN RG
- Both academia and industry.
For more information

http://www.named-data.net

Don’t miss the demos and codebase overview in the afternoon!