NDN-NP Applications Update

Jeff Burke
NDNcomm 2015

Collaborative work by many.
Reformulation

Storytelling
  => What if streaming media could be composed just-in-time?
  => What if expeditionary networks were the norm?

Health and Wellness
  => What if the user was the root of trust for their own data?

Building management / IoT
  => What if there was no assumption of perimeter security?

Real-time conferencing
  => What if it’s about what the network can deliver, not reaching a producer?
Outline

Four applications covered in this talk:

- **NDNFit** (user-centric health & wellness)

- **NDN UX** (user experience of identity / data sharing in NDNFit)

- **EBAMS** (and IoT)

- **NDN-RTC** (low-latency media / conferencing)

Also, the team has been working on scientific data apps (covered in the next presentation) and higher-level communication protocols (sync, infomax).

More about all of these in other presentations, demos, breakouts.
NDN-NP Progress So Far

*Where we started* –

- **Incorporate security**: Name-based trust definition, verification, confidentiality.
- **Practical deployment** needs that yield research challenges: publisher mobility, autoconfig, trust bootstrapping, etc.
- **Higher-level communication concepts** – e.g., sync, manifests.

*What also emerged* –

- **App / strategy relationships**.
- **Traffic measurement requirements**.
- **Performance requirements**. (Finally!)
NDNFit

Haitao Zhang, Alexander Afanasyev, Jianxun Cao, Euihyun Jung, Jiewen Tan, Jeff Thompson, Yingdi Yu, Jeff Burke, Dan Pei, Christian Tschudin, Lixia Zhang, and others.
NDNFit: Open mHealth example application

An ecosystem conceived with data exchange as the thin waist (Sim & Estrin, 2010), which is natural for NDN.

Gartner, 2014
NDNFit: Open mHealth example application

How do we conceive and build a familiar-looking application that demonstrates interaction in this ecosystem?

NDN enables a user-centric reformulation of health and wellness data management.

One simple step: the user can be the root of trust.
MOBILE TRACE CAPTURE
Ohmage on Android

PERSONAL DATA REPOSITORY
DSU

LOCATION ANONYMIZATION
DPU

ACTIVITY CLASSIFICATION
DPU

FITNESS VISUALIZER
(NO LOC. DATA)
DVU

PATH VISUALIZER
(LOC. DATA)
DVU

LOCATION-BASED CONTENT EMITTER
DVU

GEOFENCING FILTER
DPU
### NDNFit Application Architecture

For Summer 2015 Implementation

Version 2 - June 24, 2015

**NDNFit User Site (DVU)**

- **Interface to:**
  1. Sign up for NDNFit service.
  2. Download mobile application
  3. Set up omh, DSU namespace to use.
  4. Select features implemented in DPUs.

**HealthPDV (DSU)**

- **Interface to:**
  1. Sign up for DSU service.
  2. Authorize applications, like NDNFit, to access data.

**NDNF Android App**

- **Configure (Data Mgr)**
- **Capture**
- **Name, Sign, Encrypt**
- **Upload**

**ID Manager Mobile App**

- **Select/marshal keys for apps**

**NFD**

**NDN Auto-conf**

**Auto-configure support on current infrastructure, operated as part of testbed**

**NDNS**

- **Operated as part of testbed**

**Requests globally routable LINK name**

**Authorization to publish in the /org/openmhealth/<user_id> namespace**

**User-facing web sites**

**Open mHealth Namespace Assignment**

- **Interface to:**
  1. Sign up for Open mHealth namespace.

**NDNFit User Site**

**NDN in next iteration**

**Colors are used to distinguish between different administrative/ legal domains, to some degree.**

**Interface to:**

1. Sign up for DSU service.
2. Authorize applications, like NDNFit, to access data.

**Interface to:**

1. Sign up for Open mHealth namespace.
2. Download mobile application
3. Set up omh, DSU namespace to use.
4. Select features implemented in DPUs.

**Interface to:**

1. Sign up for Open mHealth namespace.
2. Download mobile application
3. Set up omh, DSU namespace to use.
4. Select features implemented in DPUs.

**RPC via NDN?**

**Sync?**

**PDV Server hosted at UCLA**

**Distribute data encryption keys**

**NDNS**

- **Operated as part of testbed**

**Requests globally routable LINK name**

**HealthPDV (DSU)**

- **Topology-independent data ns:** /org/openmhealth/<user_id>
- **Routable ns for access on global Internet:** /com/healthpdv

**Storage**

**Distribute data encryption keys**

**NDN**

**Auto-configure support on current infrastructure, operated as part of testbed**

**NDN in next iteration**

**HTTPS for now**

**User-facing web sites**

**NDN in next iteration**

**NDNFit User Site (DVU)**

- **Interface to:**
  1. Sign up for NDNFit service.
  2. Download mobile application
  3. Set up omh, DSU namespace to use.
  4. Select features implemented in DPUs.

**Classification (DPU)**

**RPC via NDN?**

**Sync?**

**Virtual hosts on UCLA box**

**Virtual hosts on Basel or UCLA box**

**NDN**

- **NAME, SIGN, ENCRYPT**
- **UPLOAD**

**Any machine**

**Web Browser accessed by user**

**HTTPS for now**

**NDN in next iteration**
Data-Centric Security

- Good fit for this application.

- Schematized trust
  - Initial design developed.
  - jNDN support for Android.

- Name-based access control
  - Names capture dimensions we want to manage.

- New pieces / future areas
  - Access control for NFN processing blocks.
  - Name confidentiality.
Challenges

• Life-long data. Enabling the user to “move” data storage from provider to provider and maintain the same namespace.

• Usable security. Schematized trust and name-based security are conceptually simple, but work to be done on how to provide this power to developers in a simple way.

• Best type of sync protocol(s) for mobile upload, storage – processing.

• Access control between processing blocks.

• Publisher mobility.

• See Haitao Zhang’s poster for more information.
User Experience for NDNFit
Dustin O’Hara, Jeff Burke
LASHP Fitness Website

Informed by REMAP's ongoing work with the Los Angeles State Historic Park (LASHP), the NDN Fitness applications will be branded to reflect the identity of the park and landscape of downtown Los Angeles.

Once the user is at the LASHP fitness website they are directed to download the fitness recorder and fitness visualizer apps through a trusted “app store” mechanism that has yet to be determined.
But this application is a new kind

• User owns the data

• Ecosystem of interoperable applications

• Named data!
Bridging the NDN architecture with users

NDN Namespaces
• How are users introduced to the application namespaces?
• How much exposure to the namespaces is necessary?

Identity management & Data signing
• How do we get users involved in signing their data?

Managing access control of personal data
• How do we make data-centric security usable for personal data?

Data-centric Interoperability
• What design choices can be made that move us towards a data-centric ecosystem rather than silo’ed applications?
ID Manager

Open mHealth Namespace

The NDNcomm App has requested that you create an ID that has an open mHealth Namespace.

MORE INFO DECLINE ACCEPT
ID Manager

RandyMan
Open mHealth Namespace

Waiting to receive a confirmation
randy@remap.ucd.edu

ID Manager
9:08
12:33 PM/View details

Someone has initiated the registration of an ID within the Open mHealth namespace. If you are the person creating this ID, then click the link below to confirm your email for the Open mHealth ID.

http://sert.openmhealth.org/sert-request/submit?token=5A2LBTz2M
Fitness Visualizer Dashboard
But this application is a new kind

• User owns the data

• Ecosystem of interoperable applications

• Named data!
The Data Sharing app will manage randomly generated data.
Confirm Data Types
Selecting Days of the Week
Challenges

NDN Namespaces
• How are users introduced to the application namespaces?
• How much exposure to the namespaces is necessary?

Identity management & Data signing
• How do we get users involved in signing their data?

Managing access control of personal data
• How do we make data-centric security usable for personal data?

Data-centric Interoperability
• What design choices can be made that move us towards a data-centric ecosystem rather than silo’ed applications?
• See Dustin O’Hara’s poster for more discussion.
Enterprise Building Automation & Management
Zhehao Wang, Wentao Shang, Jiayi Meng, Adeola Bannis, Jeff Thompson, and others.
From Enterprise to Space to Thing

Objective
Explore the Internet of Things, in the context of building automation and management, from the top-down and the bottom up.

Platforms
• Enterprise (Siemens BMS)
• Smart Space (Raspberry Pi)
• Thing (Arduino)

Moving to **system of systems** work – composing end-user experiences across these platforms.
Dealing with typical data warehousing challenges became an exploration of how to decompose SQL queries, or other standard query types, onto NDN-stored data. (Initial work by Wentao Shang.)

Looking initially at NDN-native hierarchical storage: keeping the data near the devices, then aggregating upwards.
EBAMS Test Environment: Mini-EBAMS

Hierarchical Data Storage on NDN

- Real UCLA Data
- Mini-NDN system
- NDN-JS access to it

See Zhehao Wang’s poster for more.
Hierarchical storage

- Raw data collected, kept, and batched only at the leaf nodes.
- Leaf nodes publish aggregated (min, sum, avg, etc) data at fixed time window.
- Non-leaf nodes fetch the aggregated data from all of its children, and aggregate the data after all children respond.
- Non-leaf nodes can publish aggregates with the same time window $T$; or $n \times T$, $n = 1, 2, 3...$
Mini-EBAMS

Testbed nodes
- Testbed
  - Aleph
  - Browser consumer node

Visualization nodes
- UCLA
  - Dentistry
    - A3-063
      - sensor1
      - sensor2
    - 83-055
      - sensor1
      - sensor2
  - Franz_Hall
    - C417
      - sensor1
      - sensor2
    - A173
      - sensor1
      - sensor2

Mini-ndn nodes
- UCLA
- BMS gateway nodes
  - BMS gateway node
  - ...

BMS gateway nodes
- UCLA BMS Browser Consumer Concept (Steam flow)
- Steam flow
- Time
- UCLA - Factor - a:937 a - stm.flw (Steam flow)
- Steam flow
- Time
- UCLA - Factor - stm.flw (Steam flow)
- Steam flow
- Time
- UCLA
- Steam flow
- Time
- BMS gateway nodes
- UCLA BMS Browser Consumer Concept (Steam flow)
- Steam flow
- Time
- UCLA - Factor - a:937 a - stm.flw (Steam flow)
- Steam flow
- Time
- UCLA - Factor - stm.flw (Steam flow)
- Steam flow
- Time
- UCLA
Current namespace

- **ucla**: organization
- **bms**: system
- **sensors**: category
- **young_hall**: building
- **b215**: room
- **xfmr-6**: device
- **data/electrical**: type (data, electrical)
- **inst**: data subtype (instantaneous)
- **1423123667**: data timestamp

- **ucla**: organization
- **bms**: system
- **sensors**: category
- **young_hall**: building
- **data/electrical**: type (data, electrical)
- **aggregation**: subtype (aggregation)
- **1423123670**: Start timestamp
- **1423123680**: End timestamp
Data-Centric Security

- Schematized trust
  - Initial design developed for hierarchical aggregation.

- Name-based access control
  - Primary access the
  - Provisions for links across namespaces (e.g., into user namespaces), translating previous work by Shang et al. to NBAC.

- Authenticated control
Smart Space (RPI)

- Work at UCLA, University of Arizona.

- Last year, completed a “kit” for IoT experimentation on NDN.
  [https://github.com/remap/ndn-pi](https://github.com/remap/ndn-pi)
Smart Space (RPI)

- COAP comparison - after one-time prefix registration overhead, comparable or improved communication burden.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Lighting device</th>
<th>Controller</th>
<th>Client device</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sent</td>
<td>Received</td>
<td>Sent</td>
</tr>
<tr>
<td>CoAP-DTLS</td>
<td>279</td>
<td>350</td>
<td>347</td>
</tr>
<tr>
<td>NDN-HMAC</td>
<td>140</td>
<td>99</td>
<td>265</td>
</tr>
</tbody>
</table>

Smart Space (RPI)

• UA-led work continues in the smart home context:
  • Device bootstrapping: the initial exchange of keys between the device and the controller.
  • Device discovery and configuration: how does a new device learn about existing devices in the home – further work on capabilities.
  • Access control: which device can access which data/device, and do so efficiently.
Other types of infrastructure-based sensing.

**OpenPTrack: Positional tracking for education, arts, and culture.** Based on the Robot Operating System (ROS)

Data analogous to Active RF for NFL. Low-latency NDN output, 30Hz tracking.

Leverage what we are learning from NDN-RTC.

Future project: internal ROS messaging via NDN.
Thing (Arduino)

- Demonstrated NDN-CPP library running on Arduino, extended to Bluetooth Low Energy communication at the hackathon.
- The NDN-CPP client library uses C++ features not on Arduino:
  - Standard library resizable vector
  - Standard library reference counting shared_ptr
  - Standard library function objects with bind
  - Exception handling
- But Arduino does have these C++ features:
  - Constructors and destructors
  - Method overloading
  - Namespaces
Thing (Arduino)

NDN-CPP Lite
• Lightweight C++
• No assumptions about memory model
• No support library dependencies
• Application creates and supplies memory
• E.g. initialize a MetaInfo object:
  • MetaInfoLite metaInfo;
• Shared C core between NDN-CPP and NDN-CPP Lite:
  • Packet encoding/decoding
  • Network transport (TCP, etc.)
  • Only standard C library (strlen, math.h, etc.)

Sample Application
• Register a prefix, receive an interest over TCP, return an HMAC signed data packet holding an analog measurement
• 28 kilobytes when compiled
• Modifications for Arduino:
  • YunClient bridge for TCP
  • Arduino-optimized SHA256/HMAC code
  • Use Arduino native random number generator
Challenges

• NDN – in particular, schematized trust – seems to fit the problem very well. Management of access control is challenging for outside the enterprise. Lots of interesting work to do on NDN-backed databases / query support.

• System of systems: Provisioning Enterprise, Space, and Thing systems for overlapping applications that include interaction with the Internet and personal mobile devices.
  • Namespace conventions for discovery, negotiation, data access
  • Cross-system trust and granular access control
  • Power consumption and other resources constraints
  • Core services: app-level time synchronization, etc.
NDN-RTC
Peter Gusev, Jiachen Wang, Jeff Burke, Lixia Zhang, and others.
NDN-RTC Project Goals

• Functional videoconferencing library and application:
  • Low-latency, interactive data distribution:
    • Multi-party conferences
    • Live broadcasting
  • No direct communication between peers:
    • Consumer-driven
    • More freedom for experimentation
• Wide adoption by NDN community
  • Data-centric security: schematized trust, name-based access control
• Encourage new research initiatives
• Testbed traffic generation and high-load performance testing
NDN-RTC Project 1 Year Ago

- OS X C++ library
- Console demo app
- No group chats
- Hard to use (user-unfriendly)
- Hard to setup
  - No user auto-discovery
  - Build from sources
- Mediocre streaming efficiency:
  - frequent rebufferings
  - video tearings
  - frequent audio drop-outs
NDN-RTC Project Today

- Achieves target 350-500ms latency for our conferences.
- HD-quality capable
- Improved streaming performance
  - Up to 7.5Mbit/sec over current testbed; significantly better QoE.
  - Isolated testbed tests (bidirectional streaming)
  - NDN testbed tests with multi-hop paths (bidirectional streaming)
- **ndncon** GUI OS X NDN-application:
  - automatic user discovery
  - group text chats
  - screen sharing
- Easy setup:
  ```bash
  $ nfdc register / udp://<your_hub>
  ```
Example NDN-RTC-driven Improvements

- NFD: Revised retransmissions strategy
  - App retransmission was suppressed until Interest times out in PIT
  - Varying Interest lifetime is risky when data is not produced yet or network conditions change
  - BestRoute2 strategy allows early app retransmission without giving up Interest lifetimes
- NDN-CCL: Library support for app-level PIT
  - Common low-latency case: handle Interests that arrive before data is ready
  - Need to store Interests in producer-side PIT
  - Same approached used in OpenPTrack real-time person-tracking
- Testbed/NFD: Performance stress-tests (ongoing)
  - 3-9Mbit/sec data streams per producer
  - 9Mbit/sec: ~1000 Interest/sec, ~900 data segments/sec
  - Traffic generator for the testbed
Design & Development Progress

- Design
  - “Interest Demand” concept introduction
  - Audio packet bundling
- Implementation
  - Desktop GUI application *ndncon*
    - group chats (ChronoChat2013)
    - automatic user discovery (ChronoSync2013)
    - screen sharing
  - Thread optimization
    - single-threaded architecture, decreased CPU
  - Asynchronous logging
  - Automated test environment (local testbed, NDN testbed)
- Ported to Ubuntu
  - special thanks to Luca Muscariello (Orange), Zhehao Wang (UCLA)
Interest Demand

- **Outstanding Interests** ensure latest data delivery
- The **minimal number** of outstanding Interests that ensure latest data retrieval defines "**Interest Demand**"
- **Interest Demand** driven by:
  - DRD (Data Retrieval Delay) – generalized RTT
  - Data inter-arrival delay (producer publishing delay observed by consumer)
    \[ \text{Interest Demand} = \frac{\text{DRD}}{\text{D}_{\text{arr}}} \]
- Consumer changes **Interest Demand** value in order to adjust fetching **aggressiveness**
- Data-driven Interest expression:
  - Quicker response to new network and publishing conditions
  - Faster and more robust cache exhaustion
Adaptive Rate Control

- Collaboration with Panasonic R&D department
- Established development plan:
  - NDN-RTC modifications, REMAP - October 2015
  - ARC implementation\(^1\), Panasonic - November 2015
  - Early tests - December 2015
  - Full tests – January-February 2016
  - Completion – March 2016
- Implementation details\(^2\)
  - Gapless stream switching
  - Challenging Interests for bandwidth probing
  - Ongoing monitoring of intrinsic network parameters (DRD, D_{arr}, etc.)

Future Work

- Adaptive Rate Control (*in progress*)
- Linux compatible version (*in progress*)
- Ubuntu headless app (*in progress*)
- Further tests
  - multi-party uni- and bi-directional tests (*ongoing*)
  - NFD performance stress tests (*ongoing*)
  - large-scale tests using headless Ubuntu app
- Data authentication and encryption with multi-party support
- Scalable video coding
Challenges

• **How to robustly detect arrival of latest data?**
  • Desire no **direct** producer-consumer communication
  • Current approach:
    • observe intrinsic network indicators
    • cached (stale) data arrival copies Interest expression pattern

• **How to efficiently encrypt media without losing NDN advantages?**
  • Depends on application objectives – Reformulate conferencing?
  • Leverage broadcast encryption and other schemes

• **How to achieve inter-consumer synchronization?**
  • While preserving no direct communication
  • Consider varying network conditions

• **Where is historical data stored?**
  • Depends on application objectives – drill down into use cases.
  • Audio/Video, chats, attachments, etc.
  • Historical data trust model
Opportunities for Collaboration

- NDN project team plans to use and improve ndncon. Help welcome!
- Others can use NDN-RTC library for creating more applications.
- Fundamental improvements in latency performance.
- Incorporation of scalable video coding.
- Deeper research into rate control, interest expression algorithms needed.
- We will be providing tools to use the library for traffic testing and provide feedback on strategy / forwarding.
- Test and port to other platforms
- Need to do simulations to look at algorithm performance under various caching conditions, topologies, and use cases.
- Please come visit our posters and demos!