

# MILCOM 2017

MILITARY COMMUNICATIONS AND INNOVATION - PRIORITIES FOR THE MODERN WARFIGHT

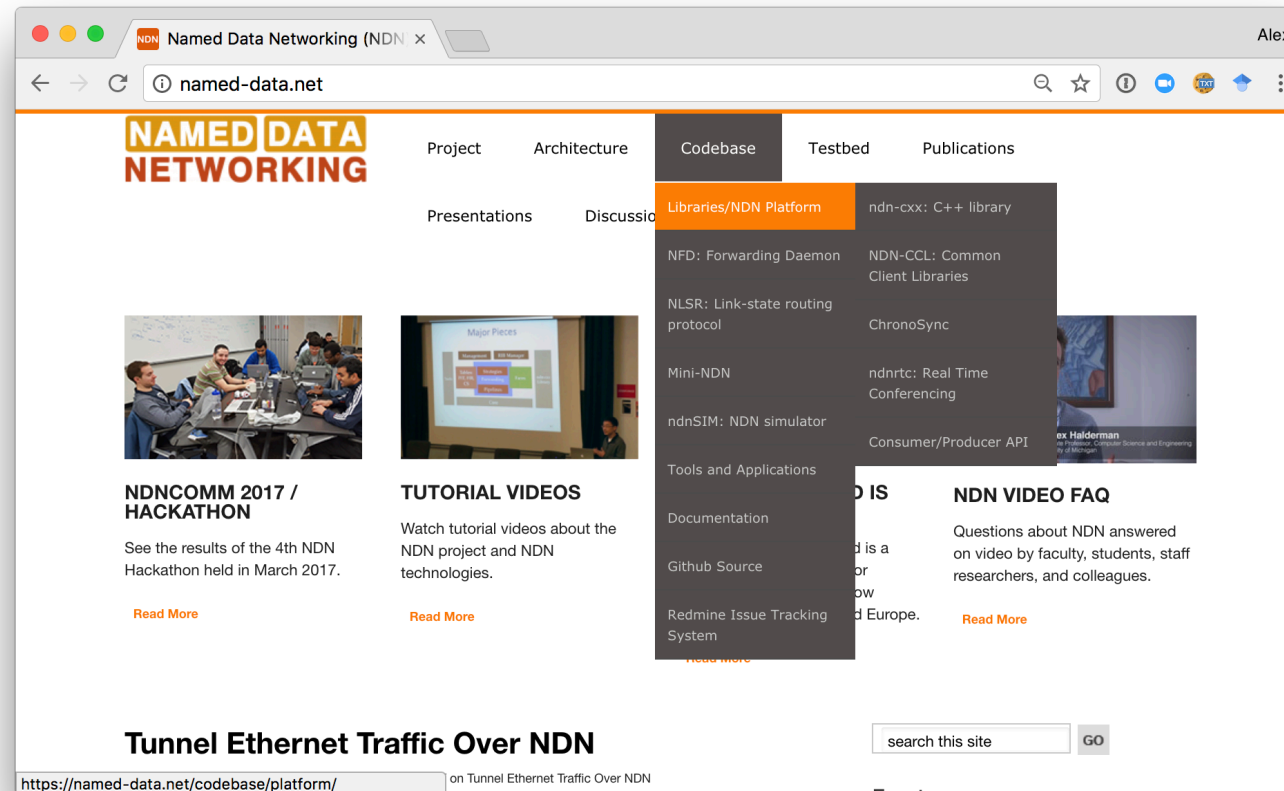
## NDN Codebase and Tools

Alex Afanasyev

Florida International University

BALTIMORE, MD • OCTOBER 23–25, 2017

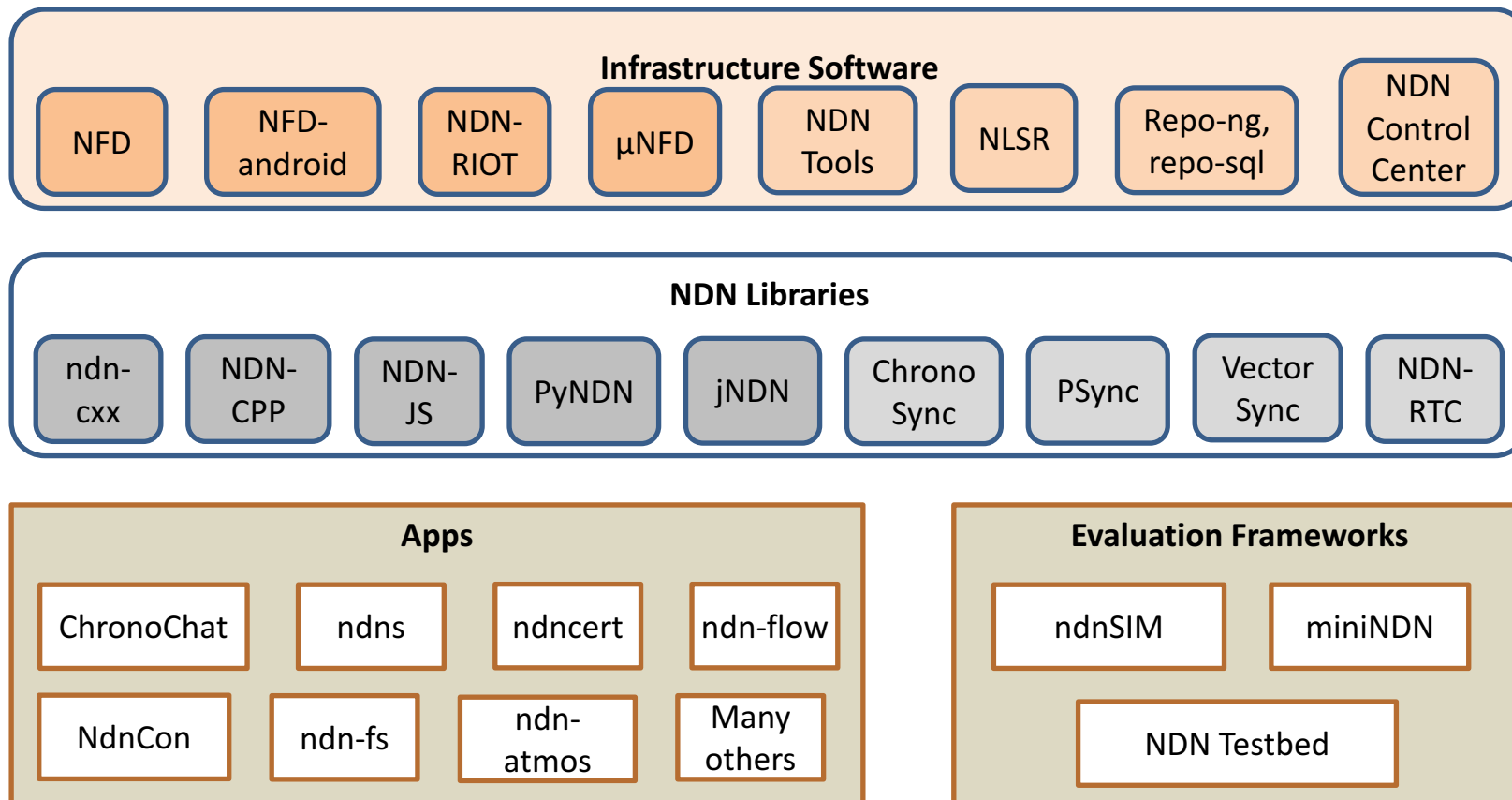
## Starting Point: <https://named-data.net/> ➡ Codebase



# Where to Find Source Code for NDN Codebase

- Most linked from <https://named-data.net> → Codebase
- Github organizations
  - <https://github.com/named-data>
    - NFD, core libraries, and other general use software
  - <https://github.com/named-data-mobile>
    - Android and related software
  - <https://github.com/named-data-iot>
    - IoT related software
  - <https://github.com/named-data-ndnsim>
    - ndnSIM core, example and real simulation scenarios

## NDN Codebase Overview

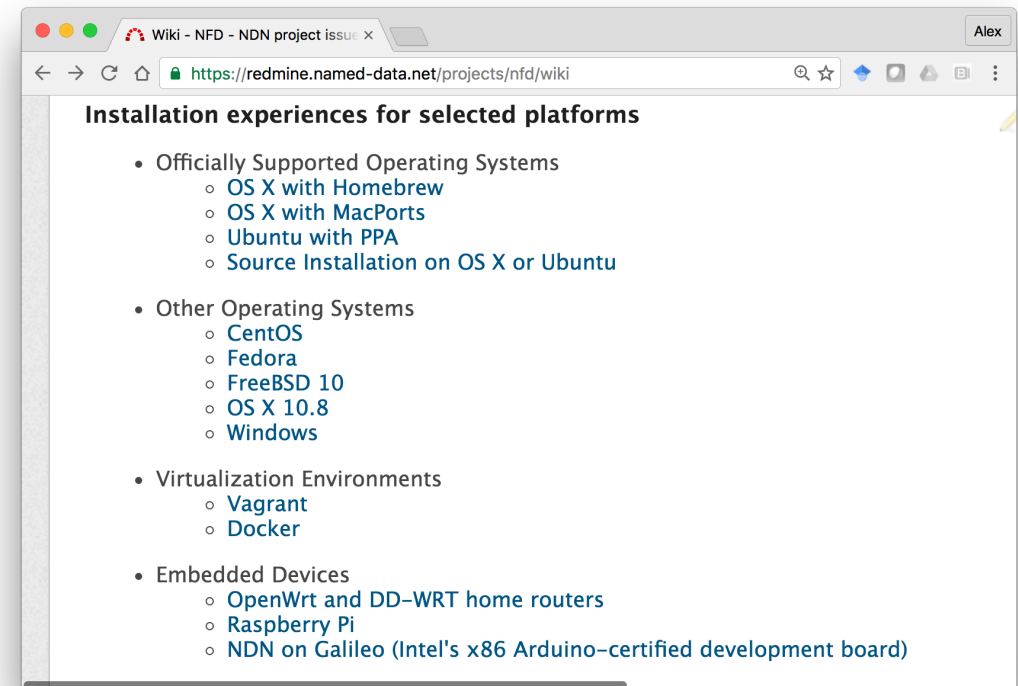




# Supported Platforms

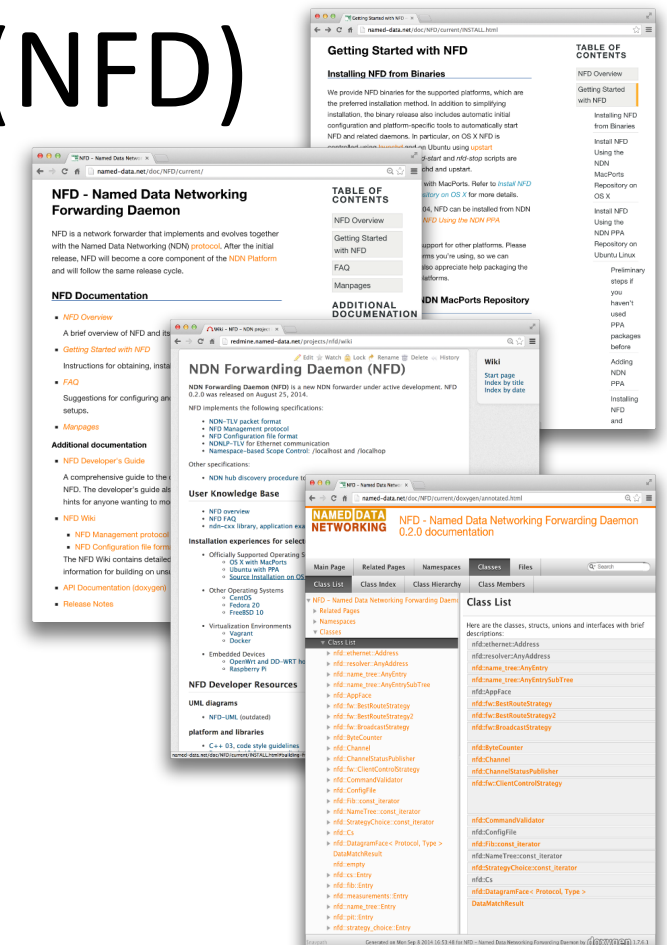
<https://redmine.named-data.net/projects/nfd/wiki>

- Desktop Systems
  - Ubuntu, OSX, FreeBSD and other Linux distributions
- Home routers
  - OpenWRT, DD-WRT
- Mobile:
  - Android, iOS (library only)
- IoT:
  - Arduino, ESP8266, RIOT-OS
  - Raspberry Pi (runs NFD, available binary packages)
- Web browser
  - NDN-JS library + microforwarder

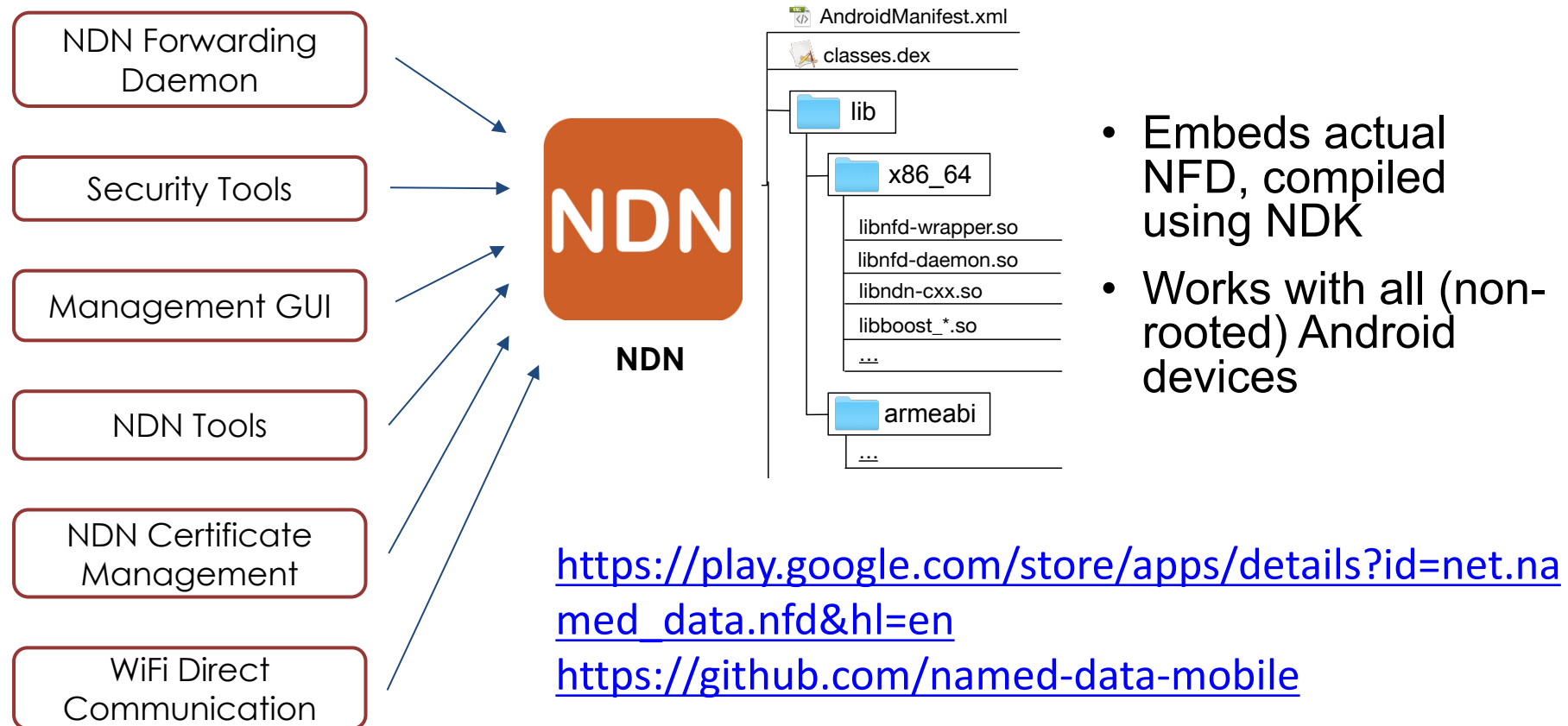


# NDN Forwarding Daemon (NFD)

- The reference implementation of NDN forwarder
- <https://named-data.net/doc/NFD/current/>
  - Overview
  - Getting started
  - NFD Developer's Guide
  - Manpages
  - Wiki
  - API documentation (doxygen)
- Feedback, suggestions, and contributions are welcome.



# NDN-Android: NDN Stack for Android



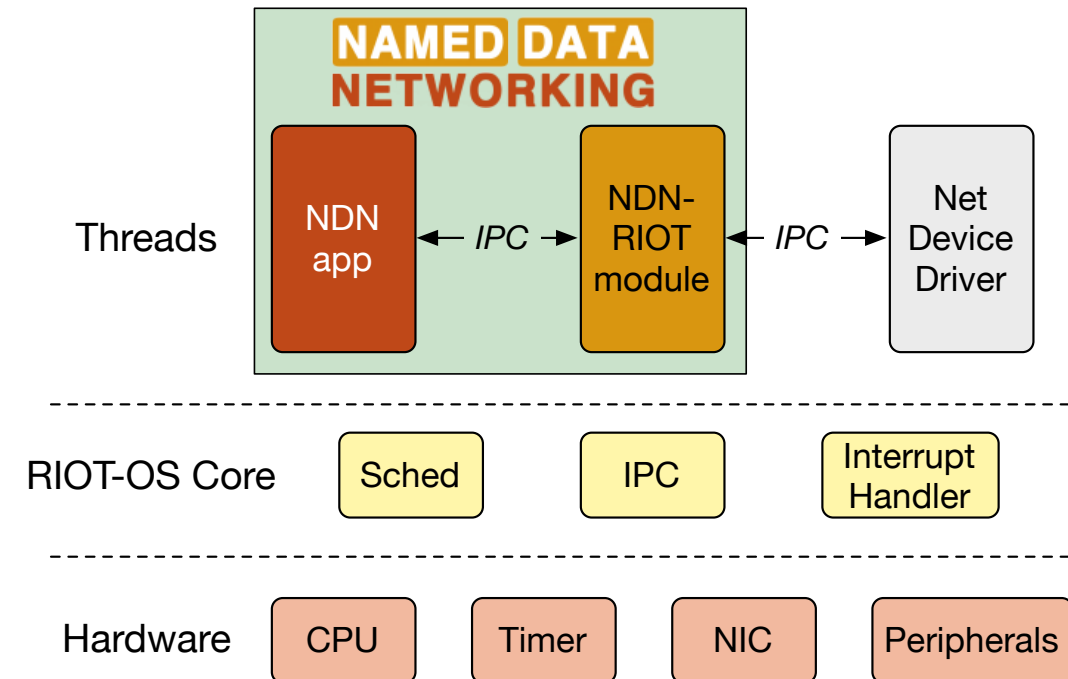
- Embeds actual NFD, compiled using NDK
- Works with all (non-rooted) Android devices

[https://play.google.com/store/apps/details?id=net.named\\_data.nfd&hl=en](https://play.google.com/store/apps/details?id=net.named_data.nfd&hl=en)

<https://github.com/named-data-mobile>

# NDN-RIOT: NDN for RIOT-OS

- Optimized for IoT apps
- Support
  - Data-centric security
  - Stateful NDN packet forwarding
  - Replaceable forwarding strategies
  - 802.15.4 and Ethernet
- Simple application APIs
- Several simple examples to get started



<https://github.com/named-data-iot>



# Getting Started with NDN-RIOT Examples

- Downloading
  - `mkdir riot`
  - `cd riot`
  - `git clone https://github.com/named-data-iot/RIOT`
  - `git clone https://github.com/named-data-iot/ndn-riot`
  - `git clone https://github.com/named-data-iot/ndn-riot-examples`
- Compiling an example
  - `cd ndn-riot-examples/<APP>`
  - For host architecture (for debugging)
    - `make`
  - For a specific RIOT board
    - `make BOARD=samr21-xpro`
    - `make flash BOARD=samr21-xpro # to flash firmware`
    - `make term BOARD=samr21-xpro # to access board via serial interface`

[ndn-benchmark](#)[ndn-consumer](#)[ndn-ping](#)[ndn-producer](#)[ndn-rtt](#)[ndn-template](#)

# NDN Tools

- ndnping, ndnpingserver
  - Reliability testing tools
- ndncatchunks, ndnputchunks
  - Segmented file transfer between a consumer and producer
- ndnpeek, ndnpoke
  - Transmit a single packet between a consumer and a producer
- ndndump, dissect, wireshark-dissect
  - Debug NDN packet flow
- repo-ng, repo-sql: NDN repositories providing managed persistent s

The logo for TCPDUMP, featuring the text "TCPDUMP" in a bold, red, sans-serif font. A thin, black, hand-drawn line loops around the "T" and "P" characters.The logo for WIRESHARK, featuring the text "WIRESHARK" in a bold, black, sans-serif font. A stylized black outline of a shark's dorsal fin is positioned above the "I" and "R" characters.

# MILCOM 2017

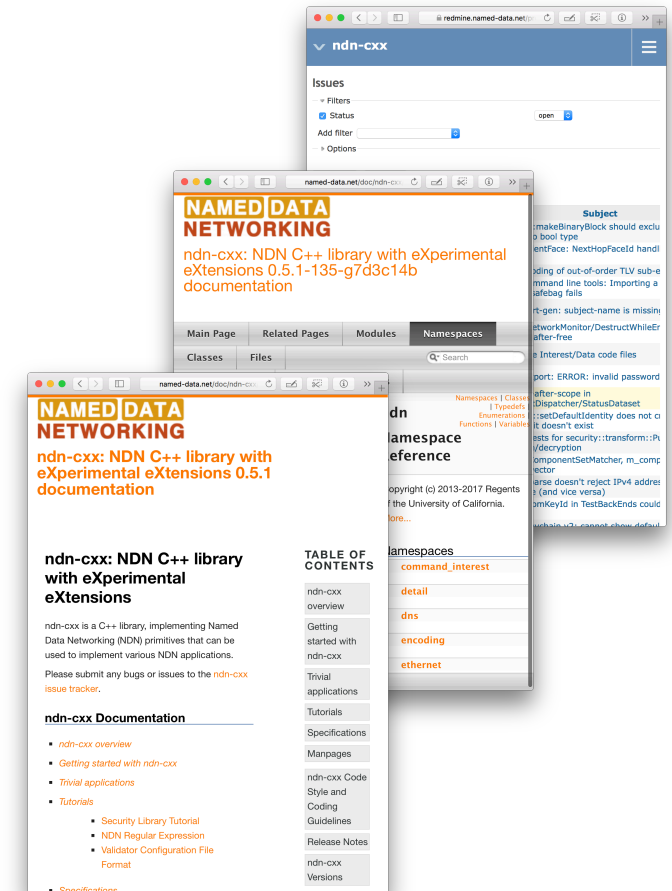
MILITARY COMMUNICATIONS AND INNOVATION - PRIORITIES FOR THE MODERN WARFIGHT

**LIBRARIES**

BALTIMORE, MD • OCTOBER 23-25, 2017

# ndn-cxx: NDN C++ library with eXperimental eXtensions

- C++11
- The reference library and security library implementation
- Used in: NFD, NLSR, ndn-tools, ChronoChat, etc.
- <https://named-data.net/doc/ndn-cxx/current/>
  - Overview
  - Getting started
  - Trivial applications
  - Tutorials
  - Specifications
  - Manpages
  - API documentation (doxygen)
- Feedback, suggestions, and contributions are welcome.





# NDN Common Client Libraries (NDN-CPP, NDN-JS, jNDN, PyNDN)

- C++, Java, Python, JavaScript, C#, Squirrel
- Used in: NDN-RTC, NdnCon, NFD-Android, etc.
- <https://named-data.net/codebase/platform/ndn-ccl/>
  - NDN Common Client Libraries API
  - NDN-CPP API
  - PyNDN API
  - NDN-JS API
  - jNDN API

**NDN Common Client Libraries (NDN-CCL) Documentation**

The NDN Common Client Libraries (NDN-CCL) are written in C++, Python, JavaScript and Java and provide a common API for client applications to use NDN. Any library in NDN-CCL suite allows an application to send interests to and receive data from an NDN forwarding daemon (NFD) and provide a large set of other functions necessary for any NDN application.

Libraries implementing the **NDN Common Client Libraries API**:

- C++ – [NDN-CPP](#), [\[language-specific issues\]](#)
- Python – [PyNDN](#)
- JavaScript – [NDN-JS](#)
- Java – [jNDN](#)

Function and class documentation: [NDN-CPP](#), [PyNDN](#), [NDN-JS](#), [jNDN](#).

Potential contributors to the NDN-CCL should review the [NDN-CCL Development Guidelines](#).

**Supported Features**

Feature	NDN-CPP	PyNDN	NDN-JS	jNDN	Notes
<a href="#">MemoryContentCache</a>	✓	✓	✓	✓	
<a href="#">ChronoSync2013</a>	✓	✓	✓	✓	
<a href="#">Name.Component from*</a>	✓	✓	✓	✓	
<a href="#">Name.Component is*</a>	✓	✓	✓	✓	
<a href="#">Name.Component to*</a>	✓	✓	✓	✓	
<a href="#">ImplicitSha256DigestComponent</a>	✓	✓	✓	✓	
<a href="#">ProtobufTlv</a>	✓ API	✓ API	✓ API	✓ API	
<a href="#">SegmentFetcher</a>	✓ API	✓ API	✓ API	✓ API	

# MILCOM 2017

MILITARY COMMUNICATIONS AND INNOVATION - PRIORITIES FOR THE MODERN WARFIGHT

A silhouette of a person in profile, talking on a mobile phone. The background is a green-toned network of interconnected nodes and lines, with some nodes glowing. The overall theme is military communications and technology.

## EVALUATION TOOLS AT DIFFERENT SCALES

BALTIMORE, MD • OCTOBER 23–25, 2017

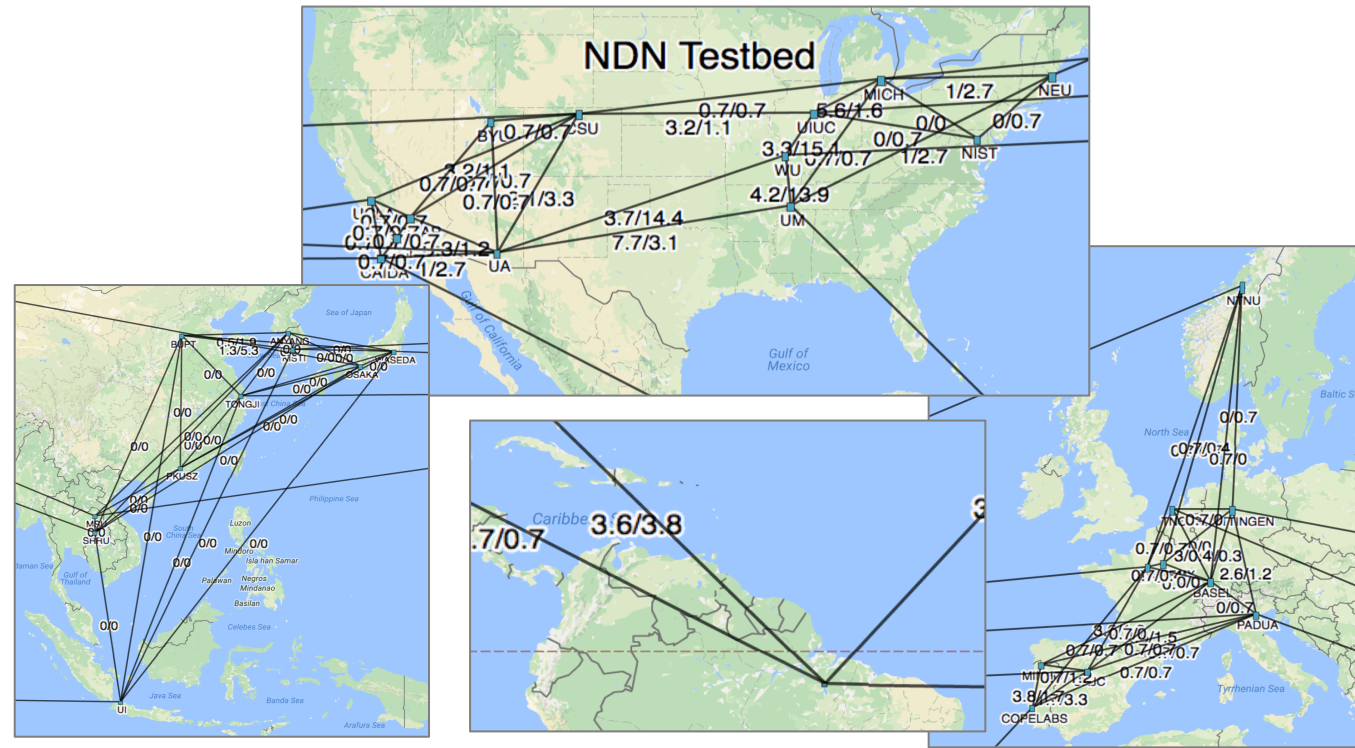
## NDN Testbed

- Network of 37 sites across 4 continents, 14 countries

**Open to join and use**

<https://named-data.net/ndn-testbed/policies-connecting-nodes-ndn-testbed/>

- Examples applications and experiments: videoconferencing, network management, virtual machine migration, strategies, nTorrent, etc.
- Small scale evaluations





# Open Network Lab (ONL)

- Remotely accessible network testbed
  - Operated and maintained by Applied Research Lab in Department of Computer Science and Engineering at Washington University in St. Louis
  - Real Hardware for running repeatable network experiments with trusted results. (NOT simulations)
- Use for NDN
  - NDN installed on each host/VM
  - NFD performance study
  - NDN Testbed Emulation to test new releases
- How to join?
  - <https://onl.wustl.edu/>
    - And “Get an account”



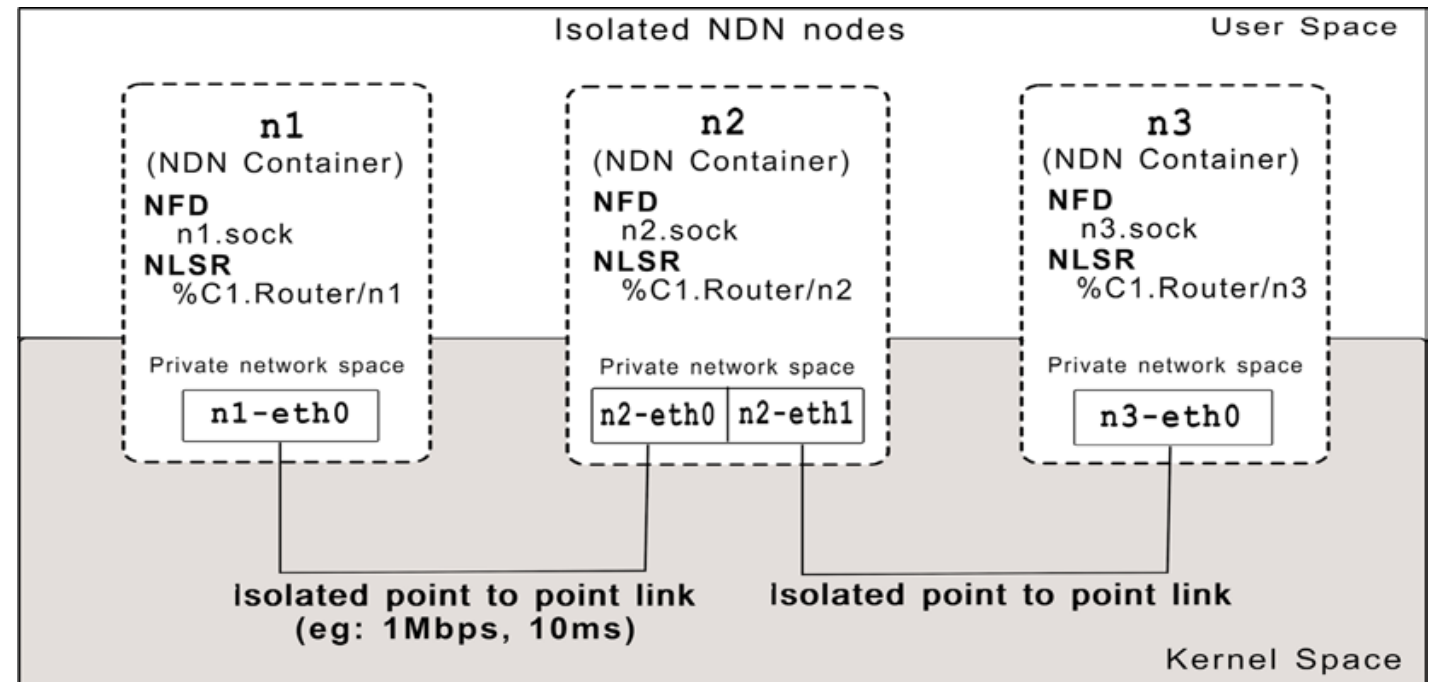


# MiniNDN: NDN Emulation Framework (Based on MiniNet)

## Runs actual instances of NFD, NLSR

### Medium-scale evaluations

- Easy to configure network emulation
- Runs any real application
- Number of emulated nodes  $\propto$  CPU power
- Cluster edition can be used to scale emulations



## ndnSIM: NDN Simulation Framework (Based on NS-3)

Fully integrated with NDN prototype implementations: NFD & ndn-cxx

Large scale evaluations

- Provide interoperability between simulation and prototyping
- Enable a two-way of experimentation and evaluation
- Enable high-fidelity NDN simulations
- 1500+ nodes with WiFi channels in the evaluation of NDN for vehicular networking

