Tutorial Intro:
Secure and Friendly Deployment (Plugging) of NDN Apps

Alex Afanasyev (FIU)

Tutorial: Power of Trust Schemas for Easy and Secure Deployment of NDN Applications
Named Data Networking as a New Paradigm

• Named Data Networking (NDN) can bring great benefits to applications
  • Directly using application data names to communicate
  • Built-in security support that secures named data directly
  • stateful dataplane with in-network caching: multipath forwarding, multicast delivery
Setting up NDN Apps

• One builds a new app, how to make it function?
  • How to bootstrap an app into the network?
  • What, where, and by whom need to be configured?
  • Putting everything on the same table: what steps to take?

• One builds a distributed app to run over multiple remote computers
  • How to securely configure remote NDN boxes?

The need for plug in order to play
Looking Back on IP Configuration

• Plug in IP networking: establish IP connectivity on a specific IP subnet
  • IP address, subnet mask, default gateway
  • DHCP automates the last step of configuring individual host
    • Network operators manually configure IP address block and subnet mask into DHCP servers

• To enable application communications
  • Need names
    • Network operators also configure DNS resolver address into DHCP servers
  • Need security support
    • PKI “trust” based on OS and browser vendors decisions to trust PKI CAs “on behalf” of users
NDN Configuration

• Plug in NDN networking: establish <what>? IP connectivity on a specific IP subnet
  • IP address, subnet mask, default gateway
  • DHCP automates the last step of configuring individual host
    • Network operators manually configure IP address block and subnet mask into DHCP servers
  • NDN apps are NDN network entities, so they are need to be bootstrapped / plugged in / (auto) configured!

• To enable application communications
  • Need names
  • Need security support
Network Model of NDN

- A networked system is made of **named entities**
  - Entities are anything produce and/or consume immutable named packets
    - services / application instances
  - Entity names are decoupled from network attachment points
    - Entity can explore available connectivity to communicate on their own

- There exist various **trust relations** among the named entities
  - Hierarchical
  - Peer-to-peer
NDN Config === NDN App Config

• Where an entity obtains its **name** and **security credentials**
• How the initial **trust relations** are configured into the entity
What About Connectivity?

• Yes, NDN entities also need to establish connectivity among each other to let NDN packets flow
  • an entity can express Interests and they will flow towards the data
  • an entity can attract Interests if it has matching data to publish

• Multiple options to establish NDN connectivity
  • forwarding state
  • forwarding strategy
  • routing state
  • overlay tunnels

Physical links are **necessary** but not **sufficient** for NDN connectivity
(same as with IP)
With Names & Security, Connectivity Can be Setup

A number of tools have been developed to help set up NDN connectivity

- **NDN Routing**
  - Exchange *secured name* prefix information and builds routing state (proactively)
- **Auto-prefix propagation / prefix readvertise**
  - Automated means to push forwarding/routing state to attract interests
- **Self-Learning**
  - Leverages forwarding strategy to reactively build forwarding state (directions where authentic data can be found)
- **NDN Over WiFi Direct**
  - Overlay management and *secured name* prefix exchange

- **ndn-autoconfig**
  - Constructs/maintains overlay tunnels to closest NDN hub
- **NDN-FCH**
  - Constructs/maintains overlay tunnels to closest NDN hub
- **NDN Neighbor Discovery**
  - Constructs/maintains overlay tunnels to neighbors

Virtual physical links, not a complete NDN connectivity solution
ndn-autoconfig

Synopsis

```bash
ndn-autoconfig [-h] [-V] [-c file] [-d]
```

Description

Client tool to run NDN hub discovery procedure.

Options

```bash
-d OR --daemon
```

Run ndn-autoconfig in daemon mode. In this mode, the auto-discovery procedure is re-run hourly or when a network change event is detected.

NOTE: if connection to NFD fails, the daemon will be terminated.

```bash
-c FILE OR --config=FILE
```

Use the specified configuration file. If `enabled = true` is not specified in the configuration file, no actions will be performed.

Showed our own lack of understanding on what kind of config NDN needs
Plugging NDN Entities into NDN Networks

• NDN’s network model requires one named entity to establish trust relations with others
  • Name
    • carrying application semantics
  • Certificate
    • enabling one to produce authenticatable data and verify received data
  • Trust anchor
    • establishing the trust relations of entities under a namespace
  • Trust policies
    • limiting the power of signing key to data with specific names
Configuring a Trust Anchor and Trust Policies

• Deployment/application parameters
  • Trust schema defining data/key name relations
  • Which key can sign what data / privilege separation

• Determine the local trust zone (and its scope)
  • Trusted microcosm “boss” (of the local trust zone)

• After trust anchor and trust policies bootstrapping
  • App can receive and authenticate data from trust zone entities
Configuring a Name and Certificate

• Application semantics
  • Depends on specific app what name is and how it is structured

• Assigned or selected
  • Depends on out-of-band (outside bootstrapping) knowledge of what it is
  • By admin, app owner, app developer, etc.

• Certificate issued based on proof-of-control over the namespace within a “trust zone”
  • ”Security challenges”, physical challenges (for proximity proof), or predefined knowledge (codes)

• After trust anchor and trust policies bootstrapping
  • App can publish authenticatable data for other trust zone entities
An Example of NDN Entities

- **App-defined**
  - `/ndnfit` self-signed certificate
  - `/ndnfit/KEY`
    - Signature
    - Trust Anchor
  - `/ndnfit/bob/KEY`
    - Signature
    - Anchor Cert

- **Local Trust Anchor**
  - Alice’s certificate
    - `/ndnfit/alice/KEY`
    - Signature
  - Anchor Cert

- **Analyzer App**
  - Digital Keys
  - Trust Policies
  - Anchor Cert

- **Sensor App**
  - Digital Keys
  - Trust Policies
  - Anchor Cert

Self selected (first come first serve basis), app approved
Terminology for the Rest of Tutorial

- Any NDN entity can become a (local) trust anchor $T$
- All NDN entities under the same trust anchor make a Trust Zone
- Owner of the trust anchor $T$ is the Controller of this trust zone

**Can be self-signed (may require different logic for name selection)**
Definition of NDN Configurations

- Plugging/Configuring a new entity $E_{new}$ is
  - Configuring $E_{new}$ into a trust zone
    - $E_{new}$ must have a name, obtained on its own, or otherwise assigned by the Trust Zone Controller
    - $E_{new}$ must have its trust anchor, certificate and trust policies installed
Logical steps of security bootstrapping: Step 1

• Mutual authentication between Trust Zone Controller and $E_{new}$
  • Trust Zone Controller authenticates $E_{new}$ to confirm its trustworthiness
  • $E_{new}$ authenticates Trust Zone Controller to be its authority
    • In order to accept the Trust Zone Controller’s self-signed certificate as trust anchor
Steps 2: Obtain Trust Anchor and Policies

- After mutual authentication, $E_{\text{new}}$ can obtain trust anchor.
- Trust anchor establishes the trust relation between $E_{\text{new}}$ and Trust Zone Controller.
Step 3: Obtain/Update Trust Policies

- $E_{new}$ fetches trust policies that Trust Zone Controller has defined for it
- Trust Zone Controller may change trust policies from time to time
- $E_{new}$ can fetch new trust policies securely in the same way as fetching other named data
Step 4: Obtain Certificate

- $E_{\text{new}}$ obtains certificate issued by Trust Zone Controller
  - $E_{\text{new}}$ validating certificate issued by Trust Zone Controller

Ready to Publish Data
Bootstrapping Mechanics

• Use-case specific mutual authentication process
• Manual or protocol-specific trust anchor and policies (schema) installation
• Manual or protocol-specific obtaining a certificate
  • Manual
  • NDN-CERT
Trust Zone Controller Returning Identity Bundle

- Trust Zone Controller can bundle \(<\text{trust anchor, certificate, trust policies}>\) in one data object as the reply to the configuration request
- Chatroom app installs the components inside bundle

more in the next part of tutorial
Exploring Problem Space in Security Bootstrapping

• How to accomplish mutual authentication
  • Solutions depend on use case scenarios

• Generalized used case scenarios

  • **Bootstrapping** $E_{\text{new}}$ in secured local environment
    • Physically secured environment:
    • No third party can communicate with either Trust Zone Controller and $E_{\text{new}}$
    • mutual authentication: the only party that can communicate with $E_{\text{new}}$ is the controller, and vise versa

• Bootstrapping $E_{\text{new}}$ in unsecured local environment

• Bootstrapping remote $E_{\text{new}}$
Moving on to part 2 of the tutorial Creating and Using Trust Schemas in 1st Use Case