NDNSEC Tools and Trust Bootstrapping

Sanjeev Kaushik Ramani (FIU)
Zhiyi Zhang (UCLA)
Contents

◊ NDNSEC Command Line Tool Suite
◊ NDN Security Bootstrapping
NDNSEC Introduction

◊ Command-line toolkit to perform various NDN security management operation

◊ NDN security data are stored and managed in two places
  o Public Information Base
  o Trusted Platform Module

◊ Usage:
  o ndnsec <command> [<args>]
  o ndnsec-command [<args>]

3
Identity, Key, and Certificate in NDN Security

- Keys, certificates and their identities are managed by KeyChain
- Real-world identity can be expressed by a namespace
  - /ndn/edu/ucla/alice
  - /ndn/edu/fiu/MeritLab/PG6142
- Identity: a named entity in NDN
- Key: owned by an identity. Each identity can have more than one keys.
- Certificate: used to certify a key. Each key can have more than one certificates issued by different issuers/authorities
NDNSEC Command Line Tool Suite

```
→ ~/ndnsec help

help           Show all commands
version        Show version and exit
list           Display information in PublicInfo
get-default    Get default setting info
set-default    Configure default setting
key-gen        Generate a Key-Signing-Key for an identity
sign-req       Generate a certificate signing request
cert-gen       Generate an identity certificate
cert-dump      Dump a certificate from PublicInfo
cert-install   Install a certificate into PublicInfo
delete         Delete identity/key/certificate
export         Export an identity package
import         Import an identity package
unlock-tpm     Unlock Tpm
```
Example 01: Try ndnsec-ls-identity -c to see your identities, keys, and certificates
Example 02: Try to create a Self-signed Certificate

\[\text{ndnsec-key-gen <your identity name>}\]

\[\text{<- Your self-signed certificate}\]
Code and Documentation of NDNSEC

◊ Manpages of NDNSEC

◊ Source code of NDNSEC (Also a good chance to learn ndn-cxx security APIs)
Trust Anchor (TA) to Bootstrap Security

◊ Trusted self-signed certificate
  o Certificate of locally trusted authority

◊ Trusted non-self-signed certificate
  o Trusting only CS department, but not the whole university

◊ Necessary option to properly validate packets
Options to Configure Trust Anchor in the Libraries

♢ Hard-coded in the code 😞

♢ Configured with trust schema 😊
  o One or multiple
  o Directly defined (base64)
    ▶ Static
  o Indirectly via file name
    ▶ Static
  o Double indirection via directory name
    ▶ Static
    ▶ Dynamic
  o “any” (disabled security)

```
trust-anchor
{
  type file
  file-name "trusted-signer.cert"
}
trust-anchor
{
  type base64
  base64-string "Bv0DGwdG...amHFvHIMDw=="
}
trust-anchor
{
  type dir
  dir/usr/local/etc/ndn/keys
}
```

ValidatorConf syntax
Dynamic Trust Anchor(s)

◊ A refresh period can be set in cases where the certificate changes during runtime

```
trust-anchor
{
  type dir
dir /usr/local/etc/ndn/keys
  refresh 1h ; refresh certificates every hour, other units include m (for minutes) and s (for seconds)
}
```
Automated Bootstrapping in Single Domain Environment (Smart Home Example Scenario)

◊ **Home Controller**
  - Android Phone (/home/controller/android-phone)
  - Linux Laptop (/home/controller/linux-laptop)

◊ **IoT Devices**
  - Living Room Camera (/home/living/camera)
  - Living Room Television (/home/living/tv)
  - Bed Room Camera (/home/bed/tv)
  - Bed Room Camera (/home/bed/camera)
Automated Bootstrapping Goals and Steps

◊ Enable device to trust the network with minimal user intervention
  o Obtain the local trust anchor
  o Every home has a global unique name along with a key-pair and a certificate of its public key
  o The certificate serves as the local trust anchor

◊ Enable device to be trusted by the network
  o Obtain an anchor-signed certificate
  o Device has a key signed by the trust anchor and issued by the controller
  o Device uses this key (referred to as communication key) to sign packets, which can be authenticated by other devices in this network
Establishing Mutual Trust

◊ High Level Requirements
  o Prevent a device from joining the wrong network
  o Prevent a malicious device from joining network
  o Prevent impersonation / replay attacks by onboarded devices in network

◊ In the current example:
  o Android Phone / Linux Laptop (controller) would bootstrap the home IoT devices.
  o After bootstrapping, each device can install the trust anchor and obtain an identity certificate.
Bootstrapping Assumptions

◊ Physical connectivity between controller and device exists before bootstrapping
  o Wi-Fi, Bluetooth

◊ Out-of-band: controller obtains bootstrapping info from device before enrolls it
  o A bootstrapping key (public key) $\rightarrow B_k$
  o By other mean like QR code scanning
Bootstrapping Steps

◊ The device initiates the process by broadcasting a request for bootstrapping; then the controller replies with the local trust anchor.

◊ Then the device generates a key-pair and requests the controller to sign its public key (the communication key); the controller signs that key by the trust anchor and returns the anchor-signed certificate to the device.

---

1. Device: sign-on request: initialize bootstrapping

2. Controller: reply with the local trust anchor

3. Device: certificate request: send public key for signing

4. Controller: reply with an anchor-signed certificate
Threat Countermeasures

◊ Device signs the first Interest with the bootstrapping key; then the controller can authenticate the device and thus perceive any fake device.

◊ Device and controller negotiate a temporal symmetric key, similar to DH, by exchanging two tokens; this key is used to sign following packets to ensure their integrity and both ends’ authenticity.

◊ Meanwhile, for every token, its presence in the next packet prevents replay attacks.
Threat Countermeasures (Contd.)

1. **Device**
   - **Interest**
   - **Data**
   - **Time to trust the other end**

   - **sign-on request:** put a random number (token1) here
   - sign with the bootstrapping key

2. reply with the trust anchor; present token1 and put token2 here
   - sign with a temporal symmetric key

3. **Certificate request:** present token2;
   - sign with the temporal symmetric key

4. reply with **an anchor-signed certificate**;
   - sign with the temporal symmetric key
Becoming Your Own Trust Anchor

◊ Demo

  o Generate self-signed certificate
  o validator.conf example
  o Proof-of-correctness using simple demo app
Manually Generating Certificates

♦ ndnsec cert gen demo