NDN-MPS: Supporting Multiparty Authentication over NDN

Zhiyi Zhang (presenter), Siqi Liu, Randy King, Lixia Zhang
UCLA, Operant Networks

ACM ICN 2021
Multiparty Authentication

- Real world business decision involves multiple parties

- Real problem we met
  - Solar energy network system
  - Inverter software update command requires approvals from multiple parties
    - Site owner
    - Site operation team
    - Manufacture QA team
Switch from Prod-Con Trust Model to Multiparty Trust

- Third party signers who are not the content producer
- Verification against a list of signers
- Coordination among the signers
Crypto: Existing Schemes vs Multisignature

● Conventional solution: obtain a list of signatures from individual signers
  ○ Large packet/signature size $O(n)$
  ○ Long verification time $O(n)$

● Multisignature: multiple signatures can be aggregated into one
  ○ Single signature $O(1)$
  ○ Single verification operation $O(1)$
What is missing?

- Trust schema for each signer to verify the producer and vice versa
- Multiparty trust schema
  - To defines signing and verification rules that involve multiple signers (and trust anchors)
- Multisignature encoding
  - To encode signature and its multi-party specific signature information
- The coordination mechanism
  - To collect and aggregate signatures from individual signers

This can be addressed by existing trust schema support

The rest are new issues
NDN-MPS: Toolkit for Multisignature based Multiparty Authentication

- Multisignature trust schema support
- An NDN-compatible multisignature encoding mechanism
- Two coordination mechanisms for multisignature generation
  - NDN Remote Procedure Call (RPC) based coordination
  - NDN sync-based coordination
Multisignature Trust Schema

- A list of required signer identities
- These signer’s certificate chains to one or more trust anchors
- Threshold policy: valid when k out of n signers sign the object
  - NDN-MPS support this with a system approach rather than using additional cryptographic primitives for simplicity of key setup and management

```plaintext
Data profile: /Site/inverters/firmware/update
All-of {   /Mfr/QA*/KEY/*
          /Site/operation/*/KEY/*
          /Site/Owner/*/KEY/*  }
Known-signer {
    ...
}

Data profile: /site/operation/command/shutdown/*
All-of {   /Site/Owner/*/KEY/* }
At-least-num 2
From {   /Site/operation/manager1/KEY/*
         /Site/operation/manager2/KEY/*
         /Site/operation/manager3/KEY/*
         /Site/operation/manager4/KEY/*  }
```
Multisignature Encoding

- New signature type
- New key locator to keep information of multiple signers
  - Must be consistent among multiple signers
  - Must tolerate changes of signer list during the coordination:
    - One required signer /site/owner/*/KEY/*
    - The producer decides to go with /site/owner/alice first
    - When Alice is down, change it to /site/owner/bob
- Solve the problem with another layer of indirection: placeholder key locator
Multiparty Signing Coordination: What is needed?

- First step: the producer publishes the unsigned data object to signers
- Second step: collect signature pieces from signers

Security objectives:
- Authenticity
- Confidentiality: just like in prod-con trust model: content is not available until it is packetized
Multiparty Signing Coordination: RPC

- RPC based: NDN-MPS RPC
  - Diffie-hellman key exchange in the first round trip to ensure confidentiality
  - Asynchronous: informed estimated processing time
  - Repo-friendly: both parameter and result can be published to repos
Multiparty Signing Coordination: Sync

- **NDN Sync based:** E.g., SVS
  - Require group-level encryption
  - Require group identity management

- **Two approaches work for different applications**
  - Already use sync?
  - Want simple setup?
Implementation and Evaluation

- A C++ library with usable APIs (works over ndn-cxx library)
  - BLS signature: no interactive key setup
  - Also integrated into ndncert as a multiparty-approved identity verification challenge

- Benchmark with different size of data and signer set
  - Confirmed $O(1)$ signing and verification time
  - Confirmed $O(1)$ signature size: 128 bit security requires 96 bytes signature regardless of # of signers
Thank you!

Q&A