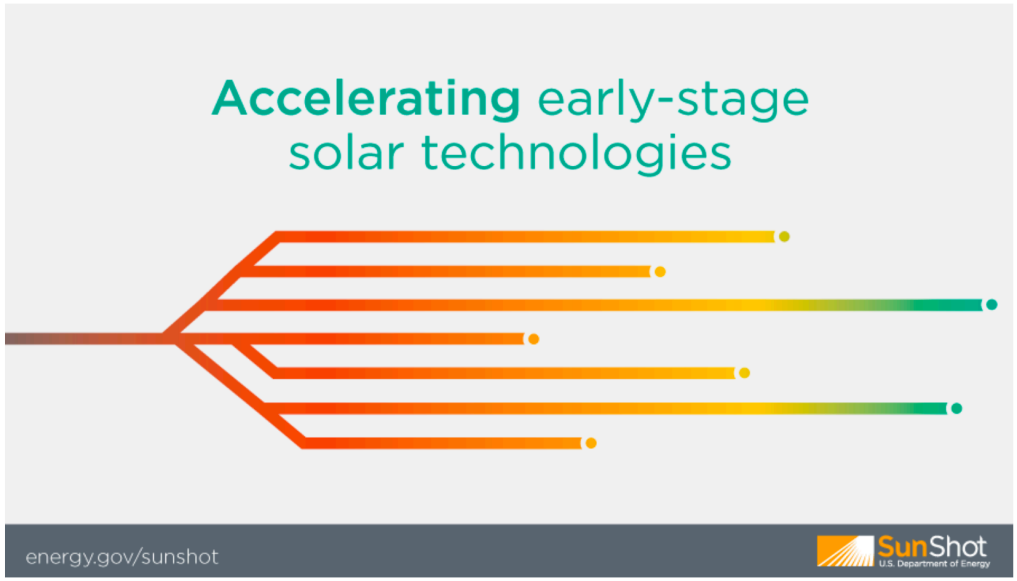
# DEMO
# Cloud-optional Home IoT w/NDN

UCLA IRL




The Technology to Market 3 (T2M3) funding program accelerates the research and development of highly impactful solar energy technologies. T2M3 encourages continued innovation within every sector of the solar economy, from manufacturers to financiers to installers. Selected projects focus on research to address early-stage, pre-commercial risks so that those projects can subsequently attract private follow-on funding. At a high level, T2M3 projects develop products to leverage new and emerging technologies, increase system values while reducing hardware costs, improve business operational efficiency, expand the investor pool for project development, and increase consumer access to solar.

## Operant Solar Corporation

**Location:** Santa Rosa, CA

**SunShot Award Amount:** $480,000

**Awardee Cost Share:** $120,000

**Project Summary:** This project will research a new concept in photovoltaic system communications using Named Data Networking, which has the potential to lower the cost of maintaining active contact with customer sites by lowering the annual disconnection rate below 1%.

# Motivation

- NDN primitives already provide benefits to IoT systems, we will demonstrate that here.
- IOTDI '16 and '17 papers discuss building higher-level functionality.

- Basic idea for today: Avoid unnecessary cloud dependence!
  - Cannot add devices or authorize users when cloud is inaccessible.
  - Additional delay even if command issuers and target devices reside on the same local network.
  - Unnecessary data exposure from the local network to the external parties may lead to security and privacy issues.

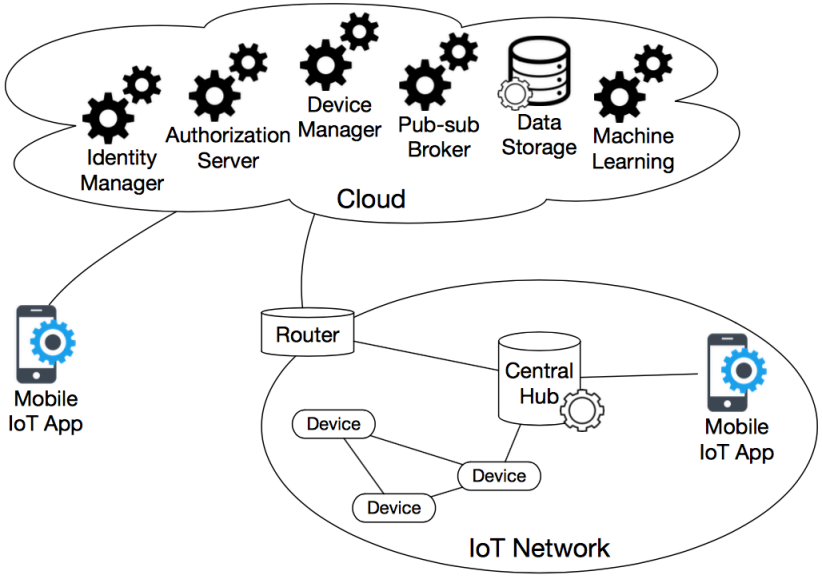
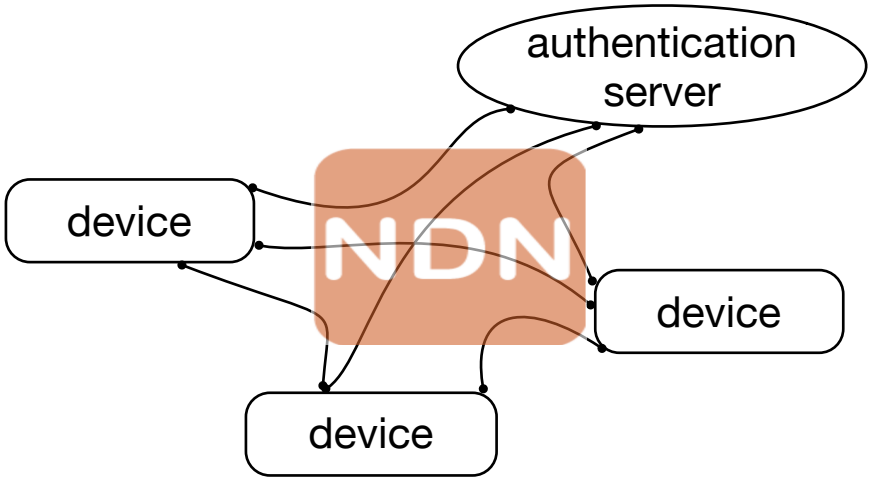
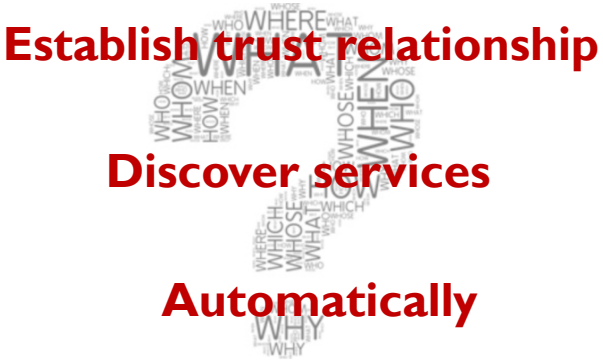



Figure 1: typical cloud-centric IoT architecture

[1] Shang, Wentao, et al. "Breaking out of the cloud: local trust management and rendezvous in named data networking of things." Internet-of-Things Design and Implementation (IoTDI), 2017 IEEE/ACM Second International Conference on. IEEE, 2017.

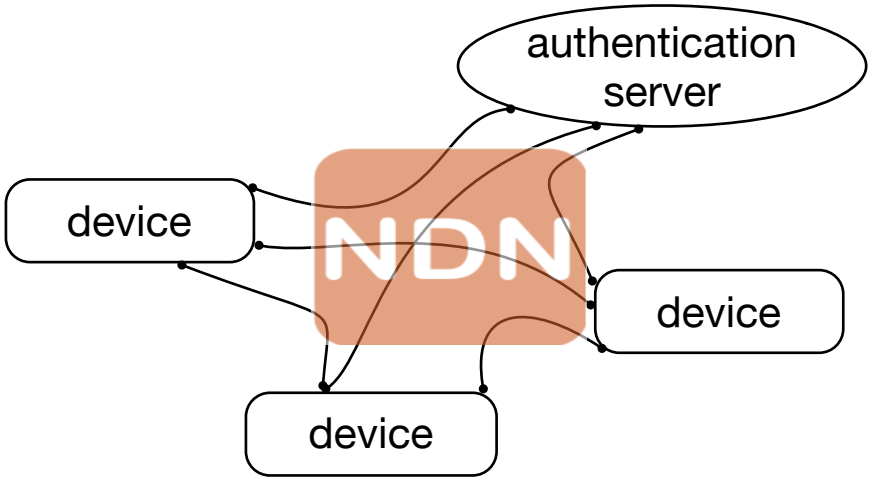
# NDN-based Home IoT

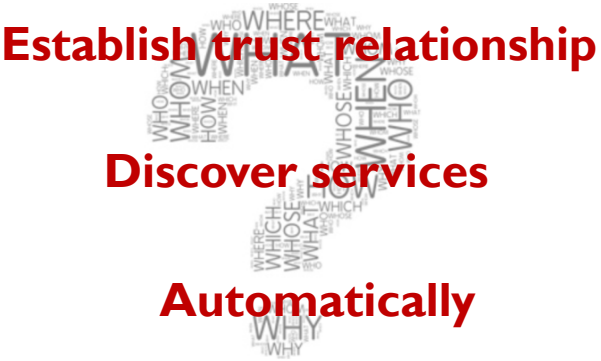



- Authentication Server (AS)
  - Serves as the trust root within this home network and a local CA
  - Be responsible to add devices and authorize users
  - May be a control hub, home router or smart phone owned by the owner of this home network.
- Devices
  - Resource producers and / or command handlers
    - Temperature sensors, camera monitors, light controllers, etc.
  - Resource consumers and / or command issuers
    - Smart phones, latptops, etc.,

# NDN-based Home IoT




- Key benefits "out of the box" from NDN
  - Schematized trust management
    - All data signed, no perimeter security required, though it can be implemented based on data naming.
    - Completely local management breaks the dependencies on Clouds
    - Hand over to external parties (i.e. CA in the cloud) smoothly whenever required
  - Data-centric and consumer-driven communication
    - Forget about where they are and how to reach them
    - Focus on what you want and how to get them
  - Cross-layer protocol stack and rich tools simplify network configuration

# Bootstrap: A few steps to join NDN-based Home IoT

- Basic assumptions
  - Each device has a physical connectivity to the authentication server
    - e.g., Ethernet, ad-hoc wifi, etc
  - Each device has an unique ID (at least locally) and keeps it as secret
    - Maybe a barcode set by its manufacturer, or a pin code assigned by the owner of this Home network
  - There is an out-of–bound way to share device's secret to the authentication server
    - e.g., scan the barcode or enter the pin code

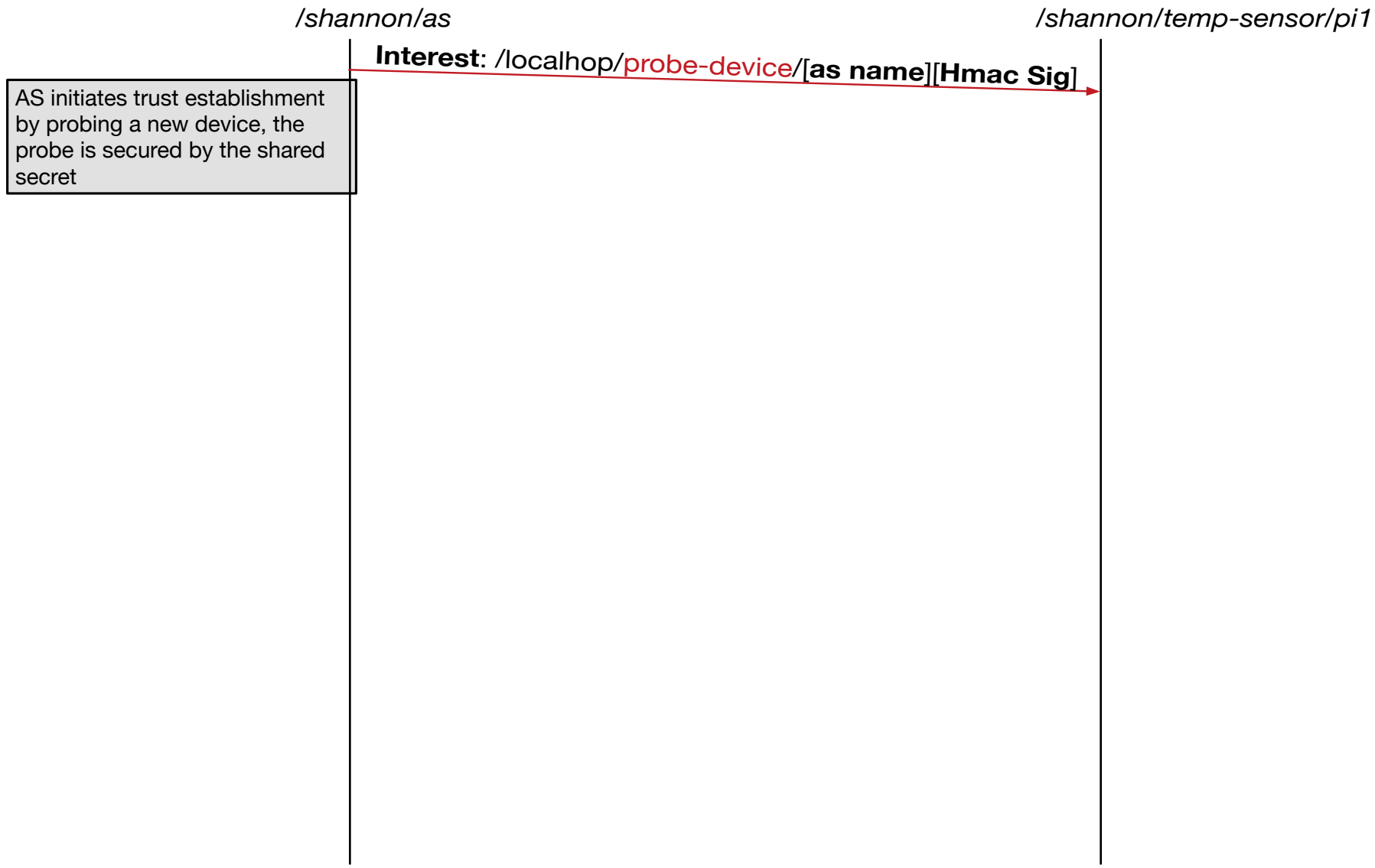
- High level bootstrap process
  - The owner gives the ID of a new device to the AS
  - Authentication server initiates the bootstrap by probing for the device
    - Secure the process by a ***secret*** shared from the target device
  - The device applies generates a key-pair and asks AS to sign it
    - AS will sign and reply with the ***signing certificate*** first
  - The device sets trust anchor and requests the ***signed certificate***

# Demo

- Implementations
  - C++
  - Ndn-cxx
  - NFD runs separately
- Basic configurations
  - A laptop acts as the AS (/shannon/as)
  - 2 Raspberry Pi emulates temperature sensors
    - /temp-sensor/pi1
    - /temp-sensor/pi2
  - Another laptop acts as the consumer who is interested in temperatures
    - /alice/macbook
- What to demonstrate
  - The bootstrap process that enable devices join the trusted network
  - The auto-discover process that help devices find out services in network

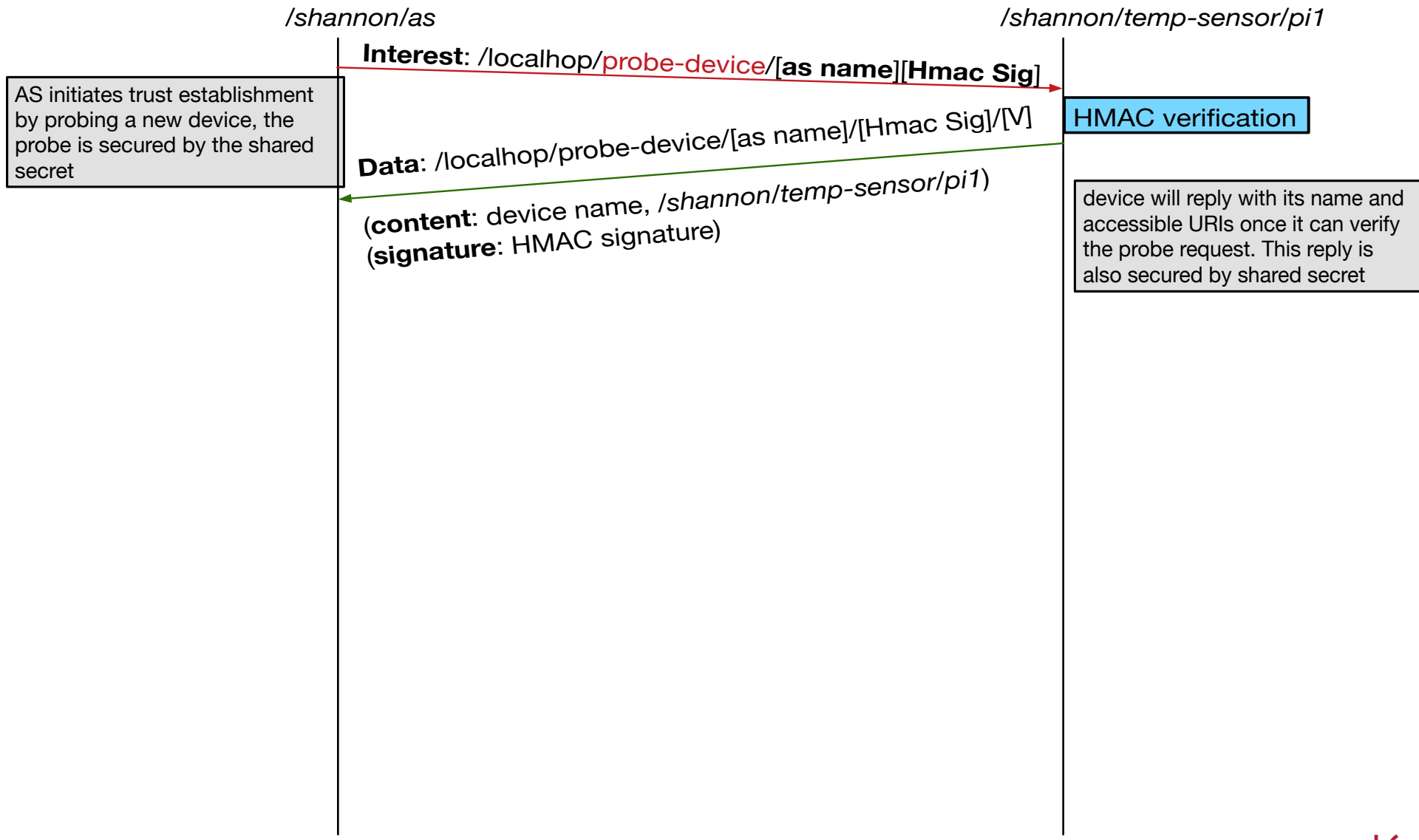
# Bootstrap: enable devices join the trusted network

## step 1 --- AS probes the device



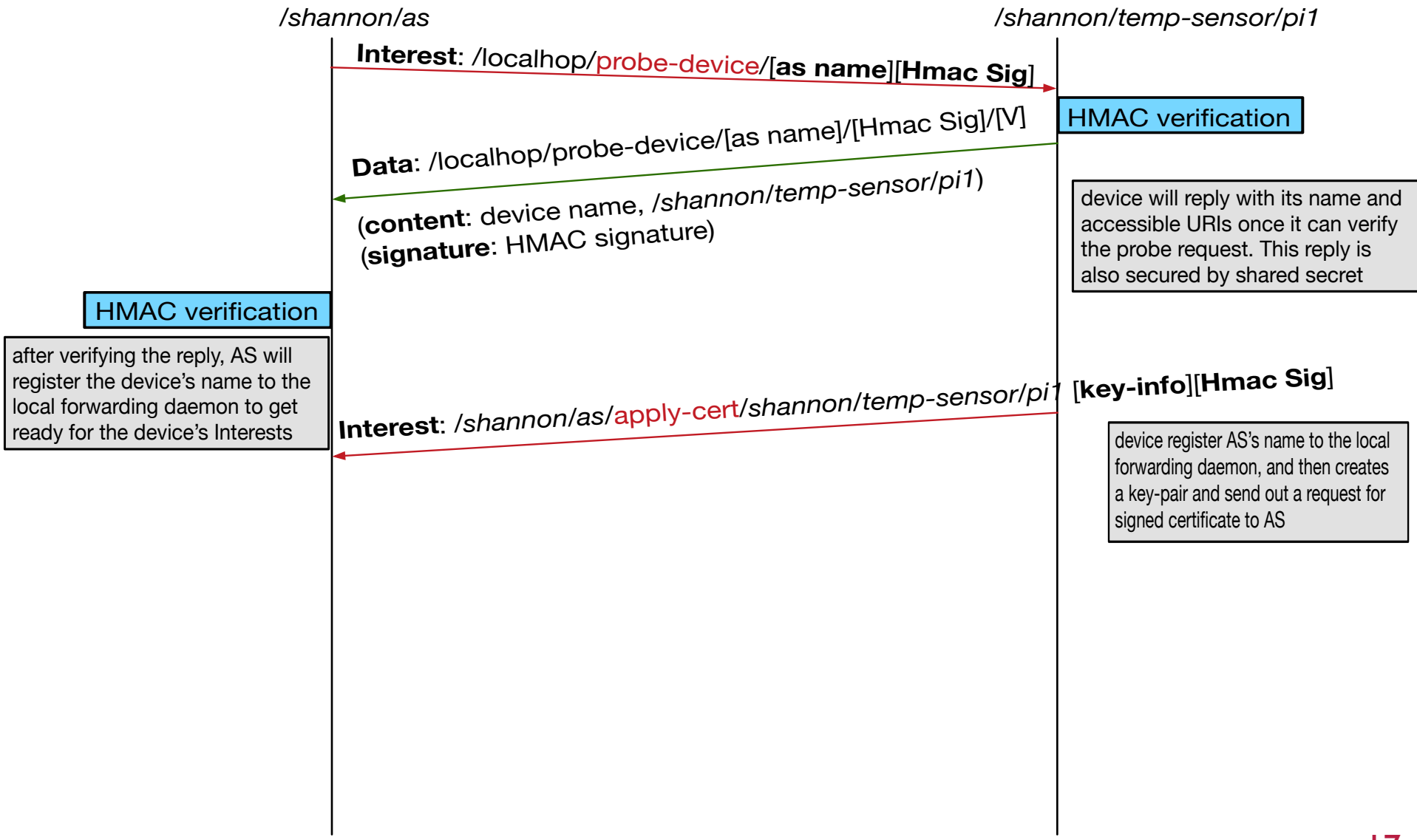
# Bootstrap: enable devices join the trusted network

## step 1 --- as probes the device



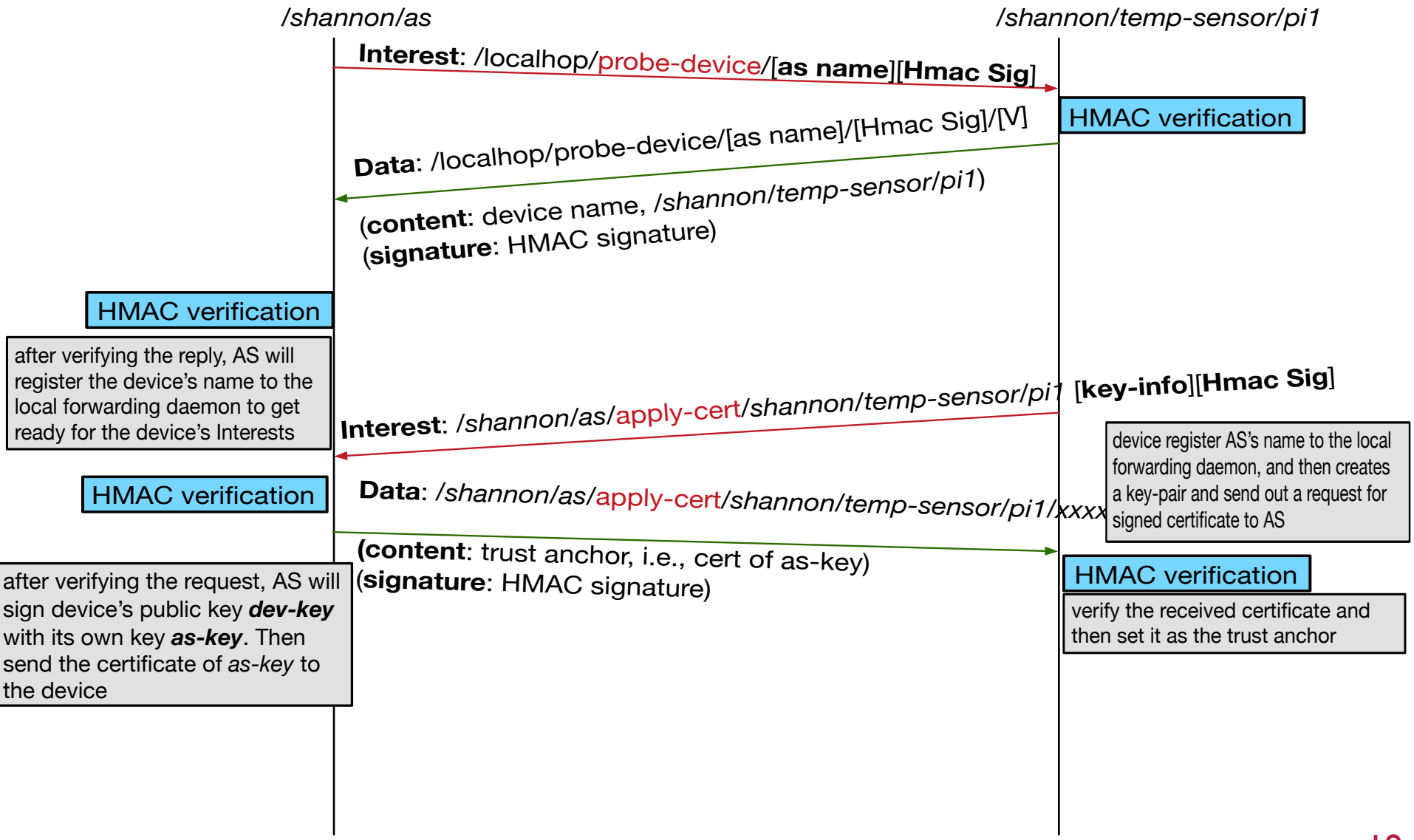
# Bootstrap: enable devices join the trusted network

## step 2 --- device applies for as-signed certificate



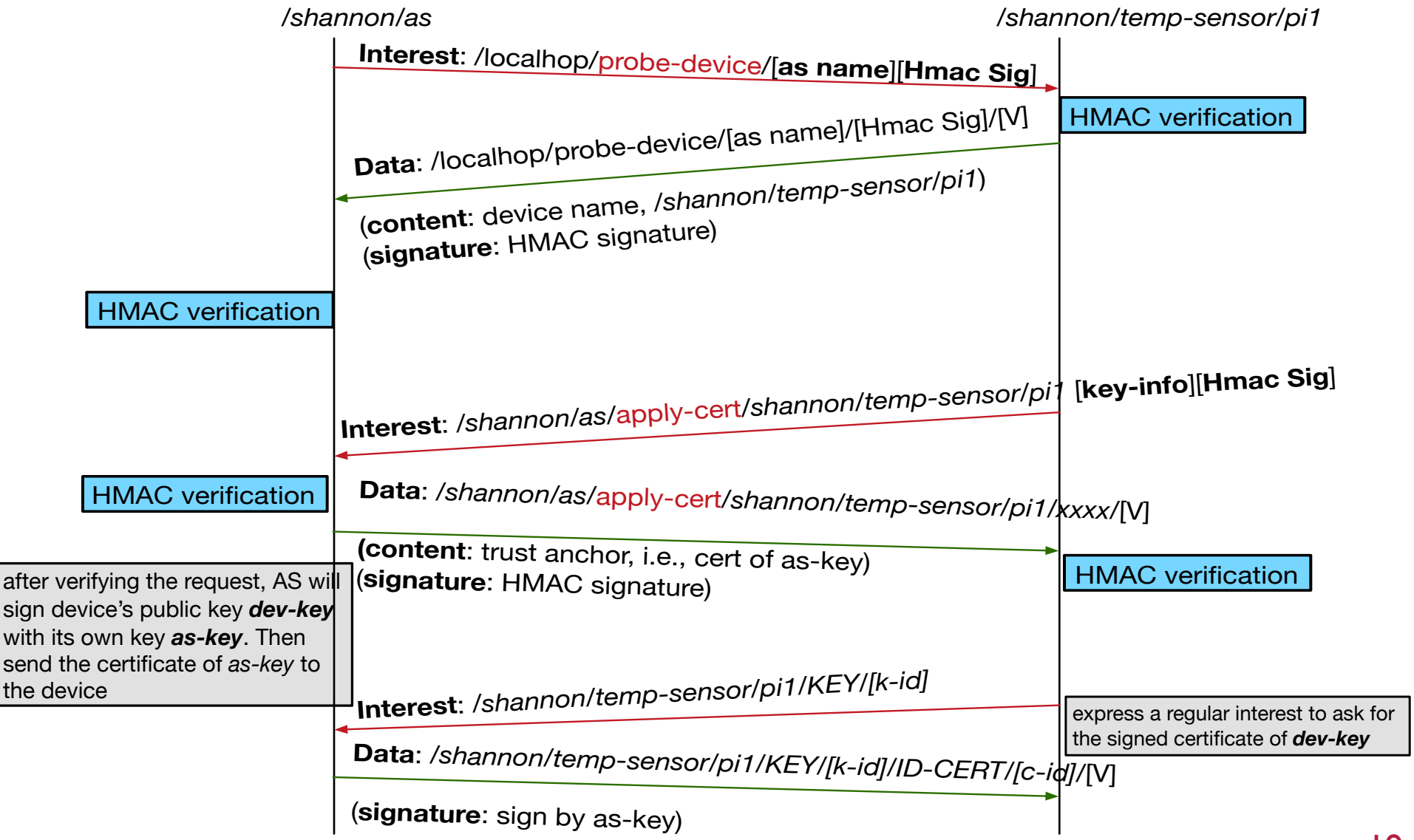
# Bootstrap: enable devices join the trusted network

## step 2 --- device applies for as-signed certificate



# Bootstrap: enable devices join the trusted network

## step 3 --- device requests for the as-signed certificate

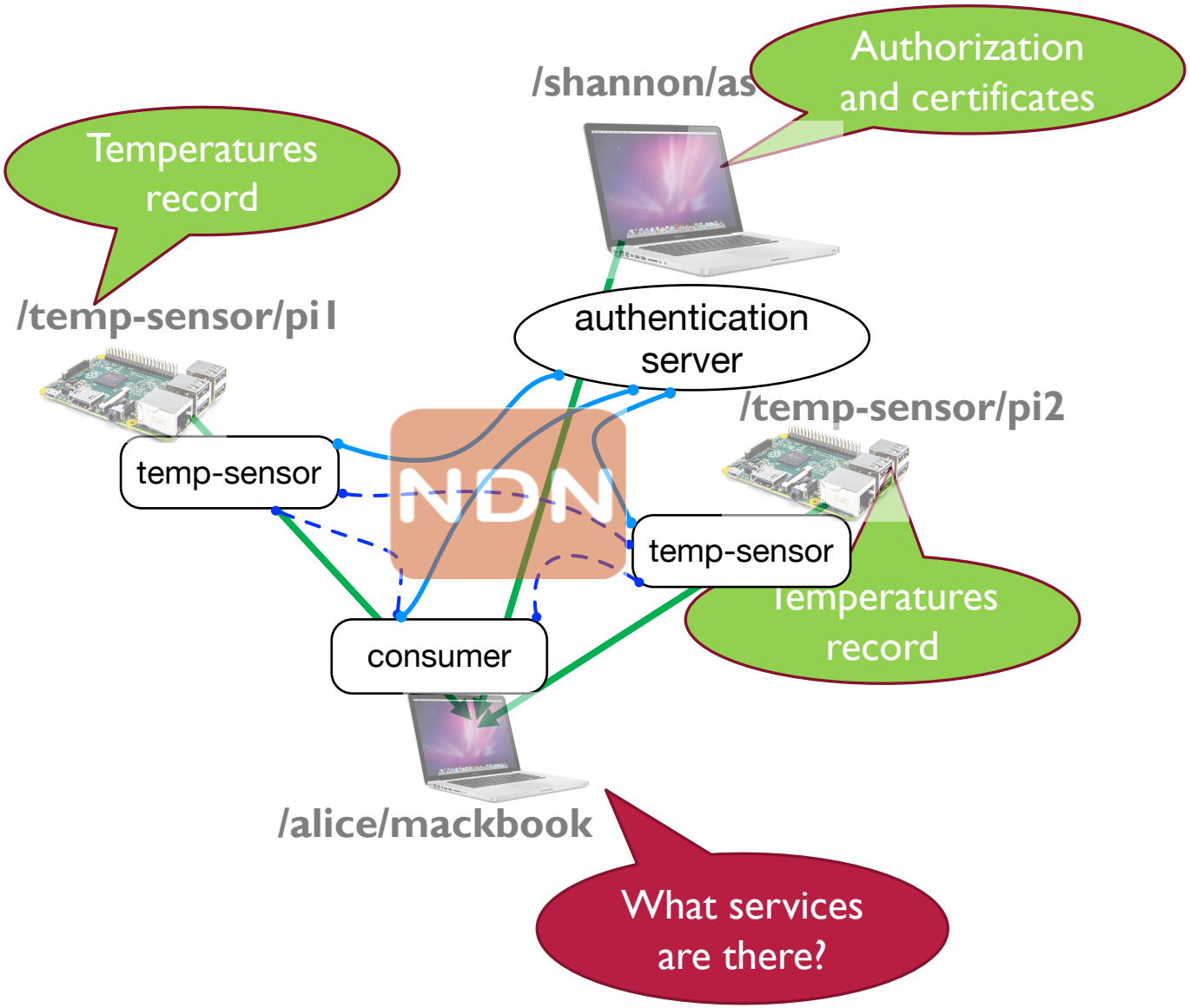


# Auto-discovery: discovery neighbors and surrounding services

- Pre-conditions
  - A device will not be able to discovery unless the Bootstrap is done
  - A device can reach potential targets physically (directly or via the AS)

- Device/Service discovery
  - A device initiates one discovery by probing its neighbors
    - Automatically triggered after bootstrap (proactive)
    - Secured by the ***AS-signed certificate***
    - With its own name embedded to enable reactive discovery
  - Any receiver reply the probe Interest if it's verifiable
    - Reply with its routable name
    - Reply with all accessible services (identified by names)

# Producers reply to trusted discovery requests with names and services



# Q&A

# Open mHealth

UCLA IRL / REMAP

# Open mHealth: Motivation

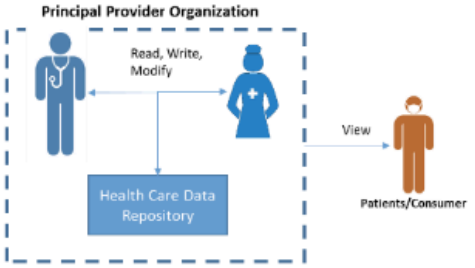



*Figure 1(a)*

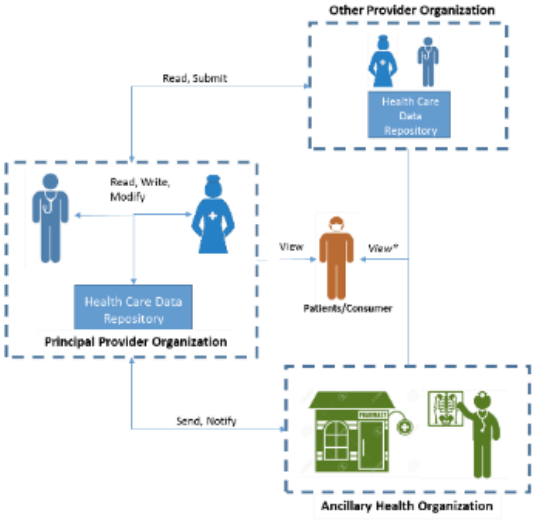



*Figure 1(b)*

**Figure 1:** *The Centralized Health System Model*

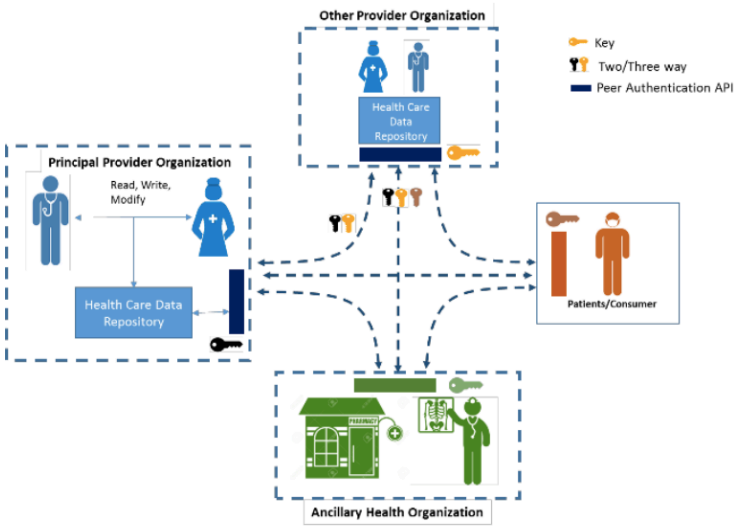



**Figure 2:** *The Peer-to-Peer Health System Model*

Prakash, R. **Adoption of block-chain to enable the scalability and adoption of Accountable Care**. 2016.
http://www.hhs.gov/about/news/2016/08/29/onc-announces-blockchain-challenge-winners.html

# Open mHealth

- Follow-up to participatory sensing work
  - Ecosystem for health data sharing
  - Leverages everyday mobile devices
  - Defines data exchange as the "thin waist"
  - Features user-controlled and privacy-aware data exchange
- Limitations of TCP/IP-based Open mHealth
  - Architecture out-of-sync with the vision of the app
  - (Administratively) centralized approach to data management: A resource server manages data point resources
  - Connection-based security managed by services

[1] D. Estrin and I. Sim. Open mHealth architecture: an engine for health care innovation. Science, 330(6005):759-760, 2010. Also, http://openmhealth.org.

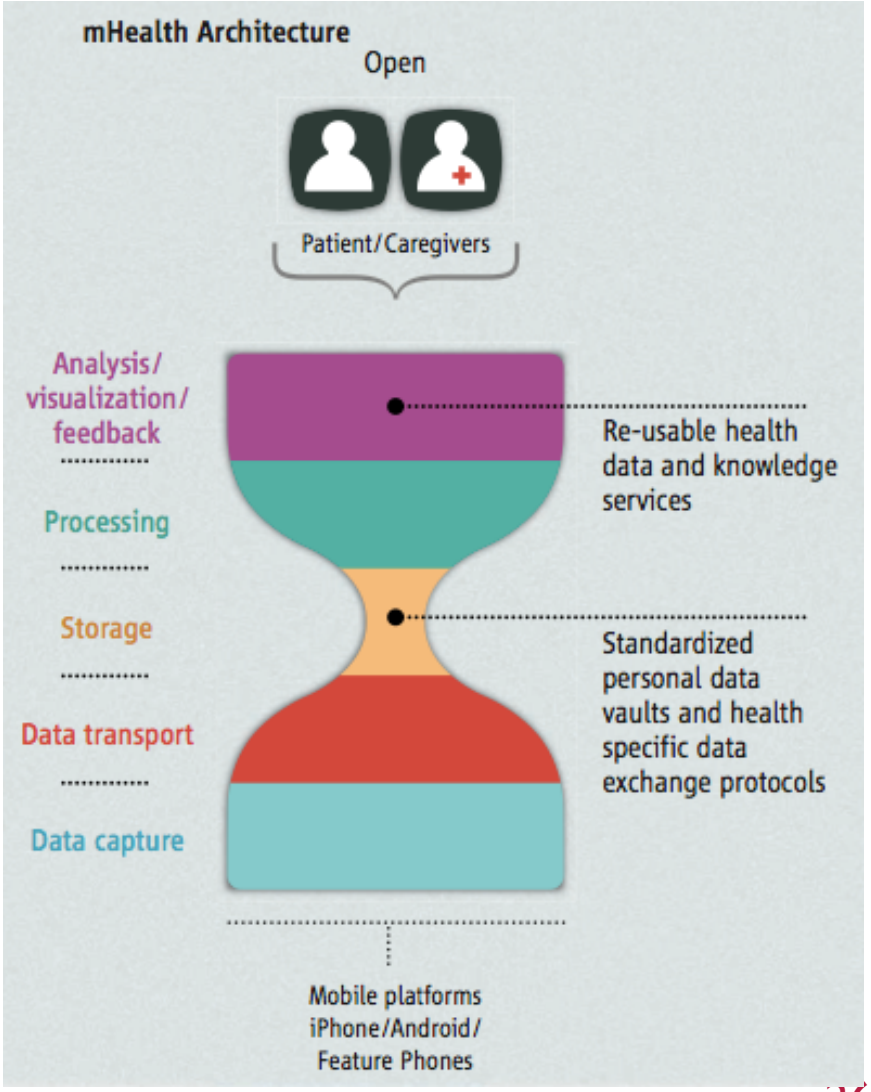
NDN CITE GOES HERE

# Why use NDN for Open mHealth?

NDN and Open mHealth share data exchange as the "thin waist" – one at app level, one at network level.

- Intuition: NDN should be a better fit.

Also, model of securing data close to capture particularly useful for a "ecosystem" with many actors.




Sim & Estrin, 2010

# Objectives

- Limitations of TCP/IP-based Open mHealth
  - Architecture out-of-sync with the vision of the app
  - (Administratively) centralized approach to data management: A resource server manages data point resources
  - Connection-based security managed by services

- Ecosystem for health data sharing
  - Leverages everyday mobile devices
  - Defines data exchange as the "thin waist"
  - Features user-controlled and privacy-aware data exchange

Open mHealth

# Demonstration

- NDNFit, example distributed system of (mobile) data gathering, processing, and visualization using schematized trust and name-based access control.

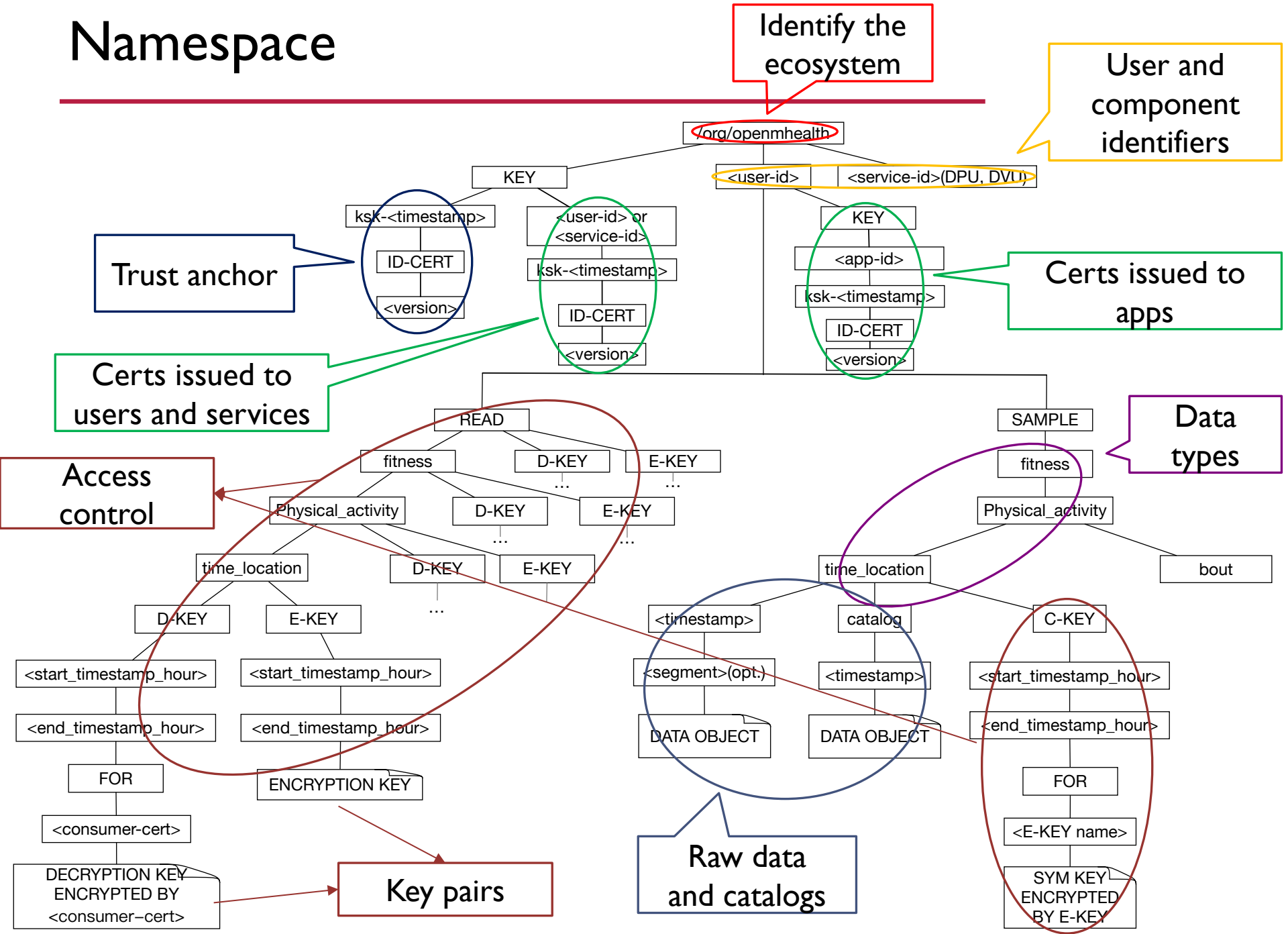
- Running on:
  - Android handset.  (NDN-Android, jNDN)
  - Storage node. (ndn-cxx)
  - Processing node.  (PyNDN2)
  - Visualization applications.  (NDN-JS)
  - Connecting via the NDN testbed.

- We'll just talk about the namespace, then show the demo.

# Namespace Design Goals

- Name data from health application perspective
  - Prefix to identify the data ecosystem
  - Component to identify the data owner
  - Components to classify data into different types
  - Fundamental types include time-series location traces
- Make common data requests using only Interest-Data exchange
- Authenticity of health data is critical: reflect the trust relationships between different components
- Health data is highly private: enable users to control access to their their data without relying on third party services

# Namespace

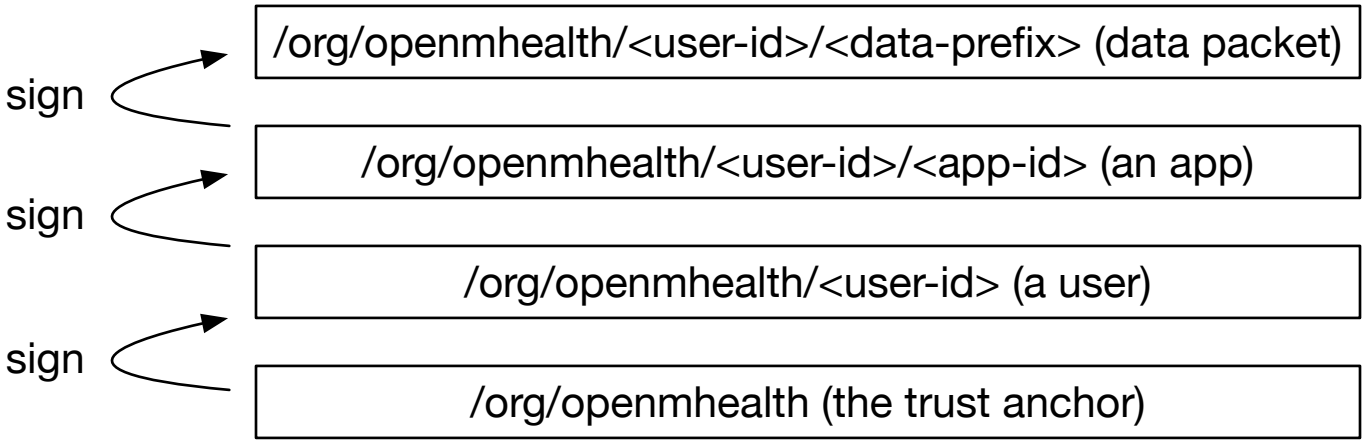

Identify the ecosystem
User and component identifiers
/org/openmhealth
KEY
<user-id>
<service-id>(DPU, DVU)
ksk-<timestamp>
ID-CERT
<version>
<user-id> or <service-id>
ksk-<timestamp>
ID-CERT
<version>
KEY
<app-id>
ksk-<timestamp>
ID-CERT
<version>
Trust anchor
Certs issued to apps
Certs issued to users and services
Data types
Access control
READ
fitness
D-KEY
E-KEY
Physical_activity
D-KEY
E-KEY
time_location
D-KEY
E-KEY
D-KEY
E-KEY
<start_timestamp_hour>
<end_timestamp_hour>
FOR
<consumer-cert>
DECRYPTION KEY ENCRYPTED BY <consumer-cert>
<start_timestamp_hour>
<end_timestamp_hour>
ENCRYPTION KEY
SAMPLE
fitness
Physical_activity
time_location
bout
<timestamp>
(opt.)
DATA OBJECT
catalog
<timestamp>
DATA OBJECT
C-KEY
<start_timestamp_hour>
<end_timestamp_hour>
FOR
<E-KEY name>
SYM KEY ENCRYPTED BY E-KEY
Key pairs
Raw data and catalogs

# Identity and trust model

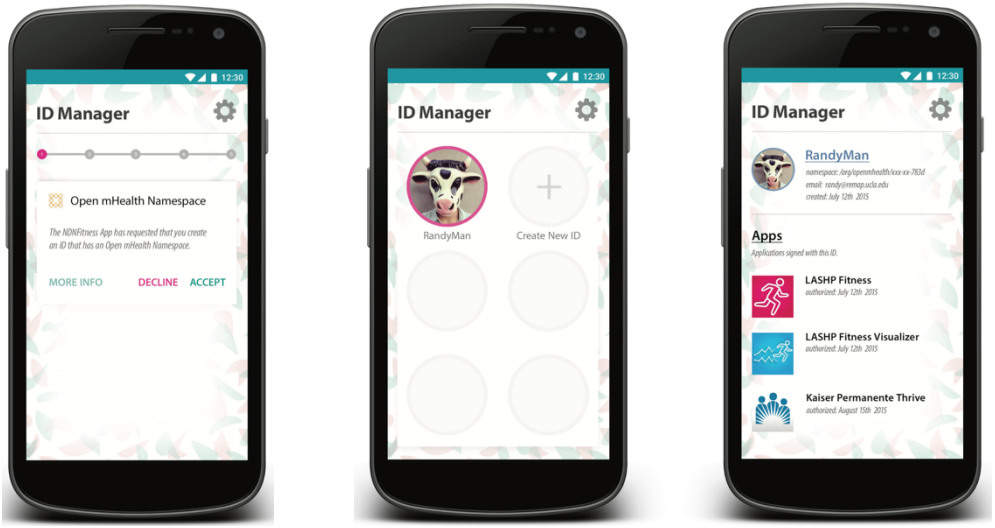
- Design goal: making trust of the data inherent in the data itself, as opposed to tied to service or connection
- Trust model definition
  - Uses *schematized trust*[1]: defines application trust via a set of relationships between data names and key names
- Open mHealth trust model
  - User as the root of trust for her/his own health data.
  - Hierarchical for the user's data; probably more complex for relationships among users.
  - A hierarchical trust model fits well for the pilot NDNFit's context, e.g user -> app-> data.

[1] Y. Yu, A. Afanasyev, D. Clark, V. Jacobson, L. Zhang, et al. Schematizing Trust in Named Data Networking. In Proceedings of the 2nd Conference on Information-Centric Networking. ACM, 2015.

# Trust in NDNFit




Hierarchical trust model for captured data




Mobile "identity manager" app manages user identities, enables their selection by the user.

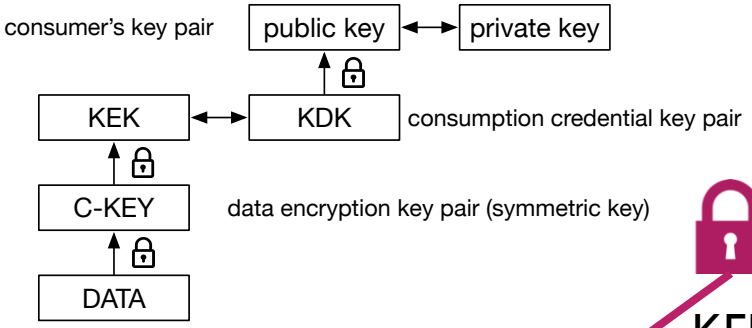
# Access control

- **Problem**: OAuth-style authentication is a significant pain point in current Open mHealth
  - Requires more federation than reasonable or desirable
  - Desire to create processing chains DSU->DPU->DPU->DVU
- Design goals:
  - Achieving access control independent of how data is exchanged
  - Enabling user-defined access control granularity
- Name-based access control (NAC)[1] developed with NDNFit as a use case
  - Data is encrypted at generation time, instead of only when it is transmitted
  - Authorization manager (controlled by the owner) grants components access to owner's data by properly naming, signing, and encrypting keys

[1] Y. Yu, A. Afanasyev, and L. Zhang, "Name-Based Access Control," Named Data Networking Project, Technical Report NDN-0034, October 2015.

# NAC in NDNFit

Consumption credential (KEK/KDK) provides one level of indirection




**Authorization manager**
(on behalf of users)



KEK

KDK

C-KEY

**Capture app**
(data producer)

Data
MAU

C-KEY

Data

KDK

**DVU or DPU**
(data consumer)

Public Key

Private Key

# Q & A