# A Discussion of Industrial Applications and NDN

Kathleen Nichols

NDN Community Meeting 2021

- It’s time for NDN to show its value in solving hard problems. ***Show,*** not tell
- Our NIST colleagues can point us at relevant problem sectors (e.g., NIST SP800-82r2 on Industrial Control Systems Security)
- Operational technologies (OT) have issues that conventional networking doesn’t address
- They also require a different approach from an NDN-based Future Internet

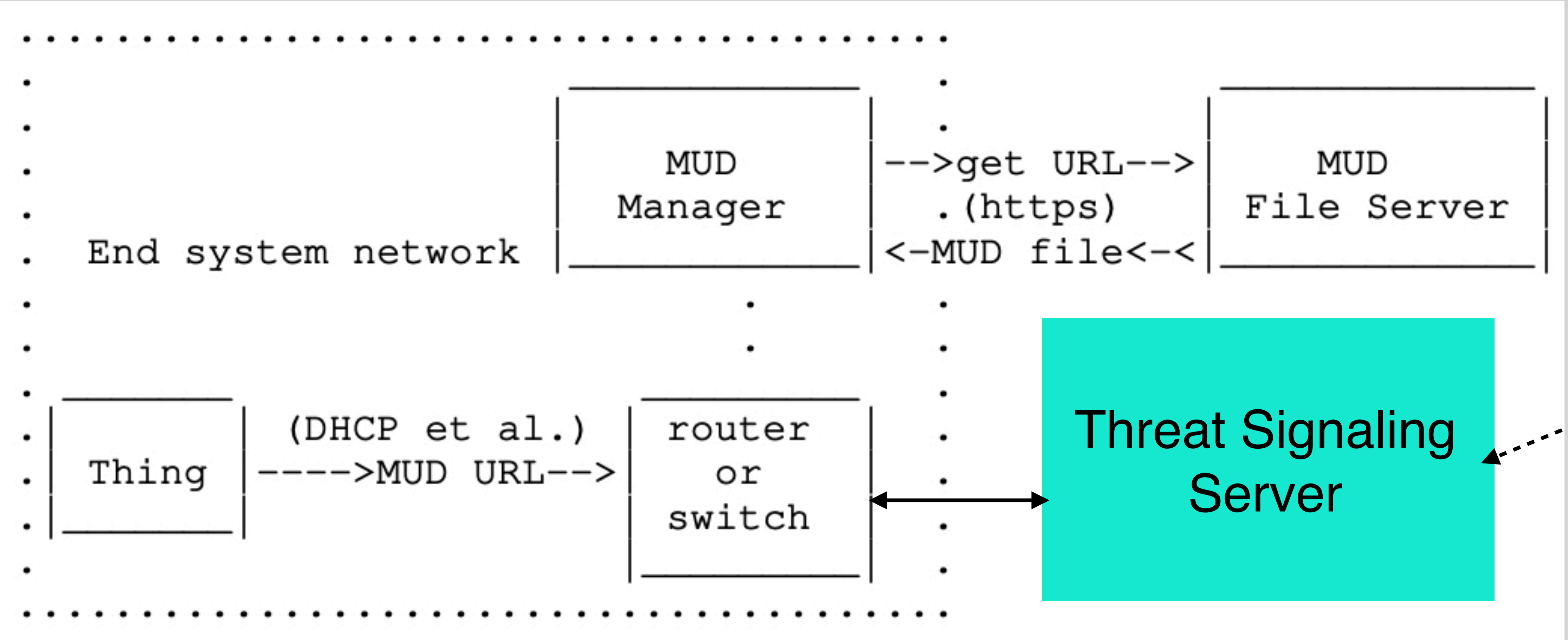
# Definitions used in this discussion

- **Operational Technology**: i-scoop.eu: “Operational technology or OT is a category of computing and communication systems to manage, monitor and control industrial operations with a focus on the physical devices and processes they use.” Wikipedia: “The term has become established to demonstrate the technological and functional differences between traditional IT systems and Industrial Control Systems environment, the so-called ‘IT in the non-carpeted areas’".
- **Conventional Networking (CN)**: the way the Internet uses the IP protocol suite. (NDN runs over IP but is not conventional networking)
- **Things**: following RFC8520, “a group of objects that are specifically not intended to be used for general purpose computing tasks”

- OT networks are increasingly wireless which obviates its traditional security approaches of firewalls, air gaps, and (wired) end-point authentication
- Zero Trust Architectures (“never trust, always verify”) can address this with TCP/TLS encryption but CN’s ad-hoc conversational communications make it hard to establish trust relationships
- Unlike IT networks, OT networks restrict roles and communications intent for their Things
- RFC 8520 notes that a small number of communication patterns follows from those restricted roles and intentions
- The patterns can be expressed as a Manufacturer Usage Description (MUD), similar to Matter, ZigBee, OneData Model descriptions

- CN can only enforce these patterns via externally applied Access Control Lists
- NIST publication SP 1800-15B discusses applications of MUD to small biz and home IoT

RFC8520
Fig 1



NIST Publication
SP1800-15B adds

- MUD is a good way to make use of existing Things and CN but use of DTLS, required by architectures like Matter, blinds deep packet inspection
- NDN could enforce the communication patterns in **every** Thing without relying on external servers and routers as well as take advantage of robust multicast enabled by NDN primitives

# VerSec architecture provides a better way to secure OT

- Key elements: trust schema that is integral to transport and use of sync protocol to distribute signing certificates and group keys as well as application Data

- An important aspect of its trust schema is a declarative approach to specifying arbitrarily fine-grained rules that can be used to check the soundness, construct valid Data Names and enforce the validity of received Data

- Communicating entities **must** publish the public certs of their signing chain before they can communicate

- The trust schema along with its particular root of trust defines a trust zone

A reference implementation (with pub/sub API) at github.com/pollere/DCT

# Data movement is governed by the trust schema

**Trust Schema**

```
devCert: /_net/_role/_roleId/_loc <= netCert
switchCert: devCert & { _role: "switch" }

lightCert: devCert & { _role: "light" }

#pub: /_net/_trgt/_typ/_loc/arg/_ts & { _ts: timestamp() }
lightPub: #pub & { _trgt: "light", arg: "on"|"off" }
sw: lightPub & { _typ: "cmd" } <= switchCert

li: lightPub & { _typ: "sts" } <= lightCert
```

**Enrollment Tools**

***One Switch's Certs***

`/nichols/switch/42/coffeebar`

***One Light's Certs***

`/nichols/light/21/coffeebar`

**Runtime**

***A Switch Publication***

`/nichols/light/cmd/coffeebar/on/1635…`

***A Light Publication***

`/nichols/light/sts/coffeebar/on/1635…`

# Data movement is governed by the trust schema

**Trust Schema**

```
devCert: /_net/_role/_roleId/_loc <= netCert
switchCert: devCert & { _role: "switch" }

lightCert: devCert & { _role: "light" }

#pub: /_net/_trgt/_typ/_loc/arg/_ts & { _ts: timestamp() }
lightPub: #pub & { _trgt: "light", arg: "on"|"off" }
sw: lightPub & { _typ: "cmd" } <= switchCert

li: lightPub & { _typ: "sts" } <= lightCert
```

**Enrollment Tools**

***One Switch’s Certs***

```
/nichols/switch/42/coffeebar
/nichols/switch/42/counter
```

***One Light’s Certs***

```
/nichols/light/21/coffeebar
/nichols/light/21/counter
```

**Runtime**

***A Switch Publication***

```
/nichols/light/cmd/coffeebar/on/1635…
```

***A Light Publication***

```
/nichols/light/sts/coffeebar/on/1635…
```

# some features of approach

- A Thing’s signing id(s) restrict what it can legally say (e.g., publish) and every Thing in the trust zone knows what is and is not legal (deliberately NOT liberal in what is accepted)

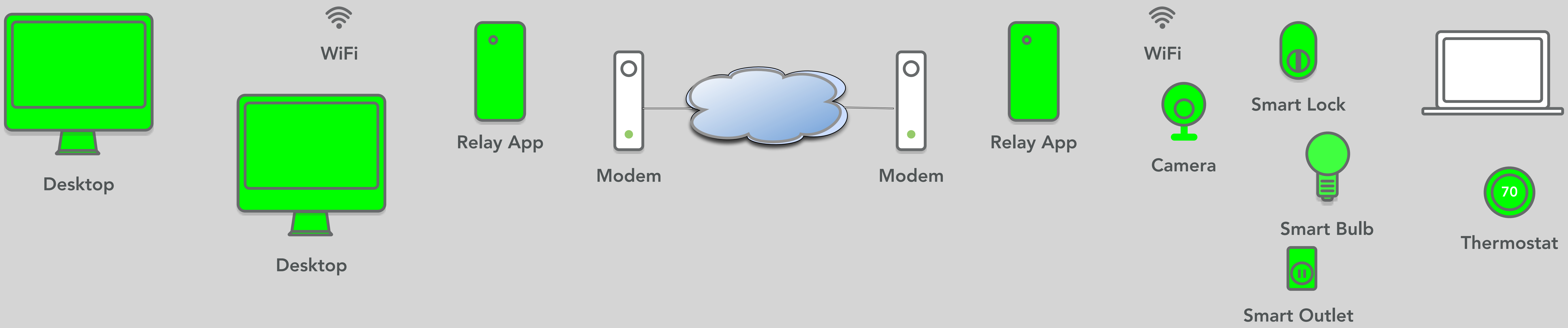
- A Thing can listen for and accept communications from multiple name prefixed Data and its trust schema-enabled transport will ensure these are valid before passing to the application.
    - A light in a building system can listen for Data to its specific location, its room, its floor, all lights.
    - The same light can only report its status and not cause any actions

- Using trust schema based enrollment, communications are enforced at every Thing (or entity) without the need for external servers and routers (no single point of attack)

- Communications exploits the broadcast nature of wireless without compromising security

# Data movement is governed by the trust schema

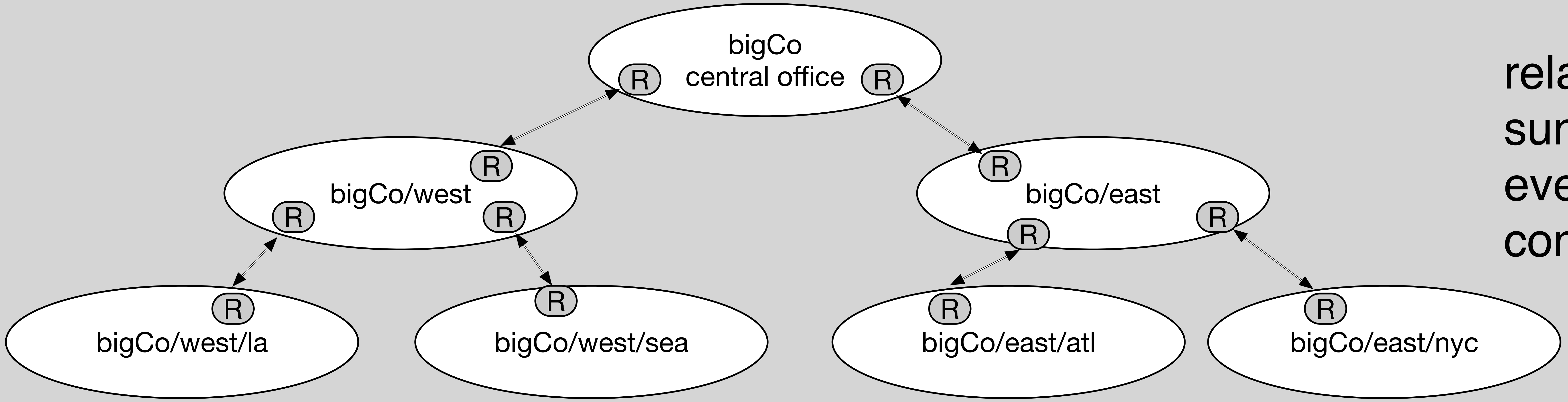
- Everything on the same network segment may not be in the same trust zone (TZ)
- Entities in the same TZ may not be on the same network segment
- Currently in DCT, one TZ per Face but working on “relay” applications to extend a TZ between Faces
- Relay applications can filter publications or just pass them through but envision a “default deny”
- Signing identities are associated with a TZ, but Data encryption keys are Face-specific



# Data movement is governed by the trust schema(s) (warning: half-baked ideas ahead)

- Operational networks can contain a hierarchy of trust zones (TZs) - just use relevant portion of trust schema in any zone. (Offices/apartments on floors in buildings, geographically distributed assets, etc.)
- To connect TZs, some entity must have a Face in each TZ, participate in a sync in each TZ, and have trust rules that govern what can pass between TZs (e.g., a type of “relay”)

Example: bigCo uses a single trust architecture for all its branches. Most of the trust rules cover branches and those publications and identities all start with the three components. The regional offices have different length publications with different trust rules but all can be in the same bigCo TS

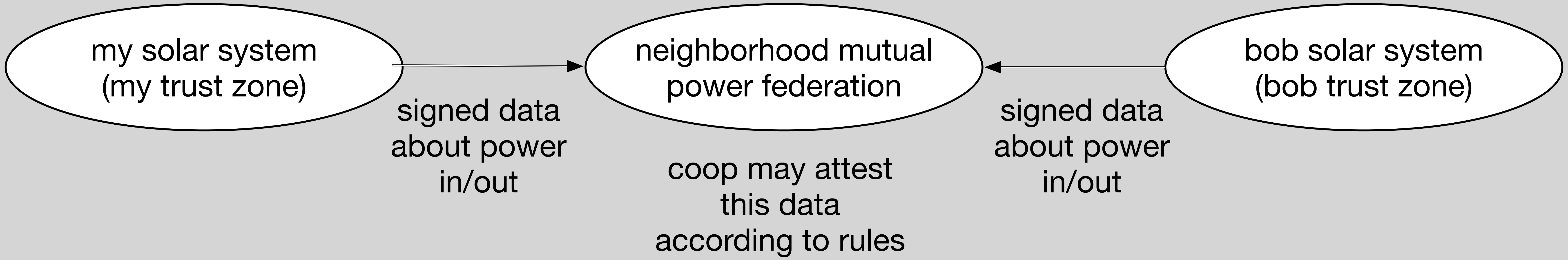


relays can aggregate, filter, summarize or even pass everything since entities can only communicate in their subzone

# Data movement is governed by the trust schema(s) (some thoughts)

- Operational networks could have several ways for totally separate TZs to interact, e.g. limited use signing credentials might be issued for TZ1 for identities that can be validated by TZ2
- For some cases, e.g., technicians to repair your networked home appliances, guests in your home, contractors in a business, short-term and limited capability signing identities might be issued to parties that can prove their identity. Self-sovereign identities could be critical to this approach, allowing a technician to prove they were sent by the company you contacted

Example: a neighborhood has a mutual power co-op


my solar system
(my trust zone)
signed data about power in/out
neighborhood mutual power federation
coop may attest this data according to rules
signed data about power in/out
bob solar system
(bob trust zone)

# Opportunities

- Solving the hard problems in a wide range of special-purpose applications can make NDN immediately relevant commercially
- Applying an NDN architecture to an OT problem should lead to creating new, secure data-centric innovations and new understanding of non-endpoint-centric communications
- Forms-based trust schema creation specific to particular classes of problems. Perhaps a MUD-to-trust schema translator?
- NDN-specific approaches to updating identities and trust schemas that make use of the increasingly available Trusted Platform Module or other Hardware Security Modules
- Extend communication models/APIs ("shims") for trust schema-enabled transports
- Listen/subscribe-only applications that create audit trails for a TZ (record Names, signers, ToD, etc.)
- Hierarchical and federated trust zones
- Security audits?
- Using identifiers/identities: e.g., use of self-sovereign identity in some cases?
- Exploit NDN characteristics to make communications that are robust to failure as well as cyber threats
- And lots more…