Schematized Trust
Design and Application

NDNcomm 2015
September 28, 2015

Alex Afanasyev
University of California, Los Angeles
Overview

NDN architecture mandates signature
• Effectiveness of the mandate depends on the implementation
• If too complex, developers will shortcut
  • “temporarily” disable
  • use non-secure/fake signatures

Need a tool to make security usable
  need automation
Data-Centric Security in NDN

- Data is named and is retrieved using name

- Name and content are bound together with a crypto signature

- Data packet includes a name of the public key to verify the signature
  - Key is also a data packet and retrievable by name

![Diagram of data packet and key retrieval](image)

<table>
<thead>
<tr>
<th>Data packet</th>
<th>Data packet (key)</th>
<th>Data packet (key)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name</td>
<td>Name</td>
</tr>
<tr>
<td>Content</td>
<td>Content</td>
<td>Content</td>
</tr>
<tr>
<td>Signature</td>
<td>Signature</td>
<td>Signature</td>
</tr>
<tr>
<td>KeyLocator</td>
<td>KeyLocator</td>
<td>KeyLocator</td>
</tr>
</tbody>
</table>

Consumers → Producer

- retrieve data
- retrieve key

Keys can be signed and verified by name.
Data Authentication

• To authenticate data, one needs a trust model
  • which keys are authorized to sign which data (trust rules)
  • one or more trusted keys
  • requires crypto properties

• Given trust model, anybody can verify data
  • applications
  • dedicated storage
  • routers

• **Trust model needs to be easily expressible**
  • help consumer to authenticate data
  • help producers to sign data
NDN Insight: Trust can be defined as a set of relationships between data and key names

Hierarchical trust relations

Cross-namespace trust relations
Desired Properties for Trust Policy Definition

• Clear definition of relationship rules
  • Use names and name patterns to define rules
    • data with /some/site prefix can be only signed with /some/site/key/<any-id>
    • keys /some/site/key/<any-id> can be only signed with /another/key/id=5
  • Pre-configured trust anchors to bootstrap trust
    • /another/key/id=5

• Least privilege
  • Limited usage scope
  • Limited time-span

• Re-use of trust models between applications
  • Define, debug, and refine common trust models

• Make security easy to use
Example: Web Blog

- Articles authored and signed by authors
- Authors are given permissions to publish on the blog by administrators
- Administrators are configured by blog configuration or other administrators
Web Blog: Name-Based Trust Relationships

- Articles authored and signed by authors
- Authors are given permissions to publish on the blog by administrators
- Administrators are configured by blog configuration or other administrators
Generalized Rules for Name-Based Trust

Relationship between data and key names

- \texttt{/a/blog/article/food/2015/3} <-> \texttt{/a/blog/author/Alice/KEY/22}
- \texttt{/a/blog/article/drink/2014/9} <-> \texttt{/a/blog/author/Zach/KEY/5}

Generalizing relationship

- \texttt{blogPrefix + “blog” + “article” + category + miscInfo} <->
  - \texttt{blogPrefix + “blog” + “author” + name + “KEY” + keyid}

Use regular-based syntax to capture the relationship

- \texttt{(<>)*<blog><article>[category]<><>} <->
  - \texttt{\1<blog><author>[user]<KEY>[id]}

9/28/15
Web Blog: Trust Schema

Regex-like pattern with grouping (group values accessible as \1, \2, \3 ...)

Data Name

article

(-*)<blog><article><><><>

author

(-*)<blog><author>[user]<KEY>[id]

admin

(-*)<blog><admin>[user]<KEY>[id]

Key Name

author(\1)

admin(\1)

root(\1)

Key

/a/blog/article/food/2015/3

/a/blog/author/Alice/KEY/22

/a/blog/admin/Bob/KEY/5

/a/blog/admin/Carl/KEY/37

Different trust anchor for different blog website

Name or other rule specializations

Regex-like pattern with grouping (group values accessible as \1, \2, \3 ...)

Key Name

root

(-*)<blog><KEY>[id]

Key

/a/blog/KEY/1 (0x30 0x82 ...)

9/28/15

NDNcomm 2015
Trust Rule Processing

```
/a/blog/article/food/2015
/_v=42/_s=1

Content (article)

Signature
/a/blog/author/Yingdi/KEY
```

```
/\a/blog/author/Yingdi/KEY

Content (article)

Signature
/\a/blog/author/Yingdi/KEY
```

```
author

(<>*)<blog><author>[user]<KEY>[id]
```

```
admin(\a)
```

```
/a/blog/article/food/2015/3 ➞  \1 = /a
```

article must be signed with the key with name expanded from author("/a")

- [user] ➞ accepts any user name (auth)
  ➞ generates use name (keygen)

- [id] ➞ accepts any key id (auth)
  ➞ generates unique key id (keygen0)
Trust Rule Processing

```
<blog><author>[user]<KEY>[id]
```

```
<blog><admin>[user]<KEY>[id]
```

```
<blog><admin>[user]<KEY>[id]
```

```
(...)<blog><author>[user]<KEY>[id]
```

```
(...)<blog><admin>[user]<KEY>[id]
```

```
(...)<blog><admin>[user]<KEY>[id]
```

```
(...)<blog><admin>[user]<KEY>[id]
```

```
/author
```

```
/admin(\1)
```

```
/author
```

```
/admin(\1)
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
```

```
/author
Trust Schema Implementation Status

- old schema (ValidatorConf)
- new schema implementation in the upcoming release

NDN-CCL: [http://named-data.net/codebase/platform/ndn-ccl/](http://named-data.net/codebase/platform/ndn-ccl/)
- NDN-CPP, NDN-JS, PyNDN, jNDN

Trust schema powers data and interest authentication in
- NFD: NDN Forwarding
- NLSR: NDN Link State Routing Protocol
- Repo-ng: NDN Data Repository
- ChronoChat: a chat application over NDN
- NDNS: NDN Domain Name System

Works! Even better implementations coming really soon
Making Trust Schema Universal Tool for Trust

Captures data/key name relationships using generalizations and patterns
- formally describes and defines trust model
- enforces trust model in automatic way
  - both authentication and signing paths

Representable in a data packet
- can be retrieved and executed by any NDN entity
- can be (recursively) authenticated using higher-level schemas

Trust schema also defines security design pattern
- regulate the behavior of applications
  - an operating system can define a trust schema to authenticate the trust schema of applications
  - only install and execute apps with authenticated trust schema