Access-Controlled In-Network Processing of Named Data

Claudio Marxer  
<claudio.marxer@unibas.ch>

Christopher Scherb  
<christopher.scherb@unibas.ch>

Christian Tschudin  
<christian.tschudin@unibas.ch>

Computer Networks Group · University of Basel · Switzerland
Problem Statement & Vision of an Ideal World

First Try

I want to compute the length of your last hike. Can you send me the time-location track?
Problem Statement & Vision of an Ideal World

First Try

No, I don’t want you to see the raw data! And I have no time to compute the result...
Problem Statement & Vision of an Ideal World

Another Try

I have time to compute the result...

data consumer

trusted intermediary

data owner
Problem Statement & Vision of an Ideal World

Another Try

@intermediary: Can you compute the result?
@owner: I need the raw data.

data consumer
trusted intermediary
data owner
Problem Statement & Vision of an Ideal World

Another Try
Problem Statement & Vision of an Ideal World

Another Try
Problem Statement & Vision of an Ideal World

Another Try

Generally useful for in-network...
- Conversion (down-scaling, reformatting..)
- Data Fusion (anonymization, statistics..)
In This Paper: Steps Towards a Solution

- Use Named Function Networking (NFN) to produce results
- Use content-based security for data protection
- Encryption of results during production
  → How to synthesize keys in a location-independent manner?
- Introduction of accompanying access control lists (ACLs)
  → How to merge new ACLs for results?
Outline

– Vision and Problem Statement

– Named Function Networking and In-Network Computations

– Solutions: Content Accompanying ACLs, Key Synthesis, ACL Merging

– Comprehensive Example

– Conclusion
NFN on Two Slides: Clients View

**Information Centric Networking:** Distribution of named content (published)

lookup: /alice/NYmarathon/track

**Named Function Networking (NFN):** \(^1\) Generation of named content (on-demand)

lookup: /get/duration( /alice/NYmarathon/track )

---

\(^1\) Sifalakis, M., Kohler, B., Scherb C., Tschudin C. An information centric network for computing the distribution of computations. ACM ICN ‘14.
NFN on Two Slides: Clients View

**Information Centric Networking:** Distribution of named content (published)

lookup: /alice/NYmarathon/track

**Named Function Networking (NFN):** \(^1\) Generation of named content (on-demand)

lookup: /get/duration( /alice/NYmarathon/track )

\[\text{named function}\quad \text{named content}\]

---

\(^1\) Sifalakis, M., Kohler, B., Scherb C., Tschudin C. An information centric network for computing the distribution of computations. ACM ICN '14.
Information Centric Networking: Distribution of named content (published)

lookup: /alice/NYmarathon/track

INTEREST[/alice/NYmarathon/track]

Named Function Networking (NFN): Generation of named content (on-demand)

lookup: /get/duration( /alice/NYmarathon/track )

INTEREST[/get/duration/@x call 2 x |alice|NYmarathon|track/NFN]
INTEREST[/alice/NYmarathon/track/@x call 2 |get|duration x/NFN]

---

1 Sifalakis, M., Kohler, B., Scherb C., Tschudin C. An information centric network for computing the distribution of computations. ACM ICN '14.
Special NFN-capable nodes dissect the interest’s NFN name, compute the result and return it.
Special NFN-capable nodes dissect the interest’s NFN name, compute the result and return it.
Ingredients for a Solution

1. Content-Based Security: “Protection and trust [should] travel with the content itself rather than being a property of the connection over which it travels.”

\[/alice/NYmarathon/track/data\]
\[\text{SymEnc}_k(<\text{time-location track data}>)\]
\[/alice/NYmarathon/track/key/<\text{clientPubKey}>\]
\[\text{AsymEnc}_{\text{clientPubKey}}(k)\]

2. Content Attendant ACLs: Accompany each secured content object with an encrypted ACL.

\[/alice/NYmarathon/track/acl\]
\[\text{SymEnc}_{k-acl}(<\text{access control list}>)\]
\[/alice/NYmarathon/track/aclkey/<\text{clientPubKey}>\]
\[\text{AsymEnc}_{\text{clientPubKey}}(k-acl)\]

Key Synthesis: Encryption Key for Derived Data

**Goal:** Symmetric encryption keys must be reproducible by *any trusted intermediary* at *any time* (and only these).

**Solution:** Key generator determined by all keys of all touched content objects.

**Example:**

\[ k_{user1} \text{ (symmetric key for /user1/NYmarathon/track)} \]
\[ k_{user2} \text{ (symmetric key for /user2/NYmarathon/track)} \]

\[ k_{\text{ranking}} \leftarrow \text{sha256}(\text{concat}(k_{user1}, k_{user2})) \]
ACL Merging: ACL for Derived Data

**Goal:** Compute new ACL compliant with all ACLs of all touched data.

**Solution:** Keep exactly these authorizations which show up in all input ACLs (Intersection).

**Example:**

```
<table>
<thead>
<tr>
<th>/user1/marathon/track</th>
<th>/user2/marathon/track</th>
<th>/get/ranking(..)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;pubKeyKaty&gt;</td>
<td>&lt;pubKeyLeo&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;pubKeyLeo&gt;</td>
<td>&lt;pubKeyLeo&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;pubKeyMaria&gt;</td>
<td></td>
</tr>
</tbody>
</table>
```

→

```
<pubKeyLeo>
```

How all this Works together: Content and ACL Channels

Consumer  Intermediary 1  ...  Owner 1  ...  Owner N

/ranking/data(/owner1/track, ..., /ownerN/track)

/owner1/track/data

\[\text{SymEnc}_{\text{track1}}(<\text{track1}>)\]

/owner1/track/key/<pubKey-I>

\[\text{AsymEnc}_{\text{pubKey-I}}(<\text{k-track1}>)\]

/ownerN/track/data

\[\text{SymEnc}_{\text{trackN}}(<\text{trackN}>)\]

/ownerN/track/key/<pubKey-I>

\[\text{AsymEnc}_{\text{pubKey-I}}(<\text{k-trackN}>)\]
How all this Works together: Content and ACL Channels

Consumer

Intermediary1...

Owner 1 ...

Owner N

/ranking/data(/owner1/track, ..., /ownerN/track)

SymEnc_{result}(<result>)

/owner1/track/data

SymEnc_{<track1>}(<track1>)

AsymEnc_{pubKey-I}(<k-track1>)

/owner1/track/key/<pubKey-I>

/ownerN/track/data

SymEnc_{<trackN>}(<trackN>)

AsymEnc_{pubKey-I}(<k-trackN>)

/ownerN/track/key/<pubKey-I>

Content Production & Key Synthesis
How all this Works together: Content and ACL Channels

Content Production & Key Synthesis

Consumer   Intermediary1…   Owner 1  …  Owner N

/ranking/data(/owner1/track, ..., /ownerN/track)

SymEnc_{k-result}(<result>)

/ranking/key(/owner1/track, ..., <pubKey-C>)

SymEnc_{k-track1}(<track1>)

AsymEnc_{pubKey-I}(<k-track1>)

/owner1/track/data

SymEnc_{k-track2}(<track2>)

AsymEnc_{pubKey-I}(<k-track2>)

/owner2/track/data

SymEnc_{k-trackN}(<trackN>)

AsymEnc_{pubKey-I}(<k-trackN>)

/ownerN/track/data

SymEnc_{k-track1}(<track1>)

AsymEnc_{pubKey-I}(<k-track1>)

/owner1/track/acl

SymEnc_{k-acl1}(<acl1>)

AsymEnc_{pubKey-I}(<k-acl1>)

/owner2/track/acl

SymEnc_{k-acl2}(<acl2>)

AsymEnc_{pubKey-I}(<k-acl2>)

/ownerN/track/acl

SymEnc_{k-aclN}(<aclN>)

AsymEnc_{pubKey-I}(<k-aclN>)

/ownerN/track/key/<pubKey-I>

AsymEnc_{pubKey-I}(<k-track1>)
How all this Works together: Content and ACL Channels

Consumer

Intermediary 1 ...

Owner 1 ... Owner N

Content Production & Key Synthesis

ACL Merging
How all this Works together: Content and ACL Channels

1. Content Production & Key Synthesis
   - Consumer
     - \(/\text{ranking/data}(\text{owner1}/\text{track}, \ldots, \text{ownerN}/\text{track})\)
     - SymEnc_{\text{result}}(\text{result})
   - Intermediary1
     - /\text{owner1}/\text{track}/data
     - AsymEnc_{\text{pubKey-I}}(\text{k-track1})
   - Owner 1
     - /\text{owner1}/\text{track}/key/\text{pubKey-I}
     - AsymEnc_{\text{pubKey-I}}(\text{k-track1})
   - Owner N
     - /\text{ownerN}/\text{track}/data
     - SymEnc_{\text{result}}(\text{result})
   - Intermediary1
     - /\text{ownerN}/\text{track}/data
     - AsymEnc_{\text{pubKey-I}}(\text{k-trackN})
   - Owner 1
     - /\text{ownerN}/\text{track}/key/\text{pubKey-I}
     - AsymEnc_{\text{pubKey-I}}(\text{k-trackN})

2. ACL Merging
   - Consumer
     - /\text{ranking/key}(\text{owner1}/\text{track}, \ldots, \text{<pubKey-C>})
     - AsymEnc_{\text{pubKey-I}}(\text{k-result})
   - Intermediary1
     - /\text{owner1}/\text{track}/\text{acl}
     - SymEnc_{\text{result}}(\text{acl1})
   - Owner 1
     - /\text{owner1}/\text{track}/\text{aclkey}/\text{pubKey-I}
     - AsymEnc_{\text{pubKey-I}}(\text{k-acl1})
   - Owner N
     - /\text{ownerN}/\text{track}/\text{acl}
     - SymEnc_{\text{result}}(\text{aclN})
     - Intermediary1
     - /\text{ownerN}/\text{track}/\text{aclkey}/\text{pubKey-I}
     - AsymEnc_{\text{pubKey-I}}(\text{k-aclN})
How all this Works together: Content and ACL Channels

1. Content Production & Key Synthesis
   - Content Channel
     - SymEnc\(_{result}\)(<result>)
   - Key Channel
     - AsymEnc\(_{<k-result>}\)

2. ACL Merging
   - ACL Channel
     - AsymEnc\(_{<k-track>}\)(<track>)

(Further propagation of merged ACL, if consumer acts as another intermediary)
Conclusion

– Content-Attendant ACLs enable to perform consumer-driven distributed computation chaining for access controlled named data.

– Intermediaries run a “consume-produce-publish loop”
  → Distributed: No central entity which is responsible for an entire computation chain.
  → Data-centric rather than location-oriented.

– Balance of data owners privacy demands and consumers needs:
  A consumer is able shift any computation into the network while the privacy of the owner is always fully respected.
Q & A
Additional Slides
Who Trusts Who?

Trust Relationships:

- Owner → Intermediates
  - Give read access to authorized parties only
  - Deliver correct results

- Consumer → Intermediates
  - Deliver correct results

- Consumer → Function Provider
  - Functions produce results as documented
Access Control Model

**Paper:** A consumer can outsource all computations to the network (but not more).

**Vision:** For some applications it is reasonable to say: A client should be allowed to read a certain computation result (e.g. statistical evaluation or anonymization) but not the primary data itself (slide 2).

**Solution: Attribute-Based Access Control?**

- ACLs state not just “*read?*” but also attributes such as “*absoluteCoordinates?*” or “*absoluteTimestamps?*”
- Functions state attributes of their result and implement according key sync.
- Example: `OriginFilter(...)` shifts starting point of time-location track to the origin of the coordinate system. Thus, output hides original coordinates (`absoluteCoordinates=false`) which means that the result might be read by more consumers than the unfiltered track.
Performance Considerations

+ Re-usage of (intermediate) results due to caching. On cache hit:
  + Less computation effort
  + Faster response times

+ Reduced network load: Execution can take place close to the storage location. Transferred result might be significantly smaller than input data.

+ Network-wide reusability of code (named functions)
  - Meta-data and key synchronization: At most four-fold increase in the number of interests.
  - Additional computation effort (ACL merging, key synthesis)