Named Function Networking
Service chaining, big picture, division of labor boundaries

IRTF ICNRG interim meeting, Boston, Jan 13+14, 2015

Christian Tschudin, University of Basel
Part I – Service chaining (Dirk Kutscher)

Part II – Gentle/general introduction

Part III – NFN in one sentence

plus

* some rants about layers and engines
* packet format implications
Part II

Gentle/general introduction to NFN
Named-Data knowledge gap – Network intelligence

location knowledge (URL), replica/version control, replica location (CND), security (https, certificates)

"pipe abstraction"

Raising the semantic level of the network API means: filling the gap

New answers necessary, different answers possible:
– redesign transport fabric to handle names
– new name-to-FIB mapping
– “name space operations” – from publish to exploration, mgmt and removal
From Named-Data to Named-Functions

• Raw data in abundance, but clients want cooked data . . . for which there are arbitrarily many recipes

• Examples:
  /downScale( /this/video )
  /getAverage( /sunShineHours/in/CA, 2014 )
  /geoFence( /my/heart/rate, /my/gps/location, 10ft )

• The goal of Named Function Networking (NFN):
  – clients *name the desired result*, server-agnostically
  – network is in charge of finding execution places
  – network optimizes execution graph, caches the results
### From Name-Lookup to Expression-Reduction

<table>
<thead>
<tr>
<th>Realm</th>
<th>Instances</th>
<th>Network Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Named Data</strong></td>
<td>“classic” ICN, key–value store, DNS</td>
<td>“name resolution” (= lookup)</td>
</tr>
<tr>
<td><em>(access to data)</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Named Functions</strong></td>
<td>“new” ICN</td>
<td>“expression resolution” (= processing)</td>
</tr>
<tr>
<td><em>(access to results)</em></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
How: a) Locate data, fct and exec place, b) Run, c) Collect

REMOTE EVAL beats “download and process locally”: find a server close to the DB!

Network does not execute: NFN only orchestrates the computation by juggling around names and triggering exec, later returning the collected result.

Christian Tschudin, U of Basel
“Routing”: Named-Data as a special NFN Case

Note: execution can, but does not have to be done in-network

D = data bits
F = byte code, binaries
@ = execution site

Christian Tschudin, U of Basel
Results as Names as Programs

“name the result . . . the network recognizes named results” – how?

- Lambda-expressions: most general approach, probably not your cup of tea
“name the result ... the network recognizes named results”  – how?

- Lambda-expressions: most general approach, probably not your cup of tea

- Other ways of expressing results. NFN enables choice:
  - /iFeelLucky( Hotel Sonesta Boston )
  - /yahoo( Hotel Sonesta Boston )

- NFN for network tasks:
  - NFV /kvsLookup( /DPI( /loadBalance( /name )))
  - IoT /mapReduce( /sensors( /my/house, temp, 2014 ), /avg)
  - Mgmt /keepInCS = /mapReduce( /topTenNames( /my/neighbor/CS ), /sort)

- NFN also for CCN/NDN semantics variety:
  - /rightMostChild( /a/node/name )
  - /exists( /a/node/name )
Results as Names as Programs

The search of the good “expression language” . . .

• exact match
• selective match
• . . .
• DB query languages
• Datalog
• intentional naming
• Prolog, $\lambda$ expressions

. . . just started!
Part III

NFN in one sentence

– and some rants
– implications for packet formats, layering
A purposefully minimal definition:

Named Function Networking is an ICN style where a request carries at least two names in order to be satisfied.
Examples where more than one name is needed

- Application – `compute(/name/of/fct, /name/of/arg)`
Examples where more than one name is needed

- **Application** – `compute(/name/of/fct, /name/of/arg)`
- **Quantifiers** – `retrieveAnyOf(/node/prefix, /pattern/star)`
  - also known as selectors (NDN)
  - also known as restrictions (CCNx’ ObjHash, KeyID)
- **Validation** (based on references to keys = names)

In this “NFN interpretation” of CCN/NDN:

*Where are the fct names?*

- sometimes not choosable: functions are “named” in the specs
- for “real NFN”: packet fmt to provide hooks for run-time-nameable
Rant slide: Stupid networks, redux?

“Extremist rant” on the ICNRG email list: Extreme contexts (high speed networking and the IoT) dominate the forwarding semantics discussion.

“The rise of stupid networks”
Isenberg’s meme from 1997 – resurging?

Contrarian view, from CES’2015 slides of Yu-Ting Yu, Qualcomm, Mario Gerla et al.:

“ICNs are network architectures allowing the network to be aware of content semantics.”

Concern that “in-network processing” becomes off-topic, is pushed to edge or app
A gradual spectrum – complementary

↑ stupid networks    ↑ CCN    ← “domesticated NFN” →    \( \lambda \) calculus ↑

Division of labor, catenet model:

- High speed or IoT forwarding substrate is a base *level*, not a base *layer*:
  - envisage nodes with different “semantic height”

- Catenet model: heterogeneous forwarding domains, routers
  - Interest might hit a CS or not
  - Interest might hit a NFN-enabled node or not, …
A gradual spectrum – implications for packet fmts

“Role-based architecture” (Braden/Faber/Handly, Hotnets 2002):
provides header space for more than one “network function”

- Roles in CCN (would be nice if NDN could add this, too):
  - **Interest name** – is for the forwarder, partly the CS
  - **Interest payload and opt header space** – for the rest of us
- Not all role parameters are generated at the edge:
  - in-network generated hints, routing diversion, name rewriting
- Organize the Interest payload and/or opt header space
A gradual spectrum – implications for “layering”

CCN’s `exactCSlookup/PITcheck/LPMfwd` is a function, NDN’s `LpmCSlookup/PITcheck/LPMfwd` is yet another function …

- Once you make these “namable”, NFN becomes the core
- ICN as assembly of several “engines”:
  - name space/expr engine (CCN style, pub/sub, Datalog, \( \lambda \) expr)
  - forwarder engine, policy engine, charging engine