CCNx 1.0  Changes from 0.x

Computer Science Laboratory
Networking & Distributed Systems

Ignacio.Solis@parc.com

IETF 90 - July 2014
CCNx 1.0 - changes from 0.x

Static header, optional header, message
TLV packet format (vs ccnb)
Validation algo (vs Signature)
Exact-match (no prefix match, selectors)
Restrictions (vs selectors/names)
Hop-limit (no nonces)
Payload in interest (vs long names)
Label-based names (no command markers)
Changes - details
Packet Format

CCN 0.x

Message

CCN 1.x

Static Header

Optional Headers

Message
Packet Format - Why change

Static header
- enables fast parsing
- contains common needs
- allows versioning

Optional headers
- allows network elements to add/modify information
Packet Encoding

**CCN 0.x**

ccnb

“Custom binary encoding format for XML to meet specific needs of CCNx”

<block><block><block>

**CCN 1.x**

tlv

type-length-value (tag-length-value)

2 byte T
2 byte L
L byte V
Packet Encoding - Why change

ccnb
flexible but complicated
relies on meta-structure
bit efficient

tlv
easy to parse
well understood
parse efficient
Message Organization

CCN 0.x

Content Object

Signature
Name
Signed Info + Meta
Payload

CCN 1.x

Content Object / Interest

Name
Meta
Payload
Validation
Message Organization - Why change

Name comes first
  fast parsing

Separate validation from metadata at the end
  modular security

Unified packet format
  simplified, fast parsing
Matching (exact)

CCN 0.x

/parc/ccn.zip
/parc/ccn.zip
/parc/ccn.zip/v2/s3
/parc/ccn.zip/v2/s4
/parc/ccn.zip/v7/s6
/parc/ccn.zip/v100/s1
/parc/ccn.zip/meta/v1/s8
/parc/ccn.zip/discussion
/parc/ccn.zip/.acl/by_group/owner/key/v1/s9

CCN 1.x

/parc/ccn.zip
Matching (no selectors)

CCN 0.x

Interest:
name = /parc/ccn.zip
minSuffixComponents=x
maxSuffixComponents=y
exclude=xxx,xxx,xxx,…
childSelector=Left/Right

CCN 1.x

Interest:
name = /parc/ccn.zip
Matching - Why change

Exact match is deterministic
   You get what you ask for

Efficient match
   Fast forwarding on single match

No rummaging of caches / traffic
   Better privacy
Matching (restrictions)

**CCN 0.x**

Interest:
name = /parc/ccn.zip
pubKeyDigest=xxx

Interest:
name = /parc/ccn.zip/abcd
minSuffixComponents=0
maxSuffixComponents=0

**CCN 1.x**

Interest:
name = /parc/ccn.zip
keyIdRestriction=xxx

Interest:
name = /parc/ccn.zip
contentObjectHash=abcd
Matching (restrictions) - Why change

No ‘real’ change
  Functionally the same

Explicit contentObjectHash matching
  Simpler matching (not intermingled)
Loop halting

CCN 0.x

Interest:
name = /parc/ccn.zip
nonce = 1234

PIT
/parc/ccn.zip : 1234

CCN 1.x

Interest:
name = /parc/ccn.zip
hop-limit = 16

PIT
/parc/ccn.zip
Loop halting - Why change

Less overhead
- No need to carry large nonce in packet
- No need to keep nonces at router (large at fast speed)

PIT takes care of most loops
- PIT halts loops, hop-limit is a stop-gap

Nonces breaks aggregation
- Interests can’t be aggregated
  (if node can treat same nonce interests as equal)
Interest Payload

**CCN 0.x**

Interest:
name = /store/cart/abc...
...defg...
...<1k component>...
...xyz/checkout

**CCN 1.x**

Interest:
name = /store/cart/...
id=1234/checkout

payload = abc...
...defg...
...<1k component>...
...xyz
Interest Payload - Why change

- Less processing at routers
  - No need to parse long names all the time

- Less storage at routers
  - No need to keep large names at routers

- Less traffic overhead
  - No need to carry copy of state back in the response
Label-based names

**CCN 0.x**

/parc/ccn.zip/…
%C1.M.K%01%02…/
%FD%04%62…/
%00%02/

**CCN 1.x**

/parc/ccn.zip/…
app<key>=1234/
v=12/
c=2/
Label-based names - Why change

No aliasing
  Defining structure eliminates aliasing

Cleaner representation
  More human readable

More powerful
  Structure allows network elements to make choices
Summary

Static header, optional header, message
TLV packet format (vs ccnb)
Validation algo (vs Signature)
Exact-match (no prefix match, selectors)
Restrictions (vs selectors/names)
Hop-limit (no nonces)
Payload in interest (vs long names)
Label-based names (no command markers)
Other changes - Summary 1

Fragmentation
Signature verification
Manifest as core object
Time is now in milliseconds
No AnswerOriginKind, Scope
Control messages
Reduced timers
Next hop (vs Face)
Other changes - Summary 2

Storage (vs Repo)
Separate discovery, chunk, versioning
Sync as protocol+service
Content Store is optional
Content Store has matching requirements
Flow control (vs static pipelining)
Other changes - Summary 3

Software (all C)
Coding style
Modular transport
Modular API
Test driven
Thank you

Nacho (Ignacio) Solis
ignacio.solis@parc.com
http://www.ccnx.org/
http://www.parc.com/ccn