CCNx 1.0 Overview: CCN 101

Computer Science Laboratory
Networking & Distributed Systems

March 2014

parc
A Xerox Company

CCN - 101

CCN - Motivation

Network Evolution

1876

Tom Watson Please

Network Evolution

1876

Tom Watson Please

1946

1-650-812-4472

Network Evolution

1876

Tom Watson Please

1946

1-650-812-4472

1976

170.124.100.133

Network Evolution

1876

Tom Watson Please

1946

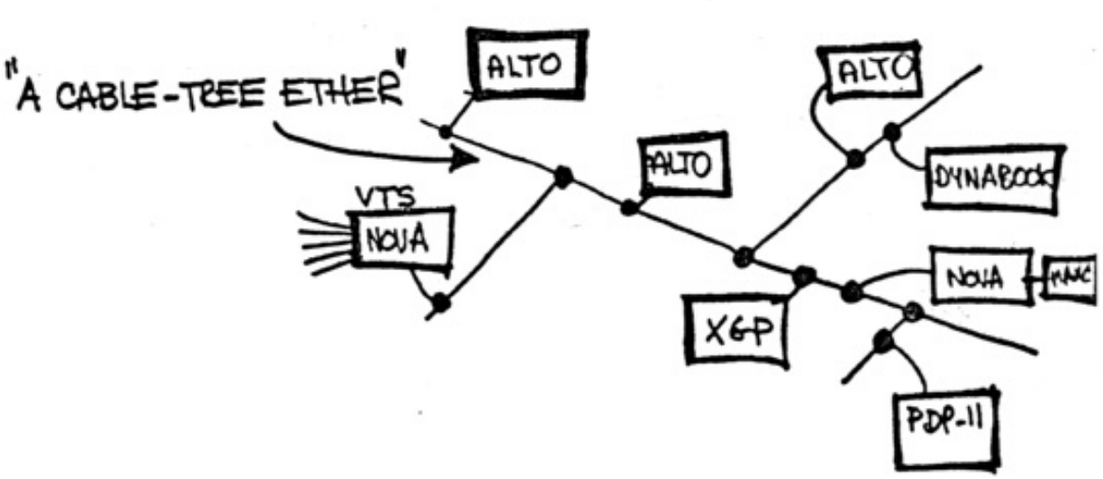
1-650-812-4472

1976

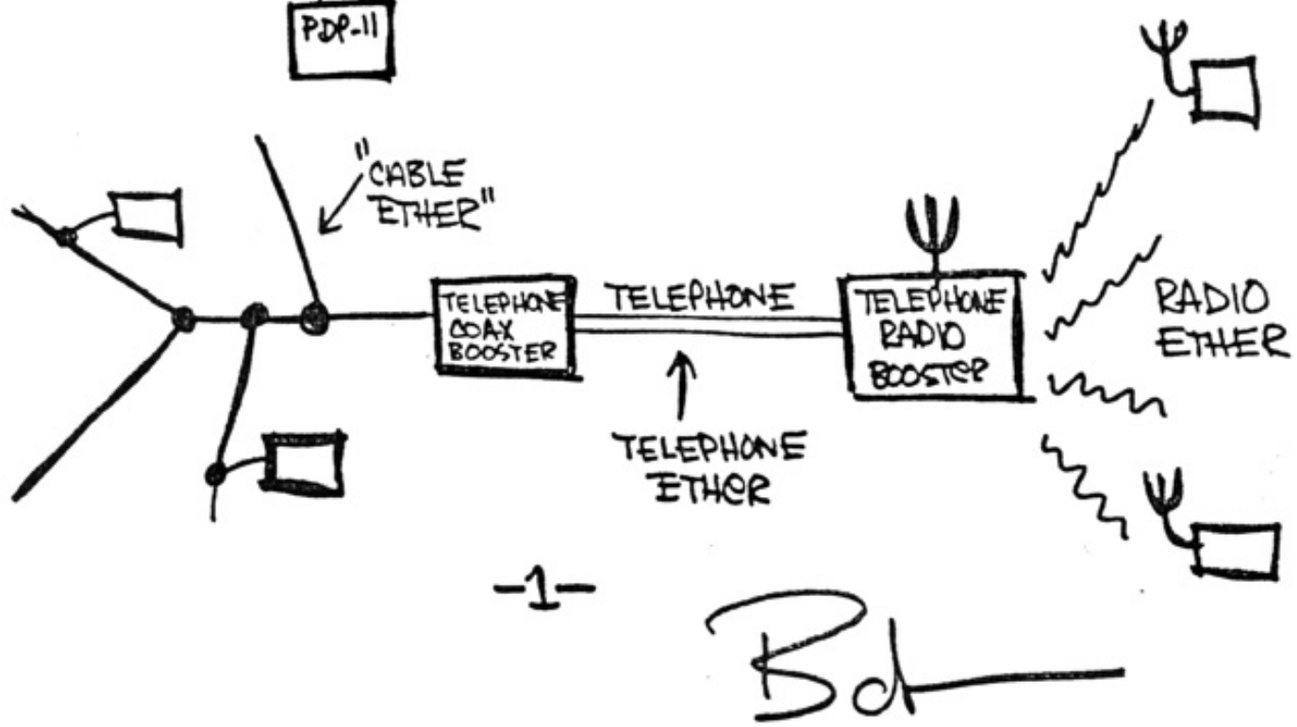
170.124.100.133

2076?

2001:1868:ad01:1::33



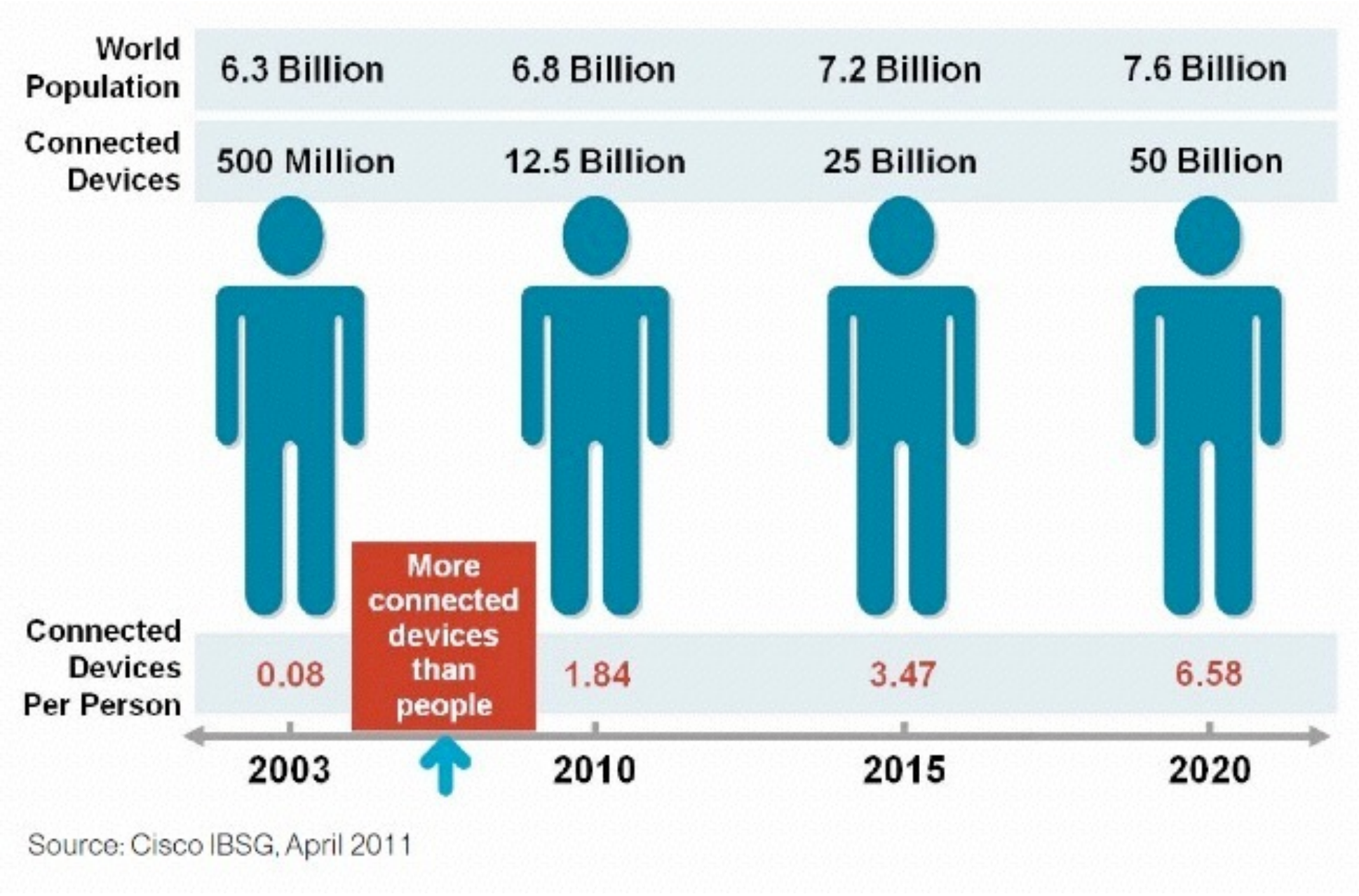
ETHER!



-1-

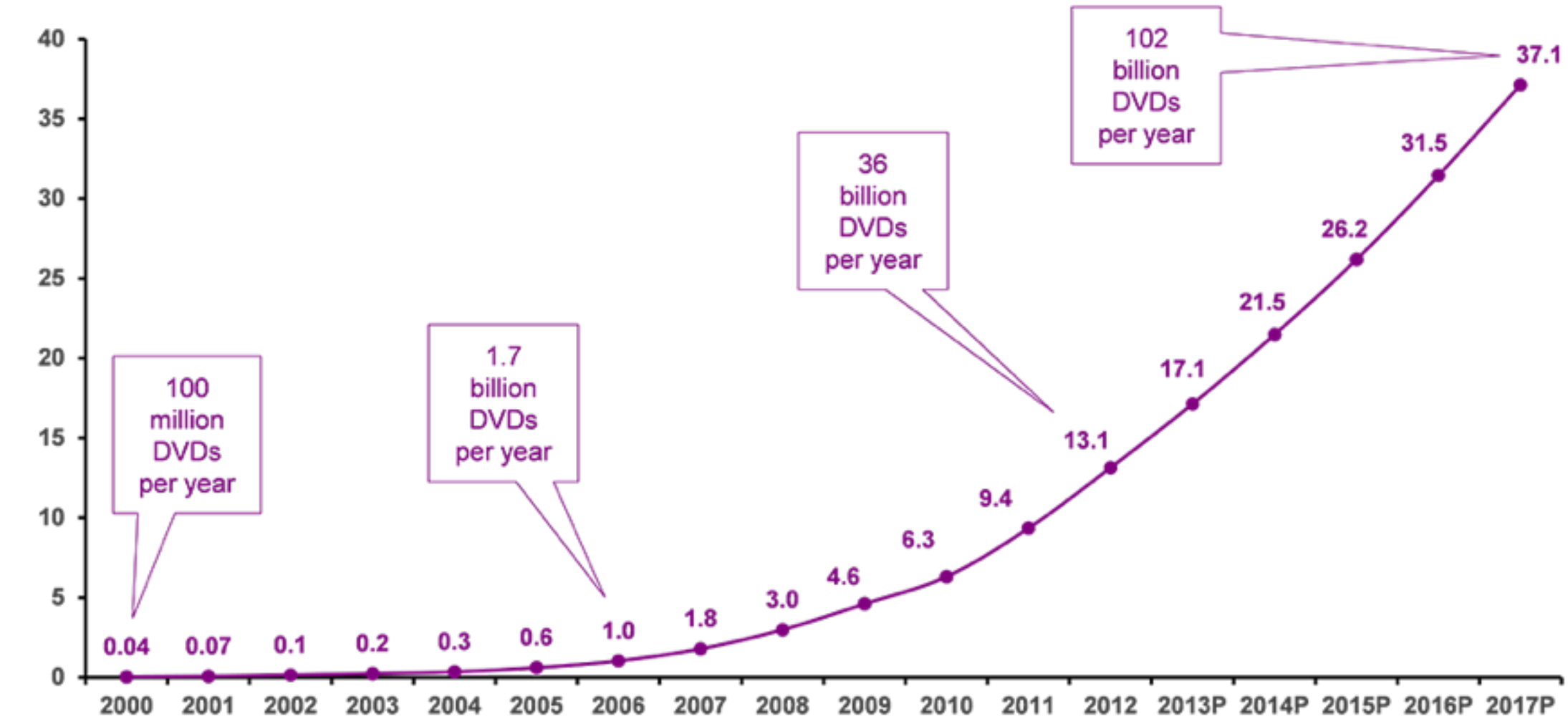
Metcalfe's Law - "***the value of a network is proportional to the square of the connected users of the network***"

Internet Growth - Devices/person



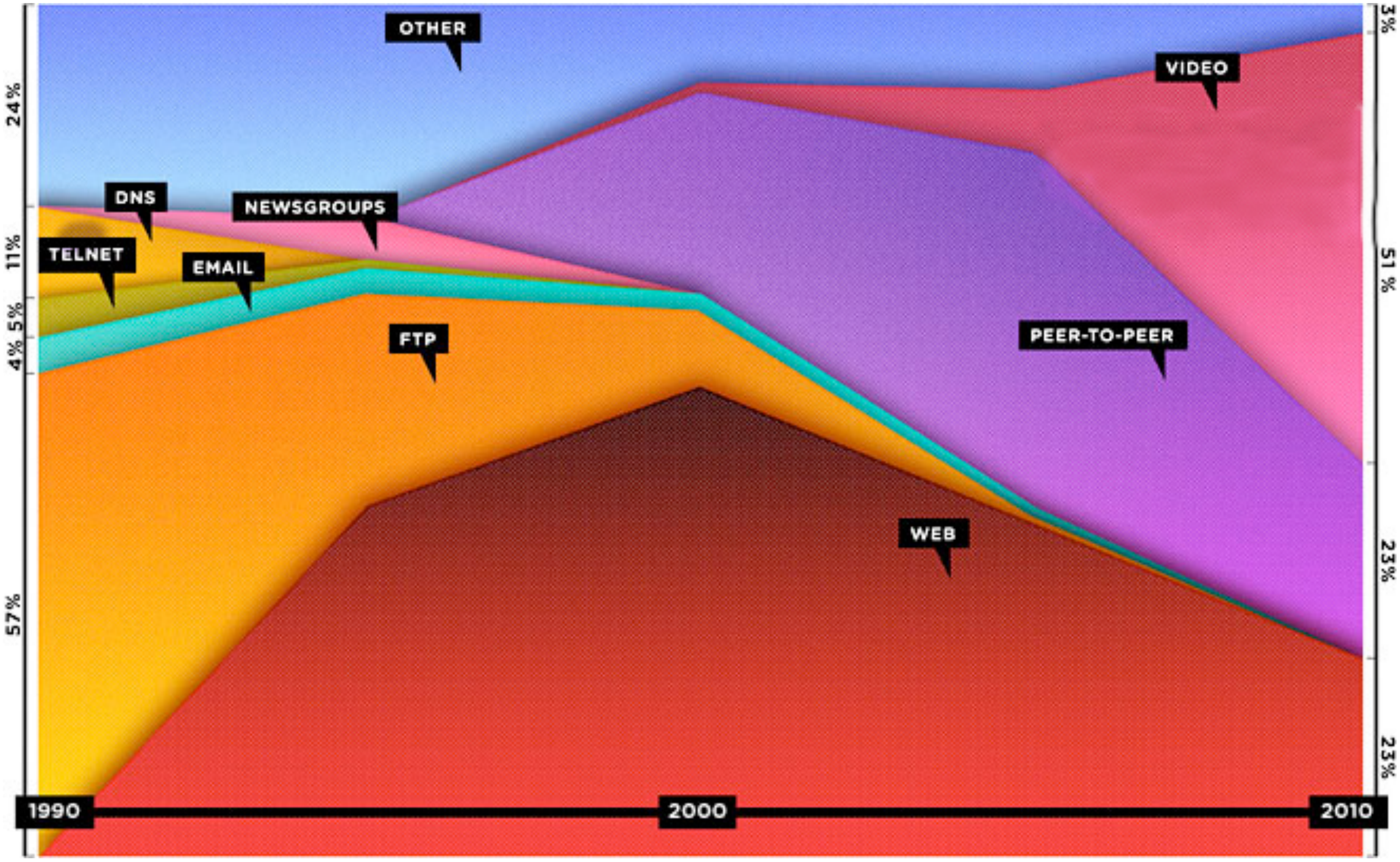
Internet Growth - Exabytes/month

Estimated U.S. Internet Protocol Traffic, 2000-2017 (Exabytes per Month)



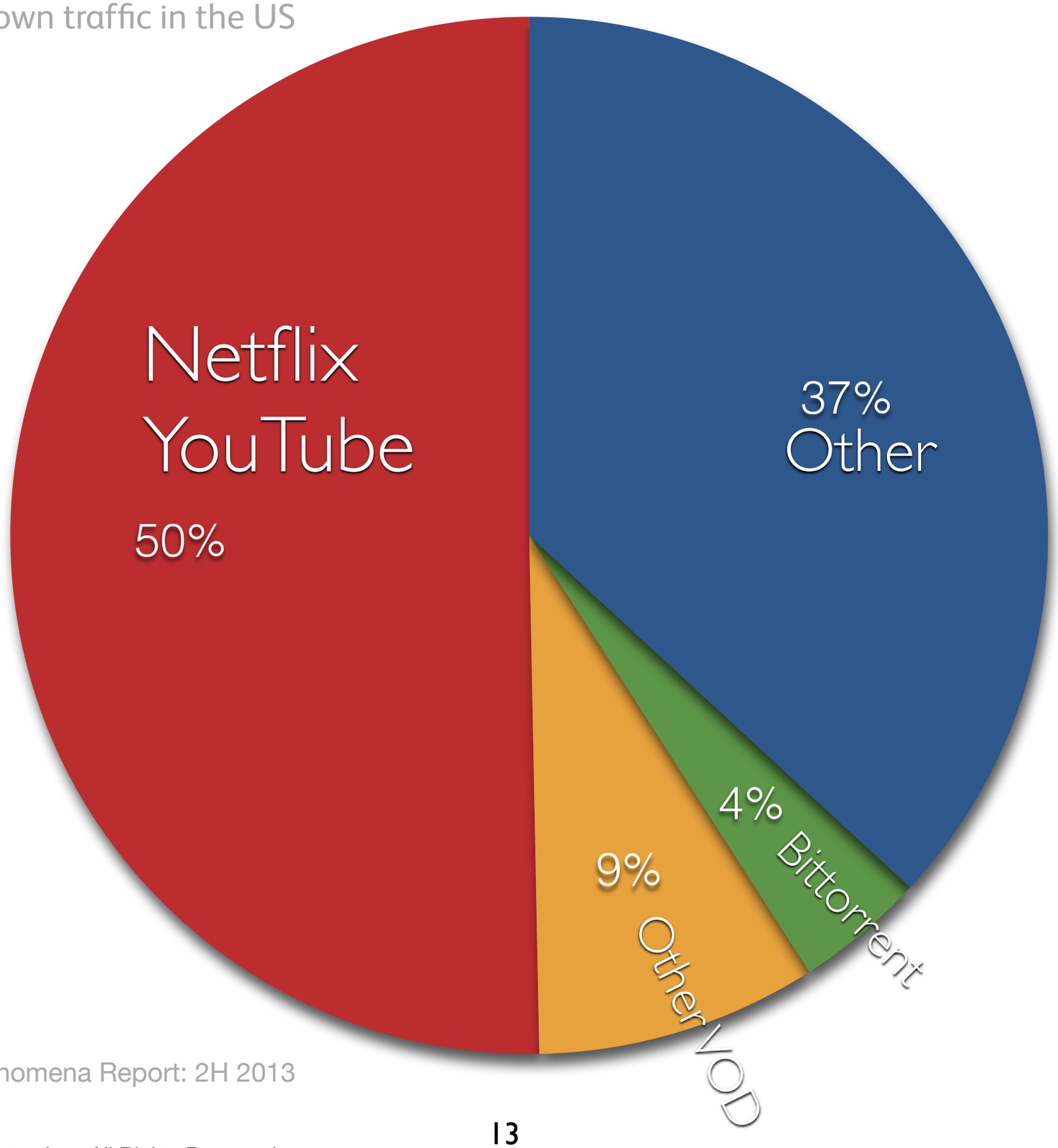
Source: Cisco Visual Networking Index (VNI) and USTelecom Analysis. A DVD is assumed to store a two-hour movie.

Internet Growth - Media types



Netflix + YouTube = 50%

of down traffic in the US



Sandvine Global Internet Phenomena Report: 2H 2013

parc
A Xerox Company

Current networks limit the "web"

Current protocols mix is not efficient

IP + TCP+ HTTP + TLS + DNS have a lot of overhead.

New protocols are being proposed to solve this (QUIC, SPDY)

IP solutions create problems

Buffers used to help, now they create bloat

IP "servers" don't scale

We need load balancers, CDNs, etc.

Current networks limit the enterprise

Network management is not simple

IP is not easy to manage, deploy and setup.

Functionality is lacking

We need to hack the network via NFV and SDN to make things do what we need them to do. IP/TCP/HTTP were not designed for virtualization.

Current networks limit the data center

TCP/IP doesn't solve the problem

TCP tries to solve congestion for the internet, not efficient transfers between blades in a rack or rack-to-rack

We are not doing point-to-point

Communication in a data center involves many decisions that are not host-to-host. We need to move large data sets to available resources and get large data sets from the nearest/cheapest place.

Current networks limit IoT

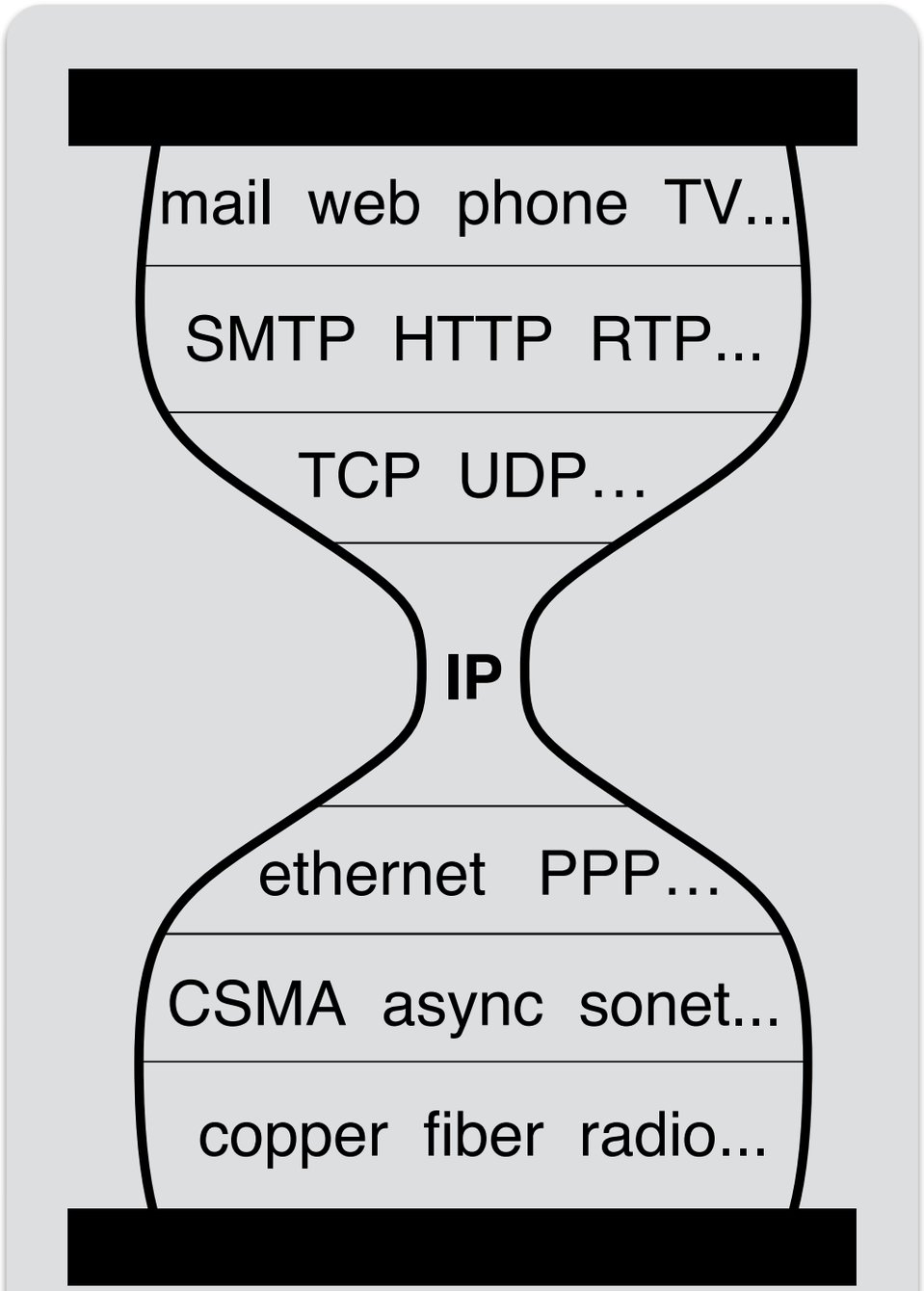
Point-to-point model doesn't fit

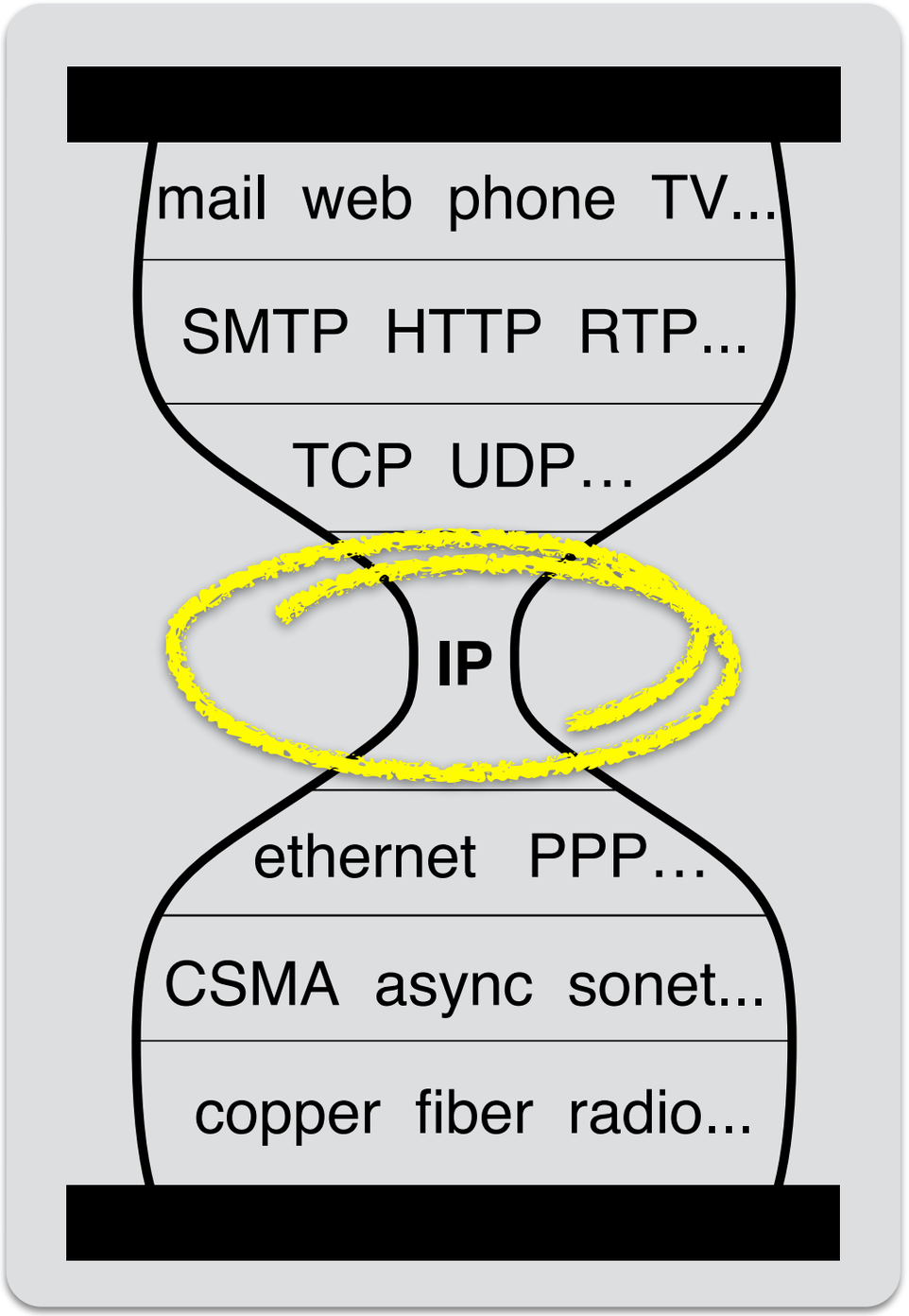
Many sensors produce data. Data might be useful for many consumers/processors.

TCP / UDP are not adequate

We need a transport protocol that deals with moving sensor data (be it temperature readings or pictures) and can support organization, security, etc. We can't rely on custom/vendor protocols.

Where's the problem?





What next?

CCN - Core idea

Content-centric Networking (CCN) transfers named data

CCN overview

Step 1 - Name the data

Name every piece of network data

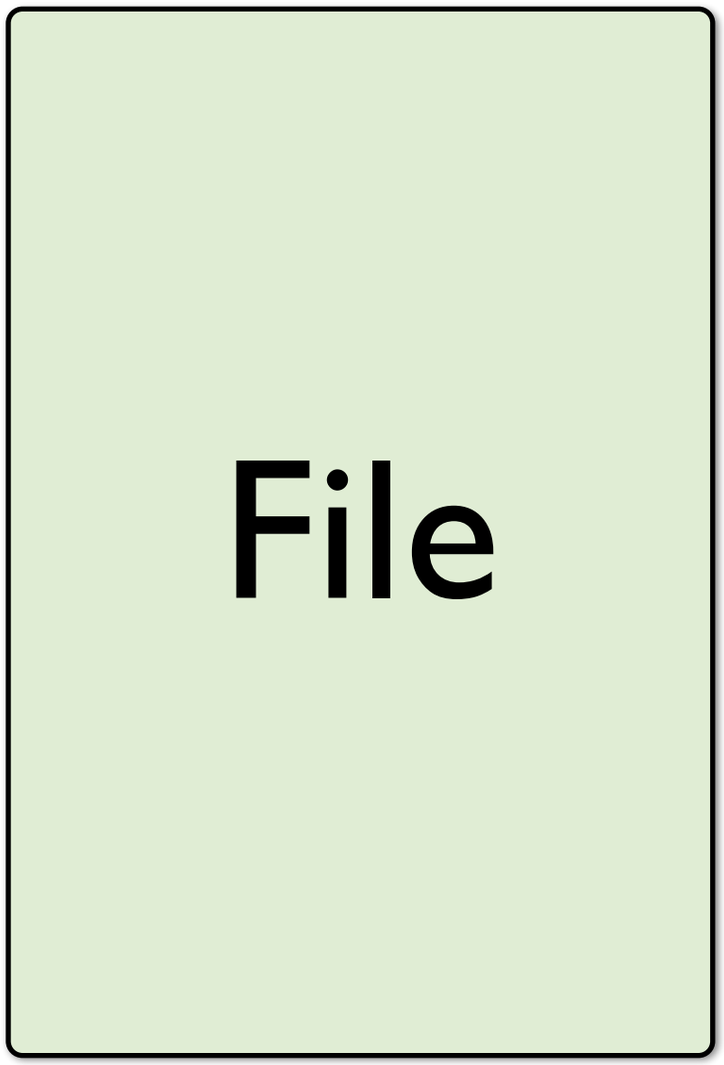
Step 2 - Secure the data

Secure every piece of network data

Step 3 - Transfer the data

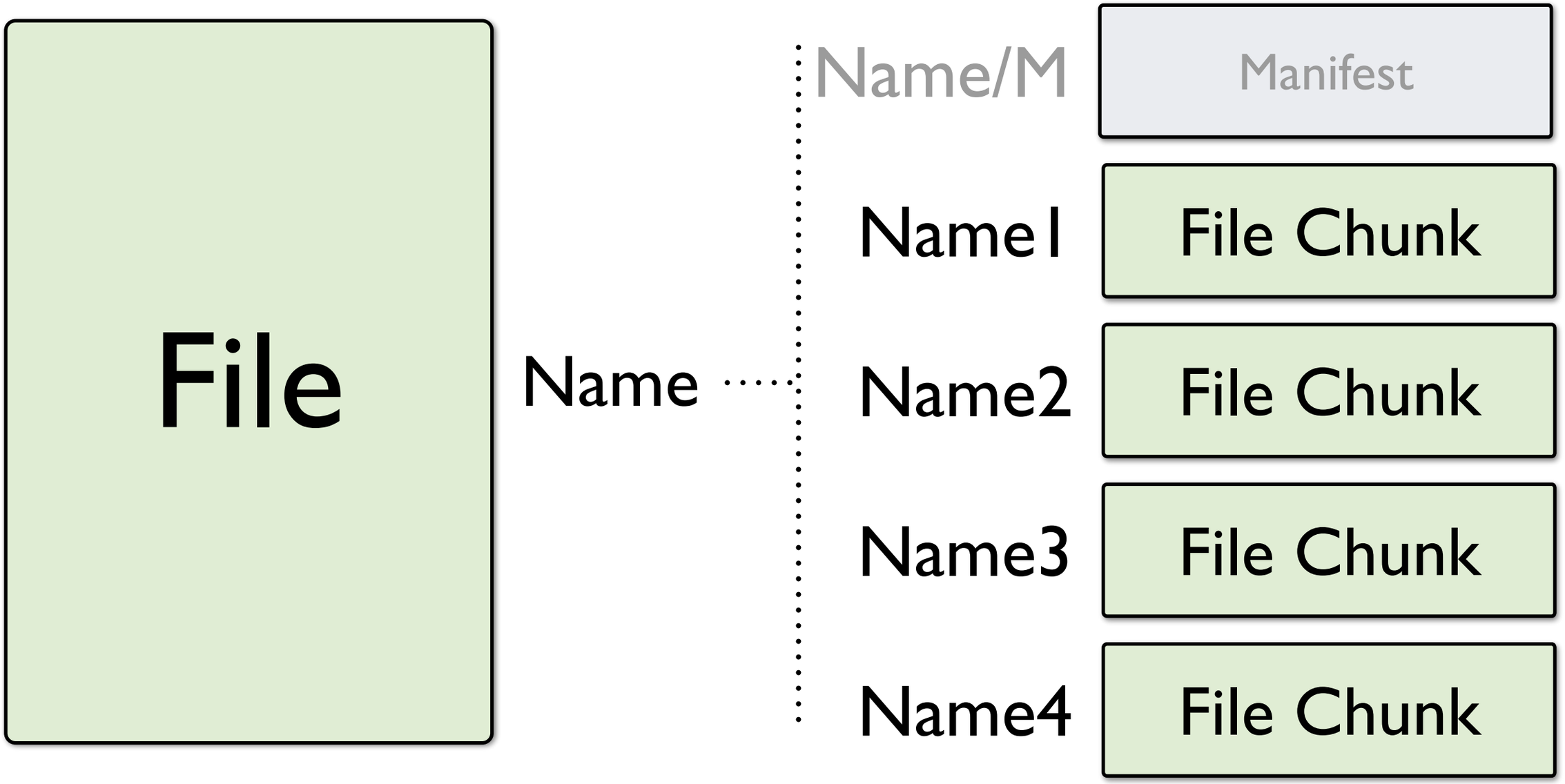
Move the data to interested recipients

name data



Name

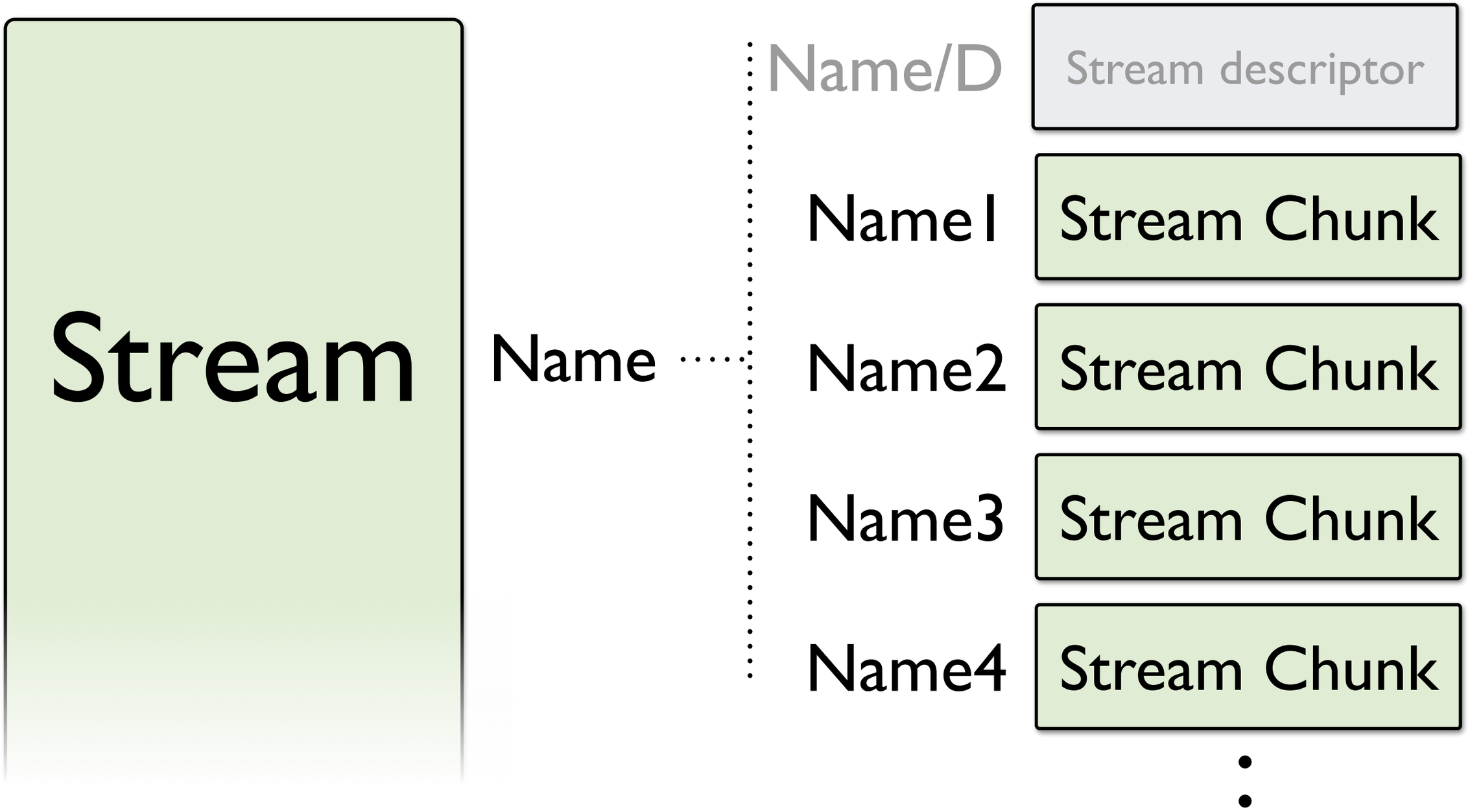
A large object like a file has a CCN name



CCN also names the network sized chunks

Name

A stream has a name



The stream descriptor is the metadata for the stream.

CCN Names

Name Segment based

Labeled binary segments

Hierarchical structure

Ordered sequence of segments

/seg1/seg2/seg3/seg4

CCN Names

/parc/ccnx/presentations/slide10/v=2/c=0

globally routable name segments

application dependent name segments

protocol dependent name segments

CCN Name origins

Routable name segments

Globally coordinated namespace (like domain names)

Application name segments

Applications can name their data any way they want (like filenames)

Protocol name segments

Protocols use conventions in naming to transmit information (like version, chunk number, etc)

CCN Name origins

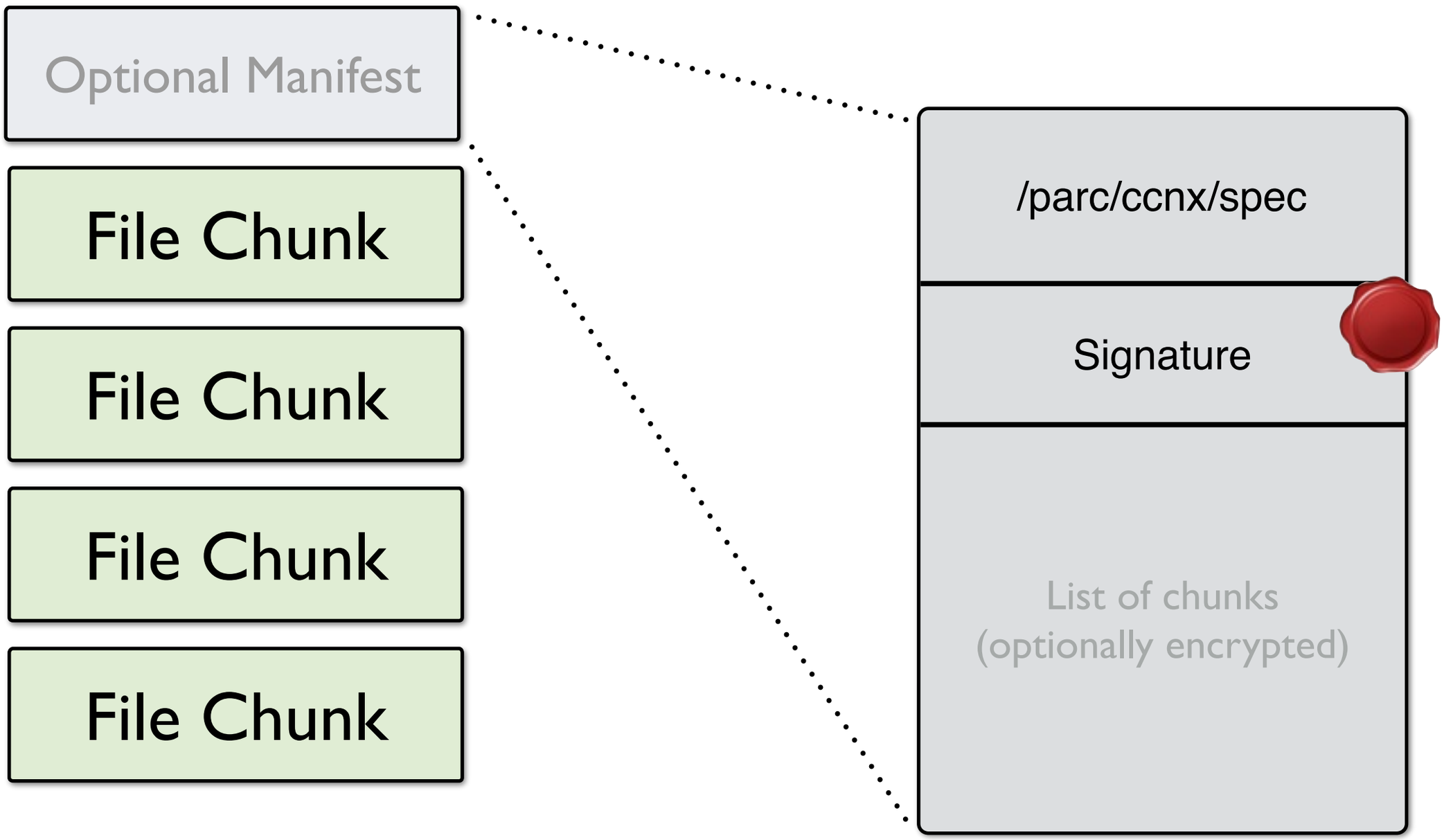
CCN names behave like URLs

CCN names can come from

- Users (typing name in / copy+paste)
- Links (from a web page / document)
- Built in (NPR app goes to NPR directly)
- Generated (Search toolbars for example)
- Scanned (QR codes, NFC, etc.)

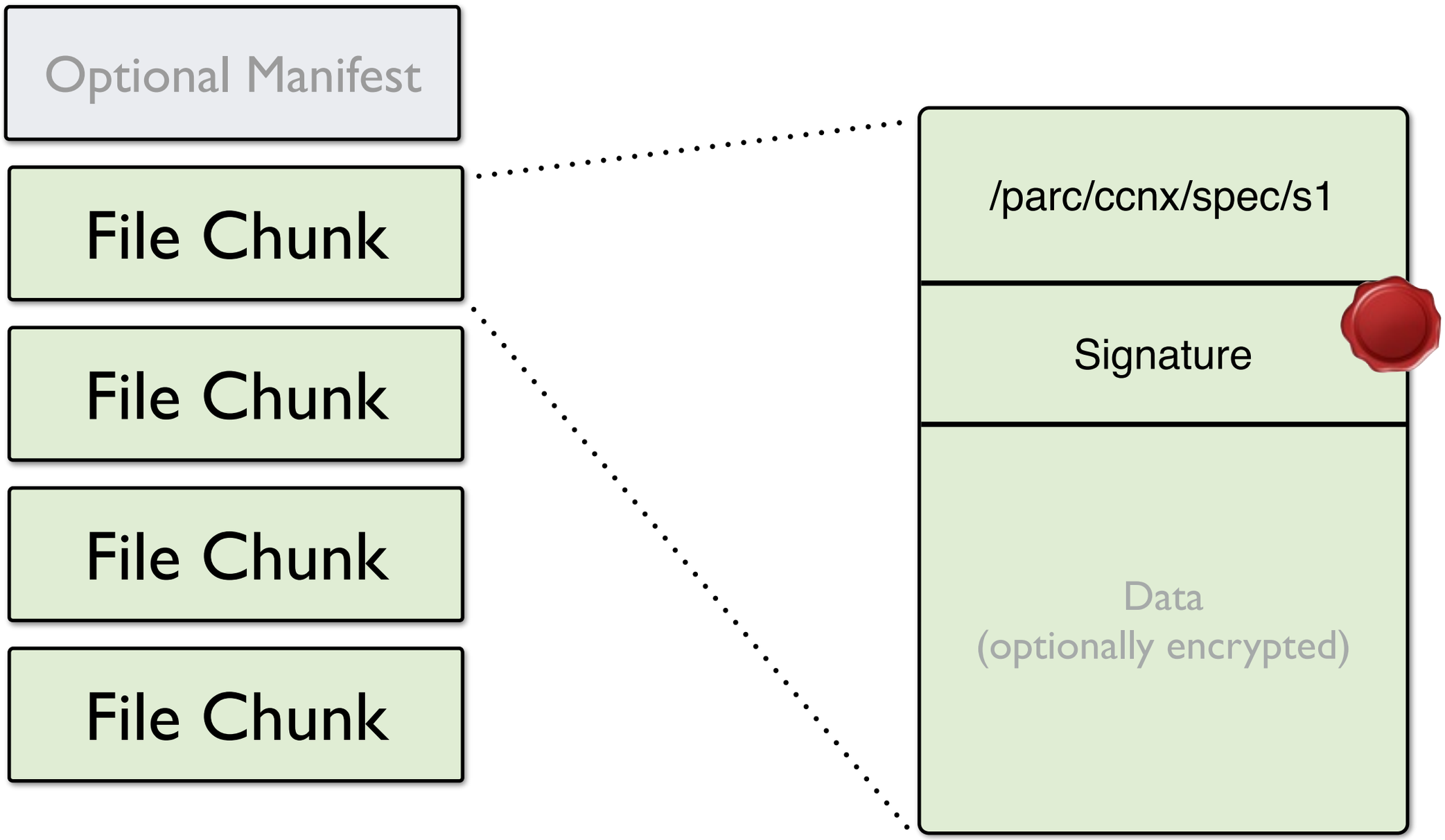
secure data

Secure whole object



CCN names and signs the file via a manifest

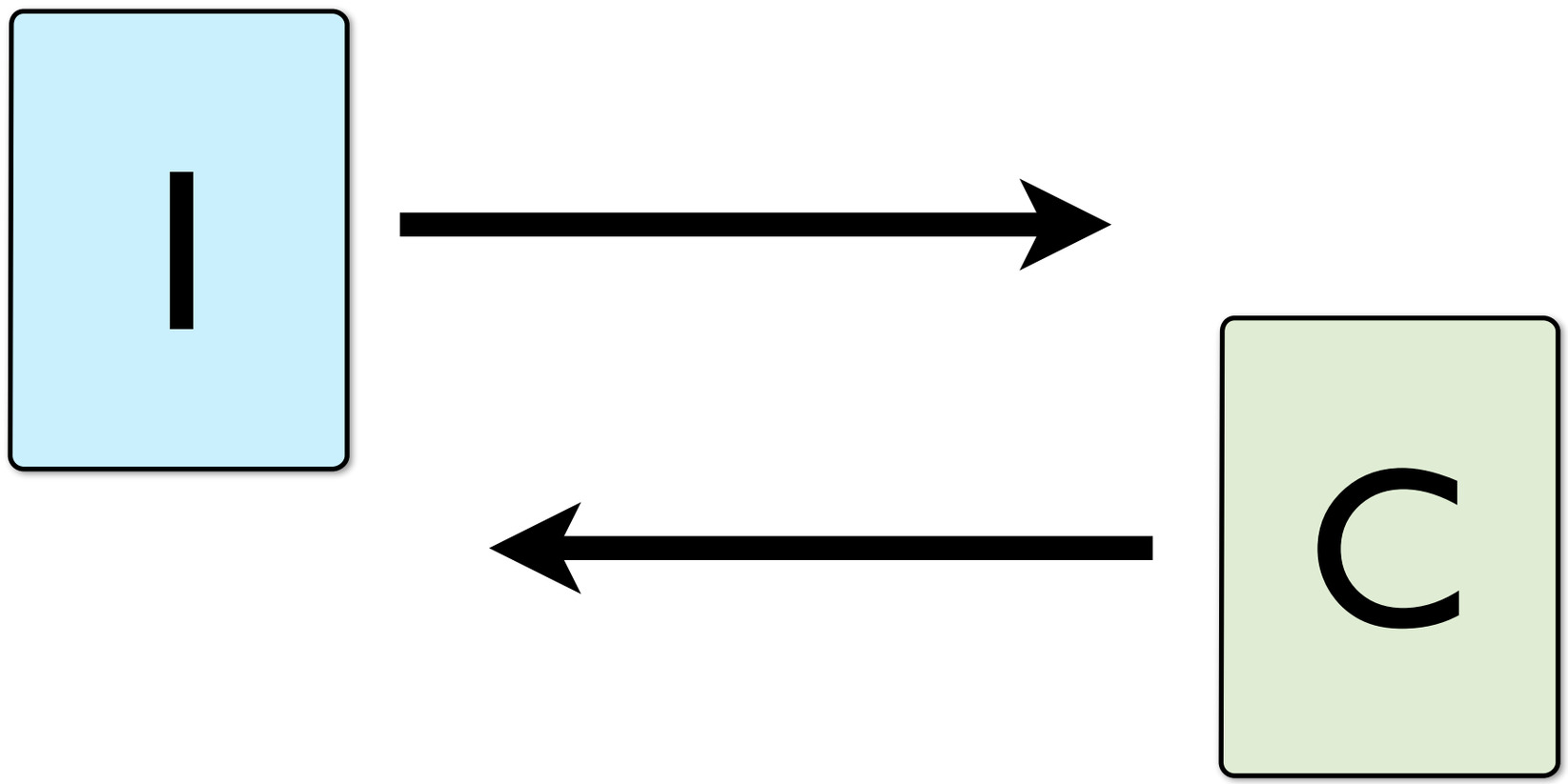
Secure single chunk



CCN names and signs every chunk

transfer data

Core Protocol



One interest packet gets one content packet

Interest

/parc/ccnx/slide1/s5 ?

Content Object

/parc/ccnx/slide1/s5

Signature

Data
(optionally encrypted)

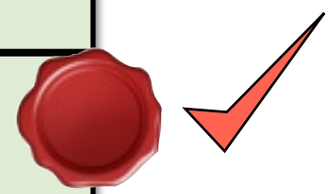
Interest

/parc/ccnx/slide1/s5 ?

Content Object

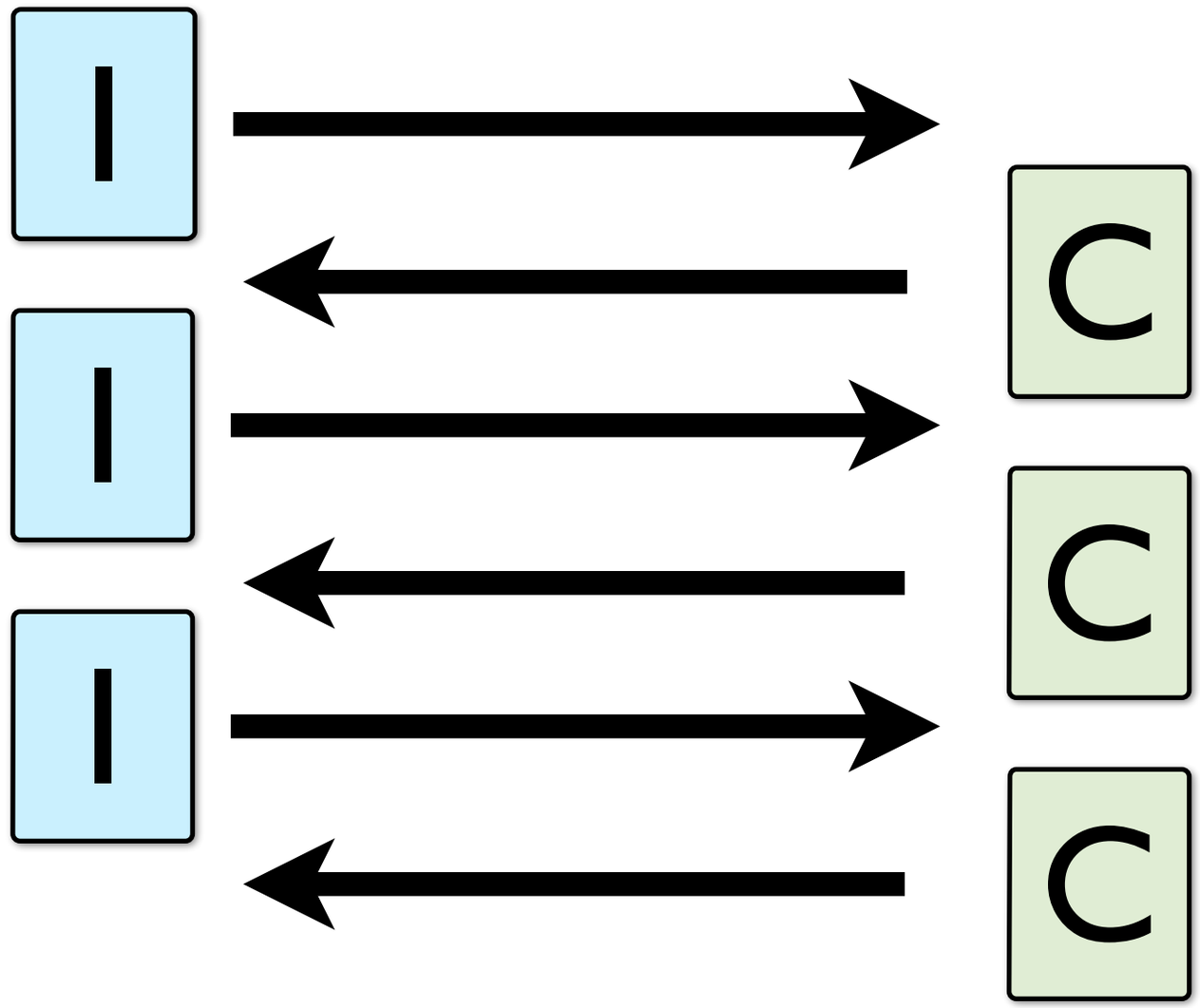
/parc/ccnx/slide1/s5

Signature



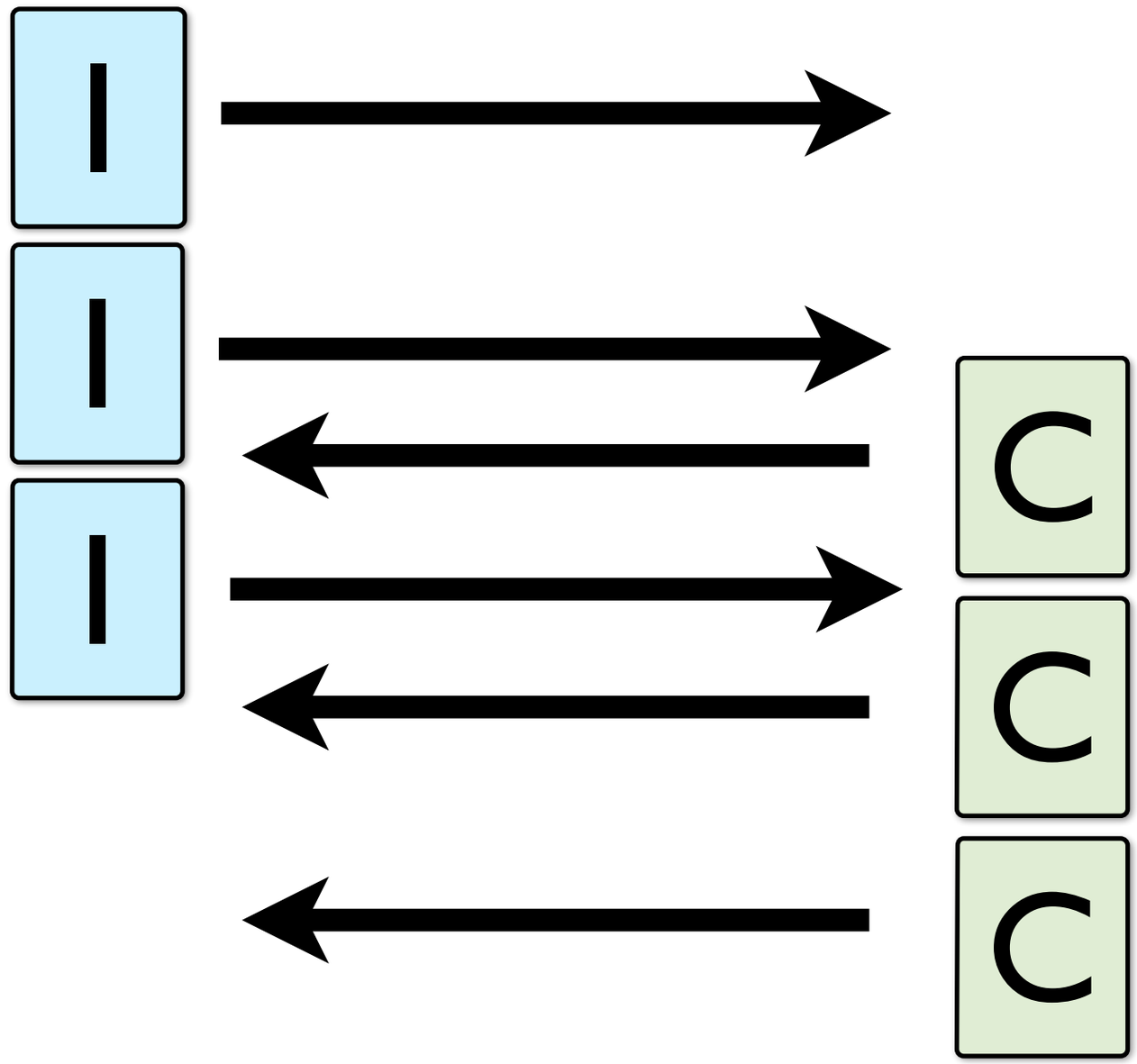
Data
(optionally encrypted)

Transport Protocols



Transport protocols are built on core exchanges

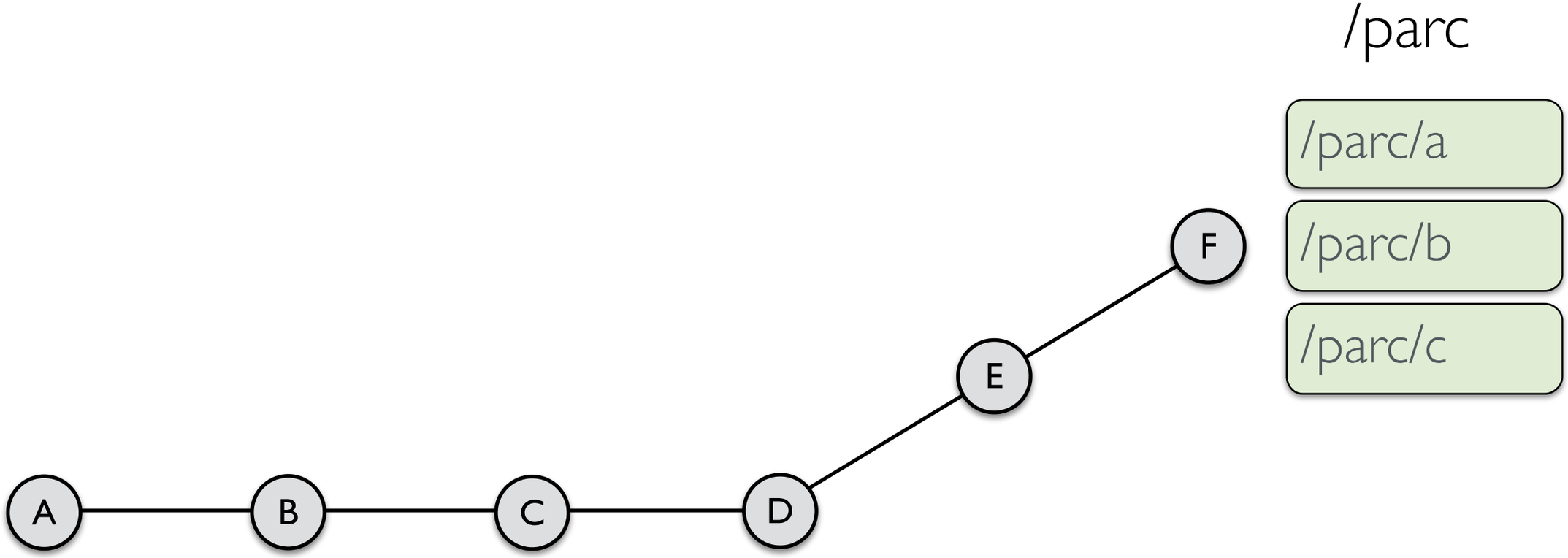
Transport Protocols



Transport protocols can do parallel requests

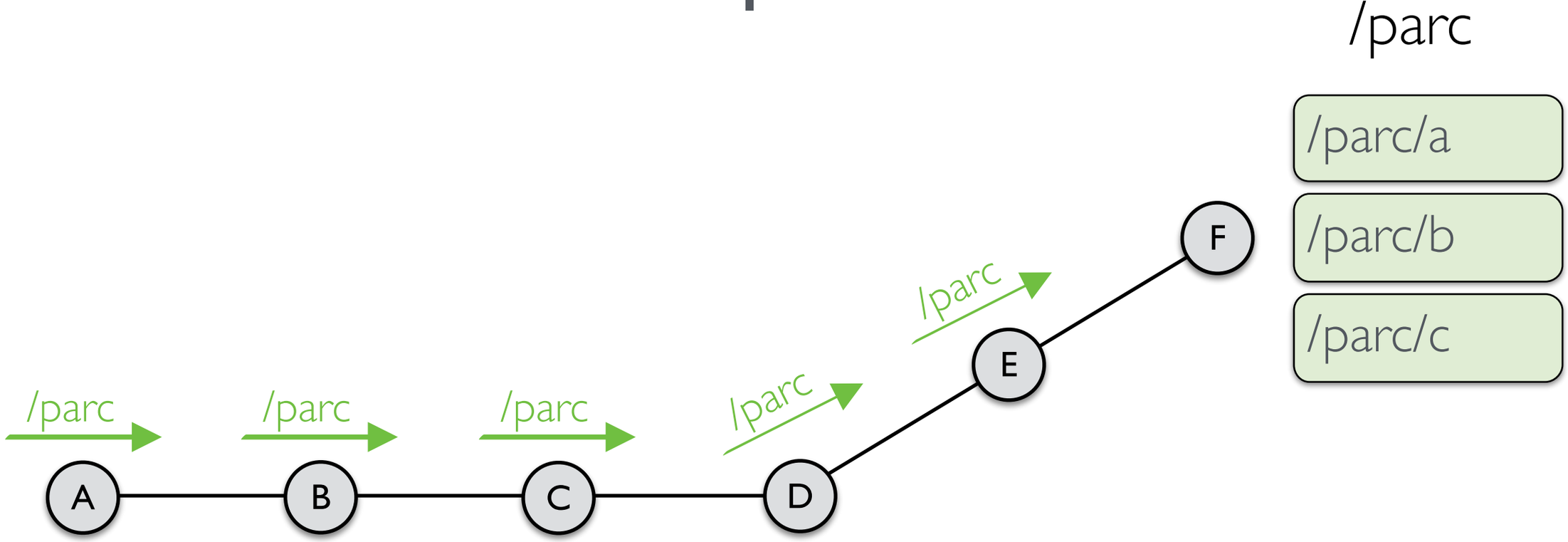
CCN - How it works

Name routes are advertised



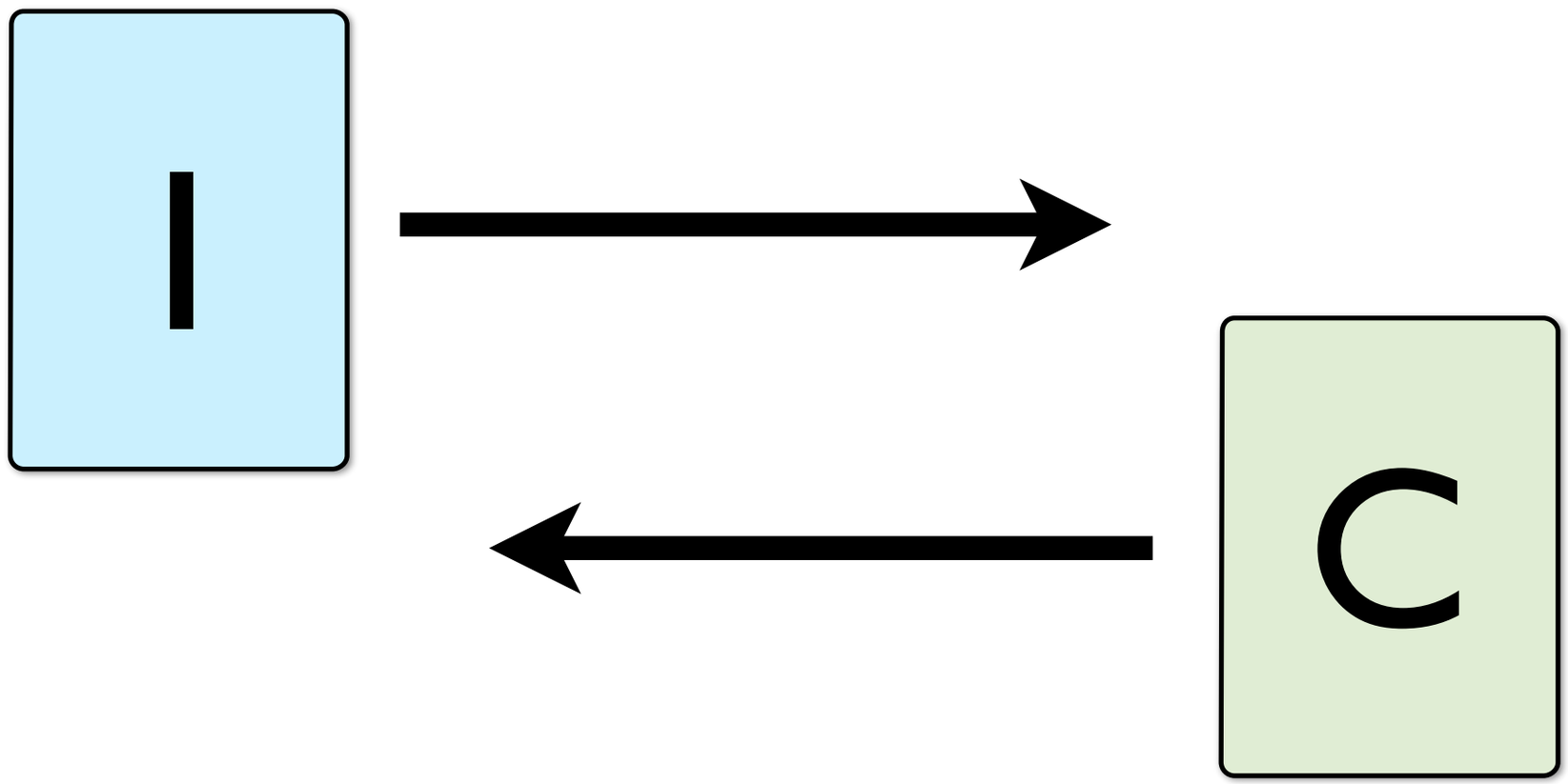
Node F serves /parc content.
Advertises a route to /parc

Name routes are set up in forwarders



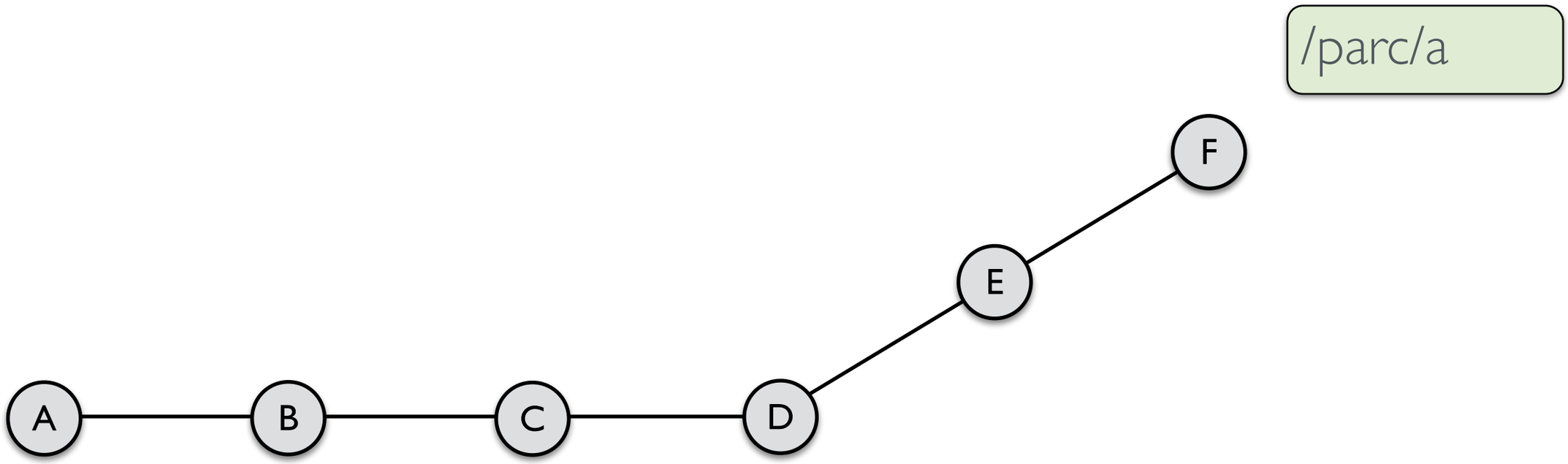
Routes are set up pointing to F for prefix /parc

Core Protocol



One interest packet gets one content packet

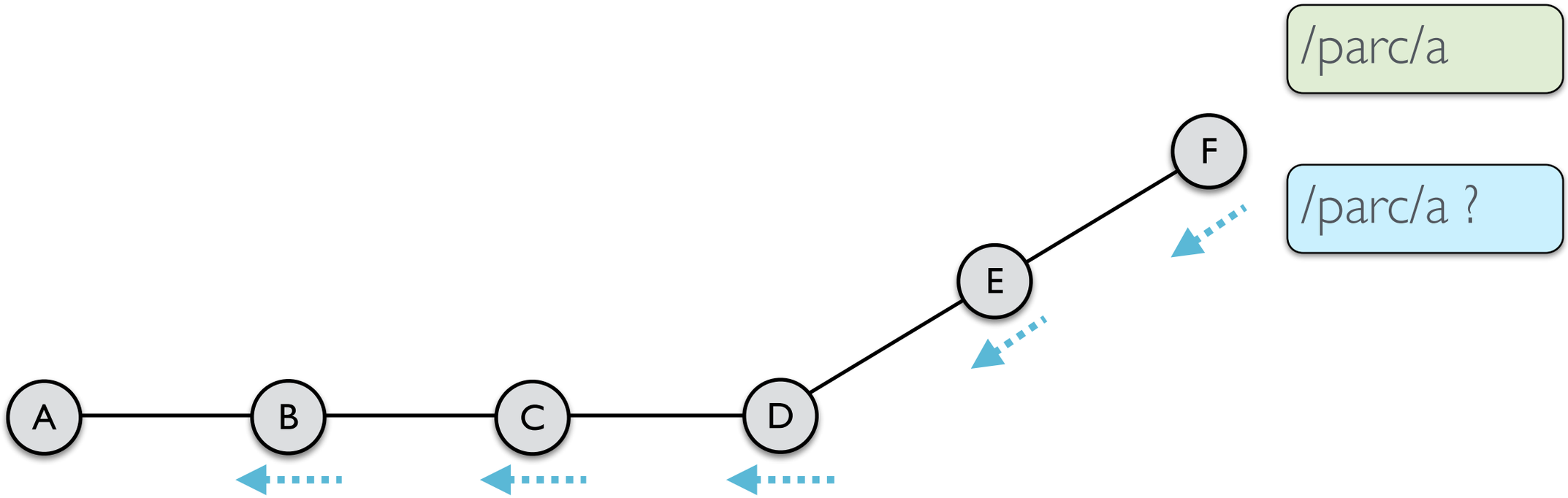
Interests follow the routes



/parc/a ?

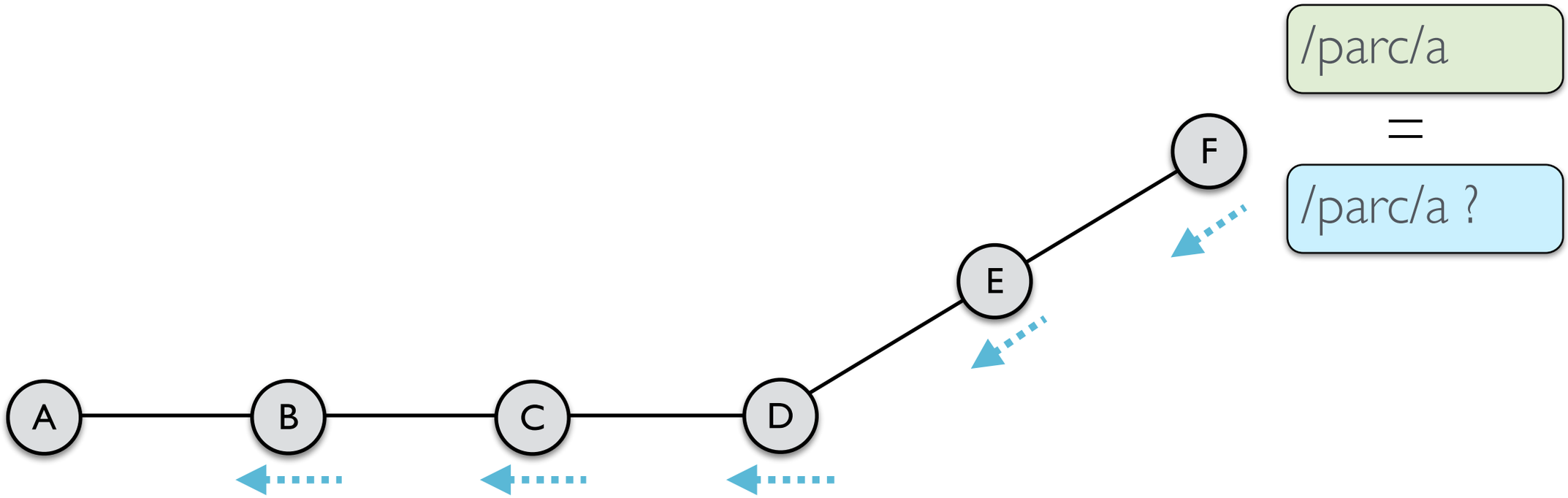
A issues interest for /parc/a

Interests leave breadcrumbs



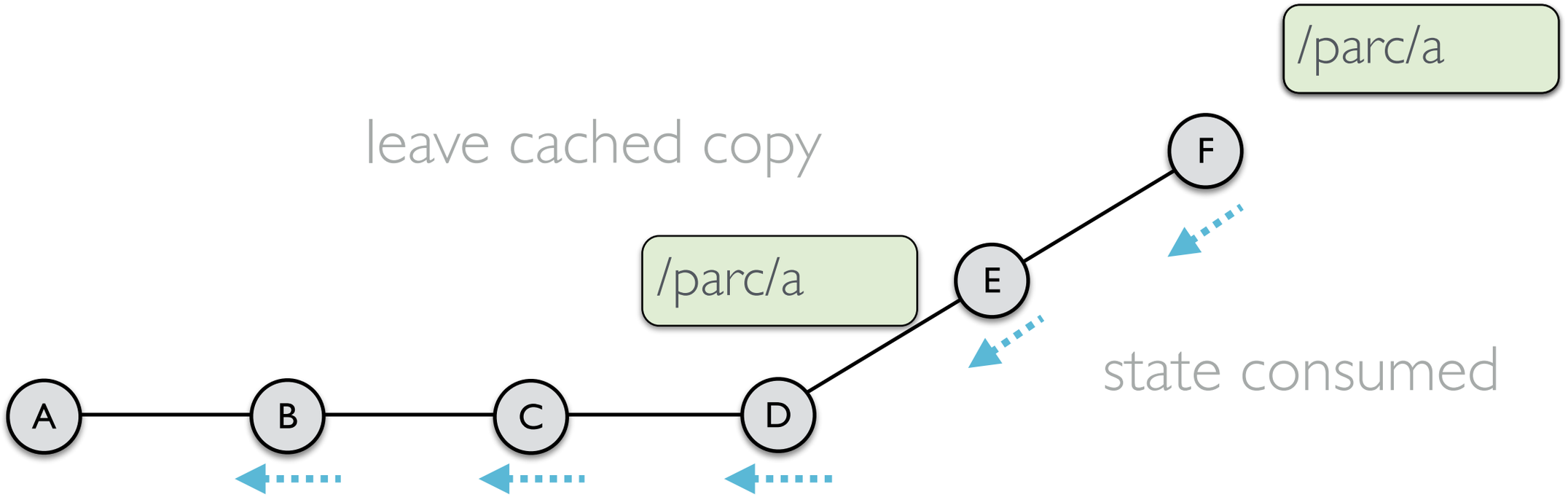
Interest leaves reverse path state in the network

Interest matches content



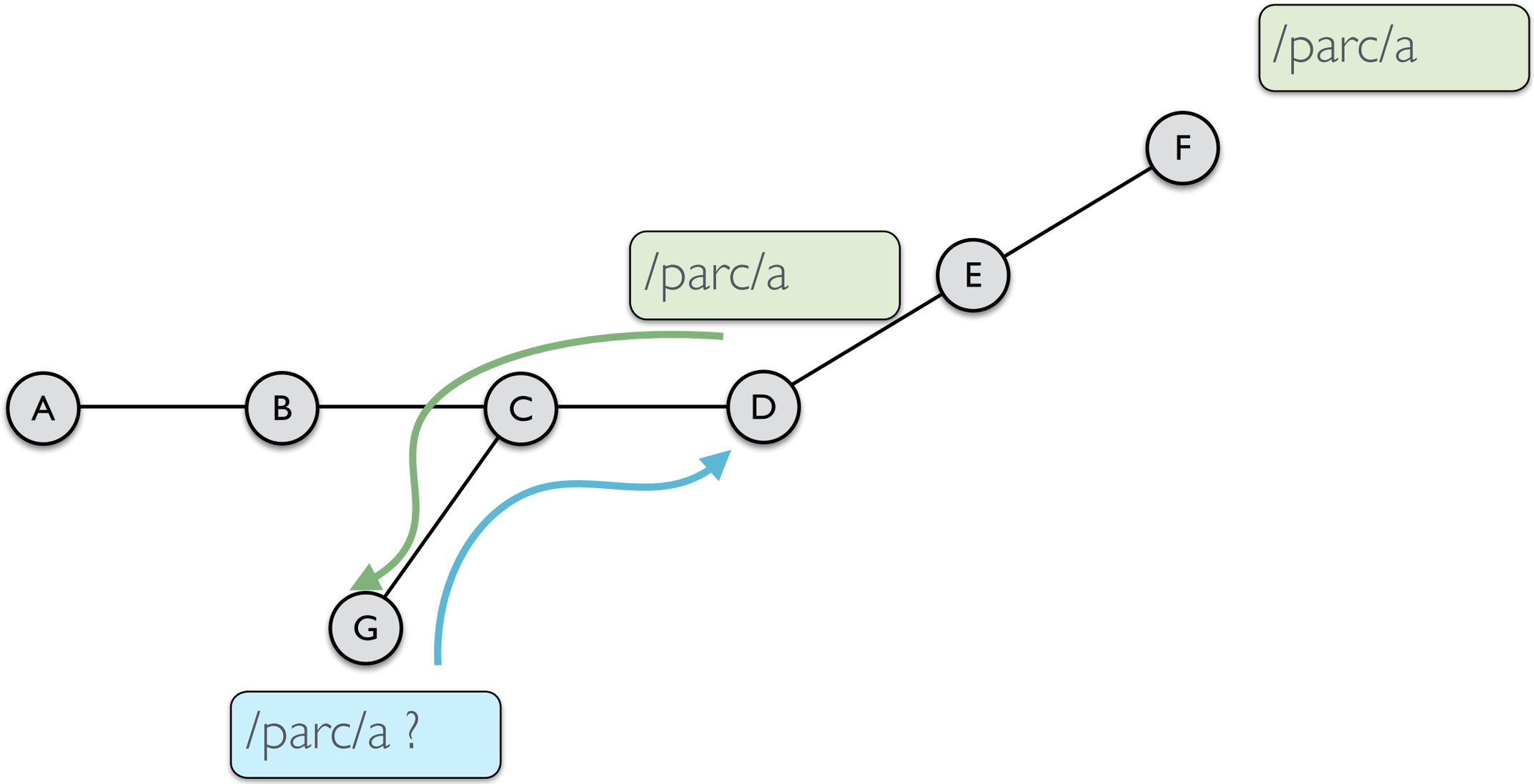
Interest for /parc/a finds a match at F

Content follows reverse path



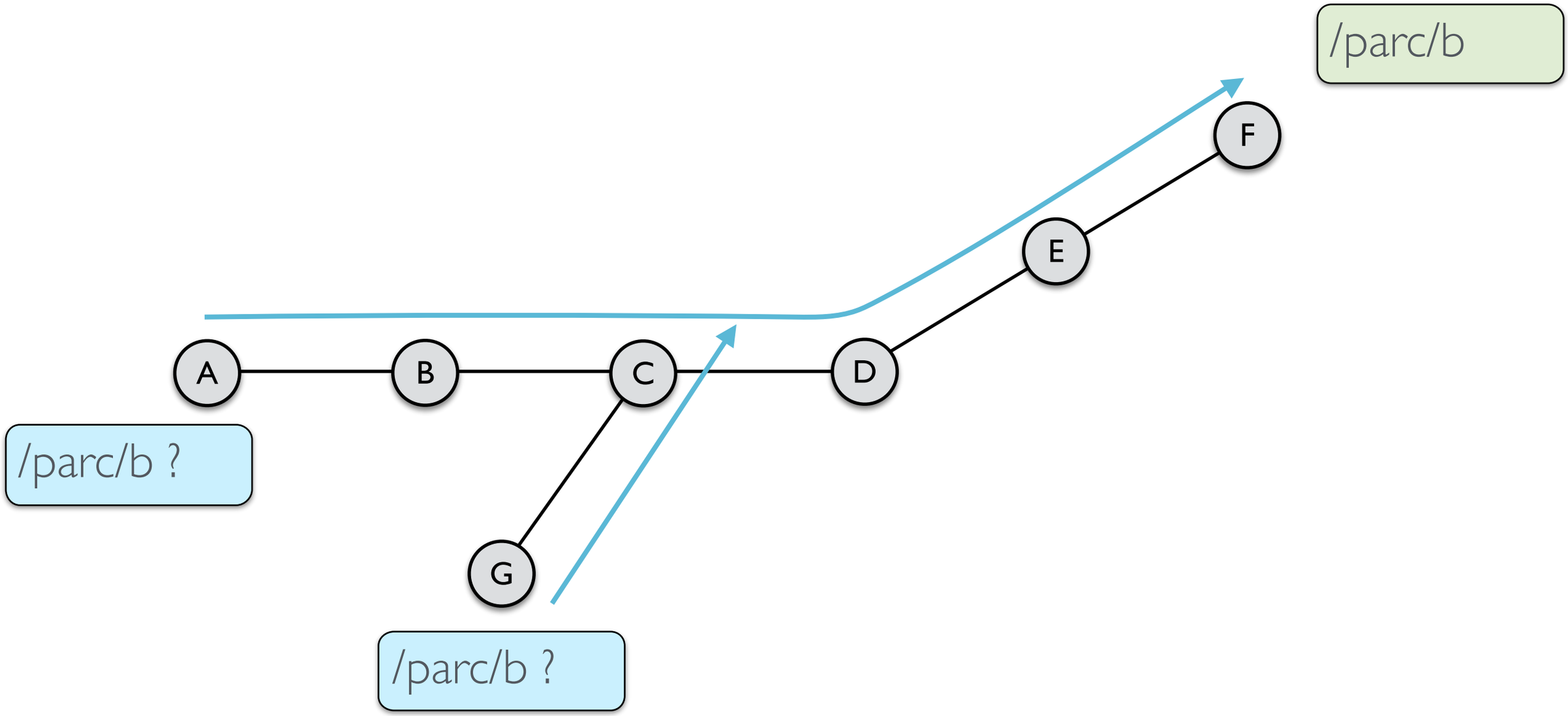
Content packet follows reverse path and consumes state

Any node can cache



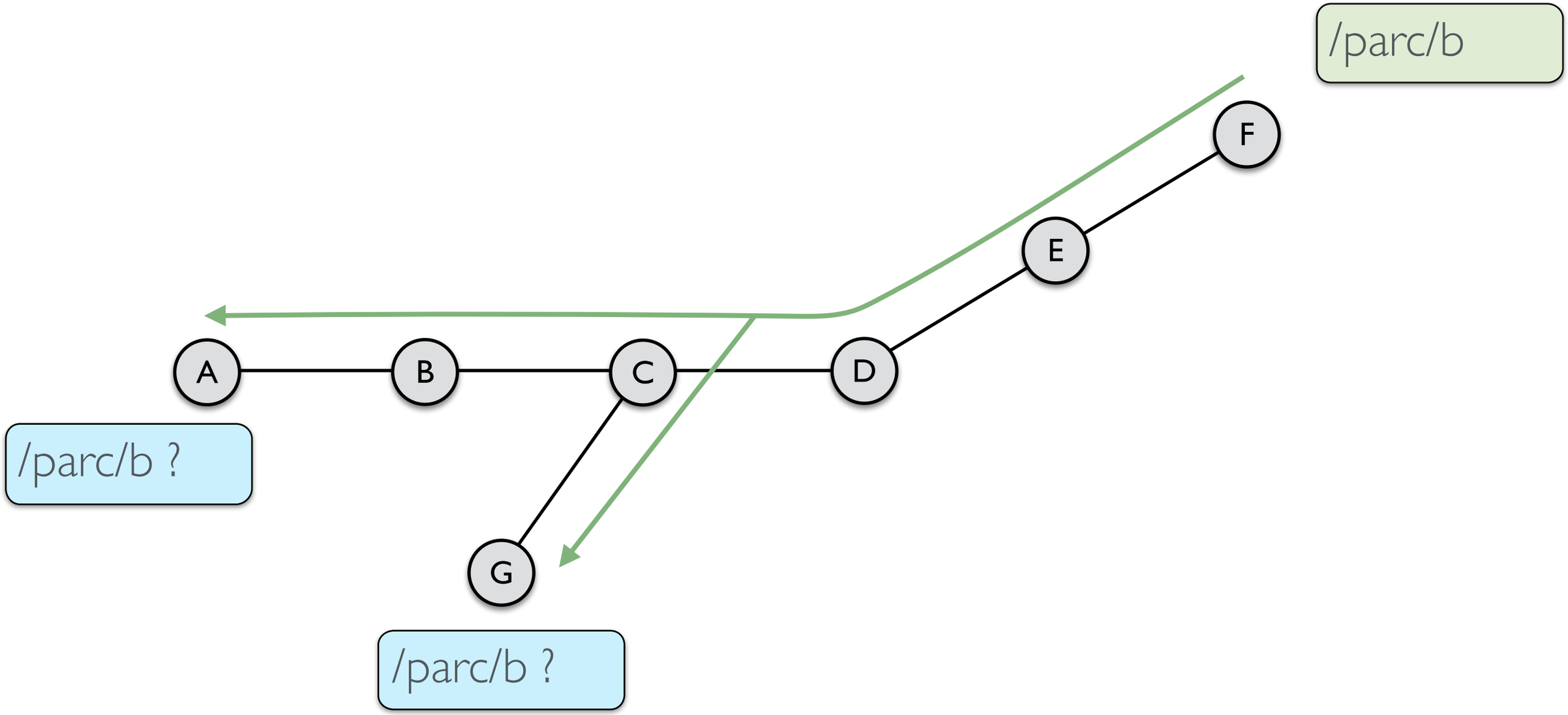
Node G requests same content, gets it from a cache in D
G authenticates content via the content signature, not the sender

Interests are aggregated



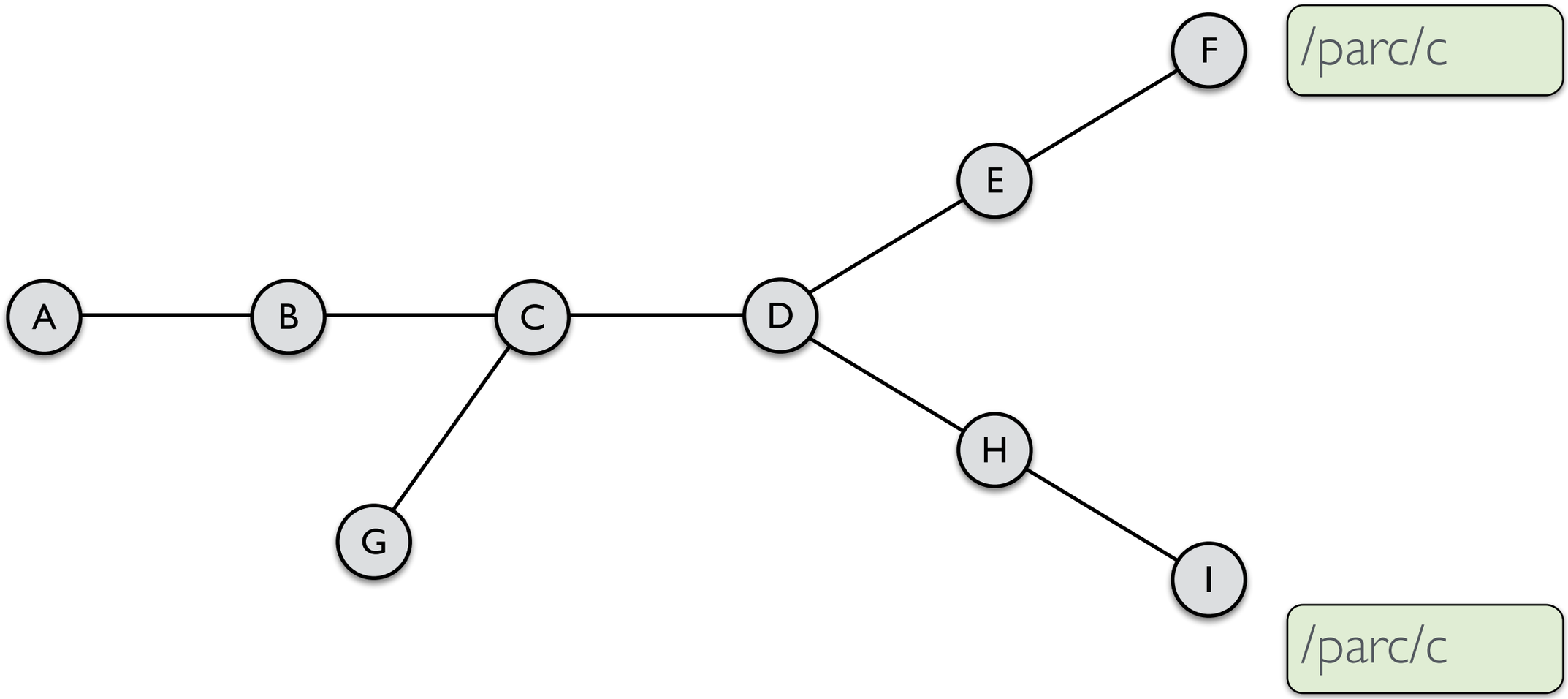
Node C aggregates the interests (only sends one upstream)

Content is sent in all interest paths



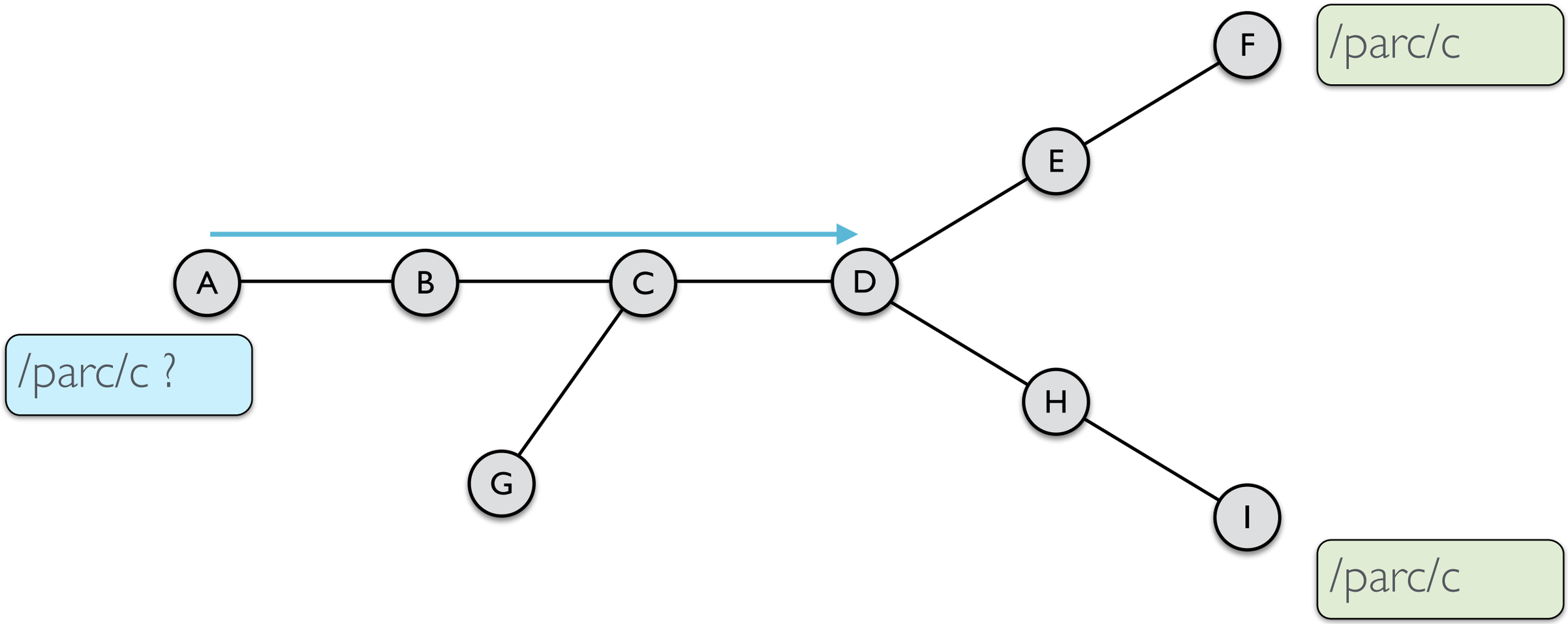
Node C sends the content packet to both B and G

Multiple nodes can advertise same prefix



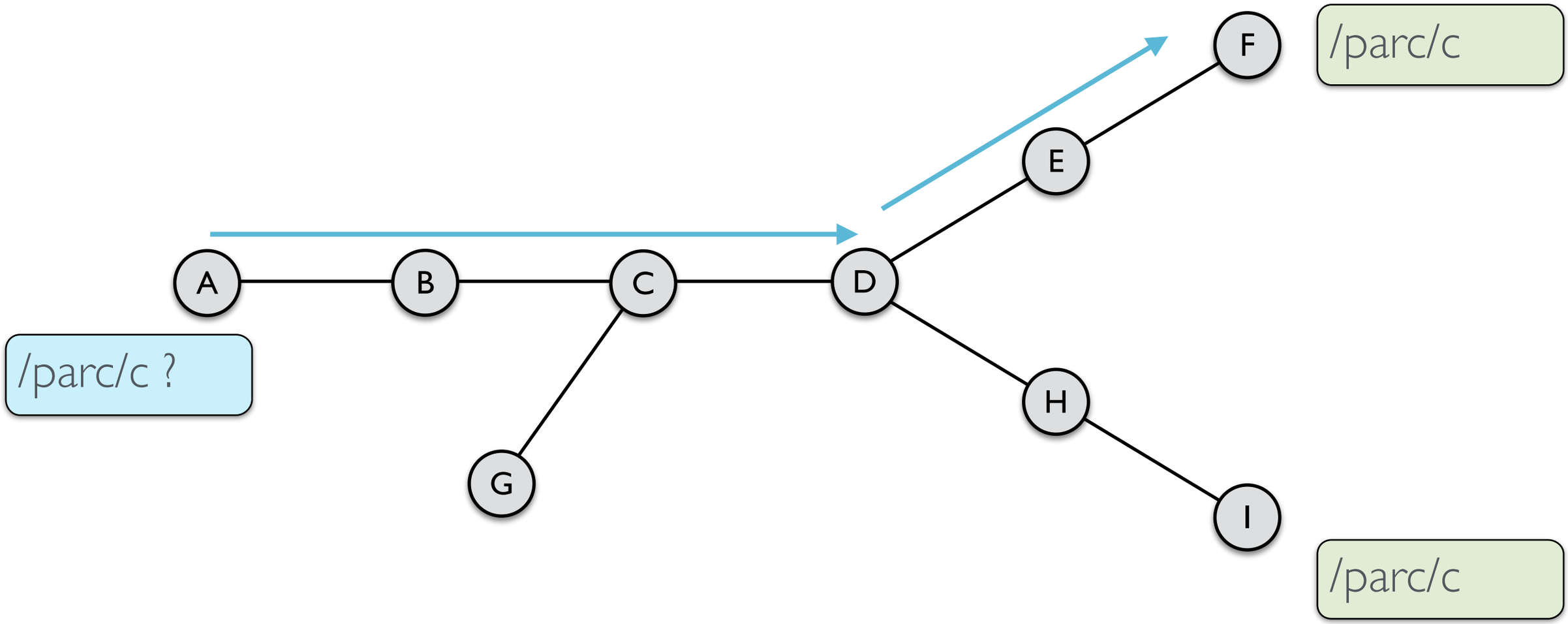
Both F and I can provide and advertise the content for /parc

Nodes can forward on either path



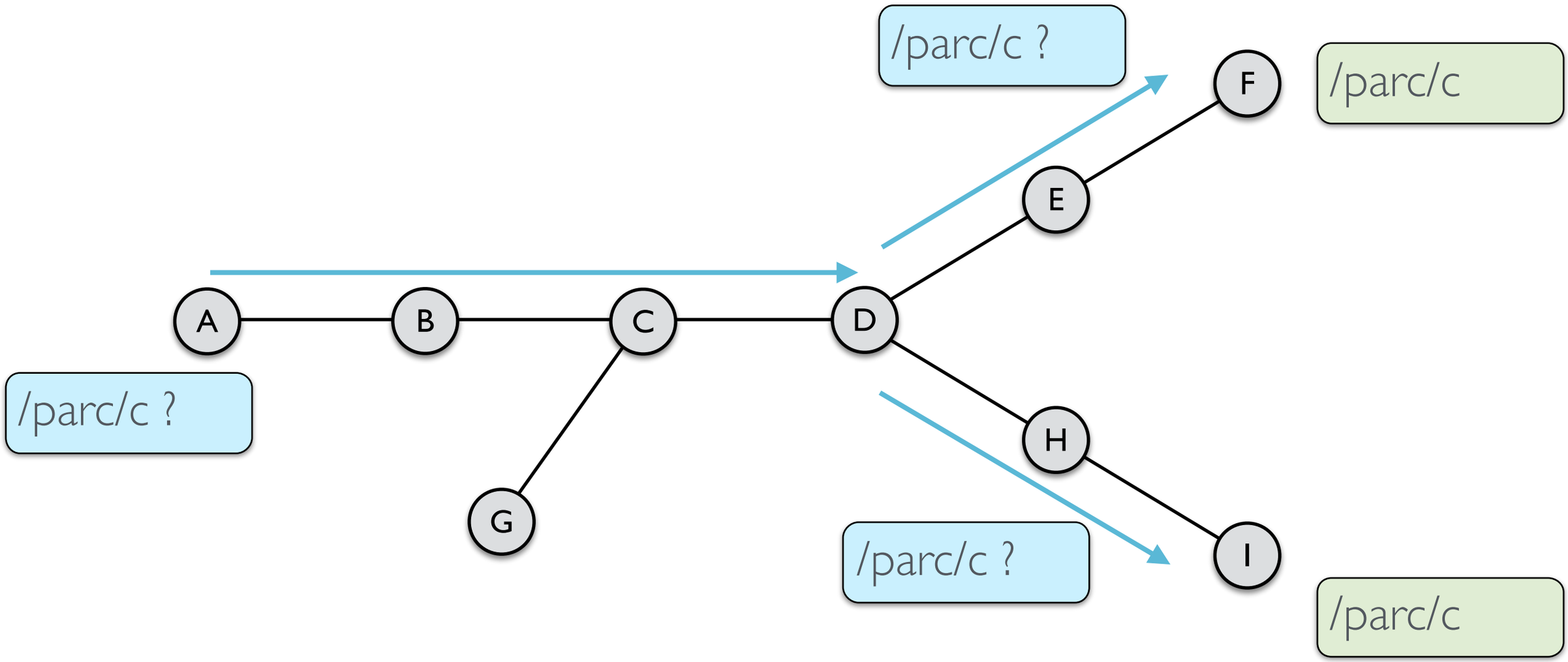
Nodes can choose where to send each interest

Forward on “best” path



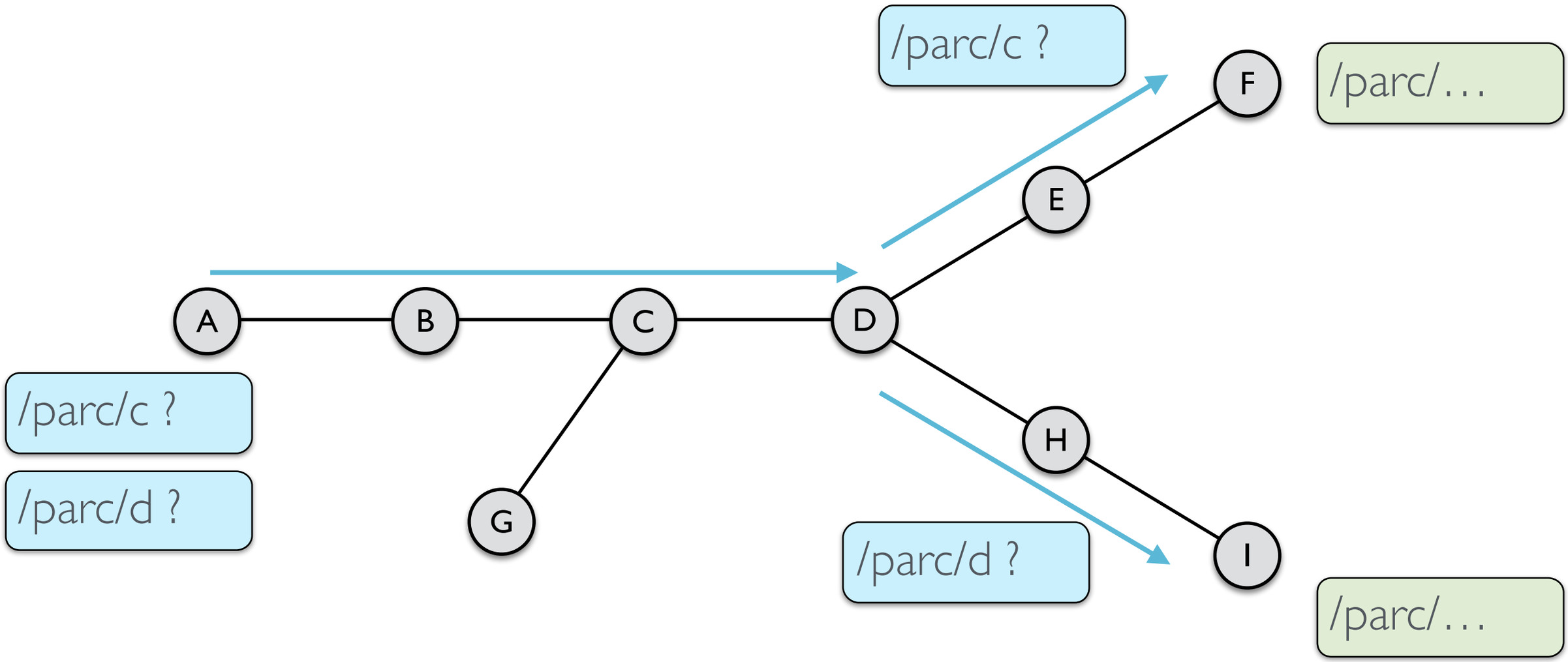
Choose one path to forward on

Forward on both paths



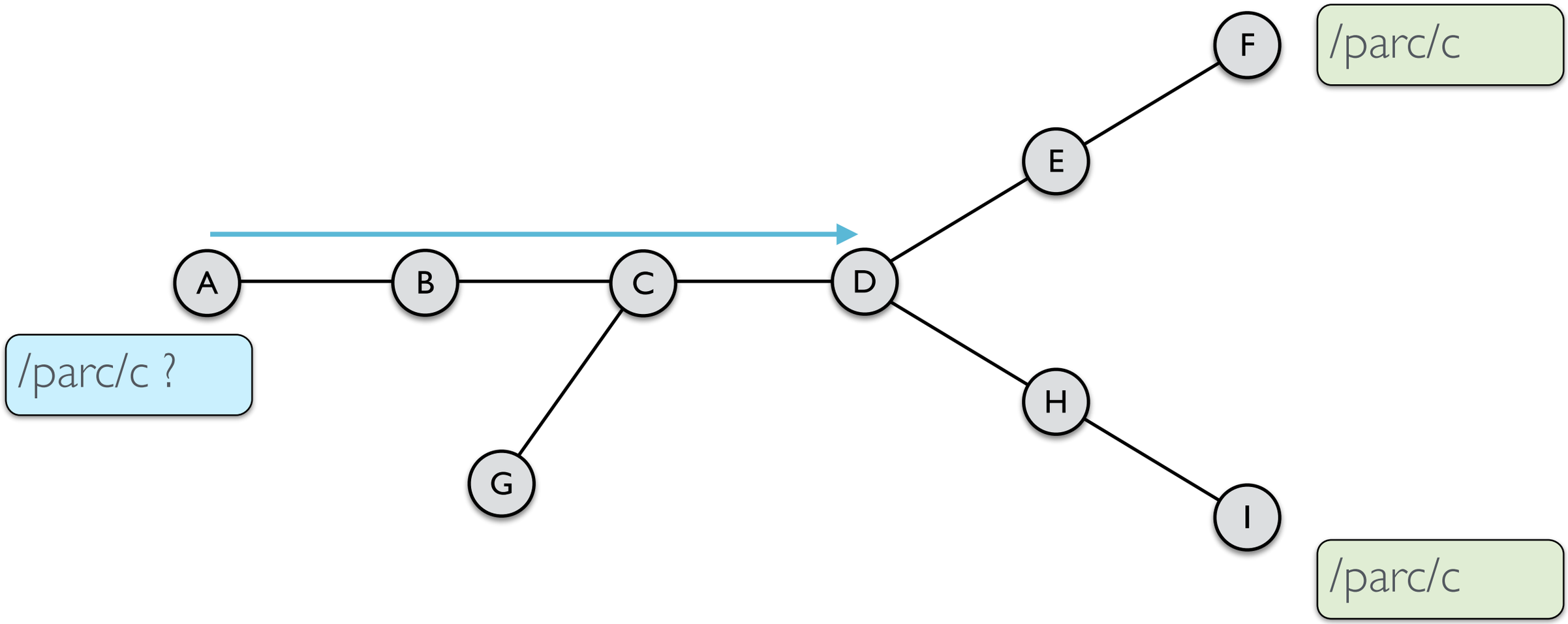
Node D will only reply to node C once
This can provide redundancy (at a price)

Forward on alternating paths



CCN is not connection oriented
Each interest can go to a different content provider

Strategy Layer determines forwarding



The Strategy Layer at a forwarder determines what forwarding policy a node follows.

CCN - Examples

Sample use case

Web page

A simplified example

Web page

Step 1 - Request web page manifest

Retrieve the manifest of the web page

Step 2 - Request web page content

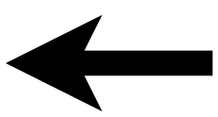
Download the web page content (html)

Step 3 - Request references

Download embedded references (images, css, etc)

Request page manifest

/parc/ccnx/index.html/manifest/c0

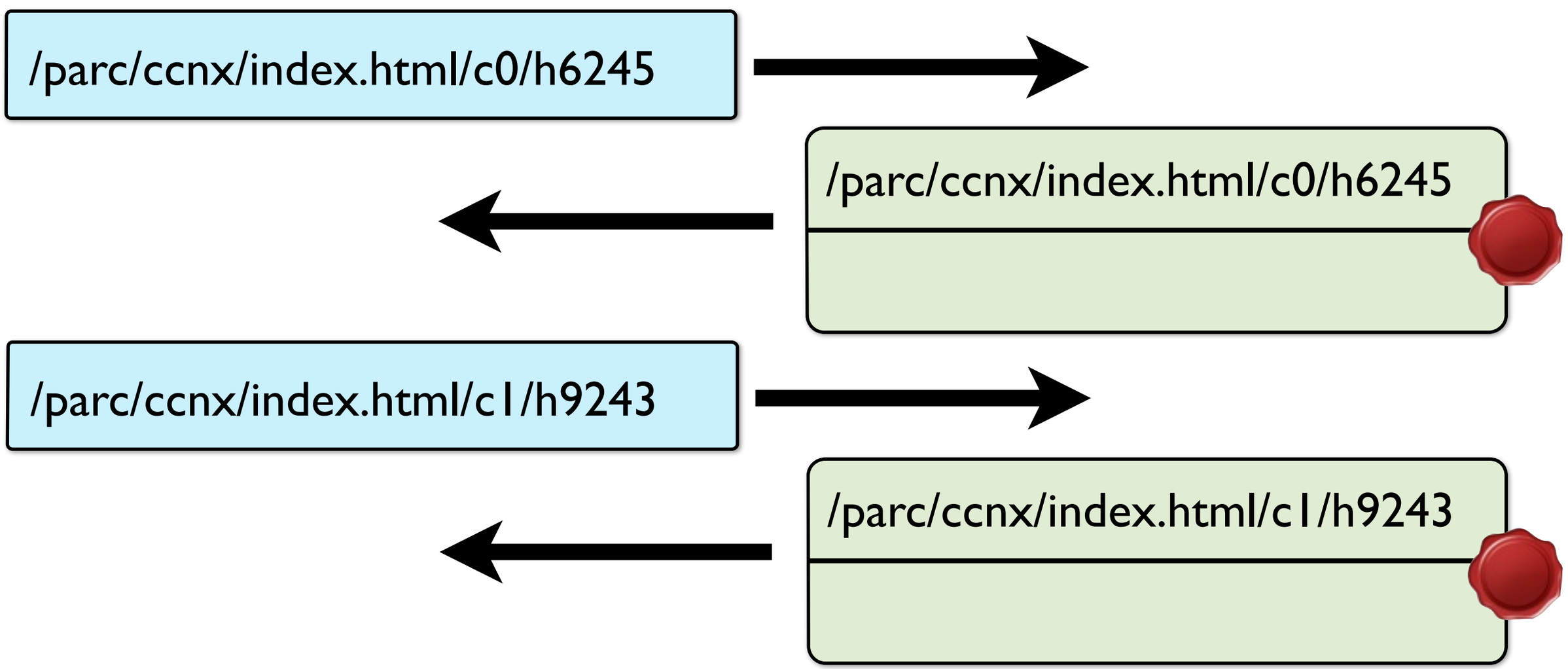


/parc/ccnx/index.html/manifest/c0

c0/h6245
c1/h9243
[EOF=s0]

Request the list of chunks for the main webpage
Each chunk identified with a hash

Request page content



Request each chunk using name and hash

Retrieve references

/parc/ccnx/style.css/manifest/c0

/parc/logo.jpg/manifest/c0

/xerox/logo.jpg/manifest/c0

/parc/ccnx/style.css/manifest/c0

…

Sample use case

Netflix

A simplified example

Netflix

Step 1 - Login

Contact Netflix to login, retrieve session key

Step 2 - Request movie network name

Get the actual network name

Step 3 - Request personal movie key

Get a key to decrypt the movie

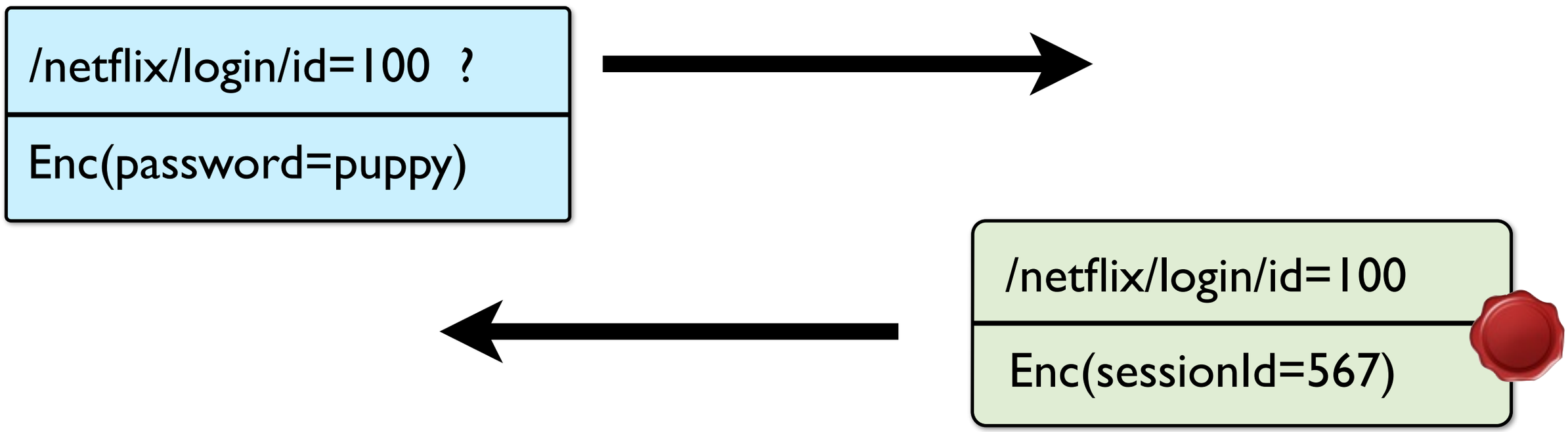
Step 4 - Request movie manifest

Get the movie manifest object

Step 5 - Request movie stream

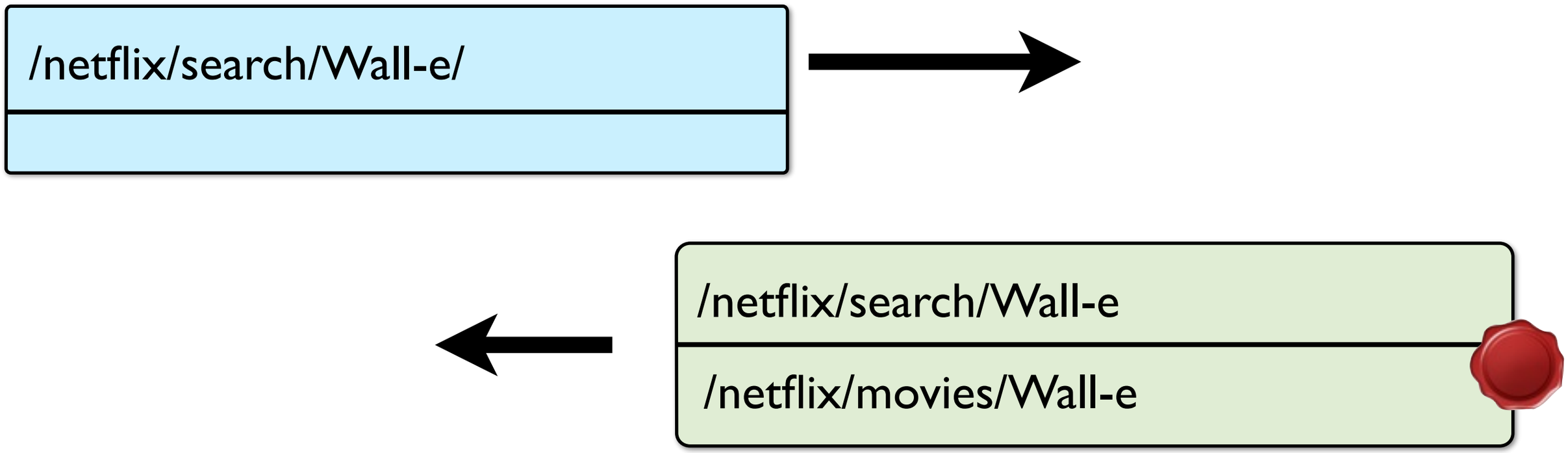
Download the actual movie contents based on the manifest.

Login to service



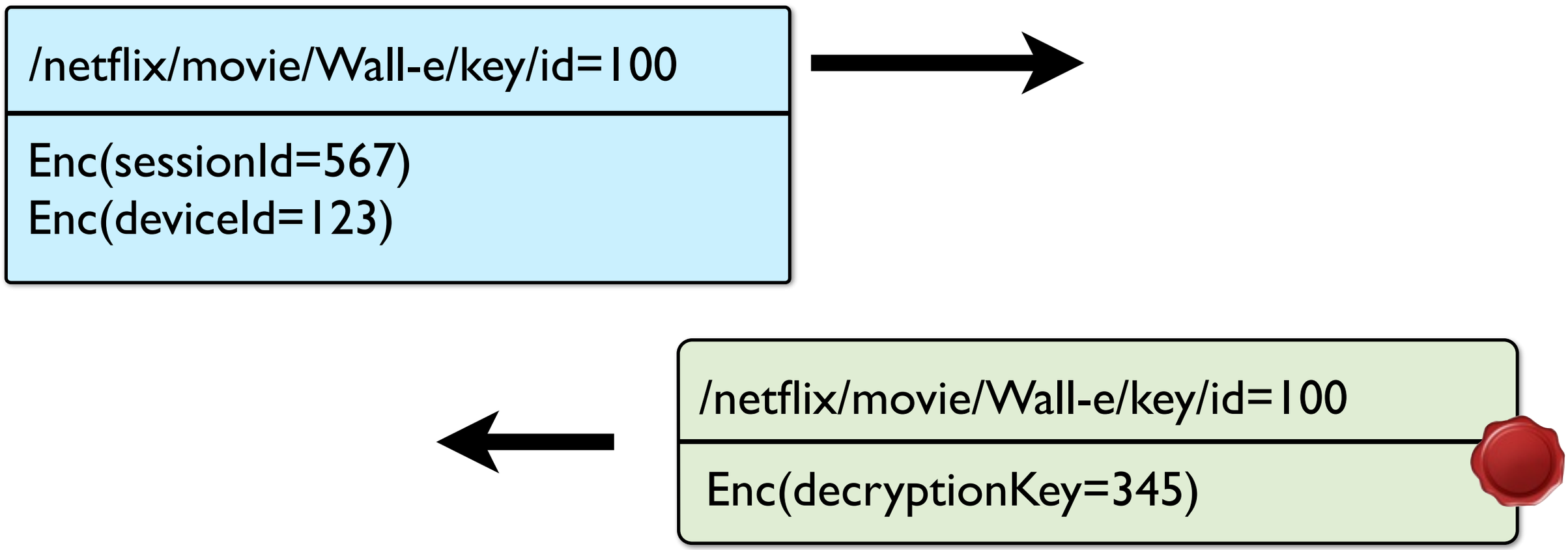
Request a session ID from Netflix using our login ID
Authenticate sending our password, encrypted

Get movie network name



Search Netflix for the movie Wall-e
Get back a network name of the movie

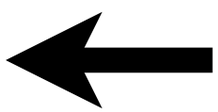
Request movie key



Request a personal key for the movie we want to watch
Send session ID and device ID

Request movie manifest

/netflix/movies/Wall-e/manifest/c0

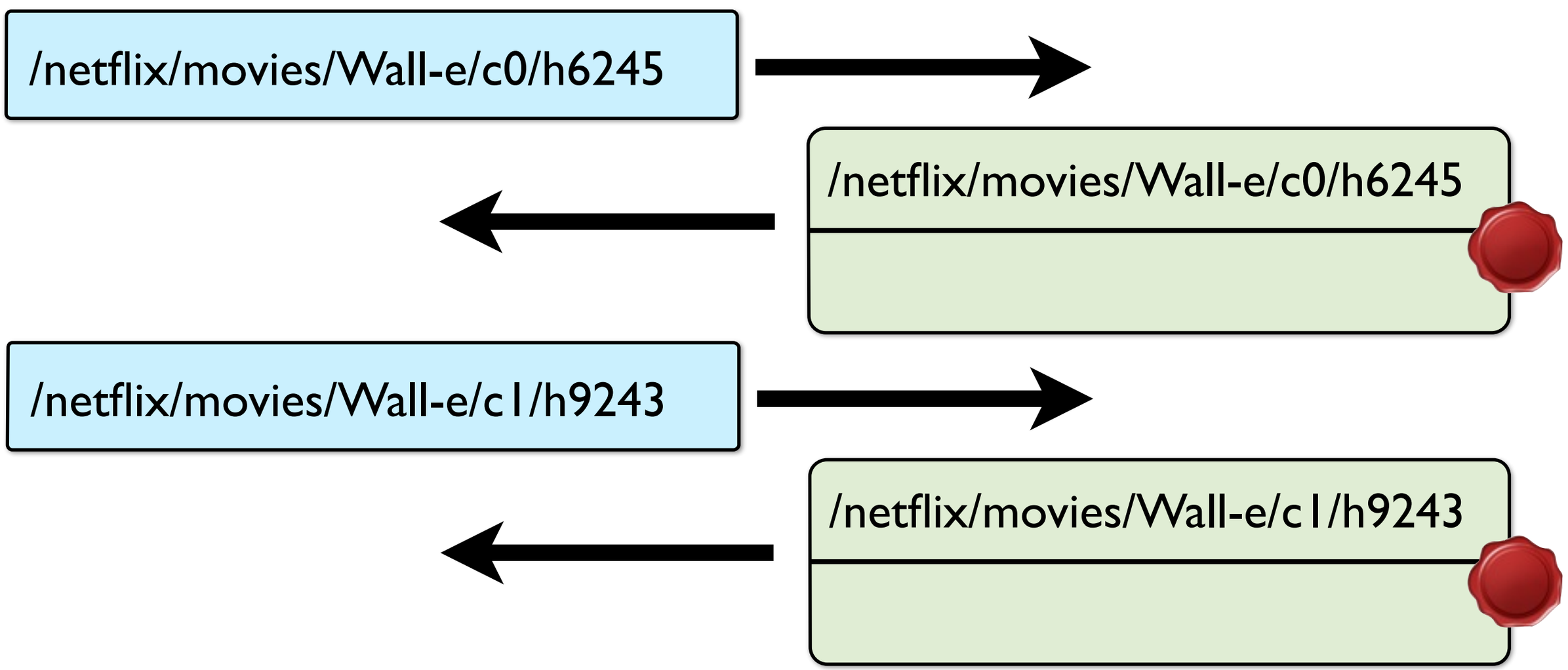


/netflix/movies/Wall-e/manifest/c0

c0/h6245
c1/h9234
c2/h8809
[EOF=s100]

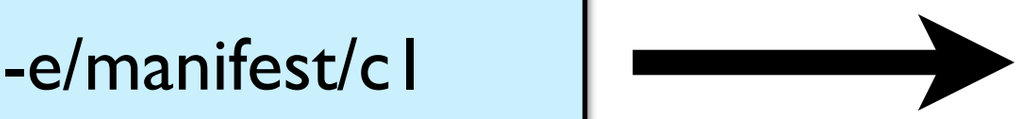
Request a list of segments for the movie
Each segment identified with a hash

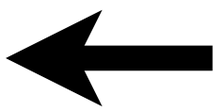
Request movie content



Request more manifest

/netflix/movies/Wall-e/manifest/c1



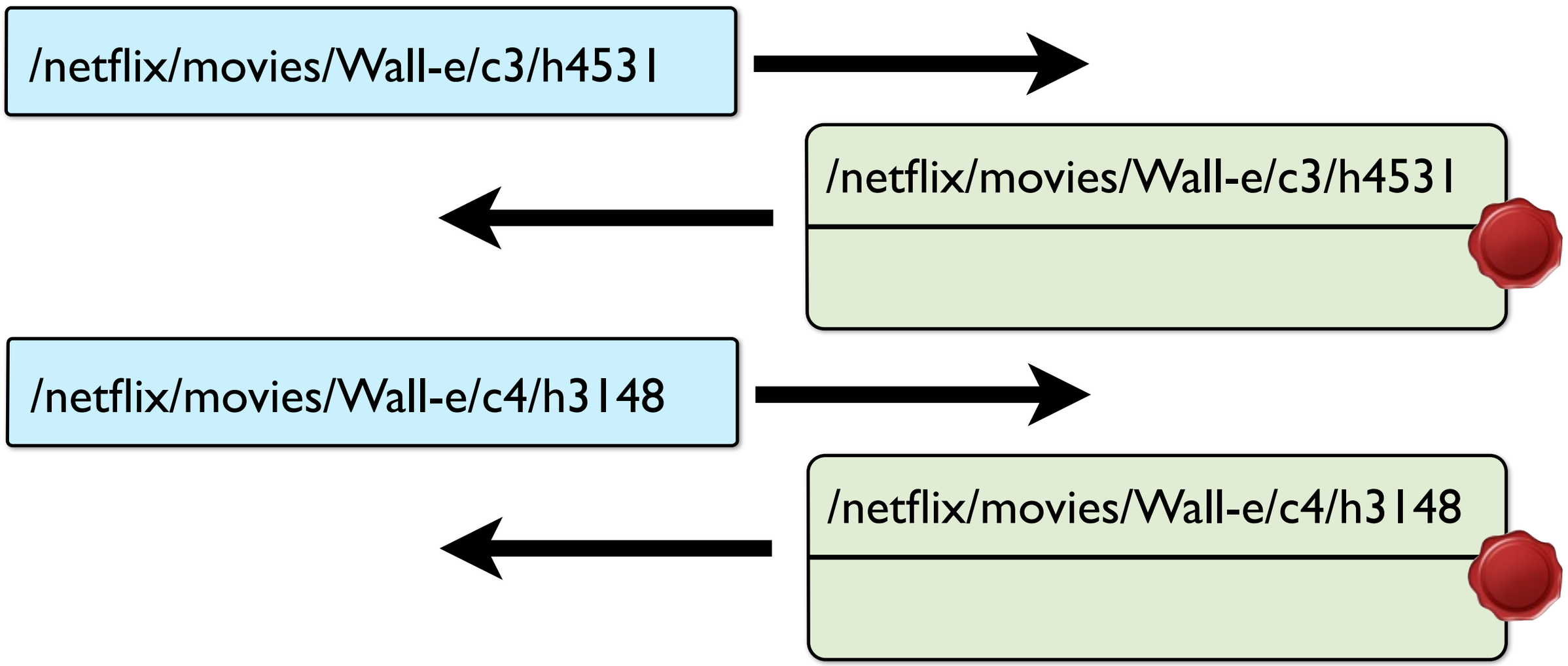


/netflix/movies/Wall-e/manifest/c1

c3/h4531
c4/h3148
c5/h1359
[EOF=s100]

Request a list of segments for the movie
Each segment identified with a hash

Request more movie content



Sample use case

Phone call

A simplified example

Phone Call

Step 1 - Setup call

Contact call agent at callee

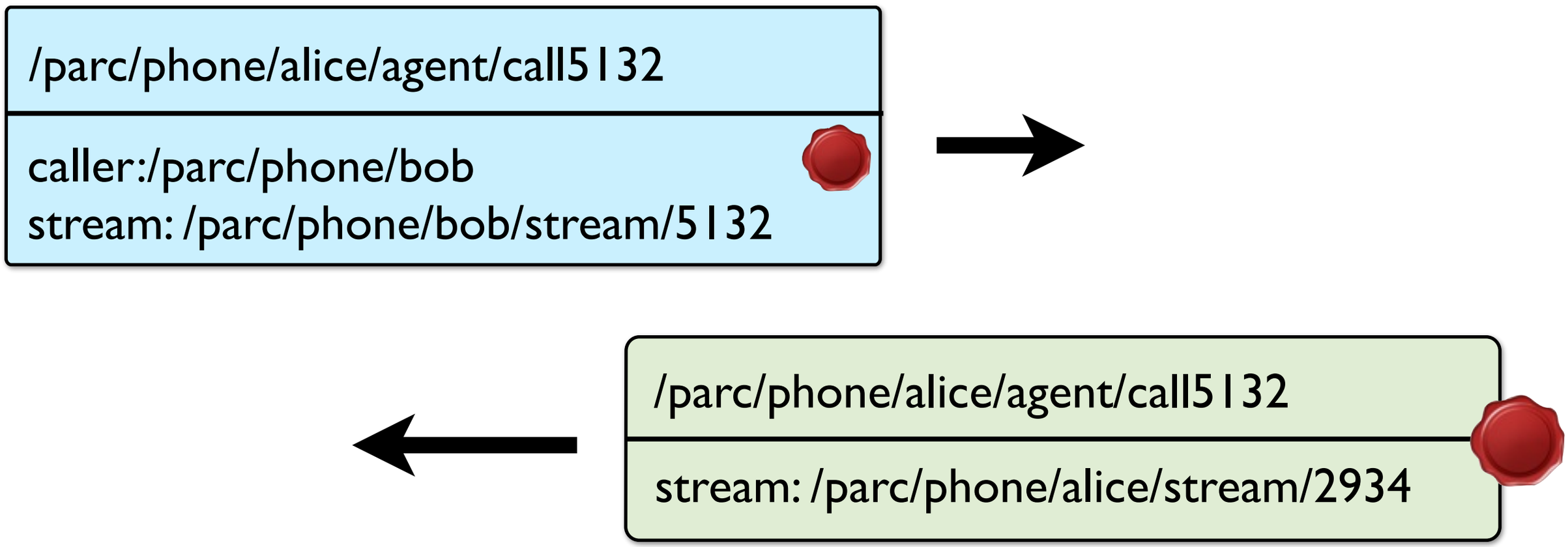
Step 2 - Request caller stream

Request audio stream from caller

Step 3 - Request callee stream

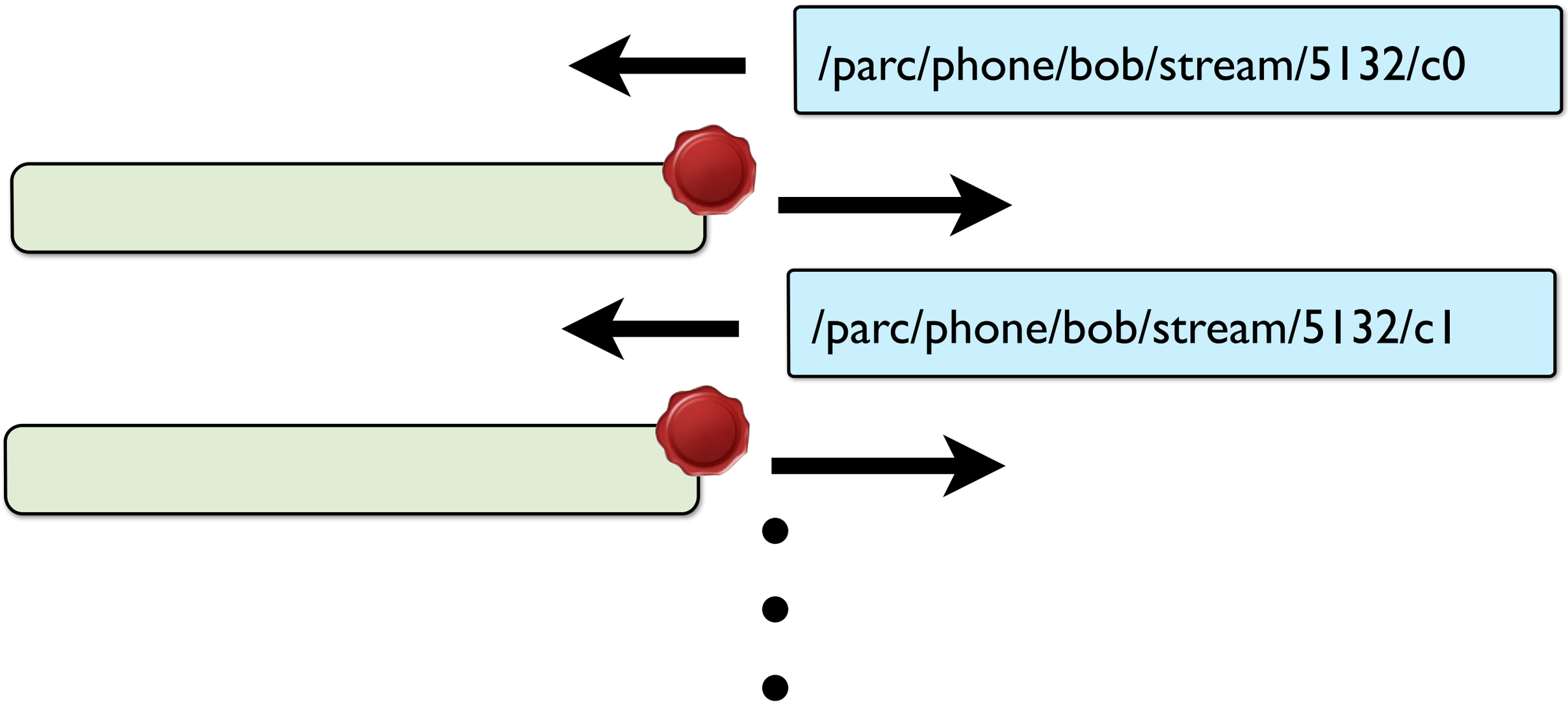
Request audio stream from callee

Setup call - Bob calls Alice



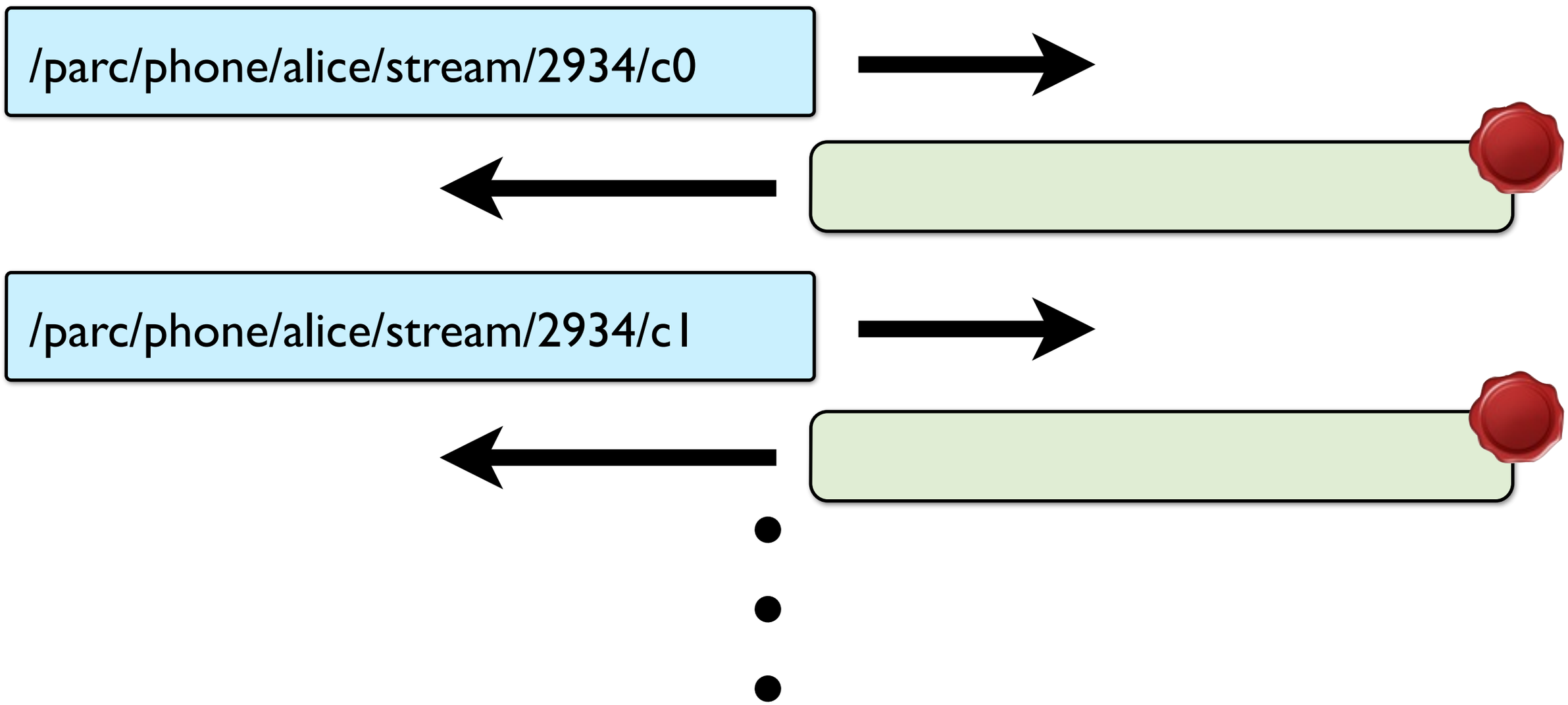
Setup a call from Bob to Alice. Bob tells Alice of the caller stream, Alice tells bob of the callee stream.

Request caller stream



Alice requests Bob's audio stream

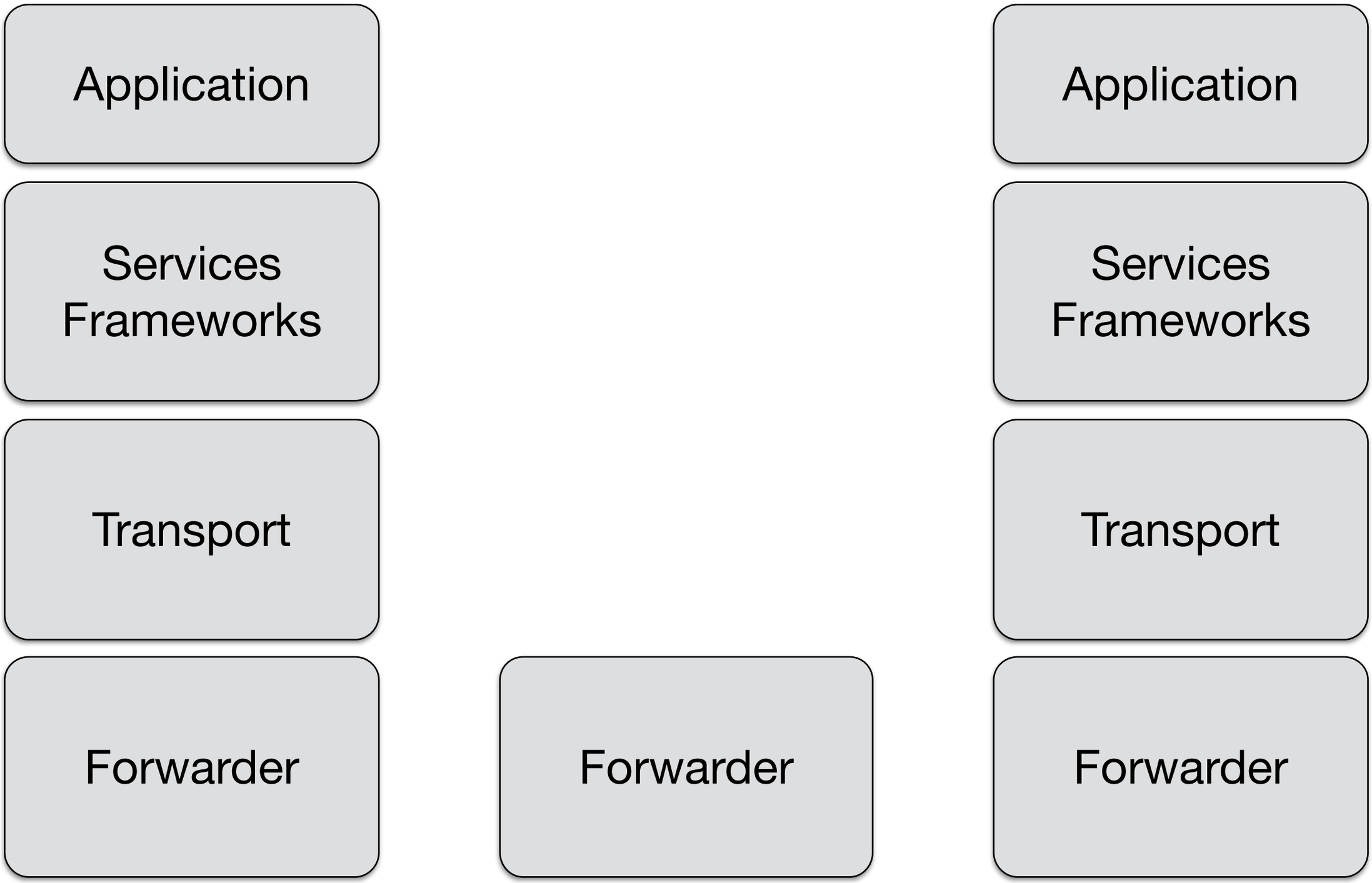
Request callee stream



Bob requests Alice's audio stream

CCN - Architecture

CCN System architecture



CCN System architecture

Application

Services
Frameworks

High level functions and network services
APIs: store, discovery, configuration, etc.

Transport

Host stack. Implements transport protocols. In charge of encoding, decoding, signing, etc.

Forwarder

Forwards packets between nodes. Required at every node. Runs the core protocol.

CCN - 1.x

Labeled Name

/N=ccnx/N=evolution/V=1/C=0

TLV Encoding

[Type (Length) Value]

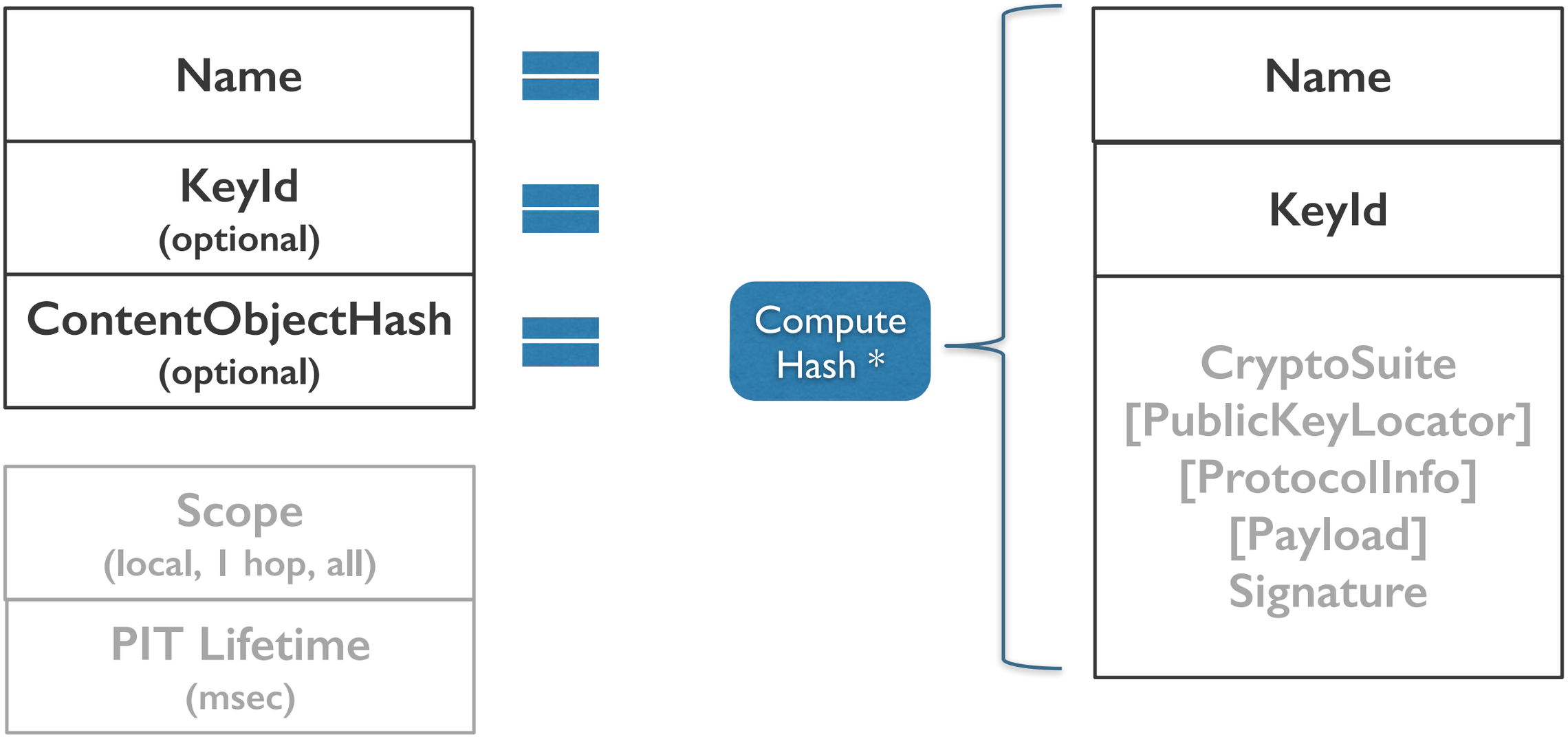
[Name = Alice]

Exact Matching

/ccnx/presentation/evolution/v1/c0

/ccnx/presentation/evolution/v1/c0

Core Protocol Primitives



Core Protocols

Everywhere

(Sensor, Servers, Spaceship)

Longest Prefix Match

FIB

Content Store

Optional
(Cache)

Discovery Protocol (Unbundled)

Chunking Protocol (Unbundled)

Versioning Protocol (Unbundled)

Label-Based Names

Discovery

Versioning

Chunking

Sync

Repo

Trust

CCN Core Protocol

Hash Forwarding

Fragmentation

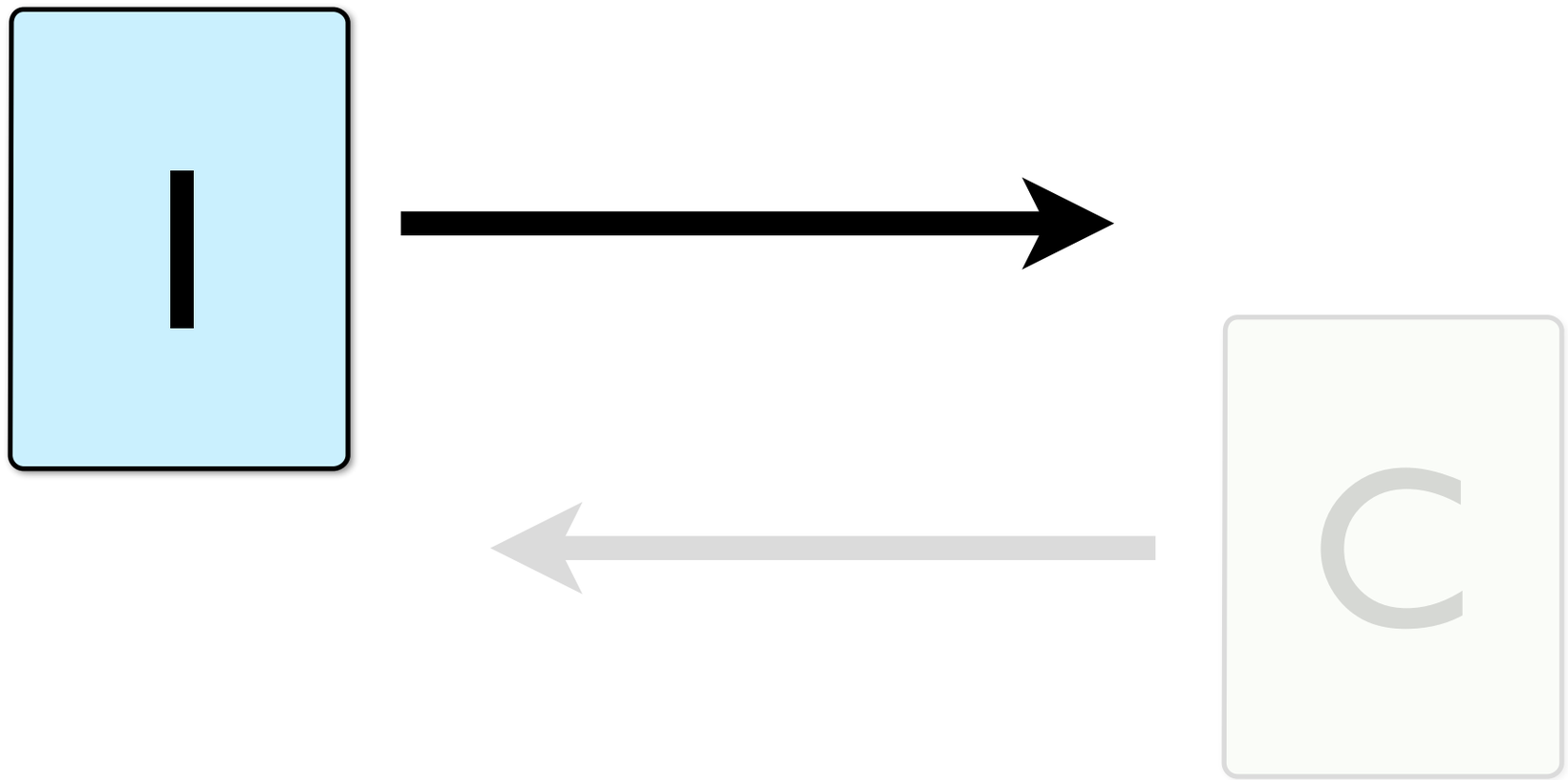
TLV Packet Format

Protocol Specifications

1. CCNx 1.0 Protocol Specification Roadmap
2. CCNx Semantics
3. TLV Packet Format
4. CCNx Messages in TLV Format
5. Labeled Segment URIs
6. Labeled Content Information URIs for CCNx
7. CCNx Content Object Caching
8. CCNx End-to-end Fragmentation
9. CCNx Content Object Segmentation
10. CCNx Publisher Clock Time Versioning
11. CCNx Publisher Serial Versioning
12. CCNx Selector Based Discovery
13. CCNx Hash Forwarding

CCN - Messages

Interest



Base CCN Message used to request Content Objects

Interest elements

Name

Hierarchical resource identifier. Used for routing and matching.

KeyId (optional)

Matching restriction based on a specific key

ContentObjectHash (optional)

Matching restriction based on a content hash

Interest elements

Scope (optional with default value)

Forwarding restriction based on network hops

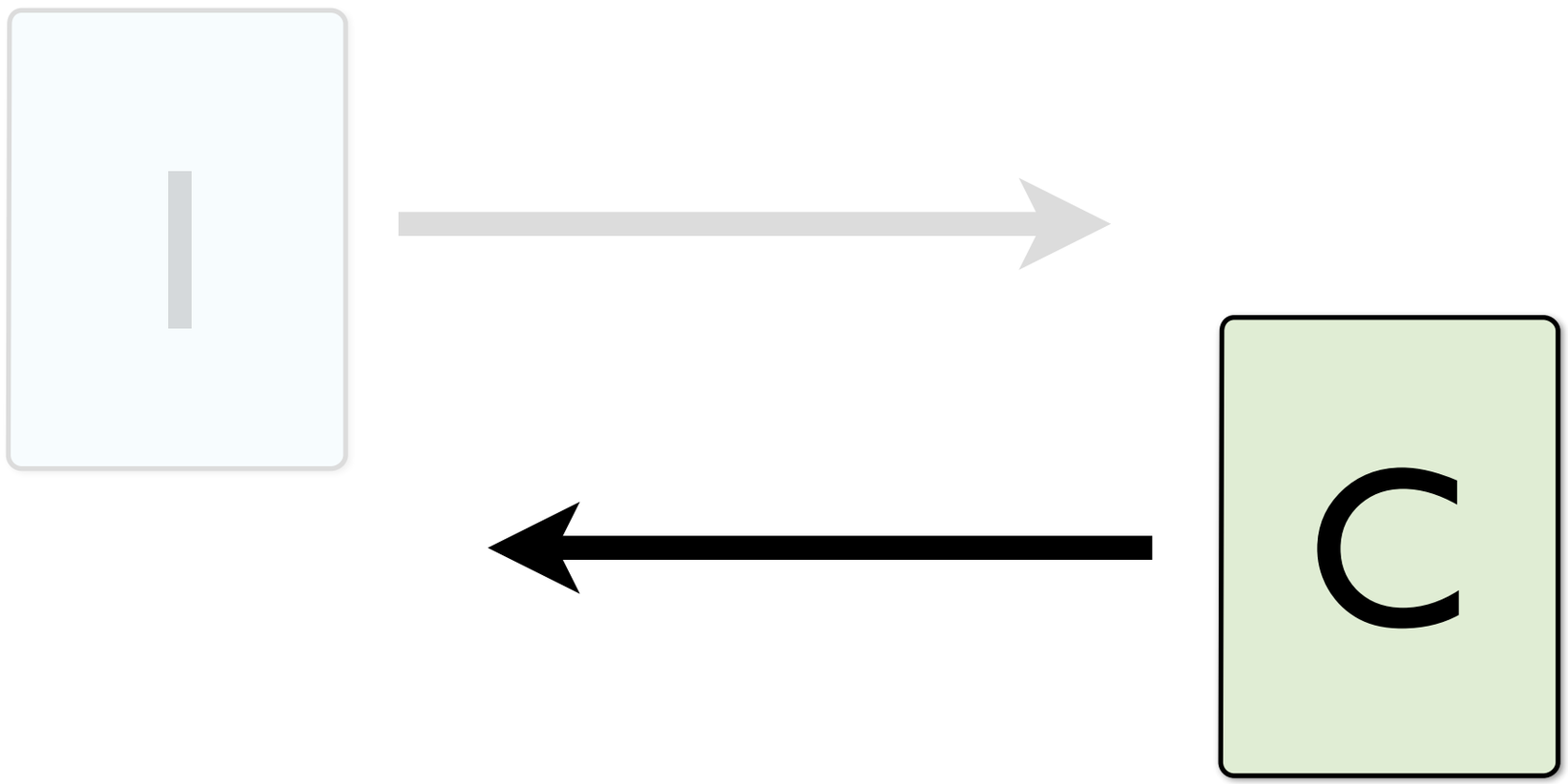
Lifetime (optional with default value)

Lifetime restriction of the Interest in-network.

Payload (optional)

Data

Content Object



Base CCN Message used to answer an Interest

Content Object elements

Name

Hierarchical resource identifier. Used for routing and matching.

Name Authenticator

KeyId of the signer

Crypto Suite used

Key locator (optional)

Content Object elements

Protocol Information (optional)

Structured information about the content object. Protocols store information here.

Payload

The data.

Signature Block

Cryptographic signature of the Content Object.

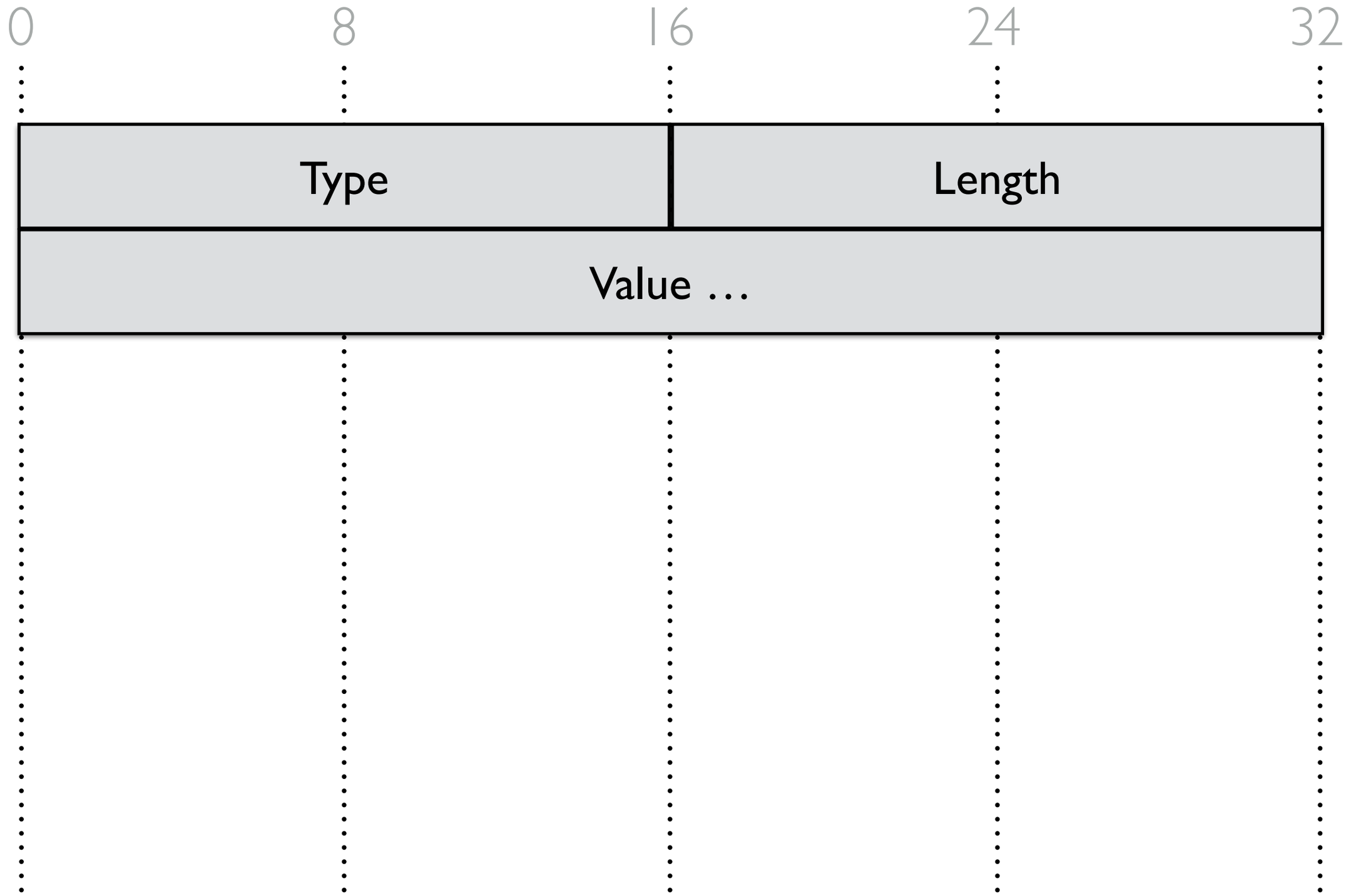
CCN - Encodings

CCN - Packets

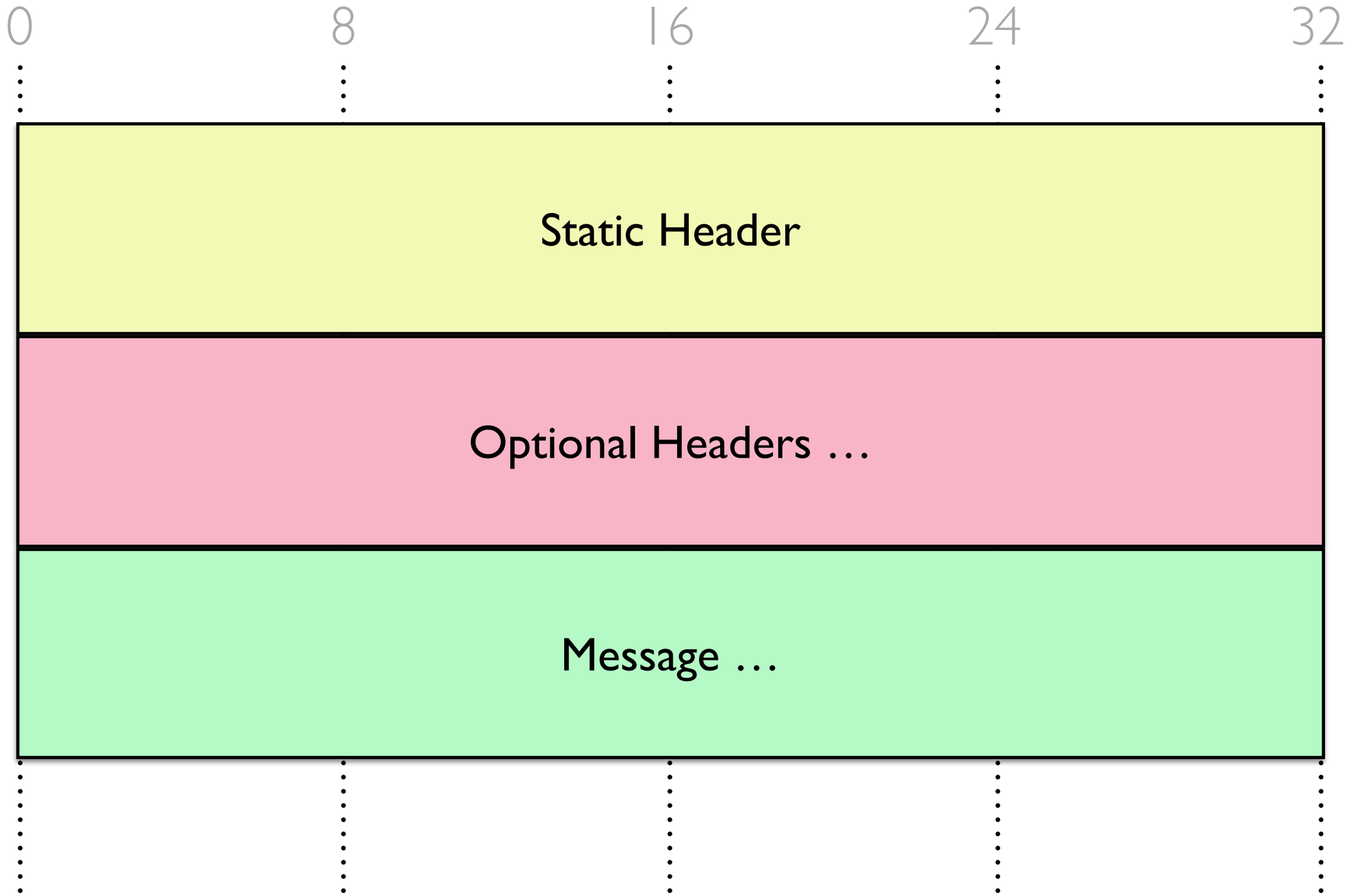
Type-Length-Value (TLV) encoding

Type	Length	Value …

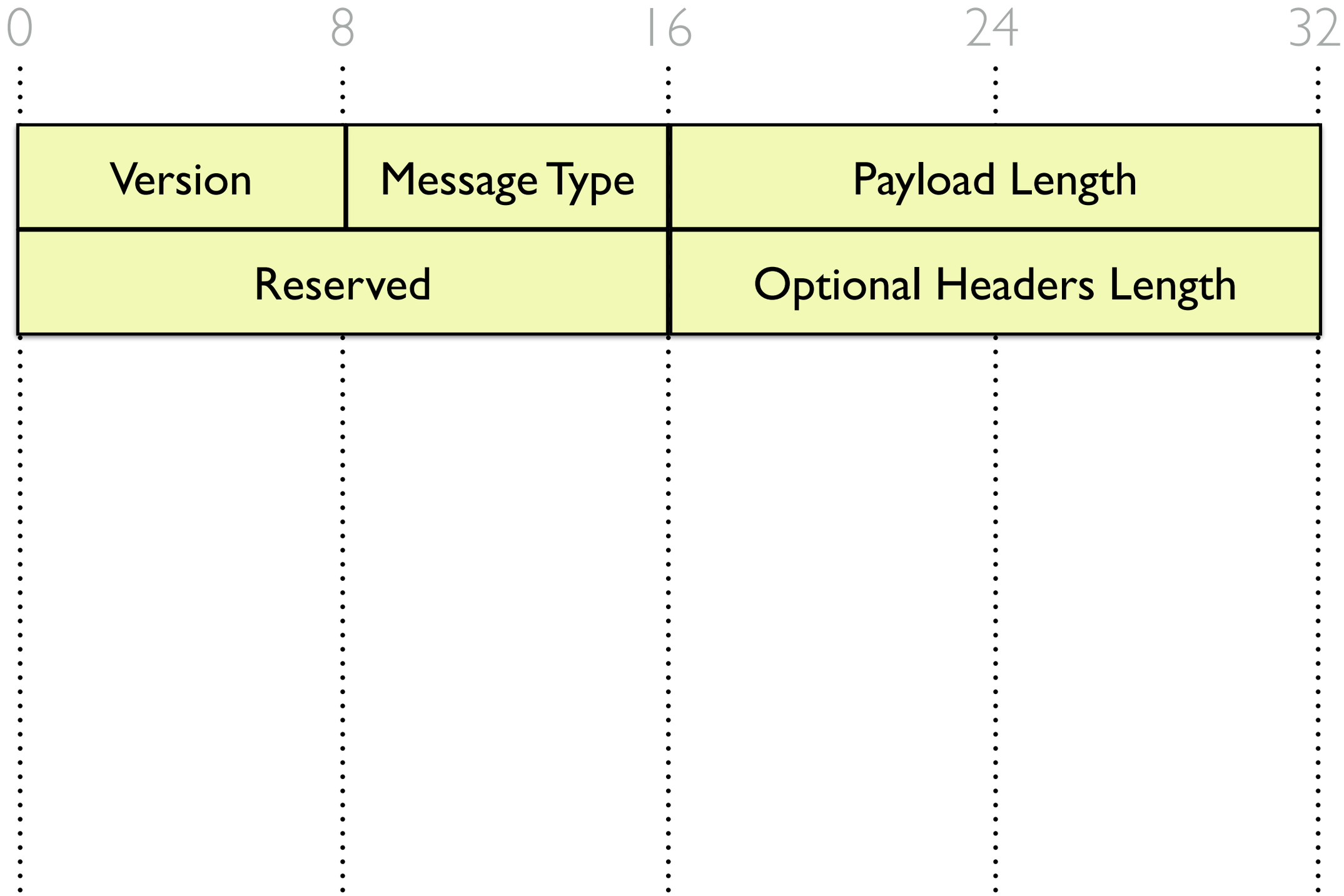
2 x 2 encoding



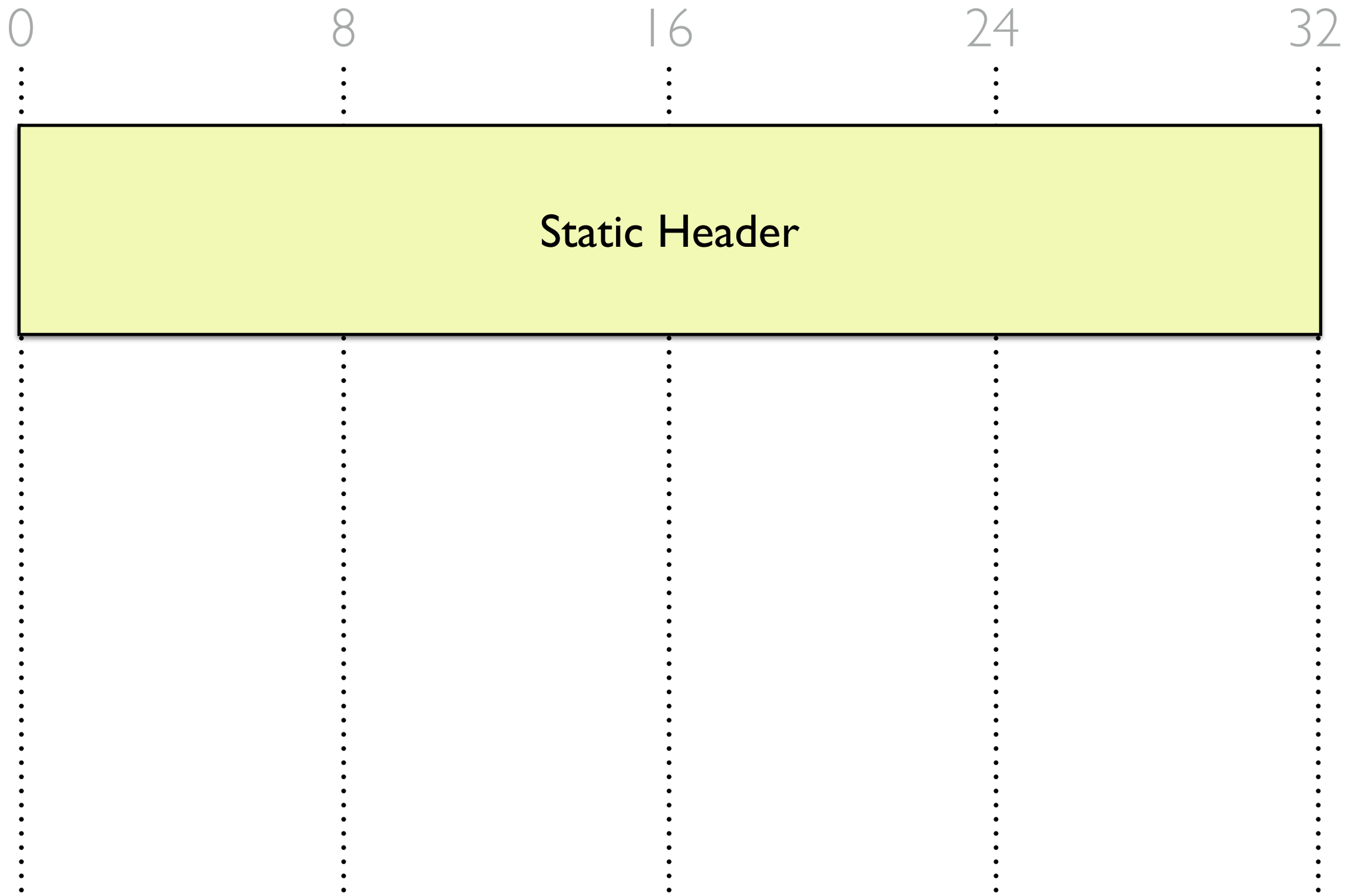
CCN Packet Structure



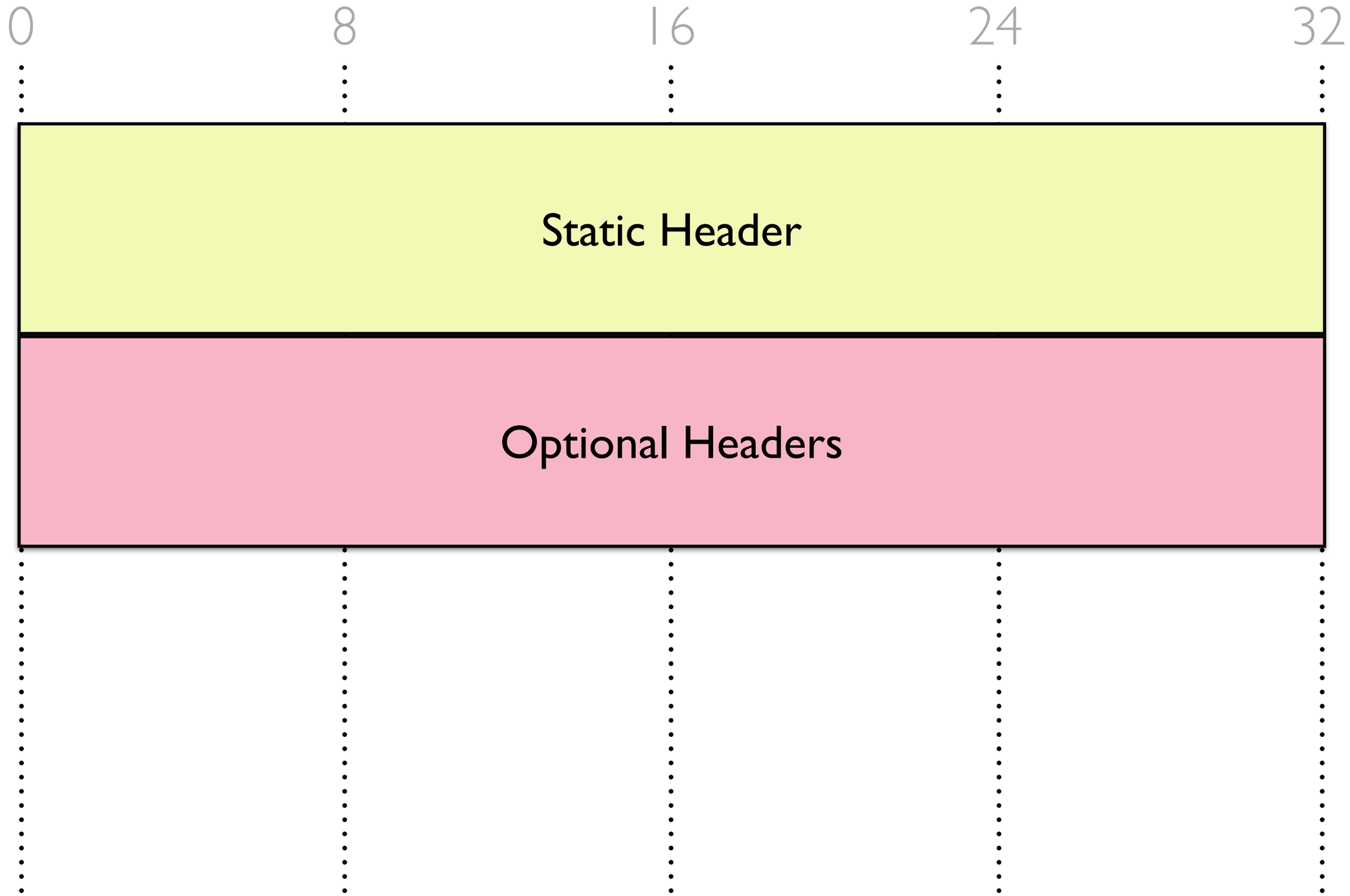
Static Header



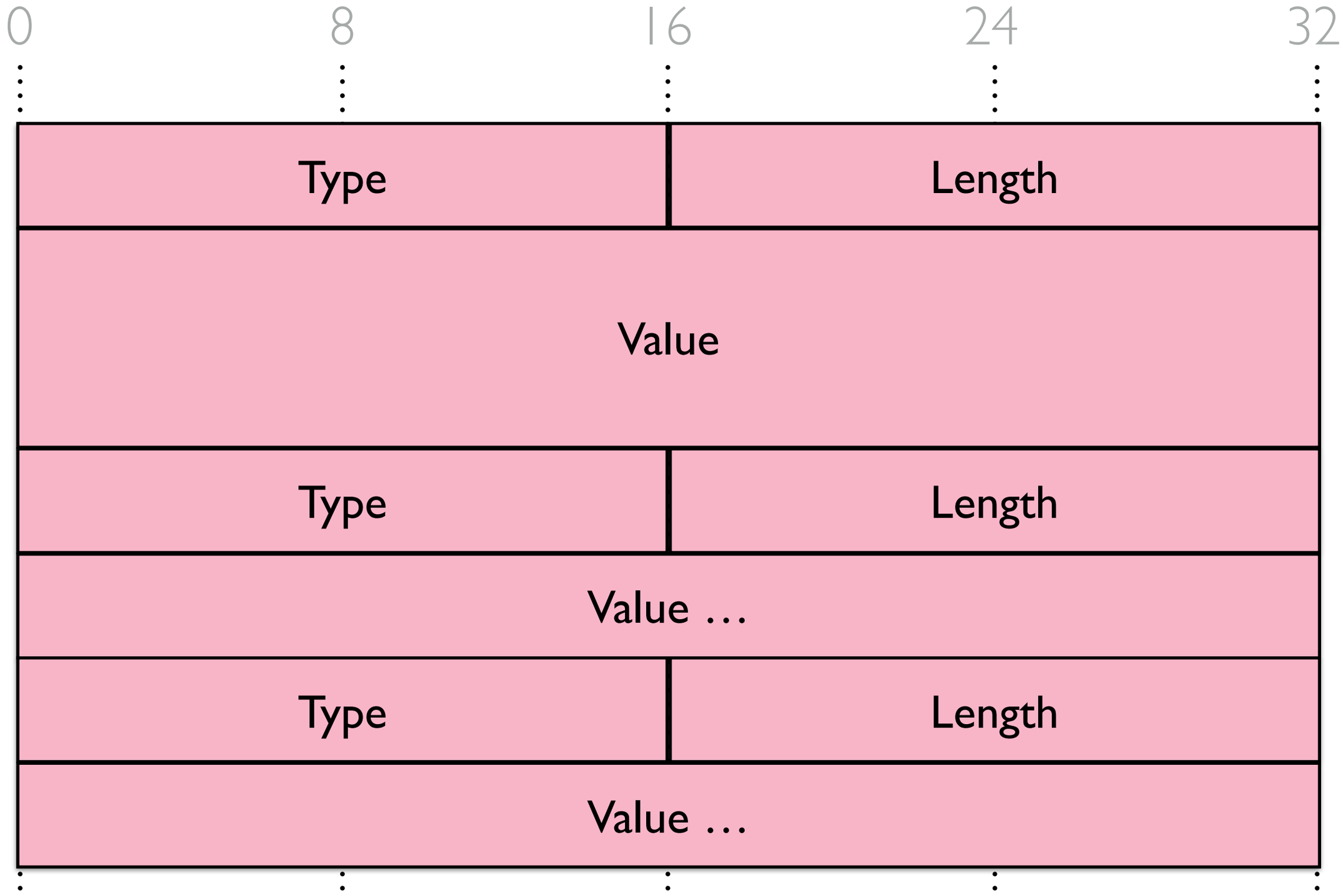
Static Header



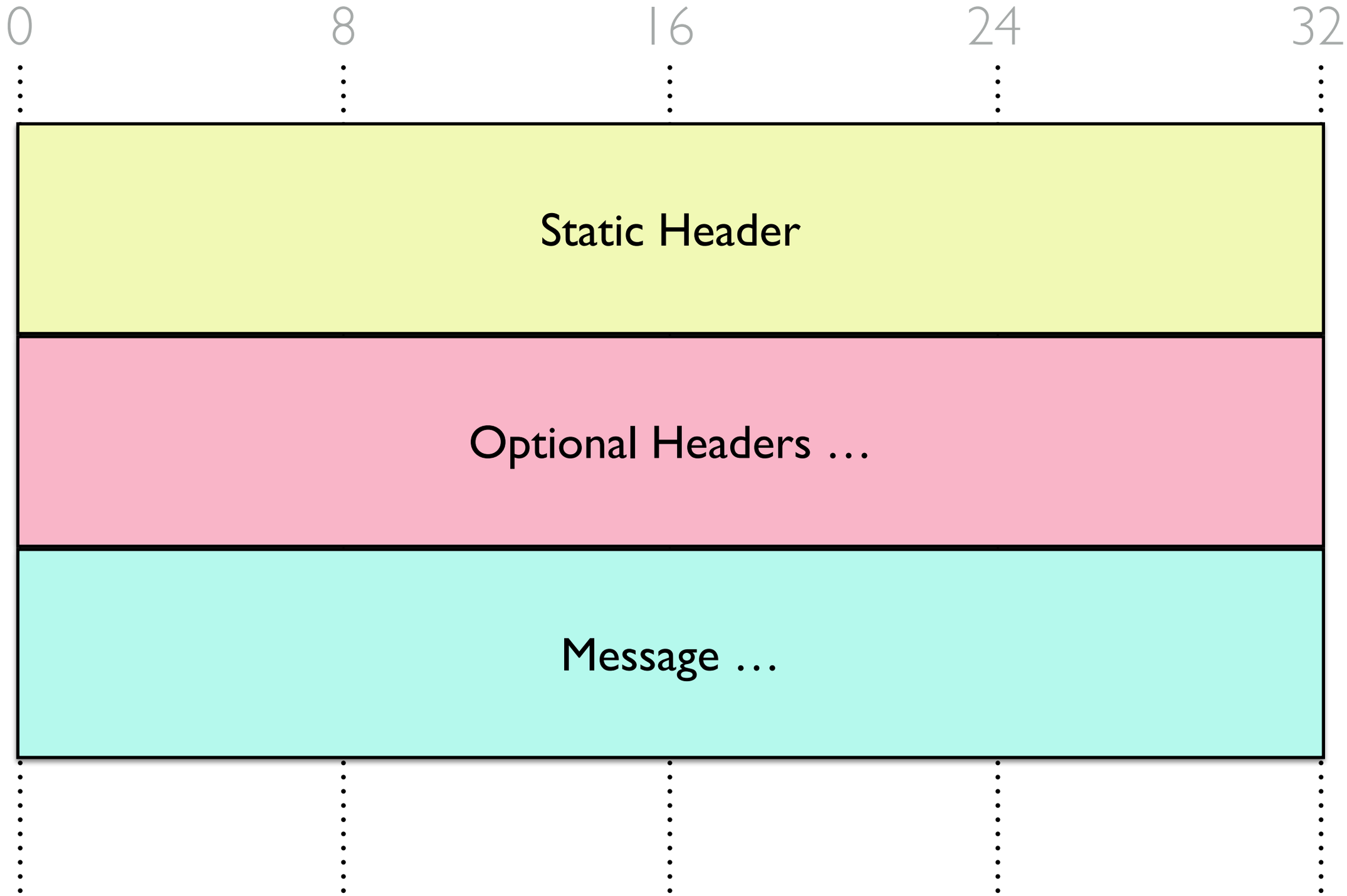
CCN Headers



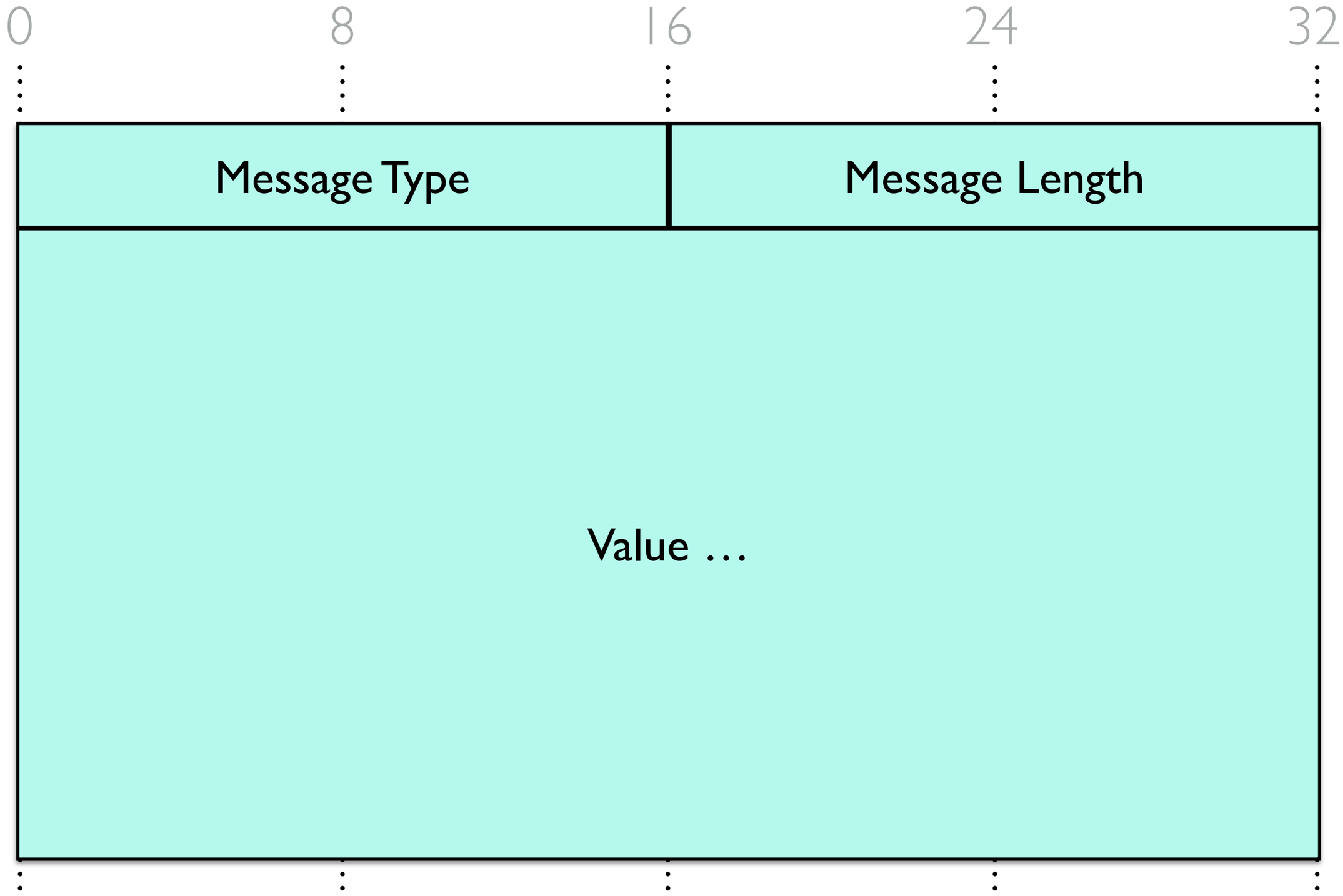
Optional Headers



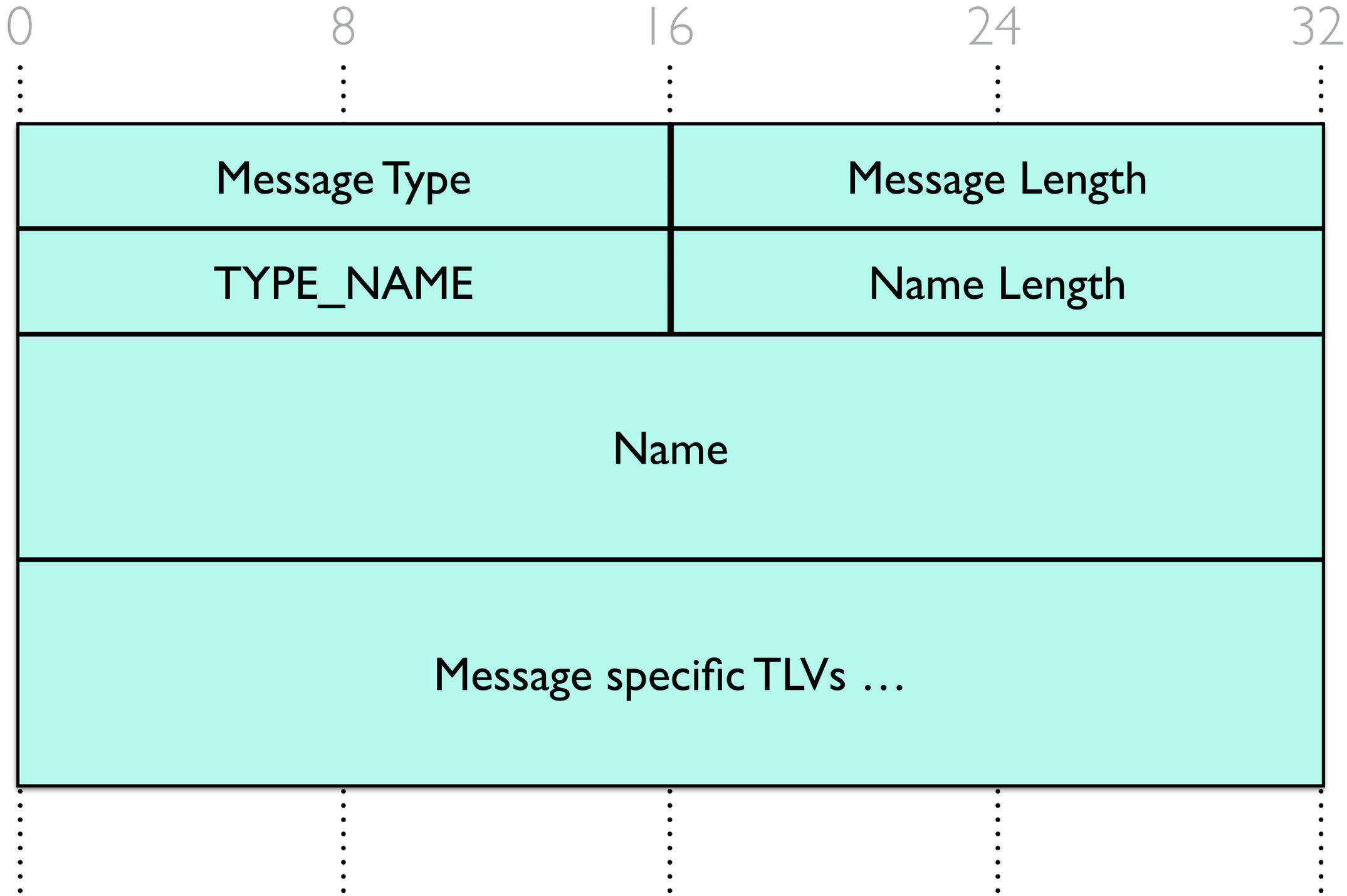
CCN Packet Structure



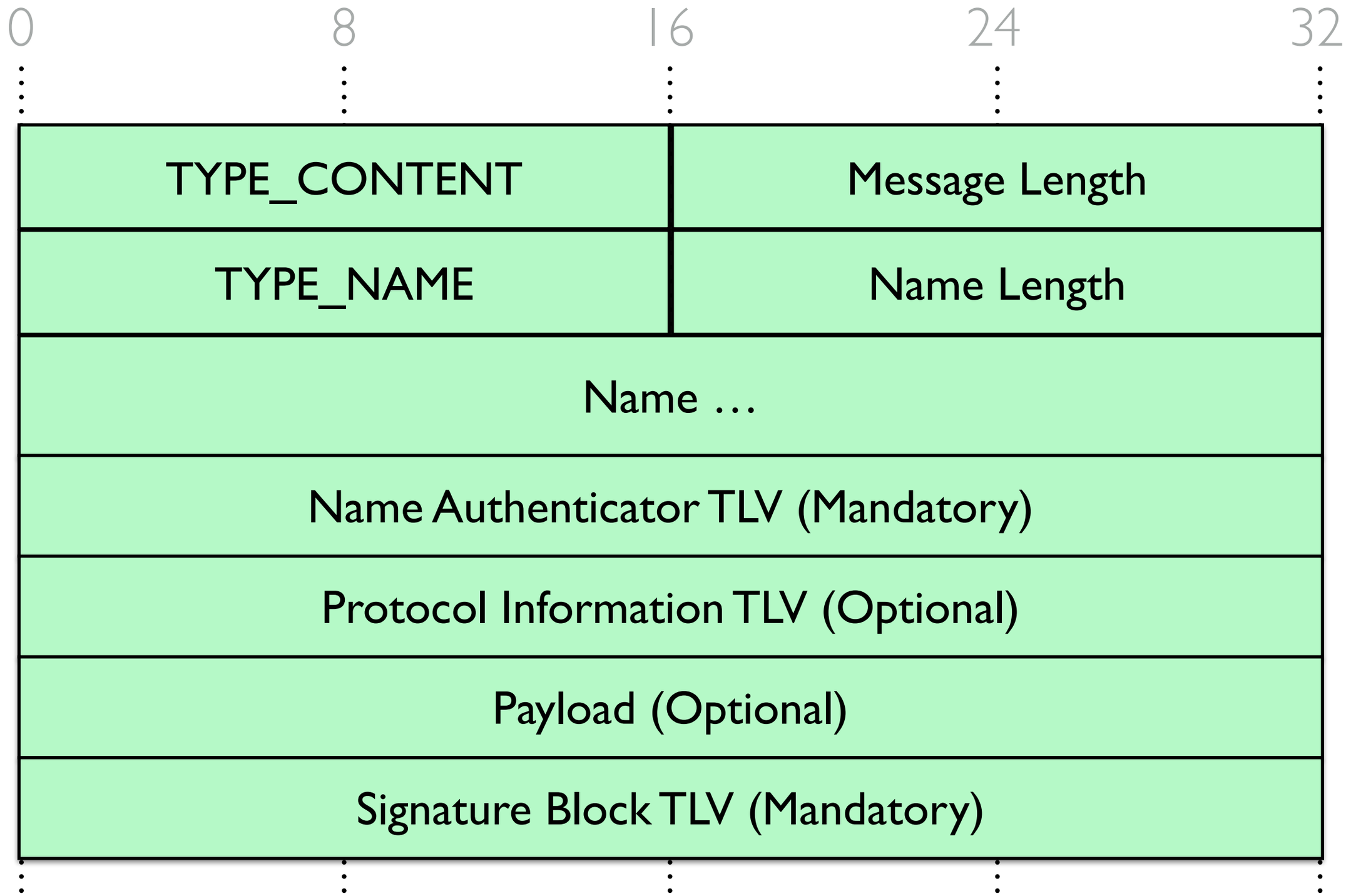
CCN Message



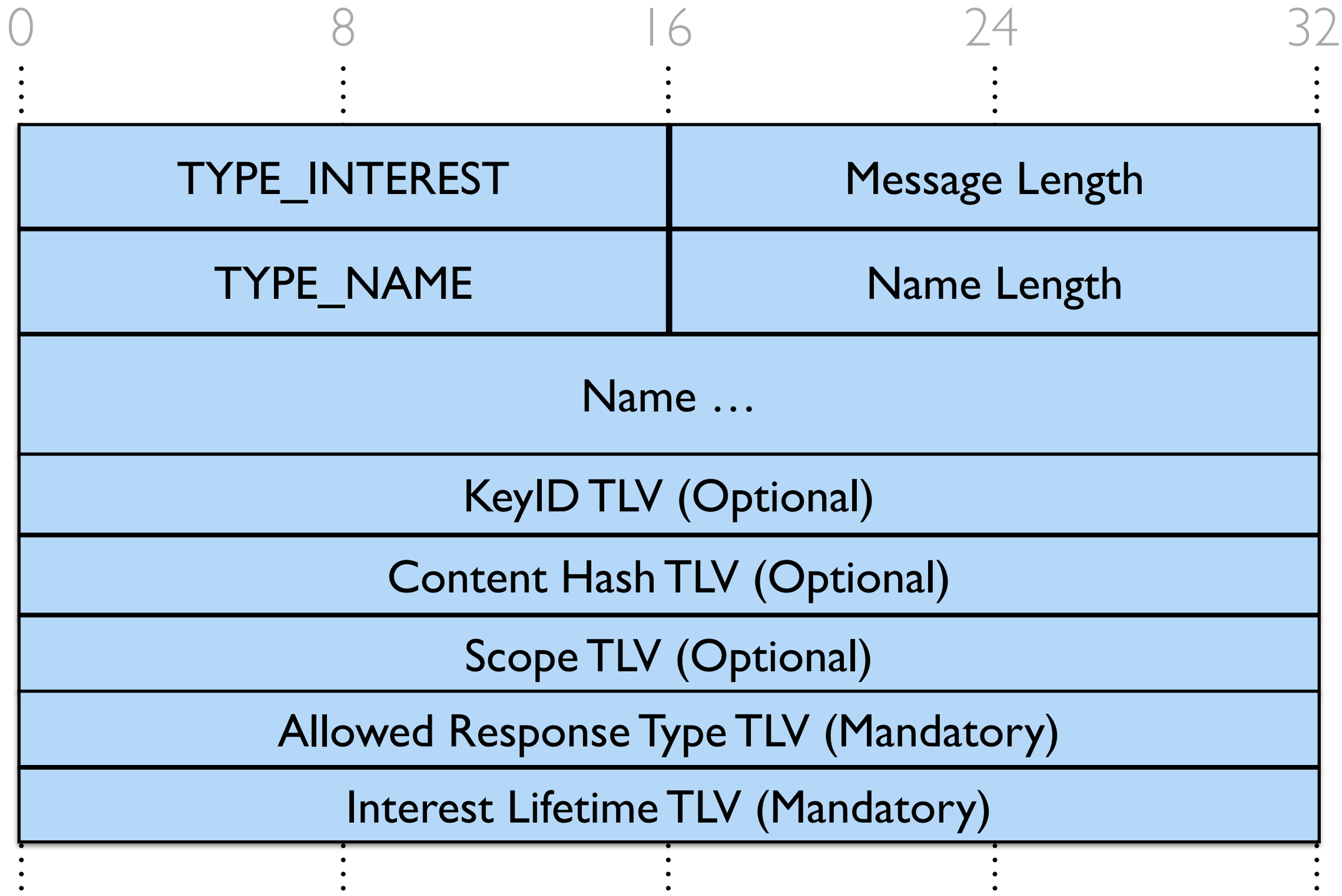
CCN Message



CCN Message



CCN Message



CCN - Names

CCN Name structure

/segment1/segment2/segment3

/parc/ccnx/presentation

/parc/ccnx/presentation/serial=2/

/parc/ccnx/presentation/serial=2/chunk=6

/parc/ccnx/presentation/serial=2/app:slide=80

CCN name segment structure

label=segment

CCN name segment structure

/ccn/presentation

/name=ccn/name=presentation

Default label = name

CCN name segment structure

chunk=6

serial=6

Chunk is used for chunked objects
Serial is used for versioning

CCN name segment structure

app:<param>=2014

app:year=2014

Application specific labels used by apps

Labeled Content Identifiers (CCN URIs)

```
lci:/parc/ccn/spec/serial=3/chunk=2
```

Labeled Content Identifiers

```
lci:/parc/ccn/spec/serial=3/chunk=2
lci:/name=parc/name=ccn/name=spec/serial=3/chunk=2
```

Labeled Content Identifiers

```
lci:/parc/ccn/spec/serial=3/chunk=2
lci:/name=parc/name=ccn/name=spec/serial=3/chunk=2
lci:/N=parc/N=ccn/N=spec/V=3/C=2
```

Labeled Content Identifiers

```
lci:/parc/ccn/spec/serial=3/chunk=2
lci:/name=parc/name=ccn/name=spec/serial=3/chunk=2
lci:/N=parc/N=ccn/N=spec/V=3/C=2
lci:/0x0010=parc/0x0010=ccn/0x0010=spec/0x00A1=3/0x00A3=2
```

Labeled Content Identifiers

```
lci:/parc/pics/spec/app:year=2014
```

Labeled Content Identifiers

```
lci:/parc/pics/spec/app:year=2014
lci:/name=parc/name=pics/app:year=2014/app:03=bob
```

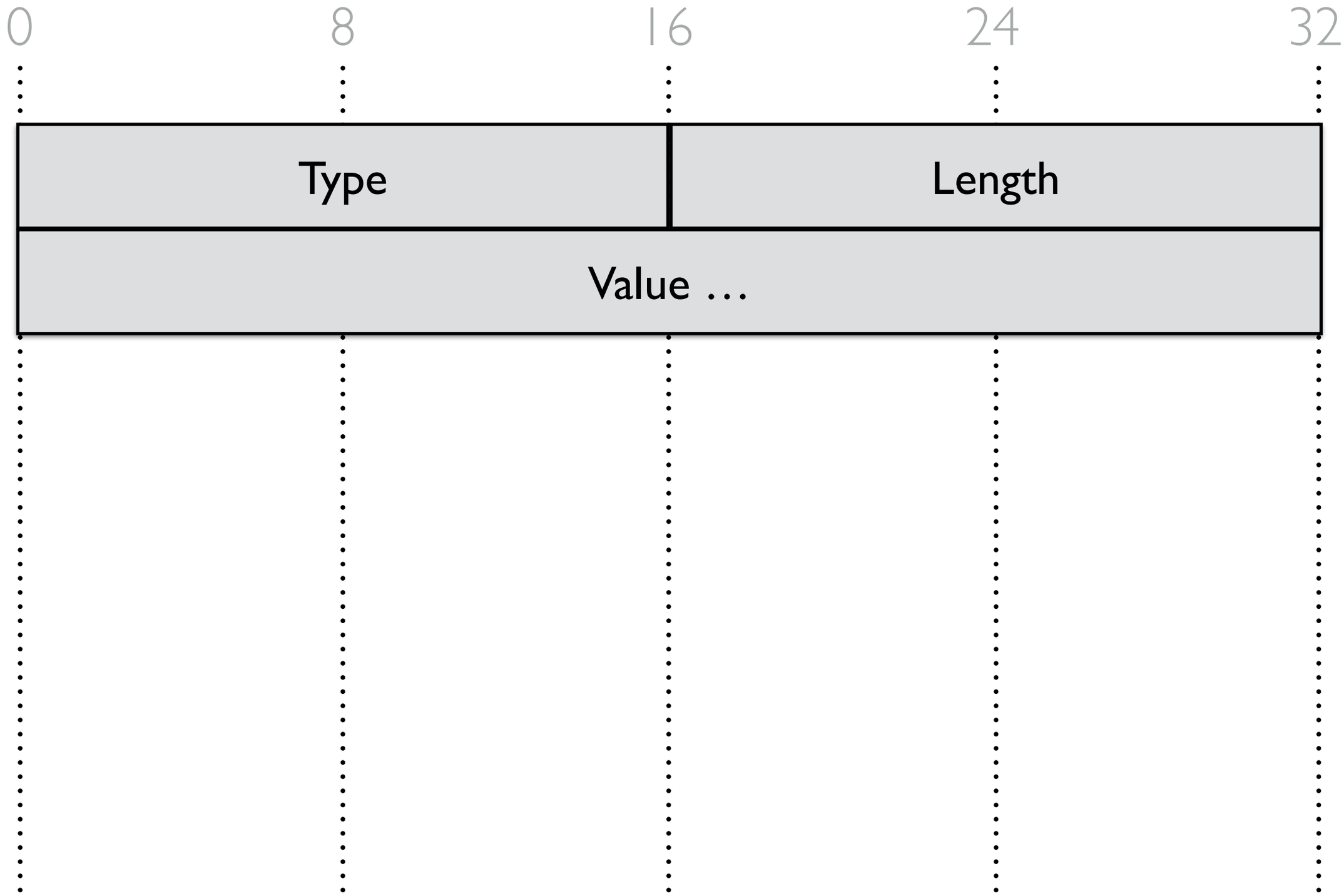
Labeled Content Identifiers

```
lci:/parc/pics/spec/app:year=2014
lci:/name=parc/name=pics/app:year=2014/app:03=bob
lci:/0x0010=parc/0x0010=pics/0xFF01=2014/0xFF03=bob
```

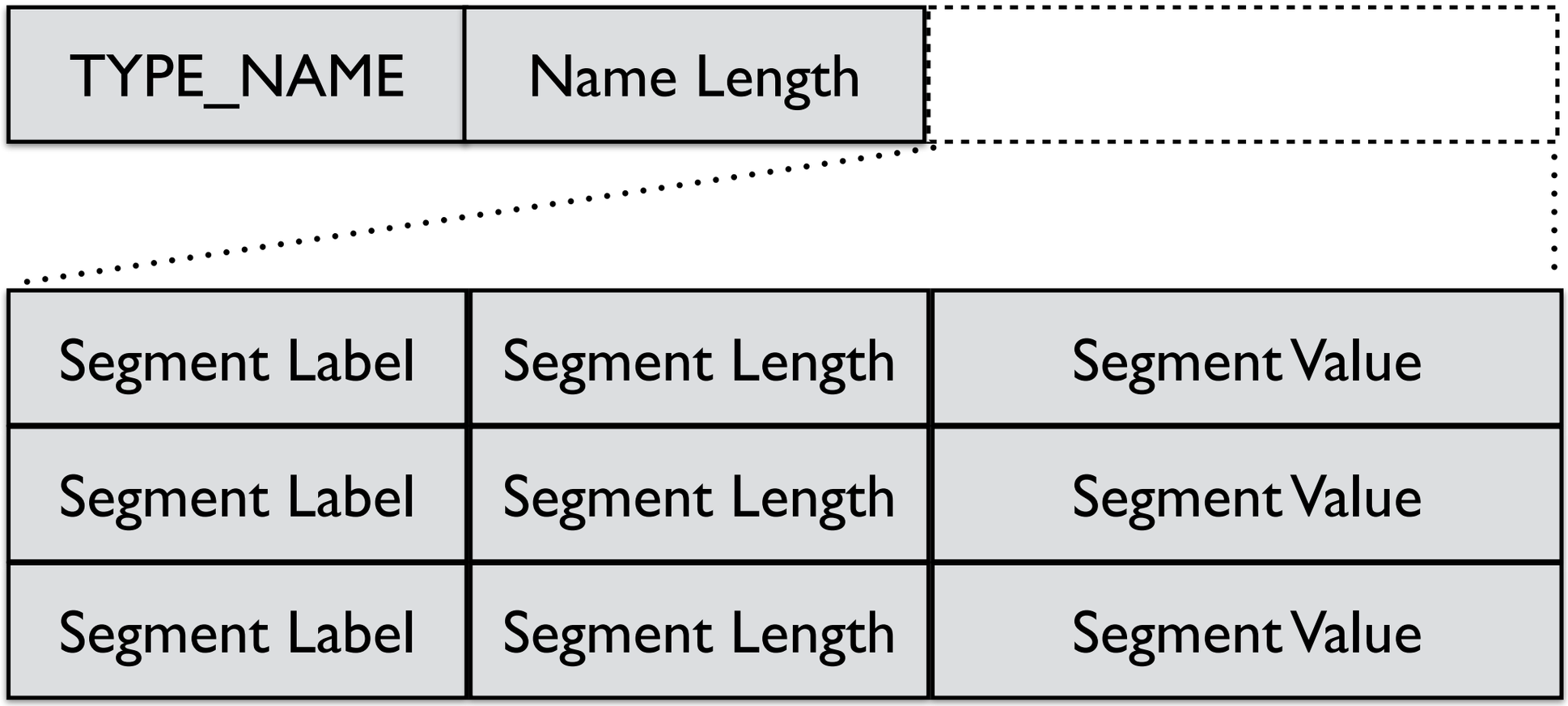
lci:/ scheme based off of URIs (RFC 3986)

```
LS-URI            = scheme ":" ls-hier-part ["?" ls-query]
                    ["#" fragment]
ls-hier-part      = ["//" authority] ls-path-absolute
ls-path-absolute  = "/" [ ls-segment *( "/" ls-segment ) ]
ls-segment        = lpv-segment / v-segment
lpv-segment       = label [":" param] "=" s-value
v-segment         = s-value
label             = alpha-t / num-t
param             = alpha-t / num-t
s-value           = *(s-pchar)

ls-query          = *1 ( lpv-component / v-component
                         *( "&" (lpv-component / v-component) ) )
lpv-component     = label [":" param] "=" q-value
v-component       = q-value
q-value           = *(q-pchar)

alpha-t           = ALPHA *(ALPHA / DIGIT)
num-t             = dec-t / hex-t
dec-t             = 1*(DIGIT)
hex-t             = "0x" 1*(HEXDIG)
ls-pchar          =  unreserved / pct-encoded / ls-sub-delims
s-pchar           = ls-pchar / ":" / "@" / "&"
q-pchar           = ls-pchar / ":" / "@" / "/"
ls-sub-delims     = "!" / "$" / "'" / "(" / ")"
                        / "*" / "+" / "," / ";"
```

CCN name TLV encoding

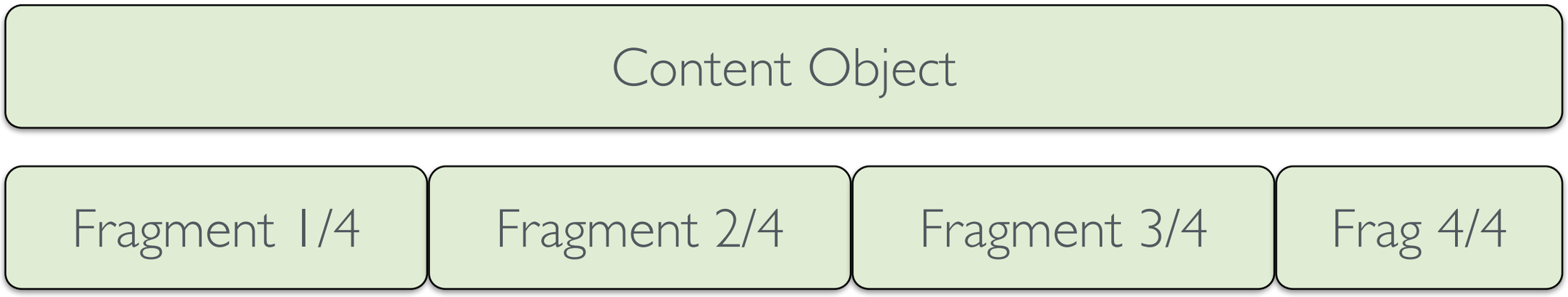


CCN name TLV encoding



CCN - Fragmentation

Fragmentation



Content Objects can be up to 64KB in size

To transmit Content Objects over network links with smaller MTU (Maximum Transmission Unit) size we need to fragment Content Objects

Fragmentation Types

Hop-by-Hop

Link specific, not part of spec

Mid-to-End

Not supported

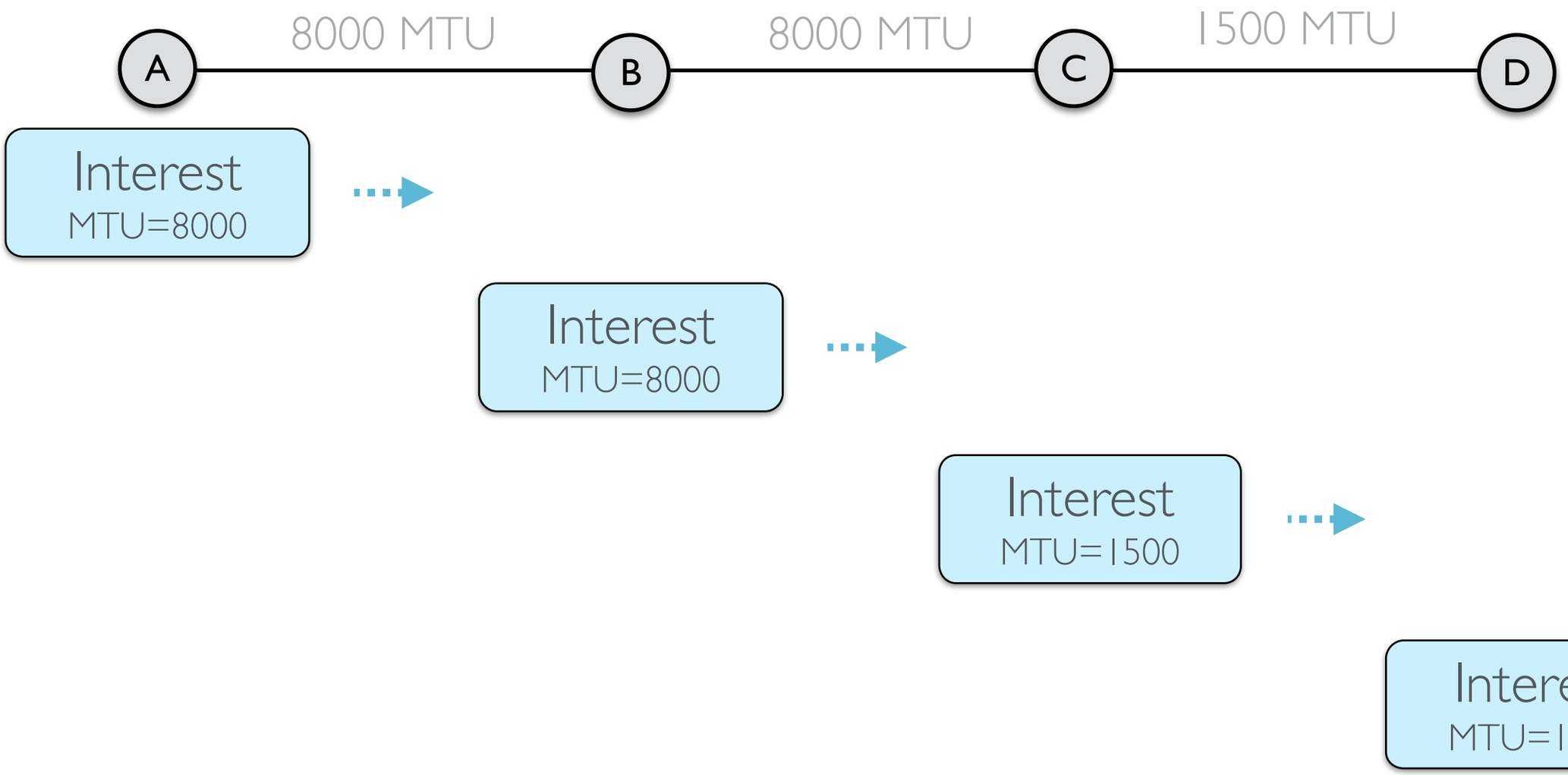
End-to-End

Interest discovers MTU

None

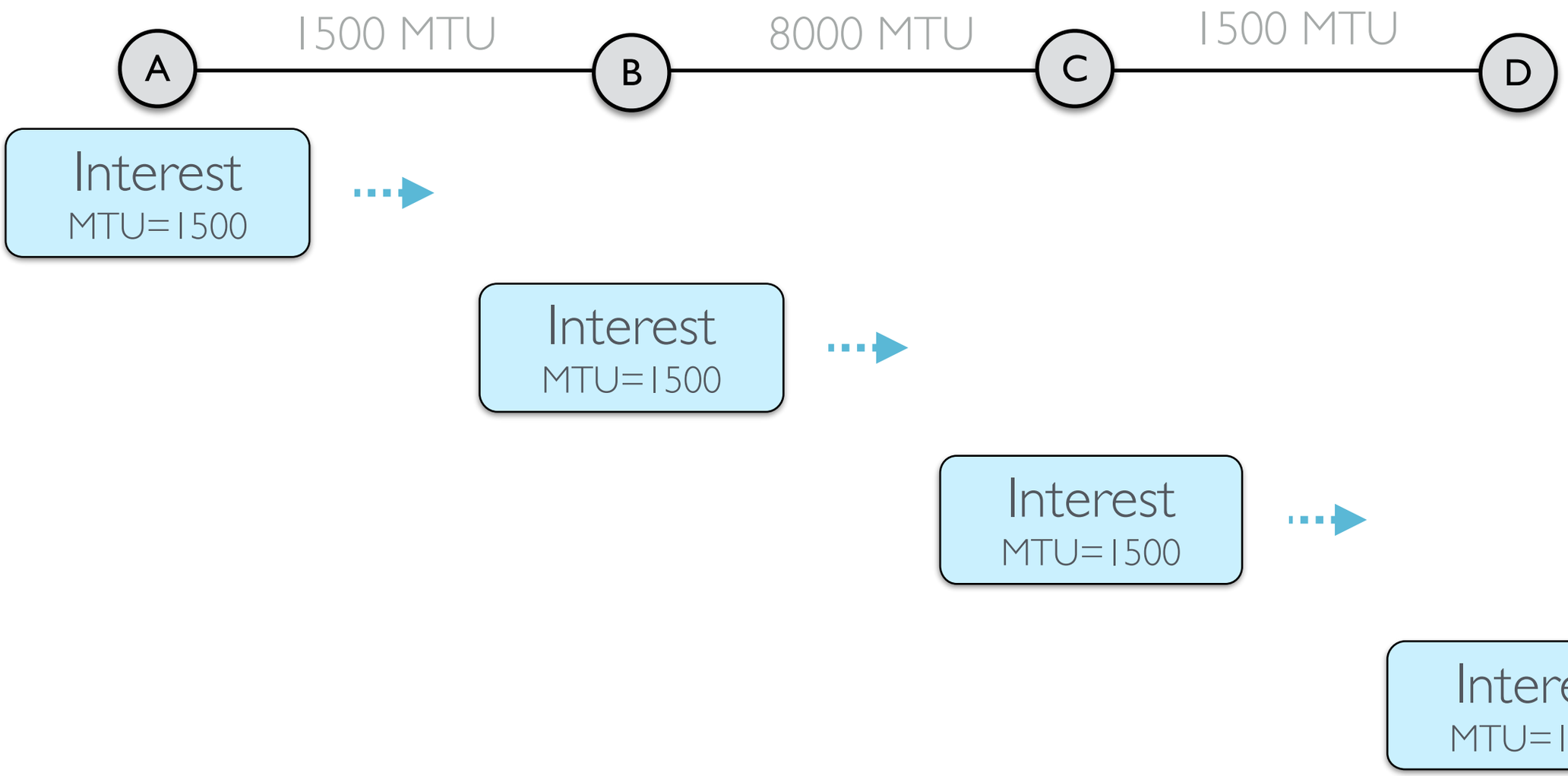
Preferred mode, potentially more efficient

End-to-End Fragmentation



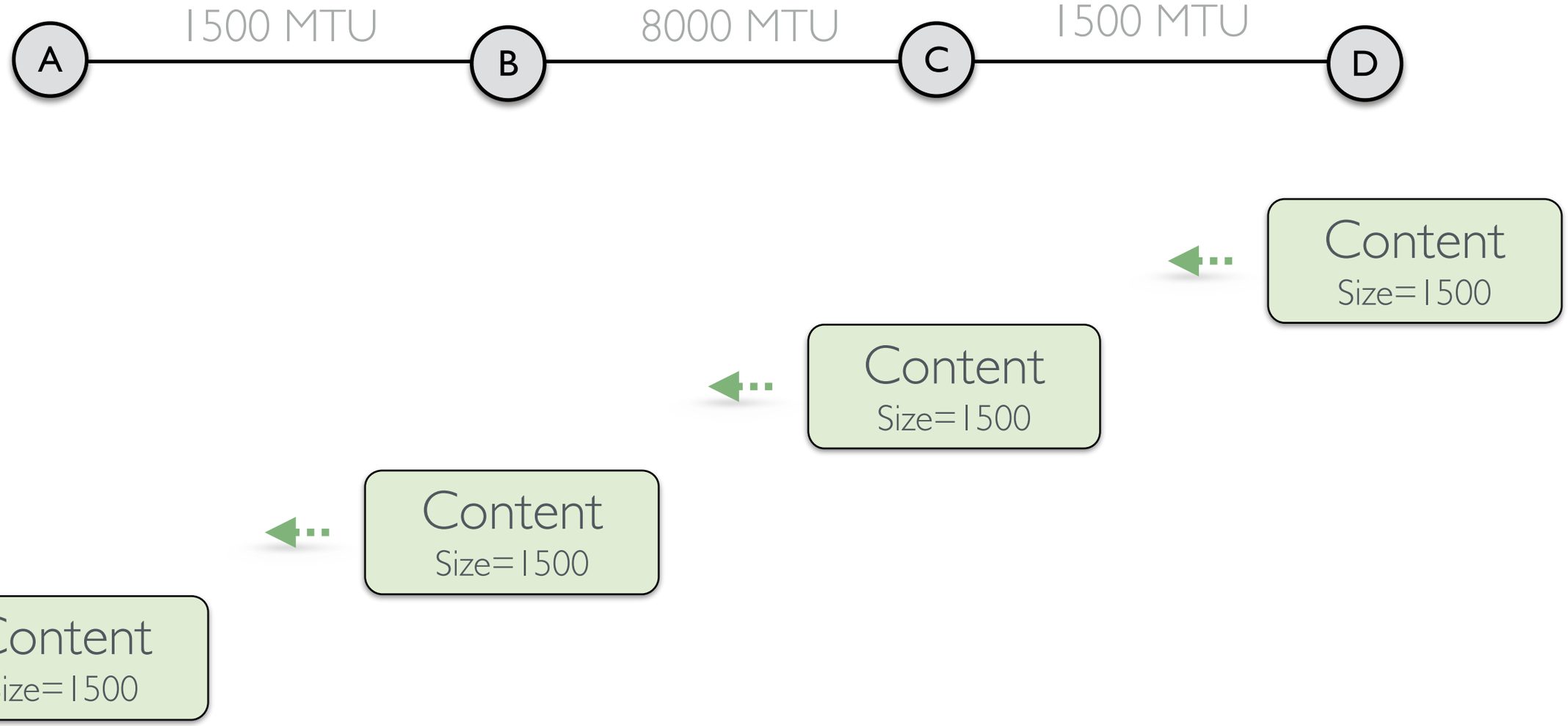
Interests discover the MTU

End-to-End Fragmentation



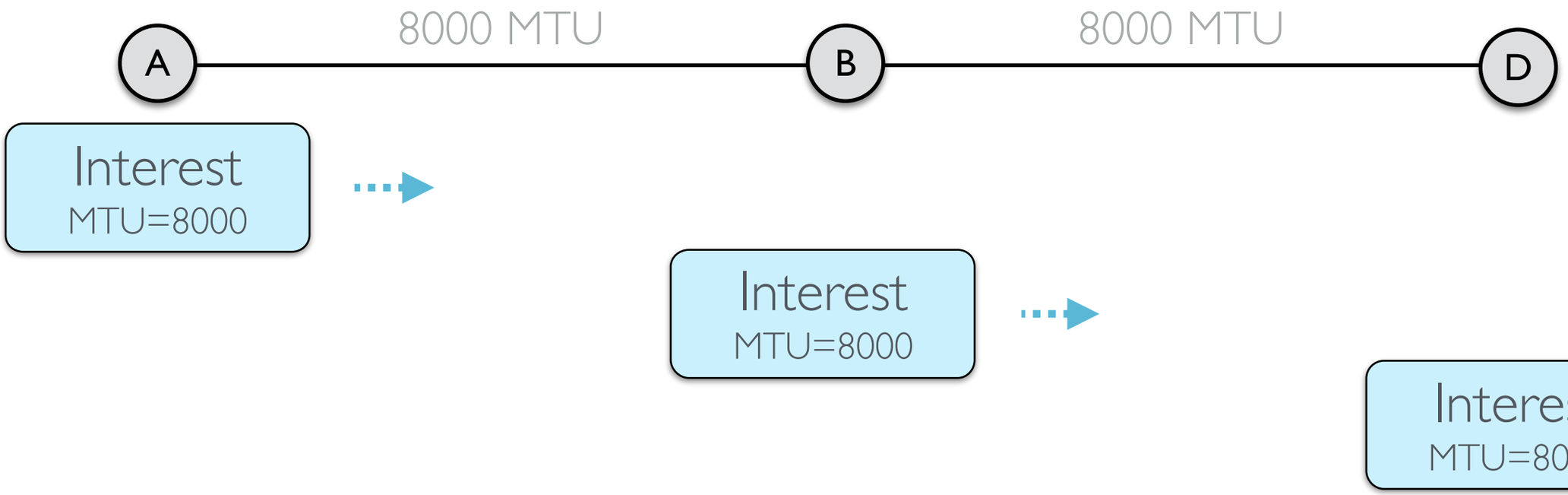
Interests are fragmented to minimum MTU
First Interest fragment must be routable

End-to-End Fragmentation



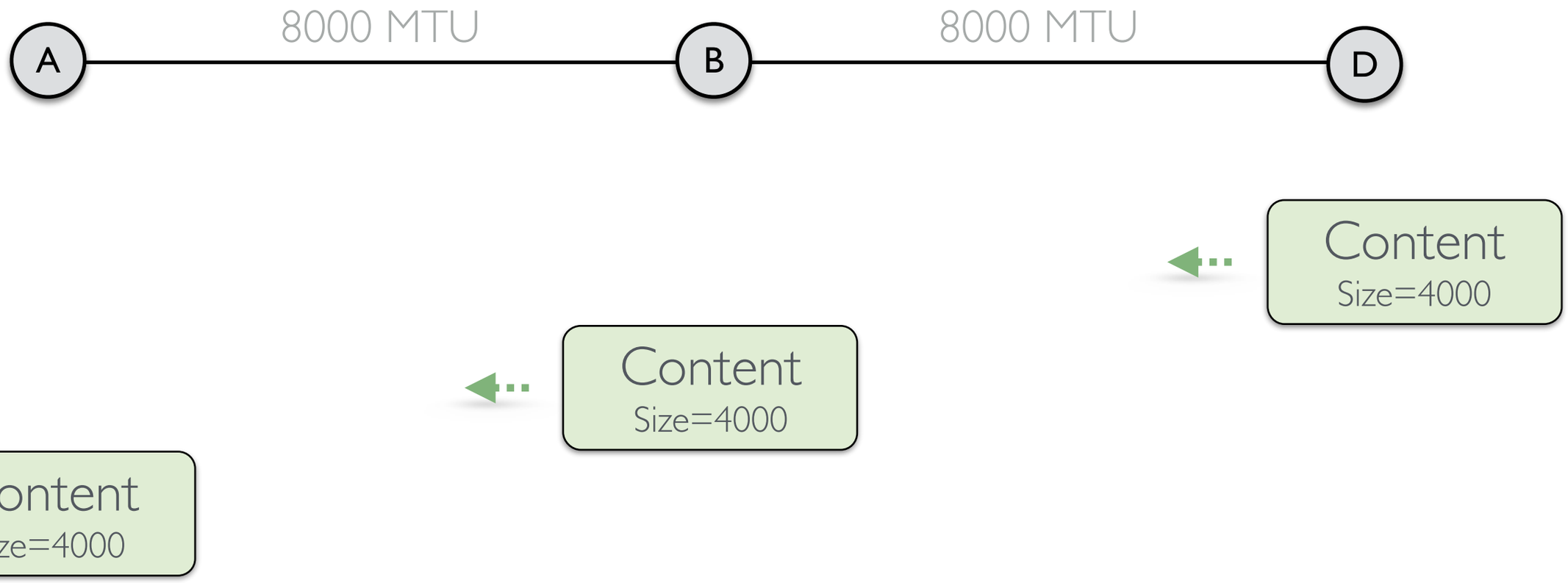
Content is fragmented to discovered MTU

End-to-End Fragmentation



Local networks might have large MTU

End-to-End Fragmentation



Reply might be fragmented smaller than MTU discovered

End-to-End Fragmentation Characteristics

Interests fragmented to minimum MTU at Requester

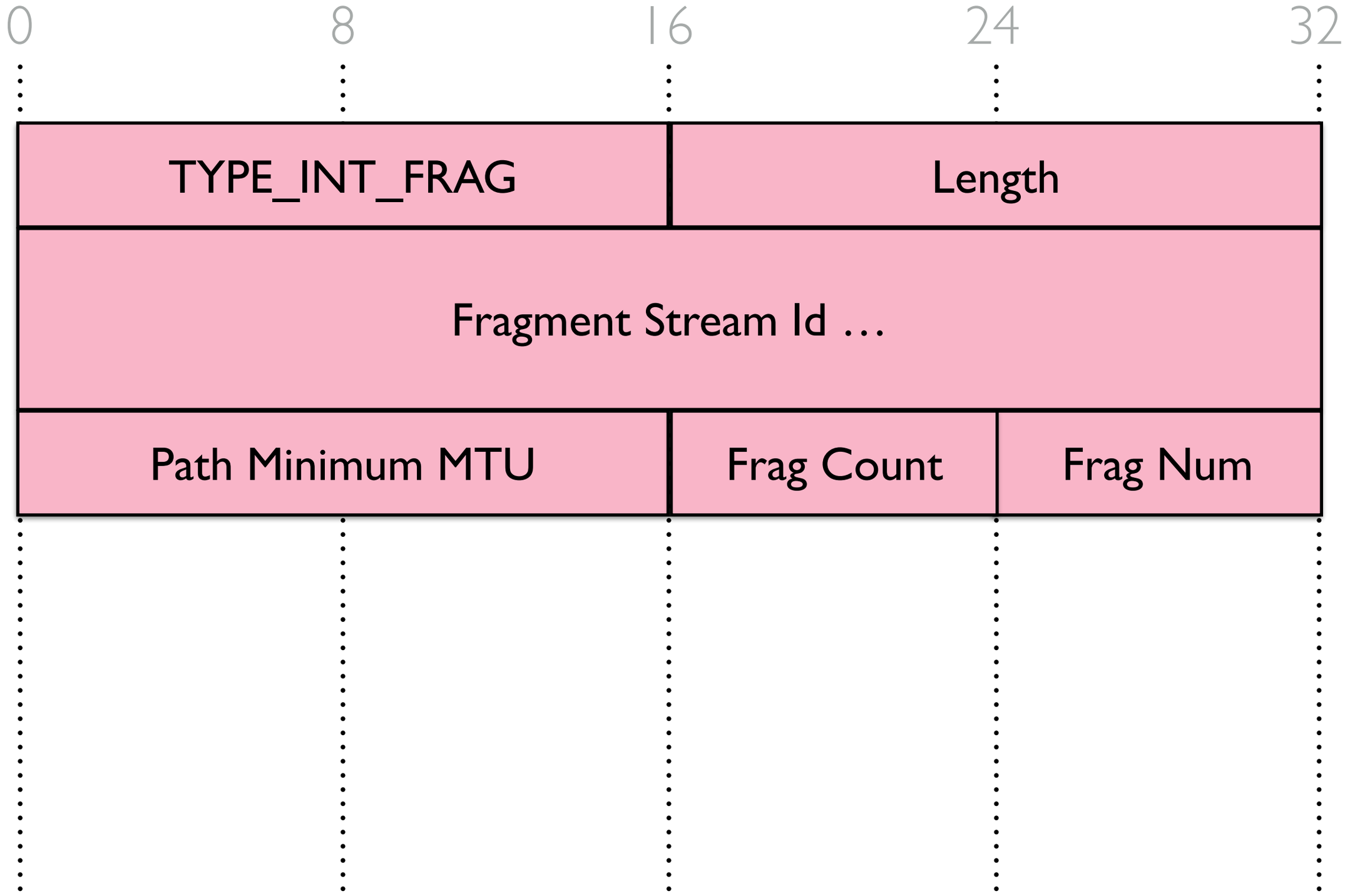
First Interest fragment must be routable

Content is fragmented to MTU size

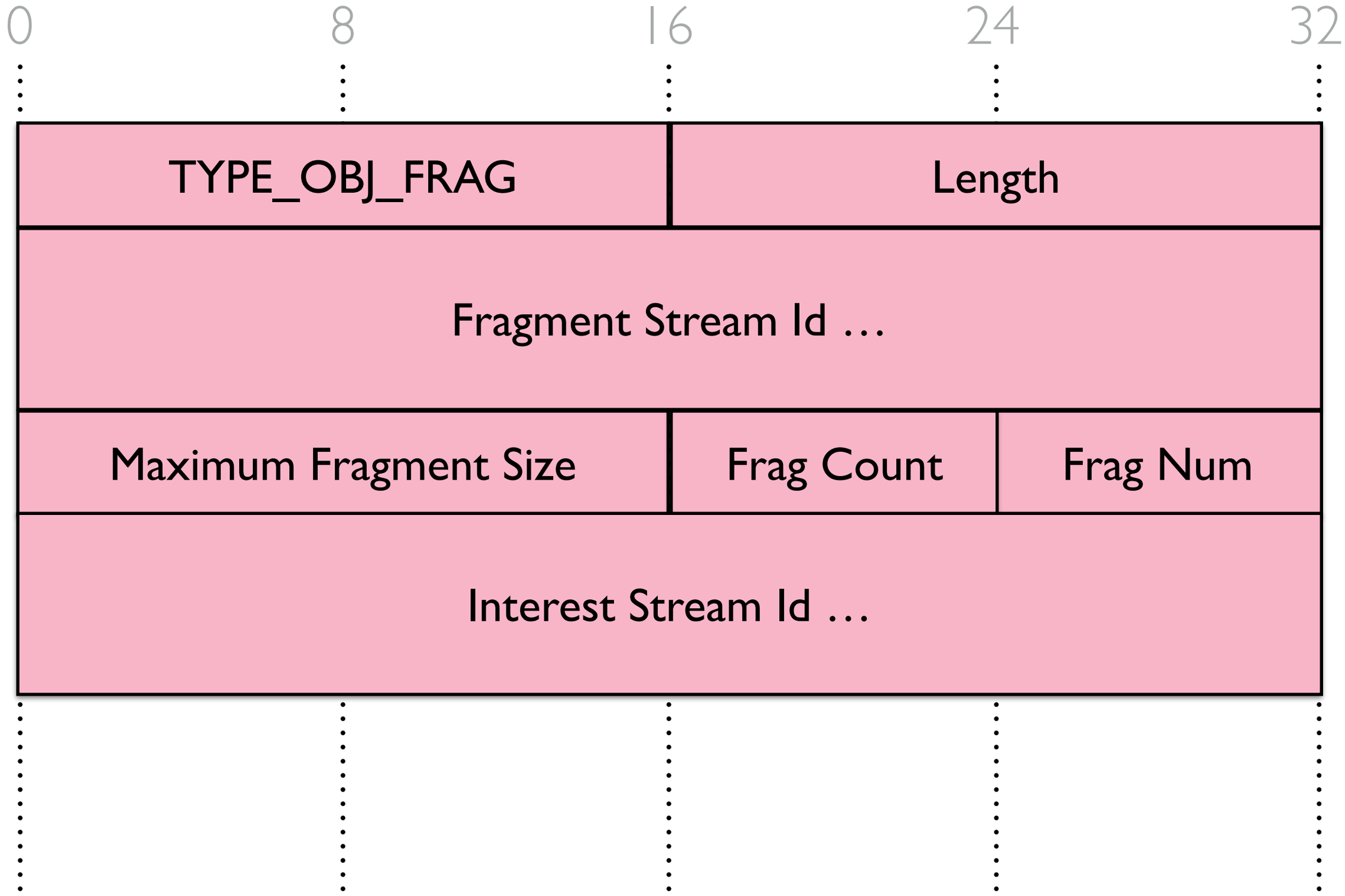
Nodes reassemble if checking Content-Object Hash

Fragments are packed ("MTU" size, except last)

Fragmentation Optional Header



Fragmentation Optional Header



End-to-End Fragmentation Summary

Cut through forwarding of interests

Cut through forwarding of content via Interest Id

Works in conjunction with Express Headers

Advanced fragmentation scheme in the works

CCN - Express

Express Headers

Used for fast forwarding

Similar to CCN enabled MPLS

Can be used together with fragmentation

Simple implementation via hash

Can be done global or local in scope

Forwarding Label

/parc/ccnx/presentations/slide10/v=2/c=0

Globally Routable Label

Forwarding Label

/parc/ccnx/presentations/slide10/v=2/c=0

Globally Routable Label

Similarity Label

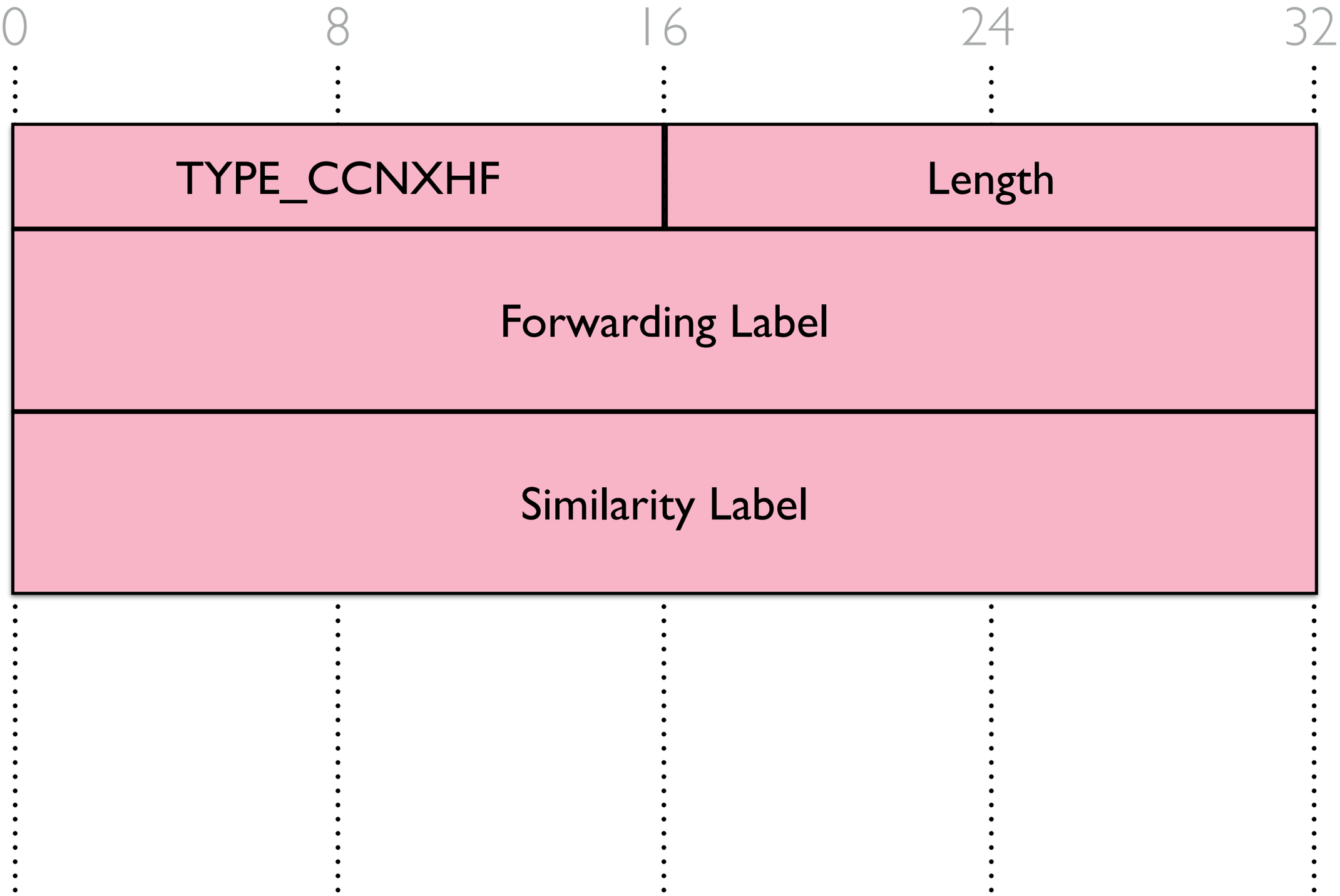
/parc/ccnx/presentations/slide10/v=2/c=0

?KeyId

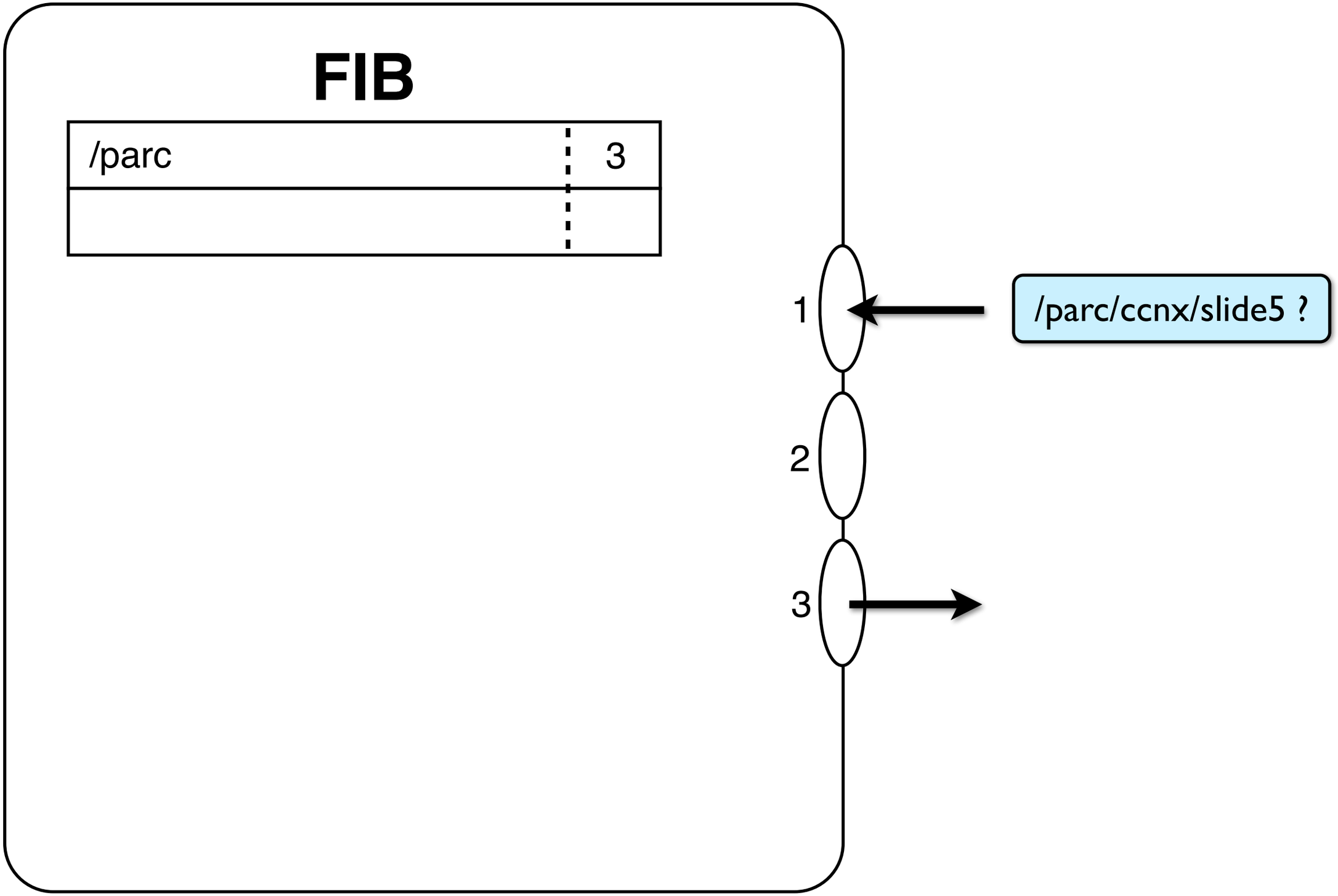
?ContentObjectHash

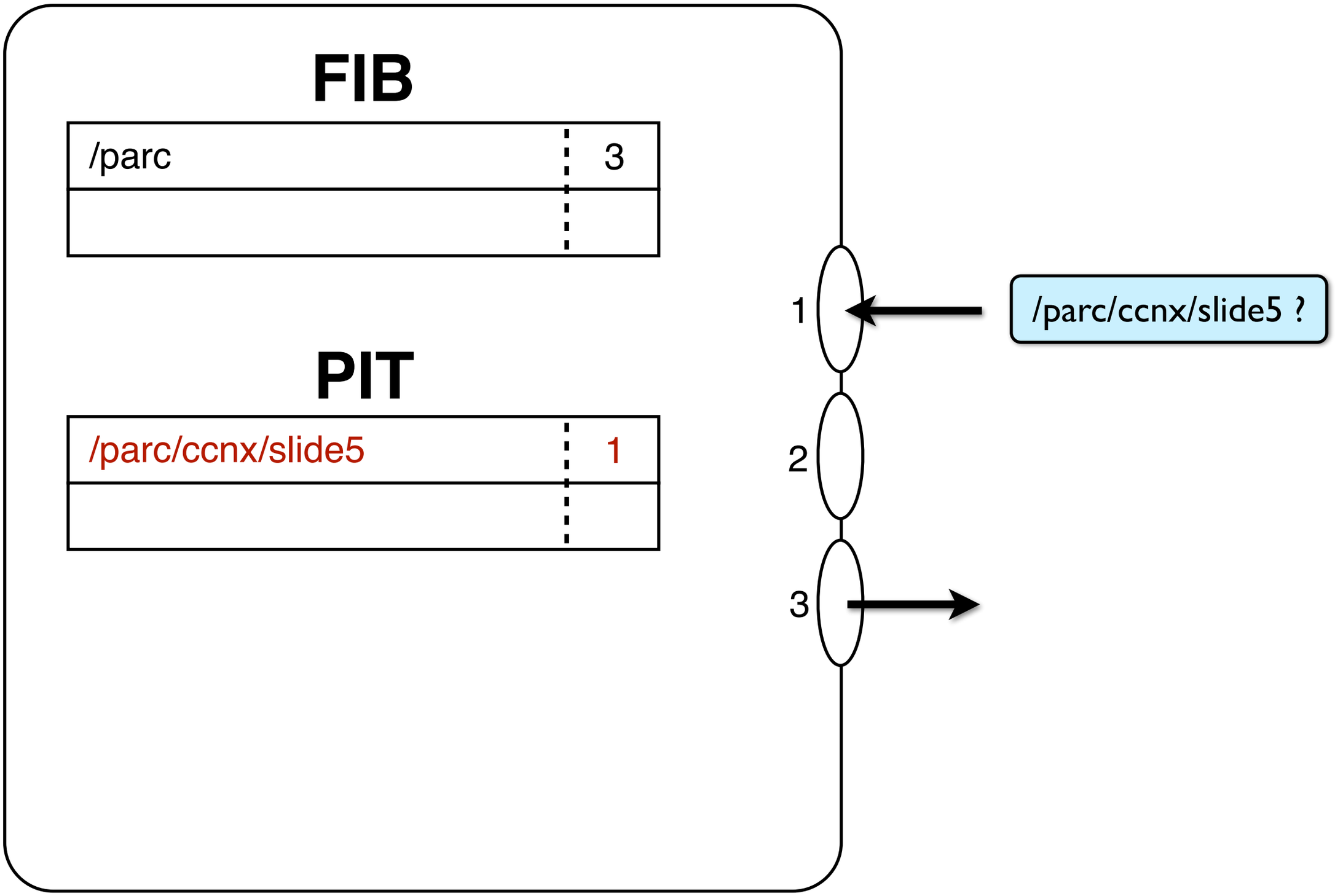
Similarity Label

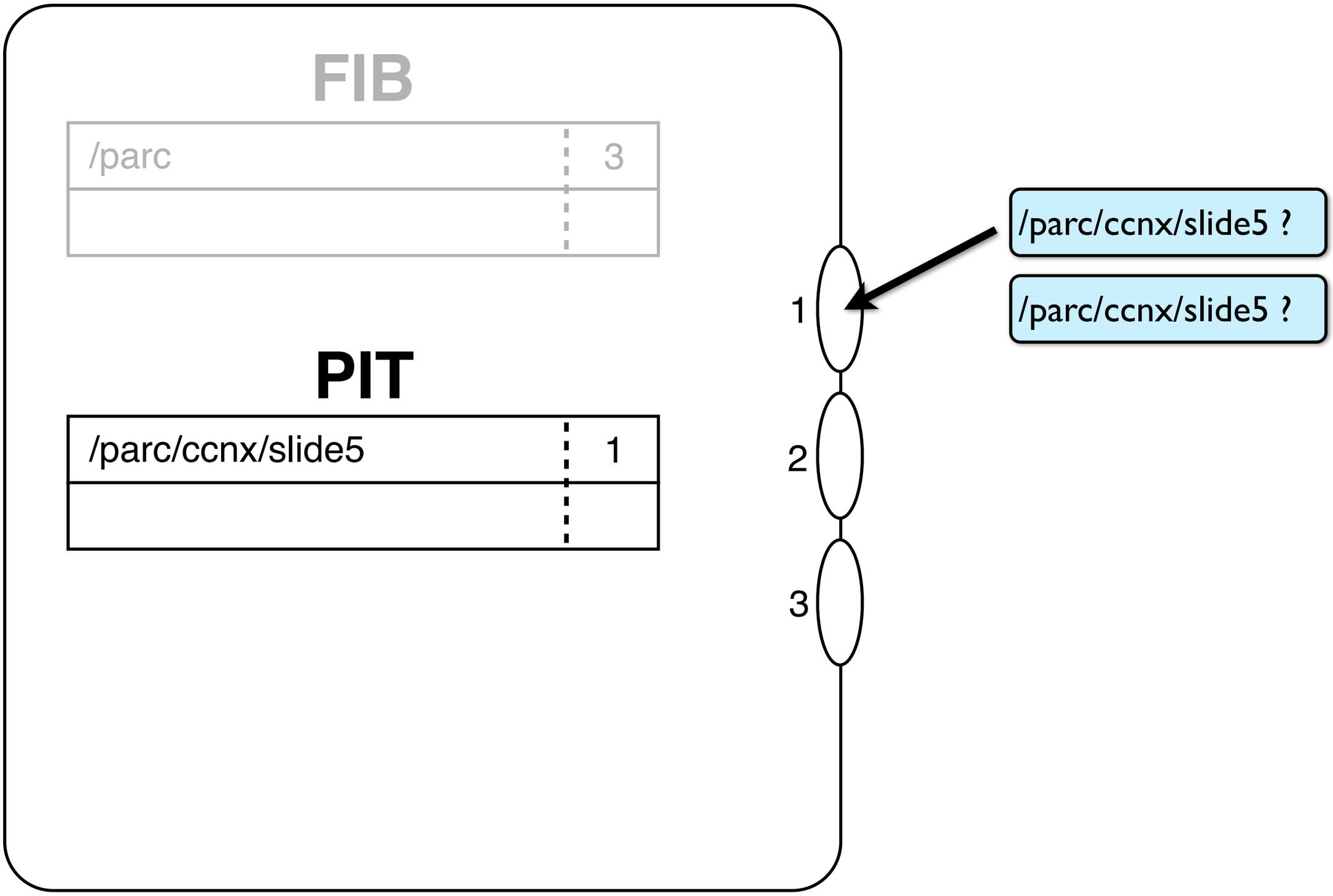
Express Headers

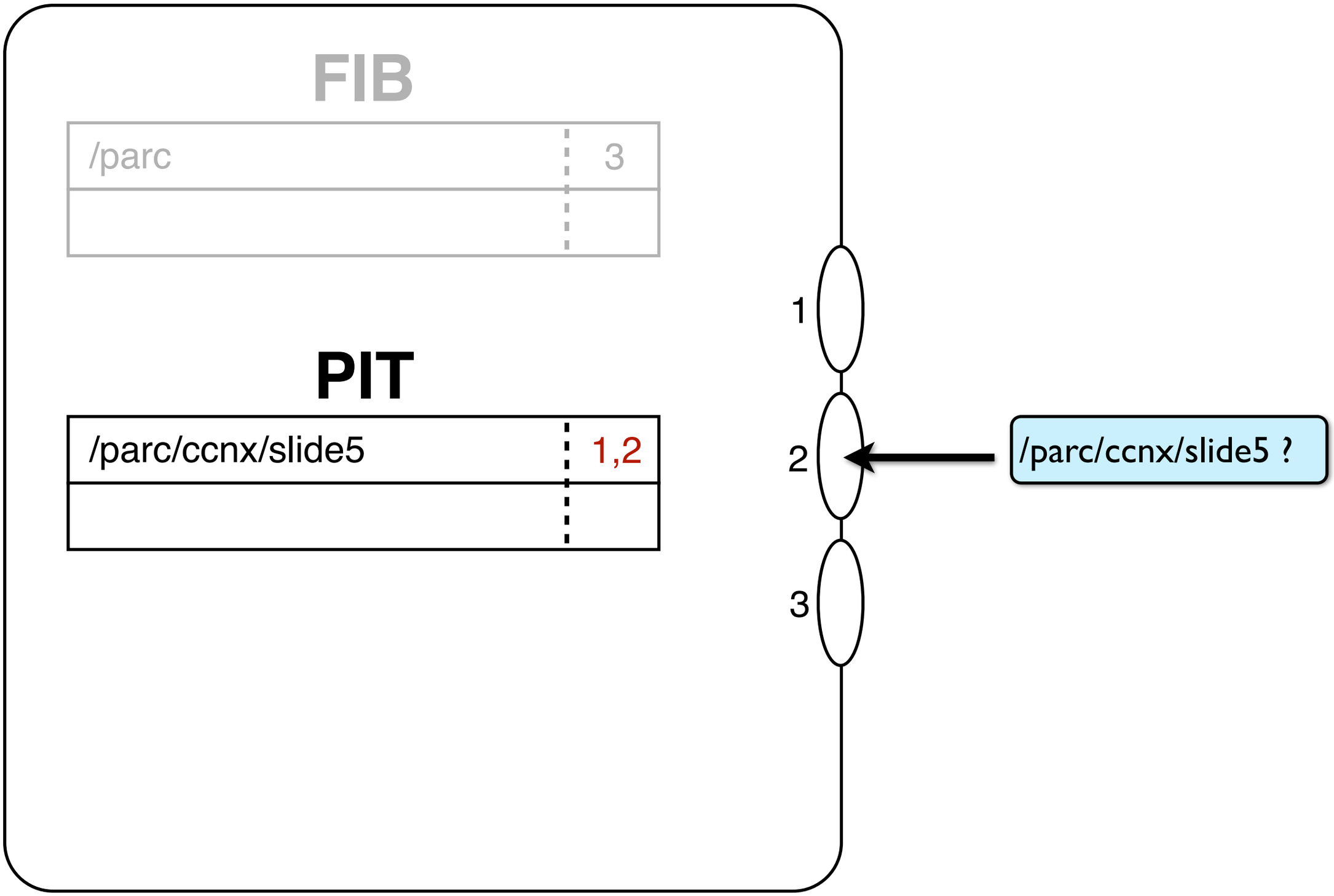


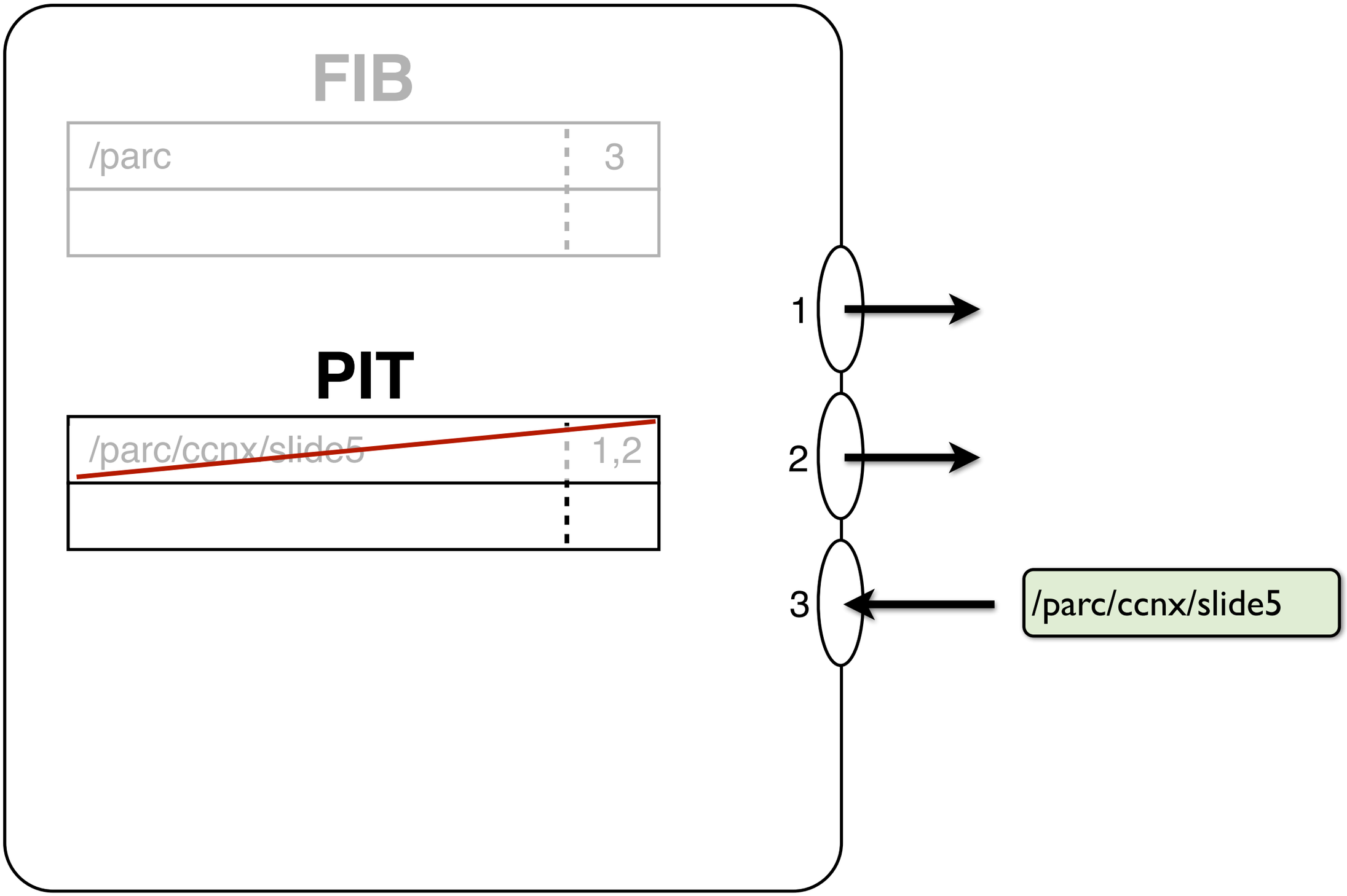
CCN - Forwarding

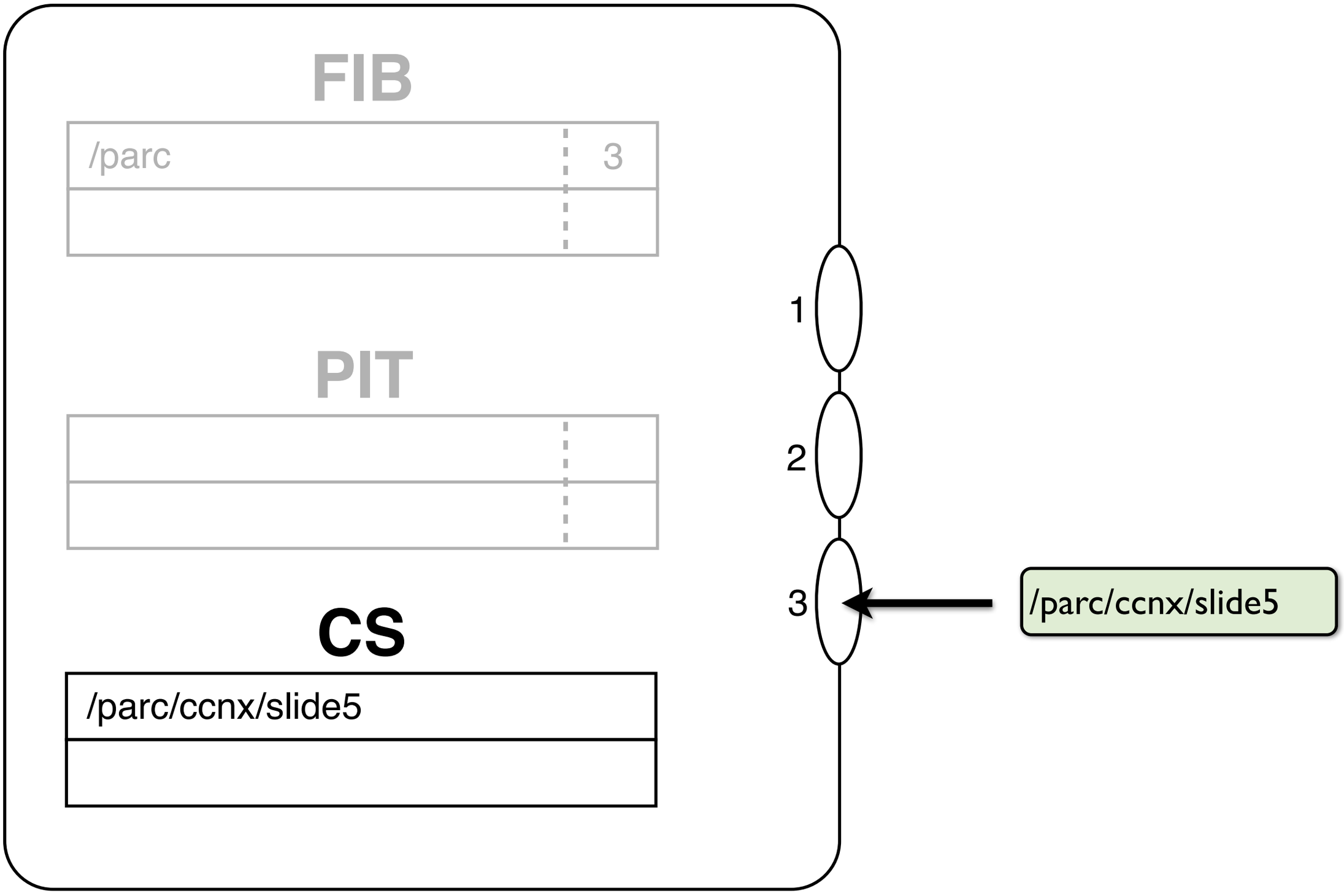


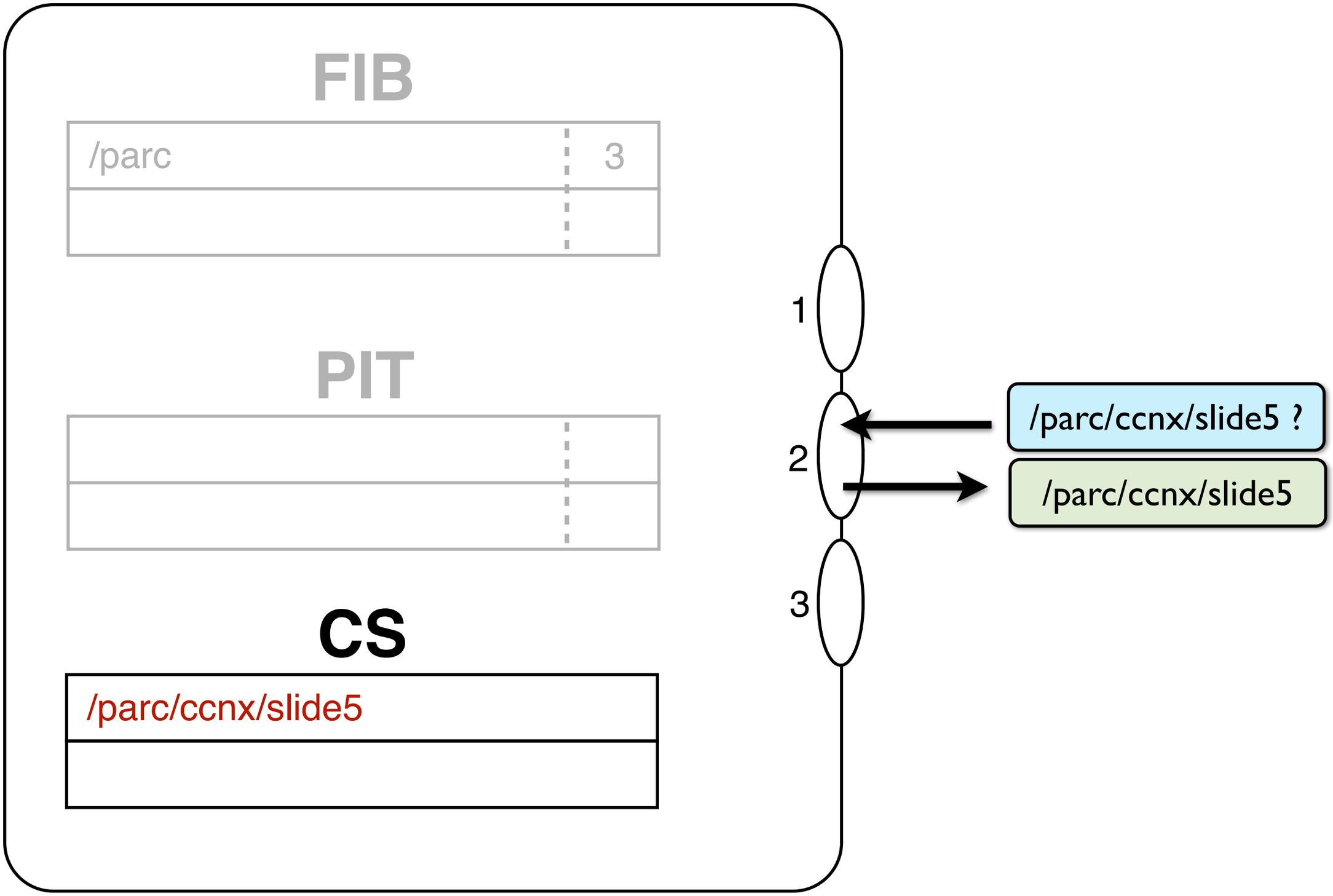












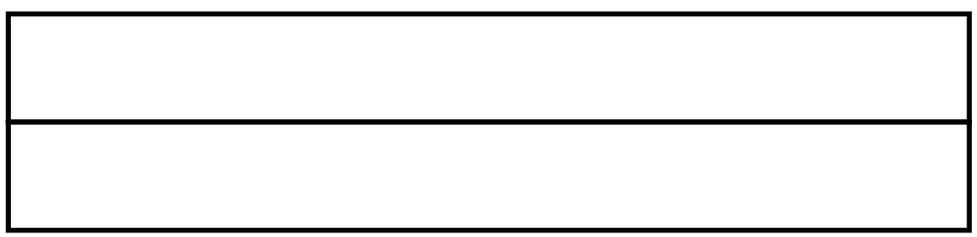
FIB

Forwarding Information Base
Store information about what face to follow to find a given name

PIT

Pending Interest Table
Store information about what interests are pending

CS



Content Store
Buffer content objects for potential reuse

CCN - Summary

Content-centric Networking (CCN)

transfers named data

CCN overview

Step 1 - Name the data

Name every piece of network data

Step 2 - Secure the data

Secure every piece of network data

Step 3 - Transfer the data

Move the data to interested recipients

Thank you

http://www.ccnx.org/

http://www.parc.com/ccn

CCN - Extra Slides

Glossary

Interest

A request for a piece of content.

Content Object

A response to an interest. A piece of data, signed.

Message

An Interest or Content Object

Chunk

A Content Object that is part of a larger piece of Data.

Fragment

A network encoded partial CCN Message
Includes static and per-hop-headers

Packet

A network encoded CCN Message (partial or complete)
Includes static and per-hop-headers

Glossary

Name Segment

A labeled name element (“`year=2014`”)

Name

A ordered set of CCN Name Segments

Name Prefix

A “left” subset of Name Segments in a Name

Content Object Hash

A hash over an encoded Content Object

Similarity Hash

A hash over the matching portion of an Interest

Interest Fragment Identifier

A Similarity Hash used to identify a fragmented Interest

Content Object Fragment Identifier

A Content Object Hash used to identify a fragmented Content Object