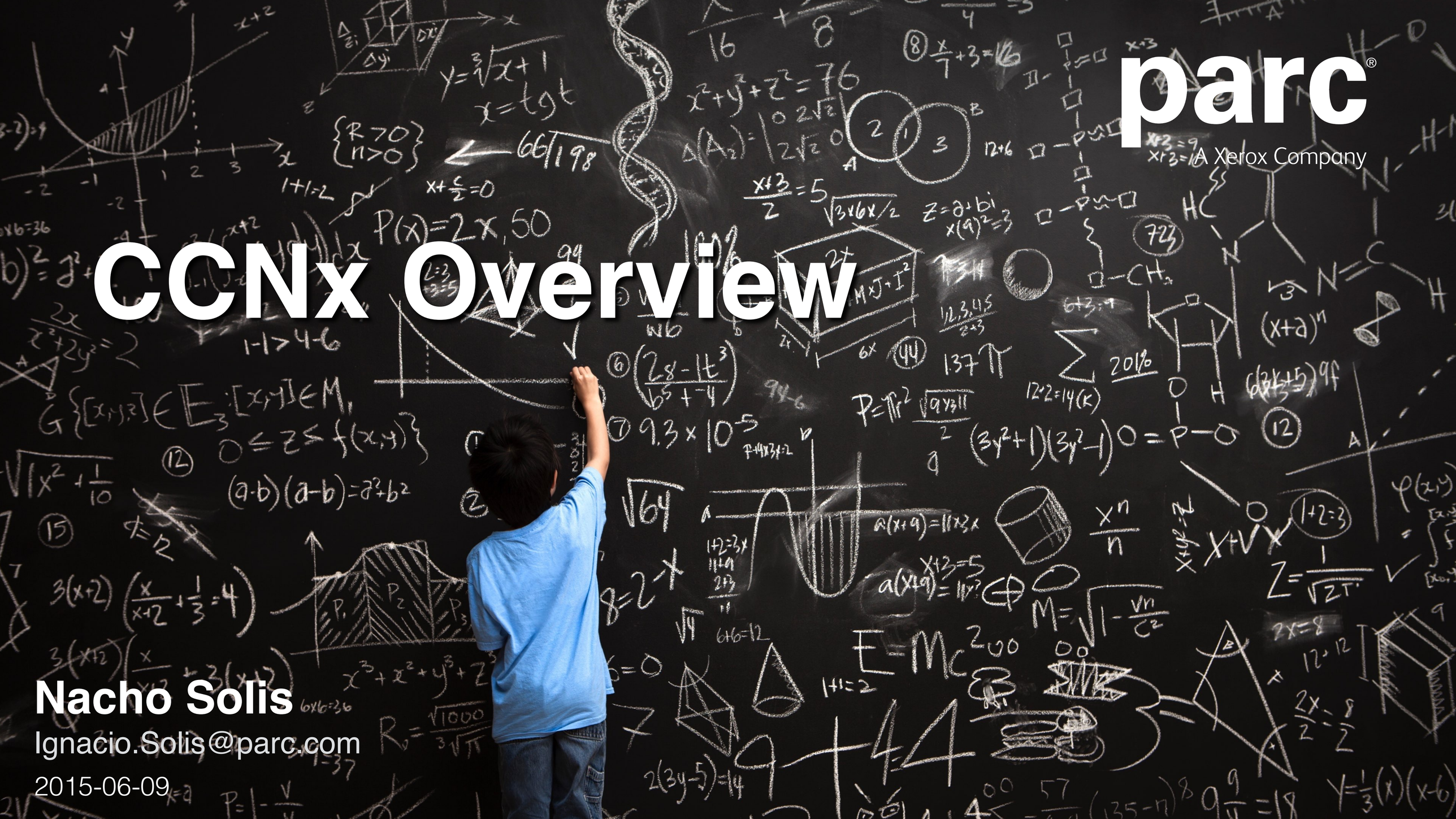
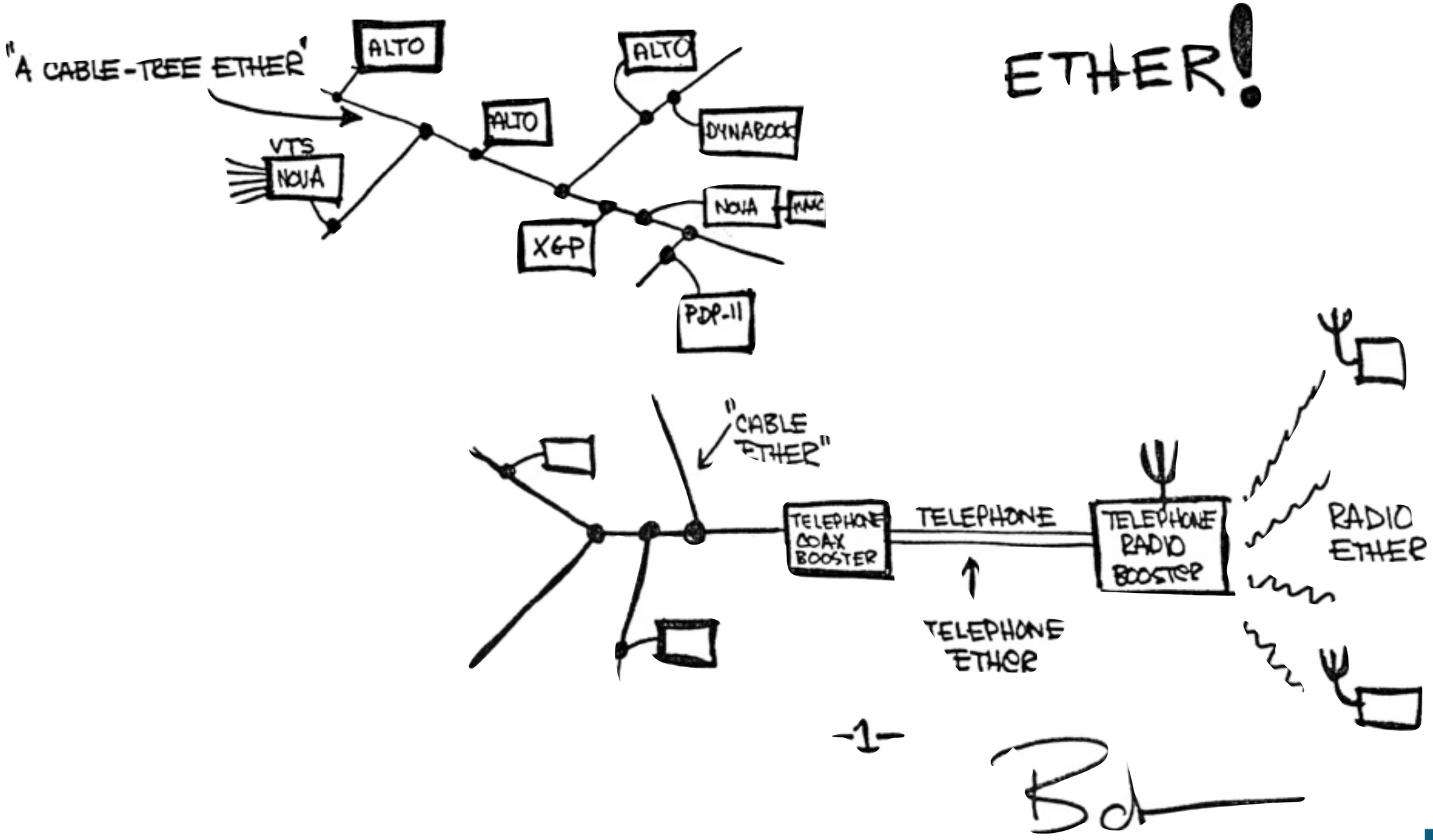


# CCNx Overview

parc®
A Xerox Company

**Nacho Solis**
Ignacio.Solis@parc.com
2015-06-09

# What is the job of the network?



# rethink the network
# rethink the stack

# The future Internet architecture

**We have a proposal that**

Is secure

Provides high availability

Transfers data independent of location

Takes advantage of storage and processing

**It does this by**

Naming all data, securing all data and communicating based on name

# CCNx (Content-Centric Networks)

is a

secure communications architecture

based on

transferring named information objects

# The CCNx communication architecture

**Applications**

User facing programs

**Services**

Base services required for network operation

**APIs**

Abstractions for interacting with the network

**Transport**

Structured and secure “end-to-end” communication

**Messaging**

Name-based, network wide communication using CCN messages

**Framing**

Transport for messages over layer 2

# The CCNx project

**Specifications**

Description of protocols and algorithms

**Software**

Reference software implementation

**Hardware**

Hardware prototypes (big and small)

**Commercial community**

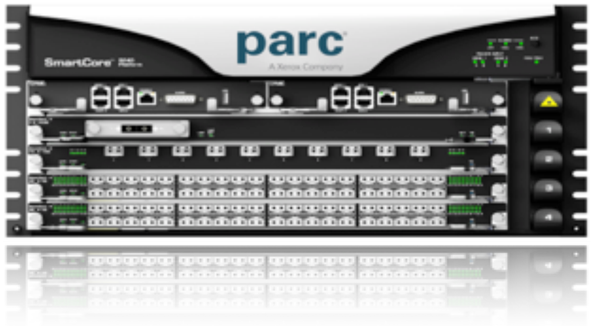
Commercial companies developing CCN

**Research community**

Researchers, faculty and students working on the CCN technology

**Developer community**

Application developers (big and small) using CCN

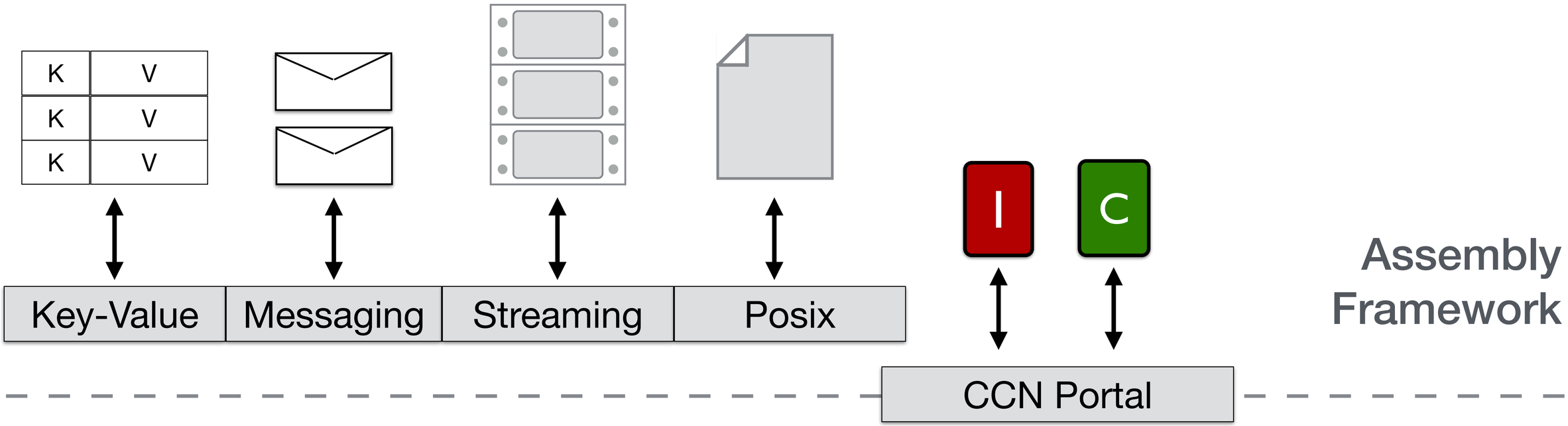


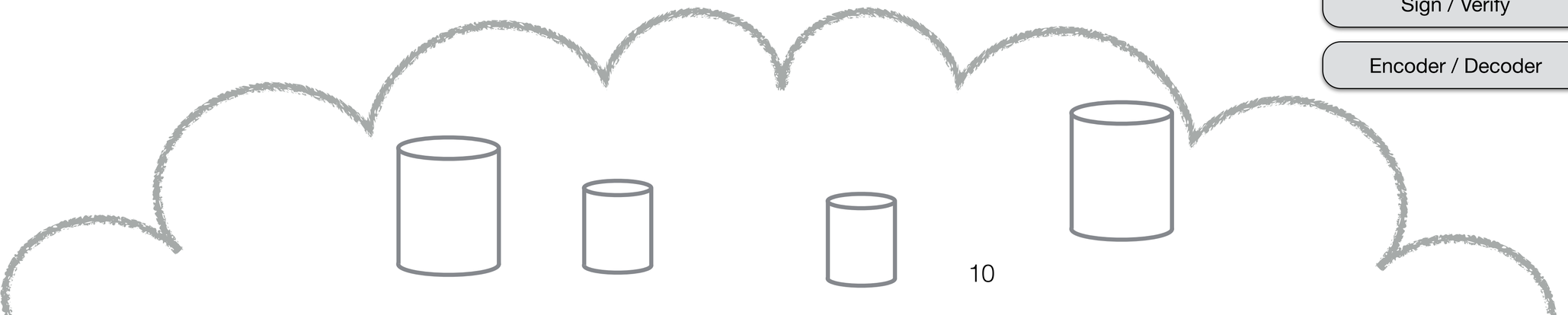



parc
A Xerox Company

# Rethink the stack?

# Network abstraction



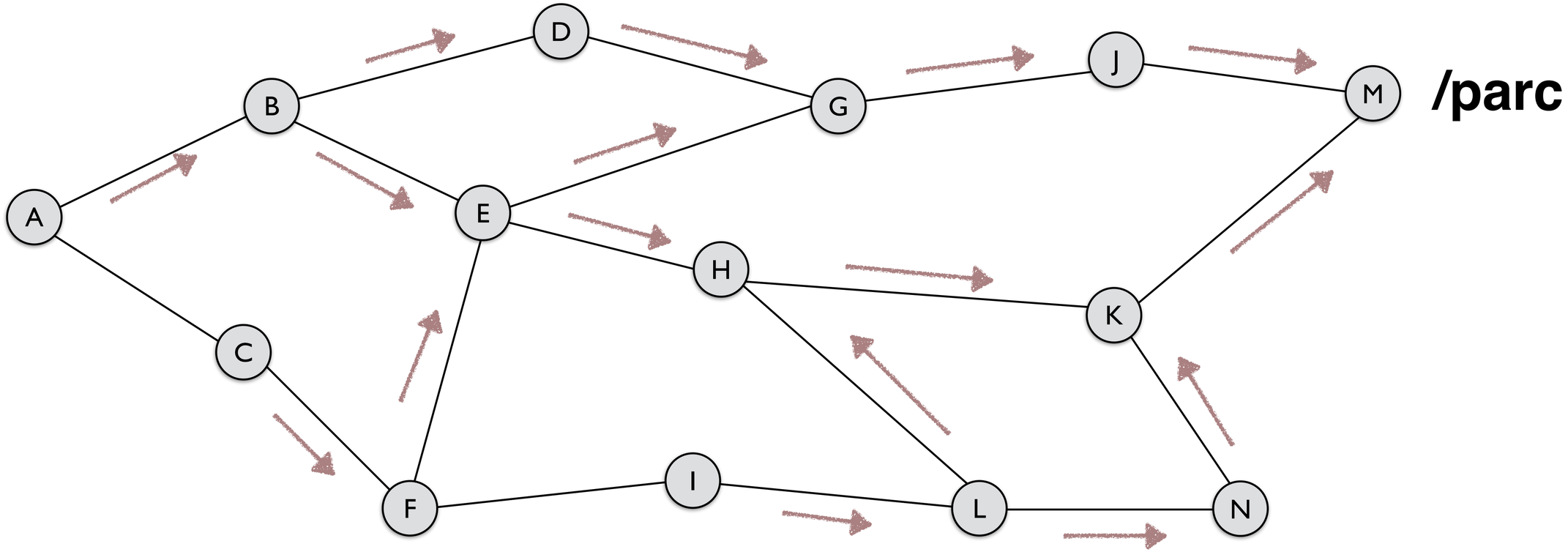





parc
A Xerox Company

# Rethink the network?

# Names are advertised via routing




Routes are set up pointing to M for prefix /parc



parc
A Xerox Company

# Core protocol

An Interest Message

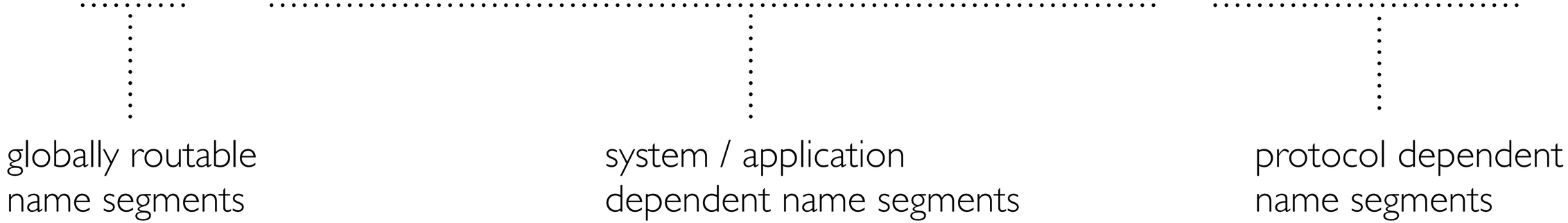
A Content Message


**A request for a named piece of content**

**A reply with a named piece of content**

# CCNx names

/parc/ccnx/presentation/slide17/v=2/c=0



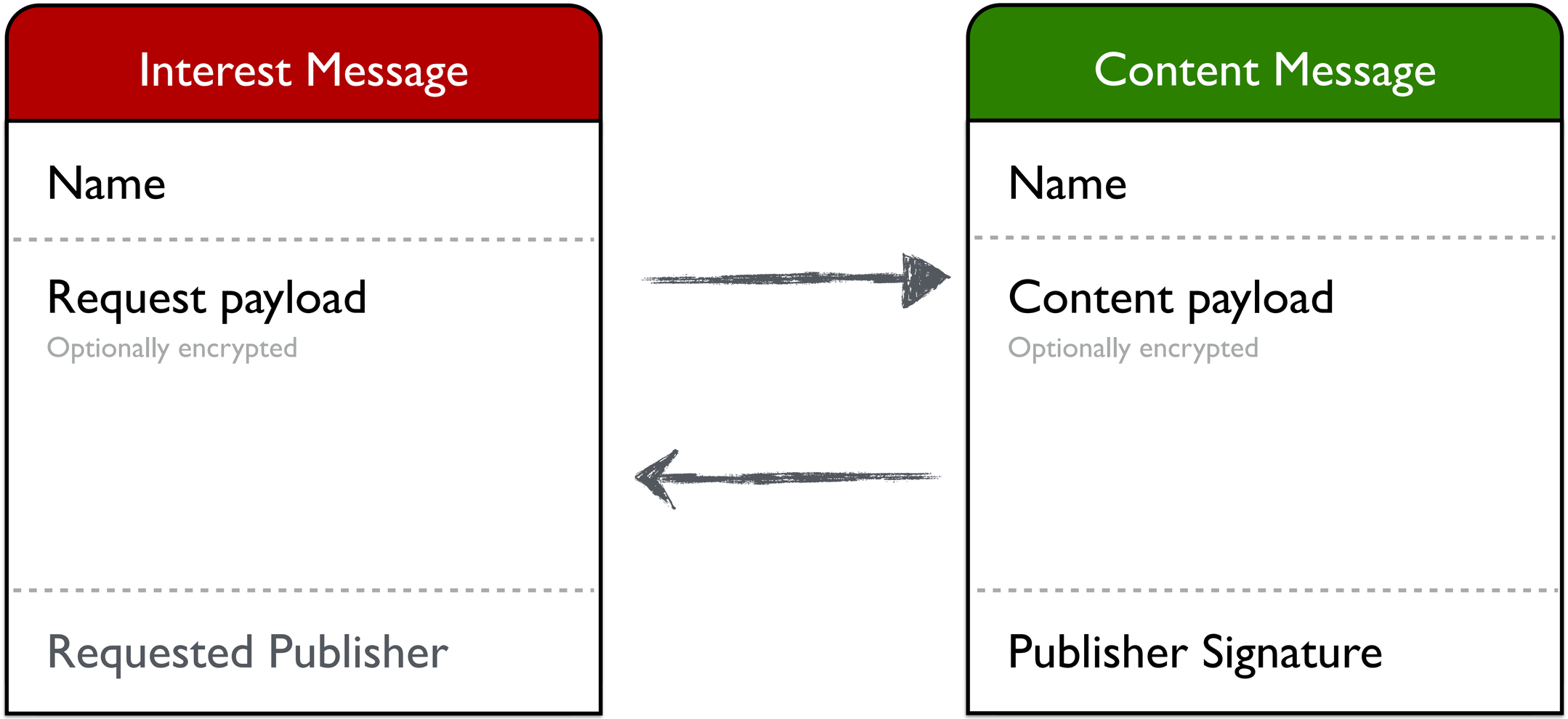

**Everything has a name**
**Hierarchical**

**Don’t have to be human readable**
**Replaces IP addresses and ports**

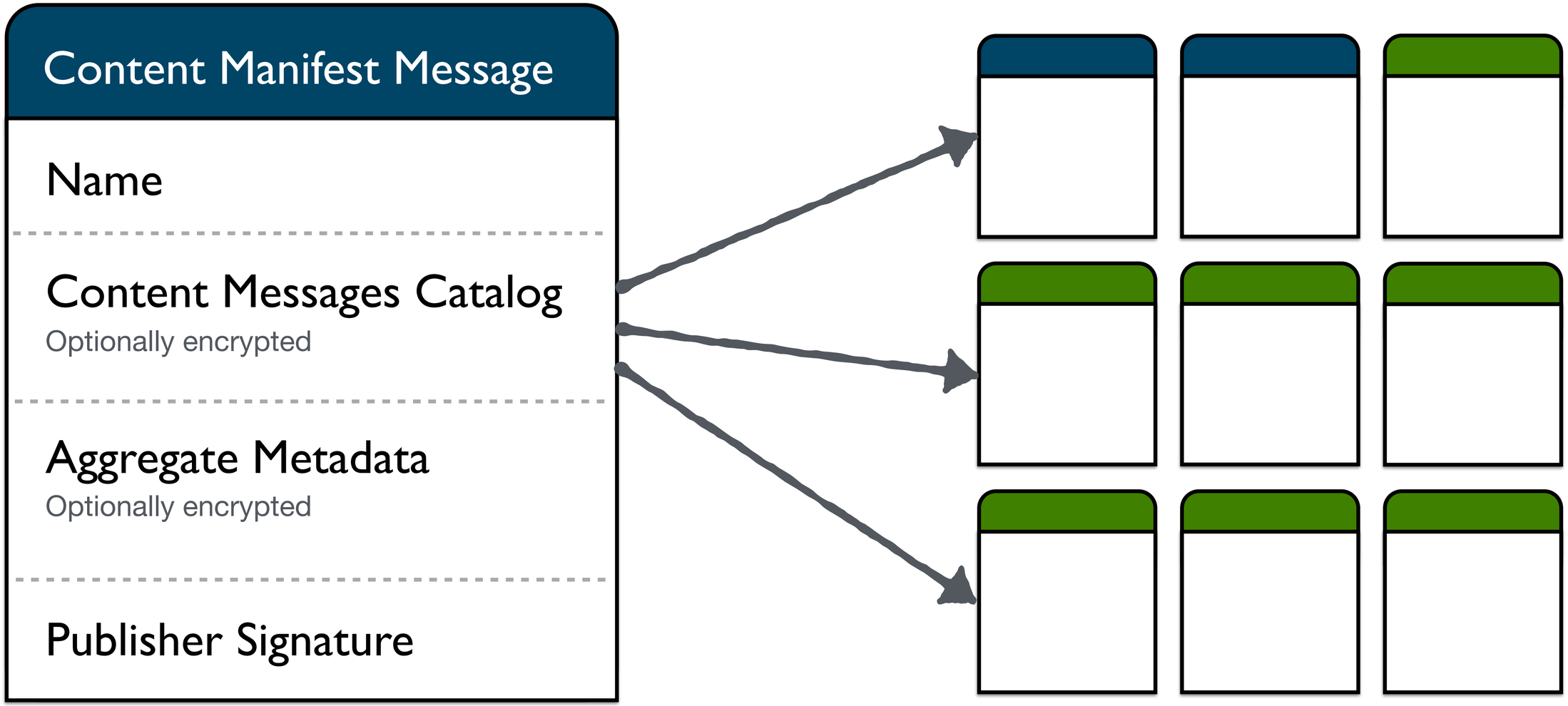


parc
A Xerox Company

# Core messages



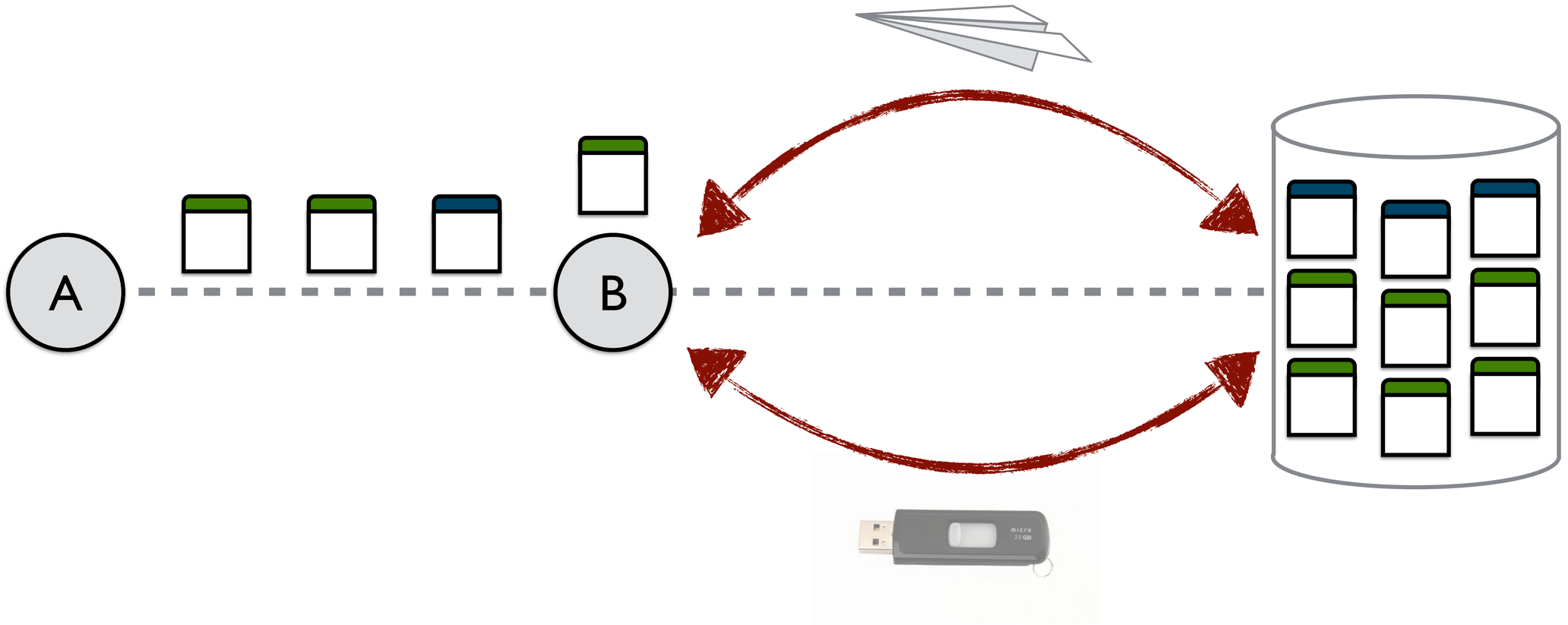
# Network data structures






parc
A Xerox Company

# Unifying network and storage



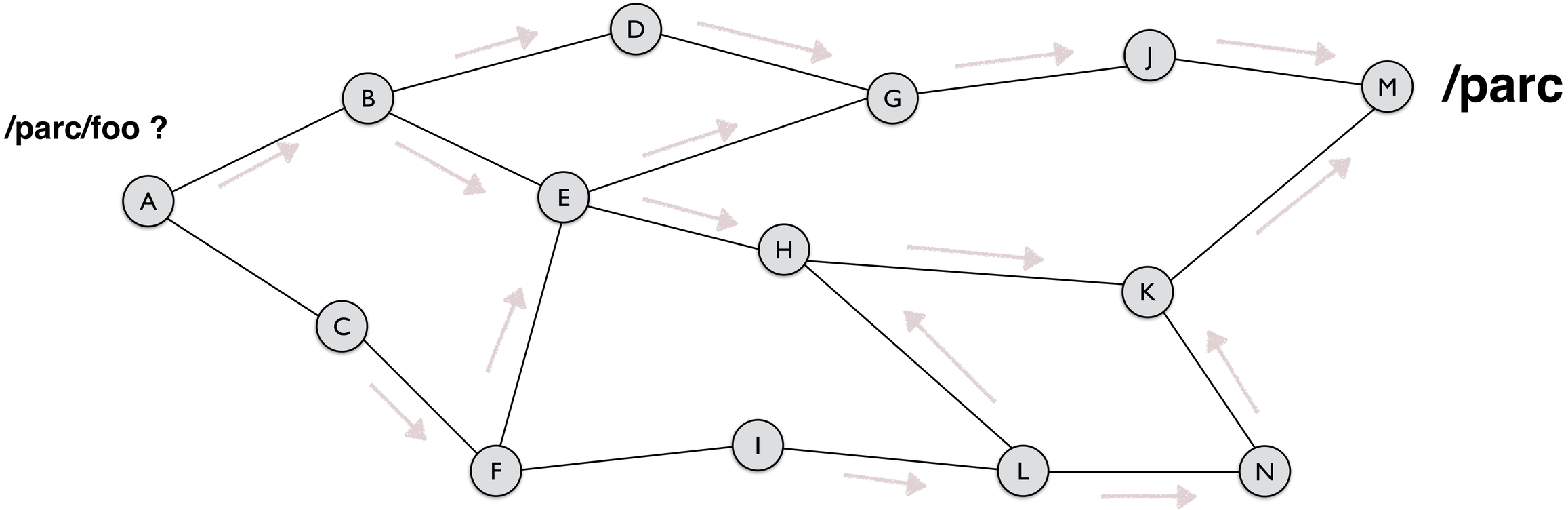

Messages are secure
(in motion and at rest)

Messages are the same
(in network and in storage)



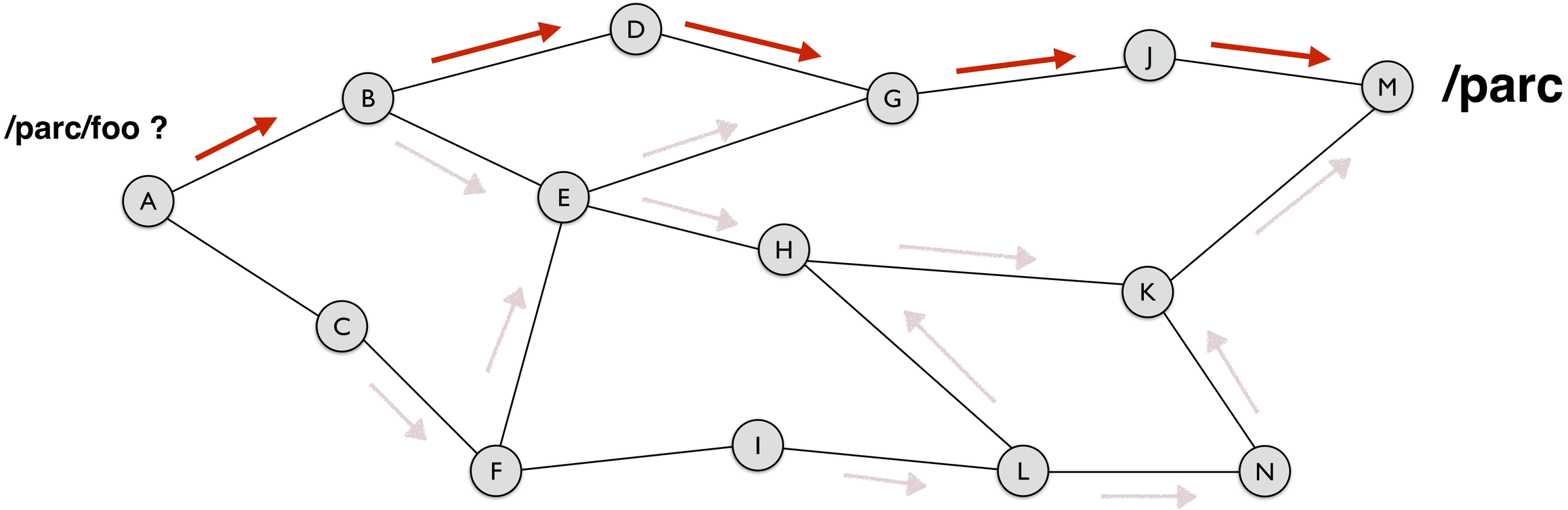
parc
A Xerox Company

# A simple example



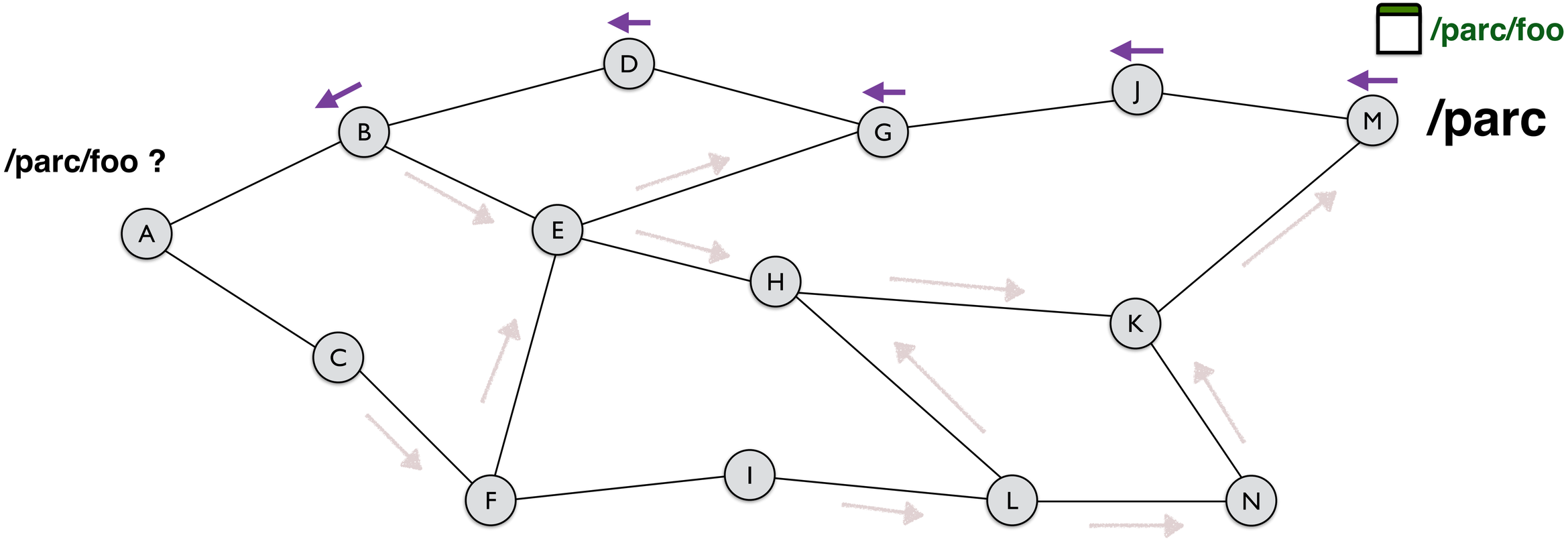

Node A has a request for /parc/foo - issues an interest

# Interests follow the advertised route



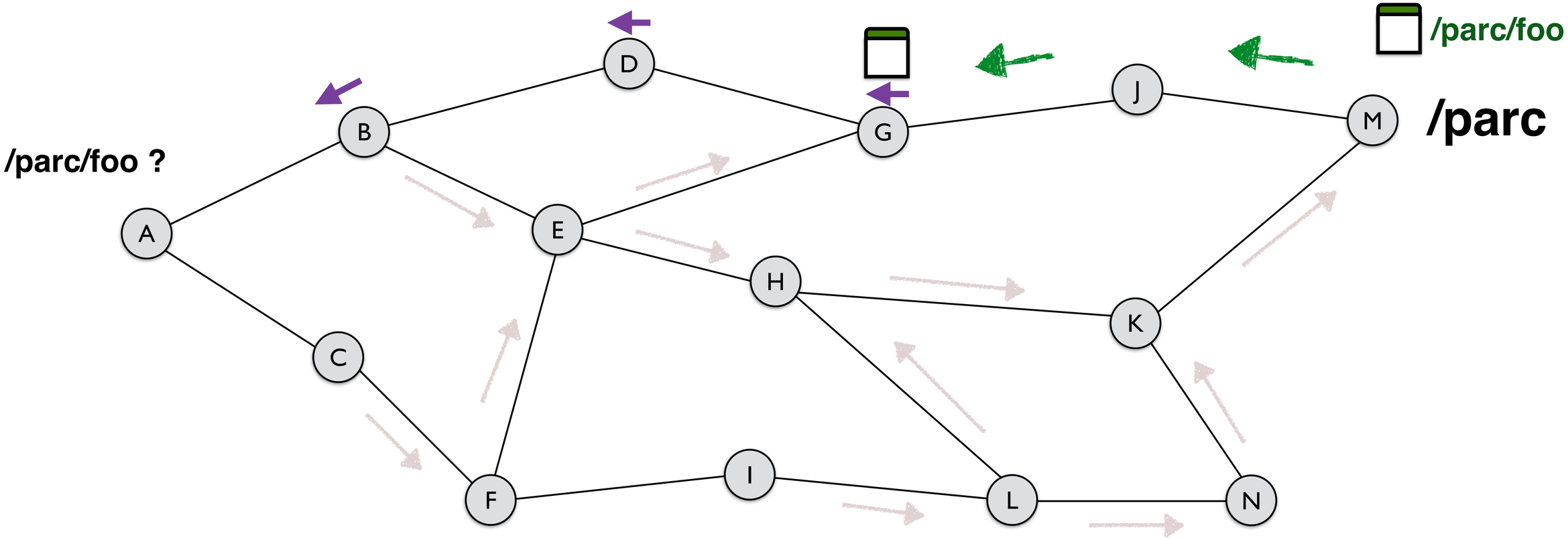

Interests are forwarded towards the node that advertised the name /parc

# Interests leave state along the path



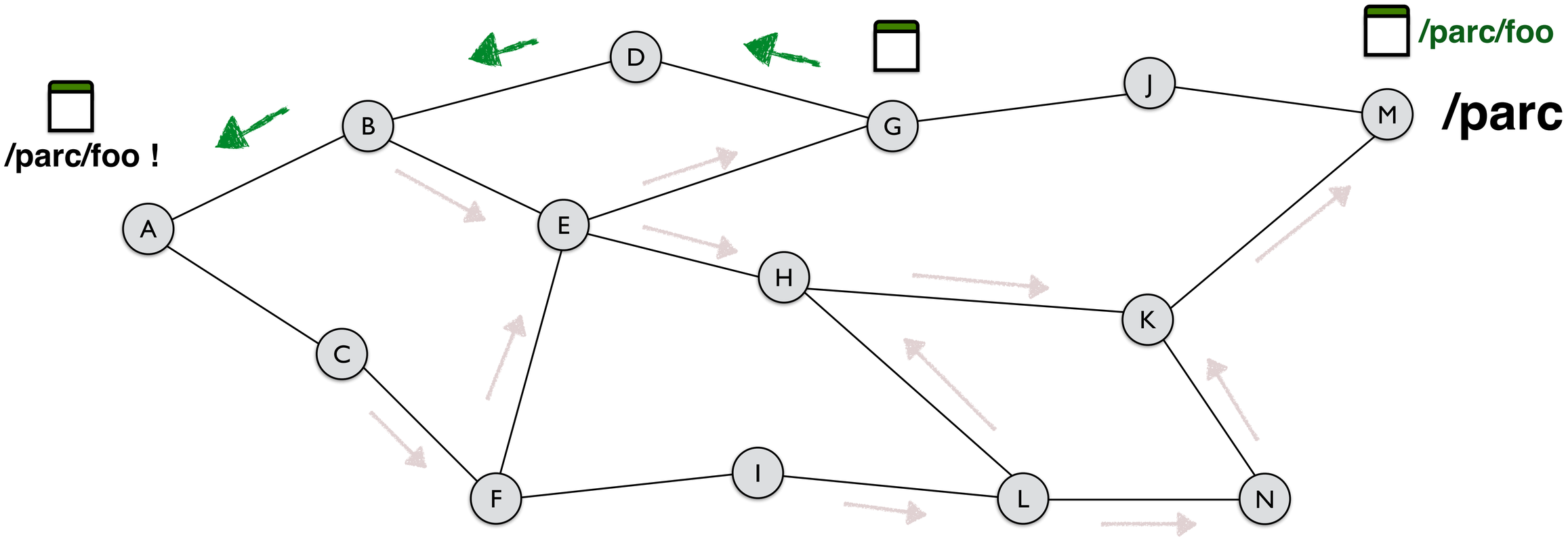

Routers keep information about the reverse path of the request

# Content Messages follow reverse path




Reverse path state gets consumed as Content Messages flow

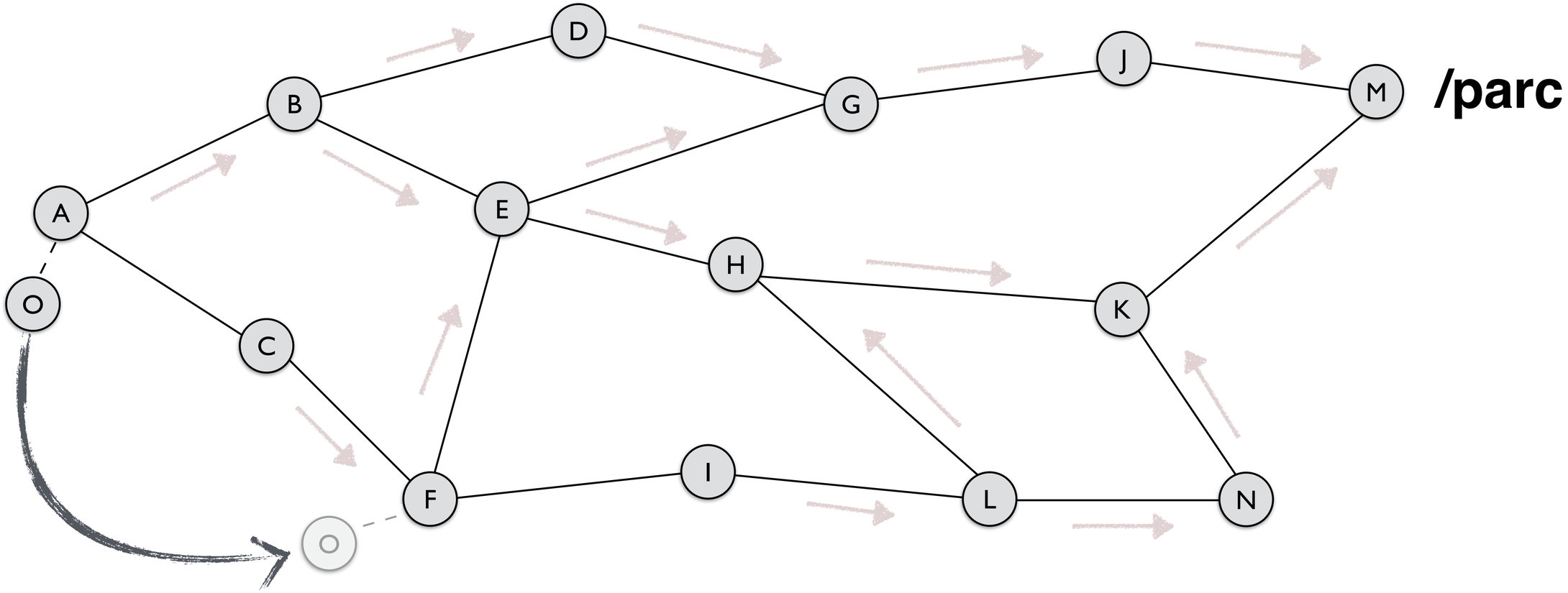
# Nodes can cache Content Messages




Caching reduces load on network

Caching makes error recovery more efficient (local retransmissions)

# Simple mobility

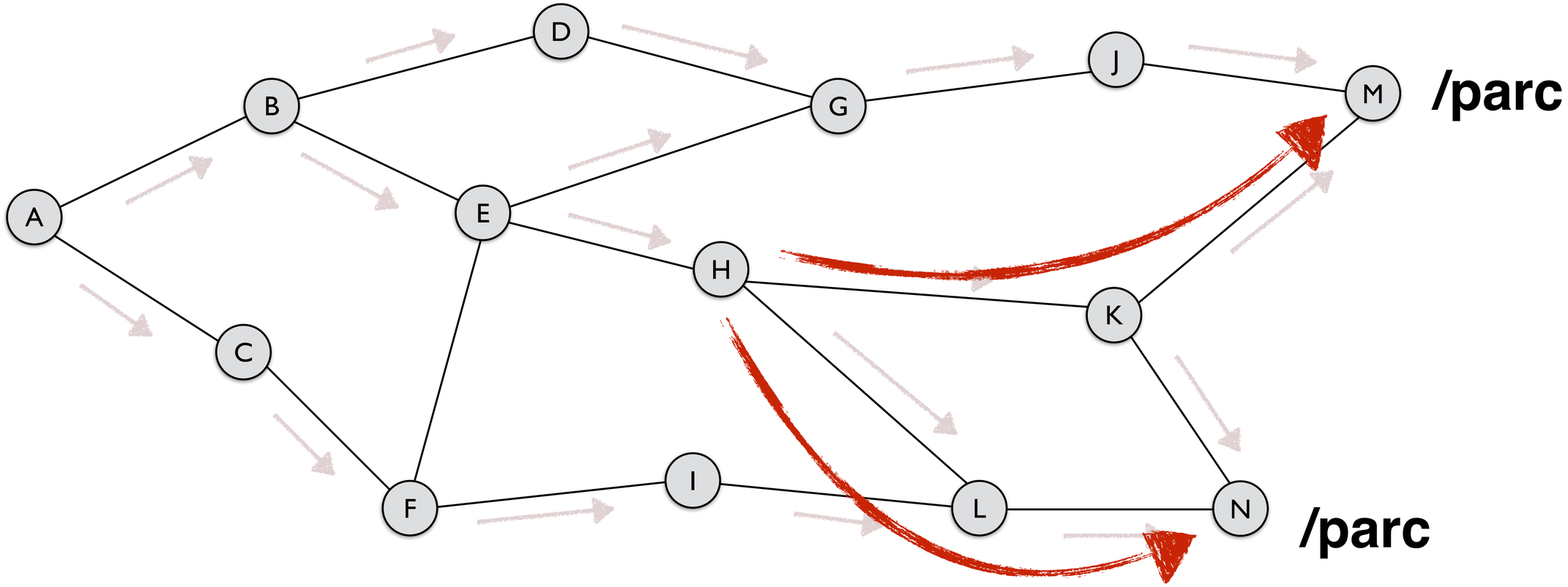



Reverse path routing allows for seamless reattachment



parc
A Xerox Company

# Load-balancing, redundancy and failover




With multi source, the core protocol allows the network to deliver load balancing, transport redundancy and automatic failover



parc
A Xerox Company

# Benefits

## Secure and Resilient

Individual content objects secure in motion and at rest

Network not susceptible to IP style network attacks

## Flexible and Efficient

Base network offers load balancing, redundancy and failover

Can handle heavy loads and perform local repair

Core protocol and messages scale up and down

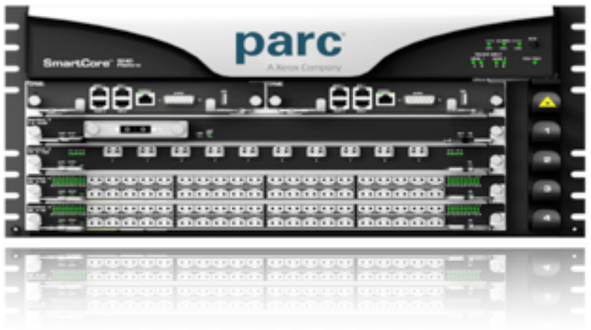
## Extensible and Comprehensive

Modular APIs to access network functions and services

Common data structures between network, APIs and applications

# Sensors, servers and supercomputers









parc
A Xerox Company

# 5G

# 5G

Flexible
Expandable
Indirection
Mobility
Security
Content Distribution
Management
Resilient
Peer-to-peer
Error recovery
Disruption Tolerant
Caching
Extensible

Communication

Storage

Processing

# Content-Centric Networking (CCNx)

Secure and Resilient

Flexible and Efficient

Extensible and Comprehensive

parc®

A Xerox Company

# Thank you

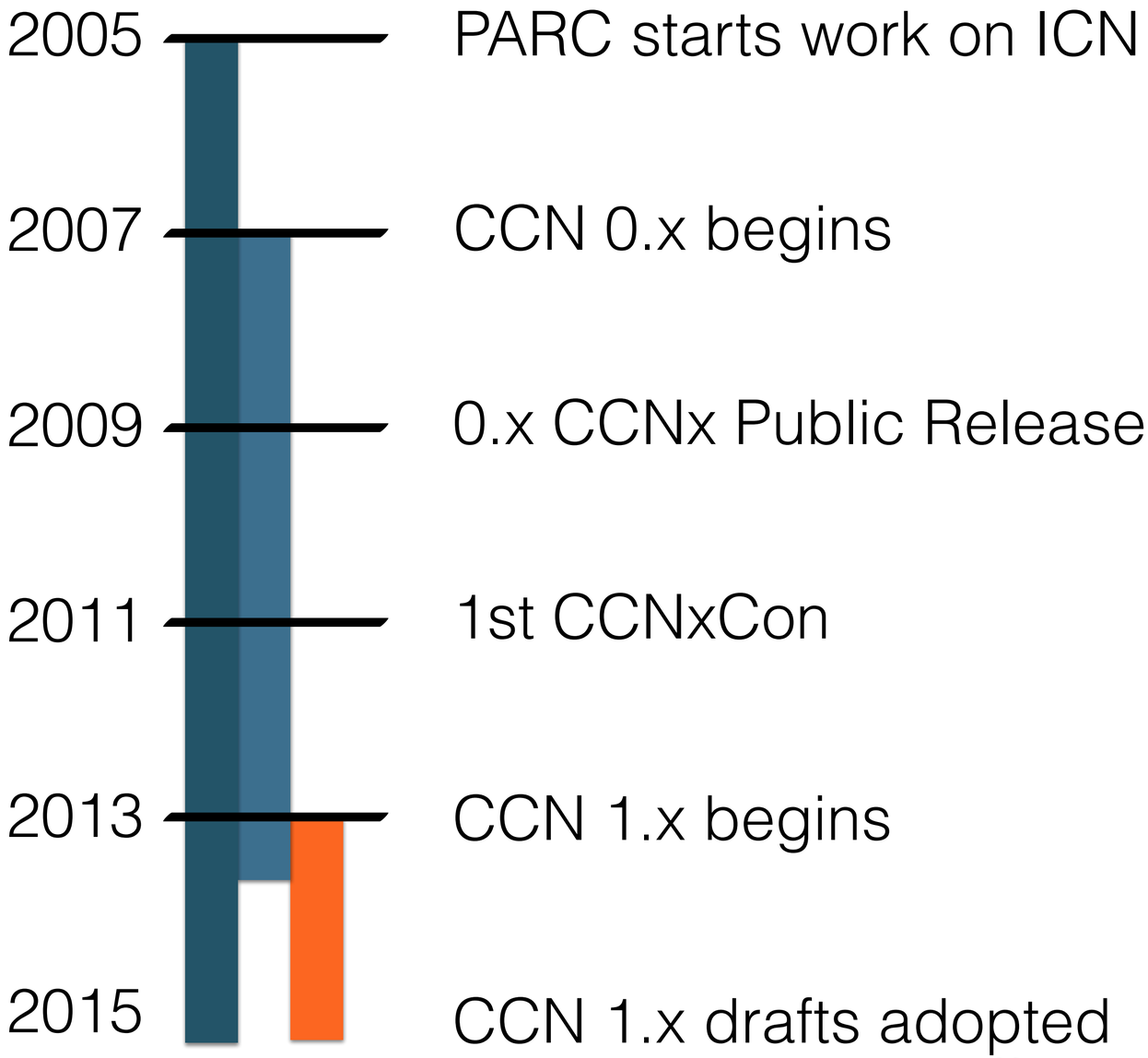
Ignacio.Solis@parc.com

http://www.ccnx.org/

# CCNx - Extra Slides

parc
A Xerox Company

# Timeline




ICN Research

CCN 0.x

CCN 1.x

NDN Formed (2010)

1st Dagstuhl Seminar (2010)

1st SIGCOMM workshop (2011)

1st INFOCOM workshop (2012)

1st ICNRG meeting (2012)

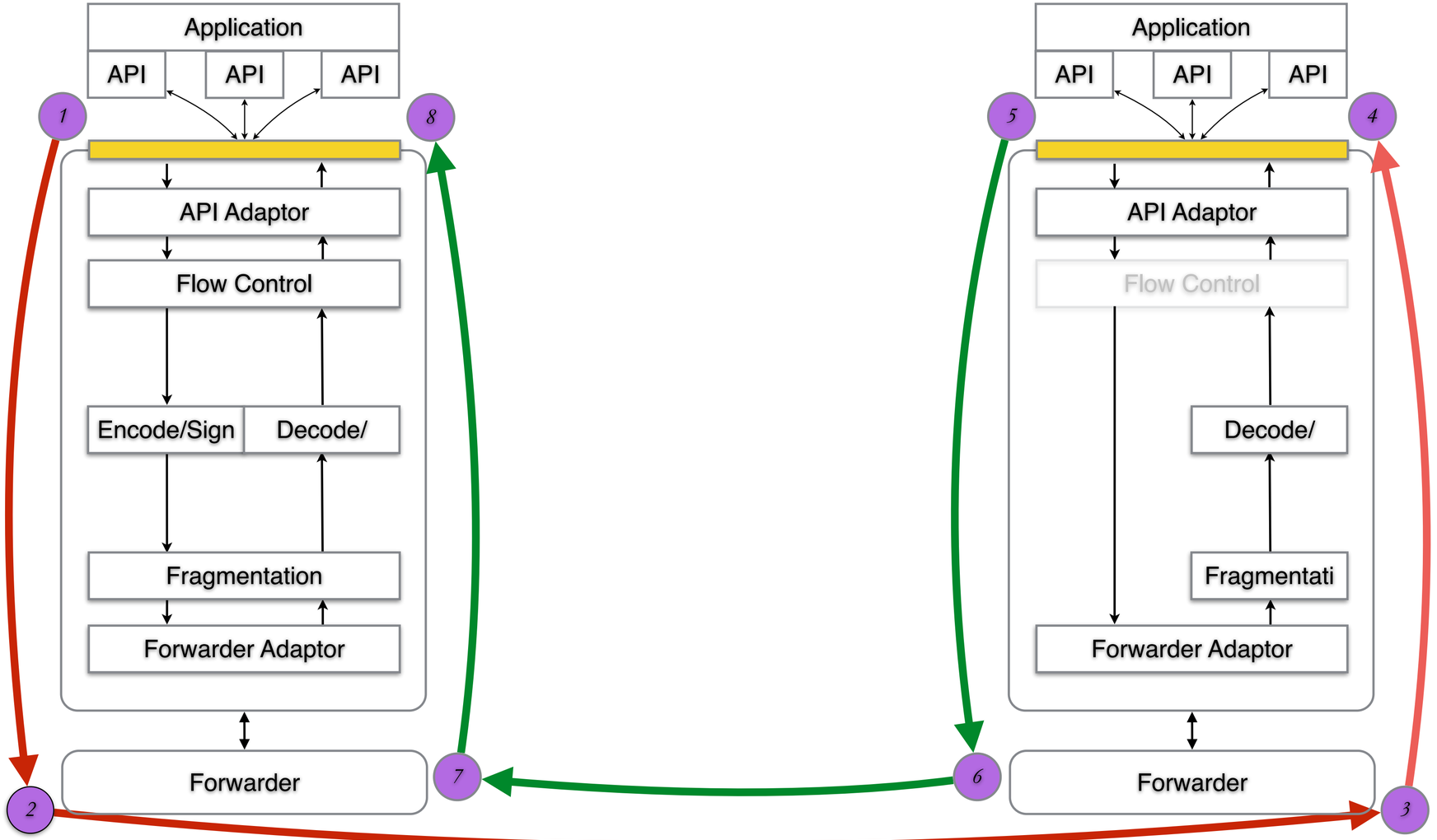
1st NDNCom meeting (2014)

1st ICN Conference (2014)



parc
A Xerox Company

# Transport Stack

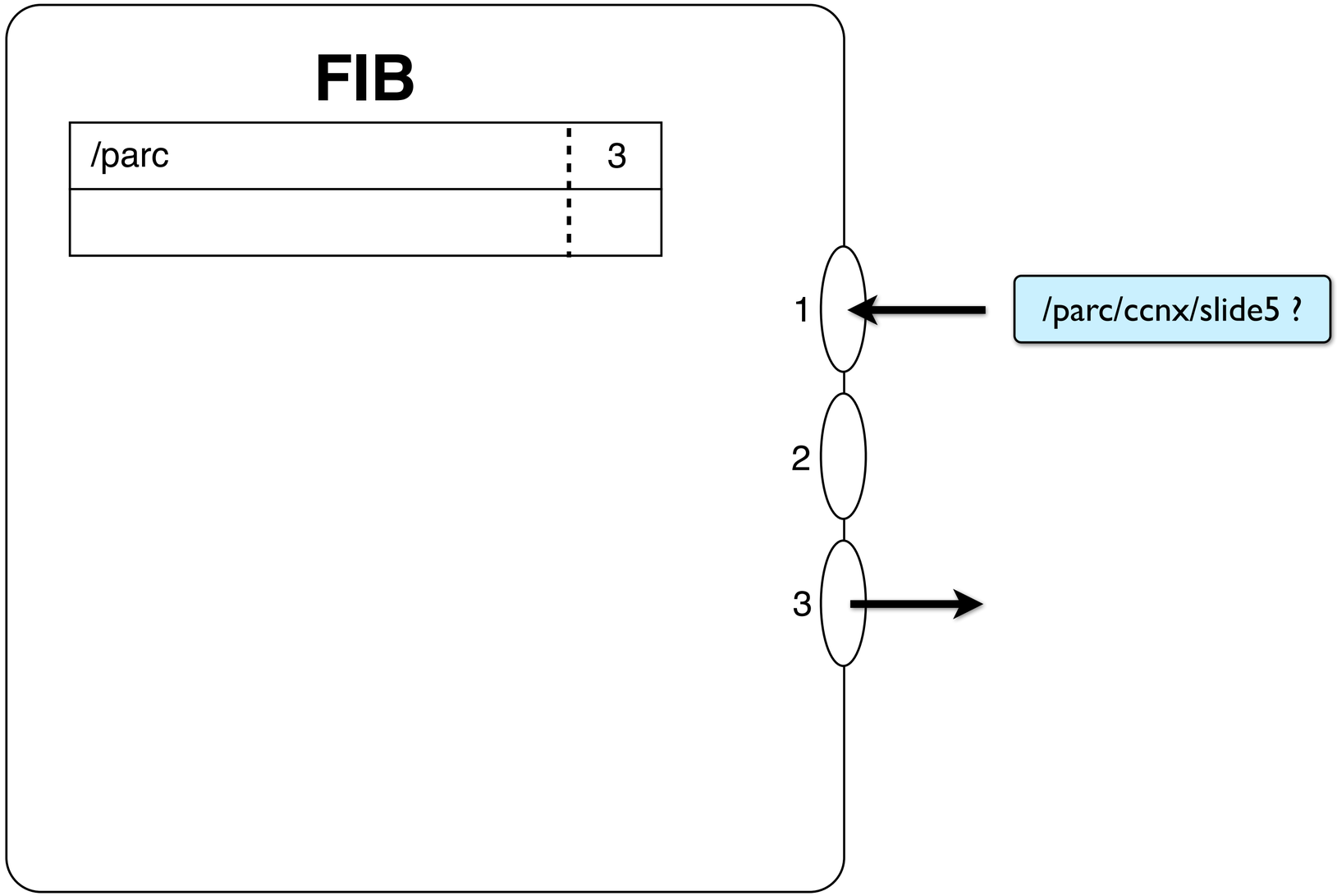


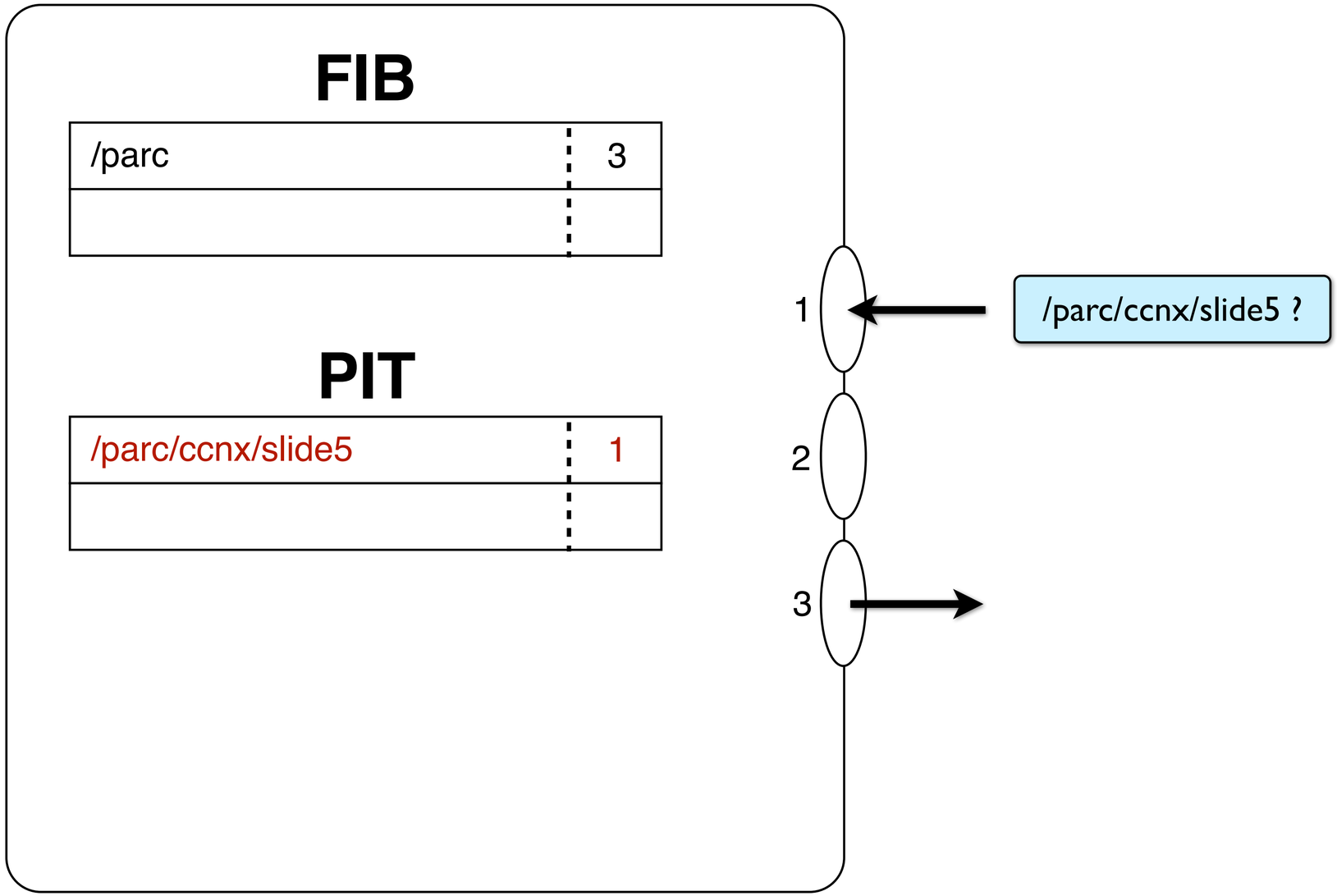


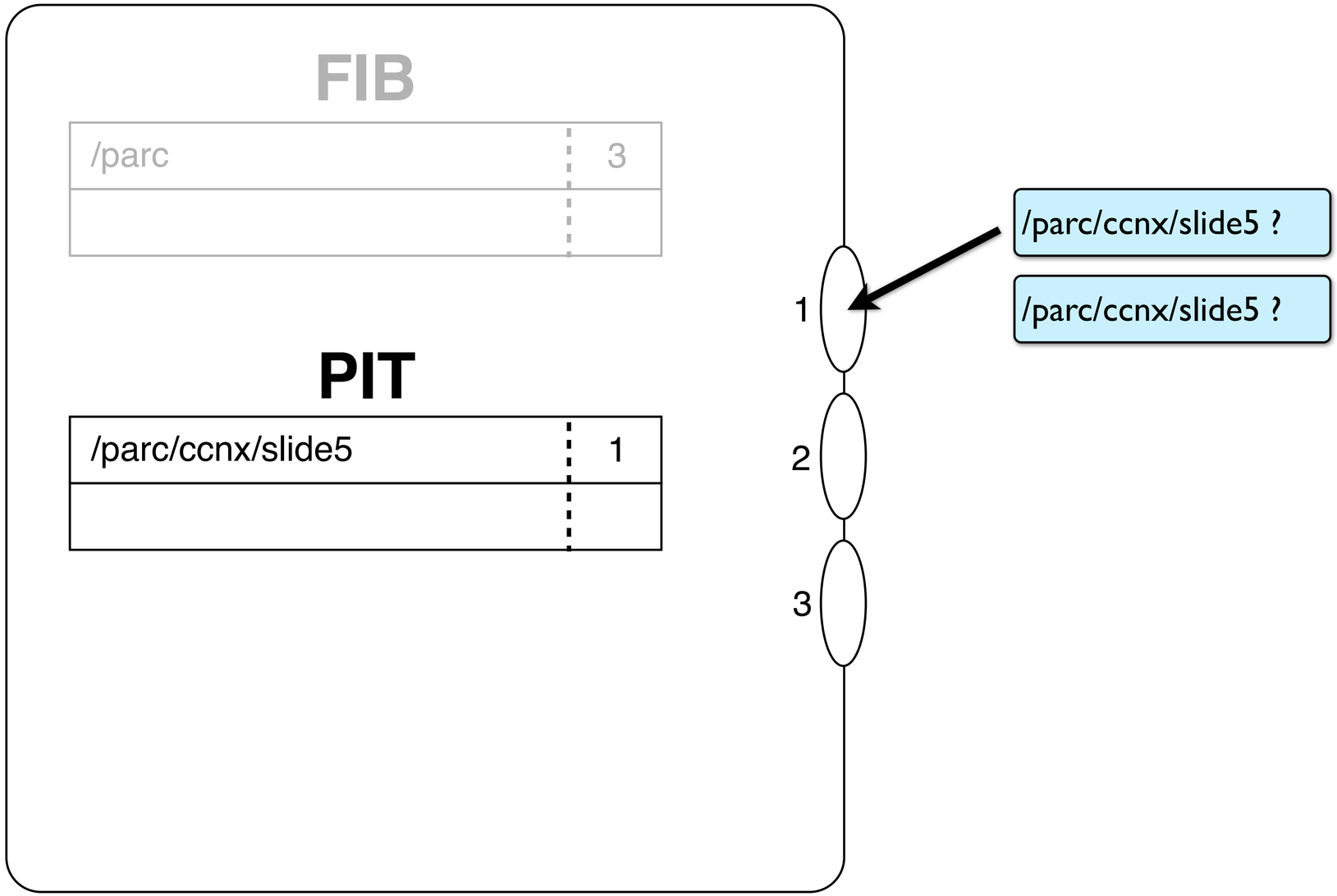

parc
A Xerox Company

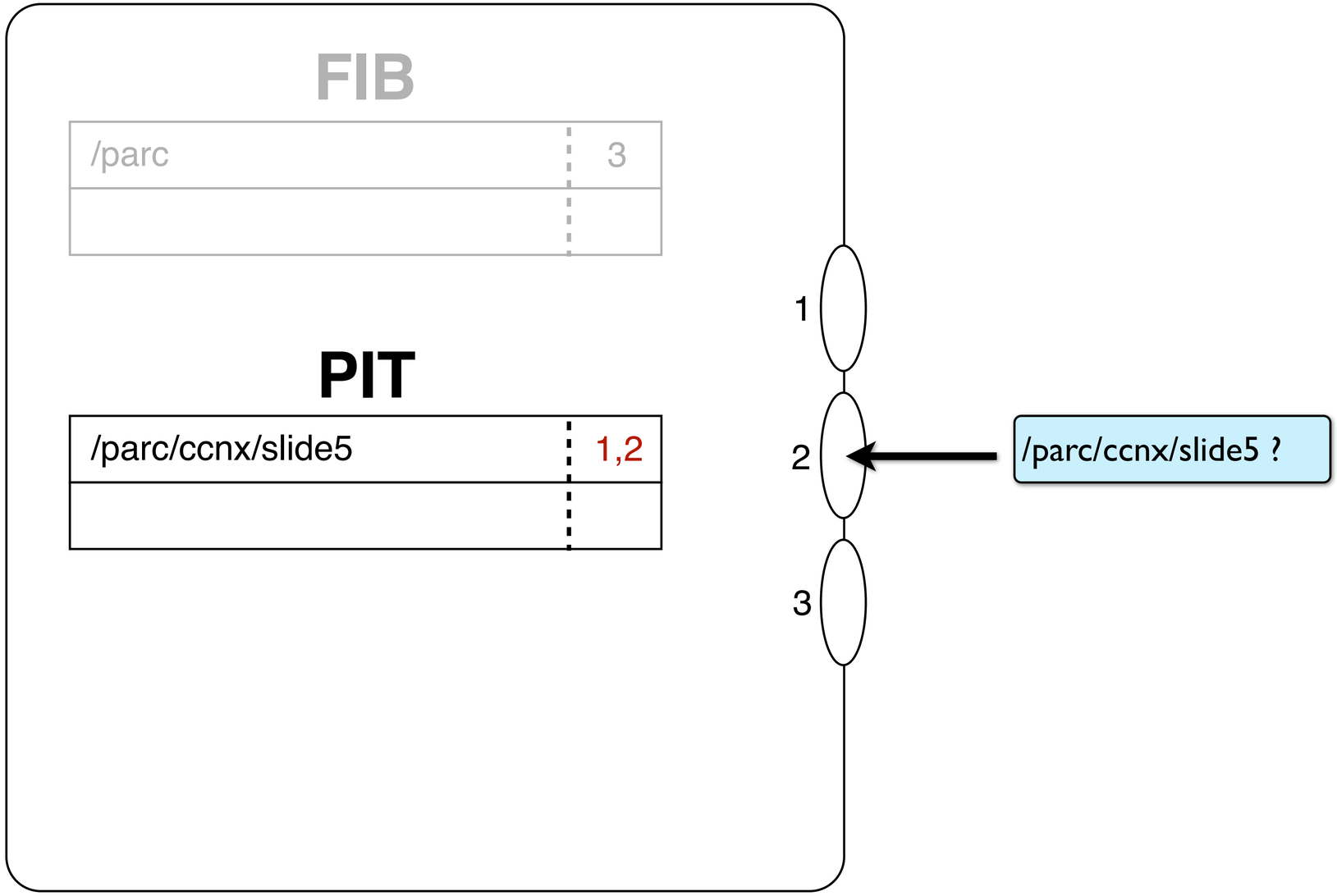
# CCN - Forwarder

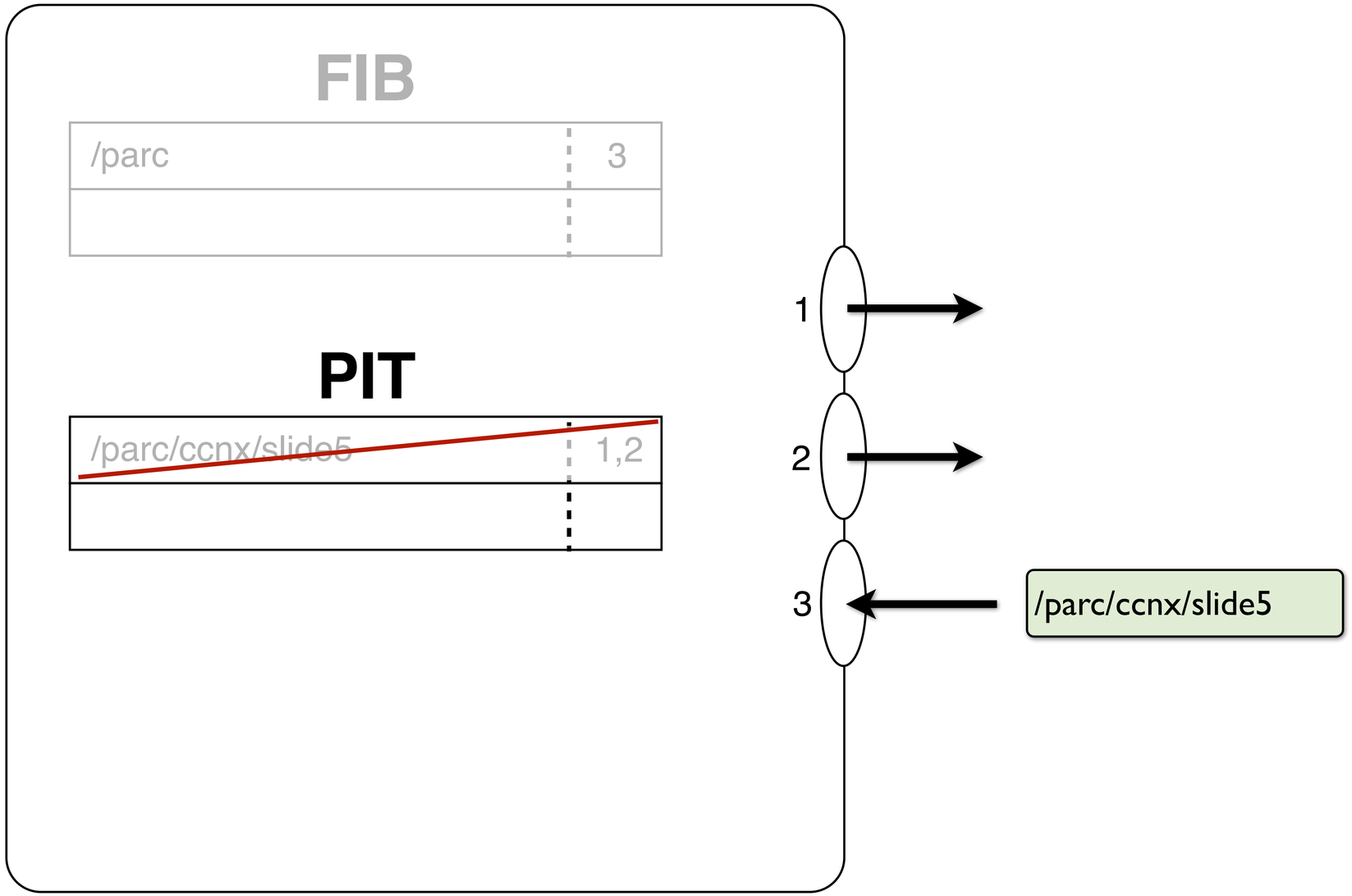
parc
A Xerox Company

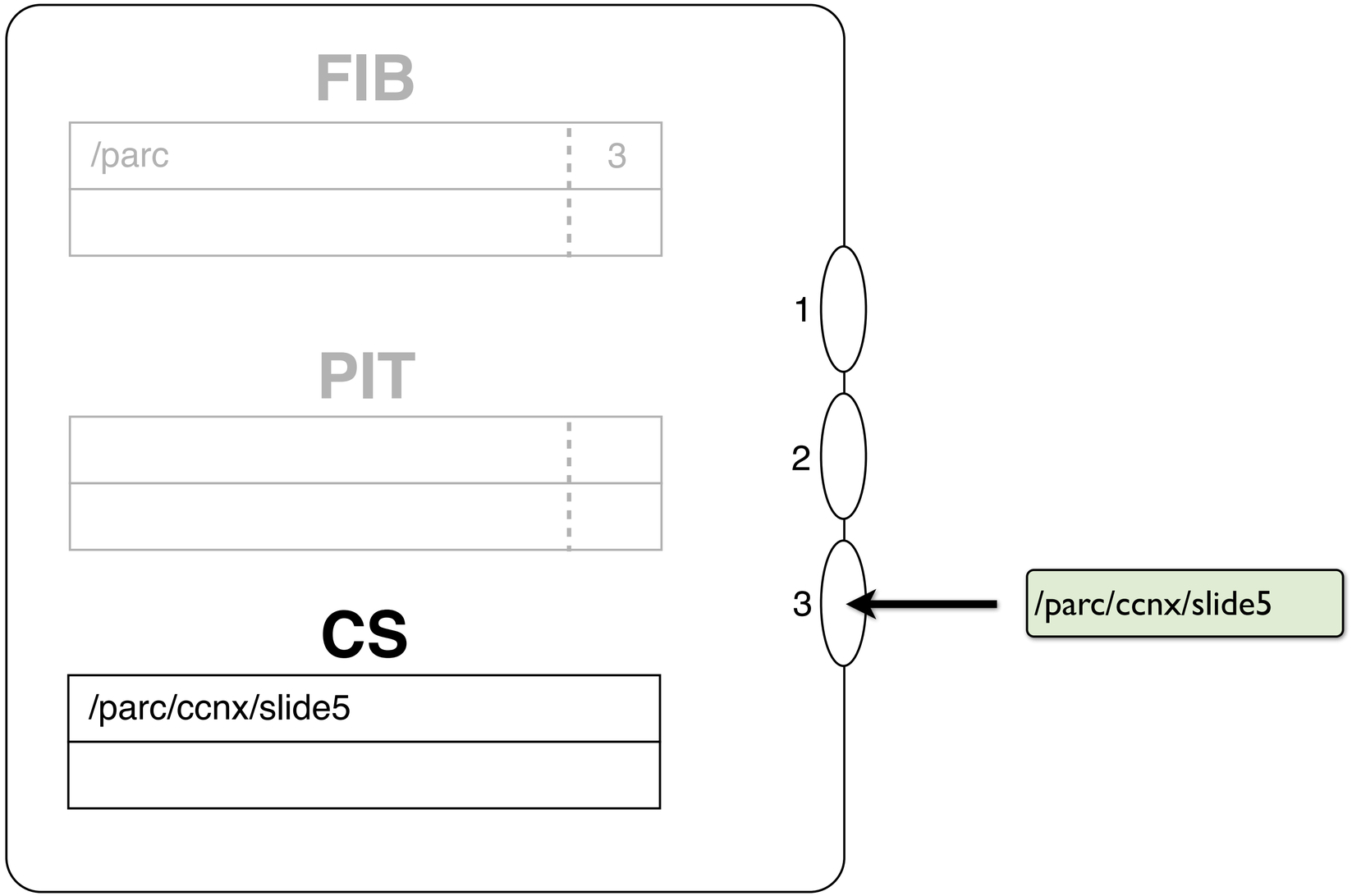


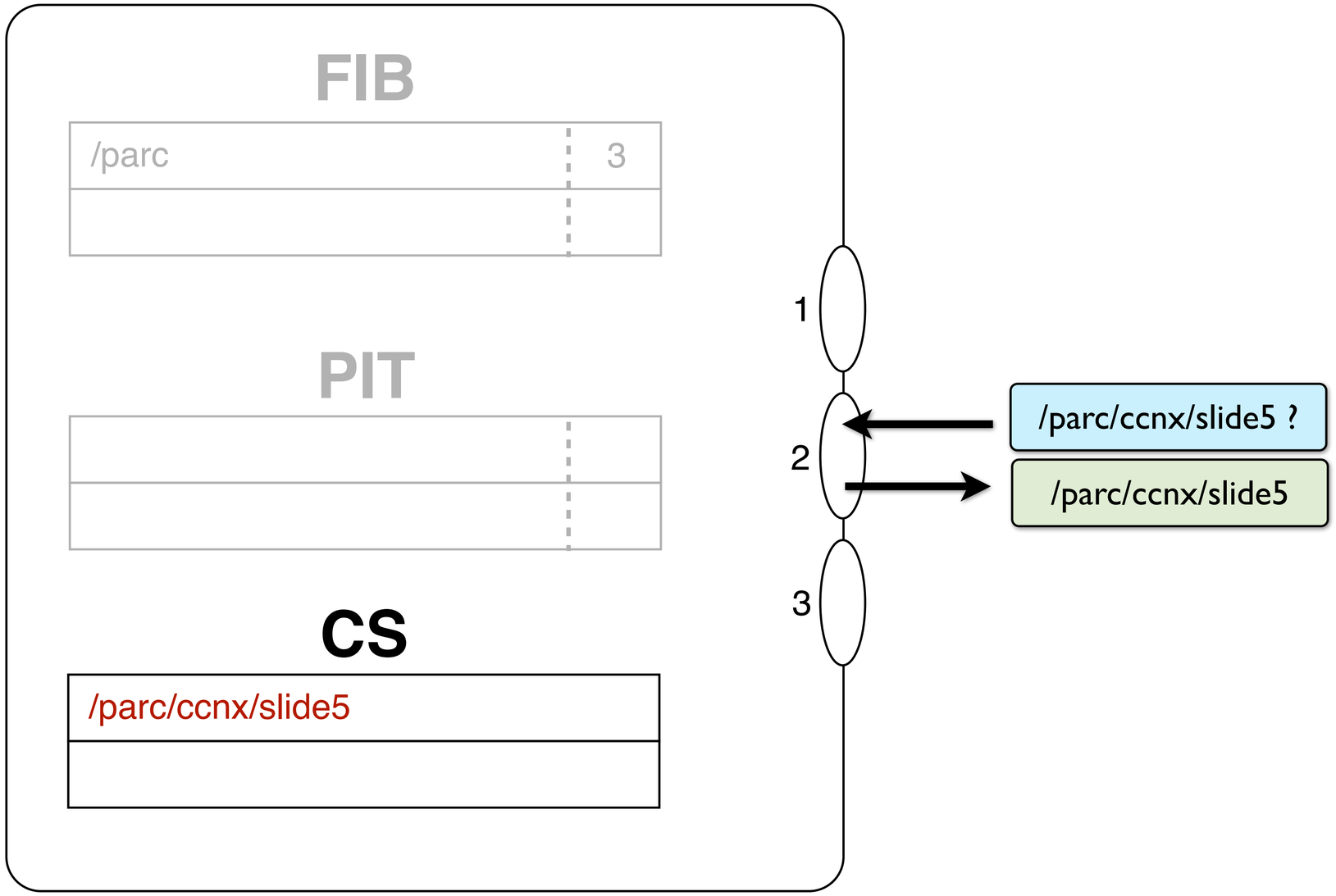


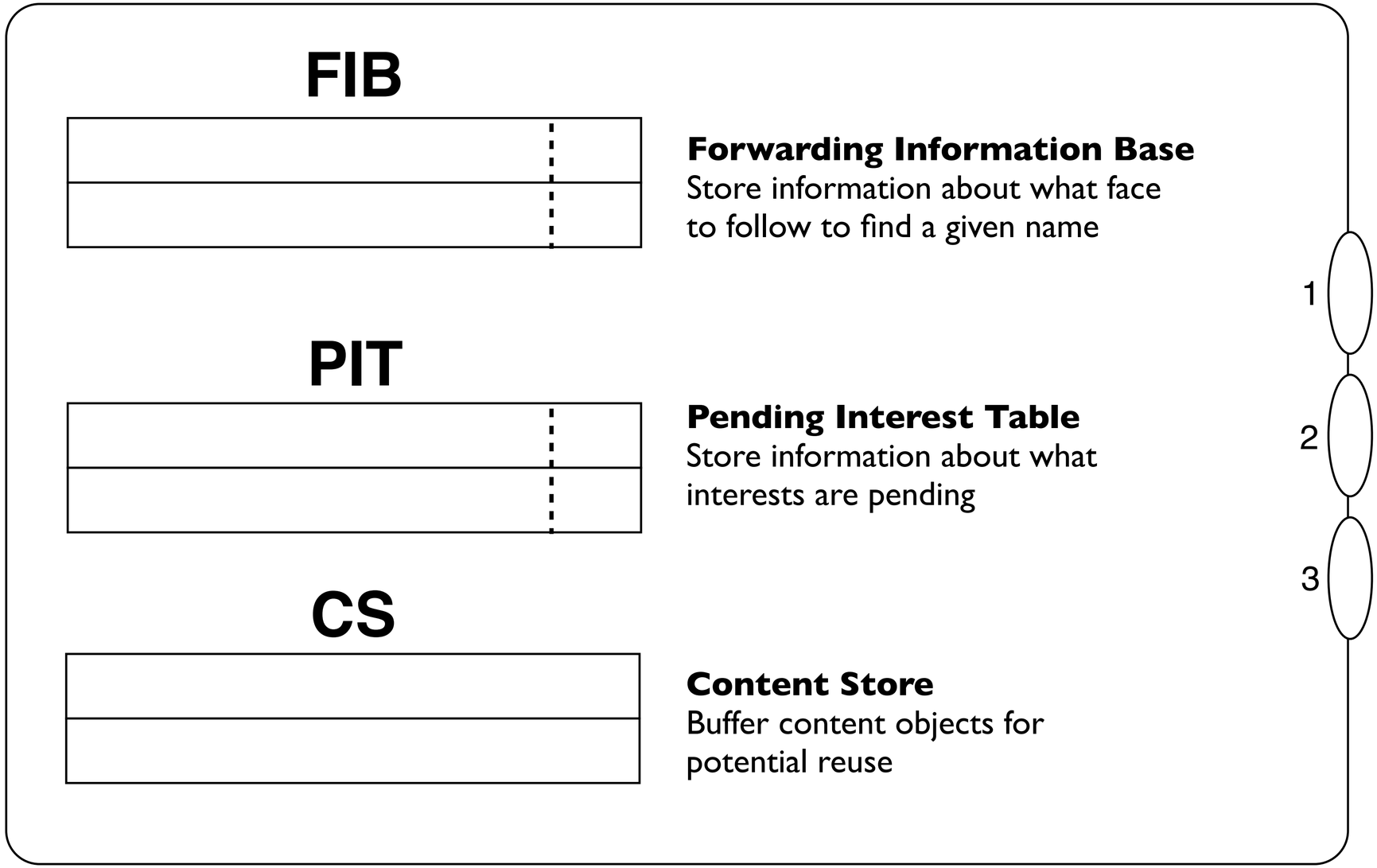












# Basic concepts

# CCN overview

**Step 1 - Name the data**

Name every piece of network data

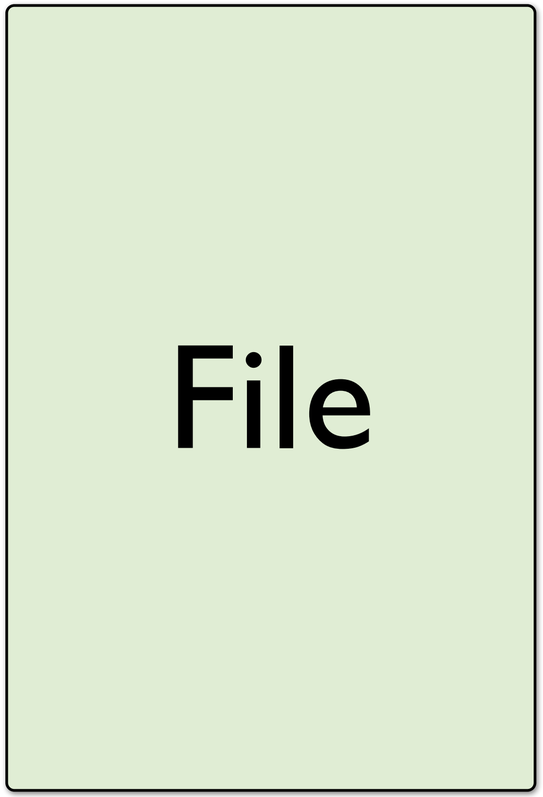
**Step 2 - Secure the data**

Secure every piece of network data

**Step 3 - Transfer the data**

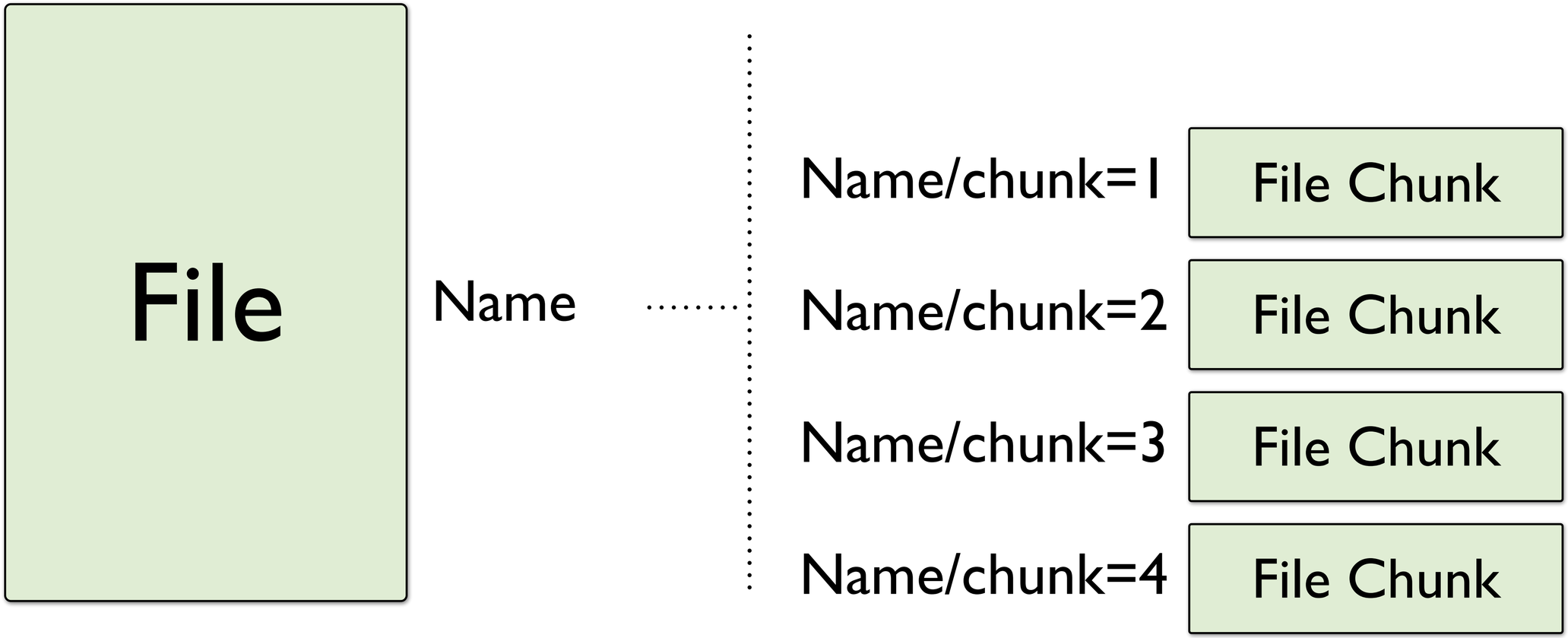
Move the data to interested recipients

# name data

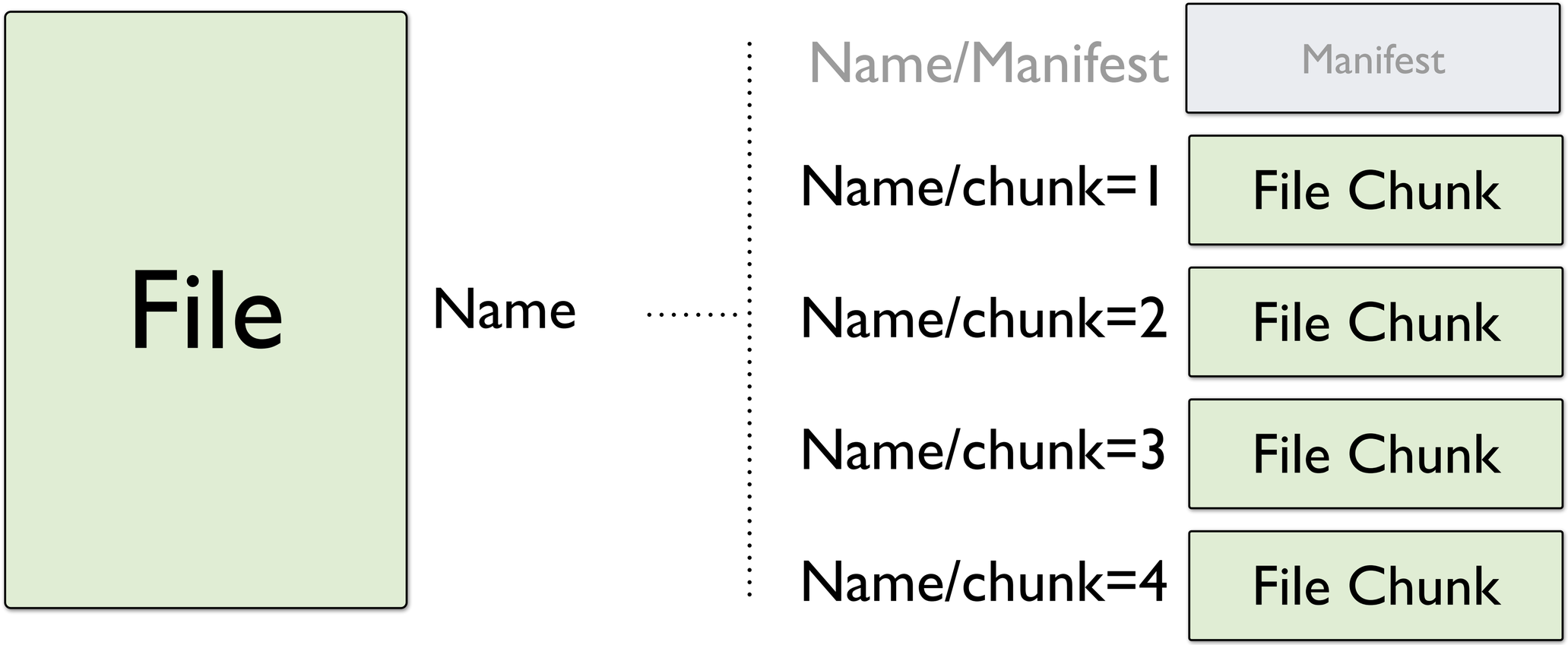



Name

A large object like a file has a CCN name



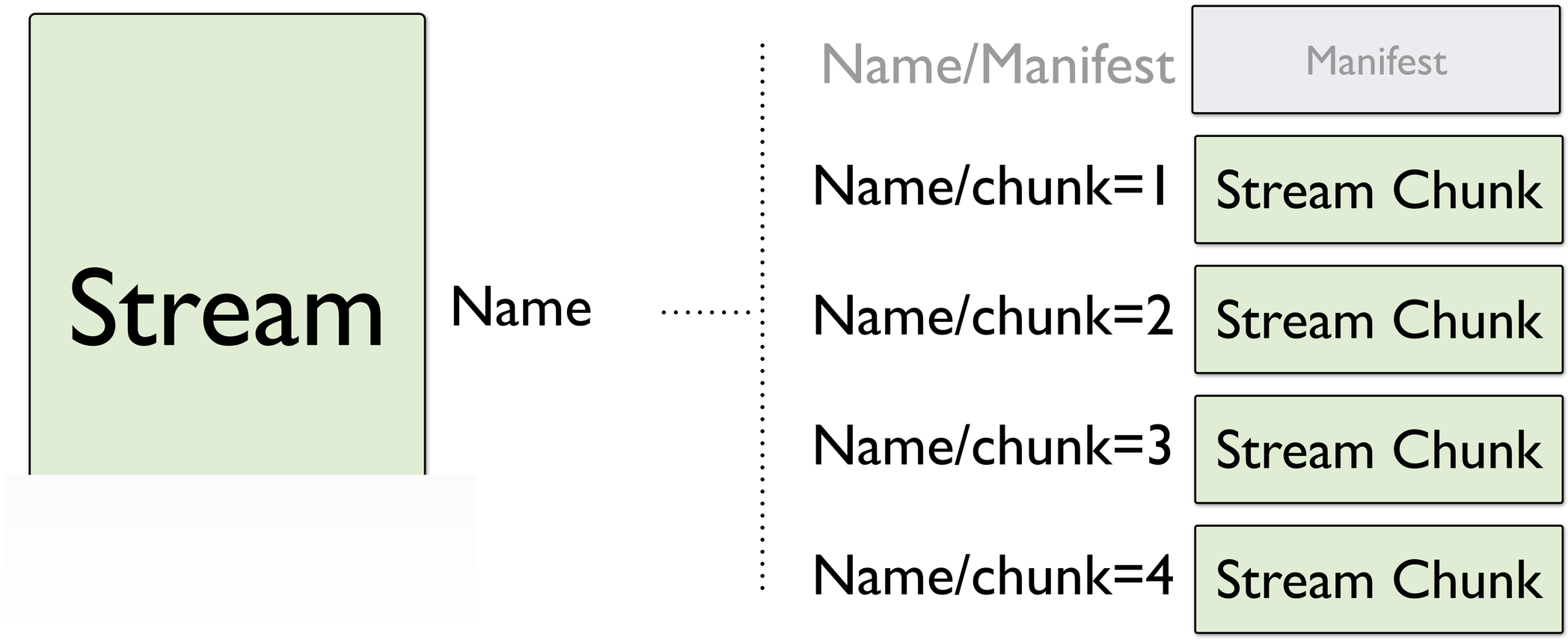

CCN also names the network sized chunks




CCN creates a manifest describing the file


Name

A large object like a file has a CCN name




The stream manifest is the metadata for the stream.

# CCN name uses

## Routing/Forwarding
Find the source of the data

## Demultiplex
Choose between options

## Matching
Determine equivalence

## Structure
Inherent information

# CCN name format

**Name Segment based**
Labeled binary segments

**Hierarchical structure**
Ordered sequence of segments

/seg1/seg2/seg3/seg4

# CCN name example

/parc/ccnx/presentations/slide20/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)
Locally coordinated namespace (like subdomains)

## 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 qualifier

/parc/ccnx/presentations/slide20/v=2/c=0

ContentObjectHash

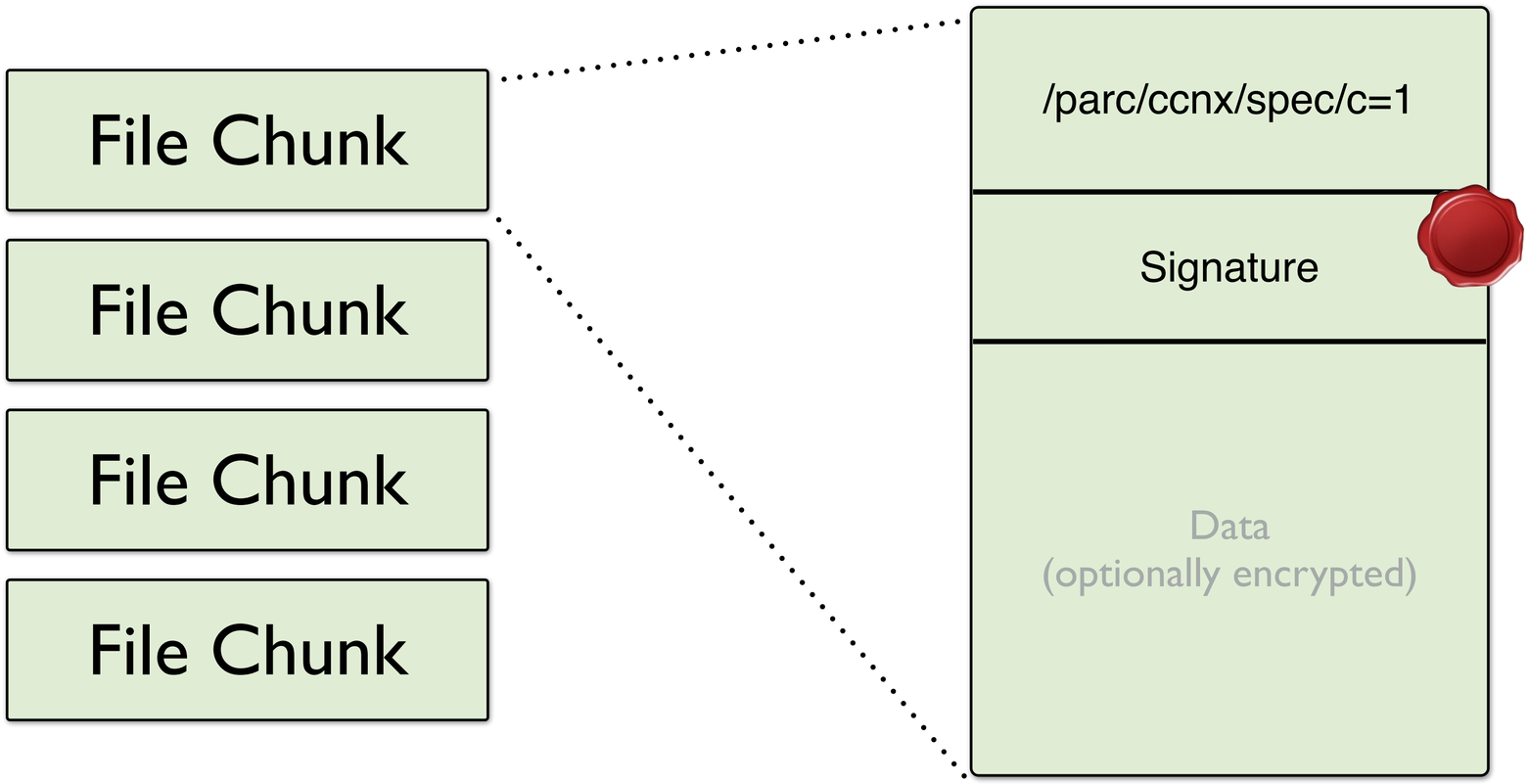
hash of the content object message

KeyId

identifier of key used to sign the object

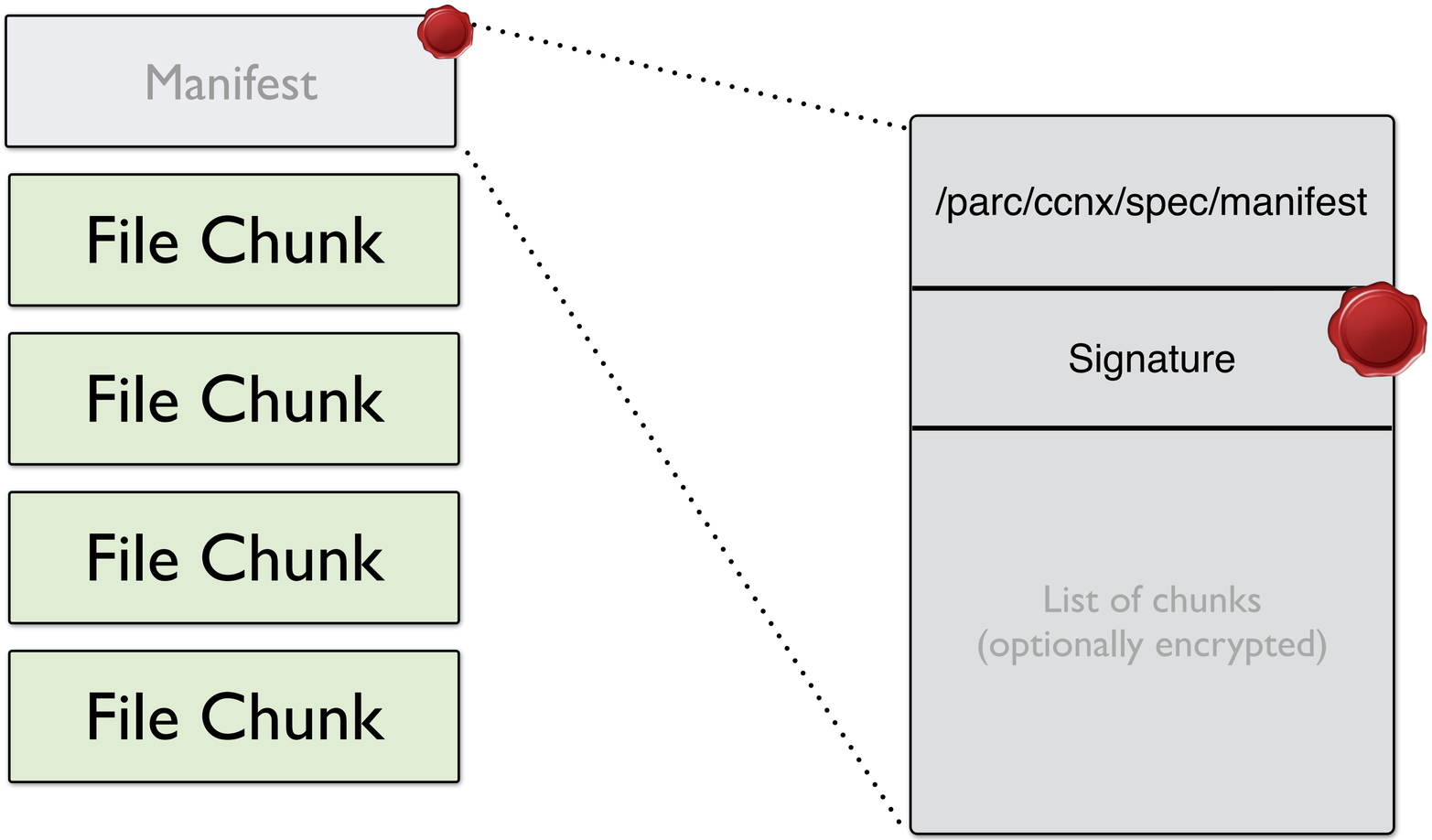
# secure data

# Secure single chunks



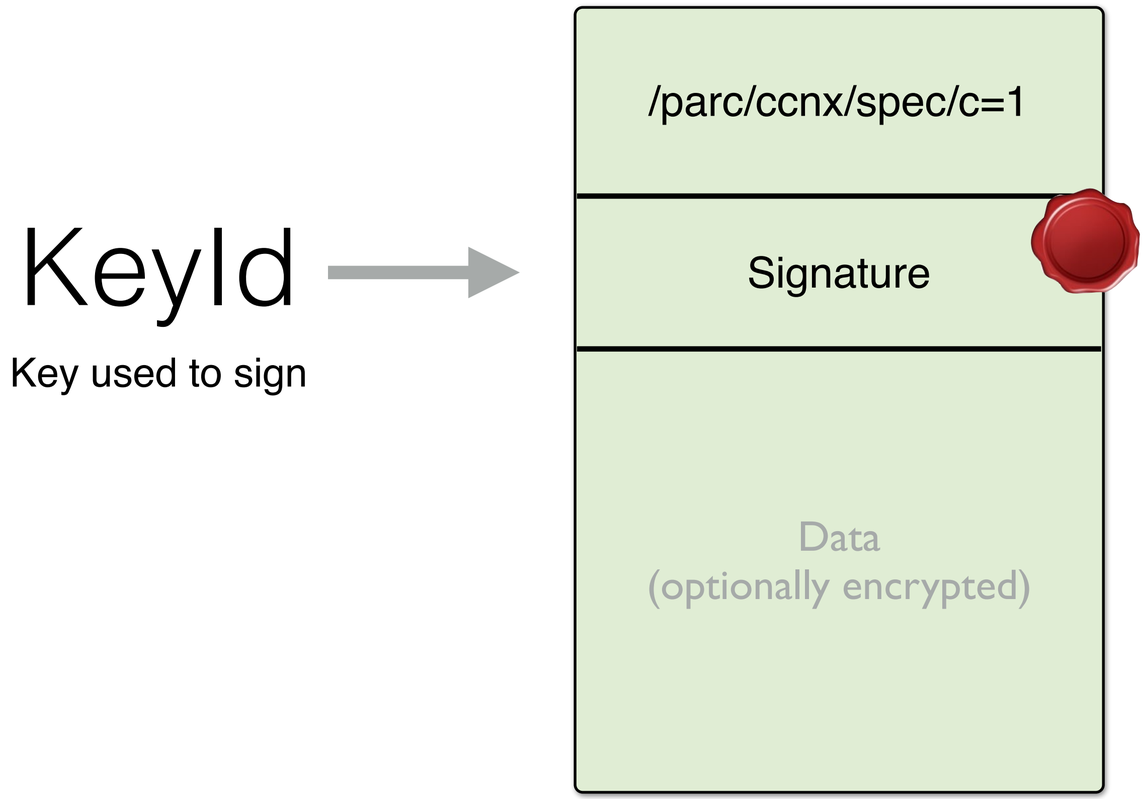

CCN names and signs every chunk

# Secure whole object via manifest




CCN names and signs the file via a manifest

# Secure single chunks




Signature binds the name to a Key

# CCN name qualifier

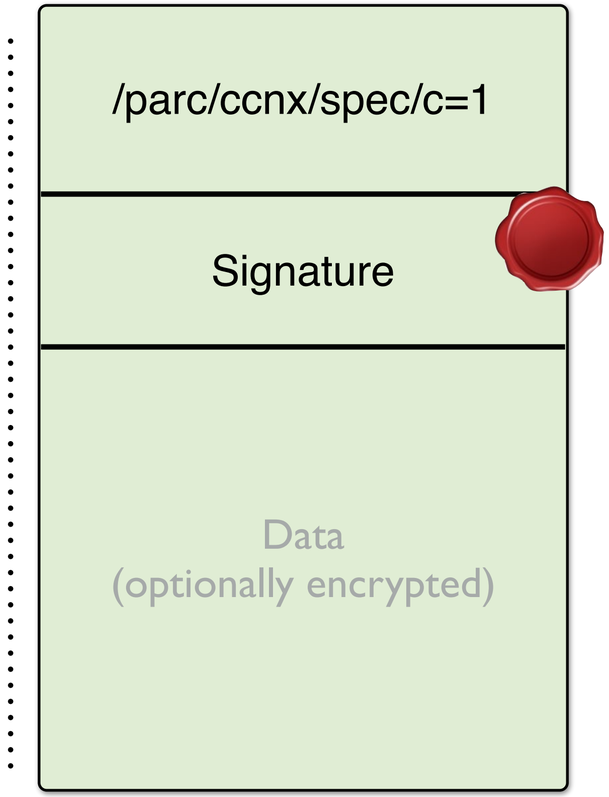
/parc/ccnx/presentations/slide20/v=2/c=0

KeyId

identifier of key used to sign the object

# Secure single chunks

Content Object Hash




Signature binds the name to a key

# CCN name qualifier

/parc/ccnx/presentations/slide20/v=2/c=0

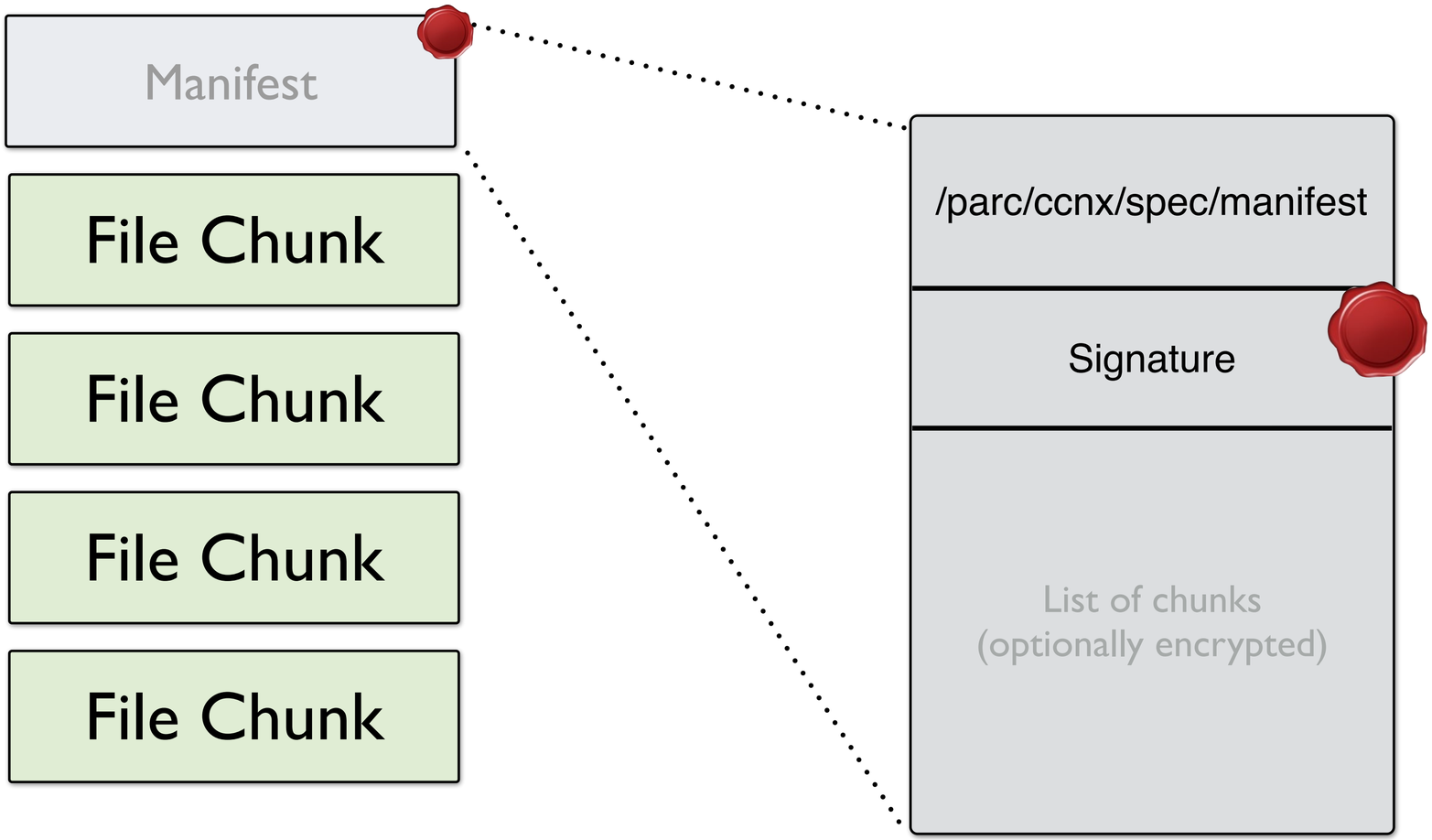
ContentObjectHash

hash of the content object message

KeyId

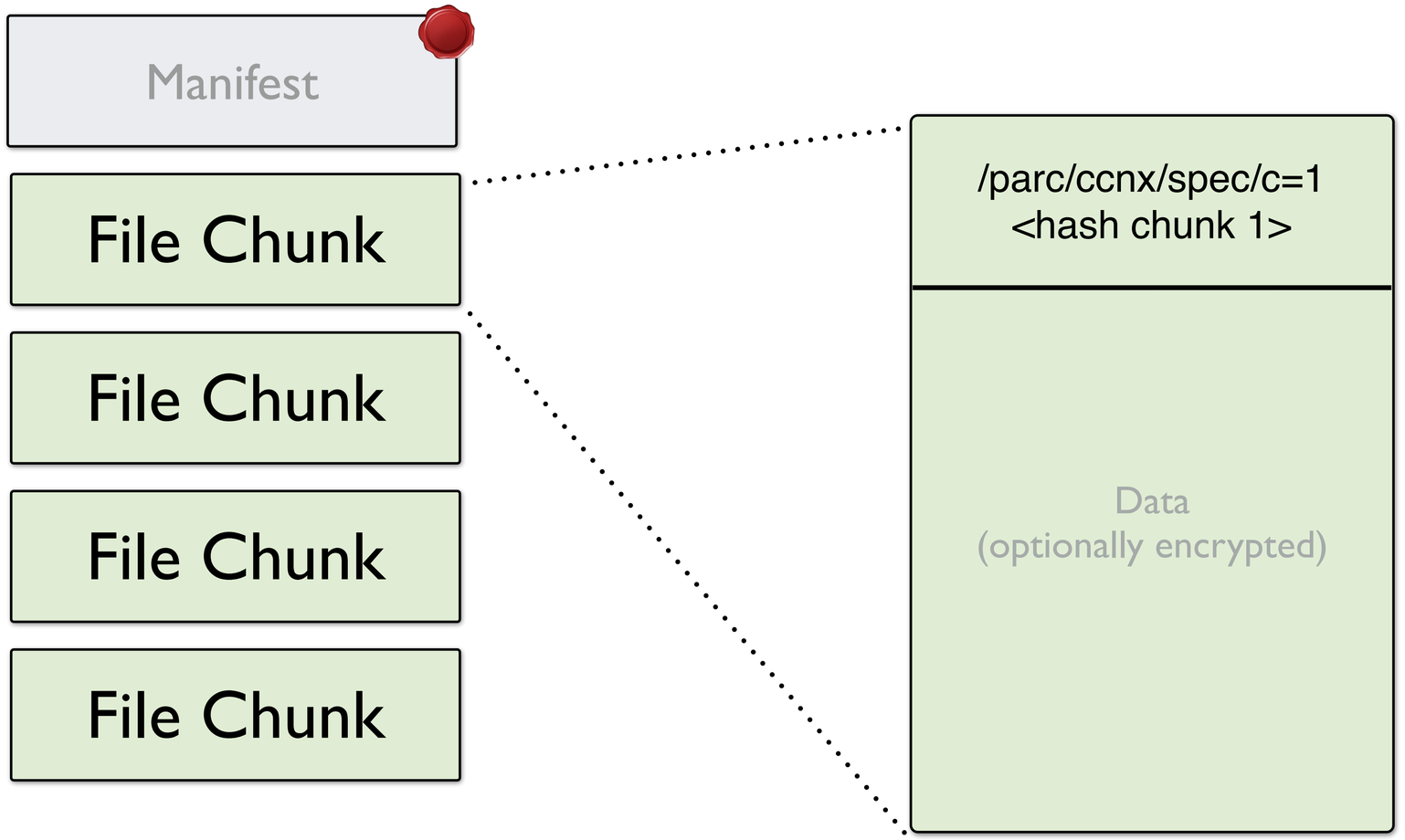
identifier of key used to sign the object

# Secure whole object via manifest




CCN names and signs the file via a manifest

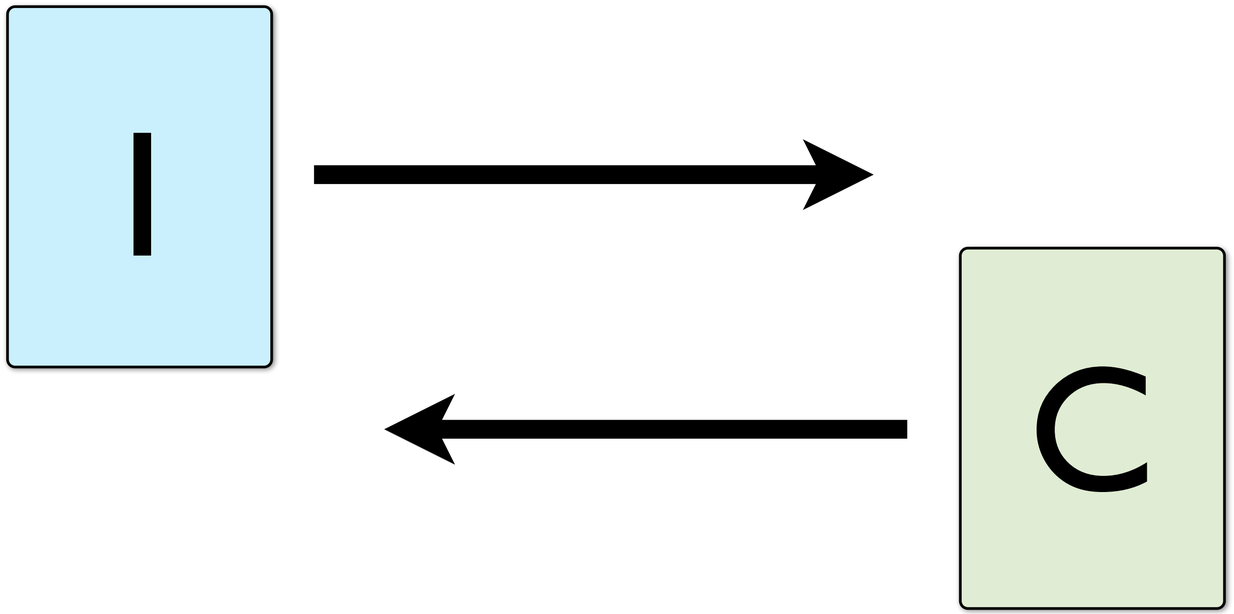
# Secure via manifest




Indirectly sign every chunk through the manifest

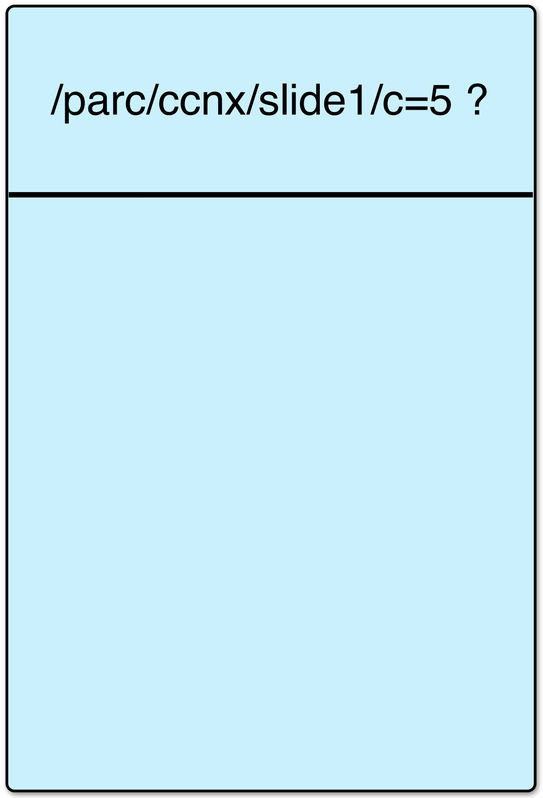
# transfer data

# Core Protocol

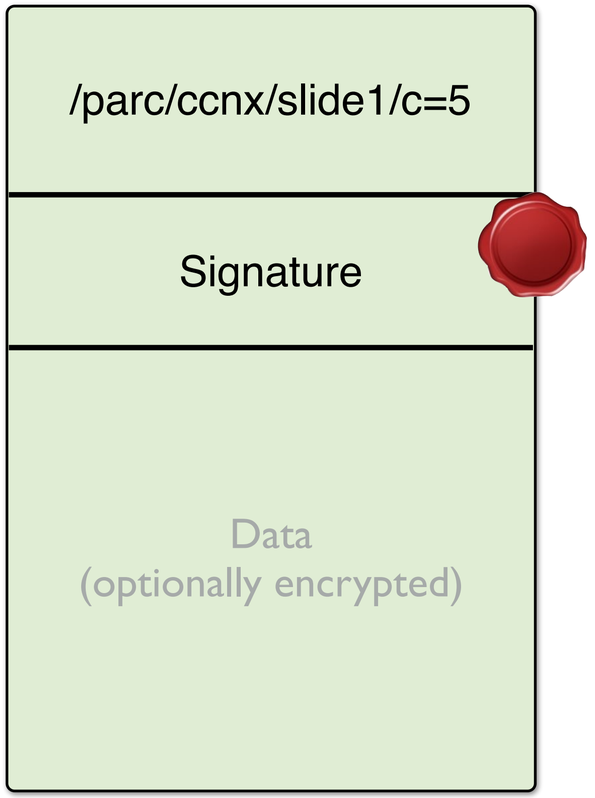



One interest packet gets one content packet

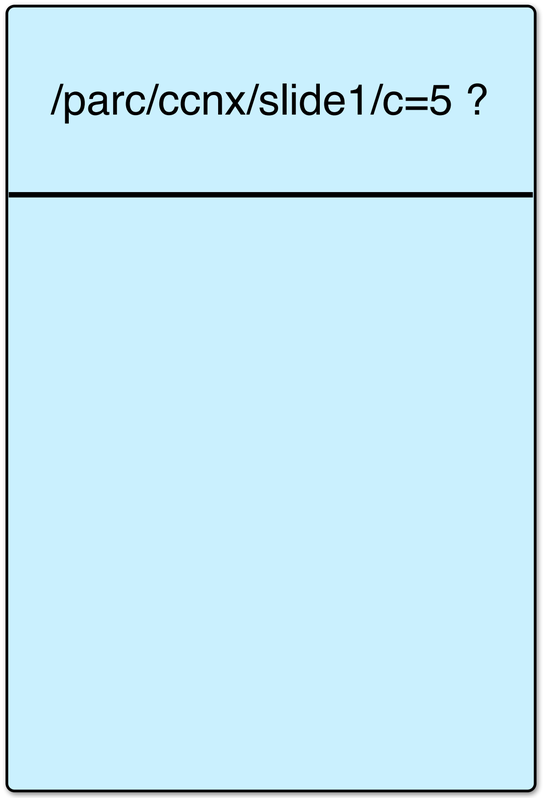
## Interest



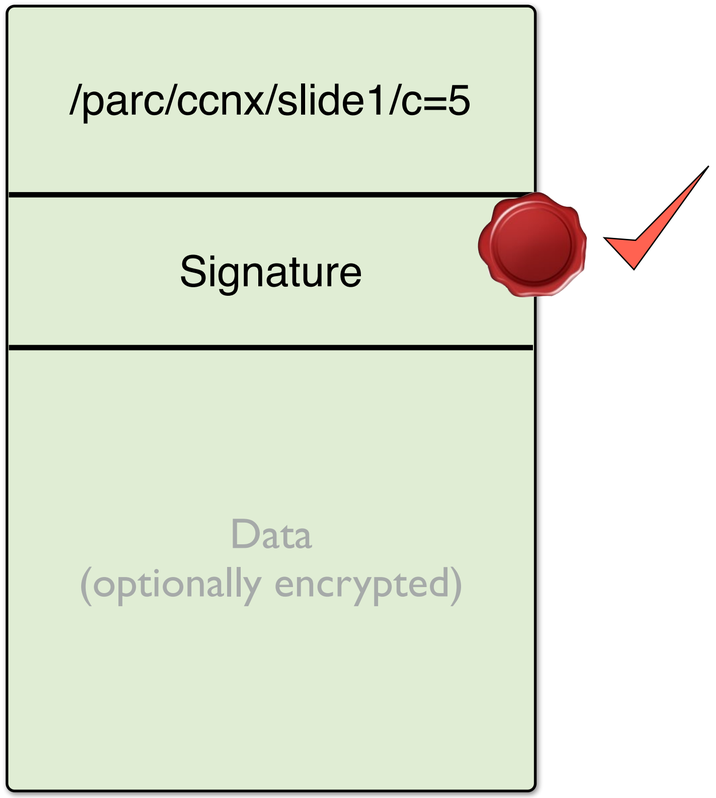

## Content Object



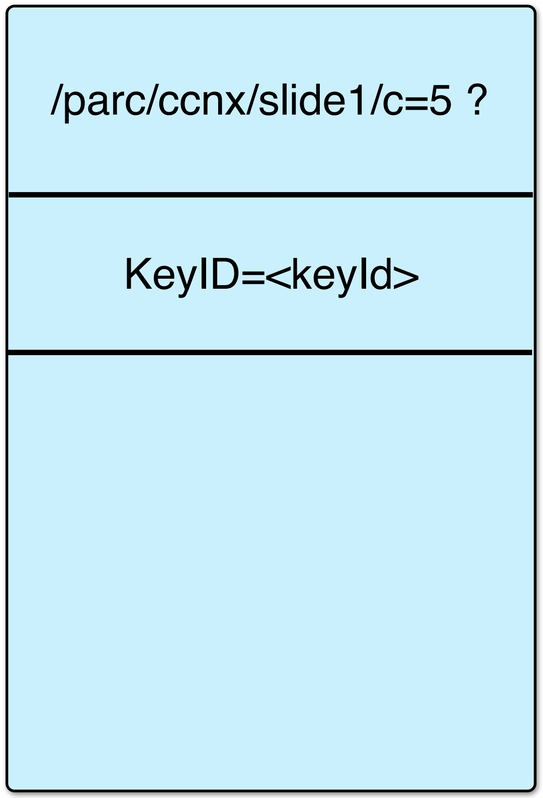
# Interest



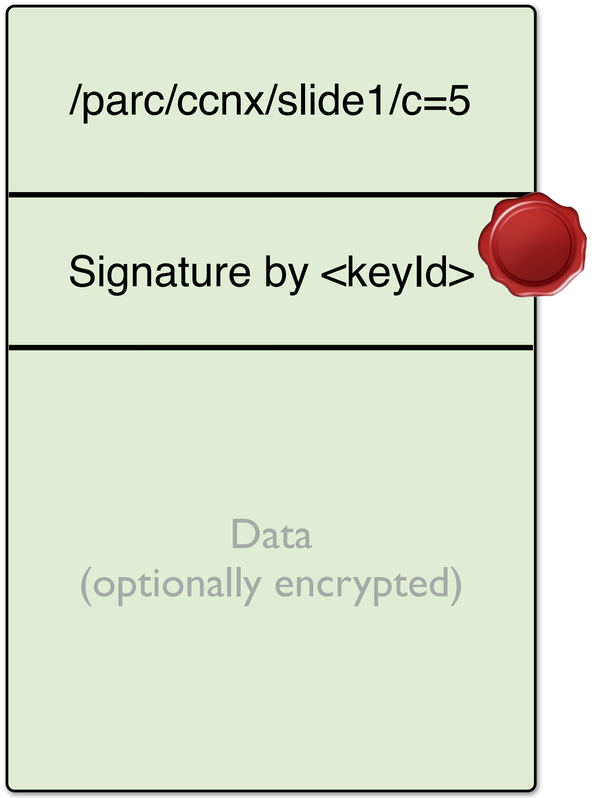

# Content Object



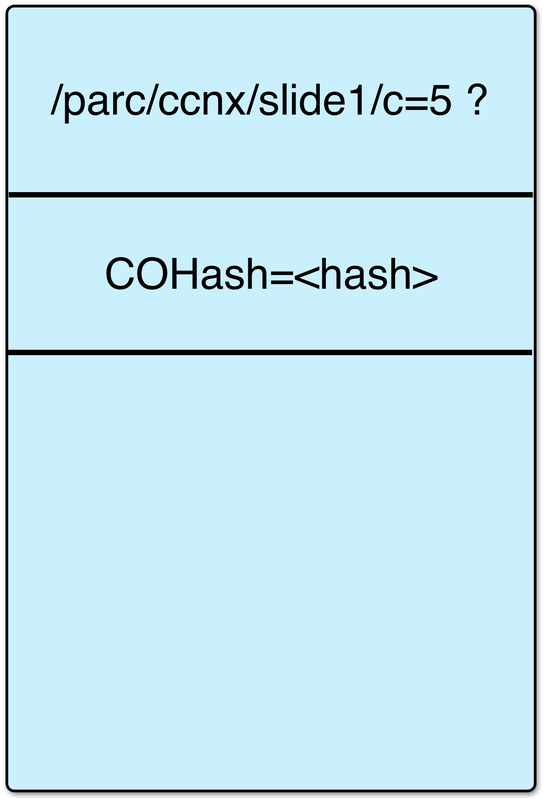
## Interest



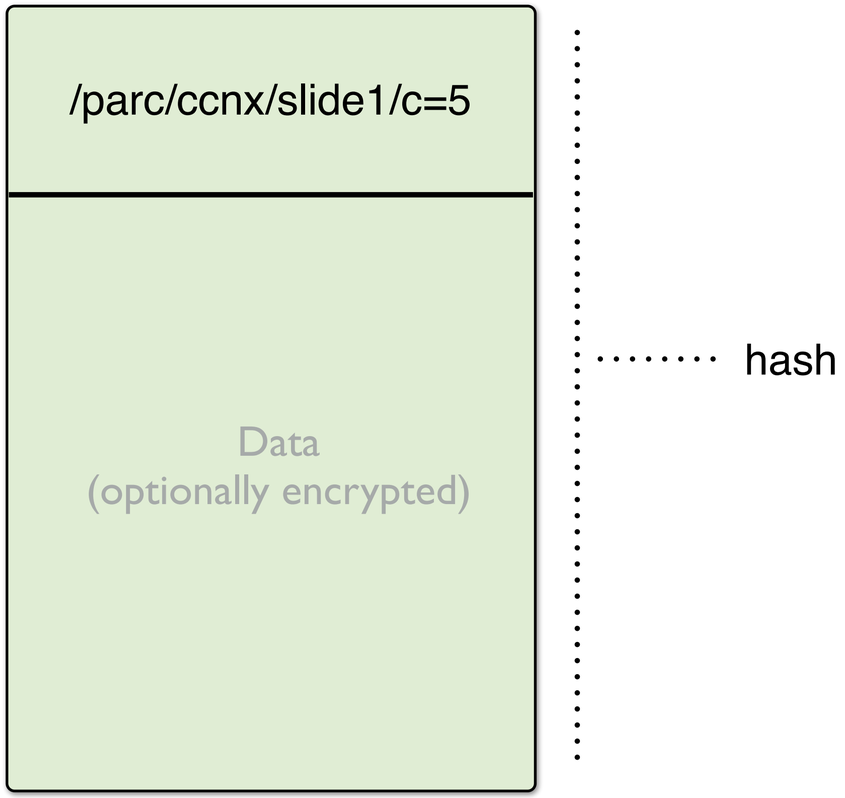

## Content Object



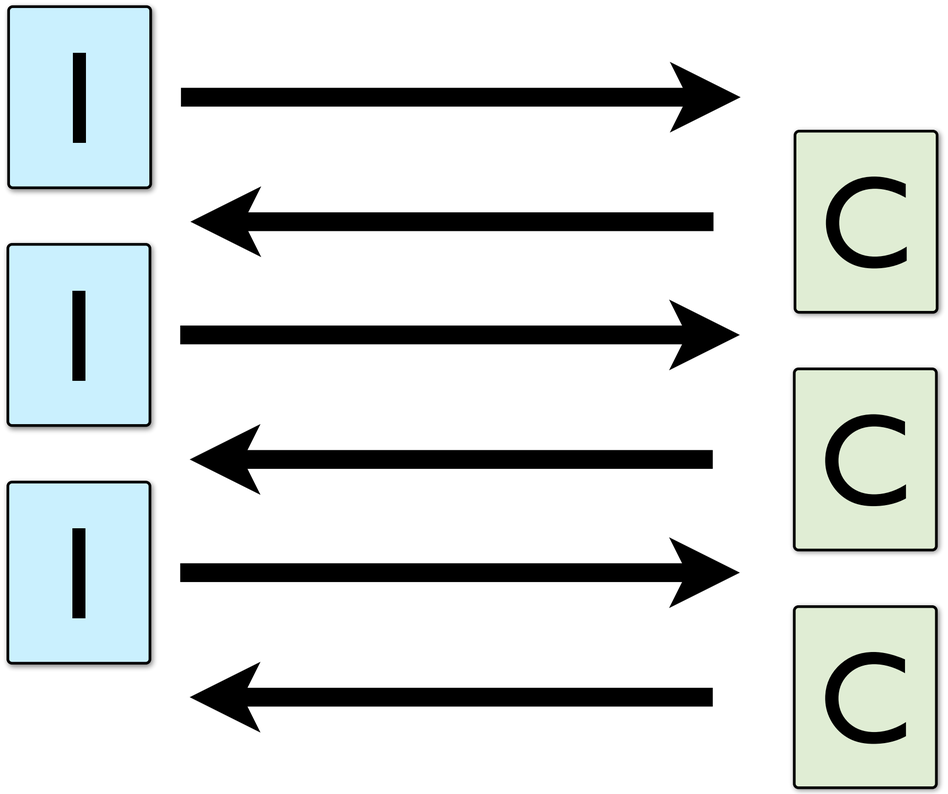
## Interest




## Content Object

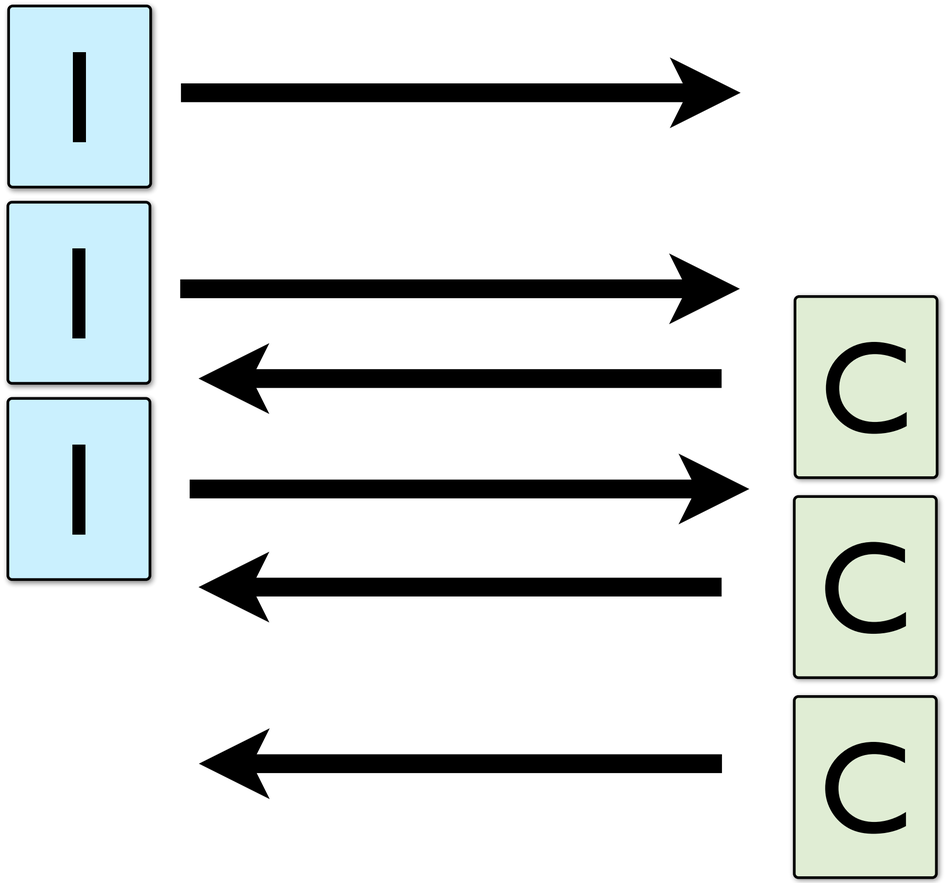


# Transport Protocols



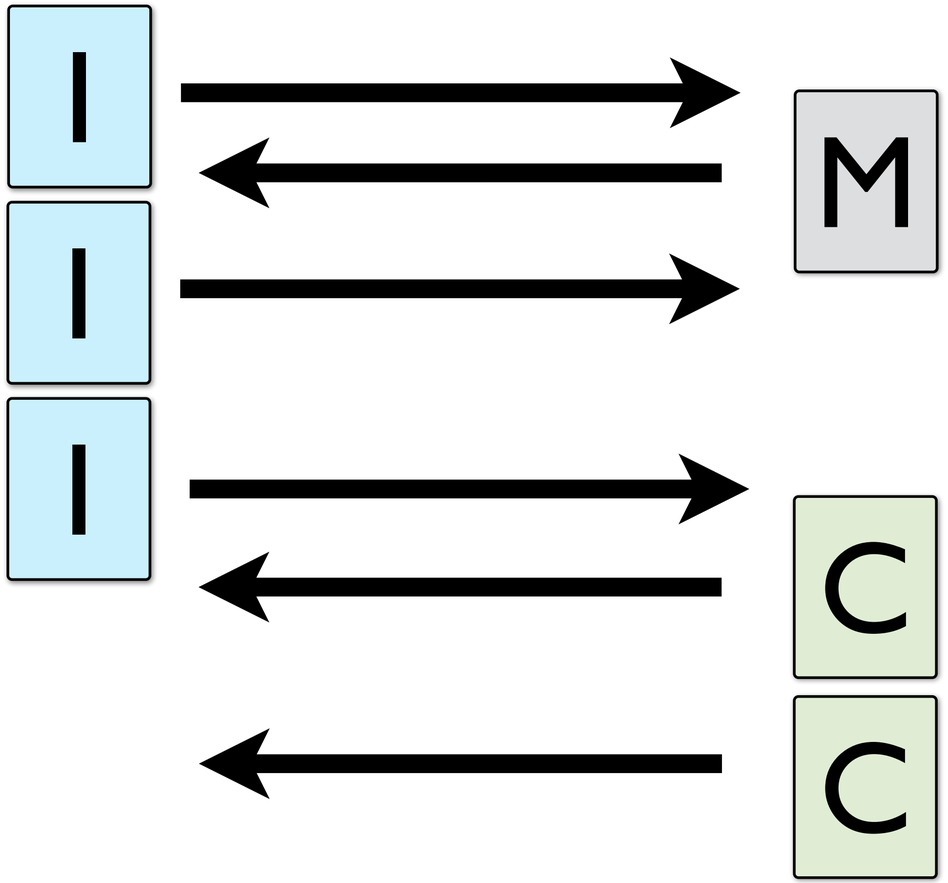

Transport protocols are built on core exchanges

# Transport Protocols



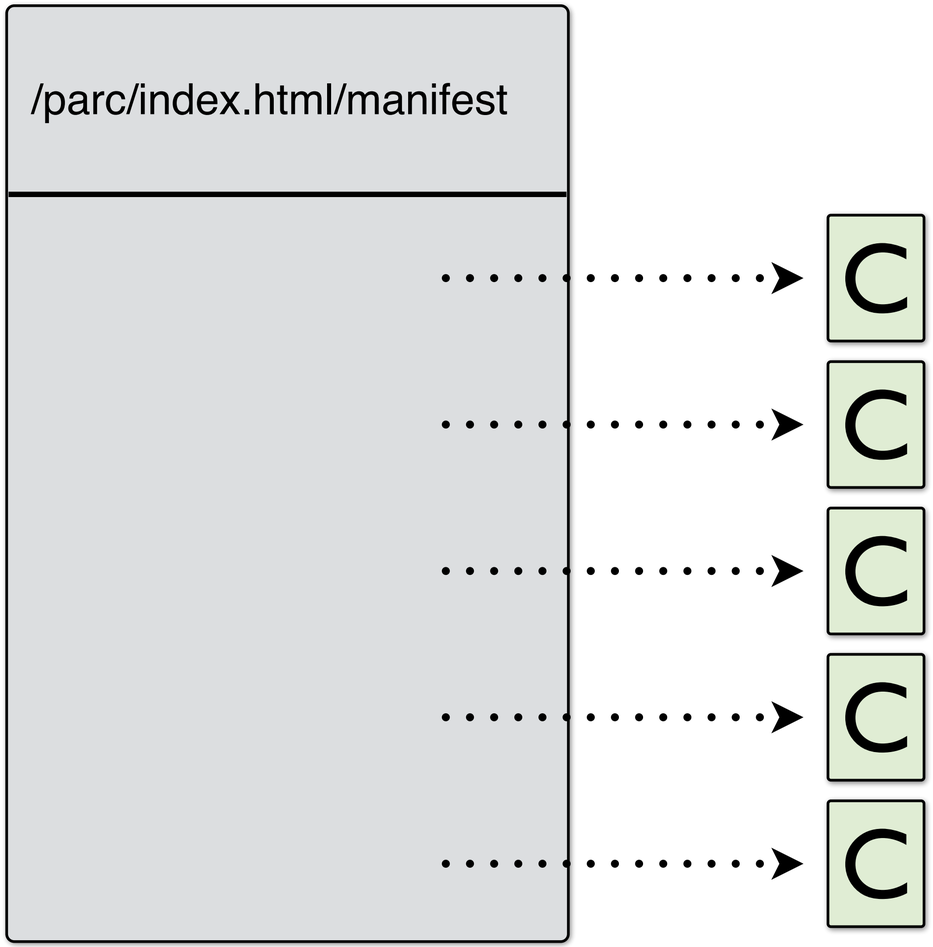

Transport protocols can do parallel requests

# Transport Protocols

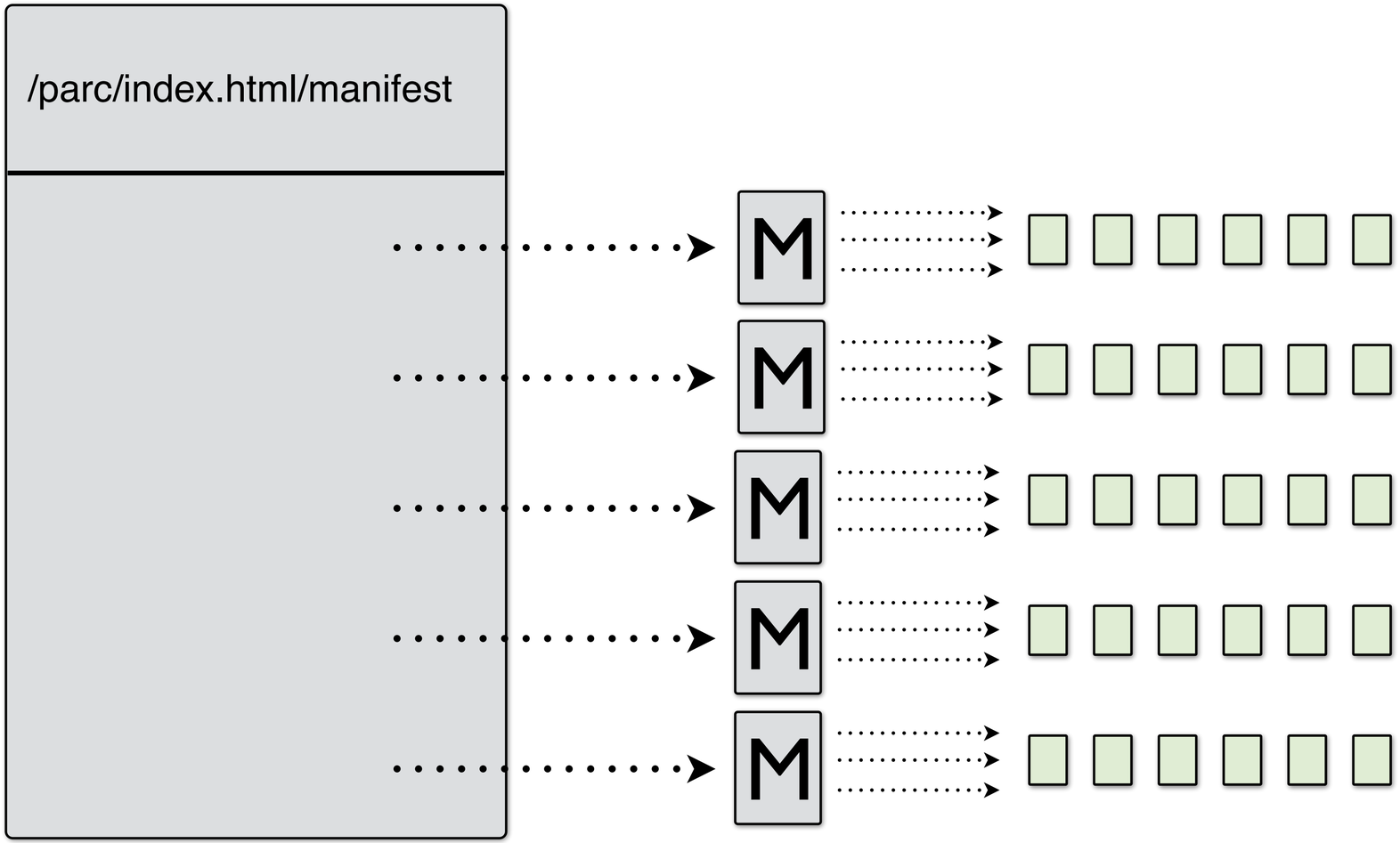



Transport protocols can use manifests

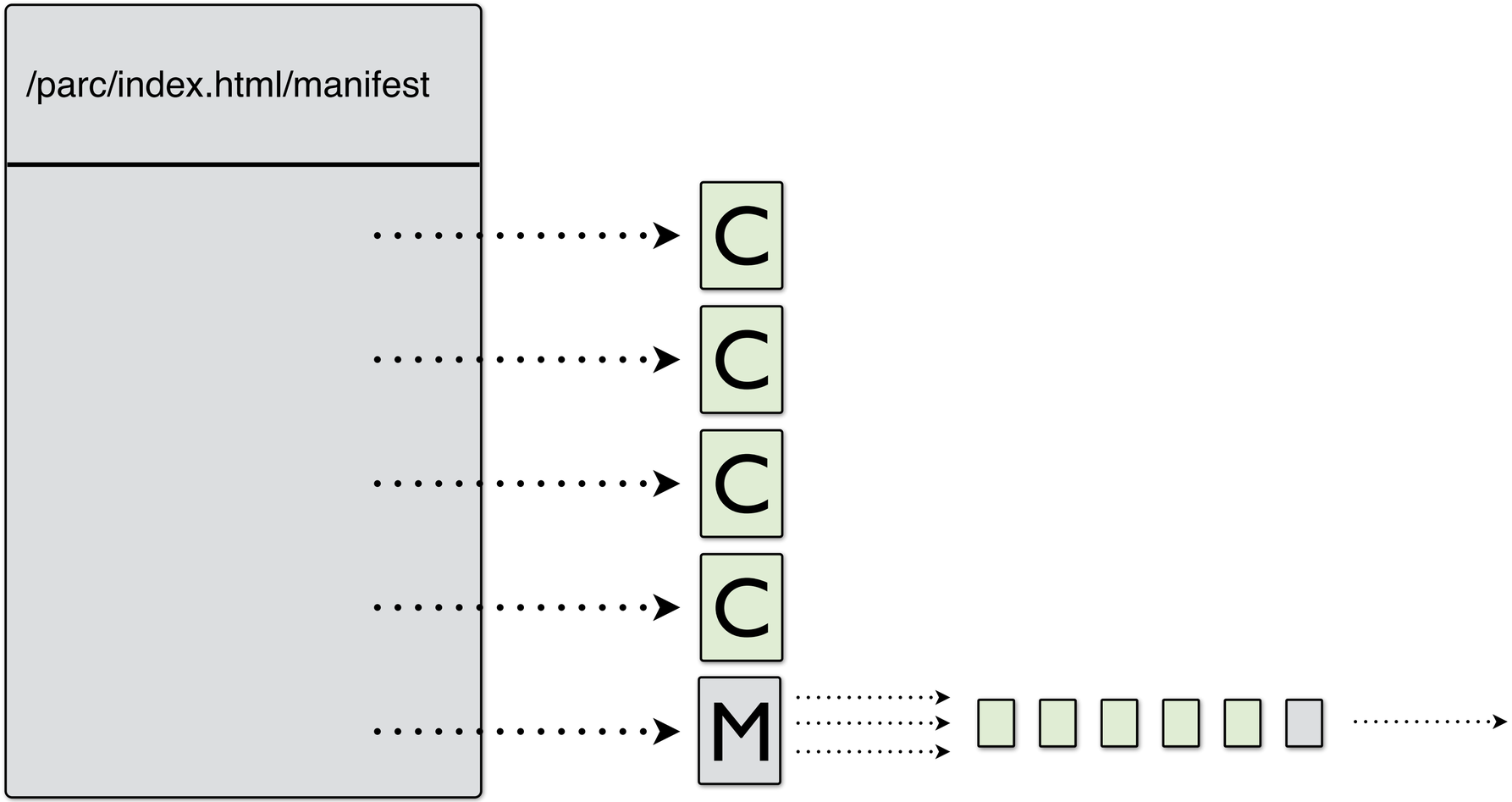
# Manifest structure



# Manifest structure



# Manifest structure



# Manifest structure