CCN Progress Report

Nacho Solis, Principal Scientist
CCN Team

2014-10-21

parc®
A Xerox Company

Quick tutorial

Content-centric Networking (CCN)

is a

communication architecture

based on

transferring named data

CCN overview

Step 1 - Name the data

Name every piece of network data

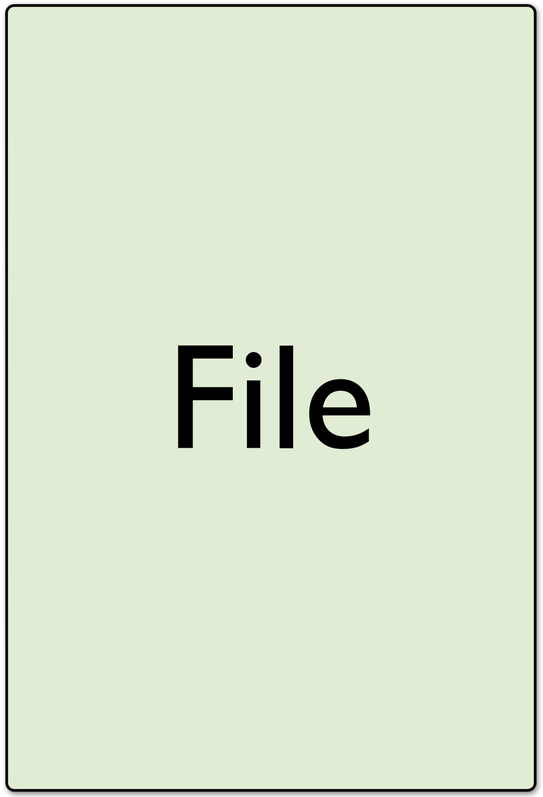
Step 2 - Secure the data

Secure every piece of network data

Step 3 - Transfer the data

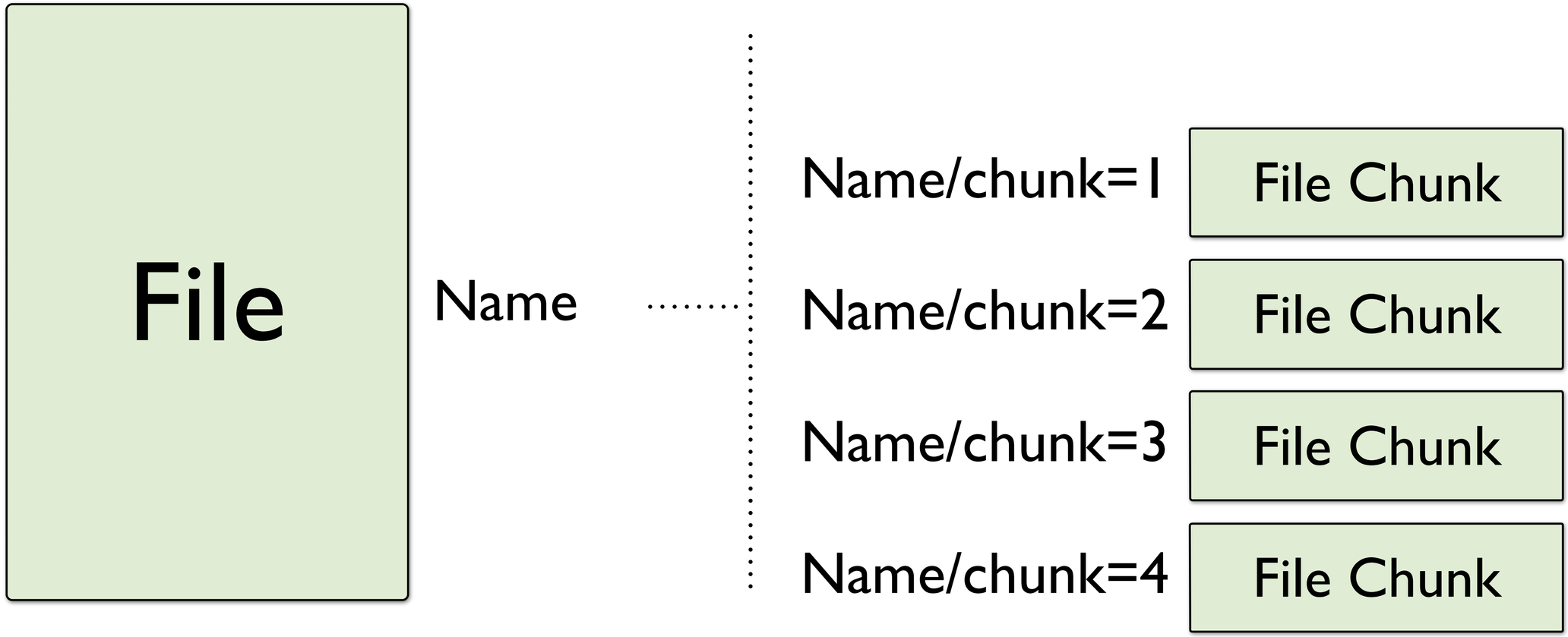
Move the data to interested recipients

name data

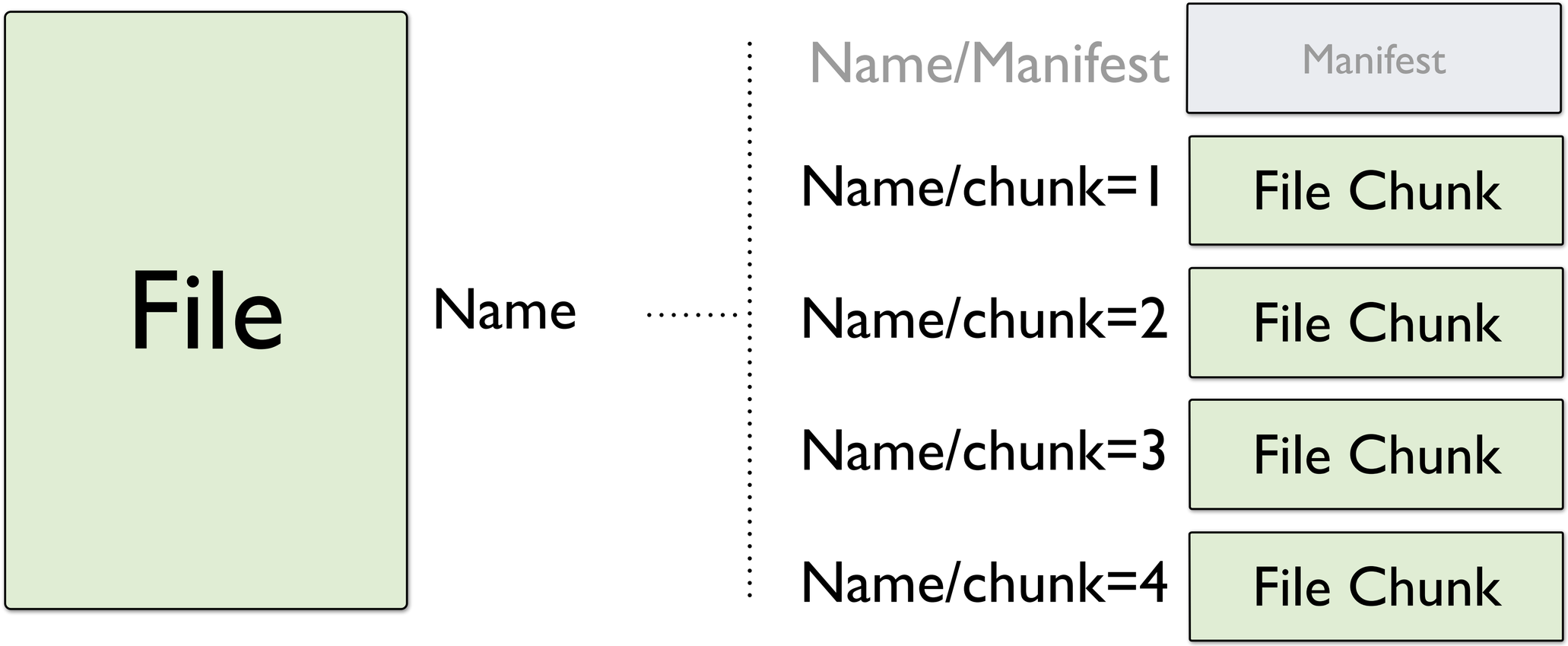


Name

A large object like a file has a CCN name



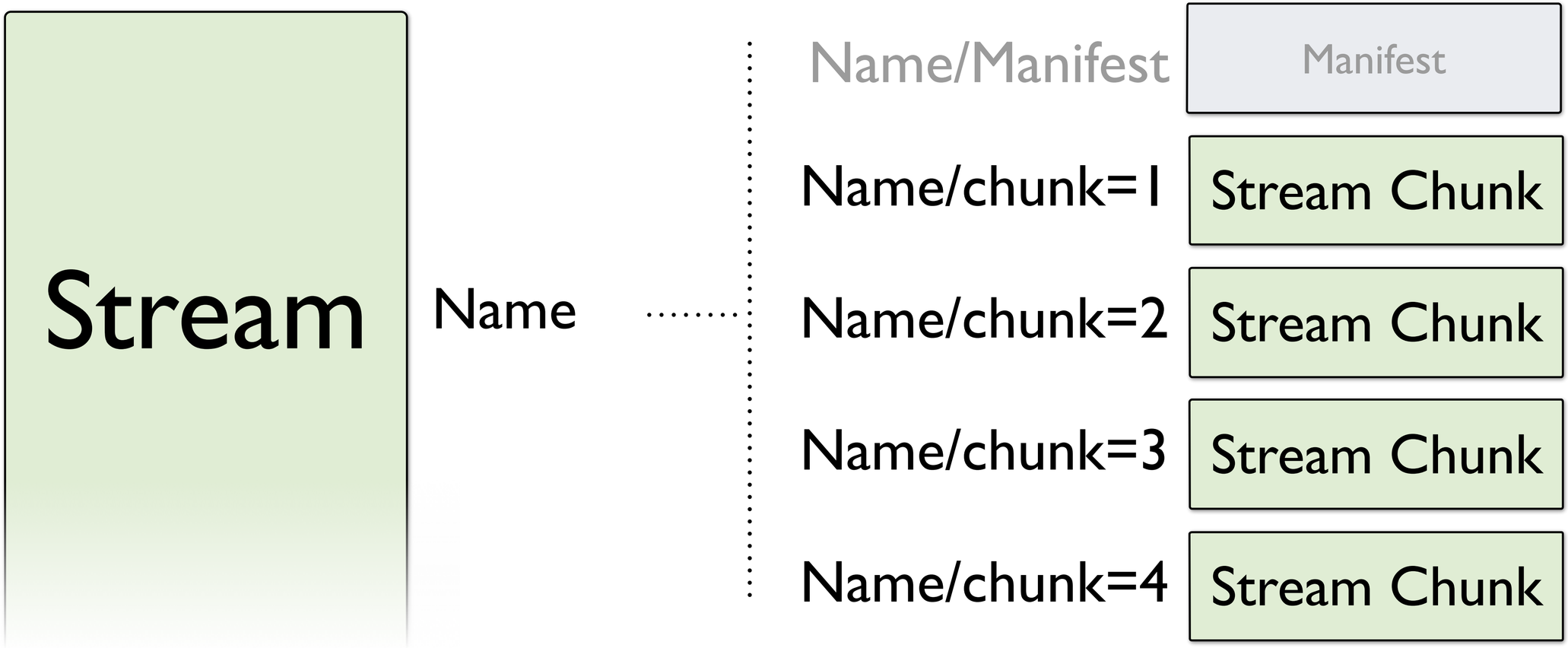
CCN also names the network sized chunks



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

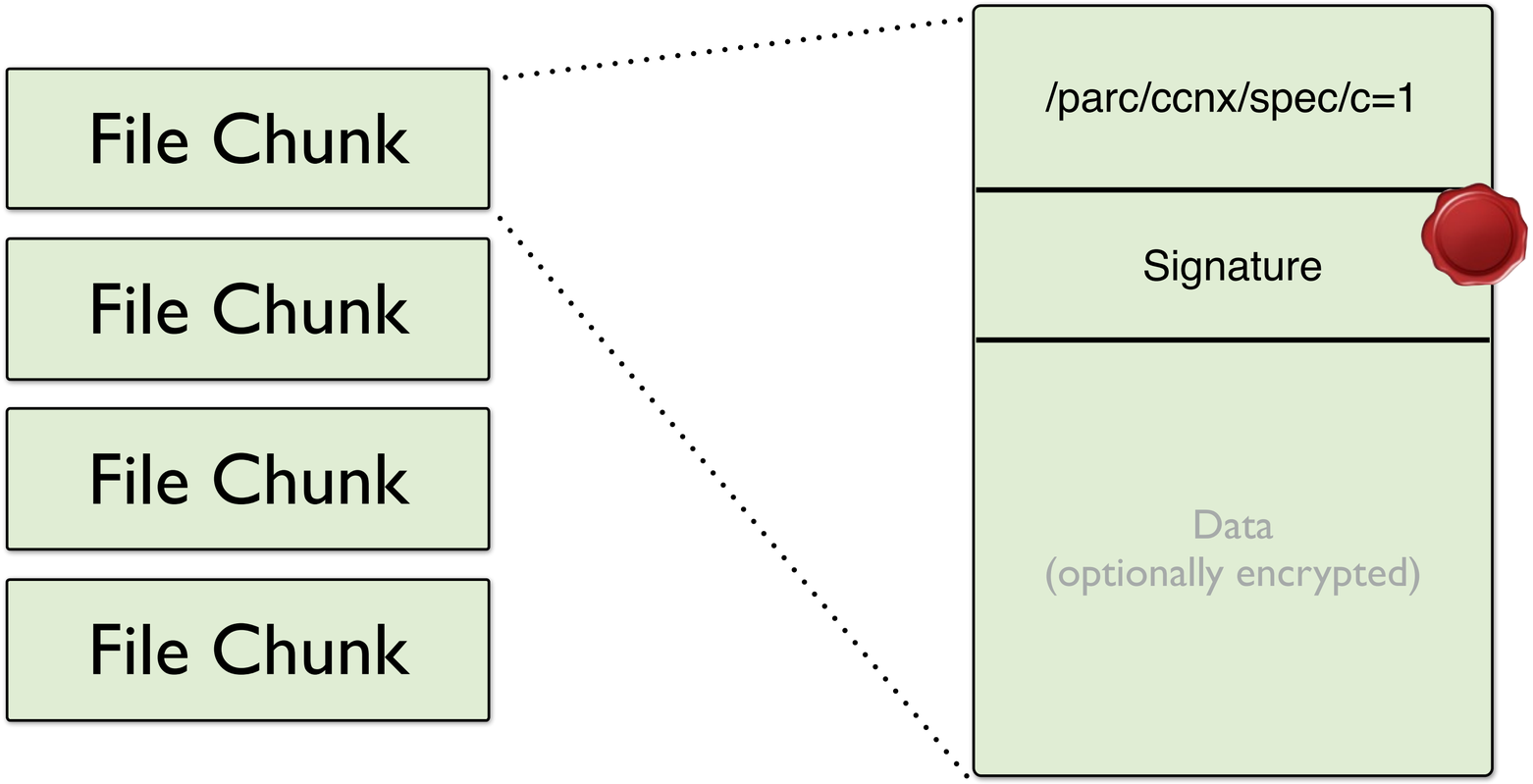
Applications can name their data any way they want (like filenames)

Protocol name segments

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

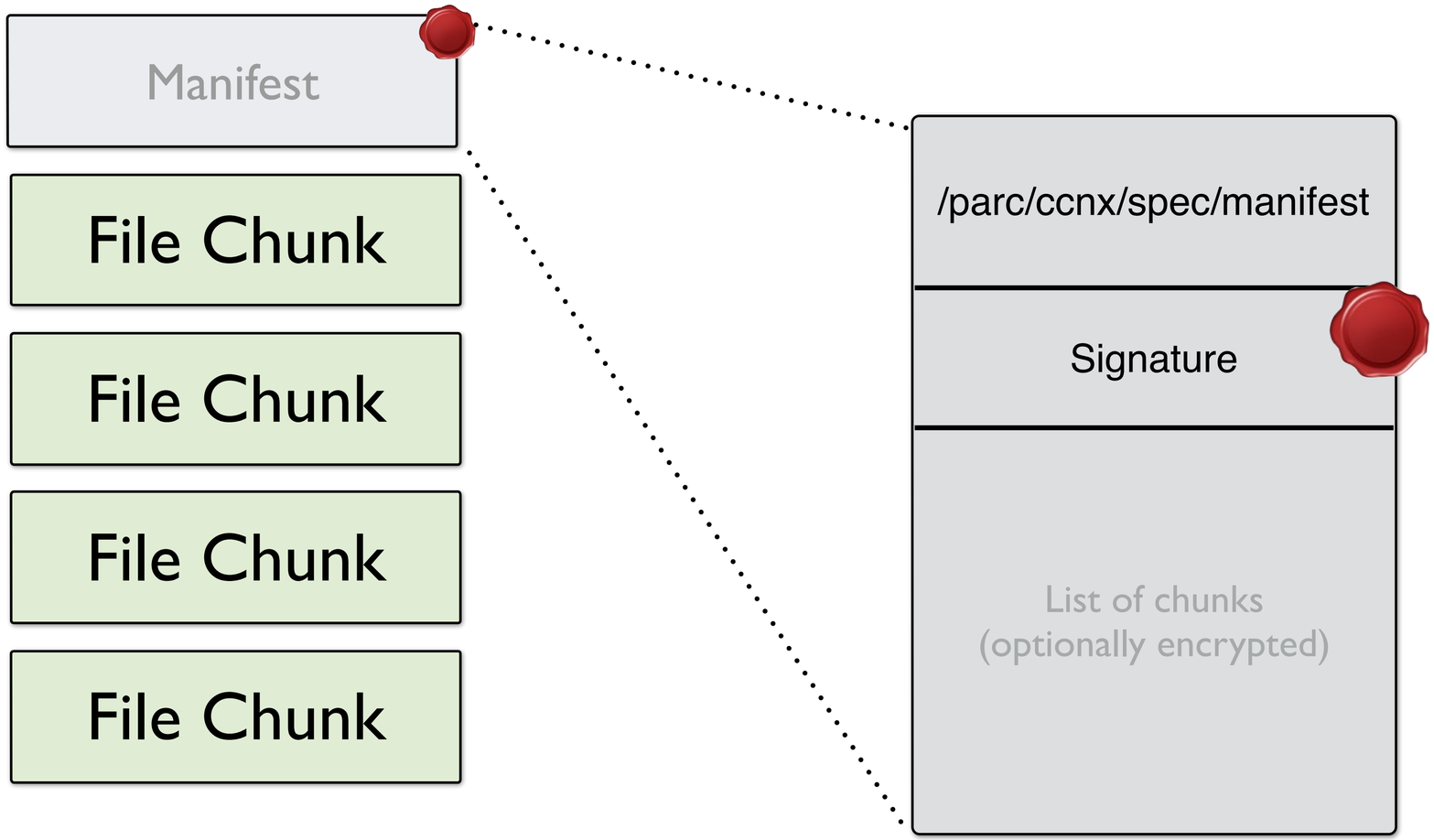
secure data

Secure single chunks



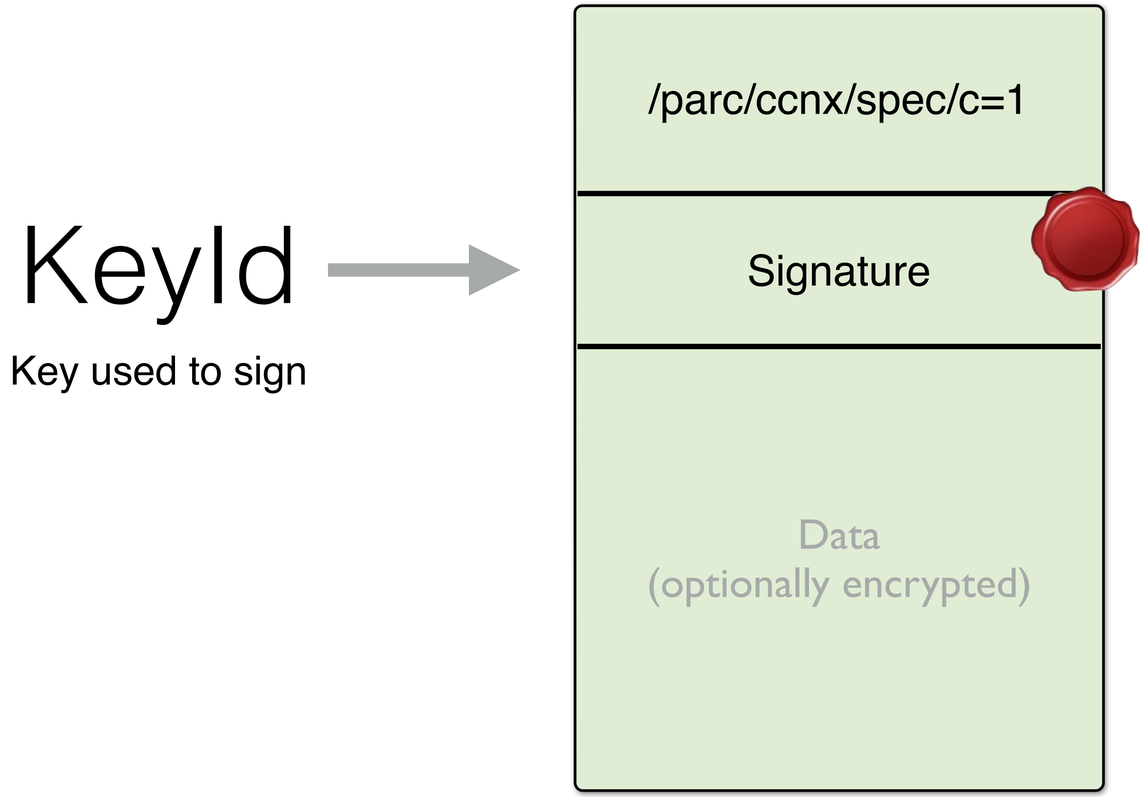
CCN names and signs every chunk

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

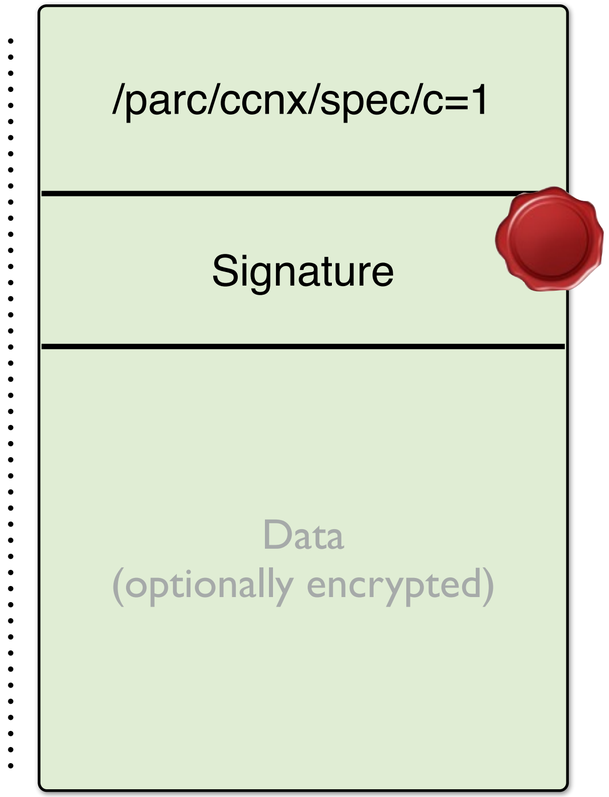
/parc/ccnx/presentations/slide20/v=2/c=0

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

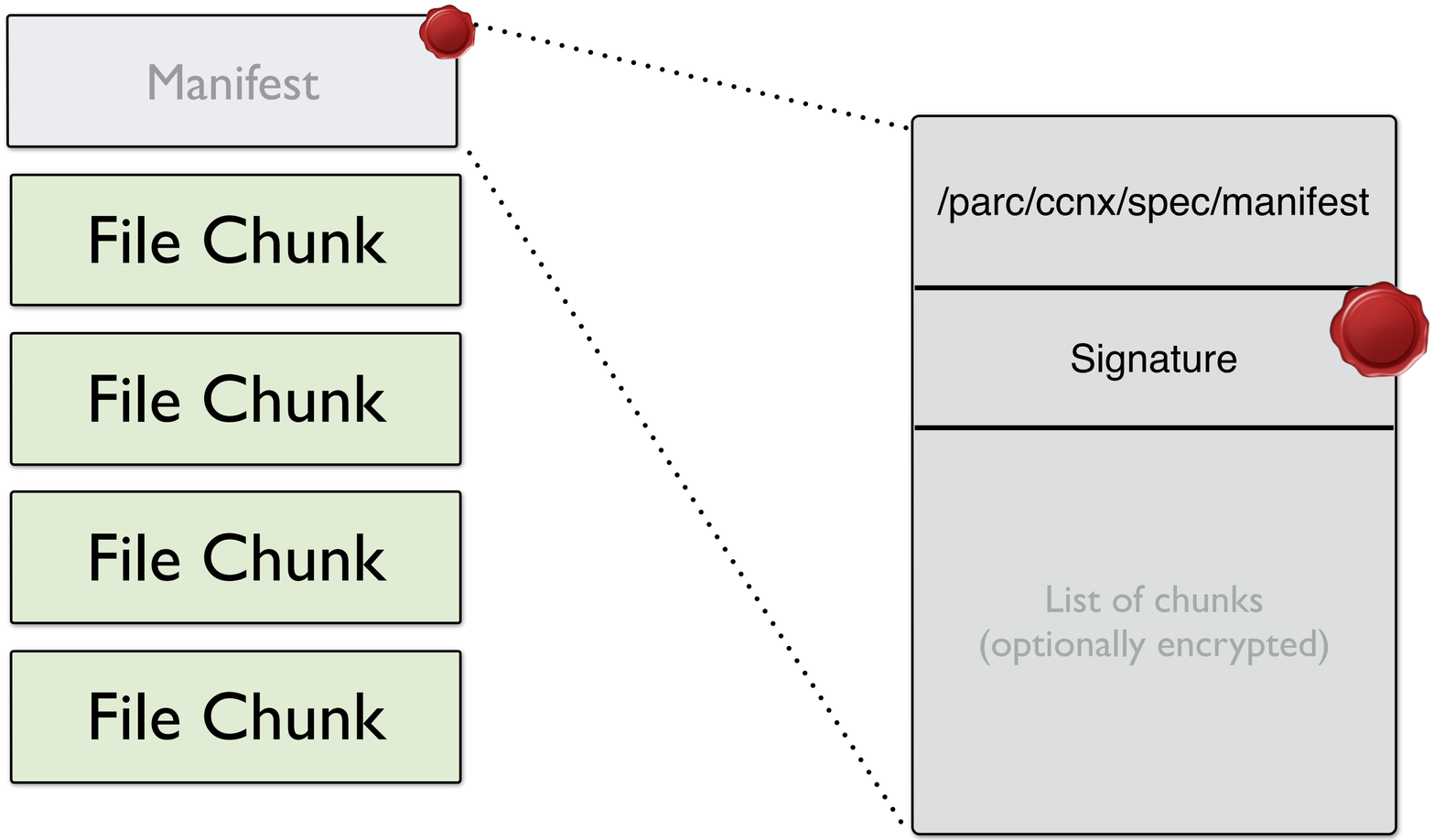
ContentObjectHash

hash of the content object message

KeyId

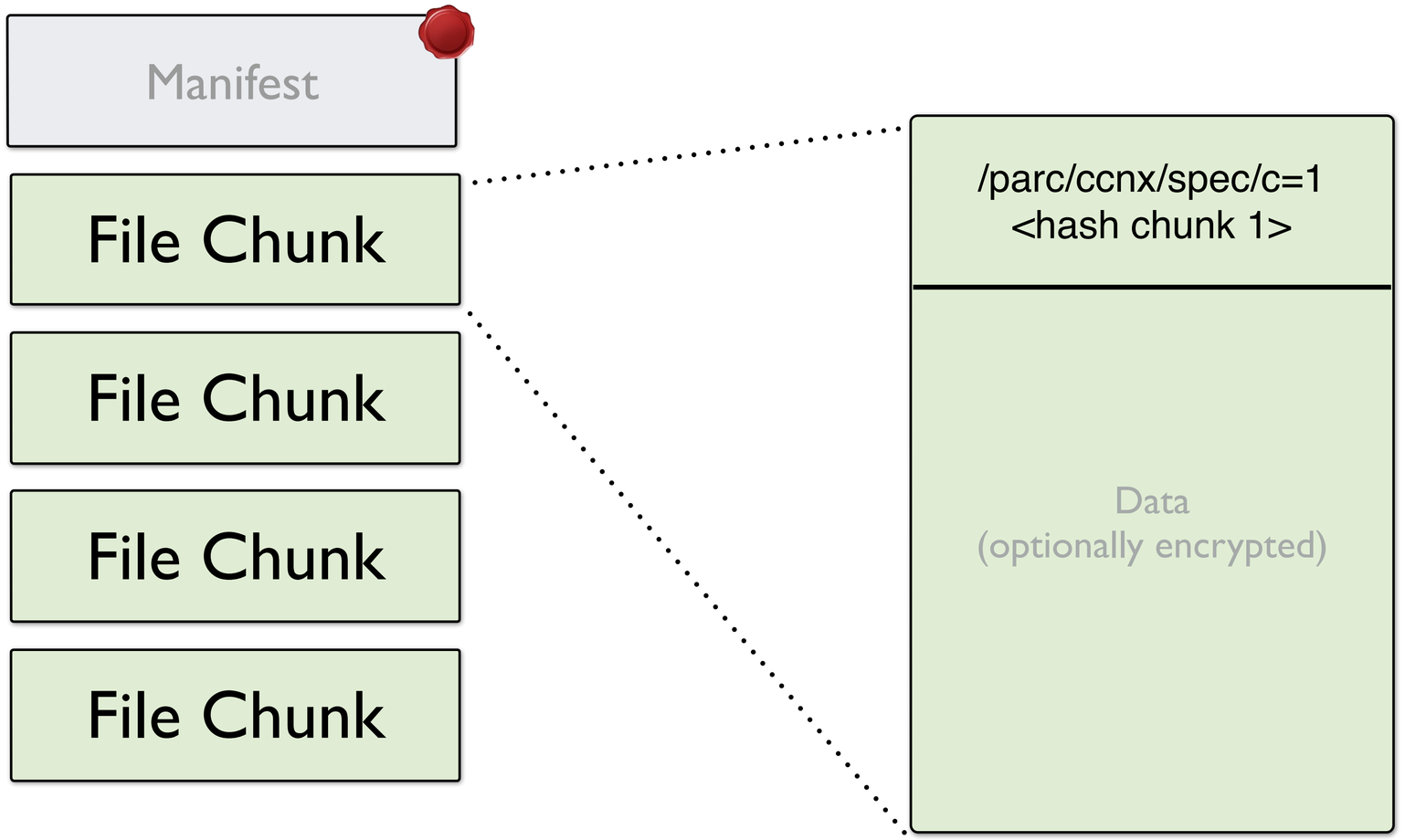
identifier of key used to sign the object

Secure whole object via manifest



CCN names and signs the file via a manifest

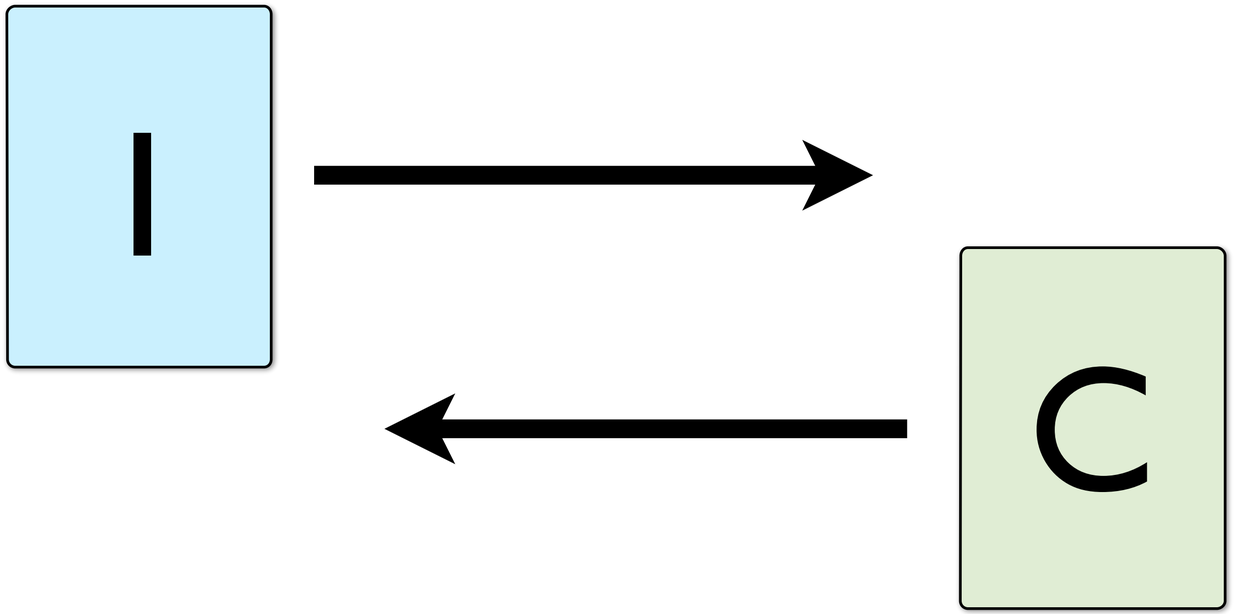
Secure via manifest



Indirectly sign every chunk through the manifest

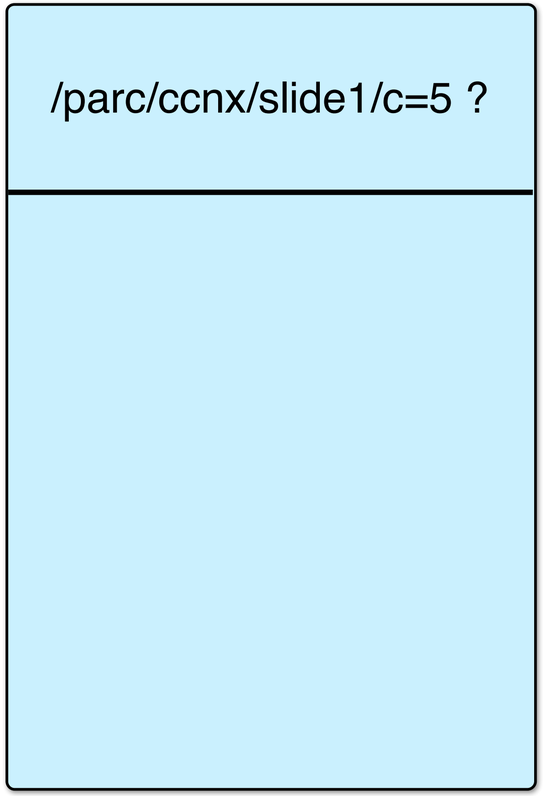
transfer data

Core Protocol

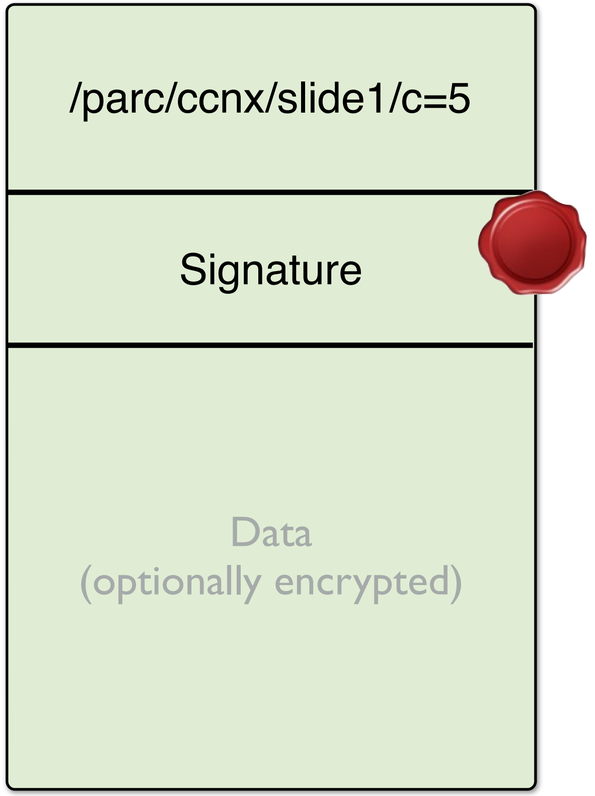


One interest packet gets one content packet

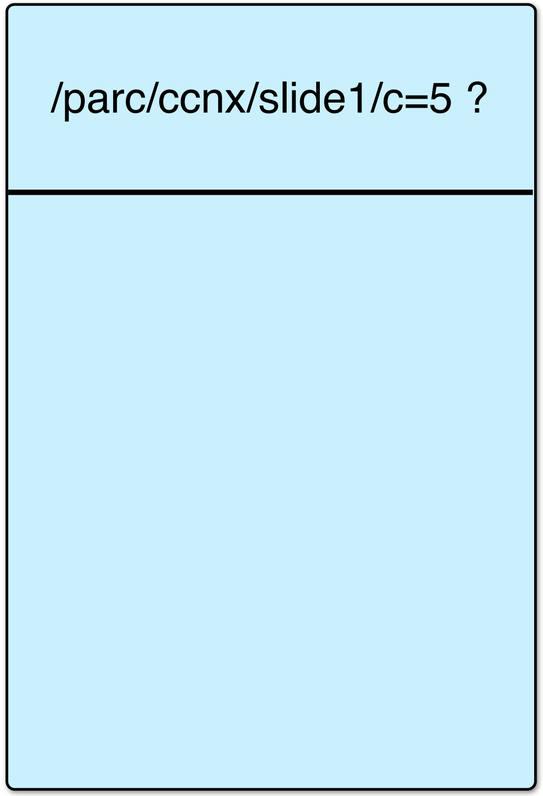
Interest



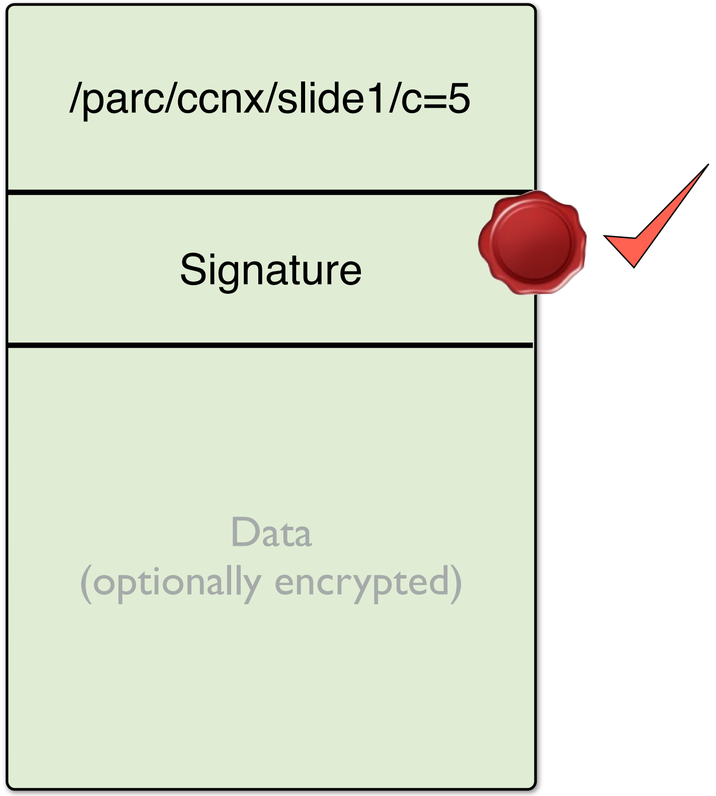
Content Object



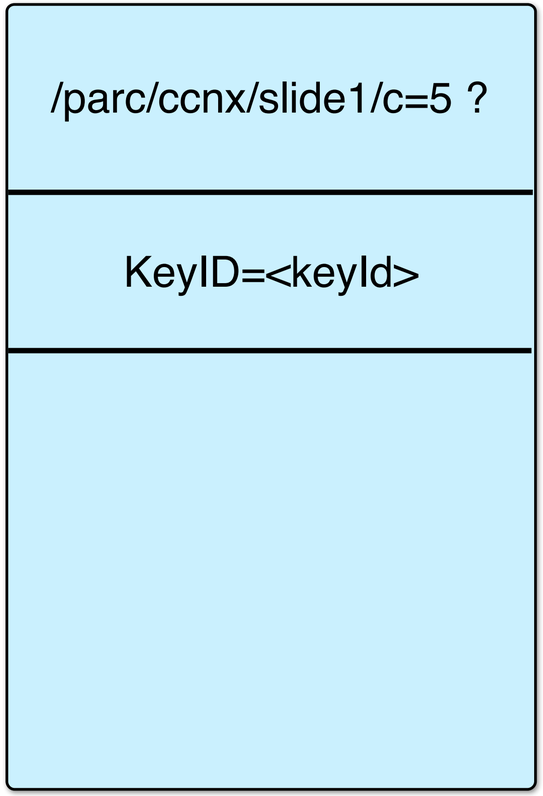
Interest



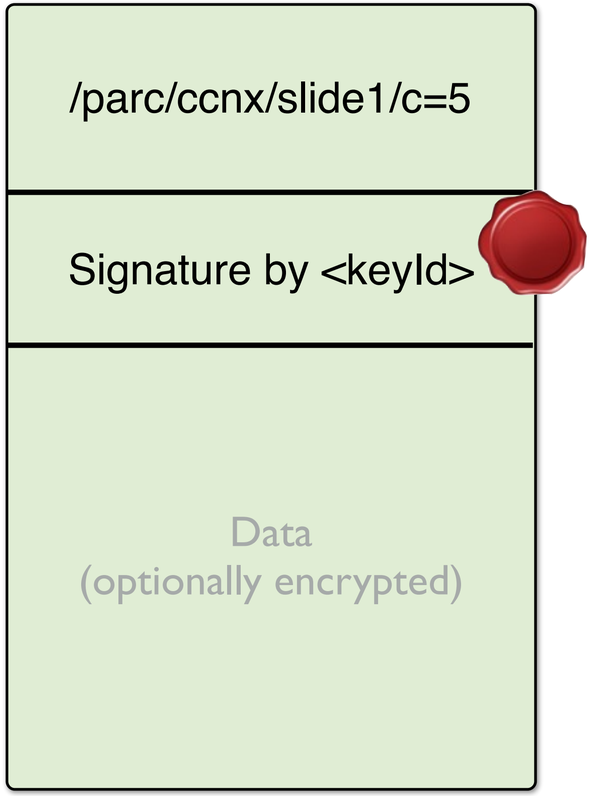
Content Object



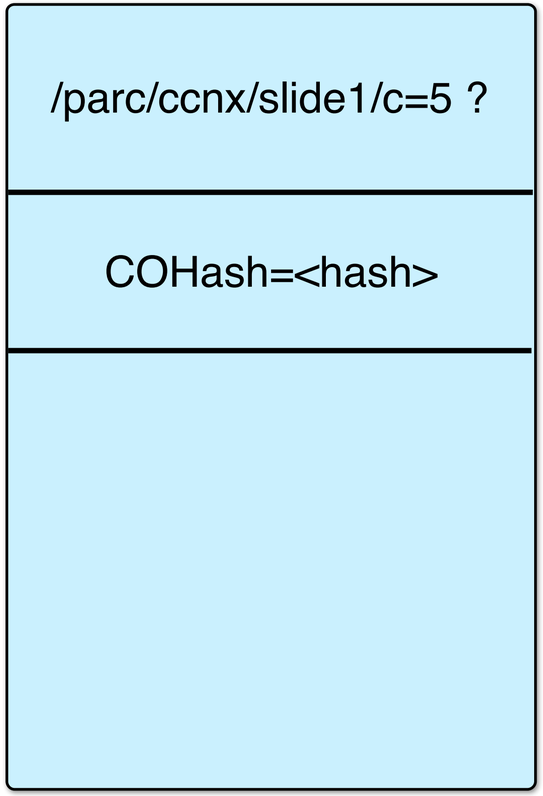
Interest



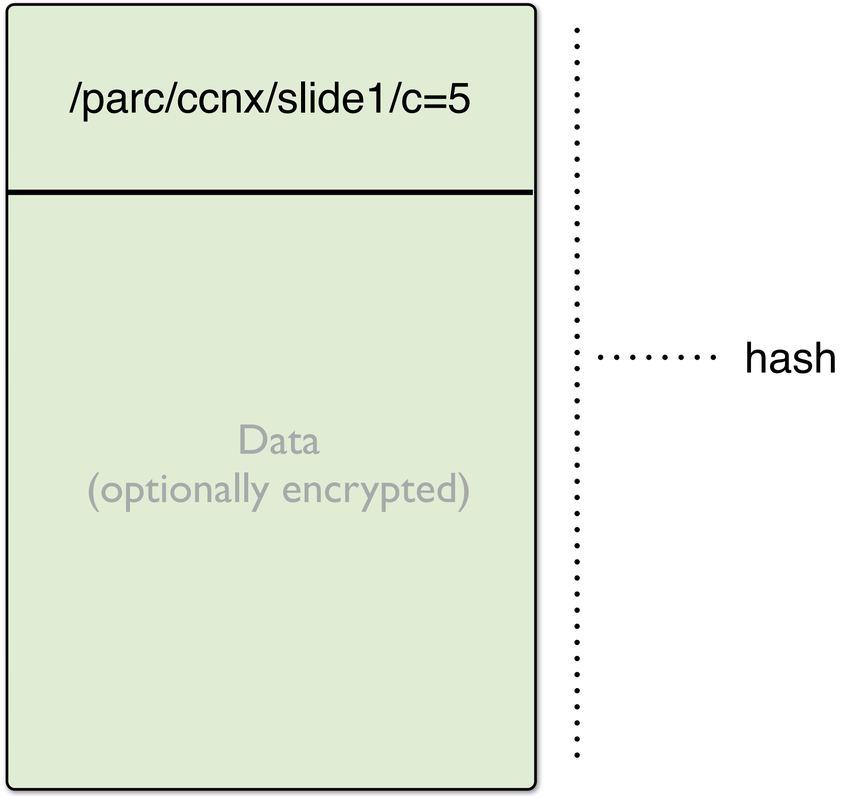
Content Object



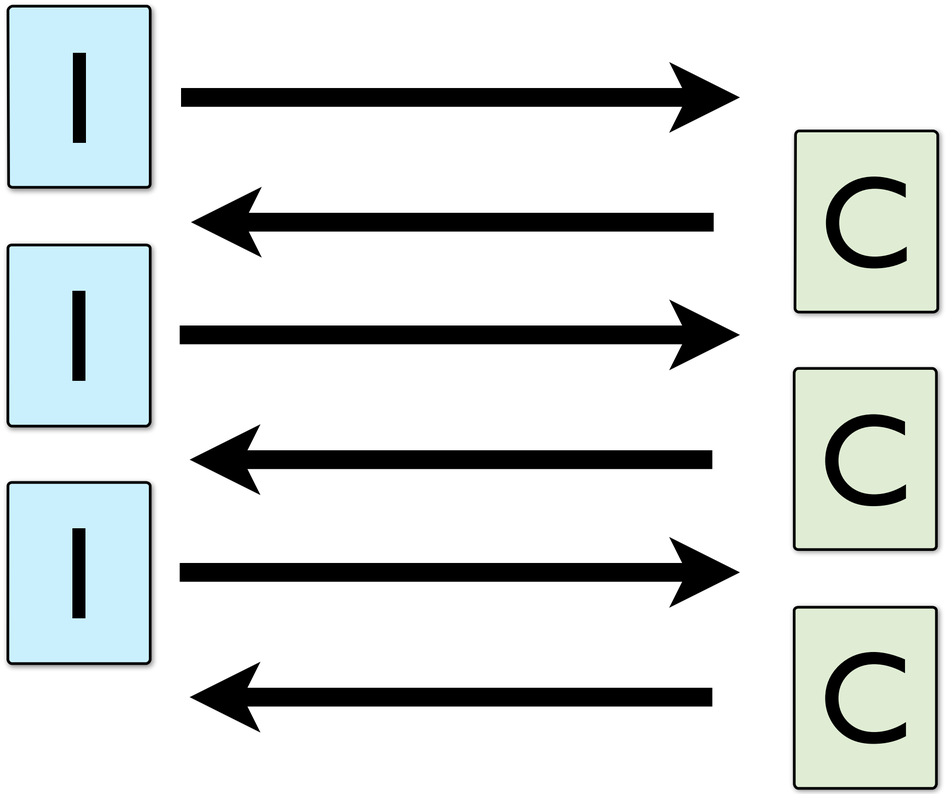
Interest



Content Object

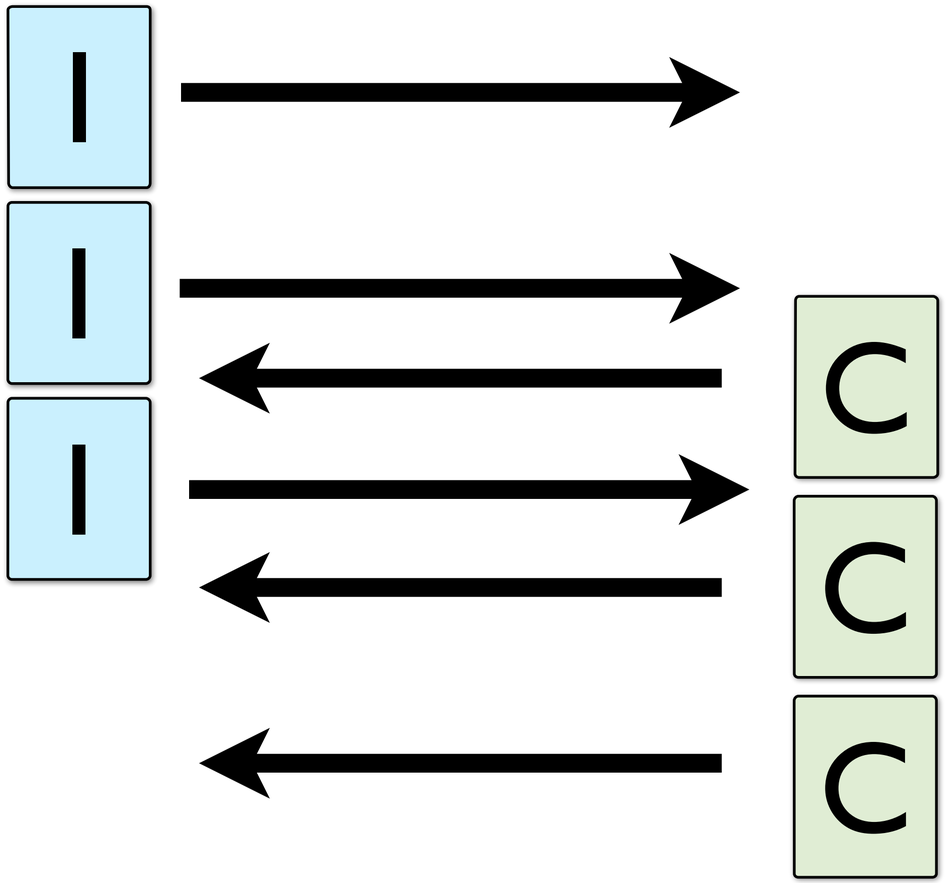


Transport Protocols



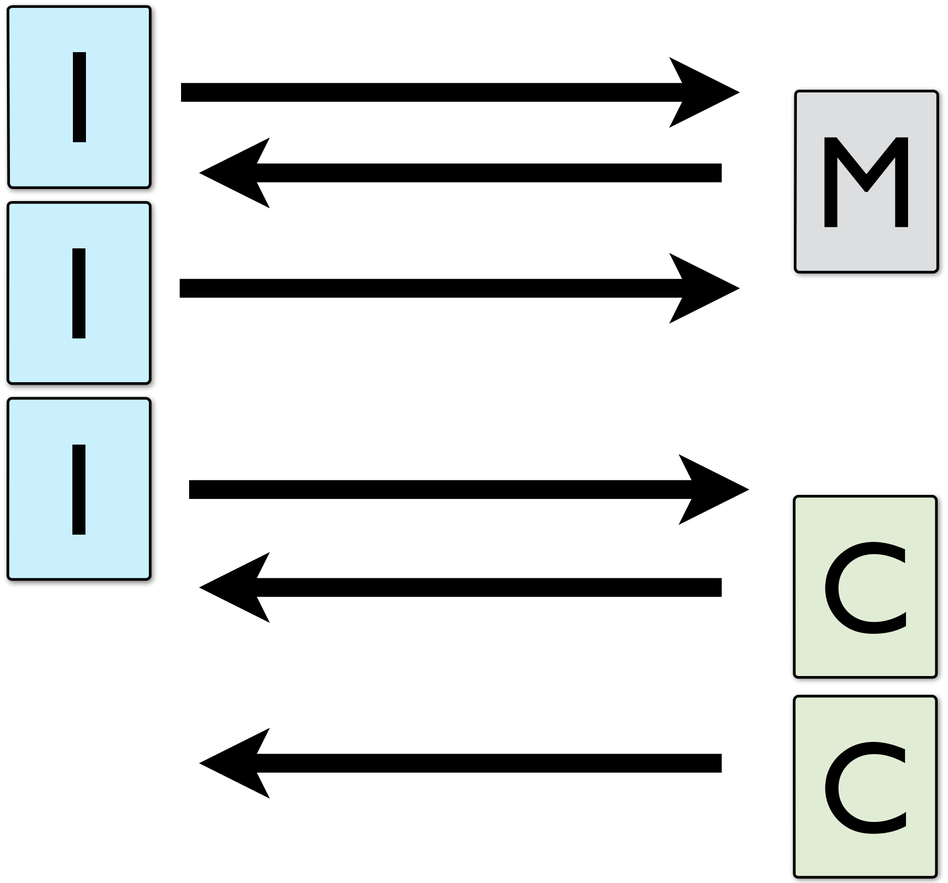
Transport protocols are built on core exchanges

Transport Protocols



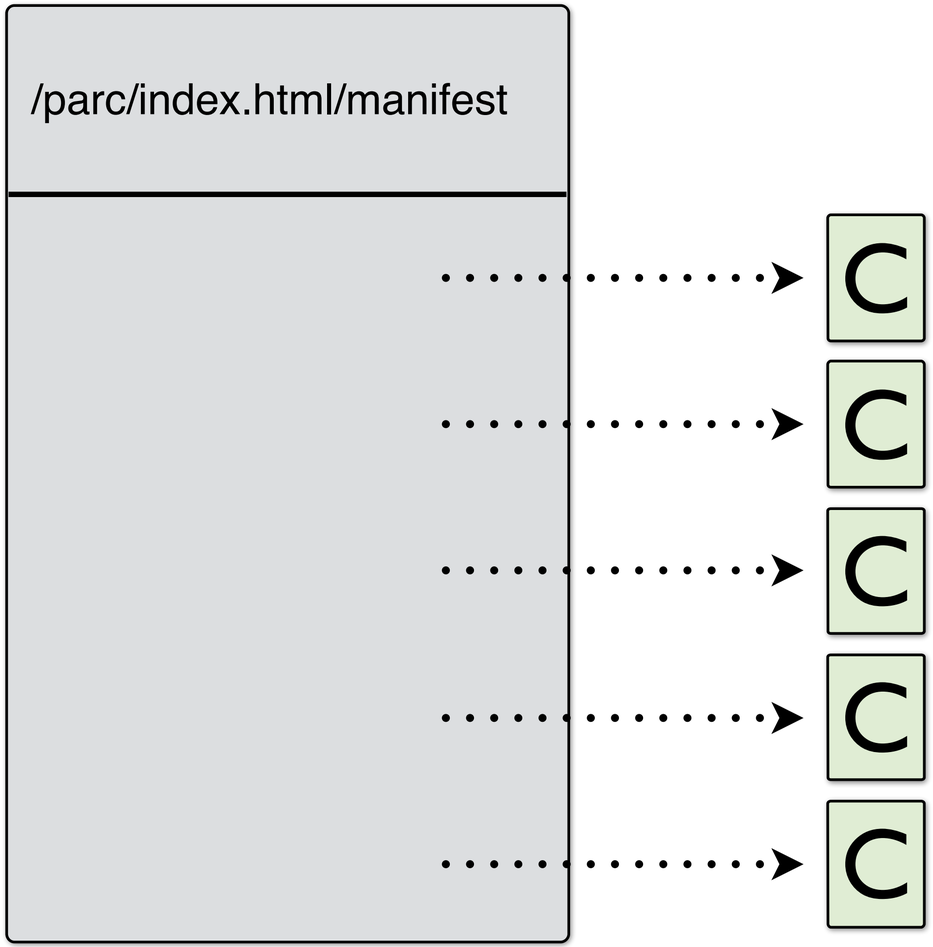
Transport protocols can do parallel requests

Transport Protocols

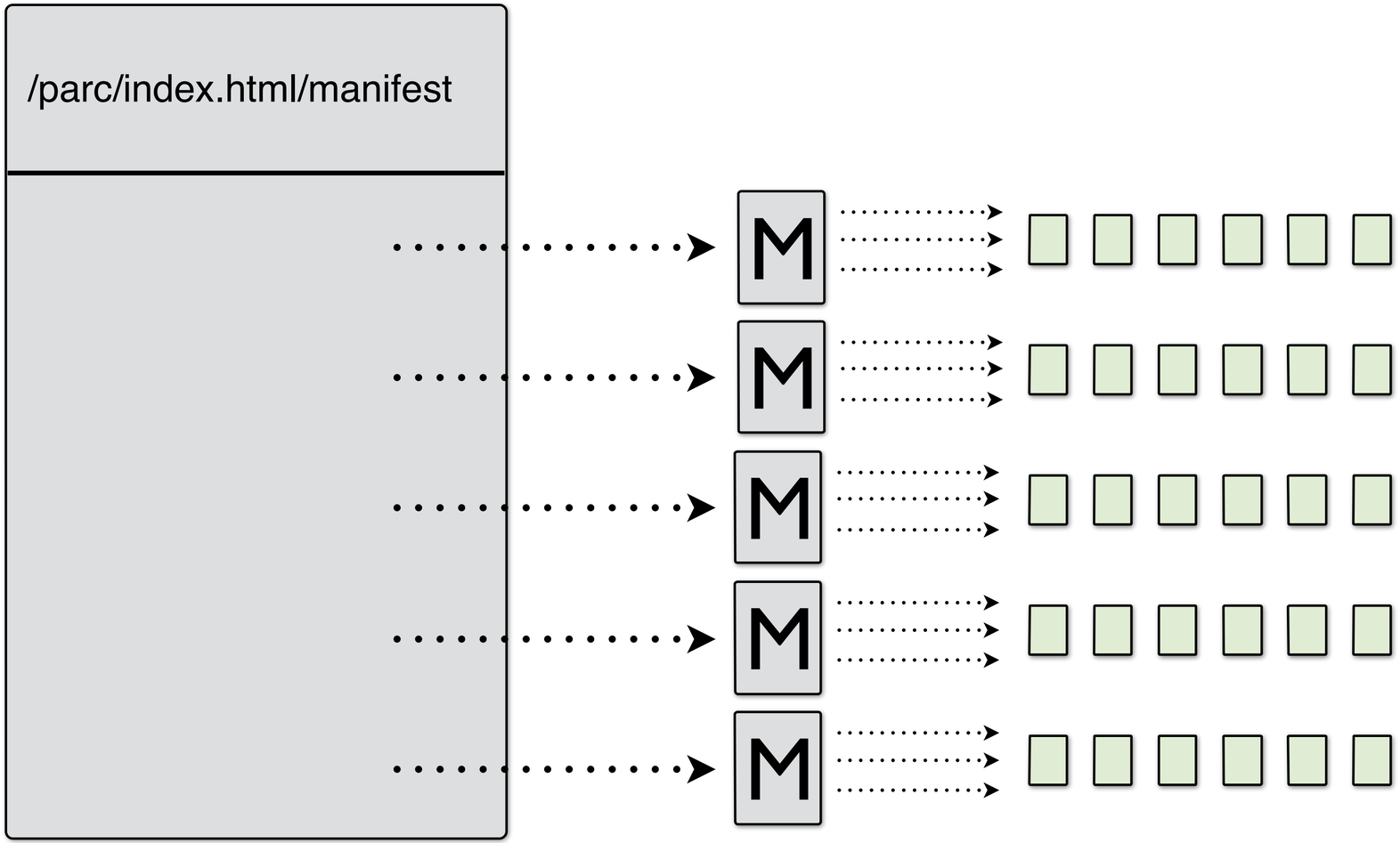


Transport protocols can use manifests

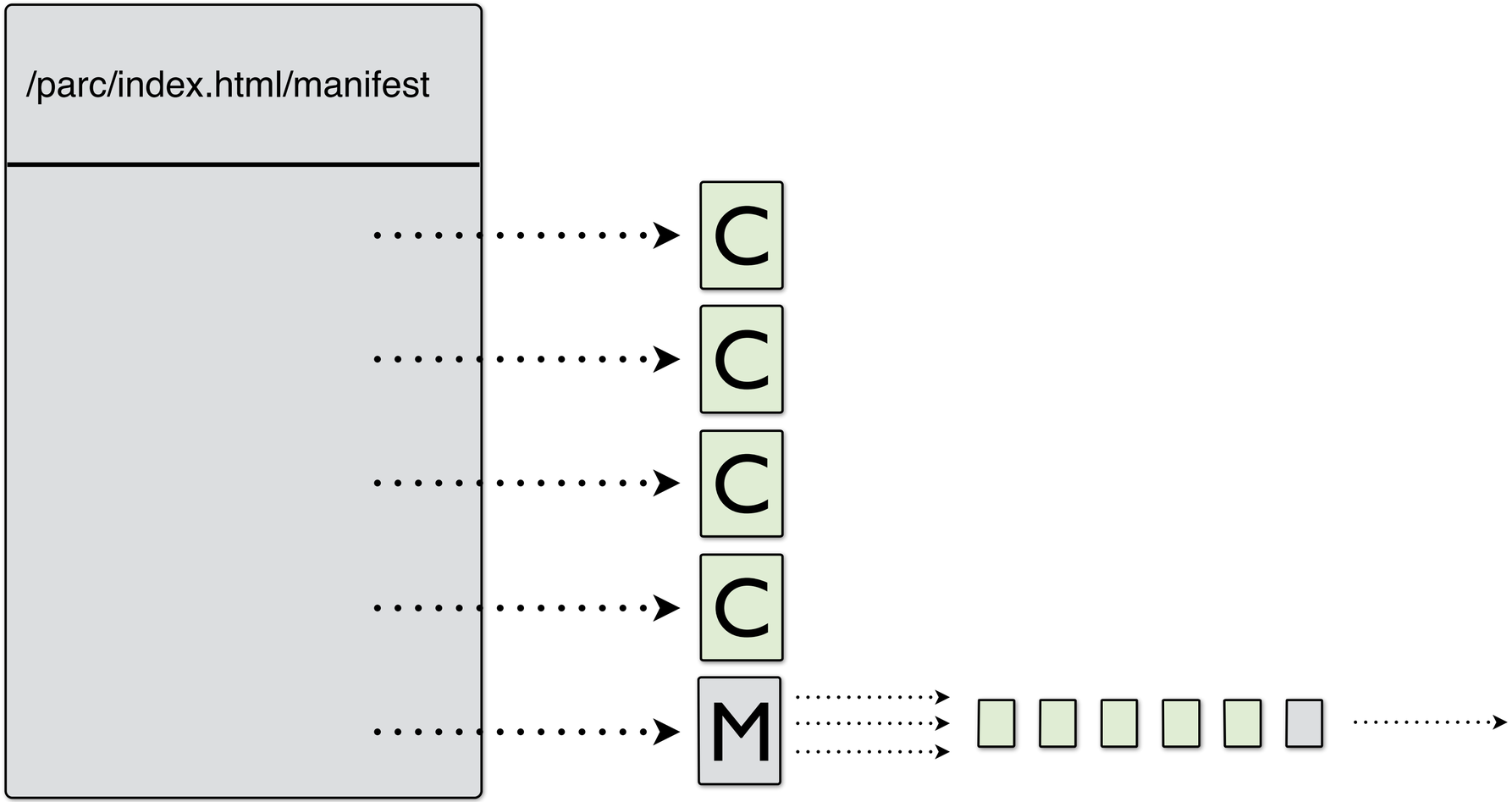
Manifest structure



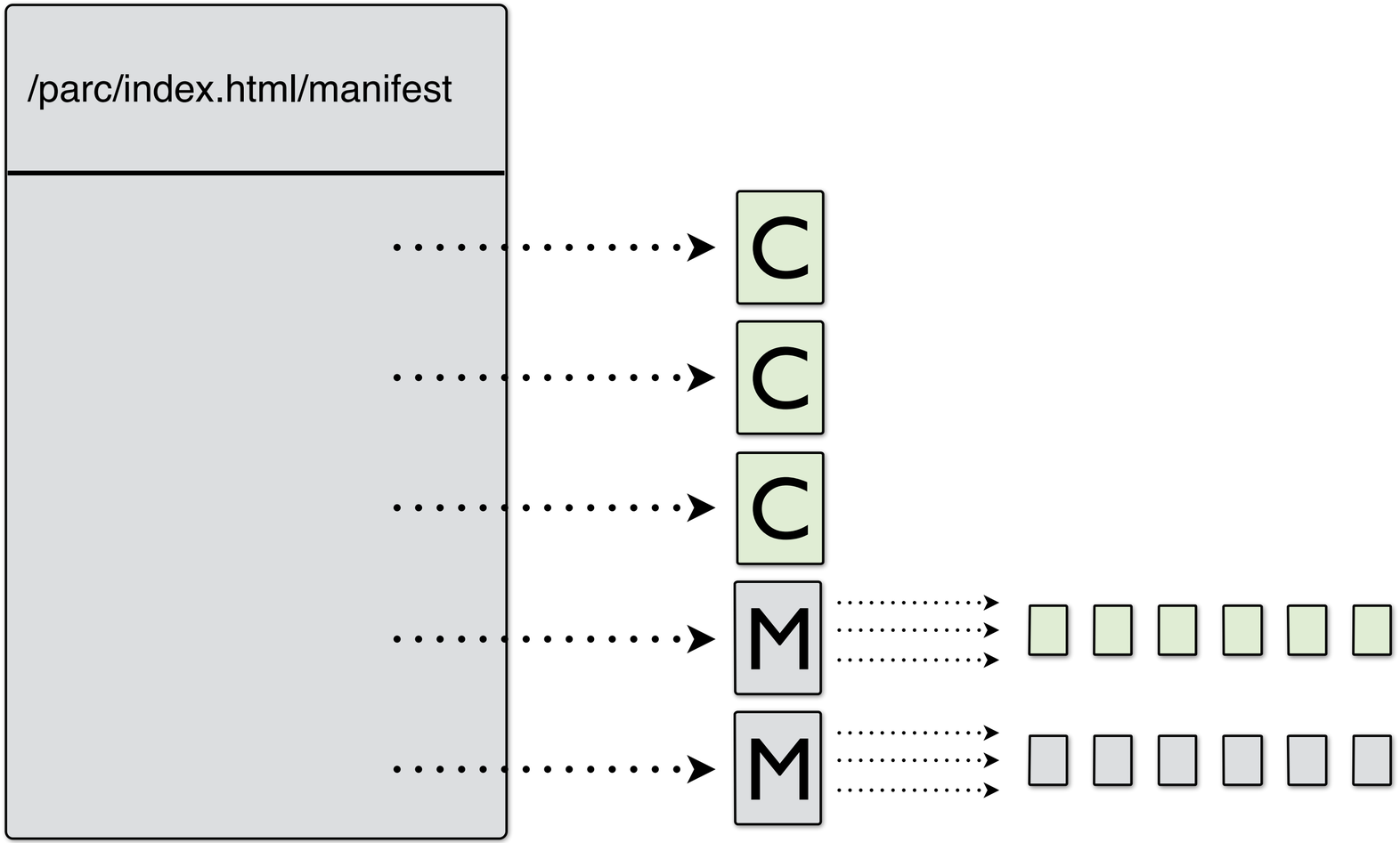
Manifest structure



Manifest structure

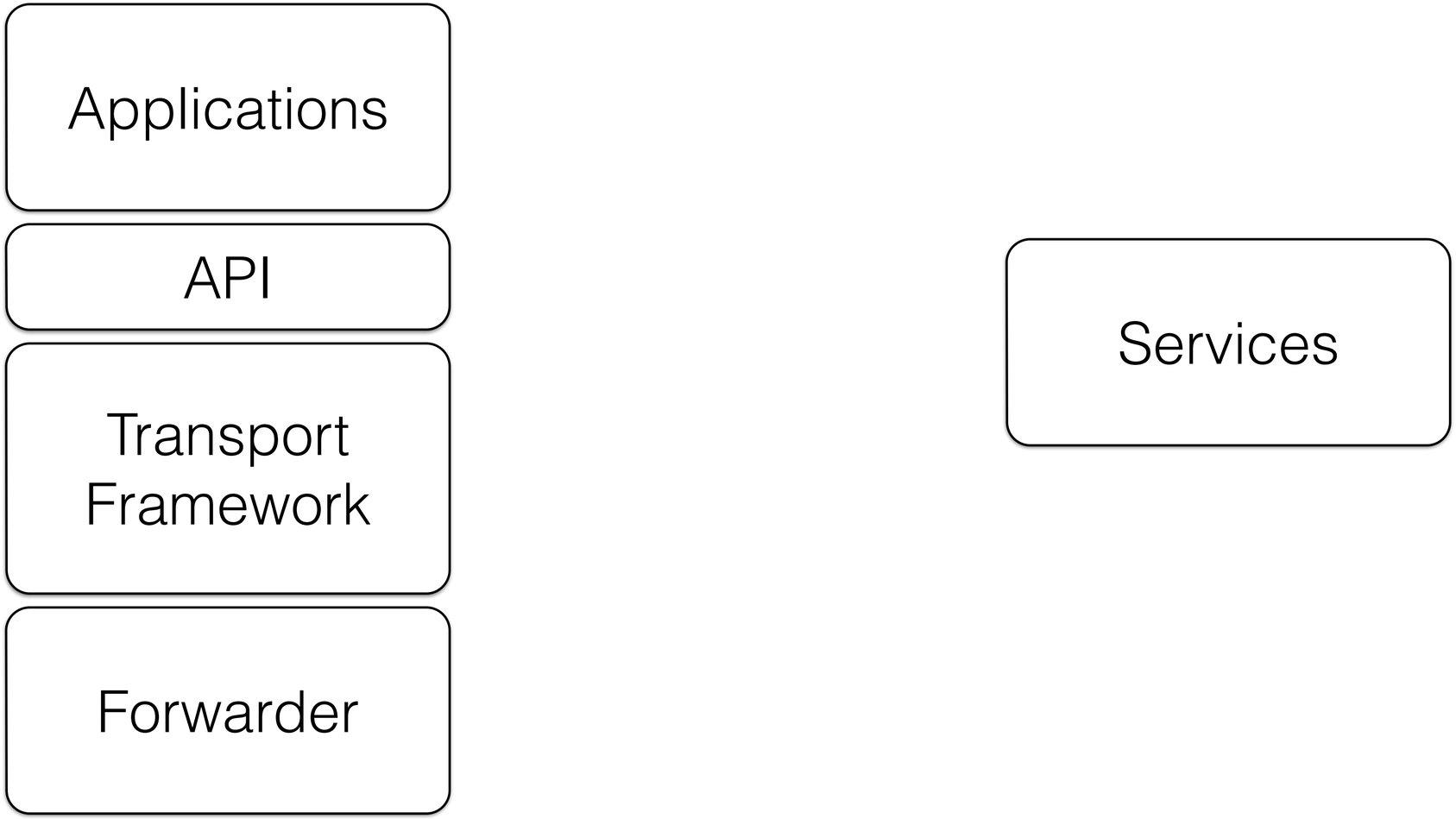


Manifest structure

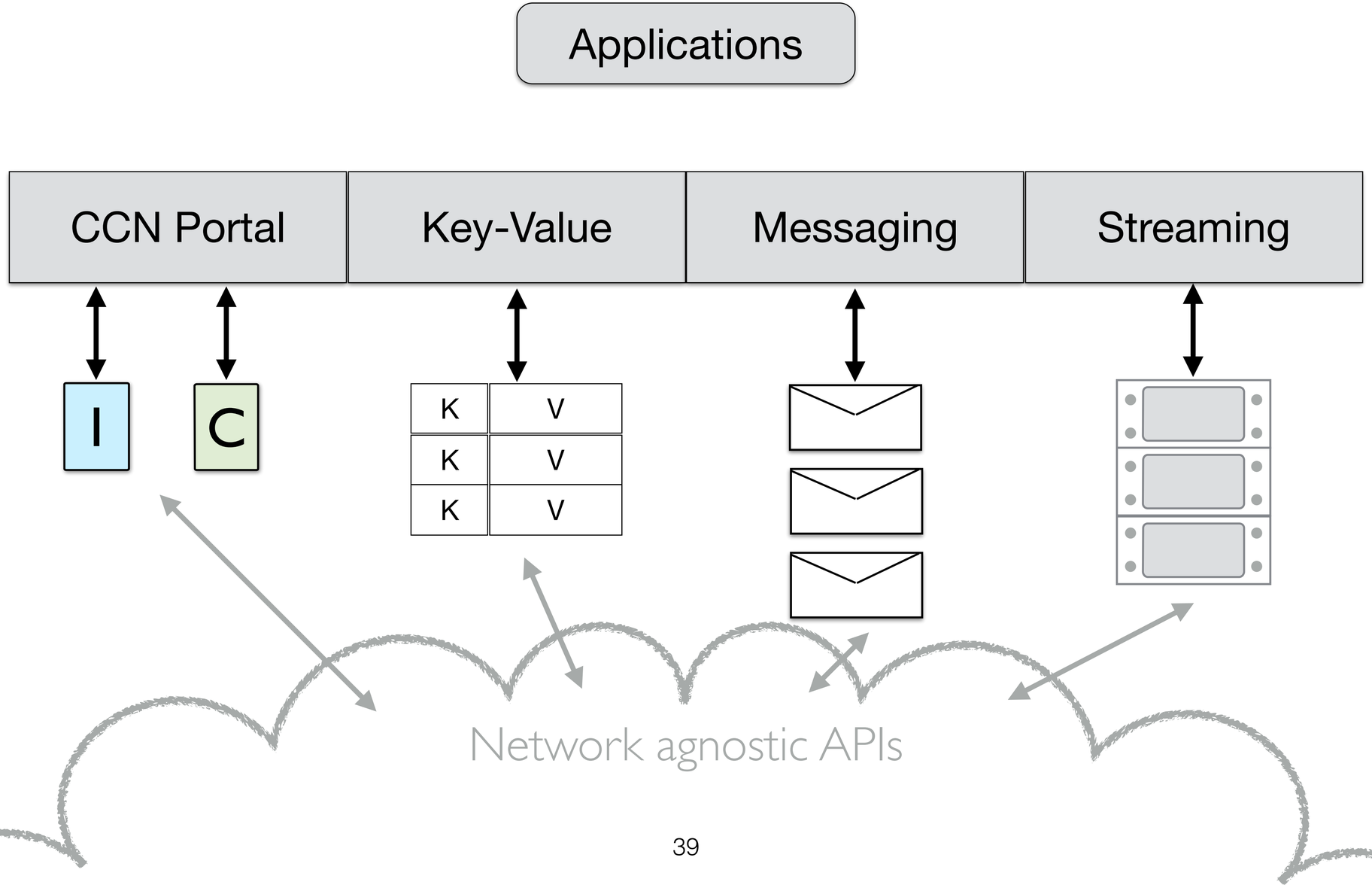


CCN

The CCN software architecture



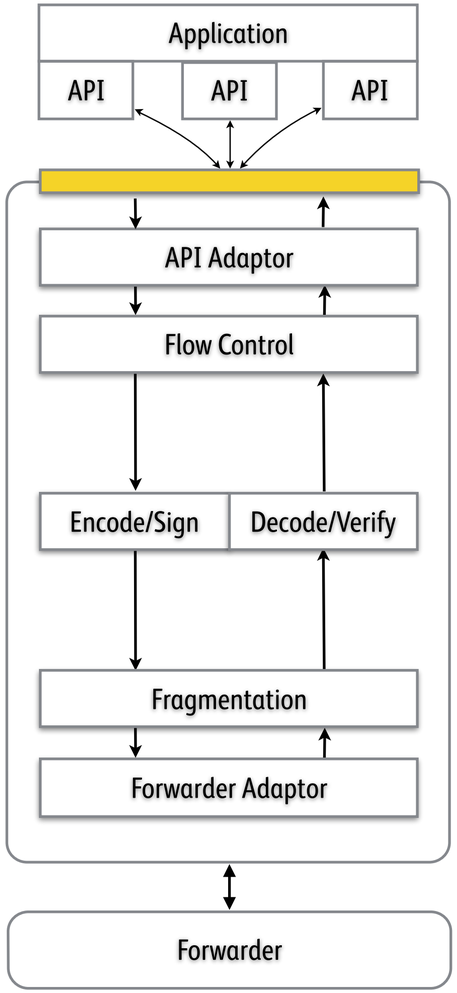
CCN Services and API characteristics



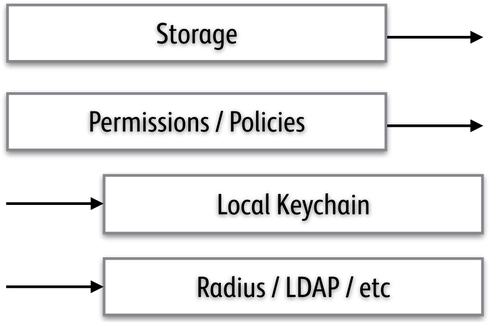
parc
A Xerox Company

The CCN transport stack

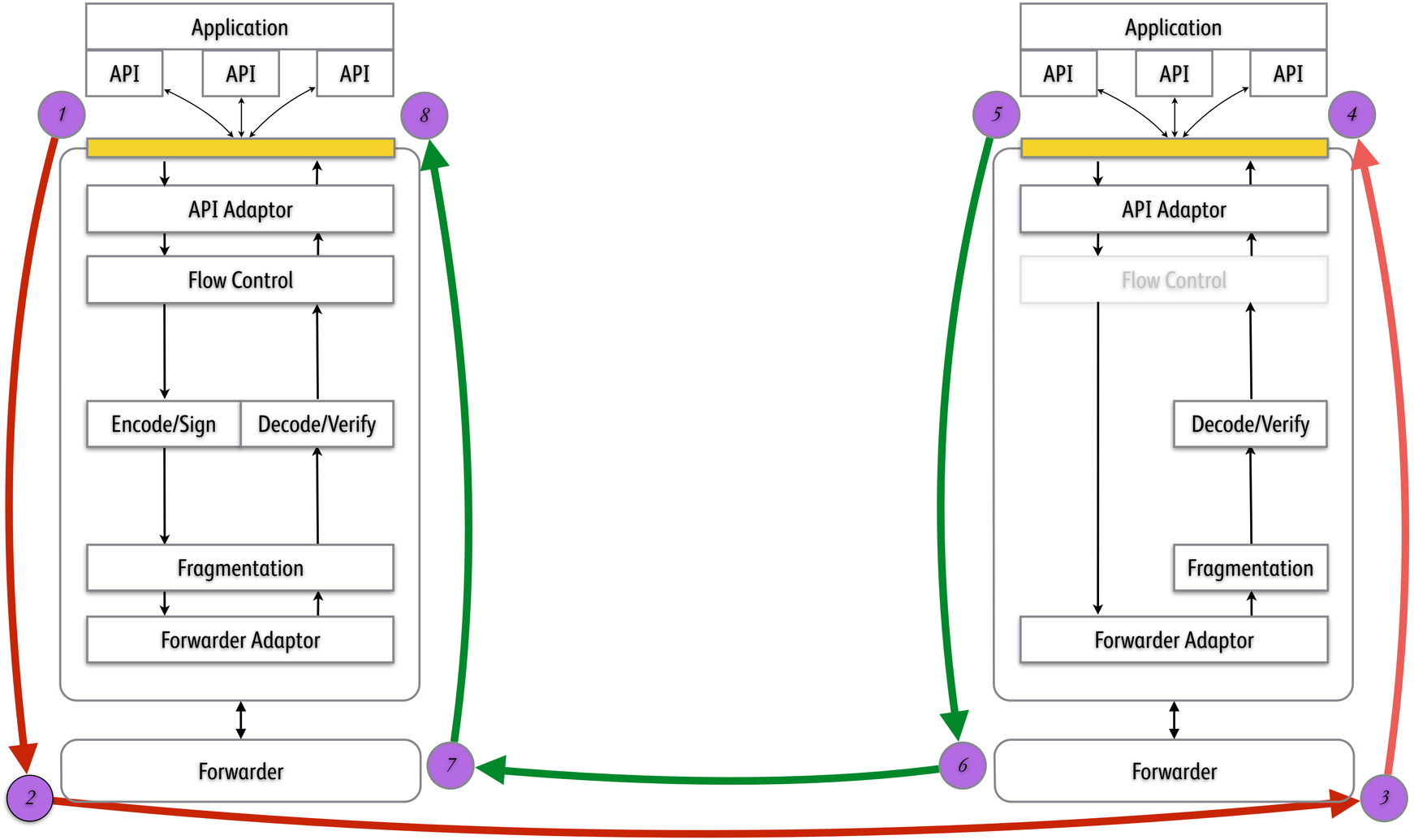
Base Stack



Optional Components



Transport Stack



CCN - 1.x

parc
A Xerox Company

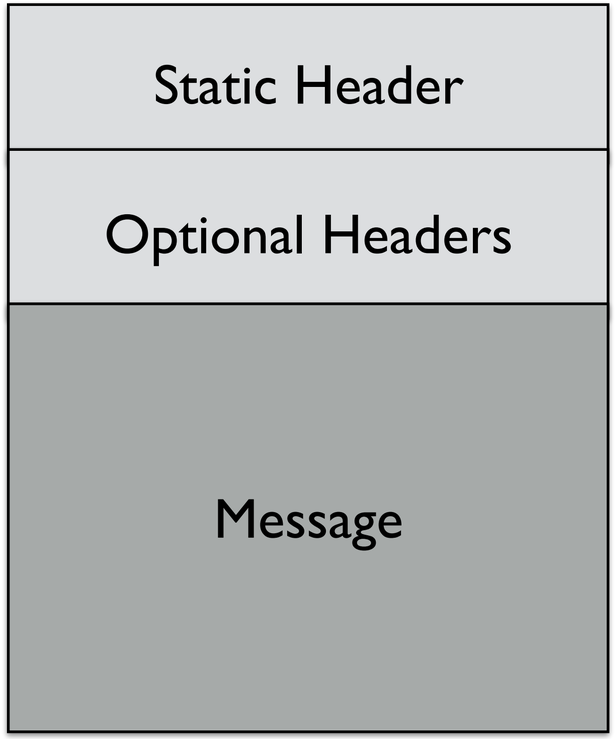
Packet Format

Static Header
Information required on every packet.
May be modified at intermediate hops.

Optional Header
Optional information. May be modified at intermediate hops.

Message
End-to-end protocol message.
Unmodified by intermediate elements.

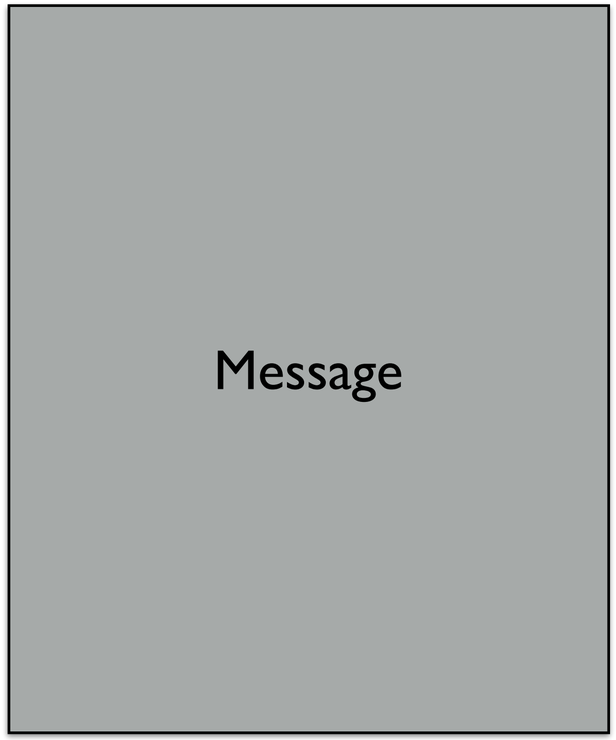
CCN 1.x



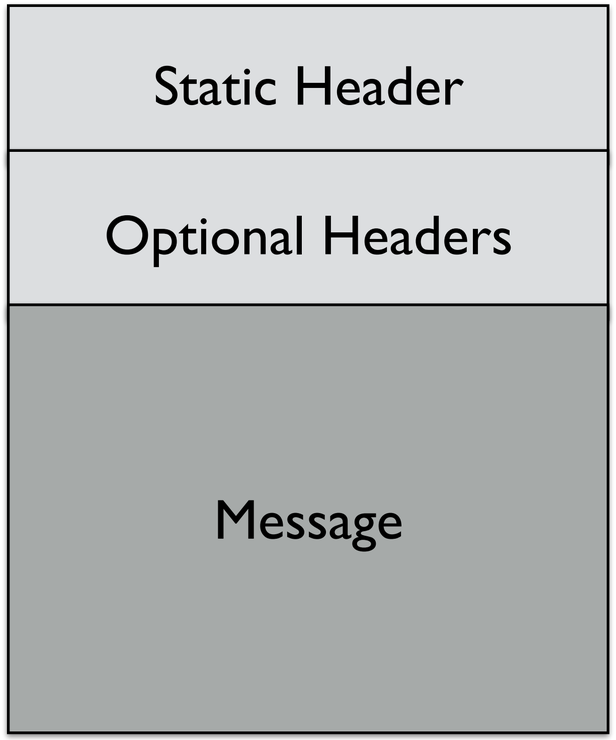
parc
A Xerox Company

Packet Format

CCN 0.x



CCN 1.x



Packet Format - Why change

Static header

enables fast parsing
contains common needs
allows versioning

Optional headers

allows network elements to add/modify information

Message Organization

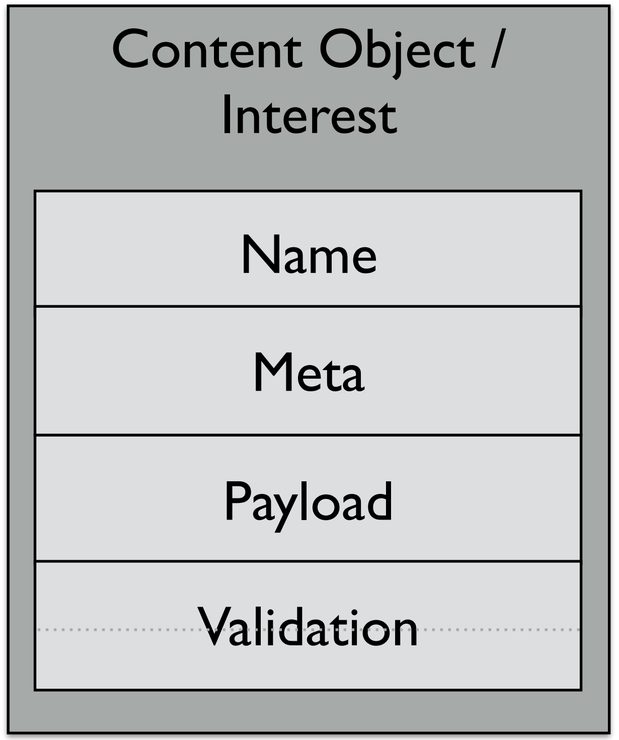
Single message format

Name in front

Payload

General Validation info in back

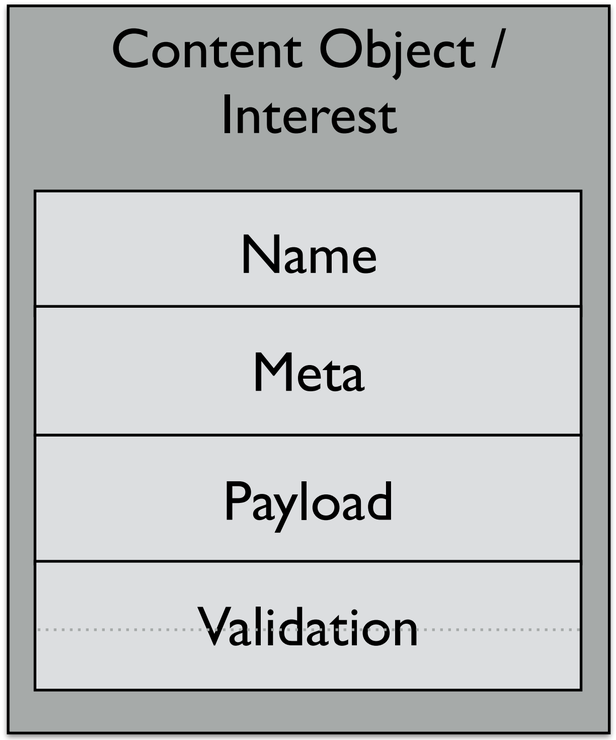
CCN 1.x



Message Organization

CCN 0.x

CCN 1.x



Message Organization - Why change

Name comes first
fast parsing

Separate validation from metadata at the end
modular security

Unified packet format
simplified, fast parsing

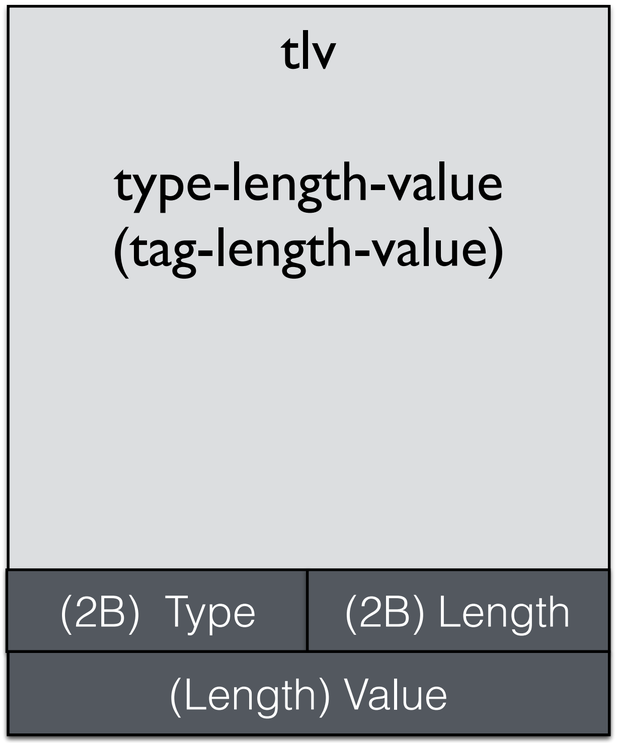
Packet Encoding

Easy to parse

Easy to skip

Unambiguous

CCN 1.x



Packet Encoding

CCN 0.x

ccnb

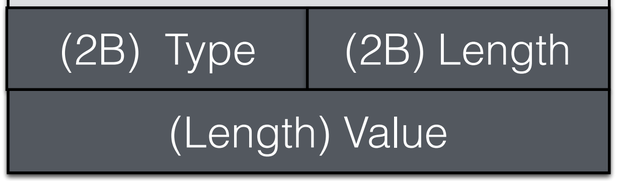
"Custom binary encoding format for XML to meet specific needs of CCNx"

<block><block><block>

CCN 1.x

tlv

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



Packet Encoding - Why change

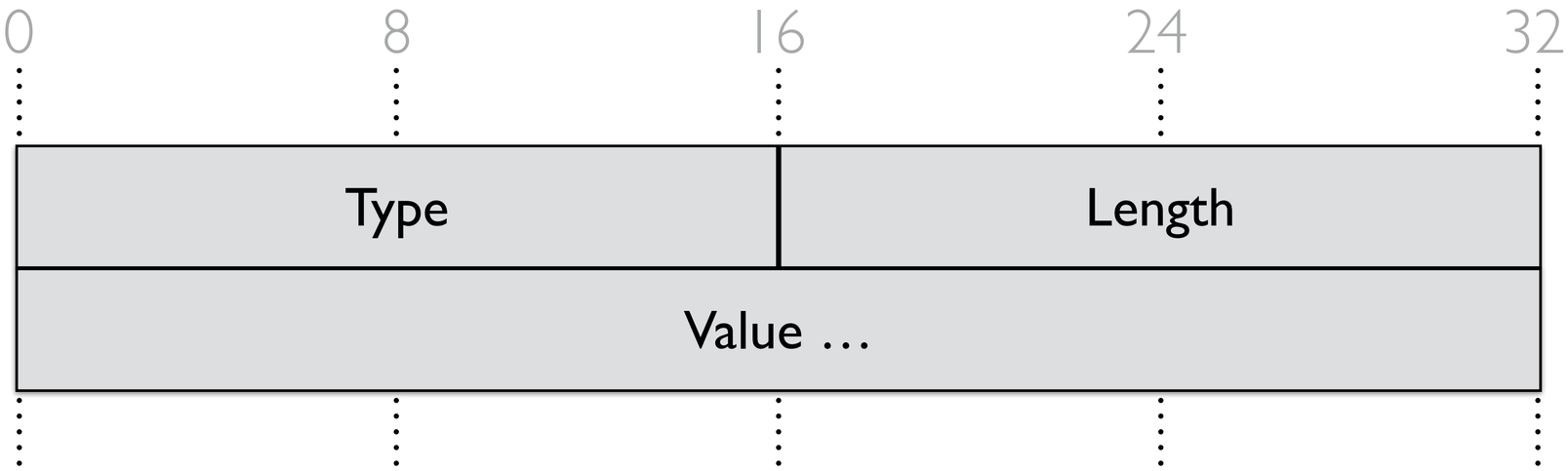
ccnb

- flexible but complicated
- relies on meta-structure
- bit efficient

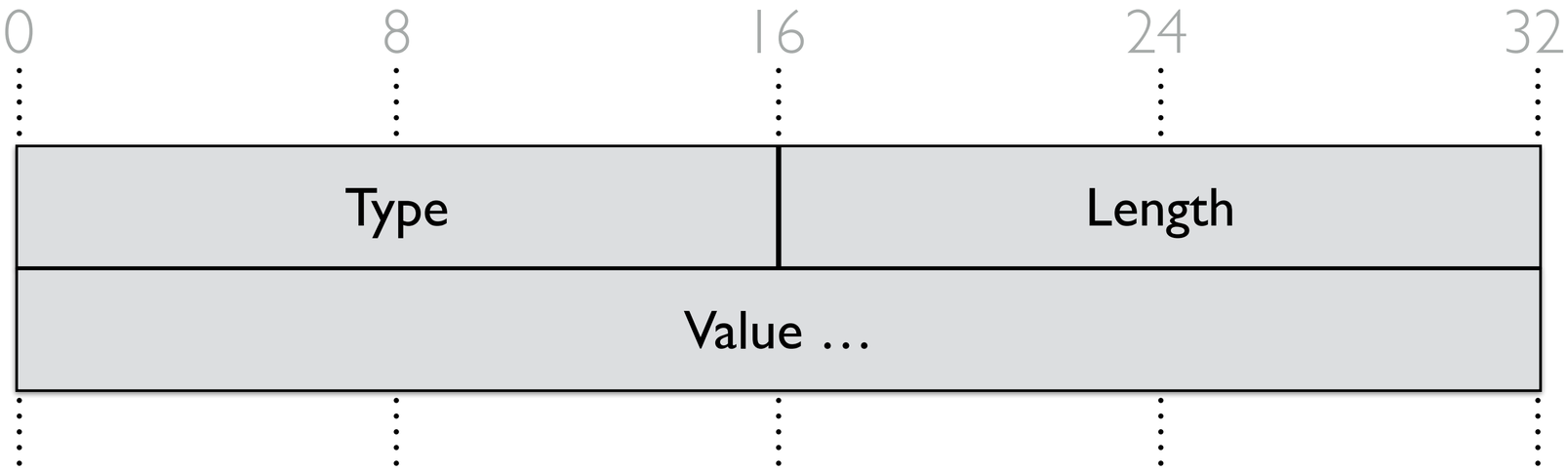
TLV

- easy to parse
- well understood
- parse efficient

2 x 2 encoding



2 x 2 encoding



2-Byte Type

Enough for expansion and experiments
Inherently hierarchical (not many types)
Fast to jump over (no parsing needed)
No aliasing (0 vs 00)
Canonical sorting

2-Byte Length

Don’t cover what doesn’t fit in a packet
Enough for large fields like payload
Fast to jump over (no parsing)
No aliasing (0 vs 00)
Canonical sorting

Label-based names

CCN 1.x

All segments have types

/parc/ccn.zip/app<id>=1234/v=12/c=2/

Types specified by protocols

Type space for applications

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<id>=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

Matching (exact)

CCN 1.x

/parc/ccn.zip == /parc/ccn.zip

Exact binary match

Matching (exact)

CCN 0.x

/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 ==
cn.zip/.acl/group/owner/key/v1/s9 ==

/parc/ccn.zip

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

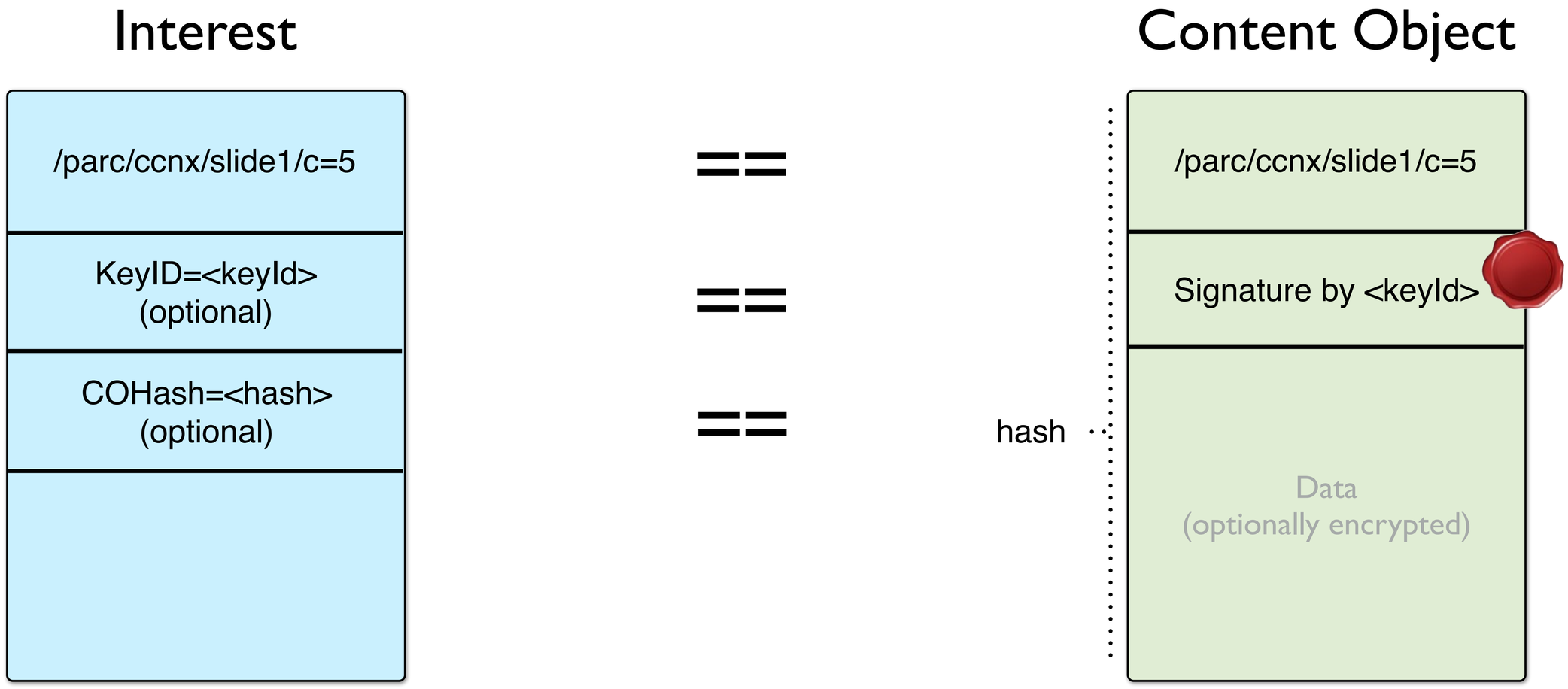
Interest:
name = /parc/ccn.zip/abcd
minSuffixComponents=0
maxSuffixComponents=0

CCN 1.x

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

Interest:
name = /parc/ccn.zip
contentObjectHash=abcd

Core Protocol Primitives



Matching (restrictions) - Why change

No ‘real’ change

Functionally the same

Explicit contentObjectHash matching

Simpler matching (not intermingled)

Loop halting

Loops halted by PIT

Hop-Limit is stop-gap

CCN 1.x

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

PIT
/parc/ccn.zip

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

Nonce-for-loops break aggregation

Interests can’t be aggregated

(if node can treat same nonce interests as equal)

Interest Payload

Interests can carry payload

CCN 1.x

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

payload = abcdefg<1k component>xyz

Interest Payload

CCN 0.x

Interest:
name = /store/cart/abcdefg…
…<1k component>xyz/checkout

CCN 1.x

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

payload = abcdefg<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

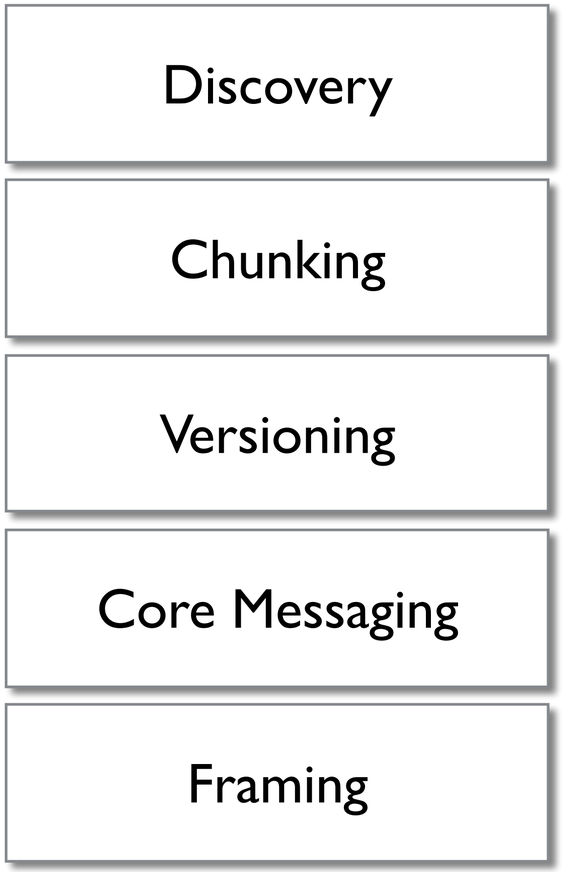
Protocol Separation

Core protocol runs everywhere

Protocol specified individually

Separation of concerns

Modularity

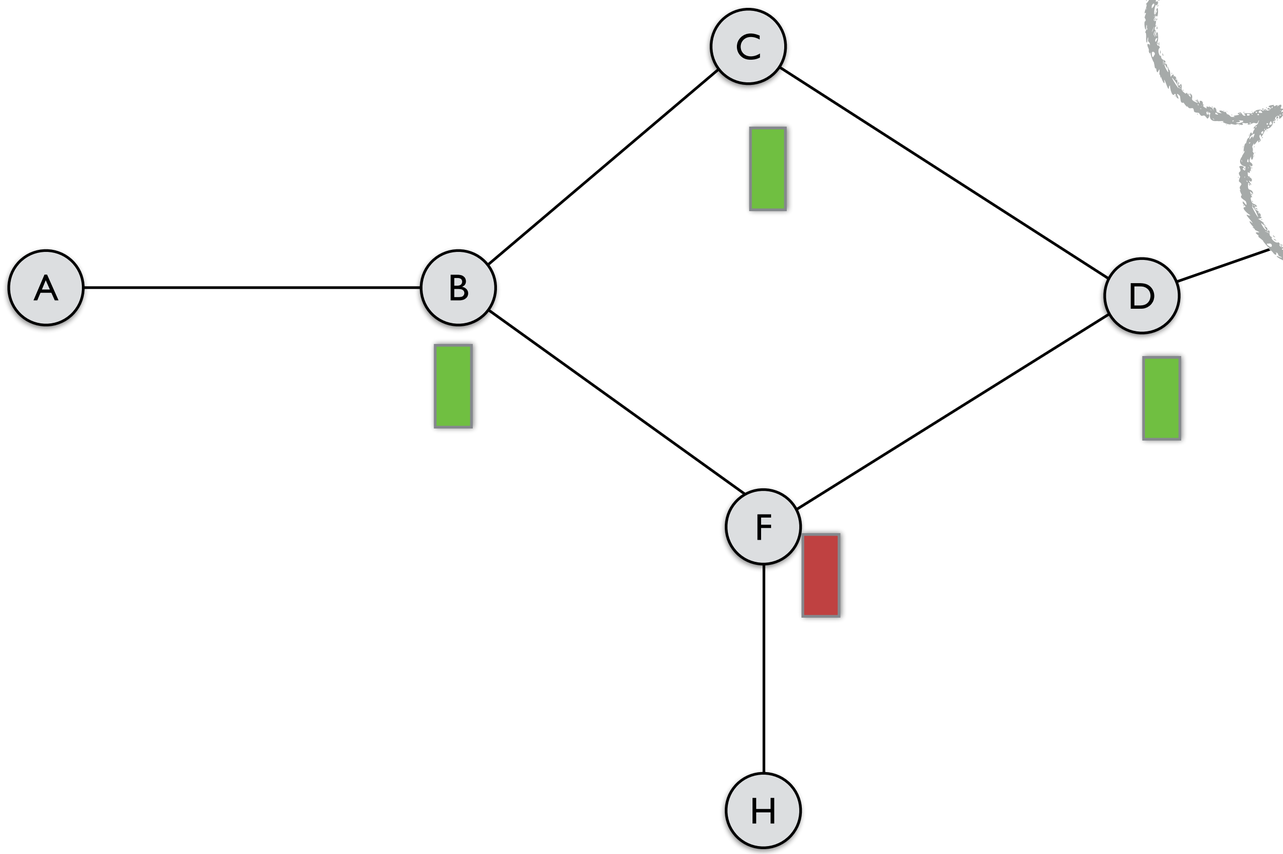


Caching

Universal

Optional

Restricted

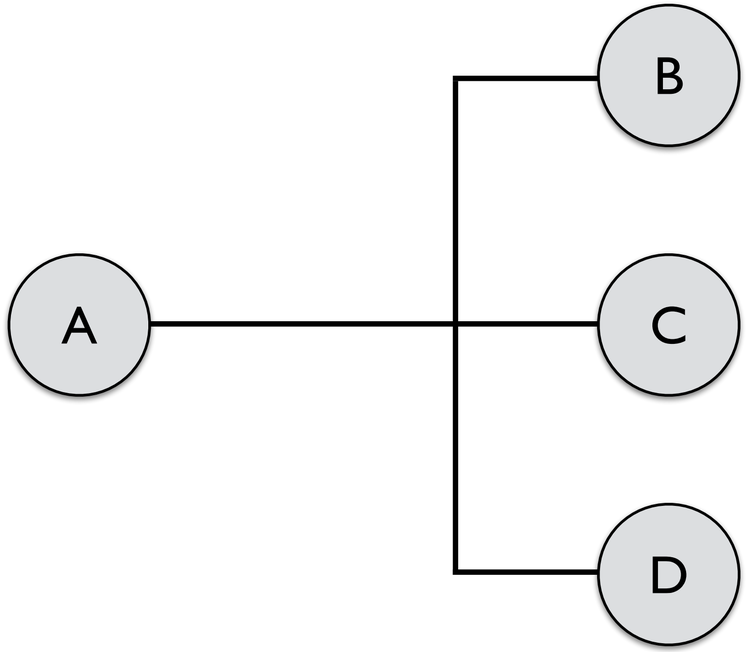


Layer 2

Next hop

Mapping to link layer (ARP)

Fragmentation



CCN Benefits

Security
Data is always secure, in transit and at rest.

Control
The network works in conjunction with the clients

Interoperable
Applications can interoperate transparently

Resilience
The network can operate with minimal interruption

Efficient
Low overhead under heavy demand

Composable
Integrate storage, communication, processing

CCN 1.0

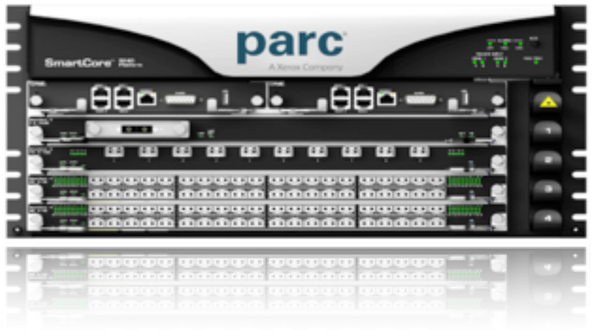
Production quality

Deployable

Feasible

Realistic for all hardware

Hardware



parc
A Xerox Company

CCN - 1.0 Software

parc
A Xerox Company

CCNx 1.0 Releases

Release	Content	Result
R0	Core Protocol	Ability to port base core protocol to new platforms (embedded, desktop or server (Mac OS X & Linux)
R1	Services	Ability to install, start, stop and manage a CCN local network
R2	Storage, Devices, Mobility & Performance	Ability to store & retrieve objects in a CCN network, use express headers & run on Android devices
R3	Platforms, Usability & Applications Framework	Ability to run on Web Browsers, Windows, iOS & FreeBSD, write & test “network aware” applications and add functionality to CCN

CCN 1.0 Software

What Is It?

codename - **Distillery**

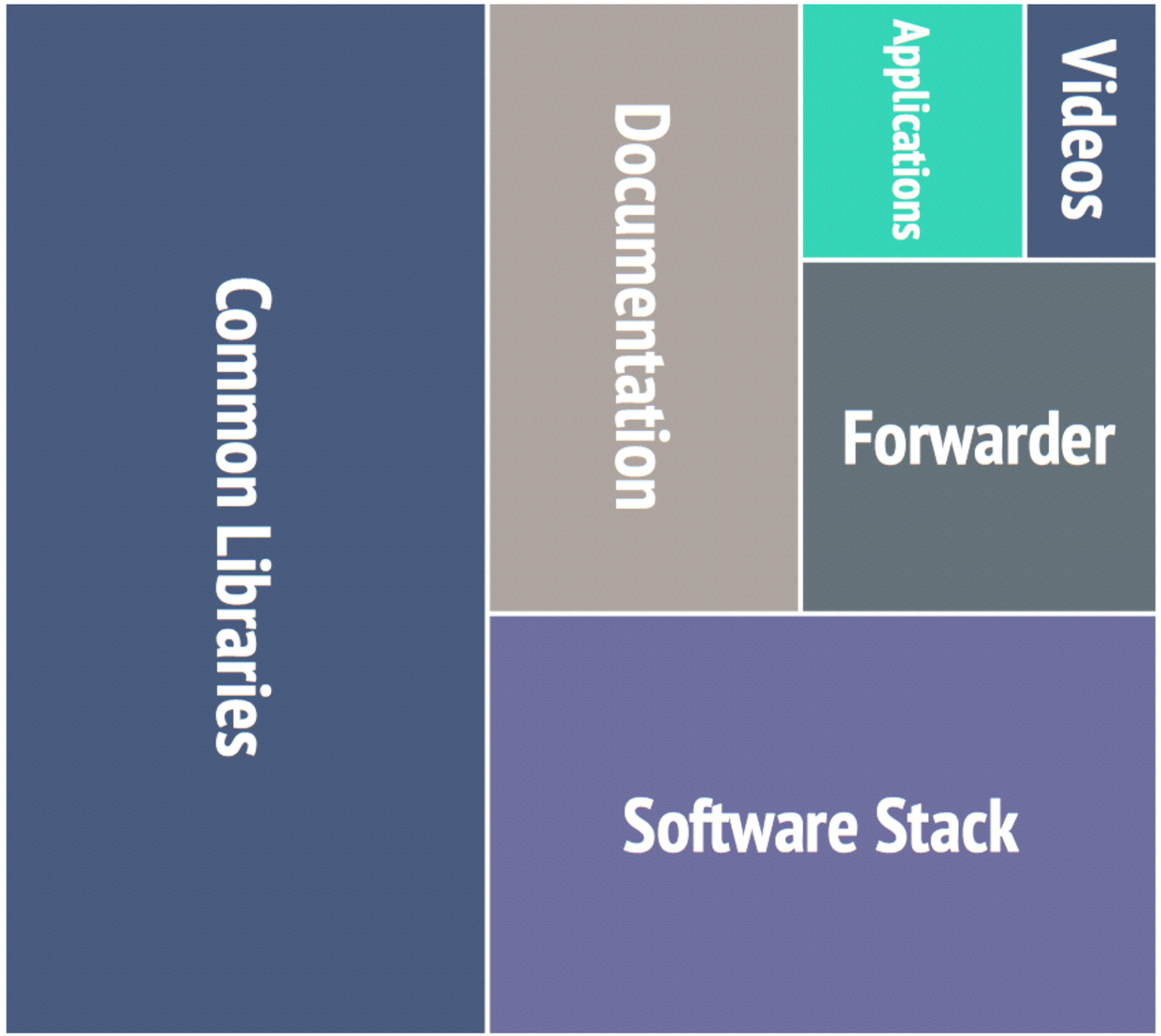
Application Programs

Software Stack

Common Libraries

Documentation

Instructional Videos



CCN 1.0 Software - Common Libs

C Support

codename - **LongBow**

Write Better C Programs

Runtime Assertions and Traps

```
assertTrue
assertFalse
assertNull
assertNotNull
```

```
trapIllegalValue
trapNotImplemented
TrapOutOfBounds
trapOutOfMemory
trapUnexpectedState
```

Native C Unit Test Framework

xUnit-style testing for C (in C)

Integrates with runtime assertions and traps

Integrated with automake, Xcode, Eclipse

CCN 1.0 Software - Common Libs

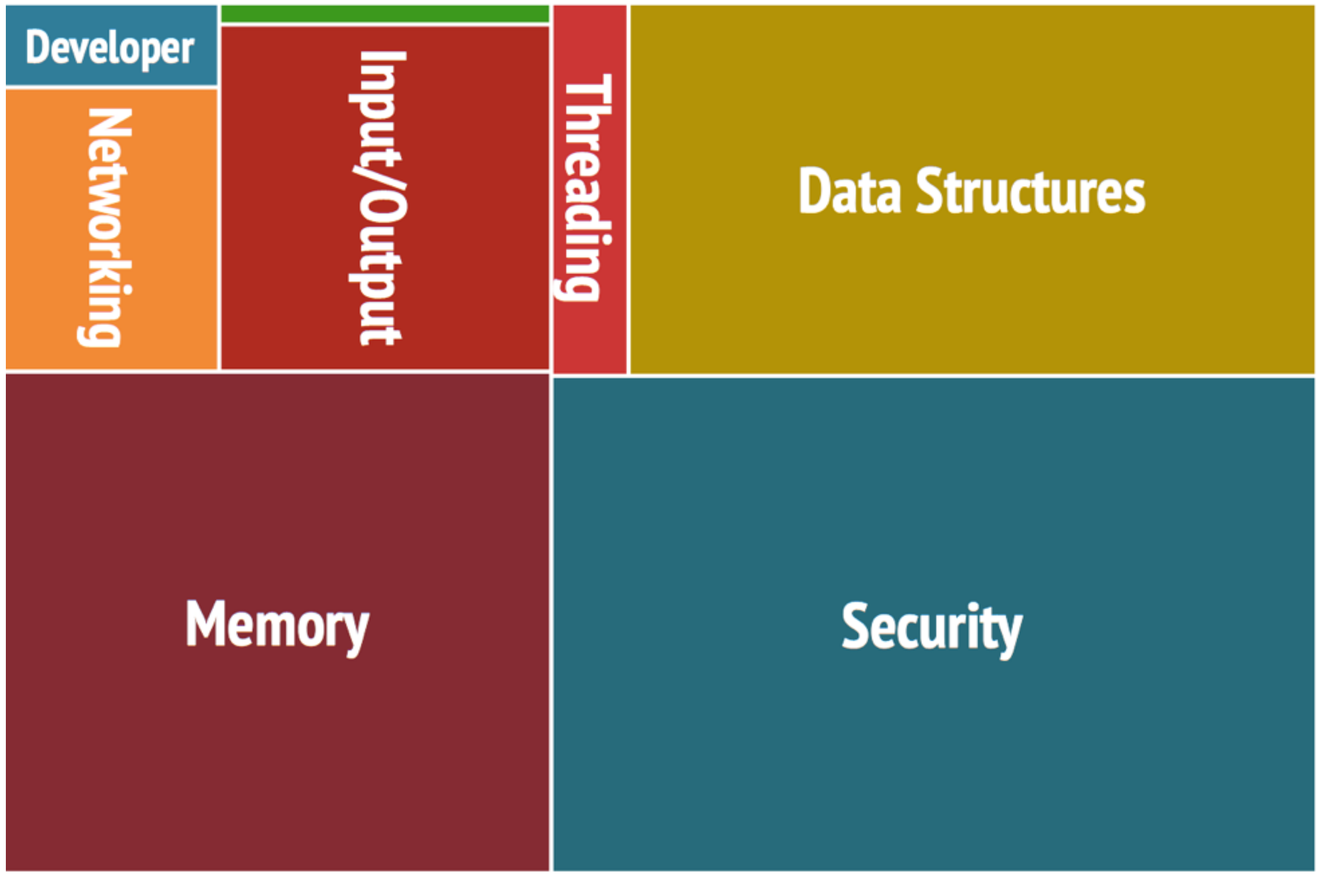
C Library

codename - **PARCLib**

Data Structures

Algorithms

Utility Functions



parc
A Xerox Company

CCN 1.0 Software - Common Libs

CCN Library

- Abstractions
- Data Structures
- Algorithms
- Utility Functions

CCN Packets

`Headers, TLVs`

CCN Names

`Components, Types`

CCN Messages

`ContentObjects`

`Interests`

`Manifests`

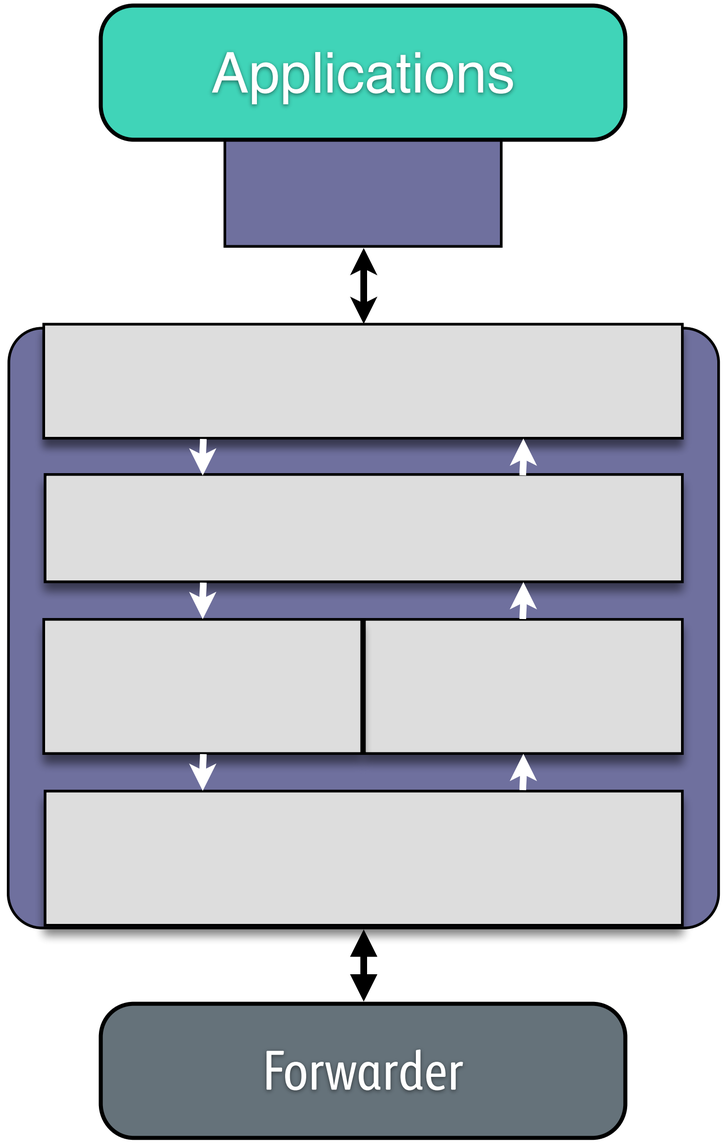
…

CCN 1.0 Software

CCN Stack

Transport Framework

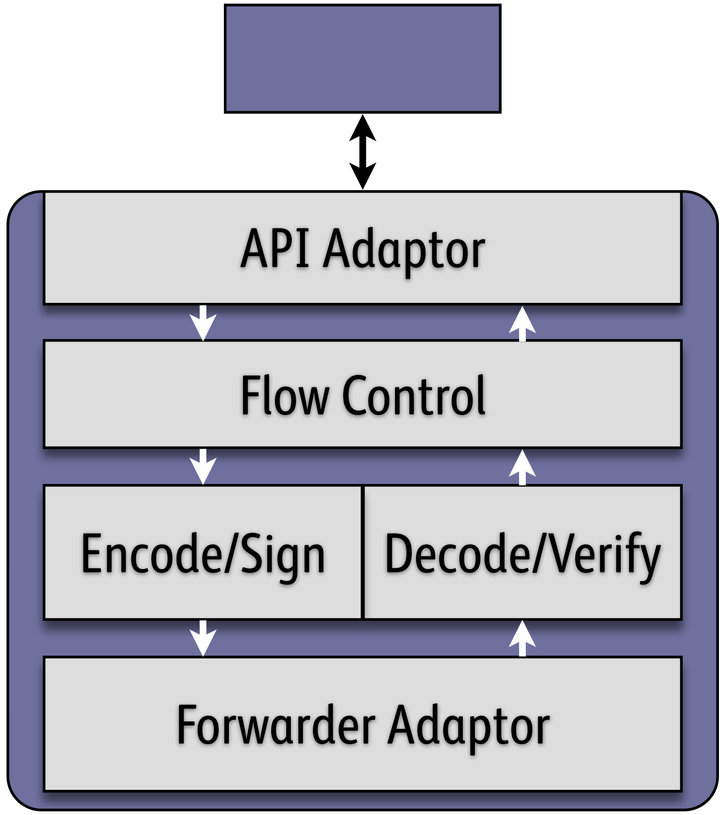
Portal API



CCN 1.0 Software - Stack

Transport Framework

- Component Based Design
- Dynamically Plumbed
- Dynamically Loaded Modules (planned)



CCN 1.0 Software - App View

Portal API

Native

Simple

CCNxNameSegmentType

CCNxKeystoreUtilities

CCNxContentObject

CCNxName

CCNxInterest

CCNxPortalMessage

CCN 1.0 Software - API View

APIs

Straight forward

Control transport

CCNxControl

CCNxMessage

CCNxName

CCNxInterest

CCNxContentObject

CCNxKeyLocator

CCN 1.0 Software - Stack view

Components

Data manipulation

Stack communication

parc
A Xerox Company

CCN 1.0 Software

Forwarder

codename - **Metis**

Built for experimentation

Efficient

Easy to Use

Simple application

No special privileges

Configurable Cache Size

In memory cache, 0 to …

CCN 1.0 Packet Format

TCP/IP encapsulation

Native ethernet

CCN 1.0 Software - Goals

Written for Experimentation

- Interface Based Architecture
  - Clear Separation of Concerns
  - Simple to Substitute Different Implementations
- Modular Design
  - Promotes Extensibility
  - Interoperability Testing
  - Graduated/Progressive Implementation

CCN 1.0 Software - Goals

Written for People

Human Factors Emphasis

Documentation

- Function Documentation
- Tutorial Guides

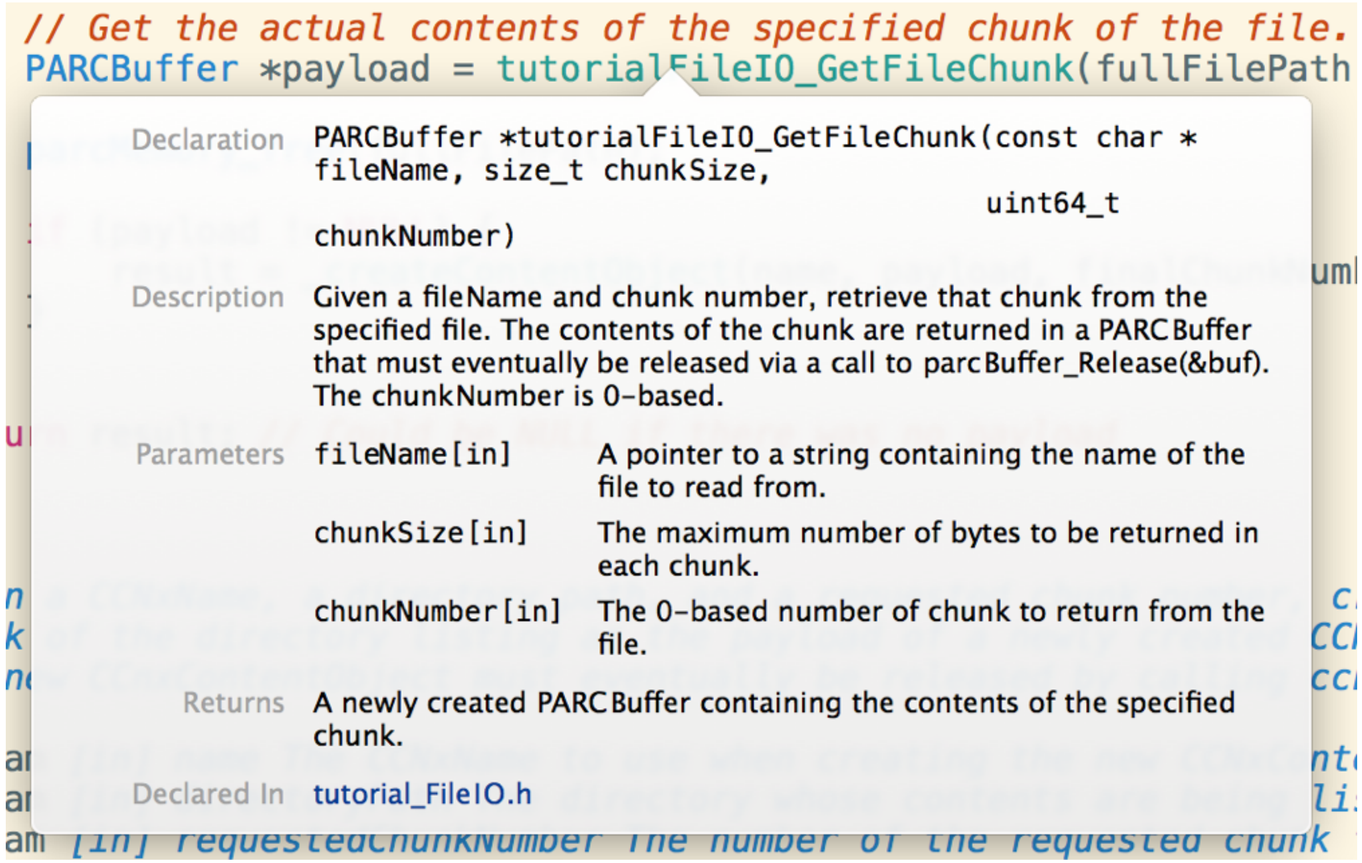
Consistent Design and Style

Measured Software

CCN 1.0 Software

Documentation

- 100% Coverage
  - Module,
  - Function,
  - Enumeration
  - Type
- IDE Integration
- Printed and Online

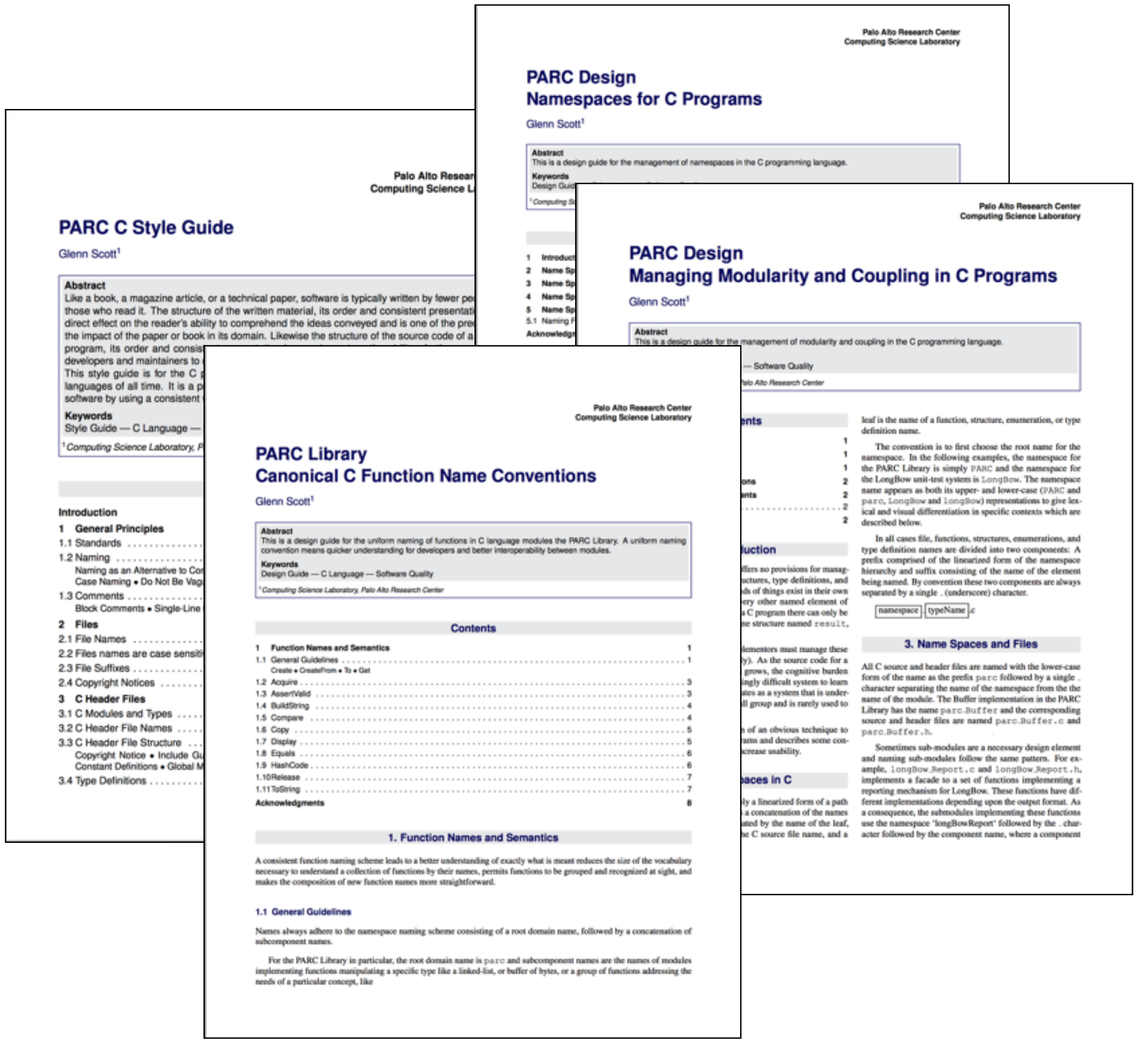


parc
A Xerox Company

CCN 1.0 Software

Consistent Design and Style

- Uniform Code Style
- Clean and Clear Naming
- High Cohesion
- Low Coupling



parc
A Xerox Company

CCN 1.0 Software

Measured

Quality Control

- Style Conformance
- Naming Conformance
- Unit Testing
- Documentation Coverage
- Complexity Management

Developer Tools

- Readiness and Acceptance

CCNx Dashboard

Source Code

	Main	*Testing*	*Test Coverage*
Forwarder	19	17	
APIs	14	8	
Transport	27	15	
CCN Library	25	14	
C Library	38	22	
C Support	10	…	
Total Lines	***136k***	***77k***	***74%***

Developer Documentation

	Print	*HTML*	*Coverage*
Forwarder	315	354	
APIs	301	239	
Transport	438	471	
CCN Library	604	234	
C Library	907	413	
C Support	225	175	
Total Pages	***2,864***	***1,972***	***92%***

Code Complexity

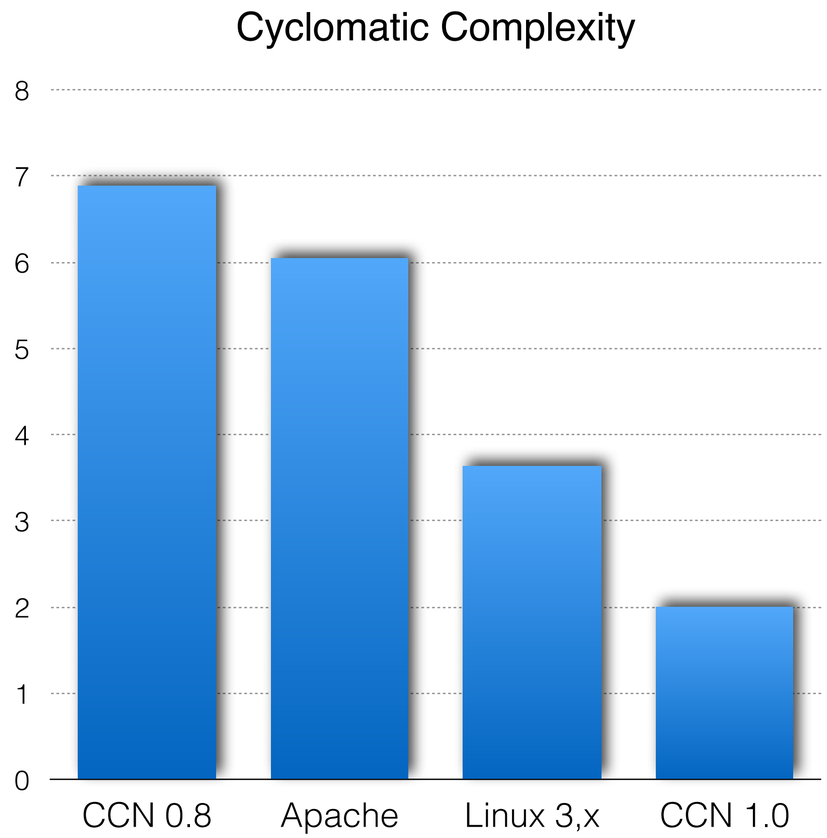
	Cyclomatic	*Vocabulary*
Forwarder	1.77	55
APIs	1.88	51
Transport	2.43	70
CCN Library	2.29	45
C Library	1.97	43
C Support	2.00	39
Average	***2.00***	***50***

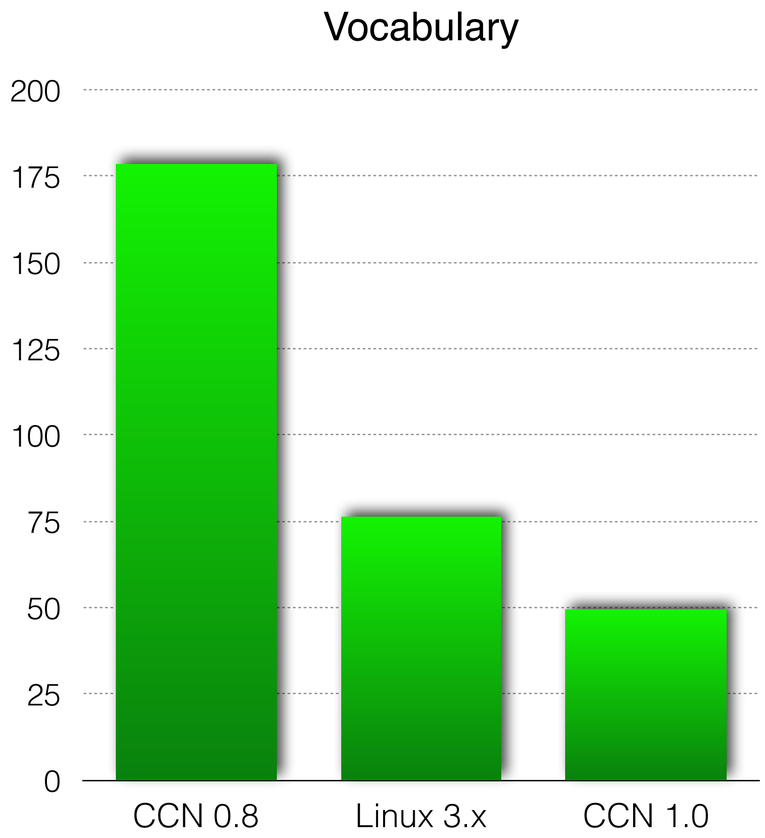
Design: Modularity

	Modularity
Forwarder	
APIs	
Transport	
CCN Library	
C Library	
C Support	
Average	***84%***

parc
A Xerox Company

CCN 1.0 Software





parc
A Xerox Company

CCN - Future

parc
A Xerox Company

Control is good

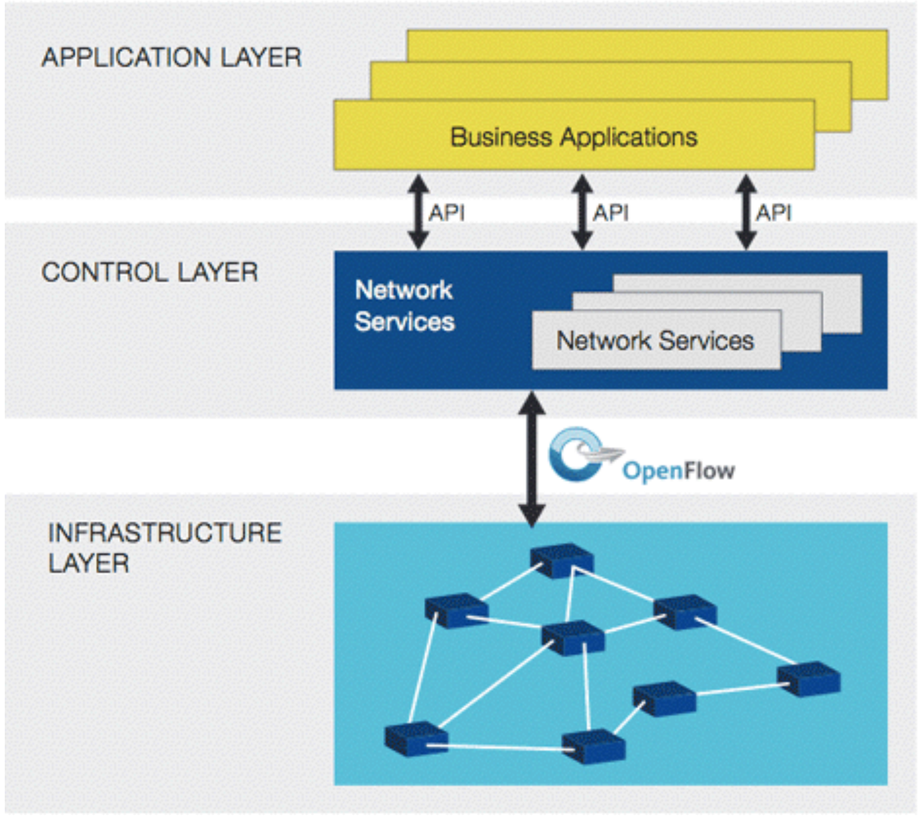
Security is good

Isolation is good

Virtualizing Operating Systems

Virtualizing Network

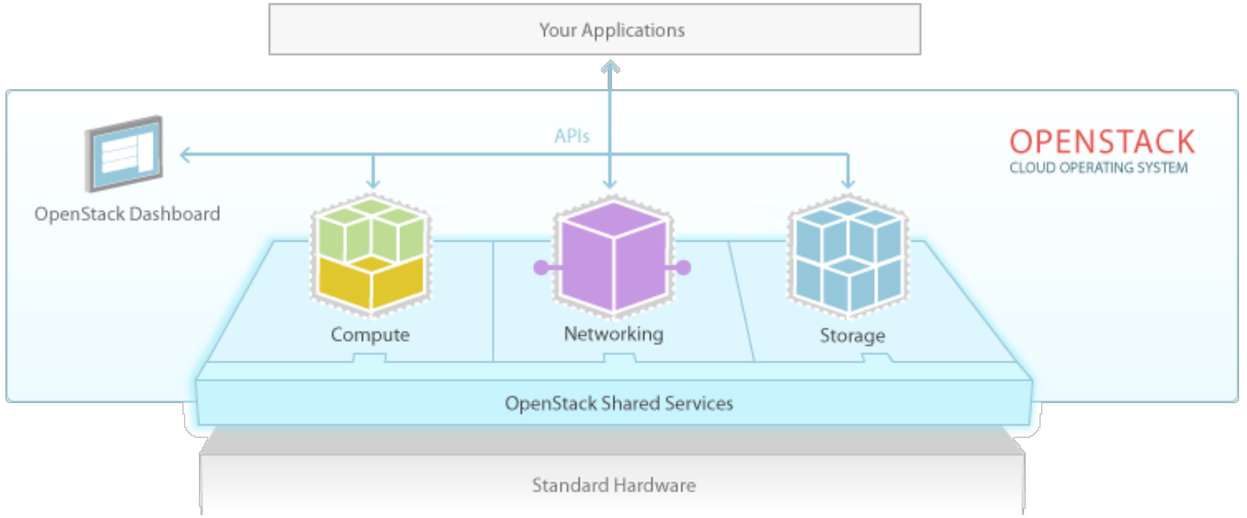
OpenFlow™



parc
A Xerox Company

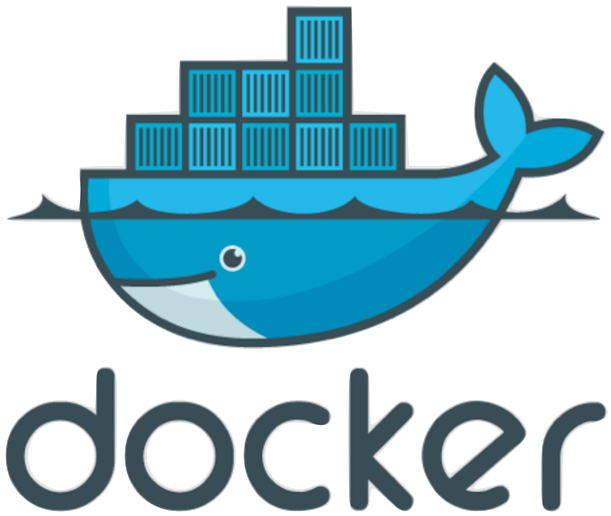
Virtualizing Cloud

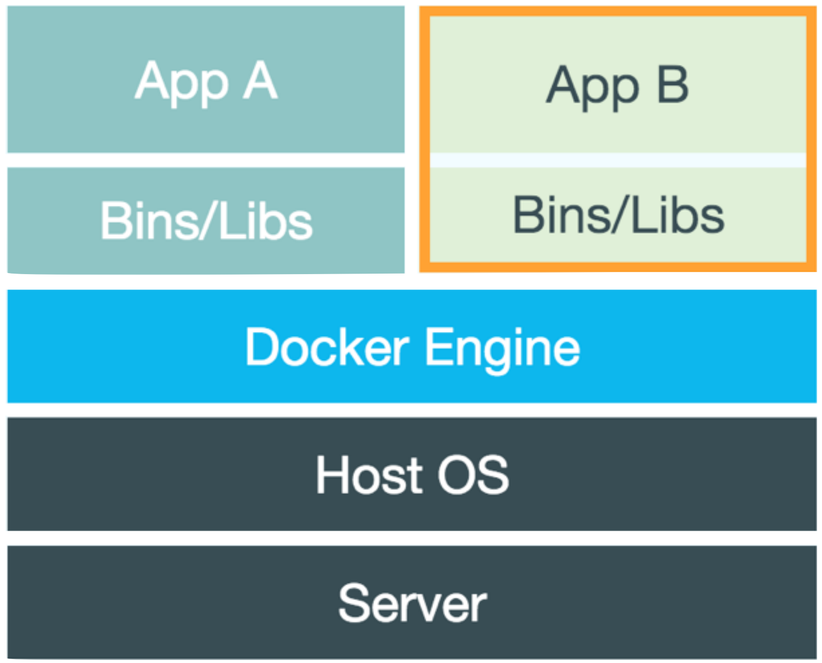
openstack™



parc
A Xerox Company

Virtualizing Applications





Solitary confinement is not for everybody

Integration

Robust and secure Integration

Virtual Platform

(the next logical step)

Platform

The base on which we build applications and services

Platform

Identity
Authentication
Security

Storage
Archival
Communication

Integration
Coordination
Federation

Platform

Identity
Authentication
Security

Storage
Archival
Communication

Integration
Coordination
Federation

CCN Virtual Platform

Codename: COPA

Platform

Identity
Authentication
Security

Storage
Archival
Communication

Integration
Coordination
Federation

parc
A Xerox Company

CCN Virtual Platform

Codename: COPA

Identity

Understands individuals, groups and organizations

Authentication

Validates and verifies identities

Security

Enables access control at various granularities

CCN Virtual Platform

Codename: COPA

Storage

Abstracts storage from applications

Archival

Keeps versions, histories and backups

Communication

Abstracts messaging between applications

CCN Virtual Platform

Codename: COPA

Integration

Allows applications to share data and resources

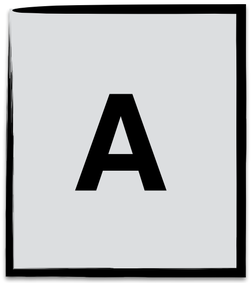
Coordination

Enables applications to work together

Federation

Bridges administrative domains

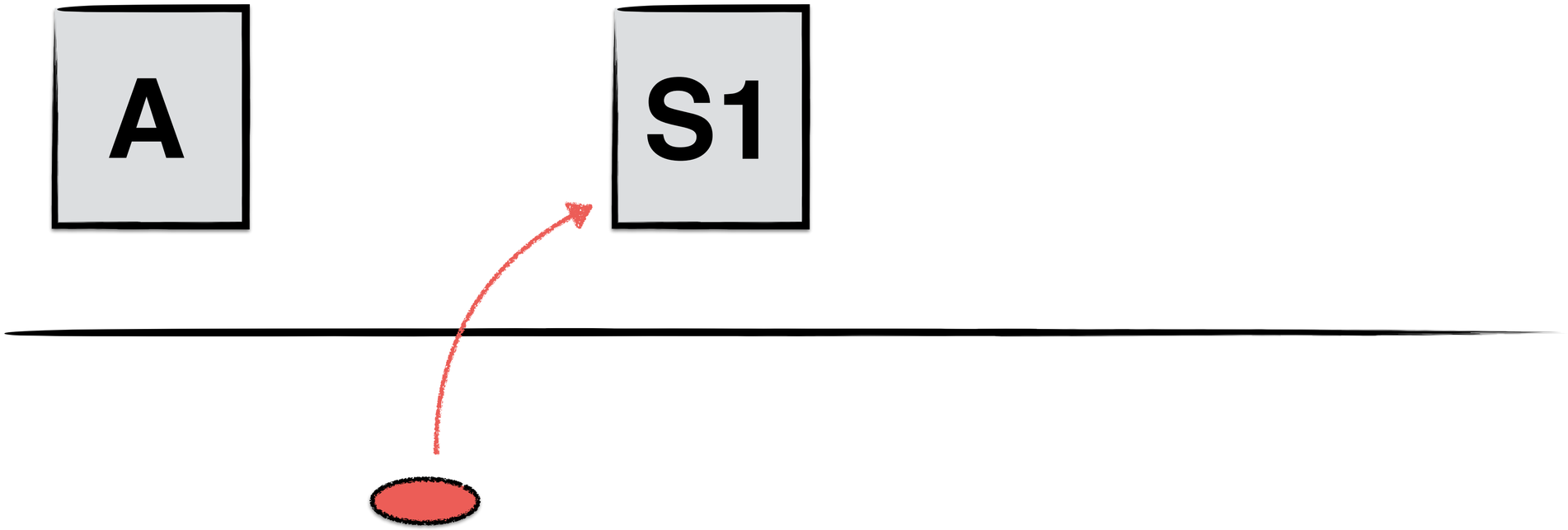
Simple Example



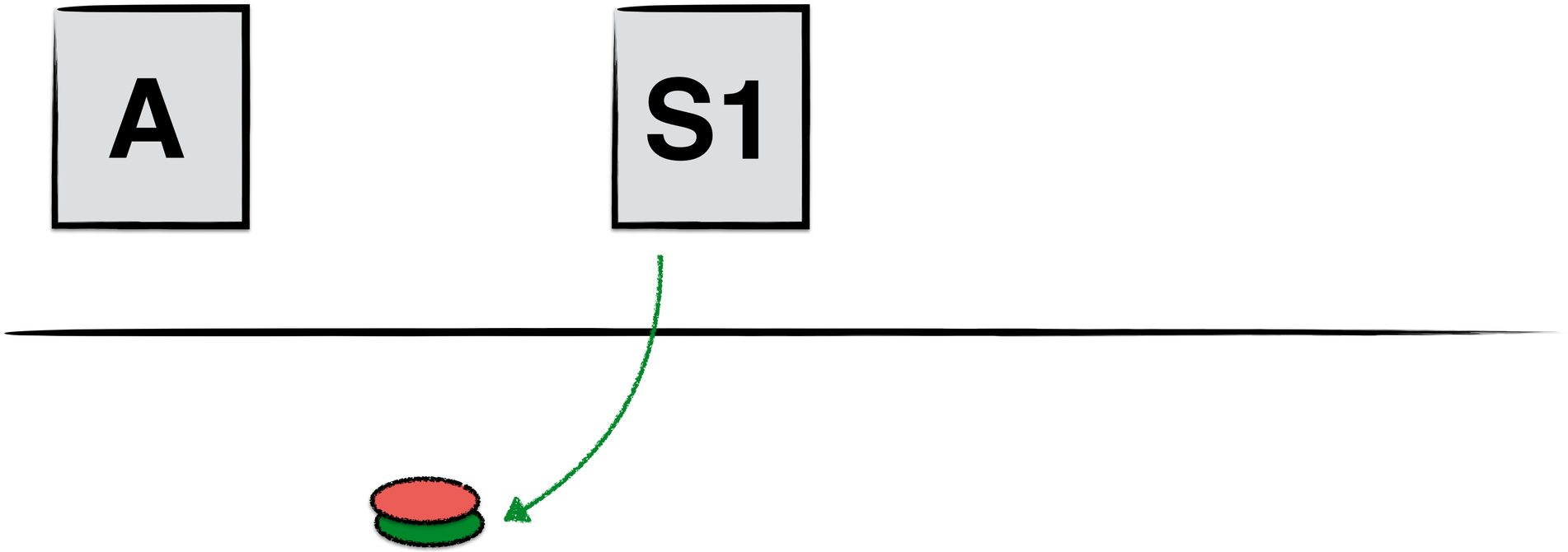
Authenticate

Store data

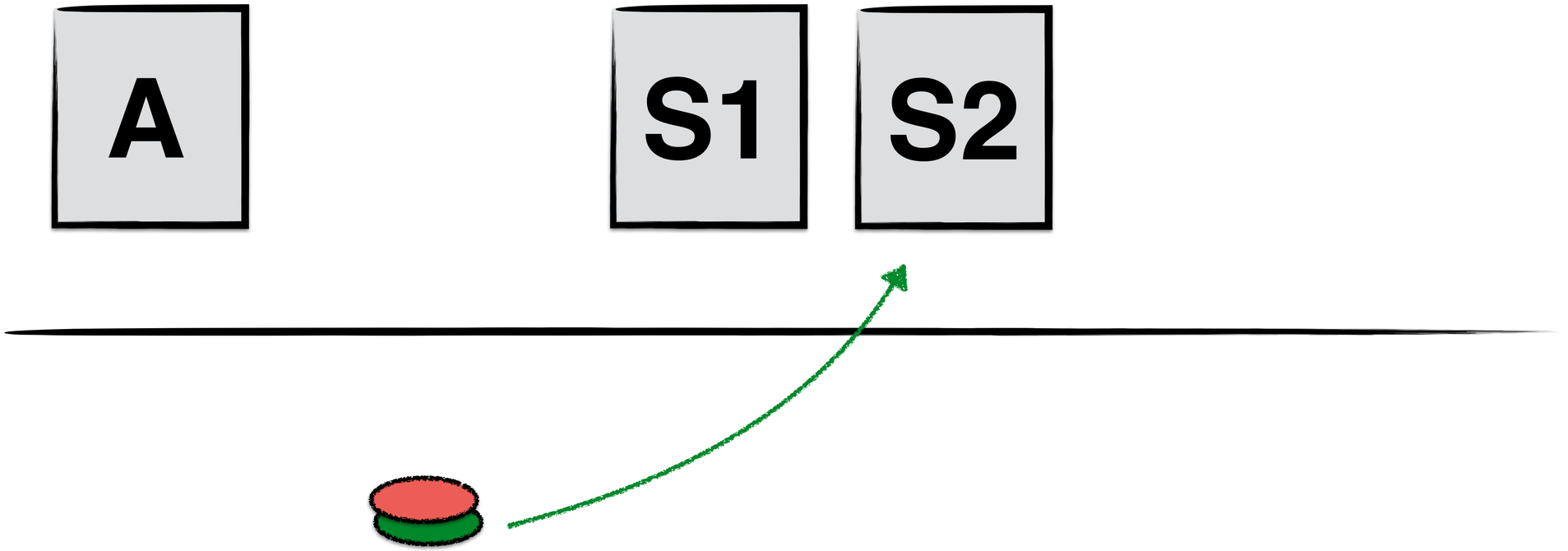
Notify Services



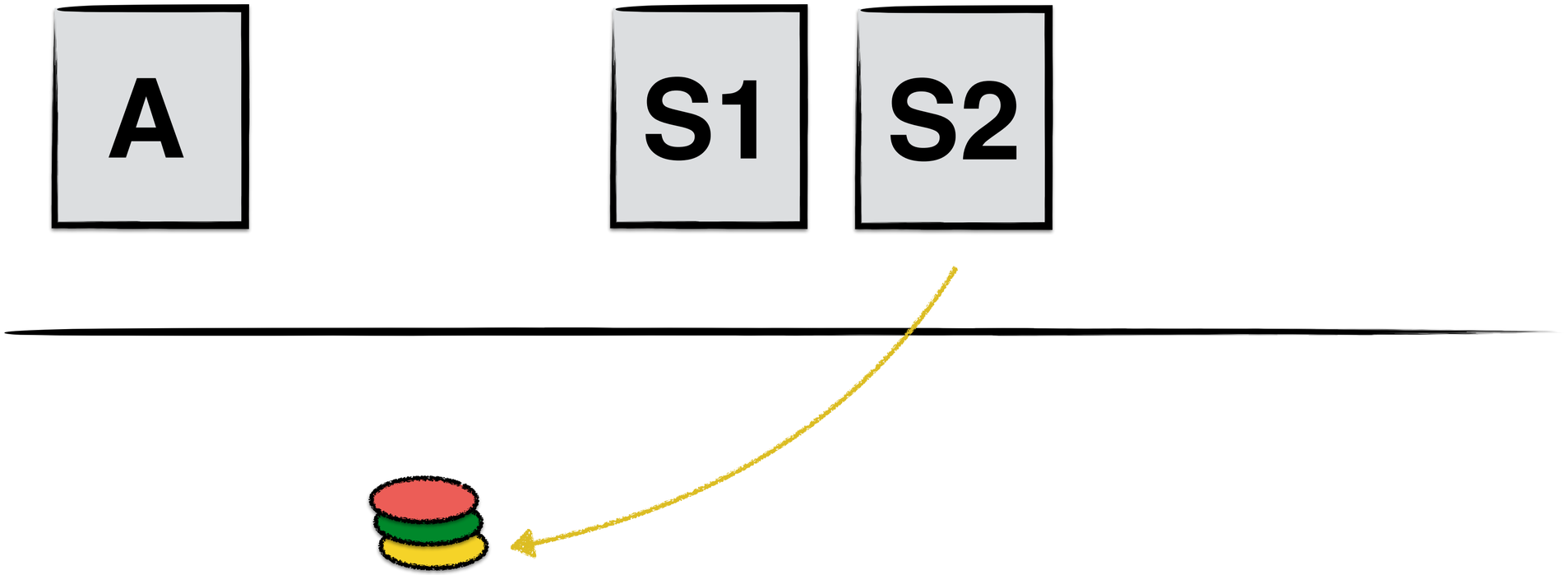
Augment data



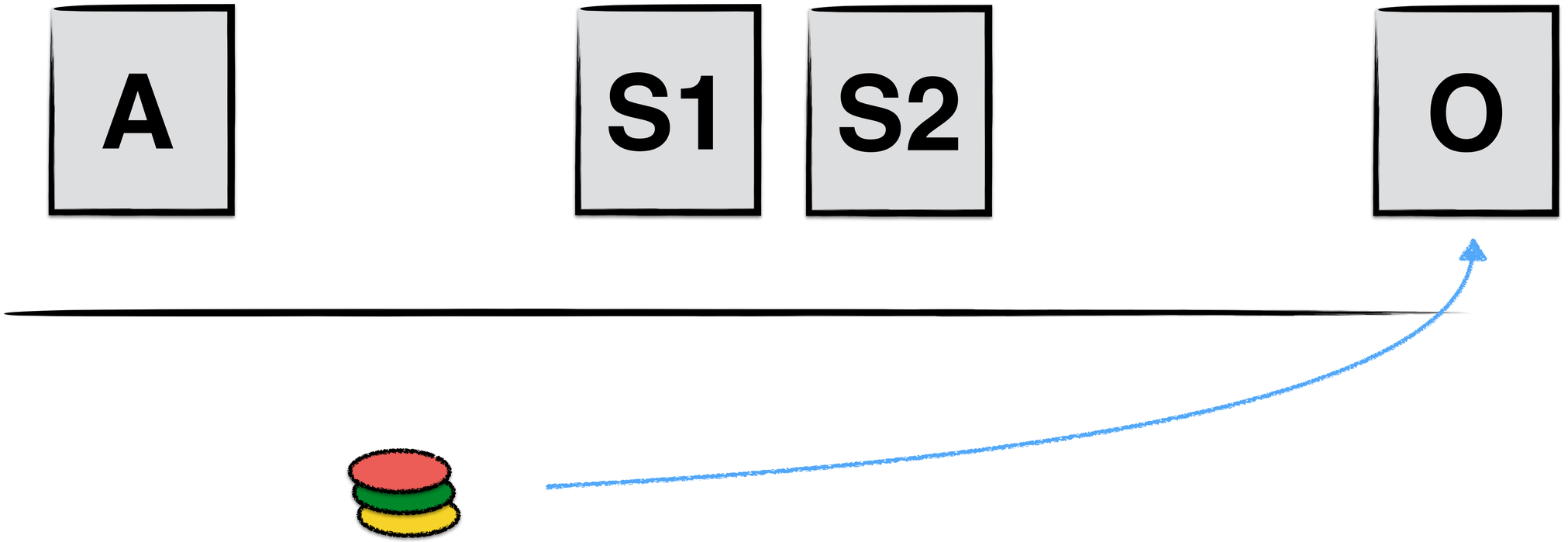
Notify Services



Augment data



Notify outputs

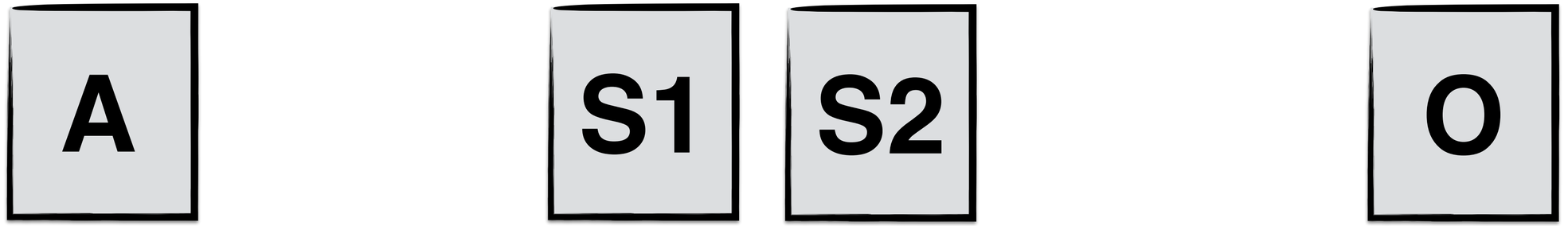


Form scan

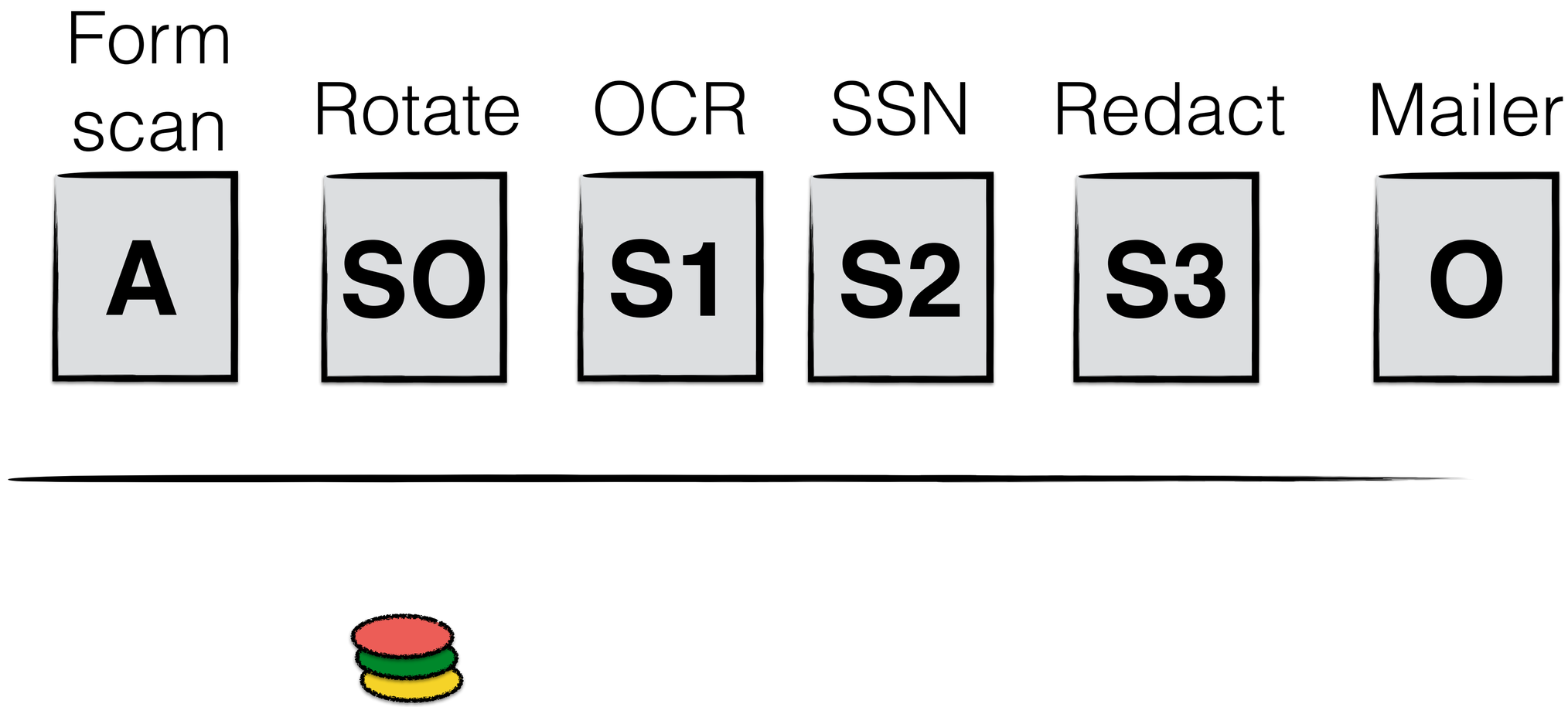
OCR

SSN

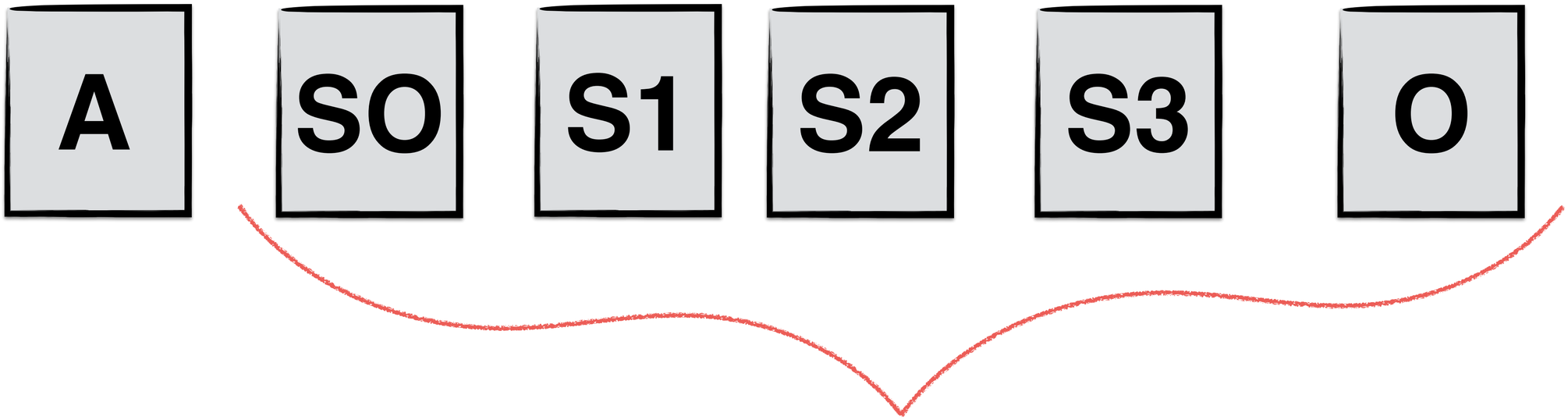
Mailer



parc
A Xerox Company



Ecosystem



Uncoordinated
Dynamic
Federated

PARC Experience

Benefits from research into workflows

Benefits from research in APIs and languages

Benefits from research in data representation

Benefits from research in planning

Built on research in networking - CCN

CCN enablers

Secure communication (data at rest and in transit)

Efficient content transfer (eliminates redundancy)

Data storage in coordination with the network (storage abstraction)

Data reuse (same data, multiple applications)

Smart network rulesets (communication decisions based on metadata)

CCN Virtual Platform

Codename: COPA

Platform

Identity
Authentication
Security

Storage
Archival
Communication

Integration
Coordination
Federation

parc
A Xerox Company

parc®

A Xerox Company

Thank you

http://www.ccnx.org/