# Multipurpose IP/NDN Gateway and Bridge for Heterogeneous Network Interoperability

Christopher A. Wood
www.christopher-wood.com
woodc1@uci.edu

May 21, 2014

# Agenda

# Today's Internet: Communication Networks as Distribution Networks

The communication-centric design enables point-to-point communcation between any two parties:

- Names and interfaces
- Supports end-to-end conversations
- Provides unreliable packet delivery via IP datagrams
- Compensates for simplicity of IP via complexity of TCP

Important observations:

- Helped facilitate today's concent-centric world, *but was never designed for it!*
- Fundamental communication model: point-to-point conversation between two hosts (IP interfaces)
- The central abstraction is a host identifier corresponding to an IP address

# Data vs Communication Networks

Distribution/data (DN) and communication (CN) networks differ in several key ways:

| | CN | DN |
|---|---|---|
| Naming | Endpoints | Content |
| Memory | Invisible & limited | Explicit (storage = wires) |
| Security | Communication process/channel | Content |

# CCN Overview

Content-centric networking flips around the host-based model of the Internet architecture

- *Content names*, rather than content locations, become addressable.
- The network is permitted to store (cache) content that is in high demand
- End result: less traffic to/from the content's original source, better usage of network resources, less latency, etc etc.

# CCN Overview (continued)

How is data actually retrieved?

- A consumer $C$ sends out an *interest* for content they desire.
- A router $R_i$ use the information in their forwarding information base (FIB) table and data in cached in their content store (CS) to handle incoming interests:
  1. If content with the same name matches what's stored in the CS, return that content
  2. Else, store the interest in their pending interest table (PIT) (including the downstream router $R_{i-1}$ or consumer $C$ that made the request), and forward the request upstream to the next router $R_{i+1}$ based on their FIB.
  3. FIBs are configured using protocol similar to OSPF
- Once the interest is satisfied in $R_i$, the PIT entry is cleared, the content is cached, and the data is sent downstream to $C$ or $R_{i-1}$.

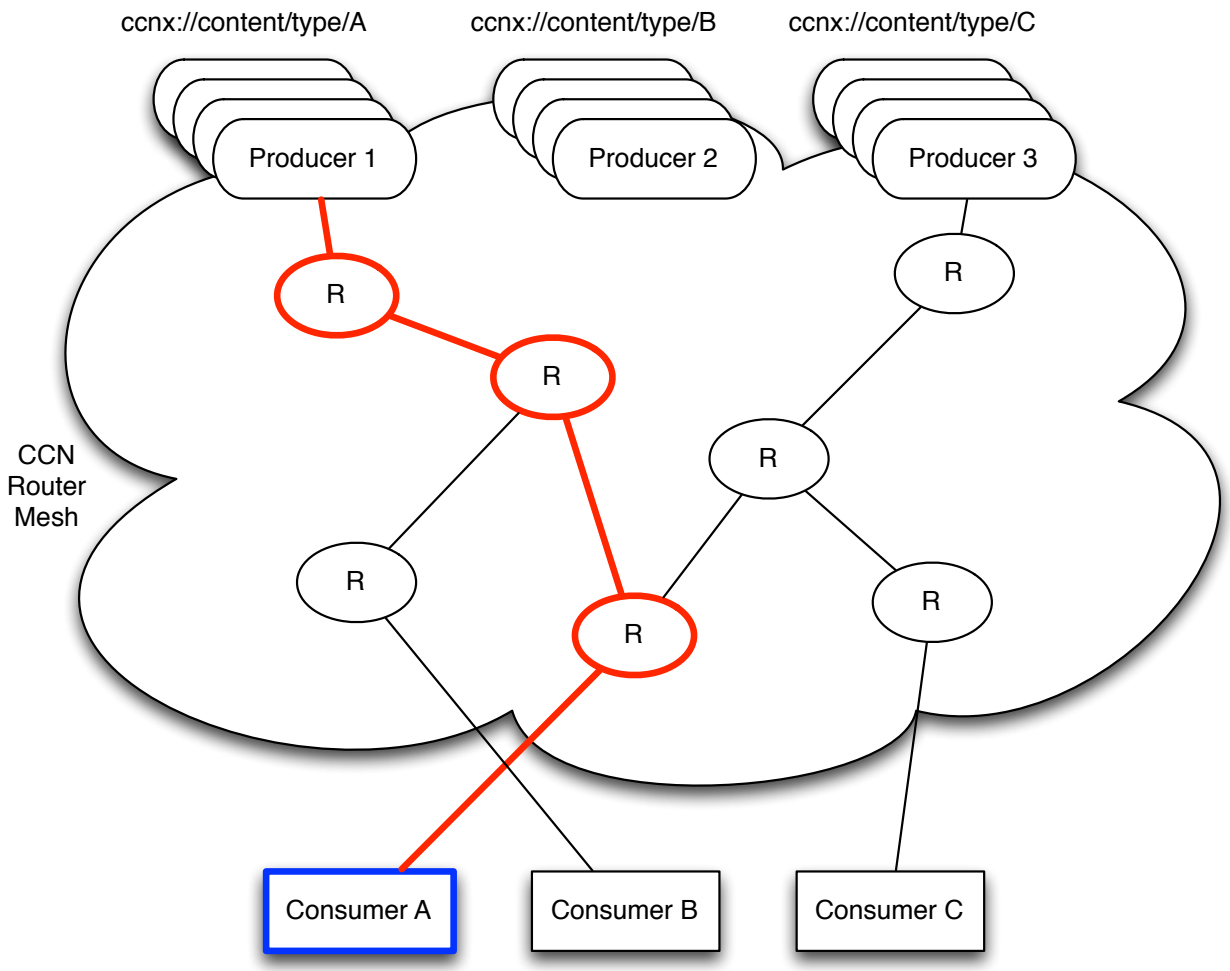
# Interest Format

- Interests are similar to URLs:

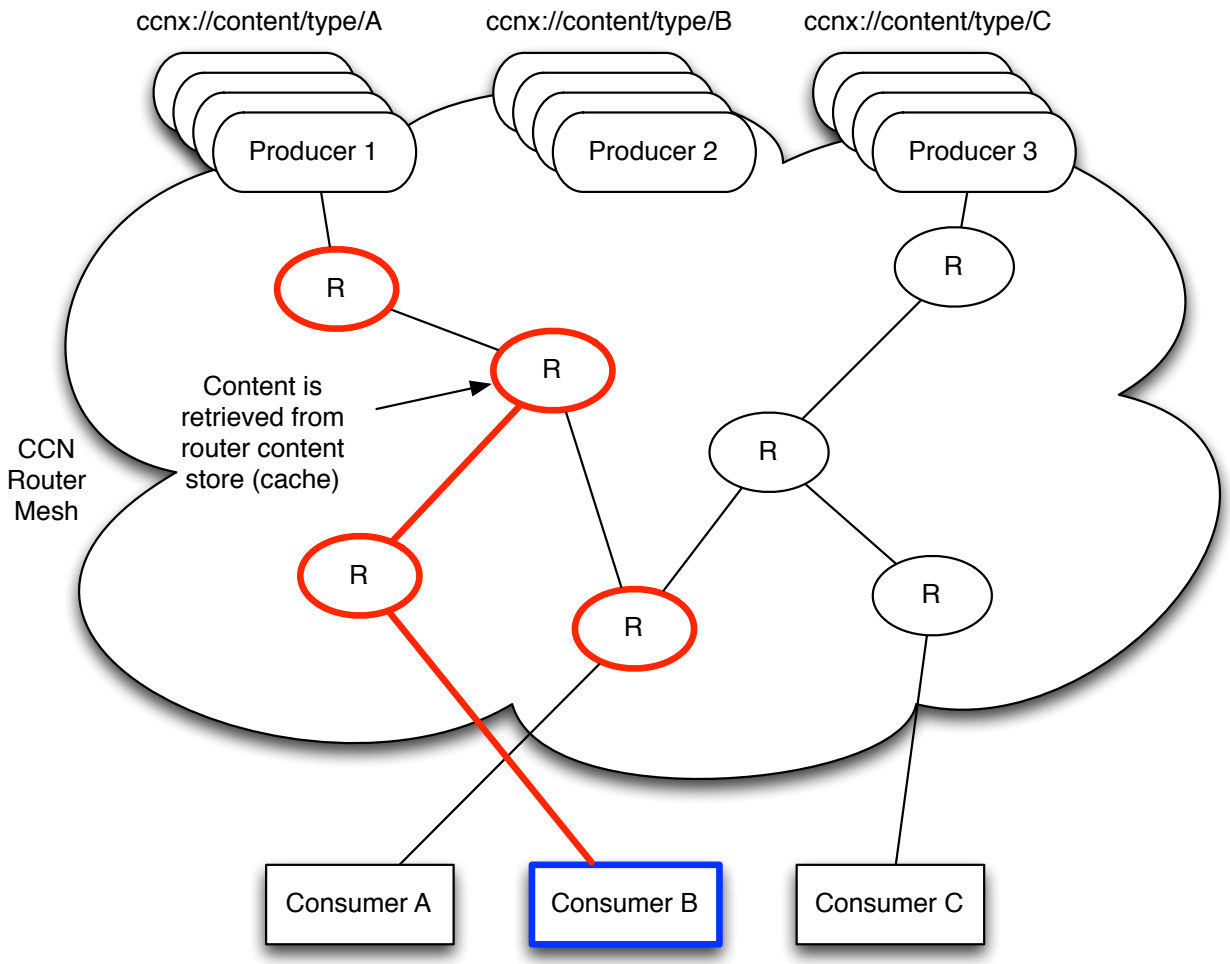
  `ccnx://rit/gccis/cs/spr/ramsey_survey`

- The / character is a delimeter that separates name *components*
- A component can be *anything*, including binary data (e.g. ciphertext)
- Interests are matched to providers in FIBs using a standard longest-prefix rule (to my knowledge, interests in CSs must match completely)

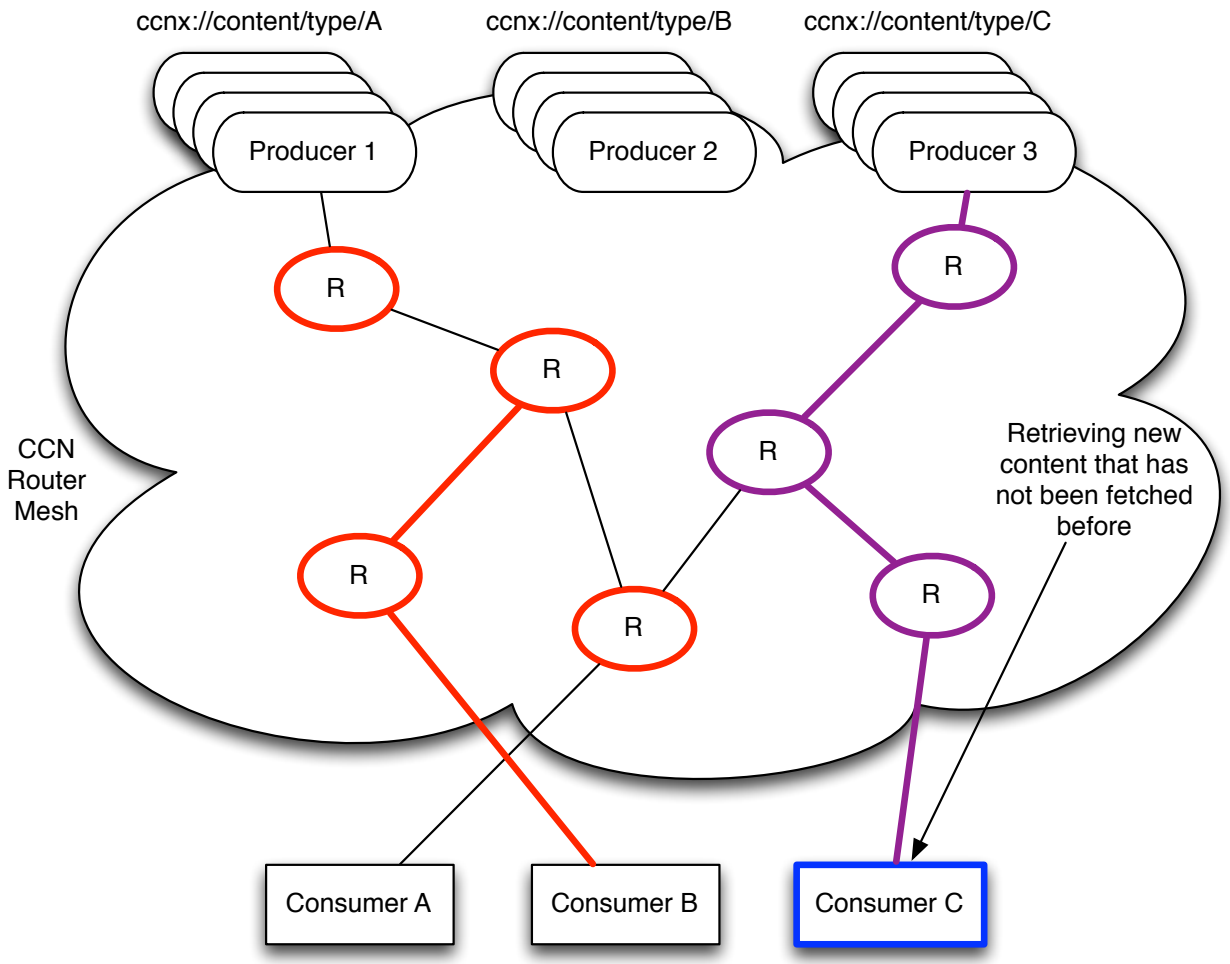
# CCN in Action - #1



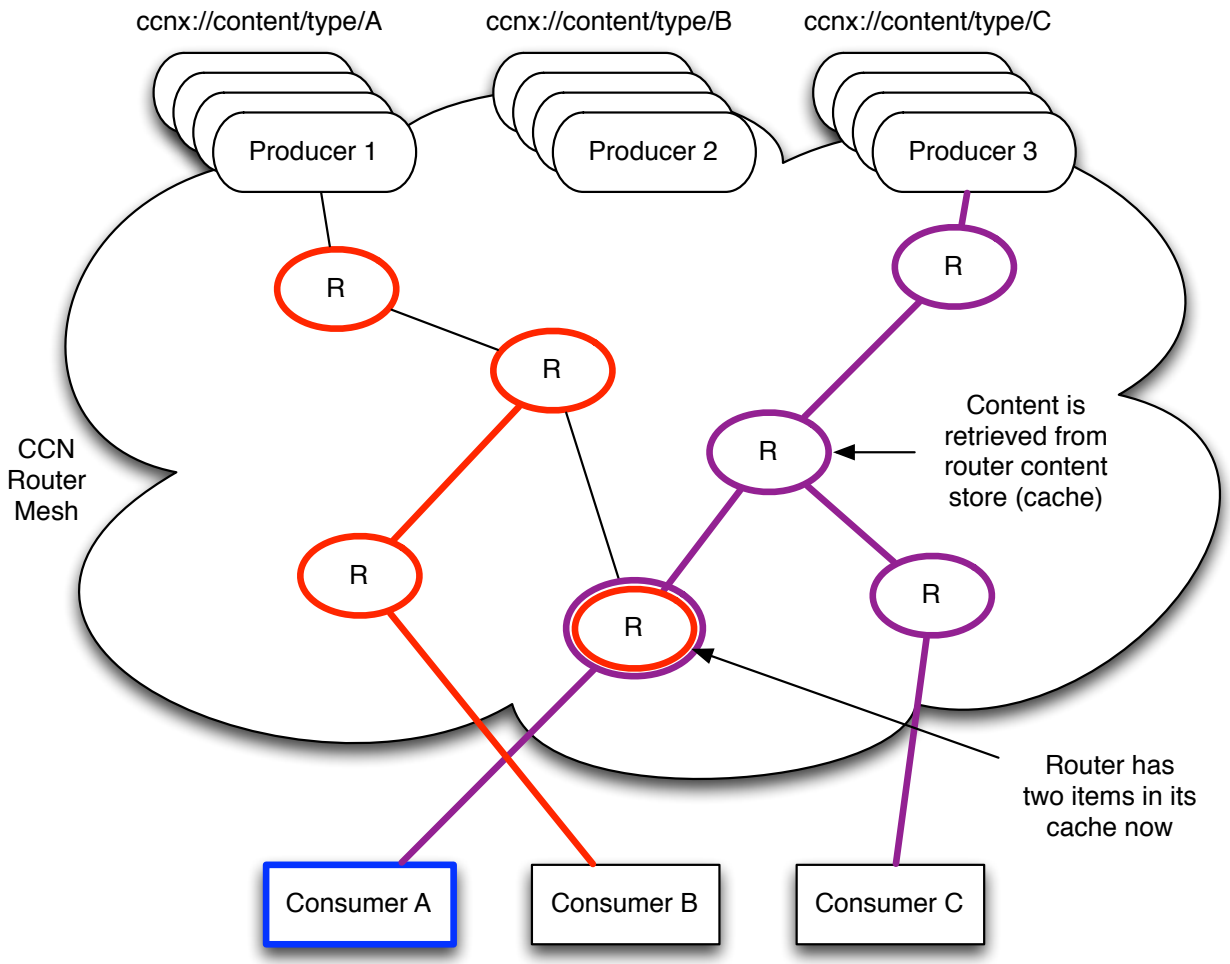
# CCN in Action - #2



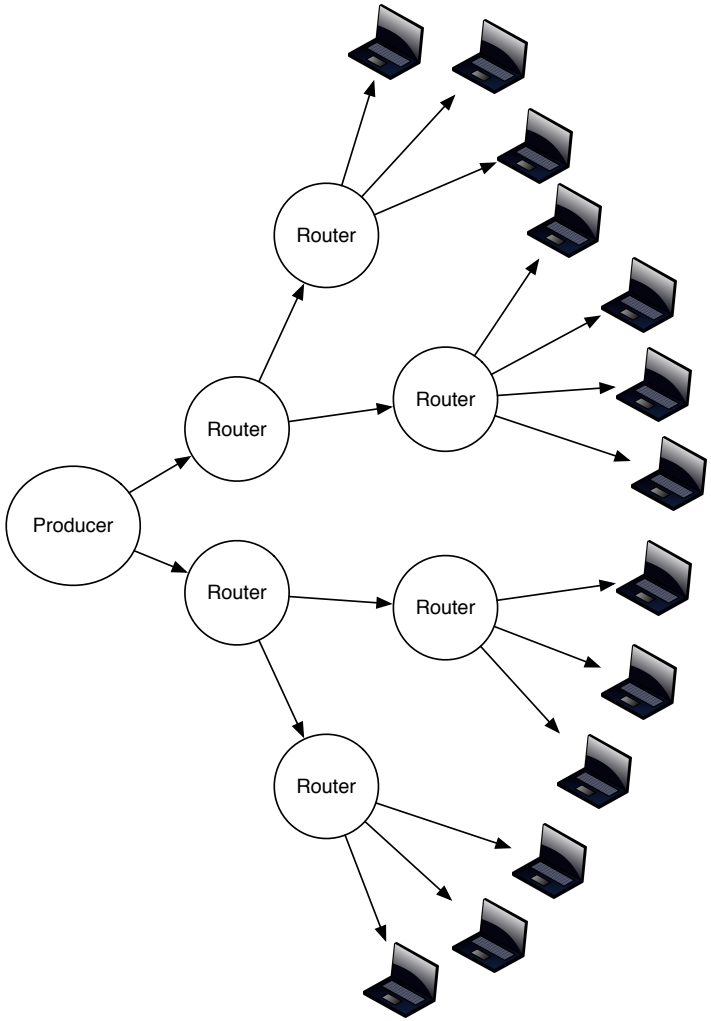
# CCN in Action - #3



# CCN in Action - #4



# CCN at a Larger Scale



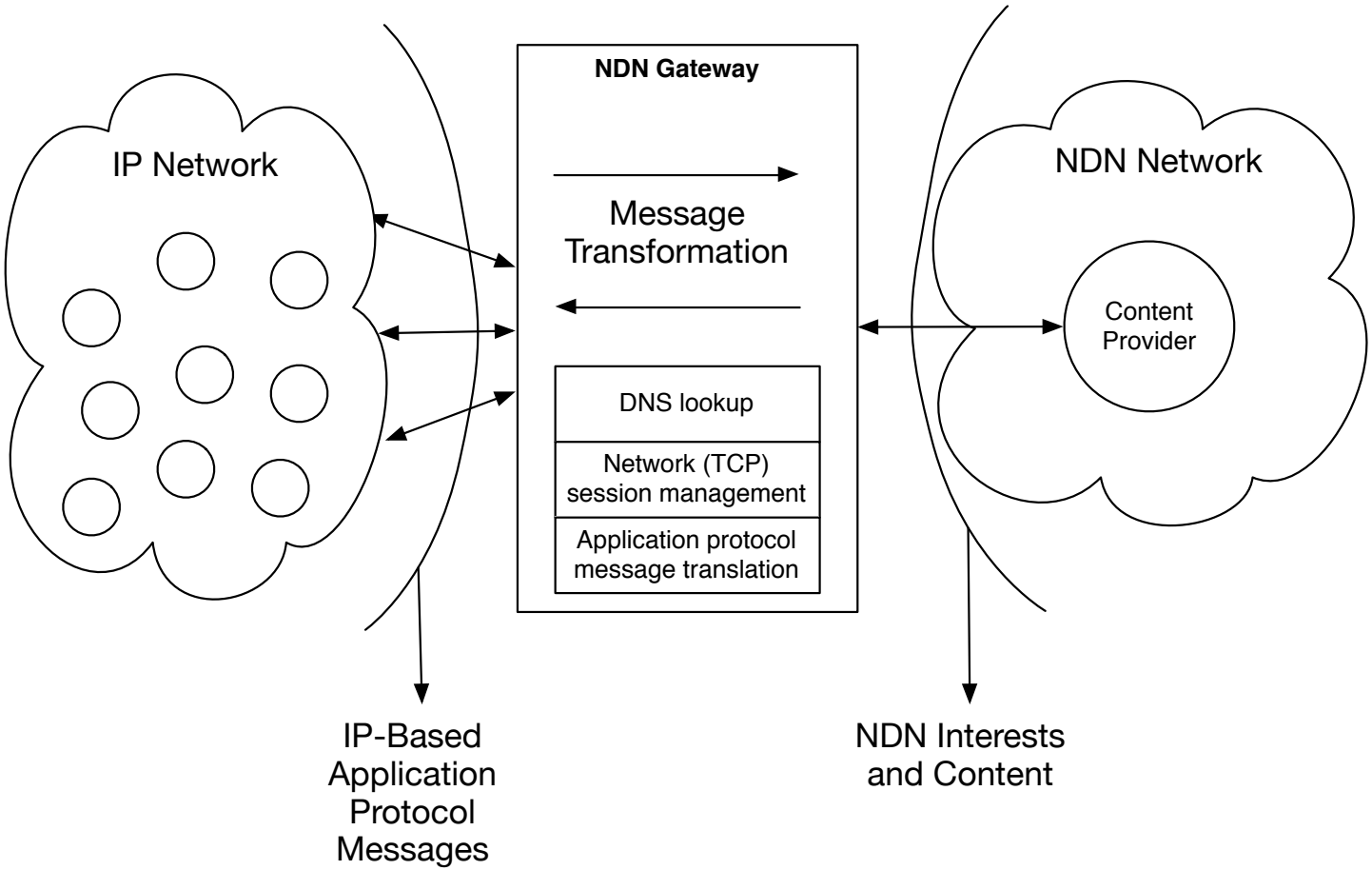
# Motivation for NDN Gateway/Bridge

**Question**: If adopted, how will NDN/CCN be deployed?

1. "Turn off" the Internet, swap in new hardware, and then flip the switch again
   - Bad idea...
2. Incrementally "roll out" NDN hardware and slowly make it interoperable with existing IP network
   - How to enable NDN-based applications to communicate with IP-based applications (and vice versa)?
   - ...and how to do this without re-writing the transport/network layer of IP-based applications to use CCNx (i.e., implement CCN functionality on top of IP)?

**Answer**: Use a NDN-network edge gateways to hide the details of NDN/IP communication mechanics and translate IP messages to compliant NDN interests (and vice versa), and use NDN-network edge bridges to connect isolated NDN "islands".

# Gateway Semantic Translations



# IP-to-NDN Traffic

- HTTP GET requests issued to get content with a similar name
  - e.g., `GET X.X.X.X:80/ndn/ccnx/name/of/content`
  - The request path is mapped to the outgoing interest name
- TCP connections established to stream data to NDN producers
  - Socket connection between IP-based client and gateway established, NDN producer name first sent, and then all remaining data is streamed
  - The gateway partitions data from the socket and packs it into an interest for the desired NDN producer

# NDN-to-IP Traffic

- Interests are encoded according to a special grammar to enable the gateway to parse interests and issue them using the appropriate IP-based protocol

<ip-interest>: ’/. . . /ip/’<protocol>.

<protocol>: ’http/’<http-cmd>[’/’<http-path>] | ’tcp/’<tcp-ident>’/’<uri-encoded-string>.

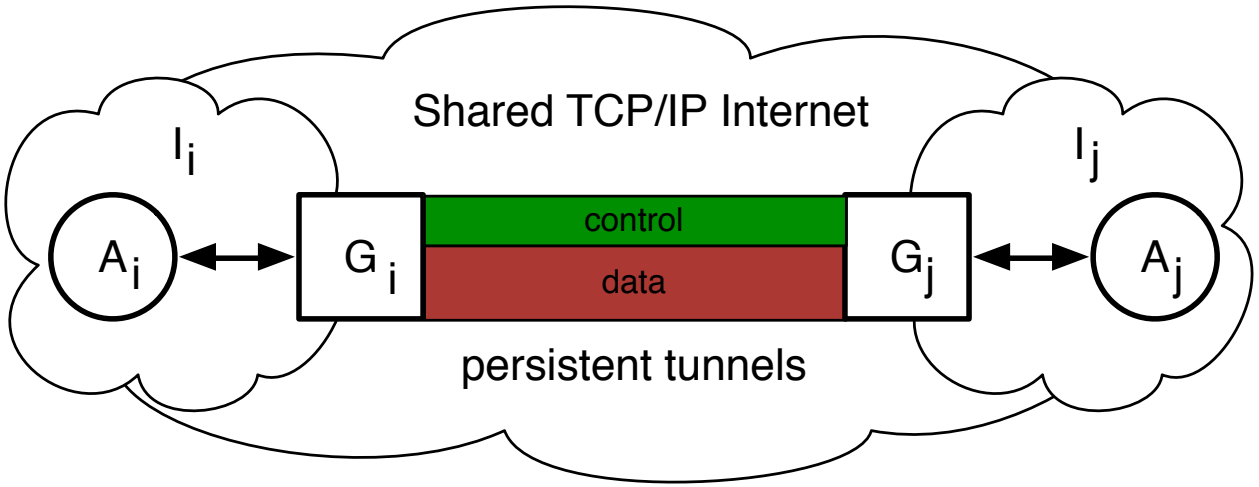
<http-cmd>: ’GET’ | ’PUT’ | ’POST’ | ’DELETE’.

<http-path>: <uri> | <ip-address>[port][’/’<uri-encoded-string>]

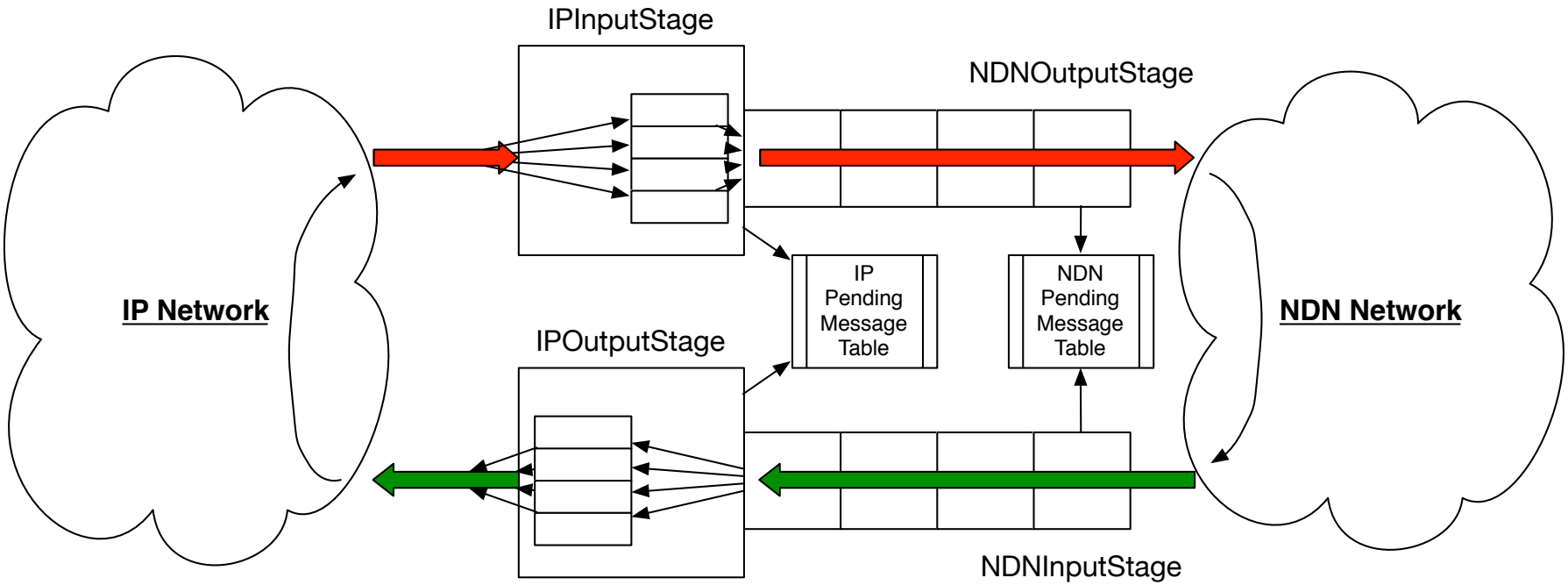
<tcp-ident>: <SHA256-hash>’/’<nonce>’/’<public-key>.

<tcp-param>: <uri-encoded-string>.

# Bridging NDN Islands



# Pipeline-Based Load Balancing Design



# Performance Measurement: Experiments and Metrics

We will assess the design and implementation performance with the following experiments:

- Bidirectional “application-layer” and “transport-layer” communication across the gateway
- Unidirectional messages sent from IP and NDN hosts

We will collect the following metrics and model them as a function of the number of gateways $n$ and estimated clients $m$:

- Unidirectional message translation overhead
- Unidirectional message trip time
- Bridge mode message latency (RTT)
- Bridge mode symmetric key establishment overhead time