UCI University of California, Irvine

Namespace Tunnels in Content-Centric Networks

Ivan O. Nunes, Gene Tsudik and Christopher Wood
University of California, Irvine
{ivanoliv, gene.tsudik, woodc1}@uci.edu

Agenda

- CCN Overview
- VPNs
- CCVPN: VPNs for CCNs
  - Design
  - Security
  - Implementation & Evaluation
- Final Remarks

CCN Overview

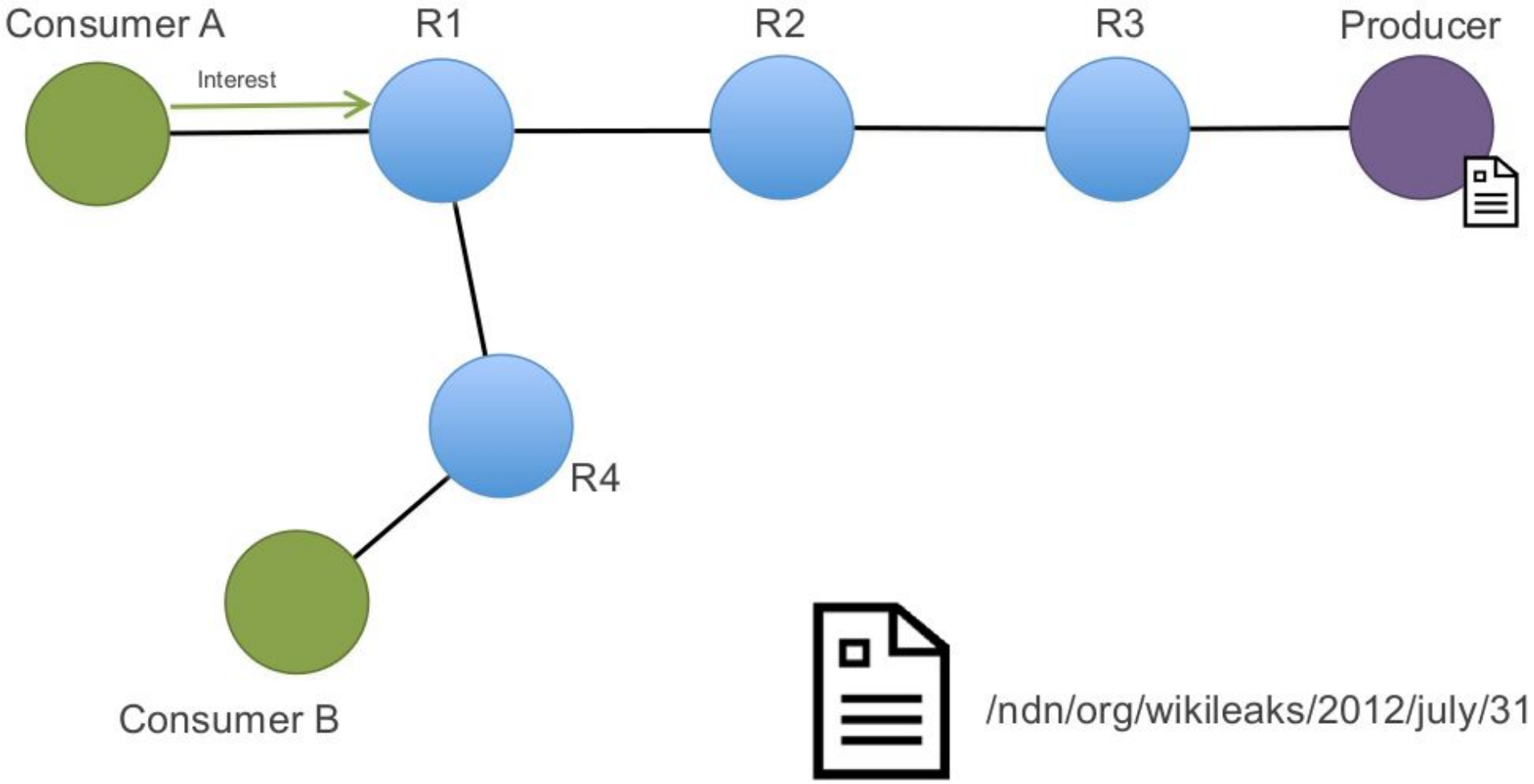
Content-Centric Networking:

- Named data, instead of host addresses
- Decouples Content from its location
- Allows in-network caching: potentially better networks utilization, lower latency...

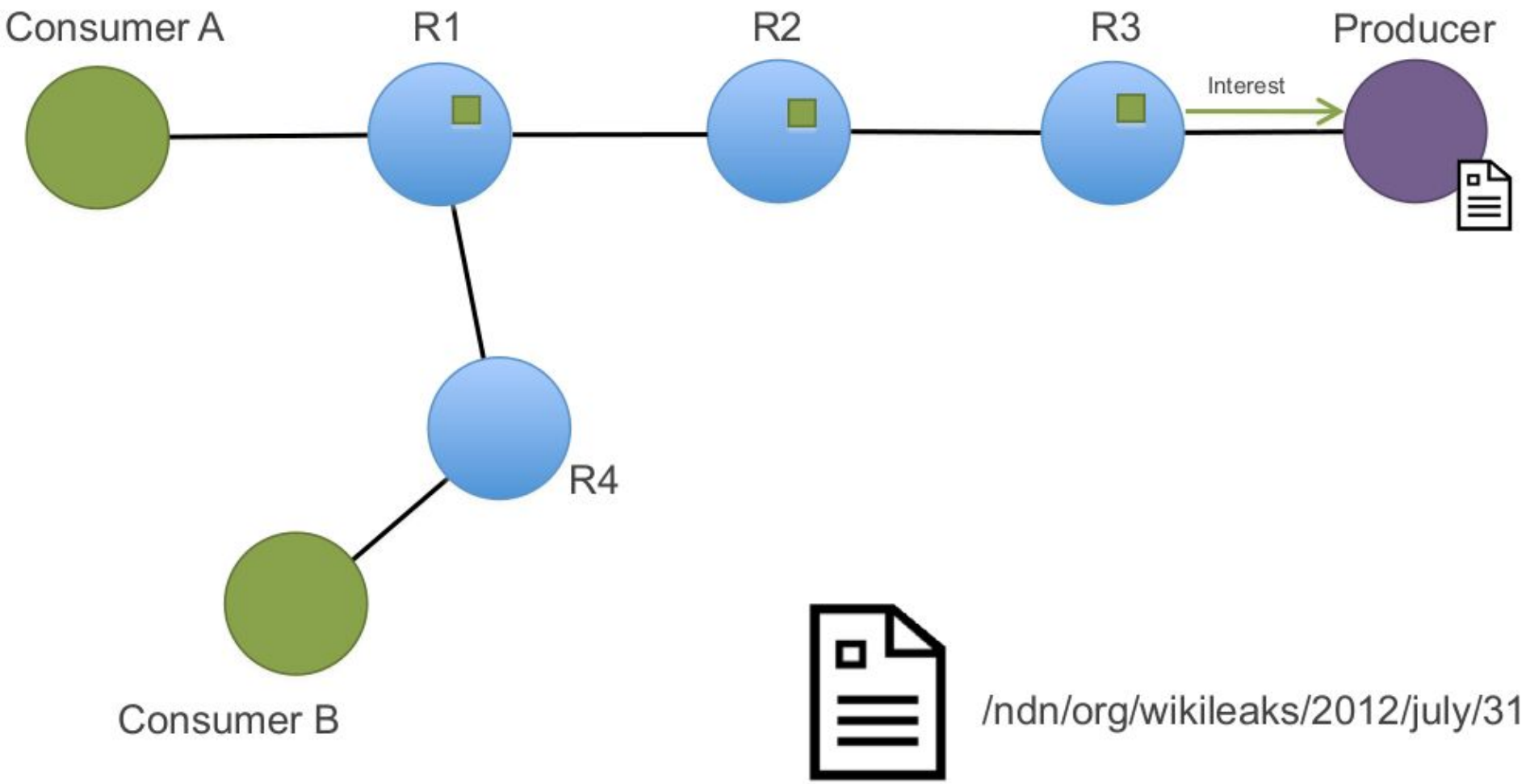
Content-Centric Networking:

- Network entities:
  - ***Producers:*** generate and publish content under unique names
  - ***Consumers:*** issue “interests” for contents containing such contents names
  - ***Routers:*** forward interests and contents
    - May cache content

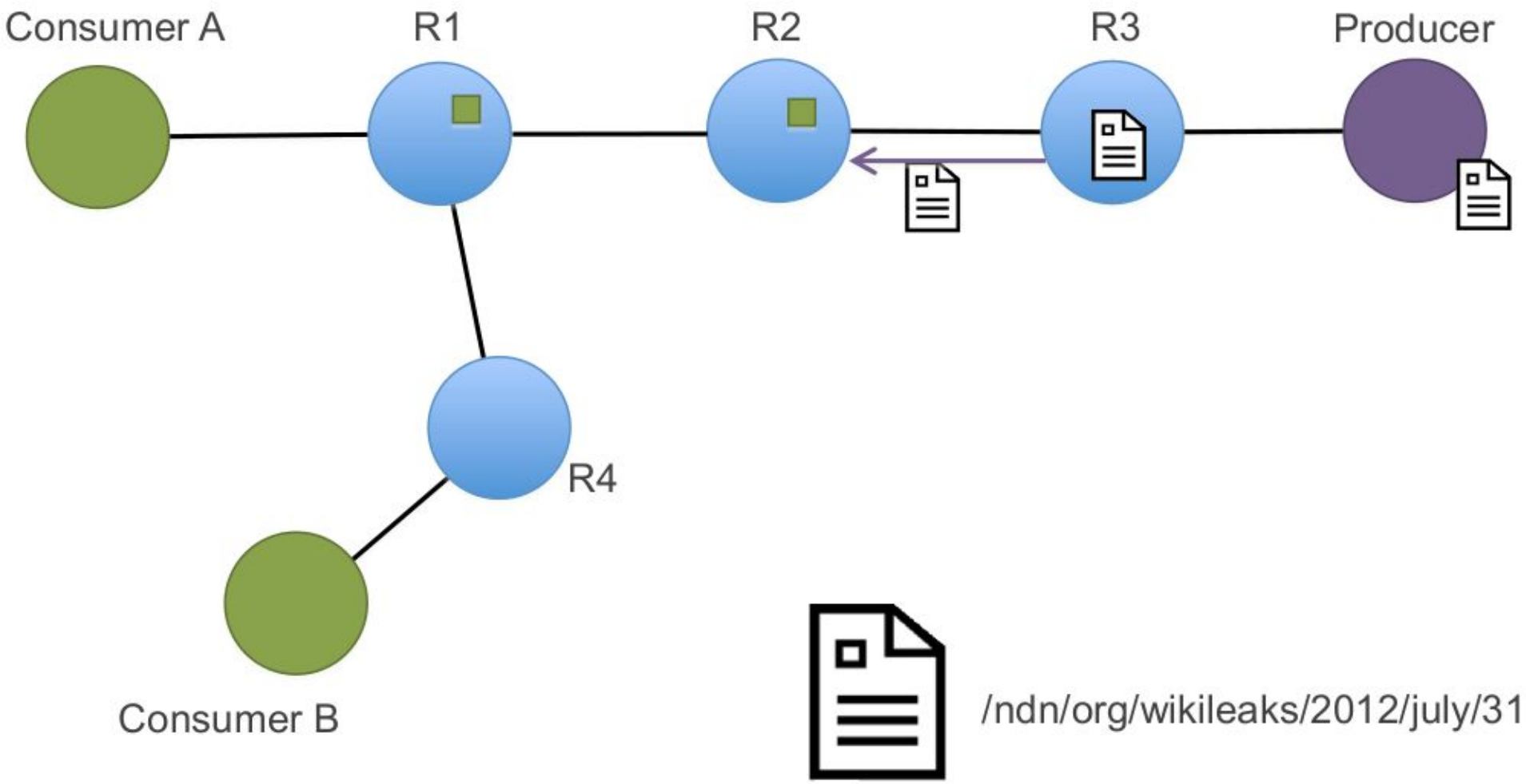
Content-Centric Networking:



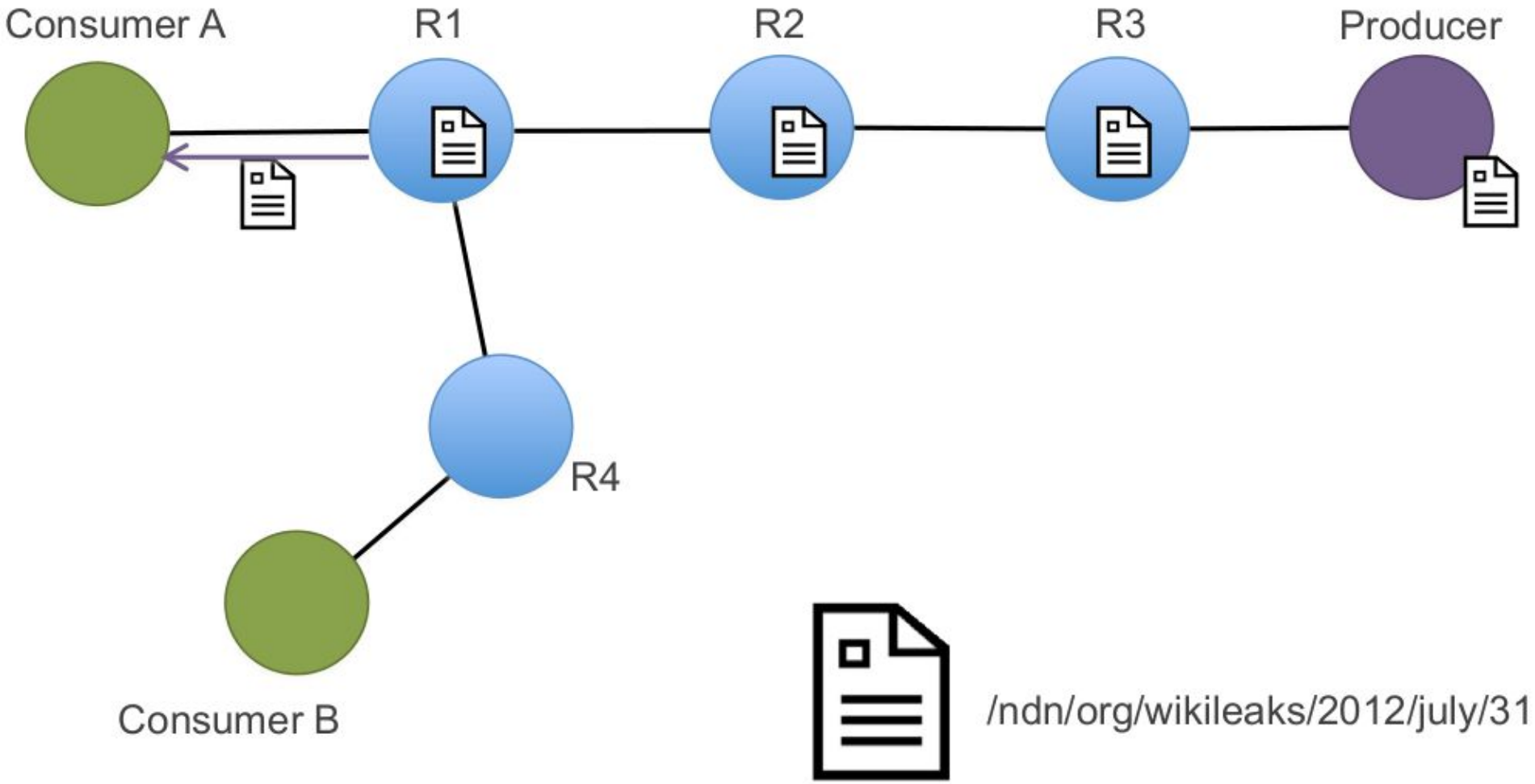
Content-Centric Networking:



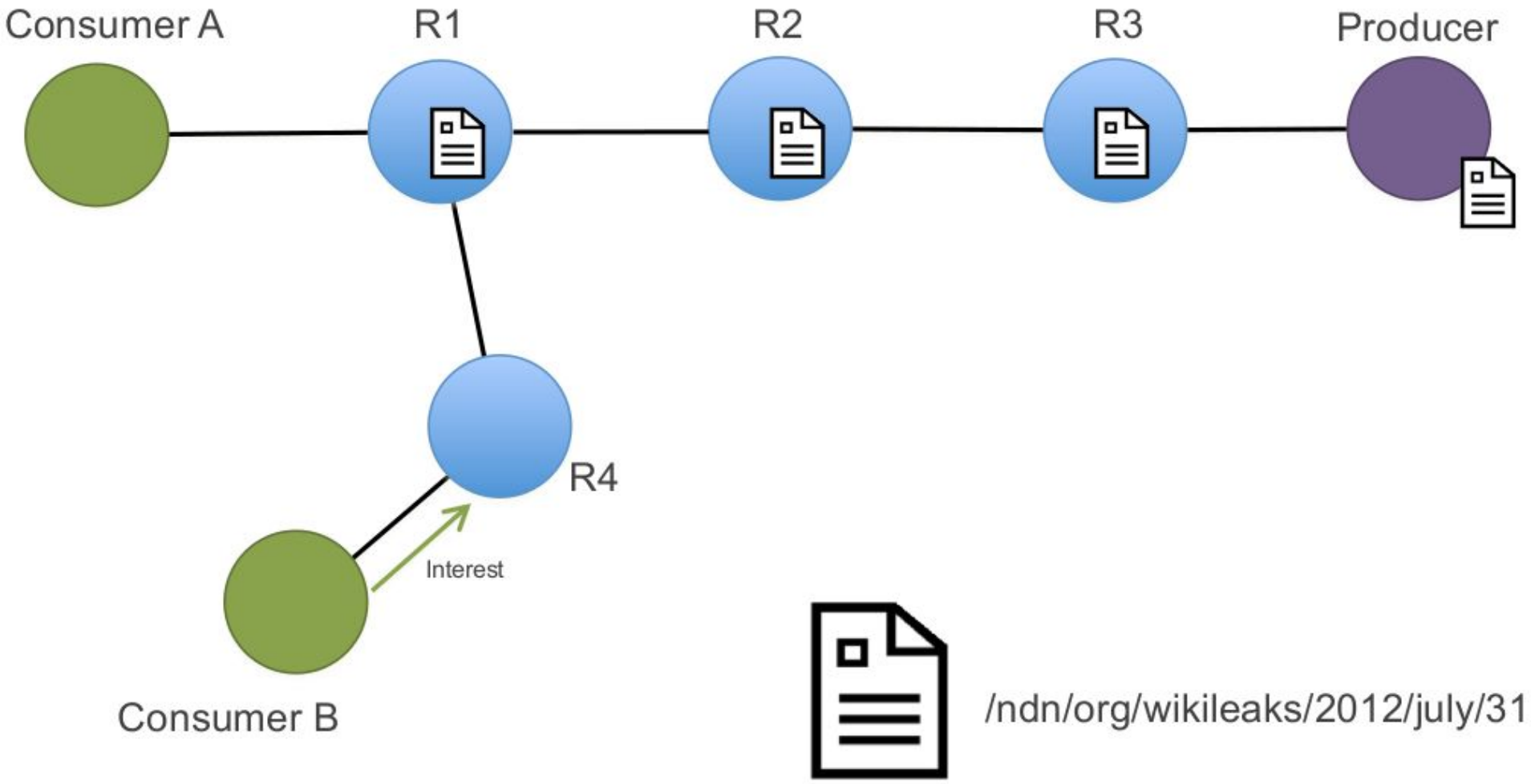
Content-Centric Networking:



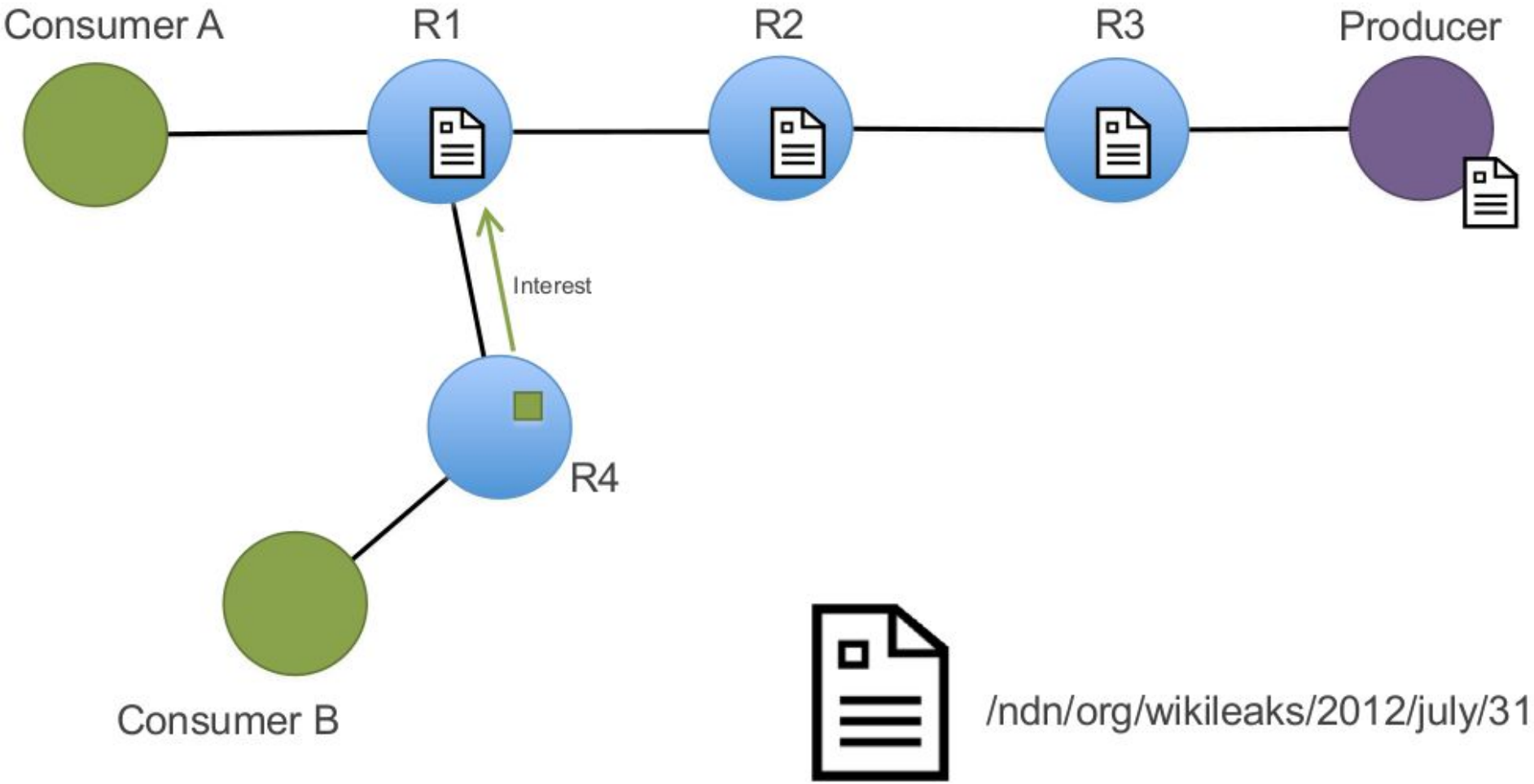
Content-Centric Networking:



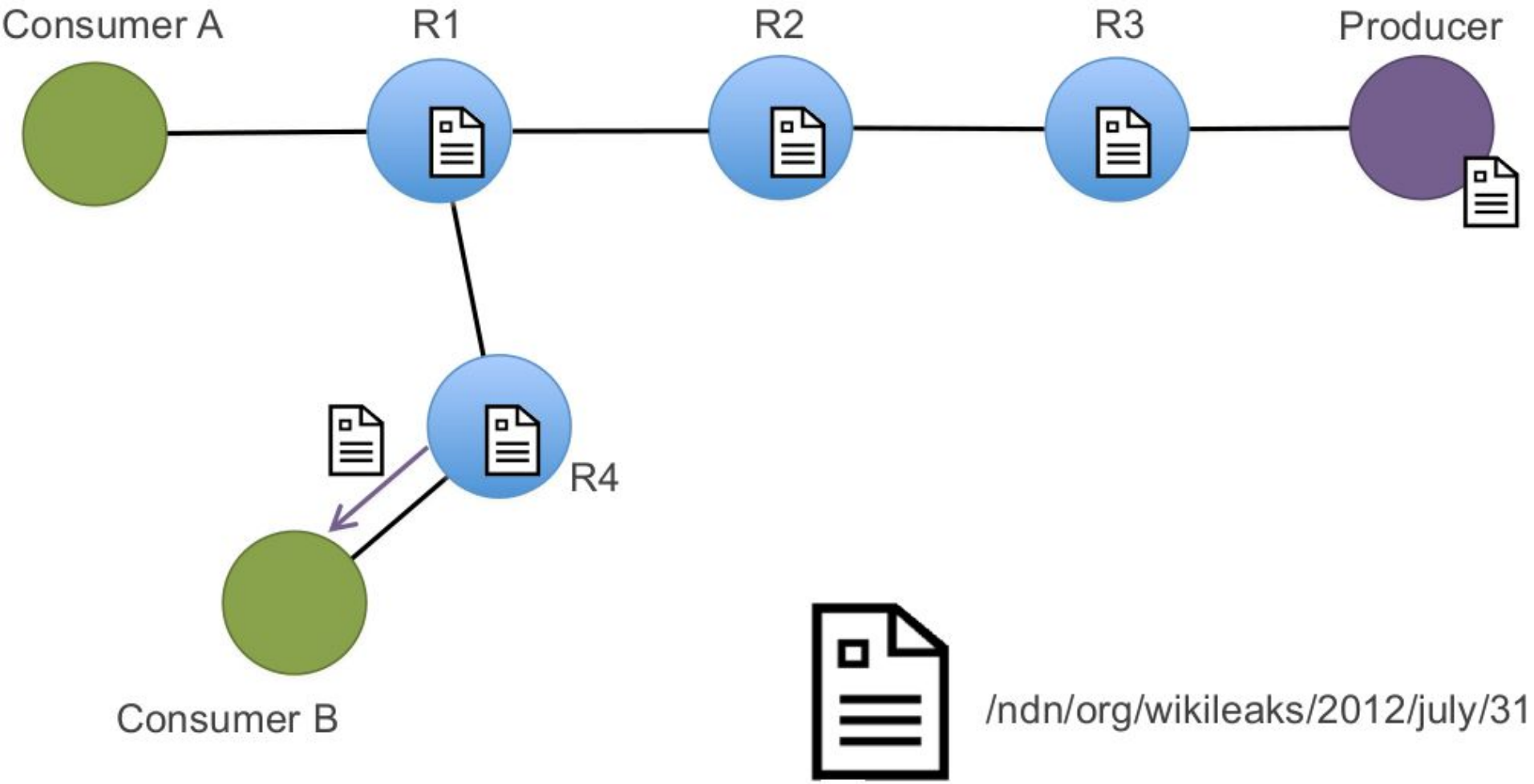
Content-Centric Networking:



Content-Centric Networking:



Content-Centric Networking:



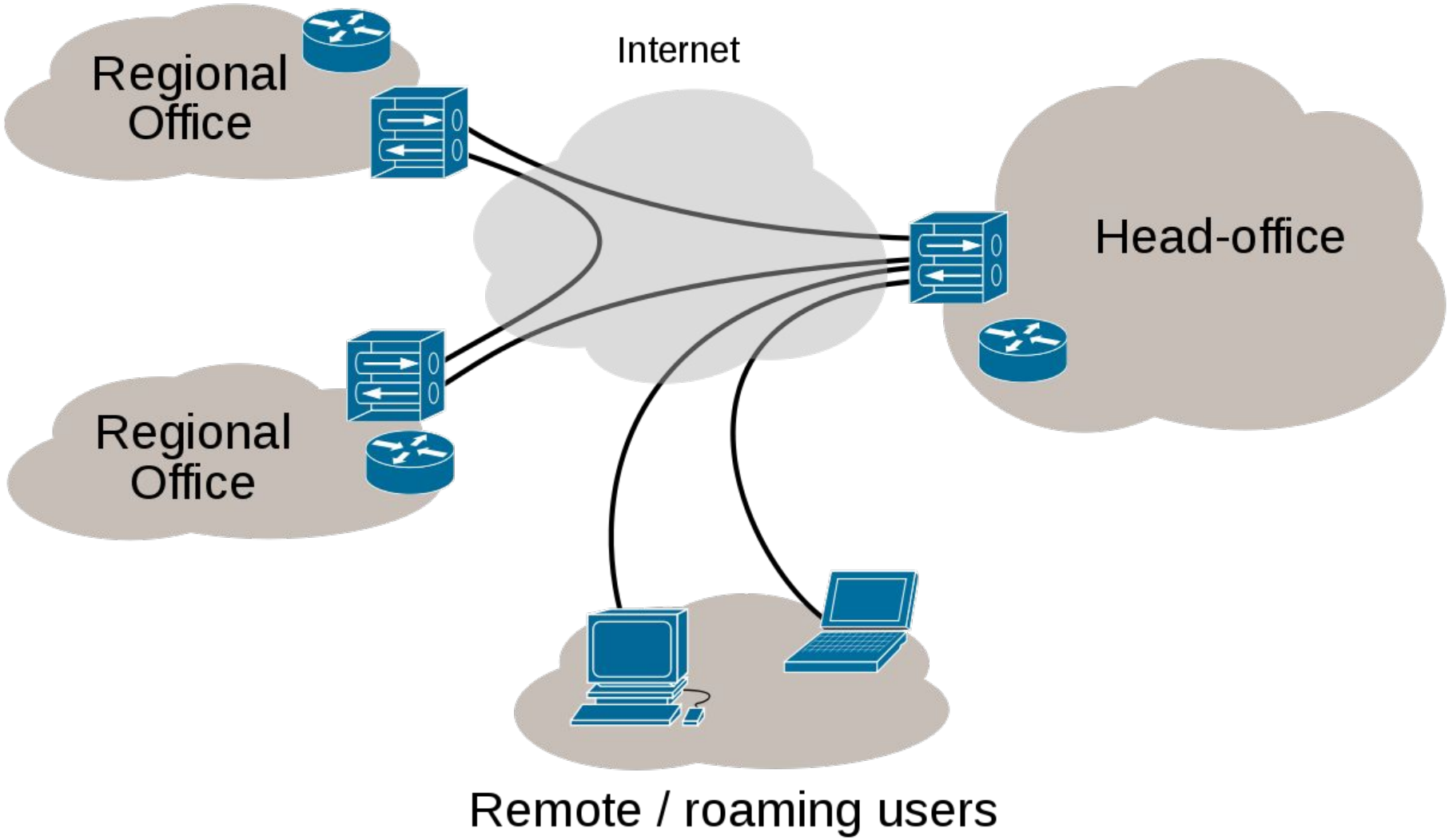
Content-Centric Networking: Overview

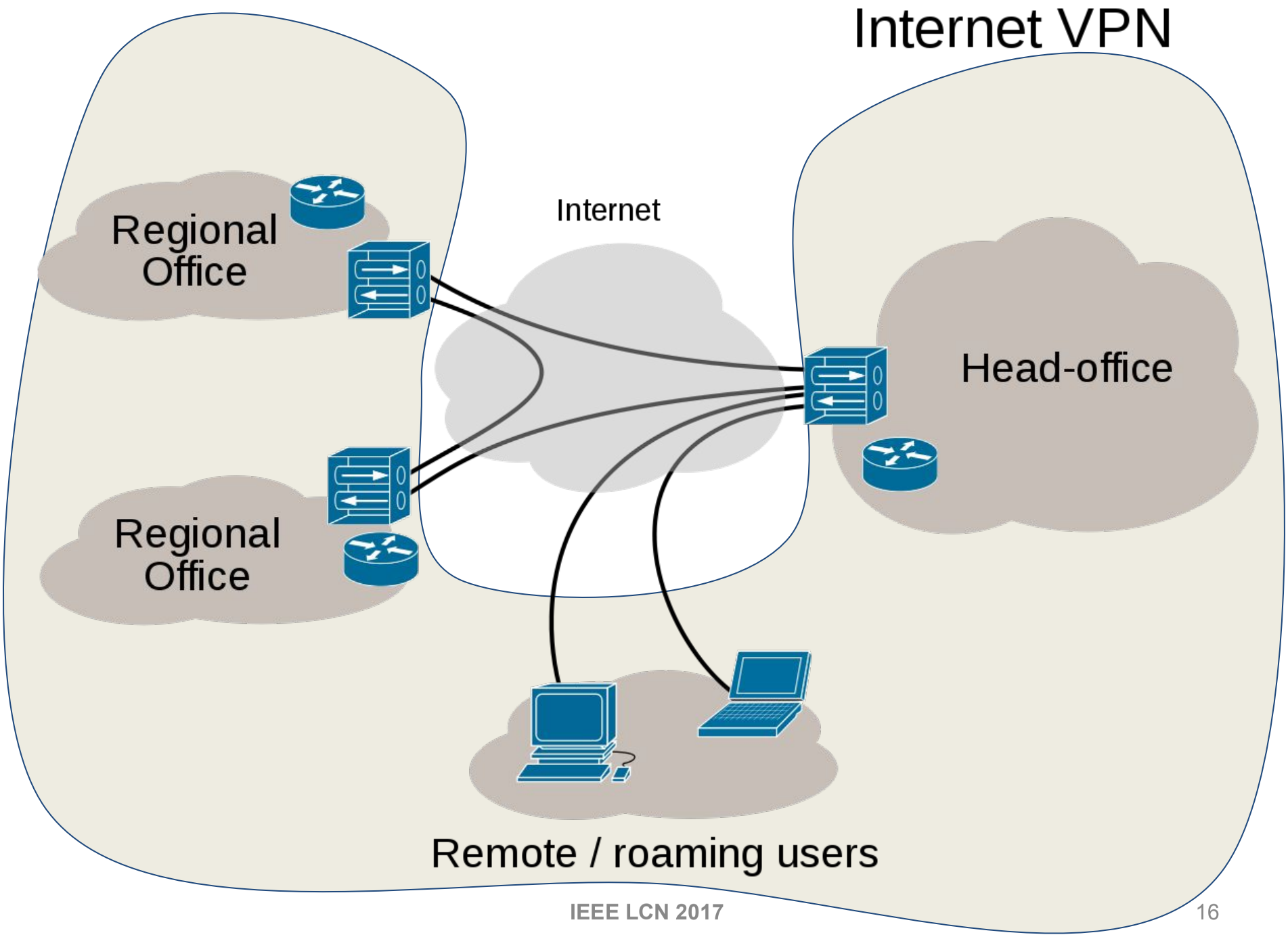
Routing:

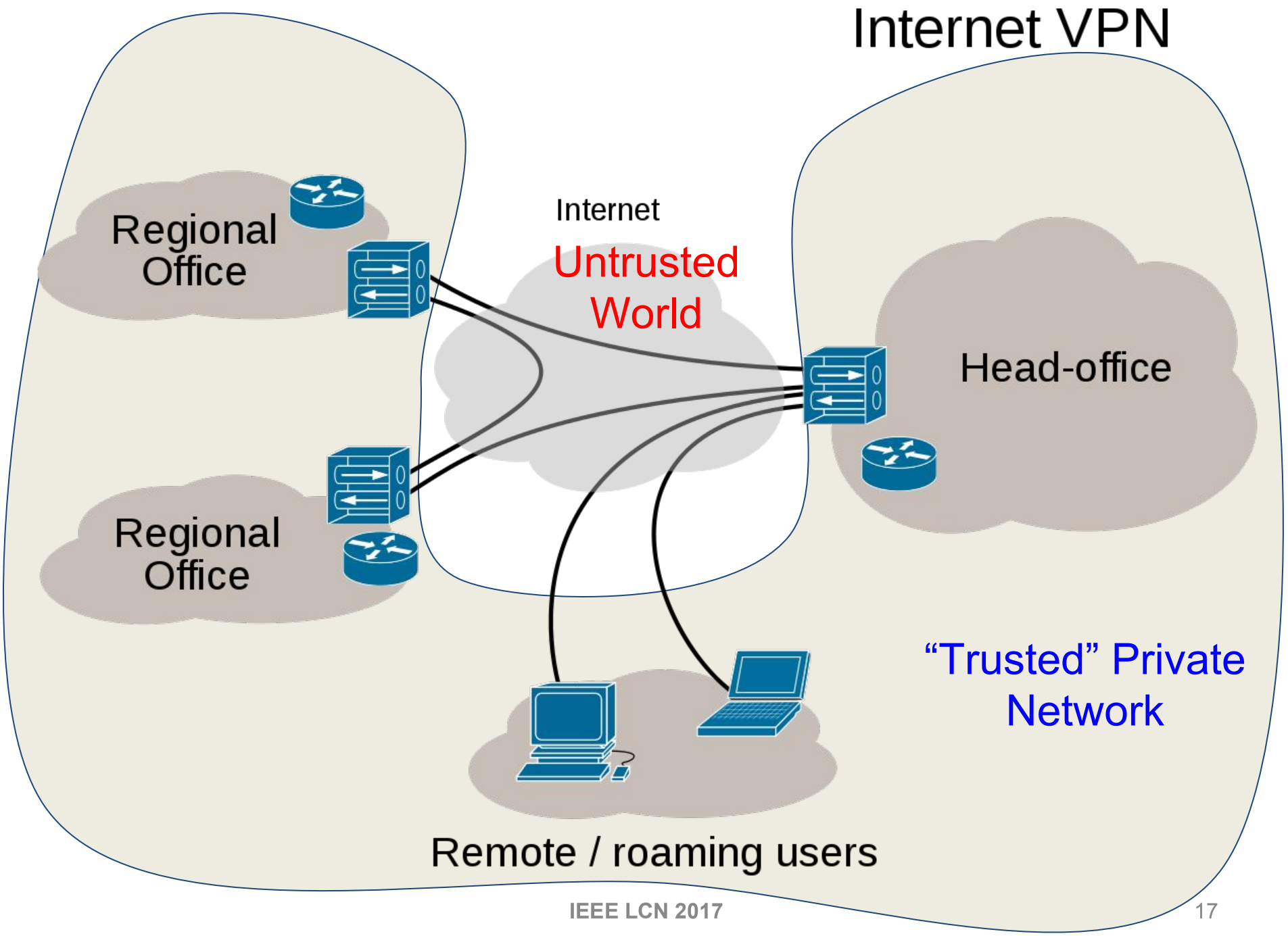
- ***Pending Interest Table (PIT)*:**
  - Table of pending interests and corresponding incoming interfaces
  - Used to route the content back to the requesting consumer
- ***Forwarding Interest Base (FIB)*:**
  - Table of name prefixes and corresponding outgoing interfaces
  - Used to route interests towards content producers (Longest Prefix Match of names)

Virtual Private Networks (VPNs)

Internet VPN




Internet VPN
Internet
Regional Office
Regional Office
Head-office
Remote / roaming users

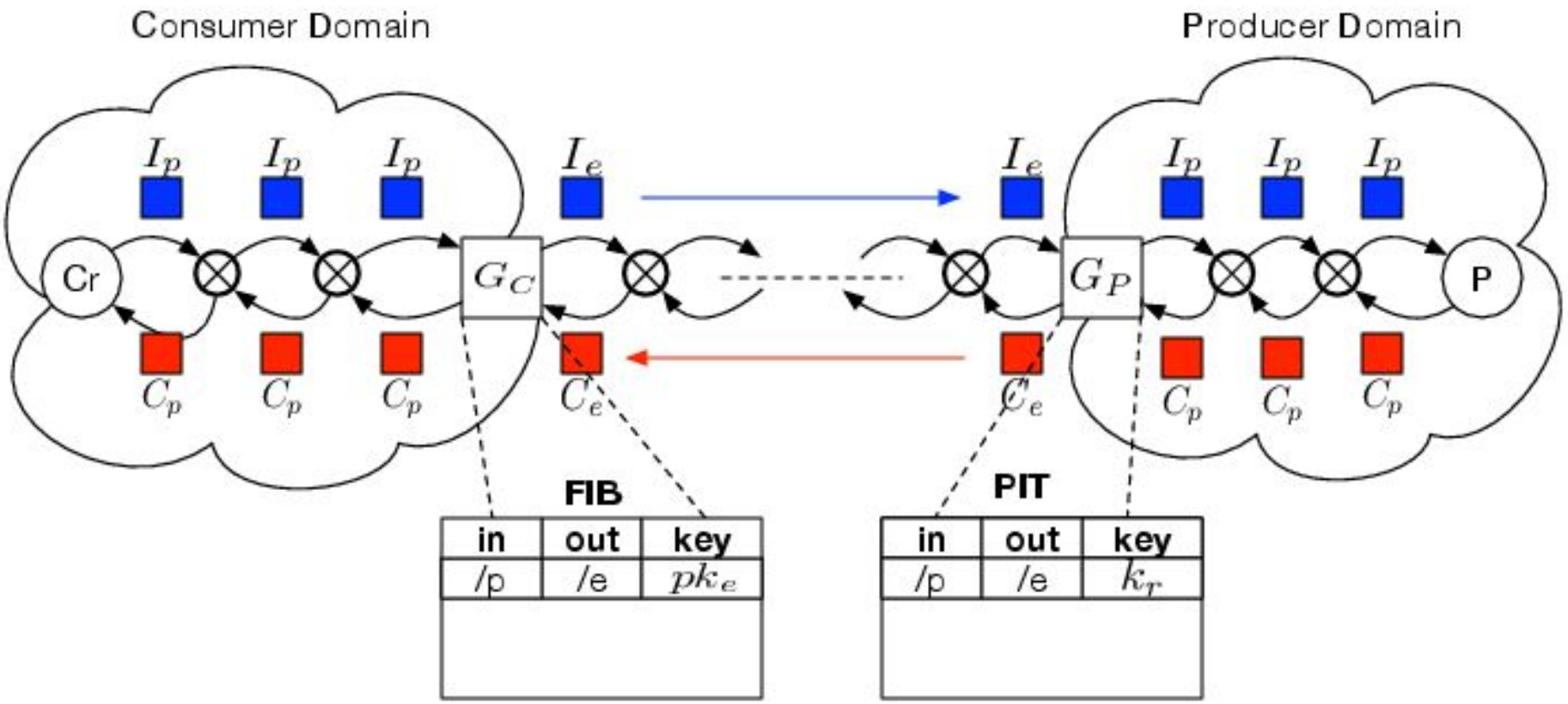

Internet VPN
Regional Office
Regional Office
Internet
Untrusted World
Head-office
"Trusted" Private Network
Remote / roaming users

Virtual Private Network

- Support secure communication across the Internet
- Allows end-points to send/receive data as if they were connected within the same physical private network.

CCVPN

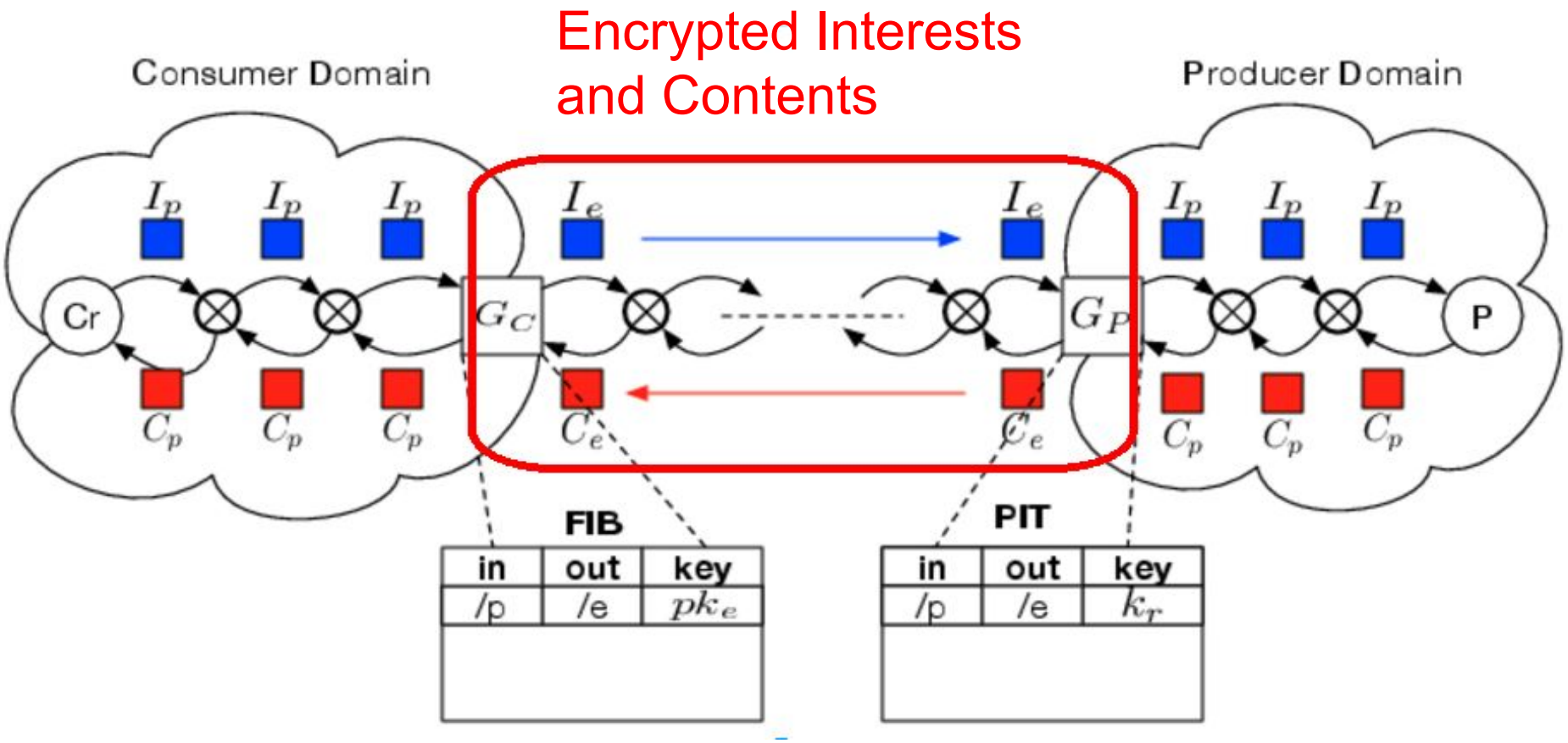
Big Picture


Consumer Domain
Producer Domain
I_p
I_e
Cr
G_C
G_P
P
C_p
C_e
FIB
in
out
key
/p
/e
pk_e
PIT
k_r

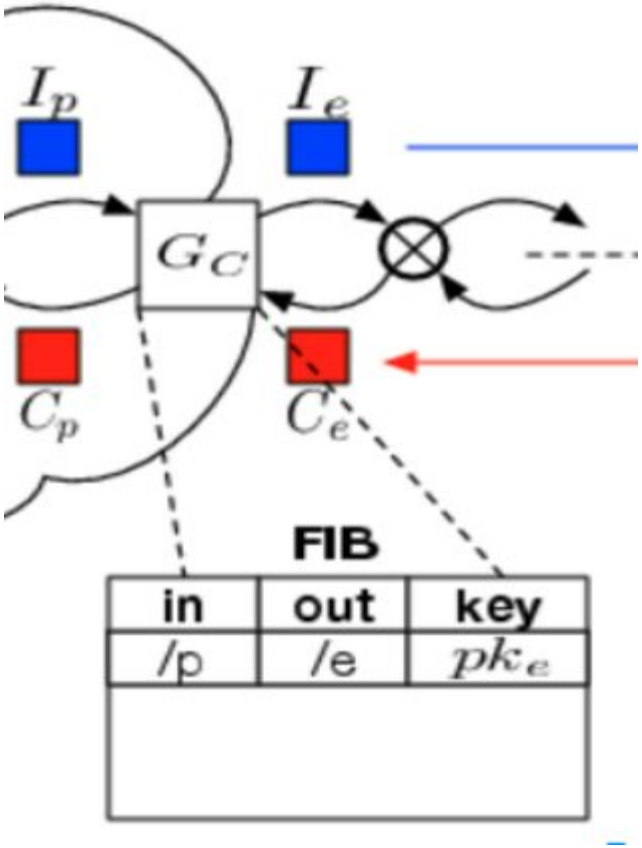
Design

- Parties:
  - ***Consumer Side GW*** ***(Gc)*** **:**
    - Encapsulates outgoing consumer-issued interests
    - Decapsulates incoming content
  - ***Producer Side GW*** ***(Gp)*** **:**
    - Decapsulates incoming encapsulated interests
    - Encapsulates outgoing content replies.

The Big Picture

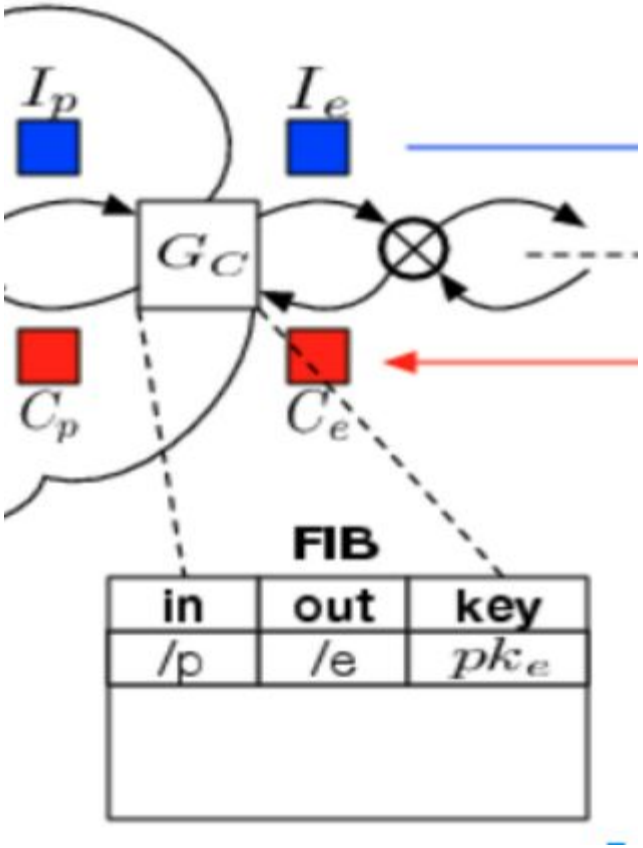


Design: *Gc* Interest Encapsulation



Assumed to i) know **Gp**'s public-key or
ii) share a symm. key with **Gp**

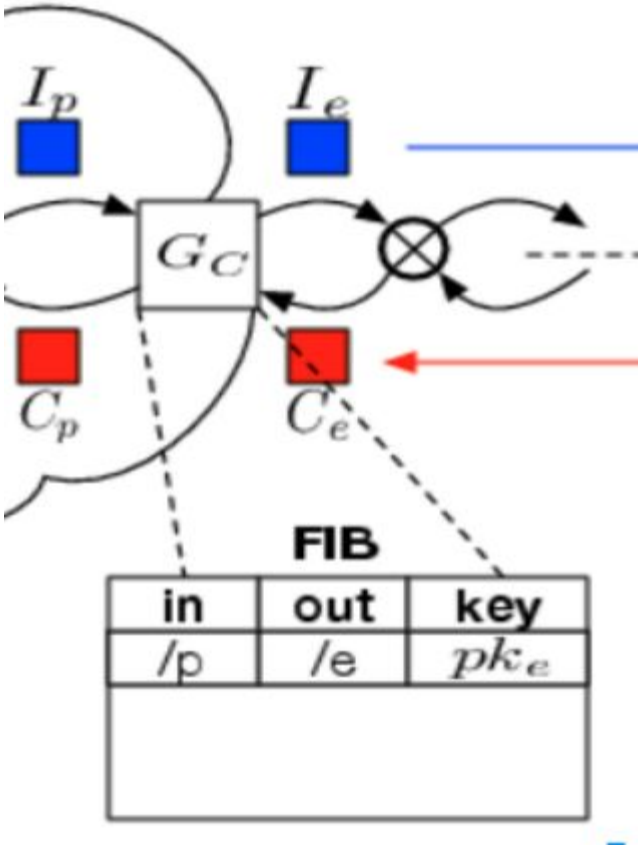
Design: **Gc** Interest Encapsulation



Assumed to i) know **Gp**'s public-key or ii) share a symm. key with **Gp**

1. Generates a fresh Symm. Key **K** (used later on) and encrypts both the Consumer-issued Interest (**Ip**) and **K** with **Gp**'s key

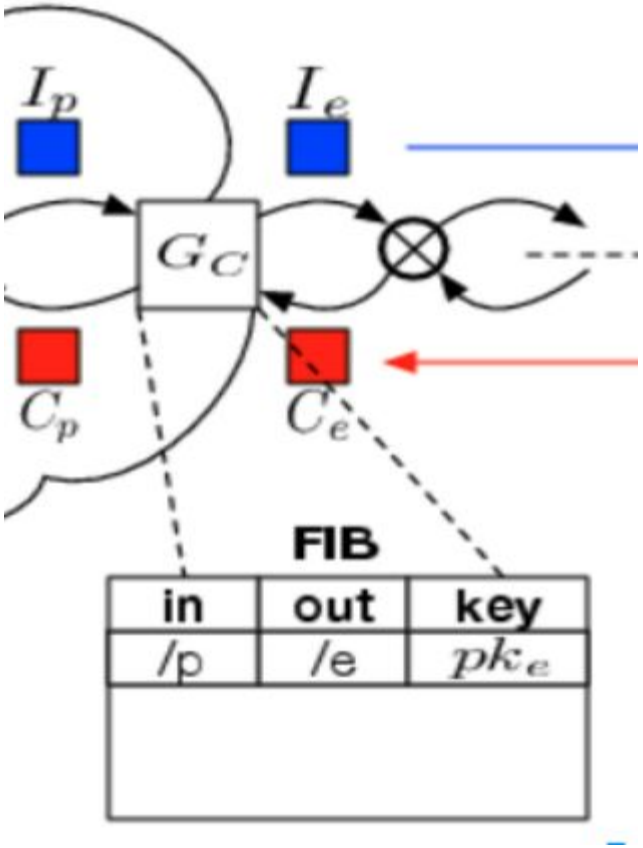
Design: *Gc* Interest Encapsulation



Assumed to i) know **Gp**'s public-key or ii) share a symm. key with **Gp**

1. Generates a fresh Symm. Key **`K`** (used later on) and encrypts both the Consumer-issued Interest (**`Ip`**) and **`K`** with **Gp**'s key
2. Issues a new Interest (**`Ie`**) with **Gp**'s namespace as prefix and encrypted **`Enc(Ip||K)`** as payload

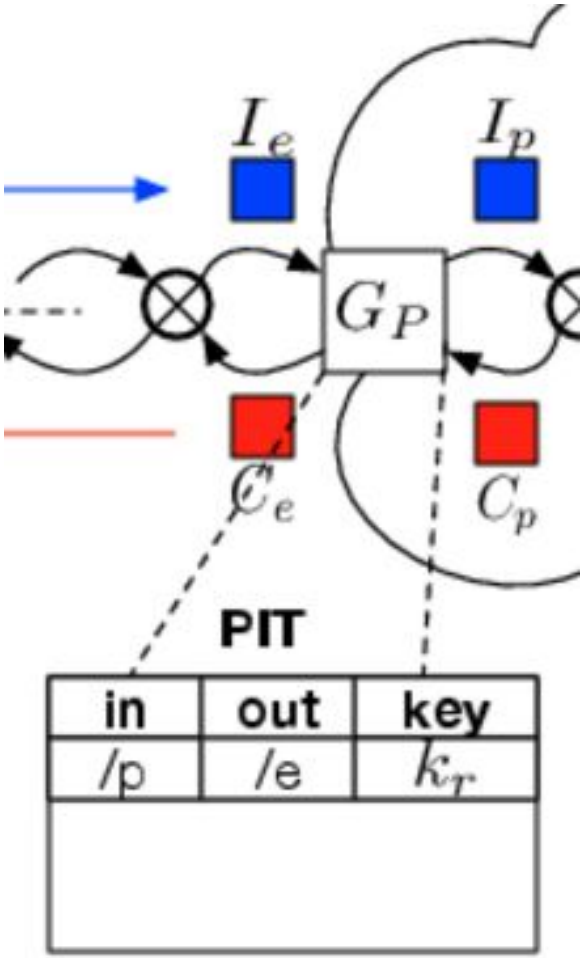
Design: *Gc* Interest Encapsulation



Assumed to i) know **Gp**'s public-key or ii) share a symm. key with **Gp**

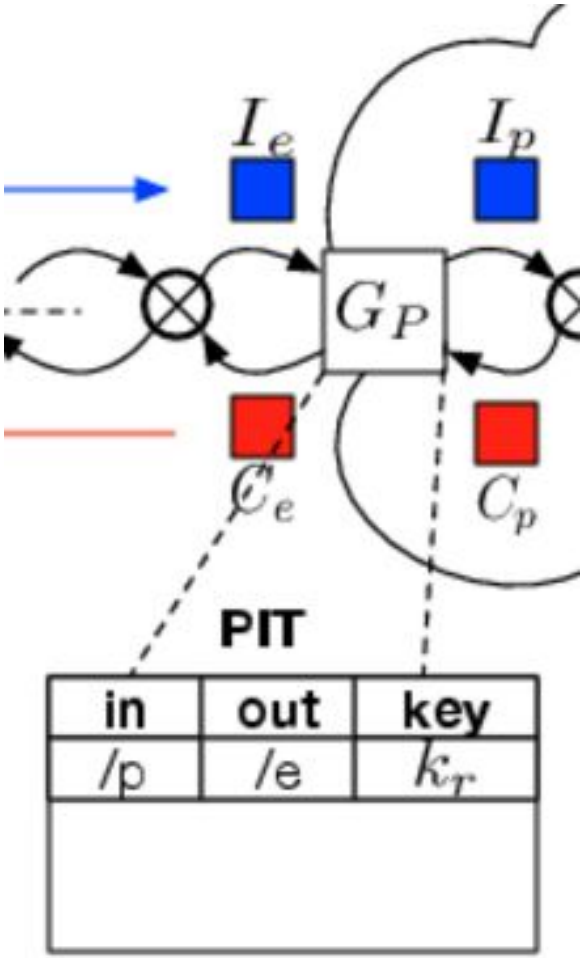
1. Generates a fresh Symm. Key **`K`** (used later on) and encrypts both the Consumer-issued Interest (**`Ip`**) and **`K`** with **Gp**'s key
2. Issues a new Interest (**`Ie`**) with **Gp**'s namespace as prefix and encrypted **`Enc(Ip||K)`** as payload
3. Store **`K`** in its PIT entry for **`Ie`**

Design: *Gp* Interest Decapsulation



Upon receiving the encapsulated interest Ie, Gp then:

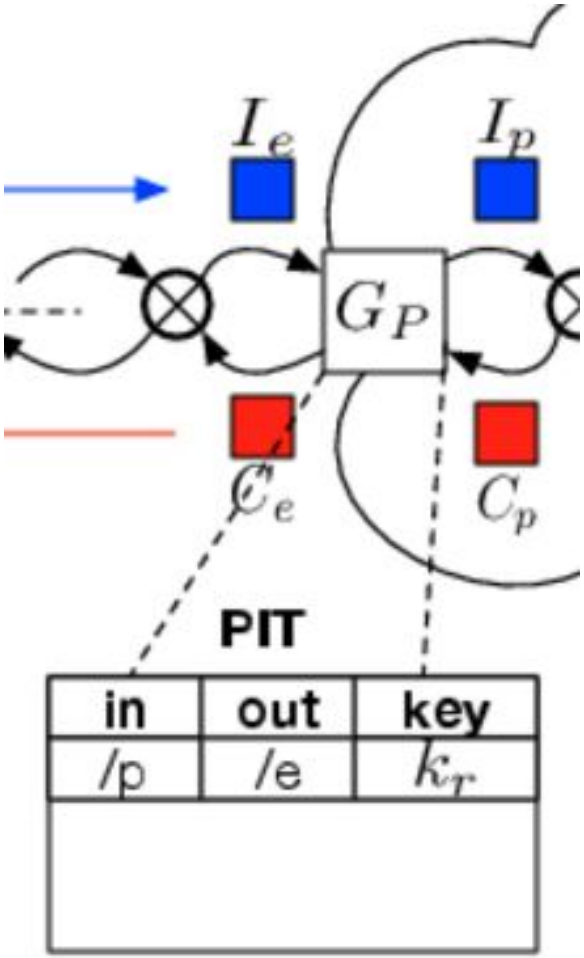
Design: *Gp* Interest Decapsulation



Upon receiving the encapsulated interest Ie, Gp then:

1. Decrypts (using the shared key or it's secret key) **`Ie`** payload retrieving **`Ip`** and **`K`**

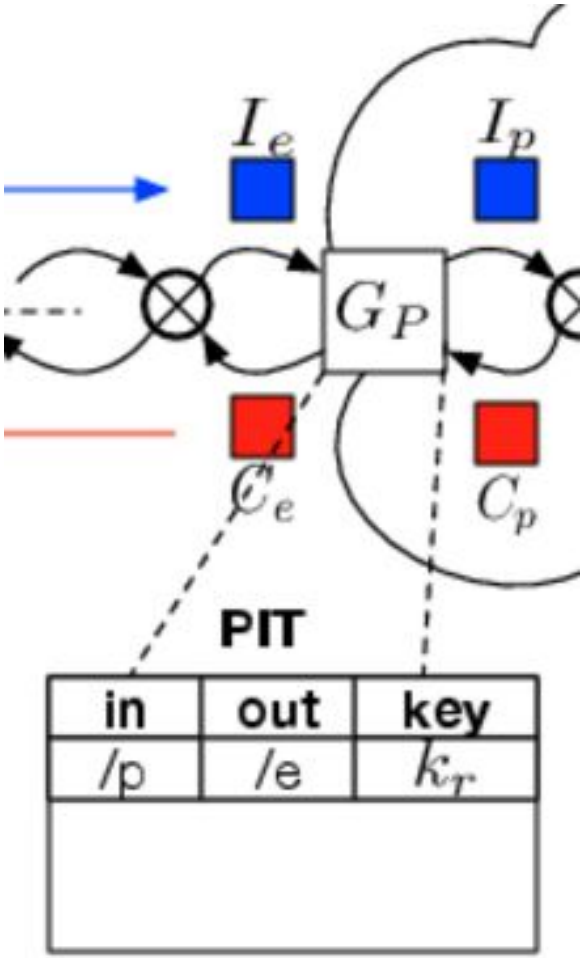
Design: *Gp* Interest Decapsulation



Upon receiving the encapsulated interest Ie, Gp then:

1. Decrypts (using the shared key or it's secret key) **`Ie`** payload retrieving **`Ip`** and **`K`**
2. Store **`K`** in its PIT entry for **`Ip`**

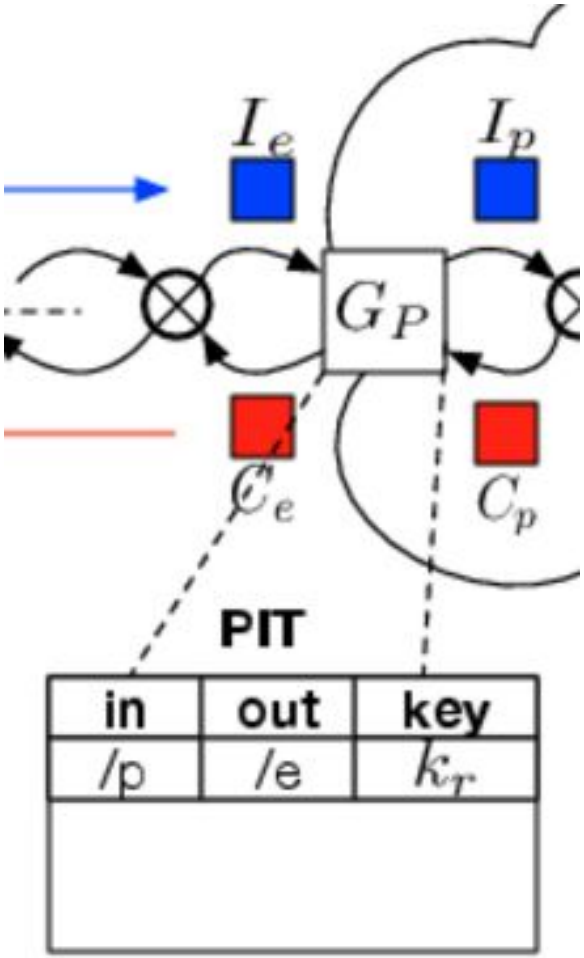
Design: *Gp* Interest Decapsulation



Upon receiving the encapsulated interest Ie, Gp then:

1. Decrypts (using the shared key or it's secret key) **`Ie`** payload retrieving **`Ip`** and **`K`**
2. Store **`K`** in its PIT entry for **`Ip`**
3. Forwards **`Ip`** towards the Producer

Design: *Gp* Interest Decapsulation



Upon receiving the encapsulated interest Ie, Gp then:

1. Decrypts (using the shared key or it's secret key) **`Ie`** payload retrieving **`Ip`** and **`K`**
2. Store **`K`** in its PIT entry for **`Ip`**
3. Forwards **`Ip`** towards the Producer

Notice that now both gateways store the fresh key K in their PITs.

Design: Content Encapsulation and Decapsulation

- Interest forwarding (w/ encapsulation and decapsulation algorithms) causes **`Gc`** and **`Gp`** share the symmetric key **`K`**

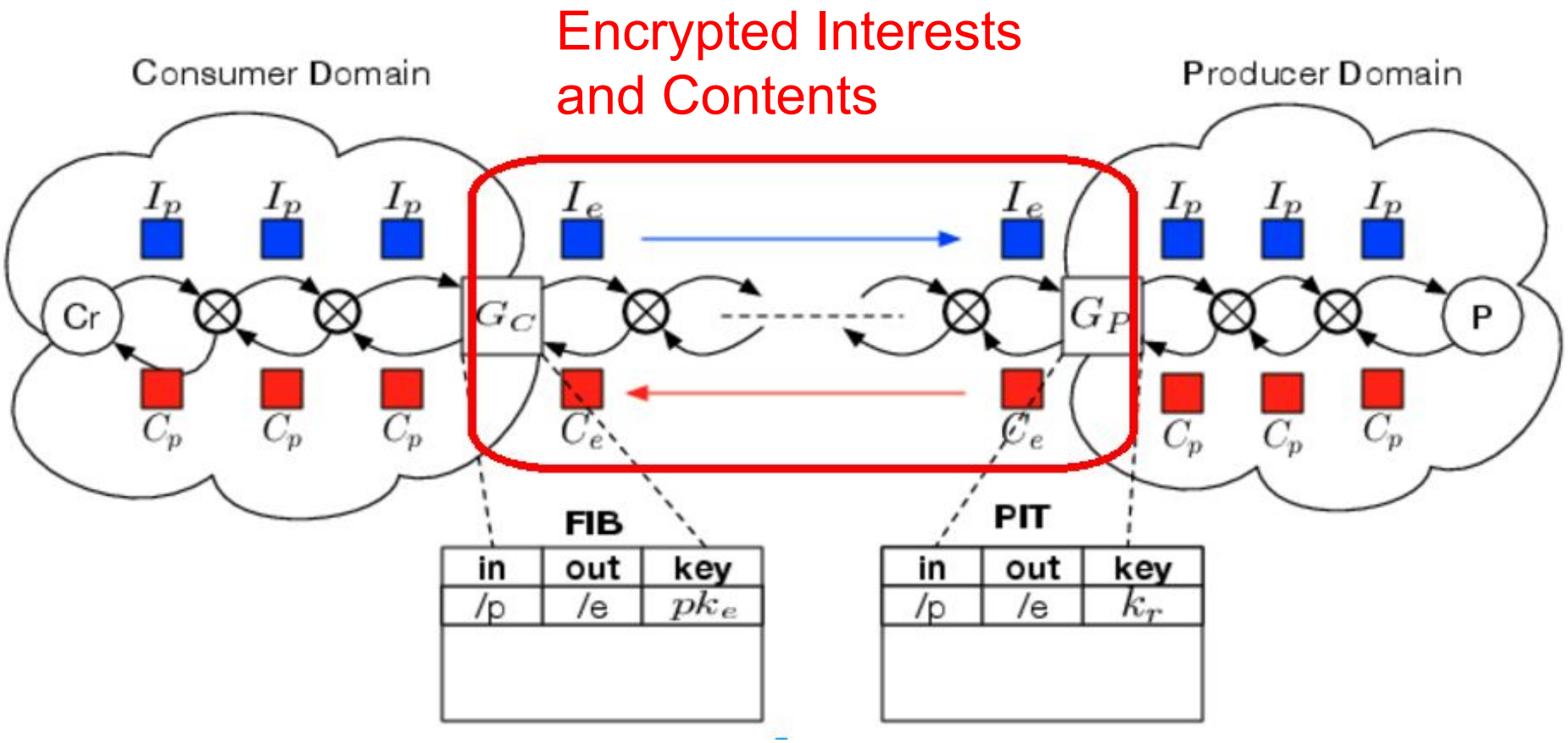
Design: Content Encapsulation and Decapsulation

- Interest forwarding (w/ encapsulation and decapsulation algorithms) causes **`Gc`** and **`Gp`** share the symmetric key **`K`**
- **`K`** is associated to the corresponding Interest names in the **`Gc`** and **`Gp PITs`**

Design: Content Encapsulation and Decapsulation

- Interest forwarding (w/ encapsulation and decapsulation algorithms) causes **`Gc`** and **`Gp`** share the symmetric key **`K`**
- **`K`** is associated to the corresponding Interest names in the **`Gc`** and **`Gp PITs`**
- Upon the arrival of the corresponding Content the gateways fetch **`K`** in their **`PIT`**s and use it to Encrypt/Decrypt, respectively

The Big Picture



CCVPN: Security

- As long as Encryption schemes are secure and network messages are padded, one can not distinguish between different encapsulated contents/interests.

CCVPN: Security

- As long as Encryption schemes are secure and network messages are padded, one can not distinguish between different encapsulated contents/interests.
- **Authenticated (Non-Deterministic) Encryption** is used to ensure confidentiality and integrity (CCA-Security).

CCVPN: Security

- As long as Encryption schemes are secure and network messages are padded, one can not distinguish between different encapsulated contents/interests.
- **Authenticated (Non-Deterministic) Encryption** is used to ensure confidentiality and integrity (CCA-Security).
- The actual Interests and contents are only visible inside the VPN

CCVPN: Implementation & Evaluation

CCVPN: Implementation & Evaluation

- Network service running ***on the gateways***

CCVPN: Implementation & Evaluation

- Network service running ***on the gateways***
- CCNx software stack (C)
- Libsodium Crypto Library (C)

CCVPN: Implementation & Evaluation

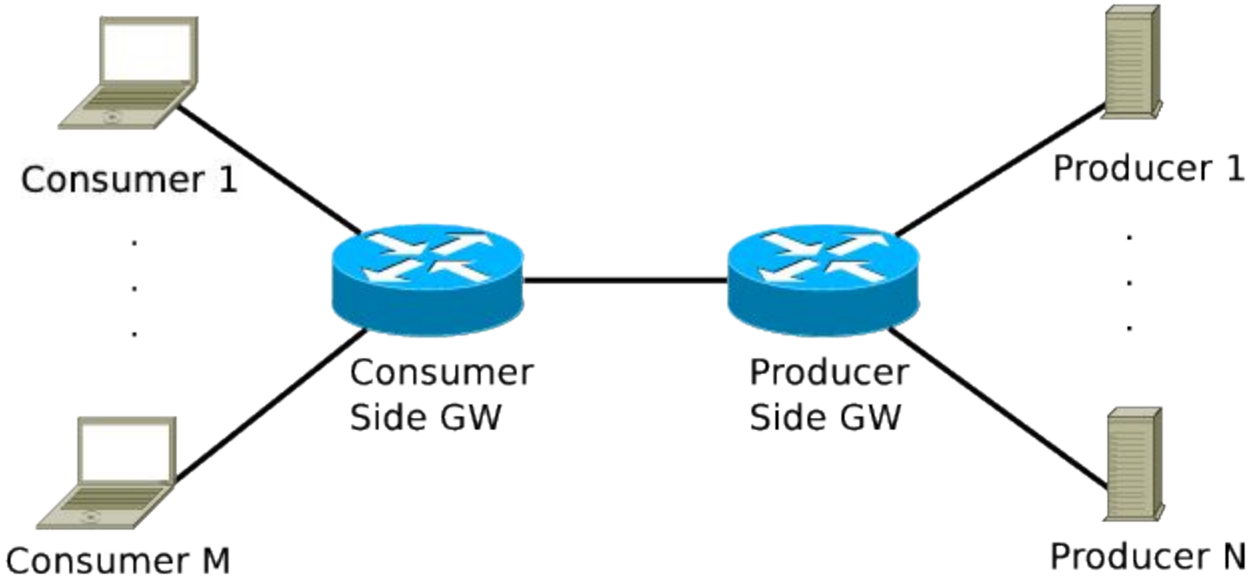
- Network service running ***on the gateways***
- CCNx software stack (c)
- Libsodium Crypto Library (c)
- Intel Core i7-3770 octacore CPU @3.40GHz, with 16GB of RAM, running Linux (Ubuntu 14.04LTS).
- Gateways as high priority processes running in a single core

CCVPN: Implementation & Evaluation

- Network service running ***on the gateways***
- CCNx software stack (C)
- Libsodium Crypto Library (C)
- Intel Core i7-3770 octacore CPU @3.40GHz, with 16GB of RAM, running Linux (Ubuntu 14.04LTS).
- Gateways as high priority processes running in a single core
- Content payload sizes set to 10 kilobytes.
- **Interests always different => worst case performance**

CCVPN: Evaluation

Testbed Network:



CCVPN: Evaluation

- Metrics:
  - Throughput (Mbps)
  - Avg. RTT (seconds)

CCVPN: Evaluation

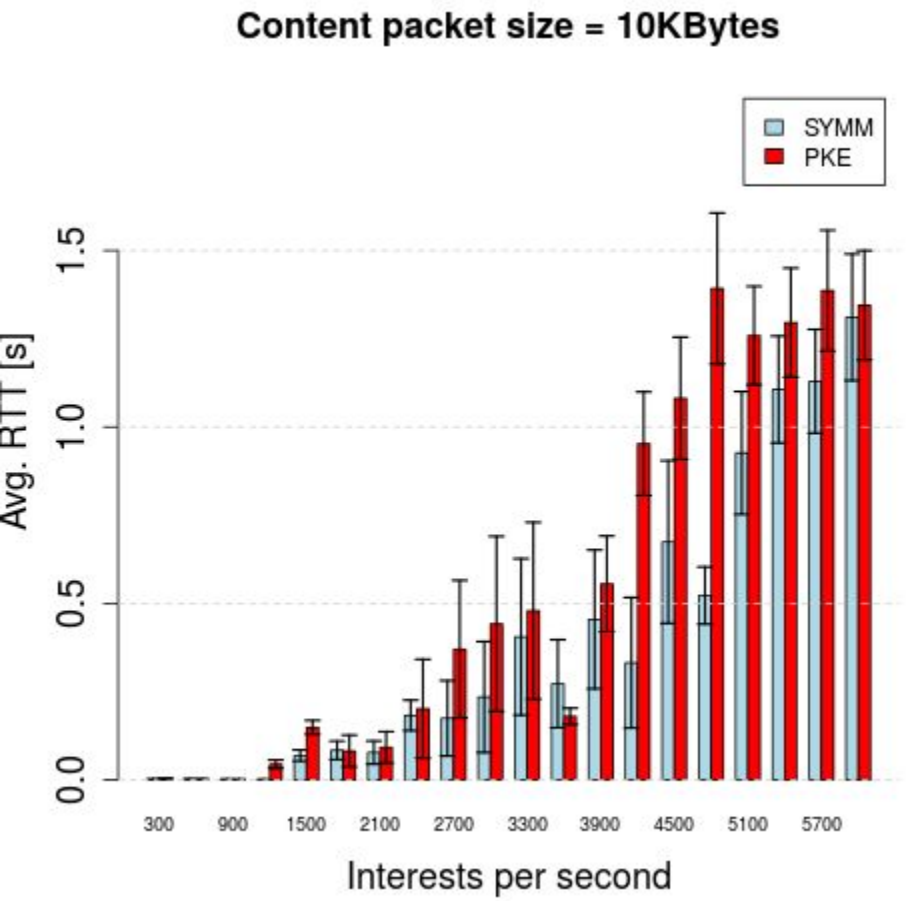
- Metrics:
  - Throughput (Mbps)
  - Avg. RTT (seconds)
- Experiments:
  - 1 consumer vs. 1 producer (w/ increasing Interest issuance rate)
  - Multiple consumers vs. 1 producer
  - Multiple consumers and producers

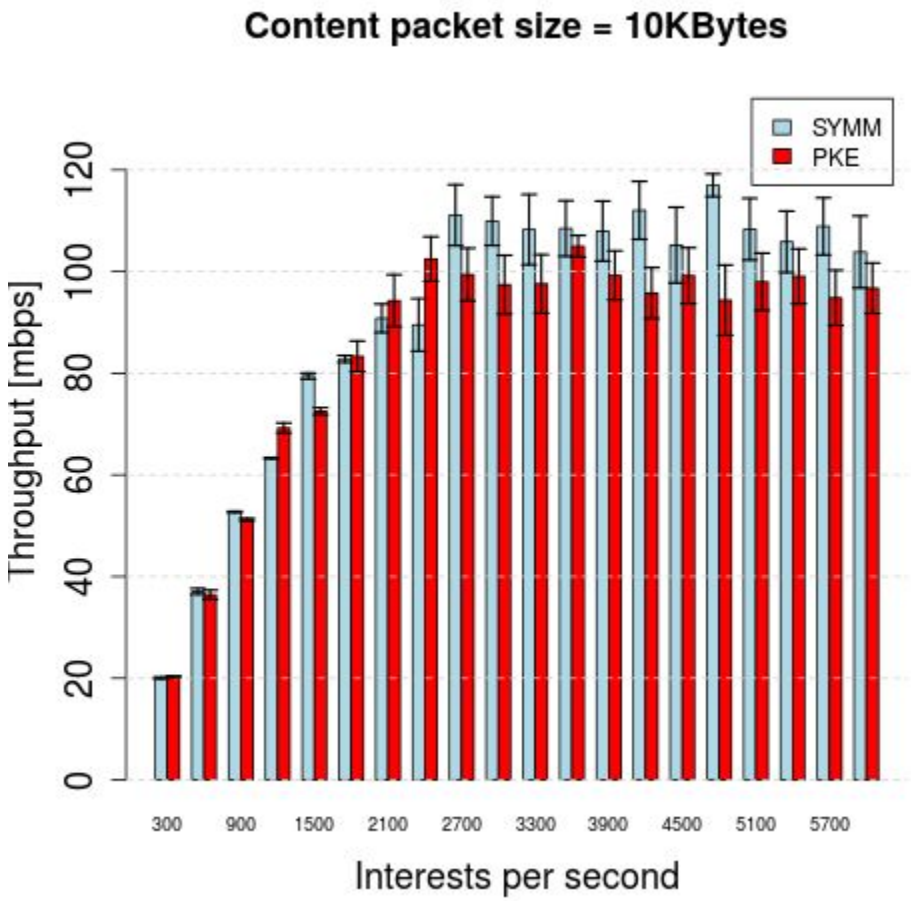
CCVPN: Evaluation

- Metrics:
  - Throughput (Mbps)
  - Avg. RTT (seconds)
- Experiments:
  - 1 consumer vs. 1 producer (w/ increasing Interest issuance rate)
  - Multiple consumers vs. 1 producer
  - Multiple consumers and producers
- 2 Versions: PKE and Symm Key

CCVPN: Evaluation

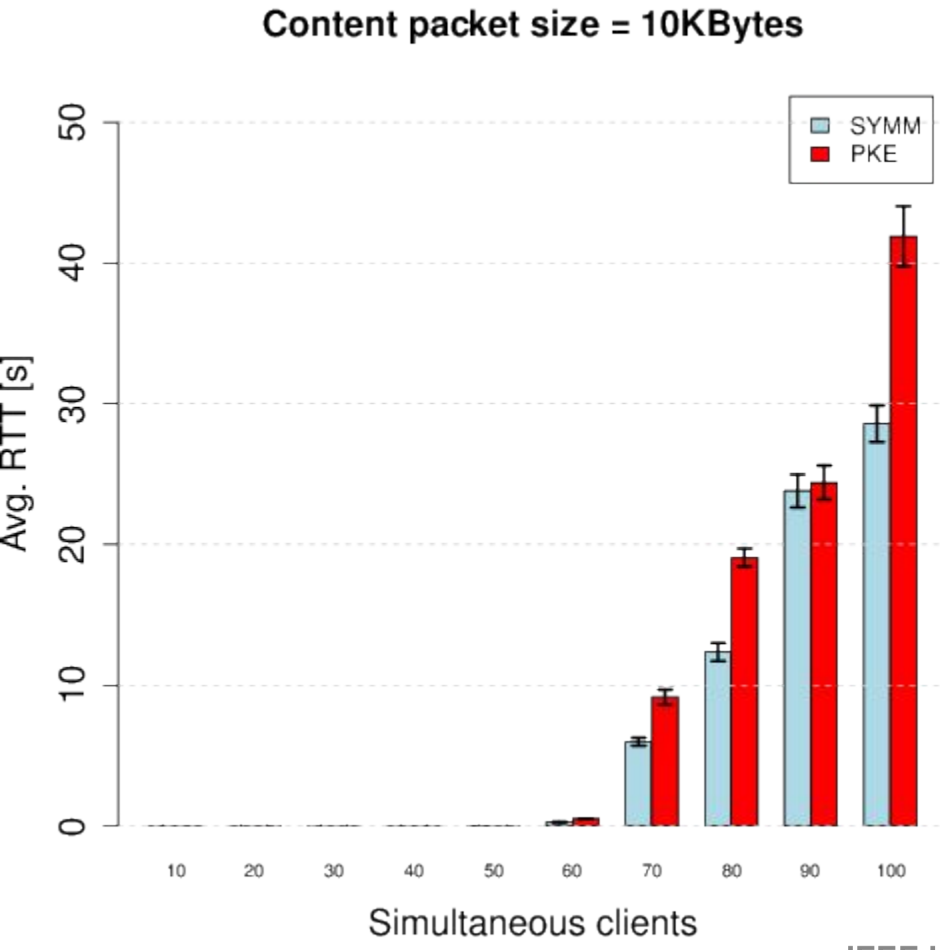
1 consumer x **1** producer:

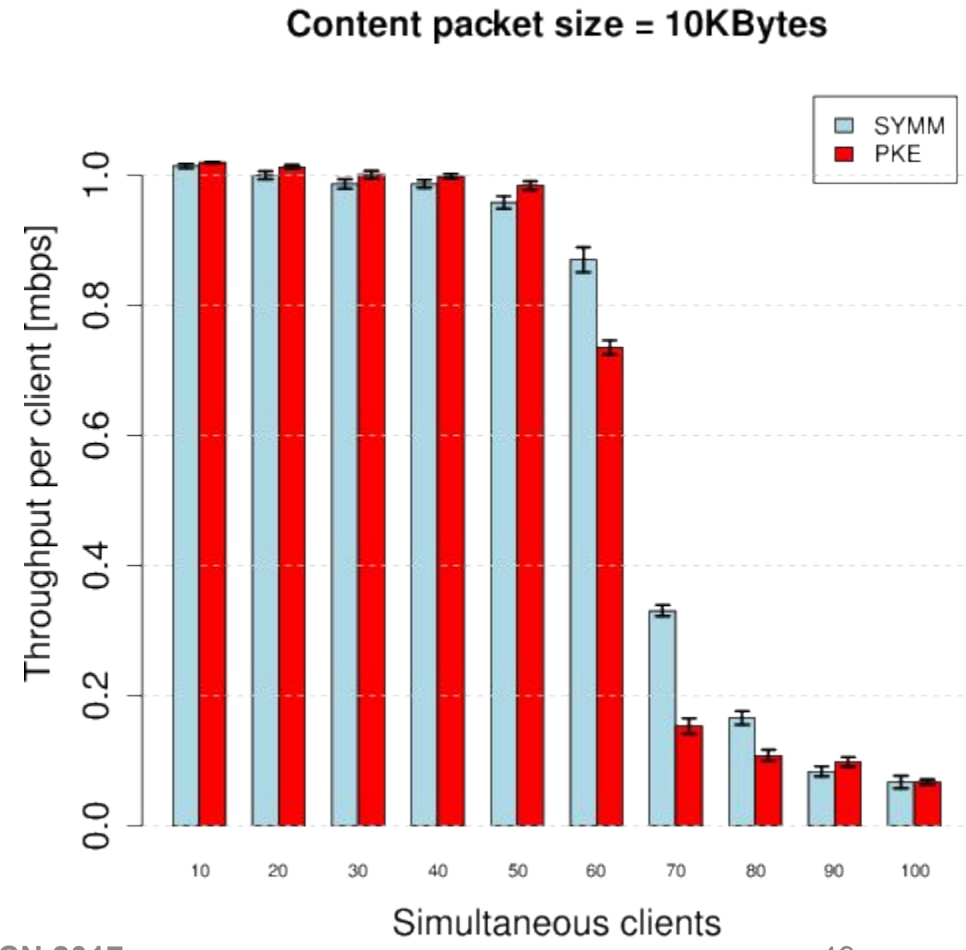




CCVPN: Evaluation

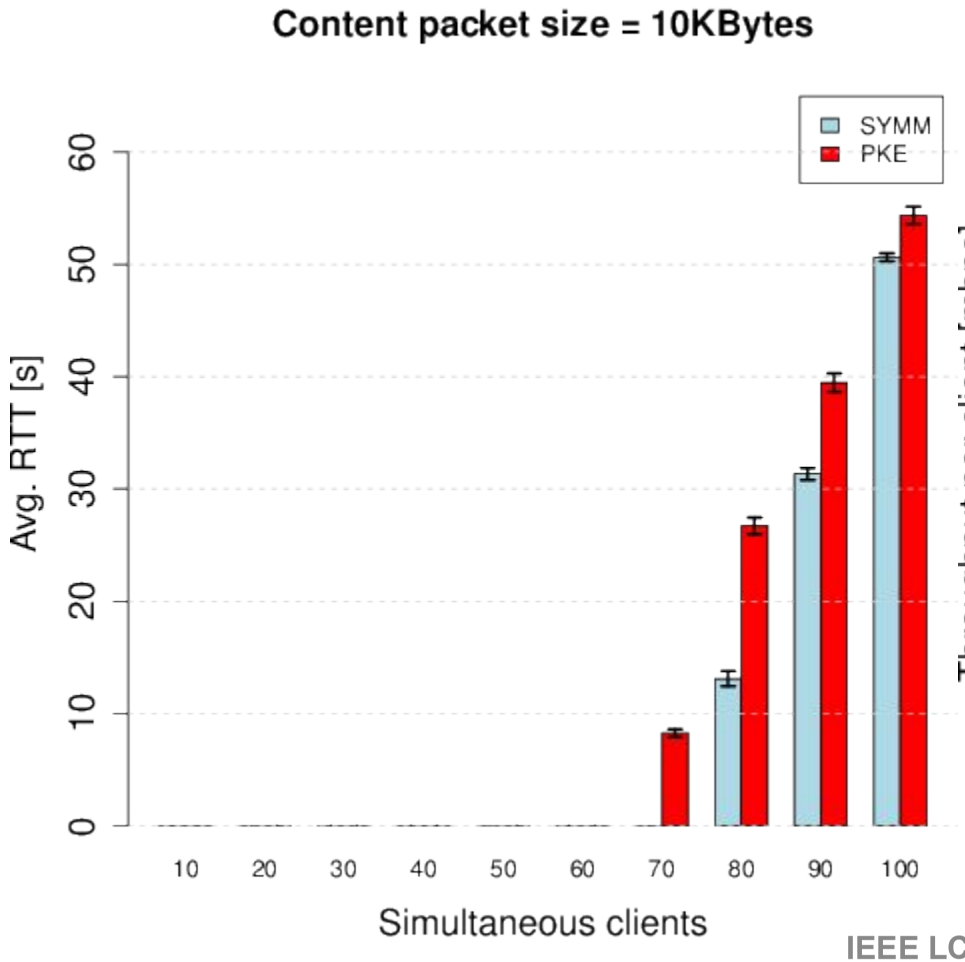
N consumers vs. **1** producer:

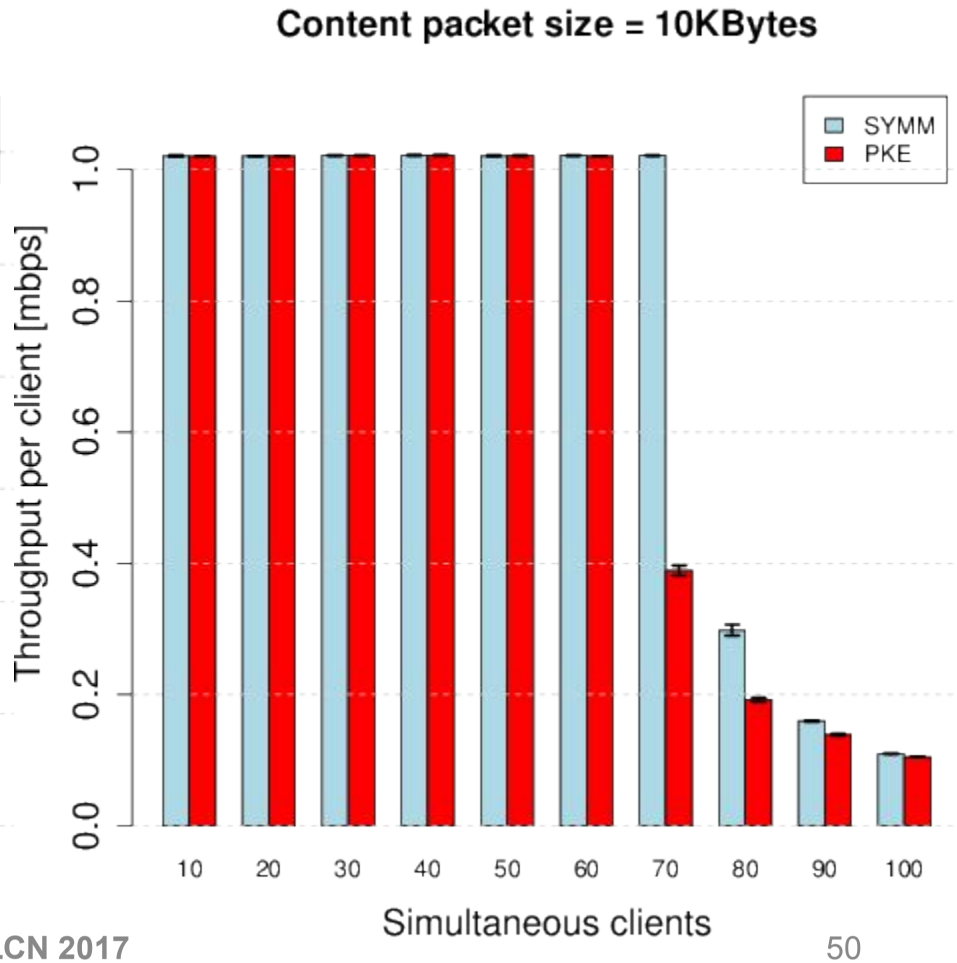




CCVPN: Evaluation

N consumers vs. **N** producers:





Discussion

- Modest processing and storage (20 MB for 100k entries) overhead in the gateways

Discussion

- Modest processing and storage (20 MB for 100k entries) overhead in the gateways
- Better performance possible with:
  - Implementation optimization (CCNx)
  - Distributed load and parallel processing
    - Multiple gateways (and multiple cores) in a single domain
  - Caching

Conclusion

- CCVPN enables VPN funtionality in ICNs
- Unlike Point-to-Point tunnels, multiple *Consumers* share the same namespace tunnel between physically separated private networks.
  - ***Enables Content-Caching inside the VPNs***

Future Work

- CCNxKE to bootstrap shared keys between gateways
- Gateway-to-Gateway Authentication
- DoS countermeasures
- Performance analysis with real-world applications (e.g., file sharing, video streaming)

Questions?