UCI University of California, Irvine

Protecting the Long Tail Transparent Packet Security in Content-Centric Networks

Christopher A. Wood
Department of Computer Science
University of California Irvine
woodc1@uci.edu

IFIP Networking 2017 — June 12, 2017

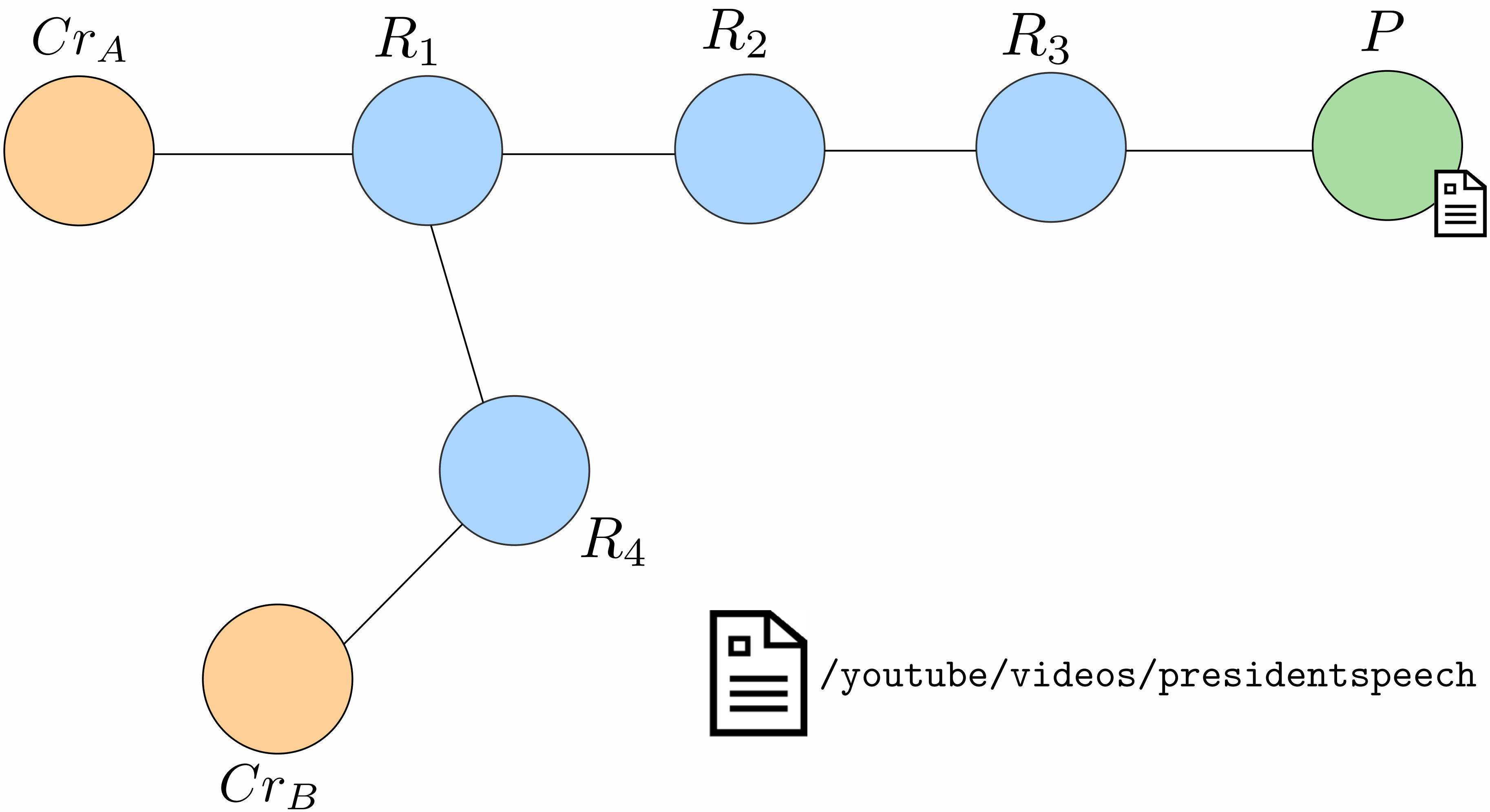
Agenda

- CCN overview
- Privacy parity and transparent encryption
- TRAPS design & features
- Experimental analysis
- Conclusion

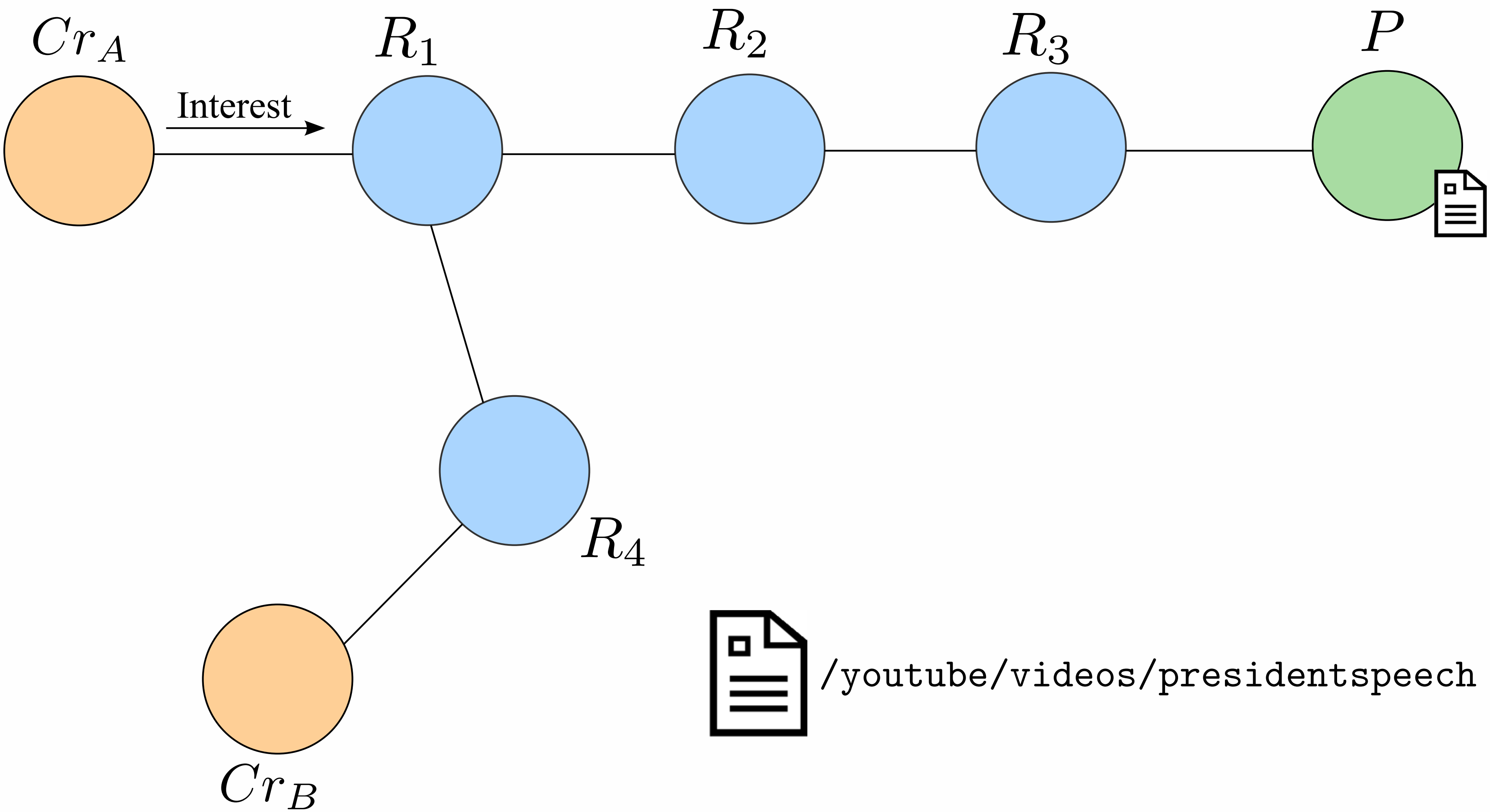
CCN Highlights

- Architecture for transferring **named data** from producer to consumer upon request
- Names are **cryptographically bound** to data
- Requests (interests) are routed based on **names** rather than endpoint addresses
- Content can be **opportunistically cached** in the network

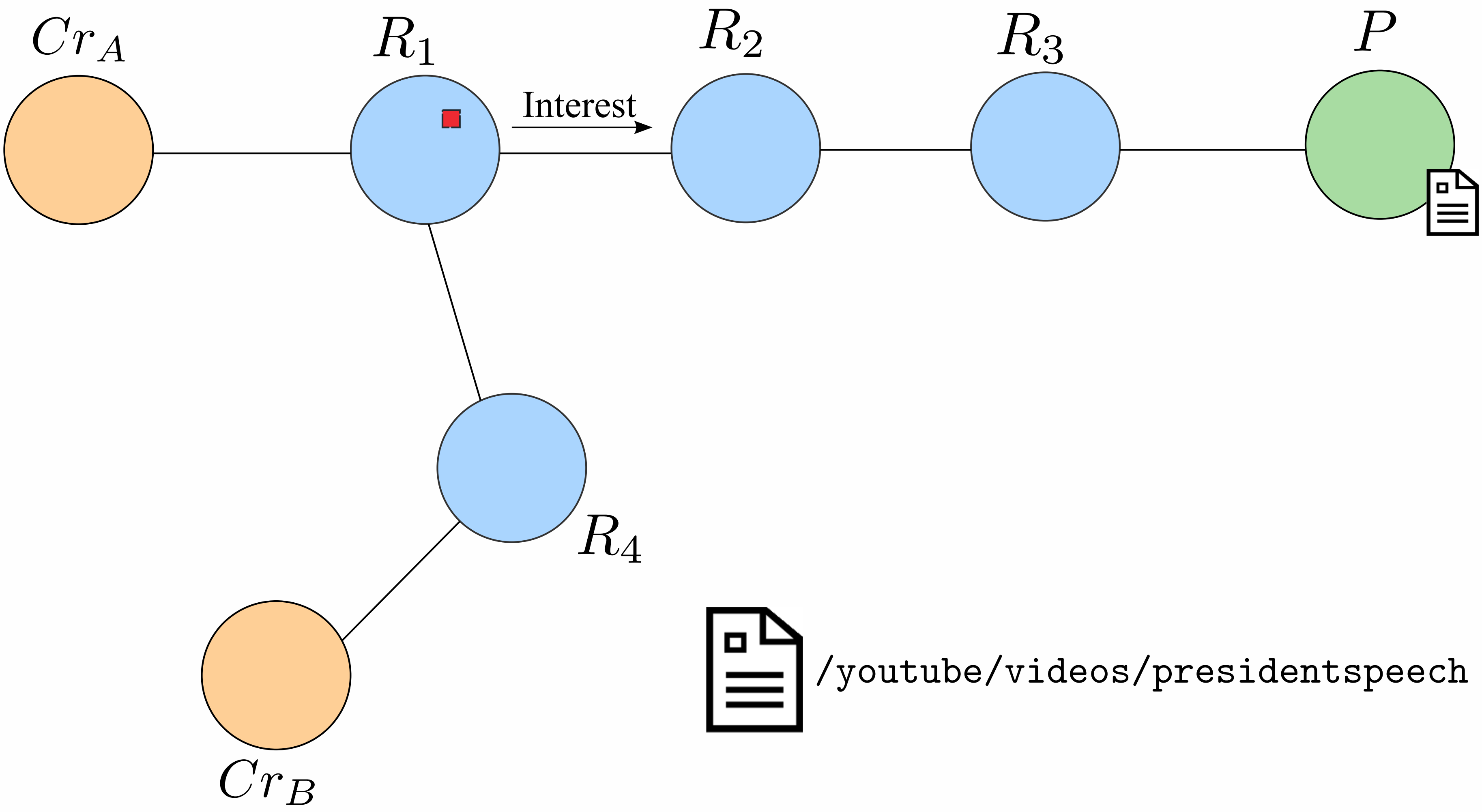
CCN Overview



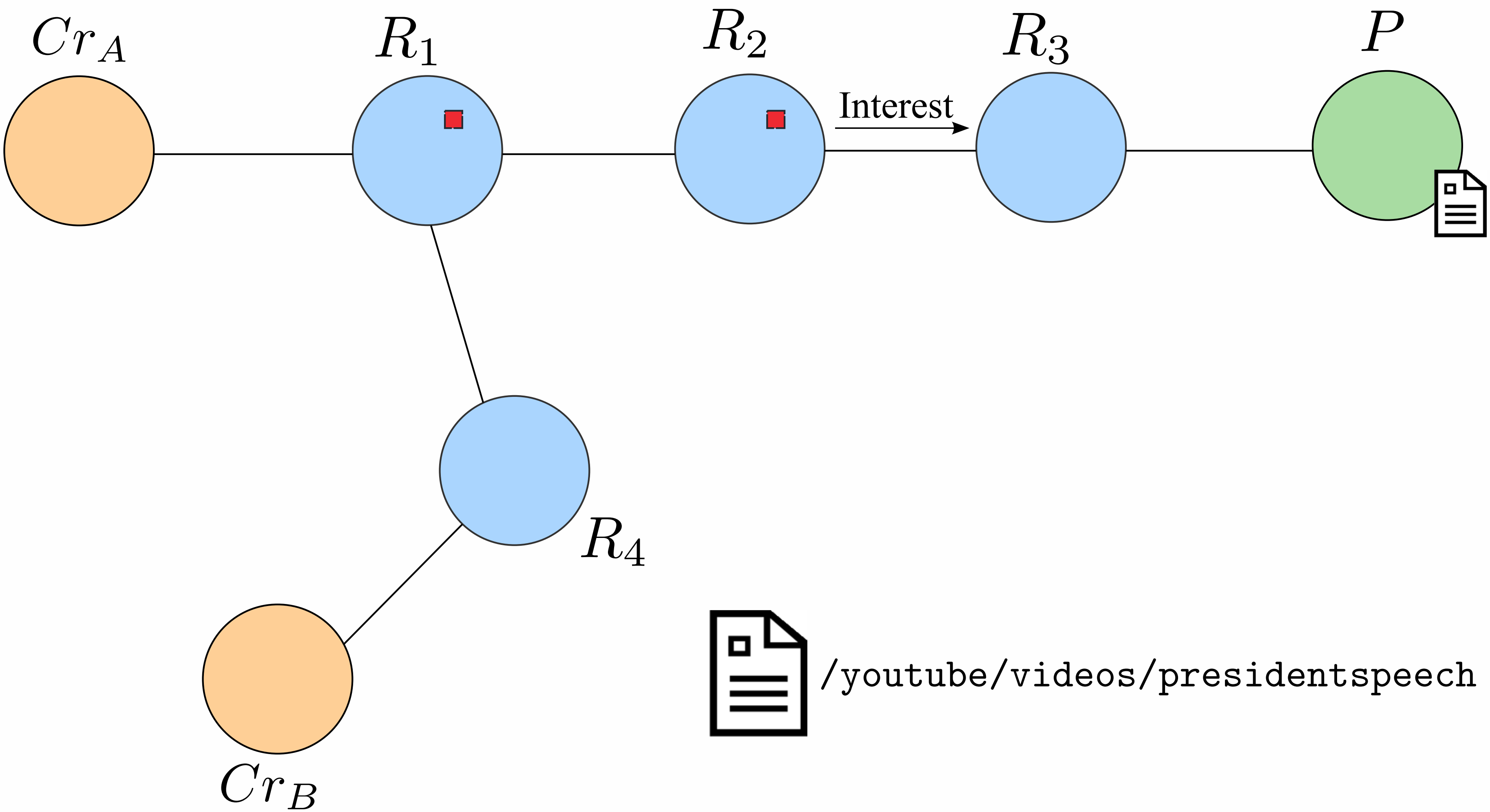
CCN Overview



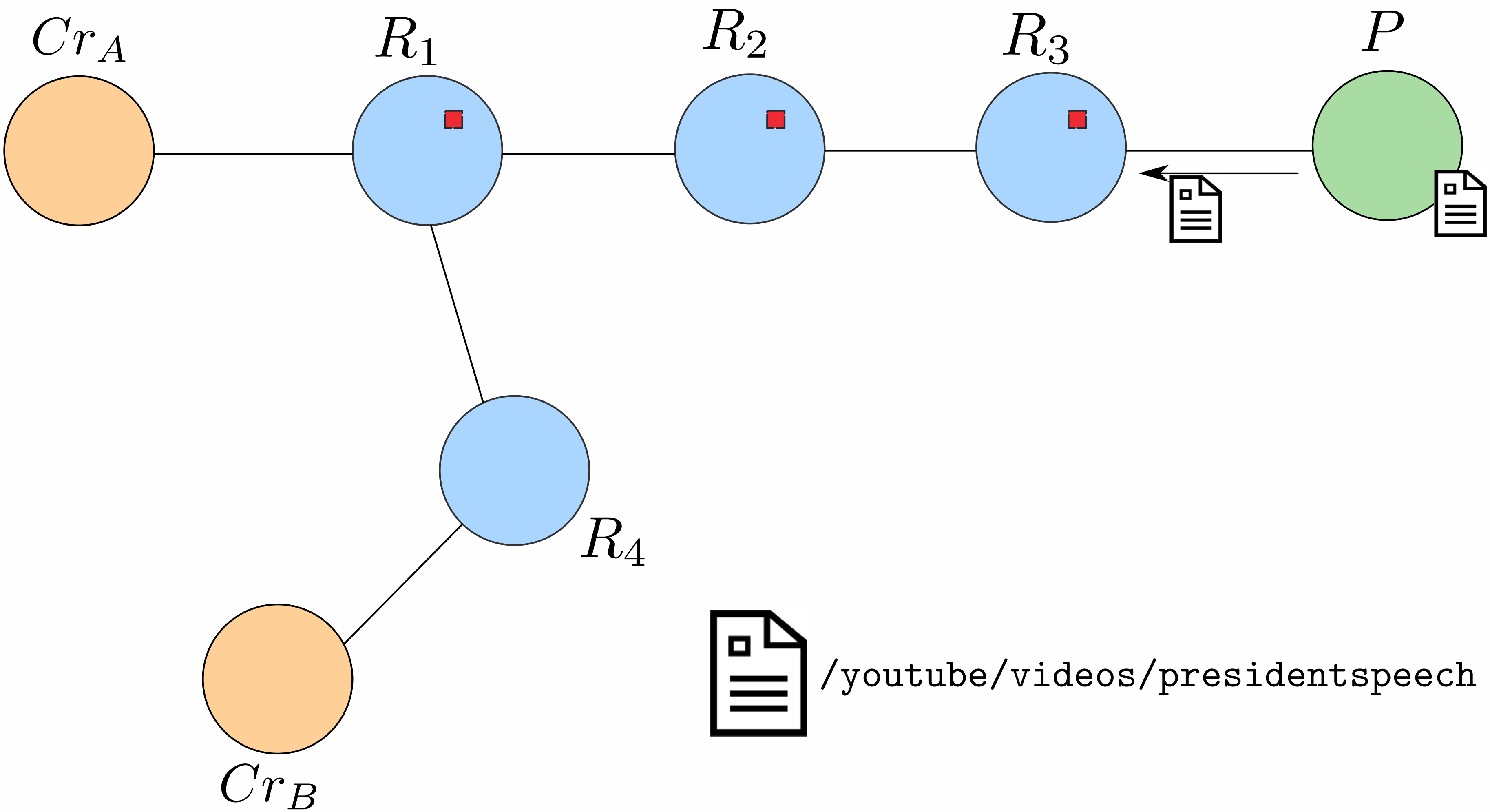
CCN Overview



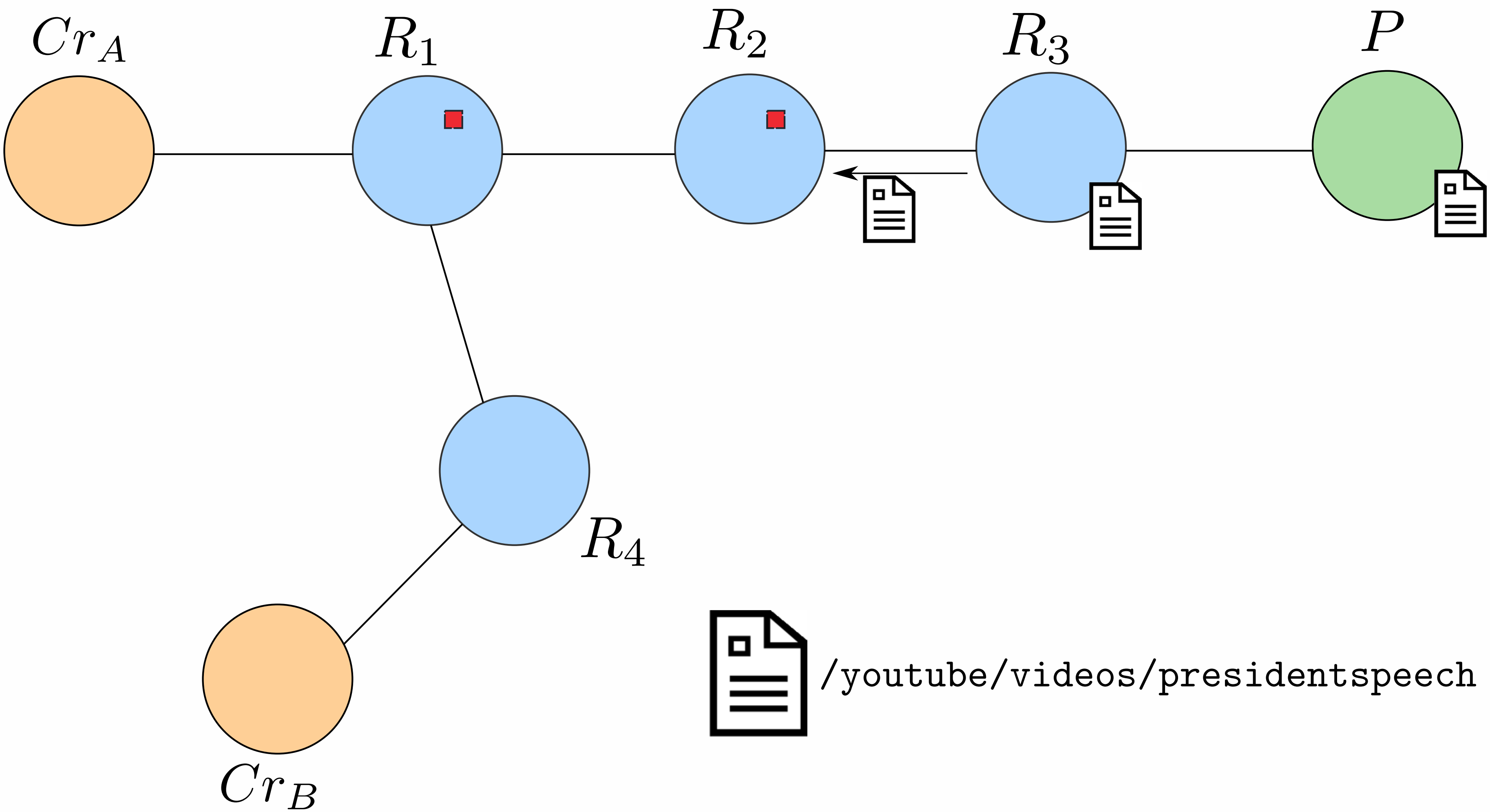
CCN Overview



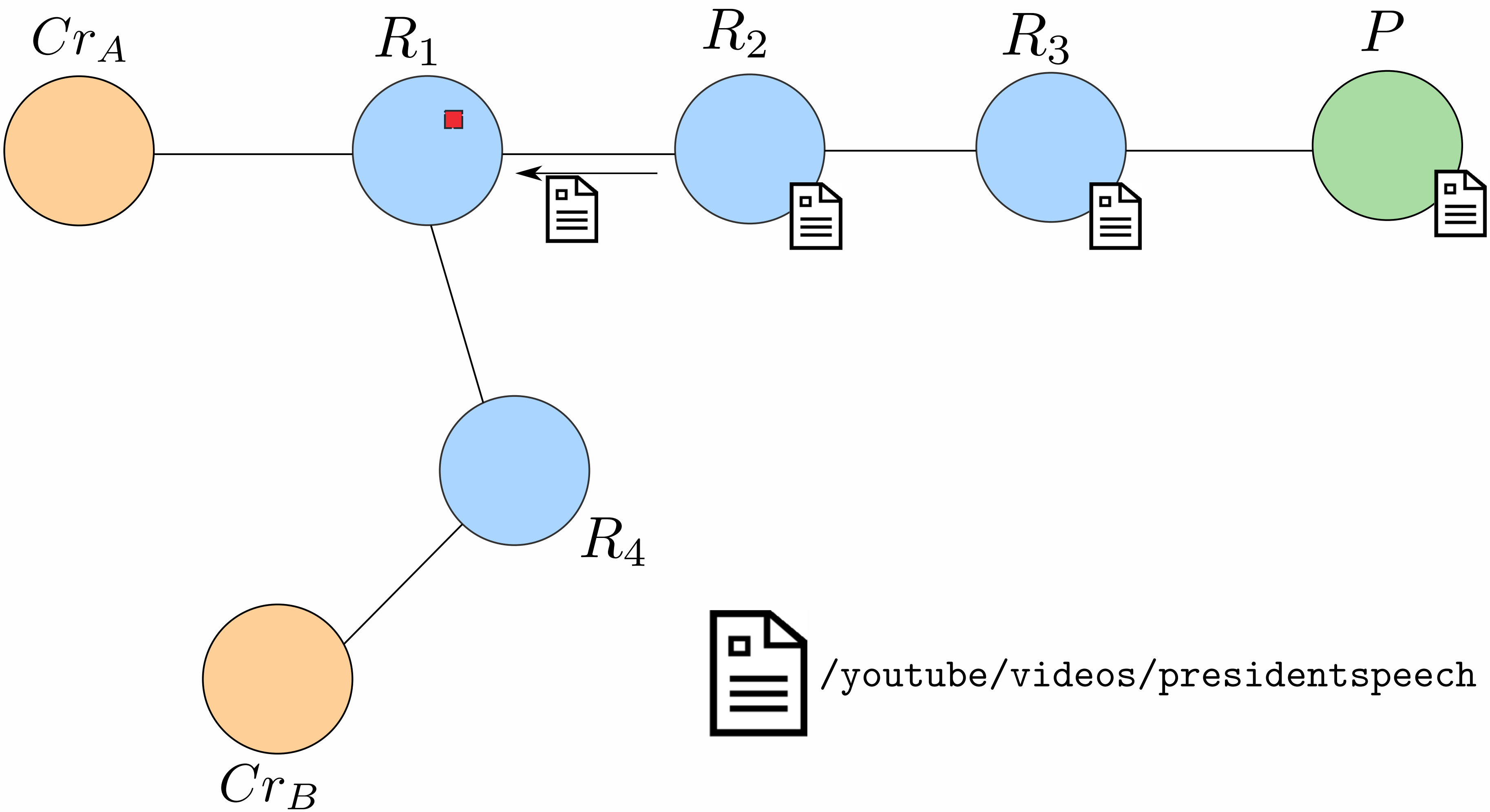
CCN Overview



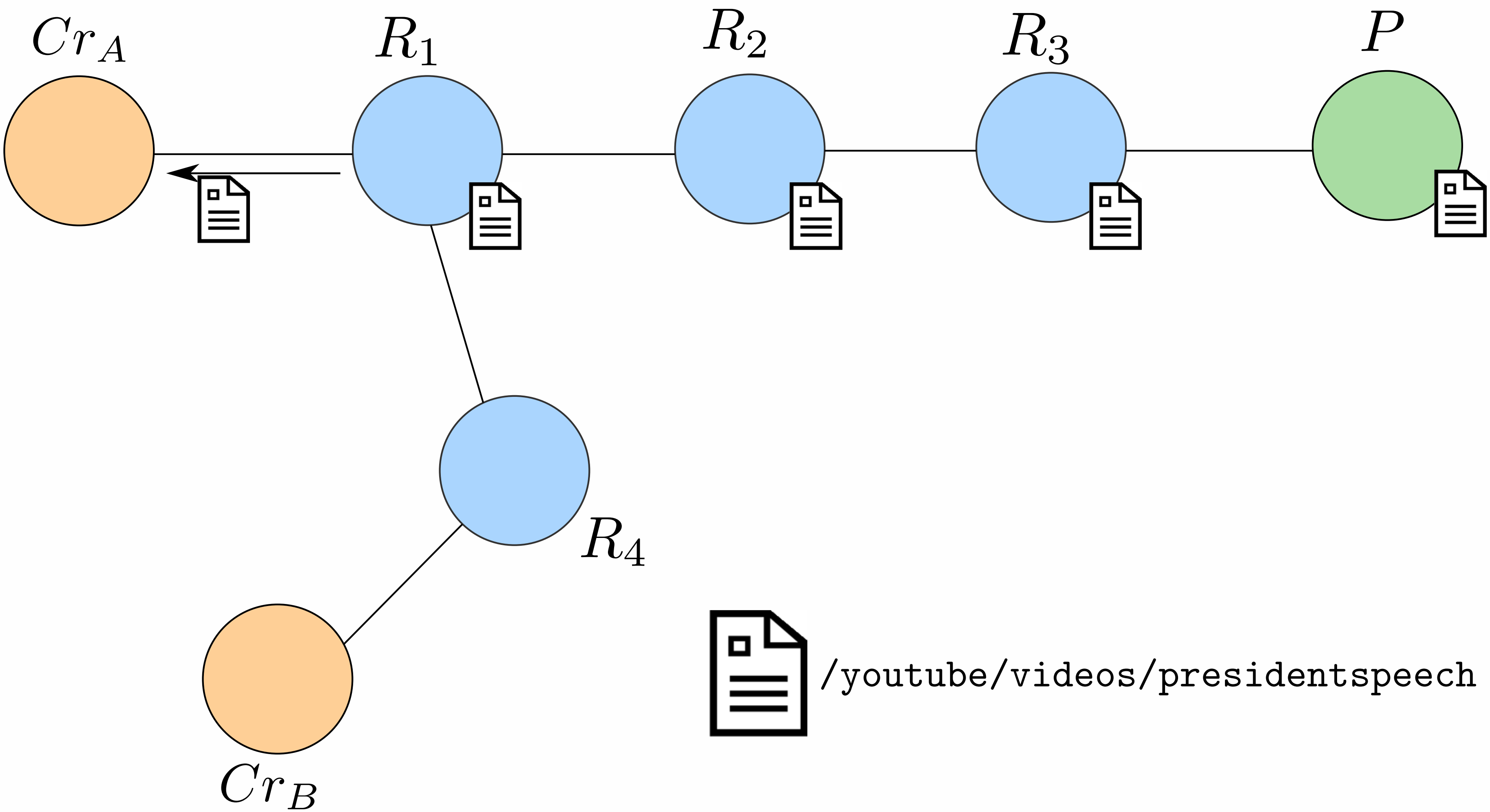
CCN Overview



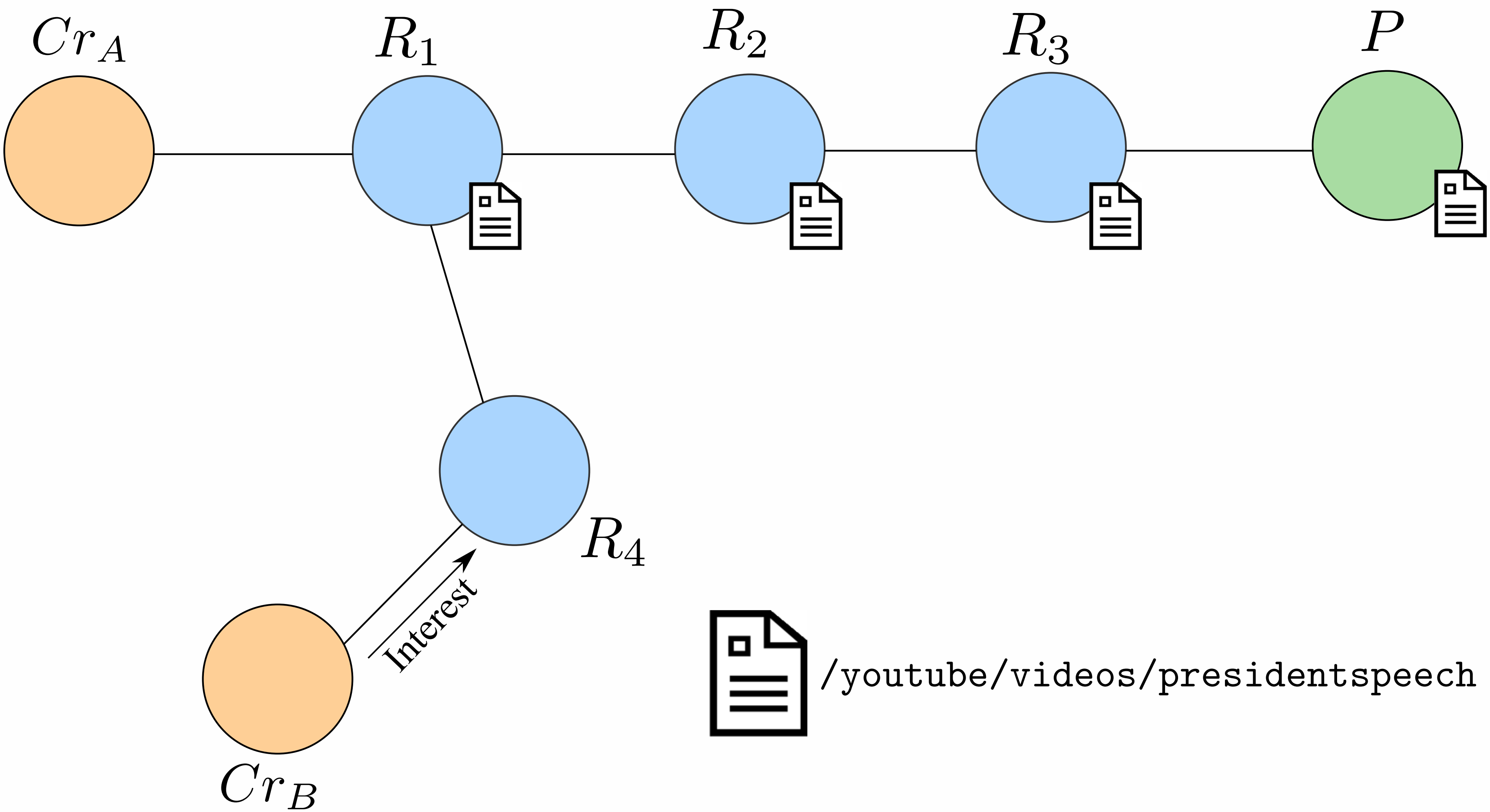
CCN Overview



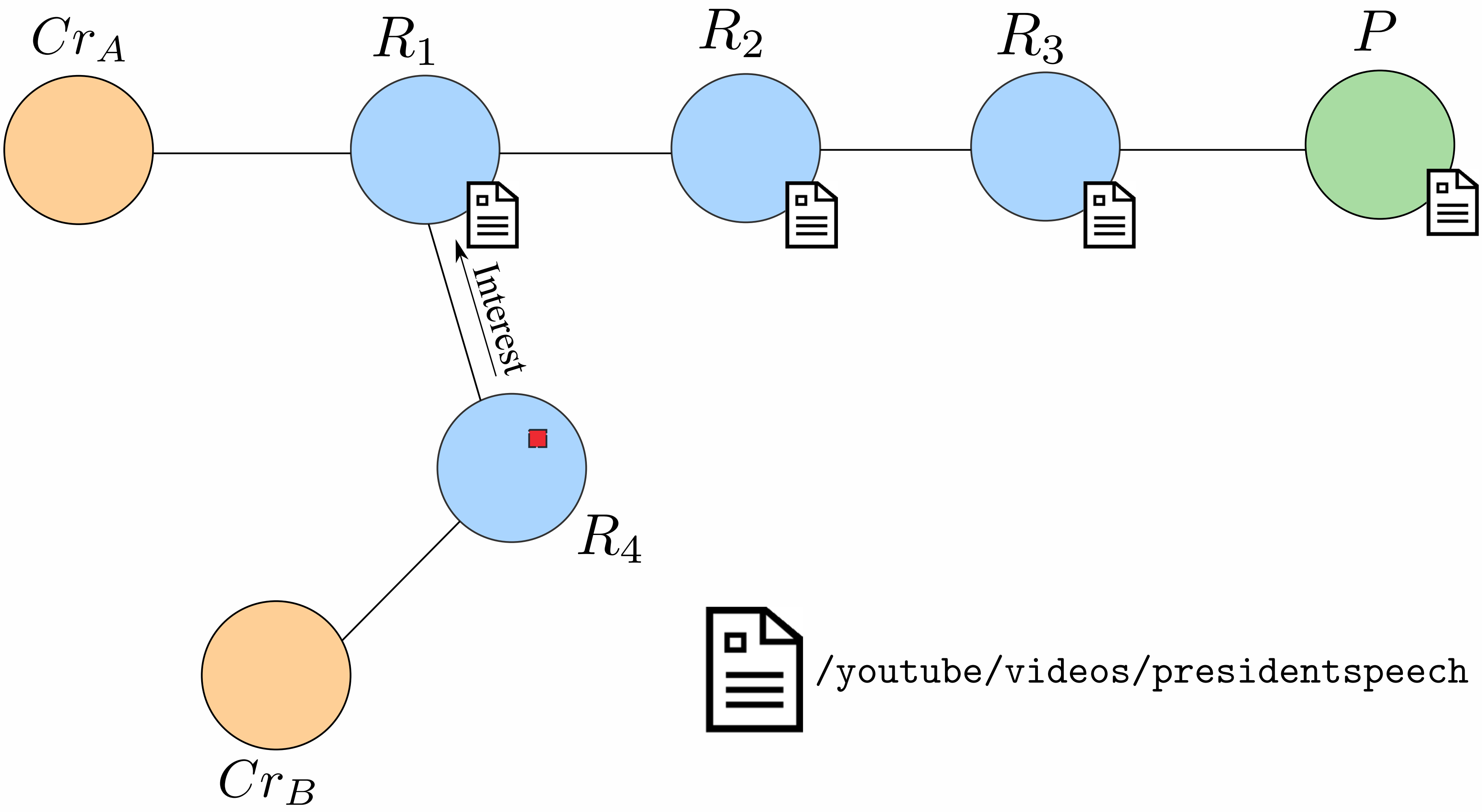
CCN Overview



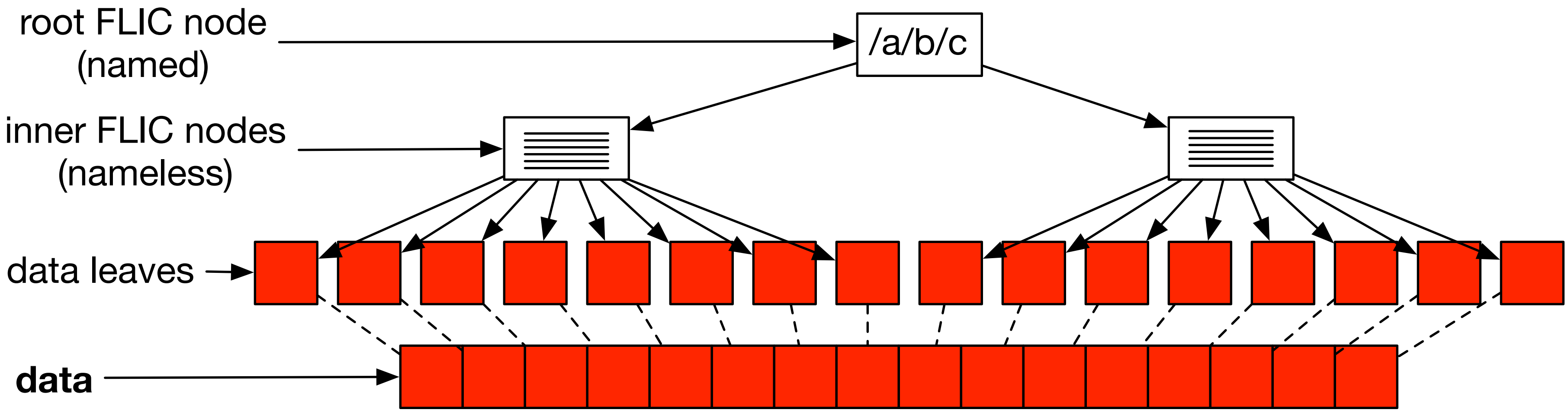
CCN Overview



CCN Overview



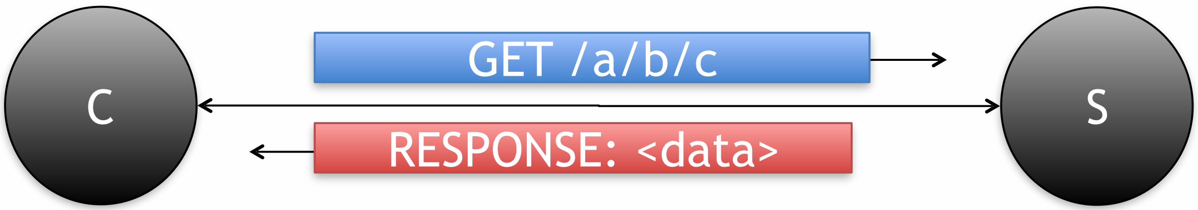
Manifests


root FLIC node
(named)
/a/b/c
inner FLIC nodes
(nameless)
data leaves
data

Onto Privacy

IP Privacy

Turns this...



IP Privacy

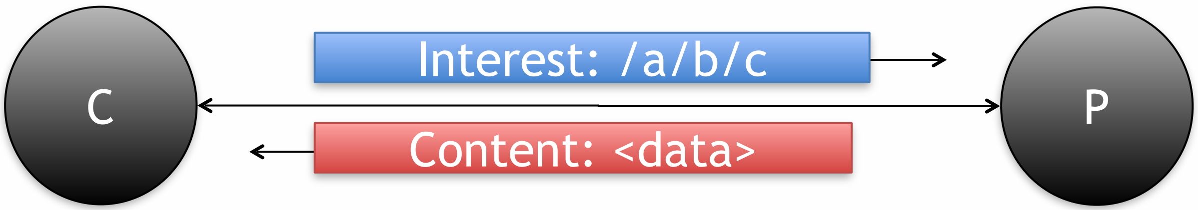
Into this… (with IPSec or TLS)

What's revealed?

- Source and destination addresses and port #
- Timing
- Packet sizes

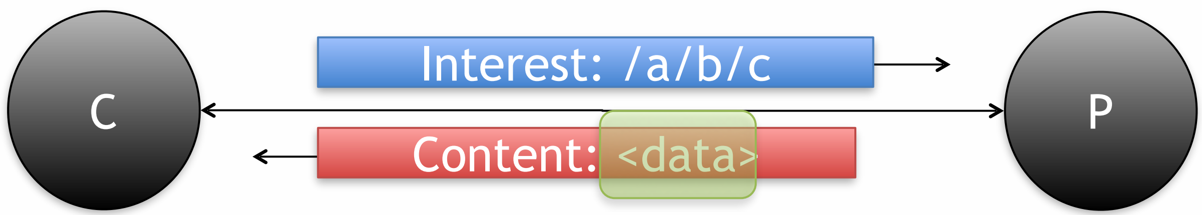
CCN Privacy

Turns this…



CCN Privacy

Into this…



encrypted content?

What's revealed?

- Consumer and producer locations
- Timing
- Packet sizes
- Interest name
- Producer identity
- …

→ Properties of the (application) **data**

Privacy Parity?

CCN

- Application names are necessarily expressed in the clear
- Content is unencrypted unless consumer and producer agree on encryption mechanism and key(s)

IP

- Sessions are encrypted and all traffic is encapsulated
- Data is encrypted at the transport layer, beneath the application

Transparent Encryption Goals

Add “transport” privacy to CCN that is:

1. Application agnostic
2. Sound by default with tunable parameters
3. Free from key exchange protocols
4. Compatible with the CCN request-response pattern

TRAPS
TRAnsparent Packet Security

Consumers and producers share one piece of knowledge: **names**

TRAPS
TRAnsparent Packet Security

Consumers and producers share one piece of knowledge: **names**

Use names as a **shared “secret”**

Key Ingredients

1. Application to network name translation
2. Name-based content encryption
3. Hash-based content encryption

(1) Application to Network Names*

/foo/bar/baz

*Ghali, Cesar, Gene Tsudik, and Christopher A. Wood. "Network Names in Content-Centric Networking." *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking*. ACM, 2016.

(1) Application to Network Names*

/foo/bar/baz

$\downarrow$ Name translation function F()

/F(foo)/F(foo/bar)/F(foo/bar/baz)

*Ghali, Cesar, Gene Tsudik, and Christopher A. Wood. "Network Names in Content-Centric Networking." *Proceedings of the 2016 conference on 3rd ACM Conference on Information-Centric Networking*. ACM, 2016.

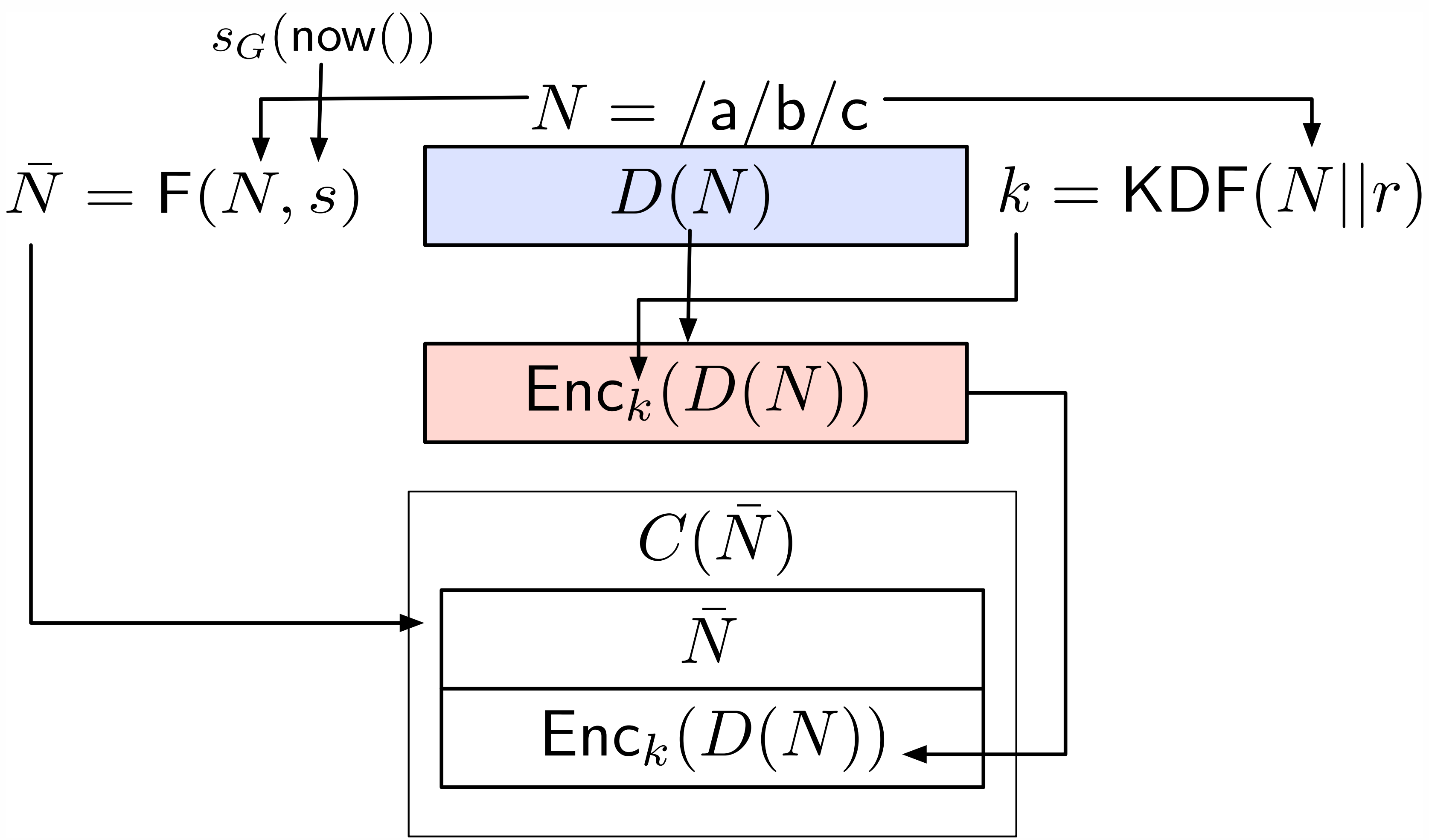
(2) Name-Based Content Encryption

$$r \leftarrow \{0,1\}^\lambda$$

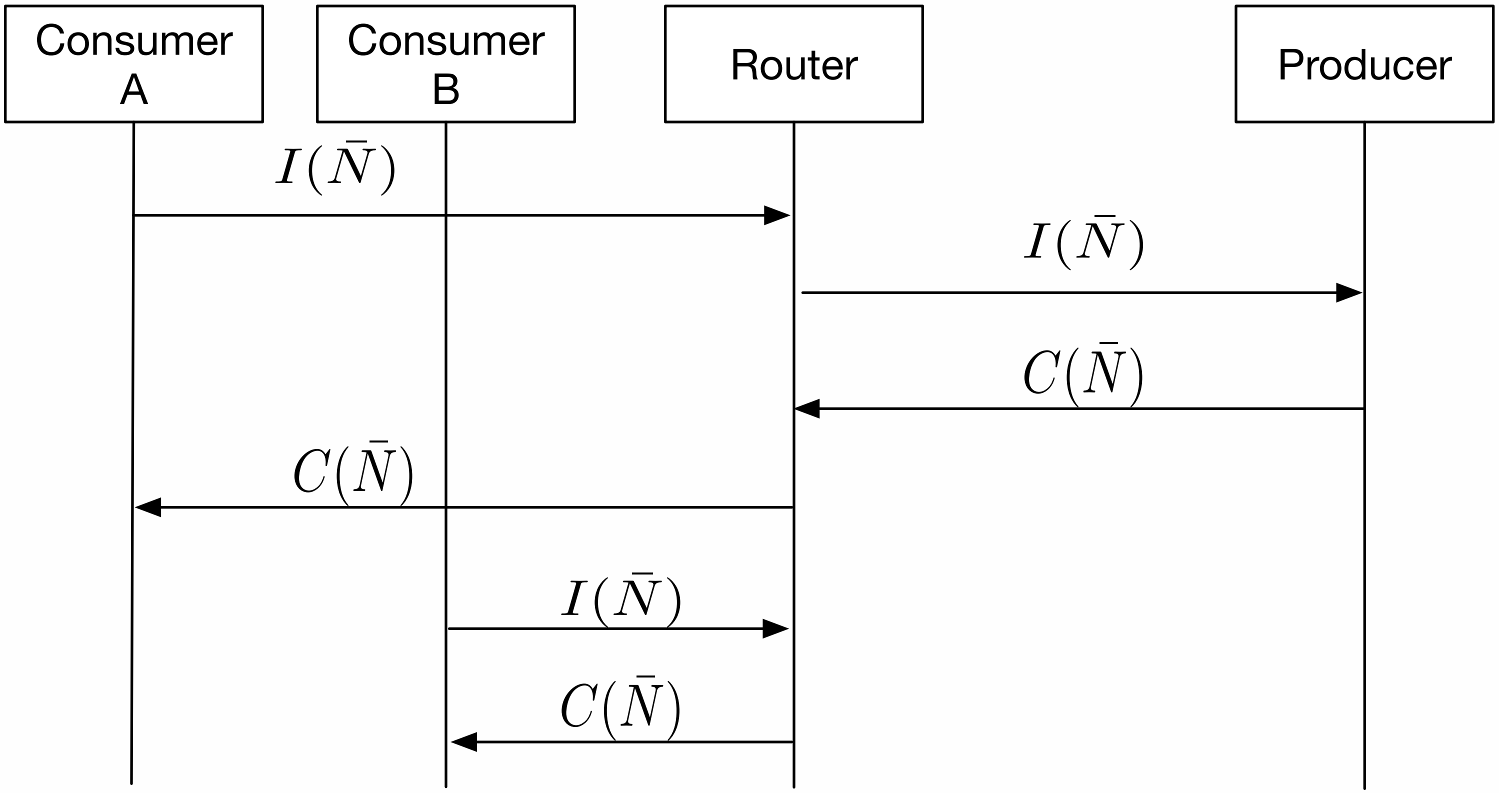
$$k = \mathsf{KDF}(N|r)$$

$$D'(\bar{N}) = \mathsf{Enc}_k(D(N))$$

Named Content



TRAPS Flow



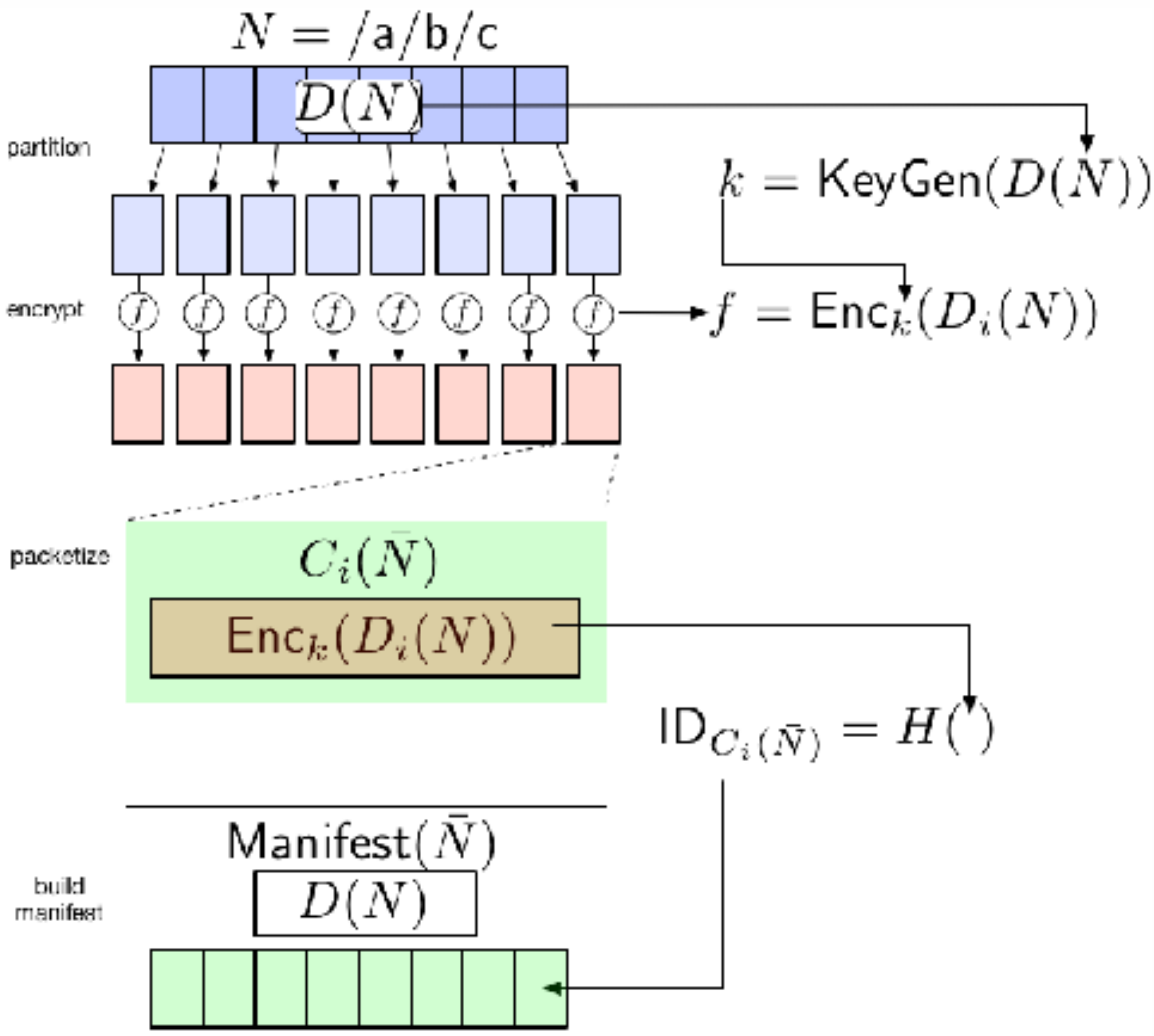
$$N = /\mathrm{foo}/\mathrm{bar}/\mathrm{baz}$$
$$\bar{N} = H(/\mathrm{foo})/H(/\mathrm{foo}/\mathrm{bar})/H(/\mathrm{foo}/\mathrm{bar}/\mathrm{baz})$$

(3) Hash-Based Encryption*

$$k = \mathsf{KeyGen}(D(N))$$
$$D'(\bar{N}) = \mathsf{Enc}_k(D(N))$$

*Bellare, Mihir, Sriram Keelveedhi, and Thomas Ristenpart. "Message-locked encryption and secure deduplication." *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer Berlin Heidelberg, 2013.

Static (Hash-Based) Content



Hash (Key) Discovery

- TRAPS applies to all content — manifests included
- Use manifest to carry hash $D(N)$
- Use CCNxKE (presented later today) to transfer manifests to consumers
  - Content hashes are encrypted in transit

Limitations

- Does not work for **dynamically generated names** — data may be dynamic
- Knowledge of the **full** application name allows one to decrypt the content
  - For hash-based content, one must **also know the content hash**

Security Model

Goal is not TLS-grade security

Security Model

Goal is not TLS-grade security

- Knowledge of N means one can decrypt the content
  - Dictionary attacks are feasible for popular names
- Content in the “long tail” popularity distribution is less prone to attack

Dictionary Attack Hardening

- Add time into the network name generation
- Add a producer-generated salt into the network name generation
- Use memory-hard functions

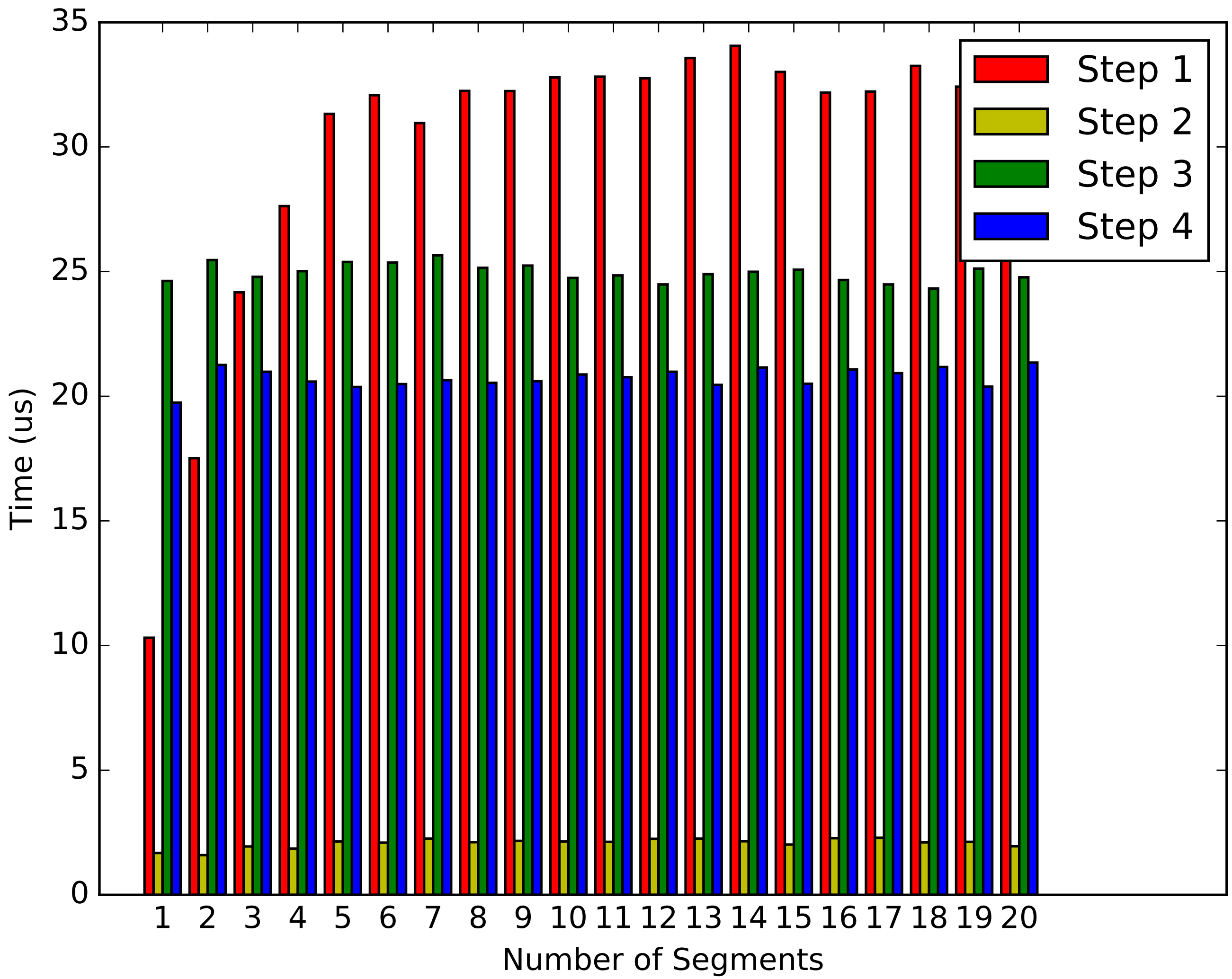
Analysis

Measure the four steps of the protocol:

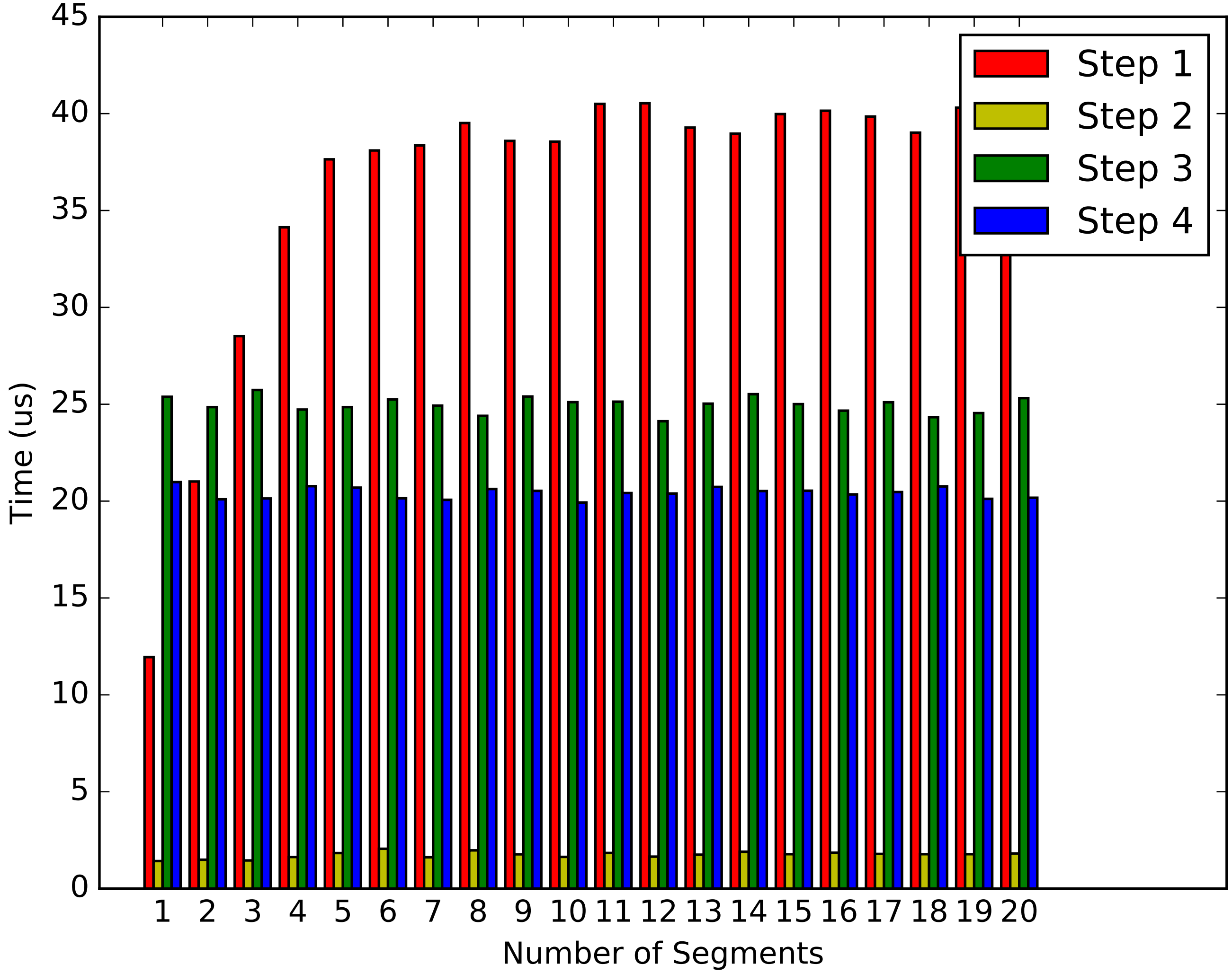
1. Name obfuscation
2. Name de-obfuscation
3. Content encryption
4. Content decryption

Use select hash functions: SHA2 and Argon2

Overhead (SHA256)



Overhead (Argon2)



Throughput Bounds

$$\frac{PT_{obf}}{L} < 1$$

Throughput Bounds

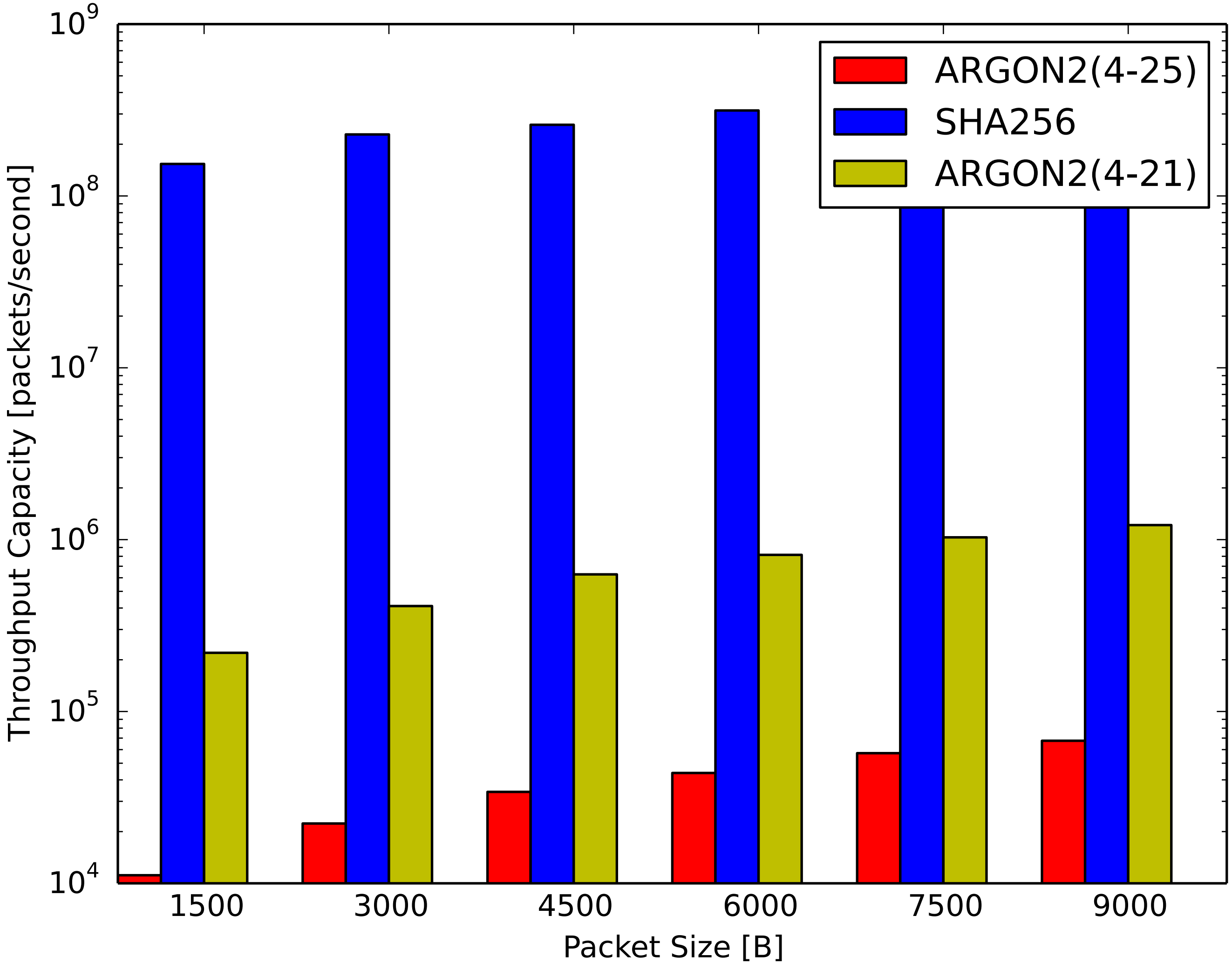
Packet bytes per second [B/s]

Worst-case processing time [s]

$$\frac{PT_{obf}}{L} < 1$$

MTU [B]

Maximal Throughput



Conclusion

- TRAPS brings opportunistic, application-agnostic encryption to CCN
- TRAPS does not offer the same security as TLS — a powerful enough attacker can break it
  - Protecting unpopular content or content with an unpredictable name (or hash!) is the goal
- Performance assessment shows TRAPS add little overhead in the fast path
- TRAPS obfuscation places a limit on the maximal consumer throughput

Questions?

Fire away!