BEAD: Best Effort Autonomous Deletion in Content-Centric Networking

Cesar Ghali, Gene Tsudik, **Christopher A. Wood**
University of California Irvine
{cghali, gene.tsudik, woodc1}@uci.edu

IFIP Networking 2016
May 18, 2016
Vienna, Austria

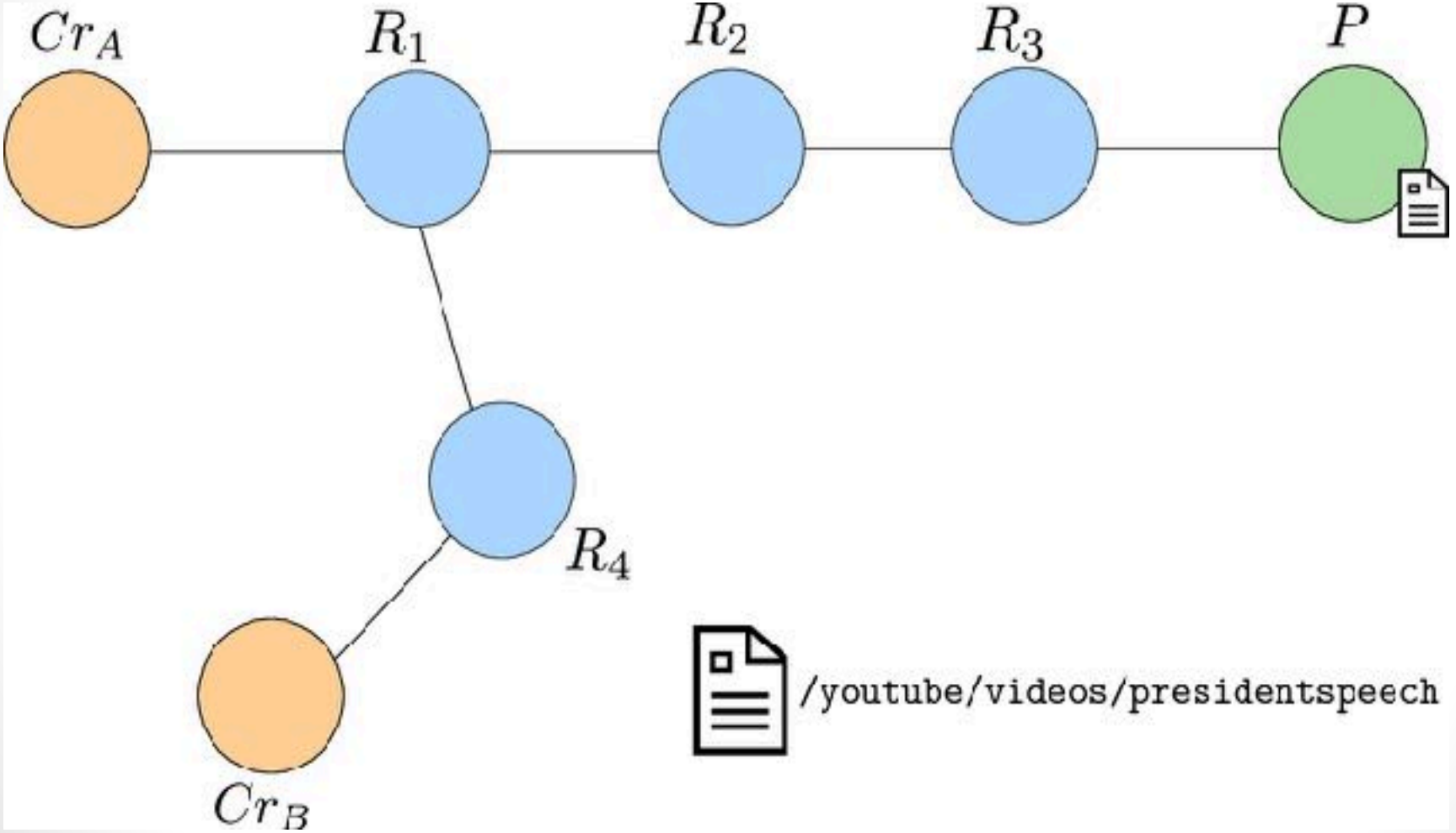
Agenda

- CCN overview and types of content
- On-demand deletion
  - Authenticated deletion requests
  - Distributing deletion requests
  - Analysis
- Experiments
- Conclusion

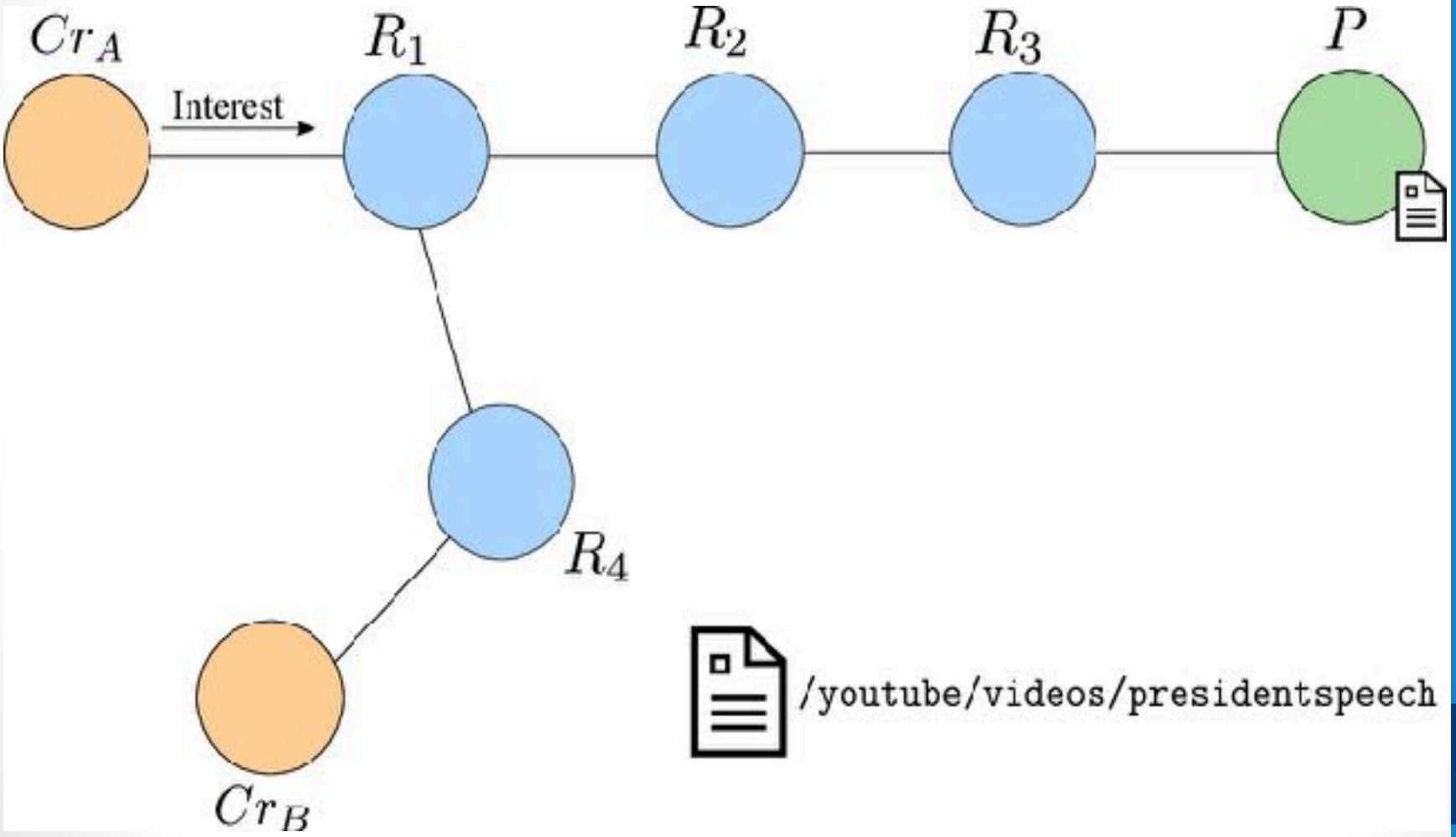
CCN Overview

- CCN is a type of network architectures for transferring **named data**
  - Data is obtained via an explicit request for the name with an **interest**
  - **Consumers** issue interests that are routed towards the data **producer** (using the name)
  - A **content object** carries the data back to the consumer

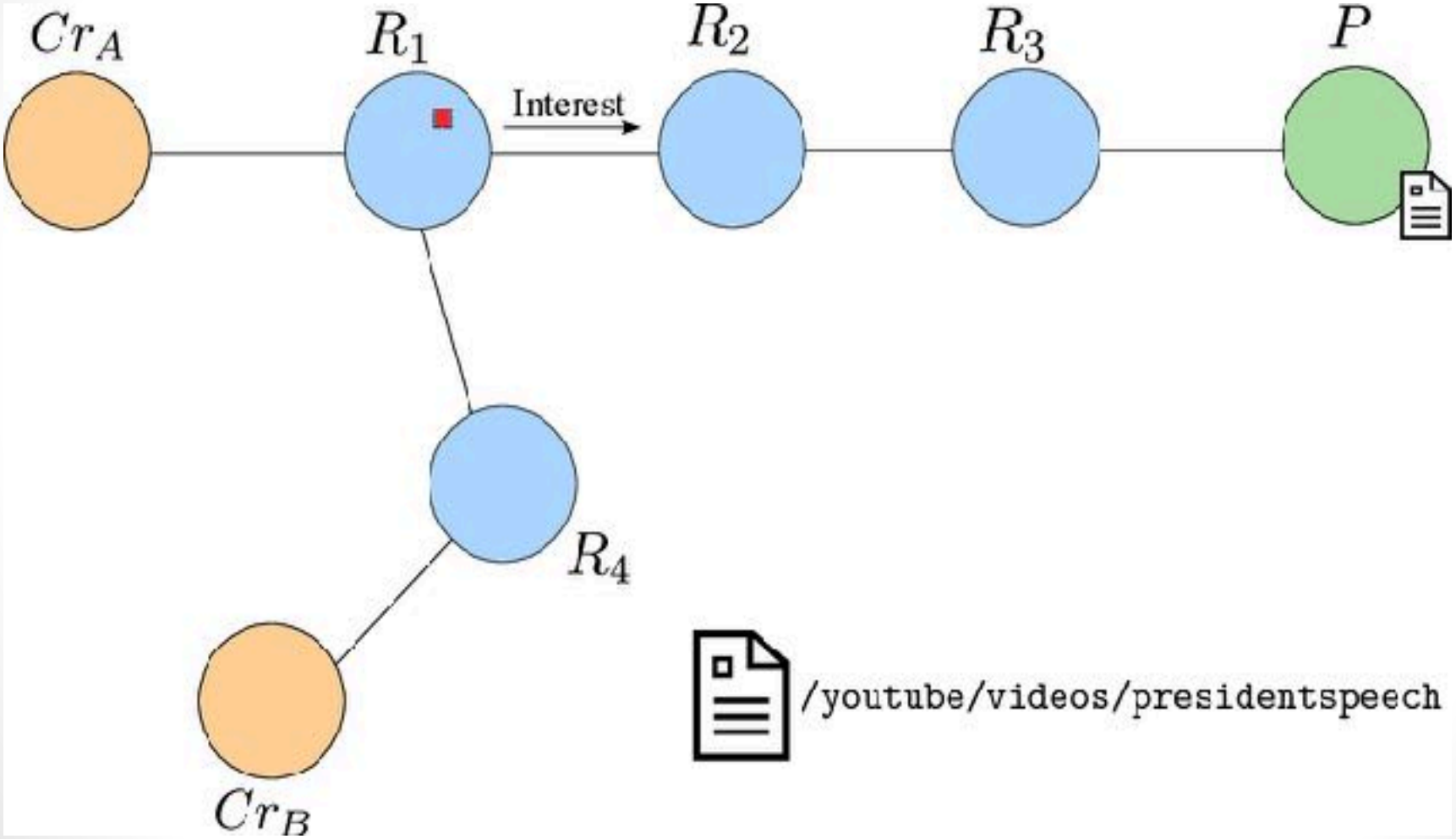
Example



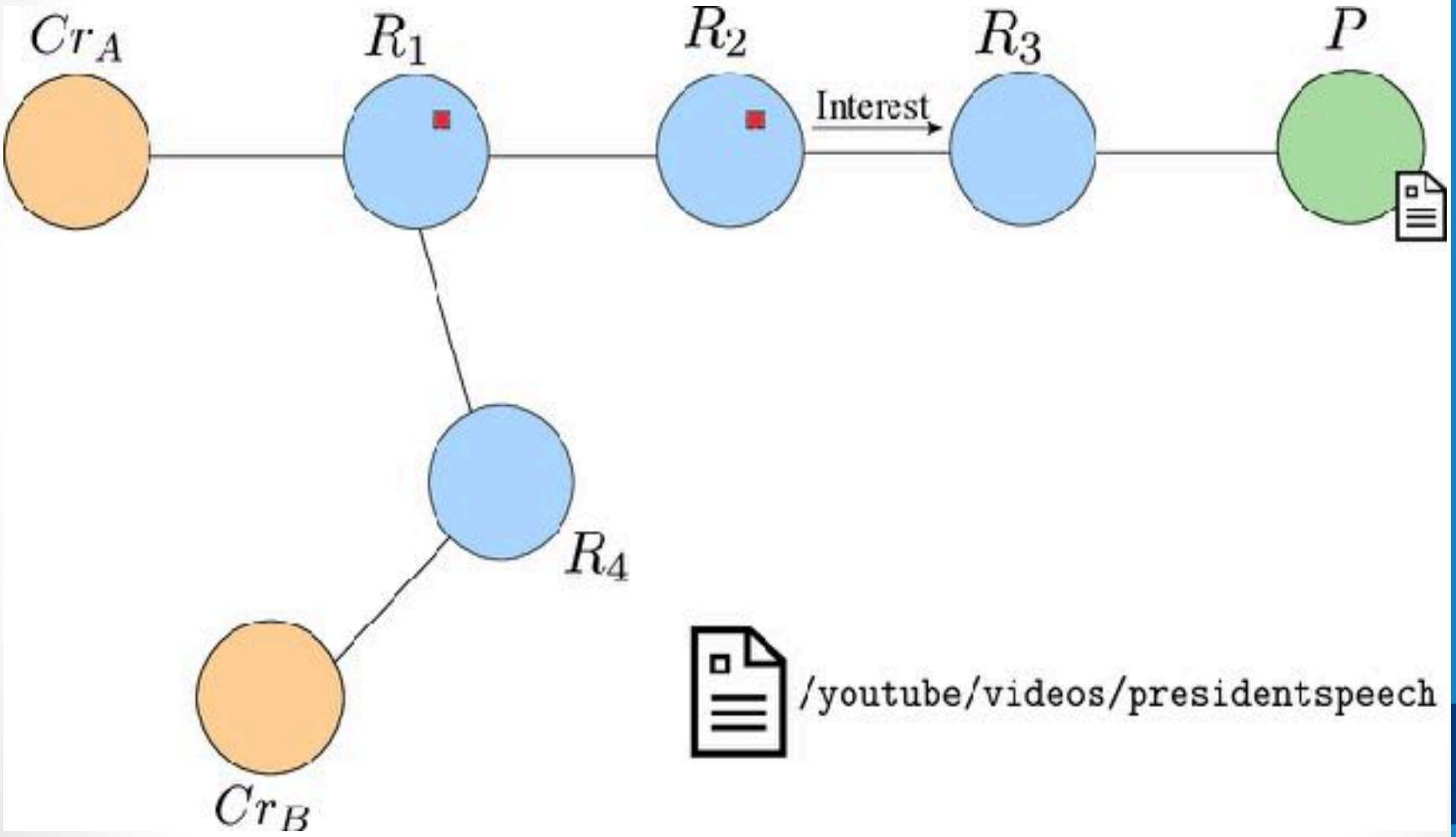
Example


Cr_A
R_1
R_2
R_3
P
Interest
R_4
Cr_B
/youtube/videos/presidentspeech

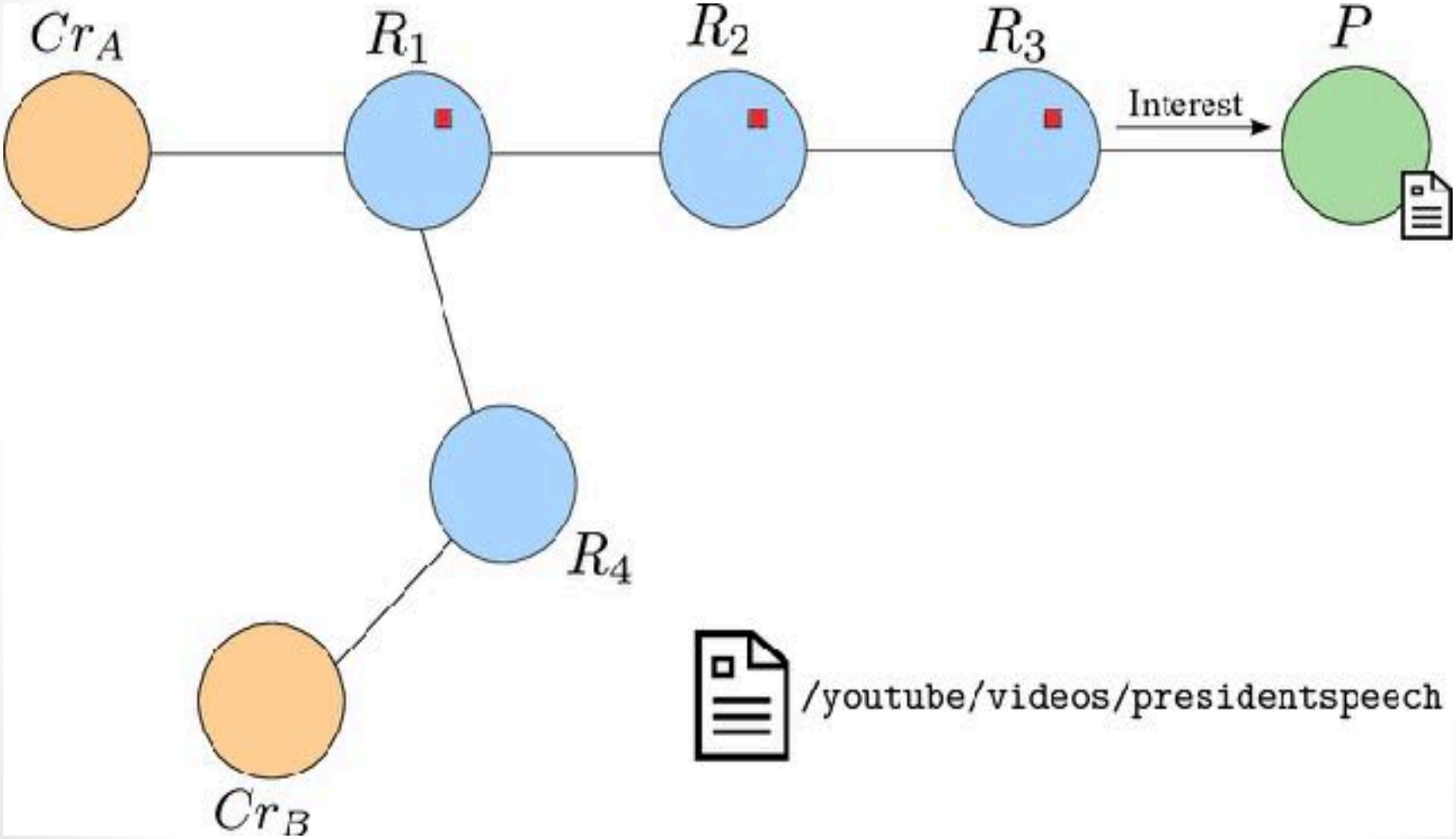
Example


Cr_A
R_1
Interest
R_2
R_3
P
R_4
Cr_B
/youtube/videos/presidentspeech

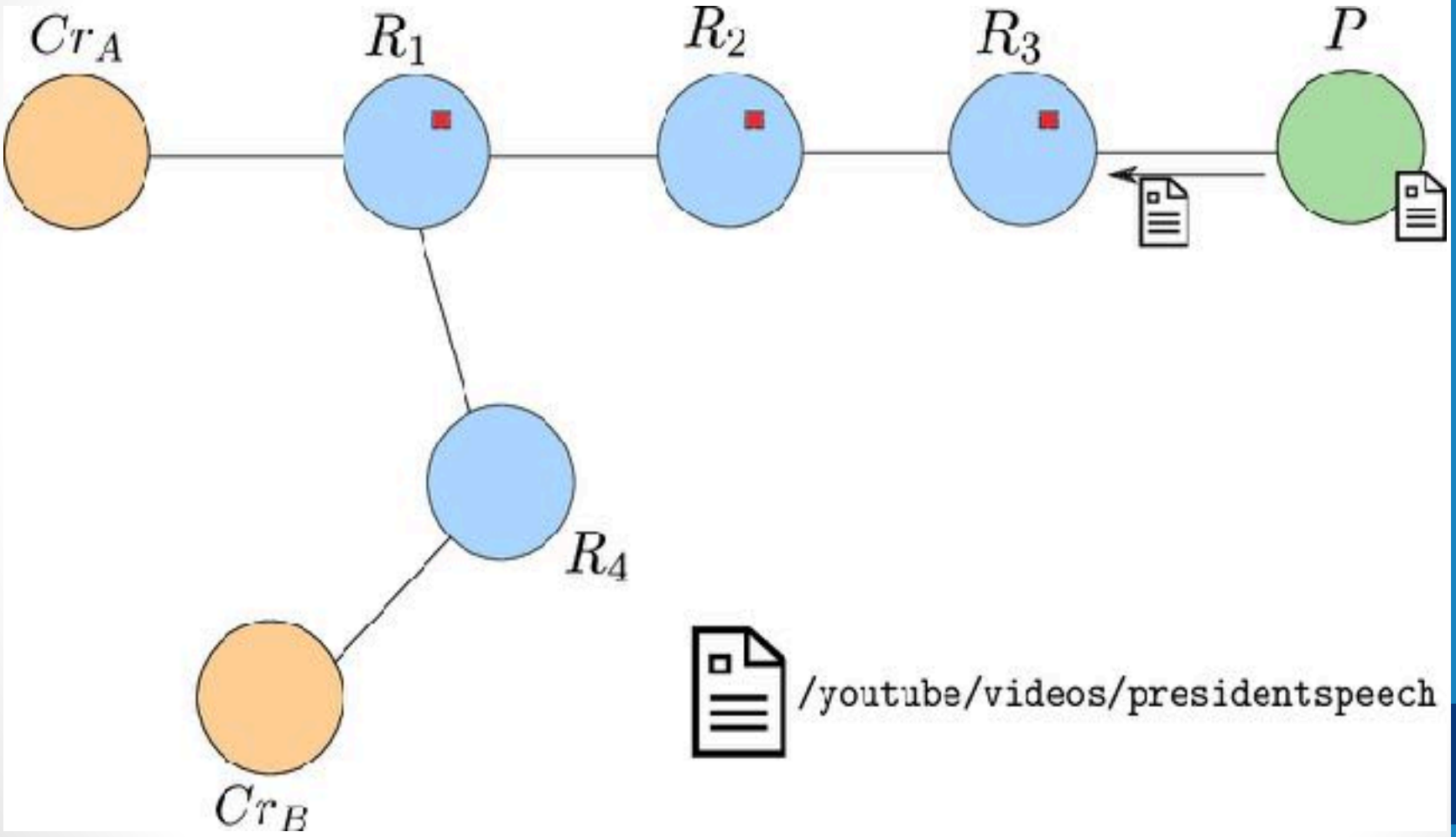
Example


Cr_A
R_1
R_2
R_3
P
Interest
R_4
Cr_B
/youtube/videos/presidentspeech

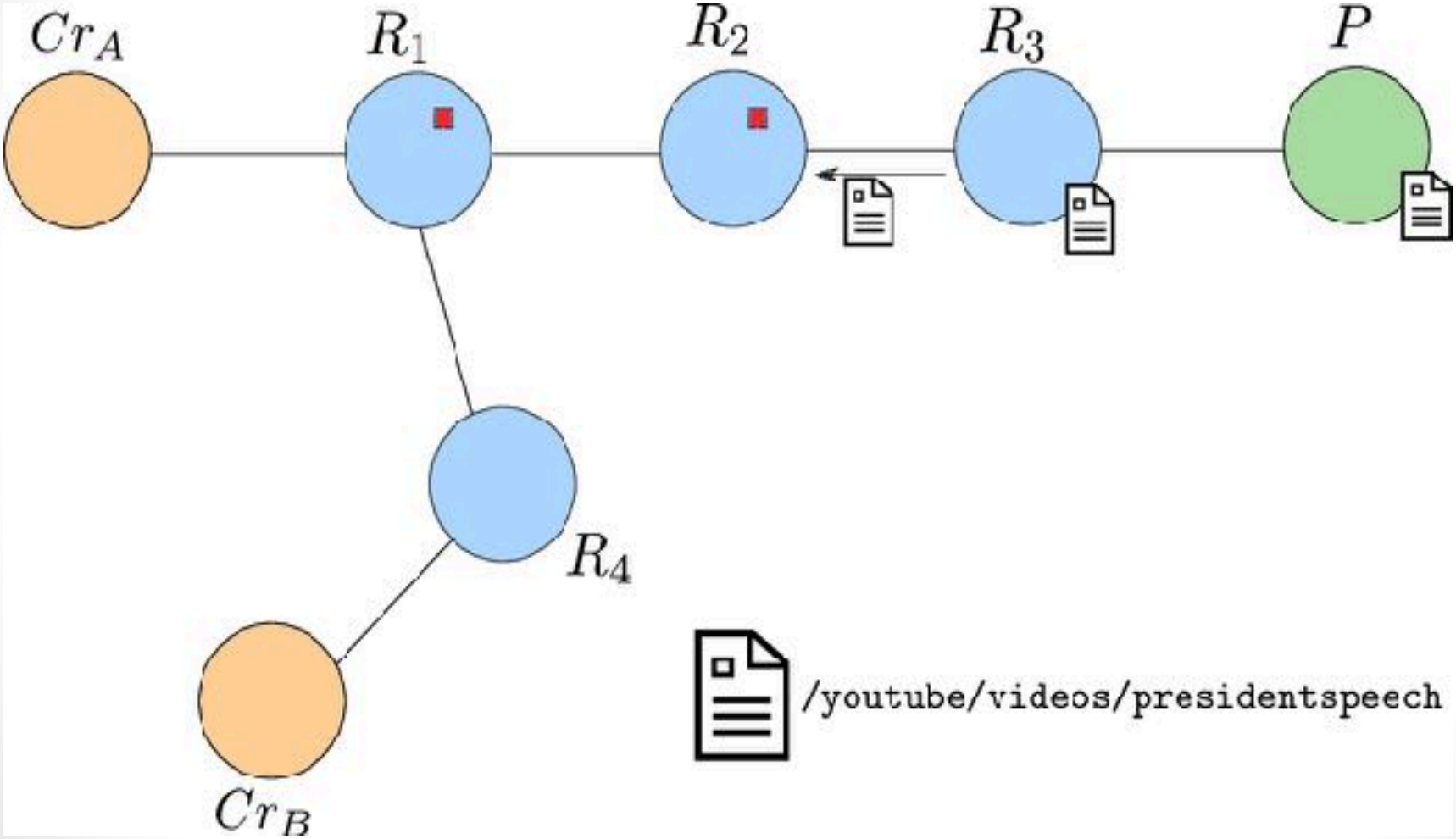
Example


Cr_A
R_1
R_2
R_3
P
Interest
R_4
Cr_B
/youtube/videos/presidentspeech

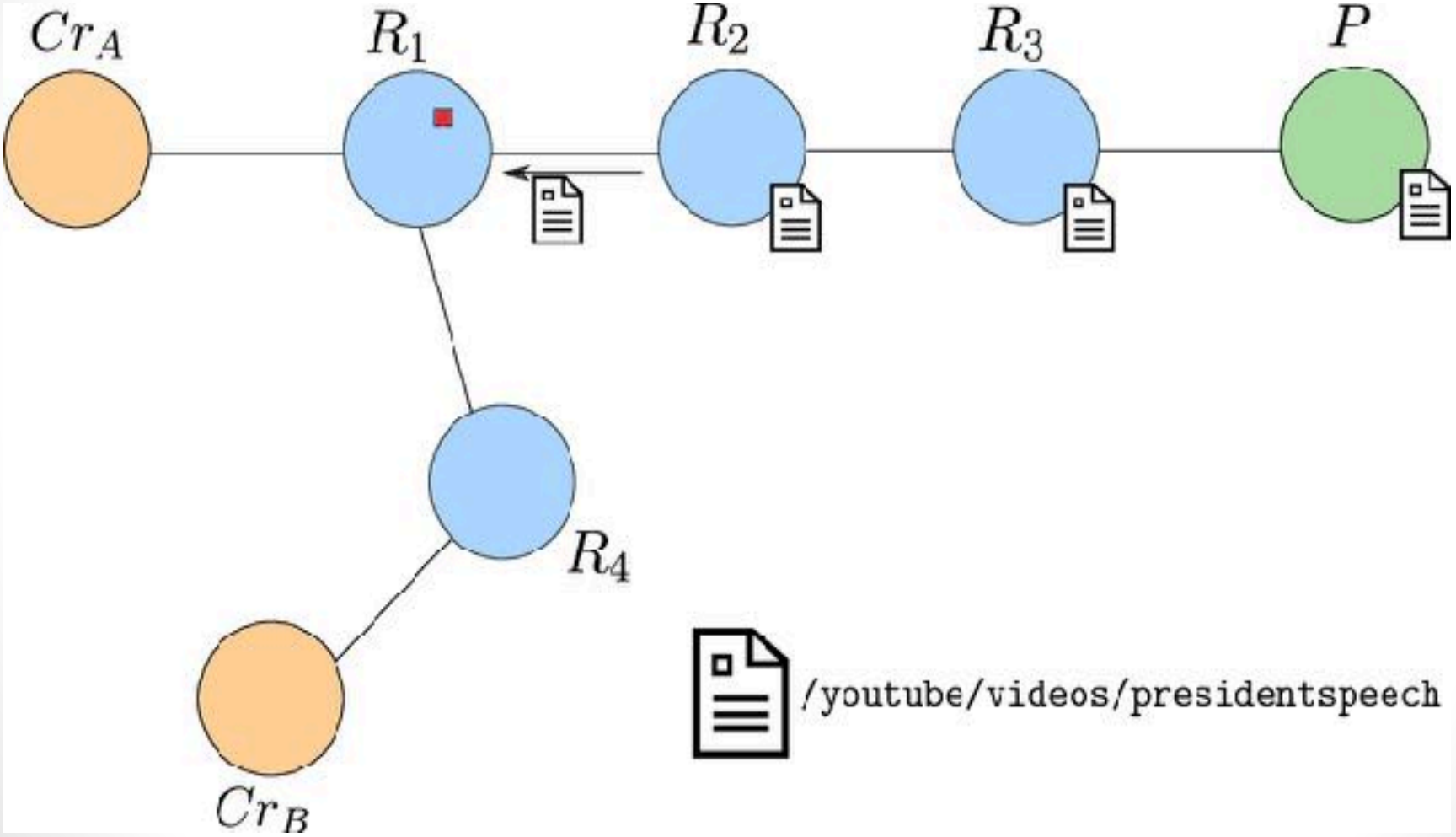
Example



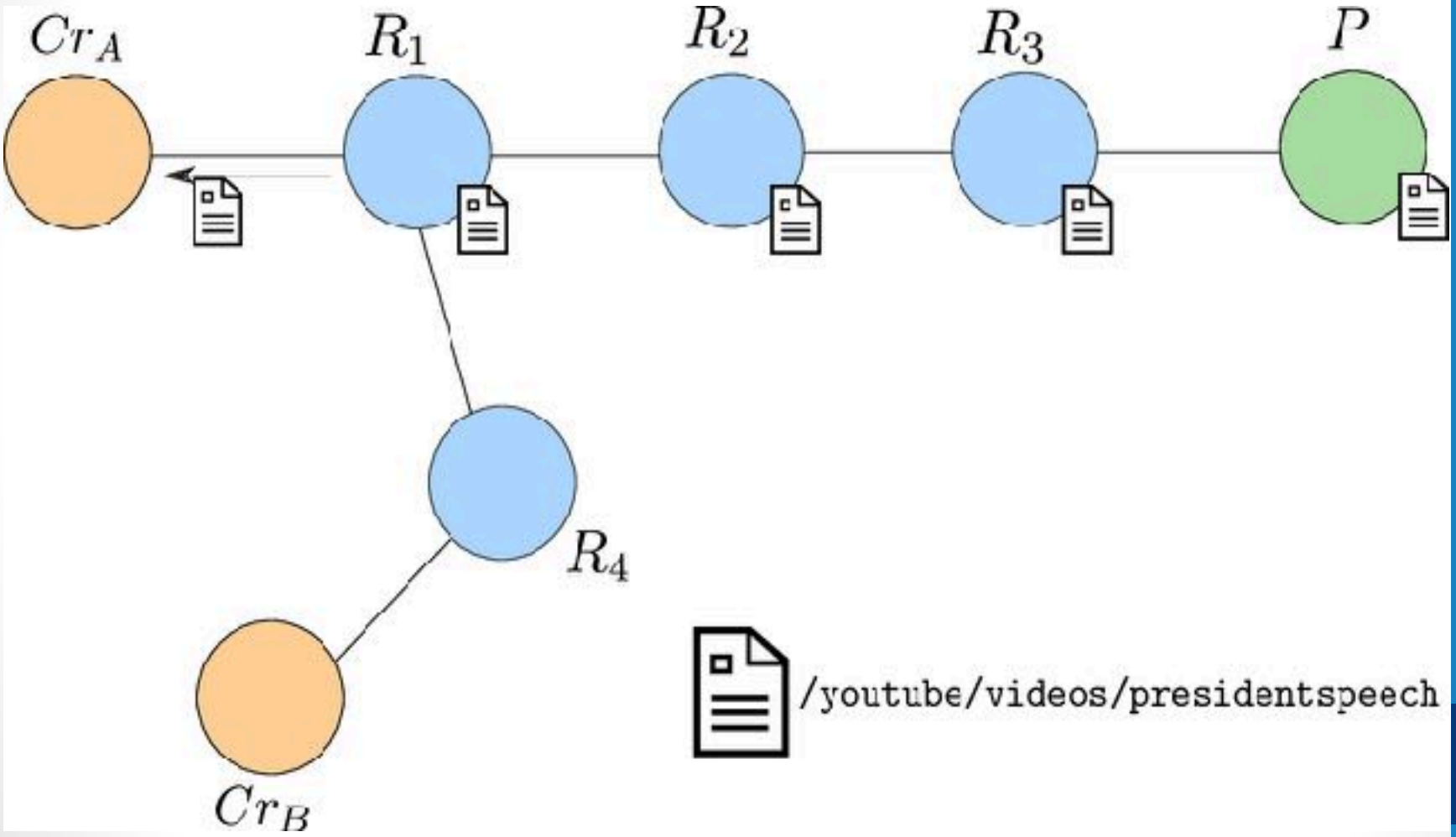
Example



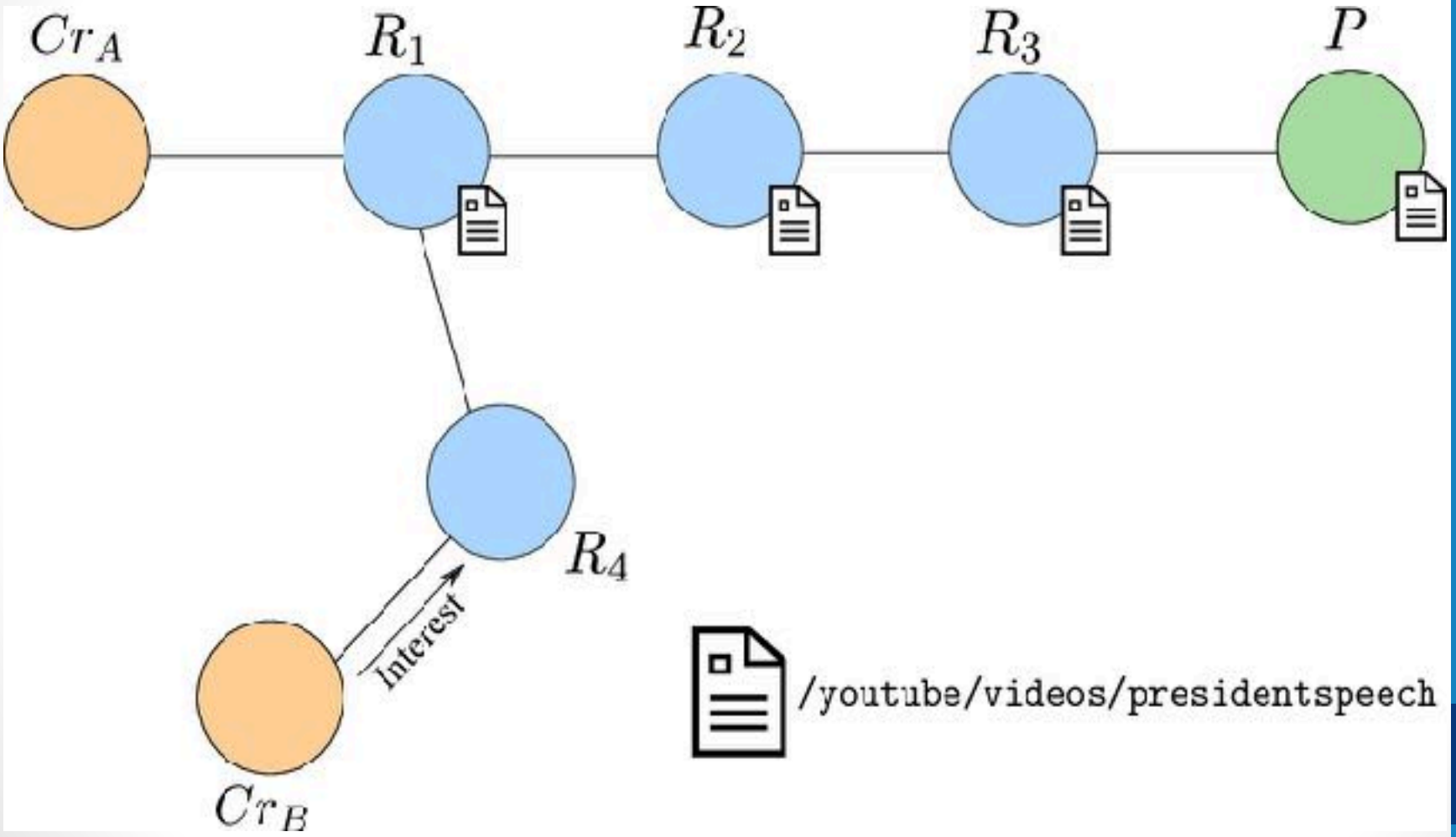
Example



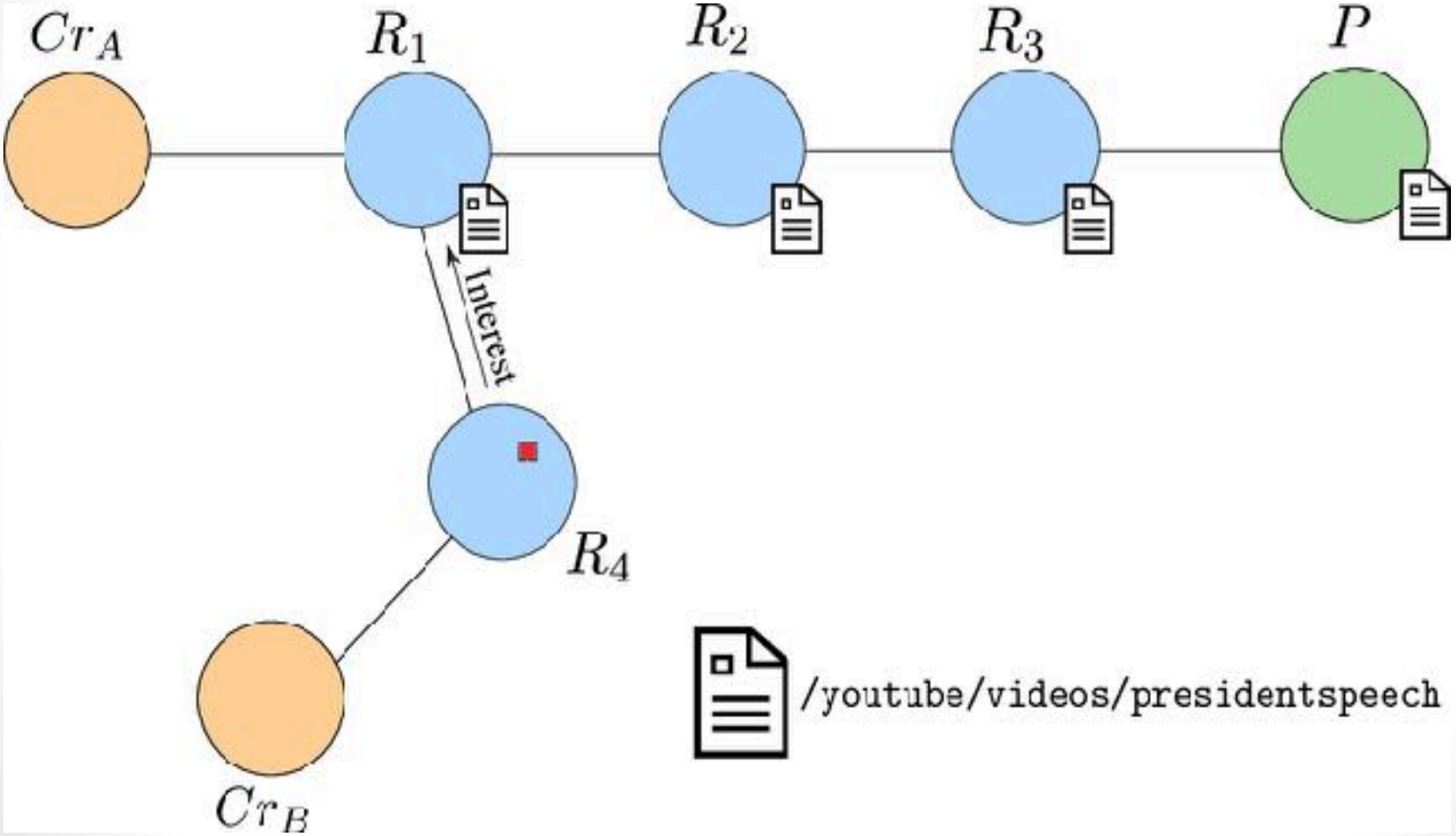
Example



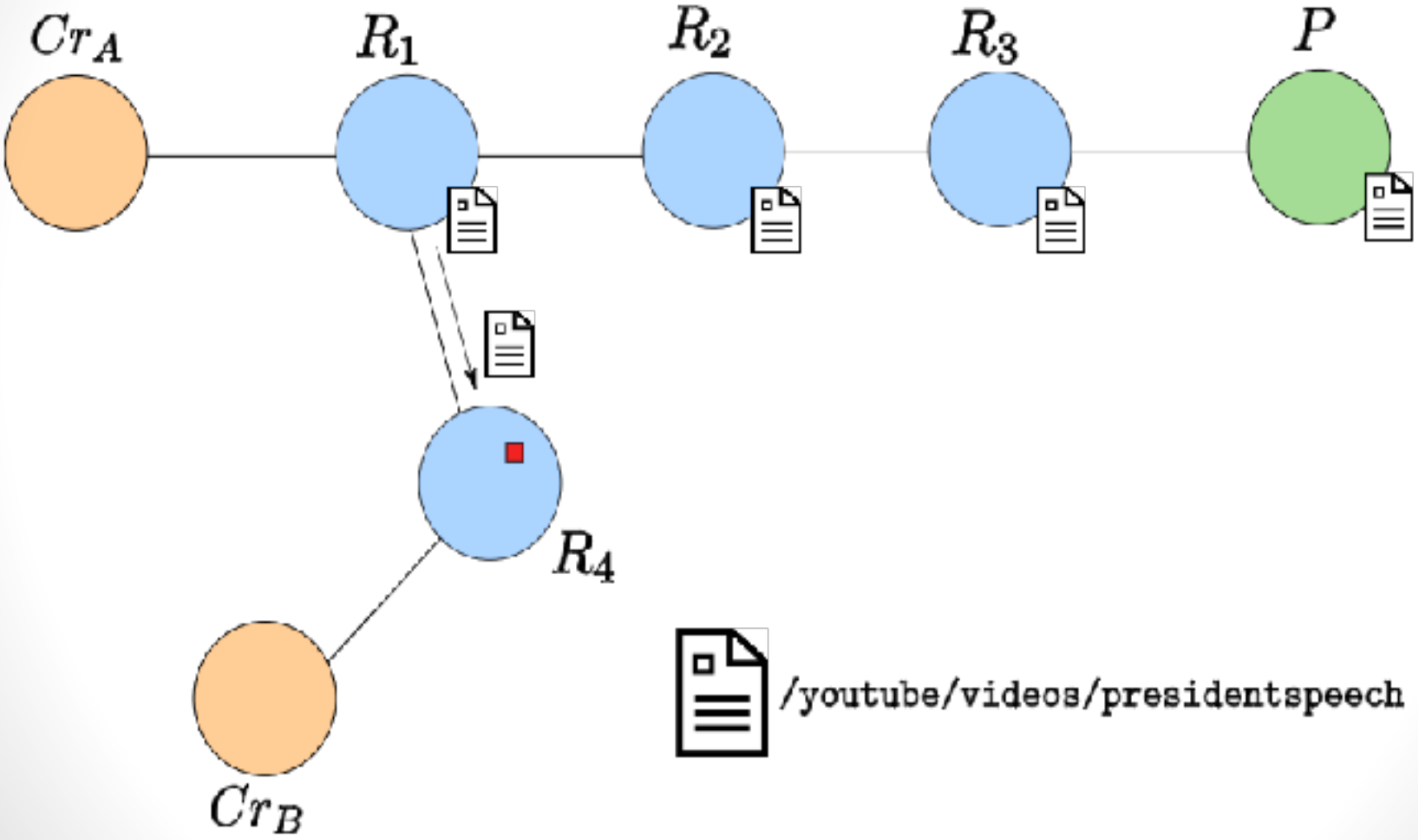
Example


Cr_A
R_1
R_2
R_3
P
R_4
Interest
Cr_B
/youtube/videos/presidentspeech

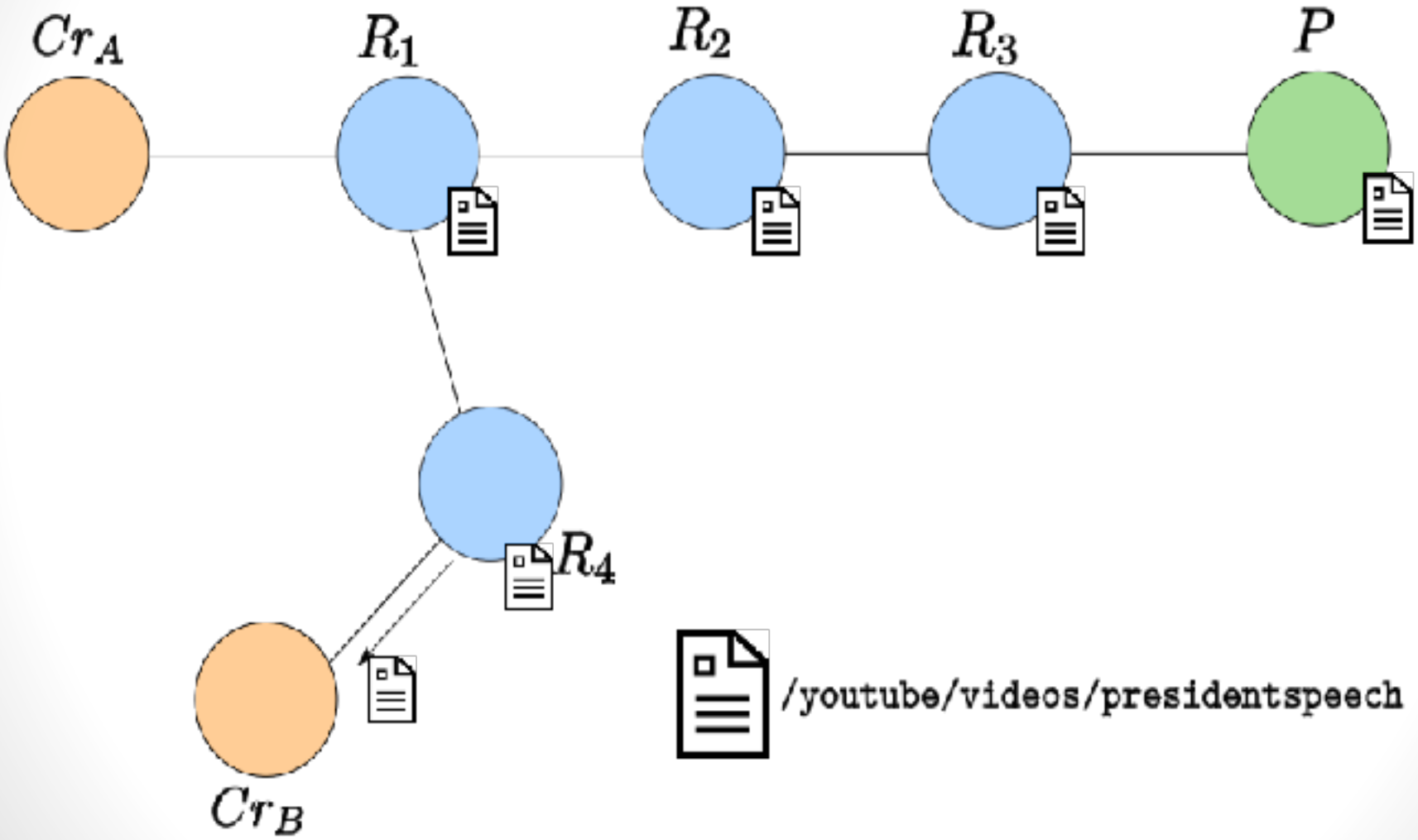
Example


Cr_A
R_1
R_2
R_3
P
Interest
R_4
Cr_B
/youtube/videos/presidentspeech

Example



Example



Data Decoupling

- Data is decoupled from its origin
- Routers may cache content to satisfy future duplicate requests
- Benefits:
  - Upstream congestion and bandwidth utilization is reduced via caching
  - Consumer latency is reduced
  - QoS improves (theoretically)
- Drawbacks:
  - Content must be digitally signed (or verified by some other means) to ensure authenticity
  - **Producers have no knowledge about where content is cached**

Types of Content

Type	Example	Cacheable?	Rate of Change?
Static	Media file	Yes	Infrequent
Dynamic	VoIP traffic	No (except for retransmissions)	Frequent
Event-driven	Breaking news	Yes	**Unpredictable**

Main Question

How can we cache event-driven content while preventing consumers from obtaining stale (out-of-date) content?

Related Work*

- Dynamically reduce the cache lifetime of content as it spreads further away from the producer
  - Helps, but doesn't mitigate the problem
- Proactive content sharing between peering routers
  - No discussion of deletion or expiration of content
- Revoke consumer access from content via online OCSP-like protocols
  - Revocation != deletion

*References are contained in the paper.

Our Answer

We need a way to **flush or erase** content from router caches **on demand**

Erasure Requirements

1. Erase messages must be authenticated
2. Erase messages must be distributed to all routers which may have potentially cached the content

Erasure Requirements

1. Erase messages must be authenticated (**easy**)
2. Erase messages must be distributed to all routers which may have potentially cached the content (**hard**)

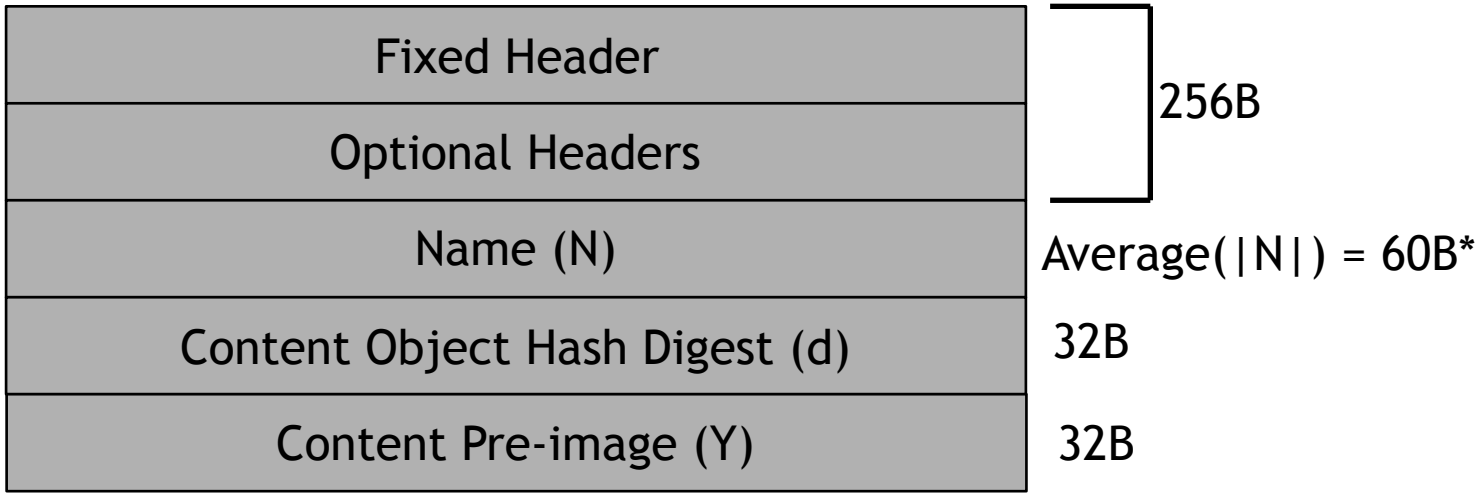
Authenticating Erase Messages

- Several things to consider:
  - Authentication should not cause a DoS on routers
  - Deletion is an idempotent operation

Authenticating Erase Messages

- Several things to consider:
  - Authentication should not cause a DoS on routers
  - Deletion is an idempotent operation
- Solution: use pre-image resistant hash functions
  1. Generate $X \leftarrow \{0,1\}^{\lambda}$
  2. Compute $Y = H(X)$
  3. Distribute Y with published content
  4. Publish X with a erase message to prove ownership of content
  5. If $Y = H(X)$ then the producer must have issued the request

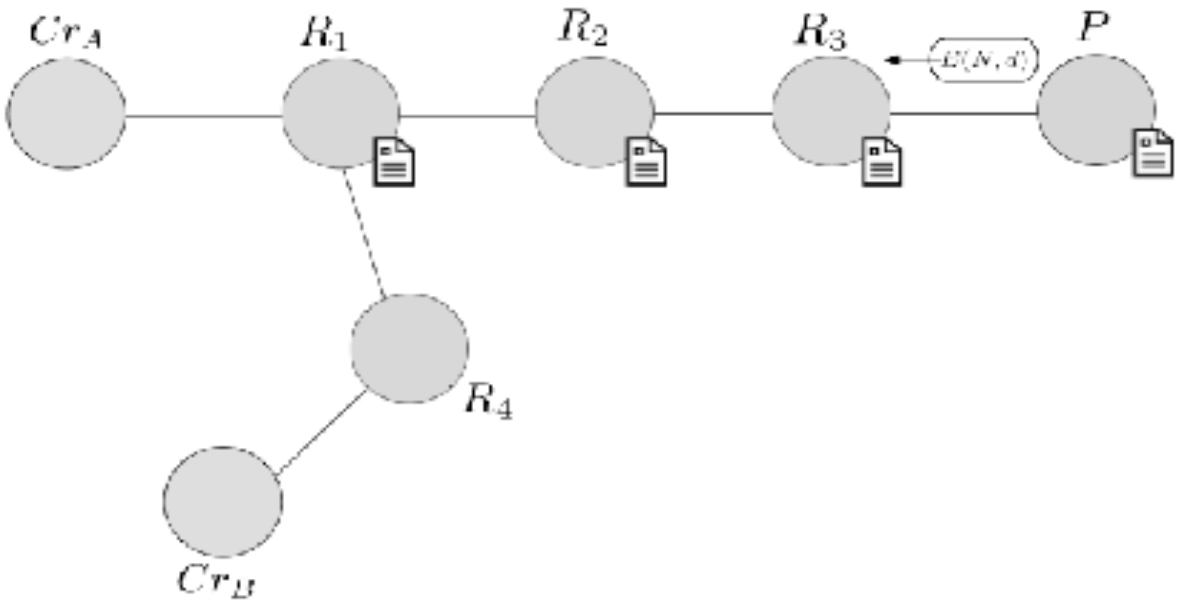
Erase Message Format



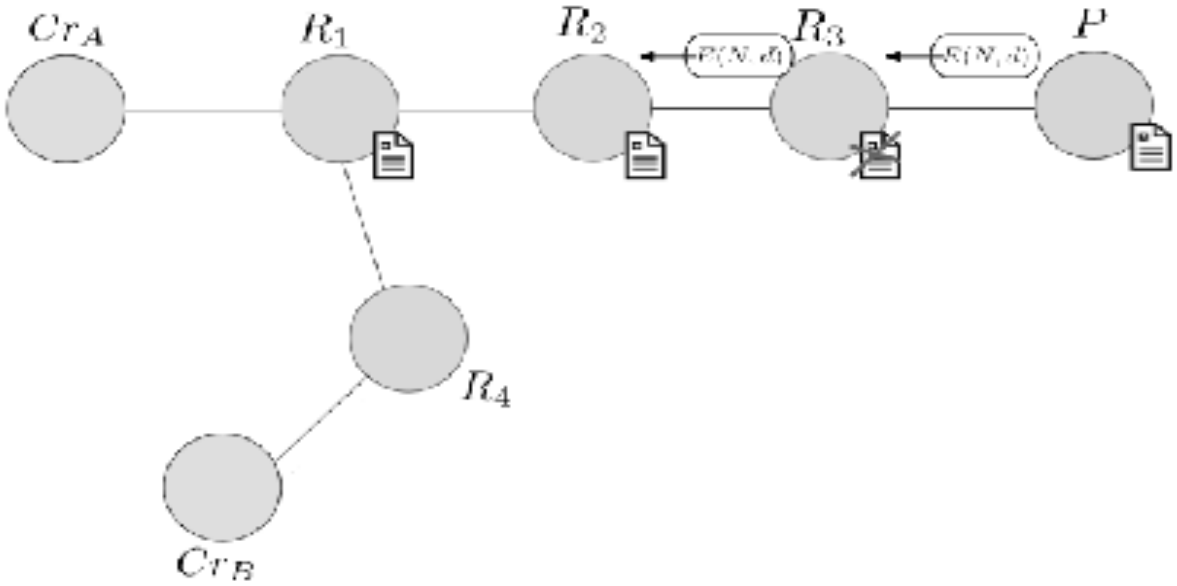
Notation: E(N,d)

*Based on unibas dataset from http://www.icn-names.net/

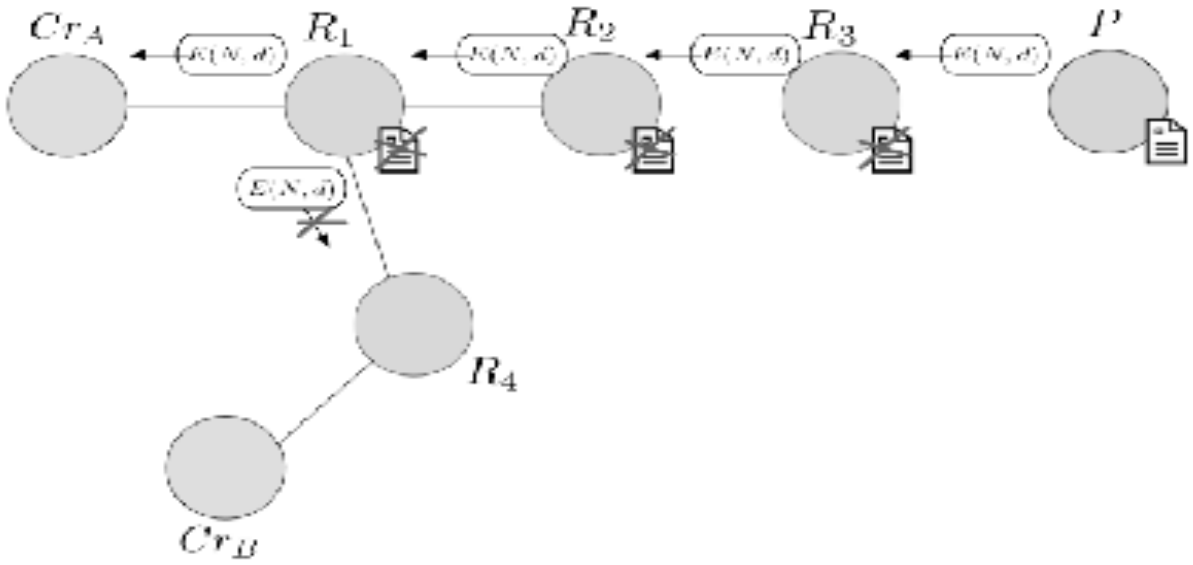
Distributing Erase Messages



Distributing Erase Messages



Distributing Erase Messages

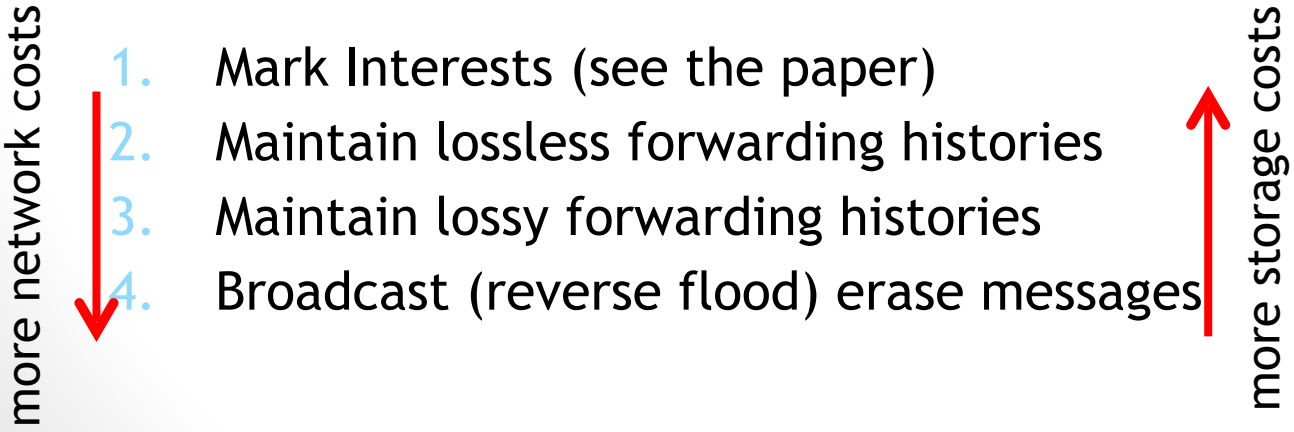


Distributing Erase Messages

- Several things to consider:
  - Producers have no knowledge about where content may be stored
  - Not all routers may cache content

Distributing Erase Messages

- Several things to consider:
  - Producers have no knowledge about where content may be stored
  - Not all routers may cache content
- Our approach: a hierarchy of distribution mechanisms



2) Lossless Histories

- Idea: remember where content was forwarded
- Implementation:
  - Caches maintain records of interfaces to which content was sent
  - When flushed, forwarding history is written to a log
- Protocol:
  - When E(N,d) arrives, search the cache for the content with digest d
  - If present, forward E(n,d) to recorded interfaces and flush the match
  - Else, search the log for d
    - If present, forward E(n,d) to the recorded interfaces
    - Else, drop E(n,d)

2) Lossless Histories

- Idea: remember where content was forwarded
- Implementation:
  - Caches maintain records of interfaces to which content was sent
  - When flushed, forwarding history is written to a log
- Protocol:
  - When E(N,d) arrives, search the cache for the content with digest d
  - If present, forward E(n,d) to recorded interfaces and flush the match
  - Else, search the log for d
    - If present, forward E(n,d) to the recorded interfaces
    - Else, drop E(n,d)

Observation: if all routers have lossless histories, this gives complete DFS covera

Lossless History Analysis

- Consumer-facing router:
  - 4GB of history storage
  - 100Mbps forwarding rate (3’200 Cps*)
  - Saturation in **~12 hours**
- Core router:
  - 1TB of flash history storage
  - 10Tbps forwarding rate (335 MCps)
  - Saturation in **~102s (ouch!)**

*assuming content objects are 4KB in size

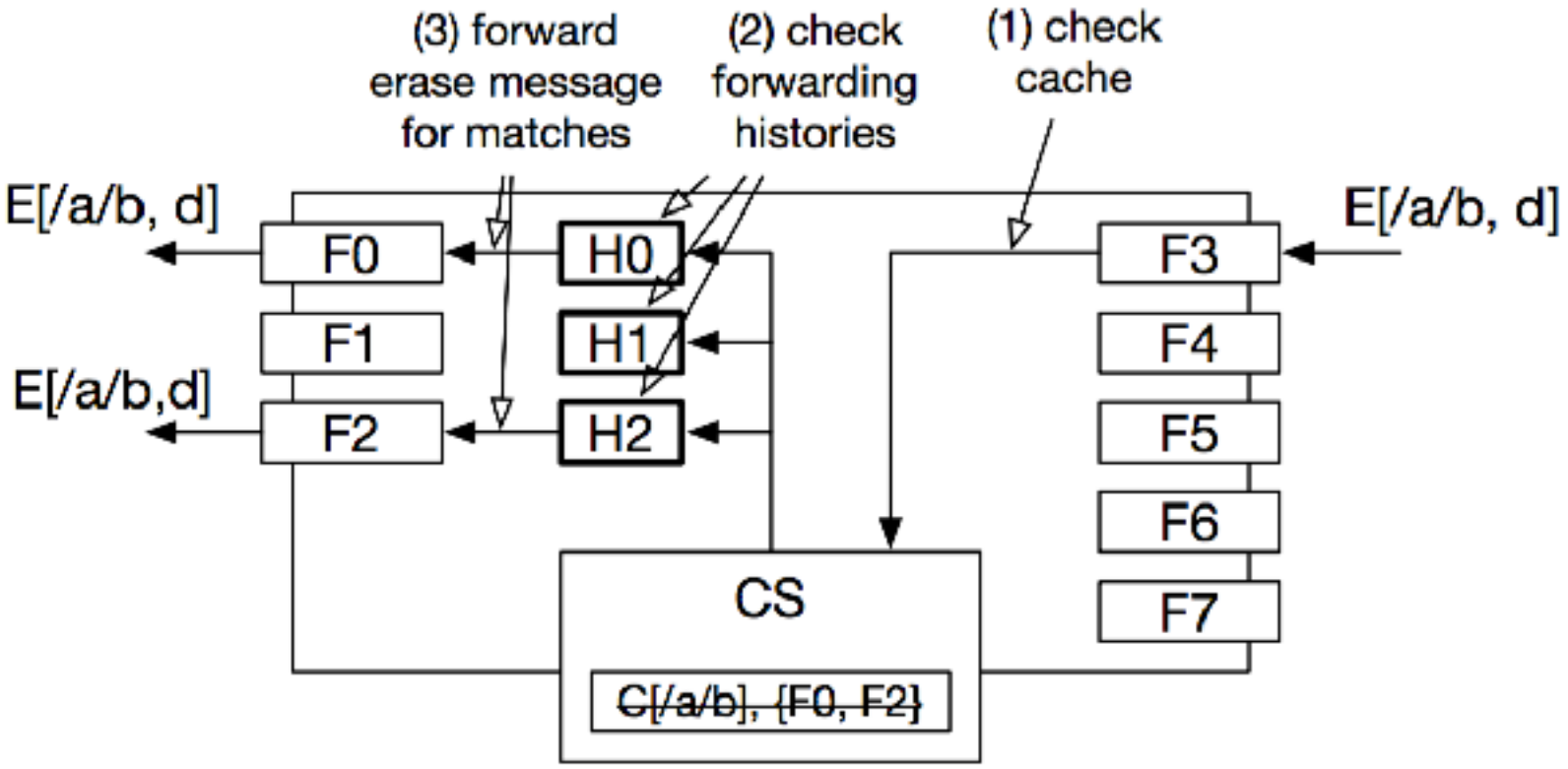
3) Lossy Histories

- Idea: remember where content was forwarded **without storing the content**
- Implementation:
  - Caches maintain records of interfaces to which content was sent
  - Maintain a Bloom Filter for each outgoing interface
  - Whenever a content is flushed, add its hash digest to each interface filter to which it was forwarded
- Protocol:
  - When E(N,d) arrives, search the cache for the content with digest d
  - If present, forward E(n,d) to recorded interfaces and flush the match
  - Check each interface filter for d
  - For each match, forward E(n,d) to the interface

3) Lossy Histories

- Idea: remember where content was forwarded **without storing the content**
- Implementation:
  - Caches maintain records of interfaces to which content was sent
  - Maintain a Bloom Filter for each outgoing interface
  - Whenever a content is flushed, add its hash digest to each interface filter to which it was forwarded
- Protocol:
  - When E(N,d) arrives, search the cache for the content with digest d
  - If present, forward E(n,d) to recorded interfaces and flush the match
  - Check each interface filter for d
  - For each match, forward E(n,d) to the interface

Observation: enables correct DFS coverage but can induce excess traffic due to false positive nature of BF


(3) forward
erase message
for matches
(2) check
forwarding
histories
(1) check
cache
E[/a/b, d]
E[/a/b, d]
E[/a/b,d]
F0
F1
F2
H0
H1
H2
F3
F4
F5
F6
F7
CS
C[/a/b], {F0, F2}

Lossy History Analysis

- Consumer-facing router:
  - False probability ceiling of 10^{-32} → at most 2×10^8 entries
  - K = 128 distinct hash functions must be used
  - Saturation in **~1 day**
- Core router:
  - False probability ceiling of 10^{-32} → at most 5.7×10^8 entries
  - K = 107 hash functions must be used
  - Saturation in **~245s (ouch!)**

4) Broadcast

- Implementation:
  - Nothing.
- Protocol:
  - If neither of the previous approaches are supported, and if the AS allows it, flood E(n,d) to all interfaces

4) Broadcast

- Implementation:
  - Nothing.
- Protocol:
  - If neither of the previous approaches are supported, and if the AS allows it, flood E(n,d) to all interfaces

Observation: use when there is no cache and in controlled ASs

Overall Recommendations

1. If interest marking is supported, use it.
2. If not, and the content is cached, forward based on the cache entry interface list.
3. If the content is not cached but there is a history, forward based on the history.
4. Otherwise, flood erase messages.*

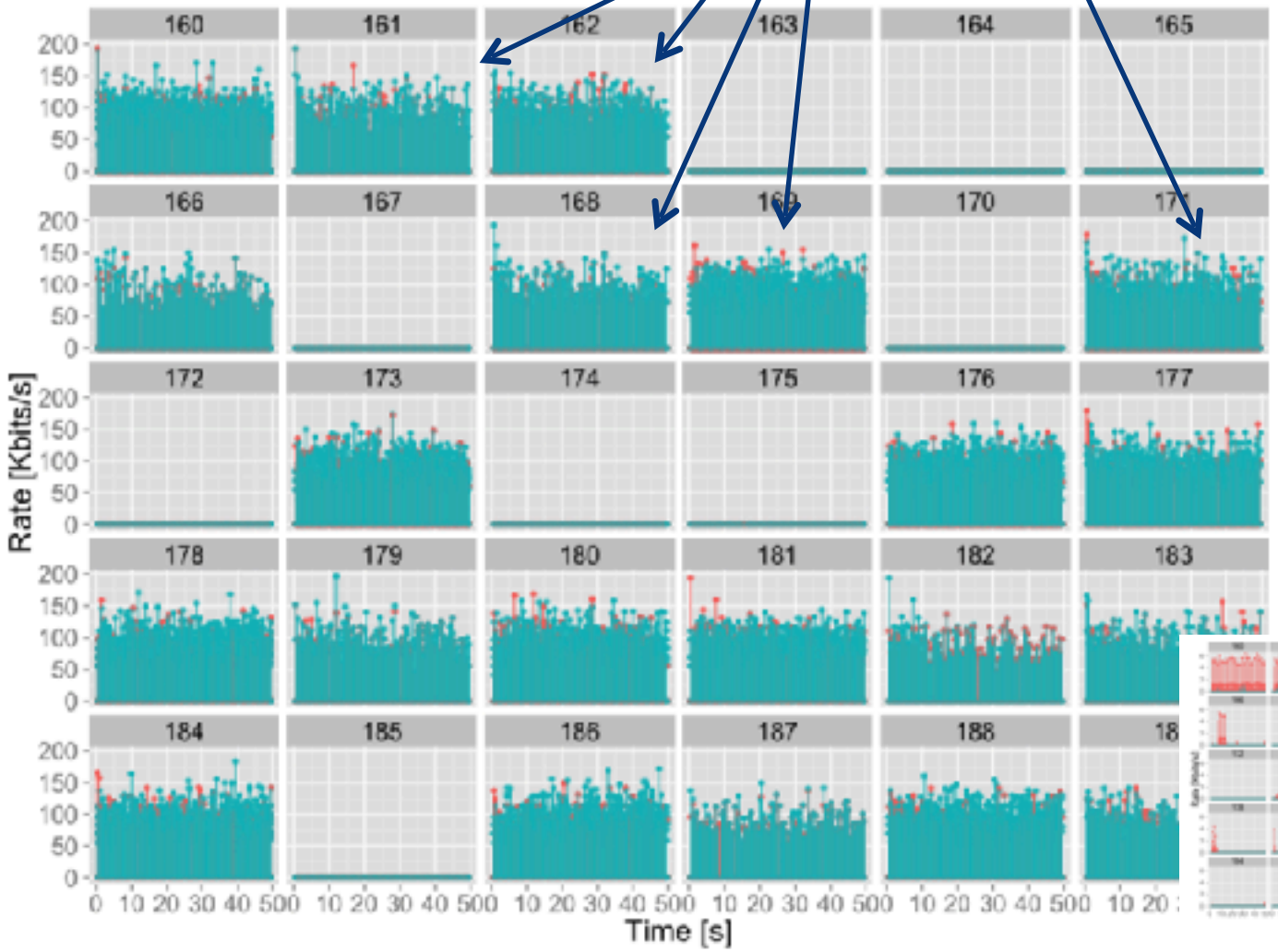
*Justification: on-demand deletion can be seen as a feature that allowed producers must **pay** for

Experiment

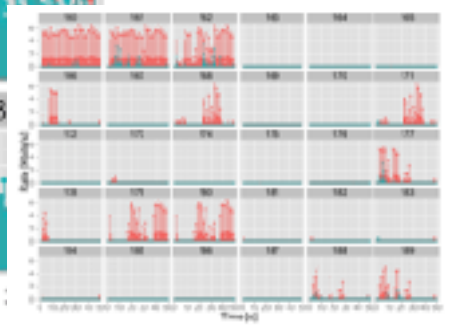
- Goal: determine the router and network overhead
- Setup:
  - Consumers requesting at 10 interest/second
  - Producers deleting a random 50% of their (previously sent) content
- Topology: large DFN and ATT topologies

Results (snippet)

Routers under heavy content load

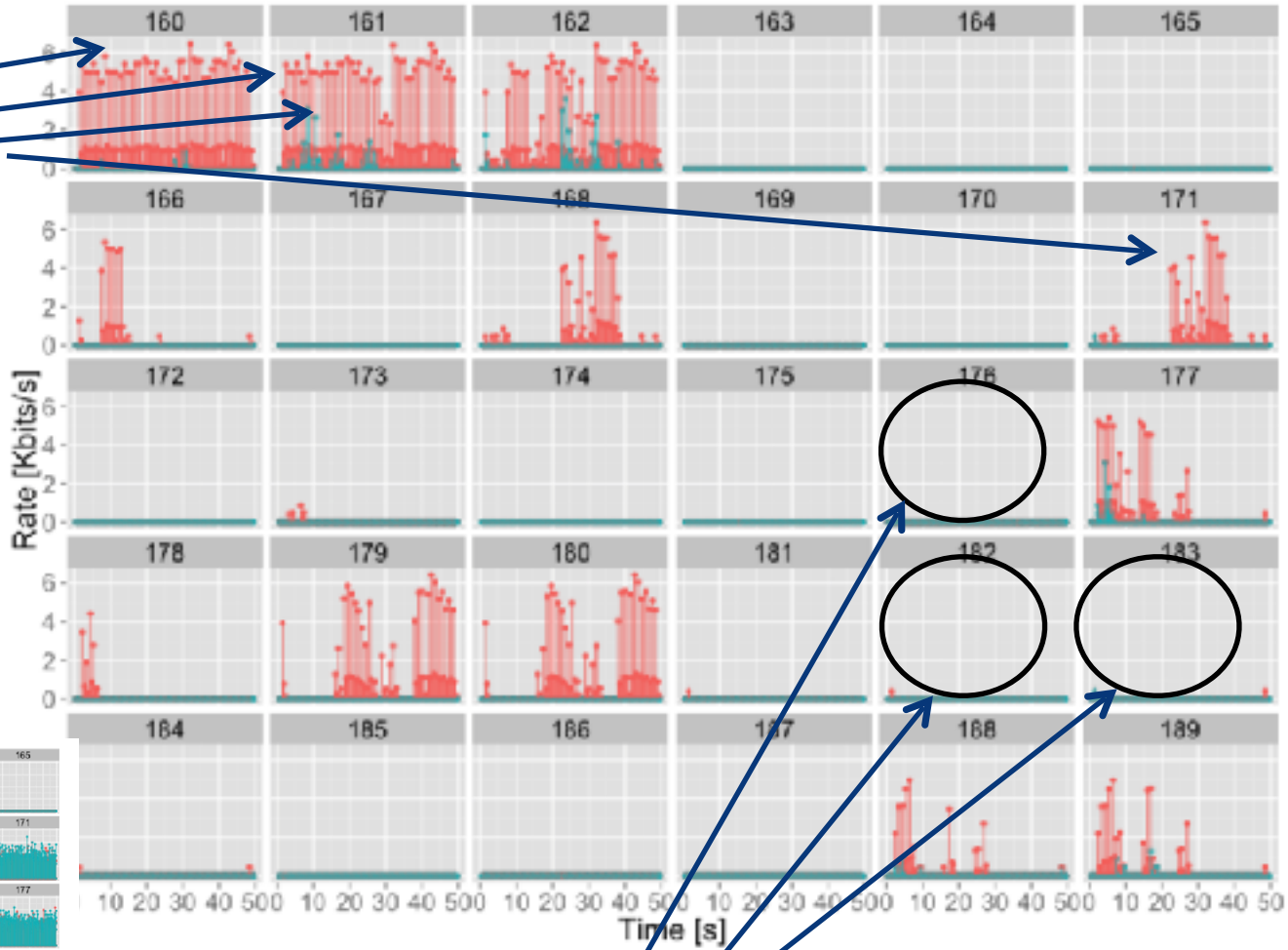


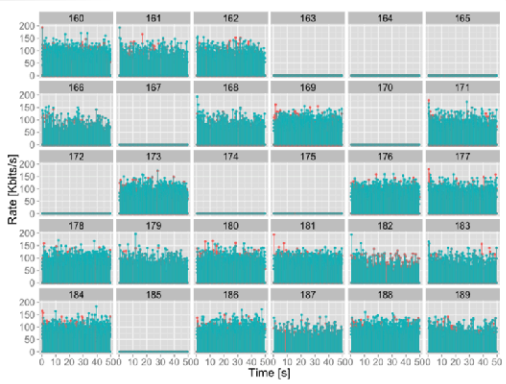
- Each cell is a router
- X-axis is simulation time
- Y-axis is forwarding rate



Results (snippet)

Routers forwarding erase messages (lower output throughput*)





*Producers may send duplicate BEADs (which are dropped at routers)

Conclusion

- Showed how to support on-demand deletion of content in CCN by…
  - Authenticating erase messages
  - Distributing erase messages to candidate routers
- Analyzed the overhead and efficacy of each distribution mechanism
- Experimentally assessed the best-case scenario

Future Work

- Continue the experimental assessment of BEAD
- Improve the interest marking mechanism so that it's private
- Combine BEAD with a secure accounting protocol for "premium CDN overlays in CCN"