Interest-Based Access Control in CCN

Cesar Ghali, Marc A. Schlosberg, Gene Tsudik, **Christopher A. Wood**
University of California Irvine

{cghali, marc.schlosberg, gene.tsudik,
woodc1}@uci.edu

August 10, 2015

Agenda

1. Introduction and Access Control Overview
   - Encryption-based AC
   - Interest-based AC
2. IBAC Security Model
3. IBAC via Name Obfuscation
   - Encryption-based Obfuscation
   - Hash-based Obfuscation
4. Security Considerations
   - Replay Attacks
   - Authorized Key-Binding Rule
5. Conclusions and Recommendations

The Tenents of CCN

- Content is named and transferred through the network from producers to consumers
- *Any consumer* can ask for content provided its name
- Producers are considered responsible enforcing access control to content object data.

The Access Control Problem

Question: How can we ensure that only *authorized users* are able to access (the body of) a content object?

1. Encrypt the payload of a content object, give decryption keys to authorized users
2. Require that all interests are forwarded to the producer for inspection and authorization checks
   - Invalidates caches...
3. Make interest names only derivable by authorized users
   - Caches are still okay!

The Access Control Problem

Question: How can we ensure that only *authorized users* are able to access (the body of) a content object?

1. Encrypt the payload of a content object, give decryption keys to authorized users
2. Require that all interests are forwarded to the producer for inspection and authorization checks
   - Invalidates caches...
3. Make interest names only derivable by authorized users
   - Caches are still okay!

Access Control Groups

Access Control is based in *groups*, where groups are allowed (or not allowed) access to the content under question.

- N - name of a Content Object
- $\mathbb{U}(N)$ - set of consumers authorized to access (read or use) the content with name N
- $\bar{\mathbb{U}}(N)$ - complement of $\mathbb{U}(N)$

Encryption-based Access Control

Main Idea: If $C \notin \mathbb{U}(N)$, then C should not be able to decrypt the body of a content object.

- A preliminary specification was first introduced in [1]
- Many variations based on different public-key cryptographic algorithms have been proposed:
  - Broadcast-based encryption [2]
  - Attribute-based encryption [3]
  - Proxy-based encryption [4]
  - CCN-AC (a general framework for all of the above) [5]

1 Smetters, Diana, Philippe Golle, and Jim Thornton. CCNx Access Control Specifications. Technical report, PARC, 2010.

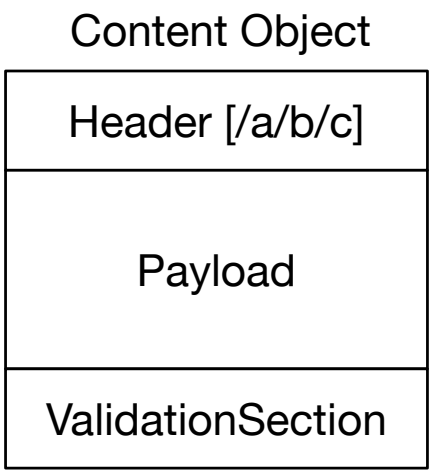
2 Misra, Satyajayant, Reza Tourani, and Nahid Ebrahimi Majd. “Secure Dontent Delivery in Information-Centric Networks: Design, Implementation, and Analyses.” Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking. ACM, 2013.

3 Ion, Mihaela, Jianqing Zhang, and Eve M. Schooler. “Toward Content-Centric Privacy in ICN: Attribute-Based Encryption and Routing.” ACM SIGCOMM Computer Communication Review. Vol. 43. No. 4. ACM, 2013.

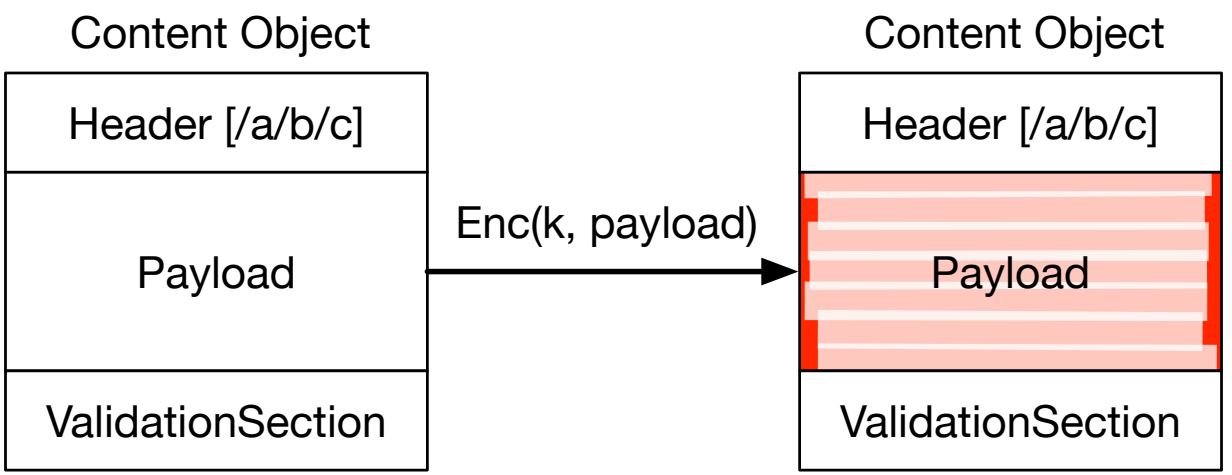
4 Wood, Christopher, and Ersin Uzun. “Flexible End-to-End Content Security in CCN.” Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th. IEEE, 2014.

5 Kurihara, Jun, C. Wood, and Ersin Uzun. “An Encryption-Based Access Control Framework for Content-Centric Networking.” IFIP, 2015.

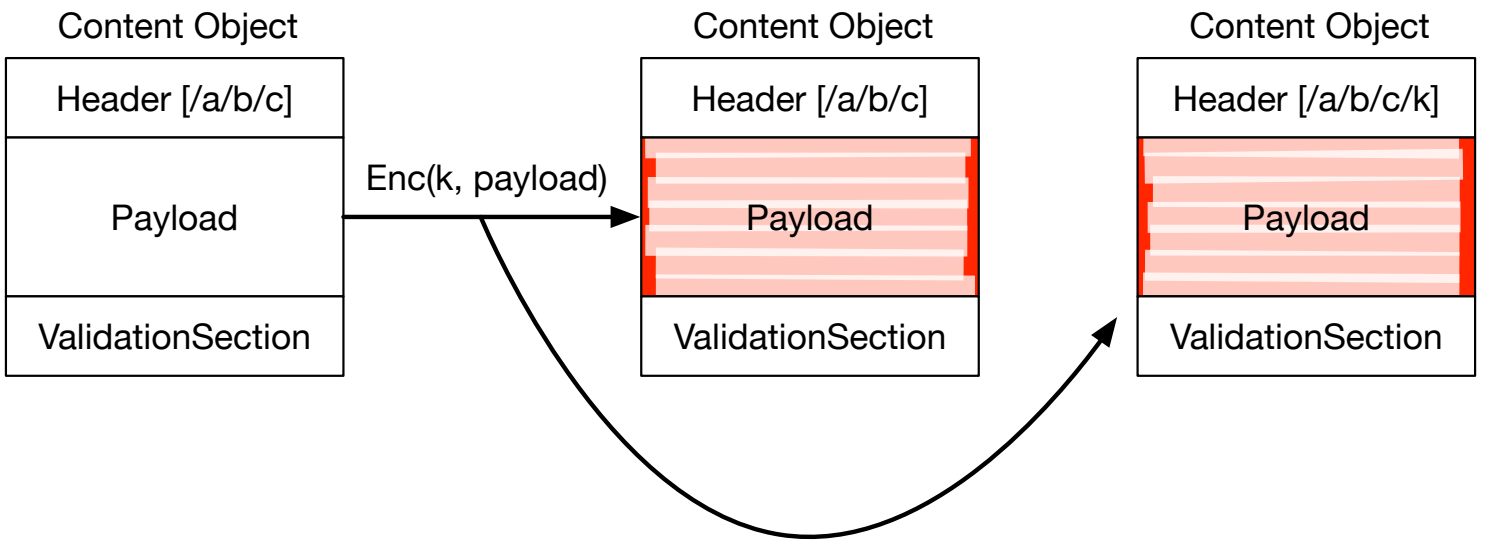
Encryption-based AC in Pictures



Encryption-based AC in Pictures (cont'd)



Encryption-based AC in Pictures (cont'd)



Interest-based Access Control

Main Idea: If $Cr \notin \mathbb{U}(N)$, then Cr should not be able to construct a "correct" interest for it.

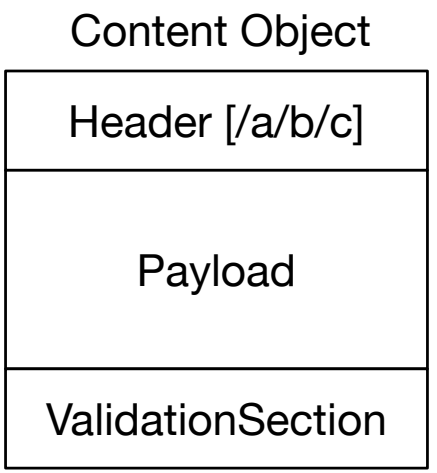
Implication: Interest names should depend on some secret that only authorized consumers know.

Interest-based Access Control

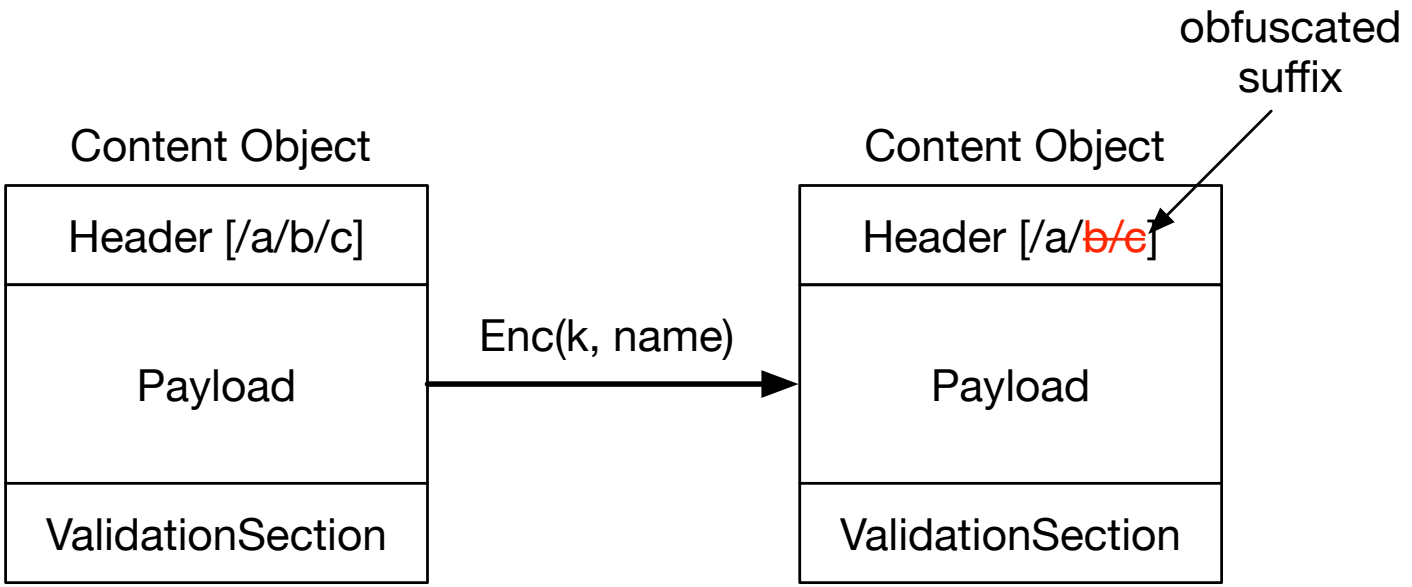
Main Idea: If $Cr \notin \mathbb{U}(N)$, then Cr should not be able to construct a "correct" interest for it.

Implication: Interest names should depend on some secret that only authorized consumers know.

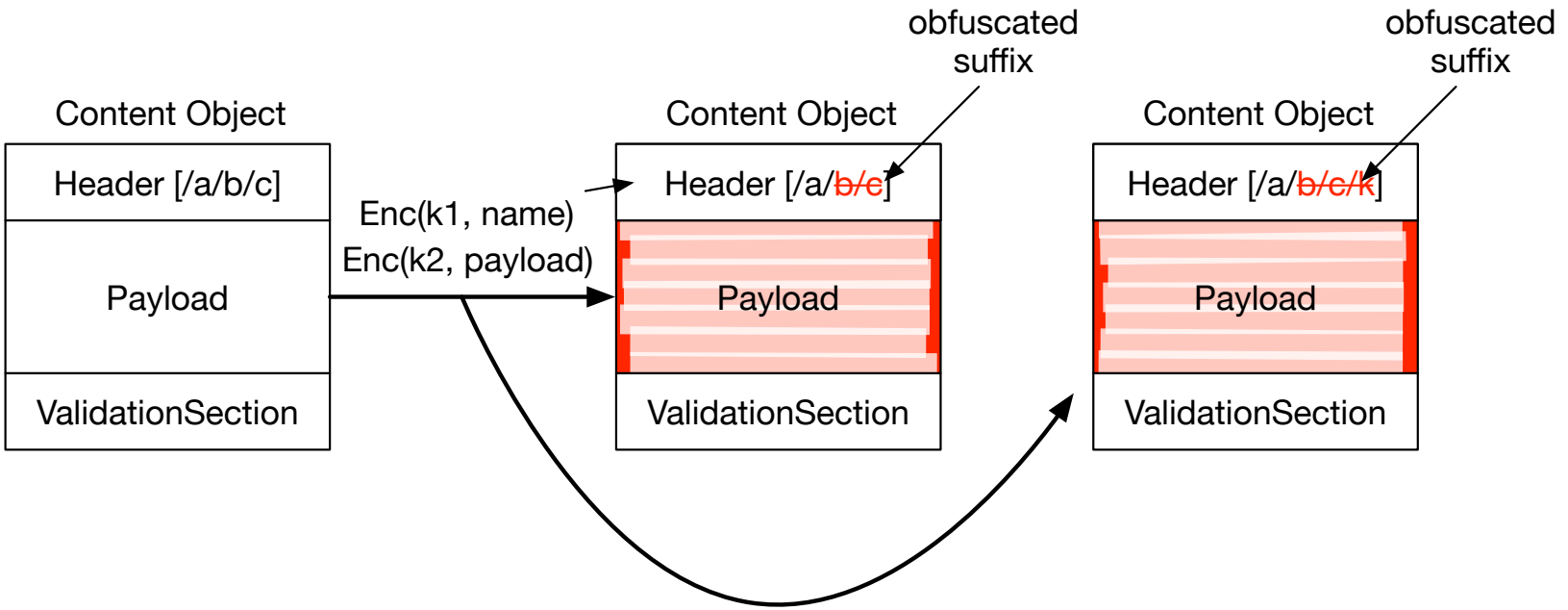
Interest-based AC in Pictures



Interest-based AC in Pictures (cont'd)



Both Dimensions of AC



Security Model

Recall that IBAC is about encrypting the name (though the payload may also be encrypted)...

Let $\text{Path}(Cr, P)$ be the set of all routers on the path between the consumer $Cr \in \mathbb{U}(N)$ and P.

We assume an adversary Adv who can deploy and compromise any unauthorized consumer or any router $R \notin \text{Path}(Cr, P)$.

- On-path adversaries can see the names!
- ... but, we will consider on-path adversaries and replay attacks later

Security Model

Recall that IBAC is about encrypting the name (though the payload may also be encrypted)...

Let $\text{Path}(Cr, P)$ be the set of all routers on the path between the consumer $Cr \in \mathbb{U}(N)$ and P.

We assume an adversary Adv who can deploy and compromise any unauthorized consumer or any router $R \notin \text{Path}(Cr, P)$.

- On-path adversaries can see the names!
- ... but, we will consider on-path adversaries and replay attacks later

Security Model

Recall that IBAC is about encrypting the name (though the payload may also be encrypted)...

Let Path(Cr, P) be the set of all routers on the path between the consumer $Cr \in \mathbb{U}(N)$ and P.

We assume an adversary Adv who can deploy and compromise any unauthorized consumer or any router $R \notin$ Path(Cr, P).

- On-path adversaries can see the names!
- ... but, we will consider on-path adversaries and replay attacks later

IBAC via Name Obfuscation

The goal of IBAC is to make the name N of a content object *unguessable* by unauthorized users, i.e., publish N under the name $N' = f(N)$ for some obfuscation function f.

There are at least two ways to do this:

- Encryption-based obfuscation
- Hash-based obfuscation

Note: the obfuscation function only encrypts the *suffix* of a name – not the routable prefix!

IBAC via Name Obfuscation

The goal of IBAC is to make the name N of a content object *unguessable* by unauthorized users, i.e., publish N under the name $N' = f(N)$ for some obfuscation function f.

There are at least two ways to do this:

- Encryption-based obfuscation
- Hash-based obfuscation

Note: the obfuscation function only encrypts the *suffix* of a name – not the routable prefix!

Encryption-based Obfuscation

$$N' = \mathrm{Enc}(k, N),$$

where k is the private key associated with an authorized user.

Supporting Multiple Groups

Question #1: What if we want group-based access control, i.e., where consumers in the same group deterministically generate the same obfuscated name?

(One) Answer: Consumers in the same group share the same encryption key.

Question #2: How does a producer identify the correct decryption key $k_{\mathbb{G}_i}$ for content?

(One) Answer: Include the group identifier $mathsf ID_{\mathbb{G}_i}$ in the payload of each interest.

$$\mathrm{ID}_{\mathbb{G}_i} = \mathrm{H}(k_{\mathbb{G}_i})$$

Supporting Multiple Groups

Question #1: What if we want group-based access control, i.e., where consumers in the same group deterministically generate the same obfuscated name?

(One) Answer: Consumers in the same group share the same encryption key.

Question #2: How does a producer identify the correct decryption key $k_{\mathbb{G}_i}$ for content?

(One) Answer: Include the group identifier $mathsf ID_{\mathbb{G}_i}$ in the payload of each interest.

$$\mathrm{ID}_{\mathbb{G}_i} = \mathrm{H}(k_{\mathbb{G}_i})$$

Supporting Multiple Groups

Question #1: What if we want group-based access control, i.e., where consumers in the same group deterministically generate the same obfuscated name?

(One) Answer: Consumers in the same group share the same encryption key.

Question #2: How does a producer identify the correct decryption key $k_{\mathbb{G}_i}$ for content?

(One) Answer: Include the group identifier $mathsf ID_{\mathbb{G}_i}$ in the payload of each interest.

$$\mathrm{ID}_{\mathbb{G}_i} = \mathrm{H}(k_{\mathbb{G}_i})$$

Supporting Multiple Groups

Question #1: What if we want group-based access control, i.e., where consumers in the same group deterministically generate the same obfuscated name?

(One) Answer: Consumers in the same group share the same encryption key.

Question #2: How does a producer identify the correct decryption key $k_{\mathbb{G}_i}$ for content?

(One) Answer: Include the group identifier $mathsf\, ID_{\mathbb{G}_i}$ in the payload of each interest.

$$\mathrm{ID}_{\mathbb{G}_i} = \mathrm{H}(k_{\mathbb{G}_i})$$

Supporting Multiple Groups (Cont'd)

Question #3: How can we prevent linkability of different interests with the same $\mathrm{ID}_{\mathbb{G}_i}$?

(One) Answer: Encrypt these identifiers using the publisher's public key pk^P:

$$\mathrm{ID}_{\mathbb{G}_i} = \mathrm{Enc}(pk^P, \mathrm{H}(k_{\mathbb{G}_i}))$$

Supporting Multiple Groups (Cont'd)

Question #3: How can we prevent linkability of different interests with the same $\mathrm{ID}_{\mathbb{G}_i}$?

(One) Answer: Encrypt these identifiers using the publisher's public key pk^P:

$$\mathrm{ID}_{\mathbb{G}_i} = \mathrm{Enc}(pk^P, \mathrm{H}(k_{\mathbb{G}_i}))$$

Hash-based Obfuscation

$$N' = \mathrm{H}(k, N),$$

where k is the same shared group key.

This method introduces more state since a producer must be able to invert H to recover N.

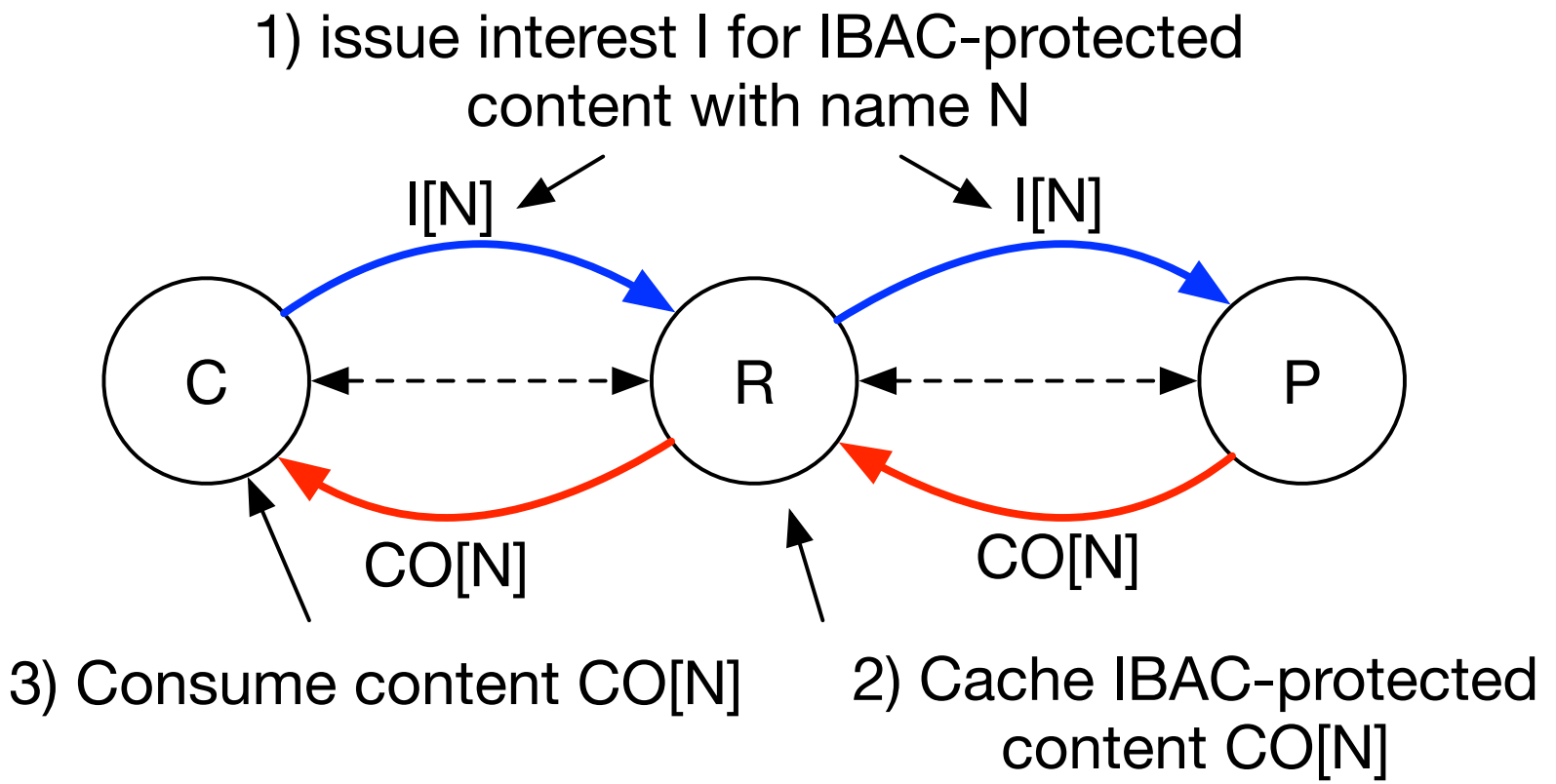
Hash-based Obfuscation

$$N' = \mathrm{H}(k, N),$$

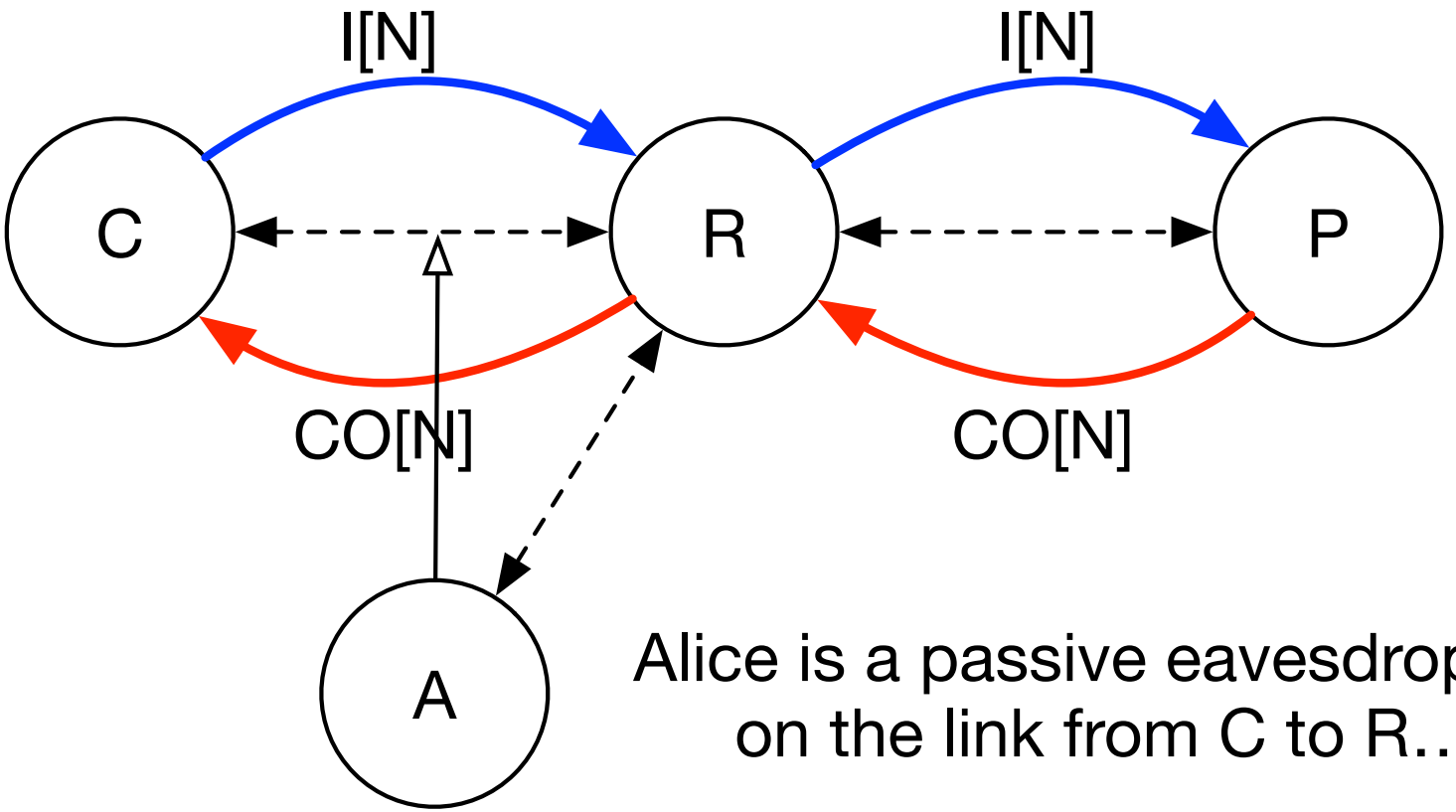
where k is the same shared group key.

This method introduces more state since a producer must be able to invert H to recover N.

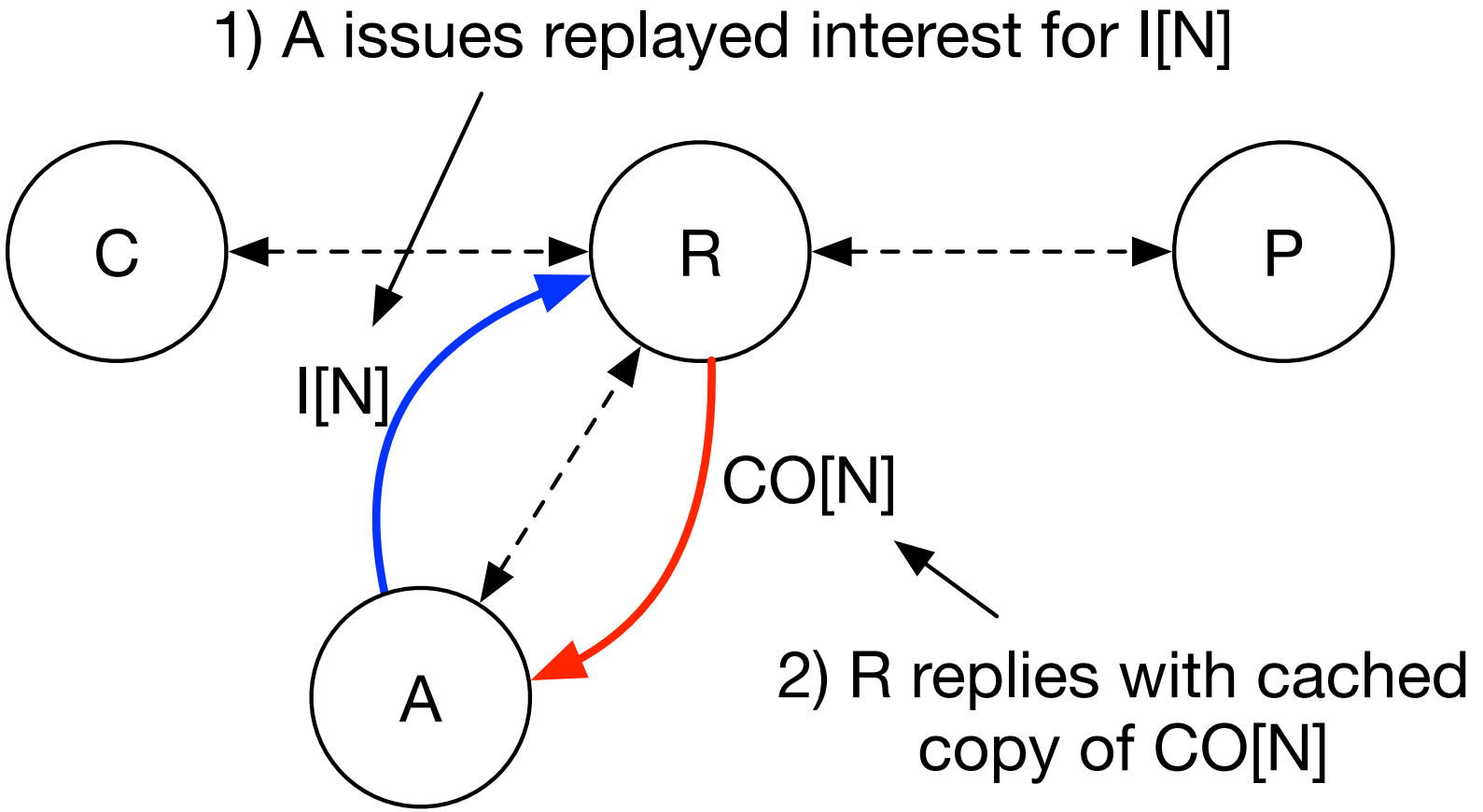
Replay Attacks



Replay Attacks (Cont'd)



Replay Attacks (Cont'd)



Replay Attacks in Detail

Any adversary can observe an obfuscated interest, replay it, and get the same content

We need **replay prevention**:

- Nonces and timestamps help prevent replay attacks
- ... in addition to consumer authentication information

$$\texttt{Payload} = \left(\mathrm{ID}_{\mathbb{G}_i}, r, t, \sigma = \mathrm{Sign}_{sk^s_{\mathbb{G}_i}}\left(N' || \mathrm{ID}_{\mathbb{G}_i} || r || t\right)\right)$$

Replay Attacks in Detail

Any adversary can observe an obfuscated interest, replay it, and get the same content

We need **replay prevention**:

- Nonces and timestamps help prevent replay attacks
- ... in addition to consumer authentication information

$$\texttt{Payload} = \left(\mathrm{ID}_{\mathbb{G}_i}, r, t, \sigma = \mathrm{Sign}_{sk^s_{\mathbb{G}_i}}\left(N' || \mathrm{ID}_{\mathbb{G}_i} || r || t\right)\right)$$

Interest Authentication

Question: How can a router check if a given (cached) content object can be returned in response to an interest?

Answer: Routers will have to verify some authenticator provided in interests (e.g., a digital signature)

Question: How does a router know what key to use for verification?

Answer: Follow the authorized key binding (AKB) rule:

> **ACKB:** Cached content protected under IBAC must reflect the verification key associated with the authorization policy.

Interest Authentication

Question: How can a router check if a given (cached) content object can be returned in response to an interest?

Answer: Routers will have to verify some authenticator provided in interests (e.g., a digital signature)

Question: How does a router know what key to use for verification?

Answer: Follow the authorized key binding (AKB) rule:

ACKB: Cached content protected under IBAC must reflect the verification key associated with the authorization policy.

Interest Authentication

Question: How can a router check if a given (cached) content object can be returned in response to an interest?

Answer: Routers will have to verify some authenticator provided in interests (e.g., a digital signature)

Question: How does a router know what key to use for verification?

Answer: Follow the authorized key binding (AKB) rule:

> **ACKB:** Cached content protected under IBAC must reflect the verification key associated with the authorization policy.

Interest Authentication

Question: How can a router check if a given (cached) content object can be returned in response to an interest?

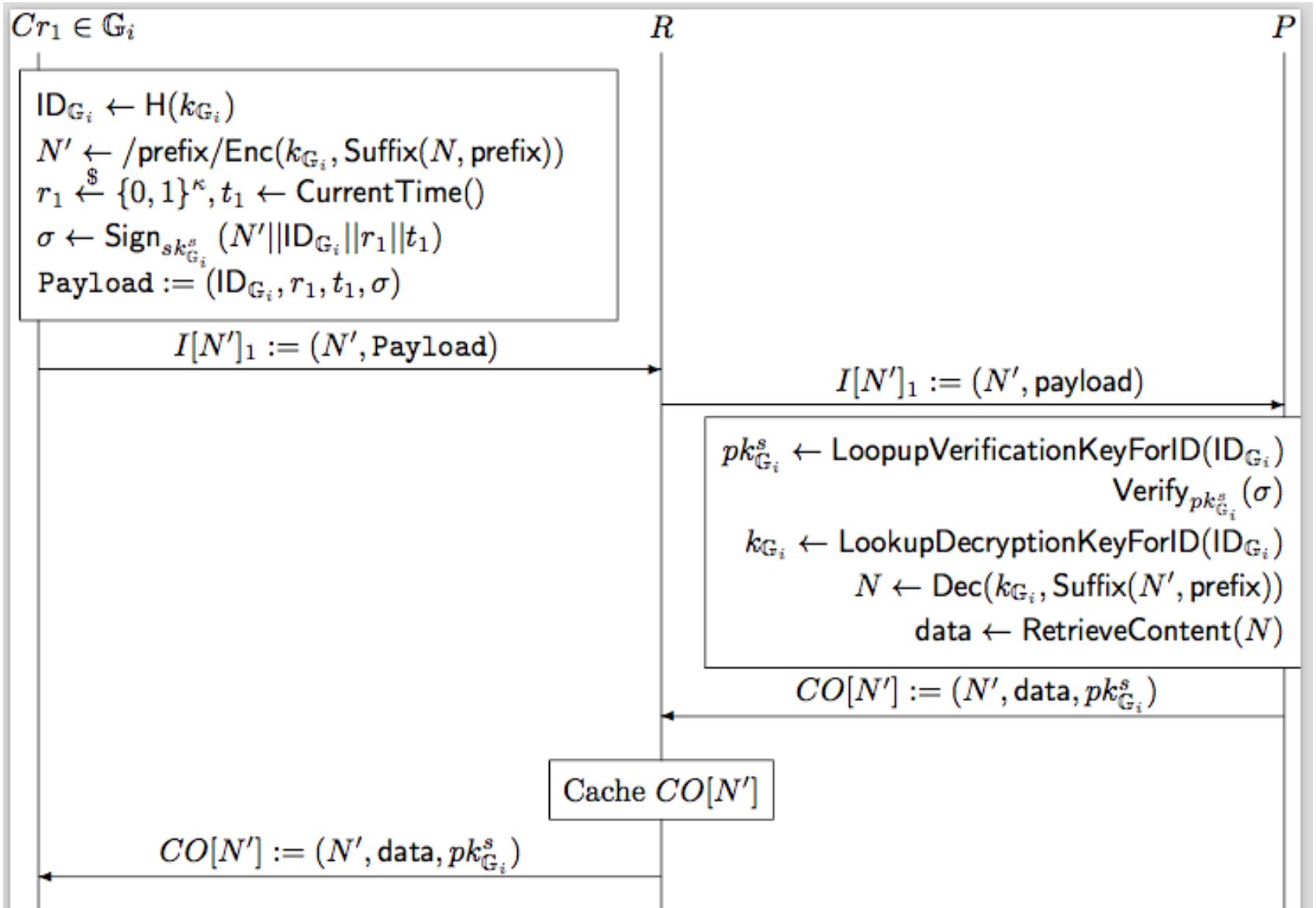
Answer: Routers will have to verify some authenticator provided in interests (e.g., a digital signature)

Question: How does a router know what key to use for verification?

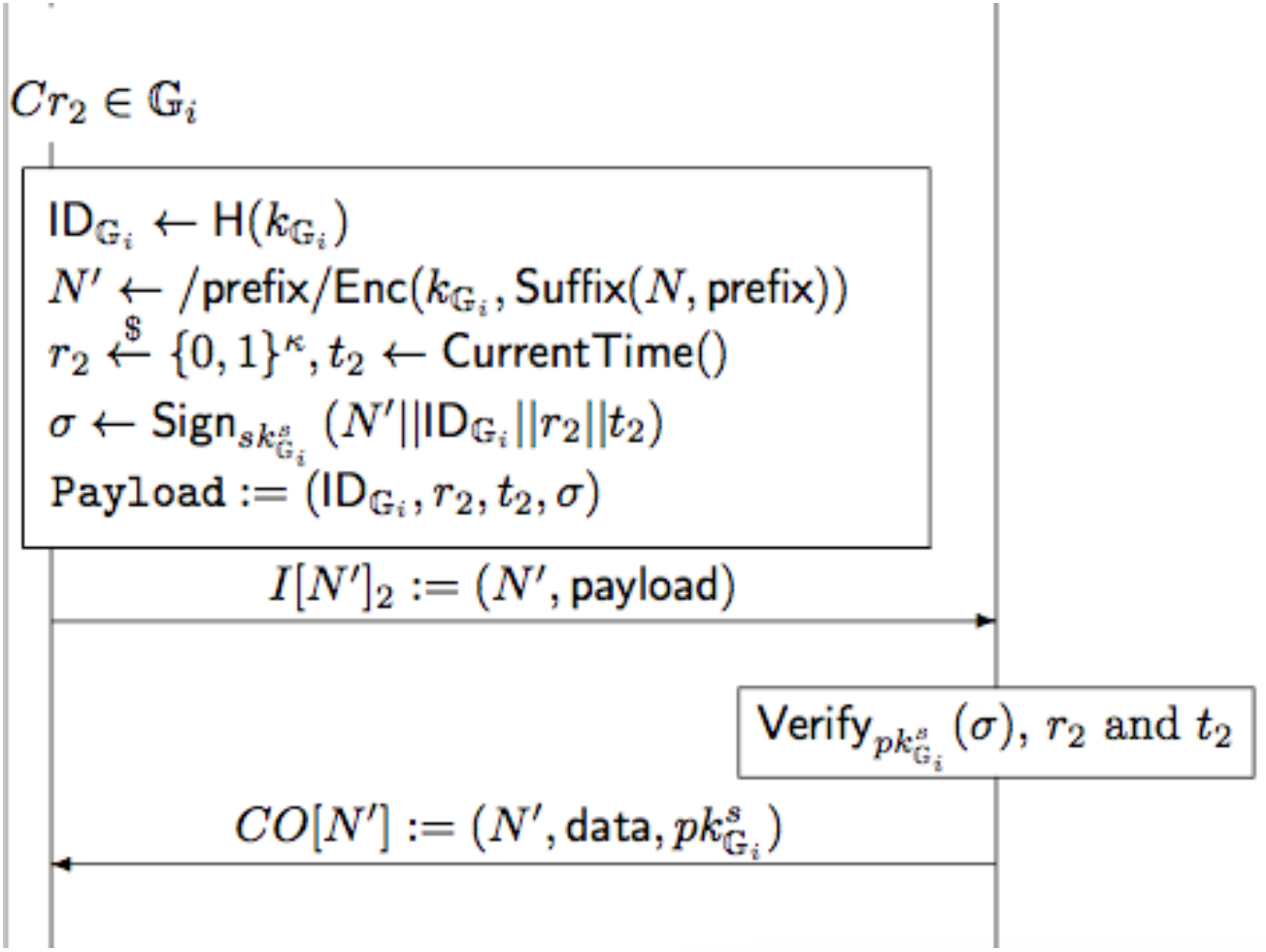
Answer: Follow the authorized key binding (AKB) rule:

ACKB: Cached content protected under IBAC must reflect the verification key associated with the authorization policy.

AKB in Action (Part 1)



AKB in Action (Part 2)



Recommendations

1. If *replay attacks* are not a concern, then consumers only need to use a name obfuscation function and include their group identity in the `Payload`.
2. If *replay attacks* are plausible and *name privacy* is a concern, then name obfuscation must be used and authorization information must be included in interest `Payload` fields.
3. If *replay attacks* are plausible but *name privacy* is not a concern, then only authorization information is sufficient.

Conclusion

1. Motivated about encryption- and interest-based access control
2. Discussed two ways to enforce interest-based access control
3. Provided an extension for to handle replay attacks in the network
4. Finished with recommendations for using IBAC

Q&A

Questions?...

Fire away!