

Efficient Content Verification in Named Data Networking

2015. 10. 2.

Dohyung Kim¹, Sunwook Nam², Jun Bi³, Ikjun Yeom¹

mr.dhkim@gmail.com

¹*Sungkyunkwan University*

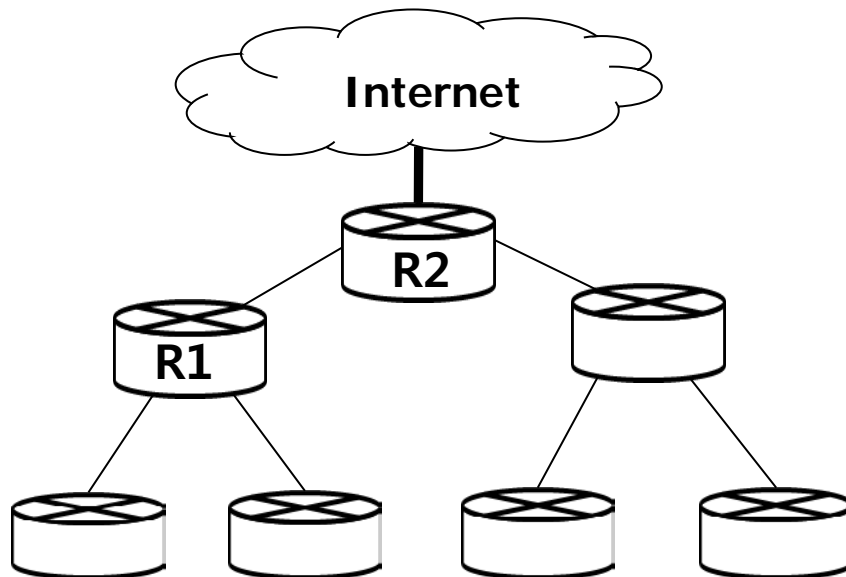
²*Korea Financial Telecommunications and Clearing Institute*

³*singhua University*

Named Data Networking (NDN)

- Name-based consumer-driven content delivery

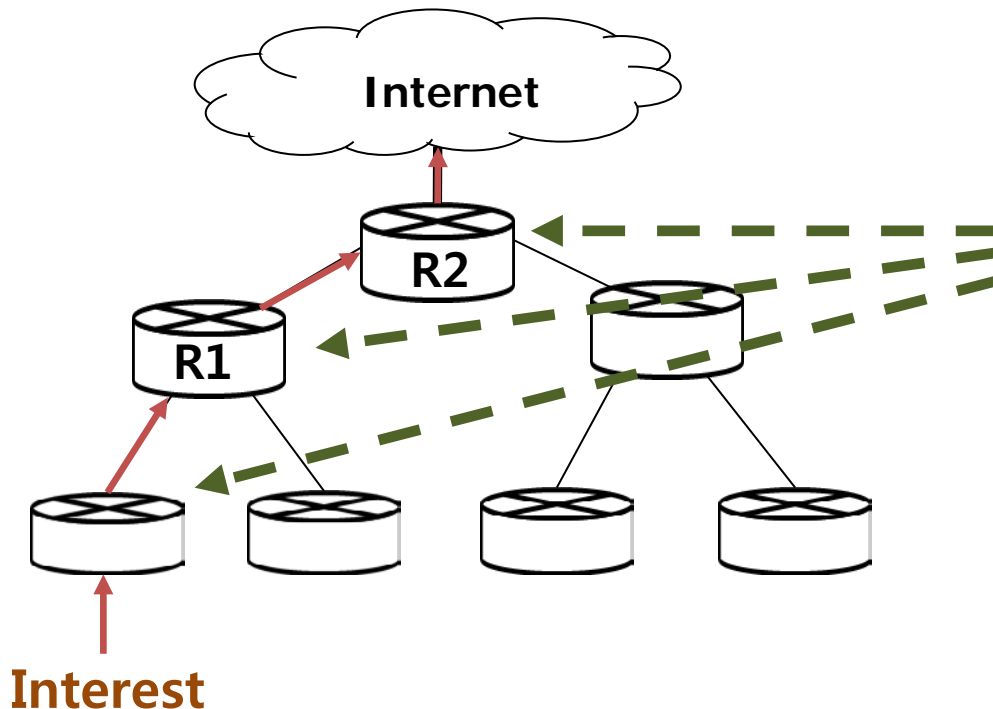
Request to Where → Request for **What**



Named Data Networking (NDN)

- Name-based consumer-driven content delivery

Request to Where → Request for **What**

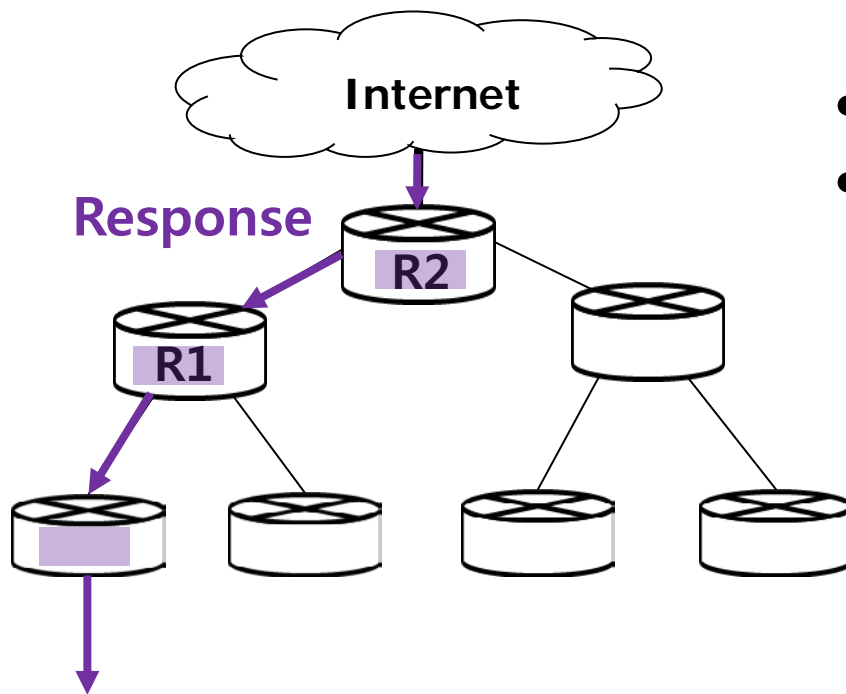


PIT entry is created
at routers

Named Data Networking (NDN)

- Name-based consumer-driven content delivery

Request to Where → Request for **What**

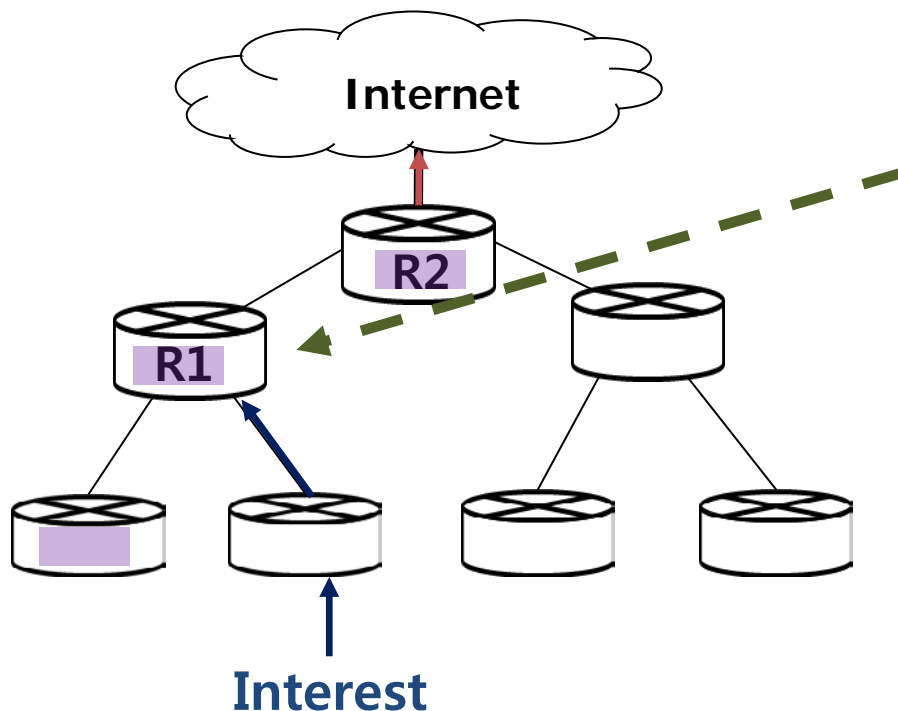


- PIT-based Content Delivery
- In-network Caching

Named Data Networking (NDN)

- Name-based consumer-driven content delivery

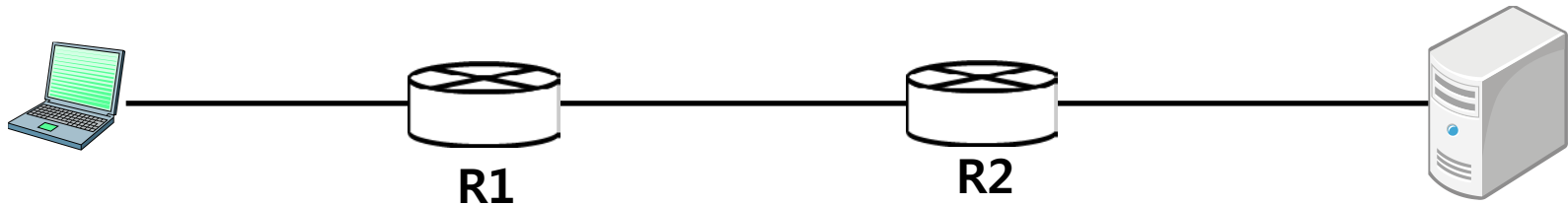
Request to Where → Request for **What**



Content is served from **in-network cache**

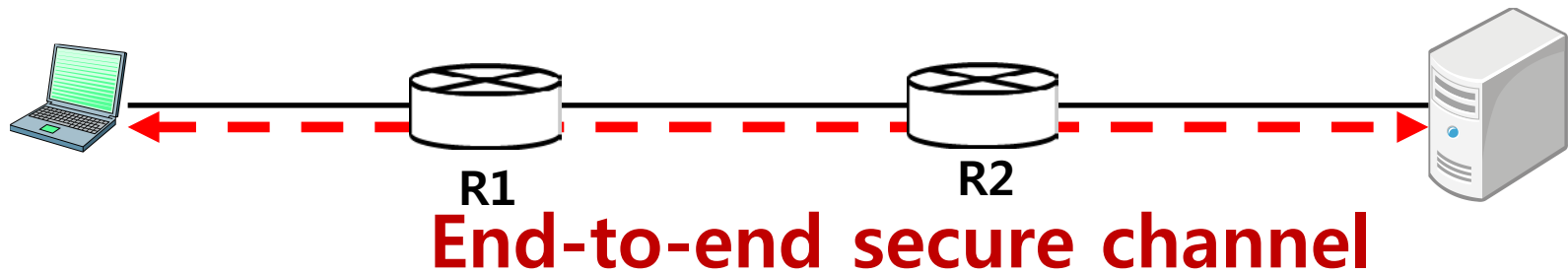
Secure Communication

- In IP networks



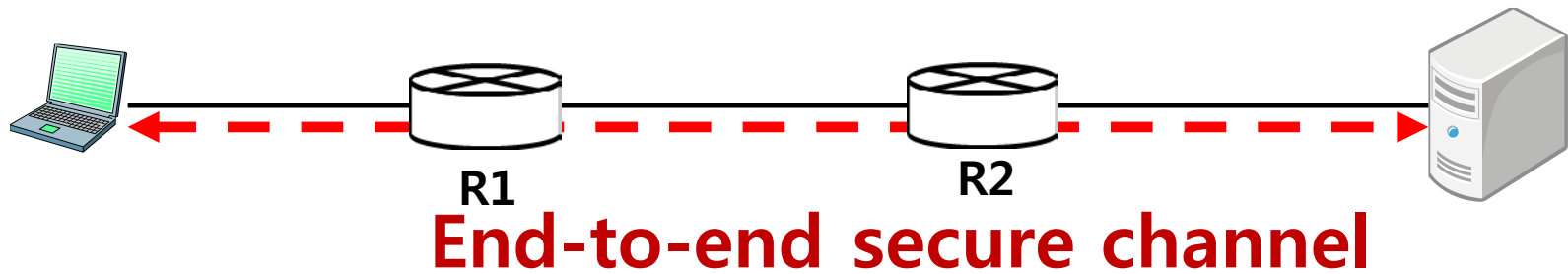
Secure Communication

- In IP networks

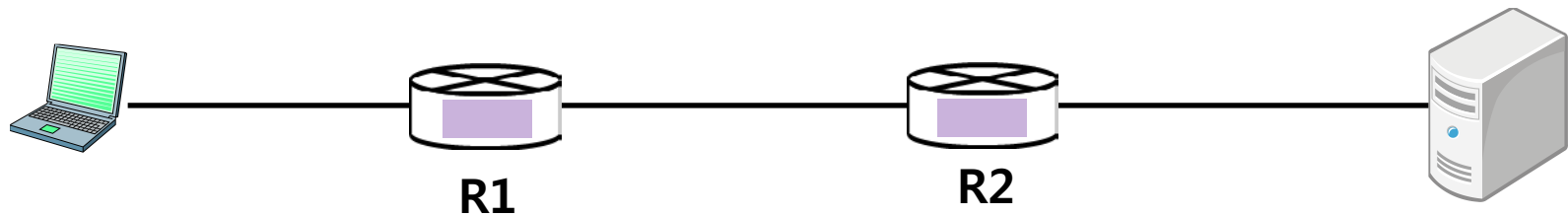


Secure Communication

- In IP networks

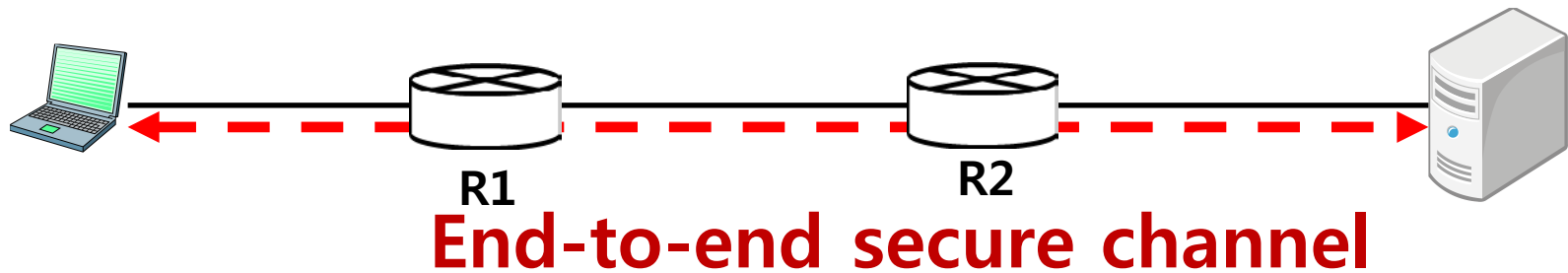


- In NDN

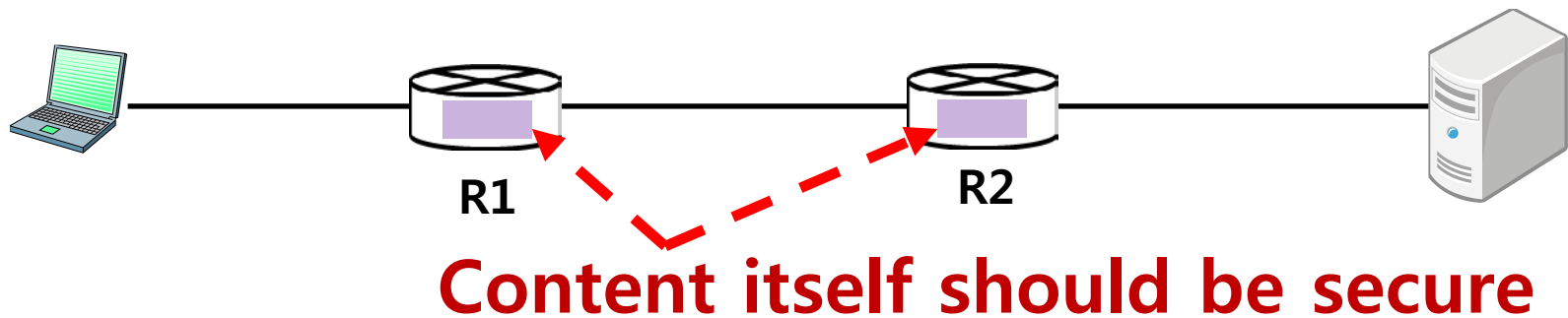


Secure Communication

- In IP networks



- In NDN

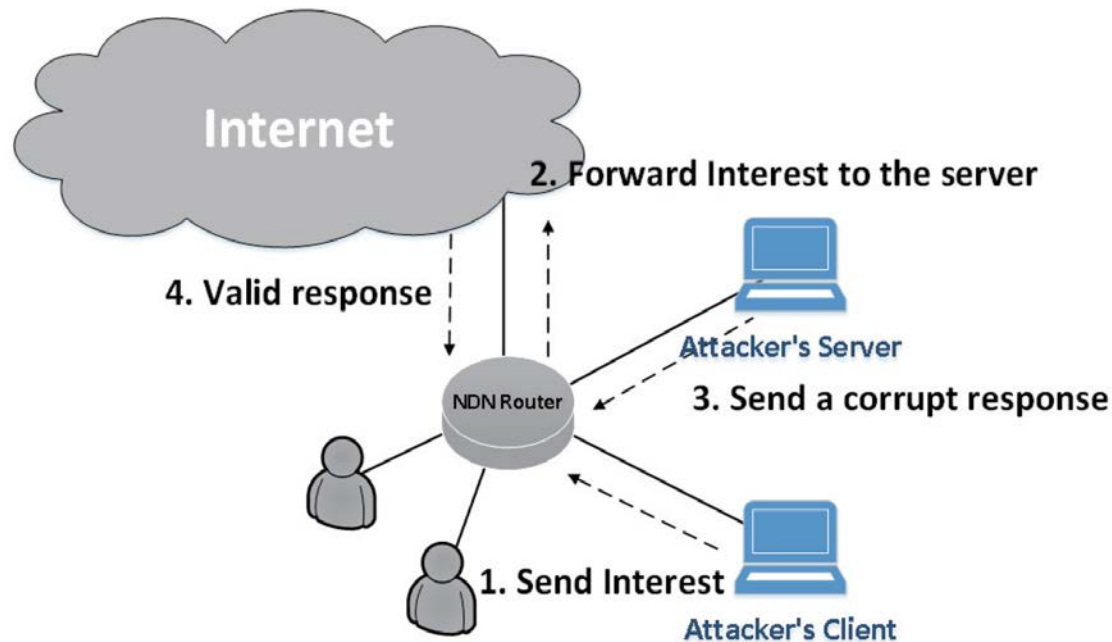


Content Poisoning Attack

- Fabricated content is placed in the content store
 - ✓ Router compromise

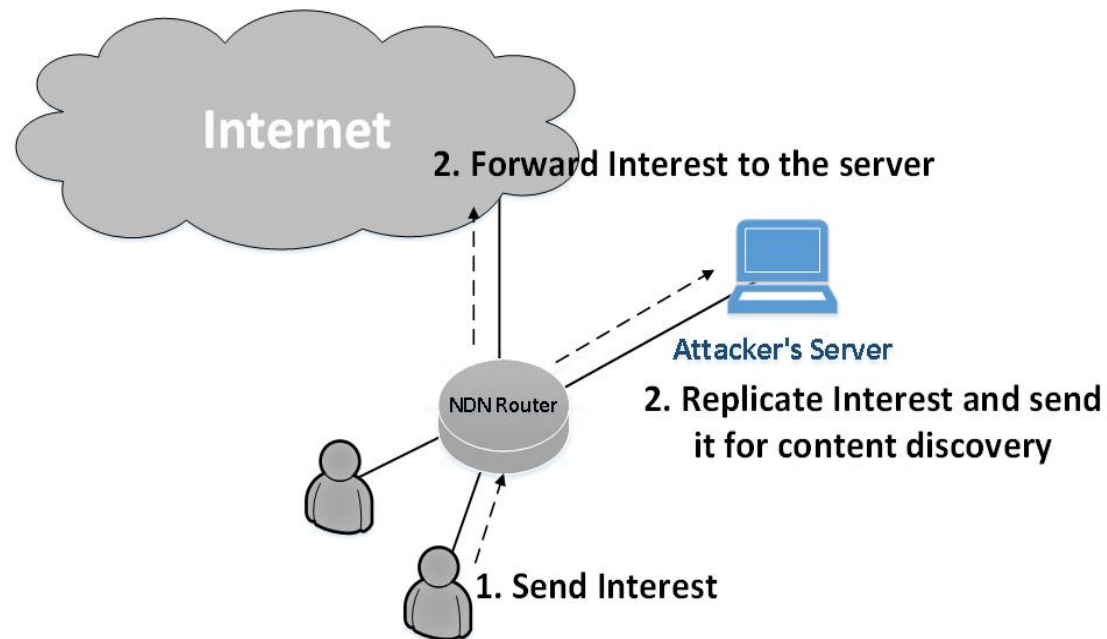
Content Poisoning Attack

- Fabricated content is placed in the content store
 - ✓ Router compromise
 - ✓ Injection from attackers' server



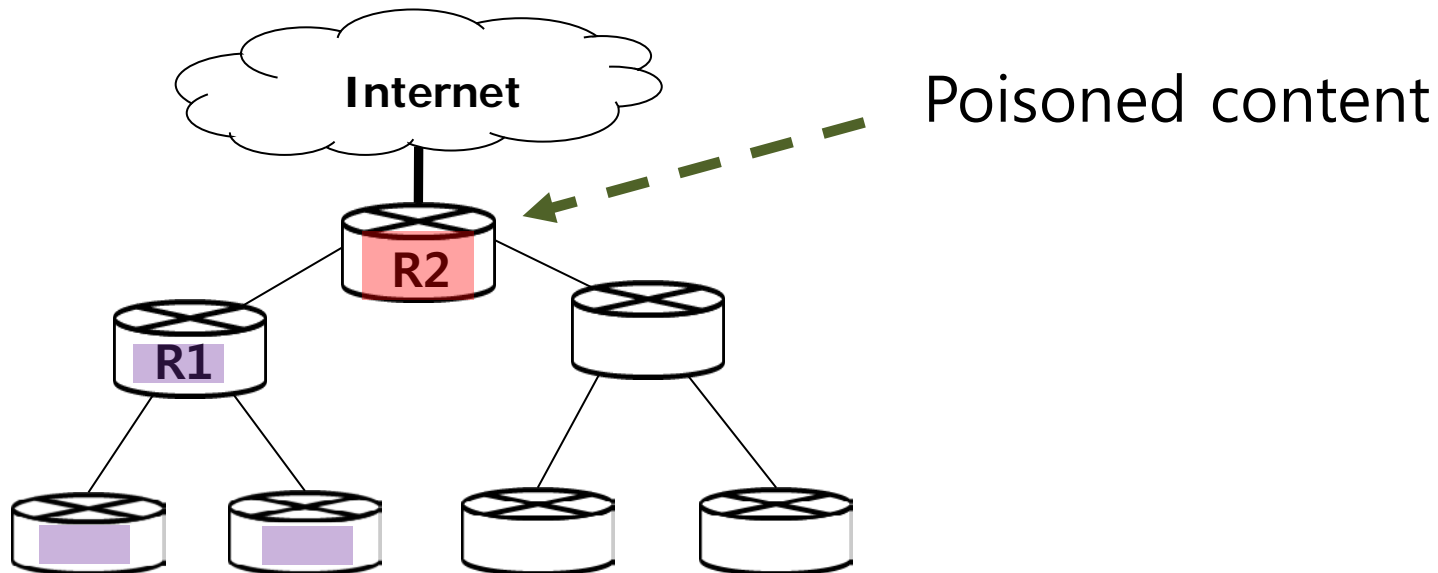
Content Poisoning Attack

- Fabricated content is placed in the content store
 - ✓ Router compromise
 - ✓ Injection from attackers' server



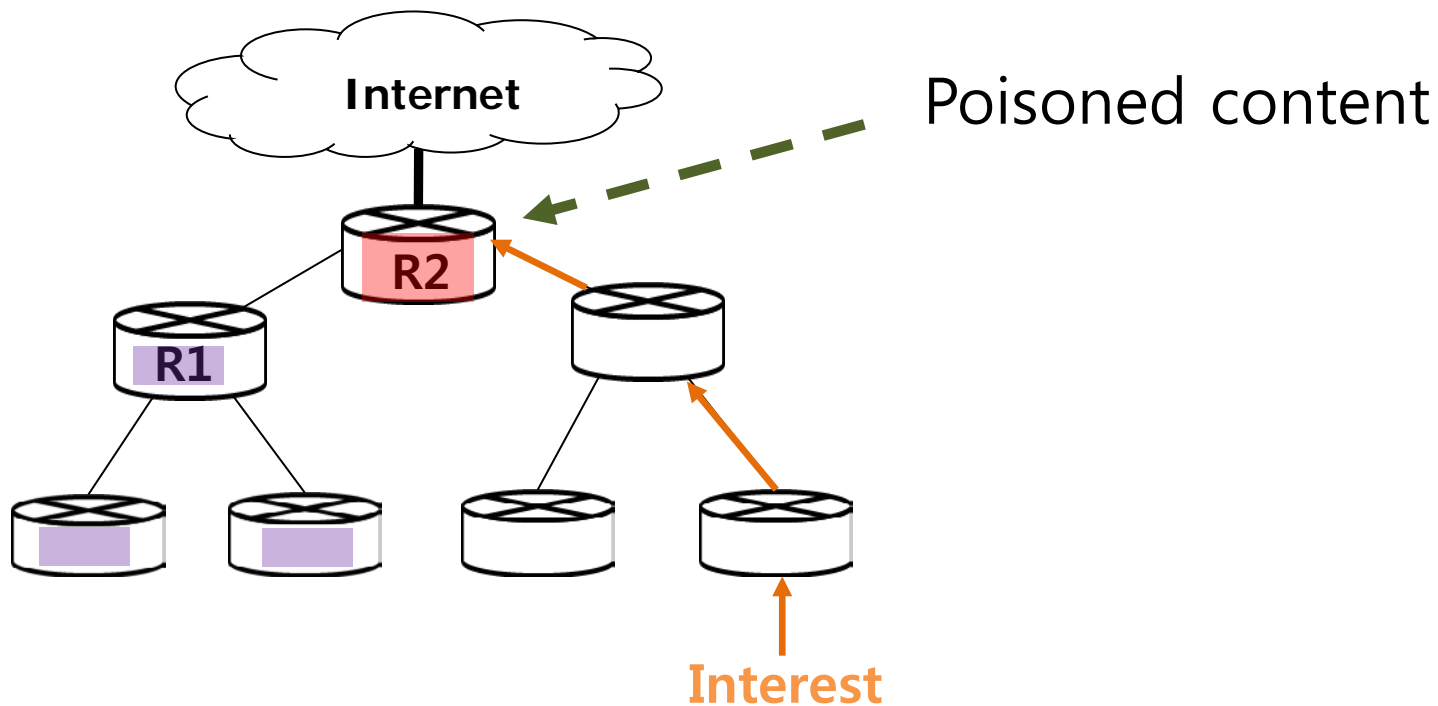
Content Poisoning Attack

- Distribution of the fabricated content



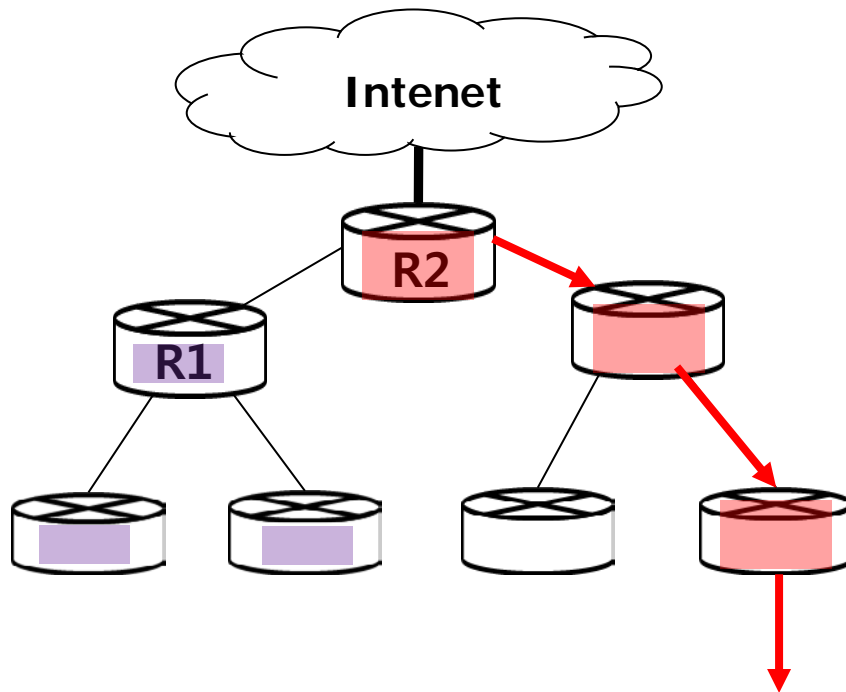
Content Poisoning Attack

- Distribution of the fabricated content



Content Poisoning Attack

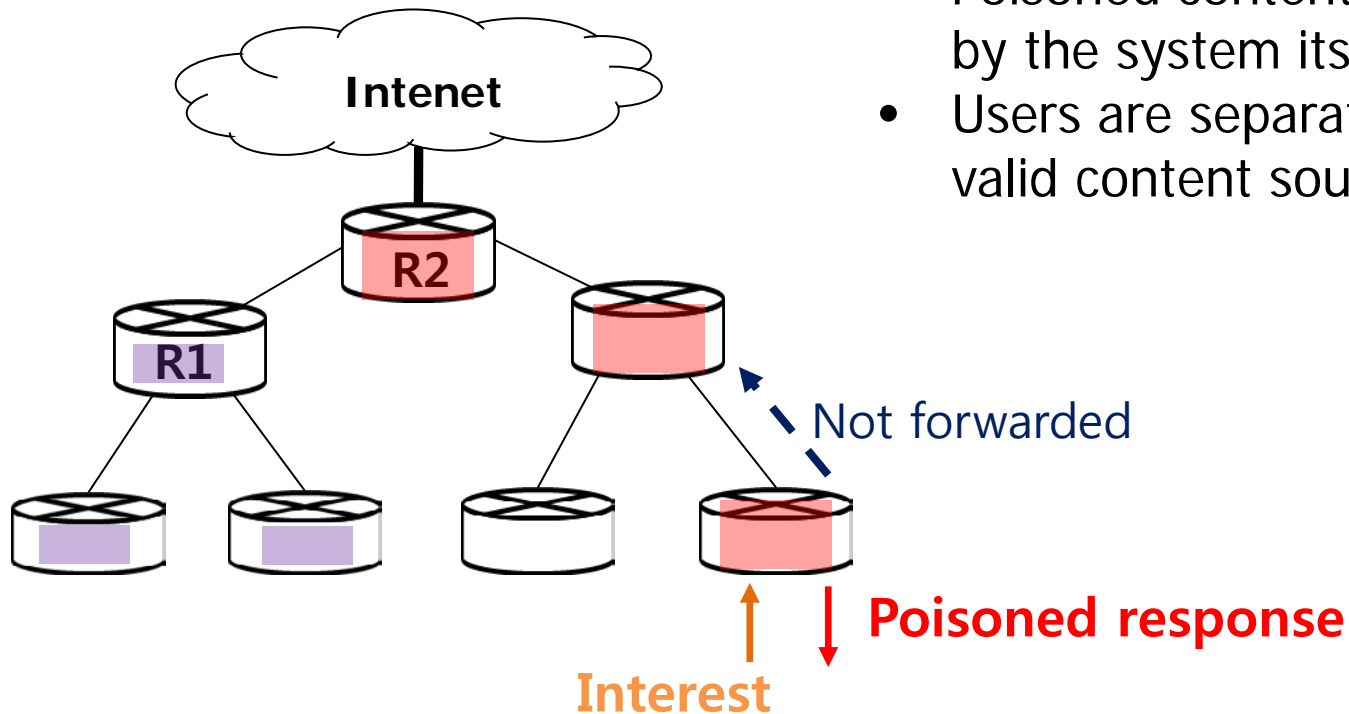
- Distribution of the fabricated content



- Poisoned content is distributed by the system itself

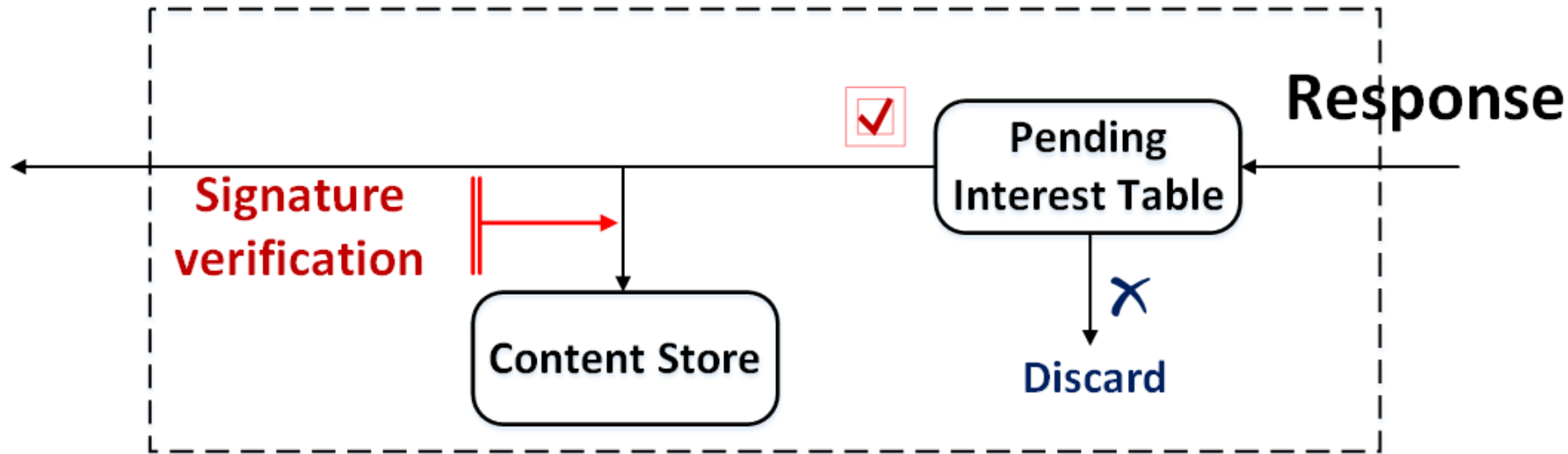
Content Poisoning Attack

- Distribution of the fabricated content



- Poisoned content is distributed by the system itself
- Users are separated from valid content sources

NDN Content Verification

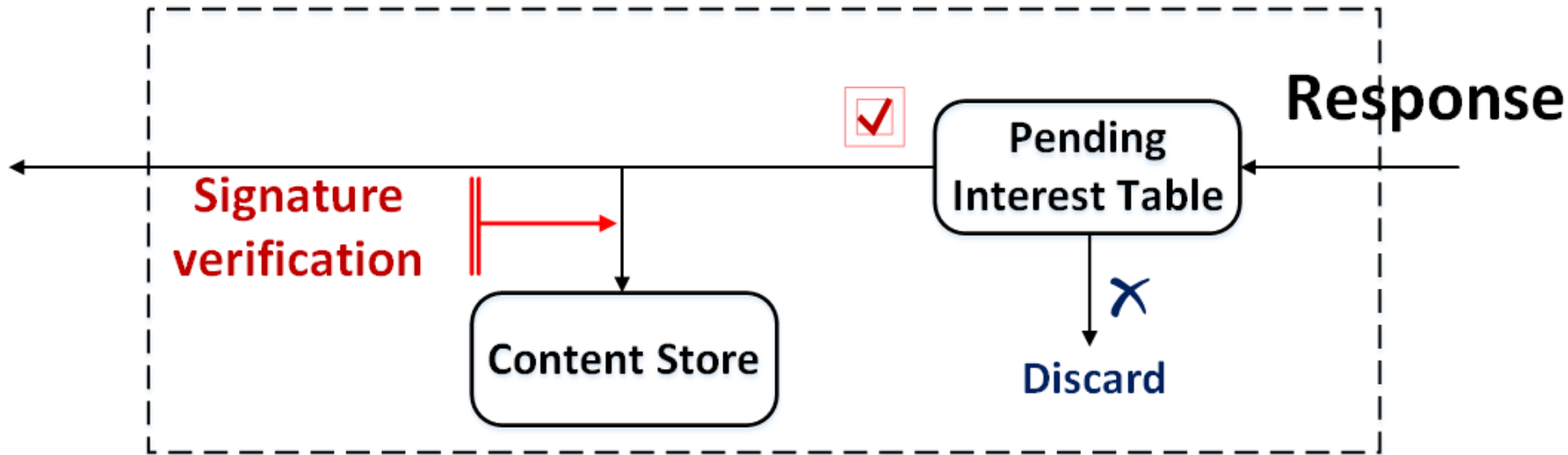


Signature verification

Response Packet

Content Name
Signature (digest algorithm, witness, ...)
Signed Info (publisher ID, key locator, stale time, ...)
Data

NDN Content Verification



Signature verification incurs
huge computational overhead

Response Packet

Content Name
Signature (digest algorithm, witness, ...)
Signed Info (publisher ID, key locator, stale time, ...)
Data

Related Work

- Probabilistic caching
 - Bianchi, Giuseppe, et al. "Check before storing: What is the performance price of content integrity verification in LRU caching?." *ACM SIGCOMM Computer Communication Review* 43.3 (2013): 59-67.
 - Verification overhead is controlled by caching probability

Related Work

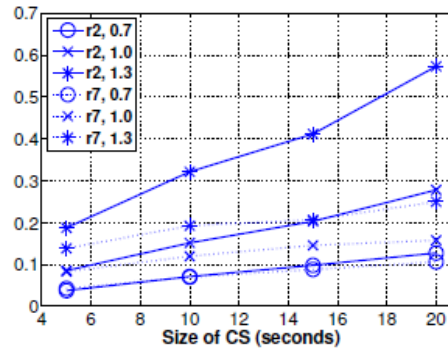
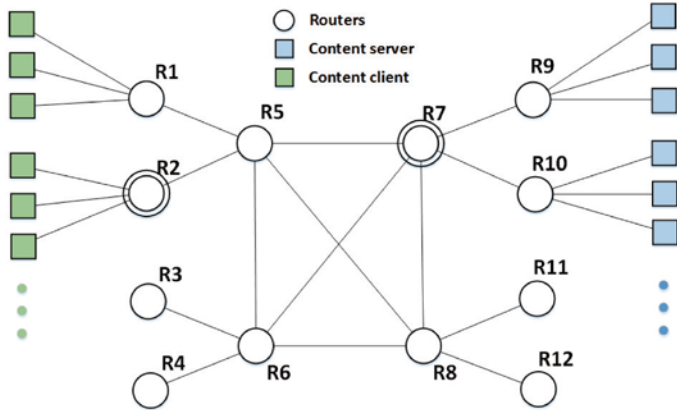
- Probabilistic caching
 - Bianchi, Giuseppe, et al. "Check before storing: What is the performance price of content integrity verification in LRU caching?." *ACM SIGCOMM Computer Communication Review* 43.3 (2013): 59-67.
 - Verification overhead is controlled by caching probability
- Limitation
 - Recency problem under dynamic content popularity
 - Limited application
 - Strongly bounded with random caching policy

Motivations

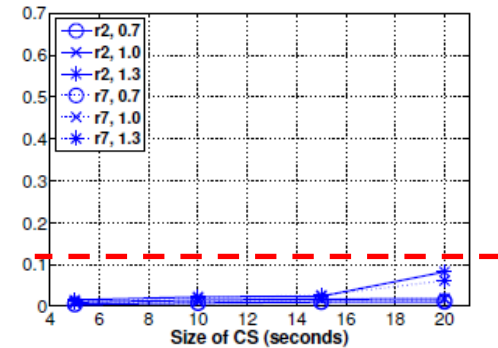
- Why do we verify even the content that is **not actually served**?

Motivations

- Why do we verify even the content that is **not actually served**?
- ns-3 simulation for estimating the amount of serving contents



Cache hit rate



Proportion of serving content in the CS

Objective

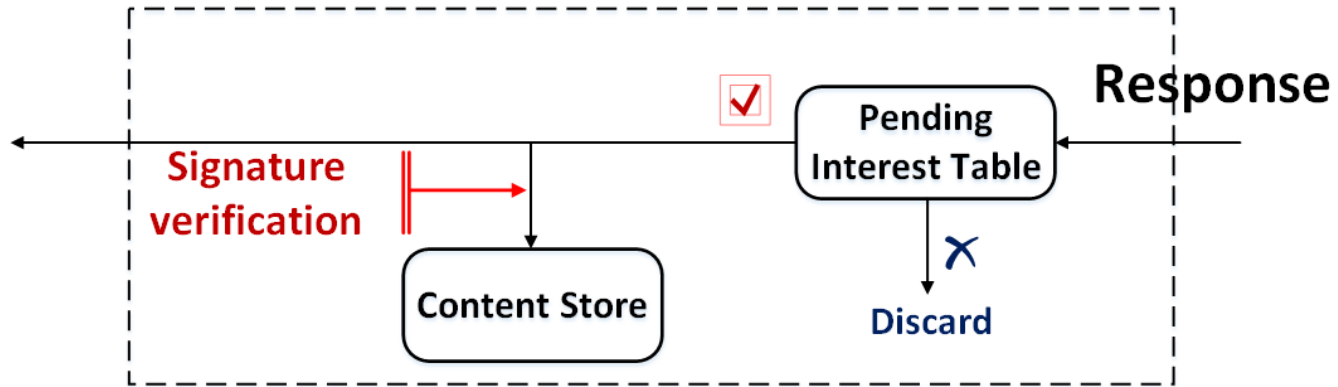
- Reduce verification overhead while preserving functionality of the built-in signature verification

The Proposed Scheme

- ✓ Verify serving contents only

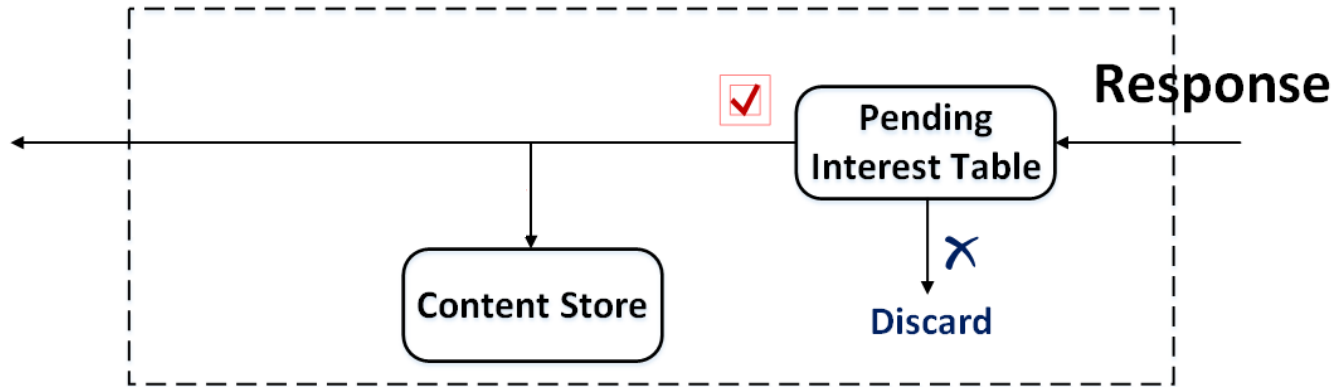
The Proposed Scheme

- Verify Serving Contents Only



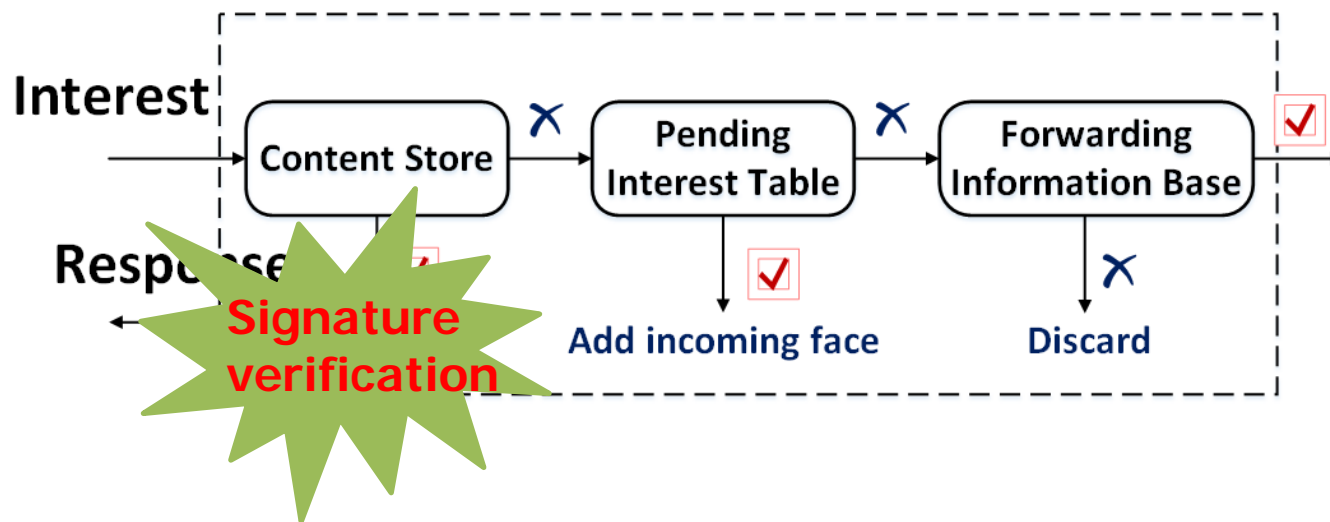
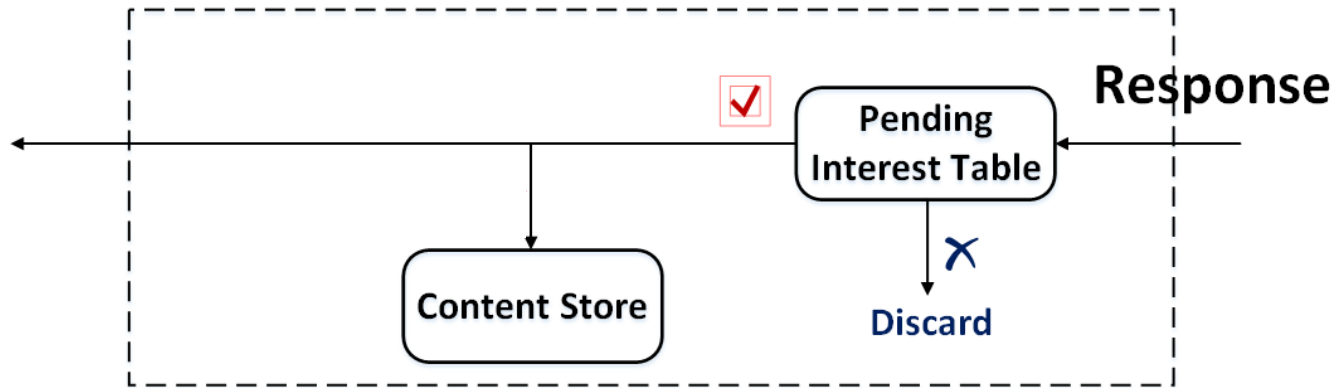
The Proposed Scheme

- Verify Serving Contents Only



The Proposed Scheme

- Verify Serving Contents Only



The Proposed Scheme

- Verify Serving Contents Only

- In the proposed scheme, poisoned content is either
 - Evicted from the content store without any damages to the network
 - Discarded by the verification mechanism before being brought out to the network

The Proposed Scheme

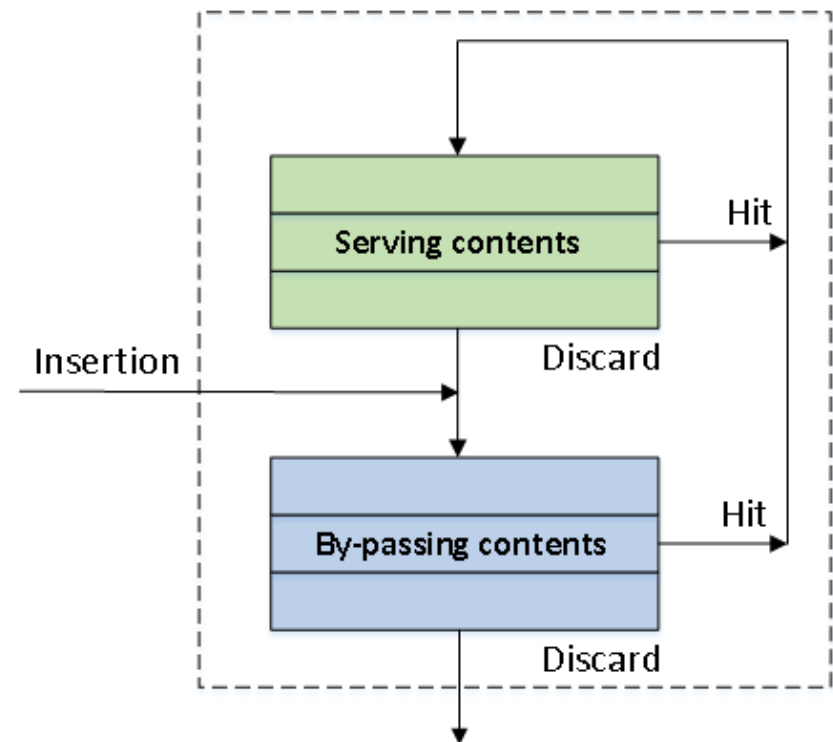
- ✓ Flag for the already verified content

**Content store
structure**

verify	name	data
False	C1Name	...
True	C2Name	...

The Proposed Scheme

- ✓ Favor the already-verified content in the content store
 - Segmented LRU prevents serving content from being evicted by by-passing content in the content store



Efficiency Analysis

- Efficiency metric

$$\kappa = \frac{N_e}{N_v}$$

- N_e : the number of examined poisoned contents
- N_v : the number of verifications

Efficiency Analysis

- In the basic scheme, κ corresponds to the proportion of the requests for the poisoned contents, e

Efficiency Analysis

- In the basic scheme, κ corresponds to the proportion of the requests for the poisoned contents, e
- In the proposed scheme,

$$\kappa = \frac{Re}{Re + RH_p(1 - e)}$$

- R is the request arriving rate
- H_p is the hit ratio for the unverified contents in the CS

Efficiency Analysis

- Hit ratio for the unverified contents

$$H_p = \sum_{\forall i} \underline{P_N(i) P_h(i) P_m(i)}$$

Proportion of requests
for content i

Efficiency Analysis

- Hit ratio for the unverified contents

$$H_p = \sum_{\forall i} P_N(i) P_h(i) \underline{P_m(i)}$$

Cache-miss probability
for the content i

Efficiency Analysis

- Hit ratio for the unverified contents

$$H_p = \sum_{\forall i} P_N(i) P_h(i) P_m(i)$$

Cache-hit probability for the content i

Efficiency Analysis

- Hit ratio for the unverified contents

$$H_p = \sum_{\forall i} P_N(i) \underbrace{P_h(i)} P_m(i)$$

Cache-hit probability for the content i

- According to Che approximation

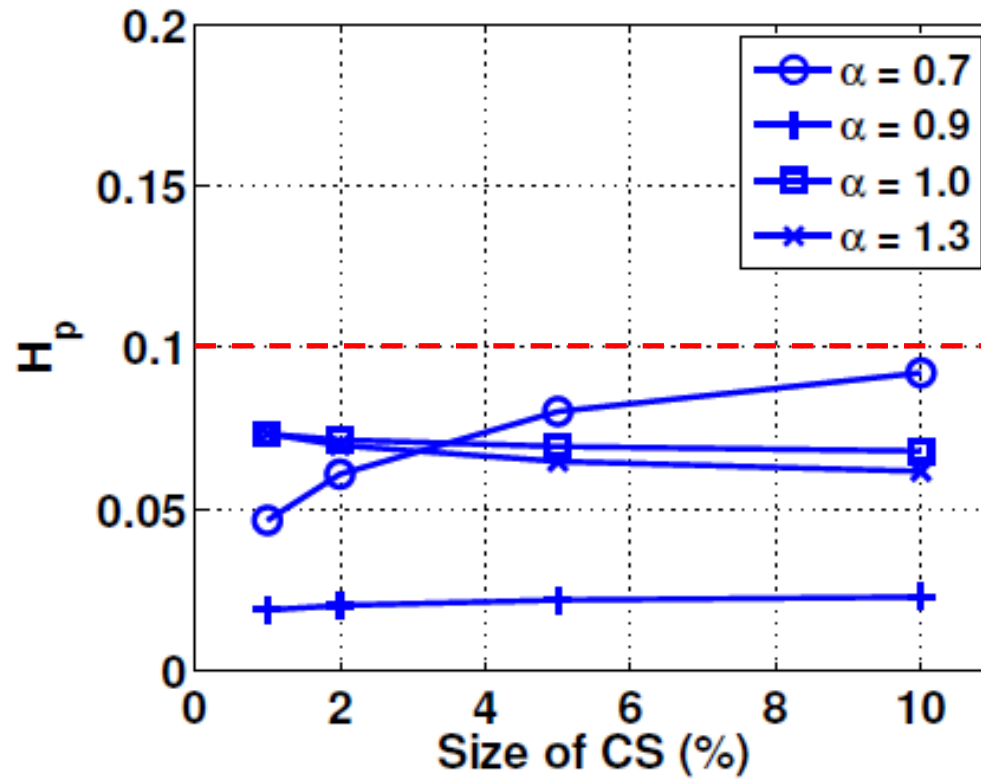
$$P_h(i) = 1 - e^{-P_N(i)tC}$$

$$C = \sum_{\forall i} (1 - e^{-P_N(i)t})$$

- C is the size of CS, and t is the residing time in the CS

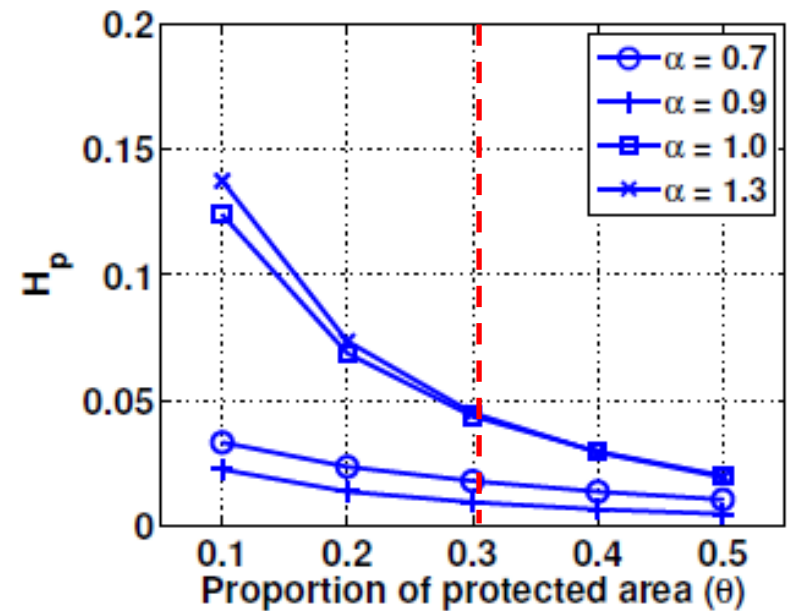
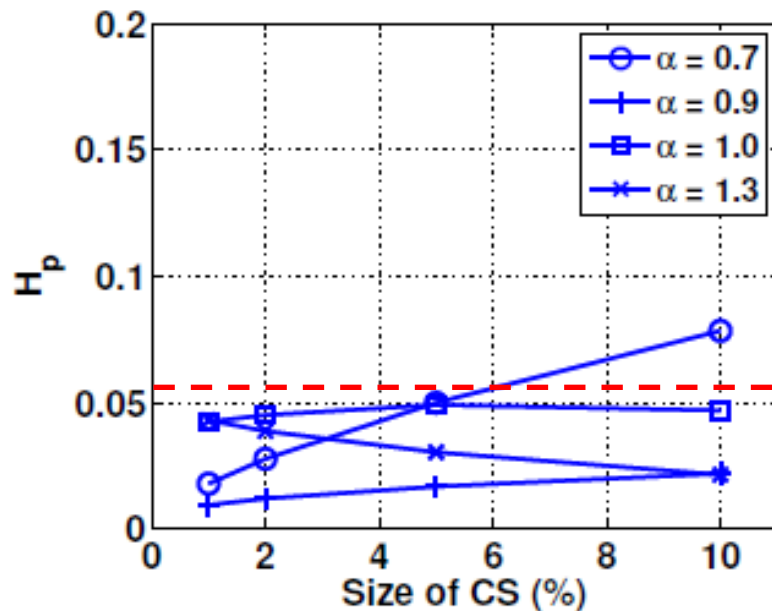
Analytic Results

- H_p without SLRU



Analytic Results

- H_p with SLRU



Analytic Results

- In the proposed scheme without SLRU

$$\kappa = \frac{e}{e + H_p(1 - e)} \geq \frac{e}{e + 0.1(1 - e)} = \frac{e}{0.1 + 0.9e}$$

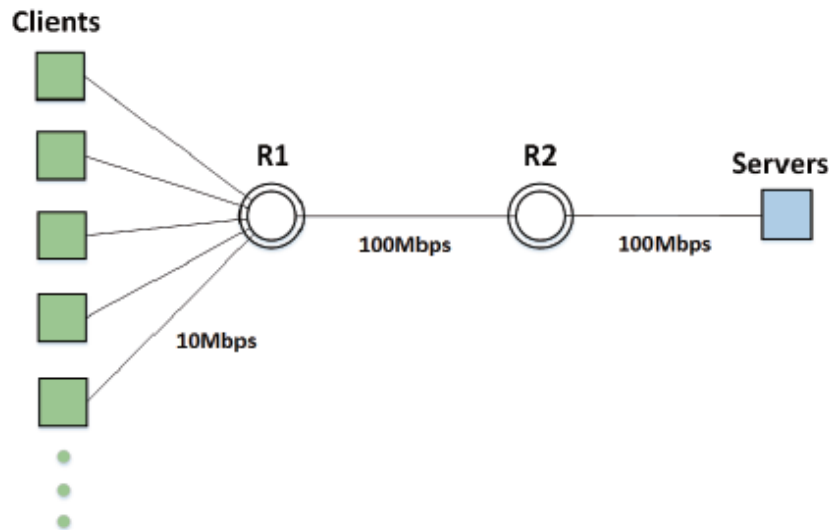
- In the proposed scheme with SLRU

$$\kappa = \frac{e}{e + H_p(1 - e)} \geq \frac{e}{e + 0.05(1 - e)} = \frac{e}{0.05 + 0.95e}$$

- When e is close to 0, the proposed scheme achieve a 10 or 20 time larger κ value of
- The value of κ is changed according to the amount of poisoned counter e

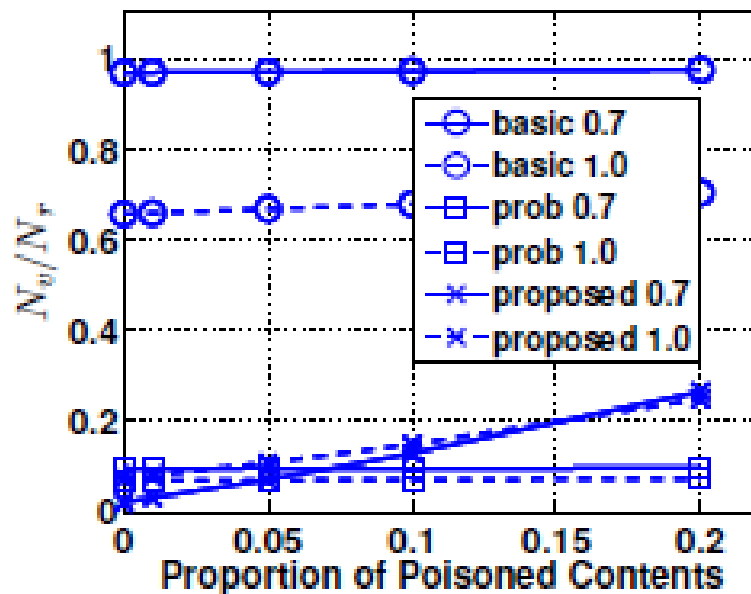
Evaluation

- Ns-3 simulation with
 - 10^6 Contents whose popularity follows Zipf-Mandelbrot distribution function
 - YouTube trace from UMASS Campus during Mar. 11-17 in 2008

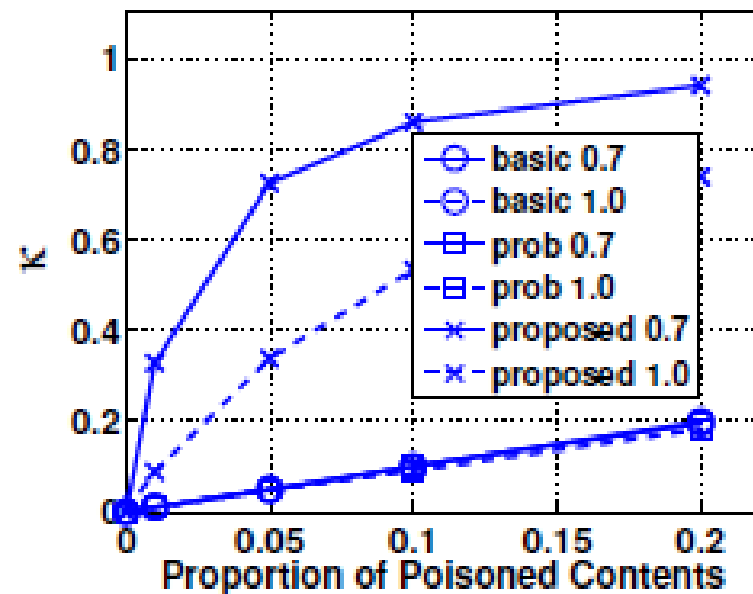


Results

- Poisoned contents



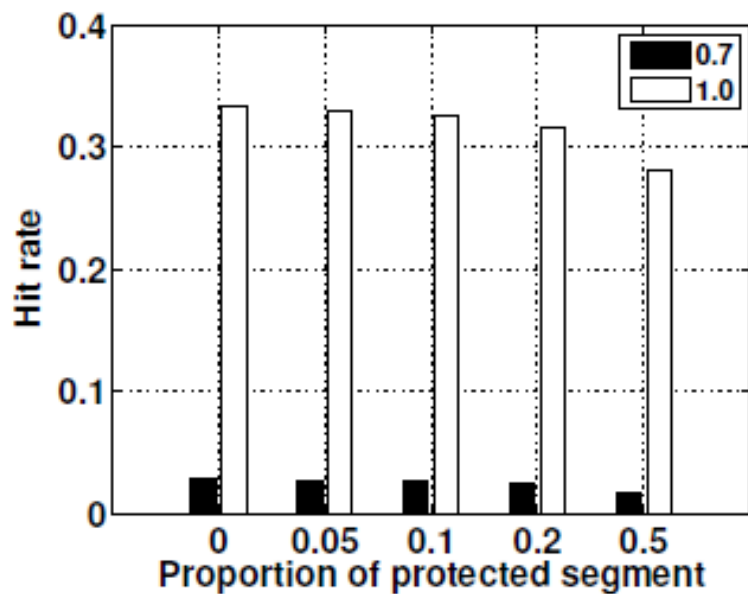
Verification overhead



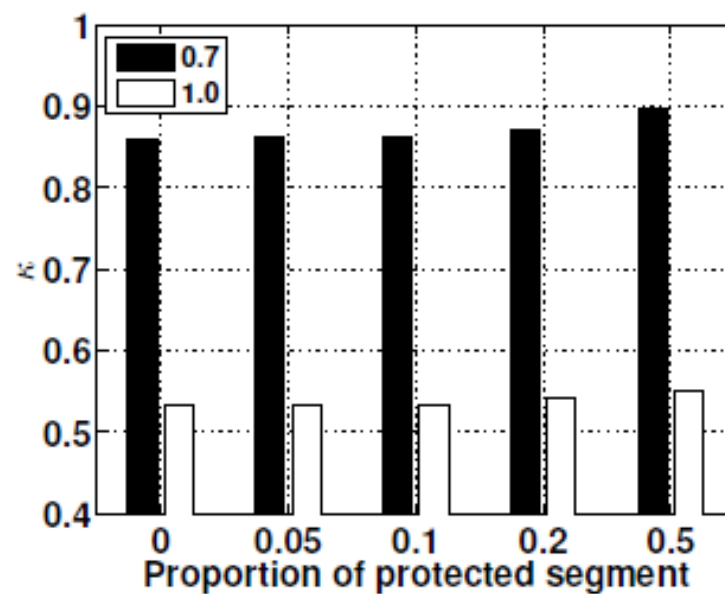
κ

Results

- Effect of Segmented LRU



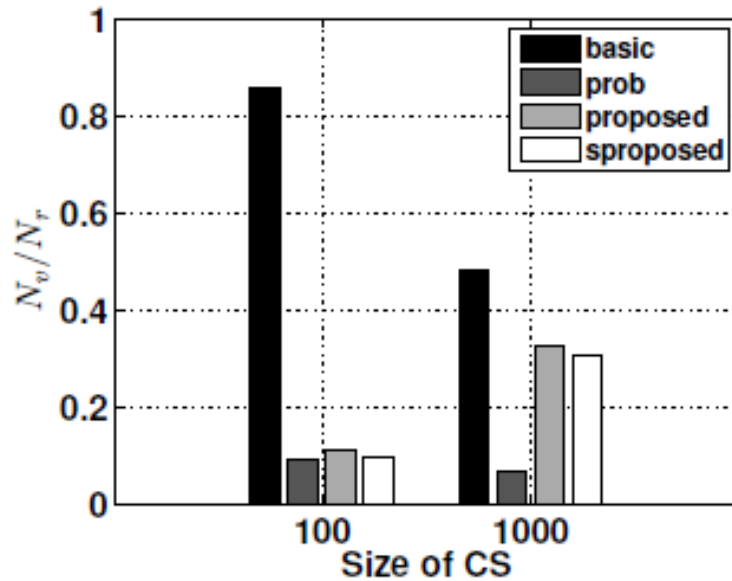
Hit rate



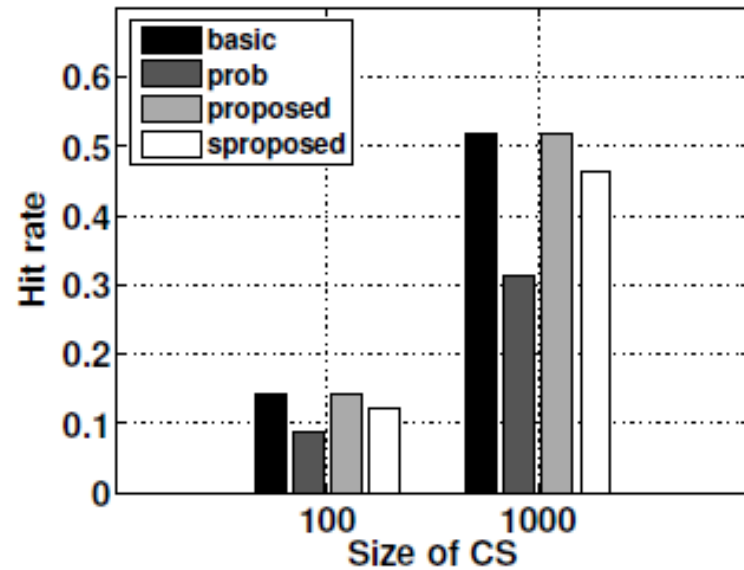
K

Results

- youTube Trace



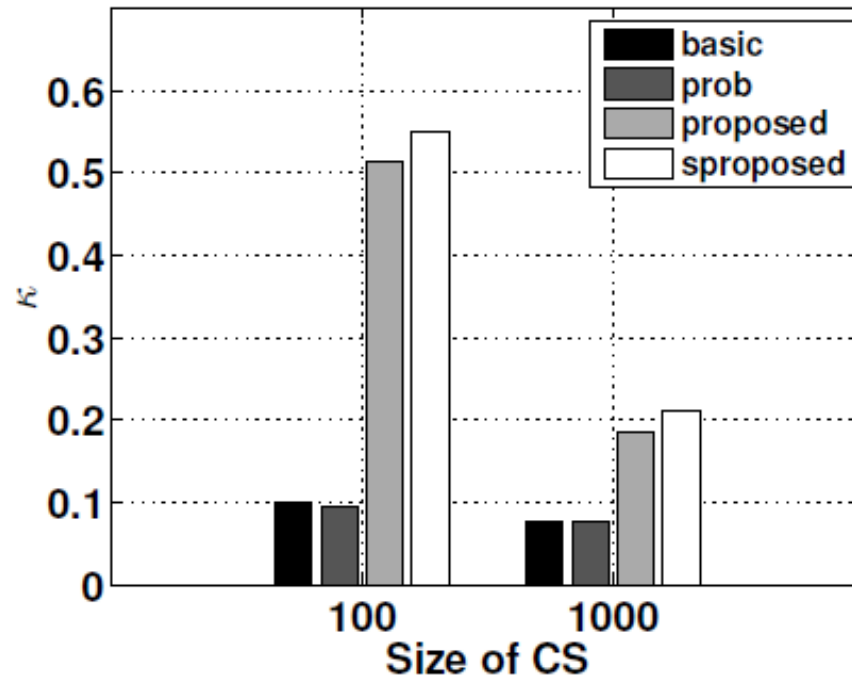
Verification overhead



Hit rate

Results

- youTube Trace



Discussion

- The access delay is increased due to the verification process
 - Limited to the first access to the content

Discussion

- The access delay is increased due to the verification process
 - Limited to the first access to the content
- Multiple pending interests may distribute the poisoned content
 - Verification for the content that is matched with multiple pending interests

Discussion

- The access delay is increased due to the verification process
 - Limited to the first access to the content
- Multiple pending interests may distribute the poisoned content
 - Verification for the content that is matched with multiple pending interests
- Cache is attacked by using unverified data
 - Might show abnormal cache-hit pattern, that is, different value of (Hit rate / amount of hit data)

Conclusion

- ✓ We look at the **content poisoning attack** in NDN
 - Implementation and its effects

Conclusion

- ✓ We look at the **content poisoning attack** in NDN
 - Implementation and its effects
- ✓ We present an efficient way to reduce overhead of content verification at routers
 - Verification of the “**servicing content**” only

Conclusion

- ✓ We look at the **content poisoning attack** in NDN
 - Implementation and its effects
- ✓ We present an efficient way to reduce overhead of content verification at routers
 - Verification of the “**servicing content**” only
- ✓ We minimize verification overhead by favoring the servicing contents in the CS
 - **Flag and Segmented LRU**

Q and A

Thank you