

# ***In-Network Retransmission in ICN*** <sup>[1]</sup>

---

Kazuhisa Matsuzono and Hitoshi Asaeda

National Institute of Information and Communications Technology  
(NICT), Japan

# Background & Motivation

## ■ Growth of low-latency and high-quality video applications

- E.g.) Teleoperation of robots requires
  - network bandwidth of 10~100Mbps [2]
  - end-to-end latency below 200ms [2]



# Retransmission-based Loss Recovery

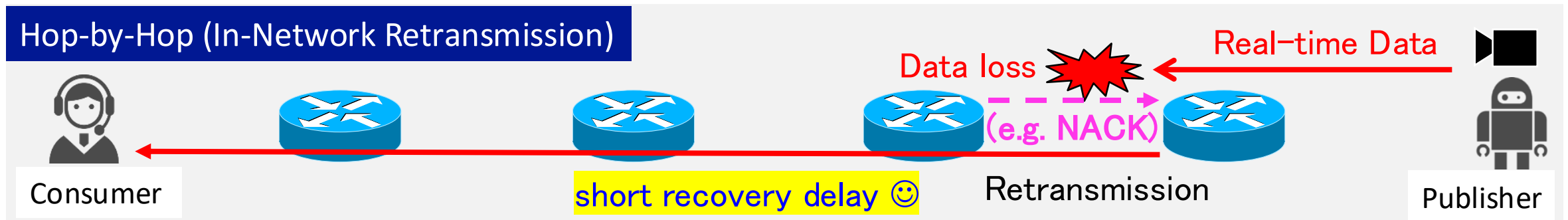
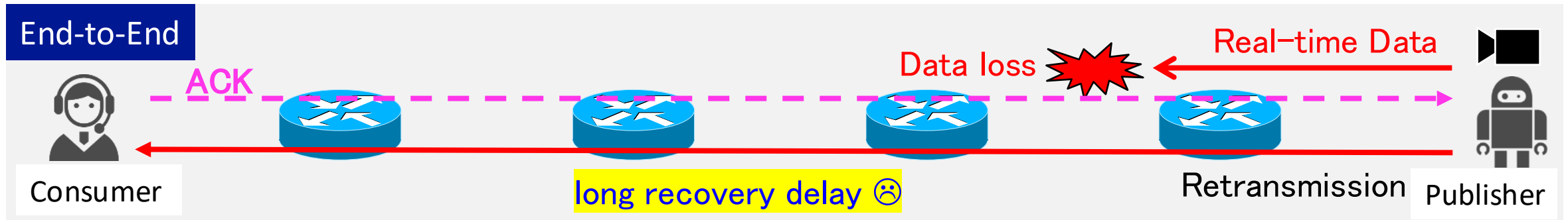
## ■ End-to-End vs. Hop-by-Hop Retransmission

### ➤ End-to-end basis:

- Long recovery delay

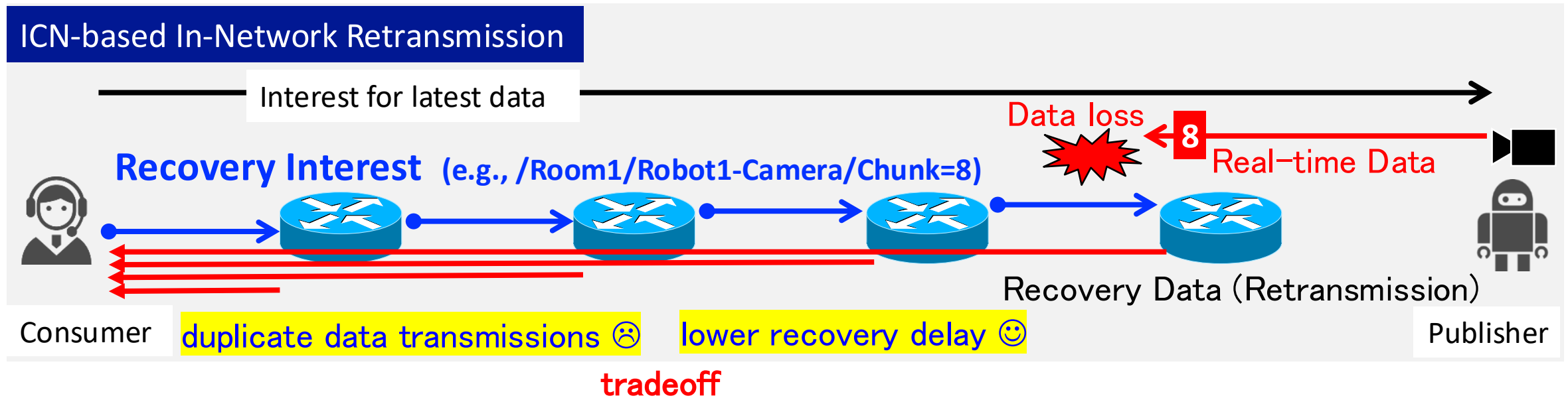
### ➤ Hop-by-hop basis (In-network Retransmission):

- Accelerate loss recovery



# ICN-based In-Network Retransmission

- ICN can potentially accelerate in-network retransmission
  - ICN provides In-network caching and hop-by-hop flow-aware transport
- **Some considerations for efficient and fast loss recovery**
  - How to detect loss (i.e., How can we recognize if it's a delay or a loss)?
  - Which intermediate node should send recovery interest?
  - When and how many times recovery interests should be transmitted?

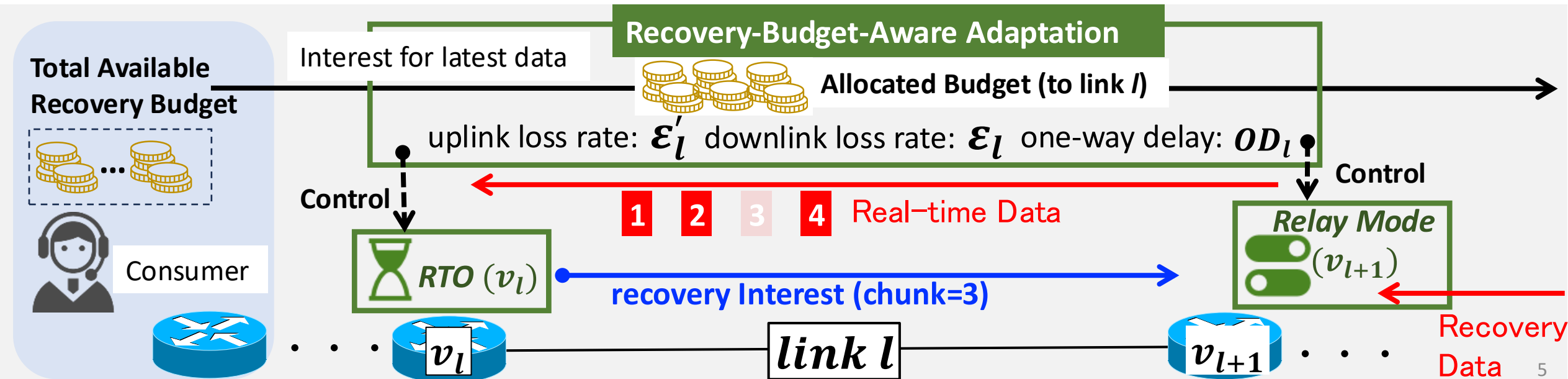
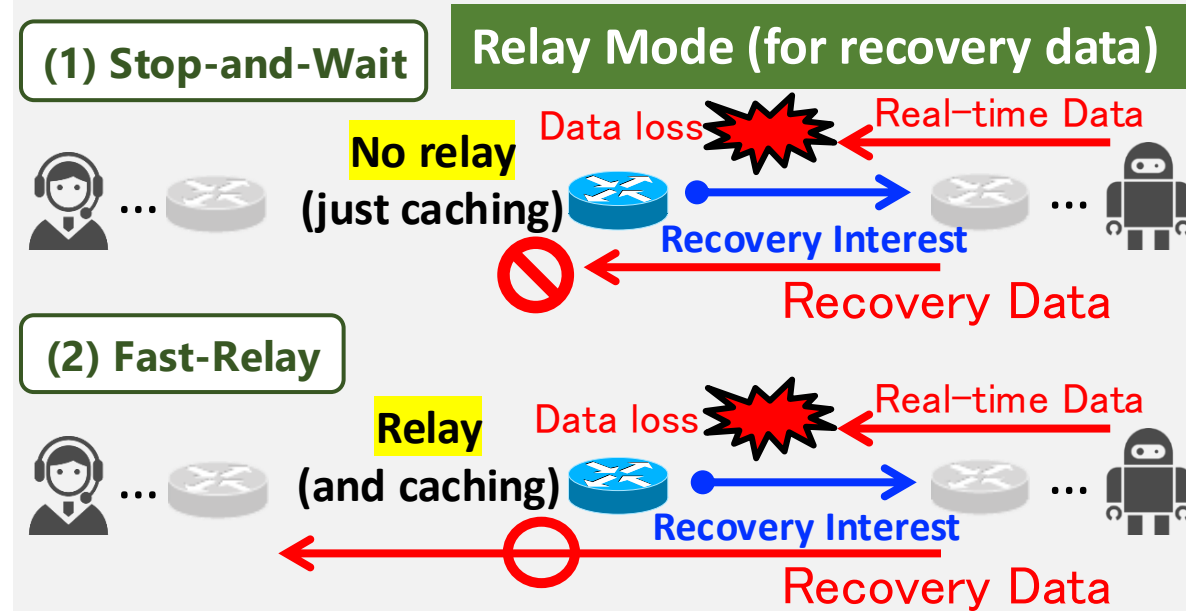


# Proposal: Key Features

## Recovery-Budget-Aware Adaptation

- Total available *recovery budget* is determined by
  - Application-specified acceptable latency
  - E2E propagation delay
- Allocate recovery budget to each link
  - Execute **per-link** retransmission-based loss recovery
- Switch between the two relay modes

(1) *Stop-and-Wait*  
and  
(2) *Fast-Relay*



# How to Allocate Recovery Budget ?

- Consumer collects (1) budget consumption and (2) budget request of each link

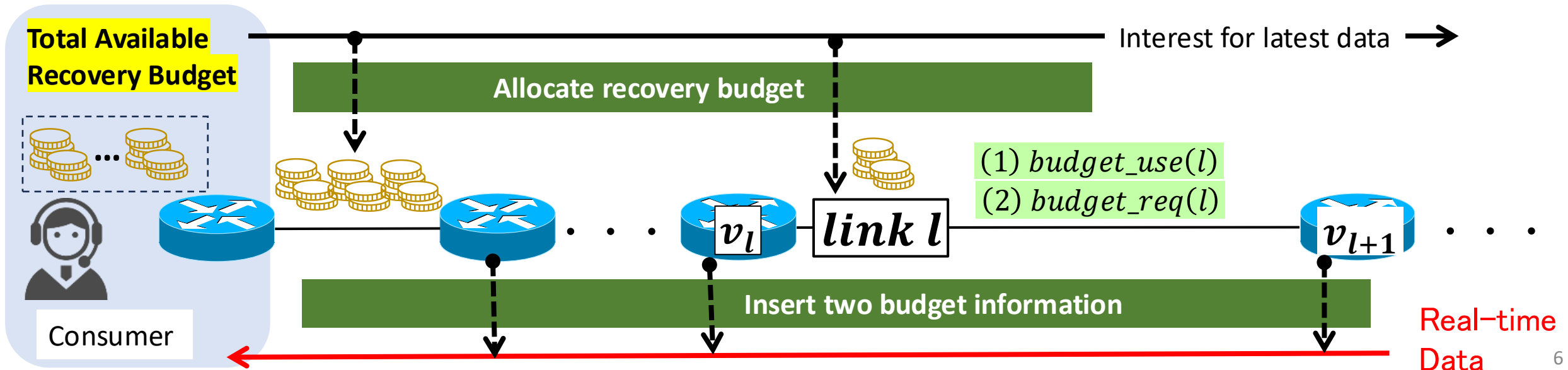
$$(1) \text{ budget\_use}(l) = \begin{cases} \sum_{x=1}^{x_l^{max}} \epsilon_l \cdot P_l^{suc}(x) \cdot R_l(x) & \text{(Fast-Relay),} \\ RTO_{v_l}^{min} \cdot x_l^{max} & \text{(Stop-and-Wait),} \end{cases}$$

$$(2) \text{ budget\_req}(l) = RTO_{v_l}^{min} \cdot x_l^{max}.$$

- $x_l^{max}$ : maximum transmission number of recovery Interest  
 $x_l^{max} = \min\{x | P_l^{suc}(x) \geq 0.99\}$
- $P_l^{suc}(x)$ : Prob. of success on the  $x^{th}$  retransmission round
- $R_l(x)$ : recovery delay when succeeding in the  $x^{th}$  retransmission round

## Budget Allocation Policy

- Preferentially from links closest to the consumer

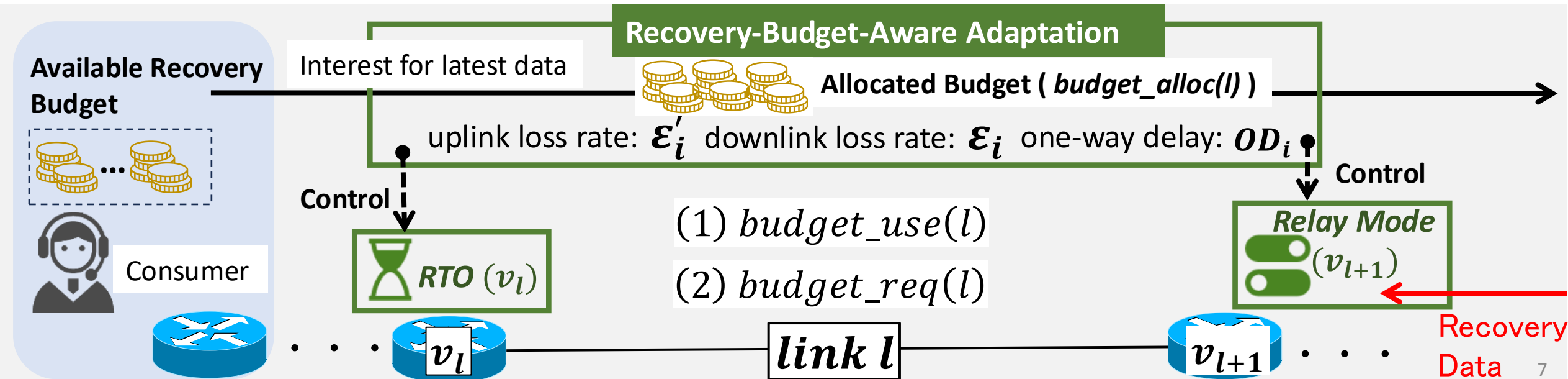


# How to Adjust the Relay Mode?

- When  $budget\_alloc(l) \geq budget\_req(l)$ 
  - Execute Stop-and-Wait
  - Set minimum RTO
- When  $budget\_alloc(l) < budget\_req(l)$ 
  - Execute Fast-Relay
  - Increase RTO as much as possible

```

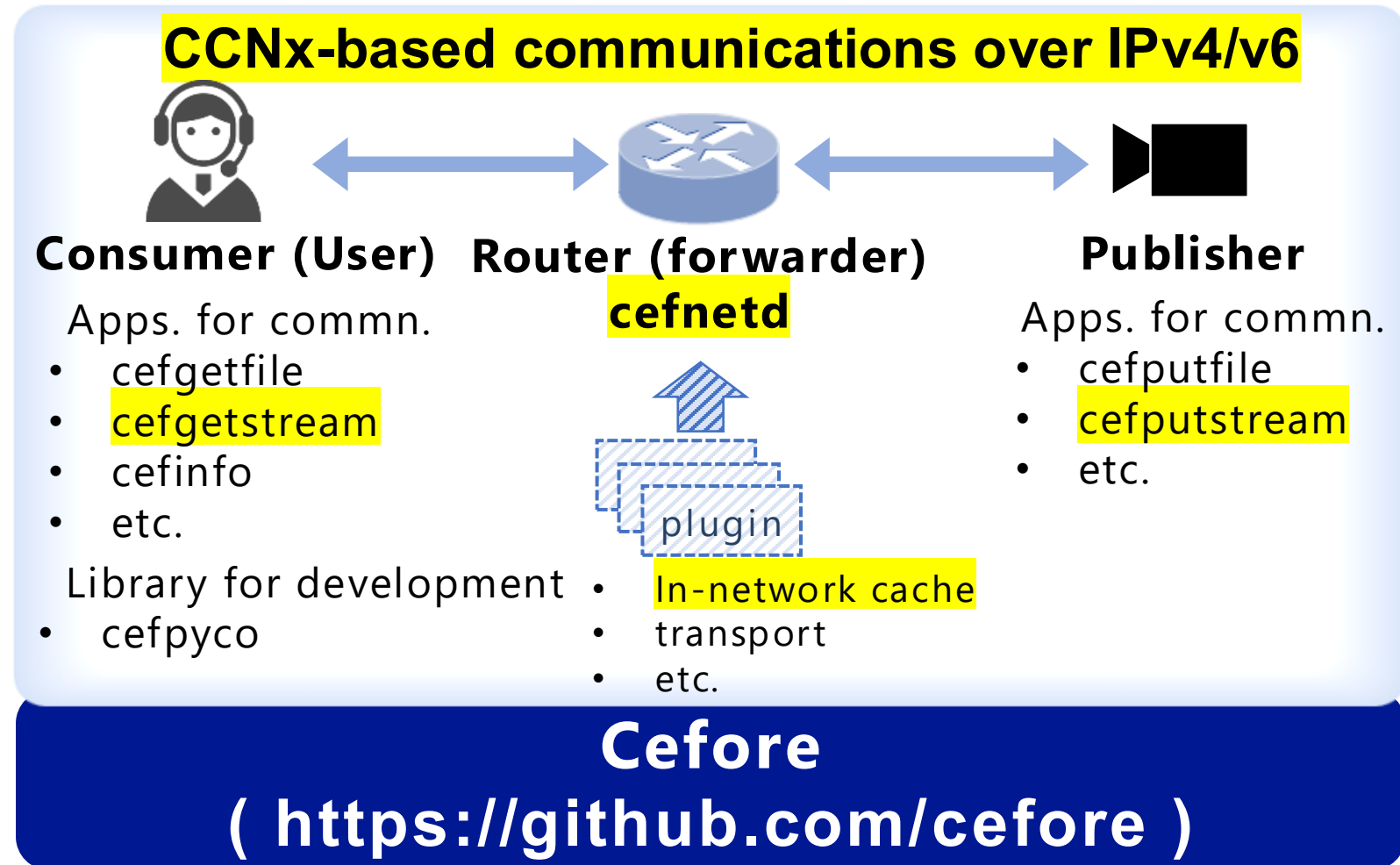
/* Determine the relay method of  $v_{(l+1)}$  */
budget_req(l) ←  $RTO_{v_l}^{min} \cdot x_l^{max}$ 
if budget_alloc(l) ≥ budget_req(l) then
     $m_{v_{(l+1)}} \leftarrow$  Stop-and-Wait
     $RTO_{v_l} \leftarrow RTO_{v_l}^{min}$ 
else
     $m_{v_{(l+1)}} \leftarrow$  Fast-Relay
    /* Increase  $RTO_{v_l}$  if possible*/
     $RTO_{v_l} \leftarrow \max\{RTO_{v_l} \mid budget\_use(l) \leq budget\_alloc(l)\}$ 
    if  $RTO_{v_l} < RTO_{v_l}^{min}$  then
         $RTO_{v_l} \leftarrow RTO_{v_l}^{min}$ 
/* Notify  $v_{(l+1)}$  of  $m_{v_{(l+1)}}$  */
Send_Interest( $m_{v_{(l+1)}}$ )
    
```



# Implementation on Cefore

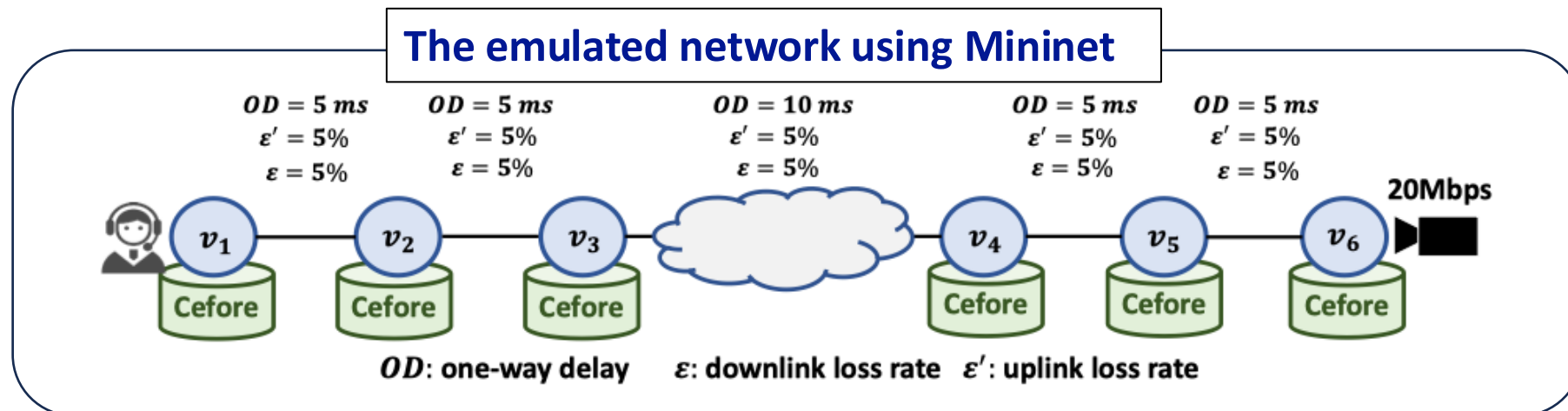
## ■ Cefore: Open-source software platform

- Compatible with CCNx-1.0 [RFC 8569] [RFC 8609]
- C language
  - Cefpyco library for app. development uses Python
- Supported OS
  - Linux (ubuntu 22.04 or later)
  - macOS (12 or later)
  - Raspberry Pi OS

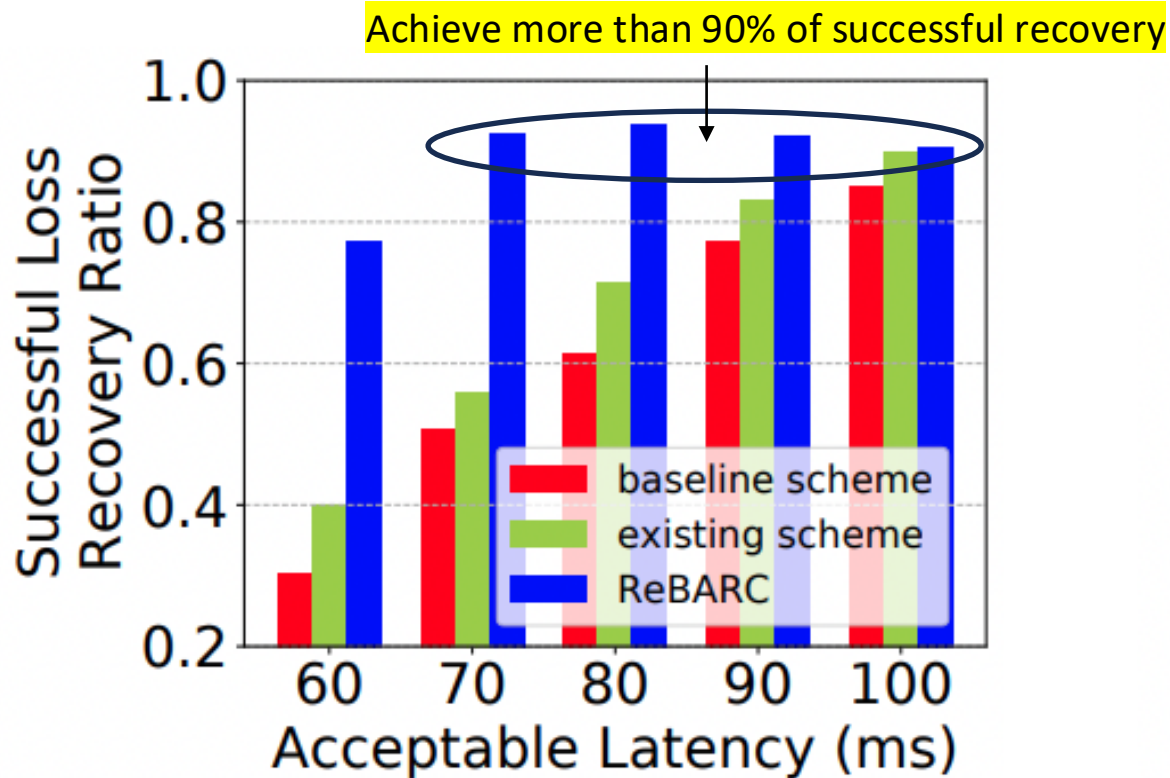


# Experimental Evaluation

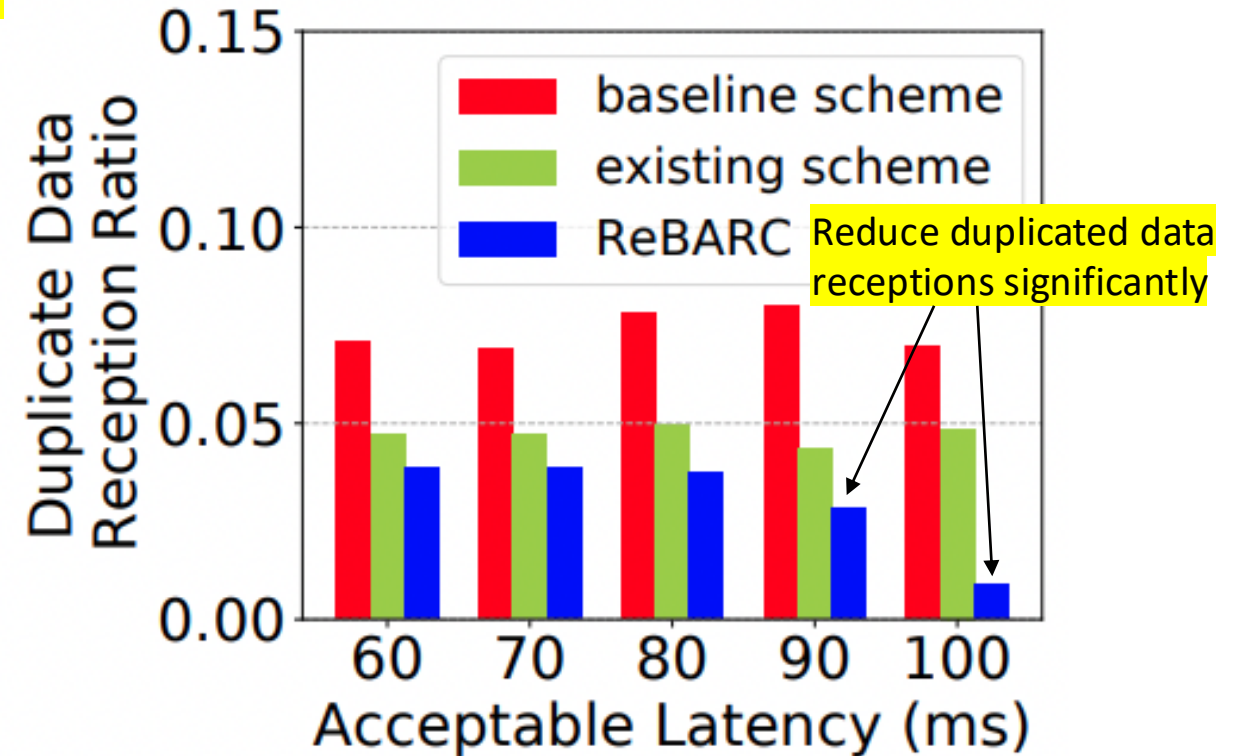
- Performance metrics
  - The ratio of **successful loss recovery** at the consumer
  - Ratio of **duplicate data reception**
- Schemes for comparison with our proposal (named *ReBARC*)
  - **Baseline scheme** : Consumer-based retransmission scheme
  - **Existing scheme** : In-network proactive loss recovery complementing consumer-based retransmission scheme [3]



# Performance according to Latency Requirements



Successful loss recovery ratio



Duplicate data reception ratio

# Summary

---

- **Proposal of ICN-based in-network retransmission mechanism**
  - **Key Features**
    - Introduce a metric called recovery budget
    - Self-regulating transmission of recovery interests
    - Consider the tradeoff between recovery delay and duplicated data transmission
- **Evaluation**
  - Experiments on Mininet with *Cefore*
  - Suppress duplicate data receptions while achieving successful loss recovery
- **Future Work**
  - Develop an effective rate control scheme using the in-network retransmission