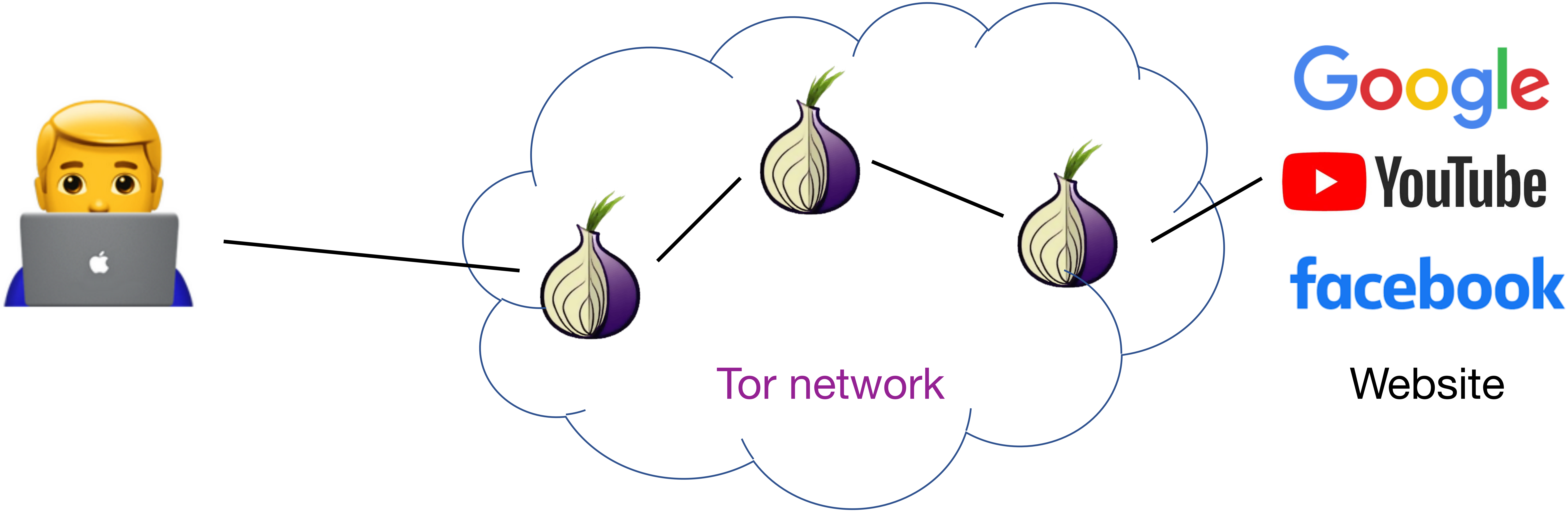
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

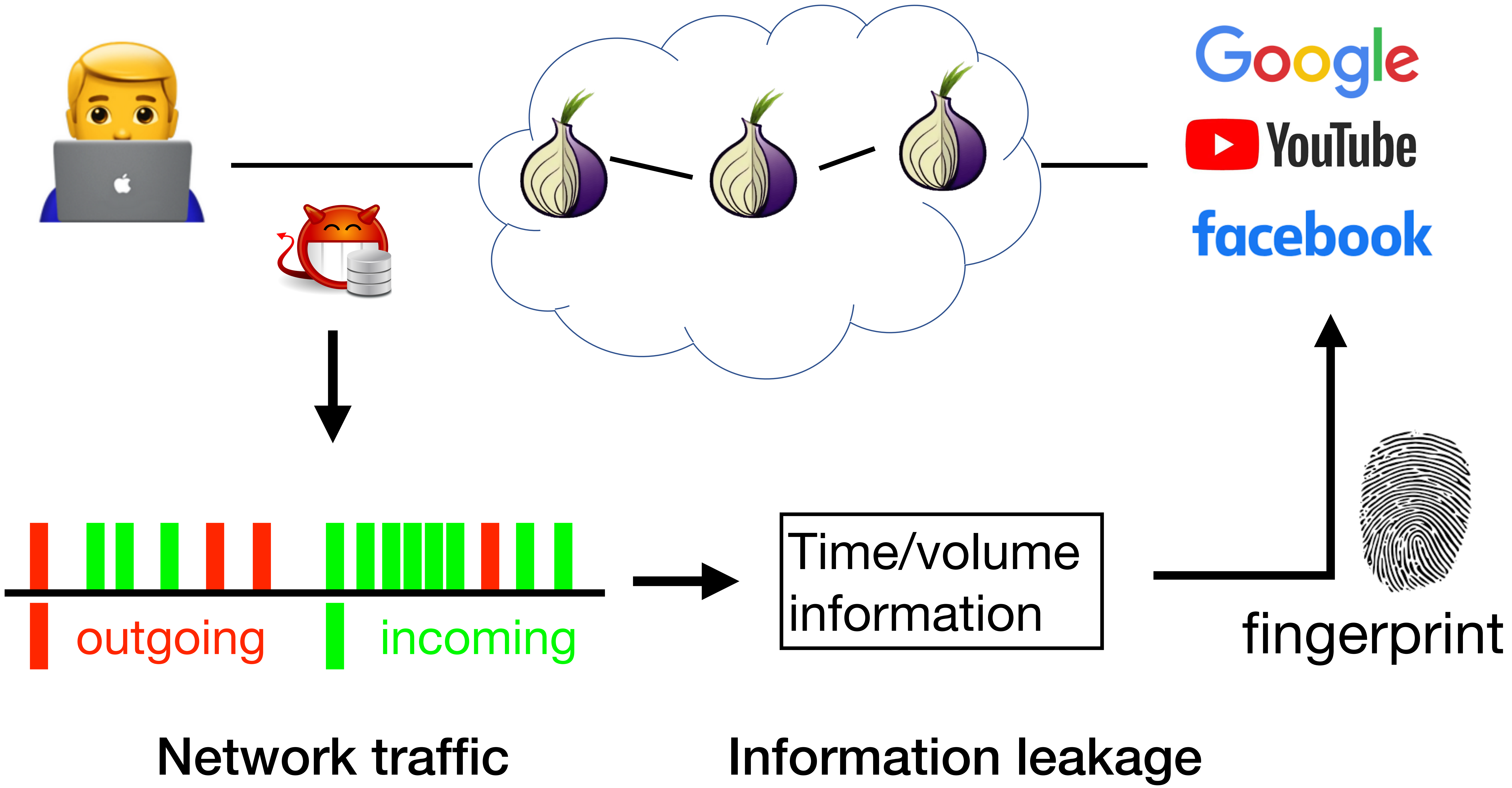
Zero-delay Lightweight Defenses against Website Fingerprinting

Jiajun GONG, Tao Wang

Website Fingerprinting

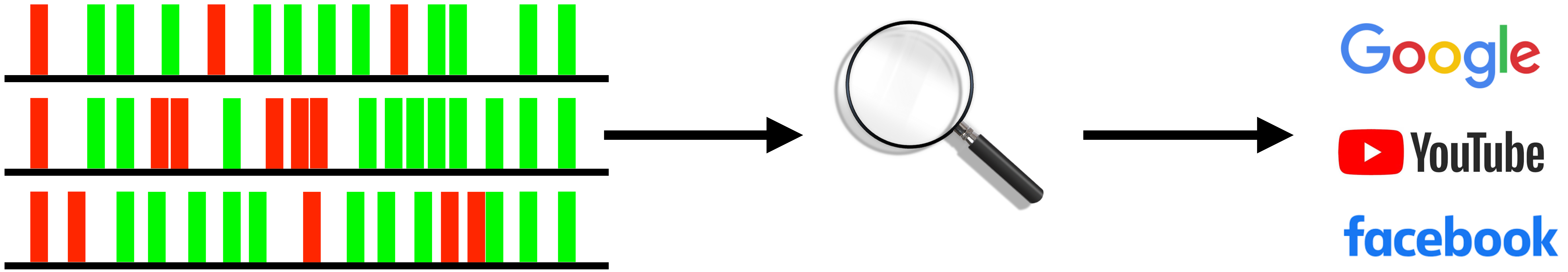


Website Fingerprinting (WF)



WF attackers: ISP, someone under the same network

Website Fingerprinting



1. Collect data 2. Train a classifier 3. Predict

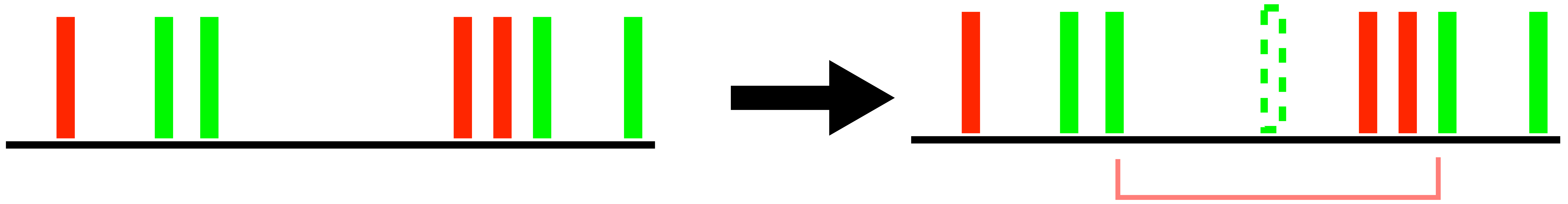
- kNN [Wang et al., 2014]
- CUMUL [Panchenko et al., 2016]
- kFP [Jamie Hayes and George Danezis, 2016]
- DF [Sirinam et al., 2018]

Threat to privacy!

> 90% recall

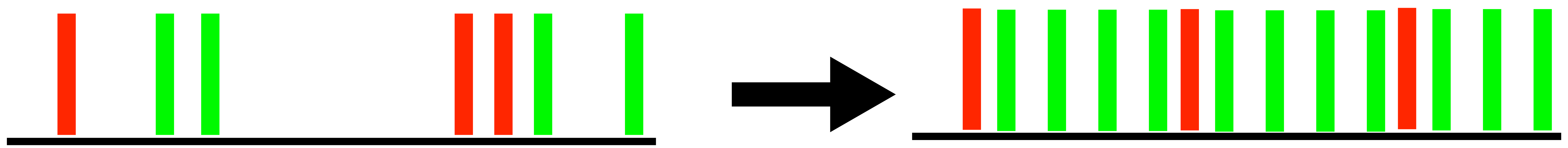
Defense

- WTF-PAD [Juarez et al. 2016]



Broken by DF!

- Tamaraw [Cai et al. 2014]



Too expensive

Evaluation of a defense

- Privacy
- Overhead:

$$\text{data overhead} = \frac{\text{\# dummy packets}}{\text{\# real packets}}$$

cost more bandwidth

$$\text{time overhead} = \frac{t_{new} - t_{old}}{t_{old}}$$

causing delay

→ Browsing experience

Defense

Question: Better defense?

0% time overhead

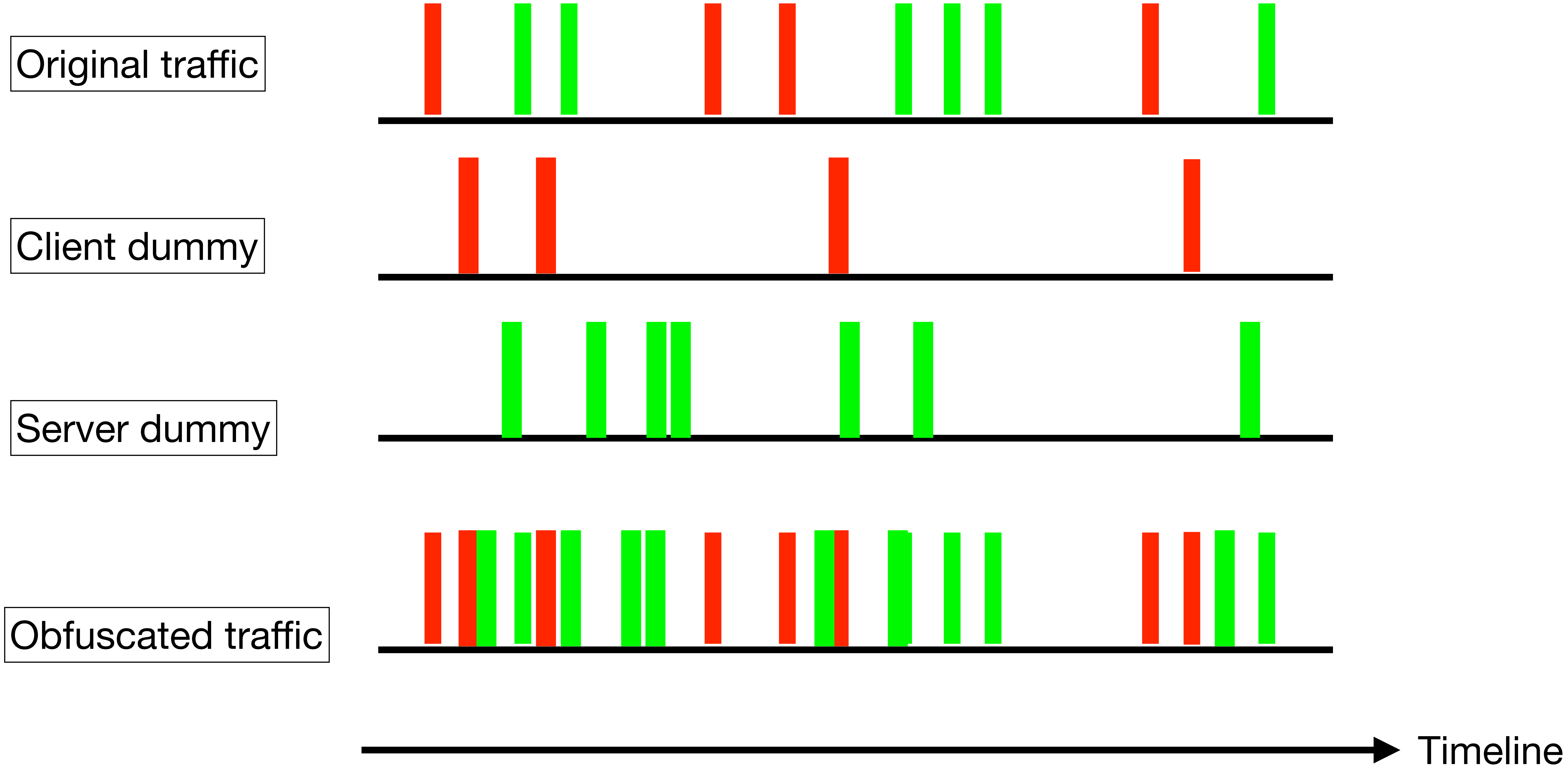
little data overhead

We proposed two zero-delay lightweight defenses:

FRONT and GLUE

FRONT

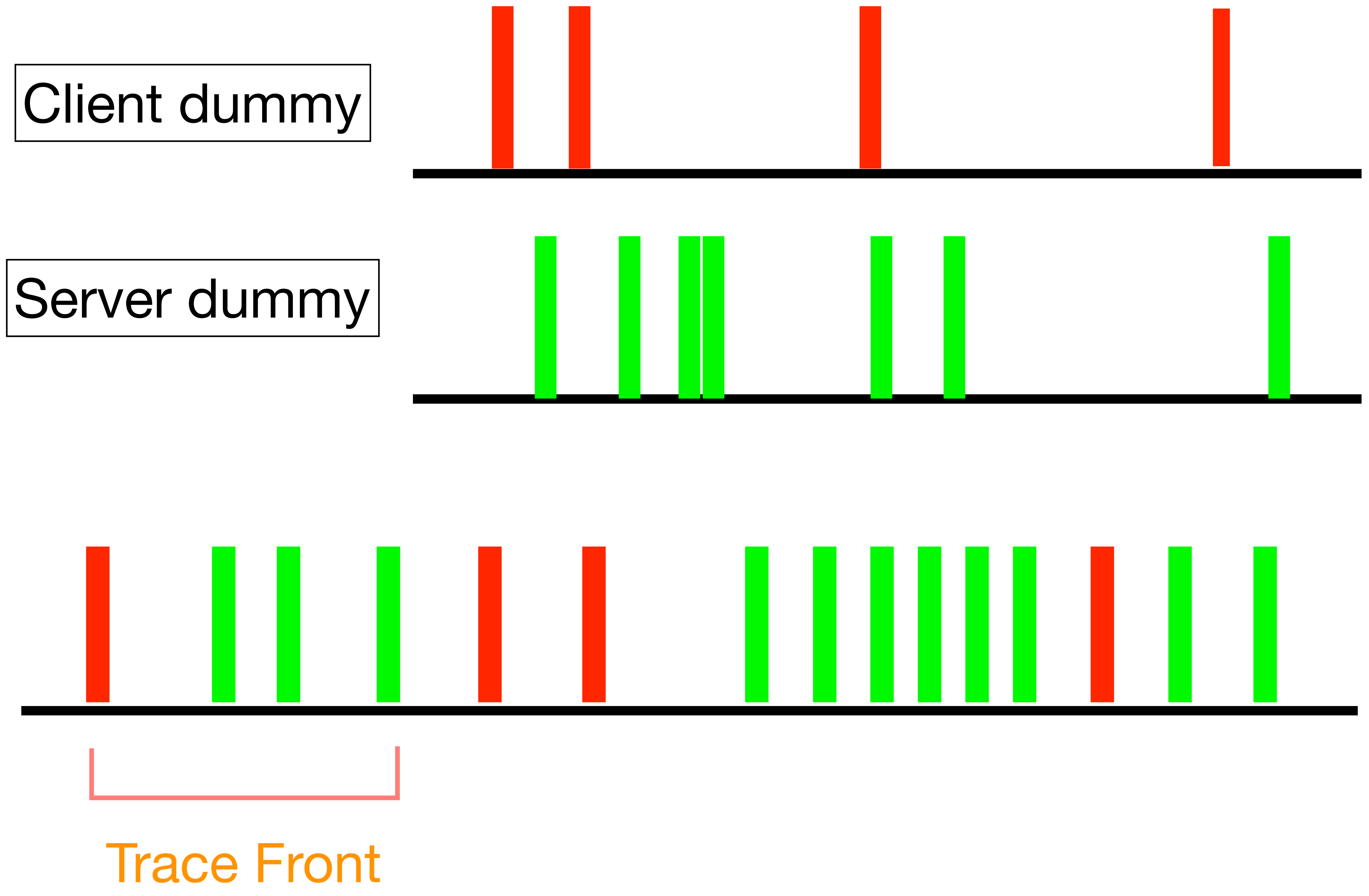
- General Idea



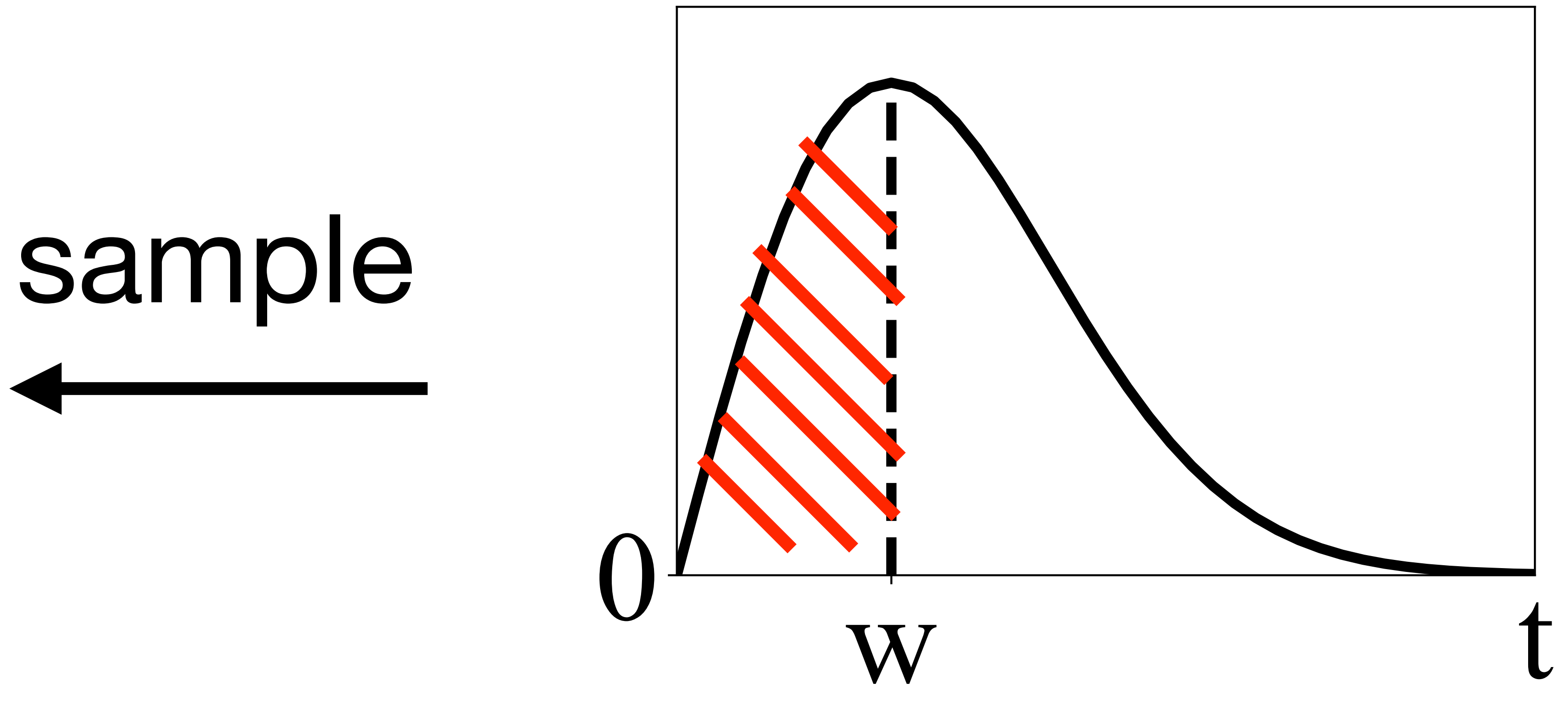
FRONT

How to schedule these dummy packets?

Intuition 1: Obfuscating feature-rich trace fronts



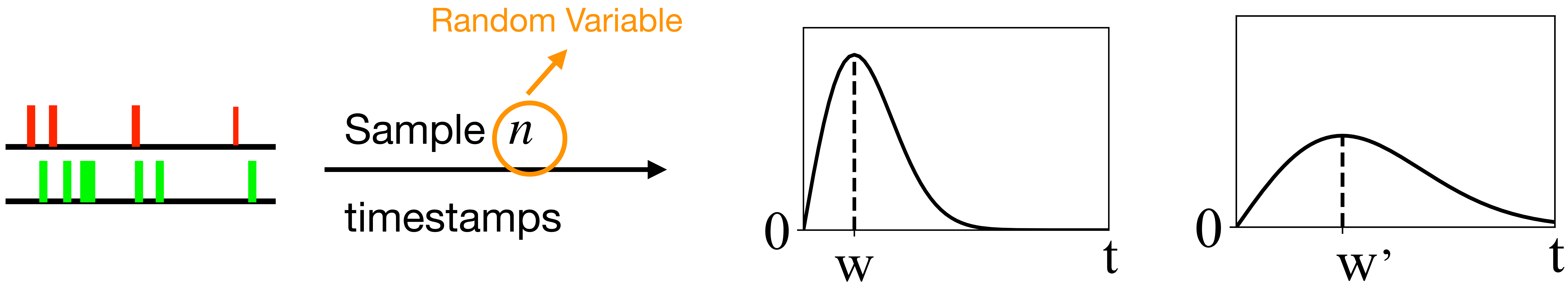
Rayleigh Distribution



Why Rayleigh distribution?

$$Pr(0 < t \leq w) = 40\,\%$$

FRONT



$$f(t; w) = \frac{t}{w^2} e^{-t^2/2w^2} (t > 0)$$

Random Variable

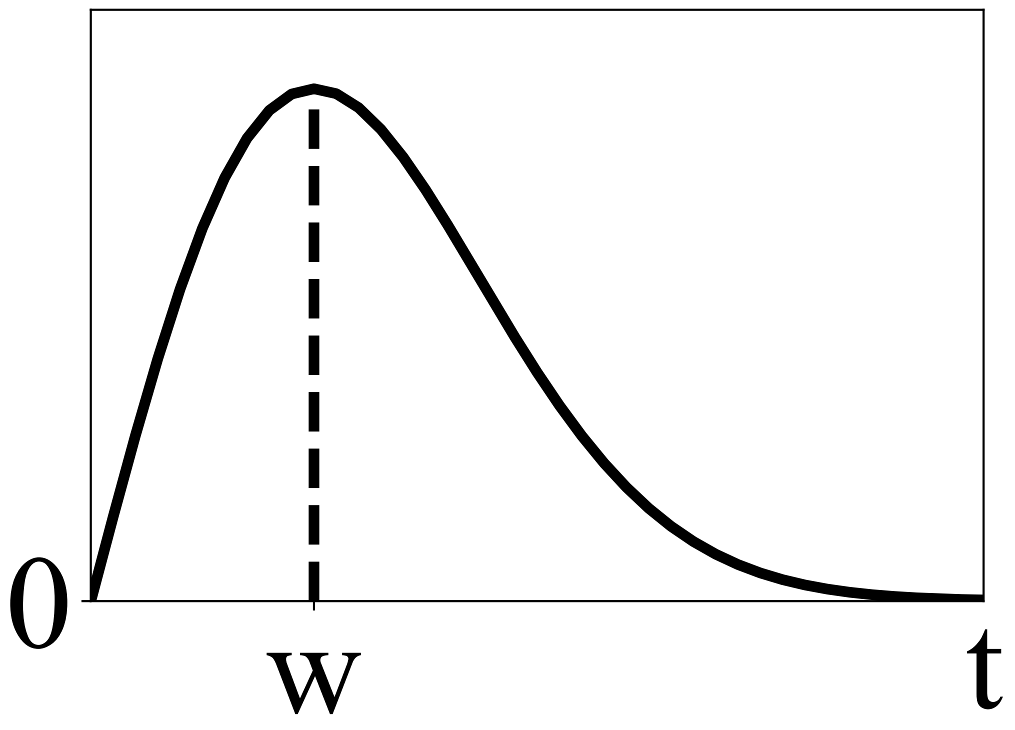
FRONT

- Set parameters: N, W_{min}, W_{max}
- Sample a number of dummy packets

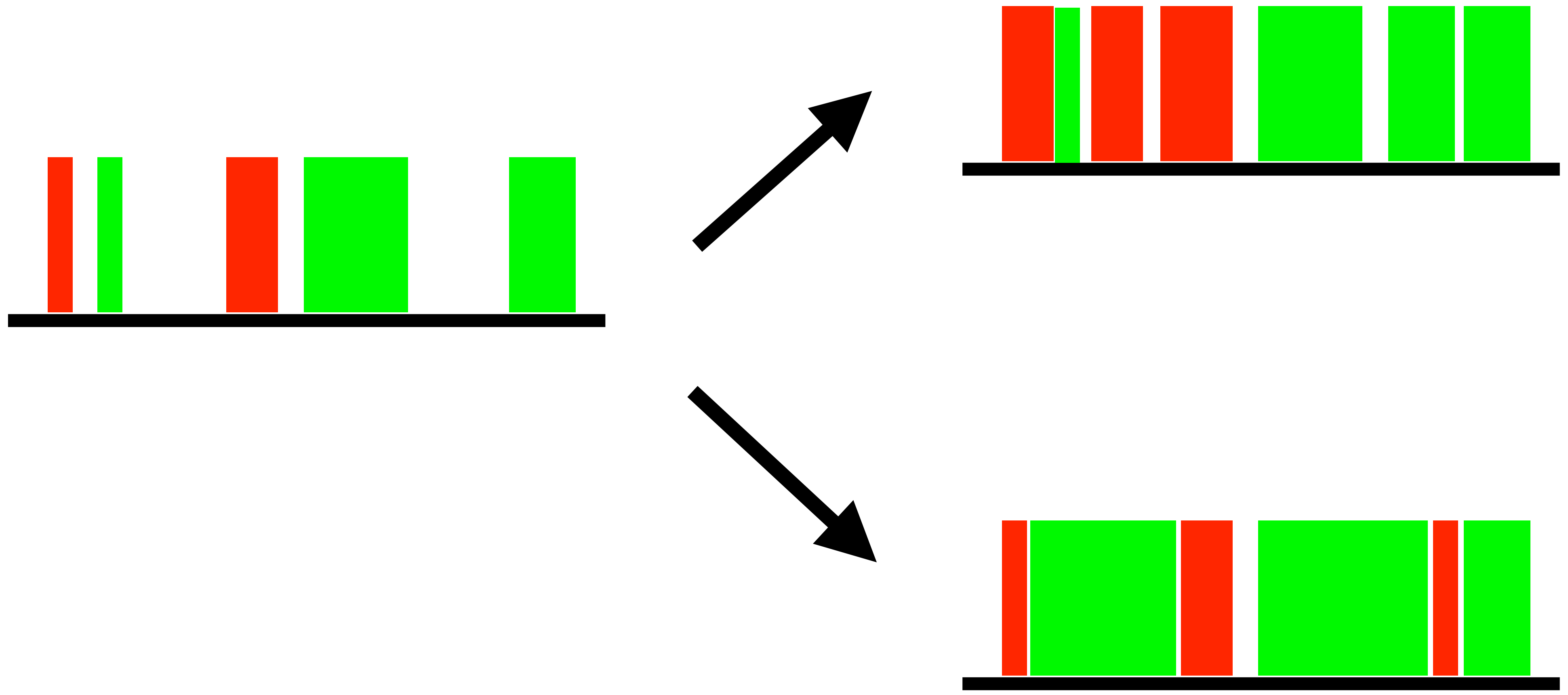
  $n \propto$ Uniform$(1, N)$

- Decide the shape of distribution

  $w \propto$ Uniform(W_{min}, W_{max})



- Sample n timestamps



Intuition 2: Trace-to-trace randomness

Experiment Setup

Dataset: 100 x 100 + 10000

Monitored non-monitored

90% training , 10% testing

Attacker's goal:

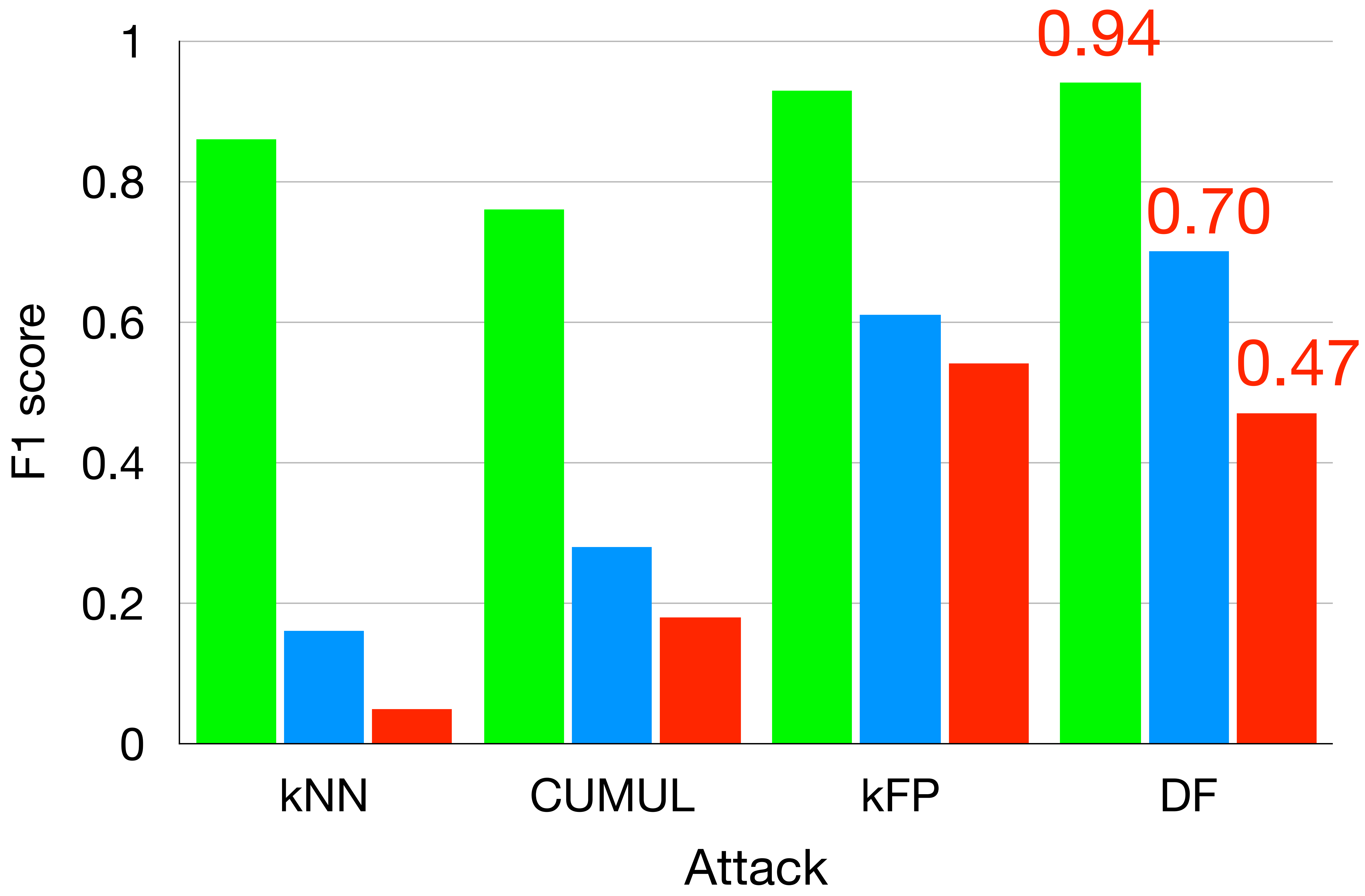
To identify whether the client is visiting a monitored page

and which monitored webpage?

Experiment Result

- Compared with WTF-PAD:

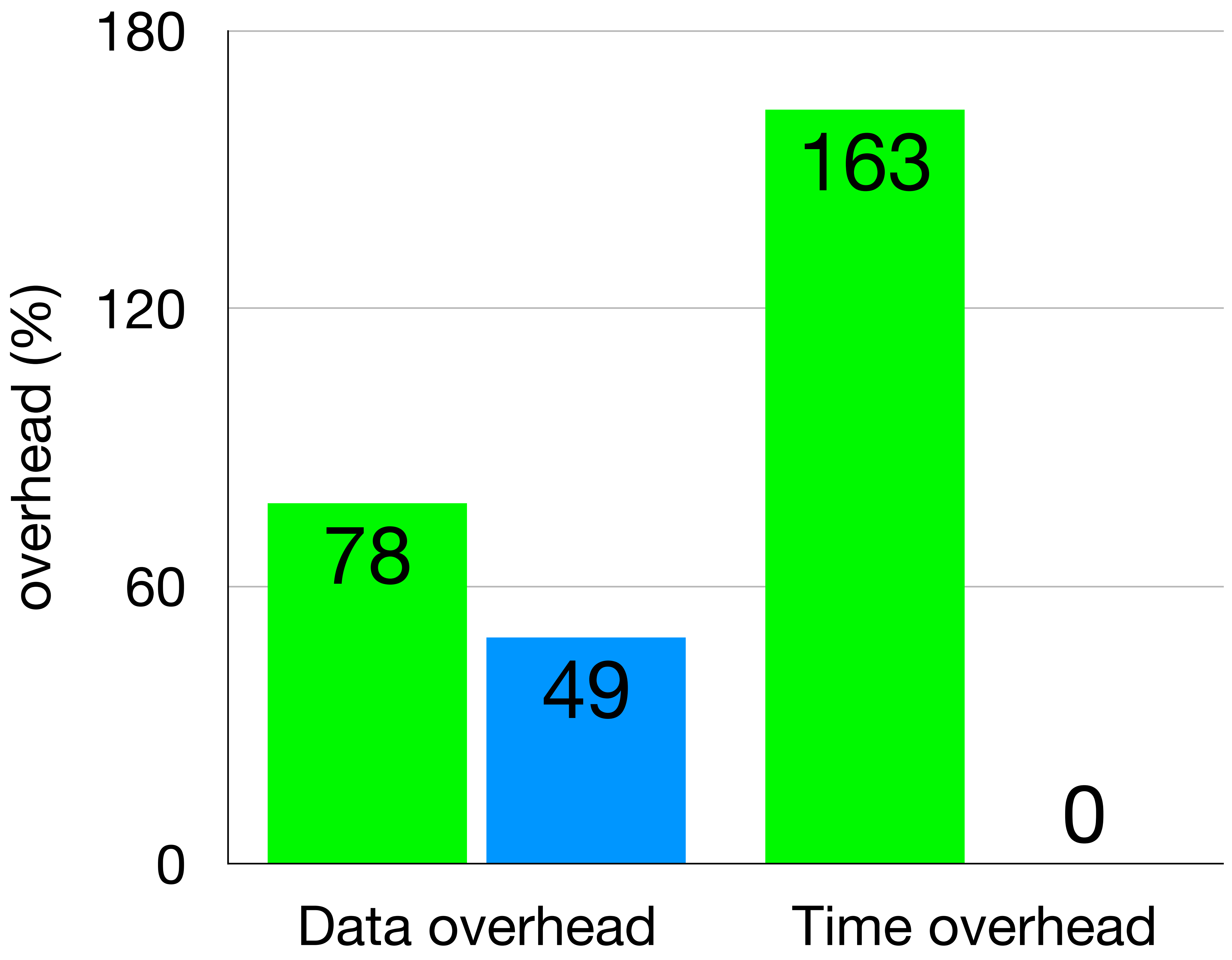
  ~ 33% data overhead, 0% time overhead



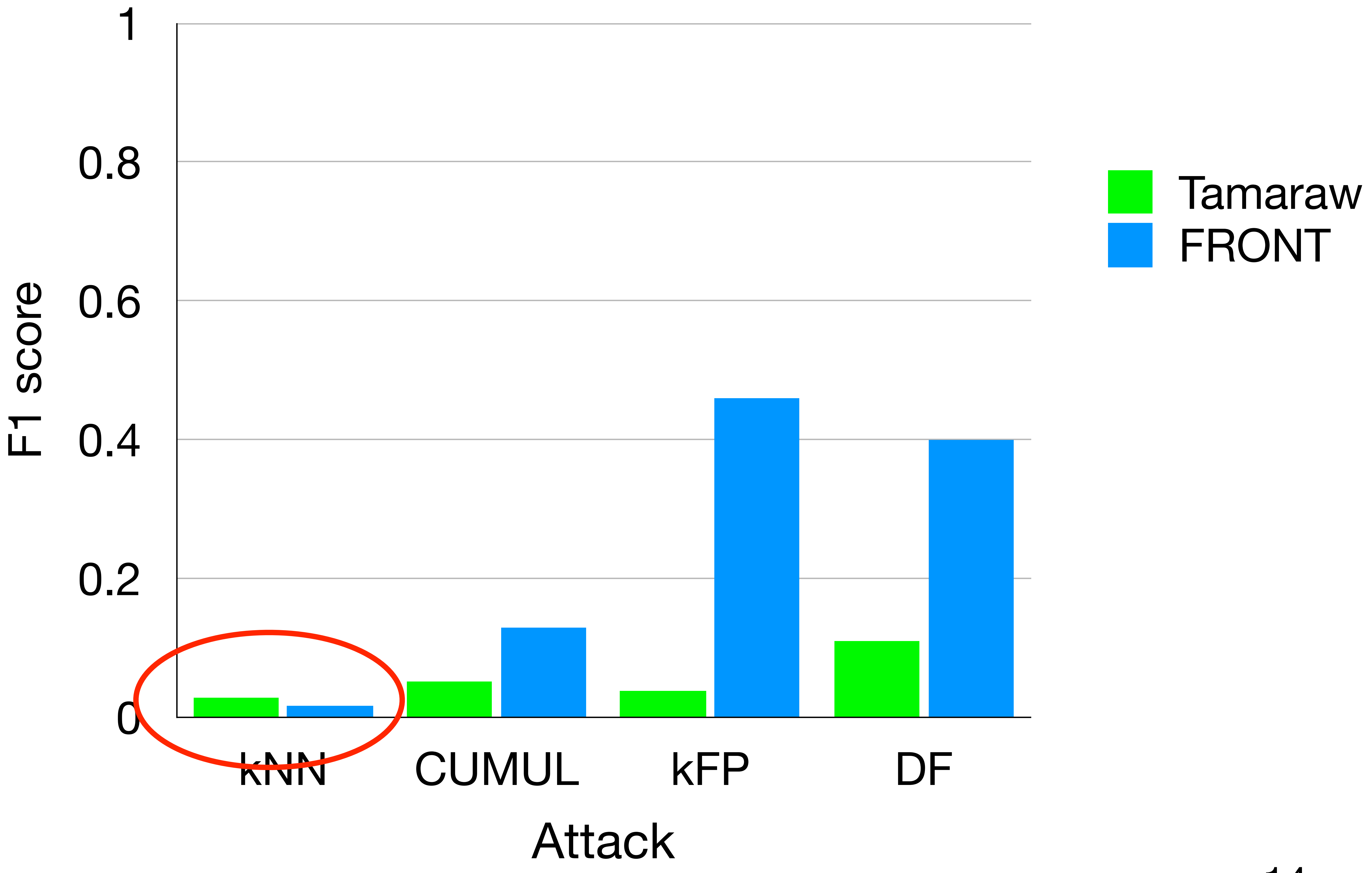
$$F1 = \frac{2 \cdot \text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}}$$

Experiment Result

- Compared with Tamaraw:



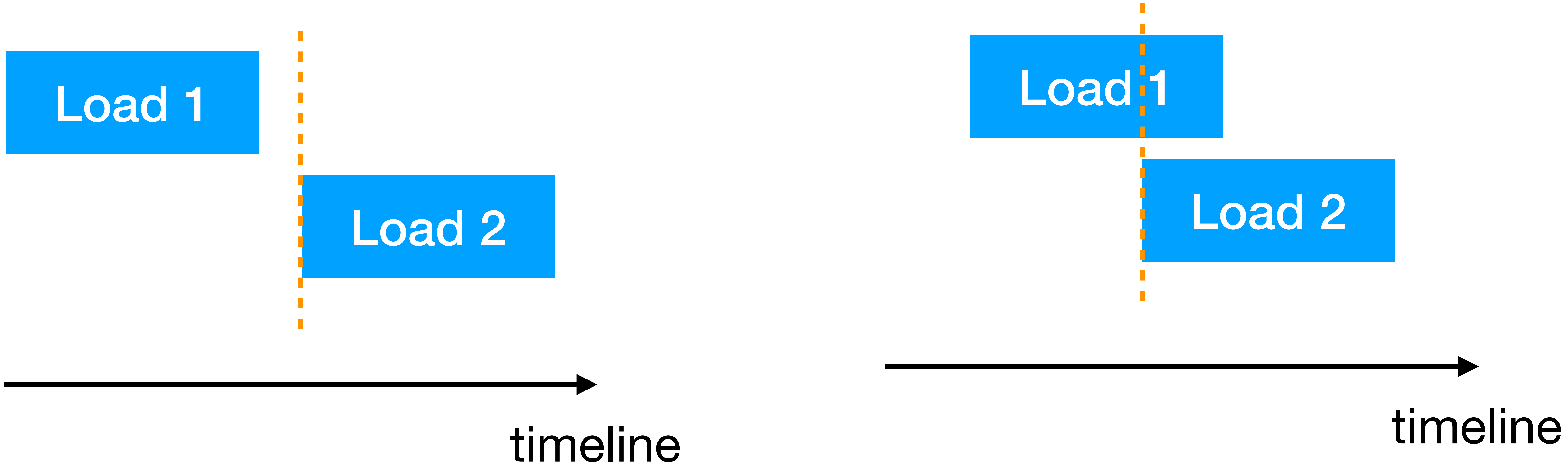
~ 5 times



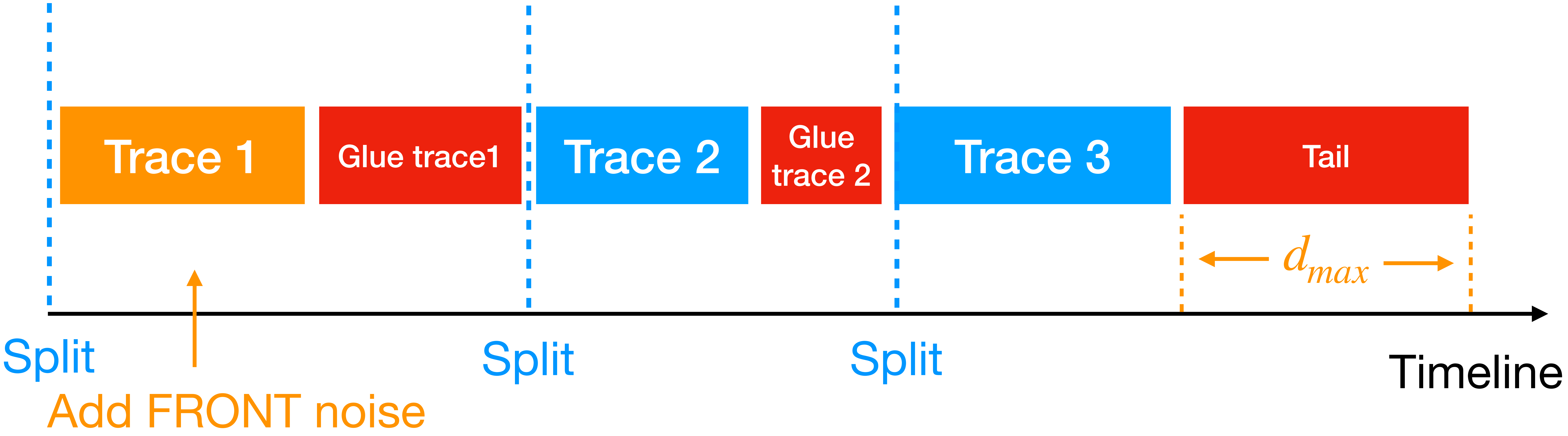
GLUE

Intuition:

difficulty of solving split problem [Juarez et al. 2014, Wang et al. 2016]



GLUE



Trace 1
Glue trace1
Trace 2
Glue trace 2
Trace 3
Tail
d_{max}
Split
Split
Split
Add FRONT noise
Timeline

GLUE

- Cover the first loading with FRONT
- "Glue" all the visits with glue traces
    - fake loading, obtained by storing the history of some webpages loaded before
- Maximum duration of a glue trace: $d_{max} \propto \text{Uniform}(t_{min}, t_{max})$

Evaluation

Scenario 1: knowing ℓ

- Randomly generated 618 ~ 4500 ℓ-traces (ℓ=2~16)
- Undefended dataset:

  82% ~ 96% recall and precision (92% split accuracy)

- GLUE dataset:

  4% ~ 54% recall and 4% ~ 20% precision

Evaluation

Scenario 1: without knowing ℓ (more realistic)

- Undefended dataset:

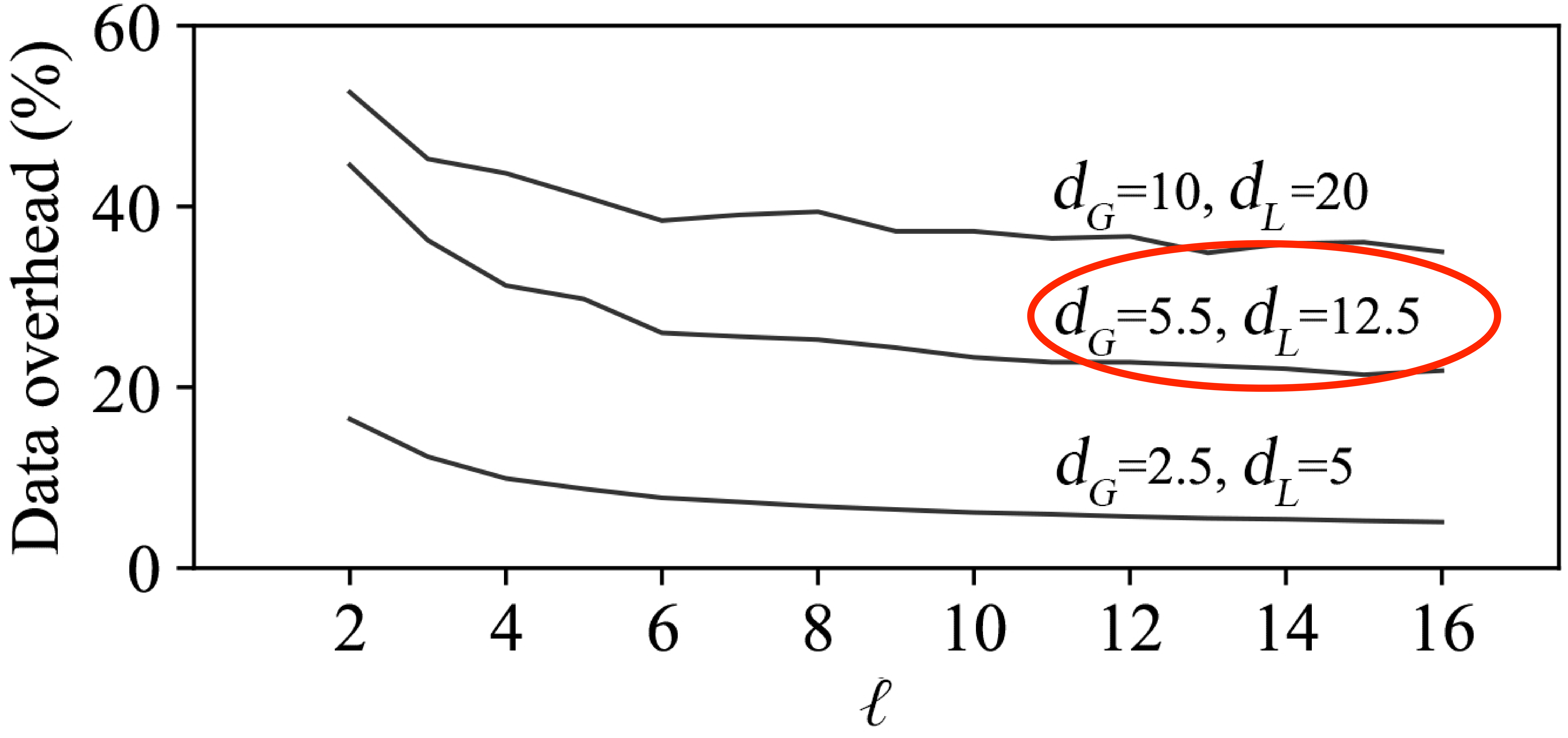
  45% ~ 75% recall and 41% ~ 77% precision

- GLUE dataset:

  3% ~ 46% recall and 1% ~ 16% precision

Overhead of GLUE

- Time overhead 0%.
- Suppose:
  - mean dwell time d_G, mean duration of tail d_L



22-44% data overhead

Summary

- Proposed two lightweight zero-delay defenses:
  - FRONT injects dummy packets in a traditional way
    - Obfuscating the trace fronts
    - Trace-to-trace randomness
  - GLUE explores a new direction for designing a defense
    - Forces the attacker to solve the split problem

- Source code

  https://github.com/websitefingerprinting/WebsiteFingerprinting/

- Contact info:

  jgongac@cse.ust.hk

Thanks for listening!