Inferring Browser Activity and Status Through Remote Monitoring of Storage Usage

Hyungsub Kim*†, Sangho Lee§, Jong Kim*

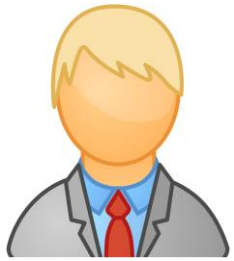
*** Pohang University of Science and Technology (POSTECH)**
† Agency for Defense Development (ADD)
§ Georgia Institute of Technology

ACSAC 2016

Presenter: Hyungsub Kim

8 December 2016

Web browser cache



<User>

First visit

Open Tab

acsac.org

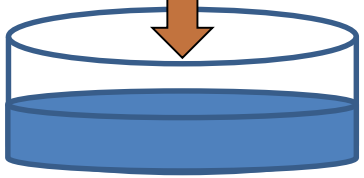
Acsac 2016

https://www.acsac.org

1. Visit acsac.org

2. Fetch the resources of acsac.org

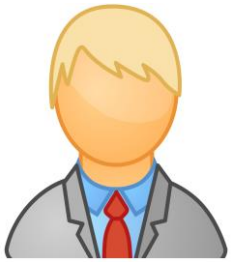
3. Store the resources



<Browser cache>

1) HTML
2) JavaScript code
3) Images
4) CSS
5) Media files
6) etc.

Web browser cache



<User>

Visit again

Open Tab

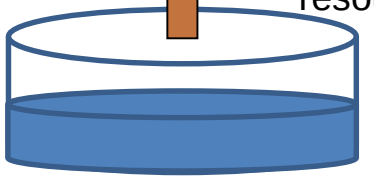
acsac.org

Acsac 2016

https://www.acsac.org

1. Visit acsac.org

2. Load cached resources



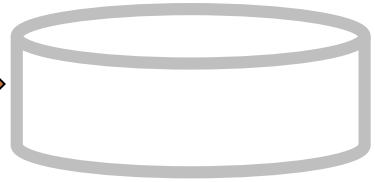
<Browser cache>

1) Not fetch the resources

2) Reducing
- Network traffic
- Load time of web pages

Quota Management API

- **Support other HTML5 storage APIs**
  - AppCache, ServiceWorker, IndexedDB
- **Manage and monitor available storage space in a web browser**
- **Two types of storage space**
  - ① persistent storage ② **temporary storage**

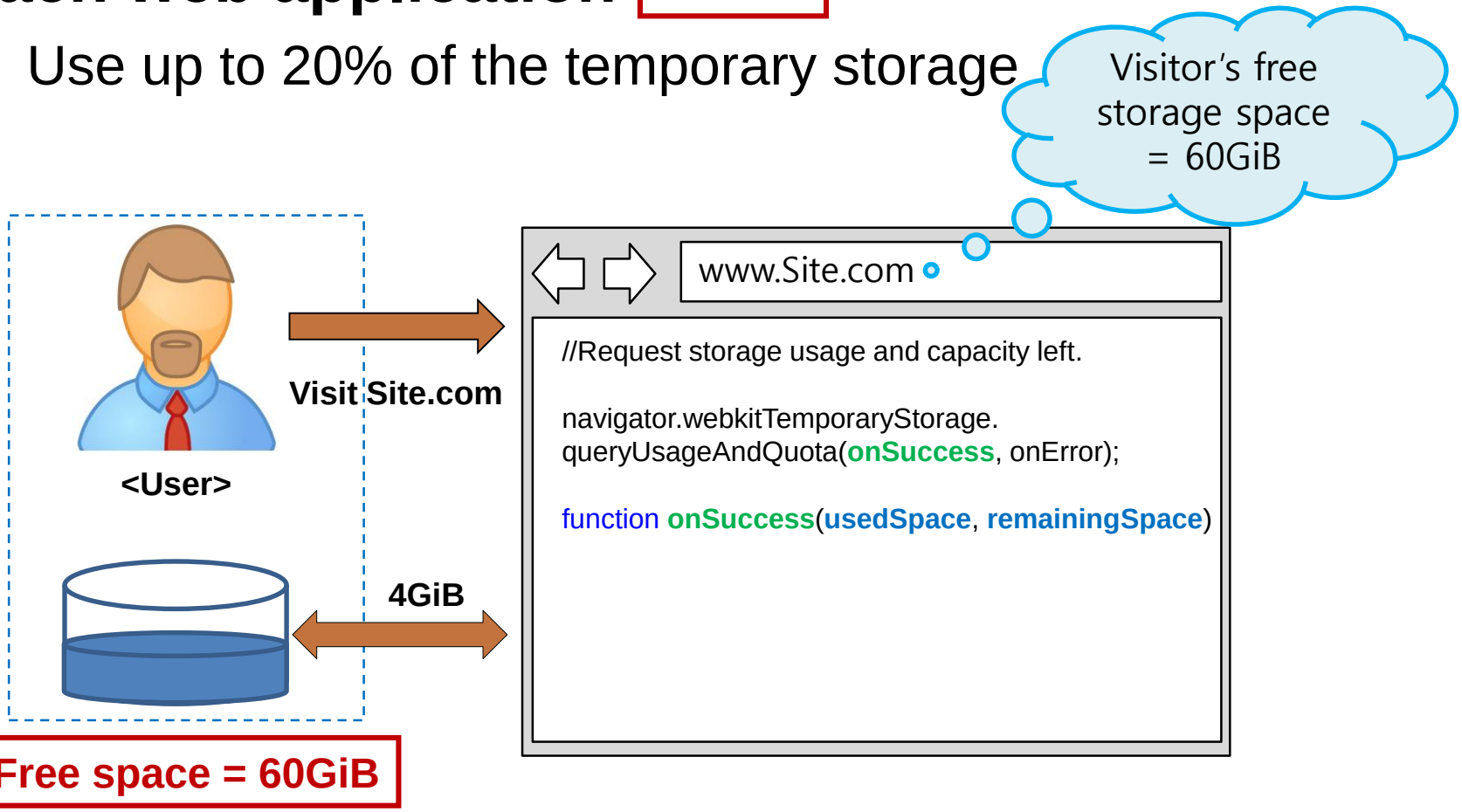


<Temporary storage>

Problem :
All web applications on a browser share temporary storage

Quota Management API

- **Temporary storage space** **= 20GiB**
  - Approximately 33% of available storage space
- **Each web application** **= 4GiB**
  - Use up to 20% of the temporary storage



Motivation & goal

- **Motivation**
  - Quota Management API's privacy problem
    - Promptly return precise remaining space of shared storage

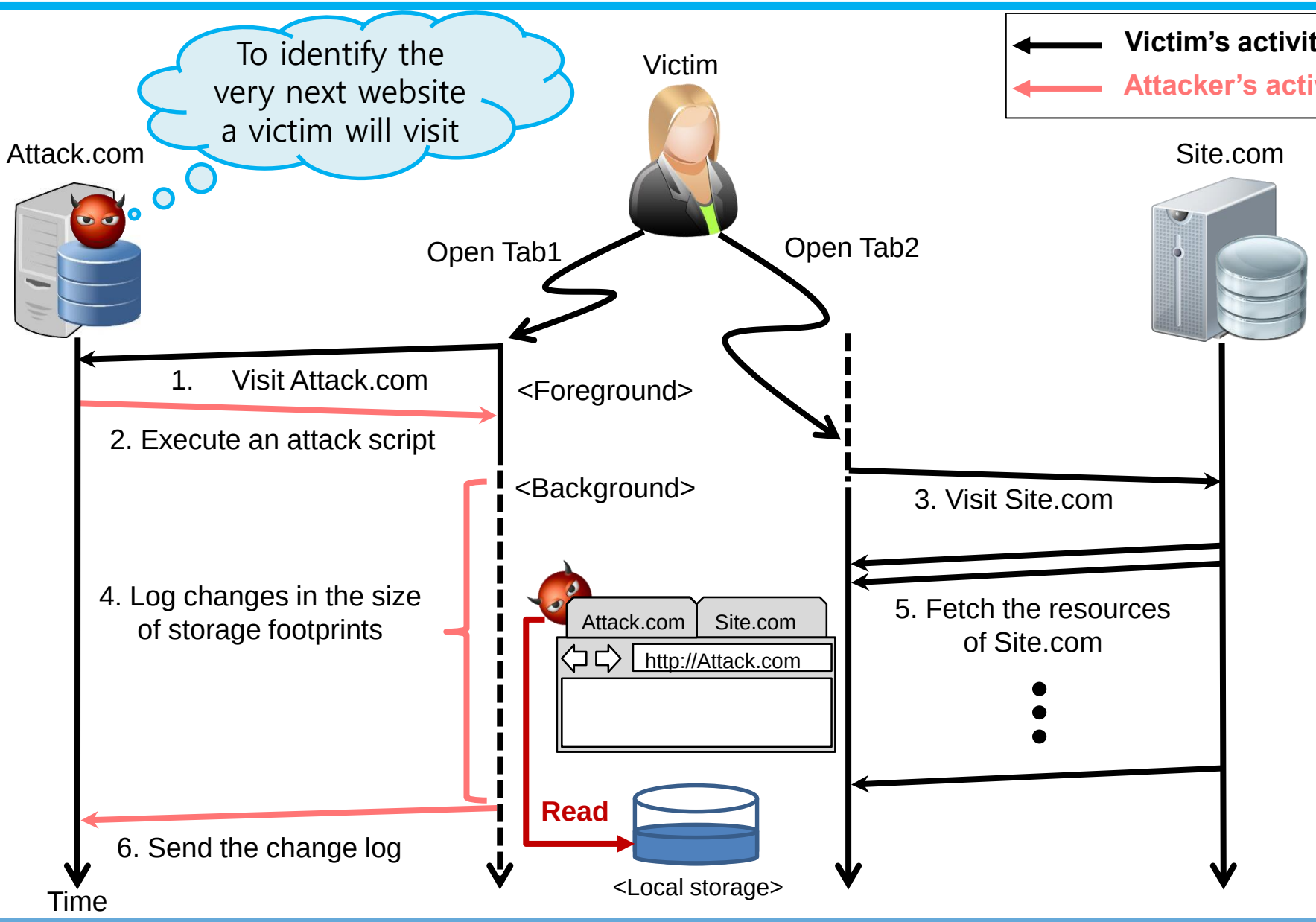
Average sampling rate: 20ms **Byte unit**

- **Goal**
  1. Attack1: infer cross-tab activity
     - Identify the very next website a victim will visit
  2. Attack2: infer browser status
     - Browsing history
     - Login status
     - Friend and group relationship on Facebook
  3. Countermeasure

Outline

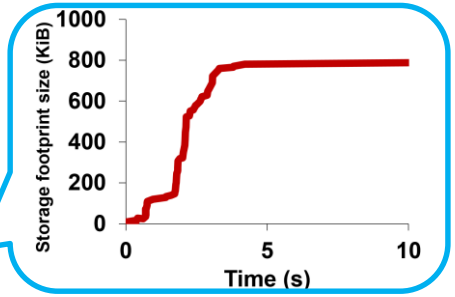
1. **Cross-tab activity inference**
2. Browser status inference
    1) Browser history
    2) Login status
    3) Friendship and group membership
3. Countermeasure

Attack procedure



Attack procedure

Different web pages use the local storage differently in terms of size and time



Attack.com

Victim

Site.com

Open Tab1

Open Tab2

1. Visit Attack.com

<Foreground>

2. Execute an attack script

<Background>

3. Visit Site.com

4. Log changes in the size of storage footprints

Attack.com | Site.com

http://Attack.com

5. Fetch the resources of Site.com

Read

6. Send the change log

<Local storage>

Time

Demonstration video

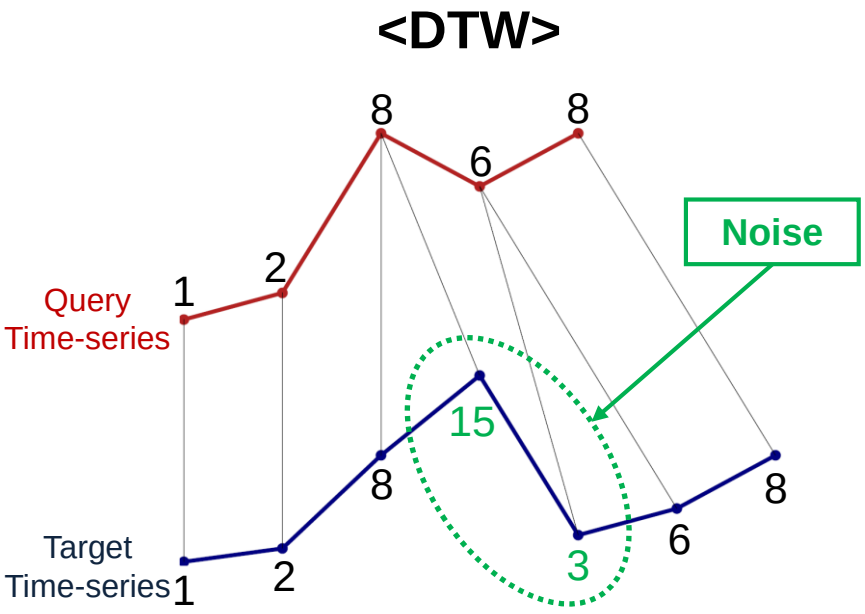
Attacker's server

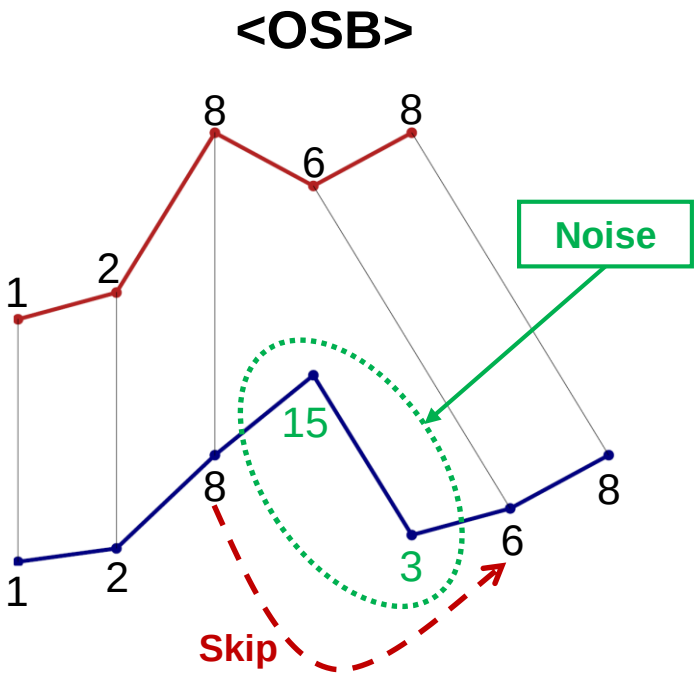
Victim's browser

https://kimhyungsub.github.io/quota_passive.html

Analyze time series

- Dynamic time warping (DTW)
- Optimal subsequence bijection (OSB)
  - Skip outliers of query and target time-series
  - Effectively deal with noise



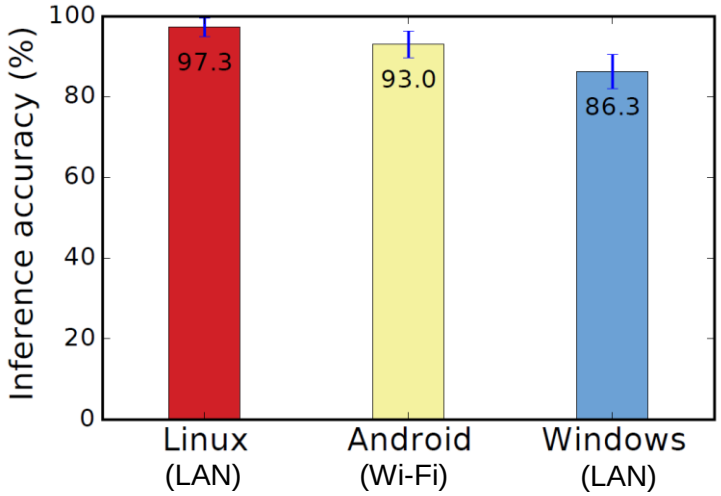


Inference accuracy

1. Visit each front page of Alexa Top 100 web sites 10 times on each platform
2. Wait for a minute to finish page loading
3. Compare the query storage footprints with the attack databases

Query storage footprints

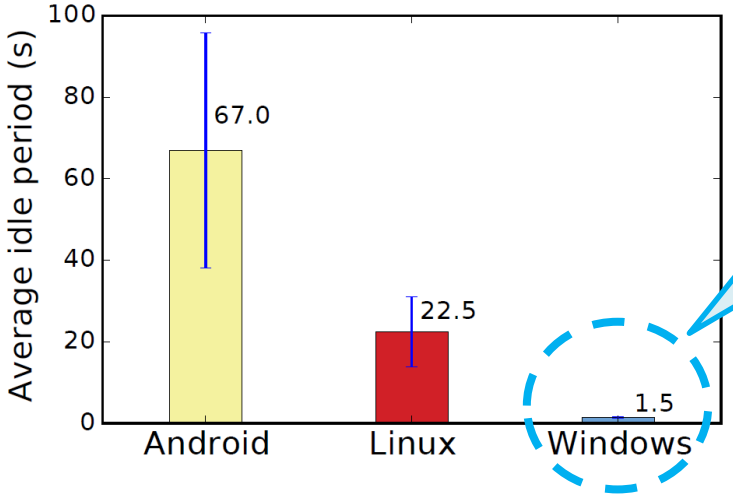
Average accuracy is 92%



<Inference accuracy of cross-tab activity attacks against non-cached websites>

Background disk activity

- Measure background disk activity within a hour
- Average idle periods
  - Android (67 s), Linux (22.5 s), and Windows (1.5 s)

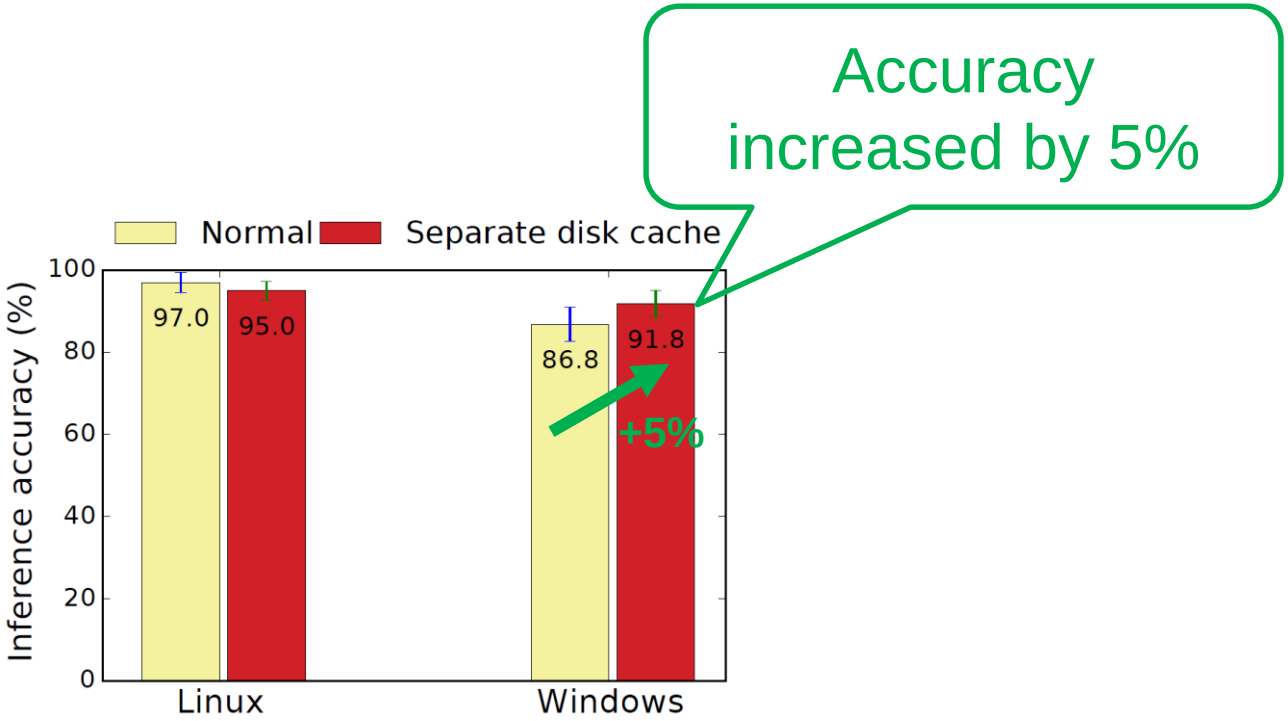


- Contain a large amount of noise
- Spoil our attack

<Statistics of idle periods>

Background disk activity

- Exclude the effects of background disk activity
  - Move the location of the browser cache to a separate disk

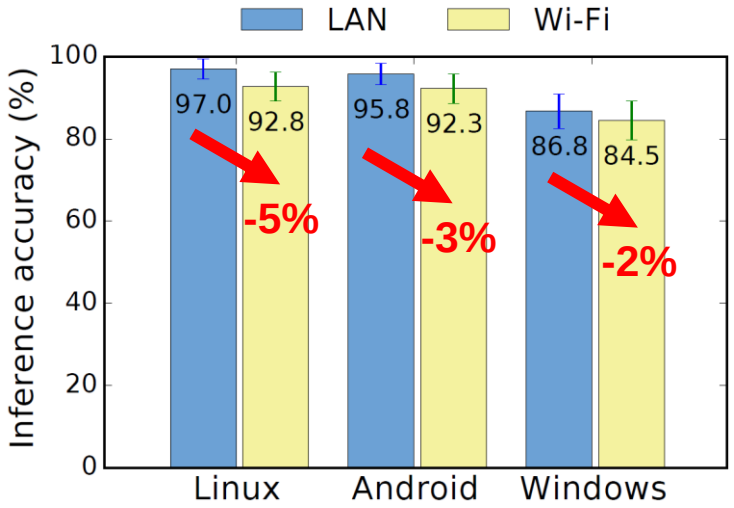


<Separate disk cache to ignore background disk activity>

Wireless network

- Wi-Fi is less stable than a wired LAN.
  - Suffer from high measurement noise

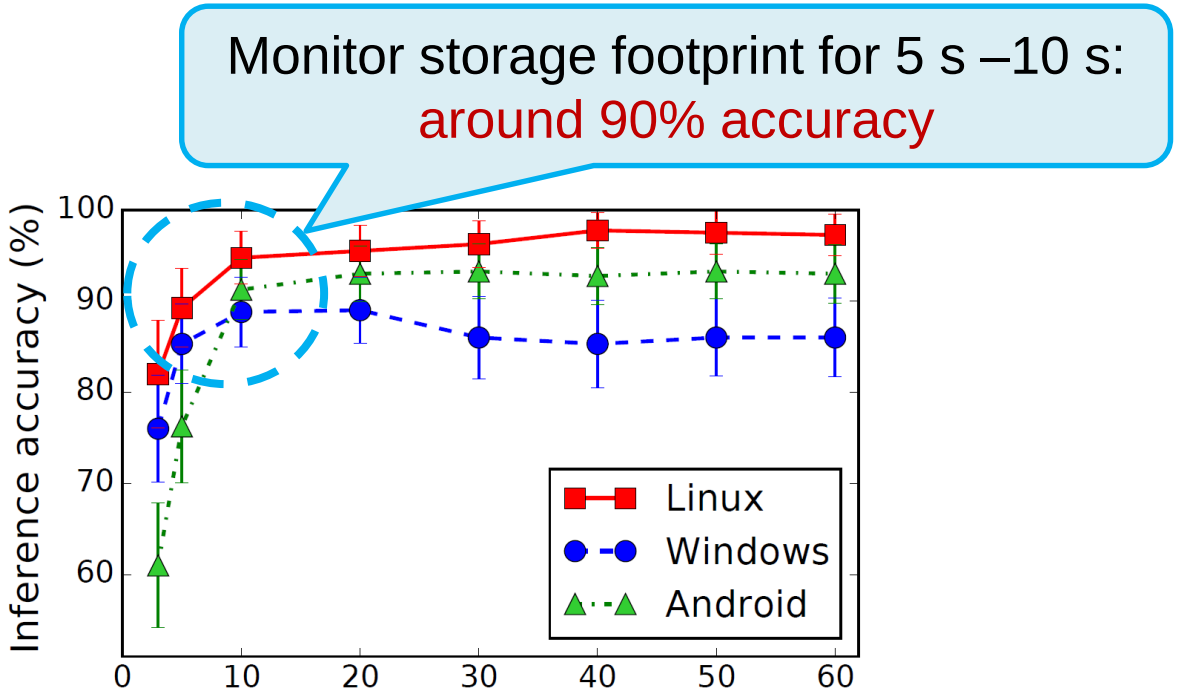
Average accuracy decreased by 3%



<Accuracy on wired and Wi-Fi networks>

Early inference

- Victim may not stay long in an attack page
- How fast can our attack infer the victim's activity?
  - Vary monitoring time from 3 s to 60 s

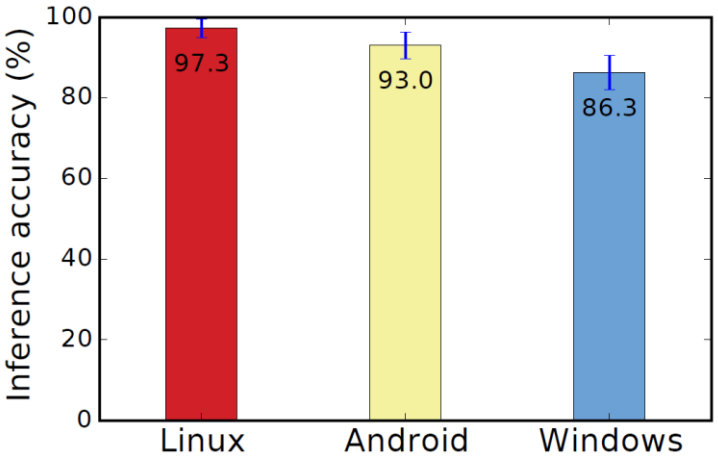


<Accuracy according to the length of monitoring time>

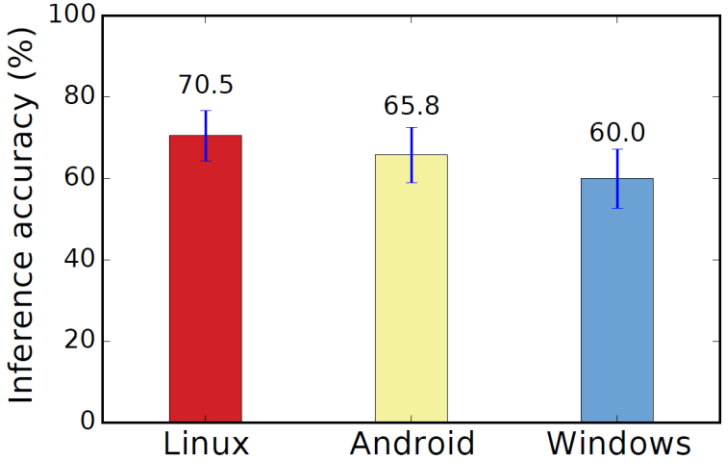
Identifying cached web sites

- Visit cached web sites
  - Only fetch and store dynamic or updated resources of the web sites
  - Changes in storage footprints are restricted

Average accuracy decreased by 26%



<Accuracy against **non-cached web pages**>



<Accuracy against **cached web pages**>

Outline

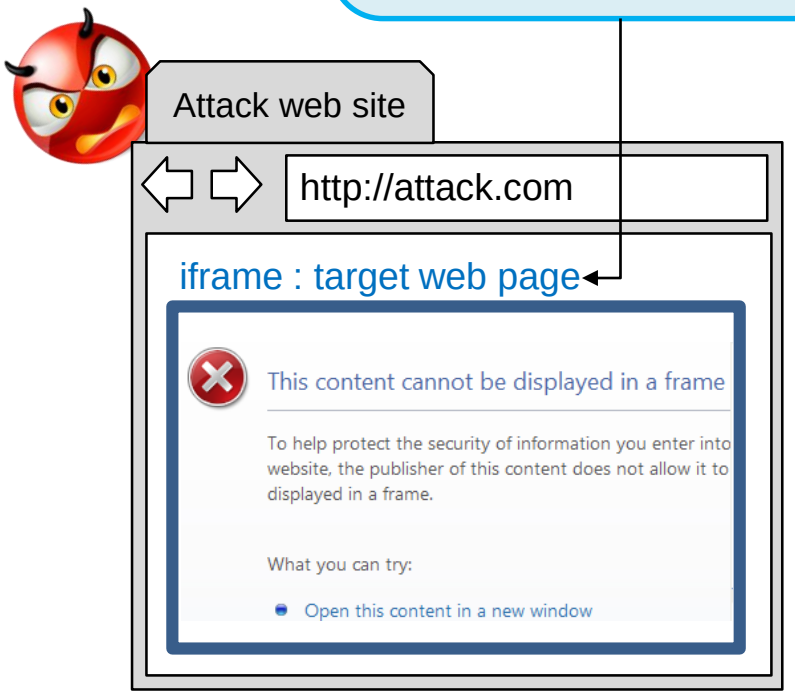
1. Cross-tab activity inference
2. **Browser status inference**
   1) **Browser history**
   2) Login status
   3) Friendship and group membership
3. Countermeasure

Load target web pages

- Conventional attacks such as clickjacking and timing attack use iframe

Web page can prevent an iframe tag from including itself

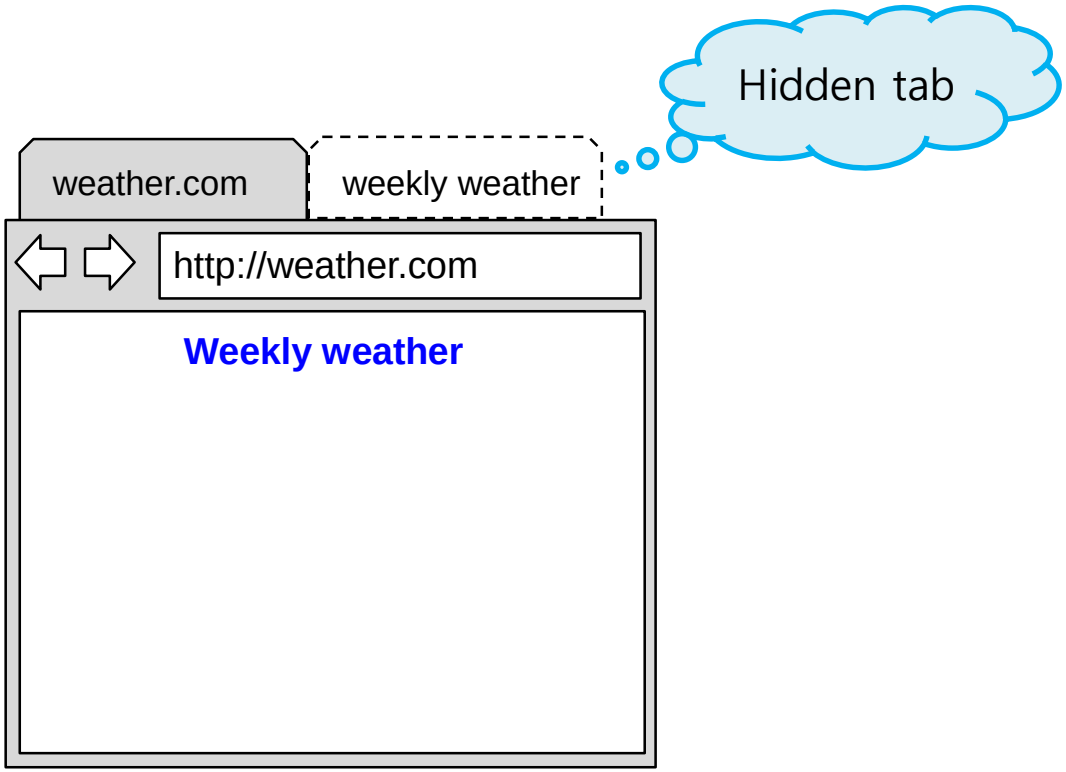
- HTTP header field X-Frame-Options
- Frame busting code



Prerendering

- Preload a web page in a hidden browser tab
- Reduce network and rendering delay

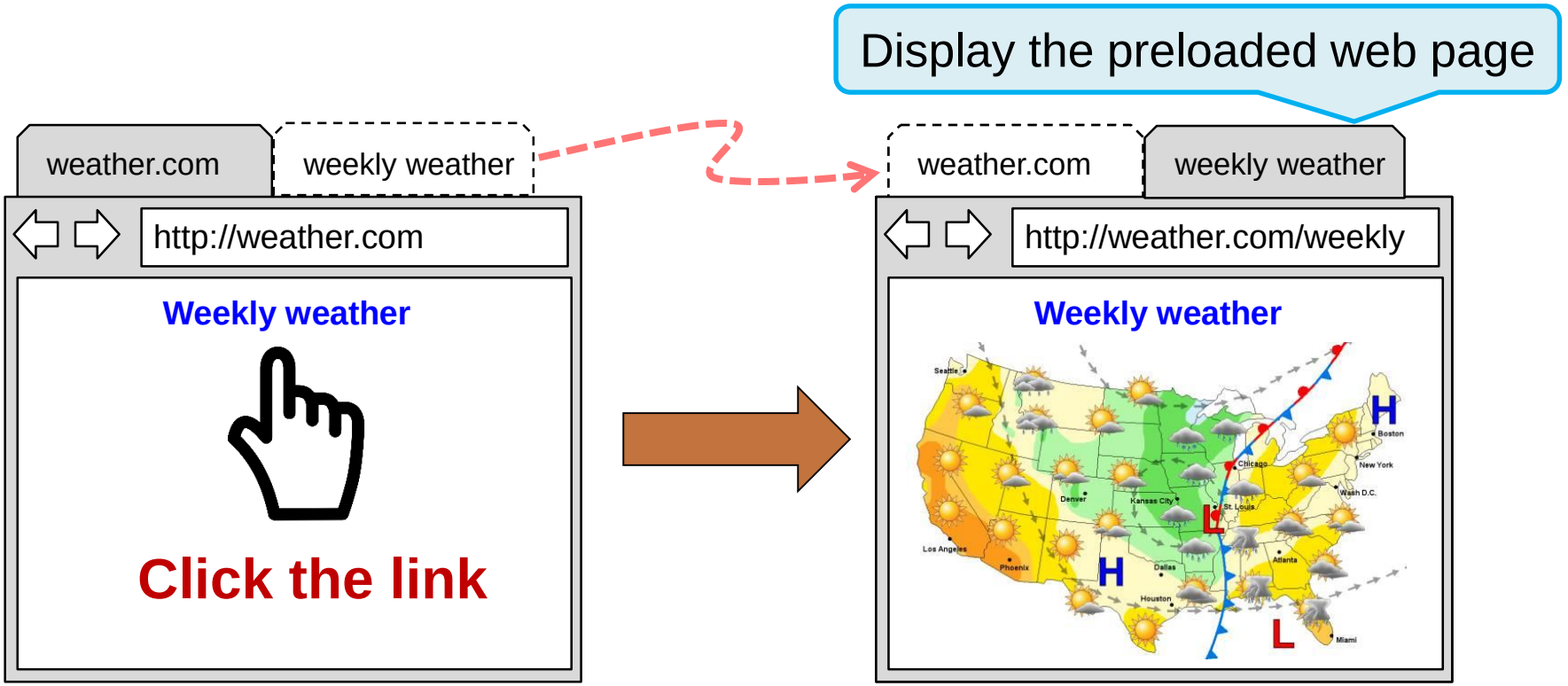
Web page cannot prevent prerendering from loading itself



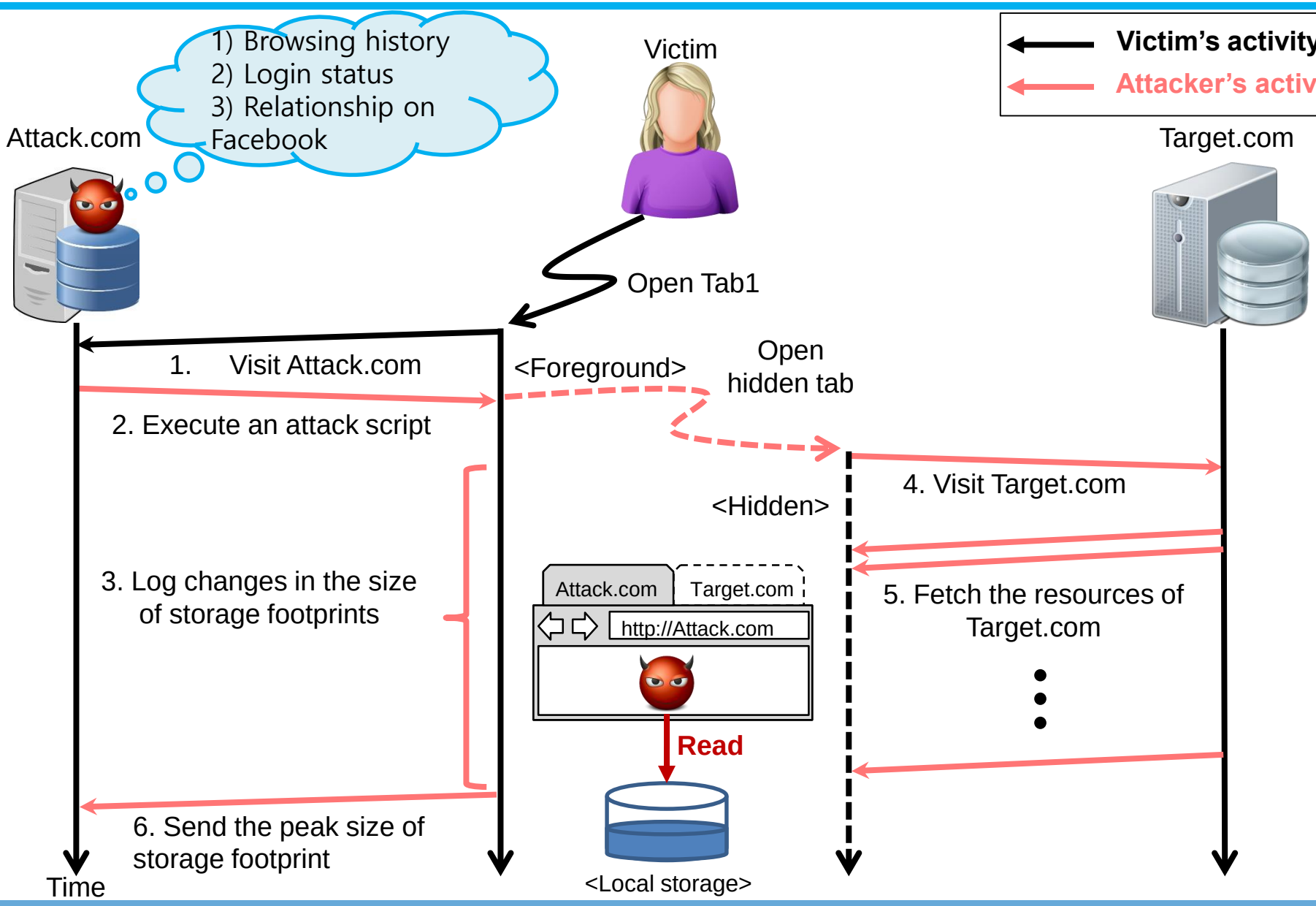
Prerendering

- Preload a web page in a hidden browser tab
- Reduce network and rendering delay

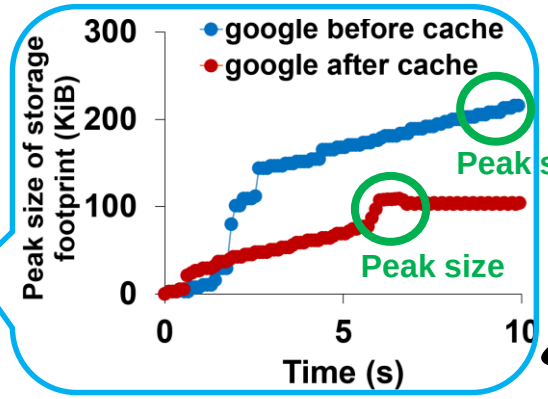
Web page cannot prevent prerendering from loading itself



Attack procedure



Attack procedure



Victim's activity

Attacker's activity

Attack.com

Victim

Target.com

Open Tab1

1. Visit Attack.com

<Foreground>

Open hidden tab

2. Execute an attack script

4. Visit Target.com

<Hidden>

3. Log changes in the size of storage footprints

Attack.com

Target.com

http://Attack.com

5. Fetch the resources of Target.com

Read

6. Send the peak size of storage footprint

Time

<Local storage>

Demonstration video

Attacker's server

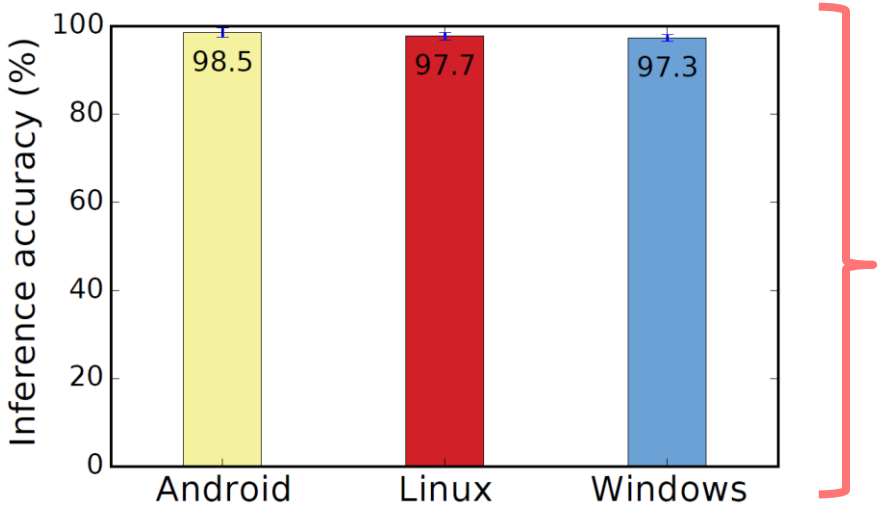
Victim's browser

https://kimhyungsub.github.io/quota_active.html

Inference accuracy

- Visit each front page of Alexa Top 500 web sites 5 times on each platform ← **Query storage footprints**
- Compare the peak size of query storage footprints with the attack databases

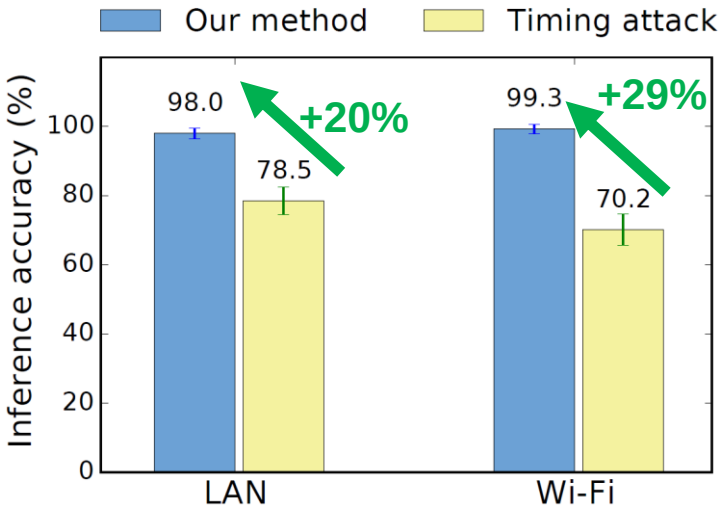
Always observe high inference accuracy regardless of OSes



<Inference accuracy of browser history stealing>

Our method vs. Timing attack

- Our method
  - Use storage footprints
  - Not affected by network condition
- Timing attack
  - Use page load times
  - Affected by network condition



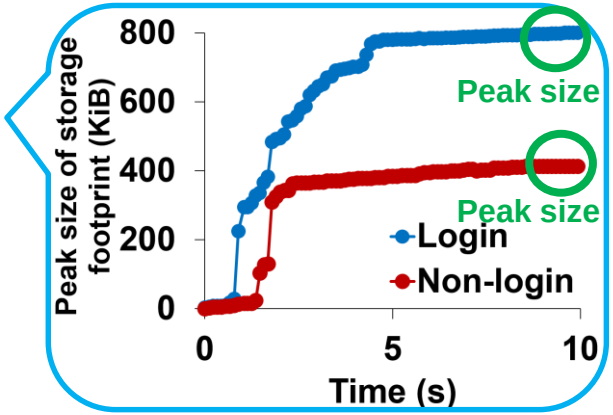
<Attacks on Linux (Alexa Top 200 sites)>

Outline

1. Cross-tab activity inference
2. **Browser status inference**
   1) Browser history
   2) **Login status**
   3) Friendship and group membership
3. Countermeasure

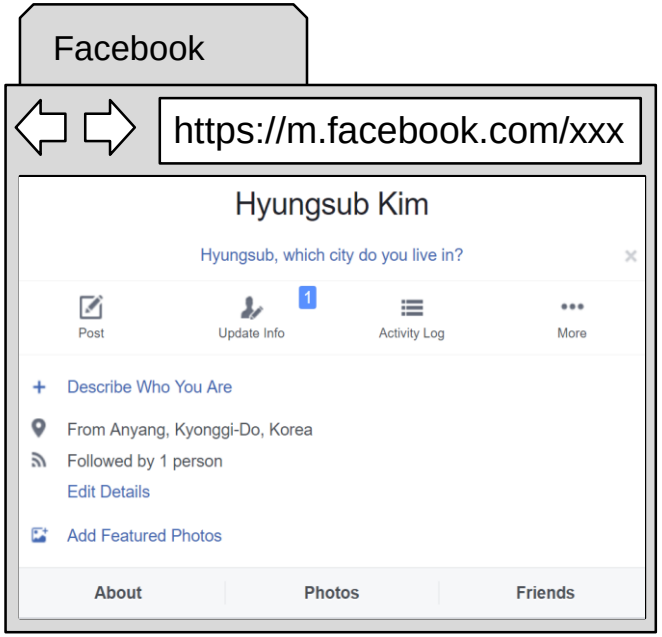
Login status identification

- Login pages return different contents according to login status



Exploit this difference!

<Non-login>



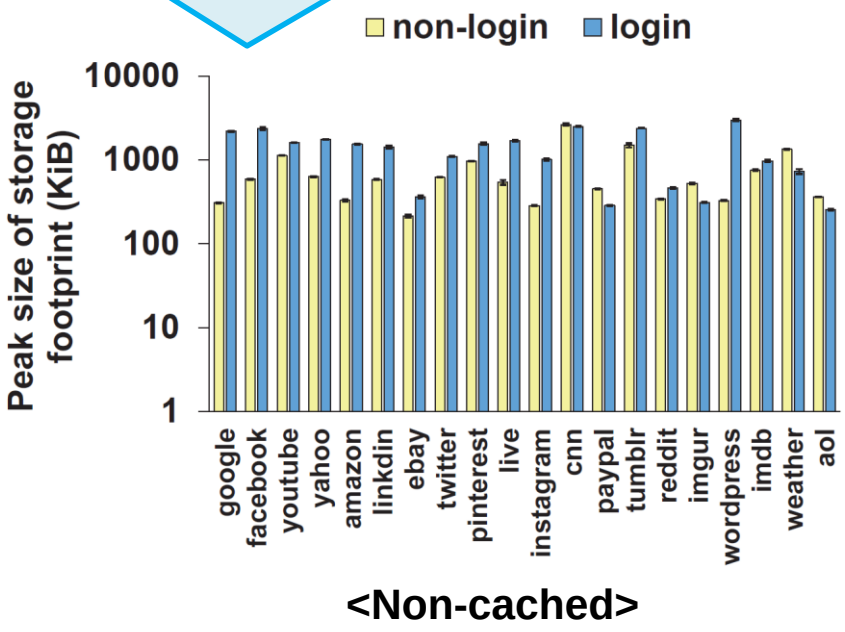
<Login>

Login status identification

- Attack the login pages of Alexa Top 20 sites

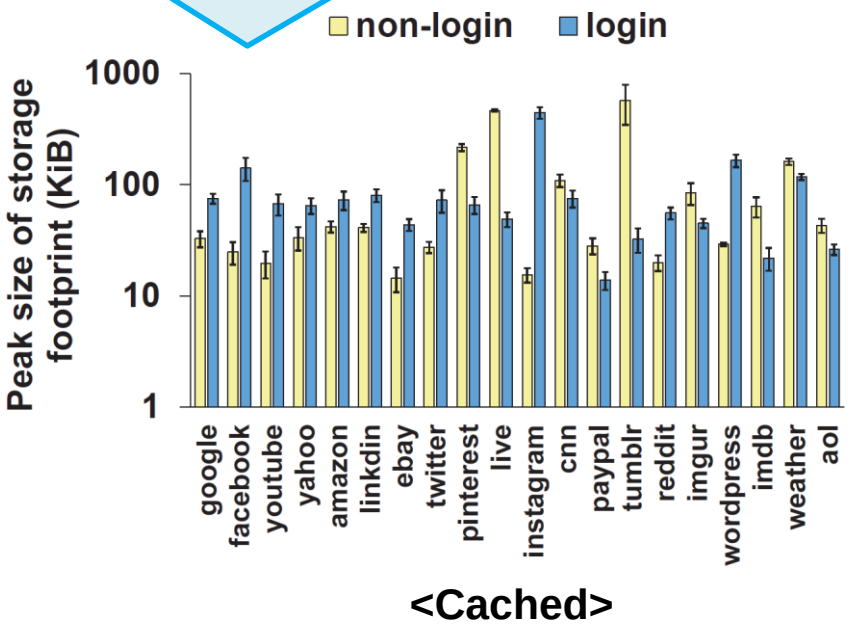
Clear differences in storage footprints

Visit non-cached web pages without and with login information



<Non-cached>

Visit cached web pages without and with login information



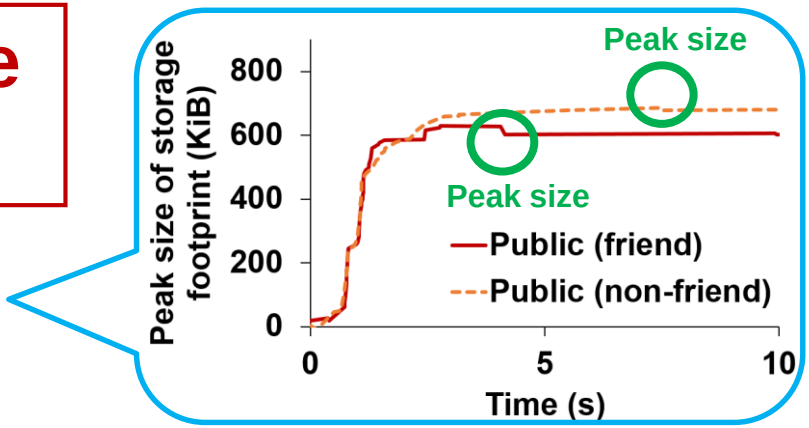
<Cached>

Outline

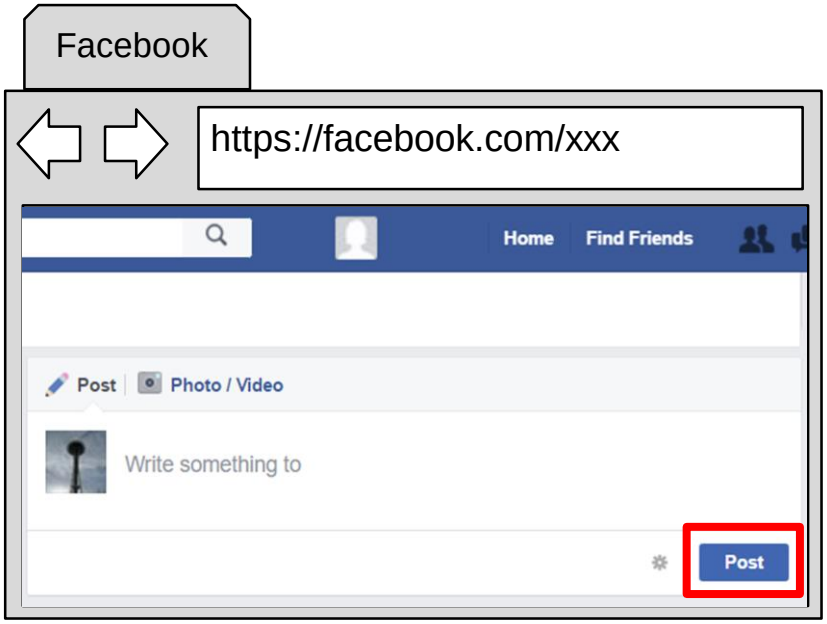
1. Cross-tab activity inference
2. **Browser status inference**
    1) Browser history
    2) Login status
    3) **Friendship and group membership**
3. Countermeasure

Inferring friend relationship

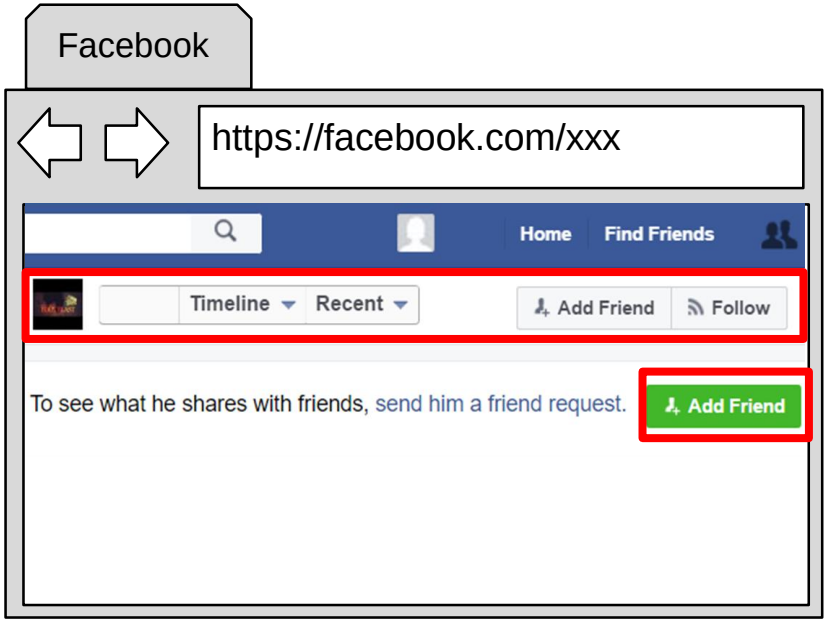
Even public Facebook timeline is personalized



- According to user permission
  - Show different web pages
  - Store different resources in the local storage



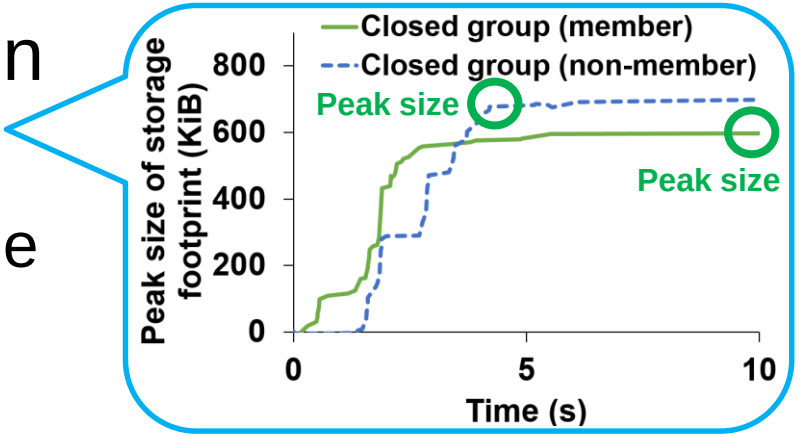
<In case of friends>



<In case of non-friends>

Inferring group relationship

- According to user permission
  - Show different web pages
  - Store different resources in the local storage



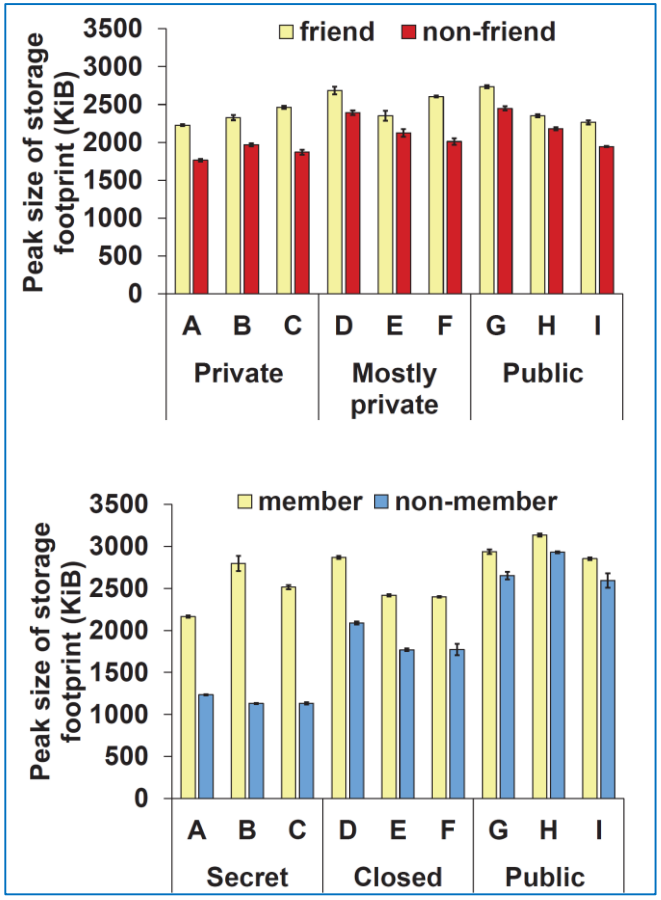
<Joined a closed group>

<Not joined a closed group>

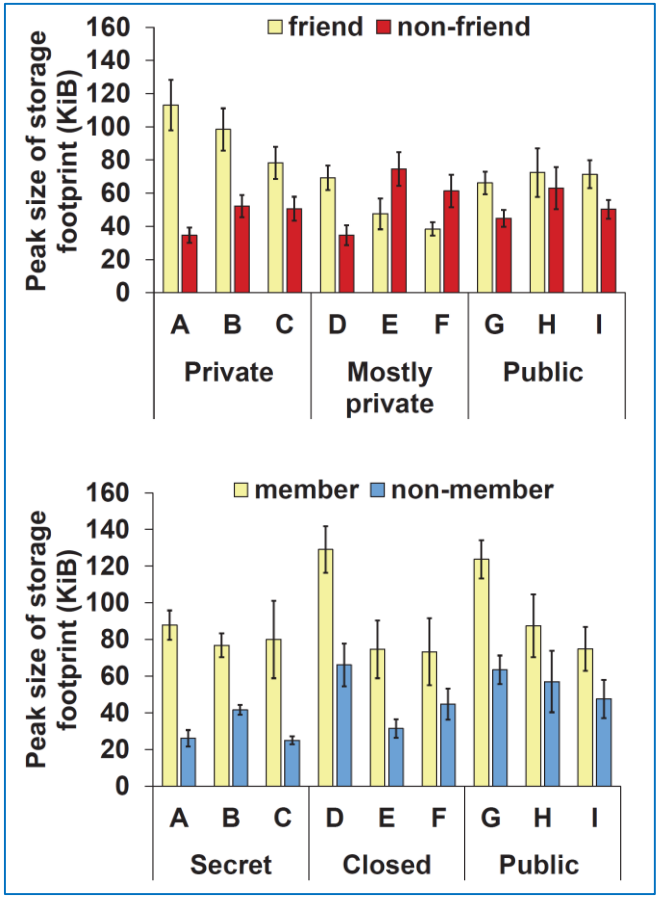
Attack results

Successfully recognize friend and group relationship

Non-cache



Cache



Outline

1. Cross-tab activity inference
2. Browser status inference
   1) Browser history
   2) Login status
   3) Friendship and group membership
3. **Countermeasure**

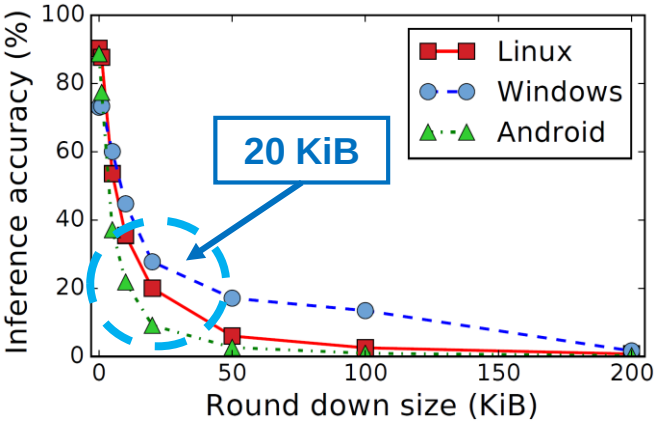
Countermeasure: Round down

- Round quota value down to the nearest multiple of a unit
- Example
  - Temporary storage: 49.99 MiB
  - Unit of round down: 100 KiB

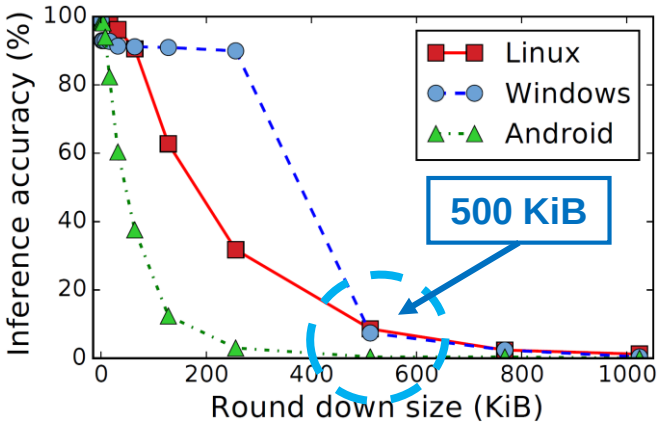
Original quota
(49.99 x 0.2) = 9.998 MiB

Rounded quota
Round(9.998, 100KiB**)**
= 9.9 MiB

Attackers can only get coarse-grained storage footprints.



<Cross-tab activity>



<Browser history stealing>

Conclusion

- **Explored the privacy problem of Quota API**
  - **Attack1**: infer cross-tab activity
    - Identify the very next website a victim will visit
  - **Attack2**: infer browser status
    - Browsing history
    - Login status
    - Friend and group relationship on Facebook
- **Proposed a round down method**
  - Effectively prevent our attacks

Thank you

Are there any questions?

hyungsubkim@postech.ac.kr