

Browser Permission Mechanisms Demystified

Presenter: Kazuki Nomoto*

Author: Kazuki Nomoto*, Takuya Watanabe † , Eitaro
Shioji † , Mitsuaki Akiyama † , and Tatsuya Mori ‡ §

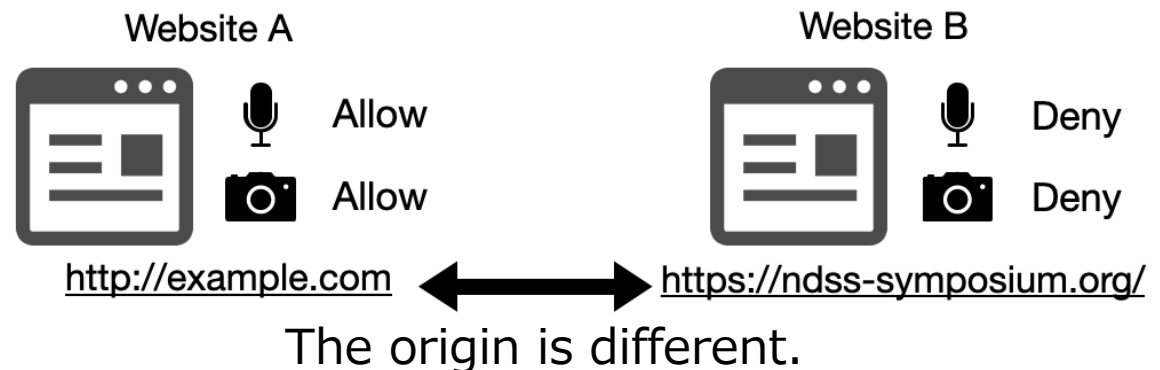
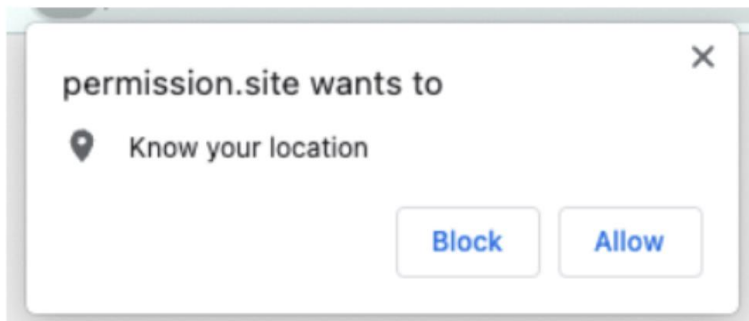
*Waseda University, † NTT Social Informatics Laboratories, ‡ National
Institute of Information and Communications Technology (NICT) ,
§RIKEN Center for Advanced Intelligence Project

Research Overview

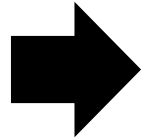
- We developed Permium, a framework that automatically manipulates web browsers and analyzes browser permission mechanisms.
 - Analysis Targets
 - 22 types of web browsers
 - 4 types of permissions
- We found 191 implementation inconsistencies caused by browser/OS differences.
- We propose/evaluate attacks and countermeasures.

Background: Browser Permission Mechanism

- Browser permission mechanisms allow browsers to control the use of cameras, GPS, etc. by websites (origins).
- Separation mechanism of permission
 - By website
 - By permission type
- No standard has been defined.



Research Outline



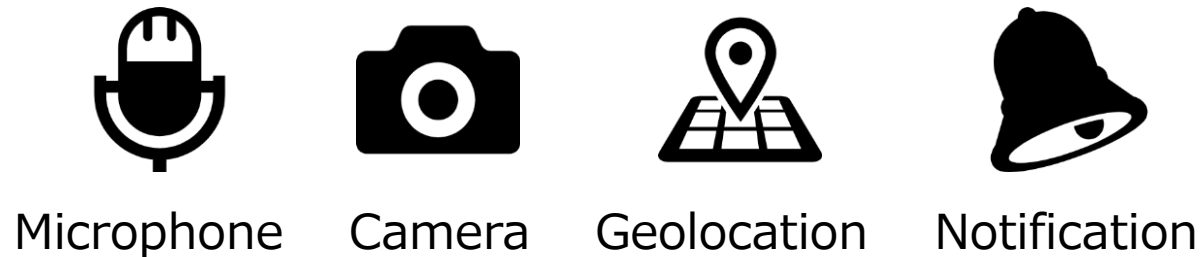
1. Analysis
 - We analyze cross-browser permission implementations.
 - We analyze browser permission implementations vs. user expectations.
- 2. Attack
 - We propose/evaluate new attack methods based on findings from our analysis.
- 3. Countermeasure
 - We propose countermeasures.

Analysis Overview - Targets

- 22 browsers (5 browsers and 5 OS combinations)



- Four types of permissions



Analysis Overview - Methodology

- We developed "Permium", a framework that automatically operates the browser.
- The analysis is divided into two steps.
- Step1. data logging
 - We define analysis scenarios.
 - Permium automatically operates/logs the browsers.
- Step2. data analysis
 - We analyze the logs to reveal the permissions mechanism.

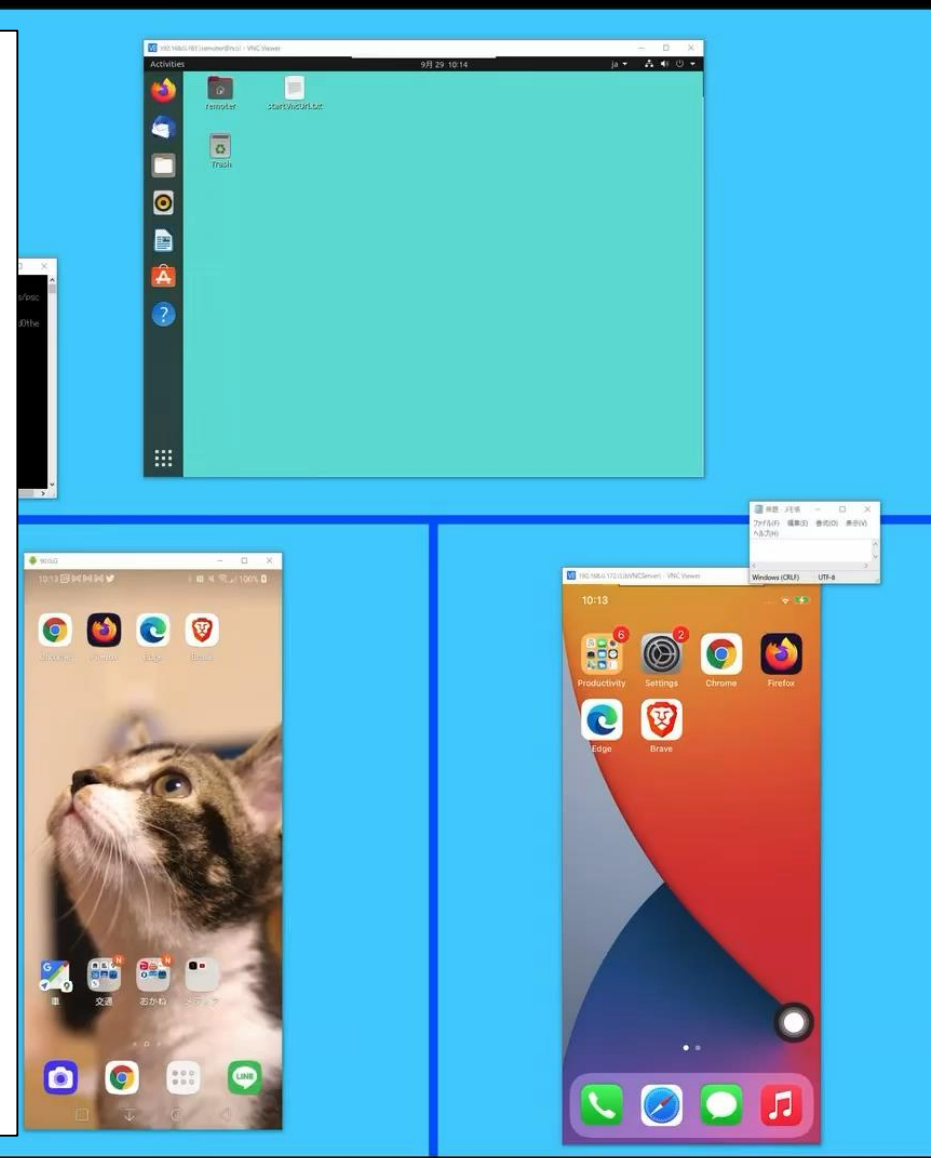
Analysis Scenario

- We defined six analysis scenarios, T1~T6.
 - T1: Is the permission state set by a user (granted or denied) correctly reflected by the browser?
 - T2: Is the permission state set by a user persistent?
 - T3: Is the permission state isolated between the browsing modes?
 - T4: Does clearing browser data and settings erase the permission state?
 - T5: How is the permission state set when the prompt is ignored?
 - T6: Does a permission request from a tab running in the background pop up in front?

PERMIUM DEMO

Technical Challenges of the PERMIUM Framework

- PERMIUM does not use existing browser auto-manipulation frameworks such as Selenium, Puppeteer, and Playwright.
 - Analysts can analyze third-party browsers on iOS, such as Chrome and Firefox.
- PERMIUM provides analysts with the abstracted operating methods that can absorb the browser UI differences.
 - Analysts can work with the 22 different browsers by simply writing a test scenario code.



Analysis Overview - Result

- 191 implementation inconsistencies were found that could lead to user privacy risks
 - Different implementations for different OS/browsers
 - Inconsistencies between browser features
- All browsers have implementation inconsistencies.



Chrome



Firefox



Edge



Brave



Safari

x



Windows



Linux



macOS



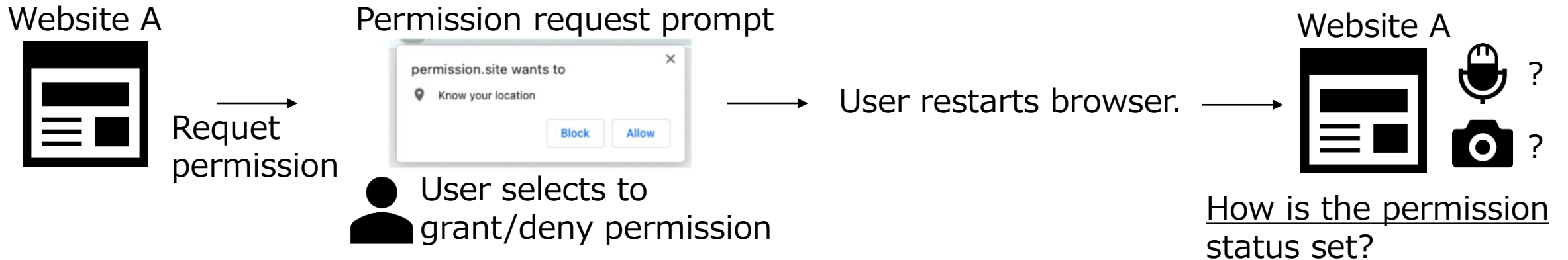
Android



iOS

T2: Is The Permission State Set By A User Persistent?

We investigate whether the permission states set by the user persist after the web browser is closed.



The Result Of Persistence Of The Permission In Normal Browsing Modes.

Permission state	Chrome					Firefox					Edge					Brave					Safari		
	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i	
Granted	●	●	●	●	G	N	N	N		G	●	●	●	●	G						G	N	G
Denied	●	●	●	●	G	N	N	N		G	●	●	●	●	G						G	N	G

● : Permission state persists for all supported resources
 N/G: Notification/Geolocation permission state persists.

W : Windows, L : Linux, M : macOS,
 A : Android, i : iOS

- In normal browsing modes in Chrome and Edge, all four analyzed permissions persisted, except for iOS.
- The conditions under which permission states persisted and the permission types that persisted vary depending on the browsers.

The Result Of Persistence Of The Permission In Private Browsing Modes.

Permission state	Chrome					Firefox					Edge					Brave					Safari	
	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
Granted					G				N	G					G					G	N	
Denied					G					G					G					G	N	

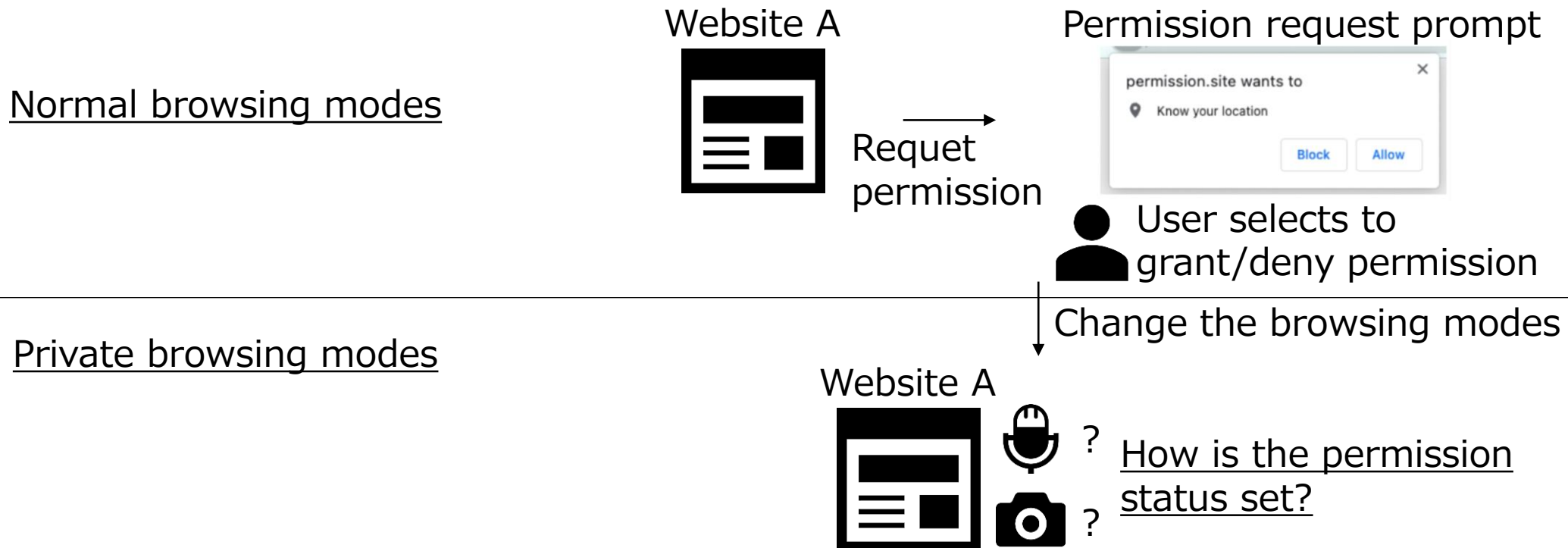
N/G: Notification/Geolocation permission state persists.

W : Windows, L : Linux, M : macOS,
A : Android, i :iOS

- We may have expected that the permission state would not be persistent.
- iOS web browser, except Safari
 - Geolocation permission persisted when a user grants the permission at least twice.

T3 : Is The Permission State Isolated Between The Browsing Modes?

- We investigate whether the permission state set by a user is isolated or shared between the normal and private browsing modes.



The Result Of The Permission State Isolated Between The Browsing Modes (From Normal To Private).

Permission state	Chrome					Firefox					Edge					Brave					Safari	
	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
Granted					G				N	G					G					G	N	
Denied	●	●	●	●	G				N	G	●	●	●	●	G	●	●	●	●	G	N	

● : Permission state of all resources is shared

W : Windows, L : Linux, M : macOS,

G/N : Geolocation/Notification permission state is shared.

A : Android, i : iOS

- The permission state was not always isolated between browsing modes in many browsers.
 - iOS WebKit browsers except Safari share the Geolocation permission state
 - Safari on macOS shares the Notification permission state
 - In Chrome, Edge, and Brave, the denied permission state set in normal browsing modes was reflected in private browsing modes.

Summary of Analysis Results

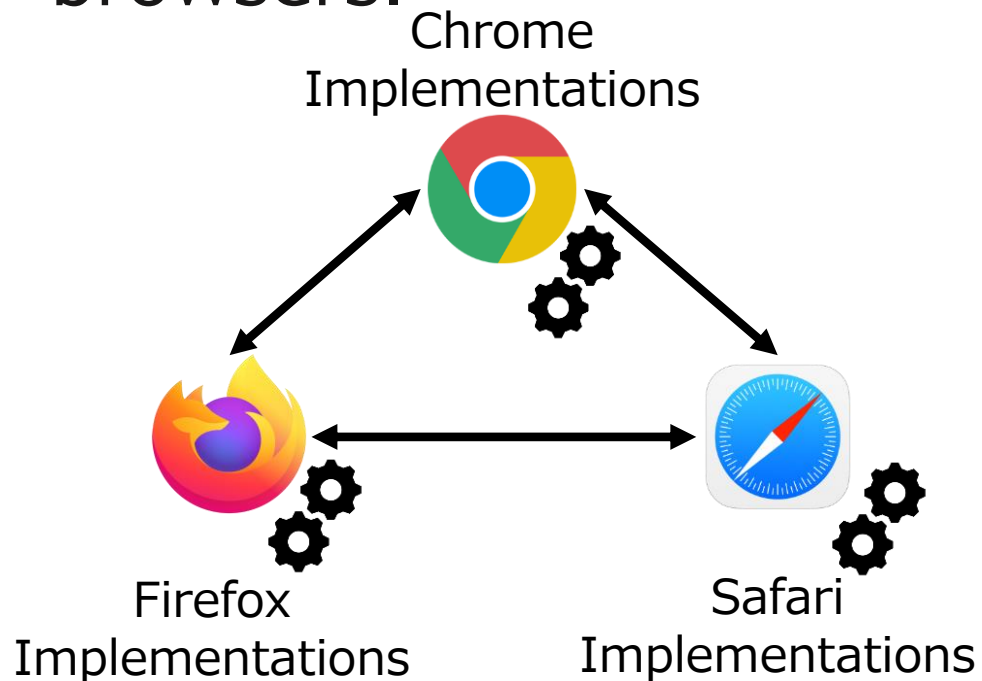
- Implementation of permission mechanisms differs widely from browser to browser and OS to OS.



Analysis

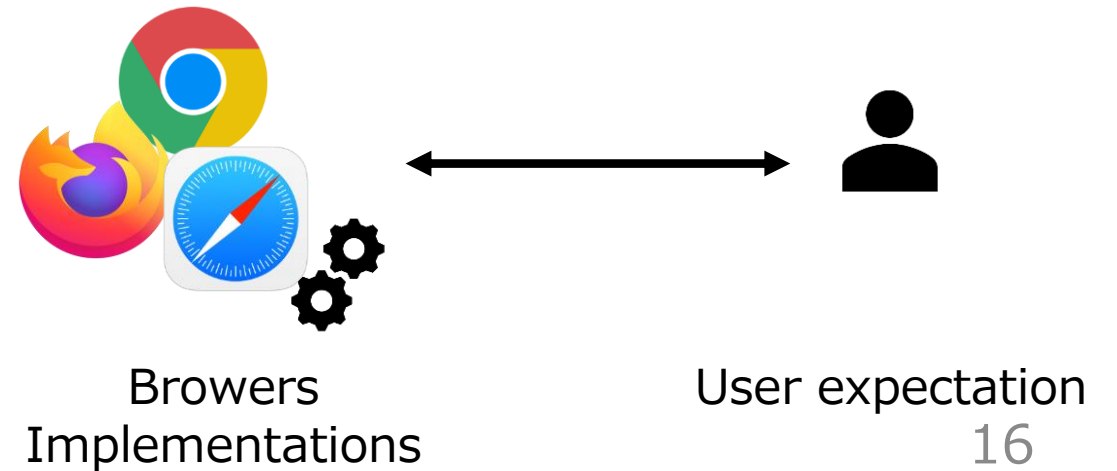
1. Browser implementations

Identify implementation inconsistencies between browsers.



2. User expectations

Identify inconsistencies between browser implementation and user expectations.



The Summary Of The User Study

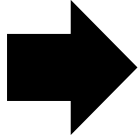
- There is gap between user expectations and browser implementations.
 - User expectations
 - 80% of users expect that the permission state is not persistent in private browsing mode.
 - 70~80% of users expect that the permission state is not inherited between browsing modes.
 - Browser implementation
 - The permission state is persistent/inherited in many browsers.
- The gaps create the risk that user privacy could be violated.

Research Outline

- 1. Analysis
 - We analyze cross-browser permission implementations.
 - We analyze browser permission implementations vs. user expectations.
- ➔ 2. Attack
 - We propose/evaluate new attack methods based on findings from our analysis.
- 3. Countermeasure
 - We propose countermeasures.

Attack Overview

- This study proposed/evaluated two attacks.

- 
1. Permission-based User Tracking Attack
 - An attacker tracks users by checking the permission state set for the attacker's websites and stored in the target's browser.
 - 2. Permission-based Phishing Attack
 - An attacker compels the target to mistakenly grant access to a resource by presenting a fake permission request.

Threat Model

- Purpose
 - Attackers want to identify users visiting a website.
- Condition
 - The attacker has some websites.

- Landing website



landing-website.example

- Tracking websites



1.example



2.example



3.example



4.example



5.example



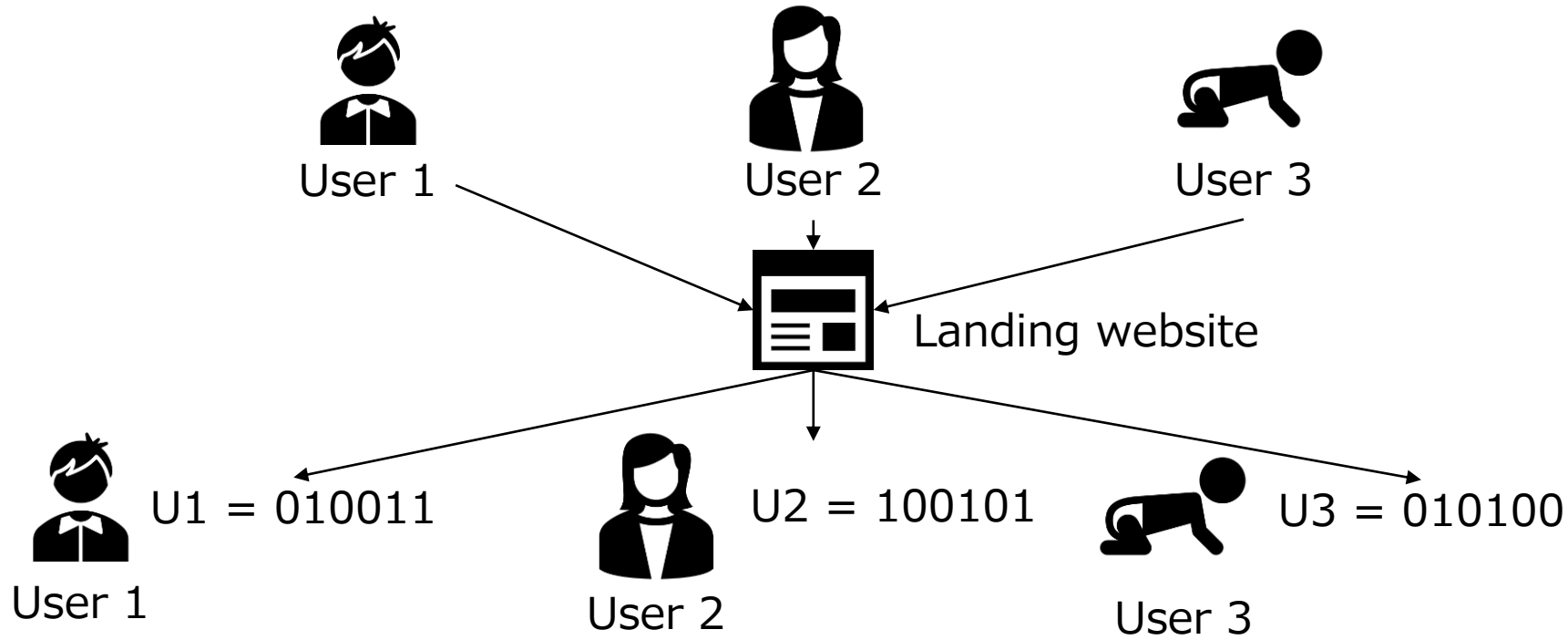
6.example

Attack Procedures

- The procedure for this attack comprises the following three steps:
 - Step 1: the user ID assignment
 - Step 2: encoding
 - Step 3: decoding
- This attack is not a normal fingerprinting attack.
- This attack actively sets the permission states corresponding to user IDs for each website.
- Attackers can identify users deterministically.

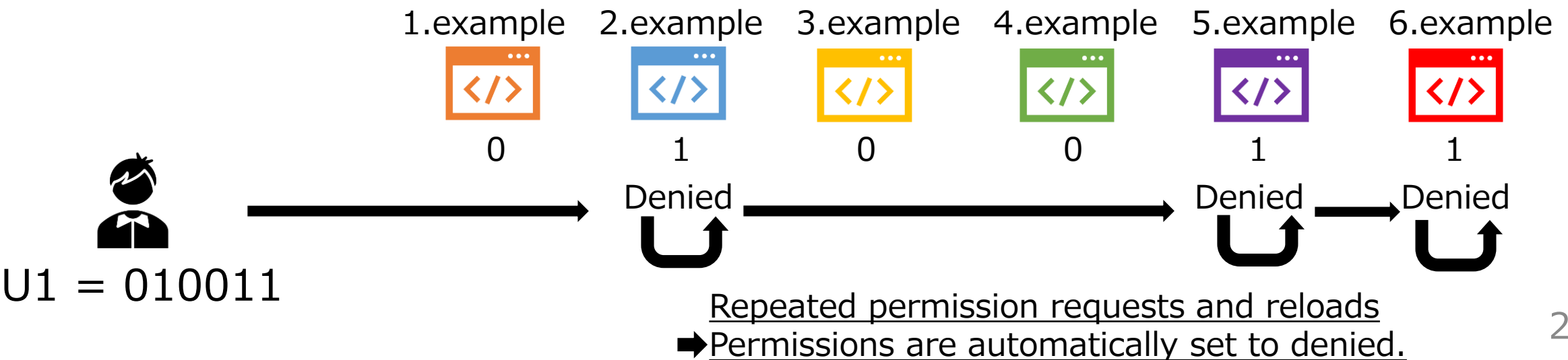
Step 1. The User ID Assignment

- An attacker assigns a unique ID to each target.
 - The user ID is a binary number of 0 and 1.
- The 0 and 1 of the user ID correspond to the permission statuses "not denied" and "denied".



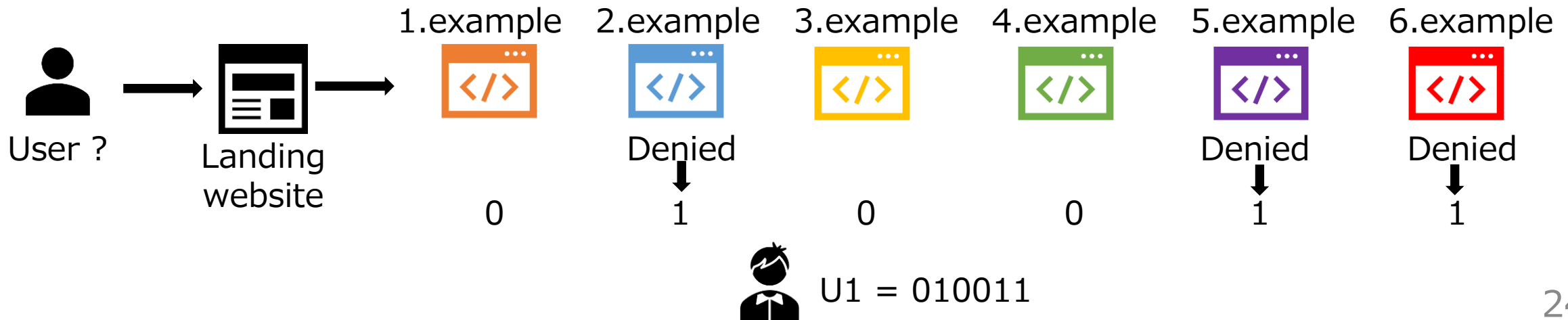
Step 2: Encoding

- An attacker manipulates the permission states of the tracking websites, following the ID generated in Step 1.
- The attacker makes the permission state for the websites to be “denied” by repeatedly requesting permission and reloading the tracking website.



Step 3: Decoding

- An attacker makes the browser access the tracking websites and checks the permission state on each tracking website when a user revisits the landing website.
- The attacker can decode the binary sequence corresponding to the permission states and obtain the ID of the user; hence, tracking the user is completed.



Attack Evaluation

- Targets

- All browsers are targets of the attack, except Firefox.



Chrome



Edge



Brave



Safari

- Tracking beyond browsing mode is possible.

- Required time

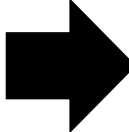
- We measured the time spent tracking 4.3 billion users.

- Result

- 7.0 [s] : Step 2. encoding (+ Step 1. the user ID assignment)
- 2.6 [s] : Step 3. decoding

- Steps 1 and 2 only need to be done once for each user, while Step 3 is done multiple times each time a user is identified. Therefore, this attack is highly effective because Step 3 can be done in less than 3 seconds.

Research Outline

- 1. Analysis
 - We analyze cross-browser permission implementations.
 - We analyze browser permission implementations vs. user expectations.
- 2. Attack
 - We propose/evaluate new attack methods based on findings from our analysis.
-  3. Countermeasure
 - We propose countermeasures.

Countermeasure

- Short/long-term perspective countermeasures are needed.
- Short-term perspective
 - Fix the implementation that this study found.
- Long-term perspective
 - Standardization / sharing of best practices.

Short-term Perspective

- We reported the issues found to the browser vendor.
 - The Brave and Firefox browsers have already fixed some of the implementations as a result of our report.
 - Other browsers are still under consideration.

Fixed



CVE-2023-23600

Firefox



Github #14765

Brave

Under consideration



Chrome



Edge

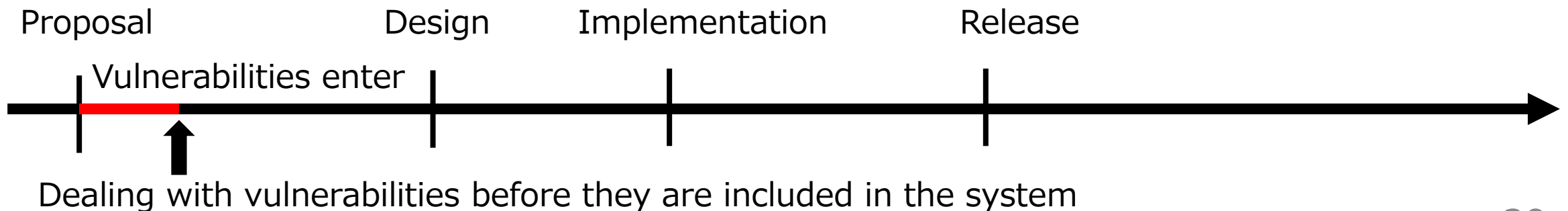


Safari

*Edge : Depends on chromium

Long-term Perspective

- In this study, the vulnerability was fixed after it had been introduced into the software and could be exploited.
- This cannot be enough to fundamentally prevent future vulnerabilities from being introduced into the software.
- It is important to identify design issues, standardize, and share best practices among browser vendors.



Long-term Perspective

- We publish a timeline of vulnerability reports on our website.
- We make presentations at conferences.
- We publish the alert information (JVNTA) with JPCERT/CC.
 - <https://jvn.jp/en/ta/JVNTA96606604/>
 - JPCERT/CC is the Japanese Emergency Response Team / Coordination Center.

Summary

- We developed Permium, a framework that automatically manipulates web browsers and analyzes browser permission mechanisms.
 - We analyzed the permission mechanisms of 22 different web browsers.
- We found 191 implementation inconsistencies differing across browsers/OS.
- We propose/evaluate attacks that exploit implementation inconsistencies.

Measurement Result T1

- T1: Is the permission state set by a user (granted or denied) correctly reflected by the browser?

TABLE II. REFLECTION OF THE PERMISSION STATE SET BY USER (SCENARIO T_1).

Browsing Mode	Chrome					Firefox					Edge					Brave					Safari	
	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
Normal	○	○	○	○	N	○	○	○	○	N	○	○	○	○	N	○	○	○	○	N	○	N
Private	N	N	N	N	N	†	†	†	○	N	N	N	N	N	N	N	N	N	N	N	○	N

○: Permission state set by a user is correctly reflected for all permission resources (Microphone, Camera, Geolocation, and Notification)

N : Notification permission is unsupported. Other permissions are correctly supported.

†: Notification permission is supported, but the state “denied” is not correctly reflected. Other permissions are correctly supported.

W: Windows, L: Linux, M: macOS, A: Android, i: iOS

Measurement Result T3

- T3: Is the permission state isolated between the browsing modes?

TABLE IV. SHARING OF PERMISSION STATE BETWEEN DIFFERENT BROWSING MODES (SCENARIO T_3).

Order	Permission state	Tab Status	Chrome					Firefox					Edge					Brave					Safari	
			W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
Normal → Private	Granted	Open	○	○	○	○	G	○	○	○	N	G	○	○	○	○	G	○	○	○	○	G	N	○
	Granted	Closed	○	○	○	○	G	○	○	○	N	G	○	○	○	○	G	○	○	○	○	G	N	○
Normal → Private	Denied	Open	●	●	●	●	G	○	○	○	N	G	●	●	●	●	G	●	●	●	●	G	N	○
	Denied	Closed	●	●	●	●	G	○	○	○	N	G	●	●	●	●	G	○	○	○	○	G	N	○
Private → Normal	Granted	Open	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	N	○
	Granted	Closed	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	N	○
Private → Normal	Denied	Open	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	N	○
	Denied	Closed	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	○	○	○	○	G	N	○

● : Permission state of all resources is shared, ○ : Permission state of all resources is not shared.

N : Notification permission state is shared. G : Geolocation permission state is shared. W: Windows, L: Linux, M: macOS, A: Android, i: iOS

Measurement Result T4

- T4: Does clearing browser data and settings erase the permission state?

TABLE V. PERMISSION STATE AFTER DELETING BROWSER DATA (SCENARIO T_4).

Mode	Chrome					Firefox					Edge					Brave					Safari	
	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
Normal	○	○	○	○	G	NMC ₁	NMC ₁	NMC ₁	○	G	○	○	○	○	G	○	○	○	○	G	○	○
Private	●	●	●	●	G	NMC ₂	NMC ₂	NMC ₂	○	G	●	●	●	●	G	●	●	●	●	G	○	○

○ : Permission states for all resources are erased. ● : Permission states for all resources are retained.

G : Permission state (granted/denied) for Geolocation is retained.

NMC₁ : Permission states (granted/denied) for Notification and permission states (granted) for Microphone and Camera are retained.

NMC₂ : Permission states (granted) for Notification, Microphone, and Camera are retained.

W: Windows, L: Linux, M: macOS, A: Android, i: iOS

Measurement Result T5

- T5: How is the permission state set when the prompt is ignored?

TABLE VI. RESULTS OF AUTOMATICALLY SETTING PERMISSION STATE TO DENIED BY IGNORING THE PROMPT MULTIPLE TIMES (SCENARIO T_5).

Mode	Tab Status	Chrome					Firefox					Edge					Brave					Safari	
		W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
Normal	Foreground	●	●	●	●	–	○	○	○	○	–	●	●	●	●	–	●	●	●	●	–	●	–
	Background	MCN	MCN	MCN	N	○	○	○	○	○	◐	MCN	MCN	MCN	N	○	MCN	MCN	MCN	N	◐	N†	○
Private	Foreground	●	●	●	●	–	○	○	○	○	–	●	●	●	●	–	●	●	●	●	–	●	–
	Background	MC	MC	MC	○	○	○	○	○	○	◐	MC	MC	MC	○	○	MC	MC	MC	○	◐	N†	○

● : Permission state is automatically set to denied for all resources, ○ : Permission state is not automatically set to denied for all resources.

M/C/N : Microphone/Camera/Notification permission state is automatically set to denied.

◐ : No results (Permission request dialog overlay display occurs and dialog cannot be ignored.), – : Analysis inapplicable.

† : Permission request dialog is overlaid, and state is automatically set to denied. W: Windows, L: Linux, M: macOS, A: Android, i: iOS

Measurement Result T6

- T6: Does a permission request from a tab running in the background pop up in front?

TABLE VII. DISPLAY OF A PERMISSION REQUEST DIALOG FROM A BACKGROUND TAB (SCENARIO T_6).

Chrome					Firefox					Edge					Brave					Safari	
W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
○	○	○	○	●§	○	○	○	○	●†	○	○	○	○	○	○	○	○	○	●†	●‡	○

○ : The dialog is not displayed on top of the foreground tab. ● : The dialog is displayed on top of the foreground tab.

§ For all permissions, both “granted” and “denied” are reflected in the permission state.

† For Microphone and Camera permissions, both “granted” and “denied” are reflected in the permission state. For Geolocation permission, only “denied” is reflected in the permission state. ‡ Notification permission will have “denied” reflected in the permission state.

W: Windows, L: Linux, M: macOS, A: Android, i: iOS

Attack Feasibility

TABLE VIII. THE TARGET OF THE PERMISSION-BASED USER TRACKING ATTACK.

	Chrome					Firefox					Edge					Brave					Safari	
	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
① Normal attack	●	●	●	●	○	○	○	○	○	○	●	●	●	●	○	●	●	●	●	○	●	○
② Packing multiple permissions	●	●	●	○	○	○	○	○	○	○	●	●	●	○	○	●	●	●	○	○	○	○
③ Background attack	●	●	●	●	○	○	○	○	○	○	●	●	●	●	○	●	●	●	●	○	●	○
④ Iframe attack	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○

● : Attackable, ○ : Not attackable, ◐ : Attackable for Encoding, W: Windows, L: Linux, M: macOS, A: Android, i: iOS

TABLE IX. THE TARGET OF THE PERMISSION-BASED USER TRACKING ATTACK ACROSS BROWSING MODES.

Order	Chrome					Firefox					Edge					Brave					Safari	
	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	W	L	M	A	i	M	i
Normal → Private	●	●	●	●	○	○	○	○	○	○	●	●	●	●	○	●	●	●	●	○	●	○
Private → Normal	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○

● : Attackable, ○ : Not attackable, W: Windows, L: Linux, M: macOS, A: Android, i: iOS

Attack Feasibility

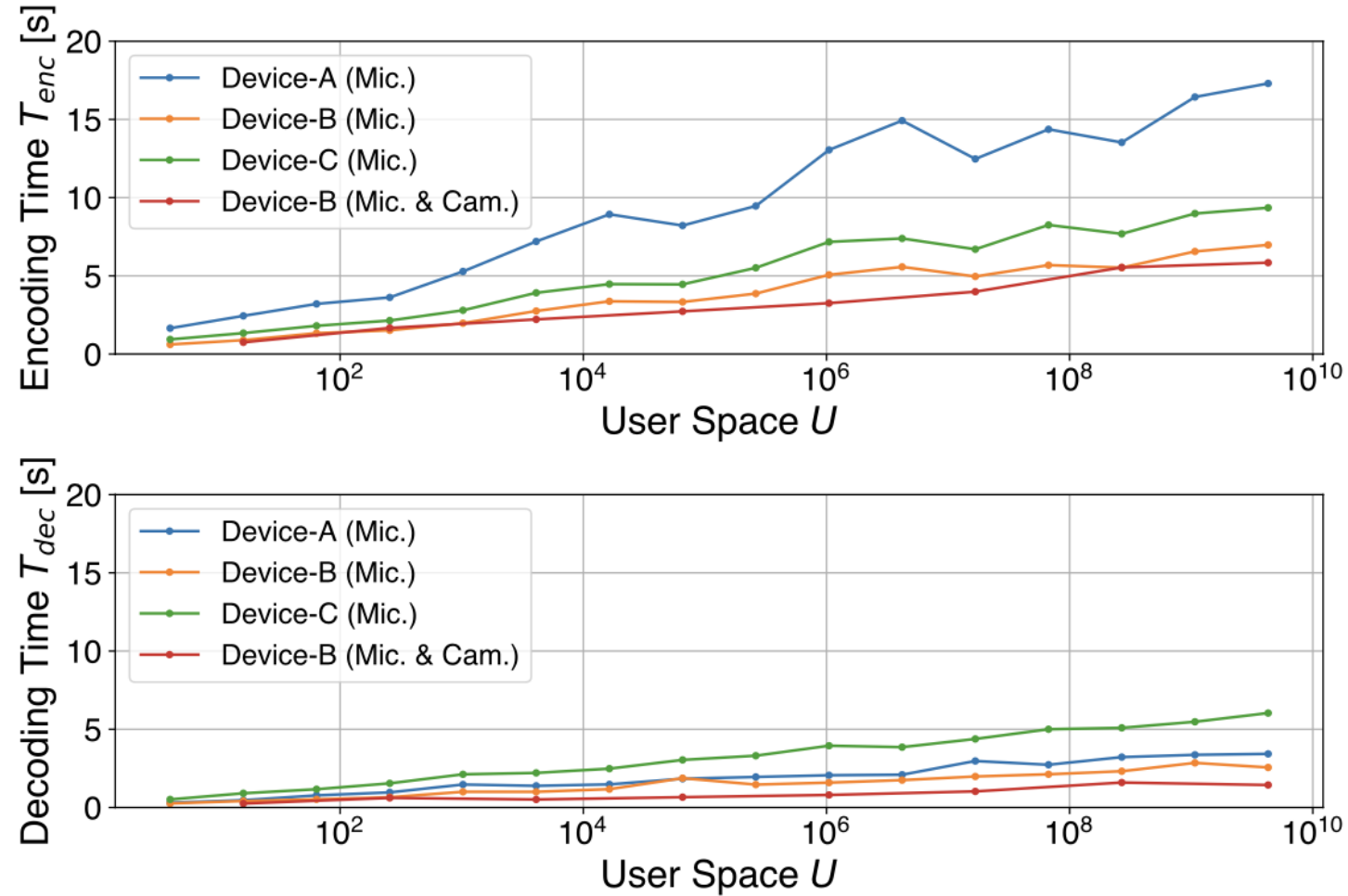


Fig. 3. U vs. required time. Top: encoding and Bottom: decoding.

Result of User Study U1

- User Expectations on the Permission Mechanisms.

Table 10 Results of U1: User expectations on the camera permission status

User expectations (recording images)	Denied		Granted	
The website can record camera footage.	3	(1.3%)	183	(78.9%)
The website cannot record camera footage.	216	(93.1%)	10	(4.3%)
Other	13	(5.6%)	39	(16.8%)
User expectations (sending iamges)	Denied		Granted	
The website can send camera footage to an external source.	4	(1.7%)	140	(60.3%)
The website cannot send camera footage to an external source.	203	(87.5%)	19	(8.2%)
Other	25	(10.8%)	73	(31.5%)

Result of User Study U2

- User Expectations Regarding the Persistence of the Permission State.

Table 11 Results of U2: User expectations on the persistence of permission status

Browsing Modes	Normal				Private							
Group	A				A				B			
Permission status	Denied		Granted		Denied		Granted		Denied		Granted	
Persistent	49	(21.1%)	107	(46.1%)	44	(19.0%)	32	(13.8%)	18	(14.9%)	12	(9.9%)
Not persistent	150	(64.7%)	88	(37.9%)	146	(62.9%)	141	(60.8%)	92	(76.0%)	91	(75.2%)
I don't know	33	(14.2%)	35	(15.1%)	41	(17.7%)	7	(3.0%)	11	(9.1%)	14	(11.6%)
Other	0	(0%)	2	(0.9%)	1	(0.4%)	52	(22.4%)	0	(0%)	4	(3.3%)

Group A : All participants, Group B : Participants familiar with Private Browsing Modes

Result of User Study U3

- User Expectations Regarding Isolation of the Permission State Across Browsing Modes.

Table 12 Results of U3: User expectations on the isolation of the permission status across browsing modes

Browsing Modes	From normal to private						From private to normal					
Group	A			B			A			B		
Permission status	Denied		Granted	Denied		Granted	Denied		Granted	Denied		Granted
Shared	48	(20.7%)	38 (16.4%)	21 (17.4%)	15 (12.4%)	34 (14.7%)	46 (19.8%)	15 (12.4%)	19 (15.7%)	97 (80.2%)	85 (70.2%)	
Not shared	152	(65.5%)	147 (63.4%)	87 (71.9%)	88 (72.7%)	171 (73.7%)	145 (62.5%)	9	(7.4%)	15 (12.4%)	2 (1.7%)	
I don't know	32	(13.8%)	38 (16.4%)	13 (10.7%)	13 (10.7%)	27 (11.6%)	38 (16.4%)	0	(0%)	3 (1.3%)	0 (0%)	
Other	0	(0%)	9 (3.9%)	0 (0%)	5 (4.1%)	0 (0%)	3 (1.3%)	0	(0%)	0 (0%)	2 (1.7%)	

Group A : All participants, Group B : Participants familiar with private browsing modes

Result of User Study U4

- User Expectations for Browsing Data Deletion Mechanisms.

Table 13 Results of U4: User expectations on the browser data deletion mechanism

Browsing Modes	Normal						Private					
Group	A			B			A			C		
Permission status	Denied		Granted	Denied		Granted	Denied		Granted	Denied		Granted
Erased	174	(75.0%)	161 (69.4%)	143 (80.8%)	135 (76.3%)	159 (68.5%)	161 (69.4%)	88 (83.0%)	84 (79.2%)			
Not erased	35 (15.1%)	45 (19.4%)	23 (13.0%)	28 (15.8%)	37 (15.9%)	29 (12.5%)	11 (10.4%)	10 (9.4%)				
I don't know	22 (9.5%)	25 (10.8%)	11 (6.2%)	13 (7.3%)	35 (15.1%)	37 (15.9%)	7 (6.6%)	10 (9.4%)				
Other	1 (0.4%)	1 (0.4%)	0 (0%)	1 (0.6%)	1 (0.4%)	5 (2.2%)	0 (0%)	2 (1.9%)				

Group A : All participants, Group B : Participants familiar with private browsing modes

Result of User Study U5

- User Expectations of Browser Behavior when Permission Requests Are Ignored.

Table 14 Results of **U5**: User expectations of browser behavior when permission requests are ignored

Expected permission status	<i>m</i> = 1		<i>m</i> = 2		<i>m</i> = 3	
The website can use/get the camera footage	12	(5.2%)	9	(3.9%)	9	(3.9%)
The website cannot use/get the camera footage	177	(76.3%)	176	(75.9%)	180	(77.6%)
I don't know	43	(18.5%)	47	(20.3%)	43	(18.5%)

Result of User Study U5

- User Expectations of Browser Behavior when Permission Requests Are Ignored.

Table 15 Results of U5: User opinions of browser behavior when permission requests are ignored

Question	Options	Count	(%)
How do user think about the repetitive display of permission request prompts?	Appropriate	61	(26.3%)
	Somewhat appropriate	98	(42.2%)
	Not really appropriate	39	(16.8%)
	Not appropriate	24	(10.3%)
	I don't know	10	(4.3%)
Do you think the mechanism † is implemented?	Yes	92	(39.7%)
	No	140	(60.3%)
The permission status should automatically become denied when permission is repeatedly requested.	Strongly agree.	54	(23.3%)
	Somewhat agree	82	(35.3%)
	Somewhat disagree	61	(26.3%)
	Strongly disagree.	28	(12.1%)
	I don't know	7	(3.0%)

† : Mechanism that prevents web browsers from repeatedly displaying permission request prompts.

Result of User Study U6

- User Expectations for the Overlaid Prompt Display.

Table 16 Results of U6: User expectations on the overlaid prompt display

Permission status	Denied				Granted			
Website	A		B		A		B	
The website can use/get the camera footage.	10	(4.3%)	10	(4.3%)	53	(22.8%)	190	(81.9%)
The website cannot use/get the camera footage.	188	(81.0%)	200	(86.2%)	129	(55.6%)	20	(8.6%)
I don't know	34	(14.7%)	22	(9.5%)	50	(21.6%)	22	(9.5%)

† : The implementation is that permission request prompts are displayed on different websites.

Table 17 Results of U6: User opinions on the implementation of the overlaid dialog display

Question	Options	Count	(%)
How do users think about the implementation? †	Appropriate	28	(12.1%)
	Somewhat appropriate	27	(11.6%)
	Not really appropriate	43	(18.5%)
	Not appropriate	126	(54.3%)
	I don't know	8	(3.4%)

† : The implementation is that permission request prompts are displayed on different websites.

Q&A: What is the Ideal Implementation?

- It is difficult to determine the ideal implementation definitively.
- It is important that discussions among browser vendors on appropriate permission implementations take place.
- In our opinion, the following points are important for the discussion.
 - Consistency of implementation and user recognition and expectations
 - Consistency of implementation across operating systems within the same browser
 - The minimum implementation necessary to protect user privacy.
 - Independence of permissions in private browsing mode
- It is not necessary for all browsers to have identical implementations.

Q&A: What was the reaction of browser vendors to the vulnerability report?

- The browser vendors responded positively to our report.
- Additionally, some browser vendors responded that fixing browser implementations can be difficult when they rely on upstream sources.
- Implemented Fixes
 - Firefox
 - Notification permissions are shared among browsing modes in Android
 - Brave
 - Denied permissions are shared from normal browsing modes to private browsing mode.

Q&A: What is the difference between a permission-based user tracking attack and general browser fingerprinting?

- A general browser fingerprinting
 - The attacker identifies general browser fingerprinting by obtaining information about the user's browser that cannot be modified by the attacker.
- A permission-based user tracking attack
 - The attacker uses JavaScript to control the user's browser state, thereby deterministically identifying the user uniquely.

The Example of Analysis Scenario

Analysis scenarios when investigating whether the permission status set in normal browsing mode is also reflected in private browsing mode.

1. `platform.startBrowser(browserName)`
2. `browser.goToUrl(targetUrl,platformName,"normal")`
3. `browser.requestPermission(platformName, mode)`
4. `browser.clickAllow(platformName,mode)`
5. `browser.openPrivateBrowsing(platformName, "normal")`
6. `browser.requestPermission(platformName, mode)`
7. `browser.checkPermissionDialogue(platformName,mode)`