On the Security and Privacy Risks of Browser Extensions

Dr.-Ing. Aurore Fass

Tenured Researcher — Inria Centre at Université Côte d’Azur (since Dec 1, 2025)

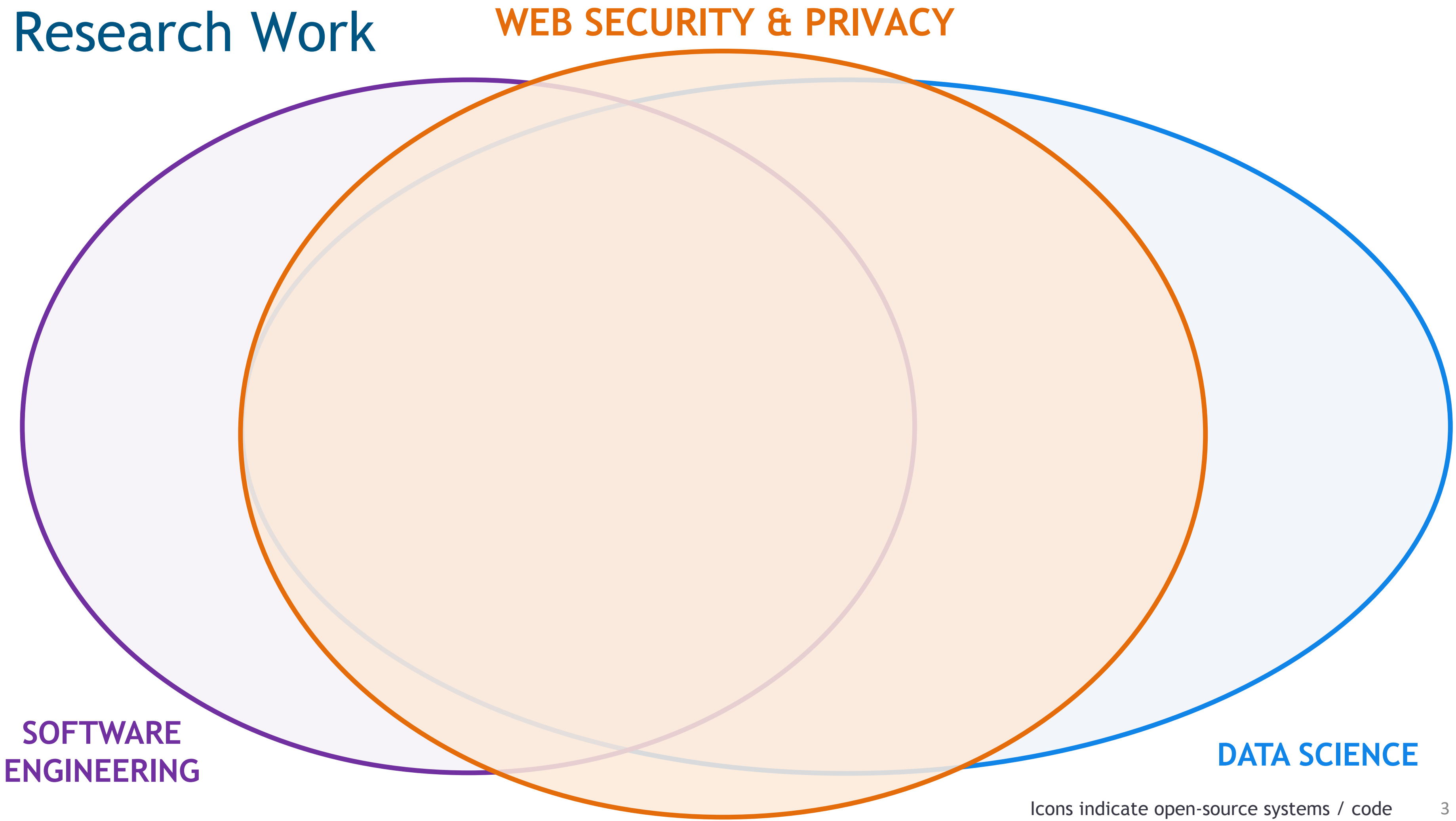
Seminar@SystemX, June 25th 2026

Dr.-Ing. Aurore (/ɔʀɔʀ/) FASS

- 🇫🇷 2017: Graduated from TELECOM Nancy (**FR**)
- 🇩🇪 2017—21: PhD Student + Postdoc at CISPA (**DE**)
- 🇺🇸 2021—23: Visiting Assistant Professor at Stanford (**US**)

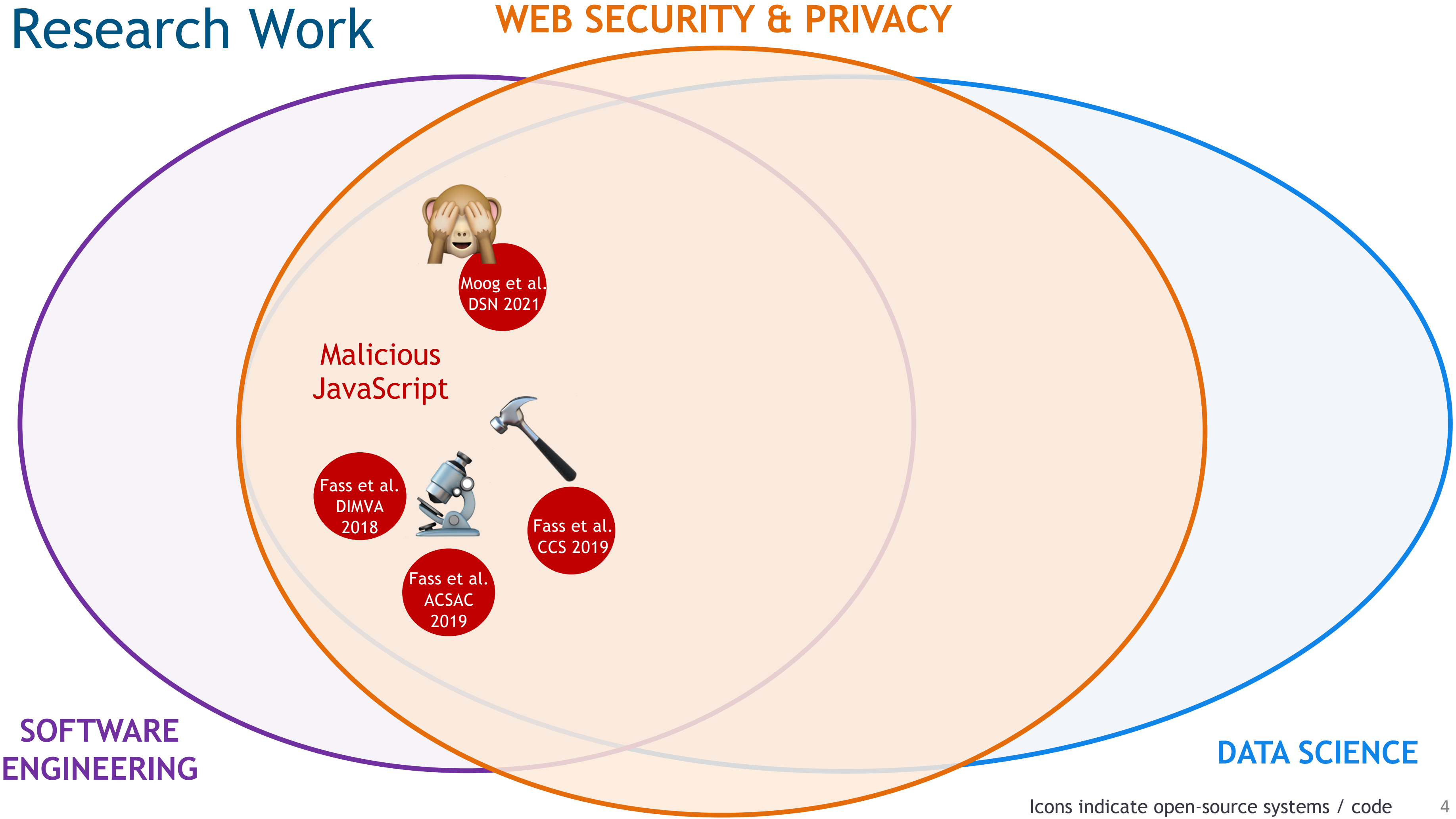
- 🇩🇪 2023—25: Tenure-Track Faculty W2 at CISPA (**DE**)
- 🇫🇷 2025—: Tenured Researcher at Inria (**FR**)

Research Work



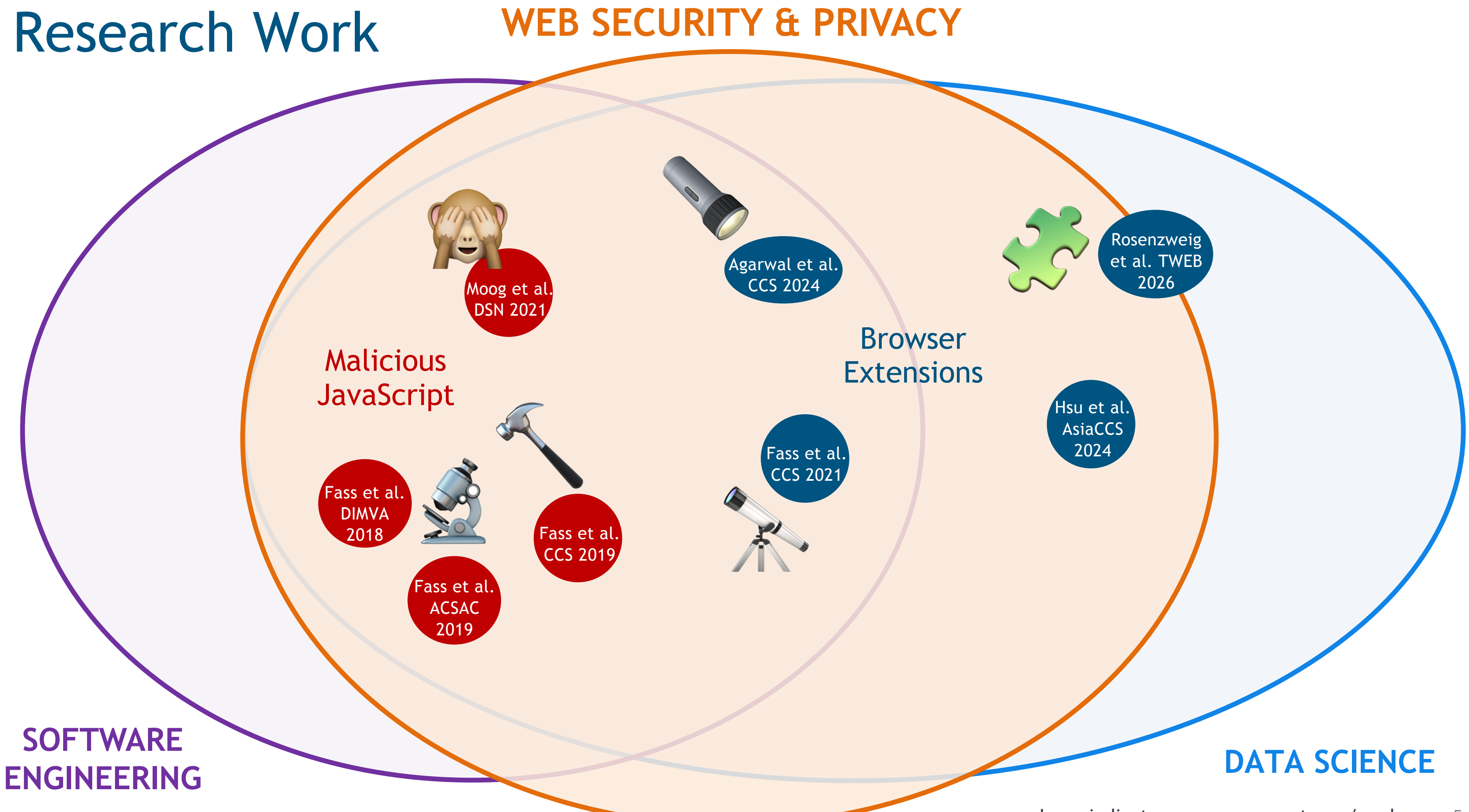
Icons indicate open-source systems / code

Research Work



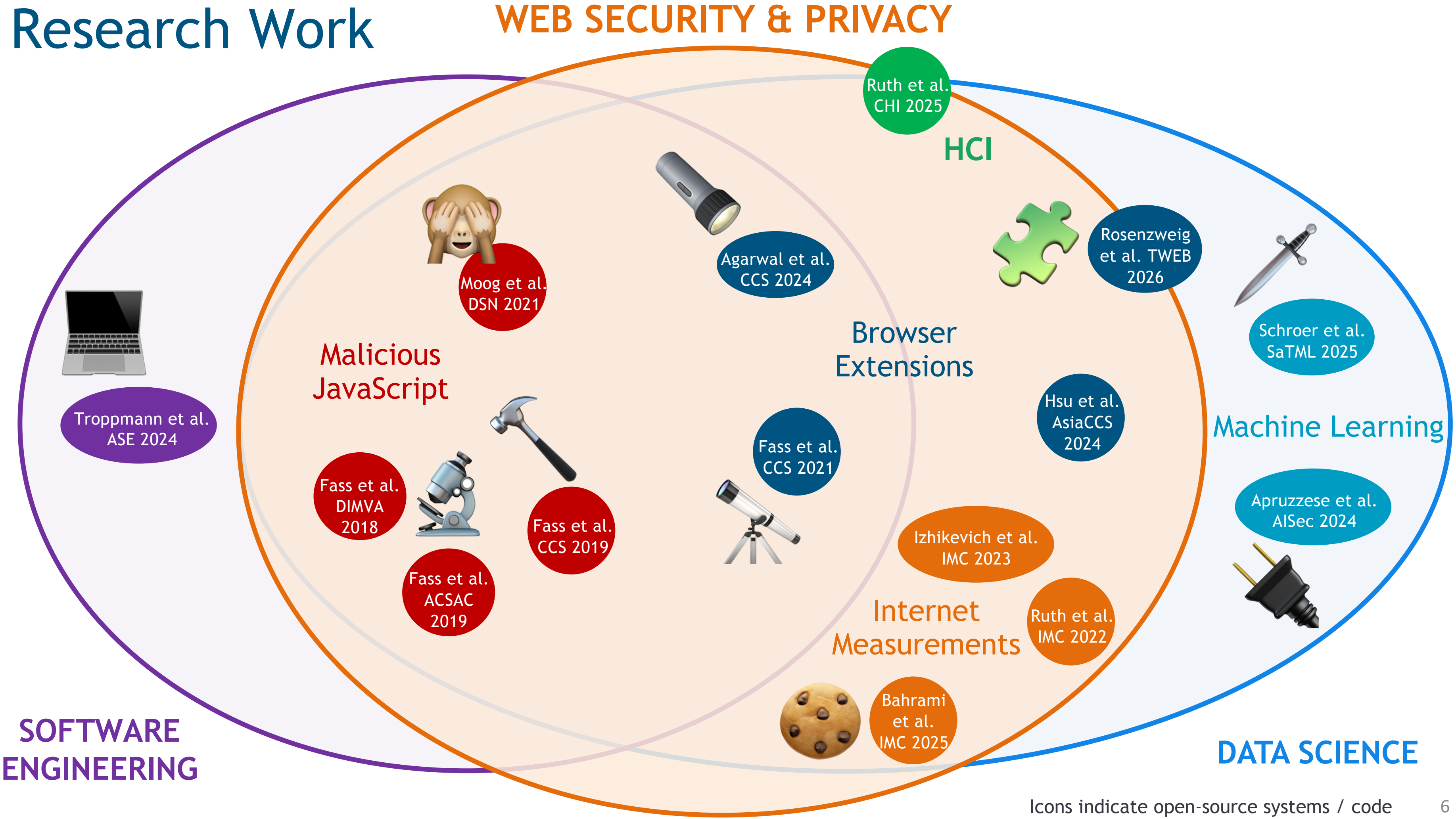
Icons indicate open-source systems / code

Research Work



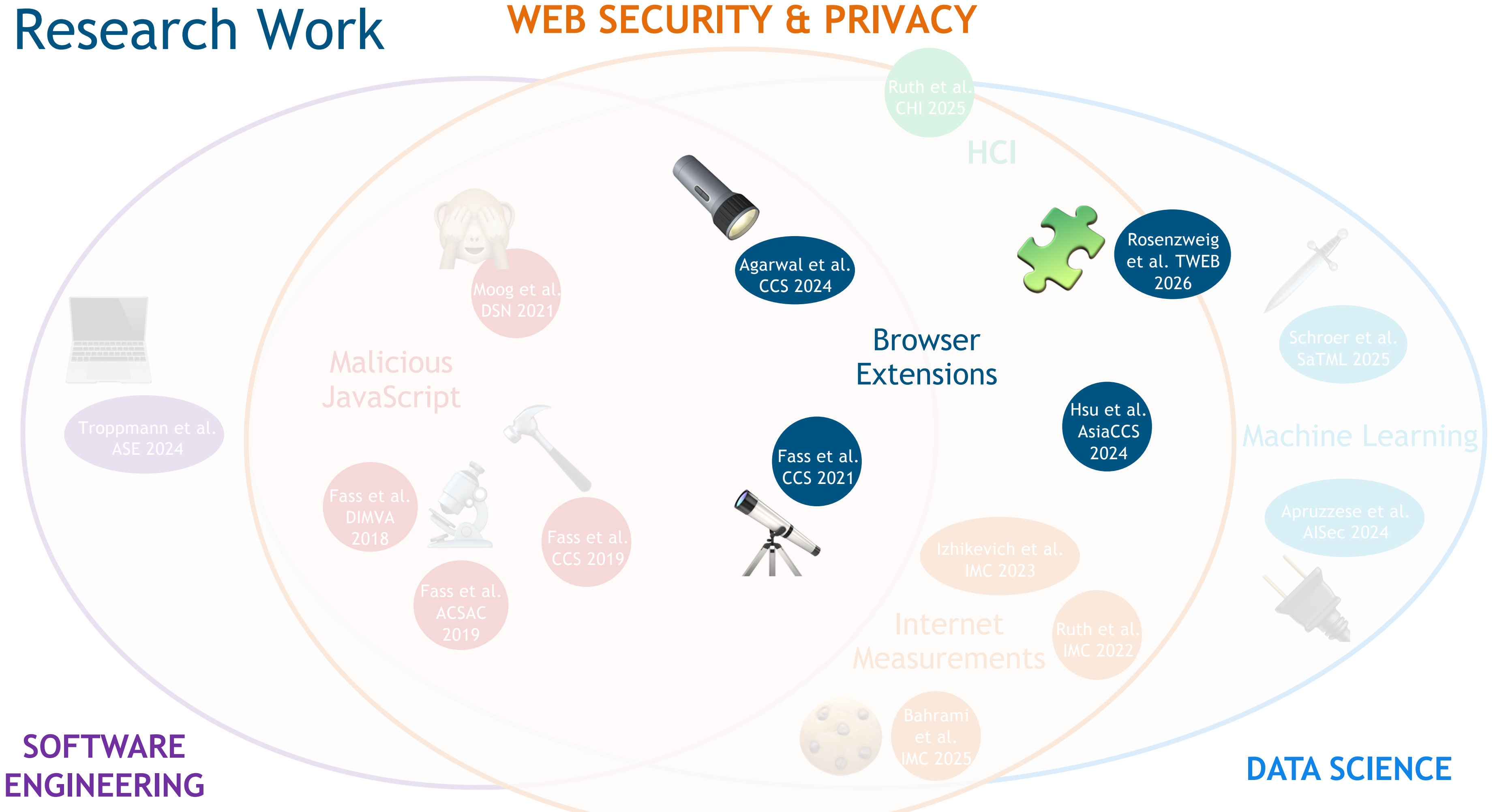
Icons indicate open-source systems / code

Research Work



Icons indicate open-source systems / code

Research Work



Icons indicate open-source systems / code

Outline

- Background: Browser Extensions
- Investigating Security-Noteworthy Extensions (SNE)
- Detecting Vulnerable Extensions
  - Threat model and automated tool (DOUBLEX)
  - Case studies, results, and potential defense strategies
- Detecting Malicious Extensions
  - Lab setting vs. real world
- Detecting Fingerprintable Extensions
  - Presentation of 3 fingerprinting vectors, results, and potential mitigations

Hsu et al. AsiaCCS 2024

Fass et al. CCS 2021

Rosenzweig et al. TWEB 2026

Agarwal et al. CCS 2024

Outline

- Background: Browser Extensions
- Investigating Security-Noteworthy Extensions (SNE)
- Detecting Vulnerable Extensions
  - Threat model and automated tool (DOUBLEX)
  - Case studies, results, and potential defense strategies
- Detecting Malicious Extensions
  - Lab setting vs. real world
- Detecting Fingerprintable Extensions
  - Presentation of 3 fingerprinting vectors, results, and potential mitigations

Hsu et al. AsiaCCS 2024

Fass et al. CCS 2021

Rosenzweig et al. TWEB 2026

Agarwal et al. CCS 2024

What are Browser Extensions?

- Third-party programs to **improve user browsing experience**

Adblock Plus - free ad blocker
Offered by: adblockplus.org

Skype
Offered by: www.skype.com

Adobe Acrobat
Offered by: Adobe Inc.

Grammarly for Chrome
Offered by: grammarly.com

Honey
Offered by: https://www.joinhoney.com

Google Translate
Offered by: translate.google.com

LastPass: Free Password Manager
Offered by: LastPass

Cisco Webex Extension
Offered by: webex.com

- ~**270k** Chrome extensions totaling ~**3B** active users
  - ➢ Necessitates a thorough **security and privacy assessment**

Background – `manifest.json`

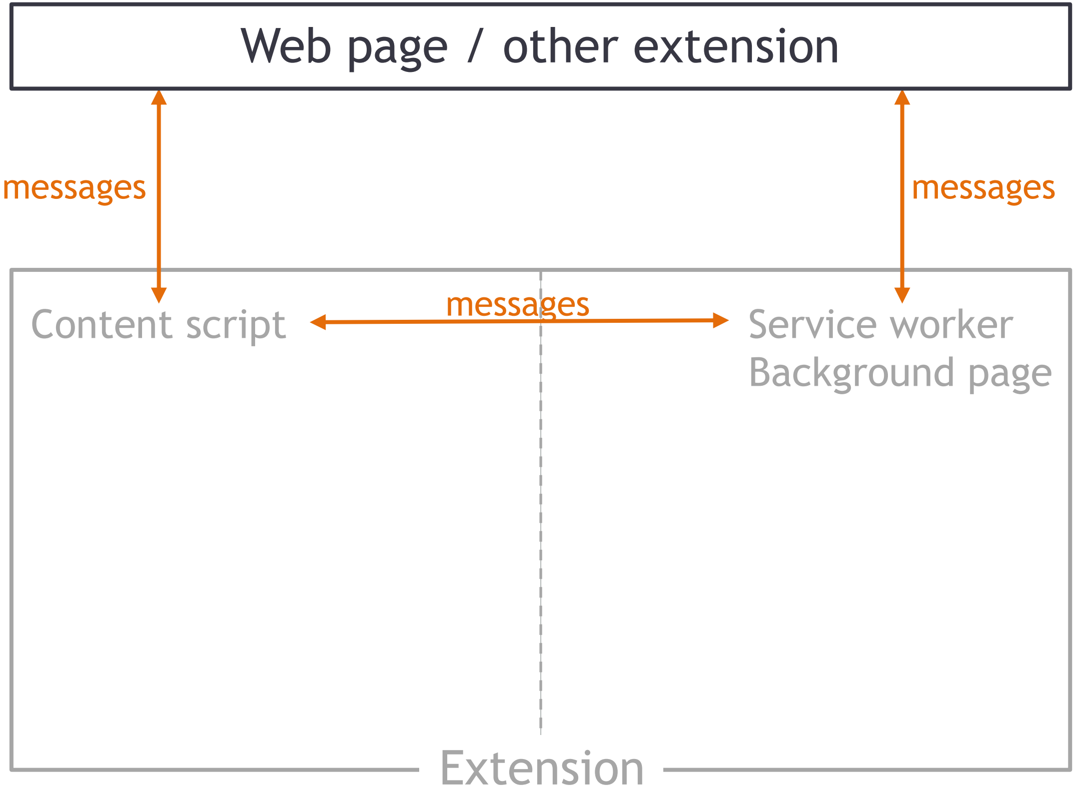
- Every extension needs a manifest written in JSON, called `manifest.json`, which gives essential information, e.g.,
  - Extension's name, version, and manifest's version
  - Main components of an extension (CS, BP/SW, …)
  - Permissions of an extension (`downloads`, `history`, …)
  - …

https://developer.chrome.com/docs/extensions/reference/manifest

Background – Extension Architecture

- Service worker (SW in MV3) / Background page (BP in old MV2):
  - Core logic of an extension
  - Executed independently of the lifetime of a tab / window
  - Privileged part of an extension
- Content scripts (CS):
  - Injected by an extension into (a) web page(s)
  - Can use standard DOM APIs to read / modify a web page
  - Similar to scripts directly loaded by a web page + some more privileges
  - Restricted access to extension APIs

Background – Extension Architecture & Messages



Background — Authorized APIs & Permissions

- Extensions only have access to:
  - APIs explicitly declared in the `manifest.json`, e.g.,
    - `storage` - store/access data from the *extension storage*
    - `downloads` - download files
    - `history` - access to a user's browsing history
    - `bookmarks`, `cookies`, `topSites`, …
  - `host` declared in the `manifest.json` = web pages an extension can access (read/write), e.g., to do some *cross-origin* requests

- https://developer.chrome.com/docs/extensions/mv3/declare_permissions/

- https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/manifest.json/permissions

Background – manifest.json -- example

Extension's name?

Manifest version?

Which extension API(s) can be accessed?

From which domain(s) can data be fetched?

On which website(s) is the CS injected?

```
{
  "name": "My Extension",
  "version": "versionString",
  "description": "A plain text description",
  "manifest_version": 3
  "permissions": ["downloads", "history"],
  "host_permissions": ["https://example.com/*"],
  "background": {
    "service_worker": ["service_worker.js"],
  },
  "content_scripts": [{
    "matches": ["<all_urls>"],
    "js": ["content_script.js"]
  }],


}
```

Outline

- Background: Browser Extensions
- Investigating Security-Noteworthy Extensions (SNE)
- Detecting Vulnerable Extensions
  - Threat model and automated tool (DOUBLEX)
  - Case studies, results, and potential defense strategies
- Detecting Malicious Extensions
  - Lab setting vs. real world
- Detecting Fingerprintable Extensions
  - Presentation of 3 fingerprinting vectors, results, and potential mitigations

Hsu et al. AsiaCCS 2024

Fass et al. CCS 2021

Rosenzweig et al. TWEB 2026

Agarwal et al. CCS 2024

How Secure are Browser Extensions?

- Browser extensions provide additional functionality…
- … so browser extensions need additional & elevated privileges compared to web pages
  - e.g., ad-blockers need to modify web page content or intercept network requests
  - e.g., extensions can download arbitrary files and access cross-domain data; while web pages cannot (blocked by the Same-Origin Policy)

➢ Browser extensions are an attractive target for attackers 😈

Dangerous Browser Extensions

→ **Extensions can put their users' security & privacy at risk:**

- **Contain malware**: designed by malicious actors to harm victims
  - E.g., propagate malware, steal users' credentials, track users [1, 3]
- **Contain vulnerabilities**: designed by well-intentioned developers… but buggy
  - E.g., can lead to user-sensitive data exfiltration [1, 2]
- **Violate the Chrome Web Store policies**
  - E.g., deceive users, promote unlawful activities, lack a privacy policy [1]
- **Be fingerprintable**: can be recognized and uniquely identified
  - E.g., can lead to user tracking or inferring of user personal information [4]

[1] Hsu, Tran, and Fass; *AsiaCCS* 2024 | [2] Fass, Somé, Backes, and Stock; *CCS* 2021 | [3] Rosenzweig, Dalla Valle, Apruzzese, and Fass; *TWEB* 2026 |
[4] Agarwal, Fass, and Stock; *CCS* 2024

Dangerous Browser Extensions

→ **Extensions can put their users' security & privacy at risk:**

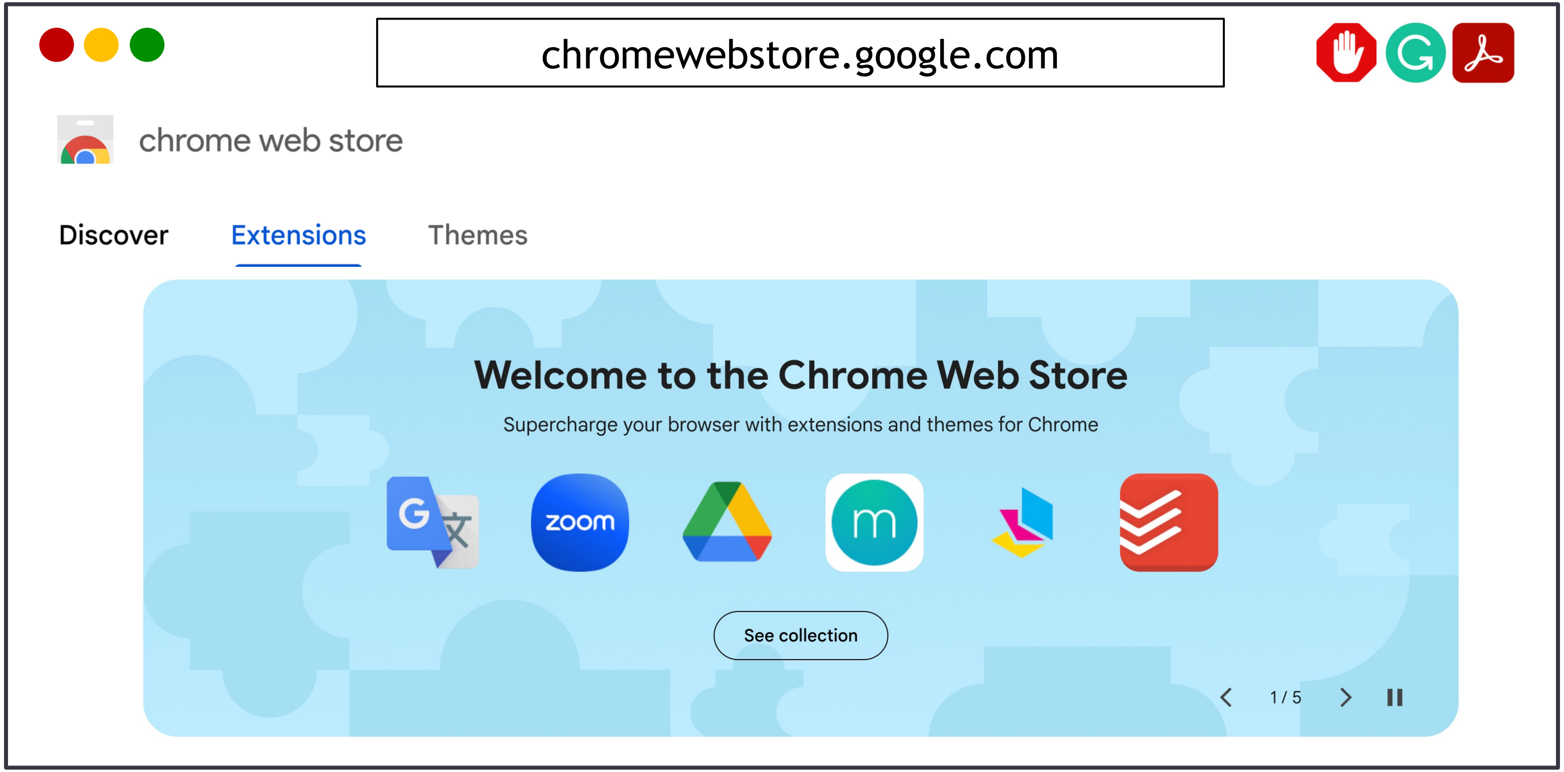
- **Contain malware:** designed by malicious actors to harm victims
  - E.g., propagate malware, steal users' credentials, track users [1, 3]
- **Contain vulnerabilities:** designed by well-intentioned developers… but buggy
  - E.g., can lead to user-sensitive data exfiltration [1, 2]
- **Violate the Chrome Web Store policies**
  - E.g., deceive users, promote unlawful activities, lack a privacy policy [1]
- **Be fingerprintable:** can be recognized and uniquely identified

➢ **Security-Noteworthy Extensions (SNE)** [1]

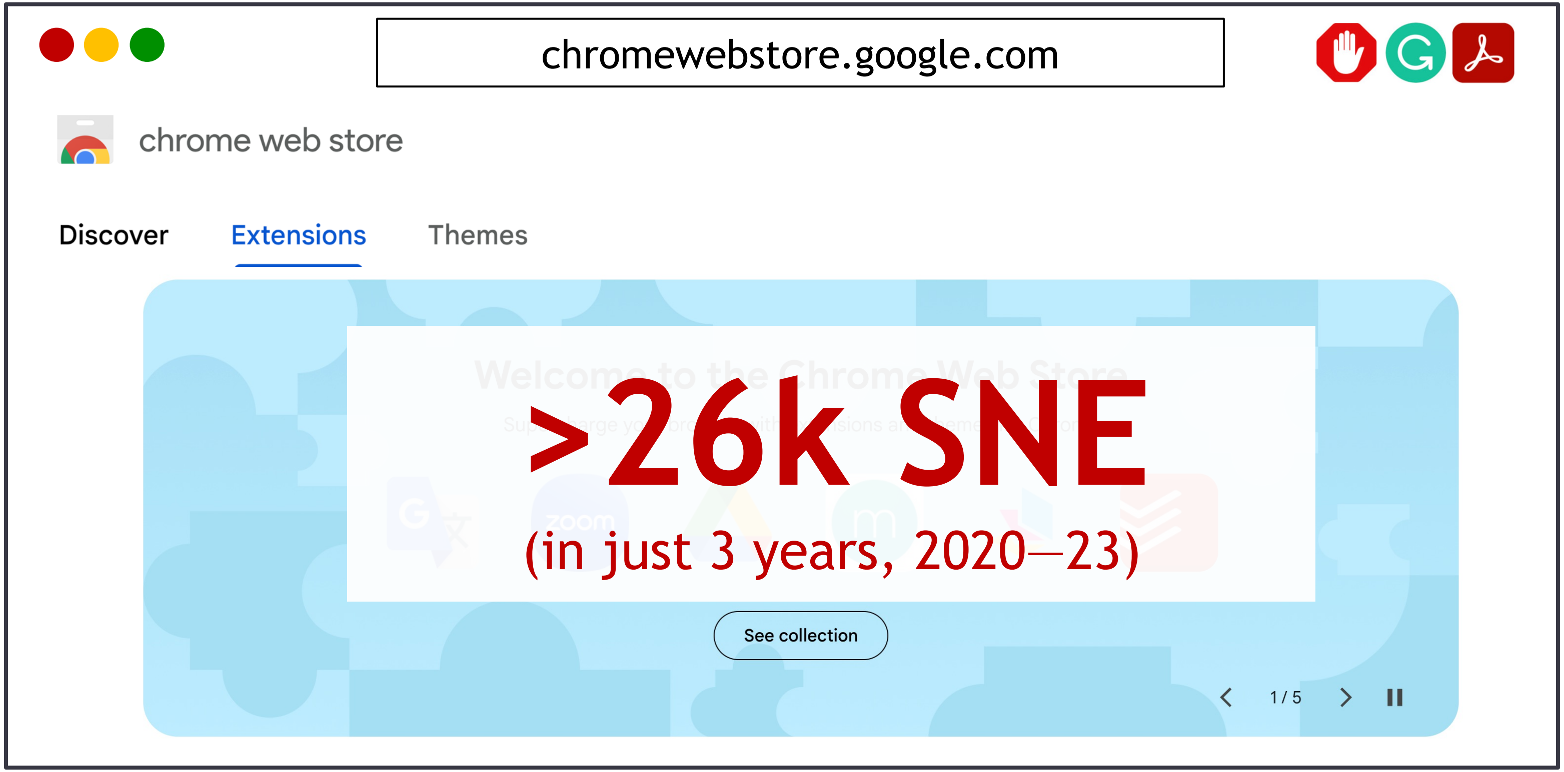
  - E.g., can lead to user tracking or inferring of user personal information [4]

[1] Hsu, Tran, and Fass; *AsiaCCS* 2024 | [2] Fass, Somé, Backes, and Stock; *CCS* 2021 | [3] Rosenzweig, Dalla Valle, Apruzzese, and Fass; *TWEB* 2026 |
[4] Agarwal, Fass, and Stock; *CCS* 2024

How are Security-Noteworthy Extensions (SNE) Installed?



How are Security-Noteworthy Extensions (SNE) Installed?



[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

Main Findings on the Chrome Web Store

- **350M users** installed **Security-Noteworthy Extensions** in 2020—2023
- These **dangerous extensions** stay in the Chrome Web Store ***for years***
- **60%** of extensions have **never received a single update**

What is in the Chrome Web Store?

Sheryl Hsu
Stanford University
sherylh@stanford.edu

Manda Tran
Stanford University

Aurore Fass
Stanford University, CISPA Helmholtz Center for Information Security
fass@cispa.de

ABSTRACT

This paper is the first attempt at providing a holistic view of the Chrome Web Store (CWS). We leverage historical data provided by ChromeStats to study global trends in the CWS and security implications. We first highlight the extremely *short life cycles of extensions*: roughly 60% of extensions stay in the CWS for one year. Second, we define and show that *Security-Noteworthy Extensions (SNE)* are a significant issue: they pervade the CWS for years and affect almost 350 million users. Third, we identify *clusters of extensions with a similar code base*. We discuss how code similarity techniques could be used to flag suspicious extensions. By developing an approach to extract URLs from extensions' comments, we show that extensions reuse code snippets from public repositories or forums, leading to the propagation of dated code and vulnerabilities. Finally, we underline a critical *lack of maintenance in the CWS*: 60% of the extensions in the CWS have never been updated; half of the extensions known to be vulnerable are still in the CWS and still vulnerable 2 years after disclosure; a third of extensions use vulnerable library versions. We believe that these issues should be widely known in order to pave the way for a more secure CWS.

CCS CONCEPTS

• **Security and privacy → Web application security**; **Browser security**.

KEYWORDS

Browser Extension, Chrome Web Store, ChromeStats, Security-Noteworthy Extension, SNE, Malware, Policy Violation, Vulnerability, Life Cycle, Maintenance, Vulnerable Library, Code Similarity

ACM Reference Format:
Sheryl Hsu, Manda Tran, and Aurore Fass. 2024. What is in the Chrome Web Store?. In *ACM Asia Conference on Computer and Communications Security (ASIA CCS '24), July 1–5, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3634737.3637636

1 INTRODUCTION

Browser extensions provide additional functionality and customization for browsers. The most popular desktop browser Chrome (with a market share of 66% [70]) has almost 125k extensions, totaling over 1.6 billion active users [6]. Unfortunately, due to their often specialized or privileged capabilities, browser extensions can either be a tool or a target for attackers. Specifically, attackers are developing *malicious extensions* to, e.g., spread malware via malvertising [1], track users [72], spy on users [14], or steal credentials and other sensitive information [22]. At the same time, other extensions have inherently benign functionalities but *contain vulnerabilities* which, if exploited, lead to, e.g., universal cross-site scripting or leaking of sensitive user data [41, 64, 74]. Finally, merely using extensions (independent of whether they have any known flaws) represents a *privacy risk* for Web users. For example, it is possible to infer the set of extensions a user has installed, by observing side effects some extensions induce (browser extension fingerprinting) [50, 59, 60, 62, 63, 67, 69]. This enables an attacker to track users across websites or infer sensitive information about them [46].

To mitigate these issues, browser vendors review extensions prior to publication, e.g., Google engineers vet extensions before their deployment in the Chrome Web Store (CWS) [32]. Despite this vetting process and a decade of work on securing extensions, malicious, vulnerable, and fingerprintable extensions are still found in the CWS [41, 54]. We argue that a potential solution to this issue requires a comprehensive understanding of the underlying ecosystem of browser extensions. Indeed, and perhaps surprisingly, very little is currently known about what lies in browser extension galleries.

This paper provides a holistic view of the browser extension landscape within the CWS. We focus on the CWS because Chrome is the most popular browser, and Chrome extensions are built using the WebExtensions API–a cross-browser technology compatible with Firefox, Opera, Microsoft Edge, etc [56].

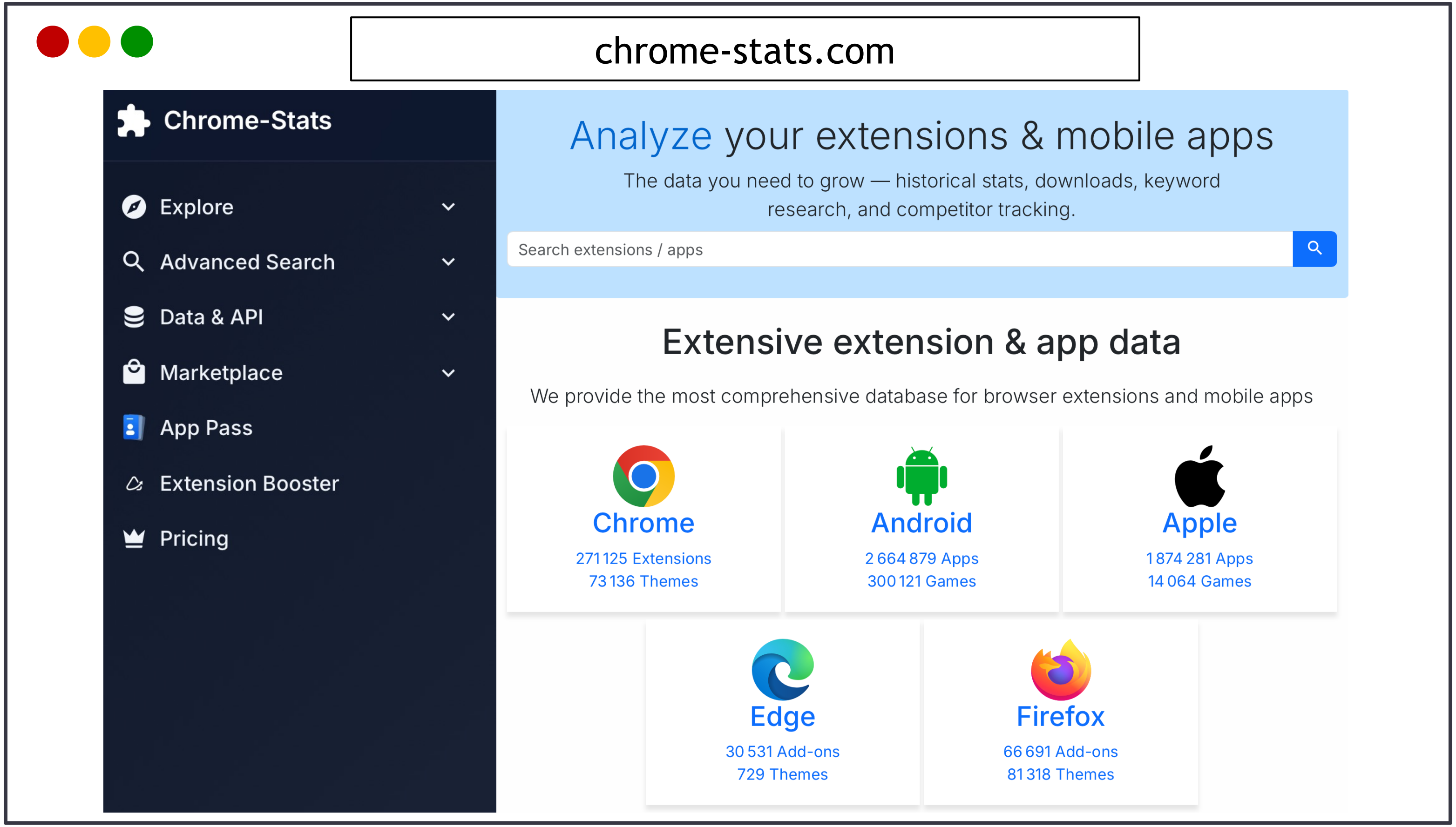
We begin our analyses by investigating overall trends in the CWS, along with potential security problems that could arise from having such a big and diverse code base. We first highlight unexpected volatility in terms of extensions in the CWS, e.g., only 60% of extensions are available for one year. Second, we define and investigate "Security-Noteworthy Extensions" (SNE): we analyze malware-containing, policy-violating, and vulnerable extensions. We find that these SNE are a significant problem: over 346 million users installed a SNE in the last 3 years (280M malware, 63M policy violation, and 3M vulnerable). In addition, these extensions are staying in the CWS *for years*, making thorough vetting of extensions and notification of impacted users all the more critical. Third, we uncover thousands of clusters of similar extensions, and we show that investigating extension codes for similarities may enable us to identify unknown SNE. Subsequently, we develop an approach to extract URLs from extensions' comments and attribute instances of code reuse to these cited URLs. We show concrete evidence that extensions reuse code from public repositories or forums directly, and we exemplify how code reuse leads to the propagation of vulnerabilities. Fourth, we show that extensions are globally not maintained: 60% of the extensions in the CWS have never been updated. This has

> What is in the Chrome Web Store?

In *ACM AsiaCCS* 2024. Sheryl Hsu, Manda Tran, and Aurore Fass

Hsu et al.
AsiaCCS
2024

Browser Extension Collection: Chrome-Stats



[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

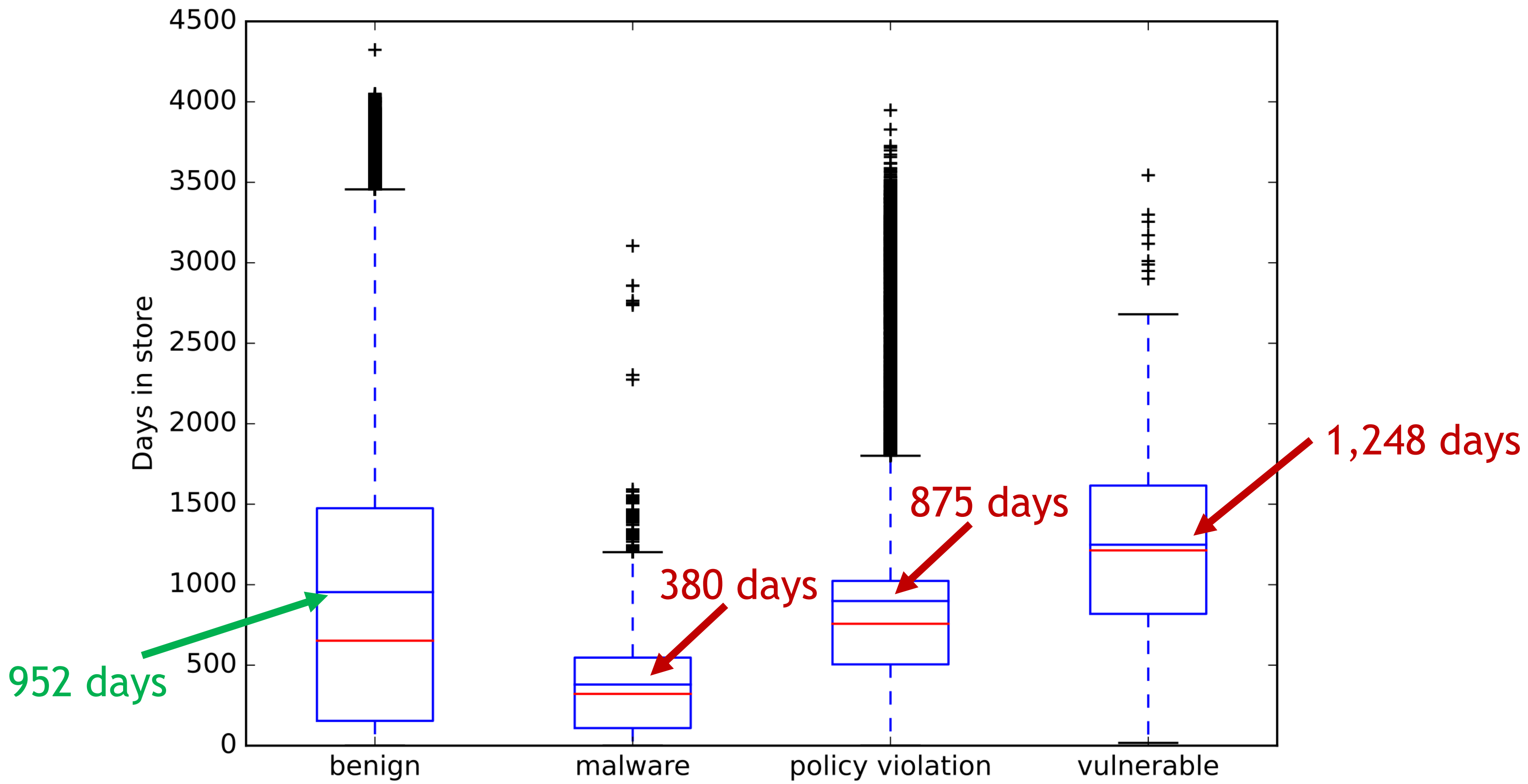
Browser Extension Collection: Chrome-Stats

Category	#Extensions Metadata collected	#Extensions Code collected	When collected
SNE	26,014	16,377	Before May 1, 2023
- Malware-containing	10,426	6,587	Before May 1, 2023
- Policy-violating	15,404	9,638	Before May 1, 2023
- Vulnerable [2]	184	152	March 16, 2021
"Benign" extensions	226,762	92,482	Before May 1, 2023

[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

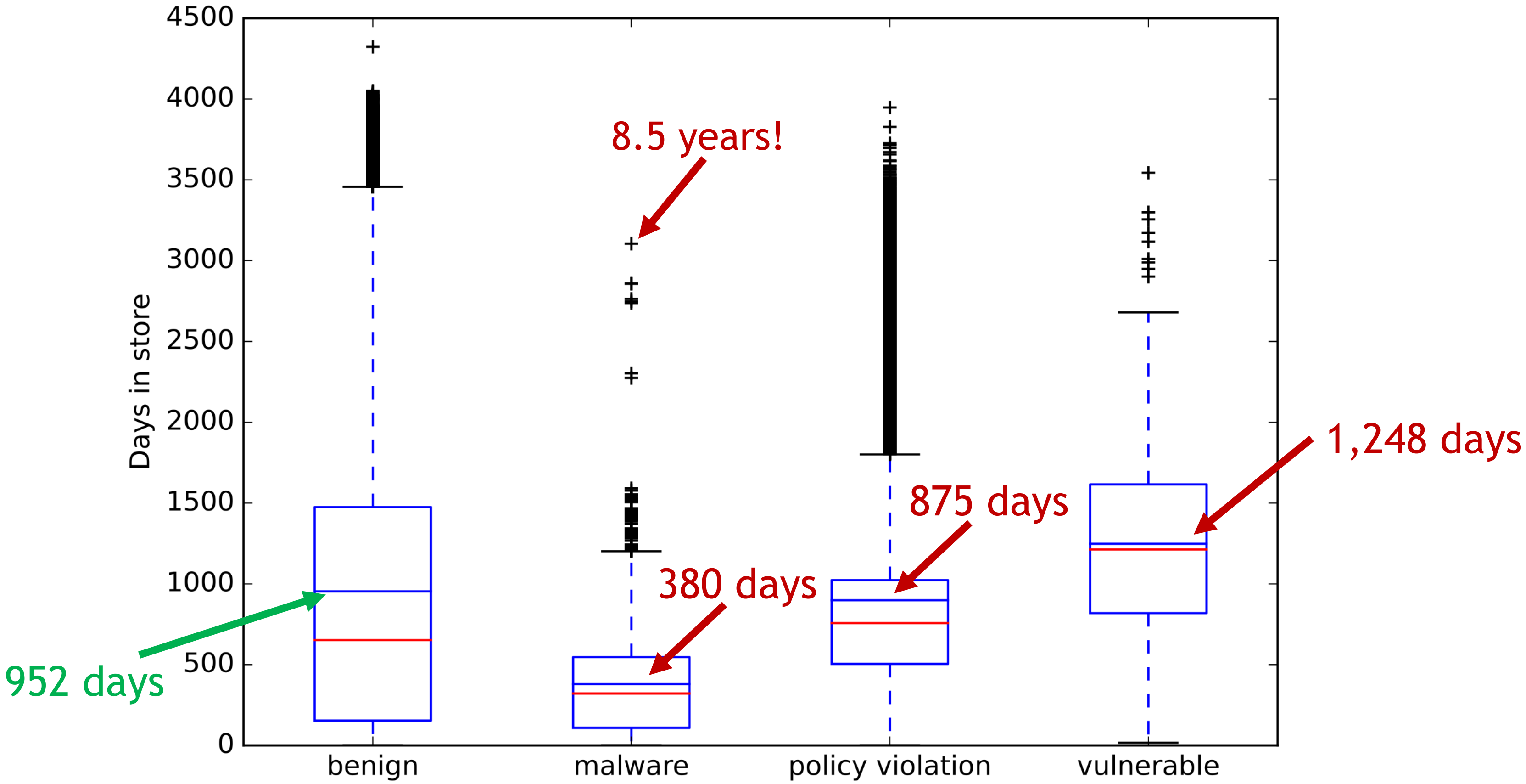
[2] Fass et al.; *CCS* 2021

Number of Days in the CWS



[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

Number of Days in the CWS



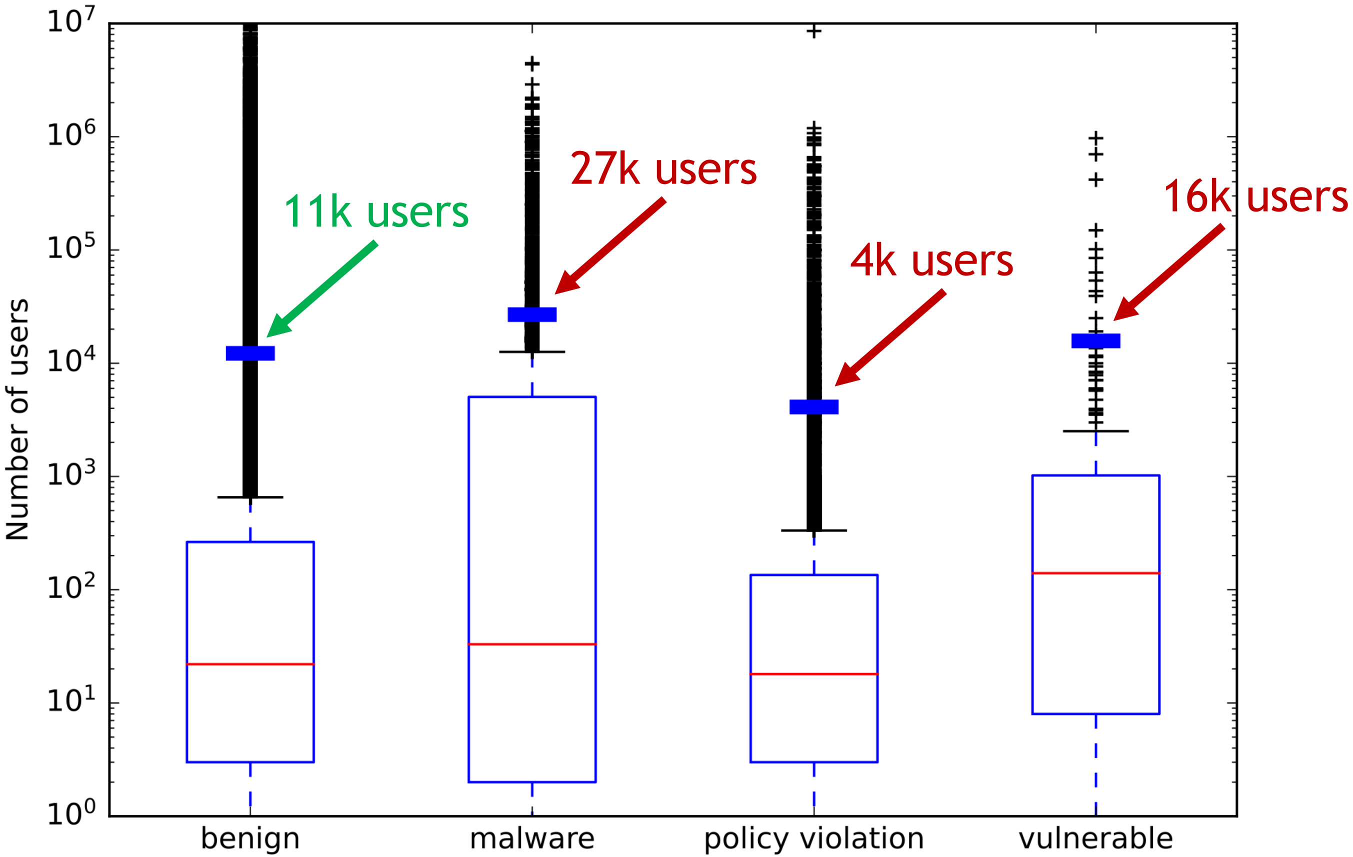
[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

Number of Days in the CWS

- **SNE put the security & privacy of Web users at risk *for years***

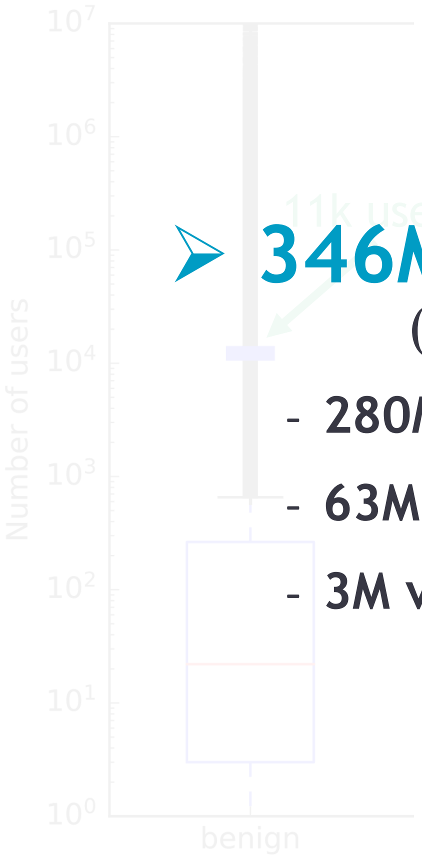
,248 days

Number of Users



[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

Number of Users



➢ **346M** users installed **SNE**

(between 2020—2023):

- **280M malware-containing**
- **63M policy-violating**
- **3M vulnerable extensions**

[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

Media Coverage

Forbes

FORBES > INNOVATION > CYBERSECURITY

280 Million Google Chrome Users Installed Dangerous Extensions, Study Says

Davey Winder Senior Contributor

Davey Winder is a veteran cybersecurity writer, hacker and analyst.

Follow

Jun 24, 2024, 06:57am EDT

How safe are Google Chrome extensions? SOPA IMAGES/LIGHTROCKET VIA GETTY IMAGES

The Register®

Risk of installing dodgy extensions from Chrome store way worse than Google's letting on, study suggests

All depends on how you count it – Chocolate Factory claims 1% fail rate

Thomas Claburn Sun 23 Jun 2024 // 10:36 UTC

TECHSPOT

TRENDING FEATURES REVIEWS THE BEST DOWNLOADS PRODUCT FINDER FORUMS

SECURITY THE WEB MALWARE CHROME

Researchers say 280 million people have installed malware-infected Chrome extensions in the last 3 years

Google claims less than 1% of all installs include malware

By Rob Thubron June 24, 2024 at 11:39 AM

ADGUARD

Subscribe to news Search blog

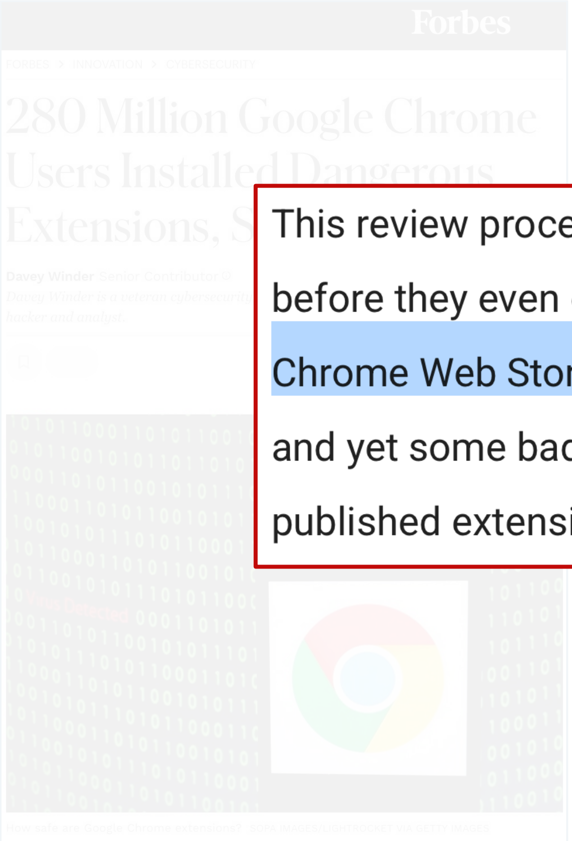
AdGuard > Blog > Google is failing miserably at weeding out bad extensions, new research indicates

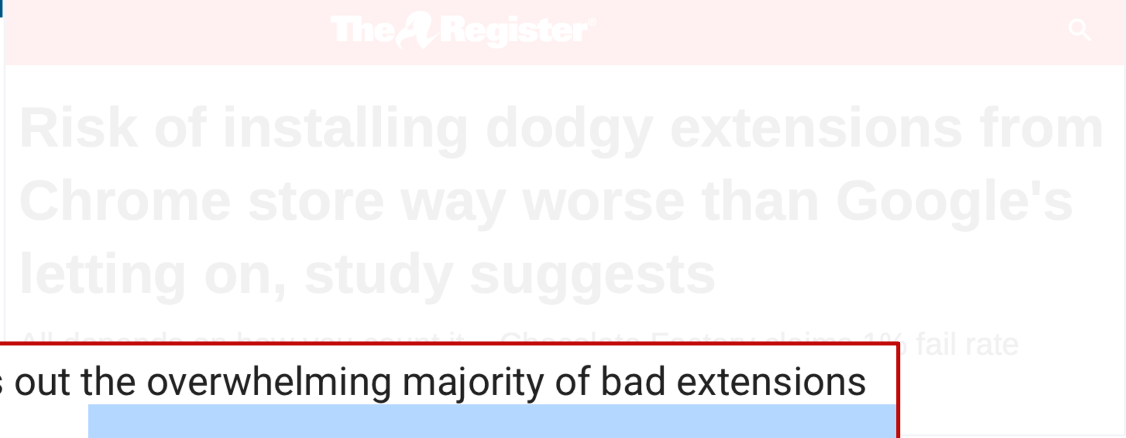
Google is failing miserably at weeding out bad extensions, new research indicates

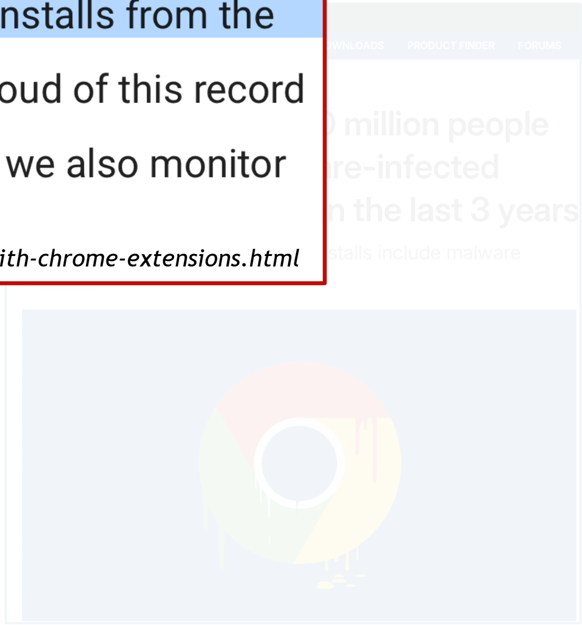
July 5, 2024 · 7 min read

[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

Media Coverage







This review process weeds out the overwhelming majority of bad extensions before they even get published. In 2024, less than 1% of all installs from the Chrome Web Store were found to include malware. We're proud of this record and yet some bad extensions still get through, which is why we also monitor published extensions.

https://security.googleblog.com/2024/06/staying-safe-with-chrome-extensions.html

[1] Hsu, Tran, and Fass; *AsiaCCS* 2024

Outline

- Background: Browser Extensions
- Investigating Security-Noteworthy Extensions (SNE)
- Detecting Vulnerable Extensions
  - Threat model and automated tool (DOUBLEX)
  - Case studies, results, and potential defense strategies
- Detecting Malicious Extensions
  - Lab setting vs. real world
- Detecting Fingerprintable Extensions
  - Presentation of 3 fingerprinting vectors, results, and potential mitigations

Hsu et al. AsiaCCS 2024

Fass et al. CCS 2021

Rosenzweig et al. TWEB 2026

Agarwal et al. CCS 2024

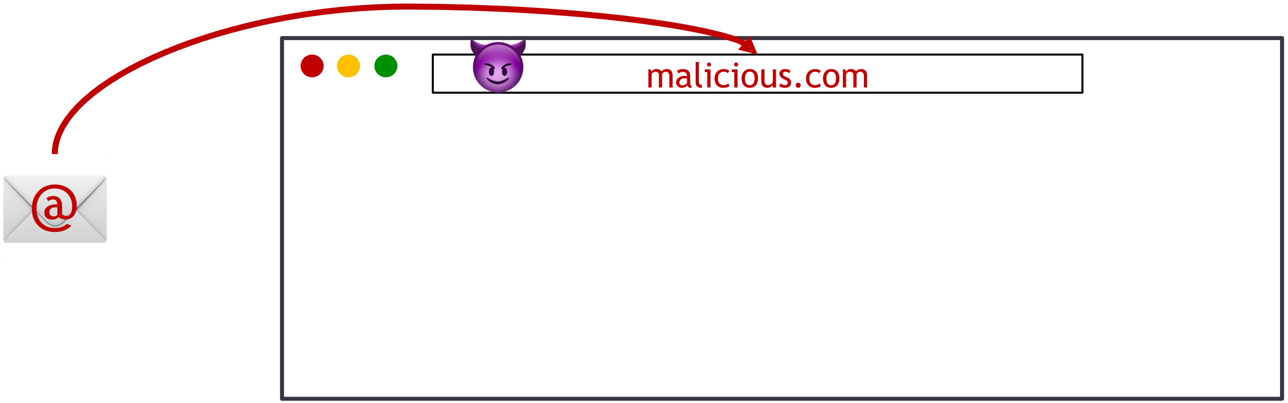
Analysis of Vulnerable Extensions: Threat Model

Challenging to detect due to their inherently benign intent (*benign-but-buggy*)

[2] Fass et al.; *CCS* 2021

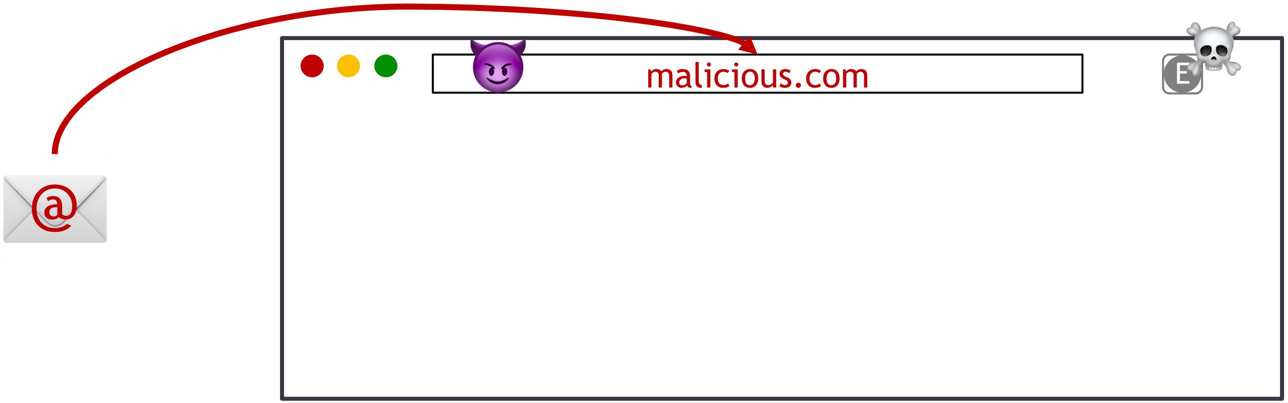
Analysis of Vulnerable Extensions: Threat Model

Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



Analysis of Vulnerable Extensions: Threat Model

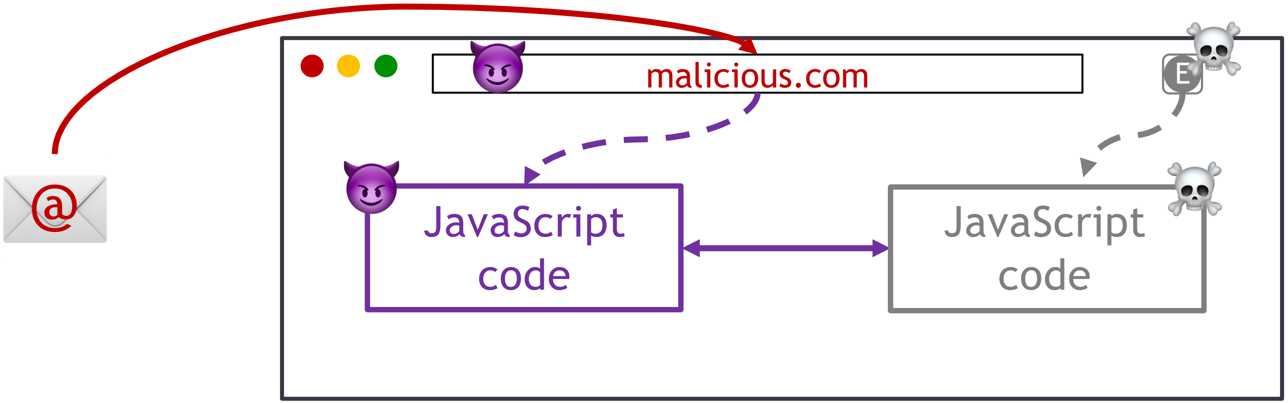
Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



[2] Fass et al.; *CCS* 2021

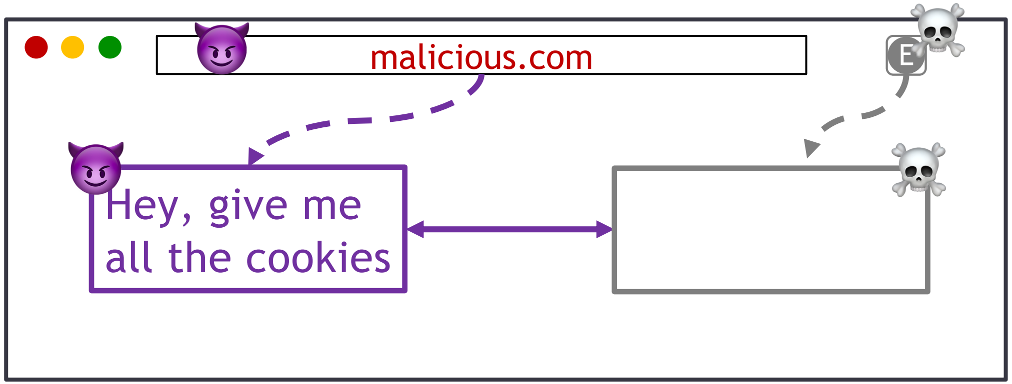
Analysis of Vulnerable Extensions: Threat Model

Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



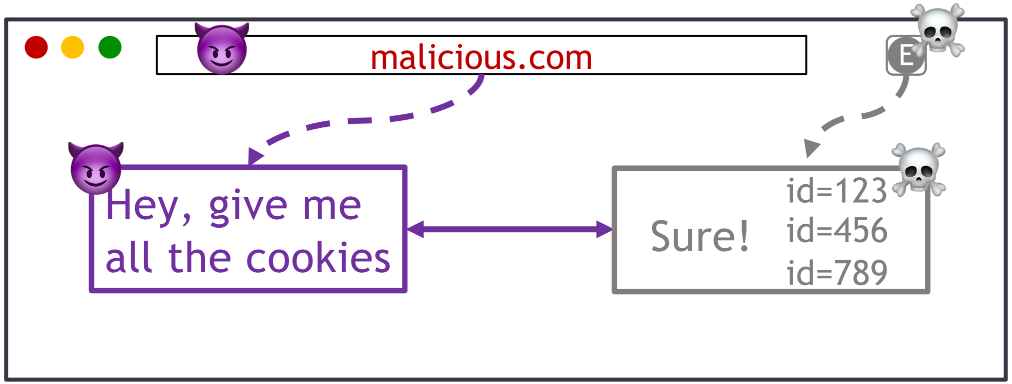
Analysis of Vulnerable Extensions: Threat Model

Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



Analysis of Vulnerable Extensions: Threat Model

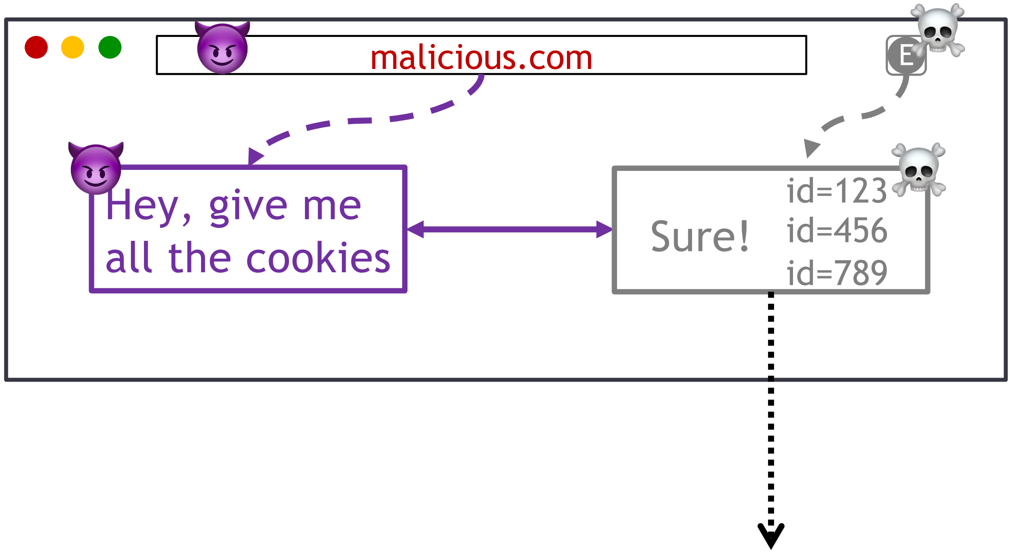
Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



[2] Fass et al.; *CCS* 2021

Analysis of Vulnerable Extensions: Threat Model

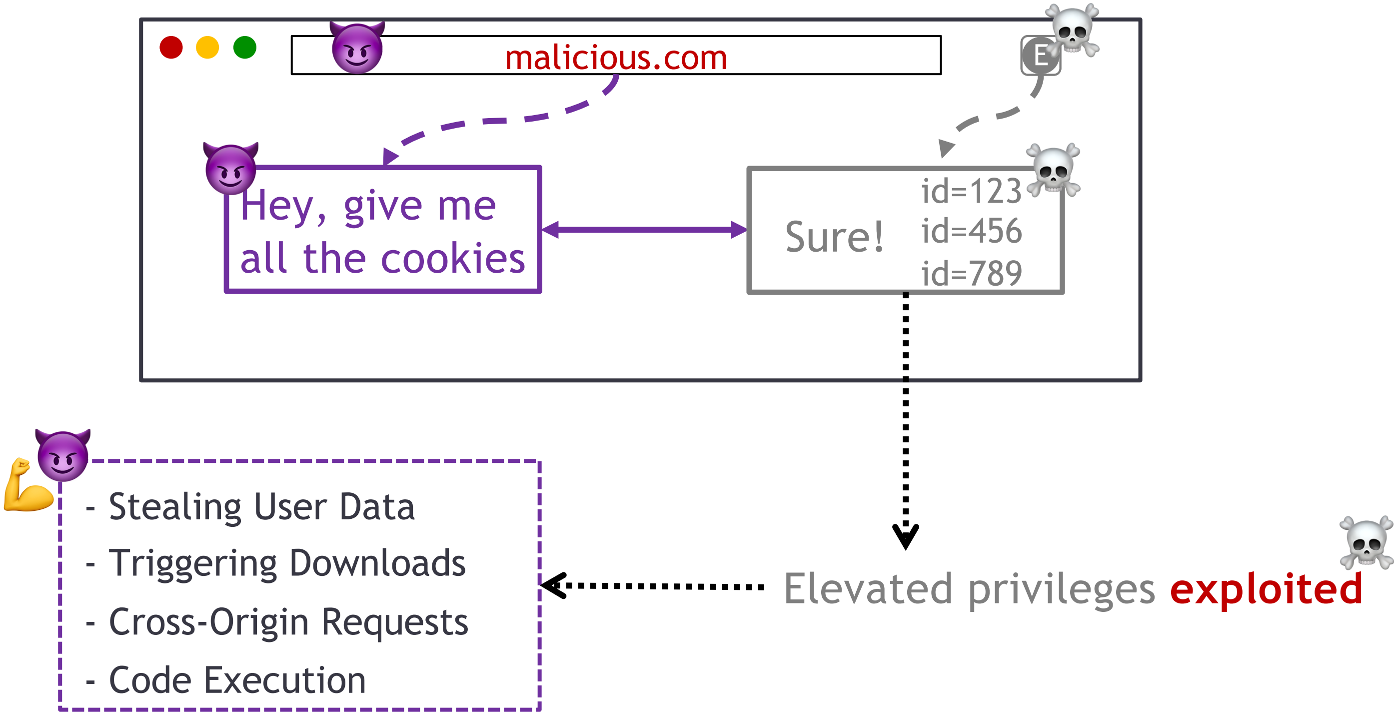
Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



Elevated privileges **exploited**

Analysis of Vulnerable Extensions: Threat Model

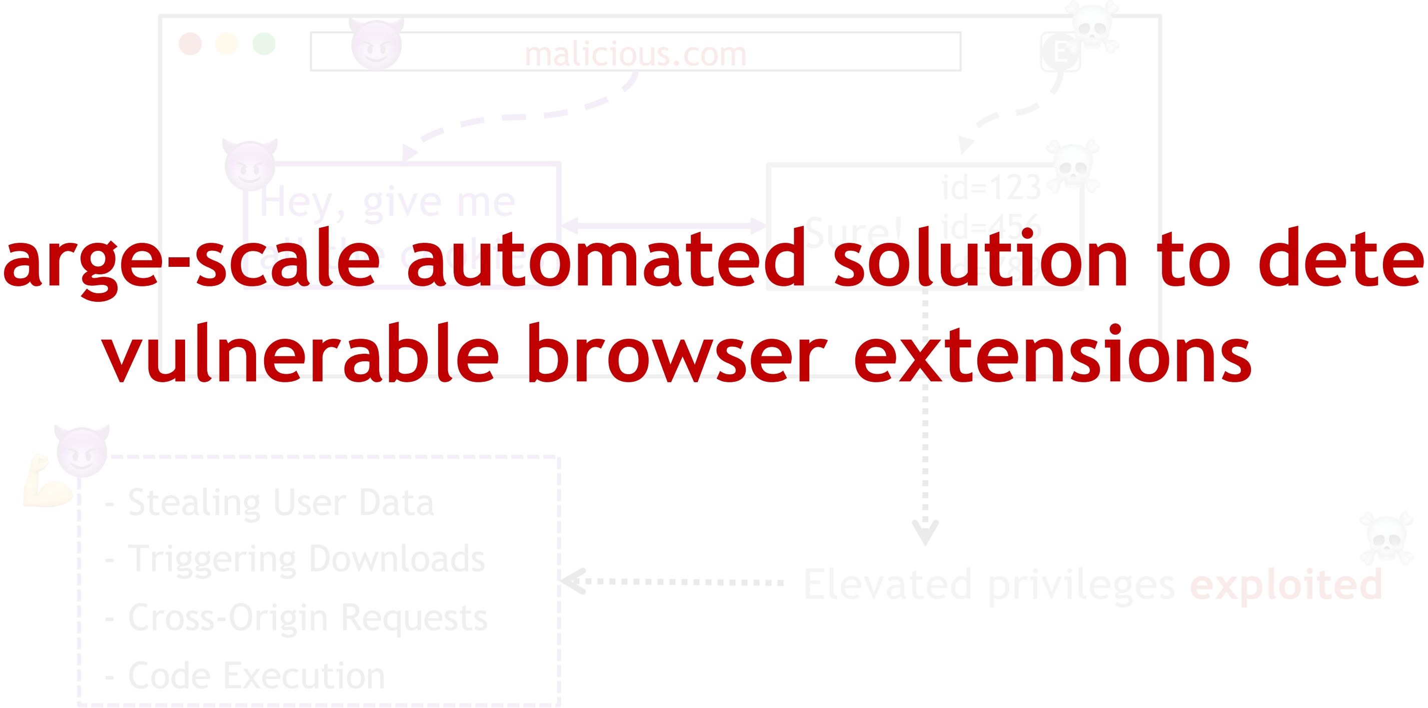
Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



[2] Fass et al.; *CCS* 2021

Analysis of Vulnerable Extensions: Threat Model

Challenging to detect due to their inherently benign intent (*benign-but-buggy*)



No large-scale automated solution to detect vulnerable browser extensions

[2] Fass et al.; *CCS* 2021

Detecting Vulnerable Extensions


DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale

Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock
CISPA Helmholtz Center for Information Security
{aurore.fass,doliere.some,backes,stock}@cispa.de

Abstract

Browser extensions are popular to enhance users' browsing experience. By design, they have access to security- and privacy-critical APIs to perform tasks that web applications cannot traditionally do. Even though web pages and extensions are isolated, they can communicate through messages. Specifically, a vulnerable extension can receive messages from another extension or web page, under the control of an attacker. Thus, these communication channels are a way for a malicious actor to elevate their privileges to the capabilities of an extension, which can lead to, e.g., universal cross-site scripting or sensitive user data exfiltration. To automatically detect such security and privacy threats in benign-but-buggy extensions, we propose our static analyzer DoubleX. DoubleX defines

1 Introduction

The Web has become a popular ecosystem used by billions of people every day. To extend their browser functionality, users install browser extensions. For the most popular desktop browser Chrome (with a market share of 67% [78]), there are almost 200,000 extensions, totaling over 1.2 billion installs [26]. While some extensions merely customize user browser interface, others serve more security- and privacy-critical tasks, e.g., to be effective, an ad-blocker needs to modify web page content or intercept network requests. To achieve this, and contrary to regular JavaScript in web pages, extensions have privileged capabilities, such as downloading arbitrary files or accessing arbitrary cross-domain data.

Fass et al.
CCS 2021

> DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions

In *ACM CCS* 2021. Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock

[2] Fass et al.; *CCS* 2021

Detecting Vulnerable Extensions

Fass et al.
CCS 2021

DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale

Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock
CISPA Helmholtz Center for Information Security
{aurore.fass,doliere.some,backes,stock}@cispa.de

Abstract

Browser extensions are popular to enhance users' browsing experience. By design, they have access to security- and privacy-critical APIs to perform tasks that web applications cannot traditionally do. Even though web pages and extensions are isolated, they can communicate through messages. Specifically, a vulnerable extension can receive messages from another extension or web page, under the control of an attacker. Thus, these communication channels are a way for a malicious actor to elevate their privileges to the capabilities of an extension, which can lead to, e.g., universal cross-site scripting or sensitive user data exfiltration. To automatically detect such security and privacy threats in benign-but-buggy extensions, we propose our static analyzer DoubleX. DoubleX defines an Extension Dependence Graph (EDG), which abstracts extension code with control and data flows, pointer analysis, and models the message interactions within and outside of an extension. This way, we can leverage this graph to track and detect suspicious data flows between external actors and sensitive APIs in browser extensions.

We evaluated DoubleX on 154,484 Chrome extensions, where it flags 278 extensions as having a suspicious data flow. Overall, we could verify that 89% of these flows can be influenced by external actors (i.e., an attacker). Based on our threat model, we subsequently demonstrate exploitability for 184 extensions. Finally, we evaluated DoubleX on a labeled vulnerable extension set, where it accurately detects almost 93% of known flaws.

CCS Concepts

• **Security and privacy → Web application security; Browser security.**

Keywords

Web Security, Browser Extension, JavaScript, Vulnerability Detection, Static Analysis, Data Flow Analysis

ACM Reference Format:

Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock. 2021. DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '21), November 15–19, 2021, Virtual Event, Republic of Korea.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3460120.3484745

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CCS '21, November 15–19, 2021, Virtual Event, Republic of Korea
© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-8454-4/21/11...$15.00
https://doi.org/10.1145/3460120.3484745

1 Introduction

The Web has become a popular ecosystem used by billions of people every day. To extend their browser functionality, users install browser extensions. For the most popular desktop browser Chrome (with a market share of 67% [78]), there are almost 200,000 extensions, totaling over 1.2 billion installs [26]. While some extensions merely customize user browser interface, others serve more security- and privacy-critical tasks, e.g., to be effective, an ad-blocker needs to modify web page content or intercept network requests. To achieve this, and contrary to regular JavaScript in web pages, extensions have privileged capabilities, such as downloading arbitrary files or accessing arbitrary cross-domain data.

Given their elevated privileges, extensions attract the interest of attackers [1, 35, 39, 81, 82]. Still, Google engineers are actively working on detecting such malicious extensions in their store. In February 2020, they removed 500 extensions that were exfiltrating user data [41]; in April 2020, 49 additional extensions that were hijacking users' cryptocurrency wallets [42]; in June 2020, an extra 70 spying extensions [48]; and, in December 2020, extensions redirecting to phishing websites [43]. In addition, before being published, extensions are reviewed by Chrome's vetting system to flag extensions requiring many or powerful privileges for further analyses [16] and to directly detect the ones that may contain or spread malicious software. This process makes it harder to have malicious extensions in the store today.

Still, malicious extensions represent only a fraction of the extensions that may lead to security or privacy issues. In fact, an attacker can also abuse vulnerable extensions to elevate its privileges through the capabilities of an extension. To this end, an attacker can leverage an extension's communication channels to send payloads to this extension, tailored to exploit its vulnerabilities. Such vulnerabilities can lead to, e.g., universal cross-site scripting (XSS) (i.e., the ability to execute code in every website even without a vulnerability in the site itself) or sensitive user data exfiltration. Due to their inherently benign intent, these vulnerable extensions are more challenging to detect than malicious ones, e.g., as they are not doing anything suspicious. Furthermore, even though they do require powerful privileges, their benign nature allows them to pass the review process. While some previous works have focussed on vulnerable extensions, they were either purely formal [7], specific to the deprecated Firefox XPCOM [52] infrastructure [5, 6], or based on primarily manual analysis [8]. To the best of our knowledge, only EmPoWeb from Somé [72] focuses on analyzing extensions' susceptibility to attacks through external messages at scale. In practice, though, his work is based on a lightweight call graph analysis and yields an extremely high number of reported extensions to manually vet: of the 66k Chrome extensions analyzed, it flagged 3.3k as suspicious, and 95% of them were false alarms. In fact, we

> DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions

In *ACM CCS* 2021. Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock

[2] Fass et al.; *CCS* 2021

Detecting Vulnerable Extensions

Fass et al.
CCS 2021


DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale
Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock
CISPA Helmholtz Center for Information Security

> DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions

In *ACM CCS* 2021. Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock

😈 Malicious web page

Content script

Background page
Service worker

Vulnerable extension

[2] Fass et al.; *CCS* 2021

Detecting Vulnerable Extensions

Fass et al.
CCS 2021

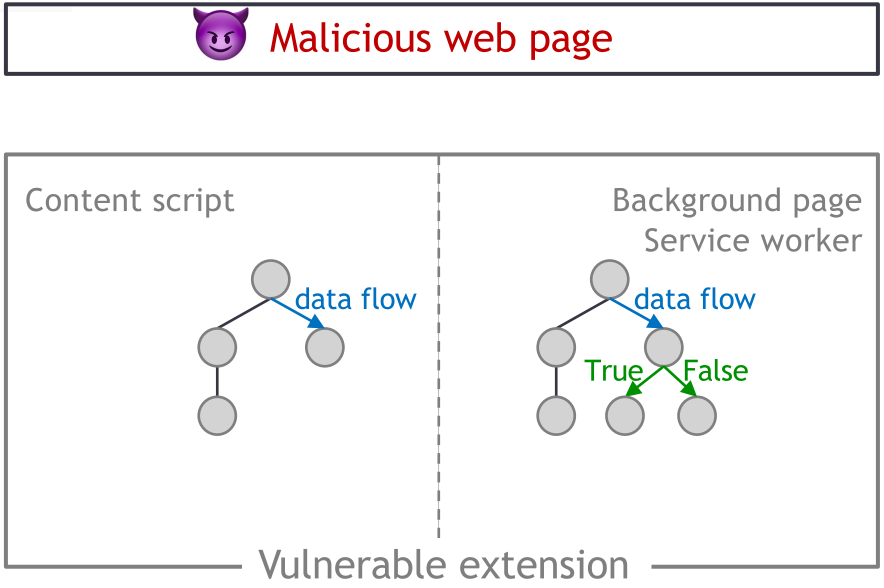
DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale

Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock

CISPA Helmholtz Center for Information Security

> DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions

In *ACM CCS* 2021. Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock



Per-component JavaScript code abstraction

- AST (Abstract Syntax Tree)
- Control flow
- Data flow
- Pointer analysis

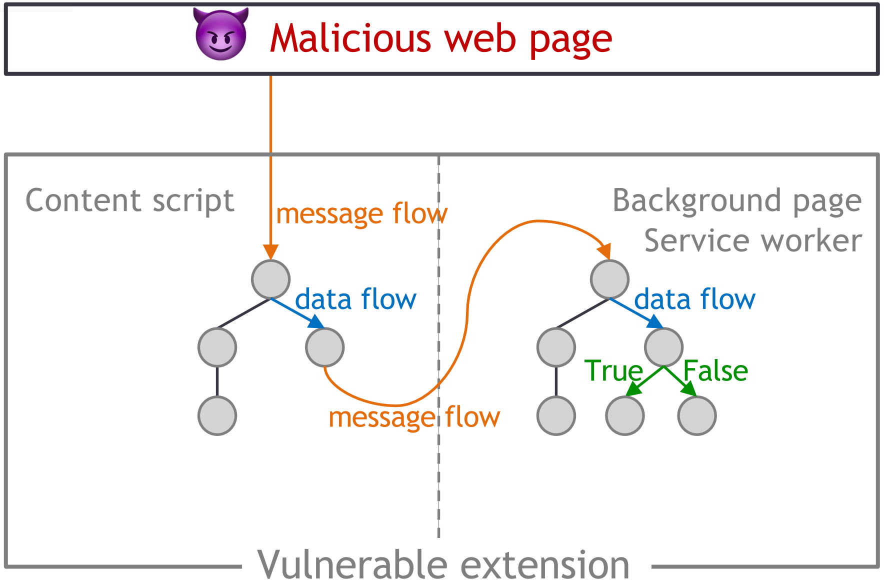
Detecting Vulnerable Extensions

Fass et al.
CCS 2021


DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale
Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock
CISPA Helmholtz Center for Information Security

> DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions

In *ACM CCS* 2021. Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock



Per-component JavaScript code abstraction

- AST (Abstract Syntax Tree)
- Control flow
- Data flow
- Pointer analysis

Extension Dependence Graph (EDG)

➢ Message interactions

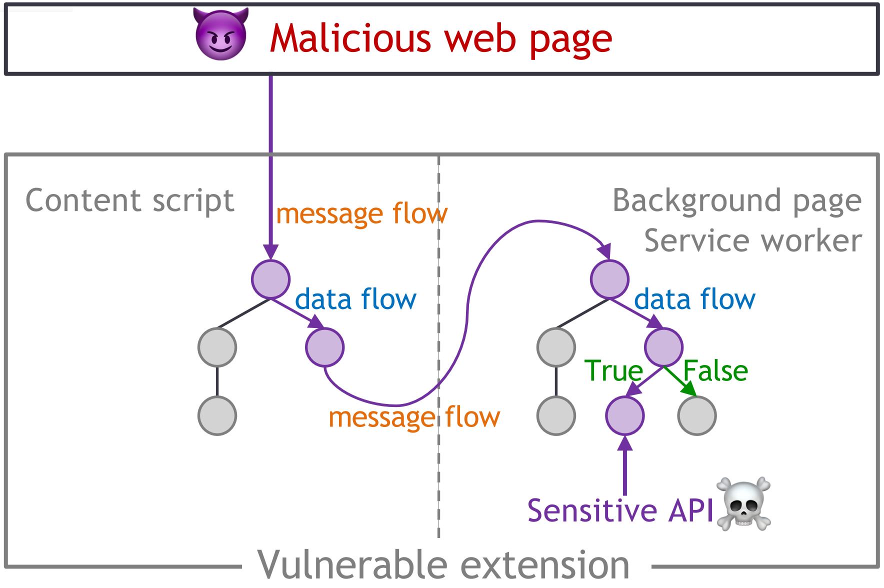
Detecting Vulnerable Extensions

Fass et al.
CCS 2021


DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale
Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock
CISPA Helmholtz Center for Information Security

> DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions

In *ACM CCS* 2021. Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock



Per-component JavaScript code abstraction

- AST (Abstract Syntax Tree)
- Control flow
- Data flow
- Pointer analysis

Extension Dependence Graph (EDG)

➢ Message interactions

Suspicious data flow tracking

➢ Detects any path between an attacker & sensitive APIs

[2] Fass et al.; *CCS* 2021

Simplified Example of a Vulnerability

```
// Content script code = from the extension
window.addEventListener("message", function(event) {
})
```

[2] Fass et al.; *CCS* 2021

Simplified Example of a Vulnerability

message received

```
// Content script code = from the extension
window.addEventListener("message", function(event) {




})
```

[2] Fass et al.; *CCS* 2021

Simplified Example of a Vulnerability

message received

```
// Content script code = from the extension
window.addEventListener("message", function(event) {

    eval(event.data);

})
```

Simplified Example of a Vulnerability

message received

```
1  // Content script code = from the extension
2  window.addEventListener("message", function(event) {
3
4      eval(event.data);
5
6  })
```


```
// DOUBLEX report
{"direct-danger1": "eval",
"value": "eval(event.data)",
"line": "4 - 4",
"dataflow": true,
"param1": {
  "received": "event",
  "line": "2 - 2"}}
```


[2] Fass et al.; *CCS* 2021

Simplified Example of a Vulnerability

```
// Content script code = from the extension
window.addEventListener("message", function(event) {

    eval(event.data);

})
```

```
// Attacker code = from the targeted web page
postMessage("alert(1)", "*")
```

(very) malicious payload

[2] Fass et al.; *CCS* 2021

Simplified Example of a Vulnerability

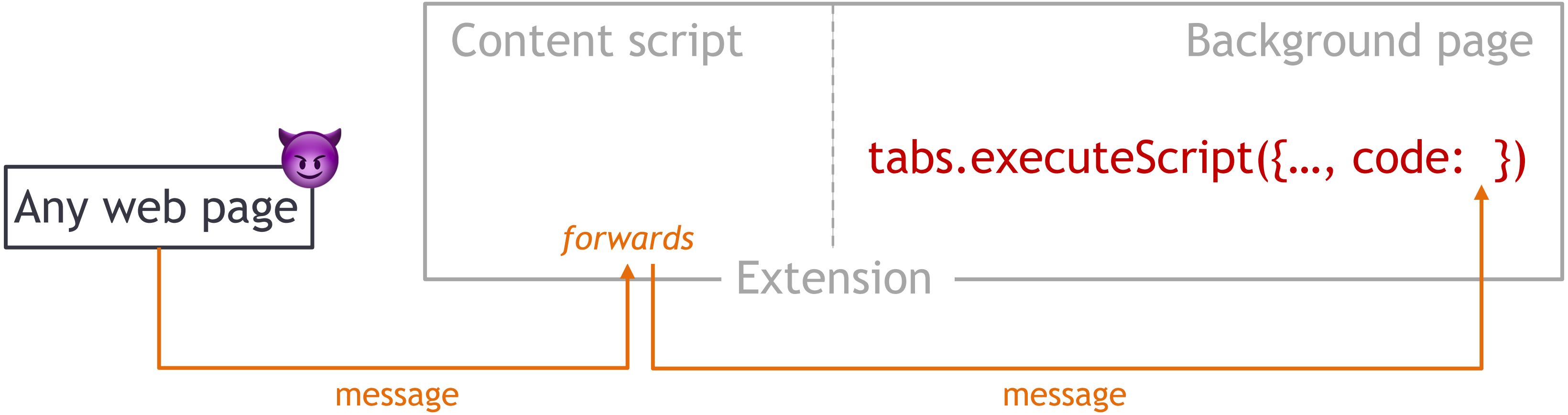
```
// Content script code = from the extension
window.addEventListener("message", function(event) {

    eval(event.data);

})
```


```
// Attacker code = from the targeted web page
postMessage("alert(1)", "*")
```

(very) malicious payload

Case Study of Vulnerable Chrome Extensions

Arbitrary code execution (*cdi…*, 4k+ users):



[2] Fass et al.; *CCS* 2021

Detecting Vulnerable Extensions with DOUBLEX

Analyzed 155k Chrome extensions from 2021 with DOUBLEX (takes 2—3s per extension)

- **184 vulnerable Chrome extensions**
- Impacting **3M users*** (* based on Google's definition)

- **Precision: 89%** of the flagged extensions are vulnerable
- **Recall: 93%** of known vulnerabilities [Somé, S&P 2019] are detected

- **Open source**, for developers and Web users

  (even in other fields, e.g., mini apps [Wang et al., ICSE 2023])

Aurore54F/DoubleX

[2] Fass et al.; *CCS* 2021

Defenses & Perspectives (for Developers)

- (Migrate an extension to Manifest V3)
- Know that communication with external actors may be dangerous
- Only allow communication with specified extensions or web pages
- Limit code execution by sanitizing messages
- DOUBLEX could provide a feedback channel for developers
- As a user?
  - **Be careful** which extensions you install
  - **Be aware** that extensions can be dangerous

[2] Fass et al.; *CCS* 2021

Outline

- Background: Browser Extensions
- Investigating Security-Noteworthy Extensions (SNE)
- Detecting Vulnerable Extensions
  - Threat model and automated tool (DOUBLEX)
  - Case studies, results, and potential defense strategies
- Detecting Malicious Extensions
  - Lab setting vs. real world
- Detecting Fingerprintable Extensions
  - Presentation of 3 fingerprinting vectors, results, and potential mitigations

Hsu et al. AsiaCCS 2024

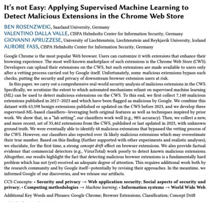
Fass et al. CCS 2021

Rosenzweig et al. TWEB 2026

Agarwal et al. CCS 2024

Detecting Malicious Extensions — Lab Setting

Rosenzweig et al. TWEB 2026



It's not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store

BEN ROSENZWEIG, Saarland University, Germany
VALENTINO DALLA VALLE, CISPA Helmholtz Center for Information Security, Germany
GIOVANNI APRUZZESE, University of Liechtenstein, Liechtenstein and Reykjavik University, Iceland
AURORE FASS, CISPA Helmholtz Center for Information Security, Germany

Google Chrome is the most popular Web browser. Users can customize it with *extensions* that enhance their browsing experience. The most well-known marketplace of such extensions is the Chrome Web Store (CWS). Developers can upload their extensions on the CWS, but such extensions are made available to users only after a vetting process carried out by Google itself. Unfortunately, some *malicious* extensions bypass such checks, putting the security and privacy of downstream browser extension users at risk.

In this paper, we carry out a comprehensive real-world security analysis of malicious extensions in the CWS. Specifically, we scrutinize the extent to which automated mechanisms reliant on supervised machine learning (ML) can be used to detect malicious extensions on the CWS. To this end, we first collect 7,140 malicious extensions published in 2017–2023 and which have been flagged as malicious by Google. We combine this dataset with 63,598 benign extensions published or updated on the CWS before 2023, and we develop three supervised-ML-based classifiers—leveraging both original features as well as techniques inspired by prior work. We show that, in a "lab setting", our classifiers work well (e.g., 98% accuracy). Then, we collect a new, and more recent, set of 35,462 extensions from the CWS, published or last updated in 2023, with unknown ground truth. We were eventually able to identify 68 malicious extensions that bypassed the vetting process of the CWS. However, our classifiers also reported over 1k likely malicious extensions which may overestimate their true number. Based on this finding (further supported with other experiments and realistic analyses), we elucidate, for the first time, a strong *concept drift* effect on browser extensions. We also provide factual evidence that commercial detectors (e.g., VirusTotal) work poorly to detect known malicious extensions. Altogether, our results highlight the fact that detecting malicious browser extensions is a fundamentally hard problem which has not (yet) received an adequate degree of attention. This requires additional work both by the research community and by Google itself—potentially by revising their approaches. In the meantime, we informed Google of our discoveries, and we release our artifacts.

CCS Concepts: • **Security and privacy → Web application security**; **Social aspects of security and privacy**; • **Computing methodologies →** *Machine learning*; • **Information systems → World Wide Web**.

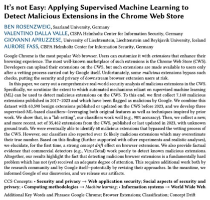
Additional Key Words and Phrases: Google Chrome; Browser Extensions; Classification; Concept Drift

> It's not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and Aurore Fass

Detecting Malicious Extensions — Lab Setting

Rosenzweig et al. TWEB 2026



It's not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store

BEN ROSENZWEIG, Saarland University, Germany
VALENTINO DALLA VALLE, CISPA Helmholtz Center for Information Security, Germany
GIOVANNI APRUZZESE, University of Liechtenstein, Liechtenstein and Reykjavik University, Iceland
AURORE FASS, CISPA Helmholtz Center for Information Security, Germany

Google Chrome is the most popular Web browser. Users can customize it with *extensions* that enhance their browsing experience. The most well-known marketplace of such extensions is the Chrome Web Store (CWS). Developers can upload their extensions on the CWS, but such extensions are made available to users only after a vetting process carried out by Google itself. Unfortunately, some *malicious* extensions bypass such checks, putting the security and privacy of downstream browser extension users at risk.

In this paper, we carry out a comprehensive real-world security analysis of malicious extensions in the CWS. Specifically, we scrutinize the extent to which automated mechanisms reliant on supervised machine learning (ML) can be used to detect malicious extensions on the CWS. To this end, we first collect 7,140 malicious extensions published in 2017–2023 and which have been flagged as malicious by Google. We combine this dataset with 63,598 benign extensions published or updated on the CWS before 2023, and we develop three supervised-ML-based classifiers—leveraging both original features as well as techniques inspired by prior work. We show that, in a "lab setting", our classifiers work well (e.g., 98% accuracy). Then, we collect a new, and more recent, set of 35,462 extensions from the CWS, published or last updated in 2023, with unknown ground truth. We were eventually able to identify 68 malicious extensions that bypassed the vetting process of the CWS. However, our classifiers also reported over 1k likely malicious extensions which may overestimate their true number. Based on this finding (further supported with other experiments and realistic analyses), we elucidate, for the first time, a strong *concept drift* effect on browser extensions. We also provide factual evidence that commercial detectors (e.g., VirusTotal) work poorly to detect known malicious extensions. Altogether, our results highlight the fact that detecting malicious browser extensions is a fundamentally hard problem which has not (yet) received an adequate degree of attention. This requires additional work both by the research community and by Google itself—potentially by revising their approaches. In the meantime, we informed Google of our discoveries, and we release our artifacts.

CCS Concepts: • **Security and privacy → Web application security**; **Social aspects of security and privacy**; • **Computing methodologies →** *Machine learning*; • **Information systems → World Wide Web**.

Additional Key Words and Phrases: Google Chrome; Browser Extensions; Classification; Concept Drift

> It's not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and <u>Aurore Fass</u>

DATASET

FEATURES

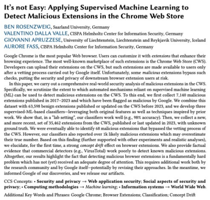
DETECTORS

Predictions

80% train | 20% test

Detecting Malicious Extensions — Lab Setting

Rosenzweig et al. TWEB 2026



It's not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store

BEN ROSENZWEIG, Saarland University, Germany
VALENTINO DALLA VALLE, CISPA Helmholtz Center for Information Security, Germany
GIOVANNI APRUZZESE, University of Liechtenstein, Liechtenstein and Reykjavik University, Iceland
AURORE FASS, CISPA Helmholtz Center for Information Security, Germany

Google Chrome is the most popular Web browser. Users can customize it with *extensions* that enhance their browsing experience. The most well-known marketplace of such extensions is the Chrome Web Store (CWS). Developers can upload their extensions on the CWS, but such extensions are made available to users only after a vetting process carried out by Google itself. Unfortunately, some *malicious* extensions bypass such checks, putting the security and privacy of downstream browser extension users at risk.

In this paper, we carry out a comprehensive real-world security analysis of malicious extensions in the CWS. Specifically, we scrutinize the extent to which automated mechanisms reliant on supervised machine learning (ML) can be used to detect malicious extensions on the CWS. To this end, we first collect 7,140 malicious extensions published in 2017–2023 and which have been flagged as malicious by Google. We combine this dataset with 63,598 benign extensions published or updated on the CWS before 2023, and we develop three supervised-ML-based classifiers—leveraging both original features as well as techniques inspired by prior work. We show that, in a "lab setting", our classifiers work well (e.g., 98% accuracy). Then, we collect a new, and more recent, set of 35,462 extensions from the CWS, published or last updated in 2023, with unknown ground truth. We were eventually able to identify 68 malicious extensions that bypassed the vetting process of the CWS. However, our classifiers also reported over 1k likely malicious extensions which may overestimate their true number. Based on this finding (further supported with other experiments and realistic analyses), we elucidate, for the first time, a strong *concept drift* effect on browser extensions. We also provide factual evidence that commercial detectors (e.g., VirusTotal) work poorly to detect known malicious extensions. Altogether, our results highlight the fact that detecting malicious browser extensions is a fundamentally hard problem which has not (yet) received an adequate degree of attention. This requires additional work both by the research community and by Google itself—potentially by revising their approaches. In the meantime, we informed Google of our discoveries, and we release our artifacts.

CCS Concepts: • **Security and privacy → Web application security**; **Social aspects of security and privacy**; • **Computing methodologies →** *Machine learning*; • **Information systems → World Wide Web**.

Additional Key Words and Phrases: Google Chrome; Browser Extensions; Classification; Concept Drift

> It's not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and <u>Aurore Fass</u>

DATASET

7,140 malicious extensions

Last update 2017—22

63,598 benign extensions

Last update 2012—22

FEATURES

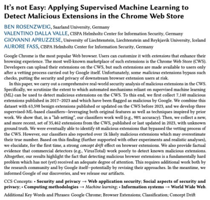
DETECTORS

80% train | 20% test

Predictions

Detecting Malicious Extensions — Lab Setting

Rosenzweig et al. TWEB 2026


It's not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store
BEN ROSENZWEIG, Saarland University, Germany
VALENTINO DALLA VALLE, CISPA Helmholtz Center for Information Security, Germany
GIOVANNI APRUZZESE, University of Liechtenstein, Liechtenstein and Reykjavik University, Iceland
AURORE FASS, CISPA Helmholtz Center for Information Security, Germany

> It's not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and Aurore Fass

DATASET

7,140 malicious extensions

Last update 2017—22

63,598 benign extensions

Last update 2012—22

FEATURES

Source code [A]

Metadata

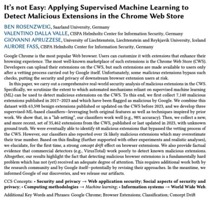
[A] Fass et al.; *ACSAC* 2018

DETECTORS

80% train | 20% test

→ *Predictions*

Detecting Malicious Extensions — Lab Setting


It’s not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store

BEN ROSENZWEIG, Saarland University, Germany
VALENTINO DALLA VALLE, CISPA Helmholtz Center for Information Security, Germany
GIOVANNI APRUZZESE, University of Liechtenstein, Liechtenstein and Reykjavik University, Iceland
AURORE FASS, CISPA Helmholtz Center for Information Security, Germany

Google Chrome is the most popular Web browser. Users can customize it with *extensions* that enhance their browsing experience. The most well-known marketplace of such extensions is the Chrome Web Store (CWS). Developers can upload their extensions on the CWS, but such extensions are made available to users only after a vetting process carried out by Google itself. Unfortunately, some *malicious* extensions bypass such checks, putting the security and privacy of downstream browser extension users at risk.

In this paper, we carry out a comprehensive real-world security analysis of malicious extensions in the CWS. Specifically, we scrutinize the extent to which automated mechanisms reliant on supervised machine learning (ML) can be used to detect malicious extensions on the CWS. To this end, we first collect 7,140 malicious extensions published in 2017–2023 and which have been flagged as malicious by Google. We combine this dataset with 63,598 benign extensions published or updated on the CWS before 2023, and we develop three supervised-ML-based classifiers—leveraging both original features as well as techniques inspired by prior work. We show that, in a “lab setting”, our classifiers work well (e.g., 98% accuracy). Then, we collect a new, and more recent, set of 35,462 extensions from the CWS, published or last updated in 2023, with unknown ground truth. We were eventually able to identify 68 malicious extensions that bypassed the vetting process of the CWS. However, our classifiers also reported over 1k likely malicious extensions which may overestimate their true number. Based on this finding (further supported with other experiments and realistic analyses), we elucidate, for the first time, a strong *concept drift* effect on browser extensions. We also provide factual evidence that commercial detectors (e.g., VirusTotal) work poorly to detect known malicious extensions. Altogether, our results highlight the fact that detecting malicious browser extensions is a fundamentally hard problem which has not (yet) received an adequate degree of attention. This requires additional work both by the research community and by Google itself—potentially by revising their approaches. In the meantime, we informed Google of our discoveries, and we release our artifacts.

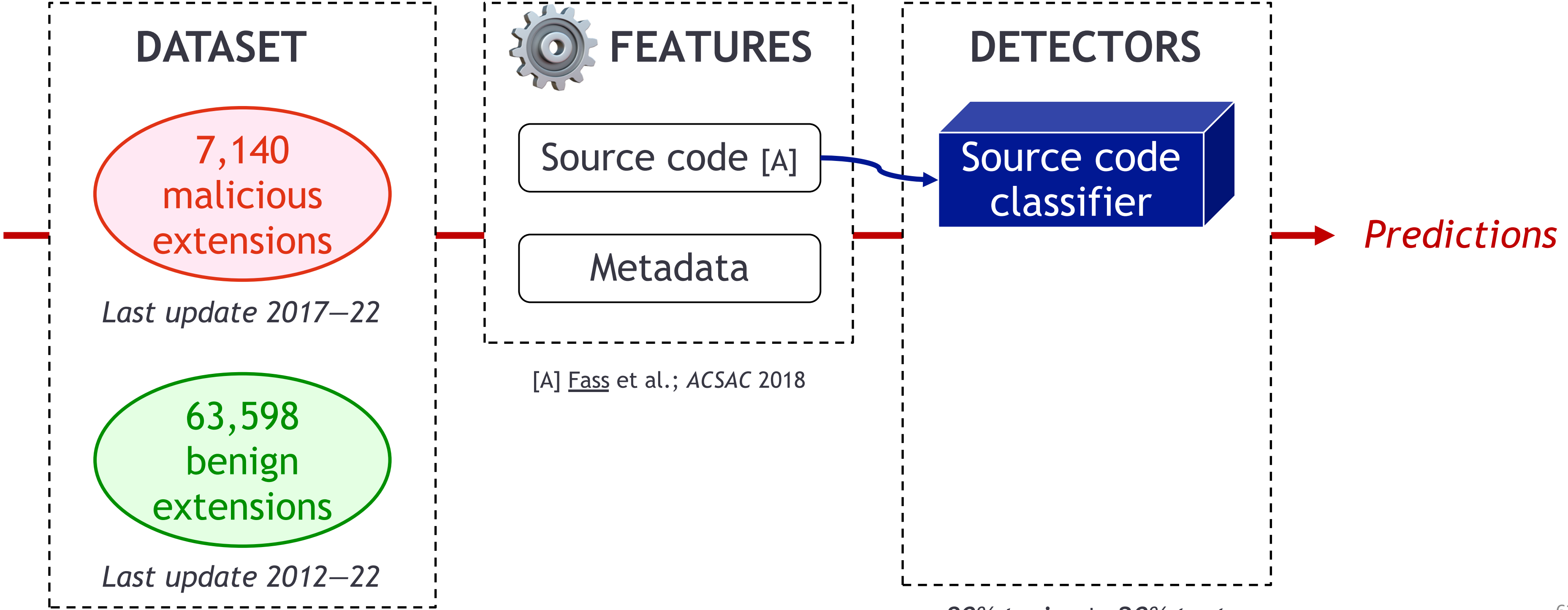
CCS Concepts: • **Security and privacy → Web application security**; **Social aspects of security and privacy**; • **Computing methodologies →** *Machine learning*; • **Information systems → World Wide Web**.

Additional Key Words and Phrases: Google Chrome; Browser Extensions; Classification; Concept Drift

Rosenzweig et al. TWEB 2026

> It’s not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and Aurore Fass

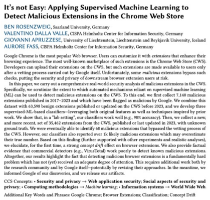


[A] Fass et al.; *ACSAC* 2018

80% train | 20% test

Detecting Malicious Extensions — Lab Setting

Rosenzweig et al. TWEB 2026



It's not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store

BEN ROSENZWEIG, Saarland University, Germany
VALENTINO DALLA VALLE, CISPA Helmholtz Center for Information Security, Germany
GIOVANNI APRUZZESE, University of Liechtenstein, Liechtenstein and Reykjavik University, Iceland
AURORE FASS, CISPA Helmholtz Center for Information Security, Germany

Google Chrome is the most popular Web browser. Users can customize it with *extensions* that enhance their browsing experience. The most well-known marketplace of such extensions is the Chrome Web Store (CWS). Developers can upload their extensions on the CWS, but such extensions are made available to users only after a vetting process carried out by Google itself. Unfortunately, some *malicious* extensions bypass such checks, putting the security and privacy of downstream browser extension users at risk.

In this paper, we carry out a comprehensive real-world security analysis of malicious extensions in the CWS. Specifically, we scrutinize the extent to which automated mechanisms reliant on supervised machine learning (ML) can be used to detect malicious extensions on the CWS. To this end, we first collect 7,140 malicious extensions published in 2017–2023 and which have been flagged as malicious by Google. We combine this dataset with 63,598 benign extensions published or updated on the CWS before 2023, and we develop three supervised-ML-based classifiers—leveraging both original features as well as techniques inspired by prior work. We show that, in a "lab setting", our classifiers work well (e.g., 98% accuracy). Then, we collect a new, and more recent, set of 35,462 extensions from the CWS, published or last updated in 2023, with unknown ground truth. We were eventually able to identify 68 malicious extensions that bypassed the vetting process of the CWS. However, our classifiers also reported over 1k likely malicious extensions which may overestimate their true number. Based on this finding (further supported with other experiments and realistic analyses), we elucidate, for the first time, a strong *concept drift* effect on browser extensions. We also provide factual evidence that commercial detectors (e.g., VirusTotal) work poorly to detect known malicious extensions. Altogether, our results highlight the fact that detecting malicious browser extensions is a fundamentally hard problem which has not (yet) received an adequate degree of attention. This requires additional work both by the research community and by Google itself—potentially by revising their approaches. In the meantime, we informed Google of our discoveries, and we release our artifacts.

CCS Concepts: • **Security and privacy → Web application security**; **Social aspects of security and privacy**; • **Computing methodologies →** *Machine learning*; • **Information systems → World Wide Web**.

Additional Key Words and Phrases: Google Chrome; Browser Extensions; Classification; Concept Drift

> It's not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and Aurore Fass

DATASET

7,140 malicious extensions

Last update 2017—22

63,598 benign extensions

Last update 2012—22

FEATURES

Source code [A]

Metadata

[A] Fass et al.; *ACSAC* 2018

DETECTORS

Source code classifier

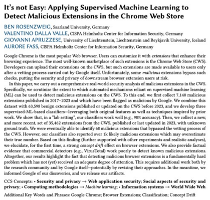
Metadata classifier

Predictions

80% train | 20% test

Detecting Malicious Extensions — Lab Setting

Rosenzweig et al. TWEB 2026



It's not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store

BEN ROSENZWEIG, Saarland University, Germany
VALENTINO DALLA VALLE, CISPA Helmholtz Center for Information Security, Germany
GIOVANNI APRUZZESE, University of Liechtenstein, Liechtenstein and Reykjavik University, Iceland
AURORE FASS, CISPA Helmholtz Center for Information Security, Germany

Google Chrome is the most popular Web browser. Users can customize it with *extensions* that enhance their browsing experience. The most well-known marketplace of such extensions is the Chrome Web Store (CWS). Developers can upload their extensions on the CWS, but such extensions are made available to users only after a vetting process carried out by Google itself. Unfortunately, some *malicious* extensions bypass such checks, putting the security and privacy of downstream browser extension users at risk.

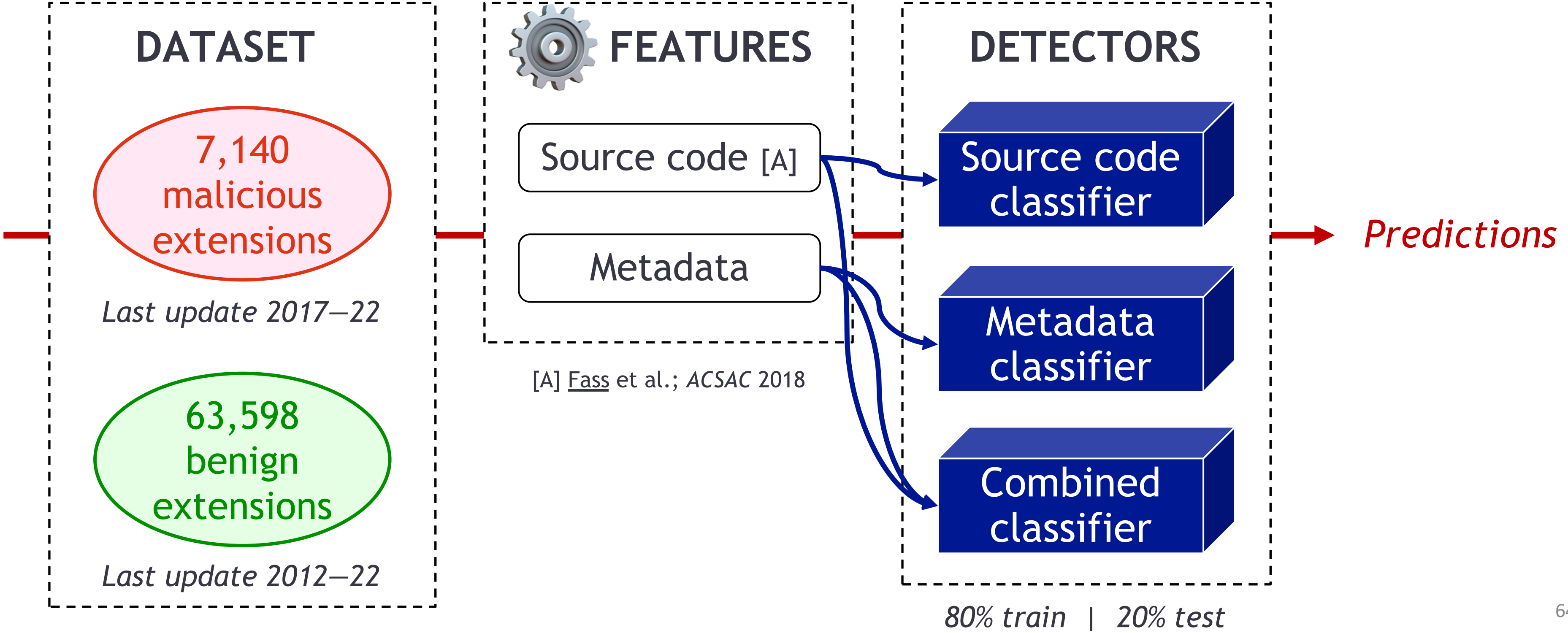
In this paper, we carry out a comprehensive real-world security analysis of malicious extensions in the CWS. Specifically, we scrutinize the extent to which automated mechanisms reliant on supervised machine learning (ML) can be used to detect malicious extensions on the CWS. To this end, we first collect 7,140 malicious extensions published in 2017–2023 and which have been flagged as malicious by Google. We combine this dataset with 63,598 benign extensions published or updated on the CWS before 2023, and we develop three supervised-ML-based classifiers—leveraging both original features as well as techniques inspired by prior work. We show that, in a "lab setting", our classifiers work well (e.g., 98% accuracy). Then, we collect a new, and more recent, set of 35,462 extensions from the CWS, published or last updated in 2023, with unknown ground truth. We were eventually able to identify 68 malicious extensions that bypassed the vetting process of the CWS. However, our classifiers also reported over 1k likely malicious extensions which may overestimate their true number. Based on this finding (further supported with other experiments and realistic analyses), we elucidate, for the first time, a strong *concept drift* effect on browser extensions. We also provide factual evidence that commercial detectors (e.g., VirusTotal) work poorly to detect known malicious extensions. Altogether, our results highlight the fact that detecting malicious browser extensions is a fundamentally hard problem which has not (yet) received an adequate degree of attention. This requires additional work both by the research community and by Google itself—potentially by revising their approaches. In the meantime, we informed Google of our discoveries, and we release our artifacts.

CCS Concepts: • **Security and privacy → Web application security**; **Social aspects of security and privacy**; • **Computing methodologies →** *Machine learning*; • **Information systems → World Wide Web**.

Additional Key Words and Phrases: Google Chrome; Browser Extensions; Classification; Concept Drift

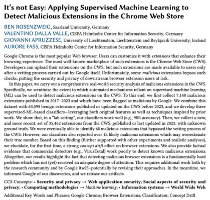
> It's not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and Aurore Fass



Detecting Malicious Extensions — Lab Setting

Rosenzweig et al. TWEB 2026



> It's not Easy: Applying Supervised ML to Detect Malicious Extensions in the CWS

In *ACM TWEB* 2026. Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and Aurore Fass

DATASET

7,140 malicious extensions

Last update 2017—22

63,598 benign extensions

Last update 2012—22

FEATURES

Source code [A]

Metadata

[A] Fass et al.; *ACSAC* 2018

DETECTORS

Source code classifier

Metadata classifier

Combined classifier

Predictions

Acc: 98.37%

TNR: 98.71%

TPR: 95.39%

80% train | 20% test

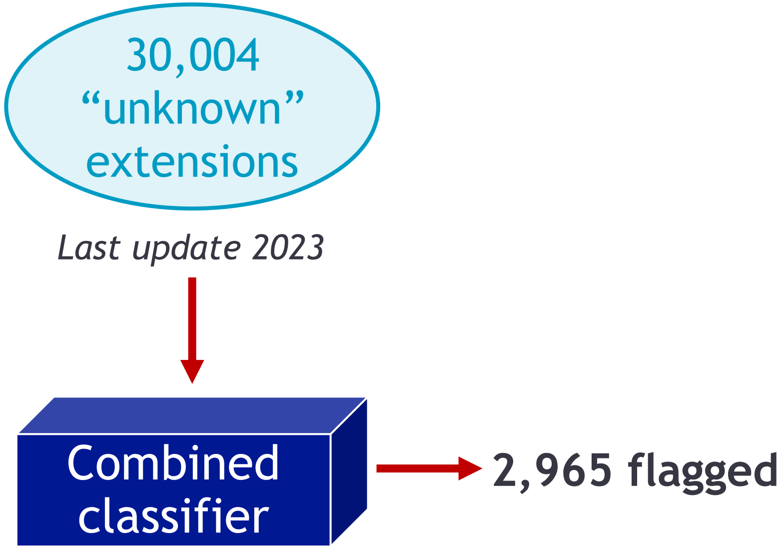
Detecting Malicious Extensions — Real World

30,004
“unknown”
extensions

Last update 2023

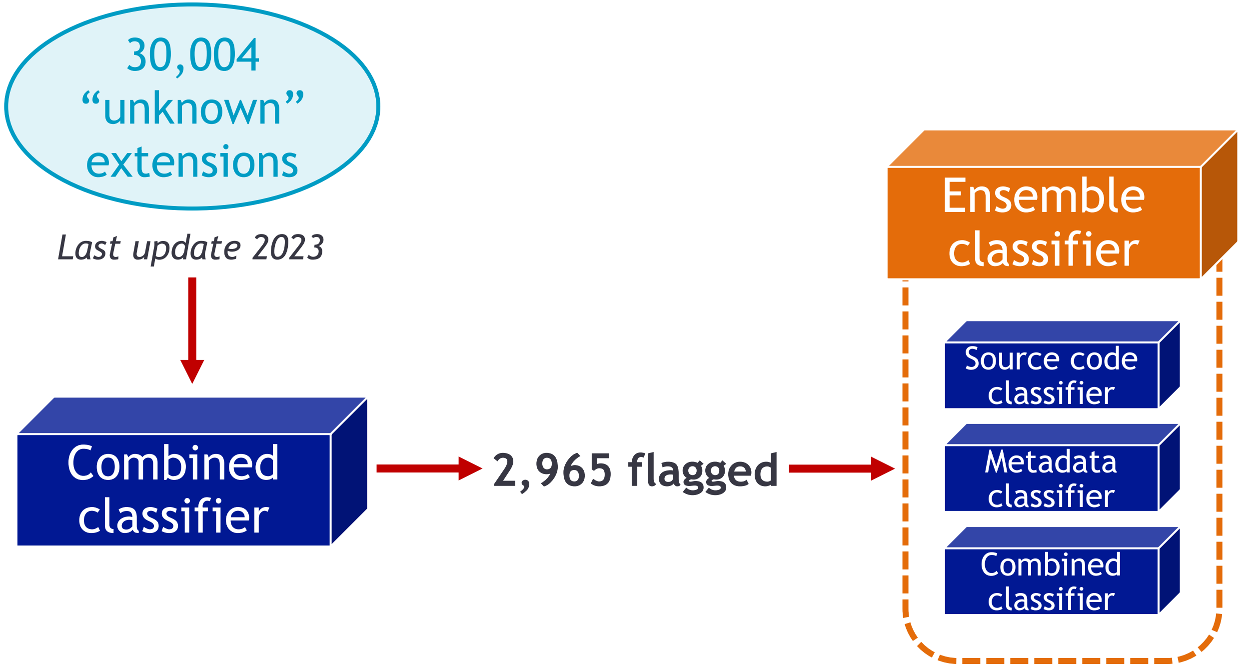
[3] Rosenzweig et al.; *TWEB* 2026

Detecting Malicious Extensions – Real World



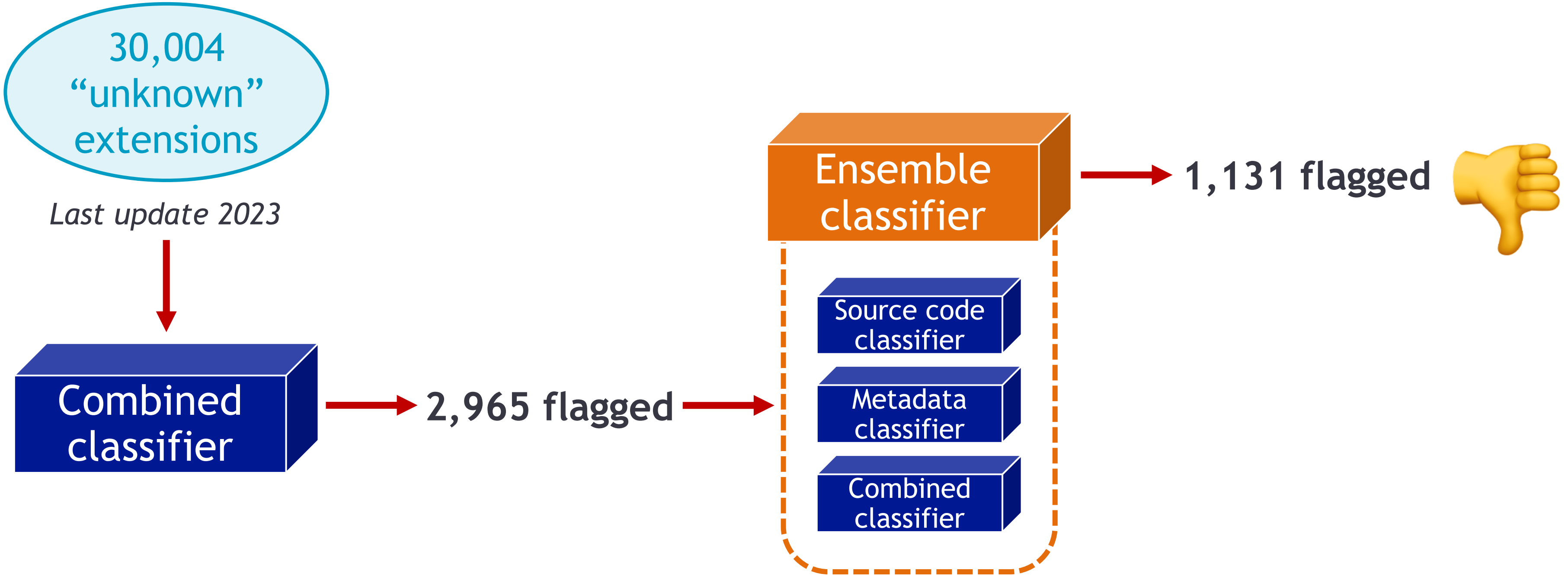
[3] Rosenzweig et al.; *TWEB* 2026

Detecting Malicious Extensions — Real World

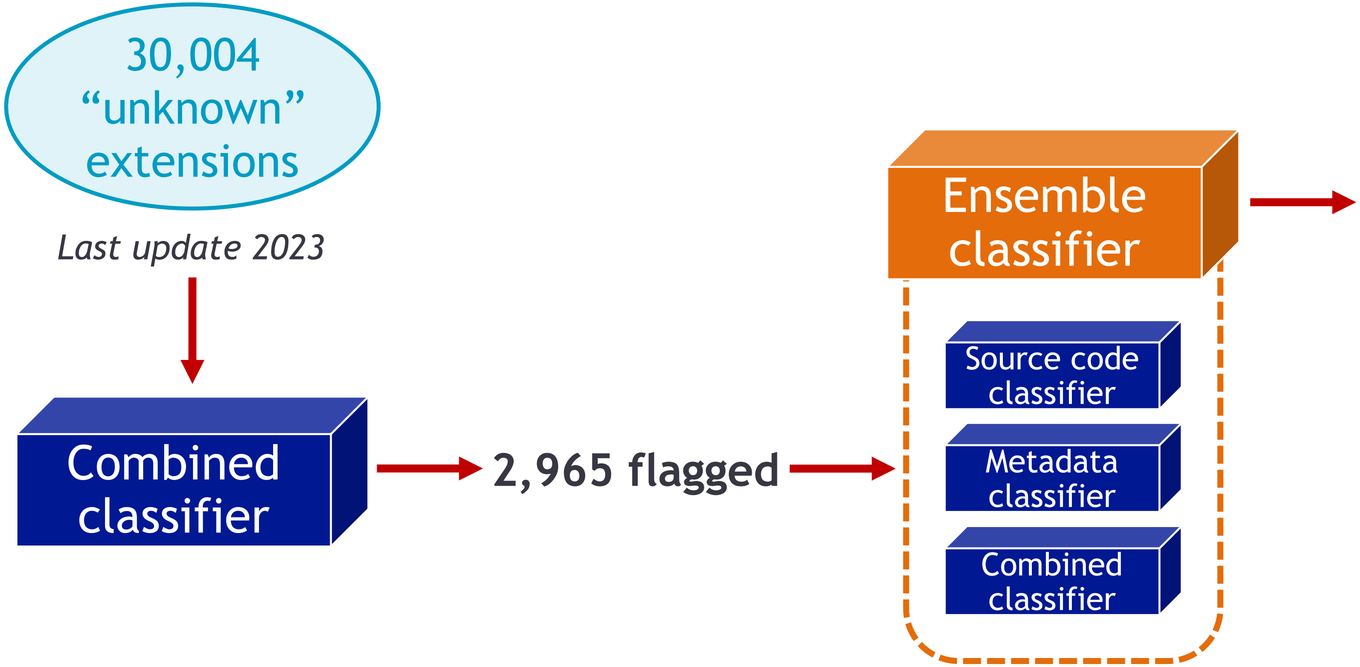


[3] Rosenzweig et al.; *TWEB* 2026

Detecting Malicious Extensions – Real World



Detecting Malicious Extensions – Real World



1,131 flagged 

- **28 malicious** extensions removed by Google, **2M+ users**

- **40 malicious** ext. (after clustering + manual analysis on 20% flagged), **11M users**

Total: 68 malicious extensions, 13M users

[3] Rosenzweig et al.; *TWEB* 2026

Detecting Malicious Extensions – Real World

30,004 "unknown" extensions

Last update 2023

Combined classifier → **2,965 flagged** →

Ensemble classifier

- Source code classifier
- Metadata classifier
- Combined classifier

→ **1,131 flagged**

- **28 malicious** extensions removed by Google, **2M+ users**
- **40 malicious** ext. (after clustering + manual analysis on 20% flagged), **11M users**

Total: 68 malicious extensions, 13M users

May 2024: **disclosure** of the 40 extensions to Google, **no response**

As of Sept. 2025: 17 / 40 malicious extensions were taken down

[3] Rosenzweig et al.; *TWEB* 2026

Detecting Malicious Extensions – Lab Setting vs. Real World

- **Lab setting:** 98.37% accuracy
- **Real world:** 1,131 / 30,004 extensions flagged | 68 verified malicious extensions

???

[3] Rosenzweig et al.; *TWEB* 2026

Detecting Malicious Extensions – Lab Setting vs. Real World

- **Lab setting:** 98.37% accuracy

- **Real world:** 1,131 / 30,004 extensions flagged | 68 verified malicious extensions

???

- **Concept drift**
  - **Intrinsic evolution** of extensions, hard for supervised ML tools to keep a (high) accuracy over time
  - **How** to validate concept drift? → see experiments in the paper
  - **Why** are extensions affected? → see paper (study feature importance)

[3] Rosenzweig et al.; *TWEB* 2026

Detecting Malicious Extensions — Takeaways

- ML evaluations can be **misleading**: detectors performing well in a lab setting are **no guarantee of practical applicability** in the real world

- Detectors need to be **retrained regularly**

- Investigate **active learning** as a **mitigation to concept drift**
  - E.g., *uncertainty sampling*: monthly retraining of the detector on X labeled extensions where the detector is the most uncertain

[3] Rosenzweig et al.; *TWEB* 2026

Outline

- Background: Browser Extensions
- Investigating Security-Noteworthy Extensions (SNE)
- Detecting Vulnerable Extensions
  - Threat model and automated tool (DOUBLEX)
  - Case studies, results, and potential defense strategies
- Detecting Malicious Extensions
  - Lab setting vs. real world
- Detecting Fingerprintable Extensions
  - Presentation of 3 fingerprinting vectors, results, and potential mitigations

Hsu et al. AsiaCCS 2024

Fass et al. CCS 2021

Rosenzweig et al. TWEB 2026

Agarwal et al. CCS 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

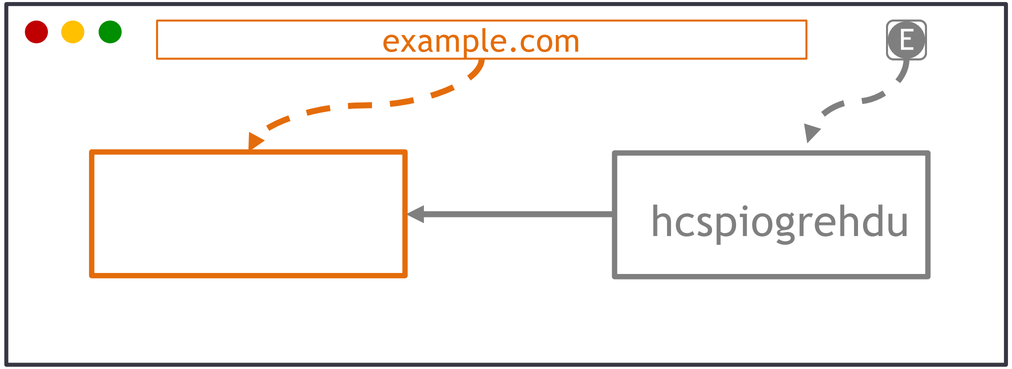
- 1) Extensions send **PostMessage** to web pages

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages...**

- 1) Extensions send **PostMessage** to web pages



Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)

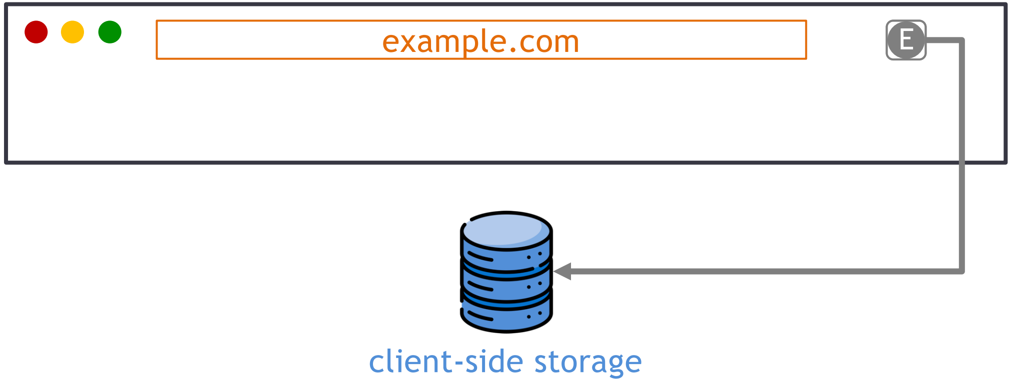
[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)

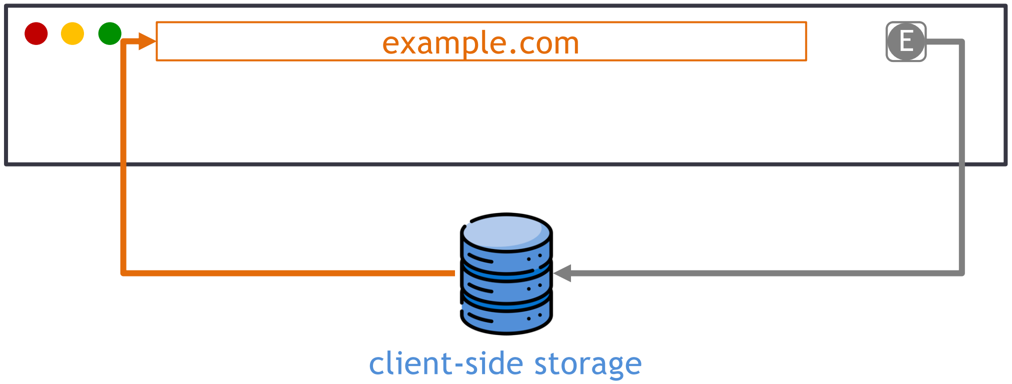


Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)

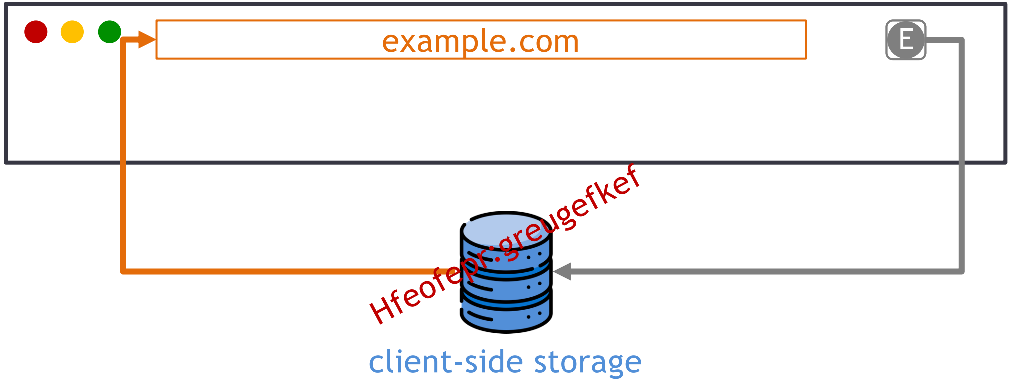


Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)



[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**
  (e.g., cookies, local/session storage, IndexedDB)
- 3) Extensions **inject JavaScript** code directly into web pages

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)

- 3) Extensions **inject JavaScript** code directly into web pages

  → registering global variables

  → invocation of global APIs and properties

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)
- 3) Extensions **inject JavaScript** code directly into web pages

  → registering global variables

  → invocation of global APIs and properties

… which leaves traces → **observable side effects**, can be seen by a “malicious” site

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)

- 3) Extensions **inject JavaScript** code directly into web pages

  → registering global variables

  → invocation of global APIs and properties

… which leaves traces → **observable side effects**, can be seen by a "malicious" site

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**
  (e.g., cookies, local/session storage, IndexedDB)
- 3) Extensions **inject JavaScript** code directly into web pages
  → registering global variables
  → invocation of global APIs and properties

… which leaves traces → **observable side effects**, can be seen by a "malicious" site

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages…**

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)
- 3) Extensions **inject JavaScript** code directly into web pages

  → registering global variables

  → invocation of global APIs and properties

… which leaves traces → **observable side effects**, can be seen by a "malicious" site

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Browser extensions can **interact with web pages**…

Why is this bad?

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, indexDB)
- 3) Extensions **inject JavaScript** code directly into web pages

  → registering global variables

  → invocation of global APIs and properties

… which leaves traces → **observable side effects**, can be seen by a "malicious" site

[4] Agarwal, Fass, and Stock; *CCS* 2024

Browser Extension Fingerprinting

Why is this bad?

- “Malicious” websites can **track users** by fingerprinting their extensions
- Attackers can know if a user has a vulnerable extension installed
- Extensions can **reveal personal user information**, e.g., geolocation, ethnicity, social/personal interests, medical issues, religion, etc. [Karami; NDSS’19]

[4] Agarwal, Fass, and Stock; *CCS* 2024

Detecting Fingerprintable Extensions

How many extensions can be ***uniquely fingerprinted*** *through these observable side effects?*

Detecting Fingerprintable Extensions

*How many extensions can be **uniquely fingerprinted***

through these observable side effects?

Agarwal et al.
CCS 2024


Peeking through the `window`: Fingerprinting Browser Extensions through Page-Visible Execution Traces and Interactions

Shubham Agarwal, CISPA Helmholtz Center for Information Security, shubham.agarwal@cispa.de
Aurore Fass, CISPA Helmholtz Center for Information Security, fass@cispa.de
Ben Stock, CISPA Helmholtz Center for Information Security, stock@cispa.de

> Peeking through the `window`: Fingerprinting Browser Extensions through Page-Visible Execution Traces and Interactions

In *ACM CCS* 2024. Shubham Agarwal, Aurore Fass, and Ben Stock

Detecting Fingerprintable Extensions with Raider

Analyzed 38k Chrome extensions from 2024 with Raider

- **2,747 fingerprintable Chrome extensions** (lower bound)
- Impacting **169M users**

- **Notified 1,967 developers** about their fingerprintable extension(s)
  - Only 30 (!) replied
  - Of those, only 16 positively acknowledged the issues
    - But: they heavily rely on our fingerprinting vectors (e.g., script injection or data storage) for their extensions' functionality

- Raider PoC is **available online** Raider-ext/raider

[4] Agarwal, Fass, and Stock; *CCS* 2024

Mitigations (for Developers)

- Global variables: scope appropriately

- Storage: use the chrome.storage API instead

- Global APIs:
  - ensure that browser extension code runs before the attacker code (inject at document_start)
  - ensure that APIs cannot be overwritten (freeze their native definition)

- As a user?
  - **Be careful** which extensions you install
  - **Be aware** that extensions can be dangerous

[4] Agarwal, Fass, and Stock; *CCS* 2024

Summary

Dangerous Browser Extensions

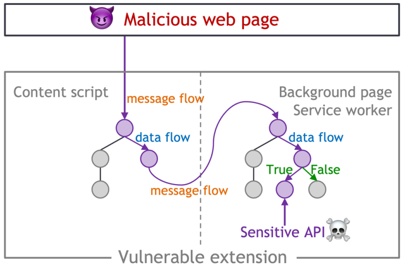
Hsu et al. AsiaCCS 2024

→ **Extensions can put their users' security & privacy at risk:**

- **Contain malware**: designed by malicious actors to harm victims
  - E.g., propagate malware, steal users' credentials, track users
- **Contain vulnerabilities**: designed by well-intentioned developers… but buggy
  - E.g., can lead to user-sensitive data exfiltration
- **Violate the Chrome Web Store policies**
  - E.g., deceive users, promote unlawful activities, lack a privacy policy
- **Be fingerprintable**: can be recognized and uniquely identified
  - E.g., can lead to user tracking or inferring of user personal information

Detecting Vulnerable Extensions with DOUBLEX

Fass et al. CCS 2021



Aurore54F/DoubleX

➢ DOUBLEX **detects suspicious data flows** in browser extensions

184 vulnerable extensions | **Precision: 89%** | **Recall: 93%**

Detecting Malicious Extensions – Lab Setting vs. Real World

- **Lab setting**: 98.37% accuracy
- **Real world**: 1,131 / 30,004 extensions flagged | 68 verified malicious extensions

Rosenzweig et al. TWEB 2026

its-not-easy/tweb25

- **Concept drift**
  - **Intrinsic evolution** of extensions, hard for supervised ML tools to keep a (high) accuracy over time
  - **How to validate concept drift?** → see experiments in the paper
  - **Why are extensions affected?** → see paper (study feature importance)

Detecting Fingerprintable Extensions with Raider

Agarwal et al. CCS 2024

- 1) Extensions send **PostMessage** to web pages
- 2) Extensions store data on the client side through **storage APIs**

  (e.g., cookies, local/session storage, IndexedDB)
- 3) Extensions **inject JavaScript** code directly into web pages

  → registering global variables

  → invocation of global APIs and properties

Raider-ext/raider

➢ Raider **detects 2,747 fingerprintable extensions** | **169M users**

Corresponding Publications

- What is in the Chrome Web Store?

Sheryl Hsu, Manda Tran, and Aurore Fass. In *ACM AsiaCCS* 2024

- DoubleX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale

Aurore Fass, Dolière Francis Somé, Michael Backes, and Ben Stock. In *ACM CCS* 2021

- It’s not Easy: Applying Supervised Machine Learning to Detect Malicious Extensions in the Chrome Web Store

Ben Rosenzweig, Valentino Dalla Valle, Giovanni Apruzzese, and Aurore Fass. In *ACM TWEB* 2026

- Peeking through the window: Fingerprinting Browser Extensions through Page-Visible Execution Traces and Interactions

Shubham Agarwal, Aurore Fass, and Ben Stock. In *ACM CCS* 2024