

Browser Extension (In)Security

Aurore Fass

Tenure-Track Faculty at CISPA – Helmholtz Center for Information Security

Research Work



Malicious JavaScript

Fass et al.
DIMVA
2018



Fass et al.
CCS 2019

Fass et al.
ACSA
2019

Moog et al.
DSN 2021

Benign / Vulnerable JavaScript

Hsu et al.
AsiaCCS
2024



Rosenzweig
et al.
Under sub.



Fass et al.
CCS 2021



Agarwal et al.
CCS 2024

Browser Extensions



Ruth et al.
IMC 2022

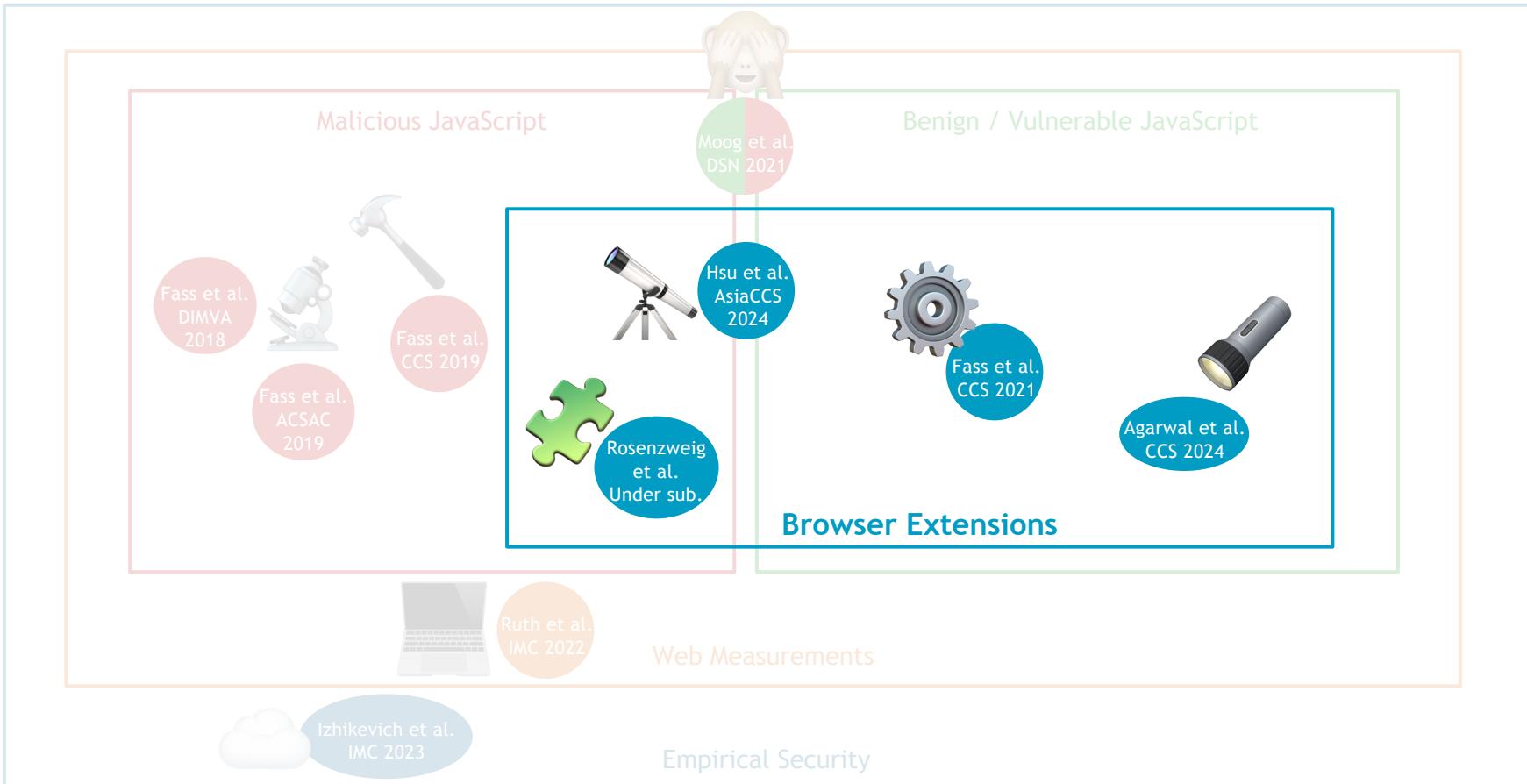
Web Measurements



Izhikevich et al.
IMC 2023

Empirical Security

Research Work



What are Browser Extensions?

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



AdBlock — best ad blocker

Offered by: getadblock.com



Adblock Plus - free ad blocker

Offered by: adblockplus.org



Adobe Acrobat

Offered by: Adobe Inc.



Avast Online Security

Offered by: <https://www.avast.com>



Cisco Webex Extension

Offered by: webex.com



Google Translate

Offered by: translate.google.com



Grammarly for Chrome

Offered by: grammarly.com



Honey

Offered by: <https://www.joinhoney.com>



Pinterest Save Button

Offered by: pinterest.com



Skype

Offered by: www.skype.com



uBlock Origin

Offered by: Raymond Hill (gorhill)



LastPass: Free Password Manager

Offered by: LastPass

- 125k Chrome extensions totaling over 1.6B active users

How Safe are Browser Extensions?

- Browser extensions provide additional functionality...
 - ... so browser extensions need additional & elevated privileges compared to web pages
- Browser extensions are an attractive target for attackers 😈

Security-Noteworthy Extensions (SNE)

→ 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

- Violate the Chrome Web Store **policies**

- E.g., deceive users, promote unlawful activities, lack a privacy policy

- Contain **vulnerabilities**

- Designed by well-intentioned developers... but contain some vulnerabilities
 - E.g., can lead to user-sensitive data exfiltration

Did you know that...

- **350M users** installed **Security-Noteworthy Extensions** in the last 3 years?
- These dangerous extensions stay in the Chrome Web Store *for years*?
- **60%** of extensions have never received a single update?

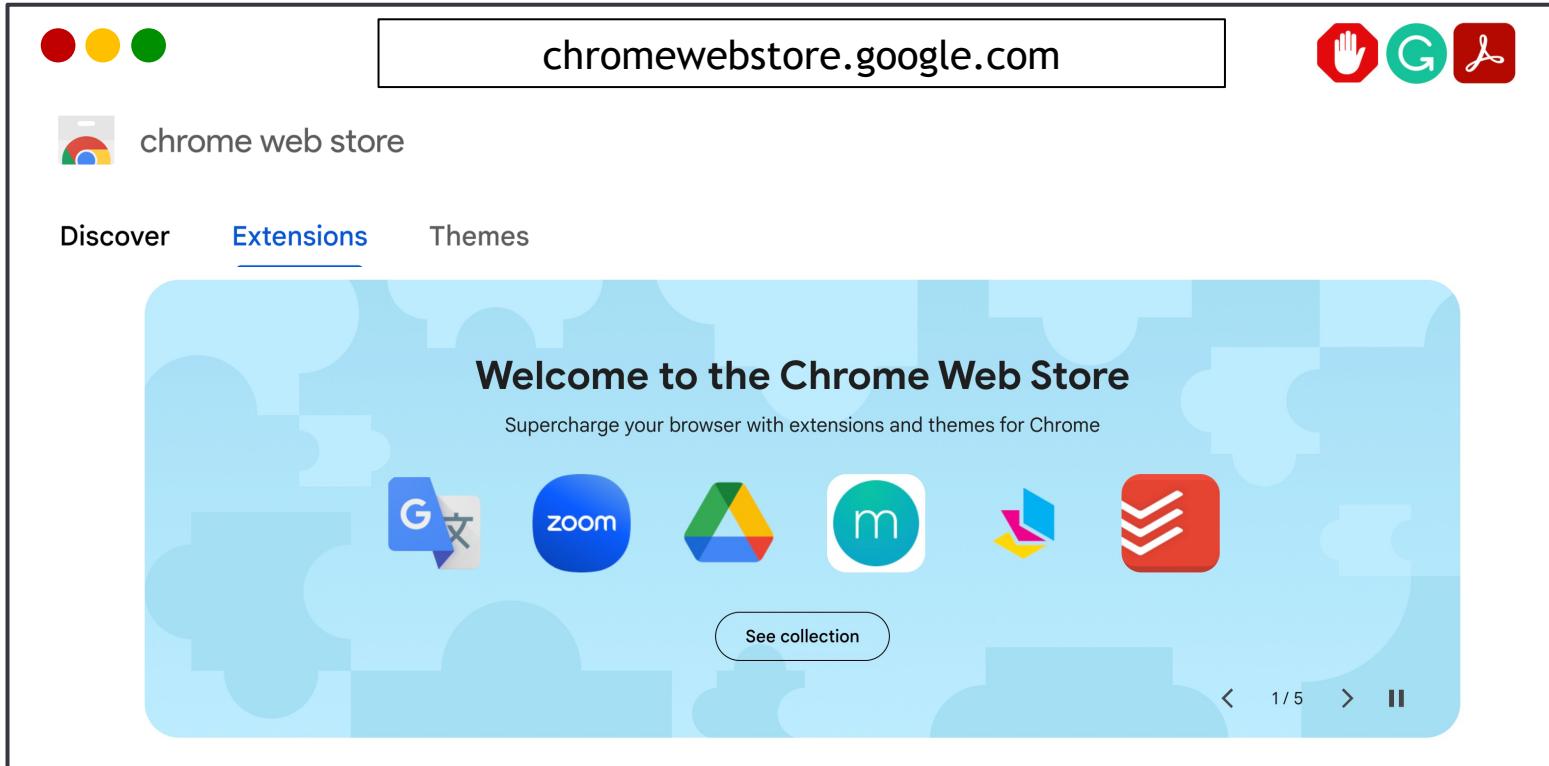


> What is in the Chrome Web Store?

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



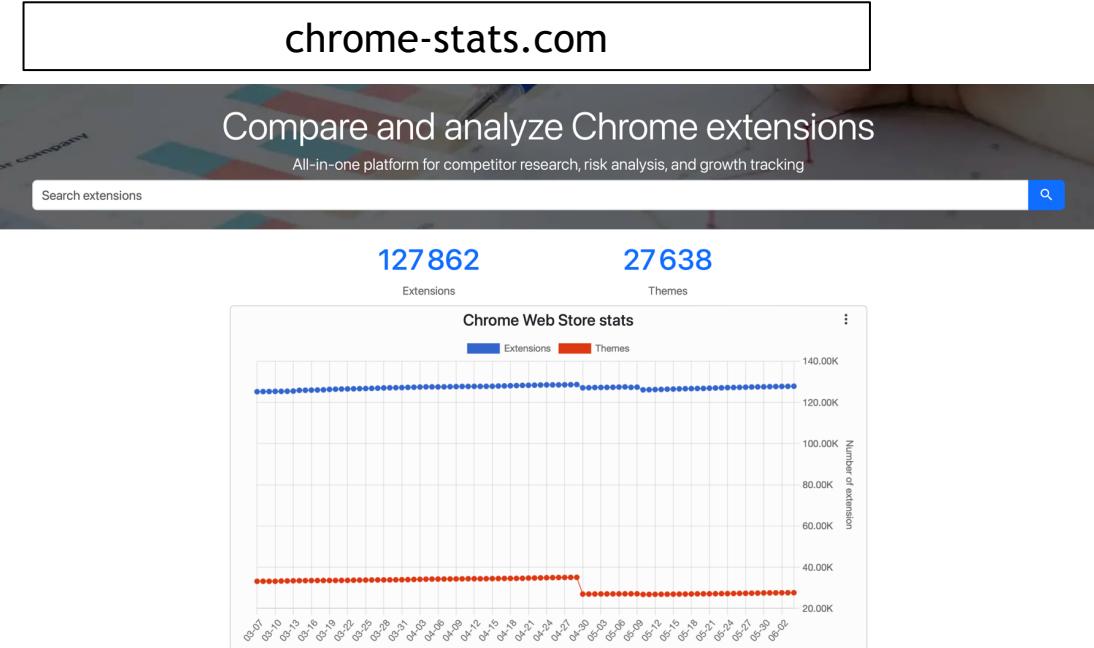
How to Install Extensions or SNE?



How to Install Extensions or SNE?



Browser Extension Collection: Chrome-Stats



The screenshot shows the Chrome-Stats.com homepage. At the top, there's a search bar with the text "chrome-stats.com". Below it, a banner reads "Compare and analyze Chrome extensions" and "All-in-one platform for competitor research, risk analysis, and growth tracking". On the left, a sidebar lists "Recently viewed" extensions and "Stats & analysis tools". The main area features two large numbers: "127862" under "Extensions" and "27638" under "Themes". Below these are two line charts titled "Chrome Web Store stats". The top chart tracks the number of extensions from March 2021 to August 2021, showing a slight dip around April 2021. The bottom chart tracks themes, also from March 2021 to August 2021, showing a similar dip around the same period. A link "Explore more Chrome extension statistics" is at the bottom of the charts.

chrome-stats.com

Compare and analyze Chrome extensions

All-in-one platform for competitor research, risk analysis, and growth tracking

Search extensions

Recently viewed

- Spotify™ & Deezer™ Music Downloader
- GS Auto Clicker:Free Download 2021
- Fraud Risk Scoring
- Autoskip for Youtube
- Maxi Refresher

Stats & analysis tools

- Chrome extension statistics
- Extension explorer
- Keyword explorer
- Publisher explorer
- Advanced search
- Raw data download
- Chrome-Stats extension

127862 27638

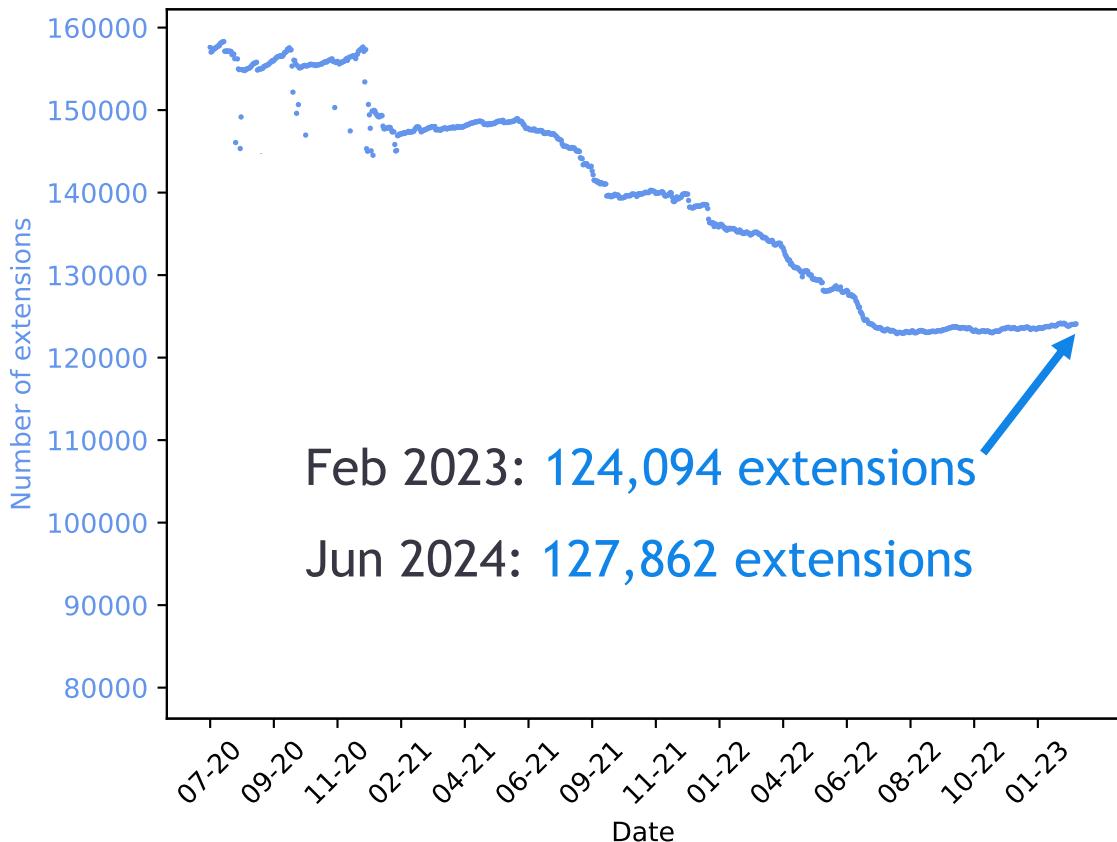
Extensions Themes

Chrome Web Store stats

Explore more Chrome extension statistics

Chrome-Stats makes Chrome extension metrics more accessible to everyone, enable competitive analysis, identify bad actors, and help support the growth of good Chrome extensions.

Number of Extensions in the Chrome Web Store

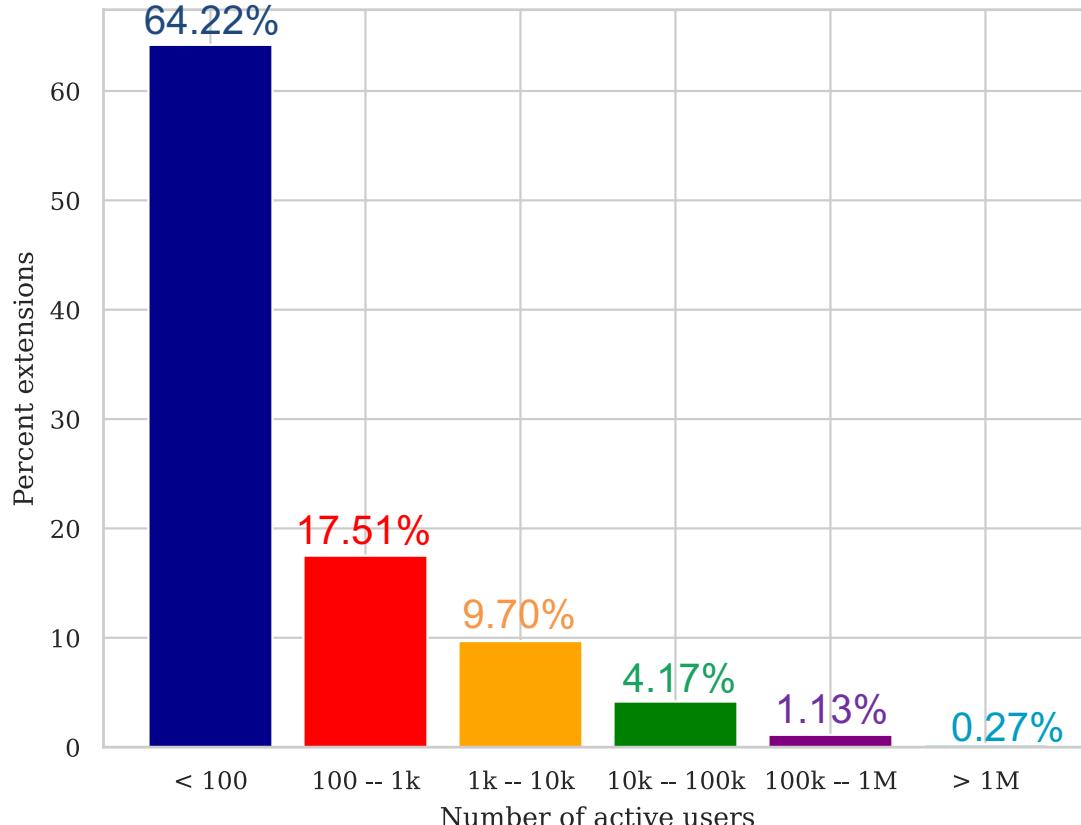


Every month:

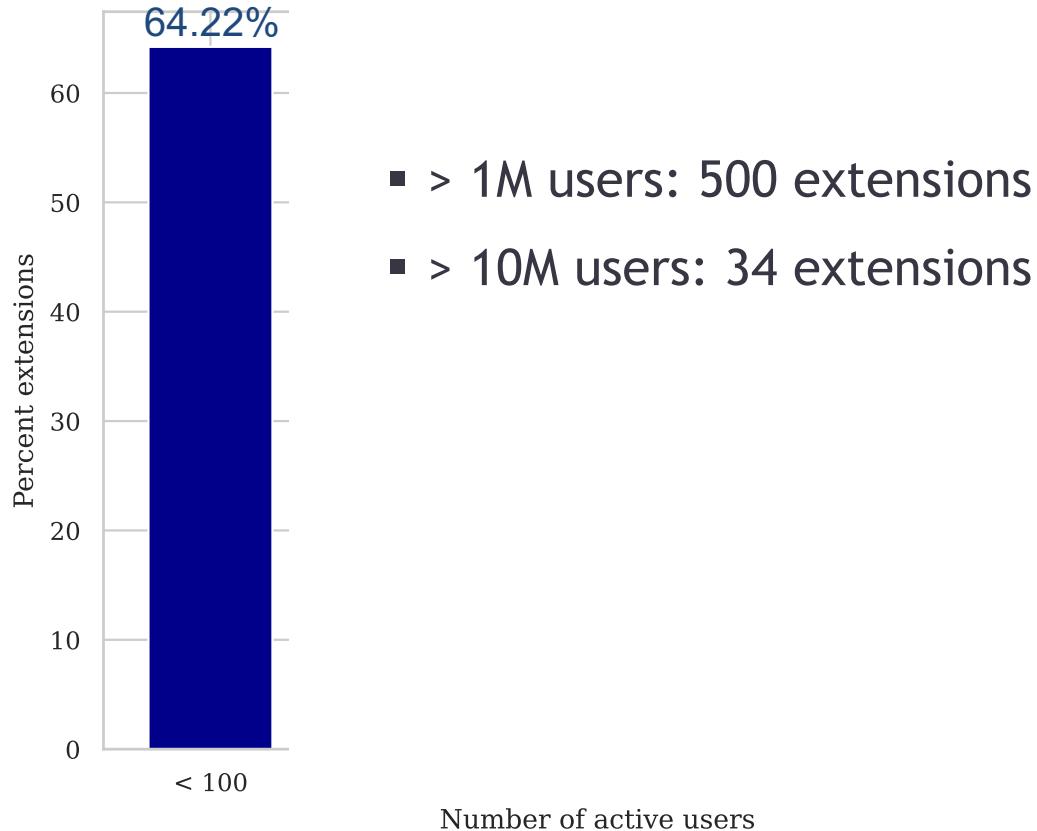
- 3,775 extensions removed
- + 2,687 extensions added

➤ **Analyses** on the CWS
should be **run regularly**

Breakdown of Extension Users



Breakdown of Extension Users



The “**number of users**” on the CWS for a given extension corresponds to:

“the number of Chromes with the extension installed that are active and checking in to [their] update servers over the previous seven days only, not for all time. It is not equal to the sum of historic installs minus the sum of historic uninstalls”

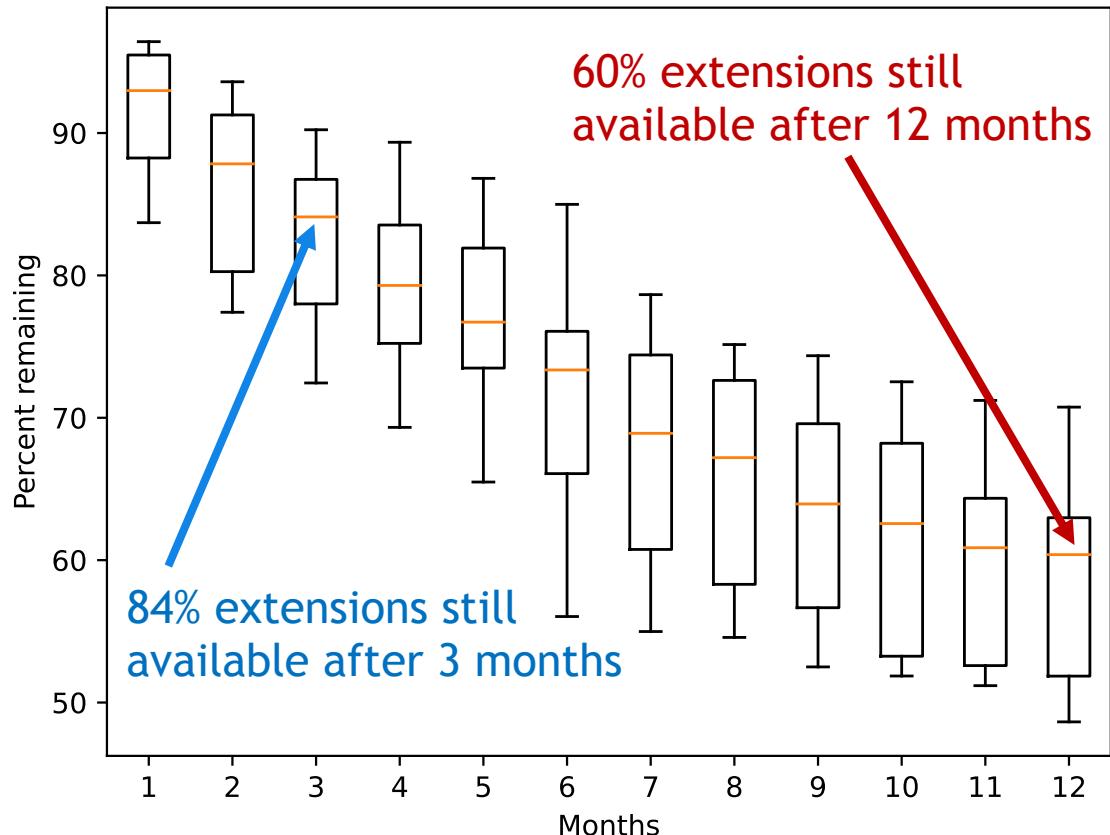
~ Chrome Web Store Developer Support

Life Cycle of Extensions

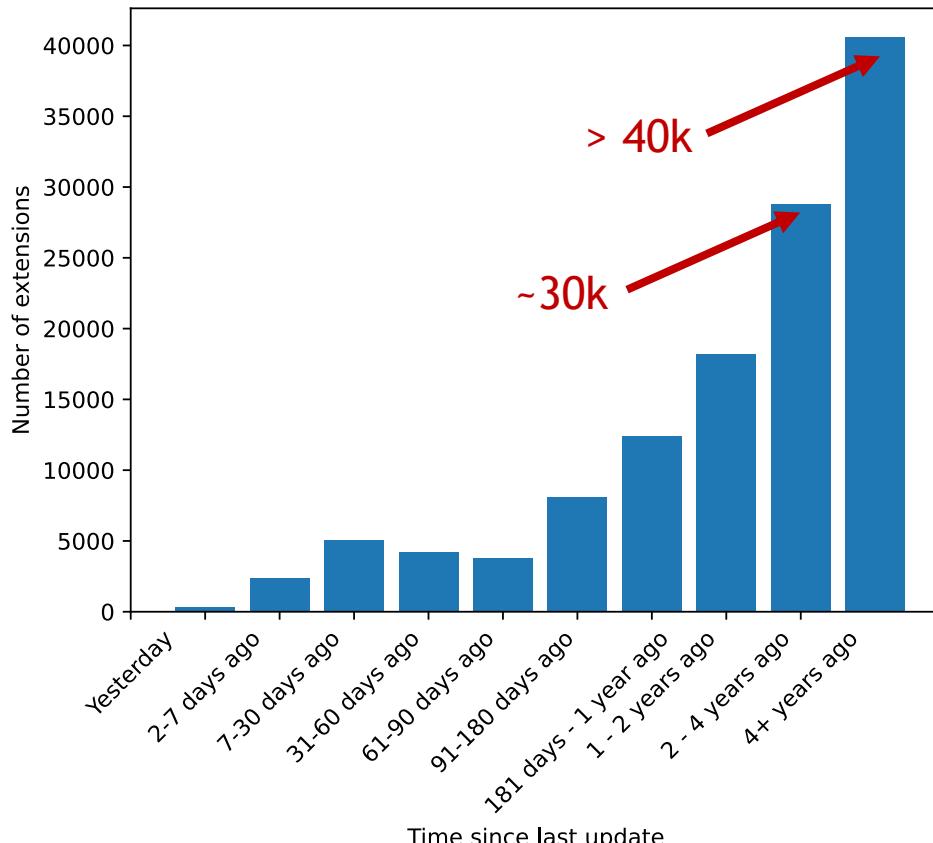
Methodology:

- Collected extensions added to the CWS in Jan–Dec 2021
- Computed the percentage of those extensions still in the CWS 1, 2, ..., 12 months later

- Extensions have a very short life cycle
- Analyses on the CWS should be run regularly



Extension Maintenance and Security



- Critical lack of maintenance in the CWS
- 60% of the extensions have never been updated
- Security & privacy implications

Malicious Extension Collection: Chrome-Stats

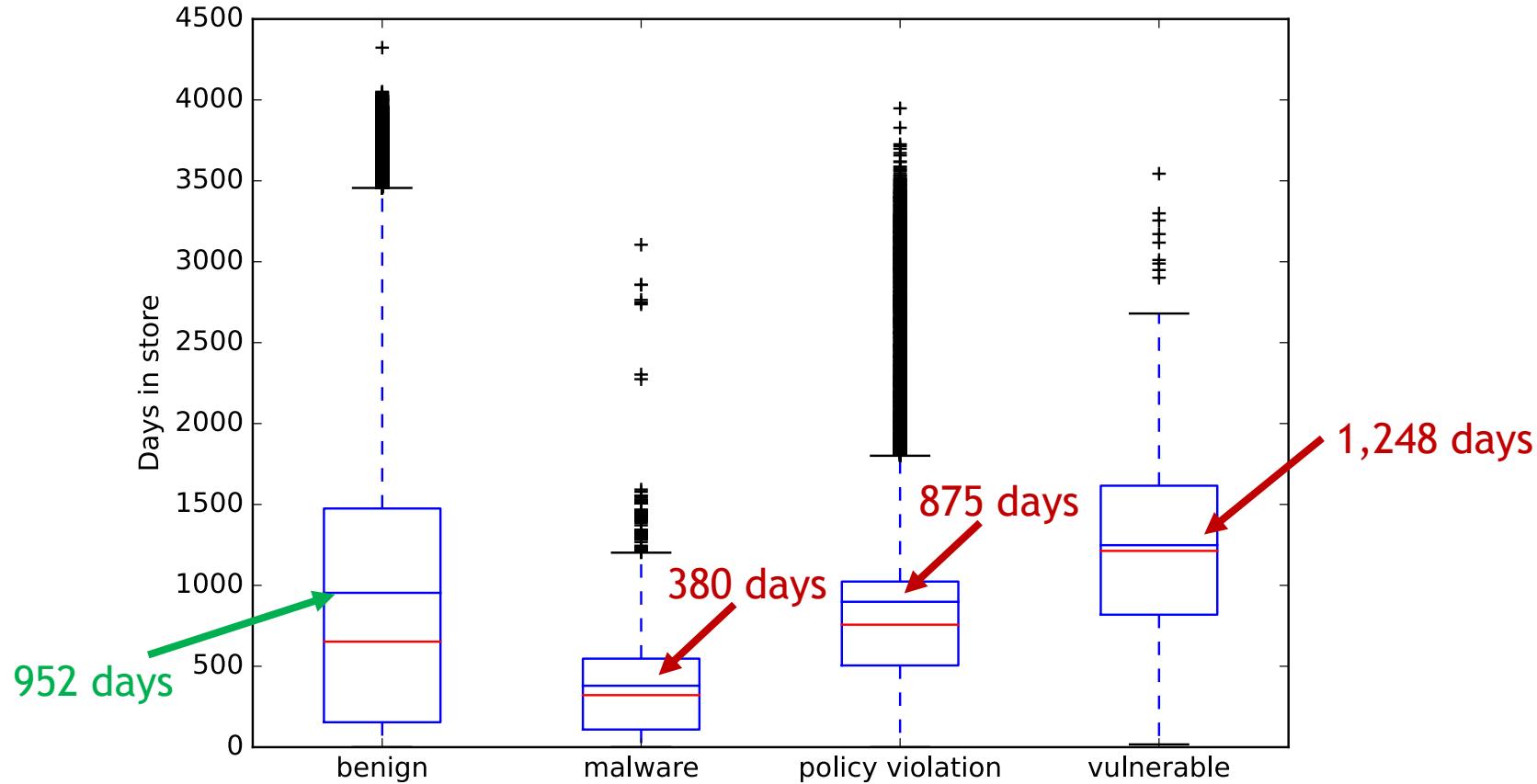
The screenshot shows a web browser window with the address bar containing "chrome-stats.com". The main content area is titled "Advanced search". A red oval highlights the search filters: "obsoleteReason" set to "malware" and the search button. Below the filters, there are buttons for "Export", "Saved query", "Visible columns", and "+ Add condition". The results table has columns: logo, name, userCount, author, ratingValue, ratingCount, obsoleteReason, lastUpdate, and creationDate. The first result is "Video downloader for Instagram™" with 100,000 users, rating 4.27, and last updated 2024-03-07.

logo	name	userCount	author	ratingValue	ratingCount	obsoleteReason	lastUpdate	creationDate
⚠️	Video downloader for Instagram™	100 000	https://instagram-downloader.instvid.site	4.27	30	malware	2024-03-07	2022-11-15
⚠️	Voice Aloud Reader for pc,windows and mac (Free Use)	11	https://voicealoudreaderforpc.blogspot.com	0.00	0	malware	2024-03-06	2024-03-06
⚠️	YTBlock - Adblock para Youtube	9 000	YTAdblock	4.91	57	malware	2024-03-01	2024-02-09
⚠️	OVO Official	30	https://ovogame.pro	0.00	0	malware	2024-02-28	2024-02-28
⚠️	Snake	50 000	https://snake.9834722.xyz	4.19	52	malware	2024-02-27	2021-10-04
⚠️	Settings for Chrome	600 000	Chrome Settings	3.75	4	malware	2024-02-27	2022-06-24

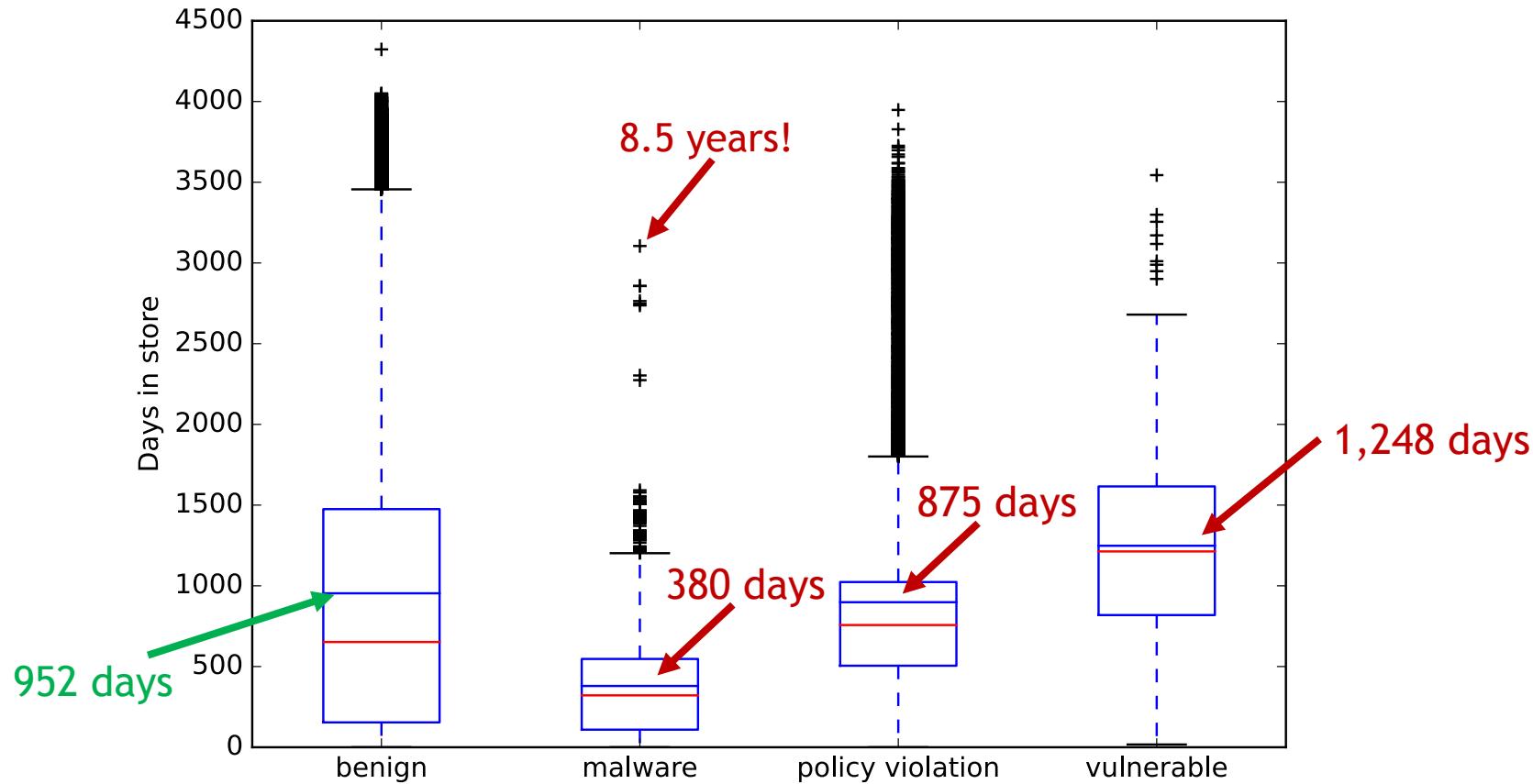
Browser Extension Collection: Chrome-Stats

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 [1]	184	152	March 16, 2021
Benign extensions	226,762	92,482	Before May 1, 2023

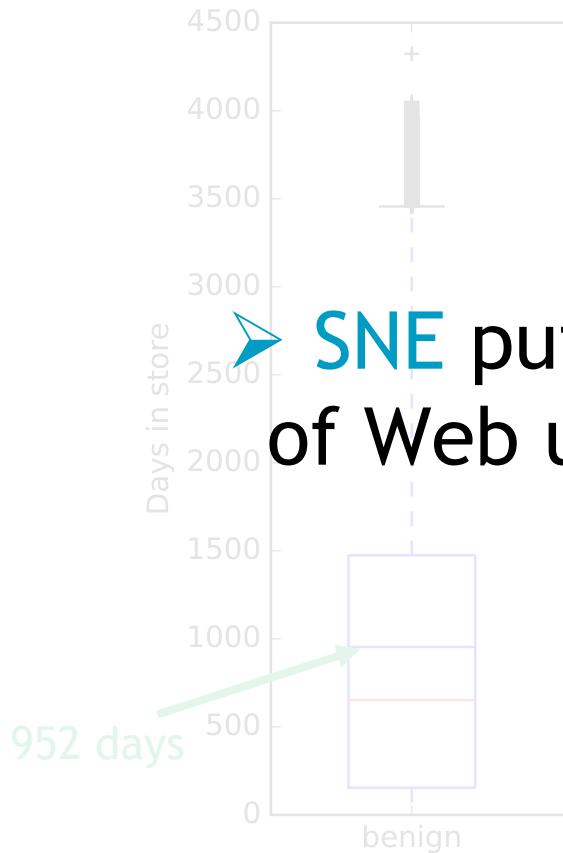
Number of Days in the CWS



Number of Days in the CWS



Number of Days in the CWS

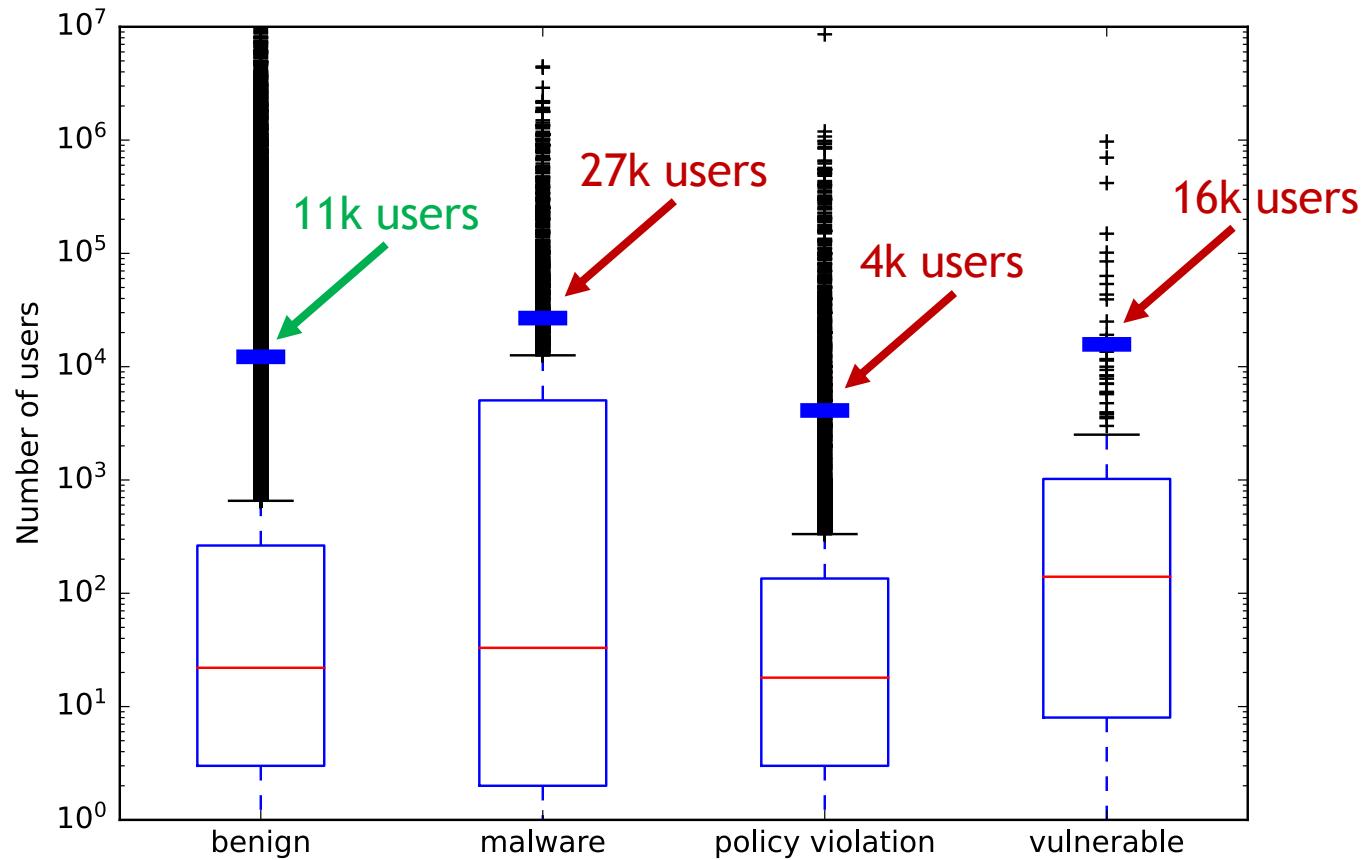


➤ SNE put the security & privacy
of Web users **at risk *for years***

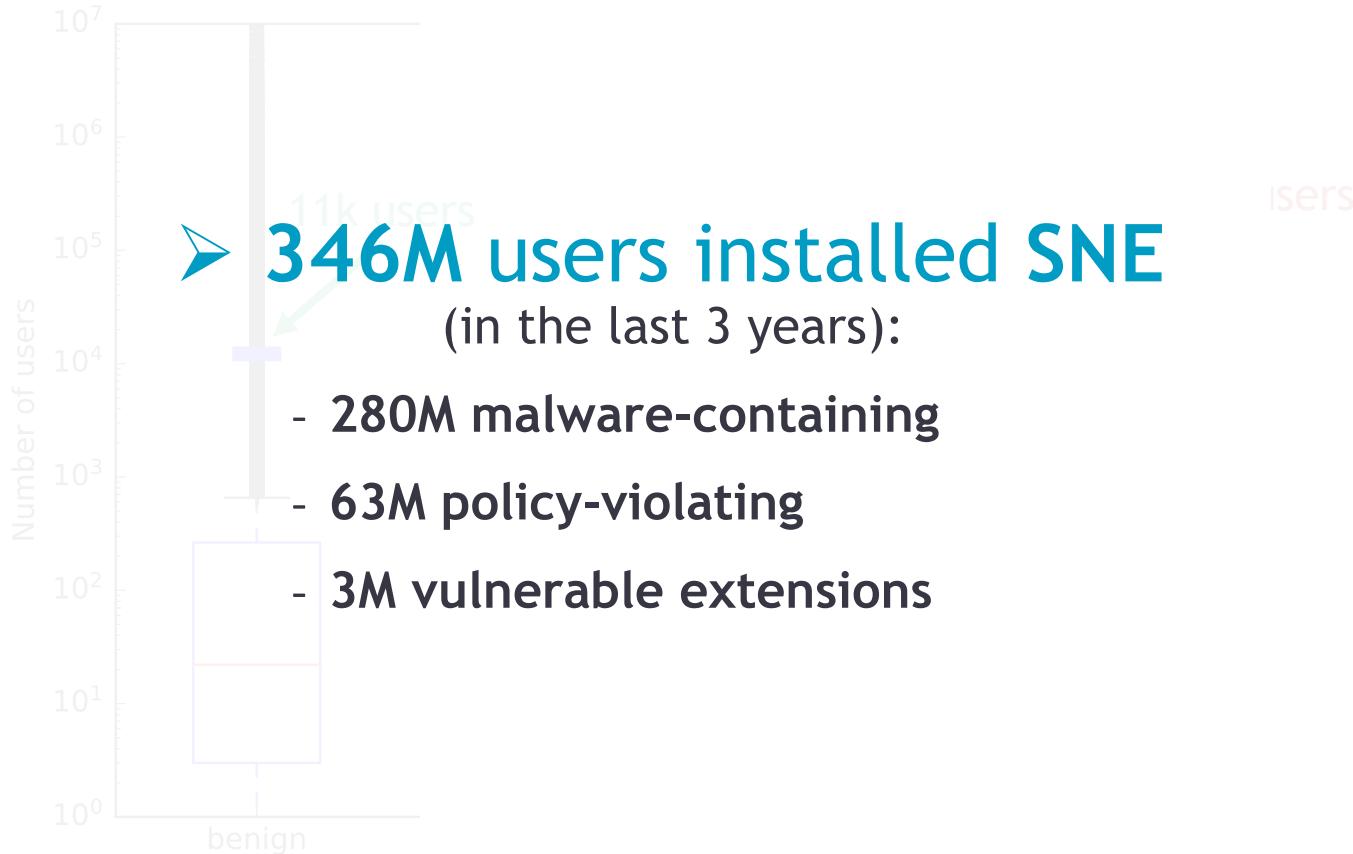
,248 days

952 days

Number of Users



Number of Users



Detecting SNE with Clustering-based Approaches

- Source-code comparison across extensions (*ssdeep* fuzzy hash)
- Clustering similar extensions together (i.e., 100% *ssdeep* overlap)
- 3,270 clusters with [2; 1,397] extensions (20,822 extensions clustered)

Detecting SNE with Clustering-based Approaches

- 3,270 clusters:
 - 2,296 **clusters** contain just **benign** extensions
 - 321 **clusters** only **SNE**
 - 14 clusters with > 100 SNE and 2 with > 863 SNE each
 - Analyzing extensions for similarities could enable to detect SNE
 - 653 **clusters** of **benign** (5,552 extensions) and **SNE** (5,126)
 - Extensions in a cluster with SNE should be flagged for more analyses

How to Detect Security-Noteworthy Extensions?

- **Contain malware**

- Designed by malicious actors to harm victims
 - E.g., steal user-sensitive data, track users, propagate malware

- **Violate the Chrome Web Store policies**

- E.g., deceive users, promote unlawful activities, lacking a privacy policy

- **Contain vulnerabilities**



- Designed by well-intentioned developers... but contain some vulnerabilities
 - E.g., can lead to user-sensitive data exfiltration

How to Detect Security-Noteworthy Extensions?

▪ Contain malware

– Exploit the web browser to download and execute malicious code

▪ Violate the Chrome Web Store policies

– Exploit the web browser to download and execute malicious code

▪ Contain vulnerabilities

- Designed by well-intentioned developers... but contain some vulnerabilities
- E.g., can lead to user-sensitive data exfiltration



Analysis of Vulnerable Extensions

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



Analysis of Vulnerable Extensions

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



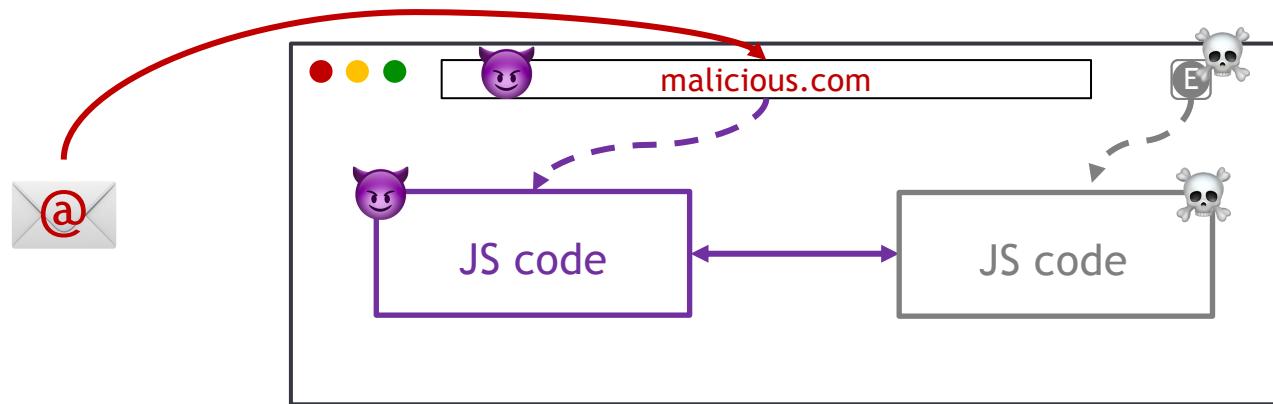
Analysis of Vulnerable Extensions

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



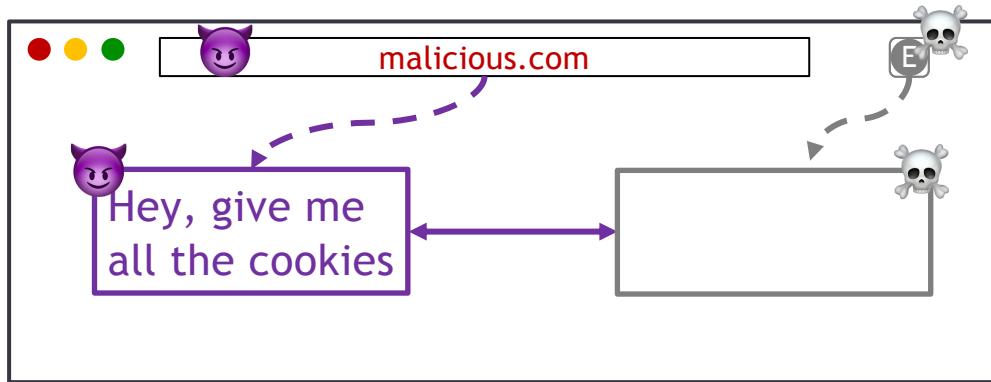
Analysis of Vulnerable Extensions

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



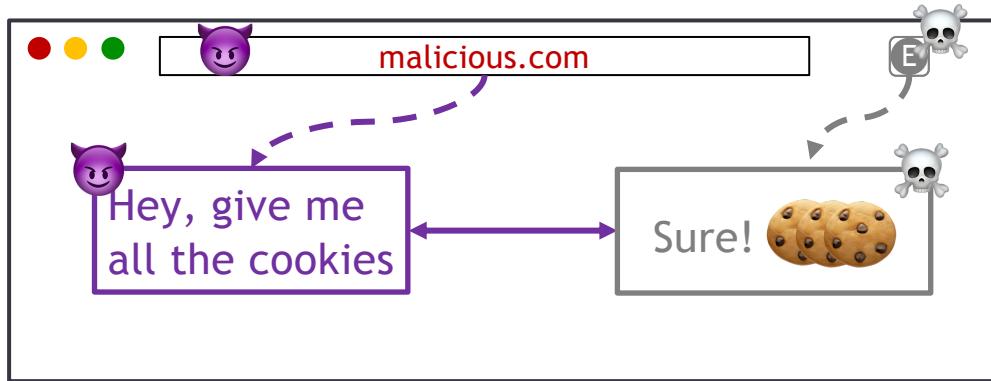
Analysis of Vulnerable Extensions

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



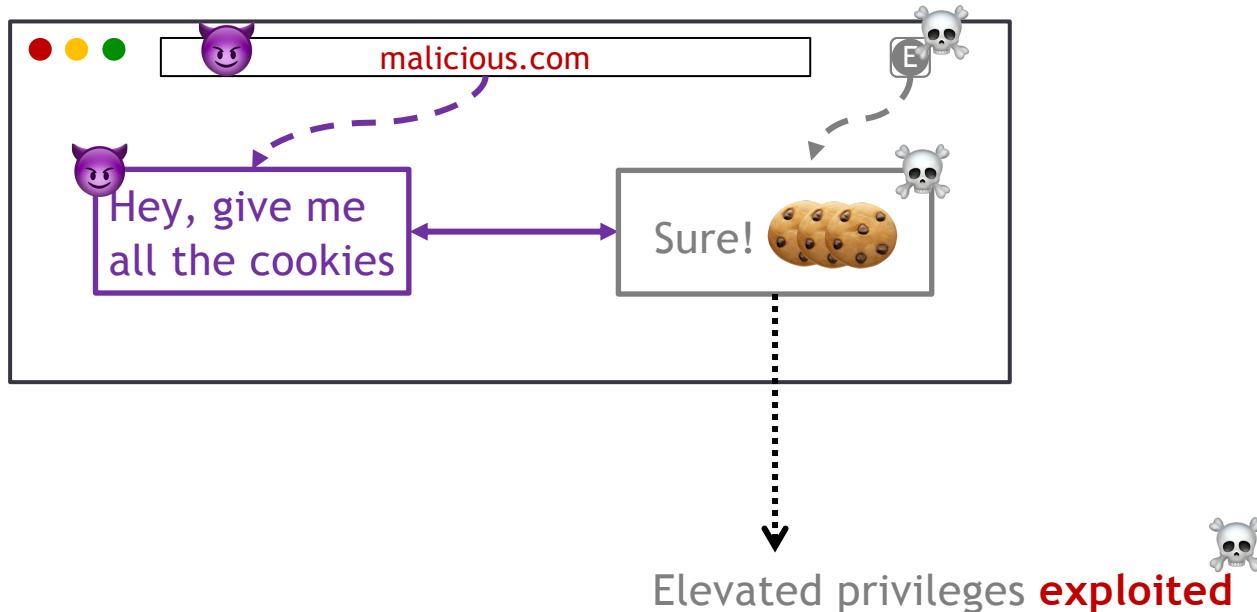
Analysis of Vulnerable Extensions

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



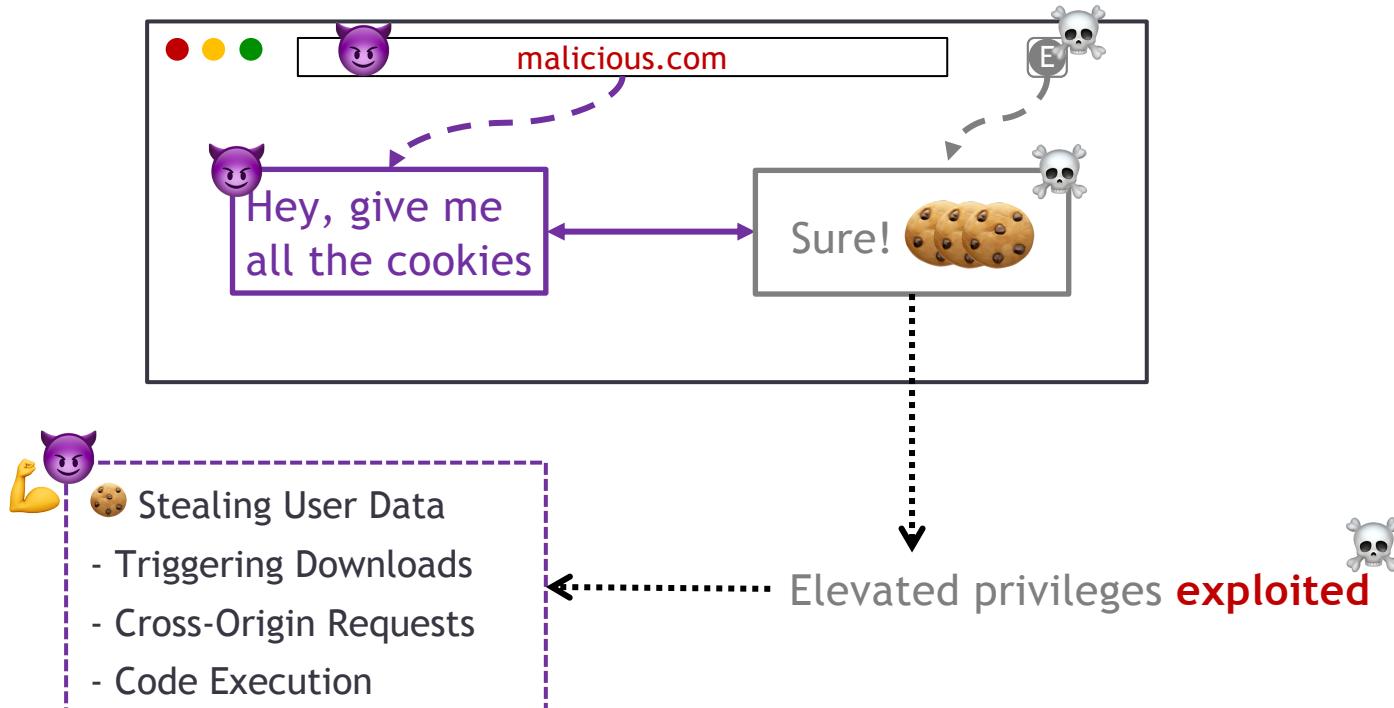
Analysis of Vulnerable Extensions

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



Analysis of Vulnerable Extensions

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



Detecting Vulnerable Extensions



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

Detecting Vulnerable Extensions



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



Detecting Vulnerable Extensions



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



Detecting Vulnerable Extensions

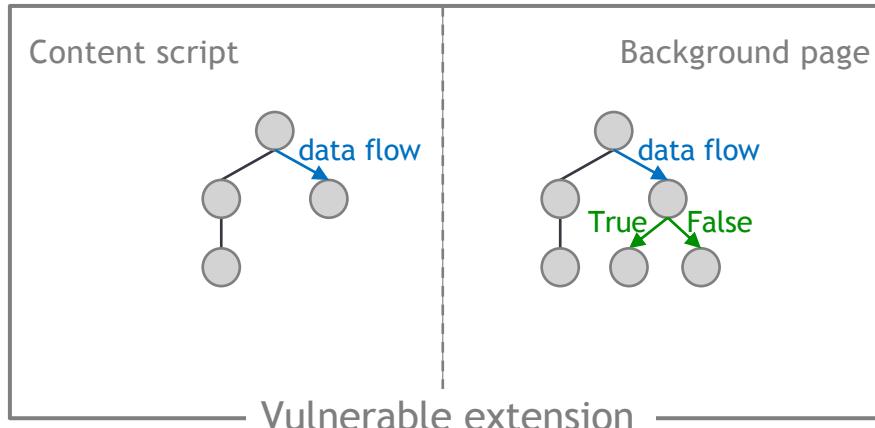


> DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions

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



Malicious web page



Per-component JS code abstraction

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

Detecting Vulnerable Extensions

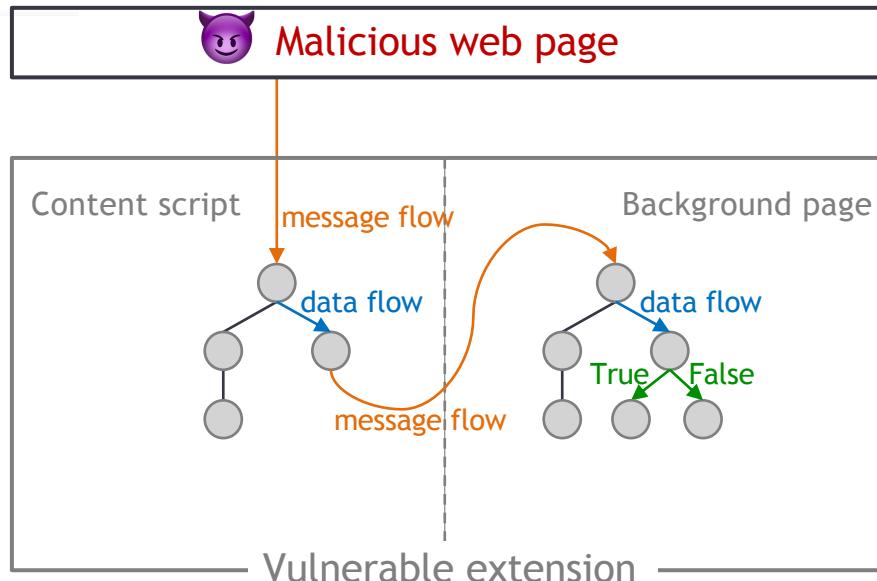


CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY



> 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 JS code abstraction

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

Extension Dependence Graph (EDG)

- Message interactions

Detecting Vulnerable Extensions

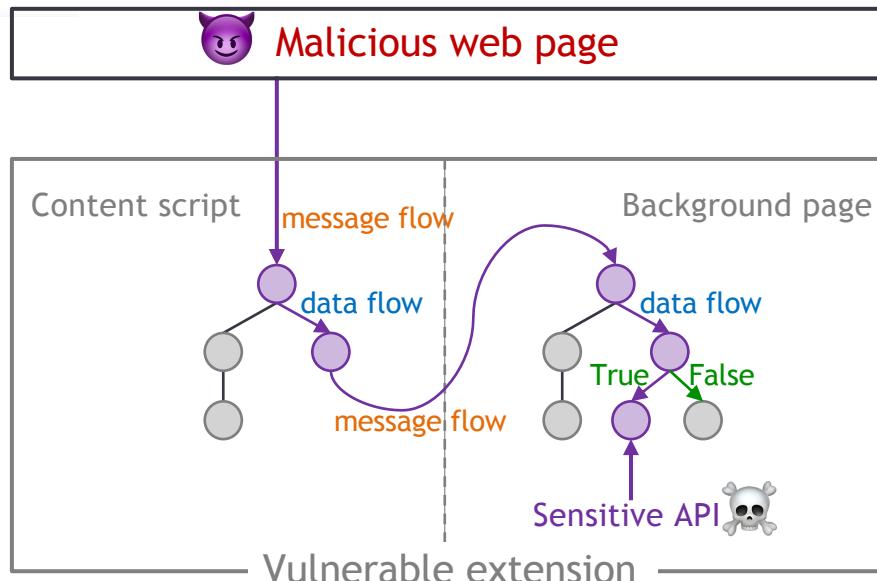


CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY



> 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 JS code abstraction

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

Extension Dependence Graph (EDG)

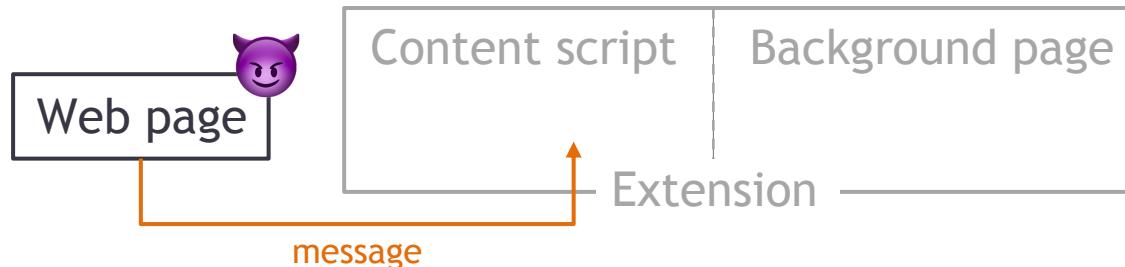
- Message interactions

Suspicious data flow tracking

- Detects any path between an attacker & sensitive APIs

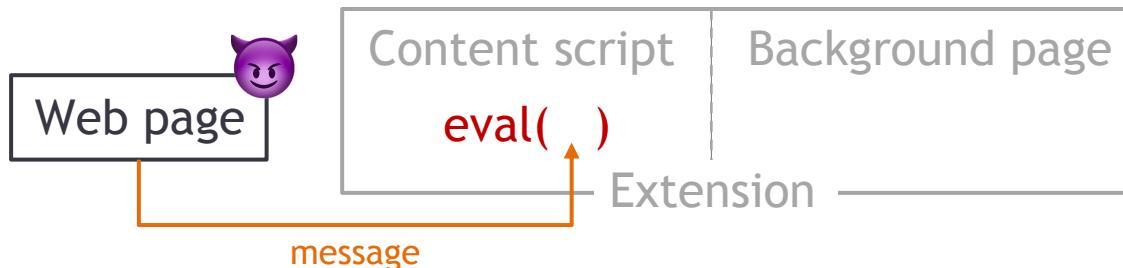
Simplified Example of a Vulnerability

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



Simplified Example of a Vulnerability

```
// Content script code  
window.addEventListener("message", function(event) {  
  
    eval(event.data);  
  
})
```



Detecting Vulnerable Extensions with DOUBLEX

Analyzed 155k Chrome extensions from 2021 with DOUBLEX

- **184 vulnerable Chrome extensions**
- Impacting **3M users**

Life Cycle of Vulnerable Chrome Extensions

- Analyzed 165k extensions from 2020 with DOUBLEX
 - 193 vulnerable extensions (184 in 2021)
 - vulnerability disclosure for 35 extensions (48 extensions when including 2021)
 - Comparison of vulnerable extensions in 2020 vs. 2021
 - not in the CWS anymore: 30 / 193
 - vulnerability fixed: 3 / 193
 - turned vulnerable: 5 / 184
 - new vulnerable: 19 / 184
- **still vulnerable: 160 (83%)! ➤ Need to prevent vulnerable extensions from entering the CWS → DOUBLEX**

Detecting Vulnerable Extensions with DOUBLEX

Analyzed 155k Chrome extensions from 2021 with DOUBLEX

- **184 vulnerable Chrome extensions**
- Impacting **3M users**
- **Precision:** 89% of the flagged extensions are vulnerable
- **Recall:** 93% of known vulnerabilities [2] are detected
- **Integration** in the **vetting process** conducted by Google
- **Available online**, for developers
 - (even in other fields!)



Security-Noteworthy Extensions

- **Contain malware**

- Designed by malicious actors to harm victims
 - E.g., steal user-sensitive data, track users, propagate malware

- **Violate the Chrome Web Store policies**

- E.g., deceive users, promote unlawful activities, lacking a privacy policy

- **Contain vulnerabilities**

- Designed by well-intentioned developers... but contain some vulnerabilities
 - E.g., can lead to user-sensitive data exfiltration

- **Can be fingerprinted**

- Designed by well-intentioned developers...
 - ... but can lead to, e.g., tracking users across sites, inferring sensitive user information



Security-Noteworthy Extensions

- **Contain malware**

- Designed by malicious actors to harm victims
 - E.g., steal user-sensitive data, track users, propagate malware

- **Violate the Chrome Web Store policies**

- E.g., deceive users, promote unlawful activities, lacking a privacy policy

- **Contain vulnerabilities**

- Designed by well-intentioned developers... but contain some vulnerabilities
 - E.g., can lead to user-sensitive data exfiltration

- **Can be fingerprinted**

- Designed by well-intentioned developers...
 - ... but can lead to, e.g., tracking users across sites, inferring sensitive user information



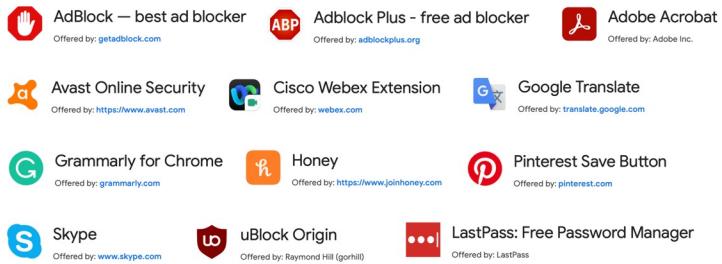
Fass et al.
CCS 2021



Agarwal et al.
CCS 2024

Takeaways – Browser Extension (In)Security

Browser Extensions are Popular



- 125k Chrome extensions totaling over 1.6B active users

Security-Noteworthy Extensions (SNE)

- Contain **malware**
- Violate the Chrome Web Store **policies**
- Contain **vulnerabilities**
- Can be **fingerprinted**



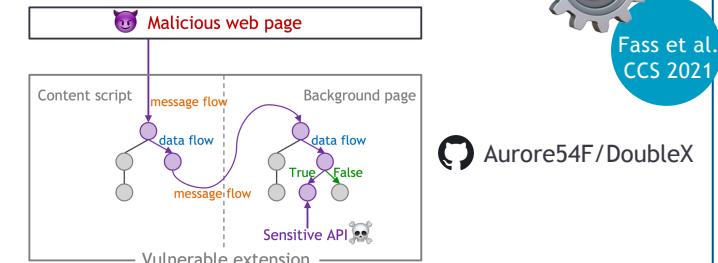
Thank you

What is in the Chrome Web Store?



- 350M users installed SNE in the last 3 years
- These SNE stay in the Chrome Web Store **for years**
- Extensions have a **short life cycle** in the CWS (60% stay 1 year)
- Critical **lack of maintenance** in the CWS (60% received no update)

Detecting Vulnerable Extensions with DOUBLEX



- DOUBLEX detects suspicious **data flows** in browser extensions
- 184 vulnerable extensions | Precision: 89% | Recall: 93%



fass@cispa.de



<https://aurore54f.github.io>



@AuroreFass

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

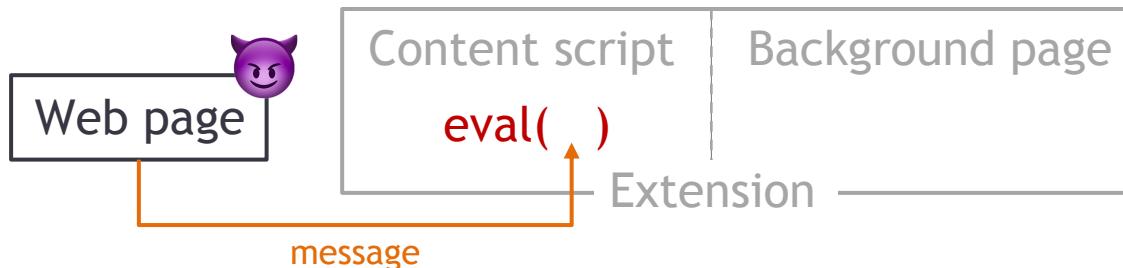
- Peeking through the window: Fingerprinting Browser Extensions through Page-Visible Execution Traces and Interactions

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

Additional slides about some questions asked

Per-Component JavaScript Code Abstraction

```
// Content script code
window.addEventListener("message", function(event) {
    eval(event.data);
})
```



Per-Component JavaScript Code Abstraction

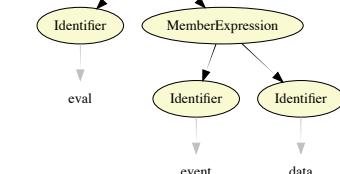
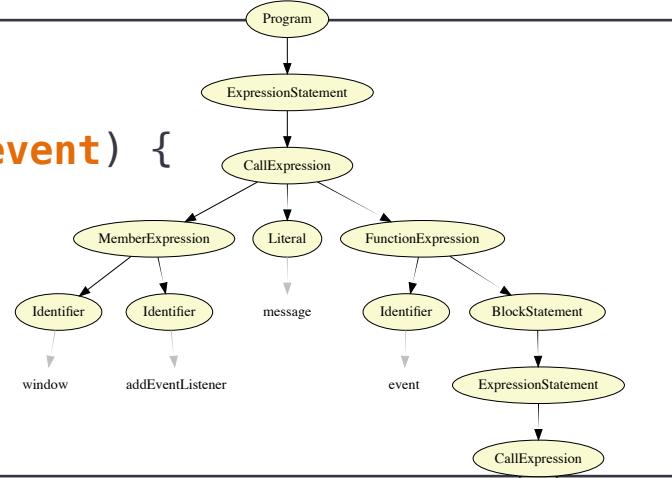
```
// Content script code
```

```
window.addEventListener("message", function(event) {  
    eval(event.data);  
})
```

Abstract code representation



AST



Per-Component JavaScript Code Abstraction

```
// Content script code  
window.addEventListener("message", function(event) {  
    eval(event.data);  
})
```



Abstract code representation



AST

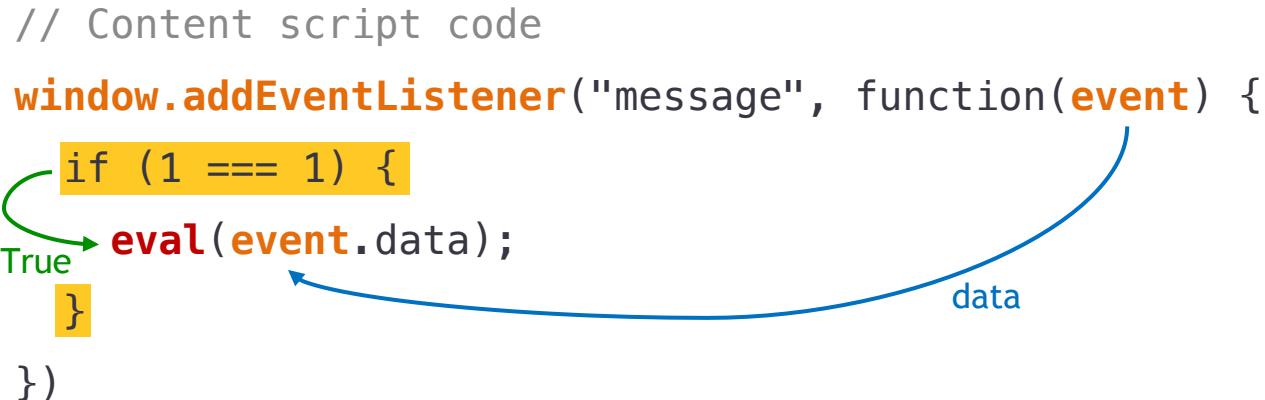
- variable dependencies



data flow

Per-Component JavaScript Code Abstraction

```
// Content script code  
window.addEventListener("message", function(event) {  
    if (1 === 1) {  
        eval(event.data);  
    }  
})
```



The diagram illustrates data flow analysis on the provided JavaScript code. A blue curved arrow labeled 'data' points from the variable 'event.data' in the code to the argument of the 'eval' function. A green curved arrow labeled 'True' points from the condition '1 === 1' to the start of the code block. The code block itself is highlighted in yellow.

Abstract code representation



AST

- conditions



control flow

- variable dependencies



data flow

Per-Component JavaScript Code Abstraction

```
// Content script code
window.addEventListener("message", function(event) {
    if (1 === 1) {
        window["e" + "val"](event.data);
    }
})
```



Abstract code representation



AST

– conditions



control flow

– variable dependencies



data flow

– variable values



pointer analysis

Extension Dependence Graph

```
// Content script code
window.addEventListener("message", function(event) {
    if (1 === 1) {
        True
        window["e" + "val"](event.data);
    }
})
```



- external messages
- internal messages

Extension Dependence Graph

```
// Content script code
window.addEventListener("message", function(event) {
    if (1 === 1) {
        window["e" + "val"](event.data);
    }
})
```

The diagram shows a snippet of content script code. A purple devil icon is positioned above the word 'event'. A blue curved arrow labeled 'data' points from the variable 'data' in the code to the devil icon. A green curved arrow labeled 'eval' points from the 'eval' keyword in the code to the same devil icon. A green arrow labeled 'True' points from the condition '1 === 1' to the same devil icon.

- external messages
- internal messages

Extension Dependence Graph

```
// Content script code
chrome.runtime.sendMessage({toBP: mess});
```

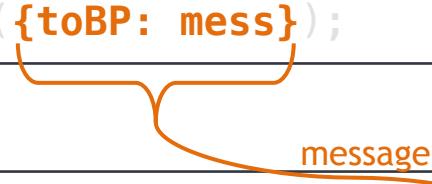
```
// Background page code
chrome.runtime.onMessage.addListener(function(request) {
    })
```

- external messages 
- internal messages

Extension Dependence Graph

```
// Content script code
chrome.runtime.sendMessage({toBP: mess});
```

```
// Background page code
chrome.runtime.onMessage.addListener(function(request) {
})
```

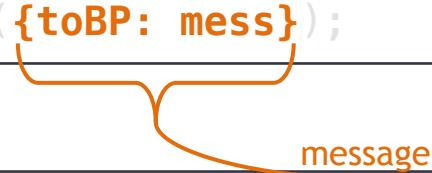


- external messages 
- internal messages 

Extension Dependence Graph

```
// Content script code  
chrome.runtime.sendMessage({toBP: mess});
```

```
// Background page code  
chrome.runtime.onMessage.addListener(function(request) {  
})
```

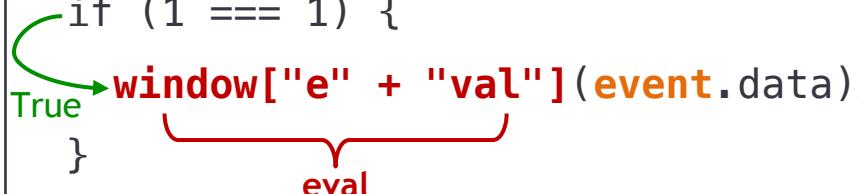


- external messages ✓
- internal messages ✓

➤ Models message interaction within and outside of an extension

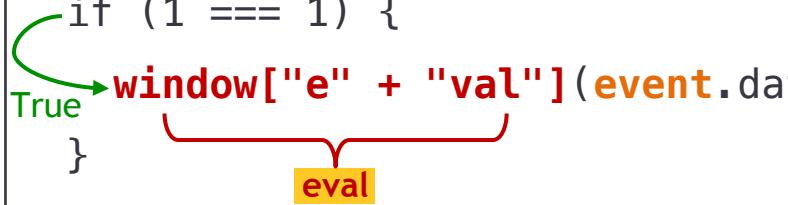
Suspicious Data Flow Tracking

```
1 // Content script code
2 window.addEventListener("message", function(event) {
3   if (1 === 1) {
4     True → window["e" + "val"](event.data);
5   }
6 })
```



Suspicious Data Flow Tracking

```
1 // Content script code
2 window.addEventListener("message", function(event) {
3   if (1 === 1) {
4     True → window["e" + "val"](event.data);
5   }
6 })
```



Suspicious Data Flow Tracking

```
1 // Content script code
2 window.addEventListener("message", function(event) {
3   if (1 === 1) {
4     True → window["e" + "val"]([event].data);
5   }
6 })
```

The diagram illustrates the flow of data in the provided JavaScript code. The variable `data` (highlighted in yellow) is passed as an argument to the `eval` function (highlighted in yellow). This `eval` function is then called by the `window["e" + "val"]` expression (also highlighted in yellow). A green curved arrow labeled `True` points from the `if` condition to the `eval` call. Above the `event` variable, a purple devil icon with a mischievous grin is positioned, symbolizing potential malicious intent or a security vulnerability.

Suspicious Data Flow Tracking

```
1 // Content script code
2 window.addEventListener("message", function(event) {
3   if (1 === 1) {
4     True → window["e" + "val"]([event].data);
5   }
6 })
```

The diagram illustrates the execution flow of the provided JavaScript code. A purple devil icon is positioned above the code. A green curved arrow labeled 'True' points from the condition in line 3 to the first argument of the eval call in line 4. A red bracket labeled 'eval' encloses the eval call. A blue curved arrow labeled 'data' points from the second argument of the eval call back to the variable 'event' in the function parameter of line 2.



Suspicious Data Flow Tracking

```
1 // Content script code
2 window.addEventListener("message", function(event) {
3   if (1 === 1) {
4     window["e" + "val"](event.data);
5   }
6 })
```

The diagram illustrates the flow of data in the provided content script code. A purple devil icon is positioned above the code. A green curved arrow labeled 'True' points from the 'if (1 === 1)' condition to the 'eval' call. A red bracket labeled 'eval' encloses the 'eval' call. A blue curved arrow labeled 'data' points from the 'data' parameter in the event object to the argument of the eval call.



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

Suspicious Data Flow Tracking

```
1 // Content script code
```

```
2 window.addEventListener("message", function(event) {
```

```
3   if (1 === 1) {
```

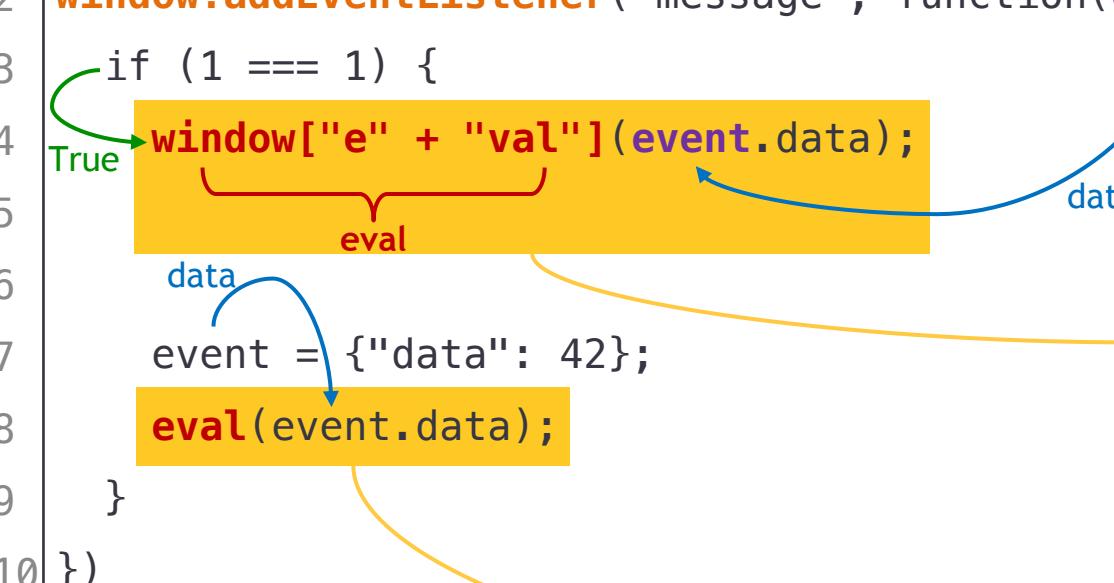
```
4     window["e" + "val"](event.data);
```

```
5       ^_____  
6       |  eval  
7       |  
8       event = {"data": 42};
```

```
9       eval(event.data);
```

```
10 }
```

```
10 })
```



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

{"direct-danger2": "eval",
"value": "eval(42)",
"line": "8 - 8",
"dataflow": false}
```

Sensitive APIs Considered

Flaw category	All components	High-privilege components
Code Execution	eval, setInterval, setTimeout	tabs.executeScript
Triggering Downloads		downloads.download
Cross-Origin Requests	\$.ajax, jQuery.ajax, fetch, \$.get, jQuery.get, \$http.get, \$.post, \$http.post, XMLHttpRequest().open, jQuery.post, XMLHttpRequest.open	
Data Exfiltration		bookmarks.getTree, cookies.getAll, history.search, topSites.get

Large-Scale Analysis of Chrome Extensions

Attacker capabilities	#Exploitable
Code Execution	63
Triggering Downloads	21
Cross-Origin Requests	49
Data Exfiltration	76
Sum	209