



# DOUBLEX: Statically Detecting Vulnerable Data Flows in Browser Extensions at Scale

Aurore Fass

Stanford Security Workshop - 04/06/2022

Based on joint work with Dolière Francis Somé, Michael Backes, and Ben Stock

# Browser Extensions...

are popular to improve user browsing experience



**AdBlock — best ad blocker**

Offered by: [getadblock.com](https://getadblock.com)



**Adblock Plus - free ad blocker**

Offered by: [adblockplus.org](https://adblockplus.org)



**Adobe Acrobat**

Offered by: Adobe Inc.



**Avast Online Security**

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



**Cisco Webex Extension**

Offered by: [webex.com](https://webex.com)



**Google Translate**

Offered by: [translate.google.com](https://translate.google.com)



**Grammarly for Chrome**

Offered by: [grammarly.com](https://grammarly.com)



**Honey**

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



**Pinterest Save Button**

Offered by: [pinterest.com](https://pinterest.com)



**Skype**

Offered by: [www.skype.com](https://www.skype.com)



**uBlock Origin**

Offered by: Raymond Hill (gorhill)



**LastPass: Free Password Manager**

Offered by: LastPass

# Browser Extensions...

are popular to improve user browsing experience



AdBlock — best ad blocker

Offered by: [getadblock.com](https://getadblock.com)



Adblock Plus - free ad blocker

Offered by: [adblockplus.org](https://adblockplus.org)



Adobe Acrobat

Offered by: Adobe Inc.



Avast Online Security

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



Cisco Webex Extension

Offered by: [www.cisco.com](https://www.cisco.com)



Google Translate

Offered by: [translate.google.com](https://translate.google.com)



Grammarly for Chrome

Offered by: [grammarly.com](https://grammarly.com)



Honey

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



Pinterest Save Button

Offered by: [pinterest.com](https://pinterest.com)



Skype

Offered by: [www.skype.com](https://www.skype.com)



uBlock Origin

Offered by: Raymond Hill (gorhill)



LastPass: Free Password Manager

Offered by: LastPass

BUT

# Browser Extensions...

may introduce security and privacy threats

e.g.,

- execute arbitrary code in *any* websites, even without a vulnerability in the websites themselves
- exfiltrate sensitive user data to *any* websites



# Browser Extensions are Highly Privileged

- Have access to privileged APIs and features
  - e.g., an ad-blocker can read/write web page content
- Can do tasks that web applications cannot traditionally do
  - e.g., are not subject to the SOP and can access arbitrary cross-domain data (even when a user is logged in)

# Browser Extensions are Highly Privileged

- Have access to privileged APIs and features
    - e.g., an ad-blocker can read/write web page content
  - Can do tasks that web applications cannot traditionally do
    - e.g., are not subject to the SOP and can access arbitrary cross-domain data (even when a user is logged in)
- Attract the interest of attackers

# Malicious vs. Vulnerable Browser Extensions

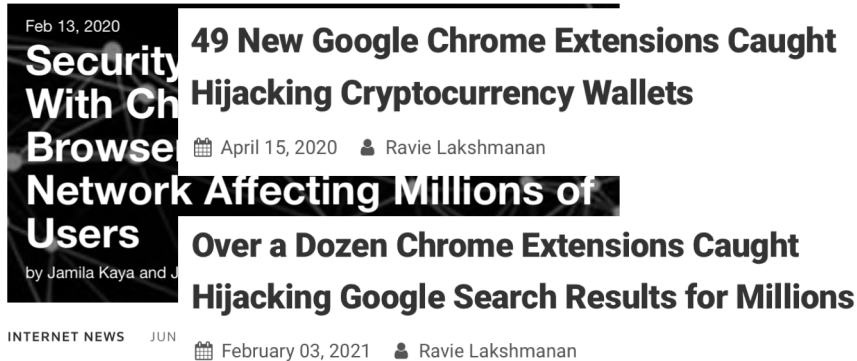
## Malicious Extensions

- Designed by malicious actors
- Aim: harming victims

# Malicious vs. Vulnerable Browser Extensions

## Malicious Extensions

- Designed by malicious actors
- Aim: harming victims

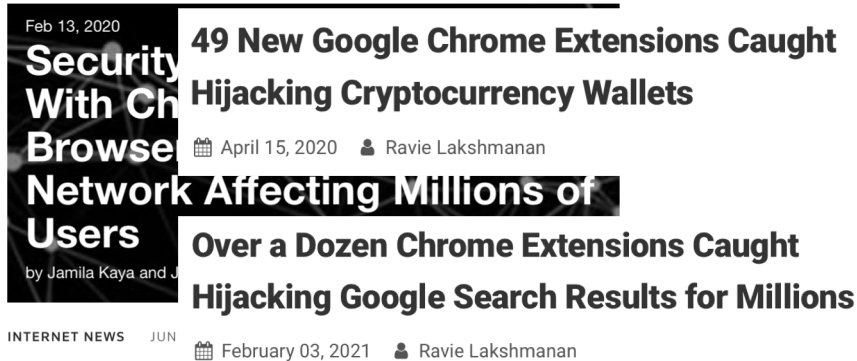


**Exclusive: Massive spying on users of Google's Chrome shows new security weakness**

# Malicious vs. Vulnerable Browser Extensions

## Malicious Extensions

- Designed by malicious actors
- Aim: harming victims
- Chrome vetting system

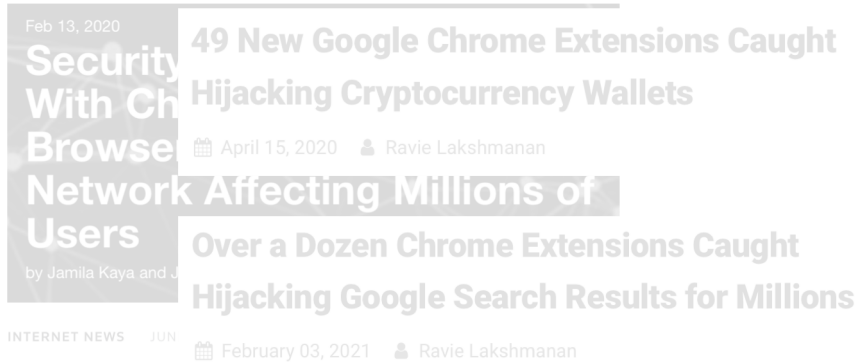


**Exclusive: Massive spying on users of Google's Chrome shows new security weakness**

# Malicious vs. Vulnerable Browser Extensions

## Malicious Extensions

- Designed by malicious actors
- Aim: harming victims
- Chrome vetting system



**Exclusive: Massive spying on users of Google's Chrome shows new security weakness**

By Joseph Menn

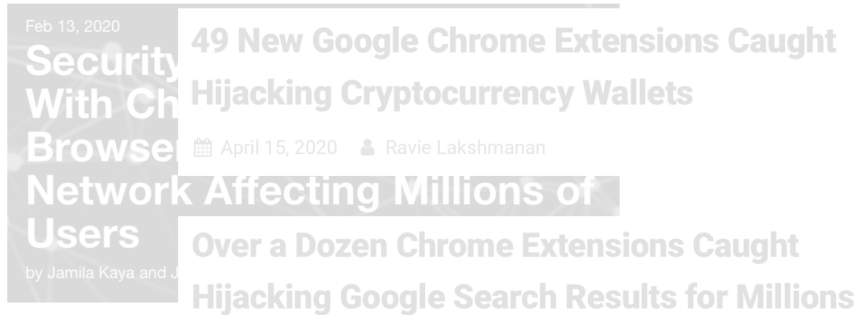
## Vulnerable Extensions

- Designed by well-intentioned developers
- ... but contain some vulnerabilities

# Malicious vs. Vulnerable Browser Extensions

## Malicious Extensions

- Designed by malicious actors
- Aim: harming victims
- Chrome vetting system



INTERNET NEWS JUN  
February 03, 2021 Ravie Lakshmanan

Exclusive: Massive spying on users of Google's Chrome shows new security weakness

## Vulnerable Extensions

- Designed by well-intentioned developers
- ... but contain some vulnerabilities

### EmPoWeb: Empowering Web Applications with Browser Extensions

Dolière Francis Somé

Université Côte d'Azur / Inria, France  
doliere.some@inria.fr

**Abstract**—Browser extensions are third party programs, tightly integrated to browsers, where they execute with elevated privileges in order to provide users with additional functionalities. Unlike web applications, extensions are not subject to the Same Origin Policy (SOP) and therefore can read and write user data on any web application. They also have access to sensitive user information including browsing history, bookmarks, credentials

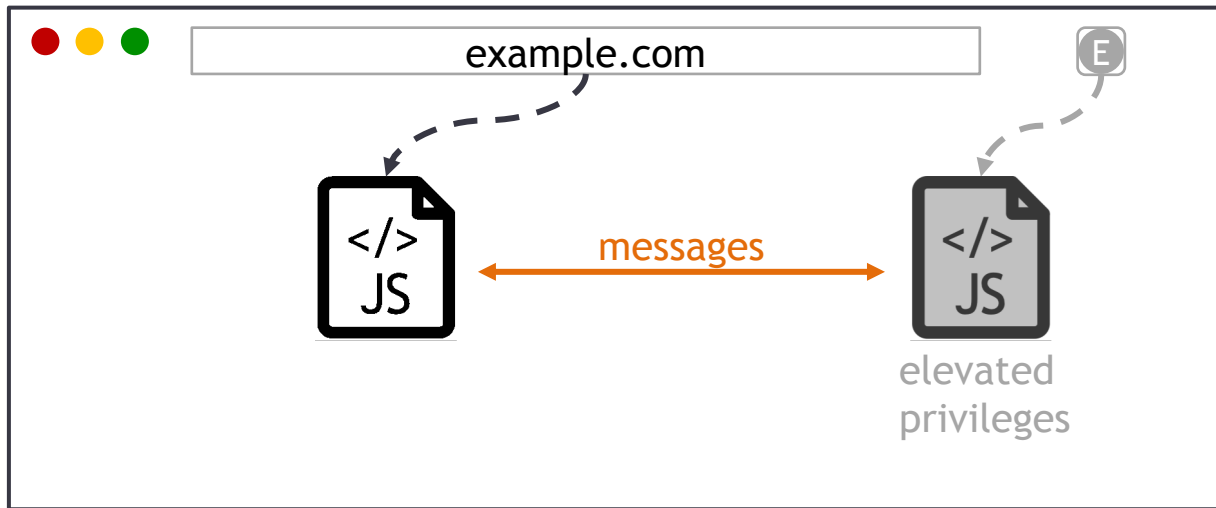
both implement mechanisms such as Cross-Origin Resource Sharing (CORS) [6].

Due to their privileged position in browsers, it is well understood that extensions pose serious security and privacy threats to user data [7], [8], [9], [10], [11], [12], [13]. Therefore, in order to limit extensions capabilities, a mandatory permission

- 66k Chrome extensions analyzed
- 3.3k flagged as suspicious → 95% were FPs

# Exploiting Vulnerable Extensions

- Web Attacker

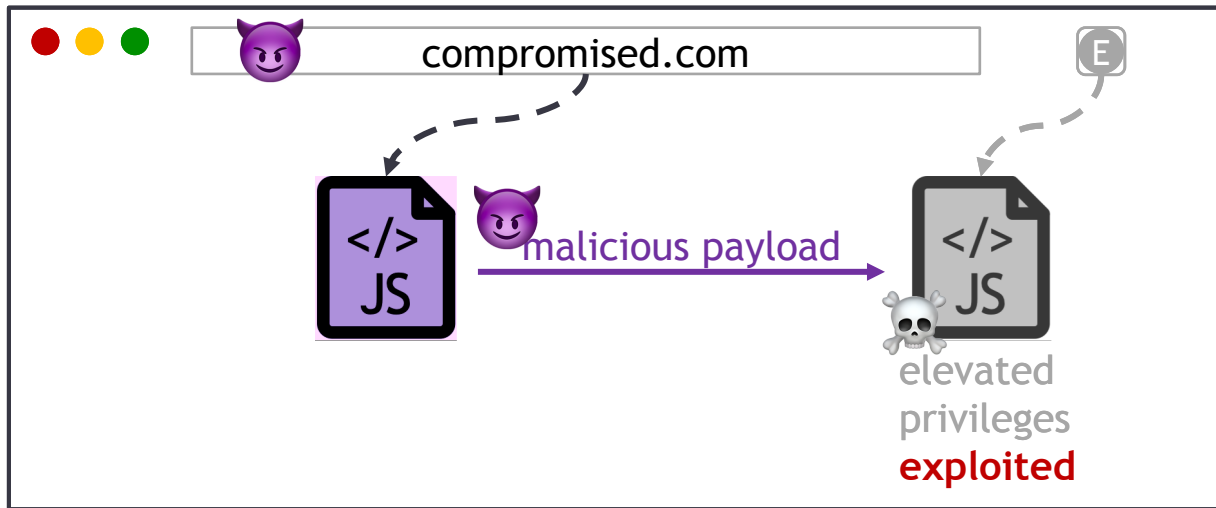


- Confused Deputy



# Exploiting Vulnerable Extensions

- Web Attacker



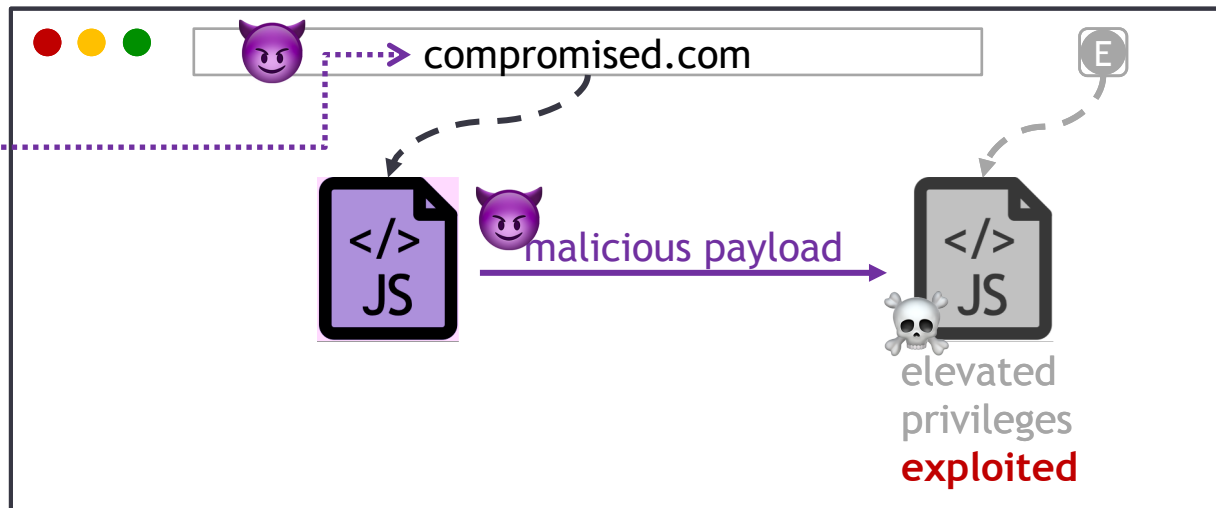
- Confused Deputy

# Exploiting Vulnerable Extensions

- Web Attacker



- 💪 Code Execution
- 💪 Triggering Downloads
- 💪 Cross-Origin Requests
- 💪 Data Exfiltration



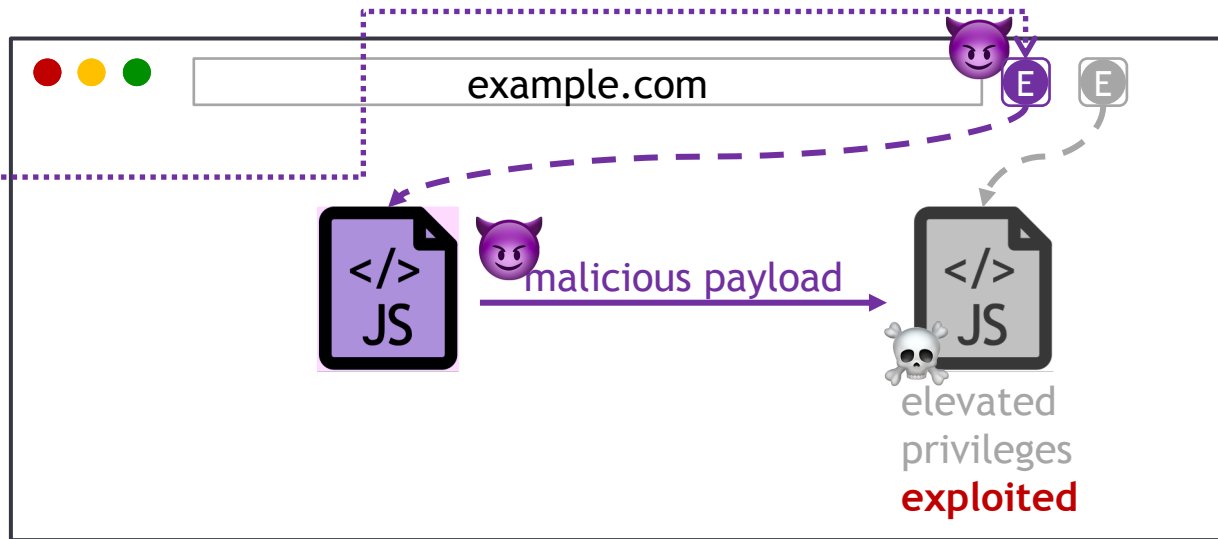
- Confused Deputy

# Exploiting Vulnerable Extensions

- Confused Deputy



- 💪 Code Execution
- 💪 Triggering Downloads
- 💪 Cross-Origin Requests
- 💪 Data Exfiltration



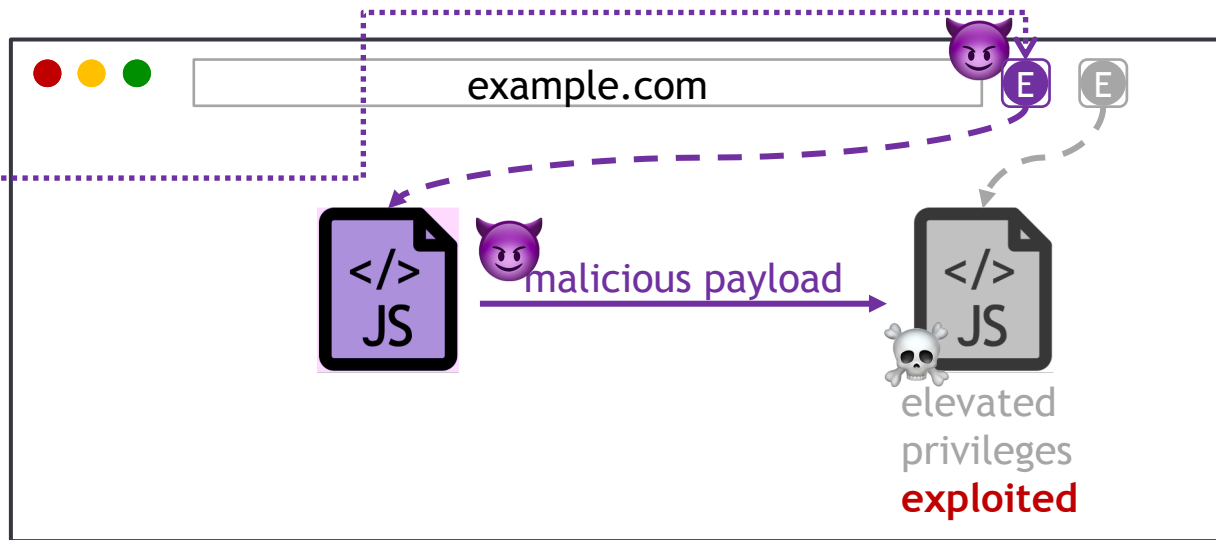
- Web Attacker

# Exploiting Vulnerable Extensions

- Confused Deputy



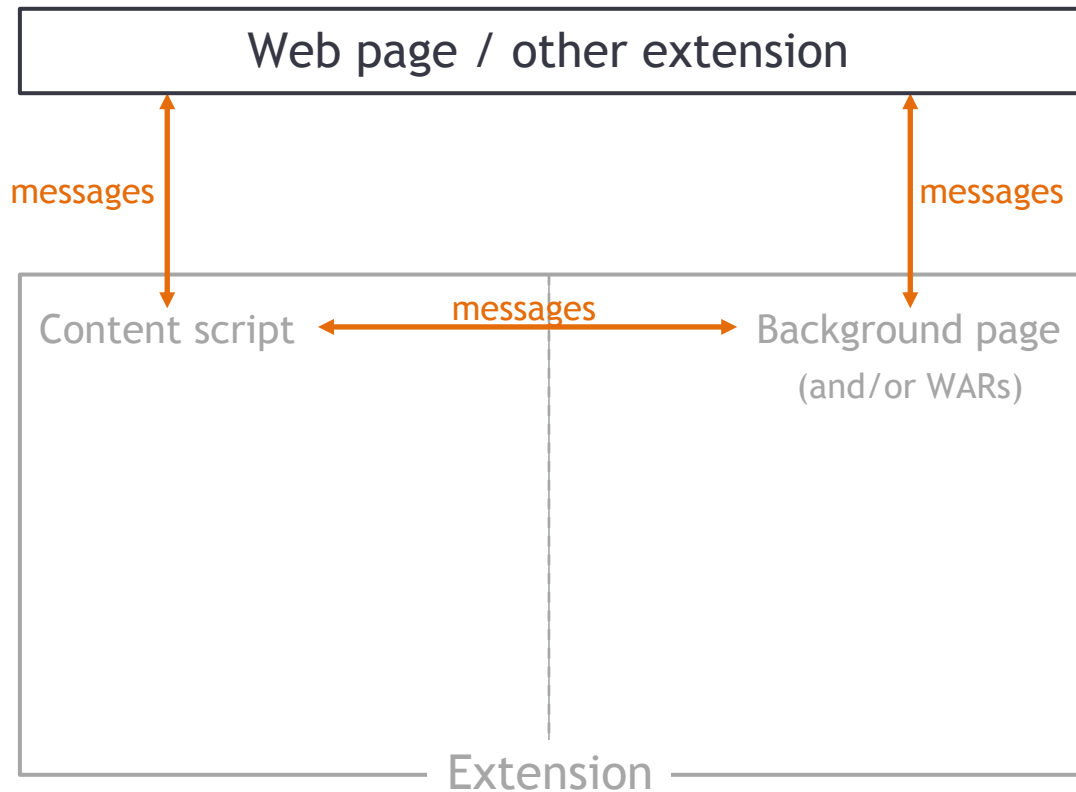
- 🦵 Code Execution
- 🦵 Triggering Downloads
- 🦵 Cross-Origin Requests
- 🦵 Data Exfiltration



- Web Attacker

- RQ: Can we statically analyze browser extensions to detect suspicious external data flows?

# Extension Architecture and Communication

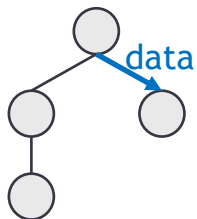


# DOUBLEX: Suspicious Data Flow Detection

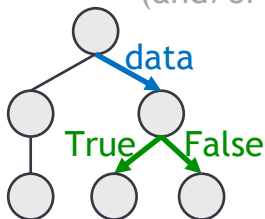
Web page / other extension

Per-component JS code abstraction

Content script

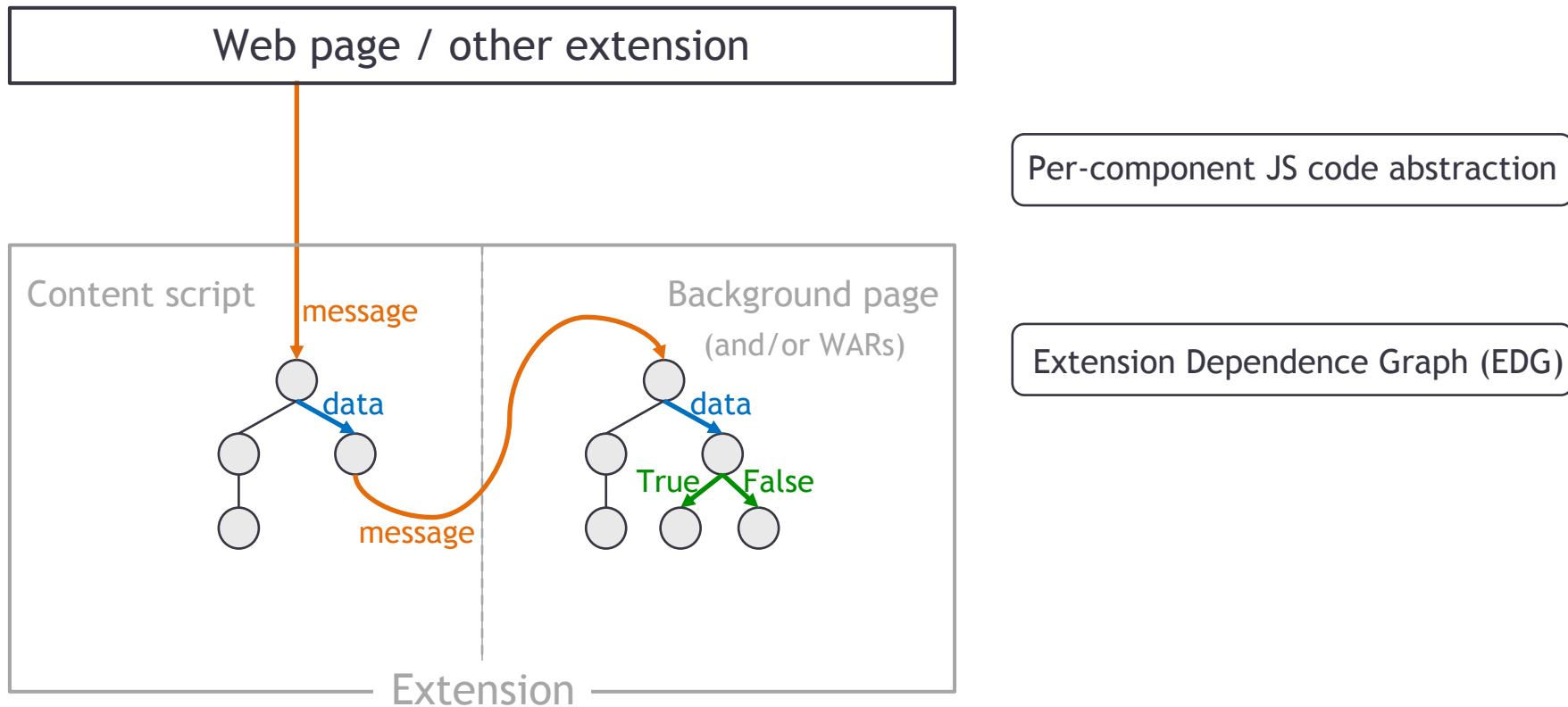


Background page  
(and/or WARs)

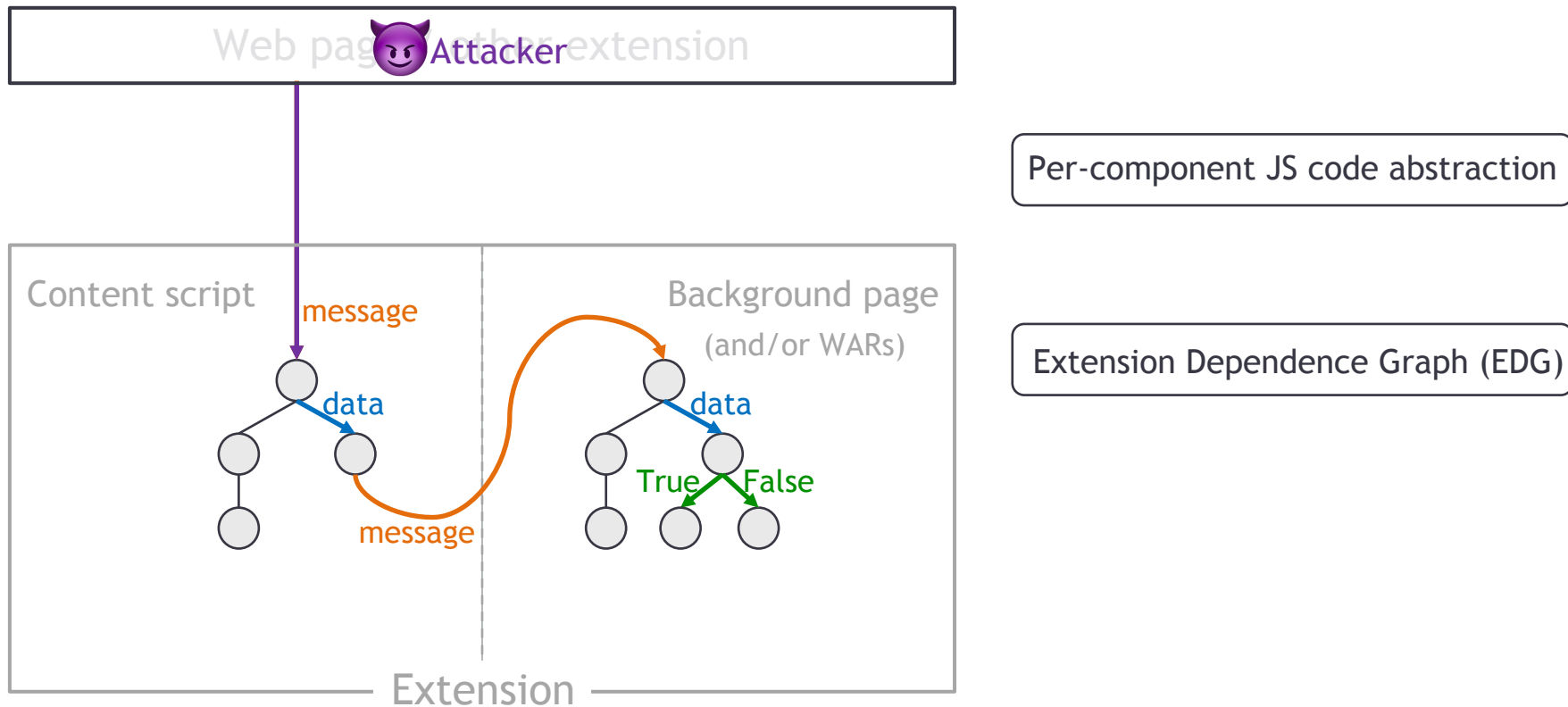


Extension

# DOUBLEX: Suspicious Data Flow Detection

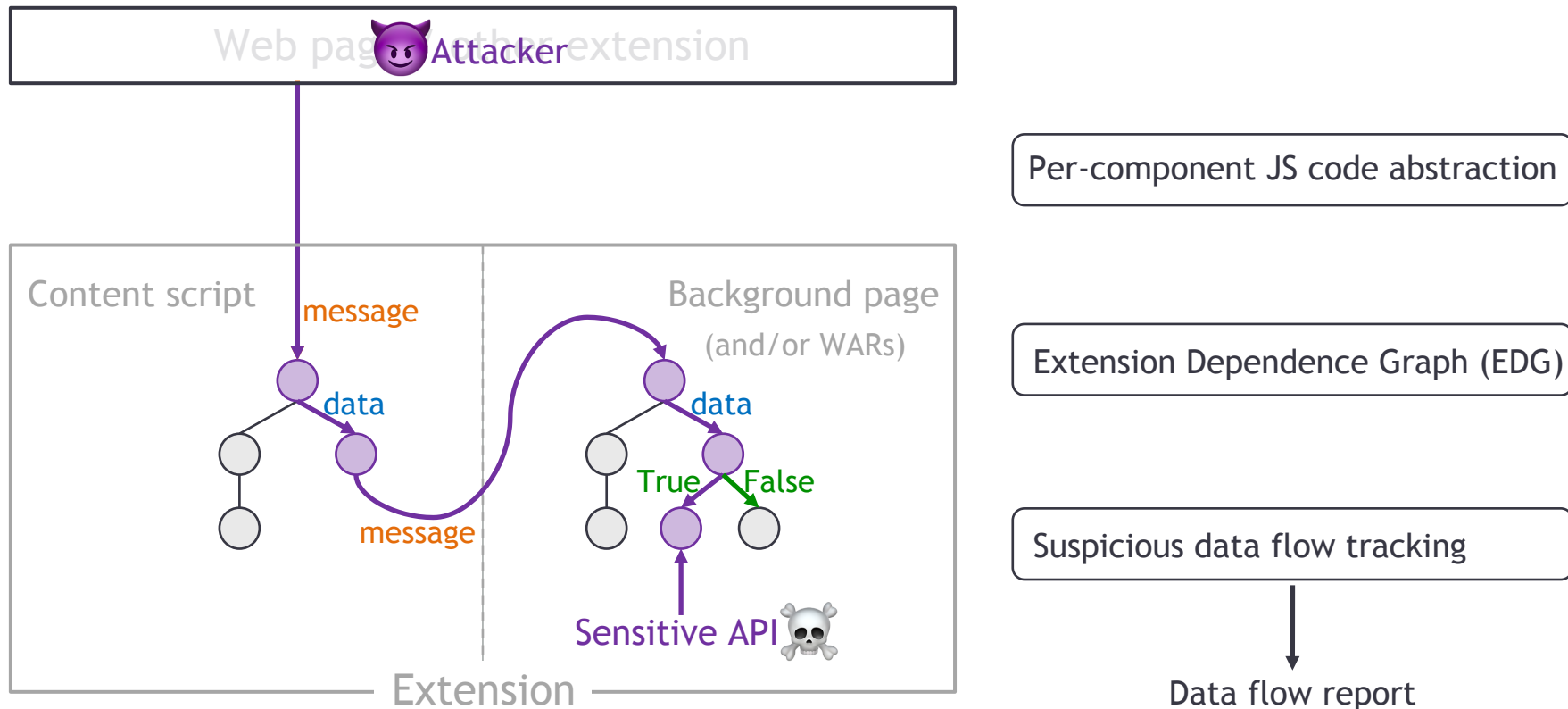


# DOUBLEX: Suspicious Data Flow Detection

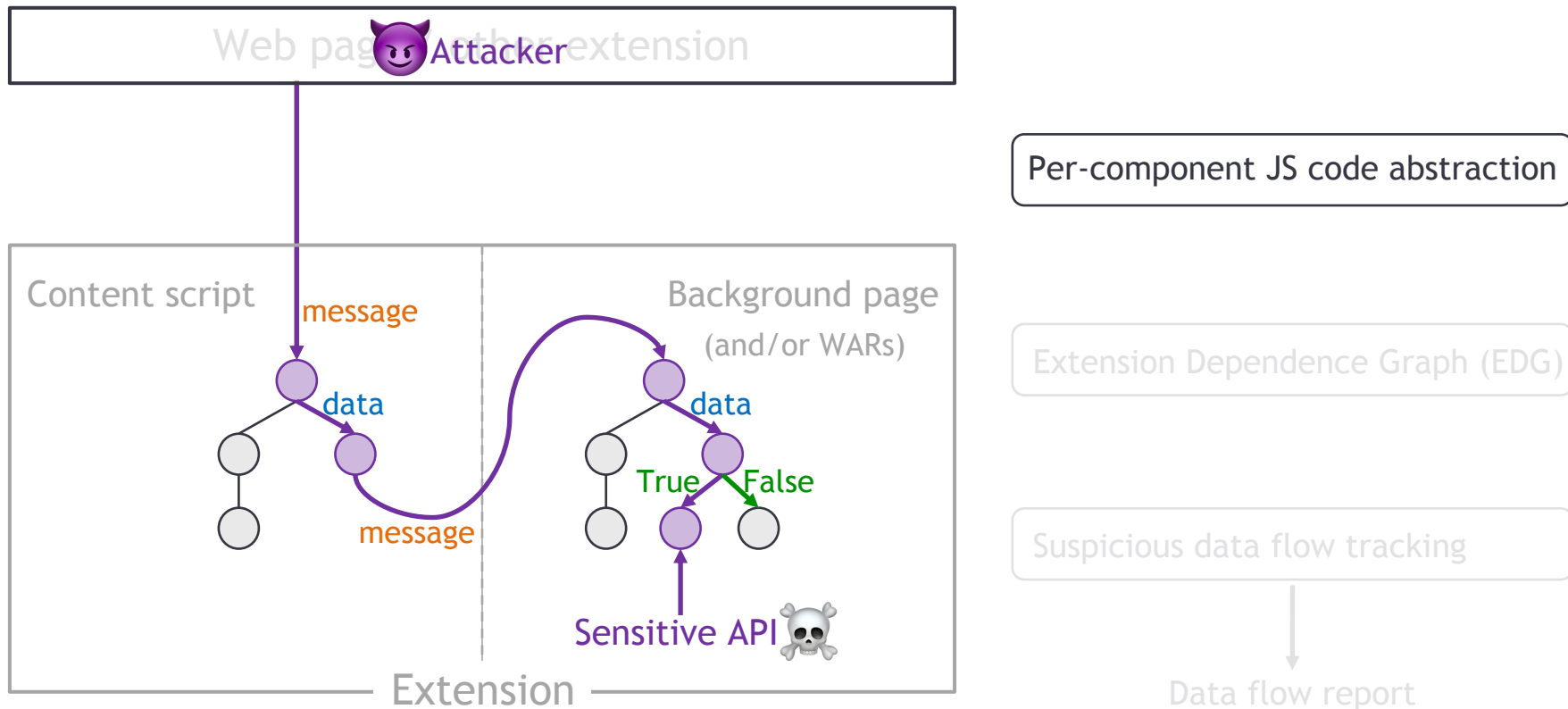




# DOUBLEX: Suspicious Data Flow Detection

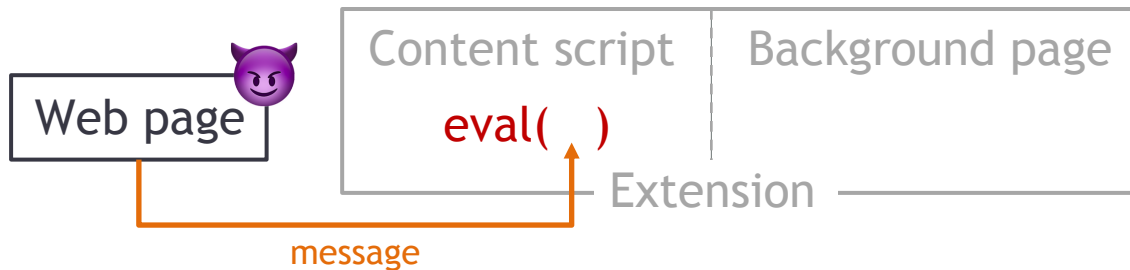


# DOUBLEX: Suspicious Data Flow Detection



# Per-Component JS Code Abstraction

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



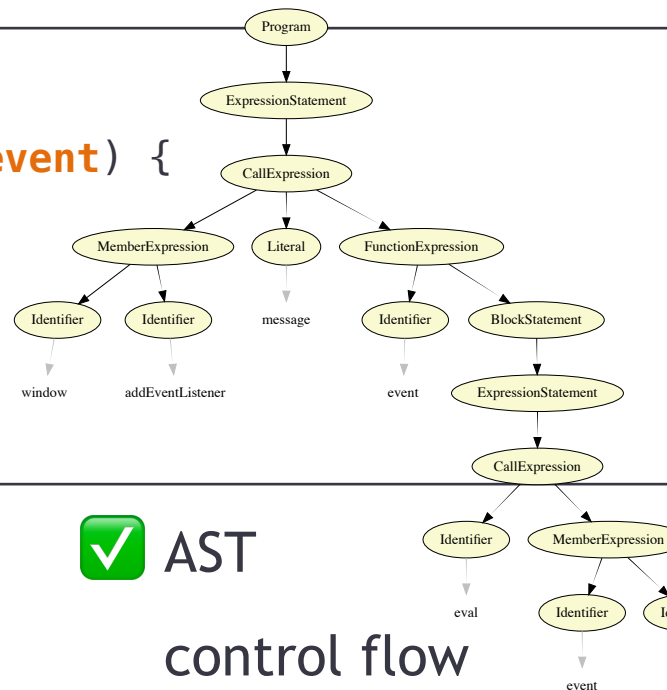
# Per-Component JS Code Abstraction

```
// Content script code
```

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

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

```
});
```



Abstract code representation



AST

– conditions



control flow

– variable dependencies



data flow


– variable values



pointer analysis

# Per-Component JS Code Abstraction

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

A blue curved arrow originates from the text 'event.data' in the function call 'eval(event.data);' and points to the 'eval' function name. The word 'data' is written in blue below the arrow's path.

Abstract code representation



AST

– conditions



control flow

– variable dependencies



data flow

– variable values



pointer analysis

# Per-Component JS Code Abstraction

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

The diagram illustrates data and control flow in the provided JavaScript code. A blue arrow labeled 'data' points from the `event.data` property access to the argument of the `eval` function. A green arrow labeled 'True' points from the `if (1 === 1)` condition to the `eval` function, indicating that the condition is always true.

Abstract code representation



✓ AST

– conditions



✓ control flow

– variable dependencies



✓ data flow

– variable values



pointer analysis

# Per-Component JS Code Abstraction

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

The diagram illustrates code abstraction for the provided JavaScript code. A green arrow labeled "True" points from the condition `1 === 1` to the function call `window["e" + "val"]`, indicating that the condition is always true. A red bracket labeled "eval" is positioned under the string `"e" + "val"`, suggesting an evaluation or simplification of this expression. A blue arrow labeled "data" points from `event.data` to the function call, indicating the data flow.

Abstract code representation



AST

– conditions



control flow

– variable dependencies



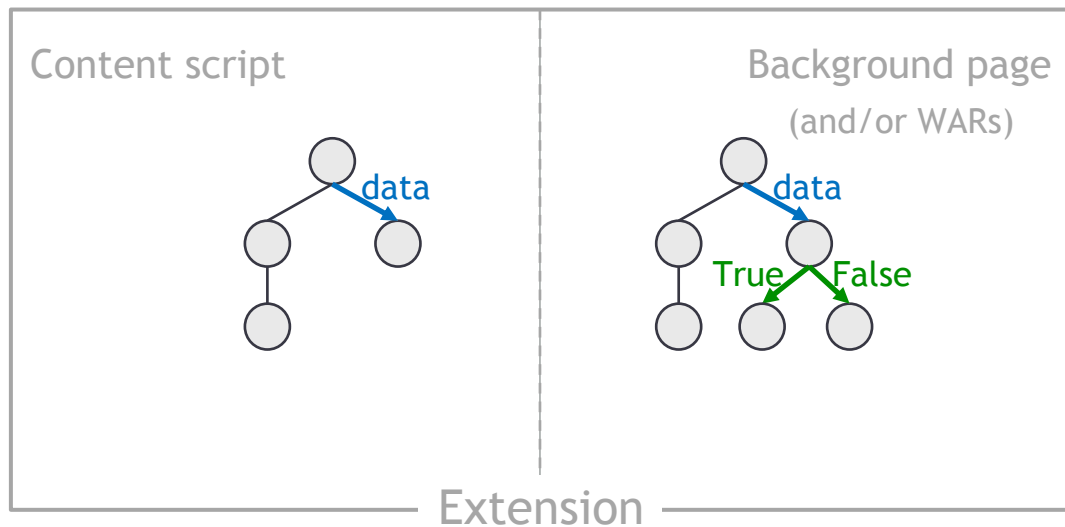
data flow

– variable values



pointer analysis

# DOUBLEX: Suspicious Data Flow Detection



Per-component JS code abstraction

Extension Dependence Graph (EDG)

Suspicious data flow tracking

Data flow report



# Extension Dependence Graph

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

The diagram illustrates the extension dependence graph for the provided code. It highlights the following dependencies:

- A green arrow labeled "True" points from the condition `if (1 === 1)` to the function body.
- A red bracket labeled "eval" is positioned under the string expression `"e" + "val"`, indicating a dependency on the `eval` extension.
- A blue arrow labeled "data" points from `event.data` to the function parameter `event`, indicating a dependency on the `data` extension.

- external messages
- internal messages

# Extension Dependence Graph

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



- 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});
```

message

```
// 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) {  
  })
```

message



– external messages

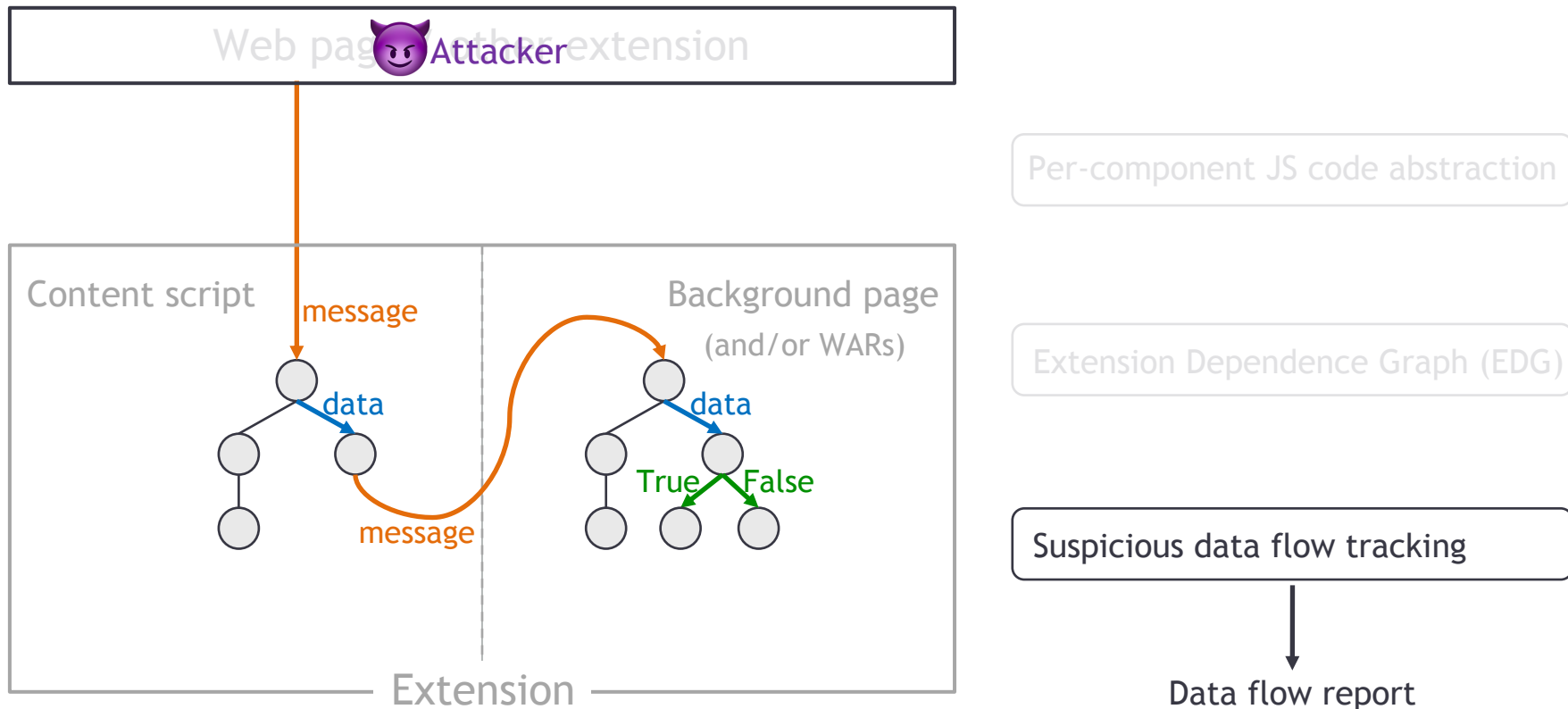


– internal messages



➤ Models message interaction within and outside of an extension

# DOUBLEX: Suspicious Data Flow Detection



# 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 data flow in the provided code. A blue arrow originates from the `event` parameter in the function signature on line 2, passes through a devil emoji, and points to `event.data` on line 4. A red bracket underlines the expression `"e" + "val"` on line 4, with a label `eval` below it. A green arrow points from the `if` statement on line 3 to the `True` label on line 4.



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

# Large-Scale Analysis of Chrome Extensions

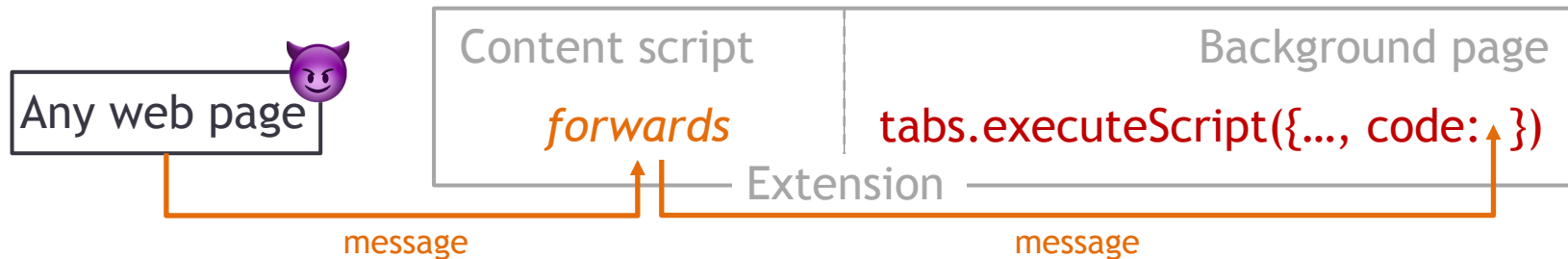
- Analyzed 155k Chrome extensions from 2021 with DOUBLEX
  - 278 suspicious extensions reported (309 suspicious data flows)
    - manual review
    - **precision: 89%** verified dangerous data flows (275 / 309)

Attacker capabilities	#Reports	#Verified data flow	#Exploitable
Code Execution	113	102	63
Triggering Downloads	21	21	21
Cross-Origin Requests	95	75	49
Data Exfiltration	80	77	76
Sum	309	275	209

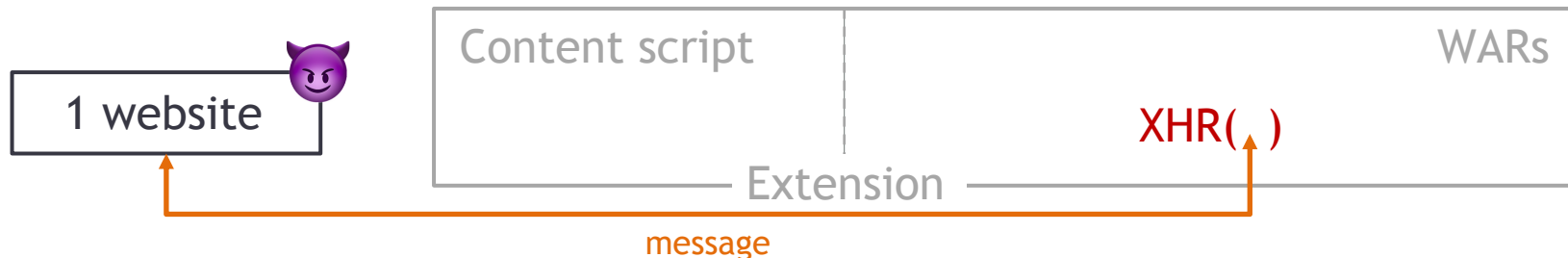


# Case Studies of Vulnerable Chrome Extensions

- Arbitrary code execution (*cdi...*, 4k+ users)



- Cross-origin requests (*koh...*, 200k+ users)



# Large-Scale Analysis of Chrome Extensions

- Analyzed 155k Chrome extensions from 2021 with DOUBLEX
  - 278 suspicious extensions reported
    - manual review
    - **precision: 89%** verified dangerous data flows
  - **184 confirmed vulnerable extensions**
    - 36% can be exploited by *any* websites or extensions
    - 2.4 - 2.9 million users impacted
- Analyzed known vulnerable extensions\* with DOUBLEX
  - **recall: 93%** of known vulnerabilities are detected (151 / 163)

# 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 Store anymore: 30 / 193
    - vulnerability fixed: 3 / 193
    - turned vulnerable: 5 / 184
    - new vulnerable: 19 / 184
- **still vulnerable: 160 (87%!)   ➤ Need to prevent vulnerable extensions from entering the Store → DOUBLEX**

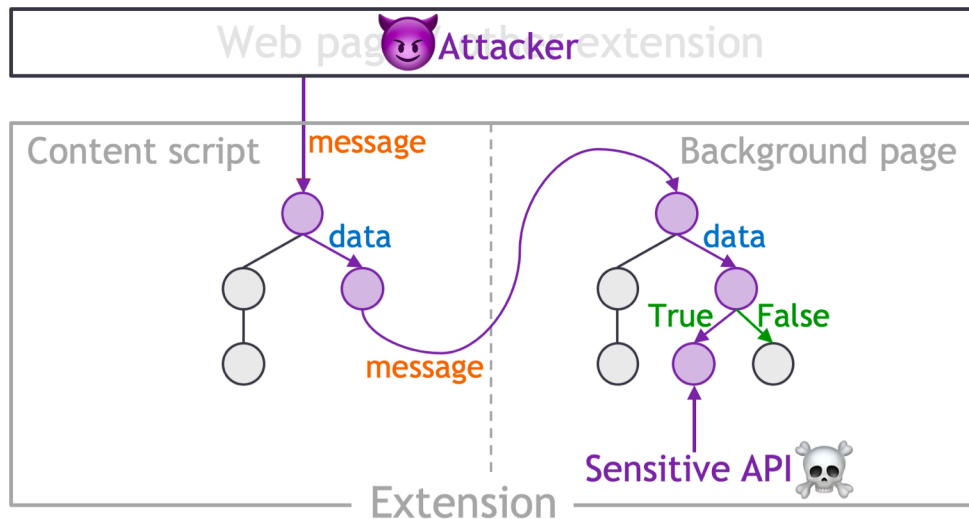
# Take-Away

unintentionally

Extensions are popular... but may introduce security & privacy threats

→ Because highly privileged

→ Due to their communication with websites / other extensions



## DOUBLEX: detects suspicious data flows in extensions

- 184 vulnerable extensions (87% already vulnerable the year before)
- Precision: 89%
- Recall: 93%



Aurore54F/DoubleX