

# FuzzOrigin: Detecting UXSS vulnerabilities in Browsers through Origin Fuzzing

Sunwoo Kim\*, Young Min Kim, Jaewon Hur  
Suhwan Song, Gwangmu Lee<sup>^</sup>, Byoungyoung Lee

\* Samsung Research

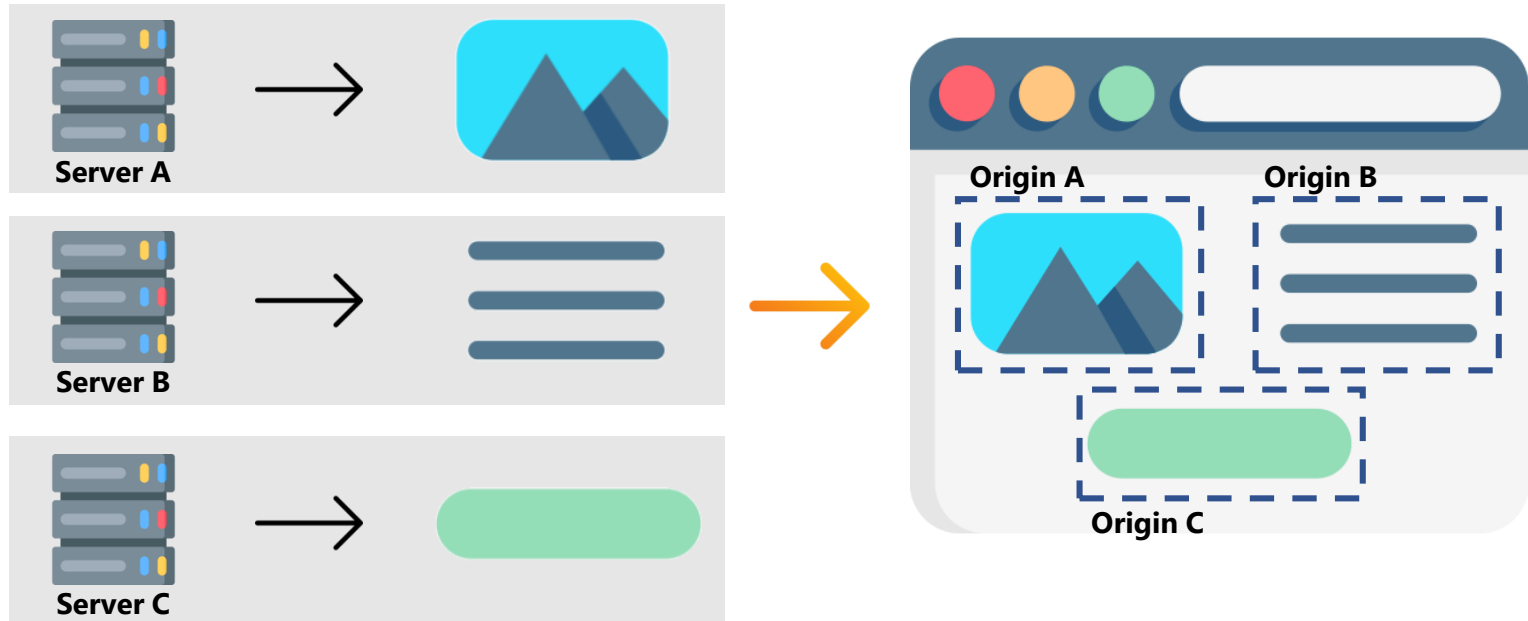


서울대학교  
SEOUL NATIONAL UNIVERSITY

<sup>^</sup> EPFL

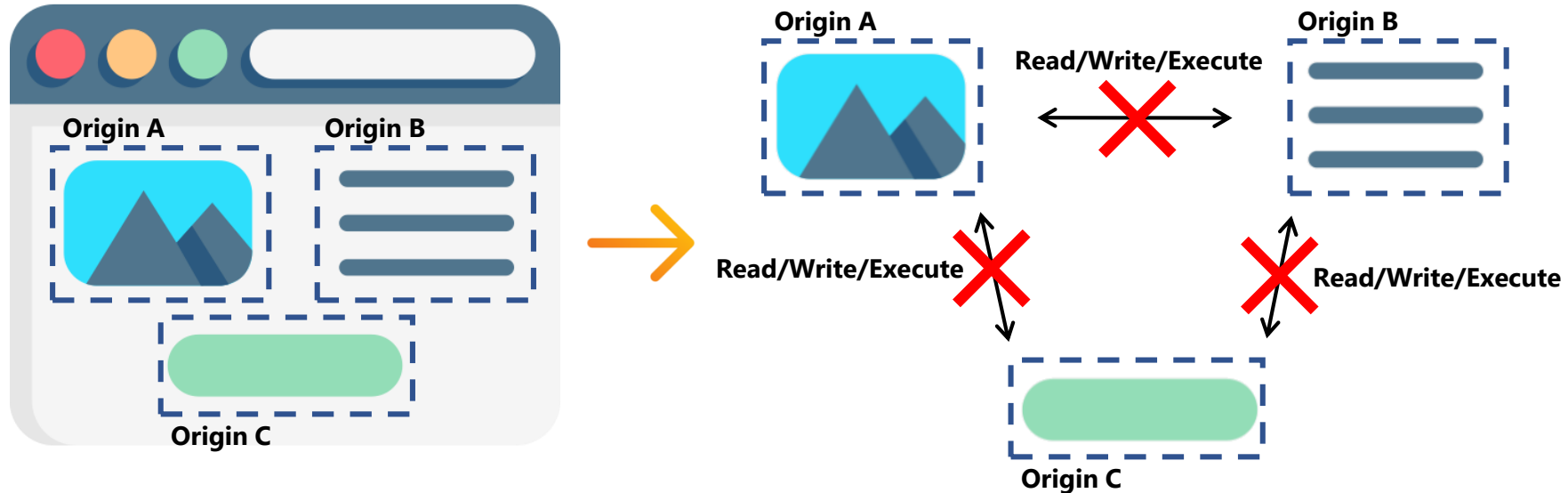
# What is Origin

- **Origin** is defined as a tuple of **scheme**, **host** and **port**.



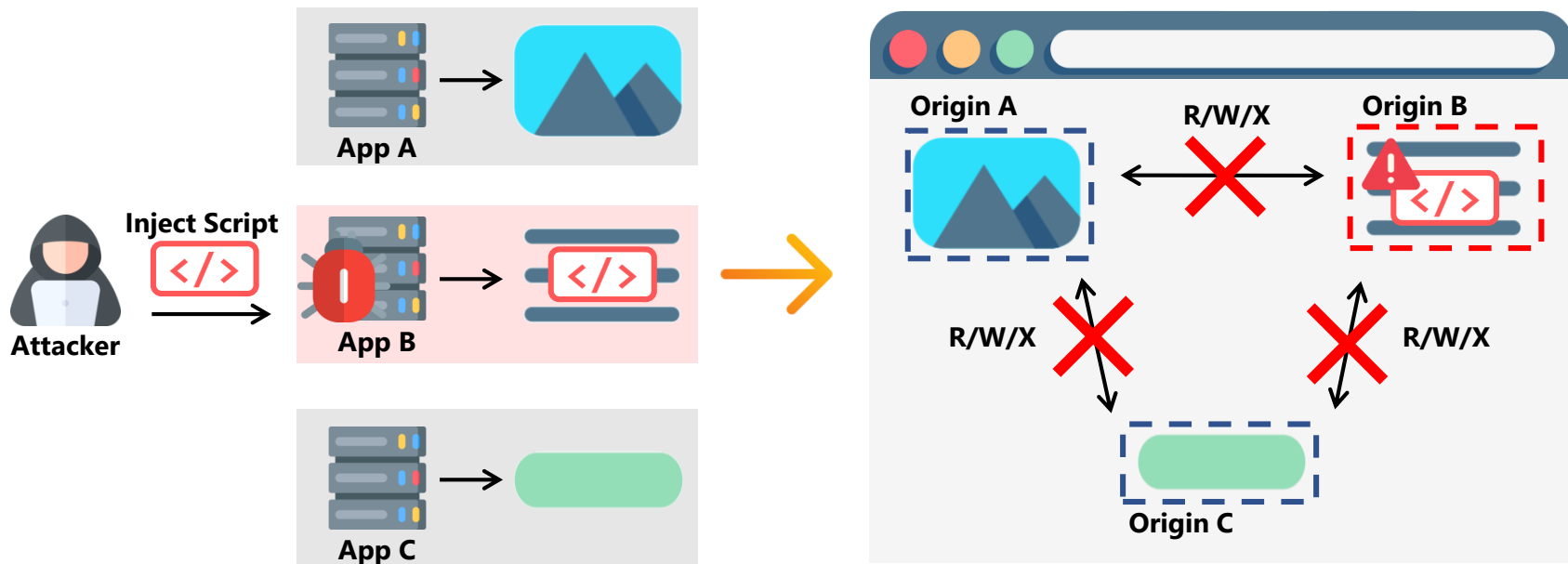
# Why Same-Origin Policy is important

- Same-origin policy constitutes a fundamental security mechanism



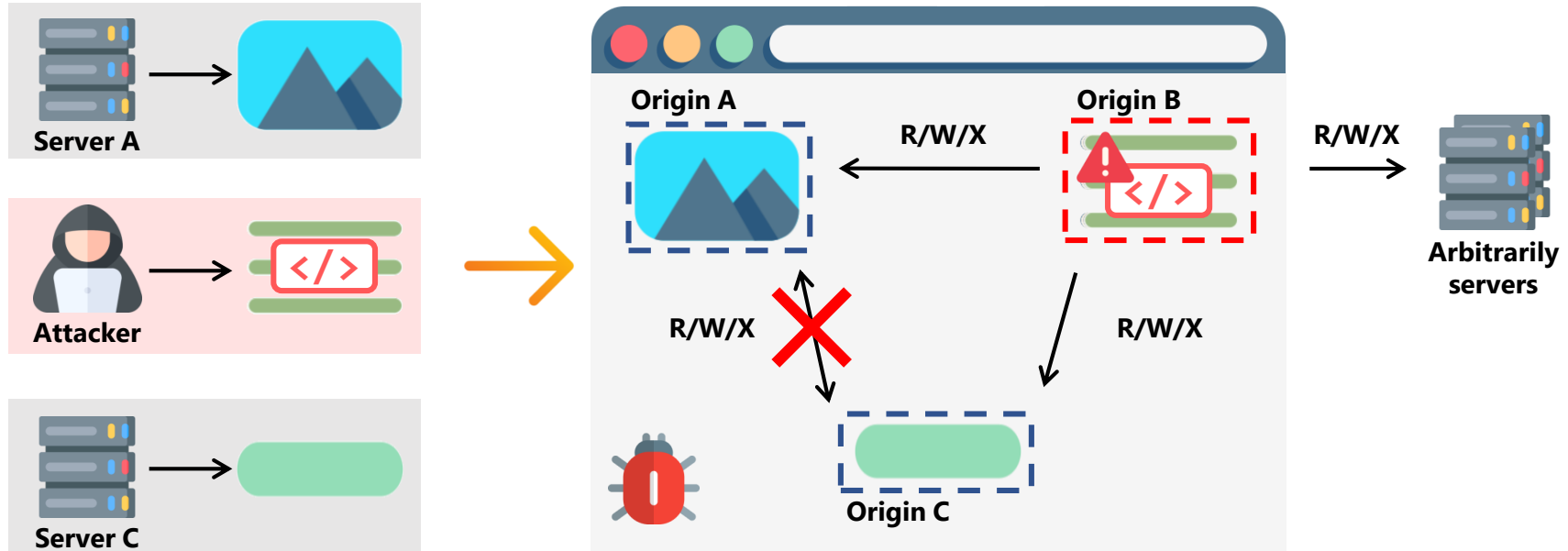
# Cross-Site Scripting

- **Cross-site scripting (XSS)** is a security vulnerability in **web application** that allows an attacker to inject client-side scripts into web pages.



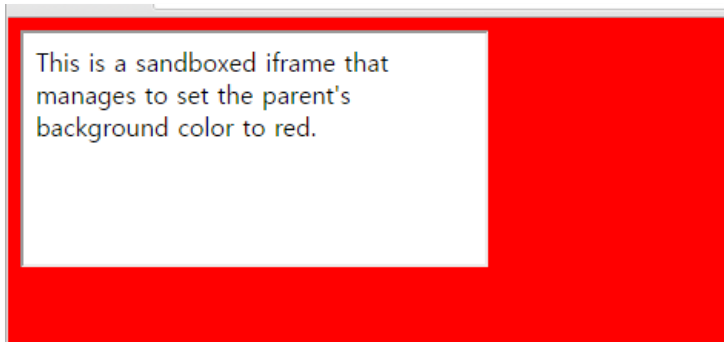
# Universal Cross-Site Scripting

- Universal cross-site scripting (UXSS) is a security vulnerability in web browsers that allows an attacker to run scripts into other web pages.

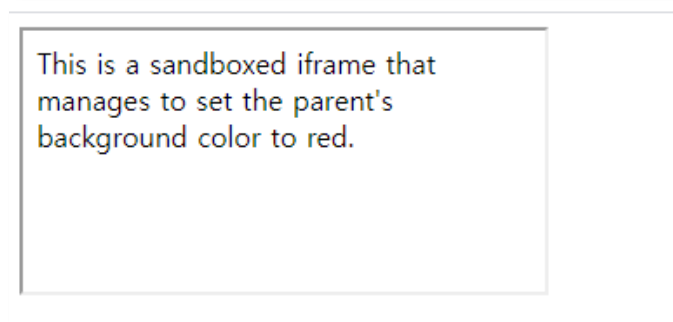


# UXSS Example (CVE-2015-1286)

- `parentFunction("document.body.style.backgroundColor = 'red';")();`

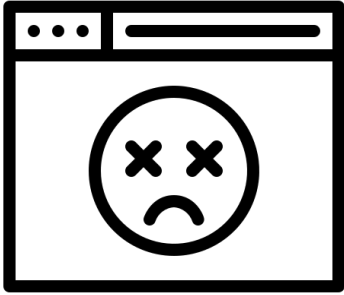


**Old(43.0.2357.0)**



**Current**

# Finding UXSS vulnerabilities is challenging

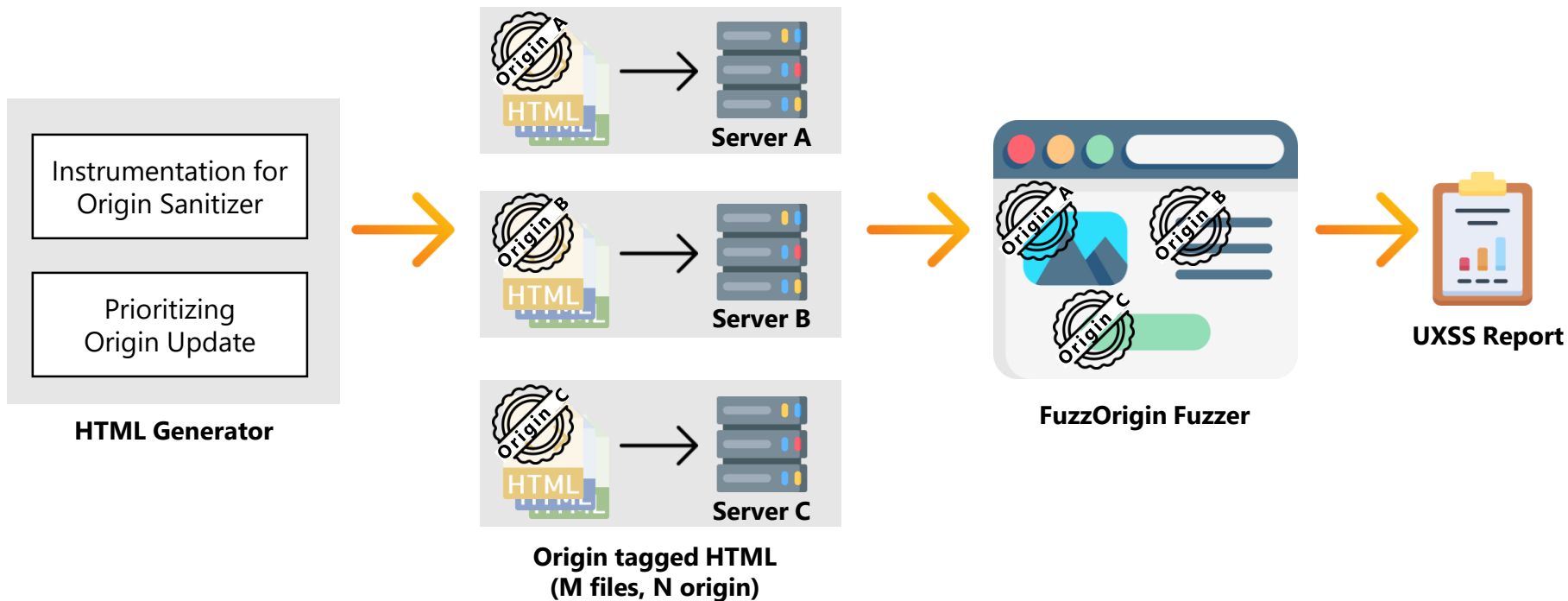


Memory corruption can be found by **crash**.

But UXSS has **no crash**. Need to check **origin**.



# FuzzOrigin Overview





# Challenges of UXSS Fuzzing

- Difficult to track the origin of **dynamically generated** JavaScript.

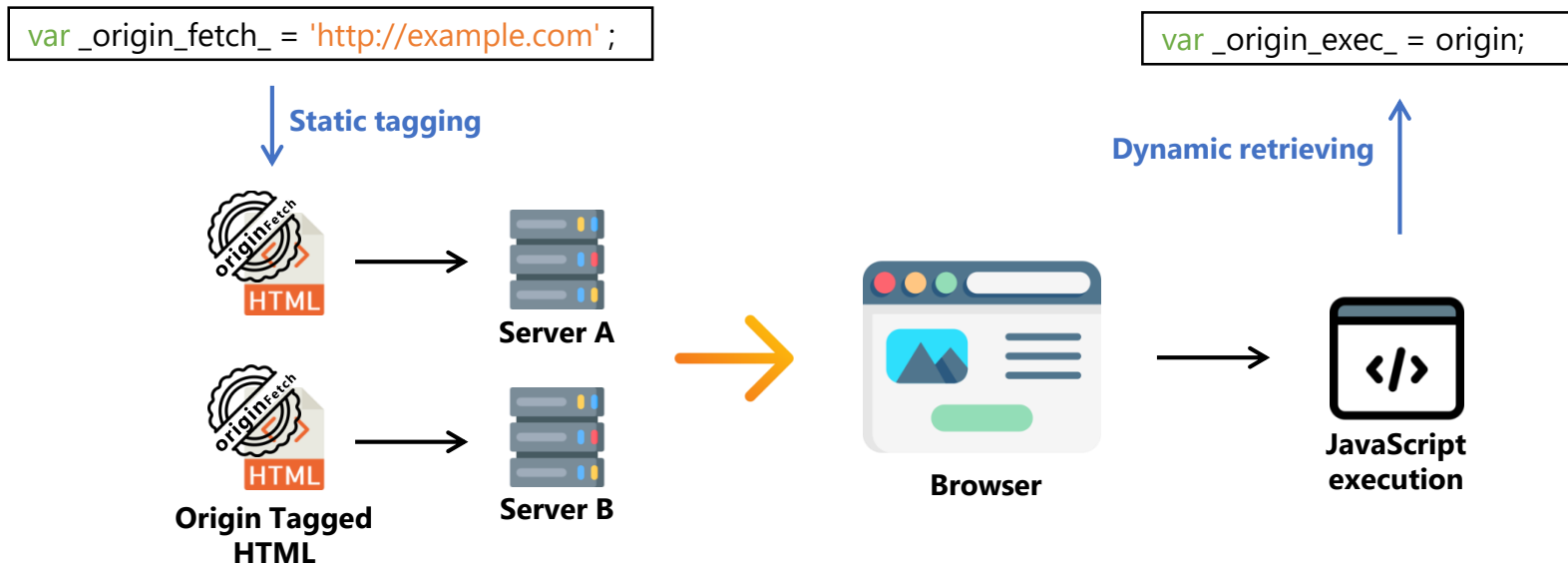


- Document is **dynamically changed** via HTML and JavaScript.



# Origin Tagging and Retrieving

- FuzzOrigin tags `originFetch` inside script as **string variable** and retrieves `originExec` by reading the **origin property** in JavaScript.



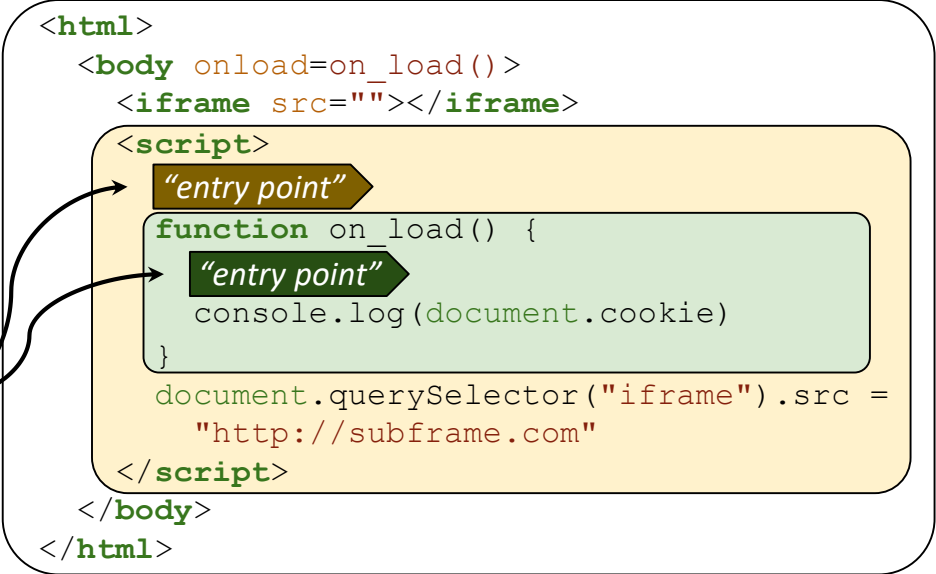
# Checking Origin Violation

- FuzzOrigin checks the origin violation for all possible entry points of script code execution.

## Origin Sanitizer

```
var _origin_fetch_ = 'http://example.com';
var _origin_exec_ = origin;
if (check_origin_violation(_origin_fetch_,
                           _origin_exec_)) {
    report_origin_violation();
}
```

```
<html>
  <body onload=on_load()>
    <iframe src=""></iframe>
    <script>
      "entry point"
      function on_load() {
        "entry point"
        console.log(document.cookie)
      }
      document.querySelector("iframe").src =
        "http://subframe.com"
    </script>
  </body>
</html>
```

The diagram illustrates the execution flow of the provided HTML code. It starts with the `<body onload=on_load()>` attribute, which triggers the execution of the `on_load()` function. This function is defined within a `<script>` tag. The code then proceeds to set the `src` attribute of an `<iframe>` element to `http://subframe.com`. Two callout boxes labeled "entry point" with arrowheads point to the `on_load()` function definition and the `document.querySelector("iframe").src = "http://subframe.com"` line, indicating these as key execution points for origin checking.

# Prioritizing Origin Update

- FuzzOrigin tries to trigger [cross-origin navigation](#) to create UXSS vulnerabilities.

Main page's  
cross-origin navigation



Sub page's  
cross-origin navigation



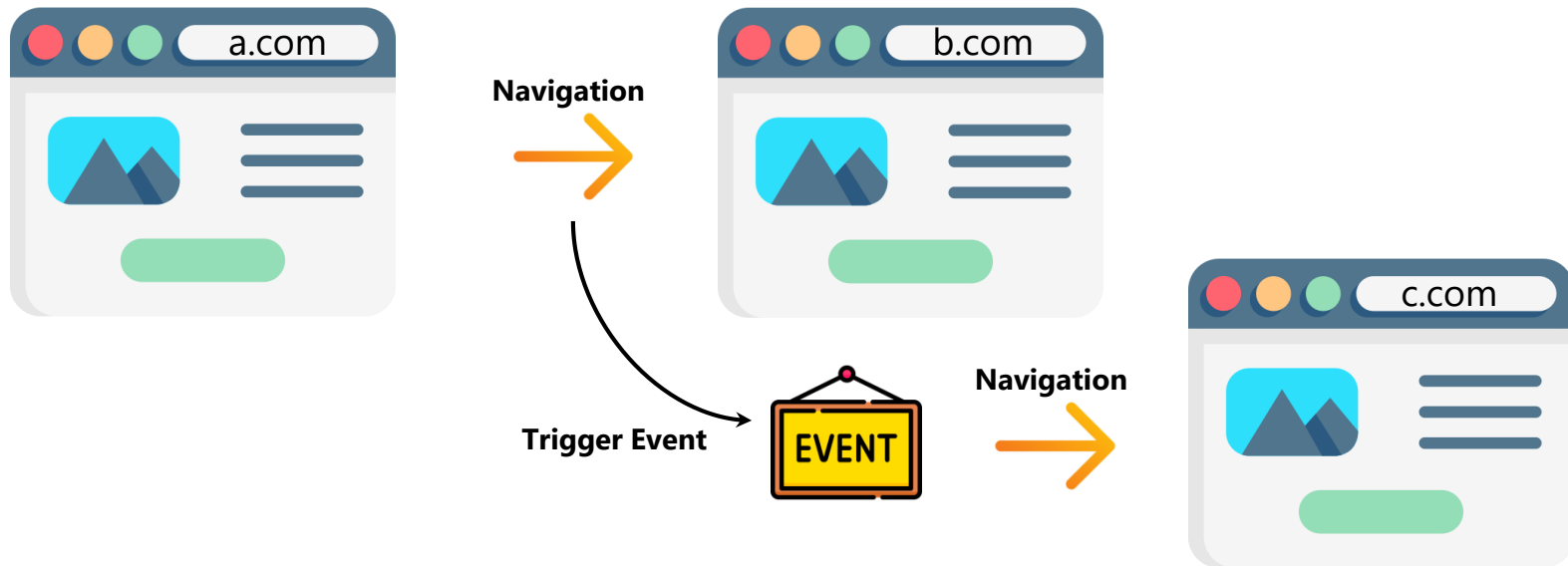
# Raising Cross-Origin Navigation

- FuzzOrigin considers using **various navigation APIs** and specifying **cross-origin navigation targets**.

Navigation APIs	Type	Generation	Target URL	Triggering Action	Dispatched Event
a.href=URL	Attribute	HTML/JavaScript	O	Click	beforeunload, unload, DOMContentLoaded, load
from.action=URL	Attribute	HTML/JavaScript	O	Submit	beforeunload, unload, DOMContentLoaded, load
iframe.src=URL	Attribute	HTML/JavaScript	O	-	DOMContentLoaded, load
history.forward()	Method	JavaScript	X	-	beforeunload, unload, DOMContentLoaded, load
history.backward()	Method	JavaScript	X	-	beforeunload, unload, DOMContentLoaded, load
history.replaceState(state, title, URL)	Method	JavaScript	O	-	beforeunload, unload, DOMContentLoaded, load
location.replace(URL)	Method	JavaScript	O	-	beforeunload, unload, DOMContentLoaded, load
location.reload()	Method	JavaScript	X	-	beforeunload, unload, DOMContentLoaded, load
window.open(URL)	Method	JavaScript	O	-	beforeunload, unload, DOMContentLoaded, load

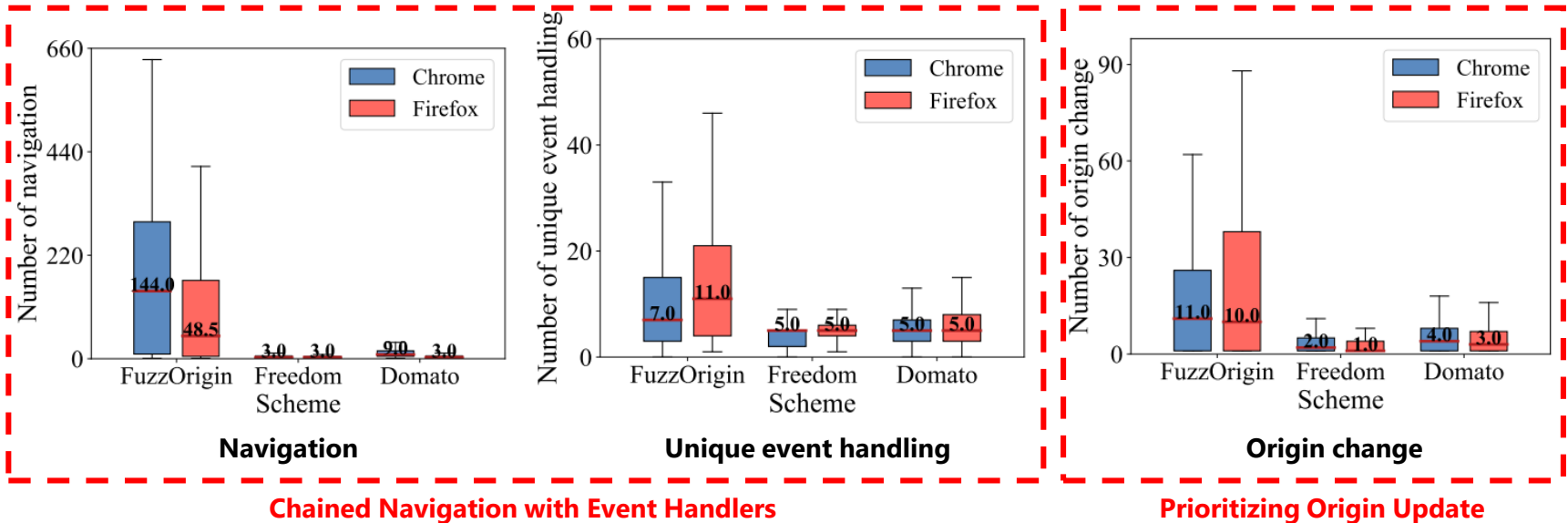
# Chained Navigation with Event Handlers

- Navigation trigger navigation related event handler and **event handler create new navigation.**



# Effectiveness of Chained-Navigation

- FuzzOrigin achieved the highest number in navigation, unique event handling and origin change.



# New UXSS vulnerabilities

- We found **four** new UXSS vulnerabilities.

Browser	Version	Bug ID	Description	Severity	Status
Chrome	96.0.4664	Issue #1280083	document.domain used in parent and child causes the origin (port) change	Low	Confirmed
Firefox	94.0b2	Issue #1741327	document.domain used in parent and child, causes script execution even if src of child window to the parent's origin	Serious	Confirmed
	94.0b2	Issue #1727480	History manipulation causes navigation to other pages on nsDocShell.	Serious	Confirmed
	94.0b9	CVE-2021-43536	Under certain circumstances, asynchronous functions could have caused navigation to fail but expose the target URL.	High	Patched



# Conclusion

- This paper presented FuzzOrigin, the first UXSS fuzzing framework.
- FuzzOrigin proposed a new UXSS detector, the origin sanitizer, as well as a new UXSS-focused HTML generation method.
- FuzzOrigin identified four new UXSS vulnerabilities.