

2025.10.03

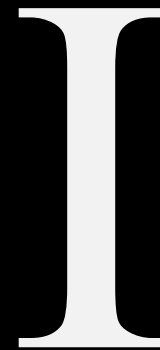
XSS-2

가디언 웹 보안 세미나 3

임준서

Security Mitigations

SOP, CORS, CSP



XSS가 왜 필요했을까?

Background

공격자가 피싱 사이트를 찔다고 하자

들어오는 사람의 브라우저의 모든 데이터 접근?

당연히 불가!

SOP가 있기 때문

Same Origin Policy

Concepts

Origin이란

`scheme://host:port`

같은 origin의 데이터만 접근 가능

DOM, Cookie, LocalStorage, Response, ... 차단

Same Origin Policy

Why?

XSS 발생

evil js code → client data로 bank 접속

SameSite=None 가정

user session을 이용해 요청을 대신 보냄

Same Origin Policy

Example

```
fetch("https://url.com")
```

요청은 문제 없이 보내진다.

1) origin = https://url.com

response 접근 허용

2) origin = https://diffurl.com

response 접근 불가

Same Origin Policy

Example

```
>> var res = await fetch("https://naver.com")
```

❗ Cross-Origin Request Blocked: The Same Origin Policy disallows reading the remote resource at <https://naver.com/>. (Reason: CORS header 'Access-Control-Allow-Origin' missing). Status code: 301. [\[Learn More\]](#)

❗ Cross-Origin Request Blocked: The Same Origin Policy disallows reading the remote resource at <https://naver.com/>. (Reason: CORS request did not succeed). Status code: (null). [\[Learn More\]](#)

❗ Uncaught (in promise) TypeError: NetworkError when attempting to fetch resource.

SOP로 인해 naver.com request 후 에러가 발생함
request는 전달된 상황

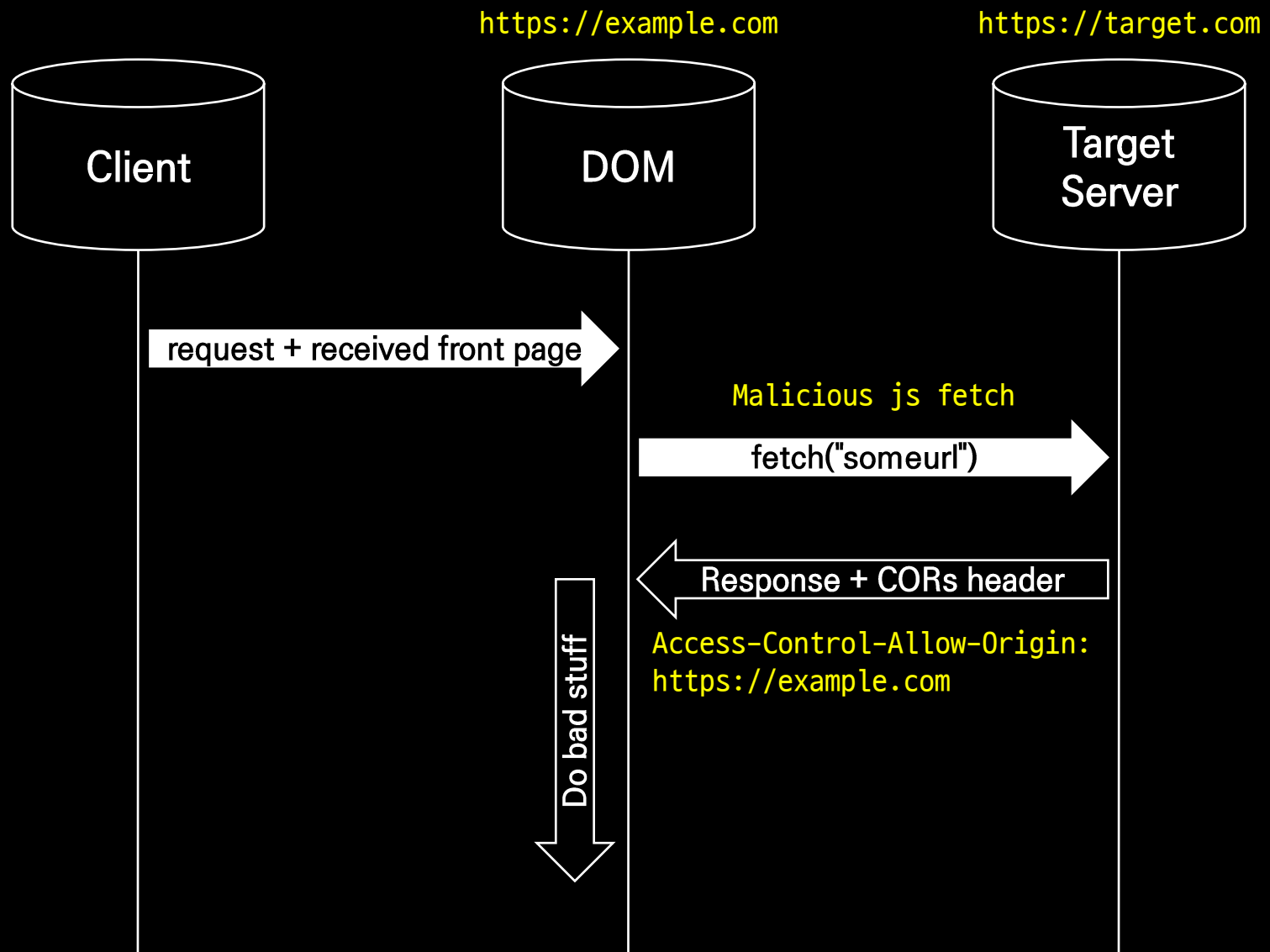
Cross-Origin Resource Sharing

Concept

모든 리소스를 거부하면 웹 개발 어려움

일부 허용된 origin은 접근 허용

Response를 보내는 서버에서 설정



잠시 XSS로 넘어와서...

Content Security Policy

혹시라도 XSS 방지가 뚫렸다면?

그리고 filter 말고 더 좋은 방법은 없을까?

Content Security Policy

CSP

"안전한" Content 만 허용하자

기본 정책 + 각 리소스별 신뢰하는 origin

Content Security Policy

How?

HTTP 헤더 "Content-Secure-Policy"

Content-Security-Policy: script-src 'self' https://apis.google.com

script-src :	js를 로드할 때
'self' :	같은 origin만 허용
apis.google.com :	추가로 이 origin까지 허용

Content Secure Policy

Nonce = Number once

script 조작 방지용 값

```
Content-Security-Policy: script-src 'self' 'nonce-ABC'
```

```
<script nonce="ABC">  
// some safe script  
</script>
```

nonce가 다르면 XSS 가능성 존재

실행하지 않는다.

여기서 중요한 점은 nonce는 다시 쓰이면 안 된다는 것이다.

XSS-2

Bypassing protections

II

Vulnerable filtering

substitution

```
_filter = ["script", "on", "javascript:"]  
for f in _filter:  
    if f in text.lower():  
        text = text.replace(f, "")
```

안전한가요?

"white list를 써야만 하는 이유들"

Vulnerable filtering

substitution

```
_filter = ["script", "on", "javascript:"]  
for f in _filter:  
    if f in text.lower():  
        text = text.replace(f, "")
```

```
text = "scscriptript"
```


Vulnerable filtering

substitution

```
_filter = ["script", "on", "javascript:"]  
for f in _filter:  
    if f in text.lower():  
        text = text.replace(f, "")
```

text = "sc~~scrip~~tscript"

Vulnerable filtering

substitution

```
_filter = ["script", "on", "javascript:"]  
for f in _filter:  
    if f in text.lower():  
        text = text.replace(f, "")
```

```
text = "script"
```

HTML, Javascript Facts

About html

HTML은 대소문자 구분이 없다

`<object data="data:text/html, <some url encoded tag>">`

URI parsing

HTML, Javascript Facts

About html normalization

1. Tokenization

태그, 속성, 주석 분류

2. DOM tree construction

3. Additional Normalization

HTML, Javascript Facts

1. Tokenization

엔티티 변환

\x00 치환

HTML, Javascript Facts

2. DOM Tree Construction

대소문자 => 소문자 통일

태그 오류 정정

HTML, Javascript Facts

3. Additional Normalization

URL 정규화

XSS 통로 중 하나

이 과정에서 URL의 공백류 제거

\t, \x01 등

HTML, Javascript Facts

About html entity encoding

특수문자를 표현하기 위한 인코딩

〈, 〉, & 같은 건 태그로 해석될 수 있음

1. Named Entity : <, >

2. Numeric Entity : A, A

HTML, Javascript Facts

About js

속성은 `document.location`, `document["location"]` 둘 다 가능하다

`document["locatio" + "n"]` 처럼 blacklist 우회

URL 정규화 `new URL("ja\4\tvascrRiPT:alert(1)", document.baseURI).href`

Unicode escape sequence: `\u0041` = 'A'

`console.log("\u0041")`

`String.fromCharCode(65,66,67)`

`1..toString`

HTML, Javascript Facts

About js

```
foo(x), foo`x`, function(foo(x))()
```

Escape seq `"\x41\x42"`

```
"alert(document.cookie)" instanceof { [Symbol.hasInstance]: eval };
```

```
Object.prototype.at = "polluted!";
```

property 탐색 순위: 자신 > prototype > 상위 prototype > ...

HTML, Javascript Facts

exploitable functionality

리소스의 on~~~ 꼴 속성들은 js를 실행시킬 수 있다.

onfocus, onerror, onload, ... video source로도 가능

iframe의 srcdoc은 **HTML**로 들어가기에 HTML인코딩을 통한 우회 가능

url, src 등은 scheme 지정 가능 → src="javascript: alert(1)"

document.body.innerHTML

script는 바로 실행되지 않는다. Event handler 필요

eval은 chrome에서는 막히기도 한다.

Other Facts

About Regex (Regular Expression)

정규식 : 문자열 패턴 찾기

좌측으로 0.5cm 떨어진 내용 하단의 설명 18px

$^[a-z]\{3,16\}$ \$

^는 시작, \$는 끝

인터넷에 리소스가 많으니, 처음 본다면 공부해두자

Guardian 2025

Other Facts

Function misuse

간혹 함수의 용도를 벗어난 경우 존재

→ 취약점이 될 수도 있다.

e.g. regex 객체의 잘못된 재사용

Other Facts

Data

Data URI를 활용한 XSS

Base64 로 인코딩 > DOM 로드 시 실행

filtering 우회

reference: www.hahwul.com/blog/2022/data-uri-xss-v2/
datatracker.ietf.org/doc/html/rfc2397

Guardian 2025

Other Facts

srcdoc

iframe의 srcdoc

Multiple encoding possible

Homework [XSS Filtering Bypass]

dreamhack.io/wargame/challenges/433

Homework [XSS Filtering Bypass Advanced]

dreamhack.io/wargame/challenges/434

2025.10.03

Q&A

질문이 있다면 하십시오

임준서

Content Secure Policy

Nouce = Number used once

혹시라도 XSS 방지가 뚫렸다면?

좌측으로 0.5cm 떨어진 내용 하단의 설명 18px

더 좋은 방법은 없을까?

좌측으로 0.5cm 떨어진 내용 하단의 설명 18px

3.5cm 떨어진 내용 3 32px

좌측으로 0.5cm 떨어진 내용 하단의 설명 18px

1cm-1cm 떨어진 주석 12px

1cm-1cm 떨어진 주석 12px

2.5cm-3.5cm 떨어진 제목 36px

제목 하단의 부제목 18px

3.5cm 떨어진 내용 1 32px

```
Git init
Git status
Git add text.txt
Git add .
Git commit
Ctrl+C
Git commit -m "genesis"
Git log
Git log --oneline

_filter = ["script", "on",
"javascript:"]
for f in _filter:
    if f in text.lower():
        text =
text.replace(f, "")
```

1cm-1cm 떨어진 주석 12px

1cm-1cm 떨어진 주석 12px

