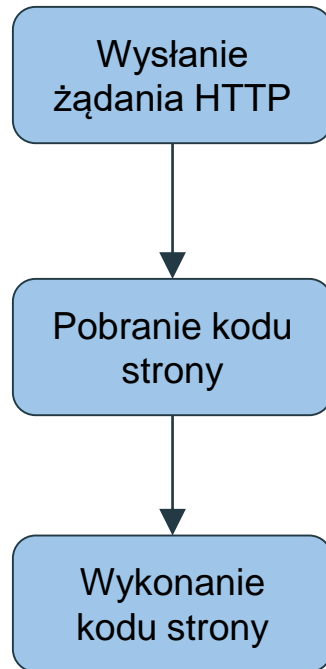


Bezpieczeństwo frontendu

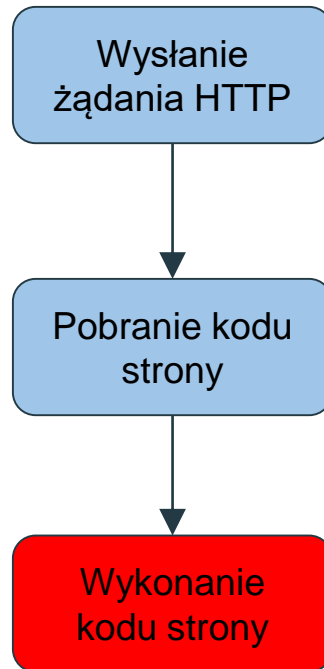
- JS
 - Angular
 - Node.js



Bezpieczeństwo frontendu

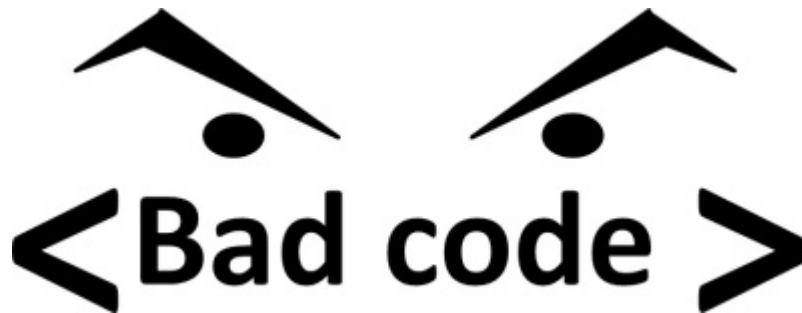
Co może pójść nie tak?

- Wykonanie kodu w przeglądarce klienta.



Bezpieczeństwo frontendu

- A3 – Cross-Site Scripting (XSS)
- V11: HTTP security configuration verification requirements
- Błędy implementacji:
 - Umożliwienie osadzenia strony w ramce.
 - Umożliwienie wykonania dowolnego kodu w przeglądarce użytkownika.
 - Dostęp do danych użytkownika, np. ciasteczek.



Osadzanie ramek



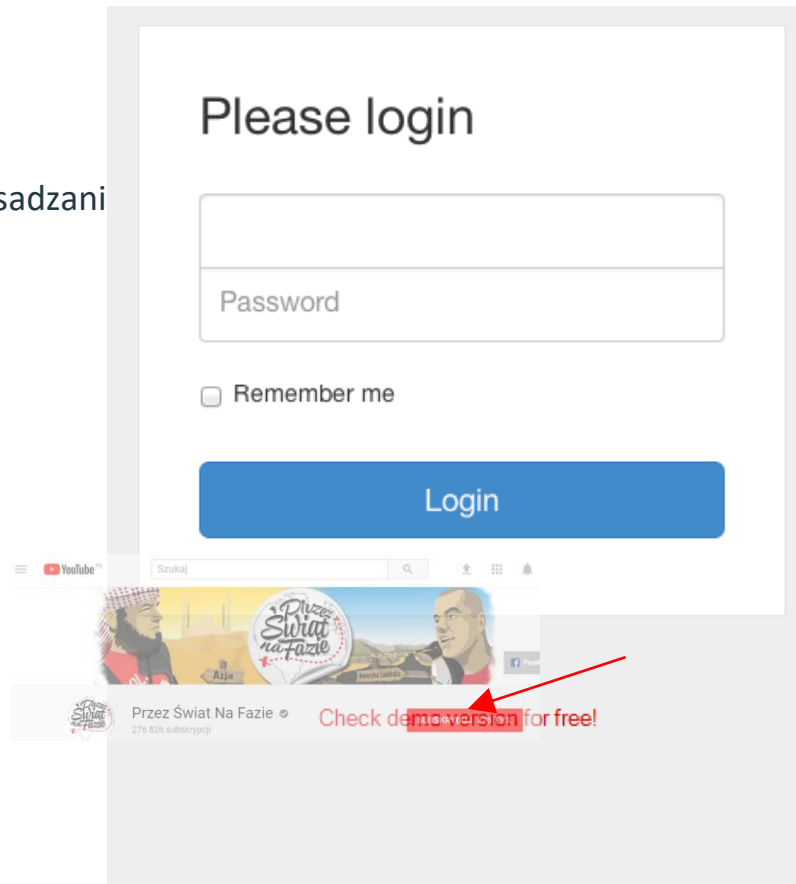
Jaka jest domena tej strony?

<http://www.wykop.pl/ramka/3925079/wojny-parkingowe-albo-bedzie-wygodnie-albo-tanio/>

```
<iframe frameborder="0" src="http://www.bankier.pl/wiadomosc/Wojny-parkingowe-Albo-bedzie-wygodnie-albo-tanio-7543005.html" id="view-frame">
    Twoja przeglądarka nie obsługuje ramek pływających.
</iframe>
```

Osadzanie ramek

- Do czego może doprowadzić zezwalenie na osadzanie
- Clickjacking



Clickjacking

- Dlaczego to jest niebezpieczne?
- Nie mamy kontroli nad tym, co będzie na stronie atakującego. Ma on pełny wachlarz możliwości.
- Jakie są ograniczenia ataku?
- Ofiara musi być zalogowana. Kto się wylogowuje ze wszystkich serwisów (Facebook, Youtube)?
- Na stronie możemy wykonać proste akcje, które ofiara wykona na stronie atakującego.
- Jak się zabezpieczyć?
- Nagłówek X-Frame-Options w odpowiedzi.

Clickjacking

- **DENY**

Żadna strona nie może osadzać w ramce mojej strony.

- **SAMEORIGIN**

Zezwala na osadzenie mojej strony na stronach tym samym: protokole, hoście, porcie.

- **ALLOW-FROM uri**

Zezwala na osadzenie mojej strony na stronie o podanym: protokole, hoście, porcie.

The screenshot shows a web browser's developer tools with the Network tab selected. A list of requests is shown on the left, with 'www.pyszne.pl' selected. The right pane shows the 'Response Headers' for this request. The 'x-frame-options' header is set to 'SAMEORIGIN', which is highlighted with a red rectangular box. Other visible headers include 'cf-ray', 'content-encoding', 'content-type', 'date', 'last-modified', 'server', 'status', and 'vary'.

Name	Headers	Preview	Response	Cookies	Timing
www.pyszne.pl	Referrer Policy: no-referrer-when-downgrade				
js?v=3.27&client=gme-takeav	▼ Response Headers				
takeaway.js?3006841259	cf-ray: 3a03be9a087c6433-FRA				
2234540350.js	content-encoding: gzip				
sdk.js	content-type: text/html				
pyszne.png	date: Mon, 18 Sep 2017 10:50:28 GMT				
android.png	last-modified: Mon, 18 Sep 2017 10:50:21 GMT				
ios.png	server: cloudflare-nginx				
blog_180.png	status: 200				
	vary: Accept-Encoding				
	x-clacks-overhead: GNU Terry Pratchett				
	x-frame-options: SAMEORIGIN				

58 requests | 38.7 KB transferred

X-Frame-Options

- Django

```
MIDDLEWARE = [  
    ...  
    'django.middleware.clickjacking.XFrameOptionsMiddleware',  
    ...  
]
```

- Apache

```
# /etc/apache2/sites-available/example.com.conf  
Header always append X-Frame-Options SAMEORIGIN  
Header set X-Frame-Options DENY  
Header set X-Frame-Options "ALLOW-FROM https://example.com/"
```

- nginx

```
#/etc/nginx/sites-available/default  
add_header X-Frame-Options SAMEORIGIN;
```


X-Frame-Options

- IIS

```
# Web.config file
<system.webServer>
  ...

  <httpProtocol>
    <customHeaders>
      <add name="X-Frame-Options" value="SAMEORIGIN" />
    </customHeaders>
  </httpProtocol>

  ...
</system.webServer>
```

Javascript

- Metody wprowadzania:

- Wczytanie skryptu JS z pliku.
- Wpisanie skryptu JS w kodzie HTML.

`<script src="..." />`

`<script>...</script>`

- Skrypt pochodzący z nieznanego źródła:

- odczytanie wrażliwych danych (ciasteczek, localStorage),
- modyfikacja zawartości strony (np. podmiana numerów kont itp.),
- instalowanie Trojanów,
- przekierowanie użytkownika na inną stronę.

Cross Site Scripting

The screenshot shows a web browser window with the address bar displaying the URL `https://blockchain.info/en/block-index/1160457/"><h1>XSS here`. The page content consists of a vertical list of 18 instances of the text `XSS here">` on the left side. On the right side, there are two highlighted boxes containing the payloads used for the attack: `"><h1>XSS here` (top) and `<h1>XSS here` (bottom). The browser's footer contains navigation links: [About Us & Contact](#), [Privacy Policy](#), [Terms of Service](#), and [Ok \(627 Nodes Connected\)](#), followed by an **Advanced: Enable** button and a dropdown menu currently set to `Bits (uBTC)`.

Cross Site Scripting

- Typy ataków XSS
 - Reflected

Dane przekazane w żądaniu HTTP są zwrócone w odpowiedzi HTTP.
 - Stored

Dane przekazane w żądaniu HTTP są zapisane na serwerze (np. w bazie danych) i są zwracane we wszystkich odpowiedziach HTTP.
 - DOM-based

Dane przekazane w żądaniu HTTP nie są obsługiwane przez serwer i nie są zwracane w odpowiedzi HTTP, jednakże oryginalny kod JS korzysta z nich po stronie klienta.
- Jaki można wymyślić scenariusz ataku, wykorzystujący XSS?

Cross Site Scripting

- Typy ataków XSS

Select your language:

```
<select><script>
document.write("<OPTION
value=1>" + document.location.href.substring(document.location.href.indexOf("default=") + 8) + "</OPTION>");
document.write("<OPTION value=2>English</OPTION>");
</script></select>
```

`http://www.some.site/page.html?default=<script>alert(document.cookie)</script>`

- Jaki można wymyślić scenariusz ataku, wykorzystujący XSS?

Cross Site Scripting - Samy

- Robak XSS z 2005r.
 - Napisany przez Samy Kamkar.
 - Uruchomiony w MySpace.
 - Przez 20 godzin rozpropagował się na milion kont!
 - Dodawał na profilu napis “but most of all, samy is my hero” i wysyłał zaproszenie do Samiego do grona znajomych.
 - Skrypt się uruchamiał po wejściu na profil Samiego.
-
- 3 lata zakazu dostępu do internetu.
 - 90 dni prac społecznych.
 - \$20k kary.









Cross Site Scripting - KNF

- Na stronie KNF wstrzyknięto złośliwy kod JS (wykorzystali dziure w serwerze JBOSS).
 - Kod selekcjonował osoby odwiedzające stronę KNF i infekował przeglądarki tylko pracowników banków.
 - Przekierowanie ofiary na stronę, która wykorzystuje znane luki we Flashu/Silverlightcie.
 - Zainfekowanie komputera ofiary.
 - Zero 0-dayów - wszystko znane.
-
- Jak się zabezpieczyć przed XSS?

Same Origin Policy

- Wbudowany w przeglądarki.
- Przeglądarka blokuje dostęp z poziomu skryptu JS do danych **innej** strony.

<http://www.example.com/dir/page.html>

URL	Rezultat	Powód
http://www.example.com/dir/page.html		Ten sam protokół, host, port
http://www.example.com/dir2/other.html		Ten sam protokół, host, port
http://www.example.com:81/dir/other.html	 	Inny port <i>(wyjątek: IE ignoruje port w SOP!)</i>
https://www.example.com/dir/other.html		Inny protokół
http://en.example.com/dir/other.html		Inny host
http://example.com/dir/other.html		Inny host (wymagany dokładnie ten sam)
http://v2.www.example.com/dir/other.html		Inny host (wymagany dokładnie ten sam)

Same Origin Policy

The screenshot illustrates a web browser window with the address bar showing `www.example.com`. The page content is a Google sign-in interface. Below the page, the browser's developer console is open, showing a JavaScript error. A red box highlights the console error, which is an uncaught DOMException. The error message states: "Failed to read the 'contentDocument' property from 'HTMLIFrameElement': Blocked a frame with origin 'http://www.example.com' from accessing a cross-origin frame." This error occurs because the script is attempting to access the `contentDocument` property of an `iframe` from a different origin, which is blocked by the Same Origin Policy.

www.example.com

Google

One account. All of Google.

Sign in with your Google Account

philippe.deryck@cs.kuleuven.be
philippe.deryck@cs.kuleuven.be

Sign in

Stay signed in Forgot password?

Sign in with a different account

One Google Account for everything Google

Elements Sources Network Timeline Profiles Resources Audits Console Security

<top frame>

```
document.querySelector("iframe").contentDocument.querySelector("input[type=password]").value
```

Uncaught DOMException: Failed to read the 'contentDocument' property from 'HTMLIFrameElement': Blocked a frame with origin "http://www.example.com" from accessing a cross-origin frame.

Cookie flags

- httpOnly - dostęp do ciasteczka jest zablokowany z poziomu JS.
- secure - ciasteczko ustawione w protokole https nie będzie wysyłane w protokole http.

```
Set-Cookie: JSESSIONID=AX765faftJqsxaxsASWe77;Domain=.example.com;Path=/;Expires=Sat, 27-Jun-2017 18:09:16 GMT;Secure;HttpOnly
```

X-XSS-Protection

- Przeglądarka blokuje wczytanie strony, gdy wykryje atak XSS Reflected.

```
X-XSS-Protection: 0  
X-XSS-Protection: 1  
X-XSS-Protection: 1; mode=block  
X-XSS-Protection: 1; report=<reporting-uri>
```

- 0 - wyłączona blokada,
- 1 - po wykryciu ataku dane są wycinane z odpowiedzi serwera (domyślna opcja w przeglądarkach),
- 1; mode=block - po wykryciu ataku przeglądarka blokuje wczytywanie strony,
- 1; report=<reporting-URI> (Chromium) - po wykryciu ataku dane są wycinane z odpowiedzi serwera, a raport z sytuacji jest wysyłany na podany adres.

X-XSS-Protection

- Jak włączyć nagłówek?

- PHP

```
header("X-XSS-Protection: 1; mode=block");
```

- Apache

```
<IfModule mod_headers.c>  
    Header set X-XSS-Protection "1; mode=block"  
</IfModule>
```

HTML Encoding

- Usunięcie z treści tagów HTML.
- Zamiana tagów na entities.

```
& --> &amp;  
< --> &lt;  
> --> &gt;  
" --> &quot;  
' --> &#x27;  
/ --> &#x2F;
```

- Twig

```
{{ user.username|escape('js') }}  
{{ user.username|escape }}
```

HTML Encoding

- Velocity

```
public static VelocityContext createContext()
{
    final VelocityContext context = new VelocityContext();

    context.put("esc", new EscapeTool());
    context.put("stringutils", new StringUtils());
    context.put("annotationutils", new AnnotationUtils());

    final IPreferenceStore store = WorkbenchCorePlugin.getDefault().getPreferenceStore();
    int maxFieldLength = store.getInt(PreferenceConstants.MAX_FIELD_LENGTH);
    if (maxFieldLength == 0) maxFieldLength = Integer.MAX_VALUE;
    context.put("maxFieldLength", maxFieldLength);

    return context;
}
```

HTML Encoding

- Velocity

```
{
# Toolbox Configuration Example

<tool>
  <key>esc</key>
  <scope>application</scope>
  <class>org.apache.velocity.tools.generic.EscapeTool</class>
</tool>
}
```

HTML Encoding

- Velocity

<pre>{ ... }</pre>	
java()	Escapes the characters in a String using Java String rules.
javascript()	Escapes the characters in a String using JavaScript String rules.
html()	Escapes the characters in a String using HTML entities.
url()	Escapes the characters in a String to be suitable to pass to an HTTP parameter value.
xml()	Escapes the characters in a String to be suitable to pass to an XML query.
sql()	Escapes the characters in a String to be suitable to pass to an SQL query.
<pre>}</pre>	

`$esc.html($html)`

Blacklisting

- Usuwanie konkretnych ciągów znaków (np. javascript).

`` -> ``

- Usuwanie konkretnych tagów (np. script).

`<script>alert('XSS')</script>` -> puste

- Usuwanie konkretnych atrybutów tagów (np. onclick).

`Click here` -> `Click here`

- **Blacklisting to zawsze zły pomysł.**

Bypass Blacklisting

- Usuwanie konkretnych ciągów znaków (np. javascript).

`` -> ``

```
javajavascriptscript:alert('XSS')
```

- Usuwanie konkretnych tagów (np. script).

`<script>alert('XSS')</script>` -> puste

```
<svg/onload=alert('XSS')>
```

- Usuwanie konkretnych atrybutów tagów (np. onclick).

`Click here` -> `Click here`

Jest 85 różnych atrybutów zdarzeniowych.

```
<a href="..." onkeyup="alert('XSS')">Click here</a>
```

- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

Content Security Policy

- Określa zaufane źródła zasobów (whitelisting).
- Zasoby:
 - skrypty,
 - style,
 - media,
 - ramki,
 - obrazki,
 - czcionki,
 - itd.

1) Default Source

2) Script Source

3) Style Source

4) Image Source

5) Font Source

6) Connect Source

7) Media Source

8) Object Source

9) Child Source

10) Frame Source

11) Worker Source

12) Frame Ancestors

13) Form Action

<https://report-uri.io>

Content Security Policy

- Jak można włączyć CSP?
 - Nagłówek HTTP Content-Security-Policy

```
Content-Security-Policy: default-src 'self' cdn.example.com; img-src img.example.com;  
script-src 'self';
```

- Meta tag Content-Security-Policy

```
<meta http-equiv="Content-Security-Policy" content="default-src 'self' cdn.example.com; img-  
src img.example.com; script-src 'self';"/>
```

Zasada: blokuj wszystko i zezwalaj wybrane.

Content Security Policy

- Wartości polityki
 - 'none' - całkowicie blokuje wykorzystanie zasobu tego typu,

```
Content-Security-Policy: script-src 'sha256-076c8f1ca6979ef156b510a121b69b6265011597557ca2971db5ad5a2743545f'
```

```
<script>var inline = 1;</script>
```

- 'unsafe-eval'- zezwala na wykorzystanie funkcji eval do generowania skryptów z napisów,
- 'nonce-<base64-value>' - zezwala na wykorzystanie zasobu opisanego noncem,
- '<hash-alg>-<hash>' - zezwala na wykorzystanie zasobu o podanym hashu (bez <script>),
- 'strict-dynamic' - zezwala na wykorzystanie zasobów wczytanych przez zaufany zasób,

“allows the execution of scripts dynamically added to the page, as long as they were loaded by a safe, already-trusted script”

```
data:text/html,<script>alert('hi');</script>
```

Content Security Policy

- <host-source>: domena, IP, protokół lub port, który określa zaufane źródło

```
http://*.example.com
```

```
mail.example.com:443
```

```
https://store.example.com
```

```
https: http:
```

```
*
```

Content Security Policy

- Raportowanie
 - report-uri <uri> - wysyła raport w formie JSON na podany adres (deprecated),

```
{
  "csp-report": {
    "document-uri": "http://example.org/page.html",
    "referrer": "http://attacker.com/attacker.html",
    "blocked-uri": "http://attacker.com/image.png",
    "violated-directive": "default-src 'self'",
    "effective-directive": "img-src",
    "original-policy": "default-src 'self'; report-uri http://example.org/csp-report"
  }
}
```

Content Security Policy

- www.example.com

```
default-src ,self; img-src mysite.com;
```

- Mogę załadować obrazek z <http://www.example.com/images/logo.png>?

```
default-src ,self; img-src 'localhost';
```

- Mogę załadować obrazek z <http://localhost/images/logo.png>?

```
default-src ,self; img-src localhost;
```

- Mogę załadować obrazek z <http://127.0.0.1/images/logo.png>?

```
default-src ,self; img-src https://othersite.com:*;
```

- Mogę załadować obrazek z <http://othersite.com/images/logo.png>?

OWASP XSS Prevention Cheat Sheet

- Never Insert Untrusted Data Except in Allowed Locations
- HTML Escape Before Inserting

```
<!--...NEVER PUT UNTRUSTED DATA HERE...-->
```

```
<div...NEVER PUT UNTRUSTED DATA HERE...</div>
```

```
<body>...ESCAPE UNTRUSTED DATA BEFORE PUTTING HERE...</body>
```

```
<div attr=...ESCAPE UNTRUSTED DATA BEFORE PUTTING HERE...>content</div>
```

```
<script>alert('...ESCAPE UNTRUSTED DATA BEFORE PUTTING HERE...')</script>
```

```
<script>...ESCAPE UNTRUSTED DATA BEFORE PUTTING HERE...</script>
```

```
<style>selector { property : ...ESCAPE UNTRUSTED DATA BEFORE PUTTING HERE...; } </style>
```

```
<style>selector { property : "...ESCAPE UNTRUSTED DATA BEFORE PUTTING HERE..."; } </style>
```

```
<a href="http://www.someSite.com/test"...ESCAPE UNTRUSTED DATA BEFORE PUTTING HERE...">
```

HtmlSanitizer - <https://github.com/mganss/HtmlSanitizer>

```
var sanitizer = new HtmlSanitizer();  
sanitizer.AllowedAttributes.Add("class");  
var sanitized = sanitizer.Sanitize(html);
```

[https://www.owasp.org/index.php/XSS_\(Cross_Site_Scripting\)_Prevention_Cheat_Sheet](https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet)

Sandboxing

Expression Sandboxing

AngularJS's expressions are sandboxed not for security reasons, but instead to maintain a proper separation of application responsibilities. For example, access to `window` is disallowed because it makes it easy to introduce brittle global state into your application.

However, this sandbox is not intended to stop attackers who can edit the template before it's processed by Angular. It may be possible to run arbitrary JavaScript inside double-curly bindings if an attacker can modify them.

But if an attacker can change arbitrary HTML templates, there's nothing stopping them from doing:

```
<script>somethingEvil();</script>
```

It's better to design your application in such a way that users cannot change client-side templates. For instance:

- Do not mix client and server templates
- Do not use user input to generate templates dynamically
- Do not run user input through `$scope.$eval`
- Consider using [CSP](#) (but don't rely only on CSP)

Angular Sandboxing

- Czy tu możemy doprowadzić do XSSa?

```
<html>
<body>
<p>
<?php
$q = $_GET['q'];
echo htmlspecialchars($q, ENT_QUOTES);
?>
</p>
</body>
</html>
```

Angular Sandboxing

- Czy tu możemy doprowadzić do XSSa?

```
<
<
<
<
$
</head>
<body>
<p>
<?php
$q = $_GET['q'];
echo htmlspecialchars($q, ENT_QUOTES);?>
</p>
</body>
</html>
```

Angular Sandboxing

- Expressions
 - Wyrażenia JS w podwójnych klamrach, które są wykonywane przez Angulara.

```
<html>
<head>
<meta charset="utf-8">
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.6/angular.js"></script>
</head>
<body>
<div ng-app>{{1+1}}</div>
</body>
</html>
```

- Wyrażenia same w sobie niewiele mogą.
- Ale, jeśli wyjdziemy z piaskownicy...

Angular Sandboxing

- Wyrażenie: `{{1+1}}`
- Ustaw breakpoint w angular.js na linii 13275.
- Wyrażenie przekształca się do:

```
"use strict";  
var fn = function(s, l, a, i) {  
    return plus(1, 1);  
};  
return fn;
```

```

var v0, v1, v2, v3, v4 = 1 && ('constructor in 1),
    v5;
if (!(v4)) {
    if (s) {
        v3 = s.constructor;
    }
} else {
    v3 = 1.constructor;
}
ensureSafeObject(v3, text);
if (v3 !== null) {
    v2 = ensureSafeObject(v3.constructor, text);
} else {
    v2 = undefined;
}
if (v2 !== null) {
    ensureSafeFunction(v2, text);
    v5 = 'alert\u00281\u0029';
    ensureSafeObject(v3, text);
    v1 = ensureSafeObject(v3.constructor(ensureSafeObject('alert\u00281\u0029', text)),
text);
} else {
    v1 = undefined;
}
if (v1 !== null) {
    ensureSafeFunction(v1, text);
    v0 = ensureSafeObject(v1(), text);
} else {
    v0 = undefined;
}
return v0;
};

```

Angular Sandboxing

AngularJS 1.4

- Podejście 1 (wymaga wywołania fromCharCode).

```
'a'.constructor.fromCharCode=[].join;  
'a'.constructor[0]='\u003ciframe onload=alert (/Backdoored/) \u003e';
```

- <http://jsfiddle.net/2zs2yv7o/2/>

- Podejście 2 (bezwarunkowe).

```
{{  
  'a'.constructor.prototype.charAt= [].join;  
  $eval('x=alert(1)')+''  
}}
```

- <http://jsfiddle.net/2zs2yv7o/6/>

Angular Sandboxing - Historia

AngularJS 1.4.0 - 1.4.9

AngularJS 1.5.0 - 1.5.8

Ian Hickey

```
constructor.prototype.  
{x = {'y': ''}.constructor.prototype; x['y'].charAt= [].join;$eval('x=alert(1)');}}  
    alert(1)//');}}}  
    } }
```

Angular Sandboxing - Historia

- AngularJS 1.6 i dalej...

Sandbox removal

Each version of AngularJS 1 up to, but not including 1.6, contained an expression sandbox, which reduced the surface area of the vulnerability but never removed it. **In AngularJS 1.6 we removed this sandbox as developers kept relying upon it as a security feature even though it was always possible to access arbitrary JavaScript code if one could control the AngularJS templates or expressions of applications.**

AngularJS >=1.6.0

Mario Heiderich (Cure53)

```
{{constructor.constructor('alert(1)')()}}
```

Angular Sandboxing - Rzyko



James Kettle (albinowax)

642

Reputation

360th

Rank

6.90

Signal

97th

Percentile

28.21

Impact

98th

Percentile

12

#125027

Reflected XSS on developer.uber.com via Angular template injection

Share:



State ● Resolved (Closed)

Severity □□□□ No Rating (---)

Disclosed publicly April 5, 2016 12:24am +0200

Participants

Reported To [Uber](#)

Visibility Public (Full)

Weakness Cross-site Scripting (XSS) - Generic

Bounty \$3,000

```
https://developer.uber.com/docs/deep-linking?q=wrtz{{(=""_.sub).call.call({}
["$=constructor"].getOwnPropertyDescriptor(.__proto__,).value,0,"alert(1)")()}}zzzz
```

Angular Sandboxing

- Zasada jest prosta jak z silnikami szablonów:

Dane przekazane przez użytkownika nie mogą być umieszczone bezpośrednio w HTMLu odpowiedzi.

Web Storage

- Przechowywanie danych po stronie przeglądarki.
- Maksymalny rozmiar danych większy, niż w przypadku ciasteczek.
- Dane nie są przesyłane do serwera.
- Per origin (per domena i protokół). Wszystkie strony z tego samego pochodzenia mają dostęp do tych samych danych.

HTML Web Storage Objects

HTML web storage provides two objects for storing data on the client:

- `window.localStorage` - stores data with no expiration date
- `window.sessionStorage` - stores data for one session (data is lost when the browser tab is closed)

localStorage i sessionStorage

- **localStorage** - przechowywane dane nie mają daty ważności

```
// Store
```

```
localStorage.setItem("lastname", "Smith");
```

```
// Retrieve
```

```
document.getElementById("result").innerHTML = localStorage.getItem("lastname");
```

- **sessionStorage** - dane są usuwane po zamknięciu okna

```
if (sessionStorage.clickcount) {
```

```
    sessionStorage.clickcount = Number(sessionStorage.clickcount) + 1;
```

```
} else {
```

```
    sessionStorage.clickcount = 1;
```

```
}
```

```
document.getElementById("result").innerHTML = "You have clicked the button " +  
sessionStorage.clickcount + " time(s) in this session.";
```

- Dane są przechowywane jako String. Trzeba pamiętać o **rzutowaniu**.

localStorage i sessionStorage

- Użytkownik ma władzę nad localStorage.
- **Nie uwierzytelniaj i nie autoryzuj** użytkownika na podstawie danych z localStorage.
- **XSS** to bezpośredni dostęp do localStorage.
- **Nie przechowuj** w localStorage danych **uwierzytelniających**.
- Wykorzystuj sessionStorage, chyba że dane **muszą być** dostępne po ponownym otwarciu przeglądarki.
- Nie przechowuj identyfikatorów sesji w localStorage. W tym celu lepiej skorzystać z ciasteczek zabezpieczonych *httpOnly*.
- Wykorzystuj **poddomeny** dla niezależnych aplikacji. W przeciwnym razie wszystkie aplikacje będą współdzielić **WebStorage**.

localStorage i XSS

