Advanced client-side web security threats

Security 2 2018-19

Univeristà Ca' Foscari Venezia
www.dais.unive.it/~focardi
secgroup.dais.unive.it

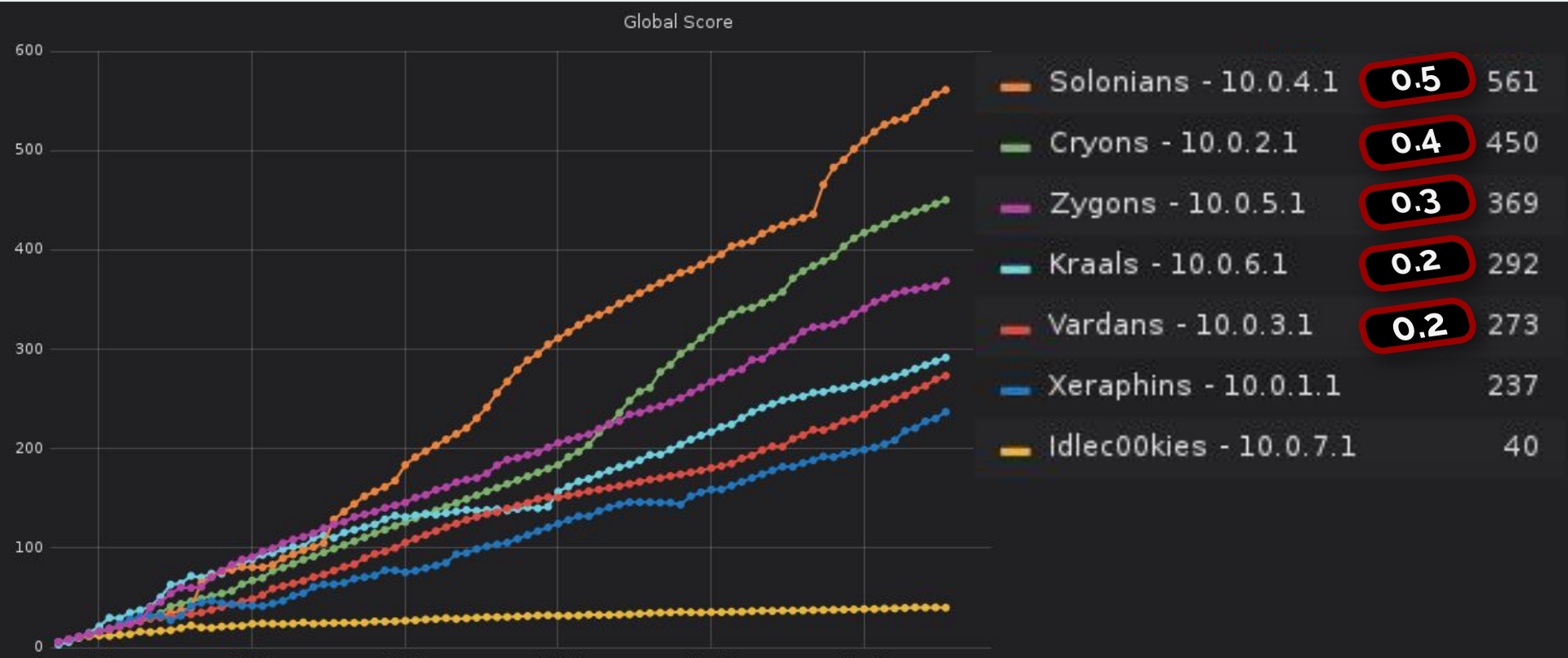
Second CTF Results

- Bonus
- Service Vulnerabilities
- Service Patches

Bonus

Second CTF

*score(position) = (6-position)*0.1*



Milkyway CTF Service

Vulnerabilities

- 11 SQLi
  - One slightly harder to fix than the others
- 1 command injection (**RCE!**)
- 1 loose comparison
- 2 logical bugs
- 2 misconfigurations

Client-side web security threats

- SOP Bypass: DNS Rebinding
- CSP and XSSAuditor Bypass: Script Gadgets

Same Origin Policy

SOP

- A standard browser policy that restricts access among documents or scripts loaded from different domains
- two pages have the same origin if the **protocol**, **port**, and **host** are the same for both pages

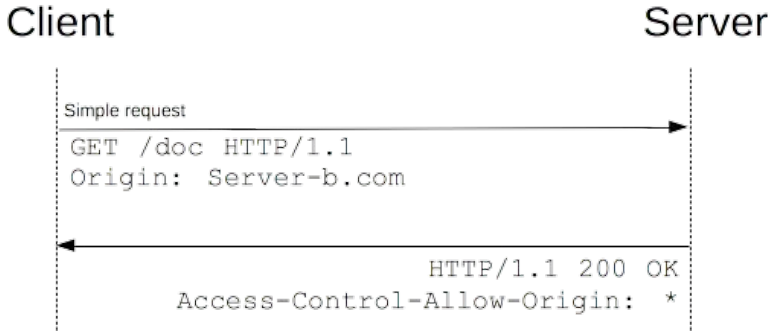
Cross-origin writes	typically allowed
Cross-origin embedding	typically allowed
Cross-origin reads	typically not allowed

https://secgroup.dais.unive.it/wp-content/uploads/2018/12/Client-side-web-security.pdf

Same Origin Policy

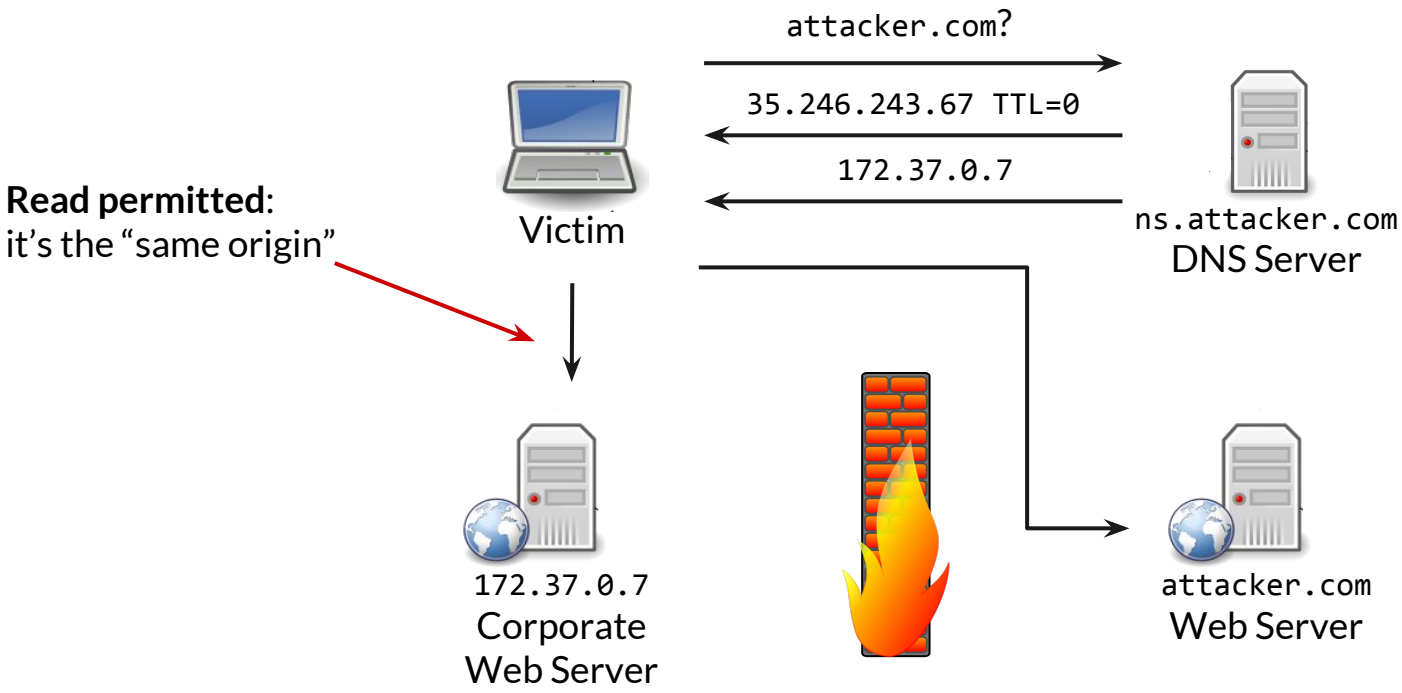
Cross Origin Resource Sharing (CORS)

- Without SOP, browsing on a malicious site will allow it to access other open pages and hijack any open session!
- How to allow cross-origin access? **CORS**!
- CORS is a part of HTTP that lets servers specify what hosts are permitted to load content from that server.



SOP Bypass

DNS Rebinding



SOP Bypass

DNS Rebinding

- Victim is tricked into visiting a website, such as `http://attacker.com/`, under the control of an attacker.
- The attacker wants to leak the file `secret` hosted at `localhost` (`http://localhost/secret`)
- The response of the following XHR is blocked by the Same-Origin Policy

```
$.get("http://127.0.0.1/secret", function(data) {console.log(data)});
```

SOP Bypass

DNS Rebinding

- We can bypass the SOP by **changing the IP** resolved by `attacker.com` from `35.246.243.67` to `127.0.0.1`, after the page is accessed by the victim

- **Note**: DNS is cached! you may need to wait several minutes for the address to change
  - DNS Server cache
  - OS cache
  - Browser cache

SOP Bypass

DNS Rebinding

DEMO

DNS Rebinding

rbndr.us

- But what if I do not have access to a DNS Server?

  https://github.com/taviso/rbndr

  DNS Rebinding Service

- To switch between 127.0.0.1 and 192.168.0.1 you would encode them as dwords[1], and then use:

  `7f000001.c0a80001.rbndr.us`

```
$ host 7f000001.c0a80001.rbndr.us
7f000001.c0a80001.rbndr.us has address 192.168.0.1
$ host 7f000001.c0a80001.rbndr.us
7f000001.c0a80001.rbndr.us has address 127.0.0.1
```

1: https://lock.cmpxchg8b.com/rebinder.html

CSP and XSSAuditor bypass

Script Gadgets (https://github.com/google/security-research-pocs)

XSS Mitigations

- WAFs, XSS filters (XSSAuditor)
  - Block requests containing dangerous tags / attributes
- HTML Sanitizers (input/output sanitization)
  - Remove dangerous tags / attributes from HTML
- Content Security Policy (CSP)
  - Distinguish legitimate and injected JS code

CSP and XSSAuditor bypass

Script Gadgets (https://github.com/google/security-research-pocs)

A Script Gadget is an **existing** JS code on the page that may be used to bypass mitigations:

```
<div data-role="button" data-text="I am a button"></div>

[...]

<script>
  var buttons = $("[data-role=button]");
  buttons.html(buttons.attr("data-text"));
</script>
```

```
<div data-role="button" ... >I am a button</div>
```

CSP and XSSAuditor bypass

Script Gadgets (https://github.com/google/security-research-pocs)

A Script Gadget is an **existing** JS code on the page that may be used to bypass mitigations:

CSP and XSSAuditor bypass

Script Gadgets (https://github.com/google/security-research-pocs)

Script Gadgets convert otherwise safe HTML tags and attributes into **arbitrary code execution**

Bypasses XSS mitigations that look for “<script>”

CSP and XSSAuditor bypass

Script Gadgets (https://github.com/google/security-research-pocs)

DEMO

CSP and XSSAuditor bypass

Script Gadgets (https://github.com/google/security-research-pocs)

- **Script Gadgets** can be found in most of the more popular web framework
  - can be used to bypass most mitigations in modern web applications