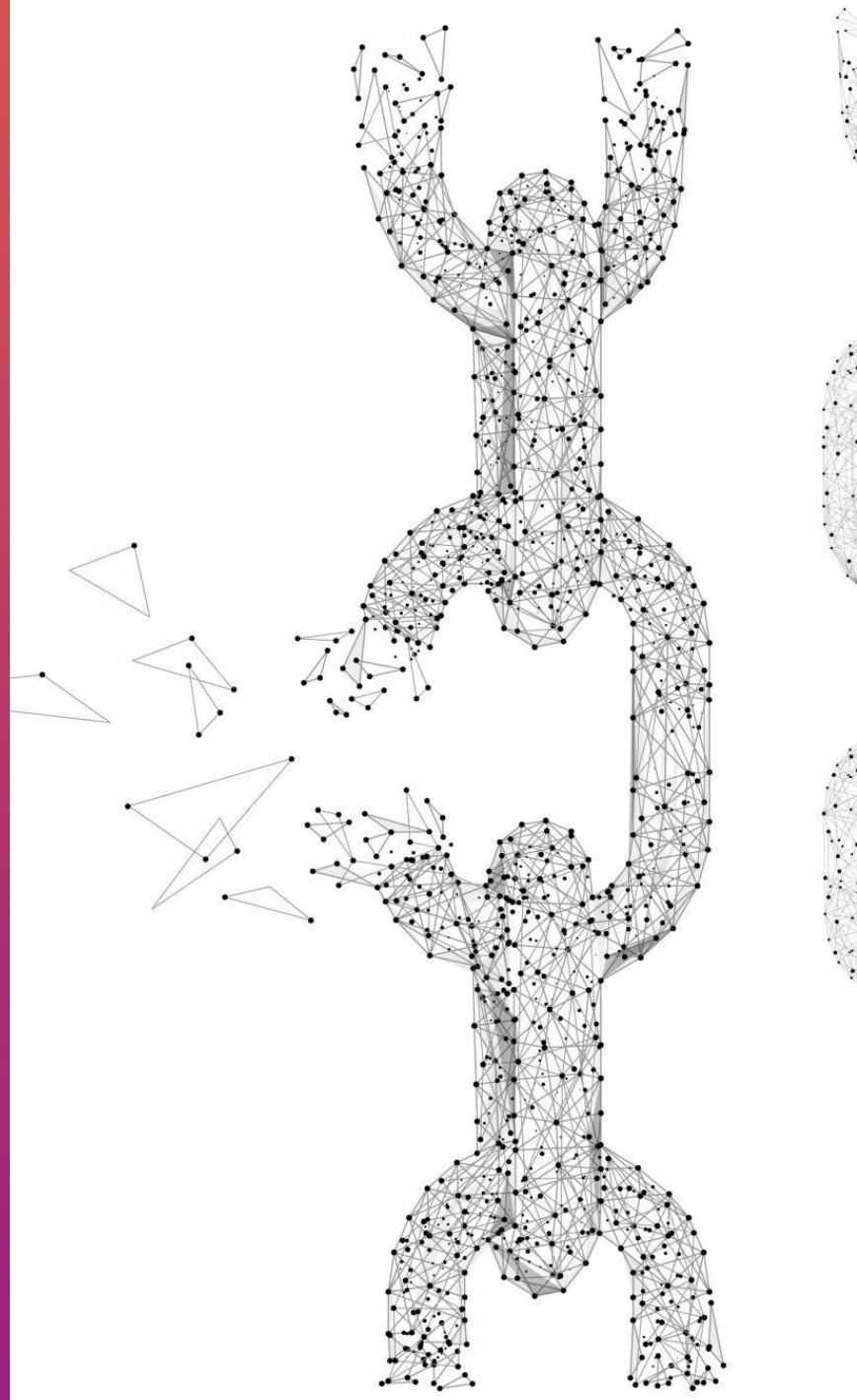
BY: HARSH BOTHRA

TALE OF CHAINING BUGS FOR ACCOUNT TAKEOVERS



WHO AM I?

- Application Security Enthusiast and Learner
- Triage @H1 | Core Lead Pentester @Cobalt.io | Community & Product Growth @Akto.io
- Author – 2 Books | Learn365 | SecurityExplained
- Blogger | Content Creator| Speaker
- Bugcrowd All Time Top 200

AGENDA

- Account Takeovers – Vulnerability Class or Impact?
- Ignored Vulnerabilities – Low Hanging Fruits
- Tale of Chaining Bugs for Account Takeovers

ACCOUNT TAKEOVERS

VULNERABILITY CLASS OR IMPACT?

POLL ON TWITTER
@HARSHBOTHRA_

IGNORED VULNERABILITIES – LOW HANGING FRUITS

Open Redirection	CRLF Injection	GraphQL Introspection	Missing Cookie Security & HTTP Security Headers
Host Header Injection	API Fuzzing (Lack of Rate Limit on Path)	Lack of Server-Side Validation	External SSRF
Prototype Pollution	Deeplink Misconfiguration	OAuth Misconfiguration	HTML Injection

TALE OF CHAINING BUGS FOR ACCOUNT TAKEOVERS

GraphQL Introspection to Account Takeover

Host Header Injection to Account Takeover

CRLF to XSS leading to Account Takeover

Open Redirection to Account Takeover

GRAPHQL INTROSPECTION TO ACCOUNT TAKEOVER

Bug Description:

- The application allowed an unauthenticated user to access and run **Introspection Queries** (Informative – In General).

- After digging and visualising their GraphQL operations, I found a couple of interesting operations allowing to **Get User ID by Email and Generate Auth Token using Email.**

- Authenticated with Attacker User and Performed the Operation using **/graphql** endpoint to query victim user's ID and later tried using it to get the Auth token but it didn't work.

- Next, tried **Logical Manipulation (or Parameter Pollution)** and supplied IDs like **attackerId, victimId** and it returned **Victim's Auth Token.**

GRAPHQL INTROSPECTION TO ACCOUNT TAKEOVER (CONT'D....)

Bug Description (Cont'd...):

- Using victim's auth token, changed their email address to **Attacker Controlled Email** and reset their password and had full control of their account.

Severity Bump: Informative to Critical

Program & Platform: Private Program (Out of Platform)

Reward Issued: $$$$$ (5-Digit)

HOST HEADER INJECTION ON EMAIL CHANGE TO ACCOUNT TAKEOVER

Bug Description:

- The application shared the same interface for external and internal users. The point of validation was the internal user's had their accounts with **@company.com** and some extra privileges.

- I had access to one of their GSuite account as part of a Pentest engagement.

- I tried Host Header Injection (mainly on password reset as we all do) but no luck on any endpoints.

- Next, I fuzzed the application using **Collaborator Everywhere** and observed that this **email change endpoint** was reflecting the External Host via **X-Forwarded-Host** header.

- Using the attacker account (external user), I requested an email change for **knownuser@company.com** with attacker controlled Host.

HOST HEADER INJECTION ON EMAIL CHANGE TO ACCOUNT TAKEOVER (CONT'D...)

Bug Description (Cont'd...):

- I was able to steal the confirmation token and use it to change email to my attacker (external user) account.

- Relogged in and got the privileges escalated to internal user dashboard that allowed to reset the password for any external user.

Result: Mass Account Takeover

Severity: Critical

Program and Platform: Private (Through Pentest)

Award: Bonus in $$$$

CRLF TO XSS LEADING TO ACCOUNT TAKEOVER

Bug Description:

- The application was vulnerable to **Self Cross-Site Scripting** via **Non-Existing Cookie Parameter. (Informative).**

- Fuzzed the application and found it vulnerable to **CRLF Injection** through double encoding.

- Used CRLF Injection to Inject the Non-Existing Cookie Parameter and Created a PoC like: something.com/<injectionpayload>=cookie:<xss payload>

- XSS was executed successfully (**Medium)**

- Now, further created a PoC to steal session token as the JWT was passed in the Cookies as well and there was no HTTPOnly flag.

- Successfully Hijacked User's Session – Changed Email – Reset Password – Full Account Takeover.

CRLF TO XSS LEADING TO ACCOUNT TAKEOVER (CONT'D...)

Result: Full Account Takeover

Severity: Informative to Critical

Program and Platform: Private

Award: $$$$ + $$$ (Bonus)

OPEN REDIRECTION TO ACCOUNT TAKEOVER

Bug Description:

- The application had multiple sub-applications and it used Auth Code to authenticate the sub applications and it was possible to access the sub-applications allowing account takeover.

- The redirection to sub-application was using **OAuth** flow and had **redirection** parameter that sent the auth token to the sub-application

- Found an open redirection that allowed to steal the auth token of the application.

- Attacker was able to successfully access the sub application. **(High)**

- Later, I also found an privilege escalation that allowed access from Sub-App to Main-App but that's a different Privilege Escalation Story.

OPEN REDIRECTION TO ACCOUNT TAKEOVER (CONT'D...)

Result: Limited Account Takeover

Severity: High

Program and Platform: Private

Award: $$$

OTHER INTERESTING ATO VECTORS

- HTML Injection to AWS Metadata Leak leading to AWS Takeover
- Insecure Deeplink allowing Account Takeover
- Password Reset Poisoning to Account Takeover
- Mass Assignment Leading to Account Takeover
- IDOR leading to Account Takeover
- Lack of Server-Side Validation in Email during Registration leading to Account Takeover

NEXT PLANS?
WILL LAUNCH AN UPDATED MINDMAP ON DIFFERENT TECHNIQUES FOR ACCOUNT TAKEOVER

SUMMARY

THANK YOU FOLKS!