**A blue and black text

Description automatically generated**

**Web Application Pen-testing**

Year 2 (2025/26), Semester 3

**SCHOOL OF INFOCOMM TECHNOLOGY**

Diploma in Cybersecurity & Digital Forensics

**ASSIGNMENT REPORT**

|  |  |
| --- | --- |
| **Weightage:** | 40% of Module |
| **Individual/Team/Both:** | **Team** |
| **Format:** | Font Arial  Font Size 11, 1.5 spacing |

|  |  |
| --- | --- |
| **Student Name** | **Student Number** |
| 1. Wyse Lee Hong Yao |  |
| 2. Ian Tan Jun Yang |  |
| 3. Toh Keng Siong |  |
| 4. Gao Yu Hao |  |

Table of Contents

[smtp2go.com 4](#_Toc206011348)

[app.smtp2go.com 4](#_Toc206011349)

[api.smtp2go.com 4](#_Toc206011350)

[Scope Exclusions 4](#_Toc206011351)

[Assessment Overview 6](#_Toc206011352)

[Finding Severity Ratings 7](#_Toc206011353)

[Executive Summary 8](#_Toc206011354)

[Vulnerability Summary 8](#_Toc206011355)

[Web Application Penetration Test Findings 8](#_Toc206011356)

[Leaked EIN Stored in Cleartext in a Publicly Accessible W-9 Form 9](#_Toc206011357)

[HTTP Header Information Disclosure 9](#_Toc206011358)

[Outdated JavaScript Library (jQuery 1.11.0) 10](#_Toc206011359)

[Passive Reconnaissance 11](#_Toc206011360)

[Authentication & Session Handling 11](#_Toc206011361)

[User & Permission Management 11](#_Toc206011362)

[Information Gathering 12](#_Toc206011363)

[Conduct Search Engine Discovery Reconnaissance for Information Leakage 12](#_Toc206011364)

[Fingerprint Web Server 13](#_Toc206011365)

[Banner grabbing 13](#_Toc206011366)

[Review Webserver Metafiles for Information Leakage 14](#_Toc206011367)

[Robots & Sitemap 14](#_Toc206011368)

[Fingerprint Web Application 15](#_Toc206011369)

[Dirbusting 15](#_Toc206011370)

[Tech Stack Detection 16](#_Toc206011371)

[Web Application Firewall (WAF) 17](#_Toc206011372)

[Wordpress enumeration 18](#_Toc206011373)

[Enumerate Applications on Webserver 19](#_Toc206011374)

[Identity Management Testing 21](#_Toc206011375)

[Test User Registration Process 21](#_Toc206011376)

[Authentication Testing 22](#_Toc206011377)

[Testing for Vulnerable Remember Password 22](#_Toc206011378)

[Testing for Weak Password Policy 24](#_Toc206011379)

[Testing for Weak Password Change or Reset Functionalities 25](#_Toc206011380)

[Authorization Testing 27](#_Toc206011381)

[Testing Directory Traversal File Include 27](#_Toc206011382)

[Input Validation Testing 28](#_Toc206011383)

[Testing for Cross Site Scripting (XSS) 28](#_Toc206011384)

[Testing for server-side request forgery 29](#_Toc206011385)

[Testing for SQL Injection 30](#_Toc206011386)

[Error Handling 31](#_Toc206011387)

[Testing for Improper Error Handling 31](#_Toc206011388)

[Comment Moderation 33](#_Toc206011389)

[XSS 33](#_Toc206011390)

[IDOR 34](#_Toc206011391)

[Appendix 35](#_Toc206011392)

[Importing scope into BurpSuite 35](#_Toc206011393)

[Passive Reconnaissance Results 36](#_Toc206011394)

[Google Dorking Results 37](#_Toc206011395)

[Sitemap results 39](#_Toc206011396)

[Banner grabbing 41](#_Toc206011397)

# Scope

|  |  |
| --- | --- |
| Assessment | Details |
| Bug Bounty Program | smtp2go.com,  app.smtp2go.com,  api.smtp2go.com |

## smtp2go.com

Standard Wordpress site hosted with WPEngine, scripting is all custom JQuery based.

## app.smtp2go.com

Flask based app running on Python 2.7, some pages are VueJS but most are scripted with custom JQuery.

Create a free account to gain login access.

To comply with the program guidelines, the test accounts created on the SMTP2GO website (<https://www.smtp2go.com/pricing/>) uses the Hacker Email Alias - [username]@wearehackerone.com and [username+test]@wearehackerone.com

## api.smtp2go.com

Most of the endpoints are handled by Flask on Python3 with Postgres as a main database.

Newer endpoints use Go on Gin framework.

Redis is mostly used for cache and rate limiting.

## Scope Exclusions

* The domain support.smtp2go.com is out of scope.
* Social engineering (e.g. phishing, vishing, smishing).
* Clickjacking on pages with no sensitive actions.
* Cross-Site Request Forgery (CSRF) on unauthenticated forms or forms with no sensitive actions.
* Attacks requiring MITM or physical access to a user's device.
* Previously known vulnerable libraries without a working Proof of Concept.
* Comma Separated Values (CSV) injection without demonstrating a vulnerability.
* Missing best practices in SSL/TLS configuration.
* Any activity that could lead to the disruption of our service (DoS).
* Content spoofing and text injection issues without showing an attack vector/without being able to modify HTML/CSS.
* Rate limiting or bruteforce issues on non-authentication endpoints.
* Missing best practices in Content Security Policy.
* Missing HttpOnly or Secure flags on cookies.
* Missing email best practices (Invalid, incomplete or missing SPF/DKIM/DMARC records, etc.).
* Vulnerabilities only affecting users of outdated or unpatched browsers [Less than 2 stable versions behind the latest released stable version].
* Software version disclosure / Banner identification issues / Descriptive error messages or headers (e.g. stack traces, application or server errors).
* Public Zero-day vulnerabilities that have had an official patch for less than 1 month will be awarded on a case by case basis.
* Tabnabbing.
* Open redirect - unless an additional security impact can be demonstrated.
* Issues that require unlikely user interaction.
* User enumeration, unless you can demonstrate that a brute force attack is likely to succeed. This includes Wordpress user enumeration.
* The ability to send an email from any domain name (not just one of your account's verified Sender Domains). This is only considered a vulnerability if the email becomes DKIM-signed with a DKIM signature at that same domain name, or the email passes DMARC. For example, you manage to send an email from hello@anotherdomain.com and it becomes signed with a DKIM signature with d=anotherdomain.com.
* Issues with notification emails, as the notification system is currently being overhauled.

# Assessment Overview

From August 1st to August 11th 2025, the engagement with SMTP2GO’s bug bounty program on HackerOne is carried out to evaluate the security posture of its web application compared to current industry best practices. This penetration test is performed under a “grey box” approach as partial knowledge of the application's endpoints is provided. All testing performed is based on the *OWASP Testing Guide’s* [*Web Security Testing Guide (WSTG) v4.2*](https://owasp.org/www-project-web-security-testing-guide/v42/)*.*

Phases of penetration testing is as follows:

1. Passive Reconnaissance – exploring the web application as a user to understand the functionalities of each endpoint and the application’s logic.
2. Information Gathering
3. Conduct Search Engine Discovery Reconnaissance for Information Leakage
4. Fingerprint Web Server
5. Review Webserver Metafiles for Information Leakage
6. Enumerate Applications on Webserver
7. Fingerprint Web Application
8. Identity Management Testing
9. Test Role Definitions
10. Test User Registration Process
11. Authentication Testing
12. Testing for Vulnerable Remember Password
13. Testing for Weak Password Policy
14. Testing for Weak Password Change or Reset Functionalities
15. Authorization Testing
16. Testing Directory Traversal File Include
17. Input Validation Testing
18. Testing for Cross Site Scripting (XSS)
19. Testing for SQL Injection
20. Error Handling
21. Testing for Improper Error Handling

# Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

|  |  |  |
| --- | --- | --- |
| **Severity** | **CVSS V4**  **Score Range** | **Definition** |
| Critical | 9.0-10.0 | Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately. |
| High | 7.0-8.9 | Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible. |
| Medium | 4.0-6.9 | Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved. |
| Low | 0.1-3.9 | Vulnerabilities are non-exploitable but would reduce an organization’s attack surface. It is advised to form a plan of action and patch during the next maintenance window. |
| Informational | 0.0 | No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation. |

# Executive Summary

This report evaluates security posture of SMTP2GO through a Web Application Penetration Test from August 1st to August 11th 2025. By leveraging the scope and rules defined under SMTP2GO’s public bug bounty program on HackerOne, no critical vulnerabilities were identified after thorough reconnaissance techniques and exploits were implemented.

There is, however, minor vulnerabilities to remediate in order to maintain customer trust, comply with security best practices, and reduce the risk of exploitation by malicious actors.

## Vulnerability Summary

The following tables illustrate the vulnerabilities found by impact and recommended remediations:

### Web Application Penetration Test Findings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 1 | 1 |
| Critical | High | Medium | Low | Informational |

|  |  |  |
| --- | --- | --- |
| **Finding** | **Severity** | **Recommendation** |
| Web Application Penetration Test | | |
| CWE-312: Leaked EIN stored in cleartext in a Publicly Accessible W-9 Form | Medium | Ensure that sensitive forms are stored in secured repositories accessible only to authorized personnel. |
| CWE-200: HTTP Header Information Disclosure | Low | Modify the HTTP headers of the web server to not disclose detailed information about the underlying web server. |
| CWE-937: Outdated JavaScript library (jQuery 1.11.0) with known vulnerabilities | Low | Upgrade to the latest version. |

### Leaked EIN Stored in Cleartext in a Publicly Accessible W-9 Form

#### CWE-312: Cleartext Storage of Sensitive Information

Severity: Medium

Base Score: 6.9 - CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N

#### Evidence

The form revealed in the search result containing sensitive information can be found [here](#_Conduct_Search_Engine) (Figure 1.2). Note: For security reasons, the EIN is obfuscated in this report.

#### Impact

* Confidentiality Breach: The EIN is a federally issued tax identifier for the business, comparable in sensitivity to an SSN for individuals.
* Fraud Risk: Attackers could use the EIN for fraudulent tax filings, opening credit accounts, or impersonating the organization in financial transactions.

#### Steps to reproduce:

1. Search ‘site:\*.smtp2go.com filetype:pdf’ on Google.
2. The form is revealed with the name “Request for Taxpayer Identification Number and Certification”.

### HTTP Header Information Disclosure

#### CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

Severity: Low

Base Score: 2.1 - CVSS:4.0/AV:N/AC:H/AT:N/PR:N/UI:A/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N

#### Evidence

Requests and responses with the highlighted sensitive information shown [here](#_Banner_grabbing_1) (Figure 2.1 & Figure 2.2).

#### Impact

Enables targeted attacks using known exploits for the identified software/version if any are discovered.

#### Steps to reproduce:

1. Send a HTTP request to smtp2go.com with openssl CLI tool.
2. The information is disclosed in the HTTP header.
3. Repeat Steps 1 & 2 for app.smtp2go.com domain.

### Outdated JavaScript Library (jQuery 1.11.0)

#### CWE-937: Using Components with Known Vulnerabilities

Severity: Informational

Base Score: 0 - [CVSS:4.0/AV:N/AC:H/AT:N/PR:N/UI:A/VC:N/VI:N/VA:N/SC:N/SI:N/SA:N](https://www.first.org/cvss/calculator/4.0#CVSS:4.0/AV:N/AC:H/AT:N/PR:N/UI:A/VC:N/VI:N/VA:N/SC:N/SI:N/SA:N)

#### Evidence

The outdated library version is discovered in the tech stack detection section [here](#_Findings).

#### Impact

The jQuery 1.11.0 is vulnerable to XSS with CVE-2020-11023 and CVE-2015-9251. Although the version was not found to be affected by any security vulnerabilities, it is recommended to keep libraries up to date.

# Passive Reconnaissance

Through browsing the web application like any other user would, the following endpoints that are found to be potentially vulnerable are listed below:

### Authentication & Session Handling

* https://app.smtp2go.com/login/
  + Test for weak login protections, 2FA bypass, vulnerable remember password.

### User & Permission Management

* <https://app-us.smtp2go.com/account/team/>
  + Test for privilege escalation vulnerabilities.
* <https://app-us.smtp2go.com/account/2fa/>
  + Test for weak 2FA implementation.

# Information Gathering

## Conduct Search Engine Discovery Reconnaissance for Information Leakage

Objective: Identify sensitive information of the application or organisation that is exposed directly on the websites in scope.

When searching for PDF files related to the smtp2go.com domain, the top result leads to a PII disclosure where the Employer Identification Number (EIN) is leaked. This is a vulnerability classified under “CWE-312: Cleartext Storage of Sensitive Information”.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 1.1 – Search result of PDF files related to smtp2go.com

A form with text and numbers

AI-generated content may be incorrect.

Figure 1.2 – W-9 form with EIN revealed

## Fingerprint Web Server

Objective: Determine the version and type of a running web server to enable further discovery of any known vulnerabilities.

### Banner grabbing

Using BurpSuite to intercept the response for each request sent for each of the domains in scope, the HTTP header may reveal sensitive information.

On the smtp2go.com domain, the response header revealed the server version, technologies used and the operating system (Apache 2.4.62 Debian). This is a “HTTP Header Information Disclosure” vulnerability classified under “CWE-200: Exposure of Sensitive Information to an Unauthorized Actor”.

The same vulnerability is found on the app.smtp2go.com domain, revealing the server used (gunicorn). This suggests a Python-based backend running on a UNIX system.

The request to the api.smtp2go.com domain did not provide much information, so the result is in the Appendix section ([Banner grabbing](#_Banner_grabbing)).

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 2.1 – Response by smtp2go.com domain

A computer screen shot of a number

AI-generated content may be incorrect.

Figure 2.2 – Response by app.smtp2go.com domain

## Review Webserver Metafiles for Information Leakage

Objective: Identify hidden or obfuscated paths and functionality through the analysis of metadata files. Extract and map other information that could lead to better understanding of the systems at hand.

### Robots & Sitemap

In robots.txt, only a sitemap is discovered at <https://www.smtp2go.com/sitemap_index.xml>, which is a file that lists the pages of the website. The sitemap did not reveal any interesting pages other than those already discovered during passive reconnaissance and its results can be found in the Appendix section ([Sitemap results](#_Sitemap_results)).

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.1 – robots.txt file content

## Fingerprint Web Application

Objective: Fingerprint the components being used by the web applications.

### Dirbusting

With the gobuster tool, directories present on the website are found by brute forcing through a wordlist. The directory that is most interesting is wp-config, which is a file commonly found in applications running Wordpress. However, since the status code is 403, we do not have authorised access to the directory. Hence, there are no findings in this section.

A computer screen shot of a program

AI-generated content may be incorrect.

Figure 4.1 – dirbusting results on www.smtp2go.com

A white and black box

AI-generated content may be incorrect.

Figure 4.2 – 403 status code when browsing /wp-config

### Tech Stack Detection

#### Server & Technology Stack

* Apache 2.4.62 running on Debian Linux (smtp2go.com & app.smtp2go.com)
* Cloudflare as a reverse proxy (smtp2go.com).
* Gunicorn for (app.smtp2go.com) login endpoint.
* Bootstrap, jQuery (versions 3.6.4 on smtp2go.com & 1.11.0 on app.smtp2go.com), and HTML5 detected.
* WordPress present on both main and app subdomains.
* IP addresses
  + smtp2go.com – 43.228.184.5
  + app.smtp2go.com – 143.42.179.218
  + api.smtp2go.com – 104.200.26.186

#### Findings

The app.smtp2go.com domain running jQuery 1.11.0 has 2 known vulnerabilities, [CVE-2015-9251](https://www.cvedetails.com/cve/CVE-2015-9251/) and [CVE-2020-11023](https://www.cvedetails.com/cve/CVE-2020-11023/). This vulnerability is classified under CWE-937: Using Components with Known Vulnerabilities.

A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.



### Web Application Firewall (WAF)

Only <https://www.smtp2go.com> Only <https://www.smtp2go.com> uses a WAF (Cloudflare). Both <https://app.smtp2go.com> and <https://api.smtp2go.com> do not have any WAF protection.

A computer screen shot of a computer screen

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

### Wordpress enumeration

A close-up of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Enumerate Applications on Webserver

With the IP addresses found during the web server fingerprinting reconnaissance, nmap scans are ran to discover services running on each open port for each host.

Known services running on 43.228.184.5 (smtp2go.com):

* SSH on port 22
* SMTP(S) on multiple ports
* HTTP on ports 80 and 12828
* HTTPS on ports 443, 2823, 2829

Known services running on 143.42.179.218 (app.smtp2go.com):

* HTTP on port 80
* HTTPS on port 443

Known services running on 104.200.26.186 (api.smtp2go.com):

* SMTP on port 25
* HTTP on port 80
* HTTPS on port 443
* SMTPS on port 465

A screenshot of a computer screen

AI-generated content may be incorrect.

A computer screen with white text

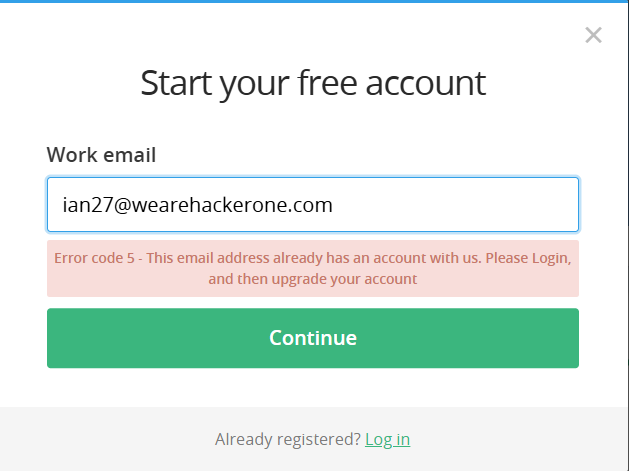
AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

# Identity Management Testing

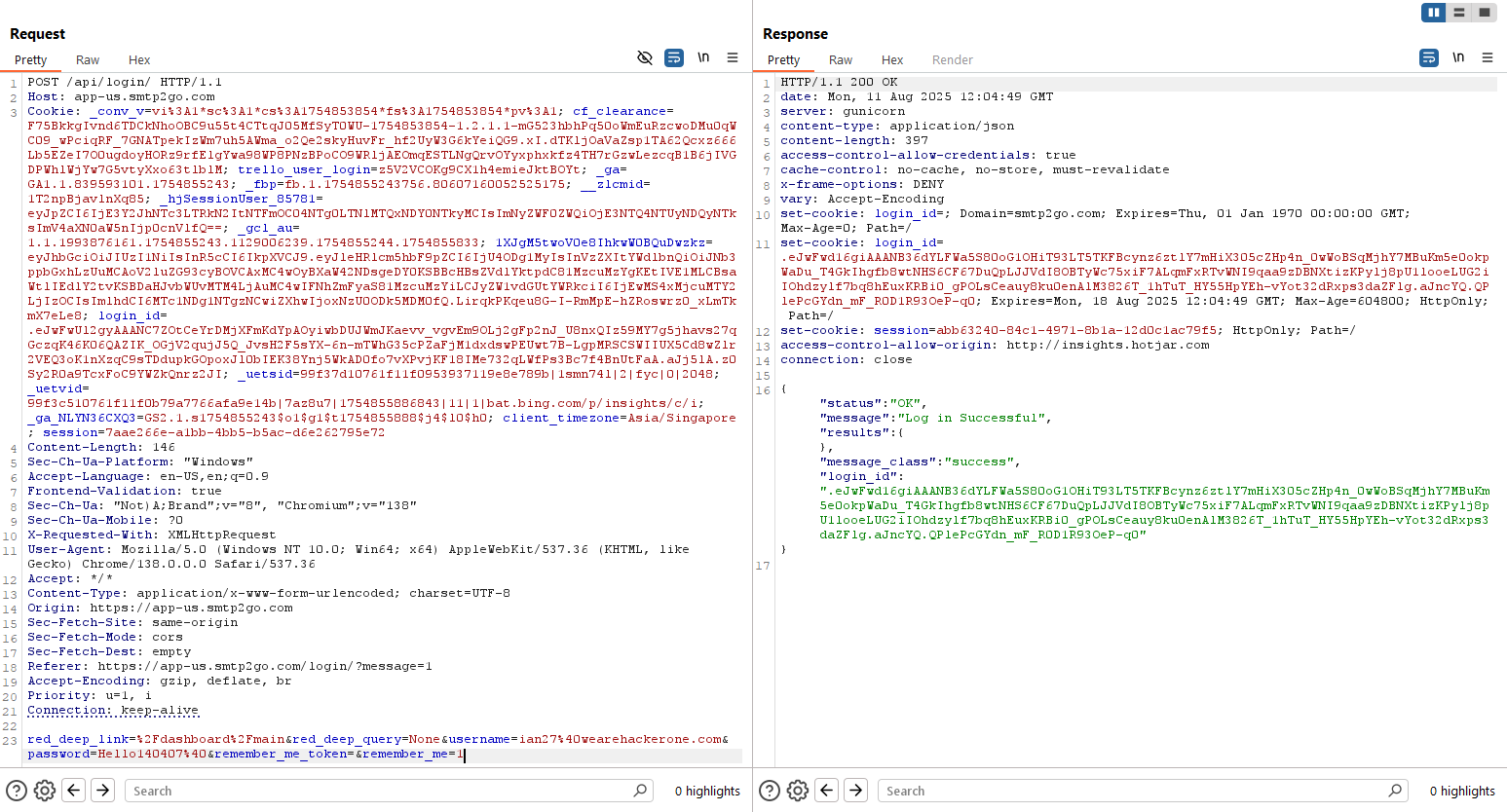
## Test User Registration Process



Tried inputting an already registered email but the server checks if it already exists. An attacker can brute-force common email addresses to confirm which ones have accounts.

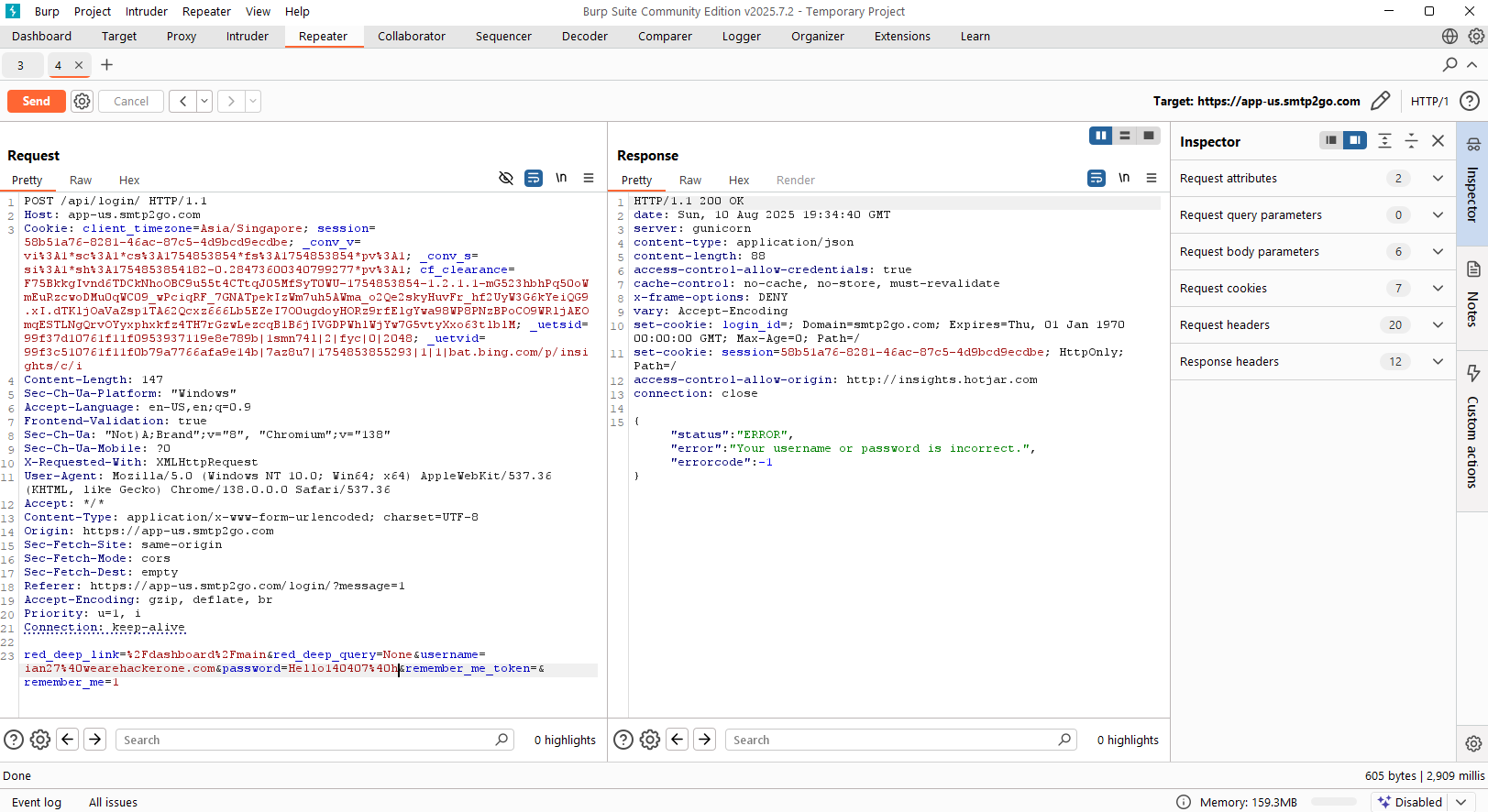
# Authentication Testing

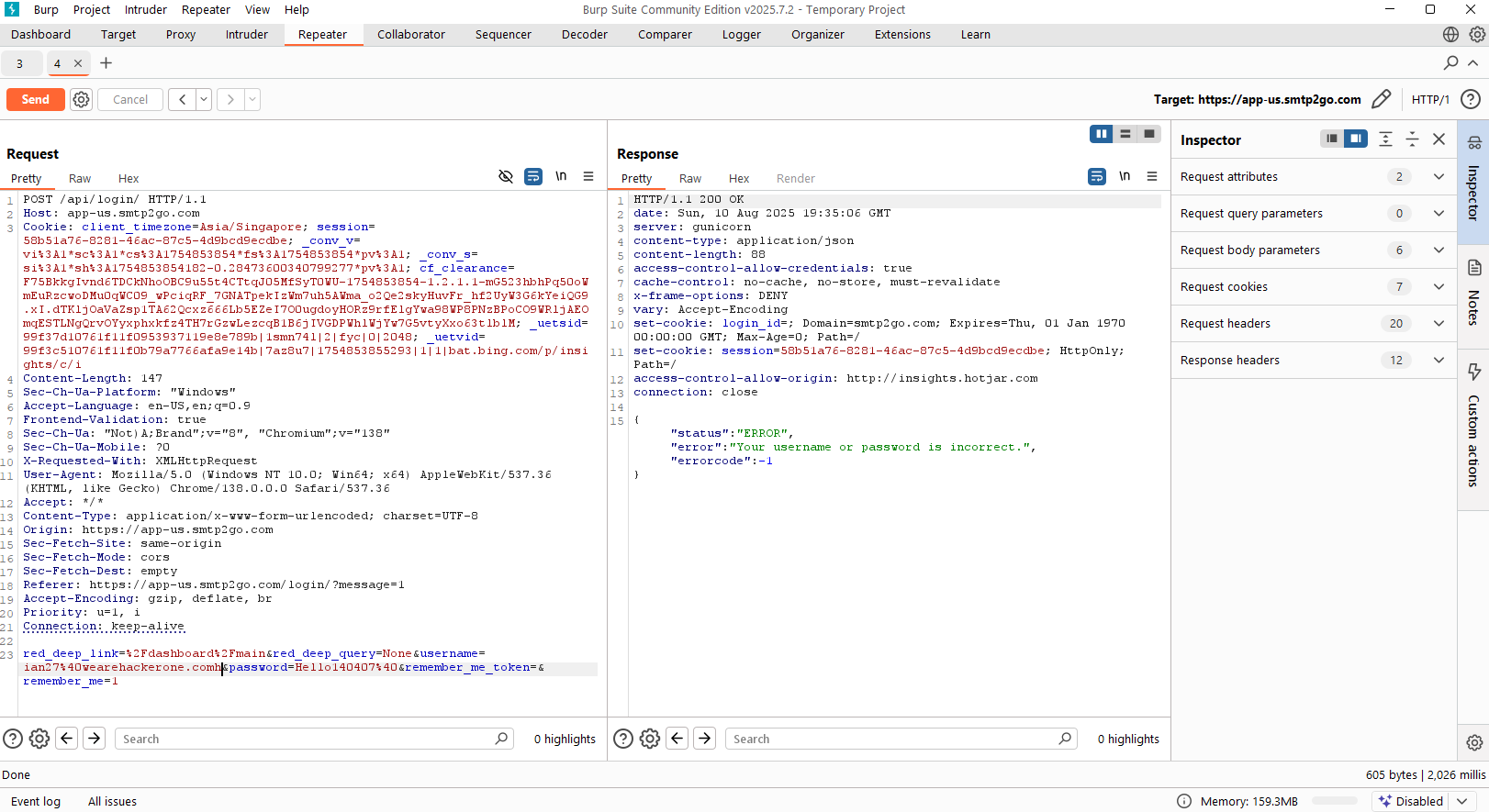
## Testing for Vulnerable Remember Password

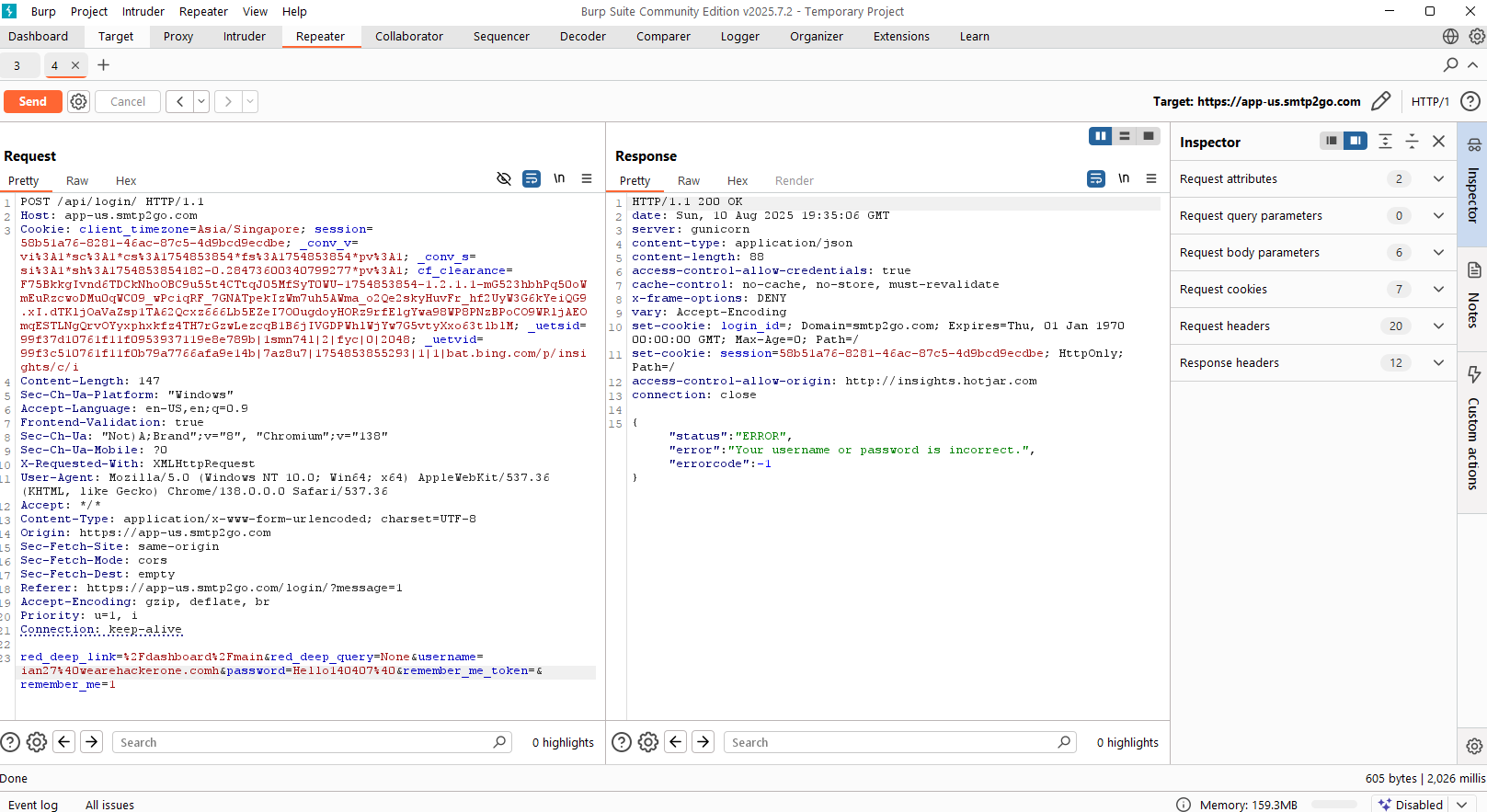




The 2 images above show a request to log in with “Remember me” option ticked, I modified the value of “remember\_me” and it seems like the server correctly disregards the tampered input. The “Remember me” feature is securely implemented and not vulnerable to parameter tampering.



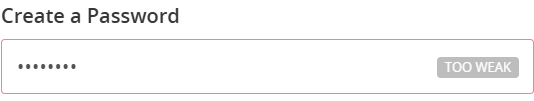




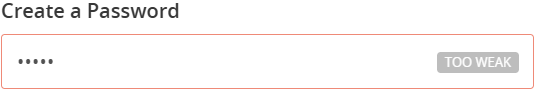
In the 3 pictures above, I tried entering wrong password, wrong username and both wrong password and username. In all test cases, the server responded immediately with a generic, well-formed JSON error message, {"error": "Your username or password is incorrect."}.

## Testing for Weak Password Policy

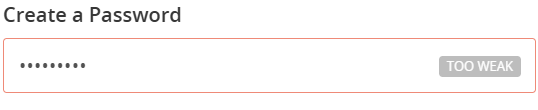
Testing password all alphabets (password)



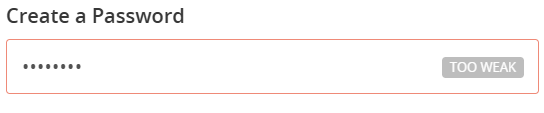
Testing all numeric (12345)



Test alphanumeric (password1)

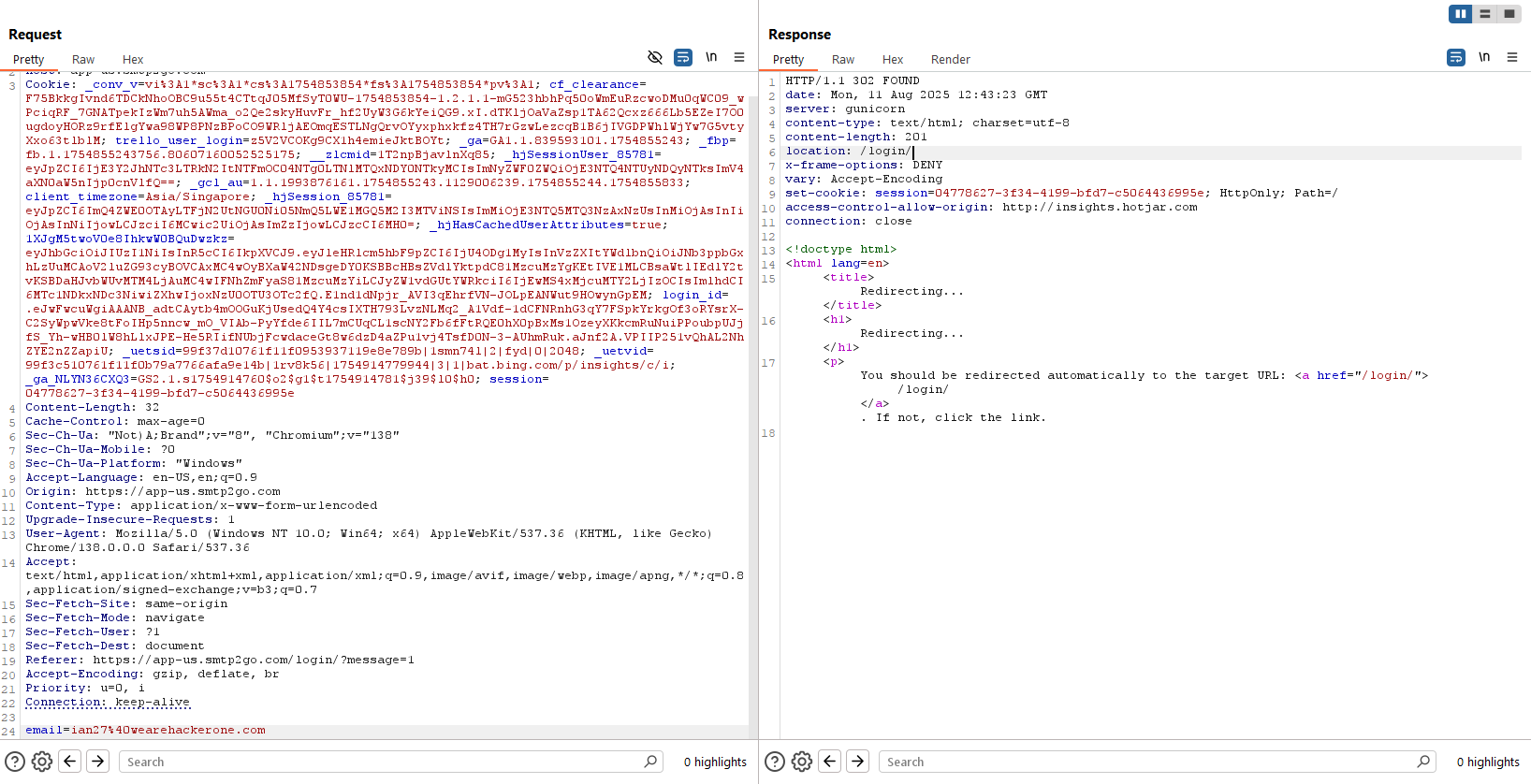


Test Capitalized password (Password)



From the results, it’s clear that the website’s password policy is effective in rejecting weak passwords. All tested variations, including alphabet-only (password), numeric-only (12345), alphanumeric (password1), and capitalized (Password), were flagged as “TOO WEAK.” This indicates that the system enforces a strong password policy that discourages common, guessable credentials and helps reduce the risk of brute-force or dictionary attacks. The test confirms that the platform requires more complex and secure passwords, which is a good security practice.

## Testing for Weak Password Change or Reset Functionalities



Upon submitting an email to the /forgot-password/ endpoint, the server responds with a generic HTTP 302 redirect to the login page without revealing whether the email address is registered in the system. This is a good practice as it prevents user enumeration, an attack where a malicious actor could confirm valid user emails. No vulnerabilities were identified in this specific area.

# Authorization Testing

## Testing Directory Traversal File Include

When logging in on https://app.smtp2go.com/api/login, there is a POST request with a parameter “red\_deep\_link” that URL decodes as “/dashboard/main”. It can also be seen that after logging in, the response JSON data redirects the user to /dashboard/main.



A close-up of a computer screen

AI-generated content may be incorrect.

To test path traversal file include, I first tried to see if removing a directory (main) such that the parameter has the value “/dashboard/” would even do anything. The response shows that the redirect now brings the user to “/dashboard” instead now.



A screenshot of a computer

AI-generated content may be incorrect.

With the change reflecting in the response, I tried many URL encoded “../” to attempt traversing into the system’s root directory and read the /etc/passwd file since the app.smtp2go.com host is running on Linux. The response shows that the redirect brings us to “/dashboard/main”, meaning that the attack failed.



A screenshot of a computer

AI-generated content may be incorrect.

# Input Validation Testing

## Testing for Cross Site Scripting (XSS)

Checking the search\_query value in the source code, all instances of the search is HTML encoded. Moreover, when a suspicious search is entered, a 403 page appears that is most likely blocking the payload. The search function is not vulnerable to XSS.

A screenshot of a computer

AI-generated content may be incorrect.











A black screen with orange and white text

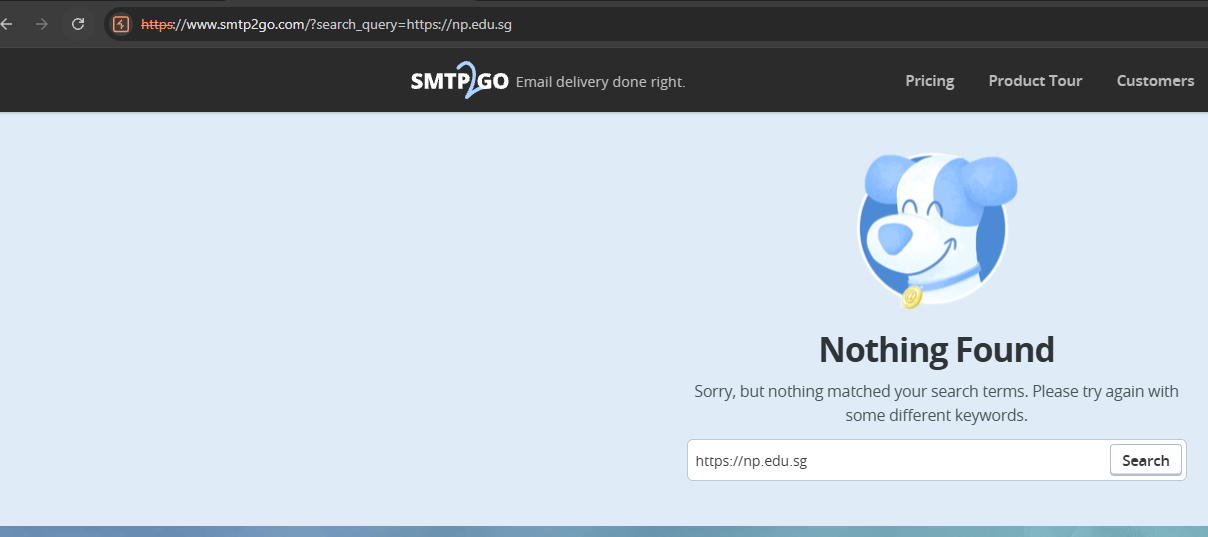
AI-generated content may be incorrect.

A screen shot of a computer

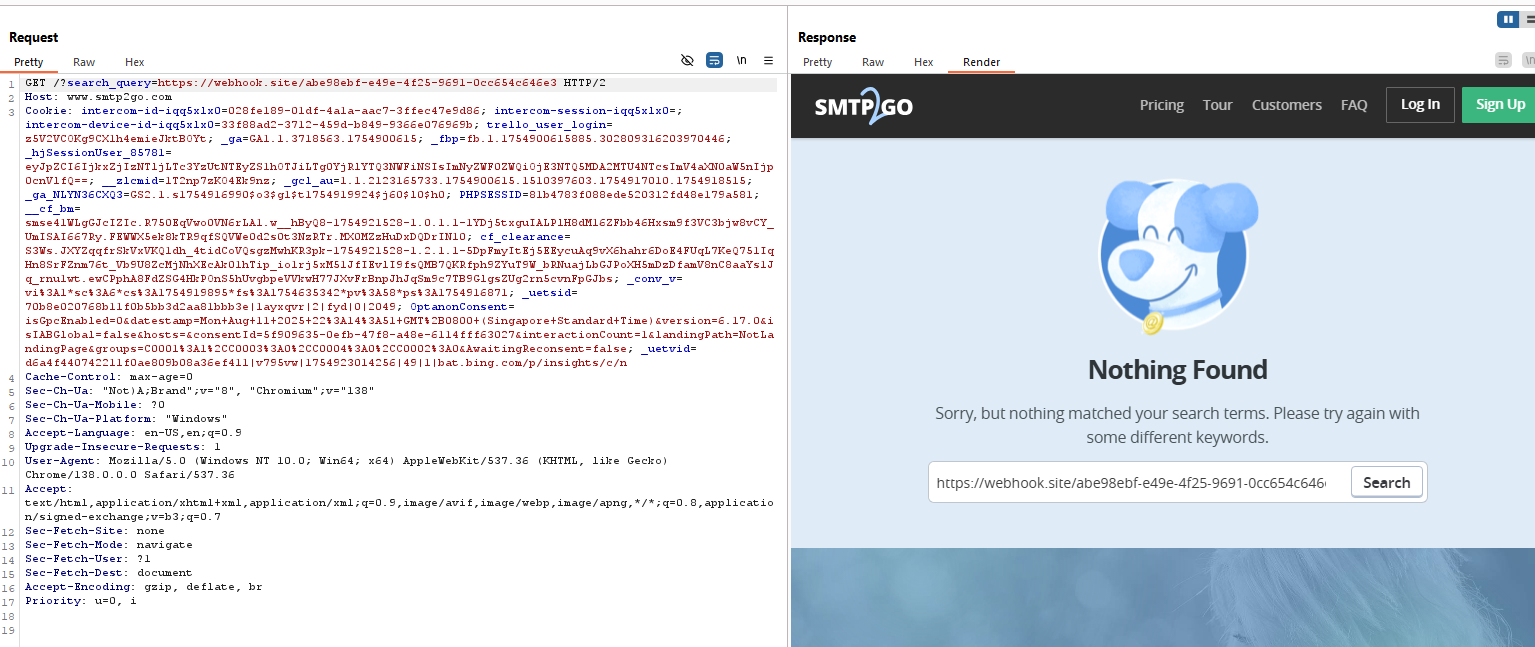
AI-generated content may be incorrect.

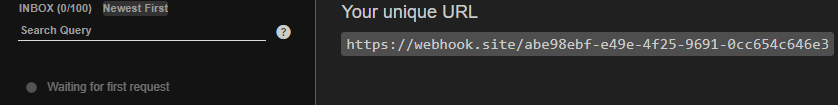
## Testing for server-side request forgery

Objective: Determine whether supplying a URL to search\_query will cause the server to make outbound requests (SSRF).



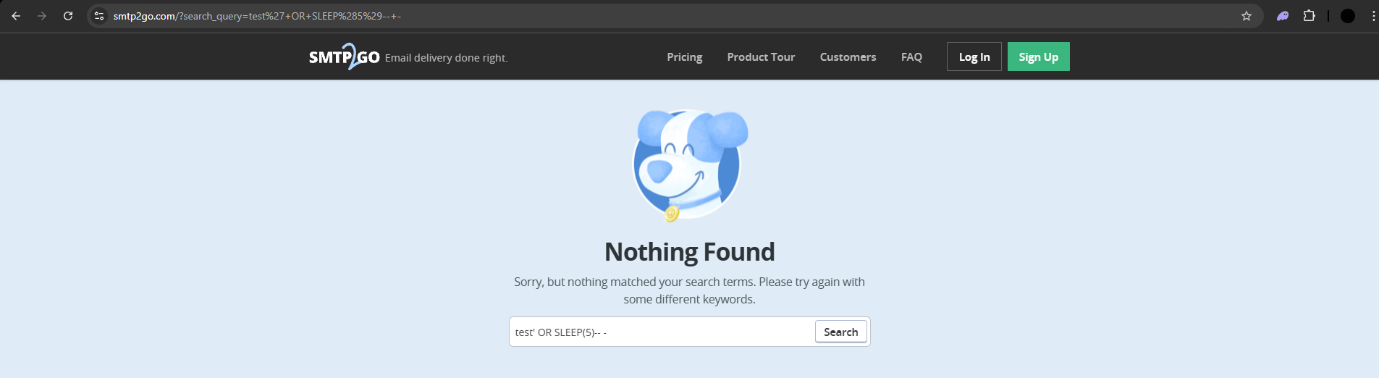
First, np.edu.sg site was submitted to the search query. The site rendered a “Nothing Found” page with the site reflected in the search box, showing that ssrf does not seem to be working.

We can double check this by submitting a unique callback URL generated via web hook.site. However, there was no request shown on the webhook inbox, further proving that SSRF is not present on the site.



To conclude, No SSRF via search\_query detected. The parameter appears to be treated as plain text and does not trigger server-side retrieval of the supplied URL. Severity: Informational (not vulnerable).

## Testing for SQL Injection



A screenshot of a computer

AI-generated content may be incorrect.

The 2 images show the search\_query parameter reflects user input into the search results page, but the output is HTML-escaped. Tested with mySQL boolean and time-based payloads but they did not show any sign of delay in execution or behavioural differences. This means that there is no exploitable SQL injection here.

# Error Handling

## Testing for Improper Error Handling

When an error occurs such as a page is not found on the smtp2go.com domain, a custom 404 page with no information disclosure is displayed. When such an error occurs on the app.smtp2go.com domain, it redirects the user to <https://www.smtp2go.com/404?loc=app>.

The error handling is done securely and there are no findings here.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

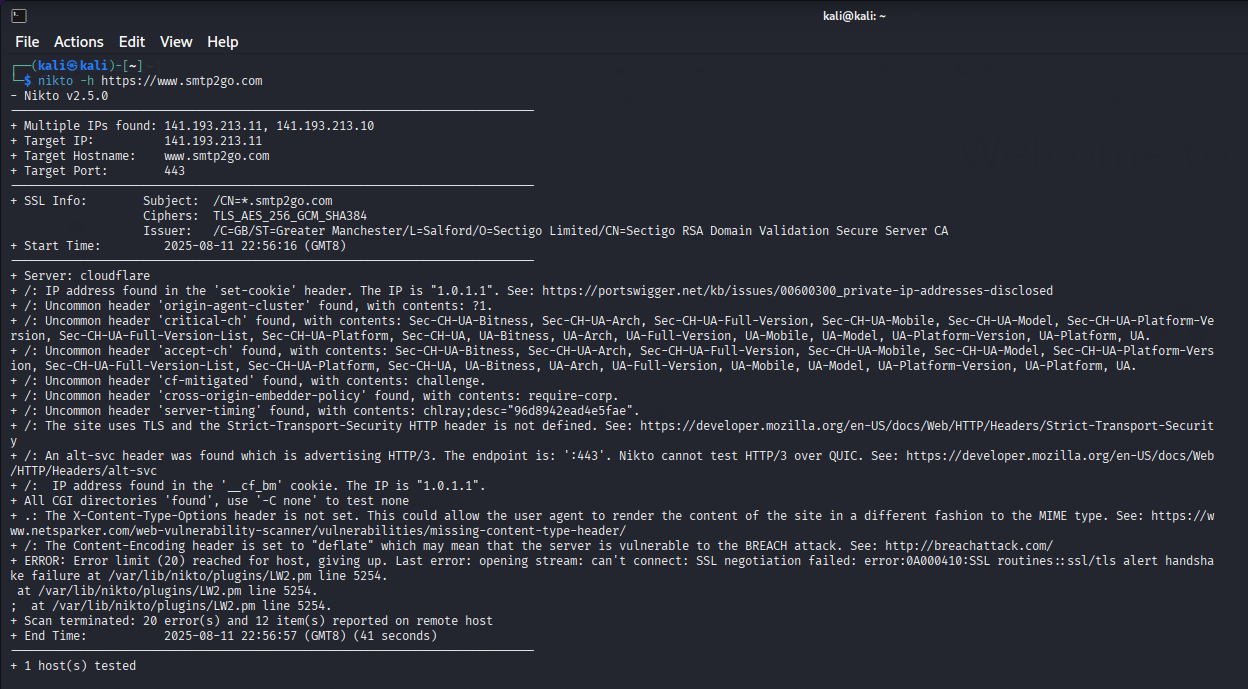
AI-generated content may be incorrect.Vulnerability scanning

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

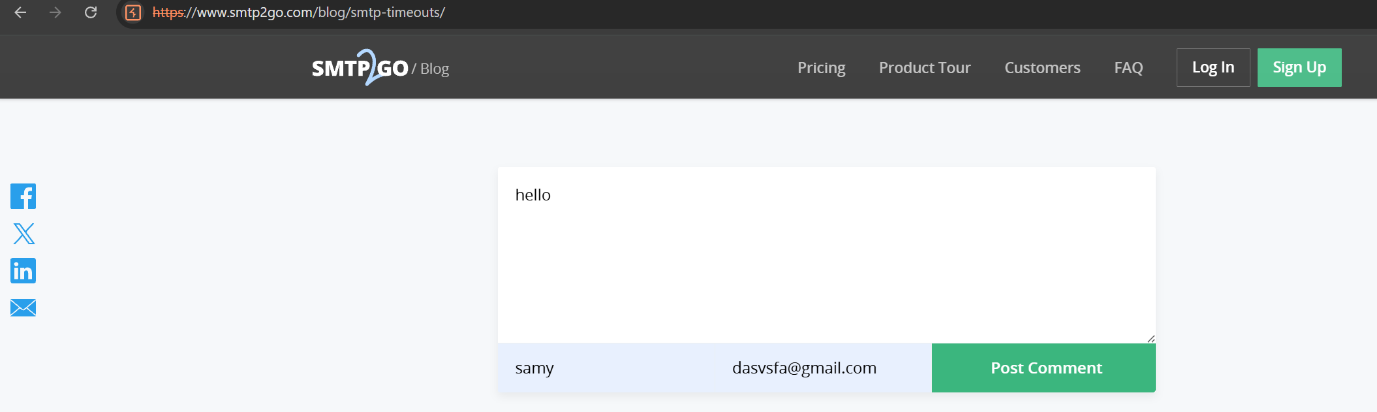


Vulnerability scan on <https://www.smtp2go.com> using nikto scanner.

# Comment Moderation

## XSS

Post a comment in “Timeouts” blog with burpsuite proxy intersection on.



Change the comment to a XSS payload, eg <script>alert(“test”)</script>

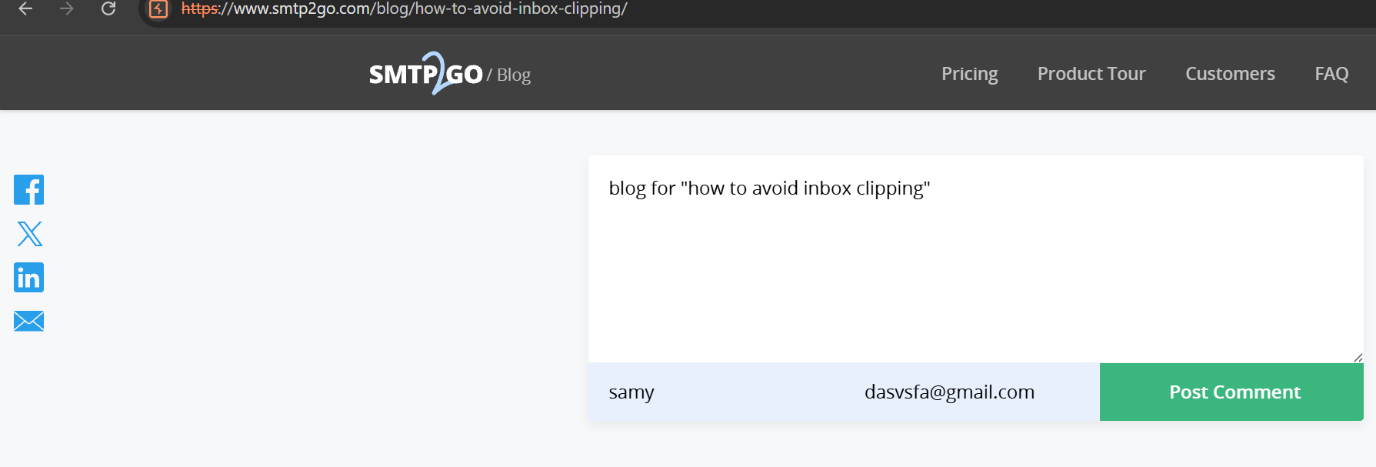


The unapproved parameter in the redirect URL indicates that the comment is pending review. Exploitation would only rely on an admin viewing the comment in the moderation panel. This makes it a stored XSS targeting administrators rather than regular users. This prevents malicious XSS payloads from being displayed to the public immediately, reducing the risk of the attack spreading to all visitors.

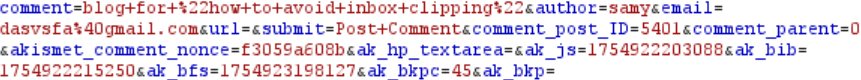


## IDOR

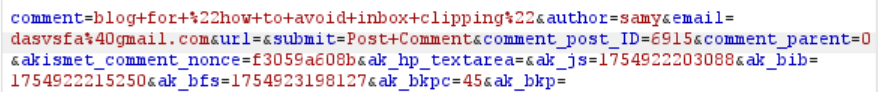
Post a comment in “how to avoid inbox clipping” blog with burpsuite proxy intersection on.

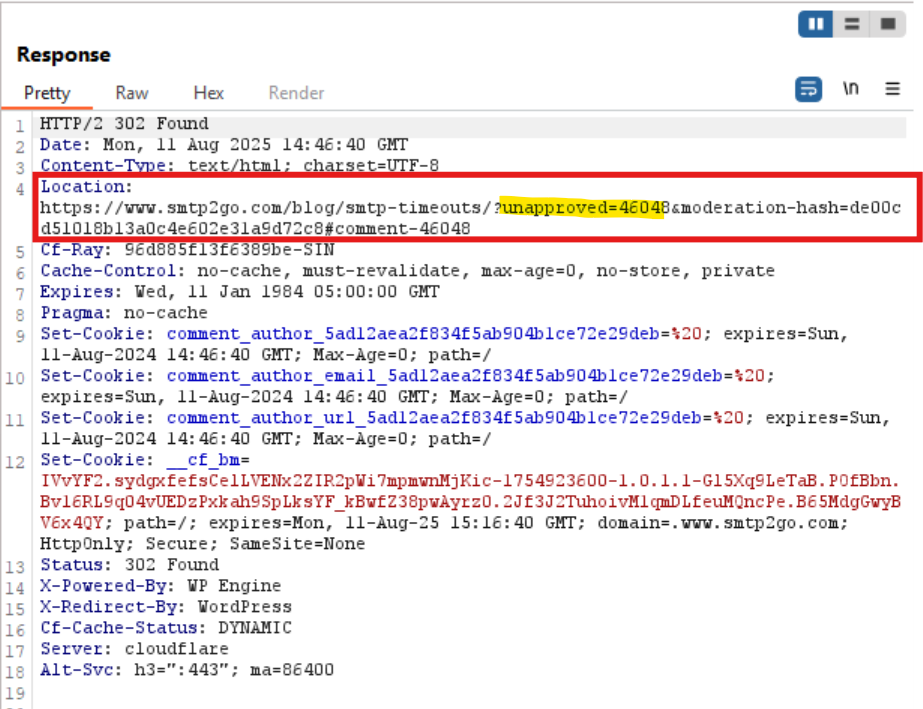


The comment post ID here is 5401 and the comment post ID for ”TimeOuts” is 6915



Change the comment post ID to TimeOuts postID 6915



/?unapproved=46048&moderation-hash=... means the comment is in the moderation queue and the comment was accepted into the database but it's not visible until an admin approves it. Which can be abused by automating requests with different comment\_post\_ID values to flood multiple posts with spam, overwhelming the moderation process  


In a nutshell, this moderation system helps the website prevent immediate exploitation during penetration testing or real-world attacks because malicious payloads are not instantly displayed to public users, giving administrators a chance to review and remove them before they reach a wider audience.

# Appendix

## Importing scope into BurpSuite

To ensure that the pentest stays in scope, the BurpSuite configuration file provided on SMTP2GO’s Bug Bounty Program on HackerOne is imported.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 1.1

## Passive Reconnaissance Results

<https://www.smtp2go.com/> - This is the root directory of the domain smtp2go.com.

<https://www.smtp2go.com/pricing/> - This is to sign up an account along with prices.

<https://www.smtp2go.com/tour/> - Talks about features of the application.

<https://www.smtp2go.com/customer-success/> - Reviews from other companies.

<https://www.smtp2go.com/faq/> - FAQ for the application.

<https://www.smtp2go.com/comparison/> - Compare application to other similar ones.

<https://www.smtp2go.com/blog/> - Displays blogs posted with a search feature.

<https://www.smtp2go.com/setup/> - Generic instructions to help set up the application with a search feature.

<https://www.smtp2go.com/setupguide/> - List of detailed set up guides.

<https://www.smtp2go.com/contact/> - The feature to send a message to ask a question or give feedback to the company. Also provides contact information.

<https://www.smtp2go.com/about/> - Information about the company.

<https://www.smtp2go.com/terms/> - Terms of service.

<https://www.smtp2go.com/privacy/> - Privacy policy.

<https://www.smtp2go.com/resellers/> - Information about a feature of the application.

<https://www.smtp2go.com/jobs/> - To send job application(s) to the company.

<https://app.smtp2go.com/login/> - Login page.

<https://app-us.smtp2go.com/dashboard/main/> - Home page to the main application.

<https://app-us.smtp2go.com/reports/summary/> - Summary of emails sent.

<https://app-us.smtp2go.com/reports/activity/> - Shows email activity.

<https://app-us.smtp2go.com/reports/charts/> - Displays charts for events, users, and demographic details.

<https://app-us.smtp2go.com/reports/suppressions/> - Block emails to and from a domain or email address.

<https://app-us.smtp2go.com/reports/data-exports/> - Export data and schedule email reports.

<https://app-us.smtp2go.com/sending/smtp_users/> - Manage users able send emails.

<https://app-us.smtp2go.com/sending/apikeys/> - Manage API keys.

<https://app-us.smtp2go.com/sending/verified_senders/> - Verify sender’s domain and email address.

<https://app-us.smtp2go.com/settings/display/> - Display settings of user.

<https://app-us.smtp2go.com/settings/sendingoptions/> - Sending email options.

<https://app-us.smtp2go.com/settings/webhooks/> - Manage webservers notified when an event happens.

<https://app-us.smtp2go.com/settings/templates/> - Email templates.

<https://app-us.smtp2go.com/settings/integrations/> - Integrate app with external services.

<https://app-us.smtp2go.com/account/team/> - Manage team members.

<https://app-us.smtp2go.com/account/2fa/> - Set up 2FA on the account.

<https://app-us.smtp2go.com/account/updatebilling/> - Account billings & invoices.

<https://app-us.smtp2go.com/account/changeplan/> - Select a plan for the application

## Google Dorking Results

Searching for any text files on the smtp2go.com domain, only 1 finding (robots.txt) was found as seen in Figure 2.1. ExploitDB did not show any results for smtp2go.com domain in their Google Hacking Database.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2.1

A close up of a computer screen

AI-generated content may be incorrect.

Figure 2.2

## Sitemap results

The full results of the sitemap findings are shown in the screenshots below:

A screenshot of a web page

AI-generated content may be incorrect.

Figure 3.1 – main sitemap page

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.2 – /post-sitemap.xml

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.3 - /page-sitemap.xml

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3.4 - /setupguide-sitemap.xml

## Banner grabbing

Here is the response given after sending a HTTP request to the api.smtp2go.com domain.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 4.1 – Response by api.smtp2go.com