**BUB BOUNTY**

A blue and orange logo

Description automatically generated

**IT NUMBER: IT22345332**

**NAME: G.P DINUJAYA THAMARA**

**WEEKEND BATCH**

**MALABE CAMPUS**

**Bug Bounty Platform – Hacker One**

**Bug Bounty Program - Booking.com**

**Scope**

**In Scope Assets**

For in Scope Assets please refer to the Scope tab

**Out-Of-Scope Applications** Any application whether owned by Booking.com or third-party vendor **not included as an in-scope asset** will be mentioned on the scope tab as out of scope.

For Out Of Scope Assets please refer to the Scope tab

**In-scope Vulnerabilities**

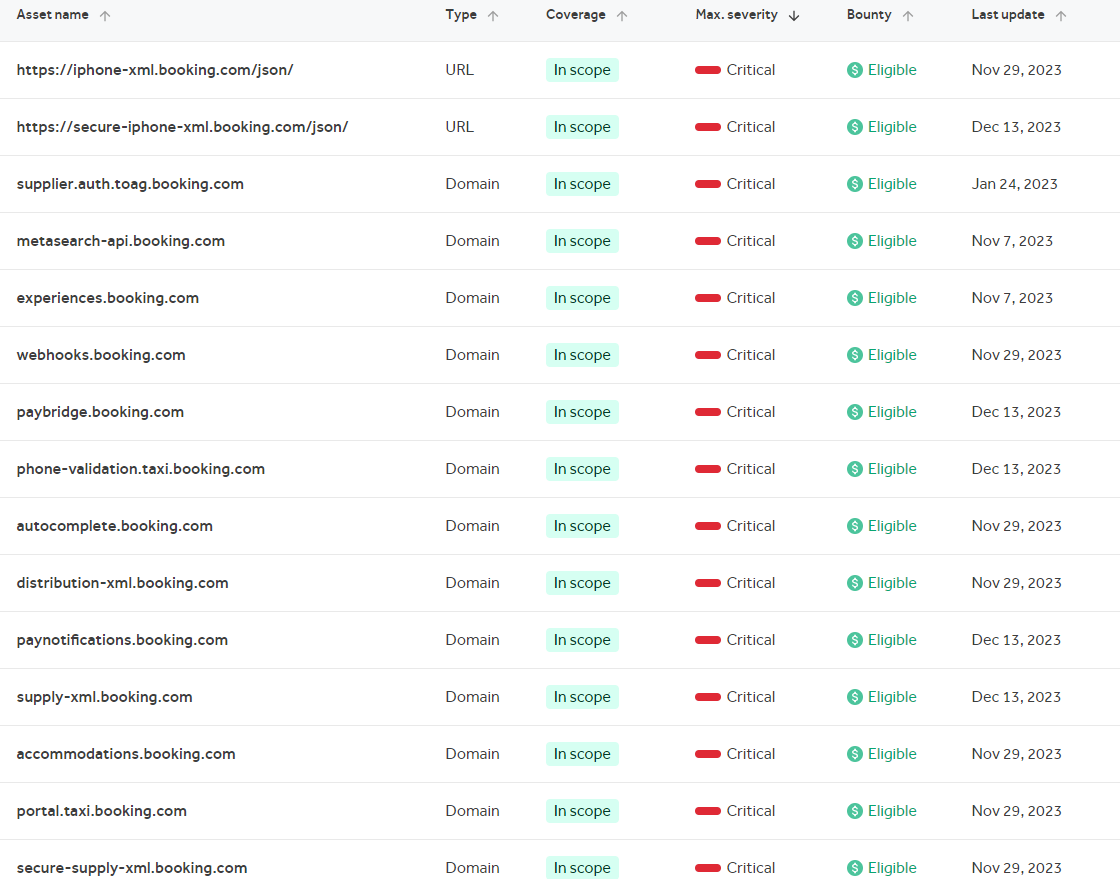
**Accepted, in-scope vulnerabilities include, but are not limited to:**

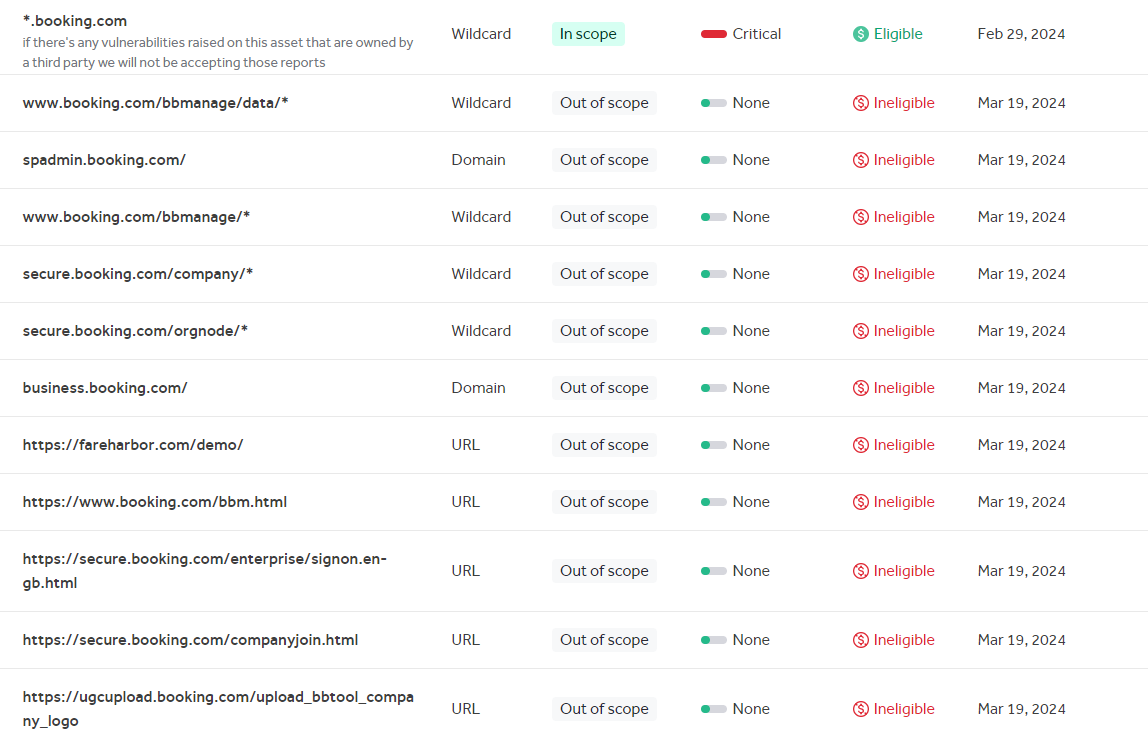
* Disclosure of sensitive or personally identifiable information
* Cross-Site Scripting (XSS) - Please note, for XSS if the same issue is reported for the different subdomains but with the same root cause, it will be considered duplicate
* Cross-Site Request Forgery (CSRF) for sensitive functions in a privileged context
* Remote code execution (RCE)
* Authentication or authorization flaws, including insecure direct object references and authentication bypass
* Injection vulnerabilities, including SQL and XML injection
* Directory traversal
* Significant security misconfiguration with a verifiable vulnerability
* Account takeover by exploiting a vulnerability
* SSRF
* XXE
* Subdomain takeover in \*.booking.com domains

**Out-Of-Scope Vulnerabilities** Depending on their impact, not all reported issues may qualify for a monetary reward. However, all reports are reviewed on a case-by-case basis and any report that results in a change being made will at a minimum receive recognition. Please note that our **program terms and rules of engagement** still apply.

**The following issues are outside the scope of our vulnerability rewards program:**

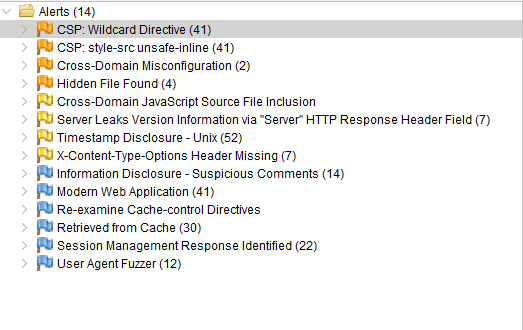
* Any vulnerability which requires access to a compromised email account or Booking.com account for successful exploitation
* Vulnerabilities on Third Party Products
* Attacks requiring physical access to a user's device or network.
* Forms missing CSRF tokens (we require evidence of actual CSRF vulnerability)
* Login/Logout CSRF
* Missing security headers which do not lead directly to a vulnerability
* Use of a known-vulnerable library (without evidence of exploitability)
* Reports from automated tools or scans
* Social engineering of Booking staff or contractors
* Denial of Service attacks and/or reports on rate limiting issues
* Not enforcing certificate pinning
* Any issues that require a rooted or jailbroken device or a compromised device
* Clickjacking
* Improper session invalidation
* User enumeration
* Host header injections without a specific, demonstrable impact
* Self-XSS, which includes any payload entered by the victim
* Any vulnerabilities requiring significant and unlikely interaction by the victim, such as disabling browser controls
* Content spoofing without embedded HTML or JavaScript
* Hypothetical issues that do not have any practical impact
* Infrastructure vulnerabilities, including:
* Issues related to SSL certificates
* DNS configuration issues
* Server configuration issues (e.g. open ports, TLS versions, etc.)





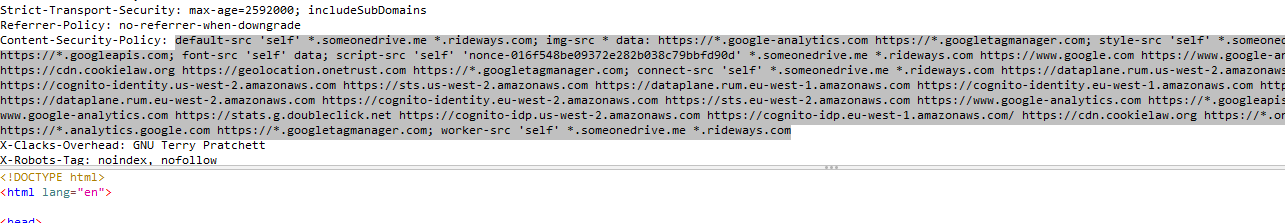
Portal.taxi.booking.com

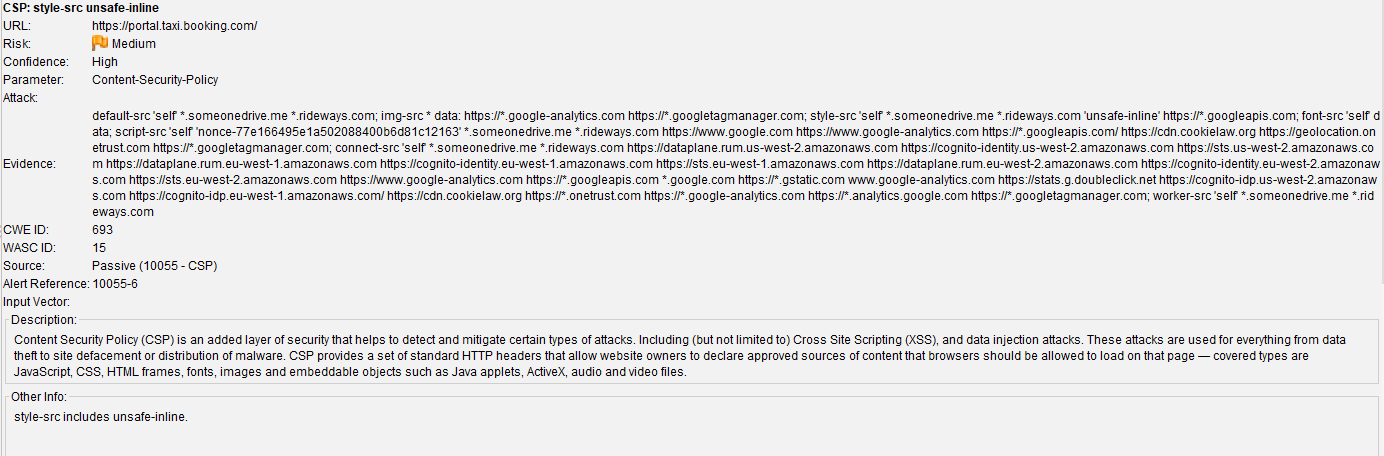
There are 27 misconfigurations in this sub domain.

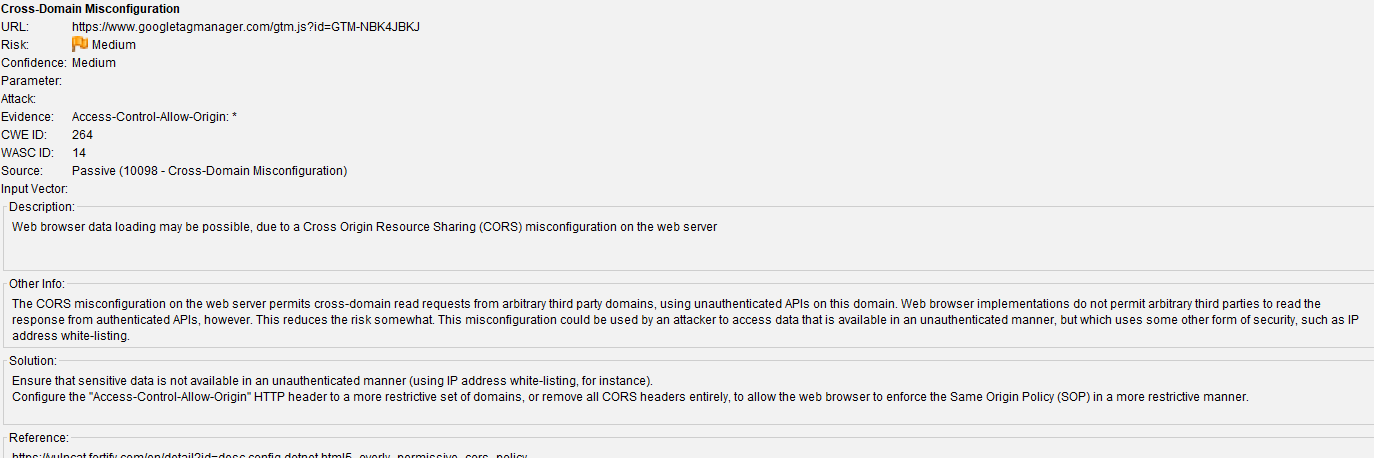


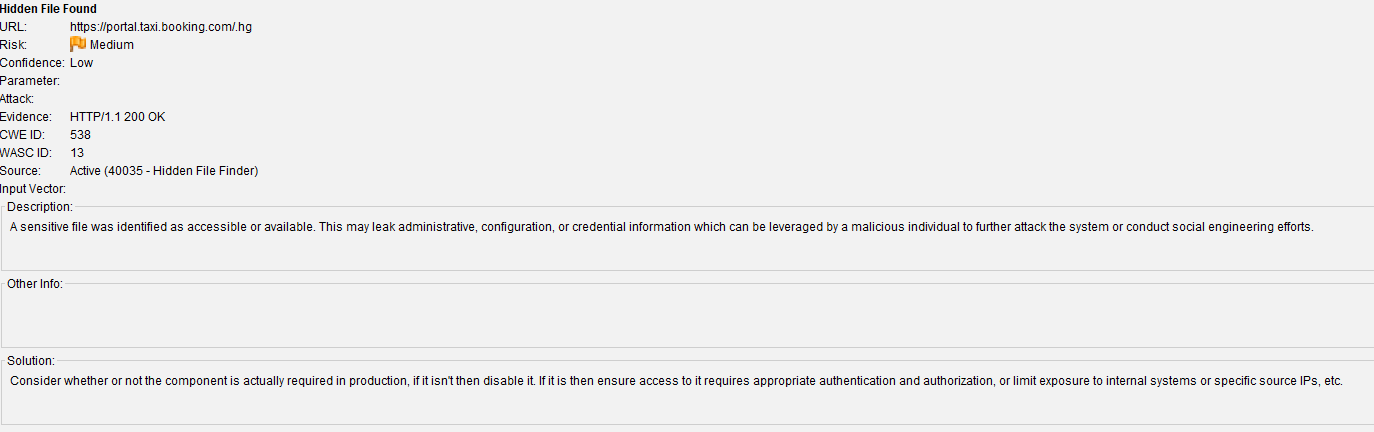
A screenshot of a computer error

Description automatically generated





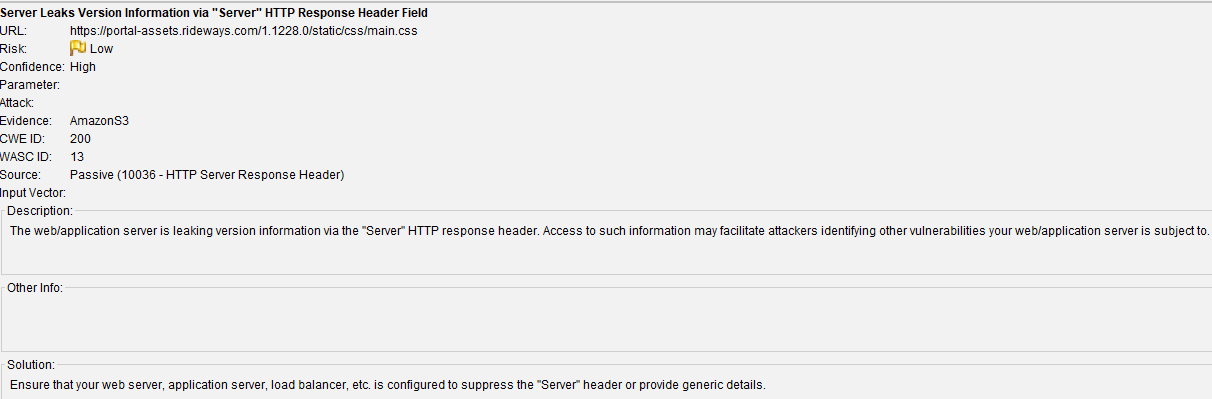




A screenshot of a computer

Description automatically generated







What is Amazon S3?

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements. According to <https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html>

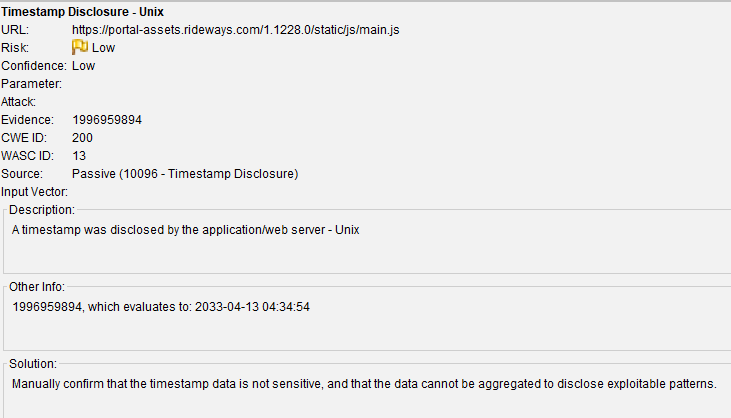
From this we can identify the following sub domain is using AmazonS3 as server lets check what are the specific vulnerabilities related toAmazonS3

According to the <https://cloudsecurityalliance.org/blog/2020/06/18/3-big-amazon-s3-vulnerabilities-you-may-be-missing> there are 3 main vulnerabilties

1: List permissions on Compute Resources

2: An over-reliance on IAM to prevent data theft

3: Non-public S3 buckets that contain public objects





To remediate the vulnerability of timestamp disclosure in Unix, the following steps can be taken:

Disable timestamp disclosure: Modify the Unix server configuration to prevent the disclosure of timestamps by the application or web server. This can typically be achieved by adjusting the server’s logging settings or by disabling the specific feature that is causing the disclosure.

Example for Apache HTTP Server:

**# Disable timestamp disclosure in Apache access logs**

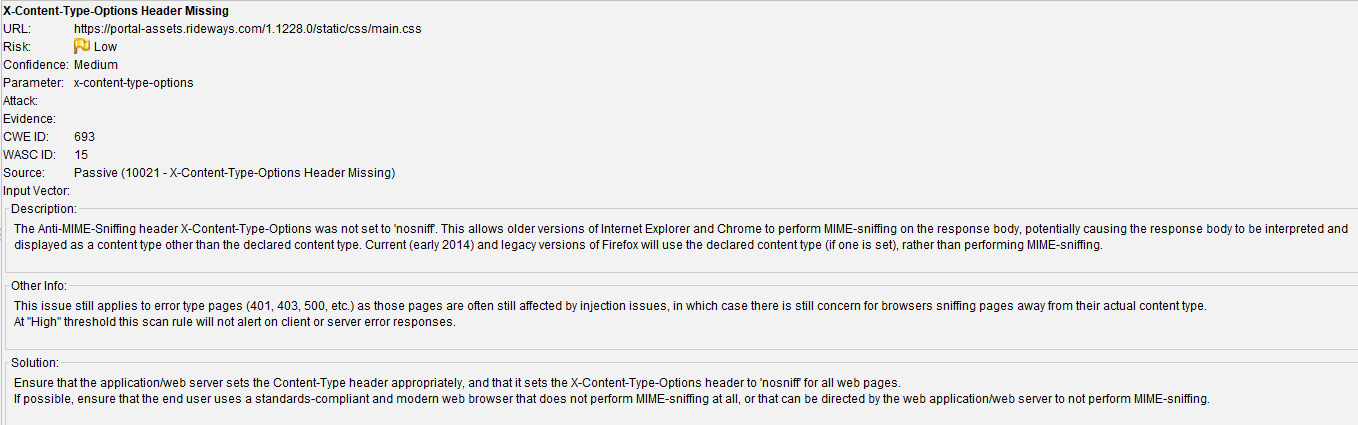
**LogFormat "%h %l %u %t \"%r\" %>s %b" common**

**CustomLog /var/log/apache2/access.log common**

Regularly update and patch the server: Keep the Unix server up to date with the latest security patches and updates. This helps to address any known vulnerabilities, including those related to timestamp disclosure.

Implement access controls: Ensure that appropriate access controls are in place to restrict access to sensitive information, including timestamps. This can involve configuring file permissions, user privileges, and network security measures.

<https://docs.stackhawk.com/vulnerabilities/10096/#:~:text=The%20vulnerability%20of%20timestamp%20disclosure%20in%20Unix%20occurs%20when%20an,server%20logs%20or%20error%20messages>.

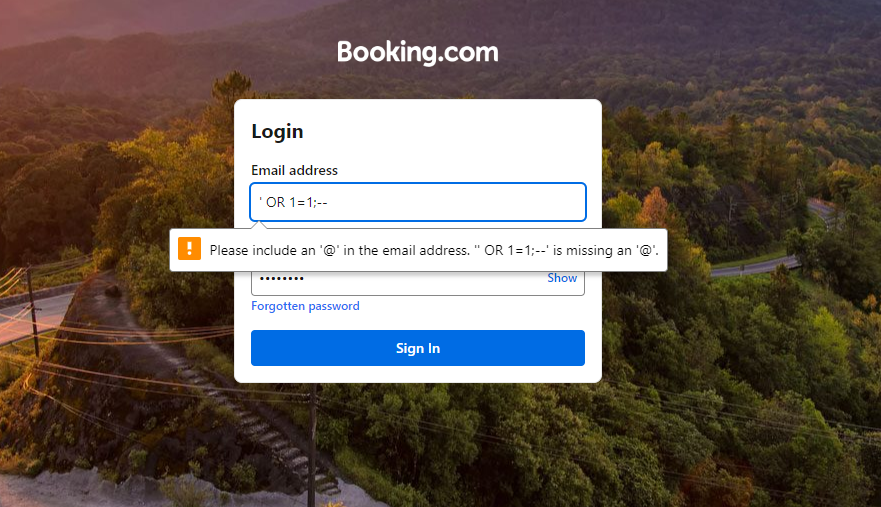


This Sub Domain uses following technologies. This is identified through Wappalyzer.



Let manually test the site manually.

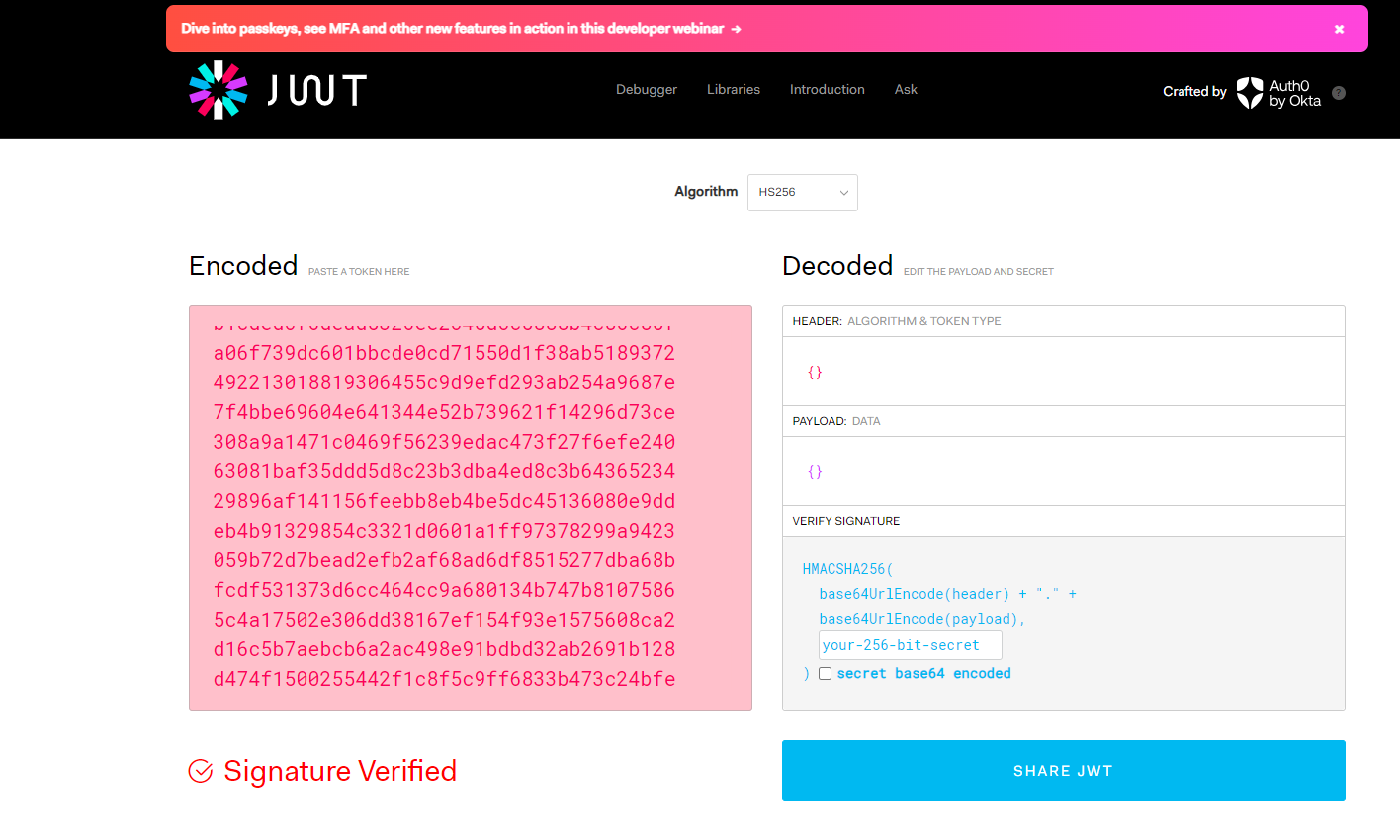
First inject the basic sql attacks to it was unsuccessful.

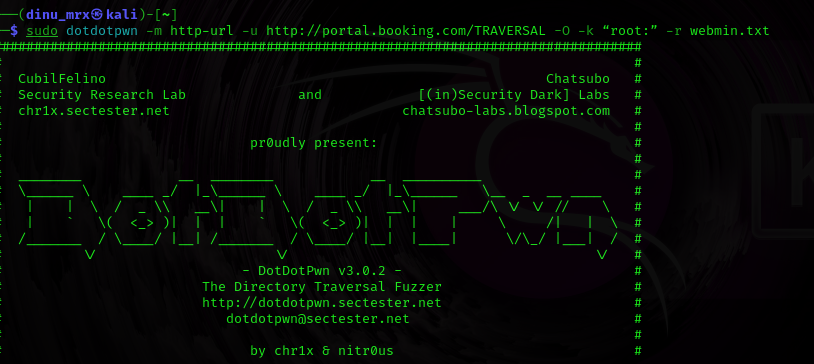


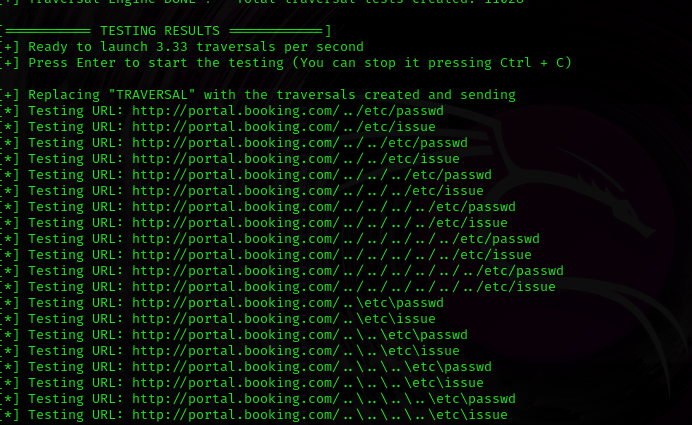
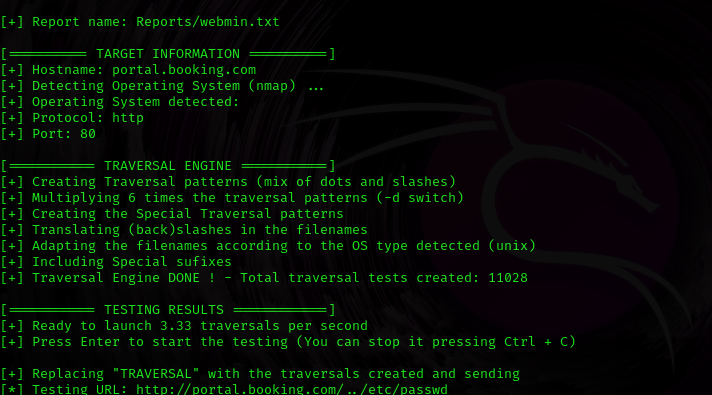
Catching the request when verifying the user

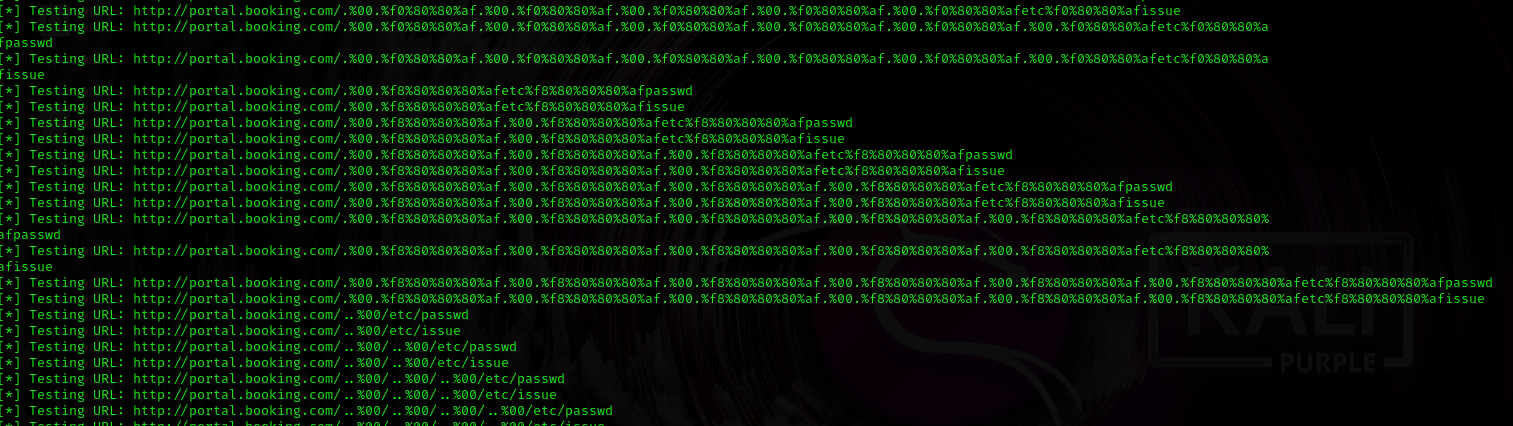


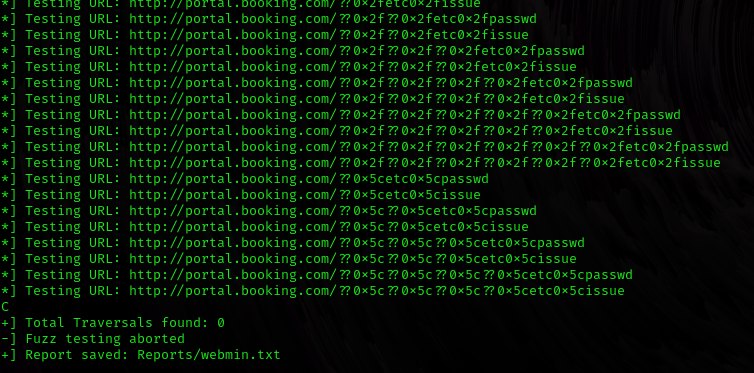
It is not decoding.



Lets try DotDotpwn to check it is vulnerable







This sub domain is invulnerable to directory traversal.