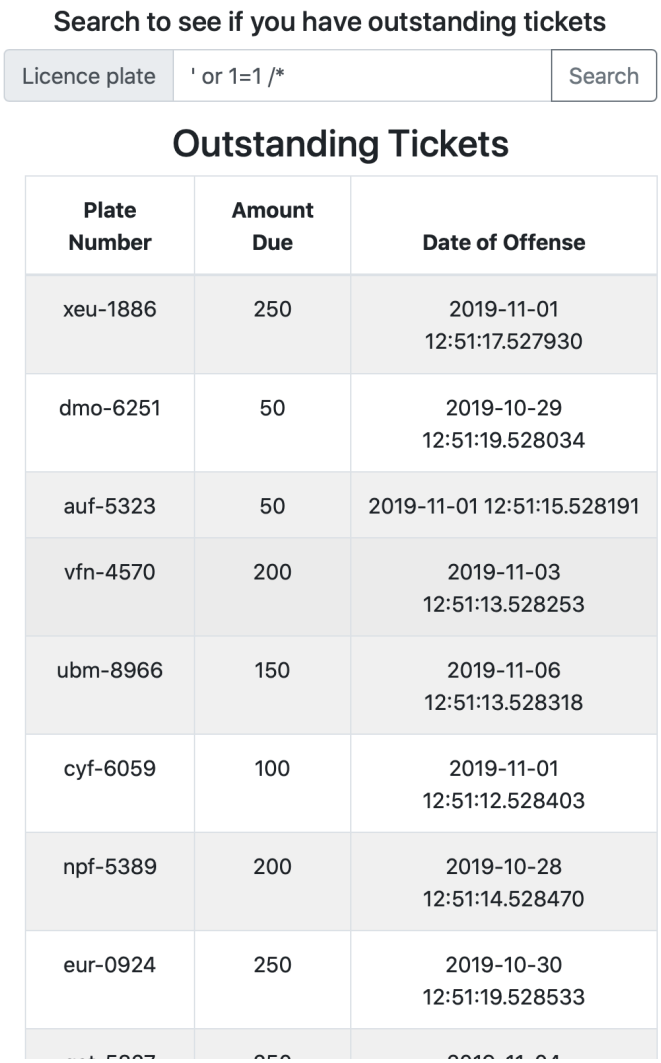
Estimate for Statement of Work #1 (Infrastructure and WebApp Audit)

Statement of Work Estimate, (the “SOW”) was prepared November 8, 2019 by **AstroX** “Jesús Fragoso”, for City of New City (the “Company”).

Firstly, we found a couple of vulnerabilities on the government websites of New City, starting out with the website http://611.newcity.city here we have 2 main problems:



The main one would be an issue with the Ticket Search subsite (located at http://611.newcity.city/search) on which there is a text field vulnerable to the following SQL injections: “ ‘ or 1=1 --”, “ ‘ or 1=1 /* ”, this is due to an issue with an unclosed quotation mark in the website’s php query to the SQL engine, which in this particular case would be sqlite. This exploit would allow an individual to potentially gain access to the complete database by using the UNION function, therefore gaining access not only to the main table, but also to other tables stored in the server, this is confirmed by the evidence on the left.

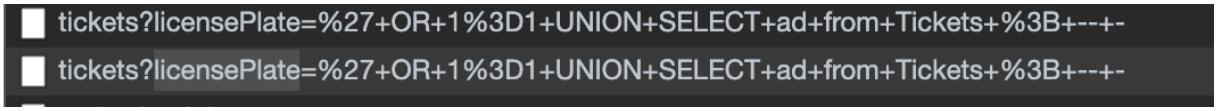
This vulnerability was found by using SQL injection techniques and by sketching out how the backend would respond to user sent queries to the server and building upon those. We started with a single quotation mark to find if we could inject code, later we found (thanks to an error code) that the database is built upon an SQLite infrastructure and that the frontend is based on the React JavaScript library.

Probable Backend:

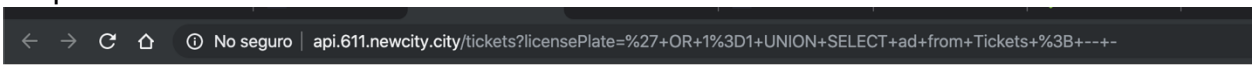
```
SELECT * FROM table WHERE x = ' or  1=1 --'
```

The error codes are the following:

Input 1:

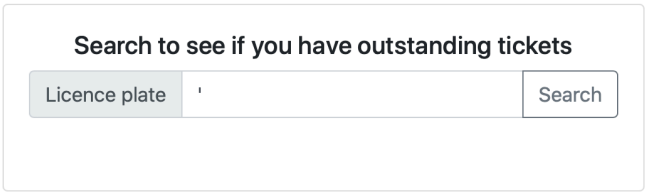


Output 1:

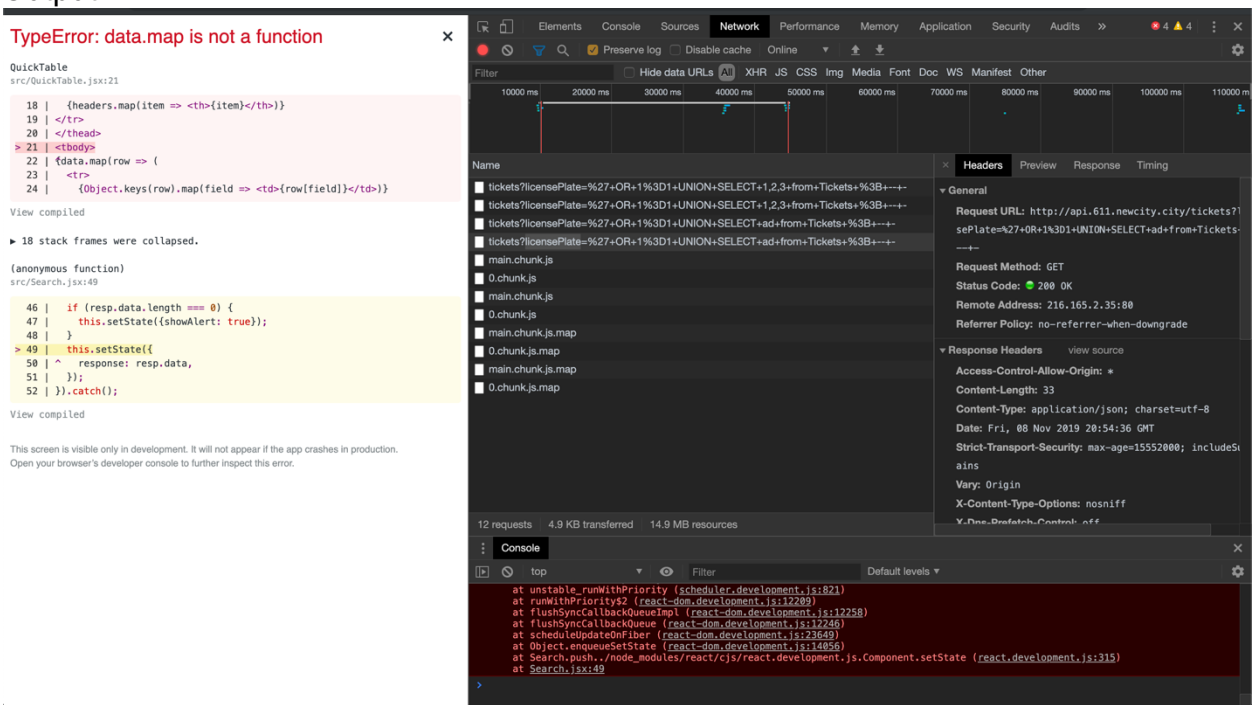


{"errno":1,"code":"SQLITE_ERROR"}

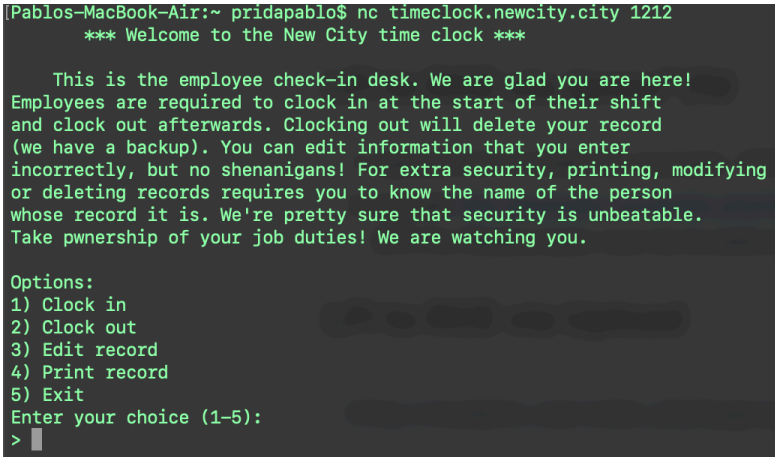
Input 2 : “single quotation mark”



Output 2:



The second issue found with this website is that, due to the fact that the Timeclock frontend is currently unavailable, the access to the timeclock program via the netcat access provided (nc timeclock.newcity.city 1212) is vulnerable to attacks. We also know that the program is running on a “gLIBC” library using the C programing language because of the access gained to the backend.



The second webpage access provided (http://payroll.newcity.city) is prone to be root accessed by anybody who is be able to perform a local file inclusion to the server via the cURL command and appending the POST function to the command and sending it off to the php login site (http://payroll.newcity.city/login.php) with a payload, either in data format, using the $GET command, or as a file sent to the server. We know this because of the fact that the server returns proof that the payload might be effective, because it tries to execute something in relation to the payload we send:

Input A

```
$ curl -d "uname=asd&pword=asd&login=uname" -X POST "http://payroll.newcity.city/login.php"
```

Output A

```
<!DOCTYPE HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no" />
        <meta name="description" content="" />
        <meta name="keywords" content="" />
        <link rel="stylesheet" href="assets/css/main.css" />
    </head>
    <body class="is-preload">
        Error message: File uname.php not found.            <header id="header">
                <a class="logo" href="index.php">CSAW RED 2019</a>
                <nav>
                    <a href="#menu">Menu</a>
                </nav>
            </header>

            <nav id="menu">
                <ul class="links">
                    <li><a href="index.php">Home</a></li>
                    <li><a href="logout.php">Logout</a></li>
                </ul>
            </nav>

            <section class="wrapper">
                <div class="inner">
                    <h2> Log in </h2>
                    <h1>Wrong username or password</h1>                    <form method="post" action="login.php">
                        <div class="row gtr-uniform">
                        <div class="col-7">
                            <label> Username </label>
                            <input type="text" name="uname" id="uname" value="" placeholder="Username" /><br>
                        </div>
                        <div class="col-7">
                            <label> Password </label>
                            <input type="password" name="pword" id="pword" value="" placeholder="Password" /><br>
                        </div>
                        <div class="col-12">
                            <button type="submit" name="login">Login</button>
                        </div>
                        </div>
                    </form>
                </div>
            </section>

            <script src="assets/js/jquery.min.js"></script>
            <script src="assets/js/browser.min.js"></script>
            <script src="assets/js/breakpoints.min.js"></script>
            <script src="assets/js/util.js"></script>
            <script src="assets/js/main.js"></script>

    </body>
```

Conclusion Reached:

The final note would be that there should be a mayor fortification of the whole system of new city, including database structure and front-end access. This is because the whole system is prone to be hacked by a third party.

The first amendment that should be put in place is the usage of techniques to prevent SQL injections and to fix the issue regarding the Timeclock program. Secondly, but most relevant, the php service at the login page should really be fixed to prevent foreign codes to execute at shell user level.

The physical evidence provided points to a corrupt electoral process in Mayor Jenkins's campaign, the files to back up such a claim are probably stored in the server, and whomever gets access to them could prove the suspected wrongdoings.