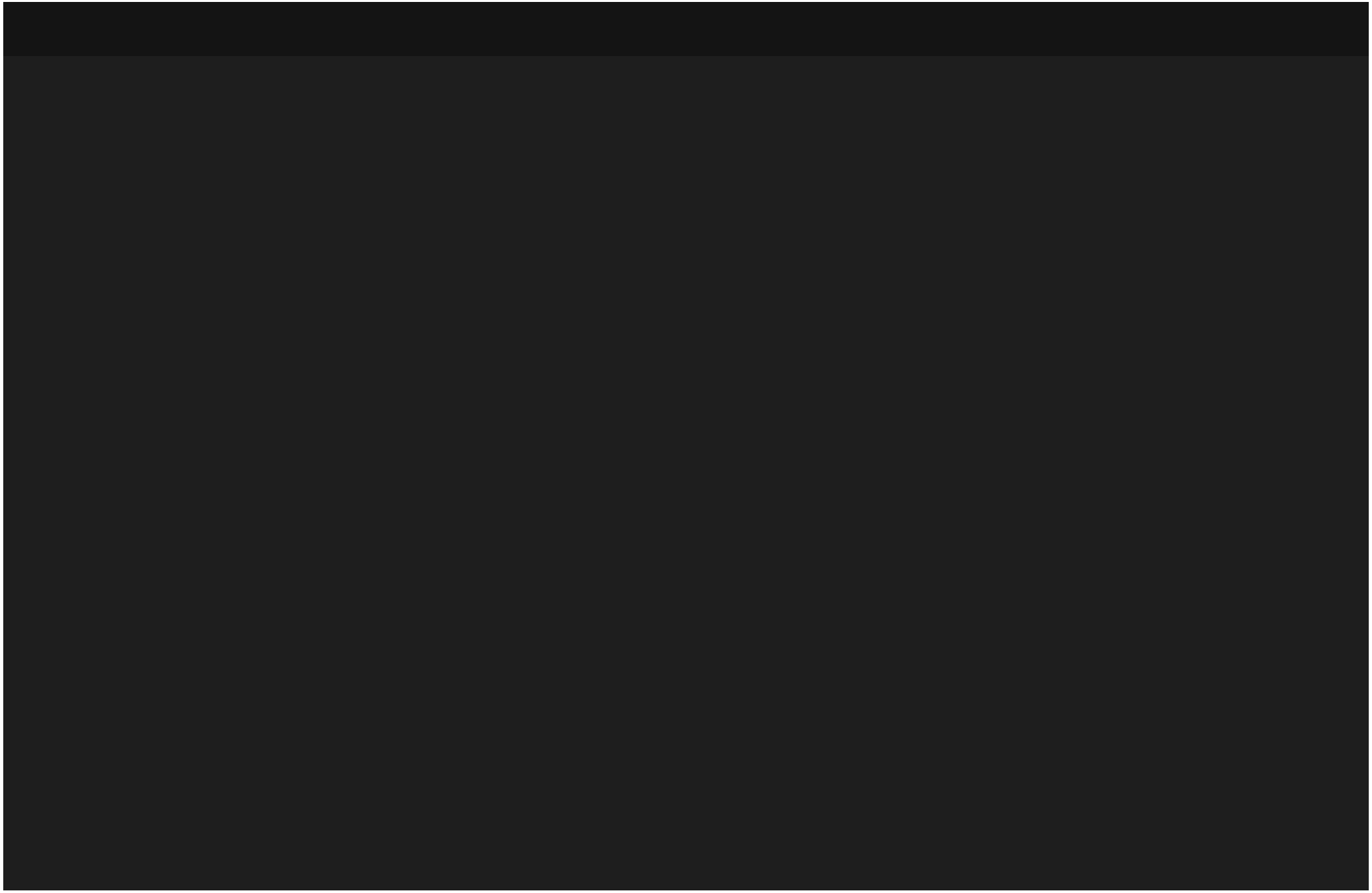
BUG HUNTING

Android App Hacking: Hardcoded Credentials

1. Unpack APK.
2. Recognize that it is a PhoneGap app.
3. View JavaScript source code to find hardcoded test credentials.
4. Login.

By bad5ect0r 17/05/2020 No Comments



Introduction

This was a relatively simple vulnerability I found for a company that deals with some potentially sensitive information. They offer paid services to their customers, but I was able to get free service by locating credentials in their Android application.

Obviously, I quickly disclosed this vulnerability responsibly and the company was very appreciative of my efforts. No bounty, but that's okay since I mainly hack for fun rather than profit 😜. I will definitely consider purchasing their services just to keep testing for them.

The Story

So as with any mobile app hacking, I started by downloading their APK onto my computer. You can easily do this with a service like APKPure. After that I ran `apktool` to unpack the APK:

```
[I] ✔ Test ls
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:29 2020 ▭  .
drwxrwxr-x osboxes osboxes   4 KB Sat May 16 23:37:31 2020 ▭  ..
.rw-r--r-- osboxes osboxes 6.3 MB Sat May 16 23:37:49 2020 珠  app
[I] ✔ Test apktool d app.apk
I: Using Apktool 2.4.0-dirty on app.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /home/osboxes/.local/share/ap
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
[I] ✔ Test ls
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:41 2020 ▭  .
drwxrwxr-x osboxes osboxes   4 KB Sat May 16 23:37:31 2020 ▭  ..
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:48 2020 ▭  app
.rw-r--r-- osboxes osboxes 6.3 MB Sat May 16 23:37:49 2020 珠  app
[I] ✔ Test cd app
```

```
[I] ✔ app ls
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:48 2020 ▭  .
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:41 2020 ▭  ..
.rw-rw-r-- osboxes osboxes 2.6 KB Sun May 17 00:15:45 2020 窍  And
.rw-rw-r-- osboxes osboxes 467 B  Sun May 17 00:15:48 2020 叫  apk
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:48 2020 ▭  ass
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:48 2020 ▭  ori
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:45 2020 ▭  res
drwxrwxr-x osboxes osboxes   4 KB Sun May 17 00:15:48 2020 ▭  sma
```

The first thing I checked was the AndroidManifest.xml file. Within this file, there was mention of Apache Cordova (PhoneGap). At that time, I didn't know what Cordova was other than hearing the name within app development circles:

```
1.  <?xml version="1.0" encoding="utf-8" standalone="no"?><mar
2.      <supports-screens android:anyDensity="true" android:la
3.      <uses-permission android:name="android.permission.INTE
4.      <uses-permission android:name="android.permission.ACCE
5.      <uses-permission android:name="android.permission.ACCE
6.      <uses-permission android:name="android.permission.ACCE
7.      <uses-permission android:name="android.permission.ACCE
8.      <uses-permission android:name="android.permission.BLUE
9.      <uses-permission android:name="android.permission.WRIT
10.     <uses-permission android:name="android.permission.RECC
11.     <uses-permission android:name="android.permission.MODI
12.
```

```
        <uses-permission android:name="android.permission.READ
13.     <application android:hardwareAccelerated="true" androi
14.         <activity android:allowTaskReparenting="true" andr
15.         <activity android:excludeFromRecents="true" androi
16.             <intent-filter android:label="@string/launcher
17.                 <action android:name="android.intent.actio
18.                 <category android:name="android.intent.cat
19.             </intent-filter>
20.         </activity>
21.         <activity android:alwaysRetainTaskState="true" and
22.         <receiver android:name="org.chromium.ChromeAlarmsR
23.         <receiver android:name="org.chromium.ChromeNotific
24.     </application>
25. </manifest>
```

With some research, I found a blog post detailing how source code could be extracted from IPAs and APKs built using Cordova/PhoneGap. TLDR: You can just go to `app.apk/assets/www/js/` to view source files.

When viewing `app.apk/assets/www/js/ViewModels/IndexViewModel.js` on this application, I was shocked to see commented out bits of code!

```
1   //var koBinding;
  1
  2 //var dataConnection = false;
  3 document.addEventListener('deviceready', onReady, false);
  4
  5 function onReady() {
  6     document.addEventListener("online", dataConnectionOnline, false);
  7     document.addEventListener("offline", dataConnectionOffline, false);
  8     setLocale();
  9     preAuthenticate(getValue(KEY_Auth));
 10 }
 11
 12 $(document).ready(function () {
 13     //setValue(KEY_IPAddress, "192.168.1.132");
 14     //setValue(KEY_IPAddress, "192.168.1.125");
 15     //setValue(KEY_IPAddress, "localhost");
 16     //setValue(KEY_APIBASEURL, "http://[redacted]:9002/");
 17     setValue(KEY_APIBASEURL, "https://[redacted]/");
 18     //setValue(KEY_APIBASEURL, "http://192.168.1.12:9002/");
 19     koBinding = new LoginViewModel();
 20     ko.applyBindings(koBinding, document.getElementById("login-box"));
 21
 22
 23 });
```

app.apk/assets/www/js/ViewModels/IndexViewModel.js

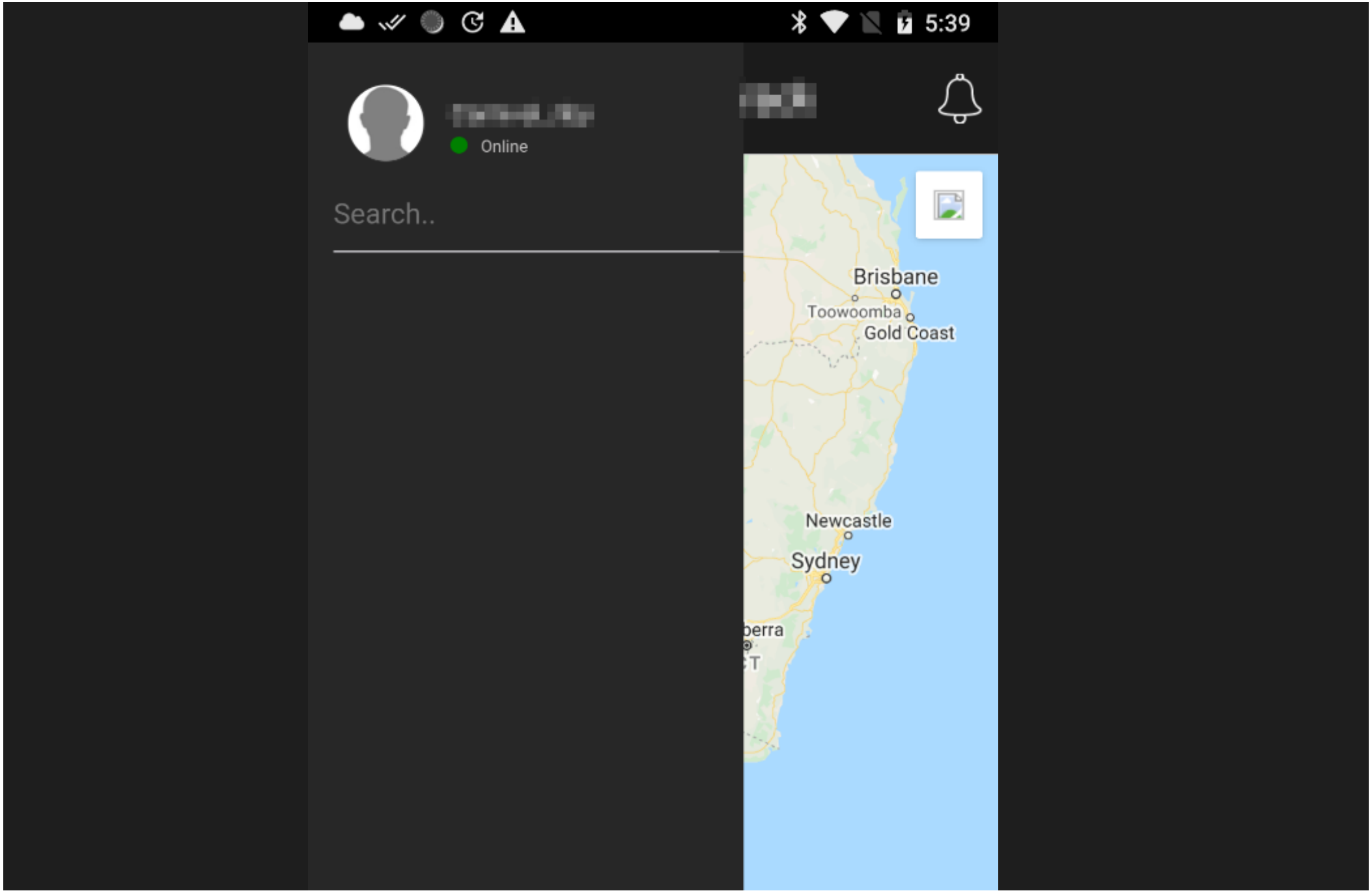
Among some of these comments, there was code that was assigning a username and password. Presumably this was done during testing to automatically authenticate the developer rather than having them manually type out the password each time:

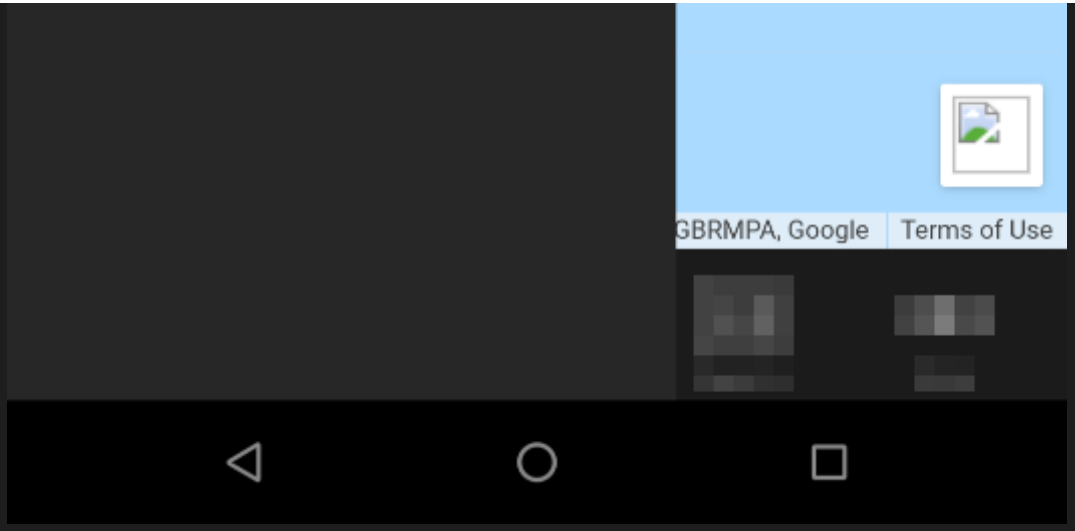
```
function LoginViewModel() {
    var self = this;
    //ko.utils.extend(self, new BaseViewModel());
    //local mob credentials following here
    //self.username = ko.observable("[redacted]");
    //self.password = ko.observable("[redacted]");
    //self.username = ko.observable("[redacted]");
    //self.password = ko.observable("[redacted]");
    self.username = ko.observable("");
    self.password = ko.observable("");
  //self.username = ko.observable("[redacted]');
  //self.password = ko.observable("[redacted]");

  //self.password = ko.observable("[redacted]");
    //self.username = ko.observable("[redacted]");
    //self.password = ko.observable("[redacted]");
    self.signinBtnClick = function () {
        self.username(self.username().trim());
        self.password(self.password().trim());
        if (isOnline()) {
            deviceReady();
            DisplaytrackedUser();
```

Plaintext credentials exposed in comments!

I tried logging into the app using these credentials but the first few failed so I started losing hope. That quickly changed when I tried the last one which allowed me to successfully authenticate!





ACCESS GRANTED

I immediately took screenshots and sent them over to the company. I found this over the weekend so no response in the first few days, but after everyone returned to work, they were quick to remove access for that account. They also updated their app packages to remove the credentials from the source code.

Takeaway

PhoneGap apps are fun to test since you get the source code!

Disclosure Timeline

- 21/03/2020 – Issue was reported to the company.
- 25/03/2020 – Follow up.
- 27/03/2020 – Acknowledged by the company.
- 03/04/2020 – Issues were fixed.
- 15/05/2020 – Partial disclosure was authorized.

Android, Cordova, Phonegap, Responsible Disclosure, VDP

← UTCTF 2020 – Observe Closely

Stored XSS Leads to Plaintext Password Disclosure →

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

POST COMMENT

 Powered by WordPress

To the top ↑