Whatsapp Account Hijacking (Exploiting QR-Code Login)

Himanshu Gwalani
(2017UCP1356)

CST 308 - Computer Network and Security
Malaviya National Institute of Technology, Jaipur

1 Introduction

This project consists of three modules :

- **Him's Bot**: This is a bot that could be used to exploit websites that use **QR** code as **Login** functionality. It establishes a connection with web.whatsapp.com and synchronizes the QR code with that on the phishing page. As the victim scans the code, it saves the exchanged information between phone and website. This information includes a **secret key**, **webId**(of browser's tab)**, version** of the browser, **location**, **type of OS**, etc (in JSON), and it is used for login from the attacker's system but disguised as the victim.
- **Face Authentication**: The above bot is secured using the face-lock mechanism, i.e. the bot will only execute when authenticated with my face.
- **Email Sender**: This is used to bait the user with a mail containing a link to the phishing page.

Web Whatsapp exchanges information with the phone as a **client-server model**, where the **phone** acts as a **server** while the **website** acts as a **client**. We bait the victim by sending a mail which includes a link to a phishing page containing QR code. This page is synchronized with web.whatsapp.com(using python scripts). Whatsapp reloads QR code every 15 seconds. The code is also reloaded on the phishing page. As the user scans it, the secret key (magic number) is captured by the **bot**, which is further used for log in to **web.whatsapp.com** disguised as the victim.
The major challenge is to synchronize the QR code (script on the attacker's side).

2 Log in with QR code

Authentication over most of platform's is dependent on traditional (id-password) based system. The credential system is dominating over others but with a few shortcomings like replay or phishing attack, password fatigue (remember the password for multiple applications), etc. To eradicate those came SSO (single sign-on) which is based on a single (id-password) pair for multiple services like google account. Then was introduced QR based login.
In a QR-code-based login, a user may only need to scan a QR code generated by the service he's trying to authenticate to, and then a client app on a trusted device such as a smartphone would scan and transmit the QR code to an identity provider in order to validate it and further authenticate the user to the destination service.
There could be several ways to authenticate using QR code :

1) The QR code may contain the URL of the server website, and when the phone scan the QR code, it will transmit the device id (maybe phone number), etc information to the server. The server will be notified that the account credentials linked with the device need to be shared with the corresponding website.
2) The mobile device can also transfer stored credentials directly to the website.

In this project I have exploited this login functionality over **WhatsApp.**

3 Process Flow of Attack

1) **Face Authentication**: This is required to begin the execution of him's bot. This includes a python script for face detection and matching with the existing information of my face stored as **flattened NumPy array in a text file**.

   It uses the KNN (K Nearest Neighbour) algorithm for matching the face.

   KNN: It is a supervised machine learning algorithm that can be used to solve both classification and regression problems.

   Hyper-parameter : k=5, distance formulae = 'Euclid's distance'

```
# s is test data

KNN(s):

        For each training data (tr):

                - Calculate distance between s and tr.

                - Add this distance with the label of s in an ordered set

                - Sort the set in non-decreasing order

                - Get the labels of first 'k' entries

- return mode of those k classes

end KNN
```

   Advantages:

   a) It is simple and easy to implement.
   b) No assumptions or training of the model is required.
   c) The algorithm is versatile. It can be used for classification, regression, and search.

   Disadvantages:

   a) The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.

   In this module, no scientific python library was used. KNN algorithm was implemented using naive python.

2) **Setting up the attack environment** :

Post authentication, him's bot will begin execution.

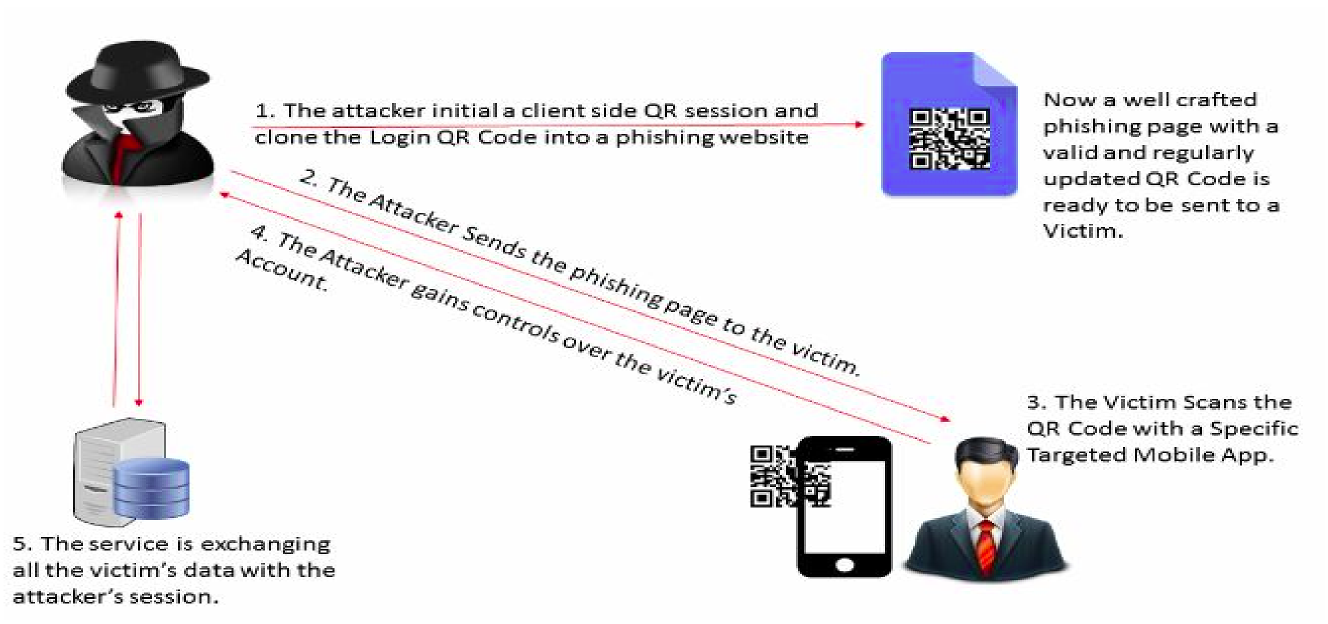
- The bot will initialize a client-side QR session. This will set up a connection with web.whatsapp.com. The site will respond as a web-page containing QR code. The bot will extract the QR code and initialize an ONCHANGE Listener over it. This QR will be cloned to our phishing_page.html. This will synchronize web.whatsapp's QR with phishing_page QR code.
- After this, the phishing website will be hosted on the attacker's (our) system on the port specified by the command line.

3) **Initialize attack** :

After the phishing website is hosted, an email is sent to the victim. The mail includes a link to the phishing page. As the victim clicks on it, he will be directed to the attacker's(our) website. This will be having the QR code. The victim will scan the QR code. This is done using the **smtplib** library of python. Features : **smtp_host** = 'smtp.gmail.com', **smtp_port** = 587,

victim's email : input

4) **Capture session information**: After victim scans the code, the phone will transmit data (magic number, location, etc.) to the server, and further the server will send the authentication credentials to our website. This information will be saved to our system in **JSON** format at /session/id.


1. The attacker initial a client side QR session and clone the Login QR Code into a phishing website
Now a well crafted phishing page with a valid and regularly updated QR Code is ready to be sent to a Victim.
2. The Attacker Sends the phishing page to the victim.
4. The Attacker gains controls over the victim's Account.
3. The Victim Scans the QR Code with a Specific Targeted Mobile App.
5. The service is exchanging all the victim's data with the attacker's session.

5) **Starting the victim's session**:

    Command : sessions -i 's_id'

    This will initiate the session with id = 's_id'. This will be done by him's bot. It will open firefox with URL as web.whatsapp.com. As the page appears, it will provide stored credentials for the corresponding victim. Since Whatsapp uses same (magic number) until it's session is ended manually from phone or desktop we will be able to access the victim's account even after restarting the browser. The only requirement is to keep the victim's phone connected to the internet since it is specified by Whatsapp.

And now, the victim's account is hijacked.

4 ClickJacking vs QR code Hijacking

Clickjacking, also known as a "**UI redress attack**" , is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when the user wants to click on the top-level page.

However, clickjacking will **not** be able to surpass **2-step verification**.

For example: If an attacker tries to somehow convince the victim to change email and password as per attacker, but if the authentication involves 2-step verification, the attacker won't be able to hijack.

On the contrary, the QR code login involves:

- SSO (Single Sign-On): use of the same credentials for multiple systems.
- 2-step verification

So QR code can be considered as the final defense line which provides usability and security. So QR code exploitation is more severe than normal clickjacking attacks.

5 Defenses

Here are a few proposed defenses for QR based session hijacking attack:

- **Use of OTP (one time password)**: This would be best to implement instead of QR based Login. Since the requirement in both cases is a device (phone) which the user can trust. So it is an efficient and much secure authentication mechanism.
- **Session Confirmation**: Implementation of a confirmation notification displaying characteristic information about the session made by the client/server.
- Not allow authentication for different networks (WAN's).
- **Audio-based authentication**: After the user scans QR code, the device (phone) generates audio which includes a message signed with the **private key** of the user. The website extracts the message using the **public key** of the user. If both messages are the same, authentication is successful. Using this, the attacker will not be able to extract the message, and the attack is unsuccessful.

6 Conclusion

In this project, I have successfully demonstrated QR based session hijacking over Whatsapp. The vulnerability of QR based login mechanism is exploited by capturing the magic number (secret key) and other necessary information for session establishment. This also includes the demonstration of face unlocks using the KNN algorithm. The technology stack mainly includes **python3**, **smtplib**, **JSON**.

The **main learnings** of this project were the implementation of KNN, extraction of session information using python scripts, synchronize web pages using threads, and communication between two applications (browser and him's bot) using **geckodriver**.

The current module of Him's bot only targets Whatsapp however, it could be further extended to applications that use QR based login mechanisms like WeChat, Yandex, etc.

7 References

- Mozilla developer resource on Content-Security-Policy frame-ancestors response header.
- https://owasp.org/www-community/attacks/Clickjacking
- https://brownglock.com/library/2016/08/02/qrljacking-attack-can-bypass-any-qr-login-system/
- https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
- https://stackoverflow.com/questions/8055132/python-sending-email-too-slow