Proposal: WhatsApp Web

Tom Curran, Romke van Dijk, Alexandros Tsiridis, Auke Zwaan

November 15, 2015

1 Introduction

WhatsApp is a proprietary cross-platform instant messaging client with more than 900 million users [5]. Released on January 21 2015, WhatsApp Web is a service through which WhatsApp can be used from any modern web browser. To configure the web client, a QR code is displayed on the website and scanned using the WhatsApp mobile application, pairing the browser to the device. Once paired, the web browser is authorized to read all messages stored on the device as well as send messages on its behalf [8].

2 Research Questions

As can be seen in section 6 there currently appears to be a lack of third party research on how this system exactly works and WhatsApp’s own documentation is both scarce and non-technical. For example, it is not clear how it is possible to access all the messages stored on the paired phone from the web browser and how the authentication mechanism operates. As a result it is difficult to determine that the implementation of that service is correct as well as if the application is secure. In addition, there is no information of the exact protocol used to achieve security and encryption.

Our research question is:

1. Are there any security vulnerabilities in the implementation of WhatsApp Web?

To answer this, the following subquestions will be answered:

1. Is it possible to eavesdrop on the communication between the service and phone?

2. Is the pairing mechanism secure, so that it prevents, for example, unauthorized access to someone’s messages?

3. Does the web browser accesses the messages stored on the mobile device securely?
As we are a group of four, there may be time for further research. The following subquestions are optional and time-dependent.

1. Is the access mechanism implementation in the client app secure?
2. Are there any potential vulnerabilities present within the current implementation of WhatsApp Web?
3. If so, is it possible to exploit them?
4. What are possible solutions to mitigate the vulnerabilities?

3 Approach

Our main approach will involve setting up a testing environment and then interacting with the application from both sides (web and mobile) in a structured way. Any requests made by the client or the phone in question will be intercepted and passed through. To set up this environment, our own OS3 servers will be used, as well as SIM cards and phones that are to be used exclusively for this project. In addition, a WiFi Pineapple will be used to set up a WiFi network that we can utilize to attempt man-in-the-middle attacks[8].

Work will be split amongst two areas; one focusing on the web client, and the other on the mobile side of the connection.

By splitting up the research on the communication in this way, controlled messaging can be done while maintaining a focus on a specific point within the communication flow. Each member has his own expertise and will be responsible for a specific part of this flow (i.e. sending receiving the messages). Besides providing the direct communication methods, the techniques used to create the front-end will also be examined using the OWASP Top Ten [3].

4 Planning

<table>
<thead>
<tr>
<th>Week</th>
<th>Task</th>
</tr>
</thead>
</table>
| 1    | Literature and technology Survey  
Research public information from the company  
Research methods  
Research tools needed  
Preparing the tools |
| 2    | Gather Data  
Apply methods  
Use the tools decided to obtain the data |
| 3    | Analyze the data gathered  
Conclusion |
| 4    | Write the report and presentation |
5 Ethical Issues

By Dutch law it is not allowed to do any reverse engineering on a computer program because this is considered as trying to copy the program. Also, the terms and conditions of WhatsApp states that users are not allowed to reverse engineer any WhatsApp applications or services[7]. However, the Dutch law (Auteurswet Article 45j) states the following about reverse engineering:

"Tenzij anders is overeengekomen, wordt niet als inbreuk op het auteursrecht op een werk als bedoeld in artikel 10, eerste lid, onder 12, beschouwd de verveelvoudiging, vervaardigd door de rechtmatige verkrijger van een exemplaar van eerder genoemd werk, die noodzakelijk is voor het met dat werk beoogde gebruik. De verveelvoudiging, als bedoeld in de eerste zin, die geschiedt in het kader van het laden, het in beeld brengen of het verbeteren van fouten, kan niet bij overeenkomst worden verboden." [2]

This means it is actually allowed to make a copy of (which also means to reverse engineer) a computer program for the purpose of finding or fixing mistakes in programs. Security issues are exactly this kind of problems, and this is what will be researched (see section 2).

In order not to involve personal information and messages of other users unaware of this project we will use four new SIM cards and four smartphones that will only be used for the experiment. Each device will have stored the contacts of only the other three. In this way, only the phone numbers and personal data of the test setup will be used.

6 Related work

It appears there is not yet any scientific research done on the inner workings of WhatsApp Web. There are however pointers given by non-scientific sources such as blog articles. Although any of the claims stated in such sources cannot be taken as valid input for our research, they may provide a step in the right direction and we attempt to verify them in the test lab.

One such web page describes the author’s personal findings when performing some initial steps in reverse engineering WhatsApp Web [6], suggesting the WebSocket protocol is used, which can be easily verified using the browser inspector in Google Chrome and was also confirmed by another - yet non-scientific - source [4]. This may be something to look into in more detail. With this knowledge, any implementation flaws can be looked for using the information given by [1] and other resources.

References


