APPLICATION ANALYSIS AND WEB SECURITY: A JOURNEY INTO THE MOST USED WEBSITES/APPLICATIONS TO SELL ITEMS ONLINE

Luca Pezzolla, Fulvio Serao

Università degli studi di Napoli “Parthenope”

Introduction:

The aim of this analysis is to verify the security level behind three of the most used applications/websites about marketplaces in Italy. We chose to analyze Vinted, Wallapop and Subito, because in the last few years they've become a landmark in their sector.

We focused on the HTTP/HTTPS packets analysis - exchanged by clients and the servers - with an eye open for the websockets (mainly used to speed up navigation).

Method:

We used two of the most known softwares, Charles Proxy and Burp Suite, to emulate a Man In The Middle attack to ourselves.

Charles is a web proxy (HTTP Proxy / HTTP Monitor), and thanks to its fake certificate, it allows anyone to display all of the data that is sent or received. Its most important feature is the SSL Proxying: it makes it easy to see SSL requests and responses in plain text, simplifying our job by a lot.

Burp Suite is a set of penetration testing tools for web applications. Although we didn't exploit its full potential, its proxy tool made us easy to look into each packet and analyze its content, allowing us to modify requests in real time. We were also able to see websocket communication, a feature which Charles Proxy lacks.

Each one of these operations has been made by using the applications/websites mentioned before and then we observed the results, respectively, on Charles Proxy or Burp Suite.

Results:

We divided our analysis into four categories, aiming to identify vulnerabilities into each one of them:

AUTHENTICATION -> We looked for login vulnerabilities: we tried to find whether it was possible or not to see our credentials after the HTTP POST sent to the server at access time.

USER DATA: WHAT CAN BE SEEN-> After the login phase, we focused on gathering as much data as possible about the accounts we looked for and the one we logged on with.

We started by looking for what sensitive data could be found. For sensitive data, in this context, we mean information that could not be obtained by only using the app or the website (because they shouldn't be available to anyone).

MESSAGING -> Finally, we tried to get information about user's messages we logged on with and tried to modify or delete them even though they had already been sent by the user himself. Additionally, we also tried to send new ones through the softwares mentioned before.

Vinted was the first one analyzed and probably the most vulnerable one when it comes to users' privacy.

At login time, we were able to see the user's credentials if he logged on the app/website with his email and password, but we couldn't see his credentials when he logged on with his Facebook or Google account.

When we looked for a specific user on Vinted, we obtained different sensitive data about him; among these:

Account's creation date, whether he verified his email/phone number, whether he used a pay method like credit card or other ones like Google Pay/Apple Pay.

Furthermore, when we looked for the so called "business accounts" we were also able to see their phone number.

When we clicked on our profile, we managed to see other relevant sensitive data such as:

Last four numbers of our credit card, its number, its expiration date and also some ciphres of our IBAN.

About messaging, we were able to send new messages to other users by just using Charles Proxy and an HTTP method called POST, modifying them with the same method. We were also able to delete them with another HTTP method called DELETE.

Finally, when it comes to security, whether it is about our credentials, our data or our messages, we definitely cannot feel safe when using Vinted's application or website.

Wallapop was the second application analyzed.

At login time, right as Vinted, we were able to see our credentials only when the user logged on with his email/username and his password, instead of when he logged on with his Facebook/Google account.

When we looked for a specific user on Wallapop, we were able to see whether he logged on with his Facebook/Google account or with his email/username and password just by looking at the specific flags displayed by Charles Proxy. By searching for a specific item, we were also able to see how many different conversations the seller had started for it.

The most sensitive data we could see about the account we logged on with were our phone number, our email, some ciphres of our IBAN and our credit card. The other ciphres had been hidden just by replacing them with the letter 'X'.

About messaging, Wallapop didn't use any POST method to send messages to other users so we weren't able to send new ones by using Charles or modifying the ones already sent. Since Wallapop doesn't allow his users to delete their messages ones they've already been sent, we couldn't do it either. The only thing we could effectively do was "intercepting" them by reading the GET methods which contained every conversation the users had already started. Thanks to Burp Suite, instead, we came to the conclusion that Wallapop used the so-called "websockets" to send messages from one user to another.

The third application analyzed was Subito, only used in our country, which was the most secure one, surprisingly.

At login time, just as Vinted and Wallapop, we were able to see our credentials only if we logged on with our email/username and password. In the other two cases we were just able to see a redirecting link which could be used to log in into the website only if we pasted it on a Browser where we had already logged on with our Facebook/Google account in the past.

When we looked for a specific user we couldn't see any sensitive data displayed about him, in contrast to what happened with Vinted or Wallapop.

About the account we logged on with, we couldn't see either his IBAN or his information about his Paypal account.

About messaging, we were able to intercept any conversation the user himself had already started just by looking at the GET method on Charles Proxy. When we sent a new message to another user on Subito, we were also able to intercept it by looking at the POST method displayed by Charles. Thanks to it, we managed to send new messages just by sending a new POST. We couldn't modify any of them or delete the ones already sent, though.