Team GRYBB 3
Daan Hanraads, s2621770
Diana-Alexandra Bozea, s2536196
Dirk van de Ruit, s2508303
Edina Depner, s2457261

Security report

Security breach	Covered?
Protection against malicious file uploads	No
Protection against Man-in-the-middle attacks	Yes
Protection against Link Injection Protection	Yes
Protection against Attribute autocomplete	No
Click hijacking protection	No

Protection against malicious file uploads

Some web applications allow you to upload files. This means that we could upload a file with a malicious script. That server where you have uploaded the file may be compromised. Normally, and this is recommended, the servers usually have different characteristics to validate the uploaded files. For example, detect possible extensions that may be a danger or the type of content.

Due to the fact that the application does not require file uploading, we have no reason to cover this topic. So that is why we don't have this covered

Protection against Man-in-the-middle attacks

A potential intruder could access website traffic that is transferred in plain text. This is a major privacy issue for users. Basically, an attacker can intercept requests to send information. If we send a message, upload a file or any kind of request, it can be intercepted. This occurs when the traffic is going over HTTP. This way it is not encrypted.

We have protected our system from Man-in-the-middle attacks by using prepared statements for the database relations for both updating and fetching information. Furthermore, the crucial data such as the passwords have been hashed and stored as such. Because our application does not handle a lot of vital information, Man-in-the-middle attacks are not considered a big threat.

Protection against Link Injection Protection

Another major problem affecting applications and websites is link injection. This could endanger our security and privacy, since we could be accessing a link controlled by hackers. How does this happen? It basically means that cyber criminals inject fraudulent links on that site. In this way, when the victim enters and accesses that link, he is not really entering a website or section that is legitimate but is directly accessing a page or server that is controlled by the attackers.

In order to protect the application against such attacks, we have used links that show the user exactly on what part of the system they are. The links we have prepared and crafted do not leave room for injecting a malicious piece of code without the user's knowledge.

Team GRYBB 3
Daan Hanraads, s2621770
Diana-Alexandra Bozea, s2536196
Dirk van de Ruit, s2508303
Edina Depner, s2457261

Protection against Attribute autocomplete

It is also another type of attack to abuse the autocomplete attribute that is usually set to off mode. The point here is that a potential attacker could have it activated, and this would allow the browser to cache information entered by the user. What could happen to this? A possible attacker would have access to the username and password entered in the browser's cache.

We do not have protection against attribute autocomplete because we want our fieldworkers to access the application quickly. Typing in a password with gloves is very difficult so we thought it would be better to let them enter it, perhaps also using a password manager system. Also we assume every field worker gets his own tablet to be able to water the trees. Finally we think that damage that can be done when a tablet is stolen is very small and not worth losing ease of access for.

Click hijacking protection

This type of attack is present mainly on platforms such as social networks. It means that an attacker has managed to infect that site with the aim of hijacking clicks. This means that if a person clicks on an element of that platform, they could end up on a site controlled by the attackers and put their security and privacy at risk.

This protection would be very important for an application where payments are done or important/private information would be shared. We assume all users know that when they are redirected to a site that asks for this kind of information our user would know from experience that our application does not ask for this kind of information. So we do not have protection against these attacks. For other pop-ups.In the case of pop-ups such as “click for a free iphone” or a sudden download button, our users would immediately know they are malicious because it is a work application with users who know it does not belong there.