Scummy Russian Hackers

# Cookie usage storage:

To handle concurrent users on the web app we decided to use cookies to remember user’s login details. The complete list of data stored is as follows: username – any sort of email address – and user type which may be user, admin or staff. Both fields are stored as plain text in the cookies.

To ensure the security goal of integrity, before any confidential code is loaded from the database, the username and type are checked against their stored values in the database. This prevents the user from escalating their privileges by changing their user type. In the case where the stored user name and the user type do not match, the user is presented with an error. Thus, the data of the person is hidden altogether till the cookie data is for a valid user. This is achieved by scanning the database for the username and checking the user type in the database to the one held in the cookie.

# Web Application Firewall (WAF)

The web application has a firewall implemented. Every piece of data given to the site passes through this filter. Additionally, special data such as usernames and passwords are sent through this filter. First, through the attack filter and then through a check for that field.

The way the firewall works with the frontend of the site as described in the following. The firewall runs as a webserver on the same machine as the front end. To distinguish between servers, a unique port number is assigned to it. The reason it is implemented as a standalone unit is that all sites running the machine can access the one central filter. This makes for higher security as this unit can be thoroughly tested. Also, since there is only on copy, it makes it easier to maintain the security of all sites.

../../../../../../Desktop/Screen%20Shot%202017-10-08%20at%2010.01.40%The filter checks the data for any potential attack; it could be XSS or a SQL injection. If the string is parsed and is clear of attacks, the server returns the result “True” back to the frontend, otherwise, a precise message is sent back detailing the error. An attack string is defined as one having illegal substrings, which are defined in the filter.  
  
Testing was done, it was found that an empty input string would result in the filter producing a 404 error. Bottle’s dynamic address resolution require that the dynamic part have one or more characters. To cope with this situation, I added another route path to make sure the empty string case was handled. e,g,   
  
  
The data passed between servers, happens only on a local machine, meaning that unless someone had access to that machine, all data passed between the web app and the filter will remain confidential.

# Interaction between WAF and the backend database

All input to the site is passed through the WAF before being processed by the application. This means that before the backend gets access to any of the data, it has already been checked for threats. This means that all data in the data base is threat free. Based on this assumption, and the fact that the website has no dynamic content outside of that provided by the database, there are not checks on data outside of the input. Simply requests make no interaction with the WAF.   
  
This decision was made because, the attack that could enter the system must occur by providing some input. Since this input is analysed, all data that successfully makes it into the system can be trusted.

* What data do we store in cookies
  + How are they handled and what checks are done to ensure they haven’t been changed.
* Communication between front end and waf server
  + Just talked about how it works.
  + Why it is designed the way it is.
  + How communcication happens between the waf and the front end.
* Interaction between waf and backend
  + Pretty much talked about how the waf comes before all operations (including the backend)
  + All data in the system and in the backend can be trusted since it must have already passed through the waf.
* What security decisions have been made