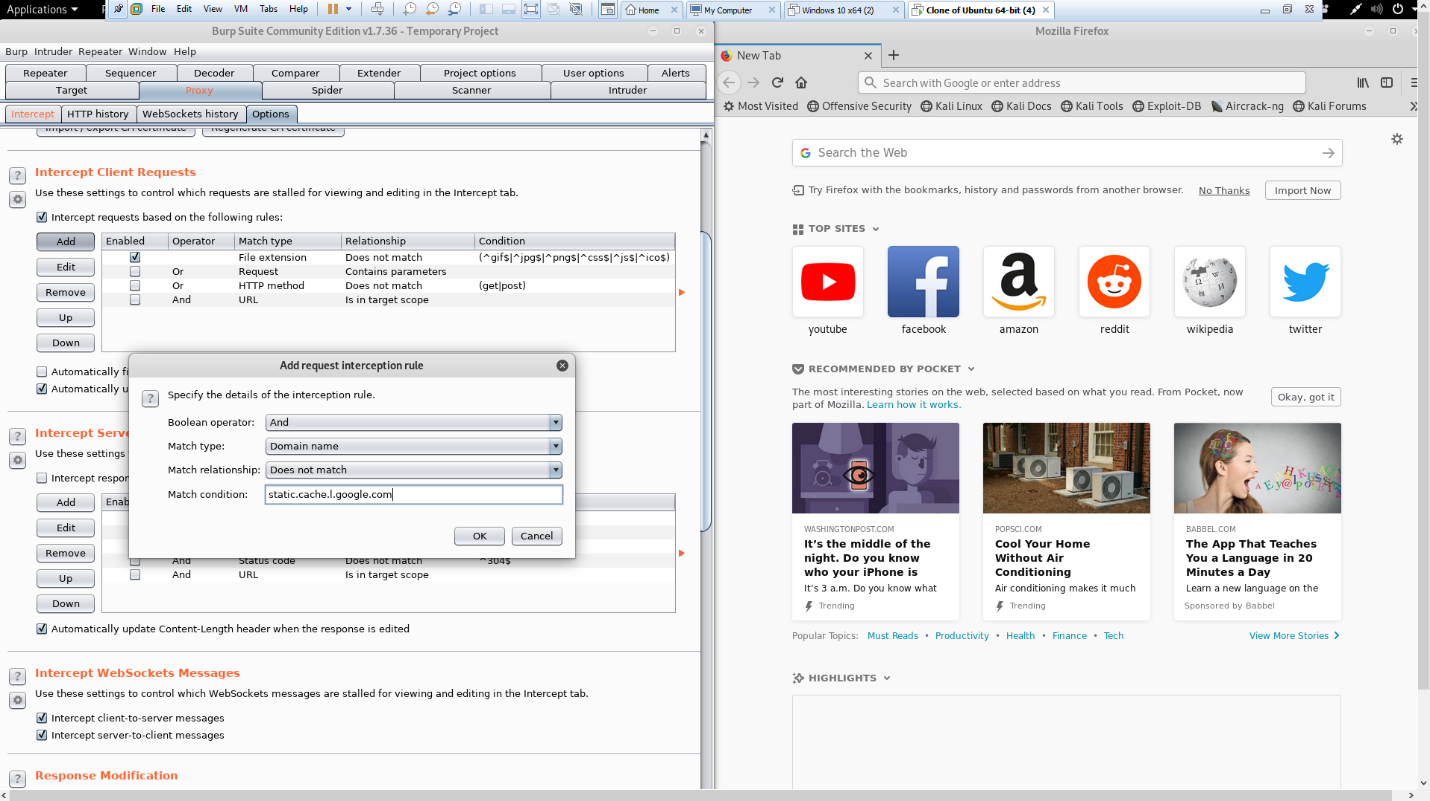
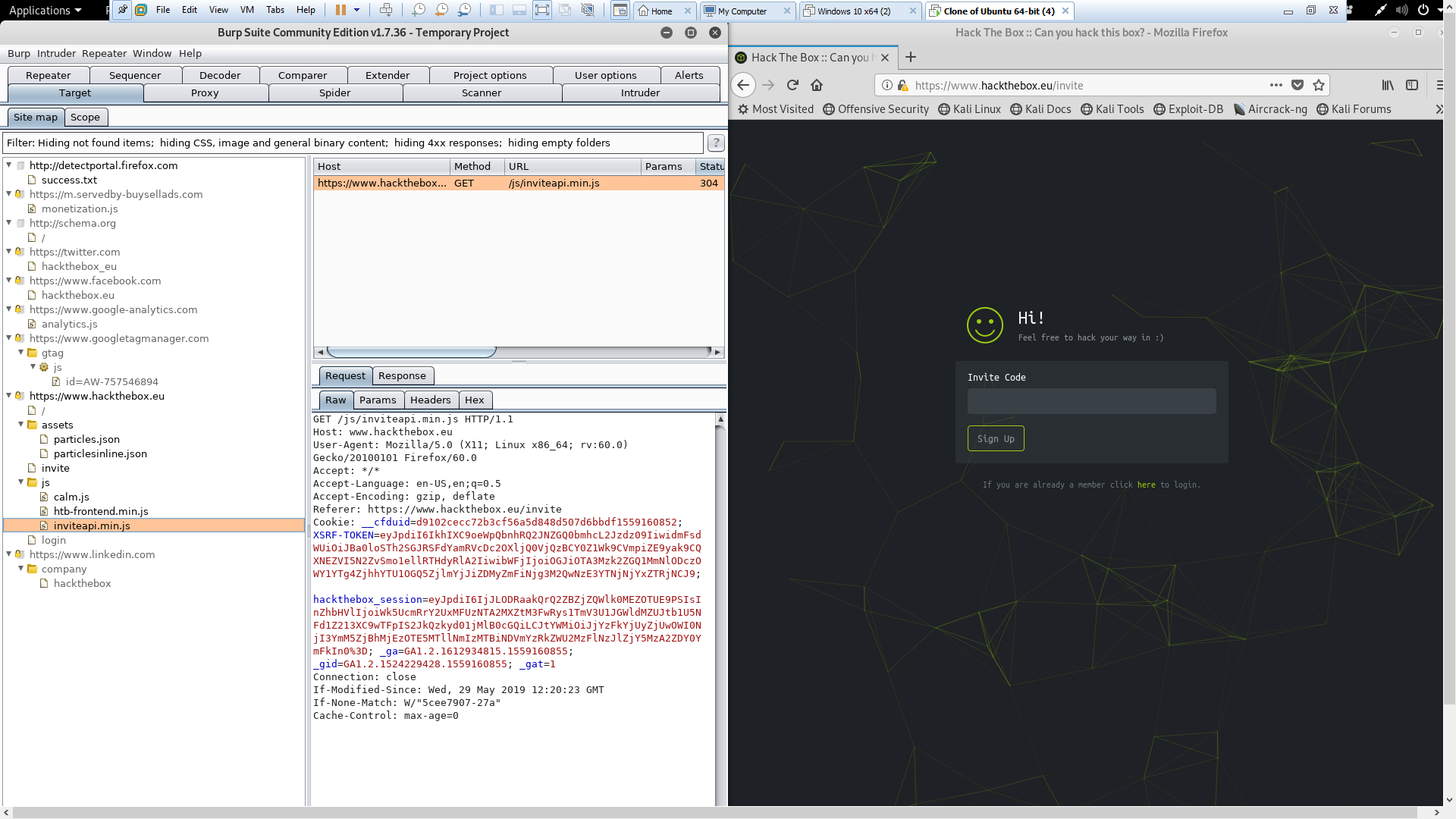
Project-04 Web attack

Software being used: Burp Suite Community Edition v1.7.36, WebGoat.

Procedure:

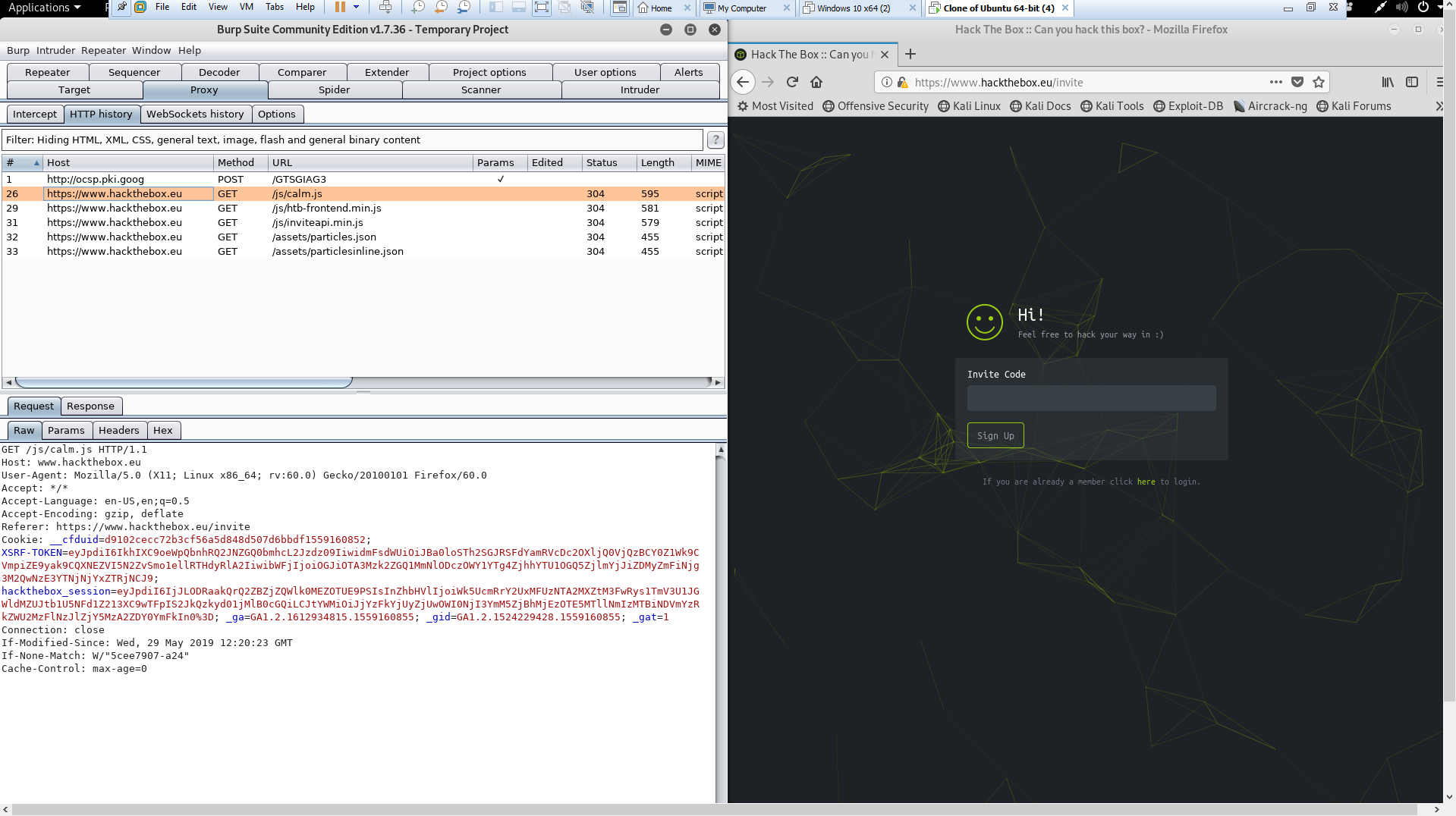


This is a viewing of the fact the burpsuite is properly setup. This portion of the lab was not as challenging as I have some prior experience with burpsuite. This also features the rule that was required for implementation via lab procedure.

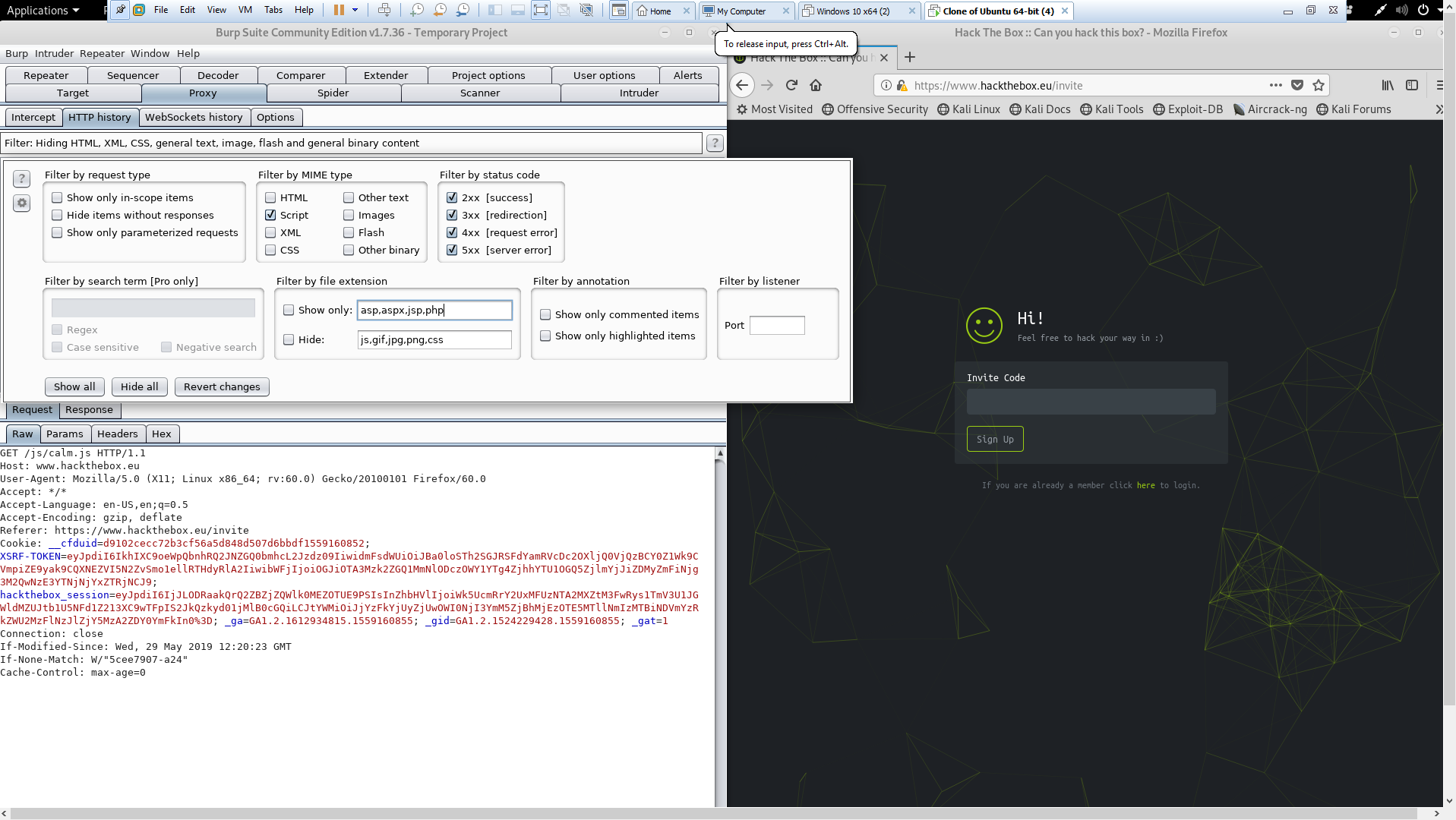


This is a view of the files associated with something we will be covering in later assignments, hackthebox invite section. This section has many areas of interest but will be covered more in depth for the final. Some interesting non-final related areas can be covered on this web page without fear of violating certain areas of concern.

After viewing some of the files associated with this website it is evident that there is a lot of intentional placement along with a few silly areas in these files and the html. You will see many instances of js or .js and hackthebox is primarily java script. This leads me to the next picture.



In this picture you can see I have a filter applied only to show java script related http history in the proxy section. The majority of information is only hackthebox as stated previously much of it is JavaScript.



Here is a further look at the filter applied in the proxy http history section.

Had much trouble getting burp suite and web goat working but later found that Firefox by default ignores proxy requests related to the local network. After removing this rule everything worked as it should.

The link used to find this answer

<https://support.portswigger.net/customer/portal/questions/17270165-burp-suite-is-unable-to-intercept-traffic-to-and-from-webgoat-localhost->

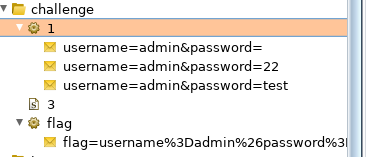
After venturing and realizing the ability you have with changing information being sent and received with vulnerable applications through burp suite the limits are endless. It is all up to the individual on whether they can exploit the system being used or not.

While messing around with web goat the primary tools used were proxy, target, intruder, and repeater. Intruder was used for payloads to attempt brute force on some of the logins.

Mission 1 “Admin Lost Password” NOTABLES

Password and username are sent to server and a response of true or false is sent back. After searching html an id was found under “AuthenticationFlaws-Passwordreset”. To base knowledge no console input is allowed. Further progress made after applying this knowledge.

Another observation made while learning how this works was that the web server logs login attempts and flag attempts as you can see in this screenshot.



Conclusion

Everything can be exploited if enough information can be gathered and an understanding can be built around the item. By using burp suite in the scenarios for this lab a real understanding of how powerful in the correct hands an application like this can be. This session and extensive time done on this project has led me to a new understanding of burp suite one I did not have prior. Applications don’t always work perfectly and there is always going to be that small sliver of information or mess up that causes more avenues to open.

Commands used

wget -nd [https://github.com/WebGoat/WebGoat/releases/download/v8.0.0.M15/webgoat-server-8.0.0.M15.jar (Links to an external site.)Links to an external site.](https://github.com/WebGoat/WebGoat/releases/download/v8.0.0.M15/webgoat-server-8.0.0.M15.jar)

18:32-5/30/19

java --add-modules java.xml.bind -jar webgoat-server-8.0.0.M15.jar --server.port=9090 --server.address=127.0.0.1

18:32-5/30/19

Proxy used 127.0.0.1:9090 instead of :8080 as burpsuite is on 8080.

15:23-5/31/19