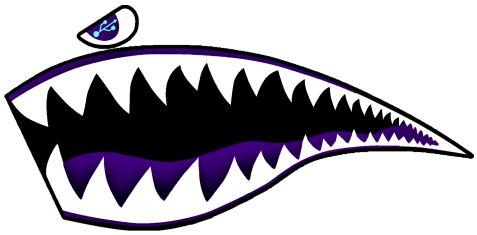
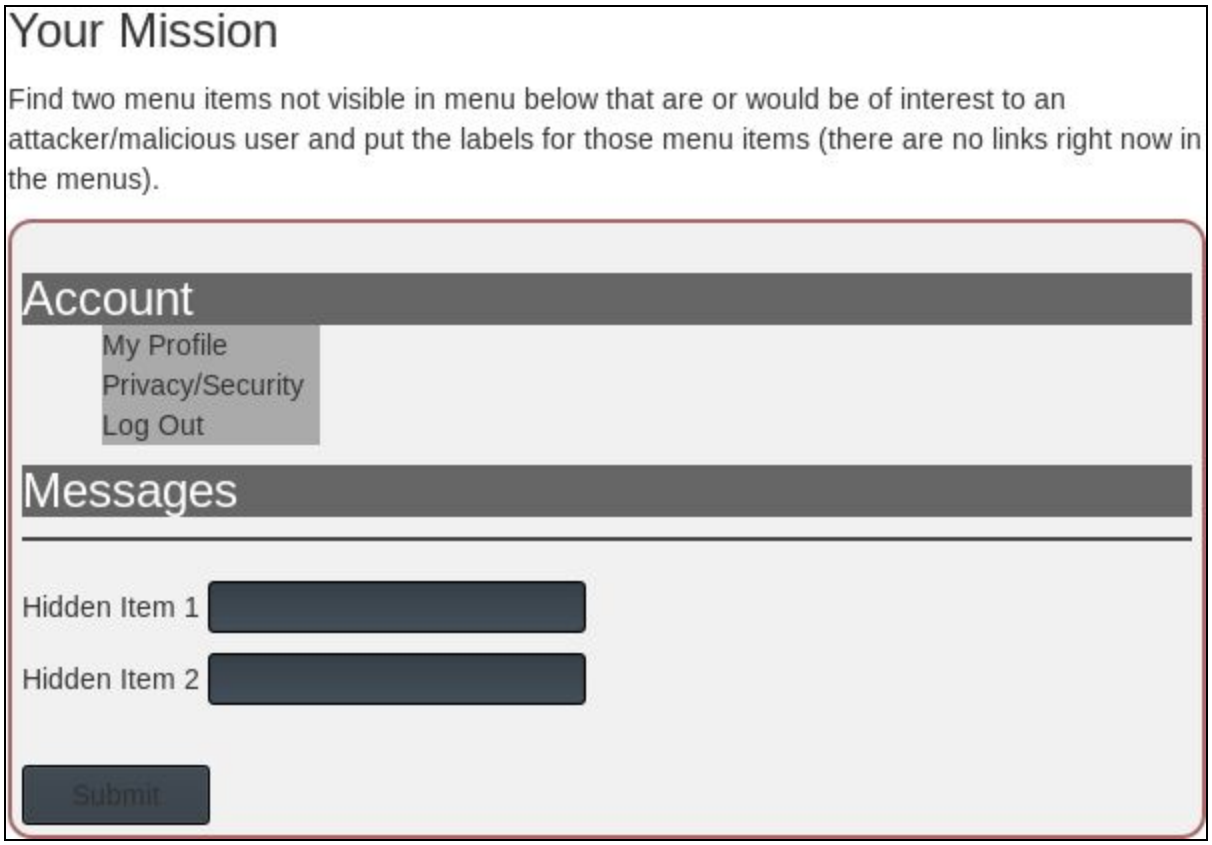


WebGoat Missing Function Level Access Controls

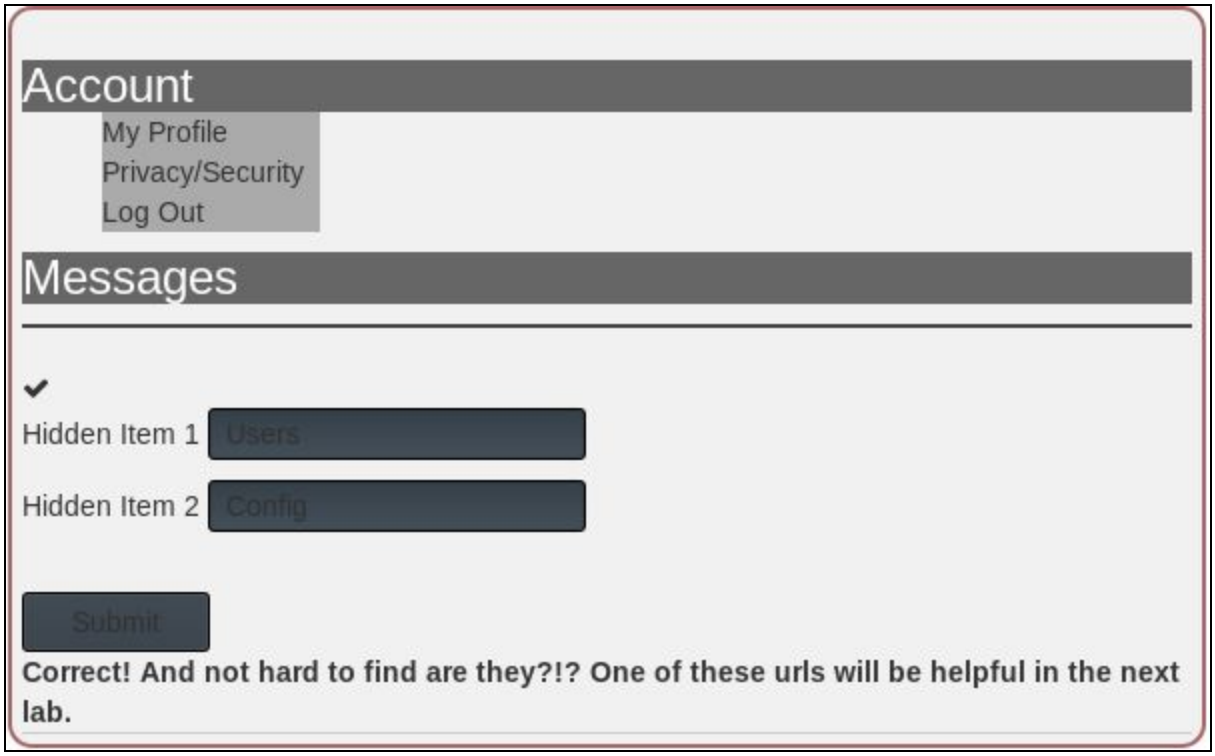
Twitter: @BlackSheepSpicy
Twitch: https://twitch.tv/BlackSheepSpicy

2. Oh good, more digging through code, thanks OWASP!

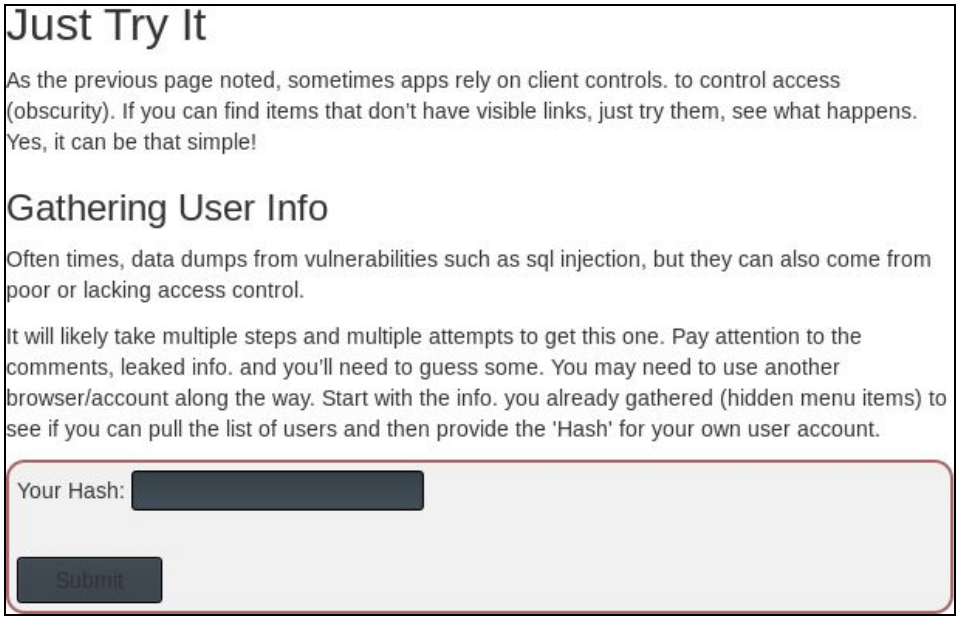


I did my best to show where these hidden items are, unfortunately it just comes down to digging through code, How I found it was I hit inspect element on the account menu and worked around there:

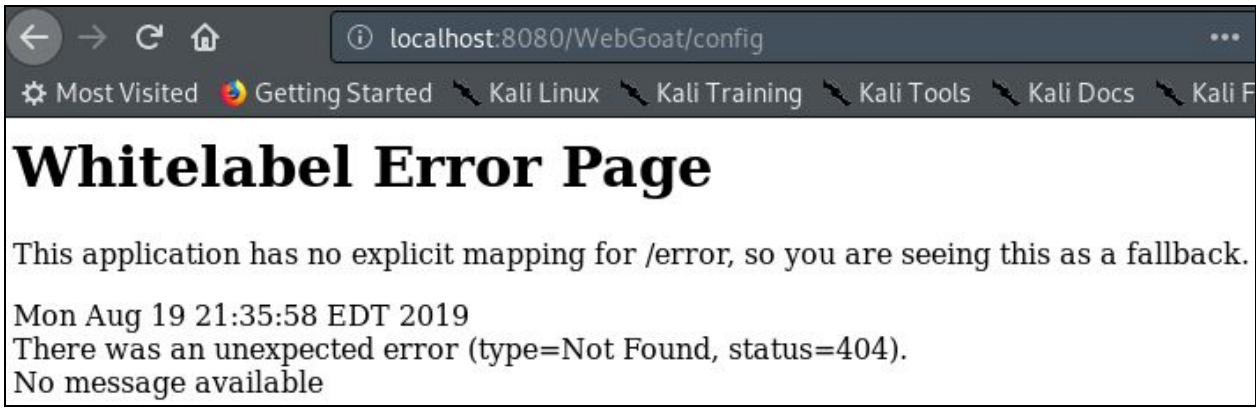
Big thing to keep in mind with this challenge, they're looking for the **label**, not the actual endpoint:



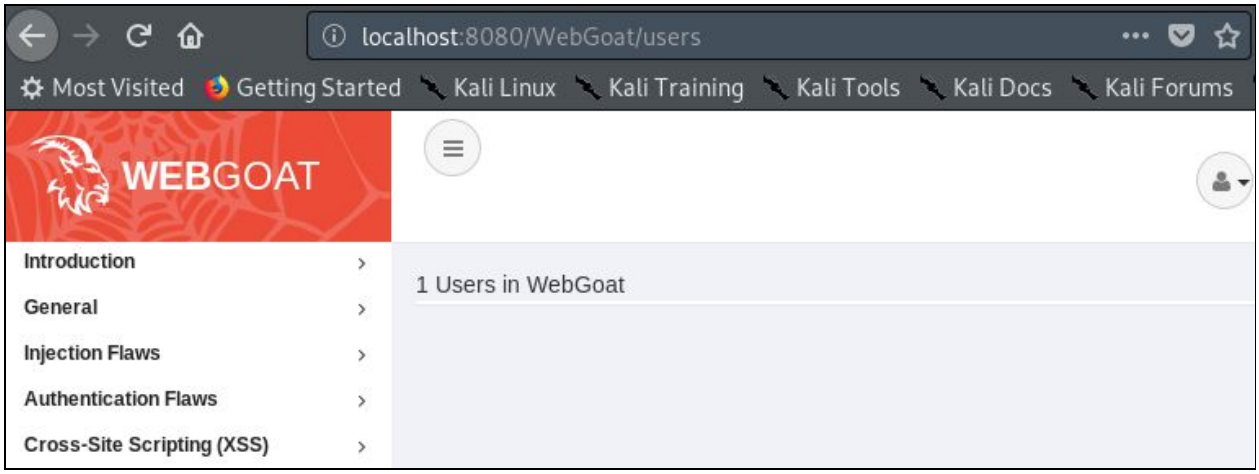
3. Building off the last exercise:



So remember those endpoints we discovered in the last challenge? They come into play here. Let's take a look at that those endpoints:

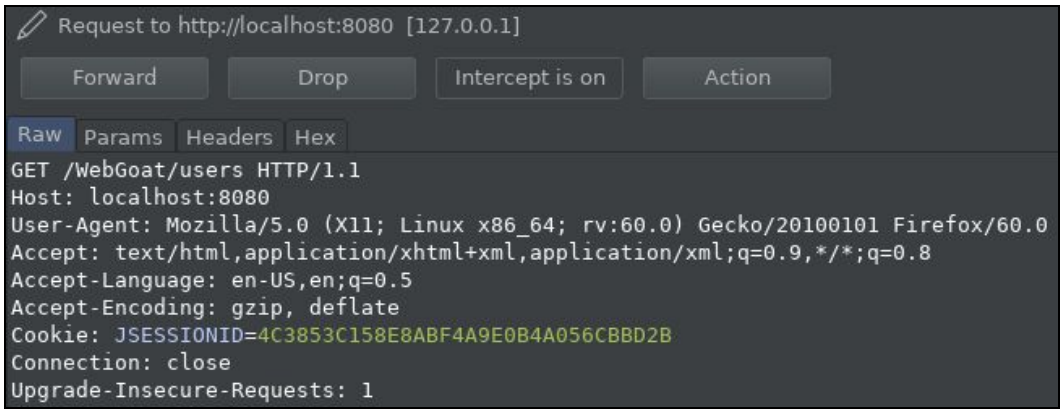


K, what's the other one?

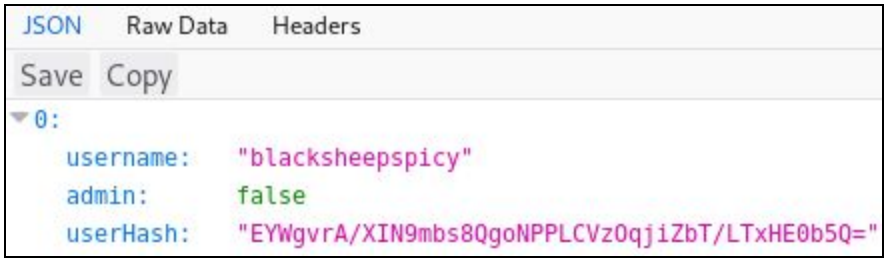


Hey rad its actually a valid endpoint, we can play around with this a bit but honestly I fully permit reading the hints on this one because this one is **weird.**

Start by intercepting the get request to the users endpoint:



Now get this shit, I have absolutely no idea why this works… at all… but if you set a **content-type** header to **application/json** and suddenly it returns the user hash OWASP is looking for:



Seriously why does this work?

Hope you enjoyed this writeup and if you want to see me attempt these challenges live be sure to drop by my Twitch when I'm live and also follow my Twitter for some quality shitposting!