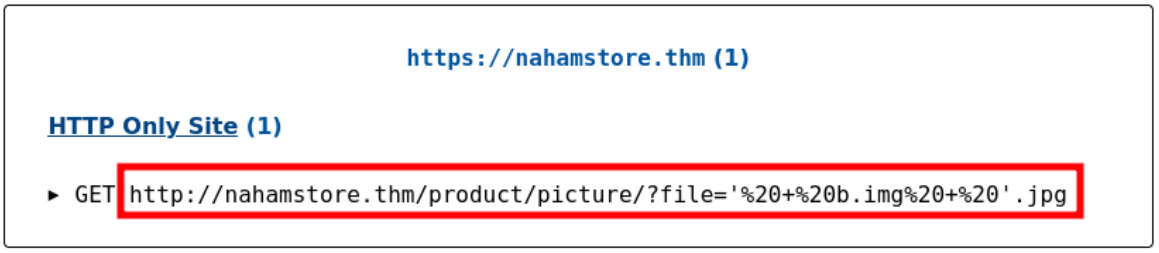
🏪🗃️ NahamStore: LFI

Introduction

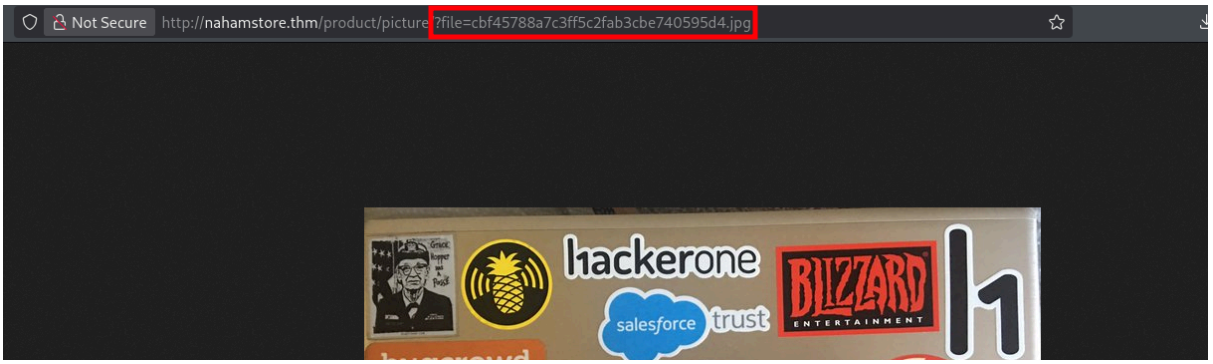
Local file inclusion (**LFI**) is the process of including files that are already locally present on the server, through the exploitation of vulnerable inclusion procedures implemented in the application. This vulnerability occurs, for example, when a page receives, as input, the path to the file that has to be included and this input is not properly sanitized, allowing ***directory traversal characters (such as dot-dot-slash [../../../])*** to be injected.

🕵️🗃️ Looking for LFI

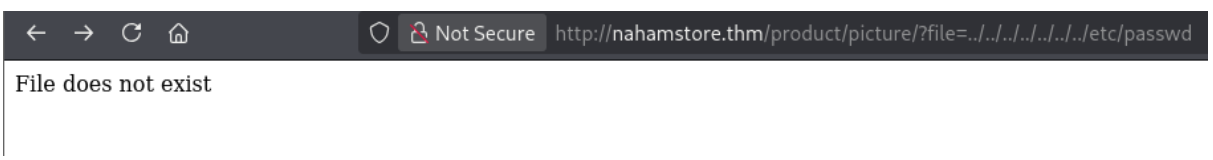
To find a potential LFI point, we need to find a section of the website that points to files on the server. In our case, we already have this information, since during the vulnerability analysis with Zap Proxy, a section of the website was found that points to files within the server:



The way to find this point manually is by going to any of the products on the website, either the *“Sticker Pack”* or the *“Hoodie + Tee”*, right-clicking on the product image and selecting *"Open Image in New Tab"*.



The interesting thing here is that we've found a section that points to files within the server, and this ***"/?file="*** is the one we'll be testing. To do this, we will use a simple LFI technique to try to read files within the server using path traversal (../../../..)

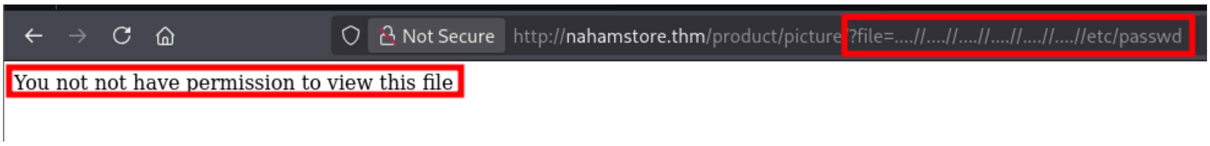


However, we see that this didn't work. What we can do from here is try to bypass it and test other *directory traversal* methods. This could take a long time, so we're going to automate this process using a tool, such as ***Burp Suite's Intruder*** or ***Caido's Automate***. But for this example, I'll use **ffuf** from the terminal. The dictionary I'll use is **LFI-Jhaddix.txt** from **SecList**, and the command will be this:

ffuf -u 'http://nahamstore.thm/product/picture/?file=FUZZ' -w /usr/share/wordlists/seclists/Fuzzing/LFI/LFI-Jhaddix.txt -fs 19 (The latter, fs 19, is used to filter out responses whose size is exactly 19 bytes, which are the "garbage" response**).**

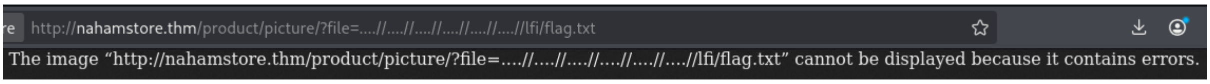
And this was the result:

And we have managed to find several directory traversal sequences that allow us to exploit LFI! Now we just need to place one of these in the ***"/?file="*** parameter and try to list files from the server.

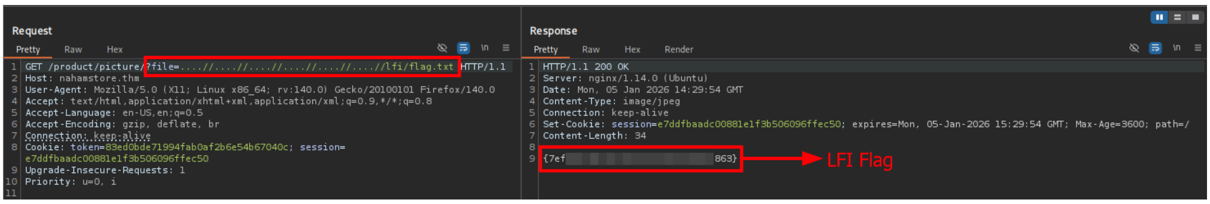


As we can see, although we cannot read the contents of the /etc/passwd file *because we do not have the privileges to do so*, we are still **interacting with the server's internal files**.

The definitive proof that this is so is to try to list the file located in the **/lfi/flag.txt** directory, which is confirmation of that challenge for the skeptics.



We see that when we try to list the **flag.txt** file, we get an error from the browser. So we're going to intercept this request from **Burp Suite** and send it to the ***Repeater***.



And with this we conclude the **LFI** section.