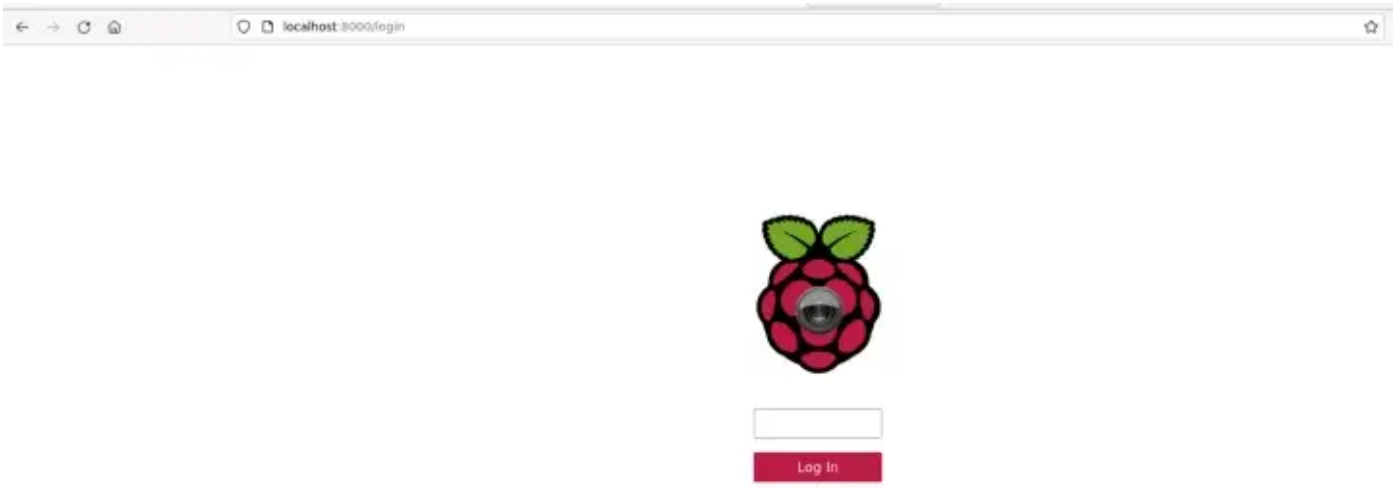
Elias Hohl Follow

Jul 25 · 4 min read · Listen

Save

Authentication Bypass vulnerability in camp, a Raspberry Pi camera server

I recently had a look at `camp`, a Raspberry Pi camera server based on Python with more than 100 stars on Github. It uses the Tornado framework and supports username-password authentication. When we start the application via `python3 server.py --require-login`, a browser window will be opened showing a login screen.



Let's see if we can bypass this. Before[...]er, I need to show you a small modification I made to the original application. I removed the real camera part and the

Just execute


using above file to disable the unnecessary parts of the application.

Looking at the part of the source code where the Tornado handlers are defined

we can see that the whole root folder is served via `StaticFileHandler` . However, `password.txt` seems to be blocked first via `ErrorHandler` . According to the Tornado documentation, rules are parsed on a “first-match-wins” basis. So there should be no problem, `ErrorHandler` should trigger before the password file can get served. Let’s try with any other file from the root folder first, for example `server.py` :

We can acutally download the python script. This is nice, but as there is no secret information in there, it also does not really help. Trying to open `http://localhost:8000/static/password.txt` will lead to a 403 Forbidden error. Well, let's test with some path traversal magic and encoding:

Surprisingly, all three of these techniques work and serve us the correct SHA-512

documentation. There should be no way to bypass such rules. One could argue that this only works when the secret file is within the served directory, but a block via `ErrorHandler` should still work. In other server applications like Apache, Nginx or Flask, such rules also work as expected. Especially in Apache and Nginx, it is quite common to have secret files within served directories and have them blocked with `.htaccess` rules or in the server configuration files. I contacted the Tornado maintainer regarding this, but he does not consider this a security bug, as the access rules are apparently “not intended to be secure”, although this is indicated nowhere in the documentation. He seems to be more interested in allowing users to serve files containing special characters like `%` in the filename, access files via percent-encoded characters or `./` magic, “because someone might be doing this intentionally”. Long story short, just avoid this exotic framework if you can.

As we now have the hash, we could crack it with a tool like `hashcat` or rainbow tables if the password is simple:


There is a very suspicious line of code at the bottom of the `server.py` file:

`PASSWORD` is defined at the top of the file:


So `PASSWORD` is not the password, but the hash taken from the `password.txt` , the hash we retrieved above. So by knowing the hash, we also know the cookie secret. We can see how the authentication mechanism works by looking at the `LoginHandler` class:

The password submitted to the endpoint is hashed and the result is compared with the hash stored in `password.txt`. If the hashes match, an authentication cookie is generated with `set_secure_cookie`. As we know the cookie secret, we can, however, skip the first step and generate our own cookie. Digging a little into the Tornado source code, I figured out that the a valid cookie can be created using the following code snippet:

Then, we set this value (the string inside the single quotes) manually as the value of a cookie named `camp` in the browser and navigate to `http://localhost:8000/` :

The maintainer replied instantly after I reported this issue and fixed it in commit bf6af5c2e5cf713e4050c11c52dd4c55e89880b1.

The vulnerability has been assigned CVE-2022–37109.

The Github repository belonging to this post:

GitHub - ehtec/camp-exploit: Authentication Bypass for https://github.com/patrickfuller/camp

You can't perform that action at this time. You signed in with another tab or window. You signed out in another tab or...

github.com

If you want to read about more vulnerabilities I discover, make sure you follow me on LinkedIn, Medium & Twitter:

Elias Hohl | Linktree

Advisor | Cybersecurity Expert | Developer | Physicist

linktr.ee

If you run a company and are looking for an expert to make sure your web applications are secure, feel free to send an email to elias.hohl@ehtec.co to receive an offer.