Congrats on passing the first step in the Hyver interviewing processes. As we discussed, the second phase in this process is the programing exrecise this document covers. The purpose of this exrecise is to assess you programing, design, and learning abilities.

The exrecise is composed of three steps. In addition to these steps, there are some extra credit steps you can choose to implement if you want to realy show of your skills.

We are also providing a simple server that should be used to test your result. In order to run it, simple run python server.py (don’t forget to pip install -r requierments.txt first ☺ )

**Step 1:**

1. Create a library called: "specific\_ip\_fetcher".
2. The library should expose the following function:
3. fetch\_binding(bindings)
4. where bindings is an array of dictionaries in the following format:

{

"url" : "<a url of the site to fetch>",

"target\_ip" : "<optional> the exact ip to use"

}

1. The function should fetch the HTML content of the given url. If a target\_ip was provided - it should fetch the content of the provided URL using the target\_ip.
2. Run server.py (Provided)
3. Run the function with the following input:

[

{"url" : "https://hyver-security.com"},

{"url" : "https://hyver-security.com", "target\_ip" : "127.0.0.1"}

]

**step 2:**

1. Create a simple flask server that will expose the following url:

/api/bindings

1. The url should receive a JSON of bindings, and return the results of the fetch\_binding method in JSON format.
2. The API should be authenticated

step 3:

1. Create a a simple shell client that will be used to invoke the above API and save the results in json file.

**Extra Credit:**

1. Complete step 1 without modifying the host file.
2. Run step 1 with the following input in under 30 seconds:

[

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

{"url" : "https://hyver-security.com?r={}".format(random.randint(1, 100)), "target\_ip" : "127.0.0.1"},

]

1. Use client side certificate authentication for the API.
2. Add a Dockerfile that will allow running the ex in Docker.

Good luck.

The Hyver dev team.