Ethical Student Hackers

Automation in Cybersecurity

The Legal Bit

- The skills taught in these sessions allow identification and exploitation of security vulnerabilities in systems. We strive to give you a place to practice legally, and can point you to other places to practice. These skills should not be used on systems where you do not have explicit permission from the owner of the system. It is VERY easy to end up in breach of relevant laws, and we can accept no responsibility for anything you do with the skills learnt here.

- If we have reason to believe that you are utilising these skills against systems where you are not authorised you will be banned from our events, and if necessary the relevant authorities will be alerted.

- Remember, if you have any doubts as to if something is legal or authorised, just don't do it until you are able to confirm you are allowed to.

Code of Conduct

- Before proceeding past this point you must read and agree to our Code of Conduct - this is a requirement from the University for us to operate as a society.
- If you have any doubts or need anything clarified, please ask a member of the committee.
- Breaching the Code of Conduct = immediate ejection and further consequences.
- Code of Conduct can be found at https://shefesh.com/downloads/SESH%20Code%20of%20Conduct.pdf

Why Automate?

- Makes repetitive tasks faster
  - Build processes in development
  - File navigation and manipulation
- Lets you do these tasks with new parameters
  - Pass IPs to scanners
  - Backup a directory
- Can create scripts for background tasks
  - Reconnaissance
  - Password Cracking
  - Web Scraping
- Learning automation useful for understanding exploits

Bash Scripting

What is Bash?

- A Unix Shell and Command Language
- Default shell in many Linux distributions

What can it do?

- Make use of stdin, stdout, stderror
- Run commands such as cat, echo, cd
- Run other scripts
- Redirect output to files
- Chain commands with &&, ||, and ;
- Anything your CLI can do!
- Plus loops, conditional statements, and input capture with read [variable]

Creating a Bash Script

- Create a file: script.sh
- Add a shebang #!/bin/bash
- Write some commands
  - Use a text editor
  - Or echo “id” > script.sh
- Make the file executable:
  - chmod +x script.sh
  - chmod 711 script.sh
- Run the script:
  - ./script.sh

Bash Scripting

Simple logical flow:

```
read name

if [ $name != “admin” ]

then

    echo “Get outta here”

elif [ -e “./checkfile” ]

then

    echo “Welcome admin!”

fi
```

Simple loops:

```
count=1

while [ $count -lt 10 ]

do

    echo $count

    count=$[$count + 1]

done

for ip in $(seq 1 10); do ping
-c 1 10.11.1.$ip; done
```

Bash Scripting - Mini Exercise

Time for a memory test!

- Create an .sh file with the right shebang
- Add some commands to do the following:
  - Output “Script running” to the screen
  - Create a new directory called “script_output”
  - Write the output of the “id” command to a file called “id” in the directory
  - Write the contents of the “/etc/passwd” file to a file called “enum” in the directory
  - Append the contents of “/etc/group” to the same file
  - Output “Done” to the screen
- Give the script the right permissions, and run it!

We’ll give you 5-10 minutes then go over the solution

Bash Scripting - Parameters

What are parameters?

- Extra Data passed to a script or binary
- Can be positional, or use flags

Positional:

- $x
- ${variable}

Flags:

- Use getopts or $@
- https://stackoverflow.com/questions/14447406/bash-shell-script-check-for-a-flag-and-grab-its-value
- https://github.com/Twigonometry/sesh-automation/blob/master/params-and-flags

Text Processing & Manipulation

Bash has a range of powerful tools for text processing and manipulation

- grep for matching regexes
- awk for searching structured data
- sed for finding and replacing

Example commands (feel free to try them):

- cat /etc/passwd | grep bash
- echo “username:password” | awk -F ‘:’ ‘{print $2}’
- echo “

Can be combined!

Cool Example - Port Scanner!

Quick intro - What is a port scanner?

What are the basic steps?

- Get a list of IPs and ports from the command line
- Use a loop to ping each one to see if it's live (not 100% reliable)
- If it's live, loop and scan the given ports (using /dev/tcp)
- Test it: Kenobi THM - https://tryhackme.com/room/kenobi

For the methodology: https://catonmat.net/tcp-port-scanner-in-bash

For a more sophisticated example: https://github.com/astryzia/BashScan

EGM

Vote here:

Nominees:

- Rong Rong (General Member)

Python Introduction

First things first to use Python for session today you should have it installed on your computer.

To Install Python, you can download it from https://www.python.org/downloads/

And explicitly for linux users, we have created a bash file that will instantly install the resources you need for today, including python, pip, selenium and chrome (if you select yes, firefox will also work).

You can download and run the bash file via the following command:

wget https://shefesh.com/session_scripts/SeleniumLinux.bash; bash SeleniumLinux.bash

Python Introduction

Python is a useful programming language which is easy to learn and has a lot of functionality

In python there are a variety of modules which have a variety of uses:

- The os module (Useful for getting operating system information and executing commands)#
- The requests module is useful for sending HTTP requests via the methods such as requests.get and requests.post, you can also add parameters to these methods and also save the response.
  - response = requests.get('https://www.shefesh.com/')
  - requests.post('https://httpbin.org/post',data={'key':'value})
- There are api libraries for use of web services such as googlemap, ospotipy , pyfacebook and more.
  - Even if there isn't a library you can use http requests to make api calls via use of the requests module.
- To install a new library you can use pip
  - pip install <ModuleNameHere>
- Python programs can also take command line arguments when executed

Python Introduction

Python is an interpreted, interactive and object-oriented programming language which also supports dynamic typing and a variety of other features.

To start off for now we'll go over basic command and instructions, so if you could all open a new .py file in your text editors and we'll start off with these exercises to help you all become familiar with the environment.

To print a string output in python all you need is the print() command, for example

- print("Hello world")

Unlike other languages such as Java, C#, and C++ you don't have to specify a main starting method as the entire python script will be loaded and ran sequentially from start to finish, though of course for example if you make a method and do not use it in the instructions it will not be ran.

Python Introduction - If Statements

Bash if statements are fairly similar to that of python, here is a conversion of the example from earlier.

Bash

```
read name

if [ $name != “admin” ]

then

    echo “Get outta here”

elif [ -e “./checkfile” ]

then

    echo “Welcome admin!”

fi
```

Python

```
import os

name = str(input())

if name != “admin”:

    print(“Get outta here”)

elif os.path.exists("file.txt"):

    print(“Welcome admin!”)
```

Python Introduction - Methods

Creating a method in python is fairly simple as well, Like if statements it is also indentation based in python. Methods can also contain parameters but types do not need to be specified as stated earlier python is a language which uses dynamic typing.

```
def HelloWorld():
    print("Hello World")

def AddNumbers(a,b):
    return a + b
```

```
HelloWorld() → Hello World

AddNumbers(1,2) --> 3
```

Python Introduction - Loops 1

Iteration in python is again very similar to that of bash, and like if statements it is again indentation based.

```
count=1

while [ $count -lt 10 ]

do

    echo $count

    count=$[$count + 1]

done
```

```
count=1

while (count < 10):

    print(count)

    count = count + 1
```

Python Introduction - Loops 2

As well as while loops there are also for loops in python, they work in a in a similar fashion to a foreach loop.

```
for i in range(0,10):
    print(i)
```

```
a = [0,1,2,3,4,5,6,7,8,9,10]
for i in range(0,len(a)):
    print(a[i])
```

```
for i in [0,1,2,3,4,5,6,7,8,9,10]:
    print(i)
```

Quick Python Practise

Now that you all have been some basic python:

Have a go at the following:

- Write a method to add all numbers in a list
- Fizzbuzz write a python program to print all the numbers 1-100
    - However if it is a multiple of 3 it will write “Fizz” instead
    - If a multiple of 5 it will write “buzz” instead
    - If a multiple of both it will write “Fizzbuzz” instead

If you need any help, please feel free to ask, this is just to teach the basics as we’ll go onto more advanced content soon.

FYI

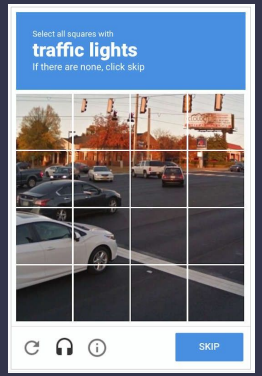
Introduction to Selenium

Selenium is a python module intended for Web Scraping (Create a bot to visit webpages and act as a user)

The module itself is very straightforward and most/all of you should've download the module and corresponding driver prior. (No problem if not, once we're in demonstration we'll be able to help if you have an issues regarding installation)

No GUI Linux Users will have to make some modifications

```
from selenium import webdriver

from selenium.webdriver.<LowerCase Browser Name Here>.options import Options

driver_options= Options()

driver_options.add_argument("--headless")

driver= webdriver.Firefox(executable_path="<Your DriverPath>",options=driver_options)

driver.get("<Webpage Address Here>")
```



Introduction to Selenium

Here is a very basic selenium example, designed to open a web browser to visit the ShefESH Website:

```
from selenium import webdriver

driver = webdriver.Chrome(executable_path="C:\\Programs\\ChromeDriver.exe")

driver.get("https://shefesh.com/")
```

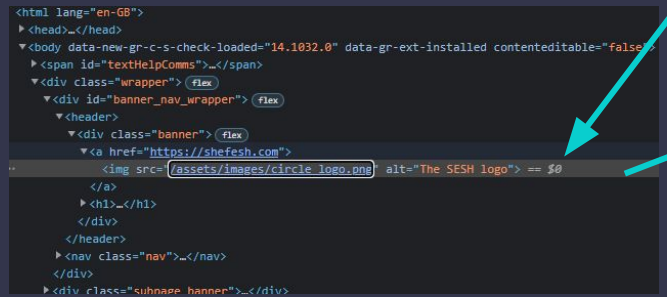
You can all give it a quick go now if you want, in a second we are going to go into more of the practical side behind selenium.

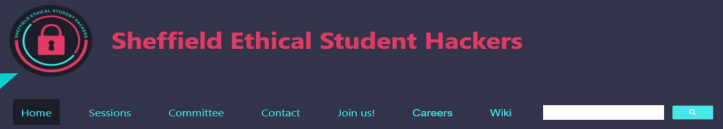
Introduction to Selenium - XPaths

XPath otherwise known as XML Path Language is a query based language used for selecting nodes in XML. XML and HTML are pretty similar but for all intents and purposes today all you need to know is that XML is used for serialization and data transmission whereas HTML is used presenting data.

An XPath in terms of web-scraping is a query that directly tells us the location of a HTML Element, for example on the ShefESH Homepage:

The XPath “/html/body/div[1]/div[1]/header/div/a/img” corresponds to the logo on our webpage.





Introduction to Selenium - XPaths

To return an element via an XPath you can use the method:

```
from selenium import webdriver

driver = webdriver.Chrome(executable_path="C:\\Programs\\ChromeDriver.exe")

driver.get("https://shefesh.com/")

shefESHLogo = driver.find_element_by_xpath ("/html/body/div[1]/div[1]/header/div/a/img")
```

Inside the shefESHLogo variable now is the HTMLElement, and now we since we can select elements on a webpage, we can start manipulating them. If we were to get the src attribute from the element, we can now see the relative url of where the image is stored on the website.

```
print(shefESHLogo.get_attribute("src"))
```

Introduction to Selenium - Elements

In Selenium you can do a variety of things with HTMLElements as well as getting their information:

Here are examples of a few commands that could be used on an element:

- `el1.click()`
  - Clicks the element as if it was a real user (Useful for forms)

- `el1.get_attribute("innerHTML")`
  - Returns all html inside the element, if a p element with text, just returns text
  - For more advanced users get_attribute will get any specified attribute of a HTMLElement and return it, most of the time as string.

- `driver.execute_script('alert()')`
  - Runs javascript on the webpage, you specify elements as parameters for some commands as well

Introduction to Selenium - Elements

More methods that could be used on HTMLElements

- `el1.find_element_by_id("")`
  - Finds element by html ID, In this case since it will only return the first element which is a child of "el1"
- `el1.find_element_by_css_selector()`
  - Finds element by css selector, In this case since it will only return the first element which is a child of "el1"
- `el1.find_elements_by_css_selector()`
  - Finds all elements by css selector, There are other methods like this that also work on the driver (Returns result in a list of HTMLElements)
- `el1.sendKeys("MyPassword")`
  - Sends key presses for text into the element, can work for brute forcing password and username logins

Introduction to Selenium - Example

Here is an example selenium program designed to get all of and only the committee photos for our website.

```
from selenium import webdriver

driver = webdriver.Chrome(executable_path="C:\\Programs\\ChromeDriver.exe")

driver.get("https://shefesh.com/")

profiles = driver.find_elements_by_css_selector( ".committee-profile")

for p in profiles:

      img = p.find_element_by_xpath ("img")

      print(img.get_attribute("src"))
```

Introduction to Selenium - Example

This final example searches through our webpage for the word “hacking”

```
from selenium import webdriver

driver = webdriver.Chrome(executable_path=r"C:\Users\Student\Chrome\chromedriver.exe")

driver.get("https://shefesh.com")

searchBox =
driver.find_element_by_xpath("/html/body/div[1]/div[1]/nav/ul/li[9]/div/div/div/form/table/tbody/tr/td[1]/div/table/tbody/tr/td[1]/inp
ut")

print(searchBox.get_attribute("outerHTML"))

driver.execute_script("arguments[0].setAttribute(arguments[1], arguments[2]);",searchBox,"value","hacking")

button = driver.find_element_by_xpath("/html/body/div/div[1]/nav/ul/li[9]/div/div/div/form/table/tbody/tr/td[2]/button")

button.click()
```

Introduction to Selenium

Exercises:

- Get a list of all our previous session titles on the “https://shefesh.com/sessions” page
- Make a script to try and visit all our pages, via checking href

If you struggle with any of the following exercises, please feel free to ask for another committee member or myself to try and help, and enjoy.

We won’t advise the use of web-scraping other websites atm for ethical and legal reasons, but for this session you can freely web-scrape our website.

Upcoming Sessions

What's up next?

www.shefesh.com/sessions

Next week

The week after that

And the week after that

And the week after that

And the week after that

Any Questions?

www.shefesh.com

Thanks for coming!