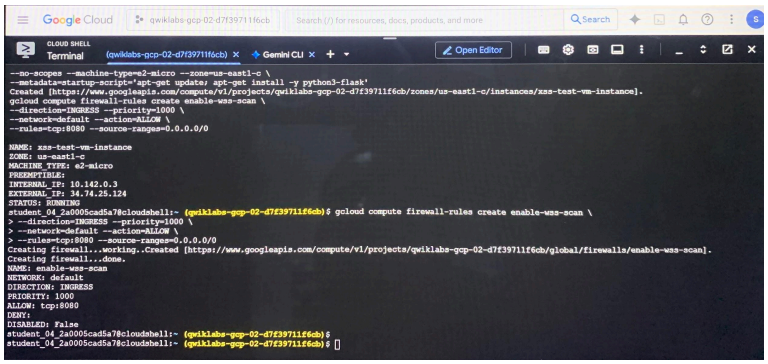
IDENTIFY VULNERABILITIES AND REMEDIATION TECHNIQUES

Cymbal Bank has developed a new banking application for its corporate clients that is set to be hosted and deployed on the new cloud infrastructure. The Chief Information Security Officer (CISO), Javier, wants to prioritize the security of this application before it is launched and customer-facing. My team lead, Chloe, has tasked me with identifying and mitigating any application vulnerabilities for this new application. I'll use the Web Security Scanner in Google Cloud to scan the application for vulnerabilities pertaining to a top OWASP® web application vulnerability known as Cross-Site Scripting (XSS).

Here's how I did this task: **First**, I created a static IP address and launched a virtual machine. **Then**, I deployed the vulnerable application. **Next**, I set up and ran the application. **Then**, I accessed and scanned the application. **Finally**, I fixed the vulnerabilities and re-scanned the application.

I activated the cloud shell terminal and typed in the following command;



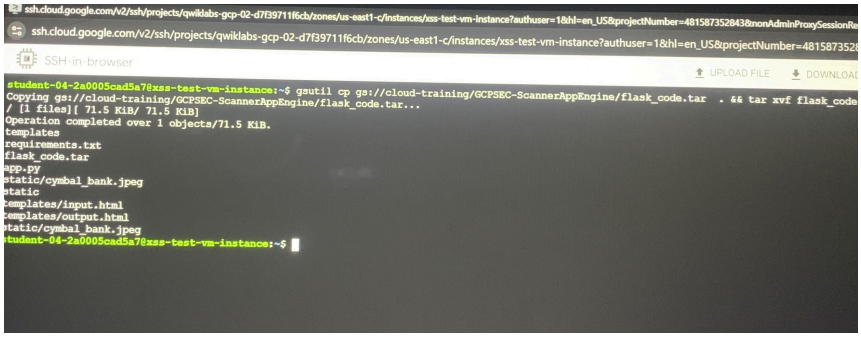
This command creates ;

1. A static IP address named `xss-test-ip-address` in the **`REGION`** region. This static IP will be used for scanning the vulnerable web application.

2. This command returns the static IP address generated.

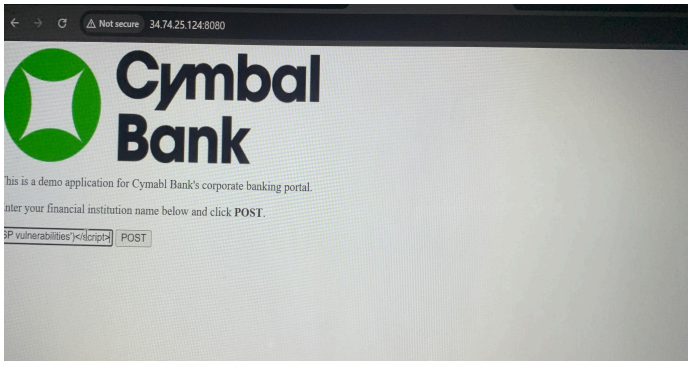
3. This command creates a VM instance to run the vulnerable application.

I typed the following in the SSH–in–browser;

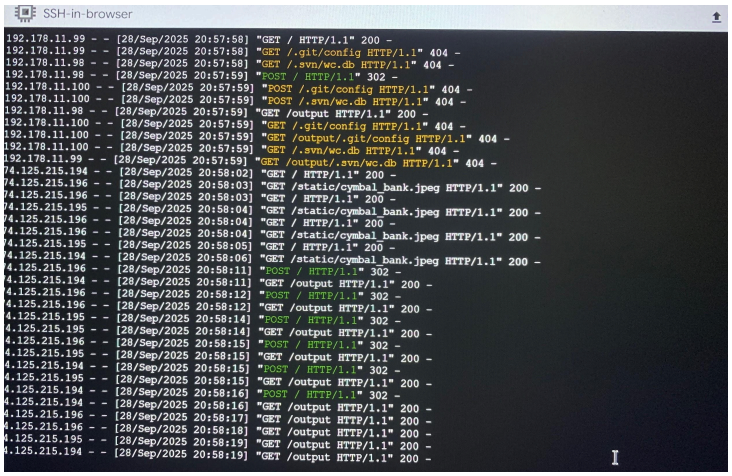


This command downloads and extracts the vulnerable web application files.

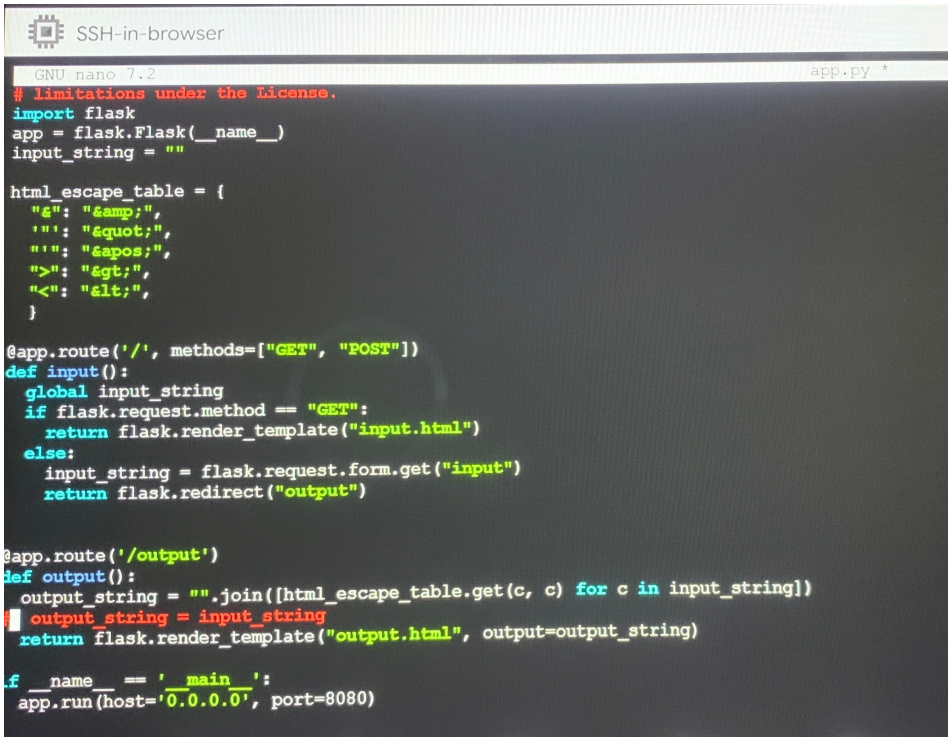
I tested my application for a vulnerability known as cross-site scripting (XSS). The injected code displayed a message back to the browser. This action by itself is not malicious, however attackers can introduce malicious code into an exploitable application to either steal data from it or implant malware onto the user's device. The alert window opens with the following message: “This is an XSS Injection to demonstrate one of OWASP vulnerabilities”.



Upon completion of the scan in the **SSH-in-browser** window, I viewed logs being generated as Web Security Scanner tests all possible URLs for potential vulnerabilities.



To fix the XSS vulnerability, I validated the output string variable in the nano code editor. The output string is the processed output of the user-supplied web form input.



Then I re-scanned to ensure no vulnerabilities were left.