Identify vulnerabilities and remediation techniques

LAUNCH A VIRTUAL MACHINE

In this task, you'll create a static IP address and launch the virtual machine to run the vulnerable application.

1. On the Google Cloud console title bar, click the **Activate Cloud Shell** ( ) icon. If prompted, click **Continue**.
2. Copy the following command into the Cloud Shell terminal:

```
gcloud compute addresses create xss-test-ip-address --region=us-east1
```

Copied!
content_copy
This command creates a static IP address named `xss-test-ip-address` in the **`us-east1`** region. This static IP will be used for scanning the vulnerable web application.

3. Press **ENTER**.

If prompted, click **Authorize**.

4. Copy the following command into the Cloud Shell terminal:

```
gcloud compute addresses describe xss-test-ip-address \
--region=us-east1 --format="value(address)"
```

Copied!
content_copy
This command returns the static IP address you generated.

5. Press **ENTER**.
6. Copy the **IP address** from the output and save it in a notepad. You'll need to use this in a later task.
7. Copy the following command into the Cloud Shell terminal:

```
gcloud compute instances create xss-test-vm-instance --address=xss-
test-ip-address --no-service-account \
--no-scopes --machine-type=e2-micro --zone=us-east1-d \
--metadata=startup-script='apt-get update; apt-get install -y python3-
flask'
```

Copied!
content_copy
This command creates a VM instance to run the vulnerable application.

8. Press **ENTER**.

SET UP AND RUN THE VULNERABLE APPLICATION

In this task, you'll download and extract the web application files for the vulnerable application, and then deploy the application in the SSH-in-browser.

First, you'll create a firewall rule that will allow Web Security Scanner to access the vulnerable application.

1. Copy the following command into the Cloud Shell terminal:

```
gcloud compute firewall-rules create enable-wss-scan \
--direction=INGRESS --priority=1000 \
--network=default --action=ALLOW \
--rules=tcp:8080 --source-ranges=0.0.0.0/0
```

Copied!
content_copy
This command creates a firewall rule that allows access to the web application from any source IP address. This allows the Web Security Scanner to access the vulnerable application and perform a scan.

2. Press **ENTER**.

Next, use an SSH connection to connect to the VM instance.

3. In the Google Cloud console, click the **Navigation Menu** (☰).
4. Select **Compute Engine** > **VM instances**.
5. On the **VM instances** page, in the **Connect** column, click on the **SSH** button next to your test instance.

This will open an SSH connection to your VM instance in a new browser window.

6. A pop-up may appear asking you to allow SSH in-browser to connect to VMs. Click **Authorize**.

Now, extract the web application files.

7. Copy the following command into the **SSH-in-browser** page (not in Cloud Shell):

```
gsutil cp gs://cloud-training/GCPSEC-ScannerAppEngine/flask_code.tar  .
&& tar xvf flask_code.tar
```

Copied!
content_copy
This command downloads and extracts the vulnerable web application files.

8. Press **Enter**.

9. **Finally**, copy the following command into the **SSH-in-browser** page:

```
python3 app.py
```

Copied!
content_copy
This command starts the application.

10. Press **ENTER**.

ACCESS THE VULNERABLE APPLICATION

1. While the application is running, open a new browser window.
2. Copy the URL below into the browser tab, and replace `<YOUR_EXTERNAL_IP>` with the static IP address of the VM you saved in a notepad in Task 1:

```
http://<YOUR_EXTERNAL_IP>:8080
```

Copied!
content_copy
A Cymbal Bank corporate banking portal with a web form should appear.

3. Copy the following HTML code including the script tags into the web form:

```
<script>alert('This is an XSS Injection to demonstrate one of OWASP vulnerabilities')</script>
```

4. Click **POST**.

SCAN THE APPLICATION

First, enable the Web Security Scanner API.

1. In the Google Cloud console, click the **Navigation menu** (☰).
2. Select **APIs & Services** > **Enabled APIs and services**. The **APIs & Services** page displays.
3. Click **+ Enable APIs and services**.
4. In the search field, type **Web Security Scanner**, and press **ENTER**.
5. Select **Web Security Scanner API**.
6. Click **Enable**.

Now, scan the application for vulnerabilities.

7. In the Google Cloud console, click the **Navigation menu > View All Products** (☰).
8. Select **Security** > **Web Security Scanner**.

If the **Web Security Scanner API** is enabled then the Cloud Web Security Scanner page displays the **Scan configs** details.

9. In the **Cloud Web Security Scanner** toolbar, click **+ New scan**.

10. In the **Name** section, name the scan **Cross-Site Scripting scan**.

In the **Starting URLs** section, the **Starting URLs 1** field should be pre-populated with your static IP address.

11. Add a colon and the port number **8080** at the end of the IP address. The **Starting URL 1** should resemble the following:

```
http://<EXTERNAL_IP>:8080
```

Copied!
content_copy

12. If present, delete **Starting URL 2**.
13. In the **Excluded URLs** section, verify that **Authentication** is set to **None**, and **Schedule** set to **Never**. Leave all other fields unchanged.
14. Click **Save** to create the scan.
15. Click **Run Scan** to start the scan.
16. Return to the **SSH-in-browser** window.

In the **SSH-in-browser** window, you should view logs being generated as Web Security Scanner tests all possible URLs for potential vulnerabilities.

17. When the scan is complete, return to the Google Cloud console.

REMIDATE THE VULNERABLITIES

1. Return to the **SSH-in-browser** page connected to your VM instance.
2. Press **CTRL + C** to stop the running application. Alternatively, you can click the **Send key combination** icon on the top right corner of the SSH-in-browser window to input the **CTRL + C** key combination.

Now, edit the **app.py** file using the nano editor.

3. Copy the following command into the **SSH-in-browser** page:

```
nano app.py
```

Copied!
content_copy
This command opens the nano code editor.

4. Press **ENTER**.

5. To fix the XSS vulnerability, you validate the output string variable. The output string is the processed output of the user-supplied web form input.

Ensure that the application does not accept user input as HTML code, instead, it will escape special characters supplied by user input. To do this, locate the two lines that set the output string:

```
# output_string = "".join([html_escape_table.get(c, c) for c in
input_string])
output_string = input_string
```

Copied!
content_copy

6. Remove the # symbol from the first line, and add it to the beginning of the next line (ensure that you indent your code properly.) The final lines must resemble the following:

```
@app.route('/output')
def output():
  output_string = "".join([html_escape_table.get(c, c) for c in
input_string])
  # output_string = input_string
  return flask.render_template("output.html", output=output_string)
```

Copied!
content_copy

7. Press **CTRL + X** to exit nano, then **Y** to save your changes, and then **ENTER** to save your changes.
8. Copy the following command into the **SSH-in-browser** terminal:

```
python3 app.py
```

Copied!
content_copy
This command re-runs the application.

9. Press **ENTER**.

RESCAN THE WEB APPLICATON

1. Return to the **Cloud Web Security Scanner** page in the Google Cloud console.
2. Click **Run** to re-run the scan.