ENPM665 - Classwork Lab 5 - Operation Broken Bridge

Task 1 - Identity

Name: Kalpesh Bharat Parmar
UMD Directory ID[redacted]
Course and Section: ENPM665 0101
Date December 3, 2025

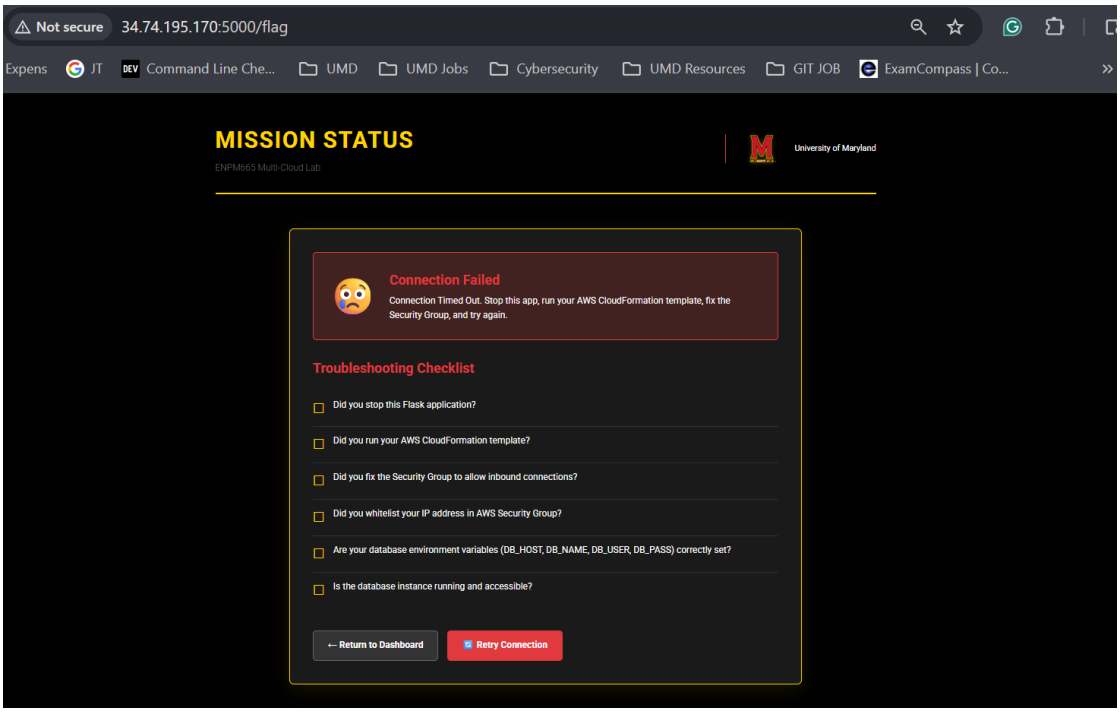
Section A: Mission Evidence

Task 2 - The Recon: Screenshot of the terminal command used to list/find the bucket.

```
kalpesh@cloudshell:~ (enpm665-demo-project)$ ls -la
total 40
drwxr-x--- 5 kalpesh kalpesh 4096 Dec 11 00:00 .
drwxr-xr-x 4 root    root    4096 Dec 10 23:59 ..
-rw------- 1 kalpesh kalpesh   18 Dec 11 00:00 .bash_history
-rw-r--r-- 1 kalpesh kalpesh  220 Mar 31  2024 .bash_logout
-rw-r--r-- 1 kalpesh kalpesh 3809 Dec  7 08:11 .bashrc
drwxr-xr-x 3 kalpesh kalpesh 4096 Dec  7 08:41 .config
drwxrwxr-x 2 kalpesh kalpesh 4096 Dec 10 23:59 .docker
drwxrwxr-x 3 kalpesh kalpesh 4096 Dec 10 23:59 .npm
-rw-r--r-- 1 kalpesh kalpesh  807 Mar 31  2024 .profile
-rwxr-xr-x 1 kalpesh kalpesh  913 Dec 10 23:59 README-cloudshell.txt
-rw-r--r-- 1 kalpesh kalpesh    0 Dec 10 23:59 .sudo_as_admin_successful
kalpesh@cloudshell:~ (enpm665-demo-project)$ gcloud storage ls
gs://enpm665-test-bucket-1/
gs://instructions-multi-cloud-gcp-aws-umd/
kalpesh@cloudshell:~ (enpm665-demo-project)$ cd ^C
kalpesh@cloudshell:~ (enpm665-demo-project)$ gcloud storage ls  gs://instructions-multi-cloud-gcp-aws-umd/
gs://instructions-multi-cloud-gcp-aws-umd/mission_briefing.pdf
gs://instructions-multi-cloud-gcp-aws-umd/multi-cloud-backend.yaml
gs://instructions-multi-cloud-gcp-aws-umd/source-code.zip
kalpesh@cloudshell:~ (enpm665-demo-project)$ gcloud storage cp gs://instructions-multi-cloud-gcp-aws-umd/
```

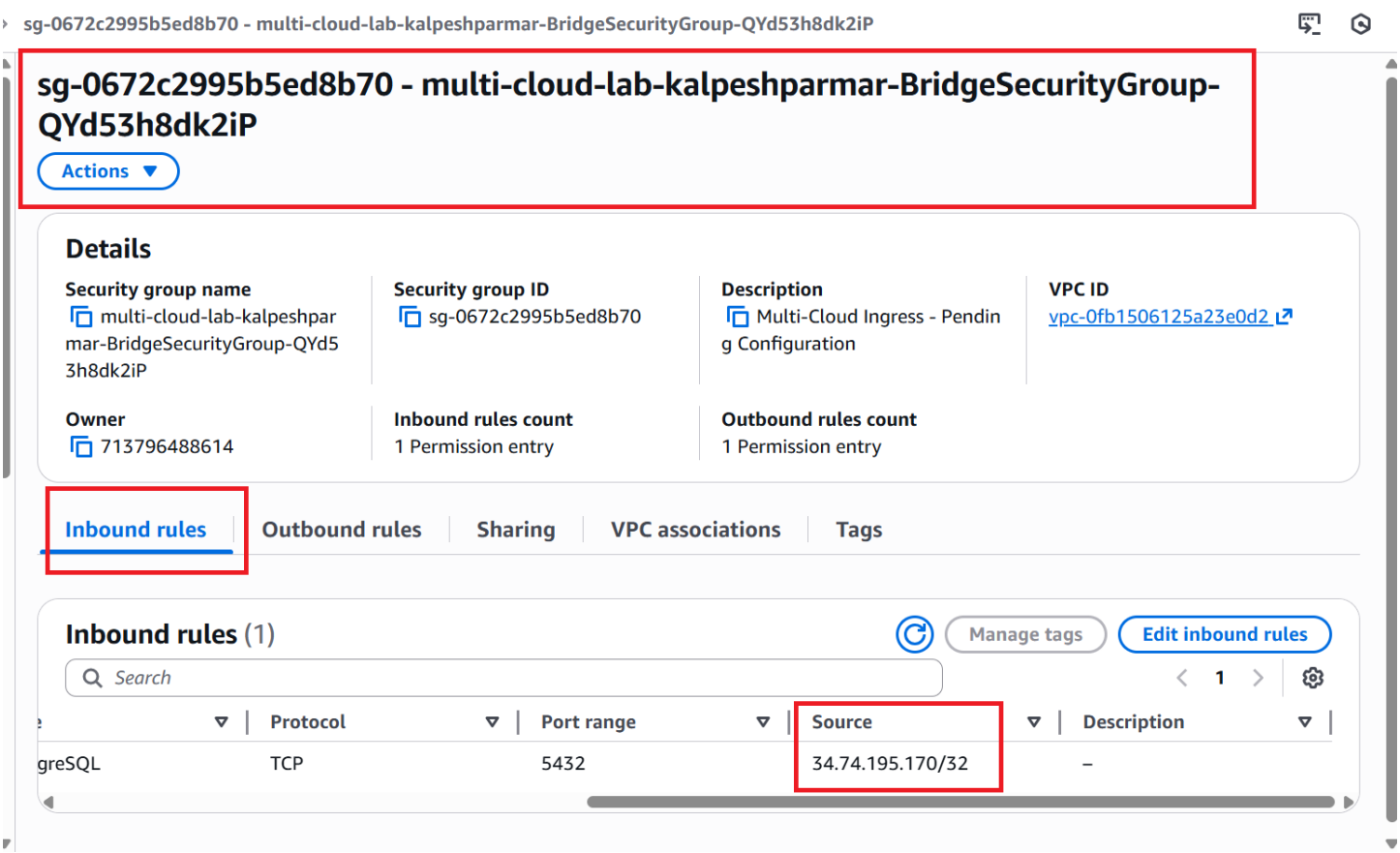
Screenshot shows recon activity where we navigated using gcloud storage command and found the three bucket/files which we download using the cp command and we found the Mission Briefing.

Task 3 - The Failure: Screenshot of the "Sad Face" / Connection Timeout on the Web App



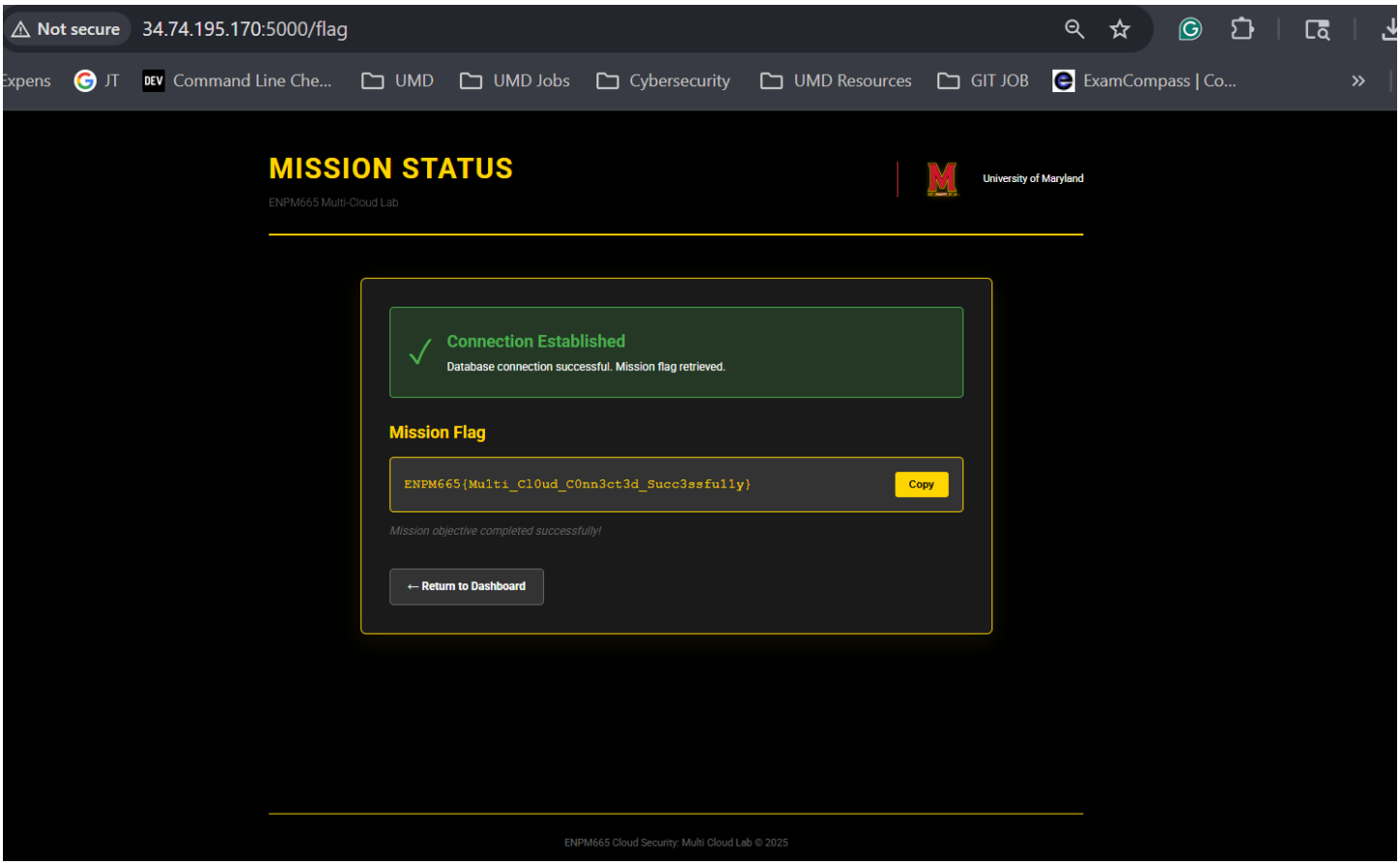
The above screenshot shows that after going through the mission briefing we deployed the application but after accessing the website we still didn't get the flag but we have been provided with few hints.

Task 3 - The Fix: **Screenshot of the AWS Security Group Inbound Rule you created.**



The above screenshot shows that we worked on the hints provided and update the inbound rules of firewall in the Security group in the AWS Console. This will enable request from our IP.

Task 4 - **The Success:** Screenshot of the "Mission Success" page with the Flag (or your best attempt)



The above screenshot shows successful implementation of security group policy which resulted in our IP being whitelisted and we successfully achieved the desired goal of the lab and found the flag "`ENPM665{Mu1ti_Cl0ud_C0nn3ct3d_Succ3ssfu11y}`".

Section B: After Action Report (Reflection)

Operational Analysis

What went well

The initial reconnaissance phase worked smoothly. Authenticating with gcloud auth login and enumerating the project storage buckets allowed me to quickly locate the deployment artifacts and retrieve the mission_briefing.pdf and backend template. Deploying the AWS CloudFormation stack also proceeded without issues, and the GCP Compute Engine instance launched correctly once I followed the instructions. The overall multi-cloud reconstruction process became clearer as I navigated through each phase of the mission.

What didn't go well

One of the first challenges I encountered involved the environment configuration. The .env.example file caused confusion initially, because the application would not load the variables until I realized that it needed to be renamed simply to .env. After reviewing the instructions and retracing the setup steps, I corrected the filename, and the application recognized the configuration properly.

In addition, diagnosing the network failure required careful troubleshooting. Although both clouds were deployed correctly, the frontend consistently showed the "Sad Face" connection error. This required checking the AWS Security

Group rules and identifying that my GCP VM's public IP had not been added as an inbound source on Port 5432. Only after modifying the rule did the bridge complete successfully.

What I would consider doing differently if deploying again

If I were to repeat this exercise, I would rely more heavily on Cloud Shell from the beginning. Initially, I installed the Google Cloud CLI locally and used my own system, but I later realized that Cloud Shell is significantly more efficient, consistent, and already configured with the necessary tools. Going forward, I will prioritize Cloud Shell for exercises like this to reduce setup friction and avoid version or configuration issues.

Additionally, I would document the network dependencies before deploying any stack—specifically which resource requires access to which service and on what port. This would help shorten debugging time when diagnosing cloud-to-cloud connectivity issues.

Security Posture Assessment

Reflecting on broader cloud security principles from this semester, there are several ways the security posture of this multi-cloud environment could be improved:

1. Identity & Access Management (IAM) Hardening

Permissions in both AWS and GCP should follow strict least-privilege principles. The service accounts and CloudFormation execution roles could be further restricted so they only have access to the specific actions required. This minimizes the attack surface and reduces lateral movement risks.

2. Secrets Management Improvements

Although the .env file worked for this lab, storing secrets directly on a VM is not an ideal production practice. A more secure approach would be to use GCP Secret Manager or AWS Secrets Manager, or even a cross-cloud secrets retrieval mechanism. This avoids plaintext storage of database passwords and improves auditability.

3. Network Architecture: Prefer Private IP Connectivity

The most significant improvement would be replacing public IP communication with a private, encrypted cross-cloud link. Options include a site-to-site VPN, VPC peering through a transit gateway, or a dedicated interconnect. This would prevent the database from being exposed to public networks entirely, even with restricted firewall rules.

4. Granular Firewall Controls and Logging

Firewalls should implement zero-trust principles by only permitting traffic from explicitly required resources. Enabling VPC Flow Logs on AWS and Firewall Insights on GCP would help detect unexpected or malicious traffic patterns and provide better visibility during incident response.

Understanding Why the Bridge Broke

The mission briefing indicated that the adversary intentionally severed the network bridge. When rebuilding the topology, the AWS Security Group did not allow inbound PostgreSQL traffic from the GCP VM's IP. This missing rule was the core reason the frontend displayed a connection timeout. Once the IP was whitelisted, the bridge was restored, and the flag retrieval mechanism worked as intended.

Conclusion -

his exercise provided valuable hands-on experience in reconstructing a multi-cloud architecture, diagnosing real-world connectivity failures, and applying security principles learned throughout the semester. By working through each phase from identifying deployment artifacts to resolving the broken network bridge. I gained deeper insight into how cloud services interact and how easily a small misconfiguration, such as a missing inbound rule or an incorrectly named environment file, can disrupt an entire workflow.

Beyond solving the technical challenges, this lab reinforced the importance of secure design practices, including least-privilege IAM, proper secrets handling, and avoiding unnecessary public exposure of cloud resources. Overall, the

mission strengthened my understanding of cloud security operations and improved my confidence in troubleshooting distributed systems across different platforms.