Deploy a New Cloud-Based Network

Dominic J. Mallo

Western Governors University

Published with Approval

Table of Contents

Summary

The Florida College of Fish Keeping is a non-profit educational institute that provides training for undergraduate students looking to become professional or hobbyist Fish Keepers. They have been in business since 2018, and have ~80 alumni, 40 actively enrolled students and 200 prospective students. Since its founding, the college has only offered its classes in person, from its campus in New Port Richey, Florida. Their campus building sits on 10 acres of land and can currently seat a maximum of 100 students. Recently, the college has been named one of the fastest-growing in their sector. Due to industry trends and market demand, they have formulated plans to start offering their courses online for the upcoming academic year.

The majority of the college's students pay for tuition with Federal Financial Aid (such as loans and the Pell Grant) in addition to Institutional Financial Aid (such as grants and scholarships). Previously, the Financial Aid process was completed entirely in person, using a paper-based system. This required prospective students to come on campus and meet with a team of two Financial Aid Advisors to apply for, review, and accept aid.

Leading up to the launch of this project, the college's administration realized their former Financial Aid system would have been a significant bottleneck in servicing the needs of remote and online students. For the college to offer remote courses, it needed to modernize its Financial Aid system with new technology. They chose to invest in an application suite, from a well-known vendor, that can facilitate the entire Financial Aid process. For students, the software offers a web-based portal to apply for & accept loans/scholarships/grants. Likewise, for staff, the software includes a desktop application to review, approve, and issue Financial Aid to students. To enable remote connections, the application is required to be deployed in a cloud network environment.

The Florida College of Fish Keeping hired Tampa AWS Architects as a contractor to assist in building a new cloud environment to run the Financial Aid application. Included in the scope of this project, Tampa AWS Architects designed, built, and secured a cloud network using the AWS public cloud. After the network infrastructure was deployed, a separate contractor configured the application. The exact configuration of the Financial Aid application was out-of-scope for this project. During the planning phase, the application vendor provided the following requirements that dictated how Tampa AWS Architects designed the network:

- Servers (Qty. 5)
    - Student Portal: One Unix Server with the Apache web-server service.
    - Staff Workstations: Two Unix Workstations with Remote Desktop Services.
    - Database: One Unix SQL Server.
    - Maintenance: One Unix SSH Server to serve as a bastion-host.
- Network Segmentation & Security
    - One Internet Gateway to enable public internet access.
    - Two subnets to isolate public and private access.
    - One NAT Gateway to route external network traffic to the private subnet.
    - Up to Four Security Groups to isolate various types of traffic.
    - One Web Application Firewall to defend the student portal.

Tampa AWS Architects approached the execution of this project in five stages:

Phase One: Tampa AWS Architects designed a logical topology for the cloud network infrastructure. For this, they used the Microsoft Visio diagramming software to draw a detailed topology showing all of the network components (i.e., servers, subnets, security groups, firewalls, etc.) and their logical connectivity to each other.

Phase Two: Tampa AWS Architects built the new network and documented all configurations.

1) First, they created an AWS cloud console account and deleted the default configurations.
2) Second, they created a new VPC (Virtual Private Cloud) named “Florida College of Fish Keeping (VPC)” and added two subnets named “Public-Subnet” and “Private-Subnet”.
3) Third, they configured separate security groups for the Student Portal, Bastion Host, Database Server, and Staff Workstation servers. Each security group limits the inbound connections to only permit specific TCP ports from other security groups, necessary for the operation of the servers within another group. For example, the Bastion Host security group only permits inbound secure shell (SSH) traffic on TCP port 22, from the college’s campus network and Tampa AWS Architect’s secure management network. Likewise, the Student Portal security group only permits inbound secure web traffic (HTTPS) from the CloudFlare Firewall (WAF) proxied IP range.
4) Fourth, EC2 server instances were spawned in each subnet/security group according to requirements defined by the application vendor. After the first Staff Workstation was configured, a custom Amazon Machine Image (AMI) was created so an identical workstation could be easily cloned for each workstation needed, now and in the future.
5) Fifth, a domain name for “fishcollege.org” was purchased from Google Domains.
6) Sixth, a CloudFlare account was registered for the WAF and linked with the domain name by changing the name servers listed in the Google Domains account.
7) Seventh, in CloudFlare, an A-type DNS entry was created for “fa.fishcollege.org” that points to the student portal public IP address issued by AWS. This domain was chosen by the college’s marketing department, where “fa” is an acronym for financial aid.

Phase Three: Tampa AWS Architects collaborated with a third-party contractor to configure the college's new Financial Aid software within the cloud environment. Although the exact configuration of the software was out-of-scope for this project, Tampa AWS Architects was still responsible for ensuring the infrastructure was capable of hosting the application.

1) First, Tampa AWS Architects connected to the Bastion Host server via SSH with the PuTTY utility.
2) Second, from the bastion host, they launched an SSH session to the public student portal and provided the software installation contractor temporary access to run the web service installation script.
3) Third, Tampa AWS Architects verified the web portal software was accessible from the public internet at the https://fa.fishcollege.org address.
4) Fourth, from the bastion host, Tampa AWS Architects launched an SSH session to the staff workstations and installed the Linux MATE desktop environment and xRDP service. These two services are used to provide a user-friendly remote desktop experience for the college's staff – who will use the workstations to process student aid applications in the cloud.
5) Fifth, the PuTTY session settings (on Tampa AWS Architect's management workstation) were modified to enable forwarding of localhost ephemeral port 3387 and 3388 to the two staff workstations' port 3389 (xRDP), proxied through the bastion host encrypted SSH session.
6) Sixth, a remote desktop session to the staff workstation was established and the application installation contractor was given temporary access to install the staff suite.

7) Seventh, from the bastion host, Tampa AWS Architects launched an SSH session to the database server and let the application vendor install the MySQL database utility.
8) Eighth, Tampa AWS Architects verified that the database was accessible from the staff workstations and student portal web server.

Phase Four: Tampa AWS Architects ensured the security controls in place were adequate to protect the Confidentiality, Integrity, and Availability (CIA) of the cloud environment. To accomplish this, the following cases were verified by attempting to reach each service the resource provides and observing the port listening status with the nmap utility:

1) The student web portal is only accessible when proxied through the CloudFlare WAF.
2) The Web Application Firewall blocks malicious traffic to the student web-portal server.
3) Staff workstations are only accessible from the bastion host proxy.
4) The database server is accessible from the web-portal server and staff cloud instances.
5) The bastion host can reach all EC2 servers, workstations, and instances.
6) The bastion host is only accessible via SSH from the college's local staff network and Tampa AWS Architect's secure maintenance environment.

Phase Five: Tampa AWS Architects provided the college's Financial Aid staff with guidance to remotely access their virtual workstations within the cloud environment. This involved outlining a procedure staff can follow when connecting to the remote desktop interface of their cloud workstations.

Review of Other Work

Article Review One: Cloud Elasticity

In an article from CloudZero, they explain the cloud computing concept of elasticity. They explain that "cloud elasticity is the ability to gain or reduce computing resources such as

CPU/processing, RAM, input/output bandwidth, and storage capacities on demand without causing system performance disruptions" (cloudzero.com). The article names "Amazon Web Services (AWS) and Google Cloud" as "public cloud providers [that] support rapid elasticity" (cloudzero.com). Likewise, they describe the various types of cloud elasticity. For example, "Scaling up or down refers to … adding/removing resources" and "Scaling out or in refers to expanding/shrinking" (cloudzero.com). For the cloud-network deployment project, Tampa AWS Architects utilized cloud elasticity, as outlined by CloudZero, to control the resources initially provisioned to the EC2 instances – leaving open the option to scale up or down as needed during the high/low seasons of Financial Aid applications.

Article Review Two: Security Groups

In a whitepaper article from Amazon Web Services (AWS), they explain the cloud computing concept of security groups. When defining security groups, they compare them to firewalls used in traditional network environments saying, "A security group acts as a virtual firewall, controlling the traffic that is allowed to reach and leave the resources that it is associated with" ("VPC: Control Traffic to Resources Using Security Groups," 2016). Similar to a firewall, the article explains, a security group has "rules that control the traffic based on protocols and port numbers" ("VPC: Control Traffic to Resources Using Security Groups," 2016). Once a group is configured, the article states, "an EC2 instance [can be] associate[d]" allowing the group to "control the inbound and outbound traffic for the instance" ("VPC: Control Traffic to Resources Using Security Groups," 2016). Tampa AWS Architects used security groups in this project to segment traffic between the student web portal and staff workstations.

Article Review Three: Bastion Hosts

In an article written by Brian Johnson (2022), a Security Engineer at strongDM, he describes the cloud computing concept of bastion hosts. To describe bastion hosts, he writes, “a bastion host is a server used to manage access to an internal or private network from an external network - sometimes called a jump box or jump server” (strongdm.com). Likewise, concerning their security, Brian writes “because bastion hosts often sit on the Internet, they typically run a minimum amount of services in order to reduce their attack surface” (strongdm.com). Finally, when describing applications for bastion hosts, Brian states “they are also commonly used to proxy and log communications, such as SSH sessions.” (strongdm.com). Tampa AWS Architects utilized the concept of bastion hosts, as outlined by strongDM, to act as an intermediate proxy between the AWS private-subnet resources and workstations connecting via the public internet.

Article Review Four: Web Application Firewall (WAF)

In an article written by CloudFlare, a leader in cloud security, they explain the cloud computing security concept of Web Application Firewalls. They write, “A WAF or web application firewall helps protect web applications by filtering and monitoring HTTP traffic between a web application and the Internet.” (cloudflare.com). They state WAF deployments “protect web applications from attacks” (cloudflare.com). When an application is public-facing it can become targeted by malicious actors with attack vectors, CloudFlare outlines, such as “cross-site forgery, cross-site-scripting (XSS), file inclusion, and SQL injection, among others” (cloudflare.com). However, WAFs can act like “a shield is placed between the web application and the Internet” (cloudflare.com). As part of the cloud-network deployment project, Tampa AWS Architects deployed the student web-portal server as an internet-facing web application.

Thus, to protect the application from cyber-attacks, Tampa AWS Architects placed a WAF between the internet and the student web portal server.

Article Review Five: MySQL Cross-Server Client Access

In an article written by Linuxize, a Linux DevOps publisher, they explain the steps to configure MySQL remote access between client and server machines. They write, "by default, the MySQL server listens for connections only from localhost, which means it can be accessed only by applications running on the same host" (linuxize.com). While this default configuration can be useful in some use-cases, they explain, that when "you [need] to connect to the remote MySQL server from your local system or a multi-server deployment … the best option is to … configure the MySQL server to accept remote connections" (linuxize.com). As part of the cloud-network deployment project, Tampa AWS Architects will be installing the MySQL database application on a dedicated server, separate from those that run the student web portal and staff applications. Thus, to enable remote connections, Tampa AWS Architects configured the database server to accept remote connections on its private 10.0.2.141 network interface.

Article Review Six: AWS EC2 Remote Desktop Services

In an article written by Akash Mahapatra, a Cloud Solutions Engineer, he explains the steps to configure RDP (Remote Desktop Protocol) on Unix-based AWS EC2 instances. He writes, "Many times having a GUI setup in your Linux servers can come in handy at times when you wish to show certain features or components of your development work" (medium.com). In the guide, he outlines the process of creating user accounts, installing the xRDP server service, installing the XFCE desktop environment, configuring the instance security group to permit RDP (TCP port 3389) traffic, and finally initiating remote sessions. Tampa AWS Architects followed this configuration guide when setting up remote desktop access to the staff workstations.

Article Review Seven: Custom Amazon Machine Image (AMI)

In a whitepaper article from Amazon Web Services (AWS), they explain the purpose and process of creating a custom Amazon Machine Image (AMI). To create an AMI, the article states to "find an existing AMI that is similar to the AMI that you'd like to create, [then] customize the instance" ("Create an Amazon EBS-backed Linux AMI," 2016). Next, the article explains when the image is captured, "During the AMI-creation process, Amazon EC2 creates snapshots of your instance's root volume" ("Create an Amazon EBS-backed Linux AMI," 2016). After the image is captured, it can be used when launching new instances – essentially creating a clone of the original instance. Tampa AWS Architects followed this whitepaper to create a custom AMI from the first EC2 staff workstation they launched. Then, the AMI was used to launch an identical EC2 instance for the second staff workstation. The benefit of this method is reduced setup time and high consistency between instances that serve identical purposes.

Changes to the Project Environment

Original

In the original project environment, The Florida College of Fish Keeping processed financial aid using a paper-based system. With this system, they had a paper template that organized student information, such as contact, FAFSA data, and eligibility for aid awards. The college then scanned a copy of this paper document onto a locally hosted Dell PowerEdge T310 file server – that ran Windows Server 2012. The printed copy would be placed in a locked file cabinet. The staff accessed this file server from their Dell OptiPlex 3090 workstations running Windows 10 Enterprise. This system made it very challenging for the Financial Aid department to run reports, such as calculating the total amount of institutional aid awarded to each student, because it required staff to tediously comb through all award letters 1-by-1.

Changes

In the new project environment, the college decommissioned their Dell PowerEdge T310 file server because (1) the security department had concerns about configuring a DMZ into the college's local area network and (2) the rapid-elasticity provided by AWS was very appealing to the administration – who is anticipating rapid growth in the coming months. Now, the role of storing student data is handled by the EC2 instance database server. Likewise, rather than filling information on a paper form, staff can now use their cloud-based EC2 workstations to process secure electronic forms. When needed, these forms can be easily shared with students via the student web portal, rather than mandating a physical on-campus visit. The college's staff did keep their Dell OptiPlex 3090 workstations, although, in the new environment, these are now primarily used as jump boxes into new cloud workstations via tunneled remote desktop services. Overall, the most significant change is that all financial aid processing workflows – such as students applying for aid and staff issuing awards – are now managed with the new software running in the cloud-based AWS environment deployed by Tampa AWS Architects.

Methodology

For this project, Tampa AWS Architects followed the ADDIE project management methodology. In this method, there were five phases followed: Analysis, Design, Development, Implementation, and Evaluation (Kurt, 2018). The phases of The Florida College of Fish Keeping's cloud-network deployment project are laid out below:

Analysis – Tampa AWS Architects met with the college to review their cloud environment requirements. This review was beneficial for Tampa AWS Architects to analyze the needs defined by the Financial Aid Processing Software. Additionally, Tampa AWS Architects

began to note the required infrastructure components – which helped to craft the project scope and timeline.

Design – Tampa AWS Architects took the information gathered during the analysis phase and created a detailed network diagram. During this phase, Tampa AWS Architects also designed a connectivity matrix between systems and defined security requirements. Microsoft's Visio graphing software was used to draw the topology showing the relationship between all the deployed AWS components, such as VPC, subnets, security groups, NAT gateway, Internet Gateway, and EC2 instances.

Development – Tampa AWS Architects developed the steps needed for creating the cloud network environment. This phase involved breaking down the project's goals into manageable objectives and deliverables.

Implementation – Tampa AWS Architects accomplished all of the project deliverables, which included deploying AWS EC2 Instances, configuring security groups, testing system connectivity, implementing a Web Application Firewall, installing the cloud-based Financial Aid processing software, and creating documentation for how the college's staff access their virtual cloud workstations.

Evaluation – During this phase, Tampa AWS Architects evaluated the cloud network's performance. Various tests were performed, such as verifying web traffic to the student portal is permitted from the public internet, private staff instances are stable when handing remote-desktop connectivity, and the security requirements were properly implemented. This was to ensure the cloud-network environment would be reliable in production. A final meeting with The Florida College of Fish Keeping's Financial Aid team was conducted to provide documentation and training.

Project Goals and Objectives

Goals, Objectives, and Deliverables Table

	Goal	Supporting objectives	Deliverables enabling the project objectives	Met/Unmet
1	Build a new cloud network for the college's new software.	**1.a.** Design the cloud network.	**1.a.i.** List infrastructure components.	Met
			1.a.ii. Build a network topology.	Met
			1.a.iii. Build a connectivity matrix.	Met
		1.b. Build the cloud network.	**1.b.i.** Deploy AWS EC2 Instances, Internet Gateways, and other essential network infrastructure.	Met
2	Secure the College's cloud network	**2.a.** Implement and validate security controls.	**2.a.i.** Deploy AWS Security Groups, Network Access Control Lists, and Router Rules.	Met
			2.a.ii. Test port-based network connectivity between cloud resources.	Met
			2.a.iii. Configure a Web Application Firewall.	Met
			2.a.iv. Ensure malicious attacks, on the public student web server, are blocked.	Met
3	Install the college's financial aid software on the new cloud servers.	**3.a.** Install the cloud-based software.	**3.a.i.** Install the web application on the public student-accessible web server.	Met
			3.a.ii. Install the database application on the database server.	Met
			3.a.iii. Install the staff software on their cloud workstations.	Met
		3.b. Provide students & staff access to the cloud-based software.	**3.b.i.** Train staff on how to access their cloud-based workstations.	Met
			3.b.ii. Demonstrate to staff how they can provide students access to the publicly accessible student web portal, to apply for Financial Aid.	Met

Goals, Objectives, and Deliverables Descriptions

The cloud network deployment for The Florida College of Fish Keeping project's primary goal was to provide a cloud environment to run the college's new financial aid software. The new software now helps The Florida College of Fish Keeping process student aid with greater efficiency and is highly scalable. The new cloud network is critical to using this software and enables the Financial Aid department to handle the college's rapid growth in student

headcount and applications. The successful completion of this goal relied on these three objectives and their deliverables:

- **Objective 1.a:** Design the cloud network. The first step in building the college's cloud network was to design a 'blueprint' for the cloud network that defines the requirements for what Tampa AWS Architects created.
    - **Deliverable 1.a.i:** List infrastructure components. A list of the necessary network infrastructure was compiled, and a unique purpose was assigned to each component. For example, two Unix-Based EC2 instances with remote desktop services were listed as required for staff to access the cloud environment.
    - **Deliverable 1.a.ii:** Build a network topology. This high-level diagram illustrated the connectivity and relationship between the infrastructure components. For example, an EC2 instance labeled "Database Server" was drawn in a Security Group box labeled "Database Server Group", within a subnet box labeled "Private Subnet", connected to a router, connected to another subnet box labeled "Public Subnet", hosting a NAT Gateway, within a VPC box, hosting an Internet Gateway, within the AWS cloud environment.
    - **Deliverable 1.a.iii:** Build a connectivity matrix. A detailed matrix was created showing which systems should be able to communicate to and from each other. For example, the private staff workstation security group is configured to deny access from the public student portal web server.
- **Objective 1.b:** Build the cloud network. The second step in building the college's cloud network was to deploy the necessary infrastructure components.

- **Deliverable 1.b.i:** Deploy AWS EC2 Instances, Internet Gateways, and other essential network infrastructure. Network infrastructure was deployed in the new cloud Virtual Private Network (VPC). This involved spinning up the necessary EC2 workstations and servers and other network infrastructure in preparation for the College's financial aid software being installed.

- **Objective 2.a:** Implement security controls. The cloud environment is used to store very sensitive Personally Identifiable Information (PII), such as student social security numbers, Expected Family Contribution (EFC), Grade Point Averages (GPA), and other information used to calculate financial aid awards. Thus, the cloud network was secured to the fullest. Once the security measures were implemented, they were tested and verified as functional.
  - **Deliverable 2.a.i:** Deploy AWS Security Groups (SGs), Network Access Control Lists (NACLs), and route rules. This deliverable involved configuring the logical enforcements for the connectivity matrix created in Deliverable 1.a.iii. For example, the student & staff systems are isolated from each other – yet access the same central database server. A unique security group was created for the staff workstations, database server, bastion host, and student portal (four SGs total).
  - **Deliverable 2.a.ii:** Test port-based network connectivity between cloud resources. After the traffic enforcement was configured in AWS, nmap was run on all EC2 resources to ensure that the security groups, NACLs, and routing rules were configured correctly.
  - **Deliverable 2.a.iii:** Configure a Web Application Firewall (WAF). A publicly accessible student portal web server was deployed to support the new financial aid

software. Since this portal is public and acts as a front-end into highly sensitive student PII, a Web Application Firewall was implemented to protect against attacks targeting the OSI Application Layer 7 – such as SQL injections and brute force attacks. In CloudFlare, managed rules were configured to block a wide range of common attack vectors and exploits.

  - **Deliverable 2.a.iv:** Ensure malicious attacks, on the public student web server, are blocked. With the WAF in place and the student web portal online, the WAF was tested to ensure malicious traffic is blocked. This was achieved by sending a sample malicious SQL injection command payload to the web portal and verifying the WAF blocked the request.

- **Objective 3.a:** Install the cloud-based software. With all network infrastructure deployed and properly secured, the college's new financial aid software was then installed. A separate contractor worked with Tampa AWS Architects to run the installation script on each server.
  - **Deliverable 3.a.i:** Install the web application on the public student-accessible web server. For this deliverable, Tampa AWS Architects ran the provided web application service installer on the EC2 Student Web Server.
  - **Deliverable 3.a.ii:** Install the database application on the database server. For this deliverable, Tampa AWS Architects ran the provided database application service installer on the EC2 Database Server.
  - **Deliverable 3.a.iii:** Install the staff software on their cloud workstations. For this deliverable, Tampa AWS Architects ran the provided financial aid processing software installer on each of the staff EC2 cloud workstations.

- **Objective 3.b:** Provide students & staff access to the cloud-based software. After the application had been deployed, it was then made available for production use by the college. This objective was focused on documenting how students & staff access the application.
    - **Deliverable 3.b.i:** Train staff on how to access their cloud-based workstations. The staff EC2 cloud workstations were configured with remote desktop services that are accessed via an encrypted bastion host proxy. For this deliverable, documentation was provided that explains the process of accessing the remote desktop terminals from the staff's physical Dell OptiPlex 3090 workstations.
    - **Deliverable 3.b.ii**: Demonstrate to staff how they can provide students access to the publicly accessible student web portal, to apply for Financial Aid. Now that the project was complete, students access their web portal by simply visiting "fa.fishcollege.org" and logging in with their given student credentials.

Project Timeline

Milestone	Planned Duration	Actual Duration	Actual Start Date	Actual End Date
List required infrastructure components.	30 Minutes	30 Minutes	8/8/2022	8/8/2022
Build a network diagram.	30 Minutes	1 Hour	8/8/2022	8/8/2022
Build a connectivity matrix.	30 Minutes	15 Minutes	8/8/2022	8/8/2022
Deploy AWS EC2 Instances, Internet Gateways, and other essential network infrastructure.	5 Hours	6 Hours	8/8/2022	8/8/2022
Deploy AWS Security Groups, Network Access Control Lists, and Router Rules.	3 Hours	1 Hour	8/9/2022	8/9/2022

Test port-based network connectivity between cloud resources.	30 Minutes	30 Minutes	8/9/2022	8/9/2022
Configure a Web Application Firewall.	2 Hours	1 Hour	8/9/2022	8/9/2022
Ensure malicious attacks, on the public student web server, are blocked.	30 Minutes	30 Minutes	8/9/2022	8/9/2022
Install the web application on the public student-accessible web server.	15 Minutes	15 Minutes	8/10/2022	8/10/2022
Install the database application on the database server.	15 Minutes	15 Minutes	8/10/2022	8/10/2022
Install the staff software on their cloud workstations.	15 Minutes	15 Minutes	8/10/2022	8/10/2022
Train staff on how to access their cloud-based workstations.	15 Minutes	15 Minutes	8/10/2022	8/10/2022
Demonstrate to staff how they can provide students access to the publicly accessible student web portal, to apply for Financial Aid.	1 hour	1 hour	8/10/2022	8/10/2022

The Tampa AWS Architects team worked diligently to keep the project very close to the projected schedule. There were no delays to the overall project structure. However, the overall duration of some deliverables did slightly fluctuate. For example, deploying AWS network infrastructure took 6 hours, which was 1 hour over the original estimate. This is because the web server needed to be terminated and re-creating using a different AMI (Amazon Machine Image / Operating System) after an incompatibility was found in the web portal software when using the default Amazon Linux 2 distribution. After re-creating the instance with an Ubuntu Linux Server 22.04 LTS AMI, the compatibility issue was resolved.

Unanticipated Requirements

An unanticipated requirement that emerged during the cloud-network deployment project was that a domain name needed to be acquired for the public student web portal. Before starting the project, Tampa AWS Architects assumed the college already owned a domain name and that they could simply add a new A-Type DNS entry. However, after a public IP address had been issued to the EC2 instance and given to the college's internal IT department, it was realized they did not own a custom domain name because their only internet presence had been on a Facebook business page. This was quickly overcome by holding an emergency meeting with the college president and marketing department, where they decided on purchasing the domain name "fishcollege.org" from Google Domains. After the domain was purchased, Tampa AWS Architects proceeded to configure the name server entries to point at the CloudFlare WAF and then issue an A-Type DNS entry for the Financial Aid portal at "fa.fishcollege.org".

Conclusions

The project's outcome is a highly functional cloud environment in the AWS platform that hosts a new financial aid processing software application. This system is now in production and available for use by the college's staff, current students renewing their aid applications, and prospective students interested in knowing how much aid they qualify for. In addition to the list of completed deliverables, which are a measure of success, the others include time savings and opening the college to new growth opportunities.

Concerning time savings provided by the cloud-network application, financial aid staff has reported previously it could take ~2 hours per student to fully calculate Federal / Institutional aid awards – but now, these equations can be performed in ~5 minutes. Likewise, previously staff reported they spent about 1-hour meeting with each student to collect information needed

for the application – but now, the self-serve cloud-based student web portal has completely removed the need for this interaction. Similarly, concerning growth opportunities, the cloud-network deployment project was a success in removing the barrier of requiring online prospective students from visiting the physical campus to apply for aid, since the application can now be completed via the self-serve cloud web portal.

Long term, this project will continue to benefit the college by enabling them to sustain a fully online version of their education program. Out of the 200 prospective students in the college's current enrollment funnel, when surveyed, 40% of prospects indicated they would be more likely to enroll if an online program were available and can only afford tuition if financial aid is offered. Thus, as demand continues to increase for a fully online program offering, the ability to remotely process aid applications virtually will become an even more critical part of the college's business strategy. Having met or exceeded all of the goals, objectives, and deliverables, this project is considered a success.

Project Deliverables

The deliverables section has nineteen figures which show the configuration of the cloud network and application in use.

Appendix A, Figure 1, shows the cloud environment network topology diagram. This was created using Microsoft's Visio graphing tool and shows the relationship/placement of all the deployed AWS components, such as VPC, NAT gateway, Internet Gateway, WAF, subnets, security groups, and EC2 instances.

Appendix B shows the domain and DNS (Domain Name System) configuration for the public student web portal server. Figure 2 shows the domain name is managed by the Google Domains registrar and has the name server entries pointing at the CloudFlare WAF. Figure 3

shows the CloudFlare DNS entries with an A-Type record for “fa.fishcollege.org” routing to the EC2 web server instance’s public IP address (after proxied by the WAF).

Appendix C shows a list of all AWS EC2 instances and subnet route tables. Figure 4 shows each EC2 instance’s public/private IP address, associated security group, instance type, and running state. Figure 5 shows the CIDR address range for the Public and Private subnets. Figure 6 shows the Private subnet route table; this is configured to forward all non-local traffic to the NAT Gateway in the Public subnet. Figure 7 shows the Public subnet route table; this is configured to forward all non-local traffic to the Internet Gateway.

Appendix D shows the inbound rule configuration for each security group (SG). Figure 8 shows the staff workstation SG permits SSH (TCP port 22) and RDP (TCP port 3389) traffic from the bastion host SG. Figure 9 shows the database server SG permits MySQL (TCP port 3306) traffic from the student portal SG, staff workstation SG, and bastion host SG; the bastion SG can also connect via SSH. Figure 10 shows the bastion host SG allows SSH traffic from the college’s staff on-campus network and Tampa AWS Architect’s management network. Figure 11 shows the student portal SG allows SSH traffic from the bastion host and HTTPS (TCP port 443) traffic from numerous CloudFlare WAF IPs; notice that the student portal SG does not allow HTTPS from 0.0.0.0/0 to prevent bypassing the WAF.

Appendix E shows the connectivity matrix between security groups and validation on the student portal. Figure 12 shows a graphic summary of allowed traffic to and from each security group; this matrix is the baseline for how the security groups were configured. Figure 13 shows the nmap network scanning tool running on the student portal web server, testing port-based connectivity to each of the other EC2 instances. The results of the nmap scan align with the connectivity matrix baseline, which guarantees the security groups were configured properly. For

example, when the student portal instance tries to reach the database server's MySQL service, that registers as open/permitted – however, connections to the SSH service are filtered/blocked.

Appendix F shows the student portal, bastion host, and staff workstations in action. Figure 14 shows a screenshot of the student web-portal home page, accessible via the "fa.fishcollege.org" public domain. Figure 15 shows the PuTTY SSH session configuration used on Tampa AWS Architect's management machine to connect to the AWS bastion host. Figure 16 shows the PuTTY SSH private key selected for authentication into the bastion host. Figure 17 shows the PuTTY SSH port tunnels used to proxy RDP traffic from the client machine to the AWS EC2 staff workstations. Figure 18 shows a screenshot from an RDP session at one of the staff workstations. Within the RDP session, simultaneous pages are open with connections to the staff-only financial aid software interface, MySQL database client, and public internet.

Appendix G, Figure 19, shows a custom Amazon Machine Image (AMI) that was captured from one of the staff workstations, enabling easy future replication if needed.

References

CloudZero. “What Is Cloud Elasticity?”, https://www.cloudzero.com/blog/cloud-elasticity.

“Create an Amazon EBS-Backed Linux AMI.” Amazon, AWS, https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html.

Johnson, Brian. “How to Create a Bastion Host in AWS.” StrongDM, StrongDM, 26 May 2022, https://www.strongdm.com/blog/bastion-hosts-with-audit-logging-part-one.

Kurt, Serhat. “Addie Model: Instructional Design.” Educational Technology, 16 Dec. 2018, https://educationaltechnology.net/the-addie-model-instructional-design/.

Linuxize. “How to Allow Remote Connections to Mysql Database Server.” Linuxize, Linuxize, 6 Jan. 2021, https://linuxize.com/post/mysql-remote-access/.

Mahapatra, Akash. “Setting up RDP in an Ubuntu AWS EC2 Instance.” Medium, Tecxperiments, 18 Oct. 2021, https://medium.com/tecxperiments/setting-up-rdp-in-an-ubuntu-aws-ec2-instance-b11044ca849b.

“VPC: Control Traffic to Resources Using Security Groups.” Amazon, AWS, 2016, https://docs.aws.amazon.com/vpc/latest/userguide/VPC_SecurityGroups.html.

What Is a WAF? | Web Application Firewall Explained | Cloudflare. https://www.cloudflare.com/learning/ddos/glossary/web-application-firewall-waf/.

Appendix A: Cloud Network Topology

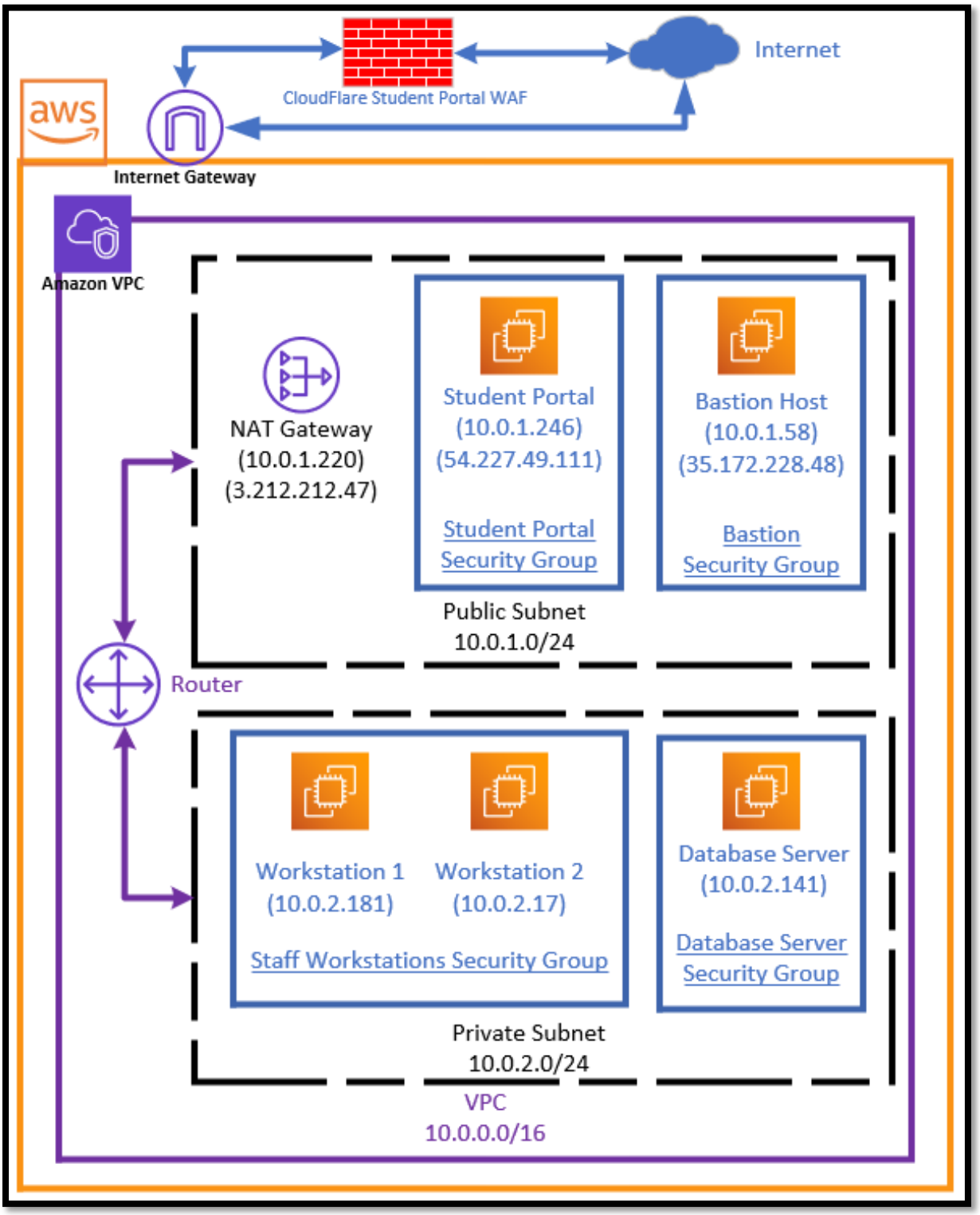


Figure 1- Cloud Network Diagram

Appendix B: Google Domains & CloudFlare DNS Management Settings

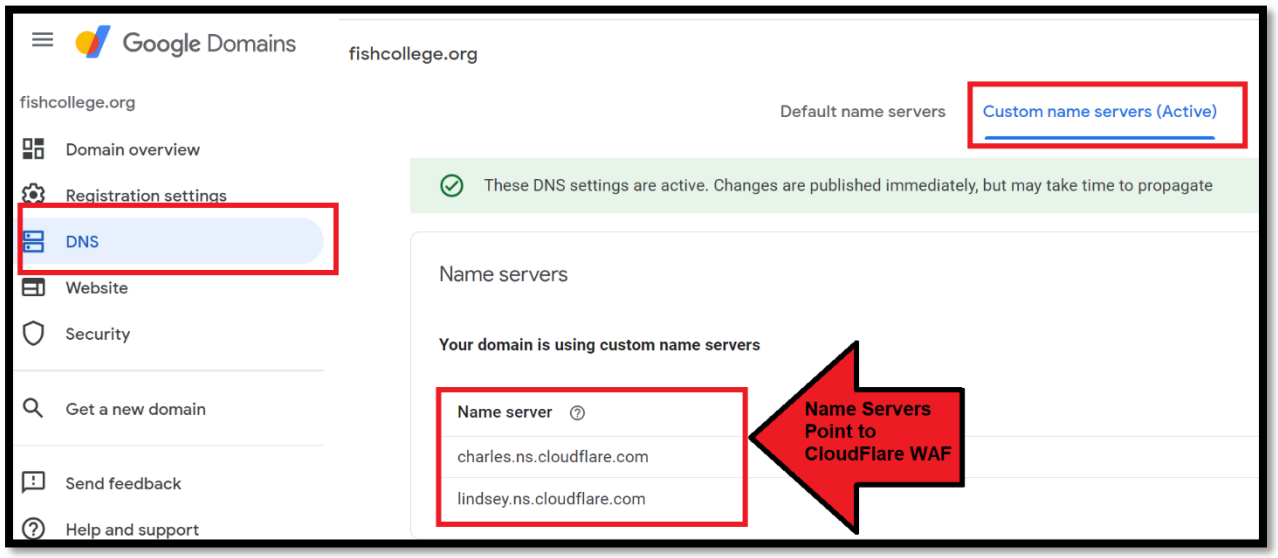


Figure 2 - Google Domains Name Server Configuration for Student Portal

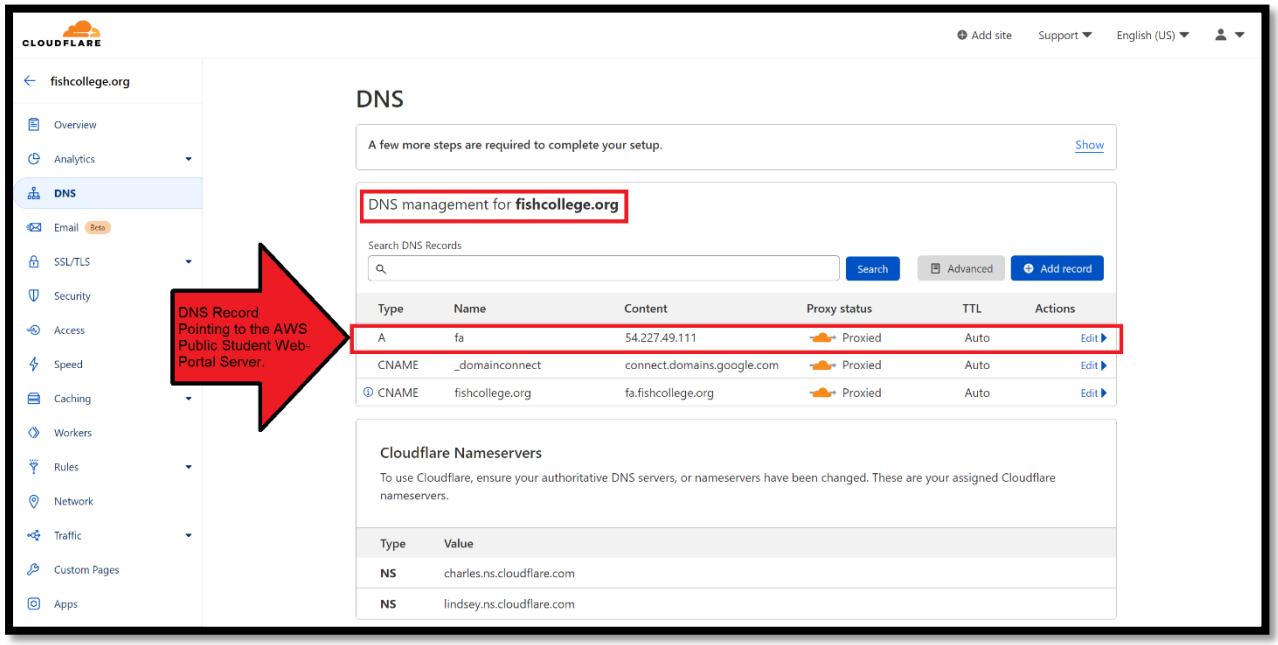


Figure 3 - CloudFlare DNS Management Settings

Appendix C: AWS EC2 Instance List and Subnet Route Tables

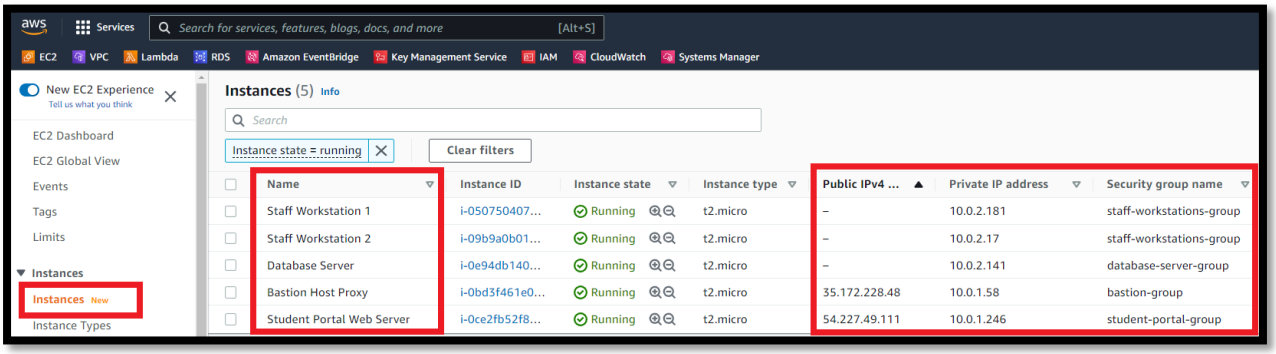


Figure 4- AWS EC2 Instance List

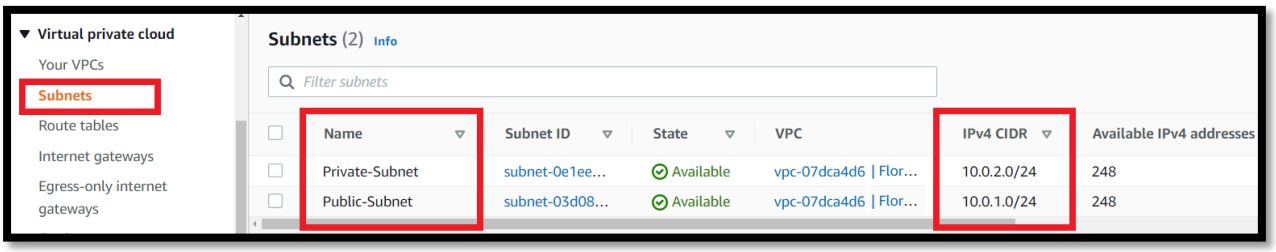
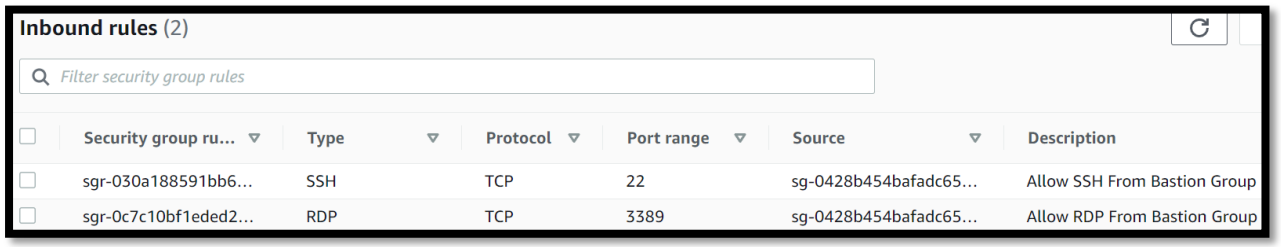


Figure 5 - VPC Subnets (Public & Private)

Figure 6 - Private Subnet Route Table

Figure 7 - Public Subnet Route Table

Appendix D: AWS Security Group Configurations

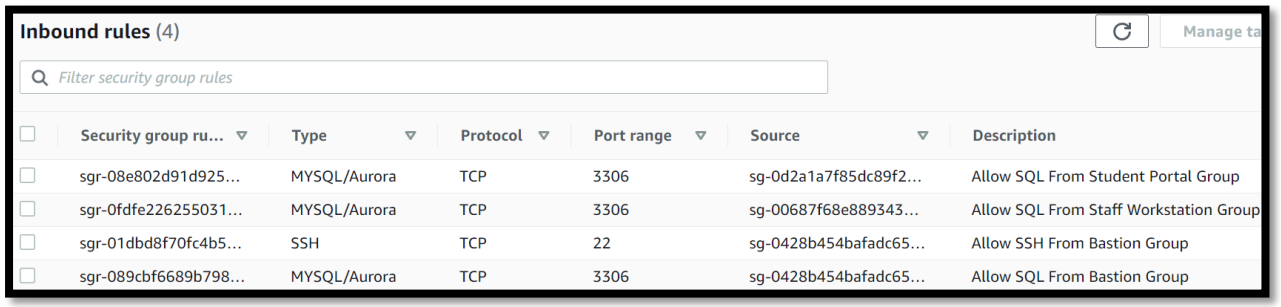


Inbound rules (2)

Filter security group rules

Security group ru...	Type	Protocol	Port range	Source	Description
sgr-030a188591bb6...	SSH	TCP	22	sg-0428b454bafadc65...	Allow SSH From Bastion Group
sgr-0c7c10bf1eded2...	RDP	TCP	3389	sg-0428b454bafadc65...	Allow RDP From Bastion Group

Figure 8 - Staff Workstations Security Group Configuration

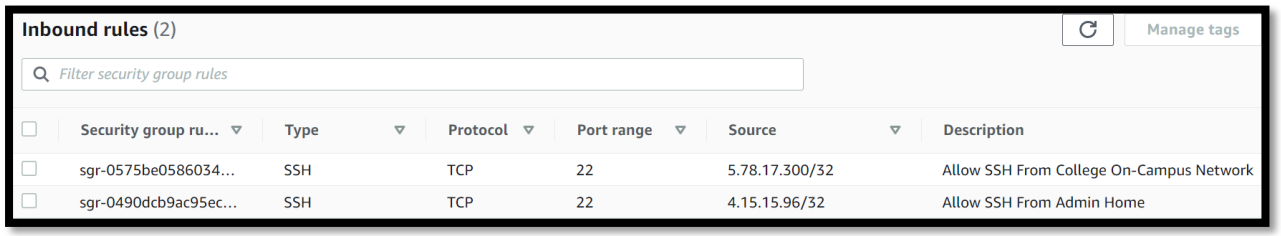


Inbound rules (4)

Filter security group rules

Security group ru...	Type	Protocol	Port range	Source	Description
sgr-08e802d91d925...	MYSQL/Aurora	TCP	3306	sg-0d2a1a7f85dc89f2...	Allow SQL From Student Portal Group
sgr-0fdfe226255031...	MYSQL/Aurora	TCP	3306	sg-00687f68e889343...	Allow SQL From Staff Workstation Group
sgr-01dbd8f70fc4b5...	SSH	TCP	22	sg-0428b454bafadc65...	Allow SSH From Bastion Group
sgr-089cbf6689b798...	MYSQL/Aurora	TCP	3306	sg-0428b454bafadc65...	Allow SQL From Bastion Group

Figure 9 - Database Server Security Group Configuration

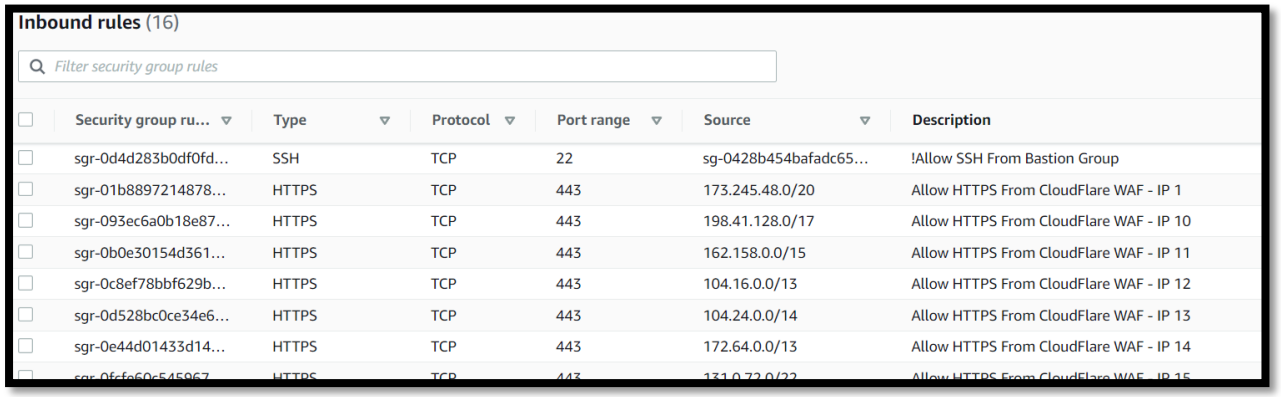


Inbound rules (2)

Filter security group rules

Security group ru...	Type	Protocol	Port range	Source	Description
sgr-0575be0586034...	SSH	TCP	22	5.78.17.300/32	Allow SSH From College On-Campus Network
sgr-0490dcb9ac95ec...	SSH	TCP	22	4.15.15.96/32	Allow SSH From Admin Home

Figure 10 - Bastion Host Security Group Configuration



Inbound rules (16)

Filter security group rules

Security group ru...	Type	Protocol	Port range	Source	Description
sgr-0d4d283b0df0fd...	SSH	TCP	22	sg-0428b454bafadc65...	!Allow SSH From Bastion Group
sgr-01b8897214878...	HTTPS	TCP	443	173.245.48.0/20	Allow HTTPS From CloudFlare WAF - IP 1
sgr-093ec6a0b18e87...	HTTPS	TCP	443	198.41.128.0/17	Allow HTTPS From CloudFlare WAF - IP 10
sgr-0b0e30154d361...	HTTPS	TCP	443	162.158.0.0/15	Allow HTTPS From CloudFlare WAF - IP 11
sgr-0c8ef78bbf629b...	HTTPS	TCP	443	104.16.0.0/13	Allow HTTPS From CloudFlare WAF - IP 12
sgr-0d528bc0ce34e6...	HTTPS	TCP	443	104.24.0.0/14	Allow HTTPS From CloudFlare WAF - IP 13
sgr-0e44d01433d14...	HTTPS	TCP	443	172.64.0.0/13	Allow HTTPS From CloudFlare WAF - IP 14
sgr-0fcfe60c545967...	HTTPS	TCP	443	131.0.72.0/22	Allow HTTPS From CloudFlare WAF - IP 15

Figure 11 - Student Web-Portal Security Group Configuration

Appendix E: Connectivity Matrix & Validation on Student Portal

FROM	TO				
	Staff Workstations	Database Server	Bastion Host	Student Portal	Public Internet
Staff Workstations	-	**MySQL**	**None**	**None**	**All**
Database Server	**None**	-	**None**	**None**	**All**
Bastion Host	**SSH & RDP**	**SSH & MySQL**	-	**SSH & HTTPS**	**All**
Student Portal	**None**	**MySQL**	**None**	-	**All**
Public Internet	**None**	**None**	**Limited SSH**	**HTTPS via CloudFlare**	-

Figure 12 - Connectivity Matrix

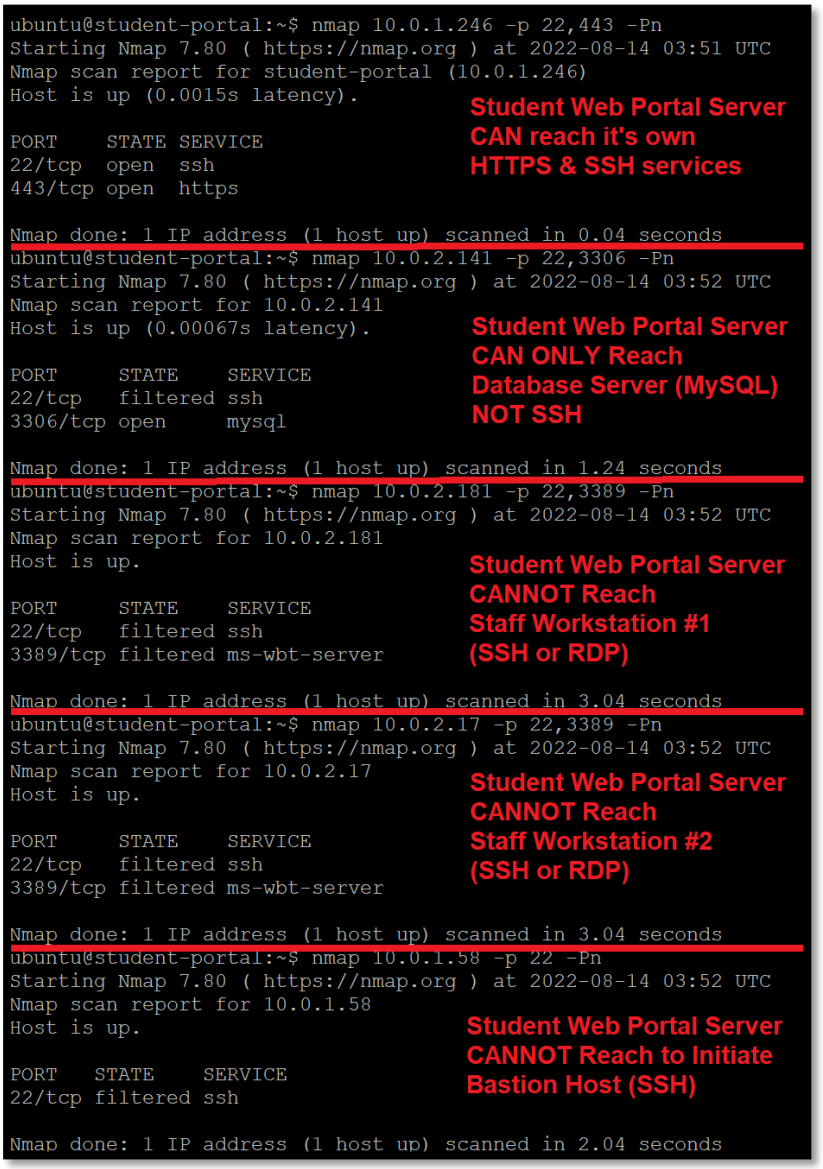


Figure 13 - Nmap Port Scan Testing Connectivity to All Other AWS Resources (Verify Security Group Rules)

Appendix F: Public Student Web Portal, Bastion Host SSH Config, and Staff RDP Session

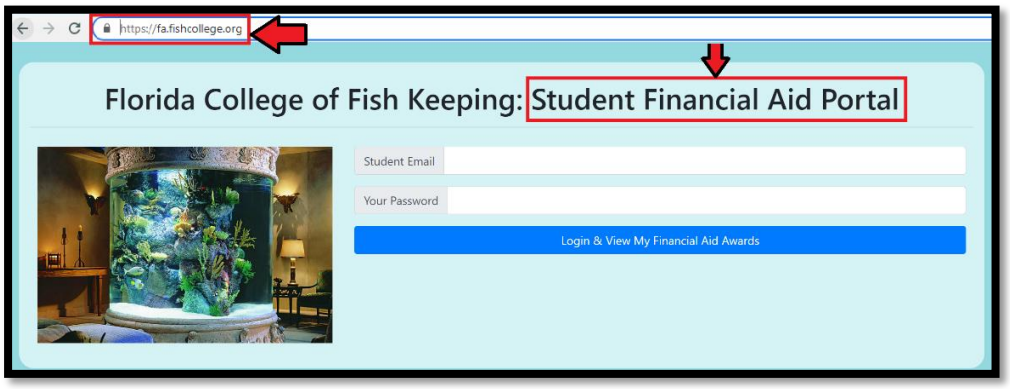


Figure 14 - Public Student Web-Portal Home Page

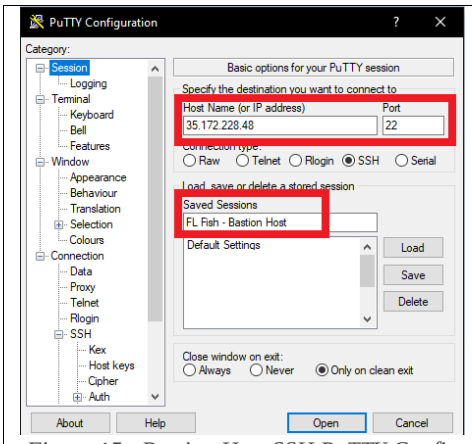


Figure 15 - Bastion Host SSH PuTTY Config (Main)

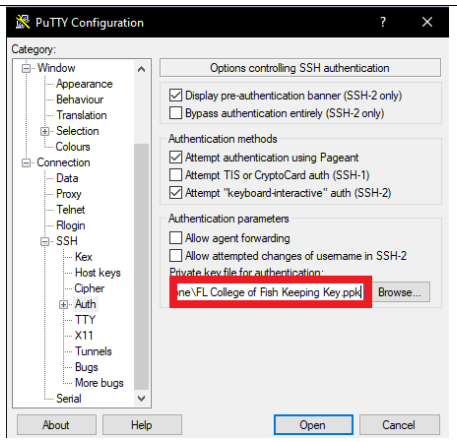


Figure 16 - Bastion Host SSH PuTTY Config (Key Auth)

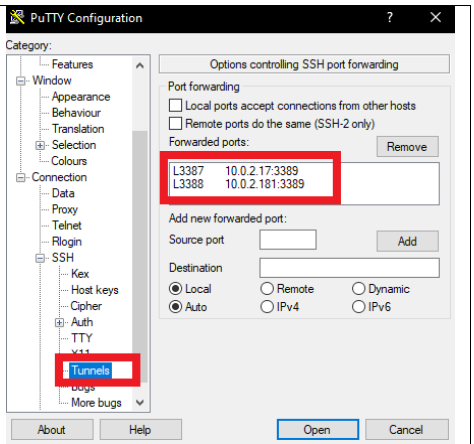


Figure 17 - Bastion Host SSH PuTTY Config (RDP Tunnels)

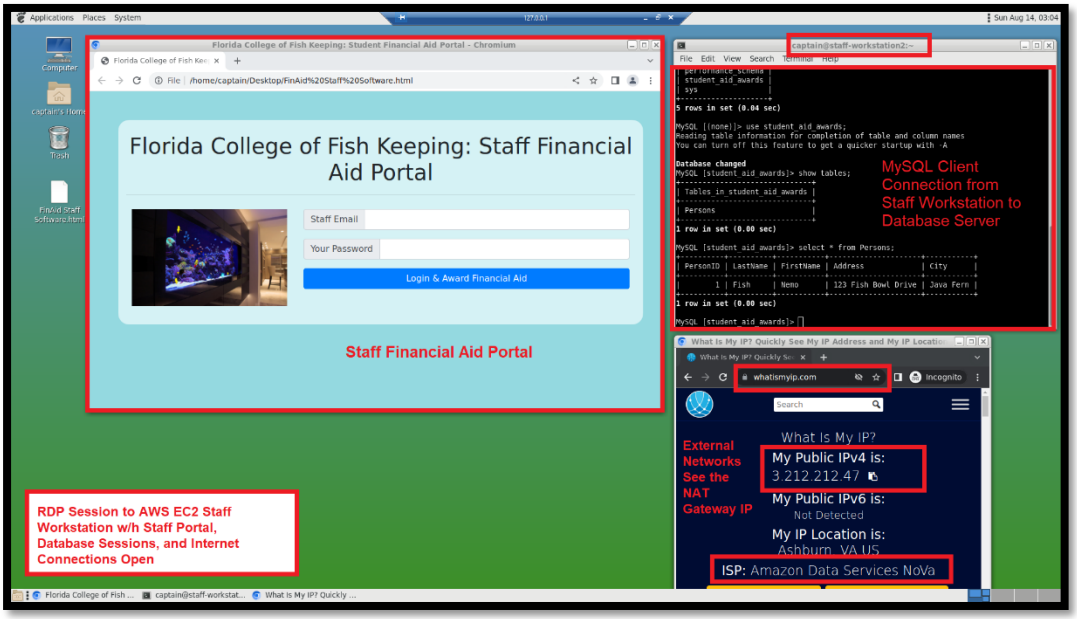


Figure 18 – Tunneled Staff Cloud Workstation RDP Session with Connections to Application, Database, and Internet

Appendix G: Custom Amazon Machine Image (AMI) for Staff Workstations

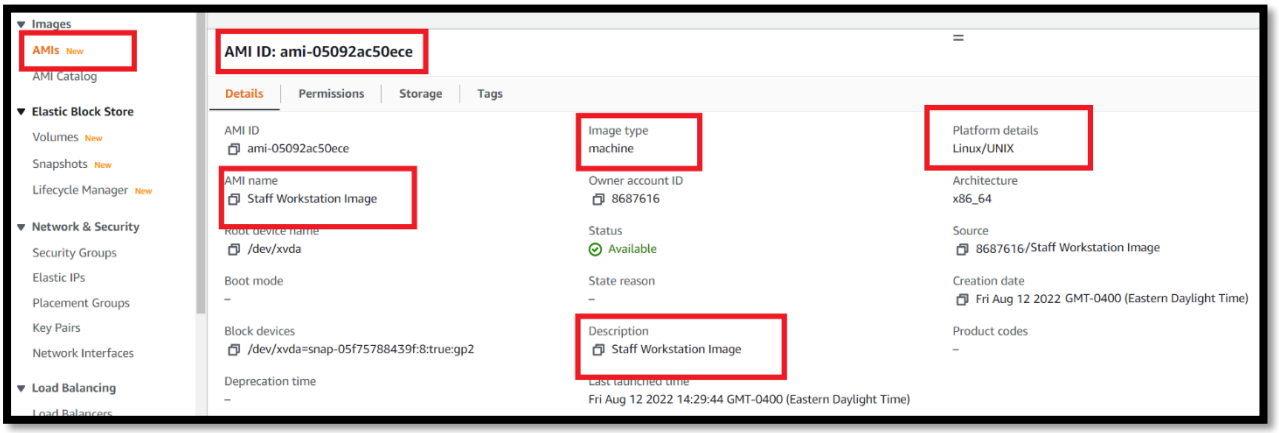


Figure 19 - Custom AMI for Staff Workstation