# Cloud Computing: Centralizing an Enterprise DevOps Tool in Amazon Web Services

CS4991 Capstone Report, 2023

Calvin Min
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
cjm9vr@virginia.edu

## ABSTRACT

Capital One, a leading bank in America, focuses on pioneering the industry by utilizing information and technology, specifically the cloud. However, when well-known banks try to incorporate technology into their business model, they become increasingly susceptible to security breaches. As a Technology Intern, I delivered an internal, enterprise-wide product that streamlined updates on company-owned virtual environments within Amazon Web Services. My team **(**two other interns and I) created a full-stack solution using Flask and AWS services to develop a user interface that incorporated a pre-existing DevOps tool called Automated Vulnerability Remediation (AVR). With the help of our full-stack application, Capital One employees can save around 400,000 hours by automating their updates within the cloud**.** In the future, the full-stack application needs to incorporate Capital One's human resources data to ensure that only authorized users can start updating jobs.

## 1. INTRODUCTION

During the 2020 COVID-19 pandemic, Capital One decided to move most of its infrastructure to AWS. The perk of this decision was to ensure their company could operate from anywhere in the world. However, with the recency of cloud computing and the unknowns about it, the public speculated that Capital One would run into security issues during its transition. Thus, AVR was one tool that Capital One created to mitigate the vulnerabilities found within the company's virtual machines.

AVR is a significant tool for the company due to its efficiency when updating virtual machines in parallel. The usual process to update an Elastic Cloud Compute (EC2) machine within a major corporation like Capital One takes numerous hours and multiple approvals. However, with the capabilities of AVR, employees who possess the right credentials can "spin up" updates on the fly. This becomes particularly necessary when vulnerabilities are found during off-hours.

## 2. RELATED WORKS

According to Alqahtani and Gull (2018), the premises of cloud computing present a lot of security issues. Some factors mentioned in their research were unidentified malware that existed under the host company's supervision, leftover credentials, and unsolicited connections. Even though cloud computing presents efficiency, a potential drawback of its use in a big corporation is the number of security vulnerabilities. My project connects with Alqahtani and Gull's research as Capital One recently migrated its infrastructure under the supervision of another company. Without direct control, Capital One is forced to rely on AWS to ensure that some areas of its infrastructure are secure.

Gorelik (2013) discussed the basic principles of cloud computing, and how any industry can apply its capabilities to its infrastructure. Specifically, Gorelik analyzed the cost structure found within cloud computing companies, and how users can effectively create a model to maximize their cloud usage while minimizing their costs. In addition to the security risks that result from the cloud, a price tag is placed on every item used. Therefore, customers of the cloud must be aware of their usage, and best practices are noted by Gorelik to decrease the amount of money spent. Gorelik's research directly impacted my project as my team researched best practices in terms of cost efficiency to create the best cloud infrastructure for our company. We prioritized scalability but kept in mind the cost that resulted from it.

## 3. PROJECT DESIGN

AVR is a complex web application that has multiple components working simultaneously. In this section, this report will review AVR's architecture, analyze the requirements for the project, and explain some of the challenges developers faced when creating the application.

### 3.1 Review of System Architecture

The web tool for AVR is a monolithic application using Python Flask. The application contained both its front-end and back-end in the same repository. The idea of this architecture was to make it easier for developers to implement new features but hindered its scalability in the future. As a monolithic application, my team was able to develop the application as a Docker image. Since the web tool is a docker image, our team could deploy the application within AWS as a running service, making the web tool accessible to computers within Capital One's domain.

AVR's deployment method was built through AWS Elastic Container Service (ECS) Fargate. Furthermore, the application's back end is integrated with AWS Simple Queue Service (SQS) and AWS Dynamo Database. The purpose of using SQS was to bottleneck the number of incoming requests for the tool. As a 24-hour service within Capital One, the tool would see job tasks constantly throughout the day. With AWS SQS, our team was able to limit the number of requests while maintaining order on a First-in First-out basis. Additionally, the purpose of using DynamoDB was to integrate with Capital One's internal API to ensure that the users logged in possessed the right credentials to use our tool.

### 3.2 Requirements

A direct outcome of the COVID-19 pandemic was companies finding themselves moving their work schedules to a hybrid or entirely remote model. With these options, employees were given free rein to work at their leisure, which suggests some employees work outside the basic 9-5. With these policies in place for Capital One, issues within EC2 instances can be found outside of the established work hours and waiting up to 18 hours for approval leaves the company in a detrimental state for a cyber-security attack. Thus, a tool like AVR was established to ensure procedures were conducted under company policies while also improving their efficiency within the cloud.

With the premise of AVR in mind, the tool was initially created in purely a DevOps environment. DevOps is a general set of practices that many software engineers use to automate their integration between software development and deployment methods. Although the use of AVR is intuitive, many of the tool's customers found themselves having a hard time using it due to the complexity of DevOps. My team was tasked with centralizing the DevOps tool by creating a User Interface that integrated with the existing tool. For the three months, I worked as an intern, my project was to develop a web

application that integrated with Capital One's internal Application Programming Interface (API) to create an experience that was easy for the tool's customers to understand.

### 3.3 Challenges

My team faced many challenges while developing the AVR web tool. The first iteration of the web tool was created as a monolithic application. However, our team wanted to move away from this architecture, and improve it by renovating the structure into two stand-alone applications, separate applications for the front-end and back-end. By doing so, we would improve the scalability of the project for future developers and maintain better coding principles. However, due to our limited time as interns, we decided against this idea because we prioritized functionality over longevity. When placed into a pre-existing application, it takes time for developers to understand what the code does. This is especially an issue when the repository contains thousands of lines of code. Since this was one of my first experiences working with industry-level code, I found it difficult to interpret the code and developer with my style at an industry level. This was especially true when intreating with tools in AWS that I had never worked with. Although there was an initial learning curve to the application, it was a valuable experience that I can leverage in the future.

## 4. RESULTS

After my three months as an intern, our team decided to transition the development of the application to a full-time employee. As the current iteration stood at the end of the summer, our tool was capable of starting update jobs through the web tool and providing custom settings for the user to integrate. With these capabilities, our web tool directly grew the application's usage within the company to around 85% and saved employees an estimated 400,000 working hours per year.

Additionally, the web tool drastically improved the customer experience and provided an option for immediate updates. However, one main concept we failed to integrate was ensuring the credentials of the user. In the future, my intern team made it clear that element needed to be the priority for an employee to work on. This was the last step needed for the application to be released to the entire company.

## 5. CONCLUSION

Automated Vulnerability Remediation is an enterprise tool used to update operating systems within the cloud on an ad hoc basis. As an intern at Capital One, I was tasked to create a user interface that interacts with the preexisting tool. As a result, my team developed a functioning web application that saves countless hours. The web tool is important to Capital One for the efficiency it brings to an everyday scenario. Furthermore, my team's project mitigates the security risk that Capital One faces by having its infrastructure within the cloud. The web tool presents functionality that differs from the preexisting tool by bypassing credentials through its integration with Capital One's internal API and introducing new customizations to start updating jobs within the cloud. In the future, Capital One plans to develop the web application even further to have AVR solely work off a user interface instead of being simply a DevOps tool.

As for myself, I learned the importance of working within a team to solve a bigger problem. For software engineers, it is easy to work independently from others. However, cooperation and transparency are imperative when developing a bigger application. In addition, I learned a great deal about Amazon Web Services and the general cloud space. I hope that the knowledge I gained from the

summer will apply to my future software engineering jobs.

## 6. FUTURE WORK

Since AVR is owned by Capital One, I can only speculate what the company intends to do with the web application after my internship. However, after promising feedback at the end of the summer, the web tool will solve other security issues within the cloud besides updating the operating system of EC2 instances. In addition, more internal features will be implemented within the web application to serve a greater variety of customer customizations.

## 7. UVA EVALUATION

Students at UVA will benefit from the following classes in terms of working in the software industry: Algorithms, Cloud Computing, Database, and Cyber Security. When interviewing for software engineering internships, it is imperative to understand the foundational knowledge taught in Algorithms. Furthermore, Cloud Computing and Database introduce students to information that applies to everyday life as a software engineer. Finally, Cyber Security implants the importance of security within code and forces students to recognize consequences that stem from unsafe code.

## REFERENCES

Alqahtani, A., & Gull, H. (2018). Cloud Computing and Security Issues-A Review of Amazon Web Services. *Int J Appl Eng Res*, *13*(22), 16077-16084.

Gorelik, E. (2013). *Cloud computing models* (Doctoral dissertation, Massachusetts Institute of Technology).