ARCHITECTING A CLOUD-NATIVE DATA ANALYSIS APPLICATION FOR ETDS

Yinlin Chen[1], Edward A. Fox[1]
[1]*Virginia Tech, Blacksburg, VA, USA*
{ylchen, fox}@vt.edu

Abstract: In this paper, we present a Cloud-native data analysis application and its architecture. This application was developed for librarians to explore useful information from the ETDs preserved in the Virginia Tech digital repository - VTechWorks. We realized the Cloud-native concepts by architecting a serverless architecture with microservices and managed services as backend, and deployed the entire application on Amazon Web Services (AWS). We detail our architecture strategies, decisions we made, and the best practices we followed. Furthermore, we share the lessons learned and cloud benefits we have gained. We believe that our proposed approach could be adopted by other ETD systems, e.g., NDLTD, and could be of benefit to the broader community.

***Keywords*:** Cloud-native, Cloud Computing, Microservice, Serverless architecture

INTRODUCTION

At Virginia Tech, we developed a digital repository - VTechWorks to preserve and provide access to graduate students' theses and dissertations (ETDs). The system is built on top of DSpace, an open source software package. Currently, VTechWorks contains over 30,000 ETDs along with metadata. The metadata records contain information about an ETD, such as author, title, abstract, published date, etc.; they afford plenty of valuable insights for us to explore. Although DSpace provides some statistical functionalities for us to use, we find it difficult to modify existing functions or add new analytic features that fit our needs, due to its monolithic architecture. To address this issue, our approach is to develop a data analysis application that can consume the data exported from the existing system and provide customized services for librarians to perform analytics tasks on their own. Due to the limited resources on the on-premise system and IT service personnel in VT Libraries, we looked for a solution to host the application with minimal maintenance, that could dynamically adjust resource allocations based on the actual system load. Thus, we adopted the Cloud-native approach to build this data analysis application directly in the cloud environment to support our data analysis needs.

The definition of *Cloud-native* has changed over time. In 2006, a Cloud-native application was simply an application hosted in an instance from a cloud provider. After more than ten years of cloud computing evolution, the definition of Cloud-native trends toward building and running applications that exploit the advantages of cloud computing [1]. There are various techniques and patterns to design Cloud-native architecture, such as microservice, serverless, and container [2]. The Cloud Native Computing Foundation (CNCF) [3], an open source software foundation dedicated to making Cloud-native computing universal and sustainable, defines three key concepts of Cloud-native: (1) Microservices oriented, (2) Containerized, and (3) Dynamically orchestrated. Microservices oriented means the application is composed of small independent services that communicate with each other [4]. Containerized means that each part of the application is packaged in its own isolated environment. Dynamically orchestrated means the deployment and operation of the application is automatically scheduled and managed to optimize resource utilization. In summary, a Cloud-native application can react to application requirement changes quickly, add new features frequently and efficiently, scale elastically, have resiliency, and take the most advantage of cloud services.

We implemented this Cloud-native data analysis application by architecting a serverless architecture, developed microservices to communicate with managed services, and then orchestrated and deployed it in the AWS cloud environment. The application processes ETD metadata from our ETD system and provides a visualization tool for users to analyze data and create visualization results.

METHODOLOGY

We defined a data analysis pipeline containing four procedure: "Collect", "Store", "Process", and "Analyze & Visualize". We adopted the Cloud-native approach, followed by the Twelve-Factor App methodology [5], and developed this data analytics application on the AWS. We selected an AWS managed service to handle tasks for each procedure and implemented AWS Lambda functions to communicate among AWS managed services. Figure 1 gives an overview of the application architecture.

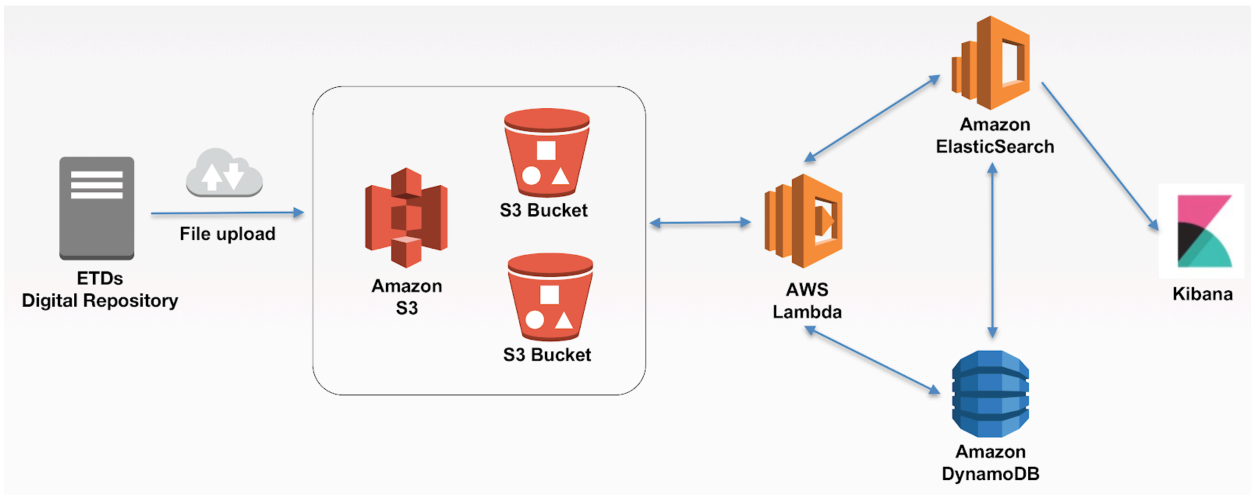

Figure 1: Architecture overview

We built a batch upload process to export ETD metadata datasets and server logs, and uploaded these files to Amazon S3 incrementally. After files are uploaded completely, Amazon S3 triggers Lambda functions to process and transform file content (ETD metadata) into the format for data ingesting into (1) Amazon DynamoDB, a NoSQL database and (2) Amazon ElasticSearch, a search engine based on Lucene. End users can explore and create analysis results using Kibana, a visualization tool. Using this Cloud-native approach, we focused only on establishing this data analysis pipeline for our needs, and delegated all the system maintenance works to AWS, such as patch update, backup, auto-scaling, etc.

FINDINGS

There are several findings from this study that demonstrate the practical usefulness of the Cloud-native approach. First, the application development process was facilitated. We dedicated our time to designing, architecting, and implementing the functionalities that will help us to bring valuable insights from our ETDs; we did not need to spend time on handling server update and backup or miscellaneous networking issues which would have contributed little. Second, we did not need to "reinvent the wheel" when there was an existing cloud service for us to use. However, knowledge to choose the right Cloud service and implement microservices to communicate with them is required, and the learning curve of each service varies. Finally, even though the loose coupling nature of Cloud-native architectures enables us to perform parallel development and deployment, and we can extend or switch different services or techniques more flexibly, as compared to with a monolithic architecture, the Cloud-native architecture is more complex, and testing such application is much more complex and difficult.

CONCLUSIONS

In this paper, we share our experiences and lessons learned by adopting the Cloud-native approach to build a data analysis application in the AWS cloud environment. We use this application to complement the functionalities we need for analysis tasks for our ETDs, instead of changing the existing monolithic ETD system. We present this Cloud-native application architecture including microservice, serverless, and AWS managed services. We list the best practices we have followed, the cloud benefits we have gained, and our findings during the entire process. We believe that our proposed approach could be adopted to other ETD systems, e.g., NDLTD, and could be of benefit to the broader community.

REFERENCES

[1] Kratzke, Nane, and Peter-Christian Quint. "Understanding cloud-native applications after 10 years of cloud computing-a systematic mapping study." Journal of Systems and Software 126 (2017): 1-16.
[2] Balalaie, Armin, Abbas Heydarnoori, and Pooyan Jamshidi. "Microservices architecture enables devops: Migration to a cloud-native architecture." IEEE Software 33, no. 3 (2016): 42-52.
[3] "Cloud Native Computing Foundation." Accessed August 10, 2018. https://www.cncf.io/.
[4] Villamizar, Mario, Oscar Garcés, Harold Castro, Mauricio Verano, Lorena Salamanca, Rubby Casallas, and Santiago Gil. "Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud." In 10th Computing Colombian Conference (10CCC), 2015 10th, pp. 583-590. IEEE, 2015.
[5] Wiggins, Adam. "The Twelve-Factor App." Accessed August 10, 2018. http://12factor.net.