# A Serverless Real-Time Data Streaming Architecture for Synchronous Online Math Competition

Yu-Che Liu
liuyuche@cityu.edu

Sam Chung
chungsam@cityu.edu

School of Technology & Computing (STC)
City University of Seattle (CityU)

## Abstract

During the synchronous online math competition, a high volume of data is continuously generated and collected from the user at every event. Those valuable data can be processed and analyzed to support many decisions in a second such as user state, cheating detection, online help desk, etc. In addition, enormous demands from different roles accessing those data, such as data analysts, data scientists, and executives, have increased recently. This paper presents a new architecture for synchronous real-time data streaming for online testing. Also, this paper identifies three challenges that need to be addressed: First, most online testing organizations rely on open-source frameworks for big data processing and streaming - Hadoop and Kafka. Second, we add the serverless architecture for synchronous real-time data streaming to an open-source learning management system, Moodle, to meet the synchronized online test requirement. Third, we discuss the benefit of the architecture in terms of high availability, cost, and technologies.

**Keywords:** Serverless, Real-time Data Streaming Architecture, Synchronous Online Math Competition, Open-Source Big Data Processing Framework, Open-Source Data Streaming Framework

### 1. INTRODUCTION

During the recent COVID-19 situation, education and industry were forced to go online. As a result, education and industry need to perform online tests for students and provide an online business platform for their employees (Wahid et al., 2015). However, building a synchronized system takes many resources for the company to hire experts and build a highly scalable infrastructure. Moreover, small to mid-size companies have a challenge with real-time data processing.

Processing real-time data is a complex job for a large-scale distribution system. Therefore, the market demand for real-time streaming has been highly increased (Liu et al., 2014). More companies are challenged to provide real-time processing for extensive data analysis due to the expensive cost, maintenance difficulty, and low performance.

The synchronous online system brings a huge benefit to companies. However, building and maintaining the system can cost some resources and energy (Kontaxakis et al., 2021). Small-size companies do not have enough resources to build and maintain this huge real-time data processing system.

The main difficulty of building a synchronized system is maintaining the users' state on the

 

server. To maintain the users' state, we need to maintain the communication protocol between producers and consumers to stream users' data to the destination (Xu et al., 2021). The options to build the communication protocols are HTTP/s and WebSocket. However, building the communication protocols can be a massive project and sometimes take several months.

Another difficulty in building a synchronized system is processing the data in real-time. Processing the data in near real-time requires several computing resources running in parallel to support instant data processing. Traditionally, companies must purchase many physical servers and set up batch processing software with stream processing capability, including Hadoop, Apache Spark, Flink, etc. However, the system was built on the on-premise server using Kafka and Flink services (Thein, 2014).

In this paper, we challenge how to maintain users' data in the synchronous online exam with high availability (Chaffai et al., 2017). Keeping users' data synchronous can bring a lot of traffic pressure on the server and require more engineering work to provide availability and reliability. Thus, we propose an AWS Serverless cloud computing solution, including AWS Kinesis, to help companies build fast, easy-to-maintain, high-performance real-time data processing platforms.

This paper proposes a serverless real-time data streaming architecture and applies the architecture to synchronous online math competitions:

1. Serverless architecture reduces development workload and maintenance costs.
2. Serverless architecture optimizes and maintains the user state data.
3. Serverless architecture provides a solution to process data in near real-time.

First, most online testing organizations rely on open-source frameworks for big data processing and data streaming - Hadoop and Kafka. Second, we add the serverless architecture for synchronous real-time data streaming to an open-source learning management system, Moodle (Chaffai et al., 2017).

There is an open-source streaming software to help us build our synchronized application without reinventing the wheels, including Kafka, ActiveMQ, and RabbitMQ (Dobbelaere et al., 2017).

An open-source streaming software can build real-time data processing architecture within a limited time. However, cloud computing can help companies manage and reduce building system workloads and budgets. For instance, Amazon Kinesis is an AWS fully managed service that engineers can focus on ingesting and processing streaming data without managing any infrastructure. Amazon Kinesis utilizes massive computing resources from AWS to create a real-time data processing platform (Gannon et al., 2017).

AWS serverless framework provides a solution to let developers build serverless applications on AWS. The solution can integrate multiple AWS services to help the developers focus on building an application without dealing with all the infrastructure. Compared with the other real-time data processing frameworks, including Hadoop, MapReduce, Spark, and Kafka, using AWS serverless framework can provide lower upfront operational cost and low maintenance (Nguyen et al., 2021)

## 2. BACKGROUND

This section will provide some background knowledge of serverless and how serverless applies to data processing.

Nowadays, mobile users are growing exponentially, which generates a lot of data from user events. The more data stored in the database, the more possibilities to find the relationship between data. Relationships can be used to build a machine learning model to solve some issues. Storing and processing massive datasets requires tons of computing resources and storage space. To support its business needs, it costs many resources to purchase equipment and human resources upfront. Not every company is capable of handling this amount of cost. The emergence of Cloud computing opens another possibility to achieve the same features but less cost and management (Nicoletti, 2016).

The emergence of cloud computing allows developers to focus on building business logic instead of setting and maintaining the server configuration (Kemal et al., 2009). The Cloud provider takes care of physical hardware settings and maintenance tasks. As a result, the developers can spin up several servers based on business requirements within several clicks on the Cloud provider dashboard without going through all the processes of setting up a physical machine.

On top of the cloud computing concept, Serverless architecture provides optimized solutions for automatically scaling up and building client and server-side applications using the same software development kit.The serverless tool brings the companies two primary benefits: cost and scalability. When the product goes live, serverless tools charge based on the volume of the service usage. Therefore, it brings substantial cost savings compared to building on-premises infrastructure (Gannon et al., 2017).

The demand for big data analysis has become higher than before. Users generate different types of data, such as logs, application status, and transactions, and send them to data infrastructures, which transfer raw data into information stored in in-memory or persistency databases. Then, data consumers retrieve the data from the database, continuously processing it with complex use cases such as machine learning models, reports, and feature engineering.

In education, there are several learning management tools such as Google Classroom, Blackboard Learning Management System, Moodle, Etc. However, the current learning management systems lack real-time data processing support, which cannot provide synchronization of online exams. The synchronization feature can provide a better user experience closer to the on-site exam.

## 3. RELATED WORK

Processing real-time data streaming has become popular these days. Data plays a vital role in streaming. In the past, people tried to figure out how to manage the data by executing SQL commands in the database. When high-speed internet comes, it provides higher bandwidth for more data to transmit.

Many research approaches analyze and process the data in real time. For example, Akanbi (2020) monitors data generate from different IoT devices and proposes a four-stage architecture to collect, stream, analytics, and store the environmental sensor data. However, their approach does not give real-time feedback based on the result of the analysis. Furthermore, all four stages are not highly cohesive; some use cloud computing, and others use open-source software on on-premise servers. This combination greatly allows the developer to choose technologies, but it increases the response time between layers.

Javed et al. (2017) states that throughput plays an important role when the team process real-time data. High throughput can have lower latency.

Table 1 compares two research studies that conduct real-time data processing platforms. Both researchers collect the data in real time. However, they do not synchronize data with front-end implementations and use cloud computing completely.

| Criteria | Adeyinka, 2018 | Javed et al., 2017 |
|---|---|---|
| Architecture | IOT, Kafka | Flink and Kafka |
| Collect real-time data | Yes | Yes |
| Synchronize data with front-end | No | No |
| Do real-time analysis | No | Yes |
| Performance | N/A | 160 records/ms |
| Serverless | No | No |
| Cost | N/A | High |

**Table 1: A Summary of Related Work**

## 4. APPROACH

The new architecture contains four parts: collecting users' data from the online test system, sending real-time data into the data pipeline, and processing and analyzing data into information stored in the database for the online testing system to synchronize data.

Figure 1 shows the serverless architecture to process, analyze, and store real-time data. To build this architecture, we build a customized online test plugin on Moodle learning management system to access users' test data and send those data back to AWS Kinesis firehose processing in real-time. After AWS Kinesis firehose receives data, data will enter a customized data pipeline to perform Amazon Kinesis Data Analytics SQL statement and query the result based on customizing data schema sending to AWS in-memory database, Amazon ElastiCache. In addition, the Moodle plugin can

retrieve users' previous status and record when users disconnect from the synchronizing exam.

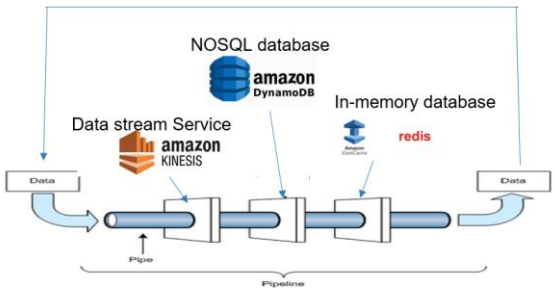


**Figure 1: Serverless Architecture**

Moodle, a learning management system, is an open-source learning tool, which provides e-learning, and flipped classroom features in various industries. We built a customized Moodle learning management system plugin that provides an online synchronization exam.

## 5. DATA COLLECTION

Based on the architecture in this paper, the data collection starts from the front-end interface, which is Moodle learning management system. When the user answers each question, the program will send the answer data to the server immediately instead of waiting for the user to answer all the questions.

A group of data is being transferred into AWS Kinesis. The different parameter is used for AWS Kinesis to compare the traditional architecture of data steaming.

Figure 2 shows how to collect synchronized user data created from the user behavior. Data is structured in a key-value pair format, allowing AWS Kinesis to retrieve and analyze the data from applications quickly.

Figure 3 shows when AWS Kinesis receives the data and will run SQL commands to analyze the user's status in real time.

Data latency is critical for the synchronized exam; this paper compares the data latency performance under different architectures. The start timestamp is recorded when the data is being sent out on the client-side, and the stopped timestamp is generated on the server-side to calculate the data latency between the front and backend.

```php
//sample data
$name = "online-test-student-data";
$content = '{"user":"spencer", "status":"000001"}';

try {
    $result = $firehoseClient->putRecord([
        'DeliveryStreamName' => $name,
        'Record' => [
            'Data' => $content,
        ],
    ]);
    var_dump($result);
} catch (AwsException $e) {
    echo $e->getMessage();
    echo "\n";
}
```

**Figure 2: Synchronized User's Status in PHP**

```
CREATE OR REPLACE STREAM "STUDENT_STATUS" ("student_id" INTEGER, "status" INTEGER);

CREATE OR REPLACE PUMP "STREAM_PUMP" as
INSERT INTO "STUDENT_STATUS"
SELECT STREAM "student_id", sum("status")
FROM SOURCE_SQL_STREAM_001
GROUP BY "student_id", STEP ("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '1' MINUTE)
ORDER BY STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '1' MINUTE), avg("status");
```

**Figure 3: Real Time Analysis in SQL**

In order to lower data latency, our approach is to collect the information from the interface like Figure 4 and restructure the data into three columns, including status, student_id, and timestamp, as shown in Table 5.

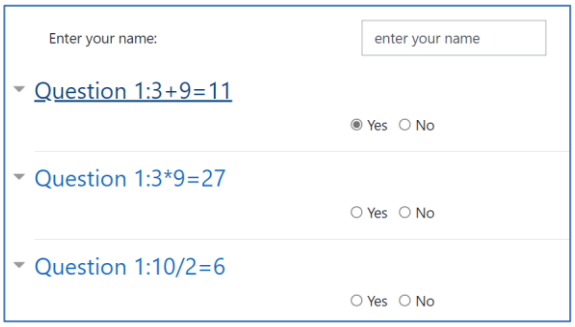


**Figure 4: User's Data on Moodle System**

| Status (VARCHAR) | student_id (INTEGER) | TIMESTAMP |
|---|---|---|
| s1120 | 3 | 1.6312066665 |
| s1100 | 701 | 1.6312909355 |
| s0000 | 977 | 1.6312718455 |

**Figure 5: Data Schema**

## 6. Data Analysis

This section presents the data collection from AWS Kinesis and Moodle system. The data is generated from AWS Kinesis API with a message buffer size of 1 MB, and each Buffer interval is 60 seconds. The idea for deciding on the buffer size is based on the size of all the questions in the online exam. We evaluate the amount of data generated from the student; each tested student has around 50 questions, and each question takes 2 bytes, which equals 0.000002MB. Therefore, one student can send 0.0001MB of data to the AWS Kinesis server for an online test. Based on the architecture purpose of this paper, Figure 6 shows that our approach can handle around 500,000 students taking the online exam simultaneously. The X-axis is the time period, and the Y-axis is the number of records our architecture can support.

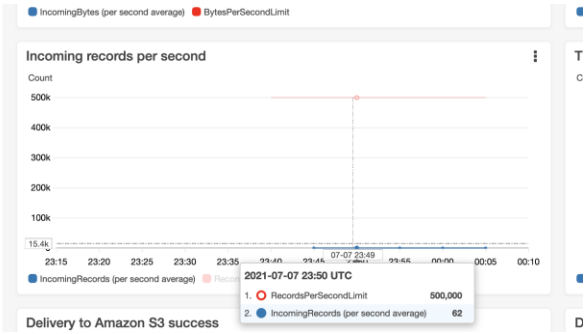


**Figure 6: Data digest in a second.**

## 7. FINDINGS

### Synchronized online test system

Our solution provides a synchronization online test system on Moodle education system. Teacher can implement a synchronized test system without struggling with technical issues and reduce the time to develop software. Students can get a similar testing experience using their devices anywhere. They do not need to worry about connection and internet speed issues, and all the answers will be streamed in real-time to the server and synchronized with the online test system.

### Cloud adoption

The traditional method to build the real-time data processing platform is made on the on-premises server. However, creating an on-premises server is a massive cost for the companies in the early stage, such as purchasing a physical server and hiring a real-time management expert. By using cloud solutions, it can have a more flexible budget and reduce the software development time.

### High availability

Cloud computing provides high availability to guarantee real-time data platform availability. In addition, all the data will be synchronized in three availability zones and one cold storage in the AWS S3 bucket to increase data durability.

Our approach utilizes the AWS cloud computing environment to build a real-time data processing platform. Compared to previous work, our approach provides real-time analysis, high performance, and lower cost using serverless architecture, as shown in Table 2 below.

| Criteria | Adeyinka, 2018 | Javed et al., 2017 | Our Approach |
|---|---|---|---|
| Architecture | IOT, Kafka | Flink, Kafka | Moodle, Amazon Kinesis, Amazon DynamoDB |
| Collect real-time data | Yes | Yes | Yes |
| Synchronize data with front-end | No | No | Yes |
| Do real-time analysis | No | Yes | Yes |
| Performance | N/A | 160 records/ ms | 500 records/ ms |
| Serverless | No | No | Yes |
| Cost | N/A | High | Low |

**Table 2: A Comparison of Previous Work and Our Approach**

## 8. CONCLUSIONS

This paper presents the new architecture for real-time data streaming based on the existing education system and identifies three challenges that need to be addressed in our architecture. First, most organizations or education areas rely on open-source learning systems. Second, on top of the synchronized learning management system, we add customized solutions for improvement and customization to give real-time ability to meet the synchronized online test requirement.

Building and maintaining real-time features for online testing systems consume more resources

for small to medium-sized companies. Adopting cloud computing saved many engineer costs and reduced the software development time. AWS cloud computing can achieve a high volume of real-time data processing and various data storage options. Connecting Moodle system with real-time data processing provides the online testing event with high availability.

The system can be improved to make the system become more cost-effectively practical and better performance. First, real-time data streaming processing brings massive data from the application. It will be a considerable cost for the companies or organizations to store the data in their system for a long time. AWS provides different layers of storage options. Data can be moved between cold and hot storage. Second, a real-time data platform will be deployed to serve multiple users from different locations. Different regions may want to track their data or query based on their time zone. The biggest challenge is to maintain the data in proper order and location.

## 9. REFERENCE

Akanbi, A. (2020, November). ESTemd: A Distributed Processing Framework for Environmental Monitoring based on Apache Kafka Streaming Engine. In 2020 the 4th International Conference on Big Data Research (ICBDR'20) (pp. 18-25).

Chaffai, A., Hassouni, L., & Anoun, H. (2017, March). E-learning real time analysis using large scale infrastructure. In Proceedings of the 2nd international conference on big data, cloud and applications (pp. 1-6).

Dobbelaere, P., & Esmaili, K. S. (2017, June). Kafka versus RabbitMQ: A comparative study of two industry reference publish/subscribe implementations: Industry Paper. In Proceedings of the 11th ACM international conference on distributed and event-based systems (pp. 227-238).

Gannon, D., Barga, R., & Sundaresan, N. (2017). Cloud-native applications. IEEE Cloud Computing, 4(5), 16-21.

Javed, M. H., Lu, X., & Panda, D. K. (2017, December). Characterization of big data stream processing pipeline: a case study using Flink and Kafka. In Proceedings of the Fourth IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (pp. 1-10).

Delic, K. A., & Walker, M. A. (2008). Emergence of the academic computing clouds. *Ubiquity*, *2008* (August), 1-1Kontaxakis, A., Deligiannakis, A., Arndt, H., Burkard, S., Kettner, C. P., Pelikan, E., & Noack, K. (2021, June). Real-time processing of geo-distributed financial data. In Proceedings of the 15th ACM International Conference on Distributed and Event-based Systems (pp. 190-191).

Liu, X., Iftikhar, N., & Xie, X. (2014, July). Survey of real-time processing systems for big data. In Proceedings of the 18th International Database Engineering & Applications Symposium (pp. 356-361).

Nguyen, K., & Chung, S. (2021). Low Maintenance, Low Cost, Highly Secure, and Highly Manageable Serverless Solutions for Software Reverse Engineering. In Proceedings of the Conference on Information Systems Applied Research ISSN (Vol. 2167, p. 1508).

Nicoletti, B. (2016, March). Cloud computing and procurement. In Proceedings of the International Conference on Internet of things and Cloud Computing (pp. 1-7).

Thein, K. M. M. (2014). Apache kafka: Next generation distributed messaging system. International Journal of Scientific Engineering and Technology Research, 3(47), 9478-9483.

Wahid, A., Sengoku, Y., & Mambo, M. (2015, January). Toward constructing a secure online examination system. In Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication (pp. 1-8).

Xu, J., Yin, J., Zhu, H., & Xiao, L. (2021, May). Modeling and verifying producer-consumer communication in Kafka using CSP. In 7th Conference on the Engineering of Computer Based Systems (pp. 1-10).