UNIVERSITI SAINS MALAYSIA

CMT321: Management & Engineering of Databases

Semester I of Academic Session 2022/2023

Assignment 1

For the attention of

Dr Suzi Iryanti Binti Fadilah

Name	Matric Number	USM Email Address
Iris Yan Ning	153410	irisyan@student.usm.my

Submission Date: 19 December 2022

<u>Table of Content</u>

Part 1

1.1 Introduction to Stream Processing

Stream processing is a form of data processing whereby the input data is continuously subjected to computations, operations, analysis or other forms of data transformation as it is being consumed by the system. The word "stream" in the term "stream processing" means the **stream of data** that enters the system. Stream processing enables the operations on data to be carried out at a rate that is almost **real-time**, made possible by the advancement of technology in terms of processing capacity and speed.

In stream processing, the system listens to the events and data emitted by a set of data producers or its environment. In social networks, the data producers comprise the users interacting with the social media application from their devices such as their laptops or mobile phones. For example, whenever a user writes a social media post caption, comments on another user's post or shares a photo, these are data that are being continuously processed according to the predefined application logic, queries and analytics in the system. The stream of generation of data to the data processing up until the arrival of data at its destination makes up the stream processing pipeline.

Twitter is a social media platform whereby users can post short sentences called "tweets". These sentences can be formed from any thought or observation, or whatever the user wants to talk about. On Twitter, any action that is taken such as posting a tweet or even a scroll or a click event, will be logged. This results in a **continuous stream of data** that is constantly fed to Twitter's backend system for further analytics. With the massive amount of data being generated at every second, stream processing is used to capture and analyse the data at real-time.

1.2 Importance of Stream Processing

With the increasingly boundless network of networks that connects the world today, the amount of data that is being processed daily is breaking 2.5 quintillion bytes of data and growing. Every single person that is connected to the network is a part of the stream processing ecosystem, and social media is a vast medium that consumes and ingests the streams of data produced by the users. The systems must keep up with the increasing volumes of data to process them at near real-time to fulfill several business functions.

However, with the gigantic amount of data being generated and ingested at every second, the complexity of processing such varied data streams to gain value from the data poses a challenge. There is a need to analyse the data in real-time to **detect anomalies, ensure instant response, perform queries** such as filtering or aggregating data and more. Hence, stream processing is the solution for systems that need to deal with massive amounts of continuously flowing data with **minimal latency** and **minimal overhead** while ensuring r**eal-time processing of high-volume data**.

In social media networking, billions of data from messages, posts and threads are being exchanged over the network at any second, making stream processing the ideal data processing mechanism, including Twitter. There are also many other fields that utilize stream processing to maximize the benefits of real-time response, including algorithmic trading, network monitoring, fraud detection, supply chain management, medical systems, Internet of Things (IoT) edge analytics and more.

1.3 Stream Processing in Twitter

Twitter utilizes the data collected from every user with thorough data processing to generate value for the users. For example, Twitter processes every individual's data to **make inferences about a person's demographics and interests**. With sufficient data, Twitter is able to deliver personalized experiences by showing **relevant content** and advertisements to the user with real-time personalization, marketing, and advertising.

In earlier days, Twitter used a batch processing system to handle their data which resulted in high latency, meaning the data cannot be accessed at real-time and instead had to be refreshed in predefined intervals to receive new data. The data would then be queried or computed over as needed. Although the batch processing system is beneficial for certain use cases, it is not a scalable option for a social media platform whereby every single engagement is instant and real-time data delivery is crucial to identify user behavior, patterns and trends.

In the early adoption of stream processing, Twitter used the **Apache Storm** software to analyse real-time data for content filtering, geolocalisation and classification. As the user community grew, Apache Storm was no longer able to withstand the increasing load, hence Twitter brought in a new replacement, the **Heron topology**, which provided relatively higher throughput and lower latency.

However, with the number of users increasing exponentially by day, subsequently the massive load of data caused Heron to not be able to keep up with the pace of the data throughput, which resulted in data loss, data inaccuracy and high latency. The quick but unsustainable fix was to restart the Heron containers, however this caused event losses which led to inaccuracy in the metrics analysis.

Hence, a new architecture is designed, which is the **Kappa architecture** which runs on Twitter Data Center and Google Cloud platform. Data preprocessing and relay event processing are performed within Twitter Data Center services. From these services, a Streaming Event aggregator collects the preprocessed events and passes them to Apache Kafka, a distributed message streaming platform. From the queue, the Streaming Event Processor applies computations, formatting or transformations to the data. The data is then streamed into Google Cloud services, which are Google BigQuery and Google Cloud Storage. The hybrid architecture enables the reduction of latency while providing high stability and accuracy when processing billions of data in real time.

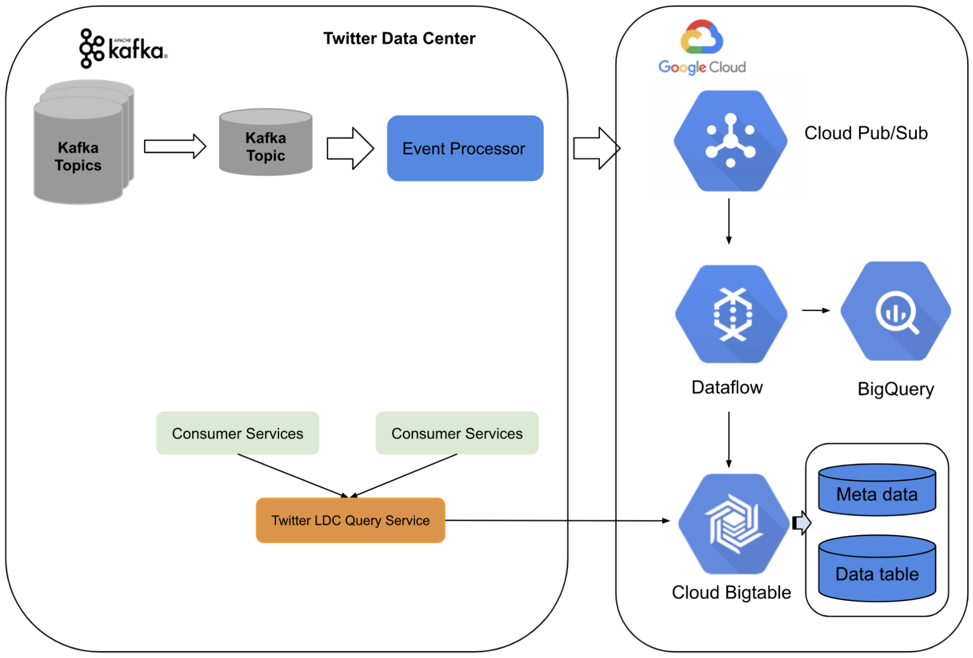

Image source: (Lu., (2021) Twitter New Architecture on Kafka and Dataflow

1.4 Comparison with Instagram

Another popular social media platform that deals with colossal amounts of data generated by 800 million monthly active users is **Instagram**. Instagram is a photo and video sharing platform with Django Python running its distributed backend services to serve millions of queries per second.

Instagram uses stream processing as a part of its data processing mechanisms. This is utilized in Instagram's **trending** module, whereby the content of trending hashtags and places are identified, ranked and displayed to users based on their relevance. The stream processing application comprises four linearly connected components which ingest and analyse streams of data from user engagement. The nodes consist of the pre-processor, parser, scorer and ranker. The data streams pass through each of them sequentially.

When the data passes through the preprocessor, metadata of the triggering event such as the user and the event content is attached to the original media as preparation for the filtering stage in the following stages. Then, the parser detects the hashtags used with the media and filters it based on a set of predefined qualities. Next, the scorer keeps track of the hashtag counter and computes a scoring function for the hashtag every five minutes. Finally, the ranker aggregates the hashtags and ranks them based on their scores to produce the top ranks list.

Comparison	**Twitter**	**Instagram**
Platforms	Twitter Data Center services, Google Cloud	Meta server
Backend components	Apache Kafka, Google BigCloud, Google Cloud Storage	Cassandra, PostgreSQL, Memcache, Redis
Architecture	Hybrid	On-premises
Flow	Publish/subscribe	Linear

1.5 Summary

The topic of stream processing in Twitter has helped me to understand the different types of data processing for different use cases and how big data generated from social media uses this mechanism of data processing to maximise the value of the data in order to generate insights for the social media platform to provide real-time personalisation of content to the users.

By studying the history of how Twitter is evolved its data architecture from batch processing to stream processing, and how Twitter is constantly improving the database management system used such as upgrading from Apache Storm to Heron, then from Heron to Google Cloud, it is crucial to adopt infrastructure with more advanced technology to ensure that the service is more reliable, resilient, accurate and scalable when withstanding increasing loads.

Platforms such as database management systems must be constantly evaluated and upgraded when necessary to quickly adapt to its ever-changing environment with the appropriate technologies in order to stay relevant in the business world.

<u>Part 2</u>

2.1 Introduction

Denial-of-service (DoS) attack is a form of cyber attack which is exactly as the name describes - **denying the usage of services to its users** by causing **disturbance to the availability of a system**. This is done by using another machine to overwhelm the targeted system with spam requests until the system is flooded with traffic, causing legitimate requests to be unable to be processed. Imagine a congested highway caused by an overflow of fake cars on the road, preventing real cars from reaching their destination. A more advanced version of DoS is distributed denial-of-service (DDOS) whereby instead of a single machine attacking the system, the attacks are launched from multiple different machines, making it more difficult to trace the attackers.

There are several different types of DoS attacks. One of them is the **buffer overflow** attacks, whereby the attacker forces a memory buffer overflow which causes the victim system to use up all its memory and CPU power, resulting in system crashes or extremely high latency. There is also the **ICMP flood** (also known as the Smurf Attack), whereby spoofed packets are flooded to all machines connected to a targeted network which was detected to be vulnerable. The **SYN flood** is another type of DoS attack which continuously sends handshake requests to a server but never completes the handshake, causing all open ports to be oversaturated with unfinished requests and blocking legitimate requests.

DoS attacks also threaten databases of systems, as they can be targeted to directly attack data centers, data infrastructure and the network that provides availability to the data users to prevent users from accessing the data.

2.2 Existing DoS Attack Prevention Techniques for Database

In the report conducted by A10 Networks in 2021, there is a continuous rise in the frequency, complexity and criticality of DoS and DDoS attacks being launched daily. On the bright side, there are many techniques that have been proven effective against such attacks if implemented correctly.

An existing framework used to prevent DoS attacks is **DDoS-Shield**, a framework used to classify attacks and calculate the **suspicion measure** of attacks based on the calculation of the **deviation from the behavior of legitimate traffic**. As DoS attacks are getting more advanced, they can be cleverly disguised as genuine requests which can bypass any monitoring tools or firewalls. DDoS-Shield works by assigning a session-based suspicion score to the incoming requests benchmarked against genuine requests to identify the presence of DoS attacks.

Correct database configuration is also a crucial step to prevent DoS attacks by reducing the attack surface, such as identifying and **eliminating unnecessary user accounts, protocols, services and database management system features**. This is because unnecessary elements are usually overlooked by database administrators, causing them to become vulnerable points of attack, especially activated unused features which may have undiscovered weaknesses.

Resource limits configuration should be implemented to limit the resources for every user. The access to resources can be based on the n**umber of queries, query timeouts, resource throttling** and more. Although this cannot completely prevent DoS attacks, it can slow down the attackers and minimise the consequences of the attacks as the response plan to dealing with the attack is carried out, such as identifying and shutting down the attackers before the attack gets worse.

Consistent **patching** of database vulnerabilities is another crucial step to prevent DoS attacks on the database. There are many vulnerabilities potentially present in database code, such as **buffer overflows and memory leaks** which can be points of entry of DoS attacks. It is also important for the **vendor** of the database management system to be responsive to security issues that arise from design flaws in database concurrency or database implementation, and the patches released from the fixes should be applied immediately by database administrators.

Database firewalls are special firewalls used to prevent the execution of **malicious queries**. Malicious queries can also be a form of DoS attack, whereby extremely **complex operations** are used in queries that result in thousands of permutations being generated to deliberately slow down the system. The database firewalls filter all database requests to restrict the structure of the query so that overly complex queries are blocked. This is also advantageous against DoS attacks in the form of buffer overflow, reducing the avenues of exploitation by attackers.

2.3 Comparison of DoS Prevention Techniques - On-Premises VS Cloud

There are three parts to dealing with DoS attacks: **preparation, prevention** and **response**. There should always be the assumption that the system database will face DoS attack at any given moment. Hence, there should be a solid plan to prepare for the attack, including **recovery plans** and **business continuity plans** which prioritize parts of the system or processes to restore based on their business value and tolerable downtime. The recovery plan should be spearheaded by the incident response team of the organisation and proper protocols to execute the plans to ensure that the consequences of the attack are minimised.

2.3.1 Prevention of DoS attacks for on-premises database

It is crucial to ensure that the **database administrator** is a part of the incident response team, or has direct communication with the team so that the administrator can take action when any suspicion of an incoming DoS attack is detected. The database administrator must be well-versed with DoS protection steps such as configuring the rate limit of the database management system and performing the appropriate blacklisting or whitelisting actions.

Database audits and vulnerability assessments should be conducted quarterly for on-premises database systems as there is always the possibility that new vulnerabilities open doors for attackers. Audits and penetration tests should be performed to dive deep into the weaknesses of the database, including checking against common database vulnerabilities such as privilege escalation, configuration management and unpatched databases, detecting bad queries and examining query metadata. These steps are crucial in order to understand the mechanism of potential attacks and form a strong defense by reverse-engineering the attacks.

The infrastructure, environment and foundations that support the workings of the database such as the **operating system** should also be monitored, such as the installation of firewalls, anti-virus software and ensuring the latest updates to patch any vulnerabilities of the operating system.

2.3.2 Prevention of DoS attacks for cloud database

Cloud databases might be safer compared to on-premises databases as there is a wide array of cloud services available to tackle DoS attacks, however it is still crucial to know how to correctly implement them. The organisation must work closely with the cloud service providers to ensure that **service-level agreements** regarding the security of the database include defence provisions against DoS attacks. While cloud providers ensure the security of the cloud, the organisation should ensure the security in the cloud.

Firstly, the scale of **network infrastructure** must be planned out clearly. As DoS attacks are designed to attack systems by flooding the traffic with massive volumes of request in attempts to consume all the system resources such as the bandwidth and computing power, it is crucial that the system can **scale** quickly, paired with **load balancers** to divert loads between paths to ensure that there is not a single route that is jam-packed to prevent a system crash. The system should also use **Content Delivery Networks (CDN)** whereby the database resources are stored in distributed servers close to users. In the case of a DoS attack, the CDN will simply redistribute the overflowing traffic to ensure that they do not directly hit the origin server.

Next, the **network perimeter** must be minimised so that the building of protections can be focused on a smaller area with a layered approach by constructing multiple levels of security strategies. Cloud services provide the **Access Control Lists (ACL) and security groups** functionalities whereby the database administrator manages the access to the database in order to block any requests coming from unexpected sources. Cloud platforms also have **monitoring systems** that can quickly identify anomalies in traffic and send alerts to the database administrator when suspicious levels of traffic are detected.

2.4 Conclusion

In conclusion, DoS attacks are still a frequent occurrence, and although attacks on databases are less common than attacks on networks, it is still a matter of concern as many organisations only address the prevention against network DoS attacks but neglect database DoS attacks as the availability of database is the foundation on which most of the system's services operate.

From this topic, I have learnt that although there is no 100% guarantee to prevent DoS attacks on databases, several counter-measures can be taken which requires a thorough understanding of the database management system used and ensuring the correct configuration of the database to minimise potential exploits. The only way to prevent any attack is by disconnecting the database from any network, which would render it useless. Hence, to keep up with the advancement of attackers' technology, constant database management system software patches must be applied besides correct database configuration to reduce any vulnerabilities.

3.0 References

1. Awan, M. J., Farooq, U., Babar, H. M. A., Yasin, A., Nobanee, H., Hussain, M., Hakeem, O., & Zain, A. M. (2021). Real-Time DDoS Attack Detection System Using Big Data Approach. *Sustainability*, *13*(19), 10743. https://doi.org/10.3390/su131910743
2. Bridgwater, A. (2016, June 16). *Twitter Open Sources Heron -- Data Streaming For Dummies*. Forbes. ttps://www.forbes.com/sites/adrianbridgwater/2016/06/16/twitter-open-sources-heron-data-streaming-for-dummies/?sh=788078b42d14
3. *Cloud DDoS Protection: How to Prevent and Mitigate DDoS*. (n.d.). https://www.stormit.cloud/blog/cloud-ddos-protection-how-to-mitigate-all-risks/
4. Engineering, I. (2018a, June 10). *Instagration Pt. 2: Scaling our infrastructure to multiple data centers*. Medium. https://instagram-engineering.com/instagration-pt-2-scaling-our-infrastructure-to-multiple-data-centers-5745cbad7834
5. Engineering, I. (2018b, June 17). *Trending on Instagram - Instagram Engineering*. Medium. https://instagram-engineering.com/trending-on-instagram-b749450e6d93
6. Hazelcast. (2019, September 25). *What Is Stream Processing? A Layman's Overview*. https://hazelcast.com/glossary/stream-processing/
7. Ortega, D. (2018, November 28). *Why Is Stream Processing Important to Your Business?* Hazelcast. https://hazelcast.com/blog/what-is-stream-processing-and-why-is-it-important-to-your-business/
8. Velimirovic, A. (2022, August 15). *How to Prevent DDoS Attacks: 7 Tried-and-Tested Methods*. phoenixNAP Blog. https://phoenixnap.com/blog/prevent-ddos-attacks