1. Self introduction

I’m Ian, from CSL. Chinese name is Yuan Hailong. Now responsible for research on big data. Today on behalf of my team, share knowledge and problems we encountered in learning spark. Thanks for their support for me. Now can we start?

1. Outline

Ok, look at the first slide, the outline of this sharing session. Next, I will follow the 4 parts to introduce.

1. The first is Technology Introduction we used. Mainly include Flume, Kafka and spark. Spark is the focus of this sharing.
2. The second is Project architecture, the most important part. Mainly introduces the project’s architecture and technical choice.
3. After a brief introduction above, I will show a few examples.
4. At last, summary this sharing session.
5. Technology introduction

Next, let me introduce the technology we used in this project:

First is flume. We use flume to collect logs. It is robust and fault tolerant with tunable ['tju:nəbəl] reliability mechanism ['mekənɪzəm]. According to the official statement, flume is a distributed [dɪ'strɪbju:tɪd], reliable, and available service for efficiently collecting , aggregating, and moving large amounts of log data. It has a simple and flexible architecture based on streaming data flows. As flume is not main content of this session, I will not introduce it too much.

Next is kafka. In this project, kafka is used to transfer and cache data. It can guarantee high availability of this project.

1. It lets you publish and subscribe to streams of records. In this respect, it is similar to a message queue or enterprise messaging system.
2. It lets you store streams of records in a fault-tolerant way.
3. It lets you process streams of records as they occur.

At the end of this part, introduce spark in detail. As said earlier, you know, spark is the key of this project, which use to process data. What is spark? First look at the official explanation   
[ˌekspləˈneɪʃn]: Apache spark is a fast and general engine for large-scale data processing. It provides high-level APIs in java, Scala, Python and R, and an optimized engine that supports general execution graphs. In short, spark is a data processing engine. It has 3 features:

Speed: We all know spark is fast. So how fast? Run programs up to 100 times faster than Hadoop MapReduce in memory, or 10 times faster on disk.

Generality:

It combines [kəmˈbaɪn] SQL, streaming, and complex [kəmˈpleks] analytics [ˌænə'lɪtɪks]

Spark powers a stack of libraries including SQL and DataFrames, MLlib for machine learning, GraphX, and Spark Steaming. We can combine these libraries seamlessly in the same application.

Runs Everywhere

Spark runs on Hadoop, Mesos, standalone, or in the cloud. It can access diverse data sources including HDFS, HBase, and S3.

Official said spark is Ease of use, but after a period of learning we found that a deeper dive on how to use spark is not so easy.

This slide shows spark framework.

We can clearly see from the figure: The top layer is a stack of libraries spark powered, which based on spark core.

Spark SQL : include DataFrame and DataSet , we can use it for querying data via SQL.

Spark Streaming: for real-time processing of live streams of data

MLlib: a library for machine learning providing algorithms for doing classification, regression [rɪˈɡrɛʃən] and so on.

Graphx: a library for manipulating [məˈnɪpjəˌlet] graphs and performing graph-parallel computations.

The middle layer is core of spark.

Spark core: it is defined by RDD(Resilient [rɪˈzɪljənt] Distributed Datasets), which is short name of Resilient Ditrbuted Datasets. It is the main programming abstraction that represent [ˌrɛprɪˈzɛnt] a collection of items distributed across many compute nodes that can be manipulated[məˈnɪpjuleɪt] in parallel.

The bottom layer is spark cluster.

Spark itself doesn’t manage the cluster, but it supports 3 cluster managers:

Standalone: a simple cluster manager included in Spark itself called the Standalone Scheduler.

Hadoop YARN: What we are using is YARN. It currently very popular.

Mesos: we have not used it yet.

Spark provides this high level programming framework that allows programmers to focus on the logic and not the plumbing[ˈplʌmɪŋ] of distributing programming, that is, the steps to be done without worrying of coordinating tasks, networking and so on.

Introducing spark has to mention Scala, because spark is implemented in Scala. If we want a deeper dive on how to use spark, learning scala is very necessary. Scala is an emerging JVM language that offers strong support for Functional Programming. So the curve of scala learning is high compare to java. In this project we use scala and sbt intead of java and maven. In the demo stage, I will show you a word count example written in scala. You can feel scala code.

1. Project Architecture

OK. Now look at the architecture of the project. We can see 2 lines from data producer to business. The first line is ELK.

ELK: I remember someone introduced it before. So I don’t want to introduce it again. You know, ELK is a set of excellent log analysis tools. It includes logstash, elasticsearch and kibana. It can quickly analyze log data through configuration. Now think about it:

If we grab pifd or fii from the logs, and we want to know which acount the pfid represents or which security fii represents. What should we do? Usually we will join it with other tables from database. But how to integrate data with ELK? How? ELK is hard to do that. This is why we construct the line below.

You can see that it can be divided into 5 layers: data collection, data transmission, data processing , data storage and data presentation. Flume collects logs, then it sends data to Kafka cluster. Kafka , as a cache, receives data from flume. When the amount of data is particularly large, Kafka will not lose data. This also ensures the high availability as I said before. Spark , as a data processing engine, can process and reorganize data quickly. It also can integrate data from multiple data sources. In this figure, we can see spark can acess sqlserver, mongoDB, Elasticsearch and so on. In this project, spark pulls log data from kafka and process. Then spark save it to elasticsearch for subsequent search analysis.

Like the question I just raised, spark gets pfid or fii from kafka cluster. It can join it with relevant table from sqlserver. At last it save the result to Elasticsearch. The line below just makes up for the short of ELK. In order to verify the flow is feasible [ˈfizəbəl], we, combined with the needs of work, made an example: blueOptimas. I will introduce to you later.

1. Demo

Now, let’s take a look a at word count example. It will count the number of occurrences [əˈkɜ:rəns] of each word. In this demo, we use spark to load an article from local file system. Then count the number of occurrences of each word. Here is the article we input. We can see that: “Hello” appeared [ə'pɪrd] 3 times, ‘Spark’,’Ian’ and ‘Scala’ each appeared 1 time. Run it.

We can see the result is consistent with what we expected. Let’s take a look at the code. We use a few transformation to build a RDD and then print it on screen. First, if we want to use spark, we should create a spark context. We use it to interact with spark. Then load a file from local file system. Here is the path of the file. Spark converts the contents of the file into a RDD. Next, we take a few transformation on RDD, including flatMap, map, reduceByKey and so on. What did Spark do with these operations? I drew a figure to explain it.

textFile method: load file to spark, convert the contents into a RDD.

flatMap: split line on whitespace.

Map: transfer words to tuples

reduceByKey: count the number of occurrences of each word.

Foreach: just print the result

Now do you feel scala code is very concise?

BlueOptimas:

Application：

可以主要应用于两个方面：

Monitor：Monitor the service of our team, such as saving service, find service. So it can save lots of human intervention and cost saving. For example, saving service monitoring we have achieved, time for trade to be processed in saving service is beyond our threshold. It will send an email to us in real time. If production needs and hardware also ready, we can monitor production application in real time.

Analysis: Which accounts are often traded? Which security is popular? It can integrate data from logs and any other data sources and analyze them.

Conclusion:

最后我来总结一下，首先我简单介绍了我们用到的技术，主要介绍了spark，详细介绍了项目的架构，并且为了验证flow是通的，我们结合实际工作需求做了一个demo： BlueOptimas，刚刚也做了展示。因为我们也是从初学，想从头到尾走一遍，来验证它的可行性。另外我还做了一个wordcount的例子，简述了spark和scala语言的特点。

我们在做这个项目的时候也遇到了一些问题：

1. MongoDB对于spark的支持并不完善，存在许多bug，我们也对此给mongo的官方提过两个jira。
2. Hadoop 集群运维成本很高，我们没有相应的support。

等等。

以上就是我今天分享的主要内容。