International Journal of Latest Trends in Engineering and Technology
IJLTET Special Issue- ICRACSC-2016 , pp.018-026
e-ISSN:2278-621X

THE DEVELOPMENT OF BIG DATA FOR SECURITY INCIDENT AND EVENT MANAGEMENT SYSTEMS

Vijay Kumar Atmaku[1], Dr.Mani Sarma Vittapu[2]

Abstract: Big Data is a collection of data sets. It is so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications. Data is accumulating from almost all aspects of our everyday lives that it becomes huge and multi-structured and has hidden useful information. The challenges with Big Data include capture, curation, storage, search, sharing, transfer, analysis, and visualization. Big Data provides materials for mining hidden patterns to support innovation mostly by data mining. The interaction research with Big Data support methods for innovation is rare at present. Knowledge discovered by data mining is novel and quantitative. However, it still lacks a uniform knowledge management model to support the innovation process effectively. The fact shows that since there emergence, big data techniques are changing very fast. In this paper, developing and maintaining the information security system to illustrate how those big data systems evolve and also describes some key design factors and challenges for future big data systems. Proposed design generic systems that can provide near real-time analytic services for many netease applications, such as spam detection, game log analysis and social community mining. No solutions can address all big data problems, especially when data size keeps increasing, more complex user requirements need to handle, the emergence of new hardware violates the old design and the old system becomes too complicated for maintenance. We face a series of technical challenges that have not been well addressed by both academic community and industry.
Keywords: Map Reduce, Real time analysis, Information Security.

I. INTRODUCTION

Nowadays the Internet represents a big space where great amounts of information are added every day. The IBM Big Data Flood Infographic shows that 2.7 Zettabytes of data exist in the digital universe today. Also according to this study there are 100 Terabytes updated daily through Facebook, and a lot of activity on social networks this leading to an estimate of 35 Zettabytes of data generated annually by 2020. Just to have an idea of the amount of data being generated, one zettabyte (ZB) eguals 10^{21} bytes, meaning 10^{12} GB. We can associate the importance of Big Data and Big Data Analysis with the society that we live in. Today we are living in an Informational Society and we are moving towards a Knowledge Based Society. In order to extract better knowledge we need a bigger amount of data. The Society of Information is a society where information plays a major role in the economical, cultural and political stage. In the Knowledge society the competitive advantage is gained through understanding the information and predicting the evolution of facts based on data. The same happens with Big Data. Every organization needs to collect a large set of data in order to support its decision and extract correlations through data analysis as a basis for decisions. In this paper we will define the concept of Big Data, its importance and different perspectives on its use. In addition we will stress the importance of Big Data Analysis and show how the analysis of Big Data will improve decisions in the future. The main importance of Big Data consists in the potential to improve efficiency in the context of use a large volume of data, of different type. If Big Data is defined properly and used accordingly, organizations can get a better view on their business therefore leading to efficiency in different areas like sales, improving the manufactured product and so forth.
Big Data can be used effectively in the following areas:

- In information technology in order to improve security and troubleshooting by analyzing the patterns in the

[1] *Nalla Malla Reddy Engineering College. Hyderabad,India*
[2] *Centre of ITSC, Addis Ababa Institute of Technology, Addis Ababa University, Addis Ababa, Ethiopia*

existing logs;

- In customer service by using information from call centers in order to get the customer pattern and thus enhance customer satisfaction by customizing services;
- In improving services and products through the use of social media content. By knowing the potential customers preferences the company can modify its product in order to address a larger area of people;
- In the detection of fraud in the online transactions for any industry;
- In risk assessment by analyzing information from the transactions on the financial market.

In the future propose to analyze the potential of Big Data and the power that can be enabled through Big Data Analysis.

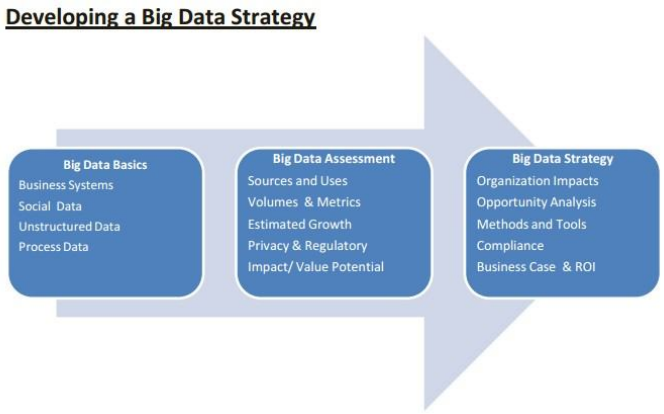
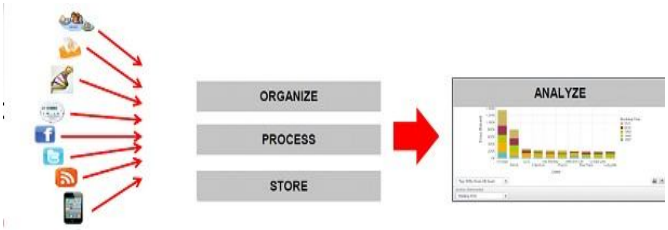


Fig 1. Developing a Big Data Strategy

The **understanding of Big Data** is mainly very important. In order to determine the best strategy for a company it is essential that the data that you are counting on must be properly analyzed. Also the time span of this analysis is important because some of them need to be performed very frequent in order to determine fast any change in the business environment.
Another aspect is represented by the **new technologies** that are developed every day. Considering the fact that Big Data is new to the organizations nowadays, it is necessary for these organizations to learn how to use the new developed technologies as soon as they are on the market. This is an important aspect that is going to bring competitive advantage to a business.
Privacy and Security are also important challenges for Big Data. Because Big Data consists in a large amount of complex data, it is very difficult for a company to sort this data on privacy levels and apply the according security. In addition many of the companies nowadays are doing business cross countries and continents and the differences in privacy laws are considerable and have to be taken into consideration when starting the Big Data initiative. In our opinion for an organization to get competitive advantage from the manipulation of Big Data it has to take very good care of all factors when implementing it. One option of developing a Big Data strategy is presented below. In addition, in order to bring full capabilities to Big Data each company has to take into consideration its own typical business characteristics. The scalability of big data solutions within data centers is an essential consideration. Data is vast today, and it is only going to get bigger. If a data centre can only cope with the levels of data expected in the short to medium term, businesses will quickly spend on system refreshes and upgrades. Forward planning and scalability are therefore important. In order to make every decision as desired there is the need to bring the results of knowledge discovery to the business process and at the same time track any impact in the various dashboards, reports and exception analysis being monitored. New knowledge discovered through analysis may also have a bearing on business strategy, CRM strategy and financial strategy going forward. See figure 2

Fig 2. Big Data Management



Big data analytics can be differentiated from traditional data- processing architectures along a number of dimensions:

- Speed of decision making being very important for decision makers
- Processing complexity because it eases the decision making process
- Transactional data volumes which are very large
- Data structure data can be structured and unstructured
- Flexibility of processing/analysis consisting in the amount of analysis that can be performed on it
- Concurrency

II. INFORMATION SECURITY SYSTEMS

Currently many organizations realize the big data security issues and actively take actions on big data information security problems. In 2011, CSA formed a working group on big data to find solutions for data security and privacy issues. In this paper, based on the status of big data research, we analyzed the current security challenges by big data, and elaborated the current information security protection method of big data.

2.1 Threats of Big Data Security

Just as Gartner said: "big data information security is a necessary fight". Today, big data has penetrated into various industries, and has become a kind of production factor which plays an important role. In the future it would be the highest point of the competition. With the development of rapid processing and analysis technology, the potential information it contained can quickly capture the valuable information in order to provide reference for decision making. However, as big data setting off a wave of productivity and consumer surplus, the challenge of information security is coming either.

2.2 Data Acquisition

The source of big data is diversity. Therefore, the first step to process big data is to collect data from source and pre-process, in order to provide uniform high quality data set to the subsequent process. As a result, due to the inundation of data acquisition, large data become more likely to be "discovered" as a sensitive target, and be more and more attention. On one hand, big data not only means the huge amounts of data, but also means more complex and more sensitive data. These data would attract more potential attackers, and become a more attractive target. On the other hand, with data assembled, the hacker could get more data in one successful attack, and reduce hacker's attack costs. The confidentiality of information refers that according to a specified requirement, information can not be disclosed to unauthorized individuals, entities or processes, or provided the characteristics of its use. A large amount of data collection includes a large number of enterprises operating data, customer information, personal privacy and all kinds of behaviour records. The centralized storage of these data increases the risk of data leakage, and not abused of these data also becomes a part of the personal safety. There is no clear definition to the proprietorship and right to use of sensitive data. And many analysis based on large data did not consider the individual privacy issues involved either.

The integrity of information refers to all the resources which can only be modified by authorized people or with the form of authorization. The purpose is to prevent information from being modified with unauthorized users. Due to the openness of big data, in the process of network transmission, information would be damaged, such as hackers intercepted, interruption, tampering and forgery. Encryption technology has solved the data confidentiality requirements as well as protecting data integrity. But encryption cannot solve all of the safety problems.

2.3 Storage of Data

The formation of network society creates the platform and channel of resource sharing and data exchange for the big data in the field of various industries. Network society based on cloud computation provides an open environment for big data. Network access and data flow provides the basis of rapid elasticity push of the resources and the personalized service. In recent years, from the chain reaction of user account information being stolen on the Internet, it can be seen that big data is more likely to attract hackers, and once being attacked, the volume of stolen data is huge.

Before big data, data storage is divided into relational database and file server. And in current big data, diversity of data type makes us unprepared. For more than 80% of the unstructured data, NoSQL has the advantages of scalability and availability and provides a preliminary solution for big data storage. But NoSQL still exist the following problems: one is that relative to the strict access control and privacy management of SQL technology; Secondly, although NoSQL software gain experience from the traditional data storage, NoSQL still exist all kinds of leak.

2.4 Data Mining

With the development of computer network technology and artificial intelligence, network equipment and data mining application system is more and more widely used,
to provide convenient for big data automatic efficient collecting and intelligent dynamic analysis. On the one hand, big data itself exits leak. Big data itself can be a carrier of sustainable attack. Viruses and malicious software code hidden in large data is hard to find. On the other hand, the technique of attack improves. At the same time of the big data technology such as data mining and data analysis gaining value information, the attacker using these big data technology either, just as the two following aspects. A large number of facts show that failure to properly handle big data will cause great violations to users' privacy. According to the different contents need to be protected, privacy protection can be further divided into location privacy protection, anonymous identifier protection, anonymous connections and so on. The threat People faced with is not only personal privacy leakage, but also prediction and behaviour of the people based on big data. In fact, anonymous protection cannot protect privacy very well. Research on social network also shows that user attributes can be found from the group features. Currently collection, storage, management and use of user data is short of specification, and regulation. Users can't determine their privacy information usage. In commercial scenario, user should have the right to decide how their information be used, and realize users' controllable privacy protection. A general view about big data Is: data itself can tell everything, the data itself is a fact. In fact, if not carefully screened, the data can deceive people, just as people can sometimes be deceived by their eyes. One of the threat of big data credibility is counterfeit or deliberately manufacturing data, and the wrong data often lead to wrong conclusions. If data application scenarios is clearly, someone could deliberately manufacturing data, and create a "false scent", to induced analysts come to the conclusion that was on their side. Because of false information often hidden in a lot of information, it make impossible to identify authenticity of information, so as to make wrong judgment. Due to the production and propagation of false information in network community is becoming more and more easy, its effects should not be underestimated and simply using information security technology to identify the authenticity of all sources is impossible.

III. DATA SECURITY PROTECTION TECHNIQUE

Key technologies in Security protection fields are in great demands to face the security challenges. In this section, we introduce important relevant fields.

3.1 Individual User

As with individual users' information in big data environment, the core and basic techniques to provide privacy protection are still in developing period. Take typical K-anonymity scheme as an example, its early version and optimized version divide quasi-identifiers into groups through tuple generalization and restraining method. When an equivalence class has identical value on some sensitive attribute, attackers are able to confirm its value. In response to this issue, researchers proposed 1-diversity anonymity. Current edge anonymity schemes are mainly based on adding and deleting of the edges. Edge anonymity can be effectively achieved by adding, deleting and exchanging edges randomly. There are problems in such methods that noises randomly added are exiguity, and protections to anonymous edges are insufficient. An important method is to perform division and aggregation operations to super nodes such as node aggregation based anonymous method, genetic arithmetic based method and simulated annealing method based method.

3.2 Internet Enterprise

Information security is critical important for Internet enterprises. System security adopts techniques such as redundancy, network separation, access control, authentication and encryption [18]. Security issues are caused by openness, boundless, freedom of the networks, the key to solve such issues are making network free from them and turning network into controllable, manageable inner system. As network system is the foundation of application system, network security becomes principal issue. Ways to solve network security issues are network redundancy, system separation and access control

3.3 Cloud Service Provider

CSPs provide following measures to prevent security issues in cloud environment. In order to prevent CSPs from peeping users' data and program, separating power and hierarchical management are needed to control access to data in cloud. Provide different authority in accessing data to service provider and enterprise to ensure data security. Enterprise should have total authority and limit authority to CSP.
In cloud computing environment data separation mechanism prevents illegal access to data, however, we should take care of

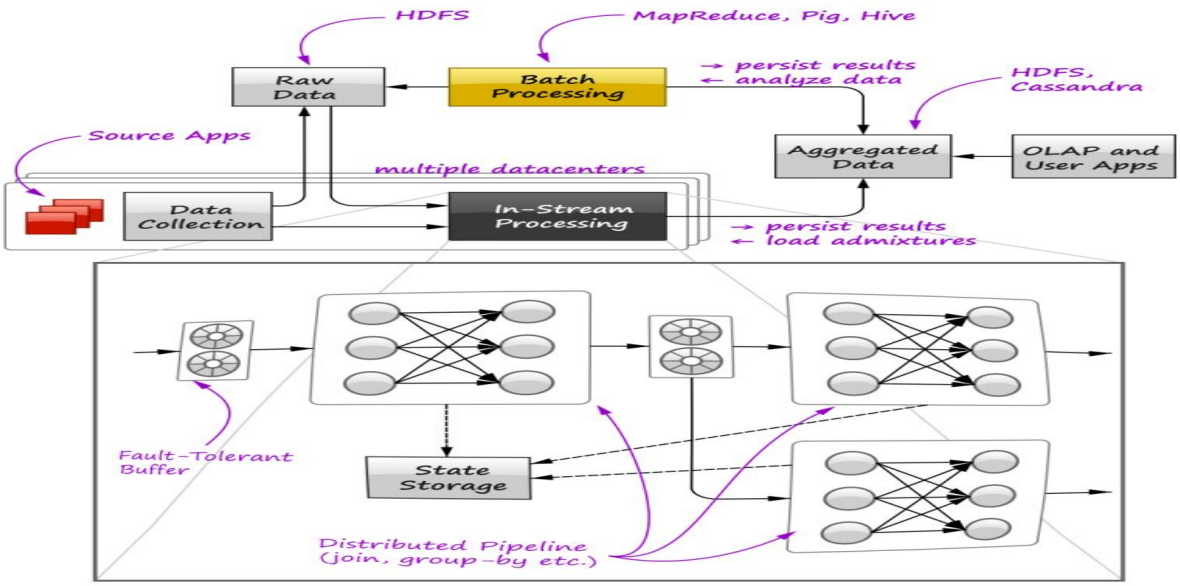
data leakage from CSPs. Mature techniques as symmetrical encryption, public key encryption are available to encrypt data and then upload data to cloud environment. In cloud environment data division is often used with data encryption i.e. encrypted data are scattered in user end and spread in several different clouds. In the way, any CSP is not able to gain complete data.

IV. IN-STREAM BIG DATA PROCESSING

The shortcomings and drawbacks of batch-oriented data processing were widely recognized by the Big Data community quite a long time ago. It became clear that real-time query processing and in-stream processing is the immediate need in many practical applications. In recent years, this idea got a lot of traction and a whole bunch of solutions like Twitter's Storm, Yahoo's S4, Cloudera's Impala, Apache Spark, and Apache Tez appeared and joined the army of Big Data and NoSQL systems. This article is an effort to explore techniques used by developers of in-stream data processing systems, trace the connections of these techniques to massive batch processing and OLTP/OLAP databases, and discuss how one unified query engine can support in-stream, batch, and OLAP processing at the same time. At Grid Dynamics, we recently faced a necessity to build an in-stream data processing system that aimed to crunch about 8 billion events daily providing fault-tolerance and strict transactioanlity i.e. none of these events can be lost or duplicated. This system has been designed to supplement and succeed the existing Hadoop-based system that had too high latency of data processing and too high maintenance costs. The requirements and the system itself were so generic and typical that we describe it below as a canonical model, just like an abstract problem statement. One can see that this environment is a typical Big Data installation: there is a set of applications that produce the raw data in multiple data canters, the data is shipped by means of Data Collection subsystem to HDFS located in the central facility, then the raw data is aggregated and analyzed using the standard Hadoop stack (MapReduce, Pig, Hive) and the aggregated results are stored in HDFS and NoSQL, imported to the OLAP database and accessed by custom user applications. Our goal was to equip all facilities with a new in-stream engine (shown in the bottom of the figure) that processes most intensive data flows and ships the pre-aggregated data to the central facility, thus decreasing the amount of raw data and heavy batch jobs in Hadoop. The design of the in-stream processing engine itself was driven by the following requirements:

A high-level overview of the environment we worked with is shown in the figure below:

Fig: 3. Streaming System



- SQL-like functionality. The engine has to evaluate SQL-like queries continuously, including joins over time windows and different aggregation functions that implement quite complex custom business logic. The engine can also involve relatively static data (admixtures) loaded from the stores of Aggregated Data. Complex multi-pass data mining algorithms are beyond the immediate goals.

- Modularity and flexibility. It is not to say that one can simply issue a SQL-like query and the corresponding pipeline will be created and deployed automatically, but it should be relatively easy to assemble quite complex data processing chains by linking one block to another.
- Fault-tolerance. Strict fault-tolerance is a principal requirement for the engine. As it sketched in the bottom part of the figure, one possible design of the engine is to use distributed data processing pipelines that implement operations like joins and aggregations or chains of such operations, and connect these pipelines by means of fault-tolerant persistent buffers. These buffers also improve modularity of the system by enabling publish/subscribe communication style and easy addition/removal of the pipelines. The pipelines can be stateful and the engine's middleware should provide a persistent storage to enable state checkpointing. All these topics will be discussed in the later sections of the article.
- Interoperability with Hadoop. The engine should be able to ingest both streaming data and data from Hadoop i.e. serve as a custom query engine atop of HDFS.
- High performance and mobility. The system should deliver performance of tens of thousands messages per second even on clusters of minimal size. The engine should be compact and efficient, so one can deploy it in multiple data centres on small clusters.
- First, we explore relations between in-stream data processing systems, massive batch processing systems, and relational query engines to understand how in-stream processing can leverage a huge number of techniques that were devised for other classes of systems.
- Second, we describe a number of patterns and techniques that are frequently used in building of in-stream processing frameworks and systems. In addition, we survey the current and emerging technologies and provide a few implementation tips.

4.1 Stream Replay

Ability to rewind data stream back in time and replay the data is very important for in-stream processing systems. This is the only way to guarantee correct data processing. Even if data processing pipeline is fault-tolerant, it is very problematic to guarantee that the deployed processing logic is defect-free. One can always face a necessity to fix and redeploy the system and replay the data on a new version of the pipeline. The system should work fast enough to rewind the data back in time, replay them, and then catch up with the constantly arriving data stream.

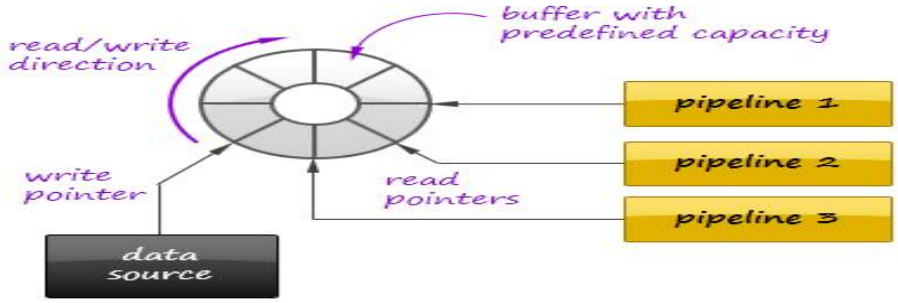


Fig 4. Big Data Stream Replay

4.2 Towards Unified Big Data Processing

It is great that the existing technologies like Hive, Storm, and Impala enable us to crunch Big Data using both batch processing for complex analytics and machine learning, and real-time query processing for online analytics, and in-stream processing for continuous querying. The key observation is that relational query processing, Map Reduce, and in-stream processing could be implemented using exactly the same concepts and techniques like shuffling and pipelining.

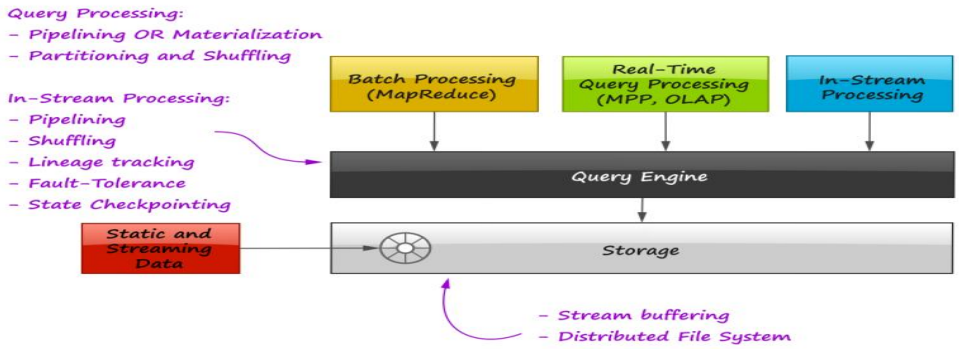


Fig. 5. Unified Big Data processing

V. A GENERIC REAL-TIME ANALYTIC SYSTEM

One major problem with streaming system is lack of flexibility. The architecture of *probability node*, *classification node* and *clustering node* is tailored for the Naive Bayes model. If we want to adopt a more comprehensive model to improve the accuracy, we need to completely change the design. Moreover, it is impossible to extend the system to support other analytic jobs such as game log analysis and social community detection. In summary, the architecture is limited to the spam detection system using Naive Bayes model. Another issue is the scalability and load imbalance problem. The interface is required to be compatible with Hadoop to reduce the efforts of migration, as most existing analytic jobs of Netease are processed in Hadoop with in–memory processing, processing updates and deployment of multiple data centers. Another issue is the scalability and load imbalance problem. Clustering nodes are the bottleneck of the system which may slow down the whole system, while probability nodes and classification nodes only perform simple computations. Because we use the streaming system, the configuration of window size affects the performance. For probability nodes and classification nodes, the window size is1. Namely, an email will be processed immediately at those nodes. On the other hand, the window size of clustering nodes equals to the predefined buffer size. When buffer is full, we invoke the clustering algorithm. Larger buffer incurs high memory overhead and delays the update of model. On the contrary, if we use a small buffer, we can update the model more frequently. However, the clustering results are not good, as some clusters only contain few members. This may make the administrators hard to decide whether the emails are spams or not and further affect the accuracy of the model. So to design a generic real-time analytic system based on the following considerations:

Scalability The system should be able to support different ana-lytic jobs of Netease in the next two years. The size of data will continuously increase (e.g., the number of email per second is expected to increase to more than ten thousands). Given limited computation resource, the sys-tem should provide scalable and efficient services. More-over, because Netease has several data centers located in Hangzhou, Beijing and Guangzhou, the analytic system should support the deployment on multiple data centers.

Update The analytic system is expected to handle both batch pro-cessing and real-time processing. For email spam detection and malicious payment detection, the system should return real-time results, while for the game log analysis and social community mining, it may process a huge size of data in an offline manner.

Model complexity The system needs to provide a flexible programming interface, so that different applications and models can be efficiently developed on top of the system. The interface is required to be compatible with Hadoop to reduce the efforts of migration, as most existing analytic jobs of Netease are processed in Hadoop.

5.1. epiC engine

epiC is based on the actor model where each compute node is considered as an individual actor, communicating with others via messages. Each actor performs its own processing specified by a user-defined function which accepts messages as its parameters. In epiC, actors do not transfer data between each other. Instead, they only use messages to indicate the location of a specific partition in the DFS. All actors will directly fetch their data from the DFS for processing. This strategy allows users to link actors dynamic message flow. The structure of the DAG can even change during the processing. This is one of the most important features of epiC, the flexibility. epiC's flexibility is also reflected on its programming model. It provides a simple *unit*interface, where users can write their customized functions. We can implement MapReduce or Pregel model on top of epiC. So previous programs written for Hadoop can be run on epiC with a few modifications. We can also develop a streaming system on top of epiC by using the asynchronous model. Currently, the streaming based spam detection system is

re-implemented using epiC. Besides Naive Bayes model, it now supports other complex classification models. epiC also supports customized optimizations, so that we can extend it to support in-memory processing to further improve the performance.

5.2. In-memory processing

Users are no longer satisfied with the performance of offline analysis. For example, Netease game designers want to have a real-time interactive tool to analyze the game log. They may start from a very general query, such as retrieving all gamers who have not logged in for three days. After examining the list, they can submit a more specific query, e.g., grouping gamers who have not logged in for three days by their characters' levels. They expect every query to be processed in real-time, so they can adjust their queries adaptively.

To build a system that can provide near real-time analytic ser-vices, we have to apply the in-memory techniques due to slow I/O performances of existing hardware. In particular, we are focusing on the following points:

• implementing a distributed memory array which follows the same design philosophy of RDD [28]to support scalable and fault tolerant data storage in memory. The memory array provides an interface for epiC's units to access its data. In this way, we extend epiC to an in-memory processing engine.

•Simply storing the data in memory cannot fully exploit the benefit of new architecture. We need some specific optimizations. For example, new index structures should be designed to maximize the hit ratio of L2 cache, instead of reducing I/O costs. Radix sort works better than quick sort and merge sort. Column-oriented storage model can adopt the late materializa-tion approach more aggressively.

•Even the price of DRAM continuously drops, it is still too ex-pensive to create a pure memory cluster to support Petabyte scale data. Therefore, a large portion of data are still main-tained in hard disks. The data partitioning strategy (which part of data are maintained in memory) is crucial for the perfor-mance. Our initial solution is to design a fast loading approach which can efficiently parse disk data into memory. The dataset for analyzing will be loaded into memory on the fly. How-ever, this approach incurs high overheads if data are frequently swapped in or out from memory.

5.3. Processing updates

As a real-time system, we cannot apply the batch update scheme which is widely adopted in the MapReduce systems. Up-dates and queries must be processed concurrently. To guarantee the consistence of analytic results, two typical strategies are employed, locking and multi-versions. In Google's Spanner [8], the conventional two-phase locking is used. Although locking affects the system's throughput, Google argues that the programmers and users should be responsible for that. If they want a better performance, they should avoid using the locks. In our system, we run analytic queries and updates together. Because some queries need to scan a large portion of data, they may lock the entire database blocking all the updates. So in our solution, we adopt the multi-version approach as in Hyper [9]. We still apply the two-phase locking to resolve the conflictions between updates. For each analytic query, we create a specific version *V*of data for it. Originally, *V*equals to current dataset. After a few updates, current dataset is newer than *V*. To handle such cases, we replicate a tuple *t*be-fore applying any update to it. Let *T*be the tuples that have not been updated during the query processing. We use *T*_to denote the set of replicated tuples. *V*is then materialized as *T*□*T*_. In other words, by using replication, we avoid locking tables for the analytic query. Note that if there are multiple analytic queries, we may create multiple versions of data. The storage overhead, how-ever, is low, because we will discard the old versions of data after those queries have been processed.

5.4. Deployment on multiple data centers

Designing a system that be deployed on multiple data centers is much more challenging. There are a few issues affecting the performance or even correctness of the data processing.

First, the network latency between data centers is much higher and unstable. When partitioning data, we should always store the data that are frequently accessed together in the same data center. An efficient analytic algorithm is partition-aware which intention-ally avoids shuffling data between nodes in different data centers. This violates our design rule, namely, providing a transparent programming model for up-layer users. Hence, we are trying to developing a module for epiC that can automatically optimize the performance by considering the network partitions.

Second, the clocks of different data centers are not synchronized. This makes timestamp based approach does not work anymore. Two consecutive messages may be handled in different orders at each data center. To address the problem, we select some nodes as time servers from each center and synchronize their clocks. Other nodes will ask the time servers to get the correct clock.

Finally, it is difficult to keep the CAP property. If one data center fails or disconnects from the network, all its data are not accessible. If we keep a replica for data in that center, we will have the consistent issue. It is too costly to keep the replica (normally in other data centers) consistent with the master copy. Besides, if we use the replica to process updates and failures, when the data center recovers, we need to start a synchronization process. To re-duce the complexity of failure recovery, when a data center fails, updates to its data will be rejected. We only use the replica to answer the analytic queries. Updates to other data centers are still allowed. This simple solution works well when we only have a few data centers. A more sophisticated approach is being developed to support more complex multi-center architecture.

VI. CONCLUSIONS

In this paper, we use the information security system in big data system evolves when users' requirements keep changing. How to design a generic system that can provide near real-time analytic services for many related applications, such as spam detection, game log analysis and social community mining. Based on our experiences, no solution can address all big data problems, especially when data size keeps increasing, more complex user requirements need to be handled, the emergence of new hardware violates the old design and the old system becomes too complicated for maintenance. New applications will emerge when we combine big data techniques with other conventional industries while in the combination process those applications will pose new requirements for big data systems, pushing us to search and propose new solutions.

REFERENCES

[1] A Navint Partners White Paper, "Why is BIG Data Important?" May 2012, http://www.navint.com/images/Big.Data.pdf.
[2] http://www.informatioweek.com/software/business-intelligence/sas-gets.hip-to-hadoop-forbigdata/ 240009035? / pgno.2.
[3] Chen Mingqi, Jiang He. USA Information Network Security New Strategy Analysis in Big Data . Information Network Security. 2012(8):32—35
[4] Narayanan A, Shmatikov V. How to break anonymity of the Netflix prize dataset. ArXiv Computer Science e-prints, 2006, arXiv:cs/0610105: 1-10.
[5]G. Caruana, Maozhen Li, Man Qi, A mapreduce based parallel svm for large scale spam filtering, in: Fuzzy Systems and Knowledge Discovery, FSKD, 2011, pp.2659–2662.
[6]Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Michael Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber, Bigtable: a distributed storage system for structured data, ACM Trans. Comput. Syst. 26(2) (2008).
[7] Alfons Kemper, Thomas Neumann, Hyper: a hybrid oltp&olap main memory database system based on virtual memory snapshots, in: ICDE, 2011,pp. 195–206.
[8]James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J.J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Pe-ter Hochschild, Wilson C. Hsieh, Sebastian Kanthak, Eugene Kogan, Hongyi Li, Alexander Lloyd, Sergey Melnik, David Mwaura, David Nagle, Sean Quinlan, Ra-jesh Rao, Lindsay Rolig, Yasushi Saito, Michal Szymaniak, Christopher Taylor, Ruth Wang, Dale Woodford, Spanner: Google's globally distributed database, ACM Trans. Comput. Syst. 31(3) (2013) 8.
[9]Alfons Kemper, Thomas Neumann, Hyper: a hybrid oltp&olap main mem-ory database system based on virtual memory snapshots, in: ICDE, 2011, pp.195–206.