An Open-Source Framework Unifying Stream and Batch Processing

Kiran Deshpande and Madhuri Rao

Abstract Log monitoring and analysis plays critical role in identifying events and traces to understand system behaviour at that point in time and to ensure predictive, corrective actions if required. This research is centered towards modelling open-source framework meant for real-time and historical log analytics of IT infrastructure of an educational institute consisting of application servers hosted over Internet and Intranet, peripheral firewalls and IoT devices. Modelling such framework has not only enhanced processing speed of real-time and historical logs through stream processing and batch processing, respectively, but also facilitated system administrators with critical security incidents monitoring and analysis in near-real time. It also allowed forensic investigations on indexed historical logs stored after stream processing by using batch processing. The modelled framework provides open-source, efficient, user-friendly, enterprise-ready centralized heterogeneous log analysis platform with fast searching options. Open-source tools like Apache Flume, Apache Kafka, ELK Stack and Apache Spark are used for log ingestion, stream processing, real-time search and analytics and batch processing, respectively, in this work. Arriving at a novel solution to unify big data processing paradigms stream and batch processing for log analytics, we propose an approach that can be extrapolated to a generalized system for log analytics across a large infrastructure generating voluminous heterogeneous logs.

Keywords Big data · Batch processing · Stream processing · Heterogeneous log analytic · Apache Spark · ELK Stack · Apache Kafka · Performance evaluation

1 Introduction

Taking into consideration the increase in demand for large-scale data processing, there is enhancement in researcher's interest in batch processing, which involves offline processing of large volumes of data at rest, as well as stream processing,

K. Deshpande (✉) · M. Rao
Thadomal Shahani Engineering College, University of Mumbai, Mumbai, Maharashtra, India
e-mail: madhuri.rao@thadomal.org

S. Smys et al. (eds.), *Inventive Computation and Information Technologies*, Lecture Notes in Networks and Systems 336, https://doi.org/10.1007/978-981-16-6723-7_45

which involves online processing of large quantities of data in motion. Since era of big data is upon us, applications requiring support for both stream and batch processing are increasing, there is a requirement to develop common framework which will unify these both big data processing methods and offer support for both. Significant research has been observed in development of extended functionalities for batch processing frameworks for offering support for stream processing due to popularity of batch processing frameworks. But since current era of big data is more focused towards stream processing, there is requirement of novel framework, where data in motion is processed through stream processing framework at foundation, and data at rest will be processed through batch processing framework built on top of stream processing framework [1, 2].

Recent research in big data has been not only focused towards in-depth analysis of the data from past to find useful insights so that decision making capability will become more precise, but also focused towards handling the notion of real-time big data [3, 4]. Using big data generated smartly to gain valuable decision support becomes important. This would only happen if we analyse all the data we have and get important insights and directions. The main bottleneck in the big data analysis is to process data as soon as it is generated if possible in real time to gain important insight [4]. Big data analysis solution which will address this bottleneck needs to be very adaptable because of information technology evolution. Increasing demand for big data analytics has given rise to development of plethora of data processing systems in recent years, offering a broad range of features and capabilities [4]. The data processing systems developed can be categorized into either a batch processing which focuses on processing data at rest or stream processing system that processes data in motion [5]. However, in current era of big data, applications that can support both processing paradigms will provide more benefits to organizations [2]. For example, the detection of abnormal and malicious events through real-time monitoring of stream-based applications. Classification of the event through offline analysis, its correlation and taking appropriate actions can be initiated [6, 7]. The framework offering both batch and stream processing needs to be modelled to provide execution environment to develop applications using both real-time and offline analysis [2]. Such framework can serve need of next generation big data analysis in huge way.

Every organization's working is heavily dependent on Internet. It becomes very important to analyse Internet and Intranet traffic so that abnormal and malicious events can be identified which may hamper organization's security and reputation [6, 7]. Monitoring the logs and system performance of critical applications like Web server, proxy server, peripheral routers and firewalls which contains crucial events related to Intranet and Internet activity [8–11]. Log data is often voluminous and is continuously growing [7, 9]. In order to have forensic analysis of events, real-time as well as historical data is required to find out in-depth correlation. In such scenario, for real-time log ingestion, processing, faster storage, retrieval and search of such big data technologies like Apache Flume, Apache Kafka, ELK Stack, ES-Hadoop, Hive, Apache Hadoop and Apache Spark can prove efficient.

Major security disasters consist of a series of steps. So if we can identify preceding steps to major security disasters, at the first occurrence itself, we may avoid them from happening [6]. Managing organization information technology infrastructure securely is critical. Until and unless respective traffic is monitored, analysed and correlated, the security flaws present in system, network and application servers deployed cannot be identified. In any medium-scale organization with IT infrastructure-dependent services, log generation is voluminous which needs to be collected and analysed in real-time and to be stored for historical analysis to keep the security posture of the organization in sync with security policy of the organization [7]. Here, stream and batch analytics of logs can play critical role and can help administrators to keep track of various security events and initiate proactive actions [6].

Logs generation is continuous process resulting in voluminous log data with different formats and rates and can be used for obtaining meaningful insights through proper and effective analysis. Introduction of Cloud computing and parallel computation frameworks supporting much required log analysis can help in managing and analysing data of large size with a high production rate [12, 13]. The distributed computing paradigm can address need of resources required for on-demand storage and computing of log mining [8, 9]. Taking into consideration current era of big data and significance of log analysis, several open-source and commercial solutions were designed for implementing log collection, storage, search, analysis and visualization. Taking into consideration advantages and disadvantages of Spark [14, 15], Kafka [16] and ELK Stack [17, 18] regarding log mining, integrating Spark and Elasticsearch capabilities with open-source tools like Apache Flume, Apache Kafka can be advantageous for creating an efficient framework for scalable log management and analysis [16, 19]. Such integration can also prove as milestone for unifying much required stream processing and batch processing capabilities considering current need of big data log analytics.

In this paper, we propose a centralized heterogeneous log analysis framework integrating Apache Flume, Apache Kafka, ELK Stack and Spark. The framework aims at presenting platform unifying stream and batch processing and providing real-time search and analytics by ensuring full use of the functionalities offered by all these platforms listed earlier. It also can serve as a guide for implementing Elasticsearch-based data analysis after performing stream processing and batch processing to cater the current need of log analytics. Thus, the proposed framework will provide open-source, enterprise-ready platform and solution for heterogeneous real-time log monitoring, analysis which will provide better insights for faster trouble shooting and can be widely used across multiple enterprises and domains. To the best of our knowledge, this is the first effort which integrates capabilities of Apache Flume, Apache Kafka, ELK Stack and Apache Spark for big data log analytics to address most of the big data log analytics requirements and challenges listed in Sects. 1.2 and 1.3, respectively.

The rest of the paper is organized as follows. In addition to Introduction (Sect. 1), Sect. 2 discusses related work, Sect. 3 includes major objectives of this research. Section 4 introduces the proposed system architecture for log analytics. Section 4

includes details of technological stack used in proposed framework. Section 5 delineates implementation architecture through FOSS Tools. Section 6 details the experimental environment as well as interprets and evaluate results achieved. The last section, Sect. 7, summarizes conclusions delivered with a brief discussion of future work.

1.1 Note on Big Data Processing Paradigms

Big data analytics is the process of using analysis methods running on powerful supporting platforms to discover potential insights hidden in big data, such as unseen patterns or unknown correlations. Big data analytics can be categorized into two varying paradigms as per their processing time requirements [1, 5].

- Streaming Processing: Streaming processing paradigm works on the assumption that the potential insights of data depends on data freshness [20]. Thus, the streaming processing paradigm ensures analysis of data as soon as possible to derive insights. In this paradigm, data arrives in a stream with continuous arrival. To find approximation results, one or few passes over the stream are made. Representative open-source systems including Storm and Kafka [3] can be used to see impact and application of streaming processing paradigm in online application which require data processing at the second, or even millisecond, level [1, 5].
- Batch Processing: Basis for batch processing paradigm is that data is first stored and then used for analysis [4]. MapReduce has become most popular and dominant batch processing model. MapReduce divides data into small chunks; then these chunks are processed in parallel and in a distributed manner to generate intermediate results. All the intermediate results are aggregated to derive final result. Batch processing MapReduce model schedules computation resources close to data location to avoid the communication overhead during data transmission [1, 5]. Wide adoption of MapReduce model is seen in bioinformatics, Web mining and machine learning [21].

1.2 Why Log Analysis Is Required

Log analysis is the process of finding meaningful insights from system, network and application server generated logs refereed as log events. Log analysis helps in providing useful metrics by which administrators can know events happened across the infrastructure and can use this information in order to improve or solve performance issues within an application or infrastructure. To ensure mitigation of risks, comply with security policies and understand online user behaviour, proactive and reactive log analysis can help organizations. Log information is much needed in different security perspective. Log analysis is required for:

- Debugging Information: Enabling logging in applications and devices can help in checking for a specific error message or event occurrence which may help in debugging.
- Performance Evaluation: Logs information can play very important role in optimizing resource utilization as well as in finding out performance related issues. To understand system, application, network health and performance over date and time information written in logs is crucial.
- Security Evaluation: To manage the system, network and application security of any organization log analysis play crucial role. To detect security breaches, application misuse and malicious events it can be helpful.
- Predictive analysis: For futuristic analysis of threats, flaws, traffic patterns, security policy design, resource optimization information gathered thorough log analysis can help administrators.
- Internet of Things and Logging: Log analysis is also very important to know the status and health of the IoT devices for their uninterrupted operation since effective management of such devices is dependent on logs and alerts generated by them.

1.3 Challenges in Big Data Log Analysis

Even if log analysis is much useful in current era taking into consideration advantages discussed earlier, but it imposes lot of challenges which needs to be resolved. Searching, analysing, correlating and visualizing system, application and network log generated by the IT and network infrastructure will play crucial role to gain meaningful insights. Manual log analysis goes beyond human capabilities. Log analysis involves following major challenges:

- Heterogeneous log format: Every application and device has different log format. Understanding different logs formats and searching across different format can be time consuming.
- Different time format: Every log record has date and time which is crucial for log analysis. Correlation of log events interpreting incorrect date and time becomes difficult.
- Decentralized log: Application servers, devices are distributed over the network, it is difficult to monitor, handle logs of application servers, devices until centralization of the logs is done.
- Storage and retrieval: Voluminous nature of logs makes storage, retrieval and processing difficult. Secure storage of logs generated is also crucial since it contain lot of security information and details.

2 Related Work

With the prospective of finding value from big data in hand, research of big data analysis means and tools is increasingly gaining popularity. The survey paper [1] is an attempt to analyse the definition, framework and typical big data processing systems of big data. This especially focuses not only on conceptual illustration and comparison of batch data processing system and stream data processing system but also focuses on hybrid processing system. There has been a lot of work on both batch and stream processing over the last decade. The seminal paper [2] presents unification of stream processing and batch processing through common computing framework. It discusses development of middle layer between MapReduce applications and the streaming platform so that benefits of stream processing can be combined with the ease of programming and familiarity of MapReduce batch processing [2]. In Li et al. [7] presented a cloud-based log-mining framework using Apache Spark and Elasticsearch to speed up log analysis process of HTTP and FTP access logs. The paper [3] represents evaluation of distributed stream processing platforms like Apache Storm, Apache Spark and Apache Flink for IoT applications with their advantages and disadvantages. With respect to issues identified in the survey paper [20], streaming analytics can be considered as an emerging research area focusing on key issues like scalability, integration, fault tolerance, timeliness, consistency, heterogeneity and load balancing. In [8] work presented by Deepak Mishra et al. on batch processing through popular big data frameworks, Apache Hadoop and Apache Spark, concludes Spark performs better than Hadoop after comparing Hadoop and Spark's performance. The seminal paper [22] discusses Apache Spark-based Analytics of Squid Proxy Logs for studying traffic behaviour and identifying threats by generating Internet traffic statistics like top domains accessed and top users. The review paper [14] discusses use of Apache Spark for big data analytics focusing the key components, abstractions and features of Apache Spark. The conference paper [23] presents a HPC log data analytics framework that is based on a distributed NoSQL database technology and the Apache Spark framework for extracting precise insights useful for system administrators and end users. Experimental analysis presented by Haoxiang and Smys in paper [21] outperforms as compared to other data mining algorithms used for privacy preservation in terms of attack resistance, scalability, execution speed and accuracy.

Few researchers proposed [24, 25] usage of Elastic Stack for easy and rapid management of big data problem of stream processing. In Liu et al. [26] presented cyberattack detection model by making use of ELK Stack for network log analysis and visualization. In this study, network log analysis and management system is designed for providing functions to filter, analyse and present network log data for further processing. DevOps teams are using Docker container technology not only to ensure faster software delivery cycle to boost operational efficiency but also to ensure application portability. In paper, Chen et al. [27] designed Docker container log collection and analysis system by using open source log collection platform ELK, lightweight log collector Filebeat and distributed message queue Kafka.

The seminal paper [28] demonstrated the use of open-source platforms showcasing the feasibility of it by comparing the performance of log analysis between commercial solutions and open-source solutions. Platform that collects [29] the variety of logs using Logstash for identifying the malicious activity in the network proposed by Sanjappa and Ahmed. Author demonstrated use of ELK ecosystem clubbed together for effective analysis of log files in order to get easily understandable insights [25, 29]. Wang et al. [30] presented work aiming to develop a monitoring system using ELK stack to address weakness or unavailable Wi-Fi signal problems.

Wide adoption of log analysis is seen in literature for addressing IT infrastructure security issues. Debnath et al. [31] have implemented machine learning to discover patterns in application logs and used these discovered patterns along with the real-time log parsing for designing advanced log analytic applications. He et al. [32] have proposed system which ensures wide use of logs for managing systems for guaranteed reliability by applying data mining methods. Son and Kwon [28] have emphasized on the reasons because of which the use of open-source tools is preferred over commercial security log analysis systems with huge pricing, complexities and resource requirement. Due to big data properties like volume, variety and rate, it is very difficult to detect the attacks using traditional detection system. More et al. [33] has represented big data technologies use for threat detection.

Data analysis performance can be enhanced through the parallel computation frameworks like Apache Spark and Apache MapReduce using data parallelism. A unified cloud platform with batch analysis and in-memory computing capacity by combining capabilities of Hadoop and Spark [8, 9] was proposed by Lin et al. Large-scale log analytics for detecting abnormal traffic from voluminous logs efficiently by using Hadoop was demonstrated by Therdphapiyanak and Piromsopa.[10]. The ELK stack, i.e. Elasticsearch, Logstash and Kibana, can play important role in developing scalable heterogeneous log analytic platform through its capabilities of automatically collecting, indexing, aggregating and visualizing log data [17, 18]. Efficient geo-identification of website user traffic through generated logs using ELK stack was orchestrated by Prakash et al. [34]. Bagnasco et al. had demonstrated effective use of the Elasticsearch ecosystem for monitoring the infrastructure as a service (IaaS) and scientific application deployed on the cloud [35]. Metha et al. articulated a streaming architecture based on ELK, Spark, and Hadoop for anomaly detection from network connection logs in near-real time [11]. Li et al. introduced a method to speed up log analysis with Elasticsearch and Spark through independent use of Elasticsearch and Spark in their framework [36].

3 Objectives

The Work in this paper is intended towards achieving following objectives through real time and offline log analysis IT Services of an educational Institute. Real-time logs are analysed by stream processing framework Apache Kafka, analysed in near real time by ELK Stack and processed logs will be made available to batch processing

framework Apache Spark for offline analysis as well as forensic investigation of historical logs. Detailed discussion of the same is done in next section.

- To ensure predictive maintenance, troubleshooting of Internet- and Intranet-hosted application servers, peripheral firewall and routers in case of critical security events identified through real-time log analysis.
- To ensure mitigation of malicious security events.
- To ensure real-time performance monitoring of data sources in consideration.
- To conduct forensic investigation through historical data analysis to crosscheck compliance with internal security policies.
- To implement IoT-based environment monitoring system that supports continuous tracking of temperature and humidity levels required to be maintained in centralized server room which hosts the log analytics setup proposed through stream processing of IoT logs of IoT network deployed.

4 Proposed System Architecture

Computing framework is one of the key aspects in improving data analytics and processing efficacy. Since era of big data is upon us, applications requiring support of both stream and batch processing are increasing, and there is a requirement to develop common framework which will unify these both big data processing methods and offer both [2]. Apache Spark Layer at top for batch processing and Apache Kafka, Elastic Stack as bottom layer in proposed framework for real-time log processing, exploration, analytic and search better analysis of security metrics can be done. In general, big data analysis framework can be represented in terms of layered structure, as shown in Fig. 1. It can be categorized into three layers, including device layer, data collection and processing layer and application layer [3–5]. This layered view can help in providing a conceptual clarity to understand working of proposed model and understand complexity of a big data system. The proposed framework can be a part of data collection and processing layer and can be used for data collection and processing through which online and offline analysis of real-time log data generated through data sources in consideration can be done.

- Device layer: Device layer includes sources of big data required for analysis, coming in from all heterogeneous sources like application servers, Web access, IoT devices and network. This layer serves as the foundation for the entire real-time big data processing and analysis in consideration.
- Data Collection and Processing: Data collection and processing is responsible for receiving data from the data sources and ensures its conversion into a format that is in sync with data analysis in consideration. Data collection layer is not only needed because of increase in data sources but also to ensure integration of multi-source, structured and unstructured data for further analysis. In this layer, streaming data will be send for processing, and the accumulated historical data will be stored so that it can be further analysed with analytical tool based on the requirements of the

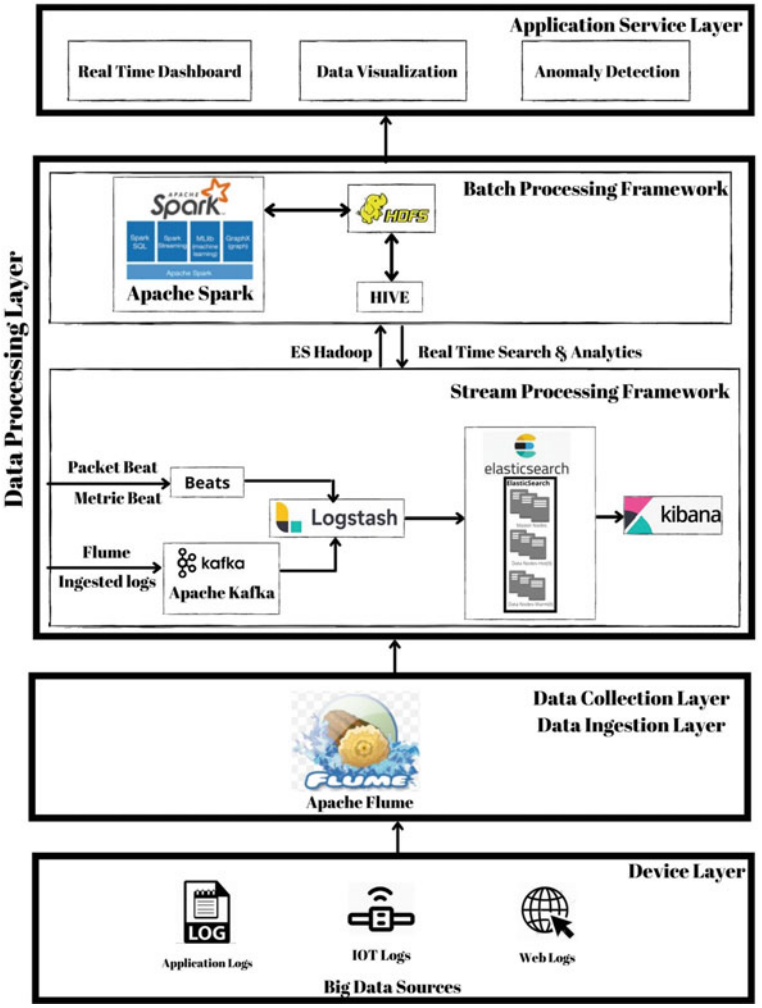

Fig. 1 System architecture

application layer. This layer is the core in processing both big data at fly and big data at rest. Since in current era of big data, sources are extremely heterogeneous in structure and content, the data processing layer ensures parallel computing, data cleansing, data integration, data indexing and so on [3, 5].

- Application Layer: Application layer is the highest layer that uses the interface provided by the data processing layer to perform various data analysis functions like querying, statistical analysis, clustering and classification [5]. It combines basic analytical methods to develop various real-time dash boards required for real-time analysis and performance monitoring of log data in consideration. Along with

the previously discussed layers, application layer can build various applications like recommendation and alert system which can be useful for end users.

5 Implementation Architecture

Figure 2 represents implementation architecture of proposed framework. This section explains working process sequence in detail for proposed framework.

1. Apache Flume is the data ingester. It collects live logs from Intranet, peripheral firewall and application servers as a source and creates a memory-based pipeline to Apache Kafka as a sink.
2. Apache Kafka gets log data from Flume and collects that log data into topics. Kafka uses java instances to manage these topics (jps). Zookeeper works with Apache kafka at the backend for replication, recovery and management of these topics. Here, Apache Kafka uses producer and consumer process (jps). Kafka producer gets log data from Flume and Kafka consumer transfers that log data to logstash for parsing.
3. Logstash is responsible for parsing and filtering of raw log data from Apache Kafka. It reads topics from Kafka consumer and does parsing of that data by creating attribute names. These attribute names are useful for creation of Elasticsearch index. At the same time, logstash perform filtering of content by applying filter plugin like grok which matches specific patterns and eliminate data out of that pattern. This helps making data structured and creation of index in Elasticsearch easy.

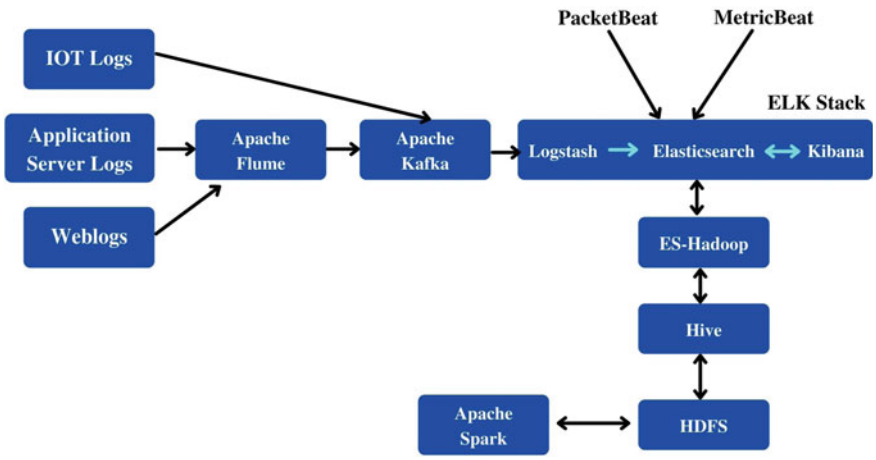

Fig. 2 Implementation architecture

4. Elasticsearch creates index and maps structured data into it. Timestamp-based index mapping can be done over log data to create visualization in Kibana. Visualizations based on Elasticsearch indices helps in creating interpretation of live data in form of bar chart, line graph, pie chart, heat map, gauge etc.
5. ES-Hadoop connector is used to transfer indexed data from Elasticsearch to HDFS. Apache hive is the interface used by Hadoop. Hive provides SQL like interface for data manipulation.
6. Hadoop architecture uses MapReduce algorithm to perform batch processing over HFDS. MapReduce has limitations of processing due to file system read–write operations latency. Here, Apache spark provides 100 times faster solution to MapReduce latency during batch processing. Apache spark architecture makes use of RDD (resilient distributed datasets), data frames and perform transitions on HDFS data by fetching it inside main memory.
7. Processed data is stored back in HDFS by spark-scala user interface. This batch-processed optimum data is again transferred to Elasticsearch through ES-Hadoop connector using hive interface. Finally, Elasticsearch can create visualizations required for deep low latency analytic of batch processed historical data using Kibana.
8. Packetbeat and Metricbeat are responsible for real-time performance monitoring of IT resources, services and Intranet by shipping related metrics to ELK Server.
9. MQTT Broker: The Message Queuing Telemetry Transport (MQTT) is a lightweight, publish–subscribe network protocol that transports messages between IoT devices. The protocol usually runs over TCP/IP; however, any network protocol that provides ordered, lossless and bidirectional connections can support MQTT. Here, MQTT broker is a server that receives all messages from the IoT clients and then routes the messages to Apache Kafka for further analysis through proposed framework.

6 Experimental Environment and Result Analysis

In this section, hardware and software specifications used in proposed framework as well as experimental environment and result analysis are explained. All server machines used for creating experimental environment are physical machines. Ubuntu 14.04 and Centos 7 with 64 bit is adopted as our operating system.

6.1 Experimental Environment

IT infrastructure network architecture of an educational organization in consideration providing services like DNS, Internet, Web, Squid, NIDS and Firewall Security which generates lot of logging and traffic data is represented through Fig. 3. Infras-

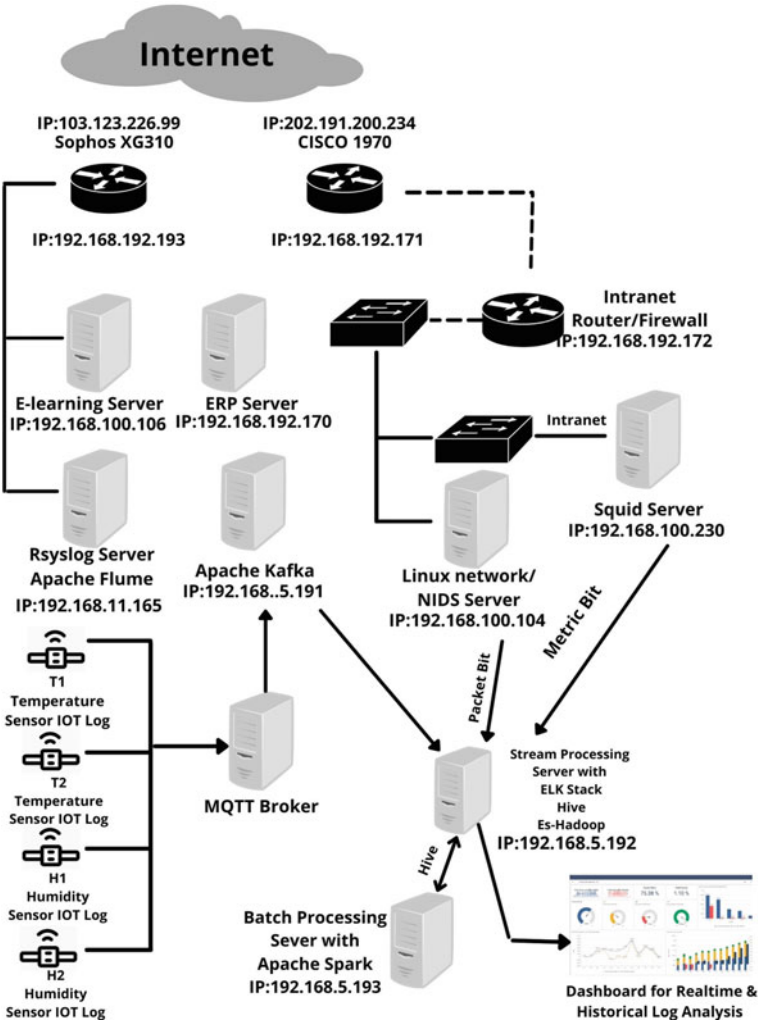

Fig. 3 Block diagram of network architecture setup used for modelling novel framework

tructure also has IoT network deployed through MQTT Broker and Kafka for creating environment monitoring system that supports continuous tracking of temperature and humidity levels required to be maintained in centralized server room which hosts the log analytics setup proposed. All these components are distributed over Local Area Network (LAN). Manually checking logs of each and every server and device on daily basis is not feasible. Syslog has been configured on critical servers and devices like peripheral firewall and Web servers to forward logs to central Rsyslog server so that logs can be made available to Apache Flume for ingesting into framework. Packetbeat and Metricbeat are configured on NIDS, Squid Server to ensure shipping their

logs to ELK Server for real-time network monitoring of Intranet and performance monitoring of Squid Server, respectively.

In implementation scenario, Apache Flume, Apache Kafka and ELK stack have been configured for real-time stream processing, analysis and centralized log management. ES-Hadoop and Hive installed on ELK Server ensure bidirectional movement of indexed logs between ELK and Apache Spark through HDFS which is used as storage for in-depth low latency analytics. This subsection also explains hardware configurations of servers, software configurations of FOSS tools and operating system environments that are used in proposed framework with its purpose.

1. Web Servers: Intel (R) Xeon (R) CPU E5 2603 v3 1.6 GHz 48 GB Memory with Centos 7 Server Operating System. This is used as source of real-time logs of Apache through which Moodle E Learning Platform, ERP Server is made accessible to 3000 users of an educational institute. Log forwarding is done through rsyslog to server with Apache Flume.
2. Squid Proxy Server with Metricbeat: Intel (R) Xeon (R) CPU X3220 2.4 GHz 8 GB Memory with centos 7 Server Operating System. Squid Proxy server is meant for providing secure Internet access in Intranet.
3. Network Intrusion Detection (NIDS) Server with Packetbeat: Intel (R) Xeon (R) CPU X3220 2.4 GHz 8 GB Memory with centos 7 Server Operating System. NIDS Server is meant for monitoring all network traffic of Intranet including switches and routers.
4. Rsyslog Server with Apache Flume: Intel (R) Xeon (R) CPU E5 2603 v3 1.6 GHz 48 GB Memory with Centos 7 Server Operating System. Apache Flume ingests Apache logs of Web servers, peripheral firewall to ELK Stack. This server can be used as sink for multiple heterogeneous log sources.
5. Server with ELK Stack, Apache Spark and Hive Instances deployed: Intel (R) Xeon (R) CPU E5 2603 v3 1.6 GHz 48 GB Memory with Ubuntu 18 Server Operating System. ELK Stack is used for real-time log processing, and Spark is used for offline processing of historical logs. Here, hive uses ES Hadoop connector to transfer indexed data to HDFS from Elasticsearch. Similarly, hive gets batch processed data from HDFS and inserts back to Elasticsearch for in depth analytics.
6. Sophos XG310 (SFOS 18.0.5 MR-5-Build586) Firewall: It also serves as log source for Rsyslog Server along with Web server logs through Syslog service configured.

In implementation architecture for log shipping, stream processing, real-time search and analytic, batch processing and unifying stream processing framework to batch processing network following open-source tools are used.

1. Log Ingestion: Apache Flume 1.5.0.1
2. Data Pipeline Tool: Apache Kafka 2.7.0
3. Batch Processing of Historical Logs: Apache Spark 3.0.1 with Prebuilt for Apache Hadoop 3.1
4. Real-time search and analytic: Elastic Stack 7.7.0
5. Connector of ELK Stack and Apache Spark: ES Hadoop 7.11

6. Lightweight shipper for network data: Packetbeat 7.13.1
7. Lightweight shipper for performance metrics: Metricbeat 7.13.1.

6.2 Result Analysis

In this section, we will discuss some precise insights through IT Infrastructure log analysis of an engineering institute through proposed framework. This Engineering Institute is A. P. Shah Institute of Technology located at Mumbai, Maharashtra, India. There are 3000 active users including students, faculties and administrative staff who accesses the IT Infrastructure in consideration resulting in 10 million logs per day, approximately. This voluminous log analysis is intended to ensure security of critical infrastructure components like routers, firewall, Proxy server, e-learning server and ERP server of this institute as well as identifying the malicious activities. Log analysis represented in this section helps institute to keep its security posture in sync with security policy framed at institute level which will be discussed later in this section. Big data log analytics of unstructured heterogeneous logs represented here helps administrators to keep track of various events related to security and take proactive actions on basis of that. Also, along with real-time log analysis managed through Kafka and ELK Stack, analysis of historical logs through batch processing implemented through Apache Spark can form basis to revise, develop future security policies required for institute. In addition to log analysis, the proposed framework also ensures real-time monitoring of performance metrics related to IT resources which helps in shifting from reactive to proactive monitoring.

Result analysis presented in this section focuses on identifying following security breaches on Apache Web Service configured on Application Servers in consideration. Following are the Apache Service-related security events in consideration around which result analysis will revolve.

1. Using blocked request method: In most cases, GET and POST are the only request methods required to operate a dynamic website. Allowing more request methods than are necessary increases site's vulnerability. Thus, finding request methods which are blocked through real-time log analysis can help administrator.
2. Using blocked user agents: Blacklisting user-agents revolves around the idea that every browser, bot and spider that visits server identifies itself with a specific user-agent character string. Thus, user-agents associated with malicious, unfriendly or otherwise unwanted behaviour may be identified and blacklisted in order to prevent against future access. Thus, finding malicious user-agents which are blocked through real-time log analysis can help administrator.
3. Identifying and tracing access request to Web server with HTTP request result status code 404: Attempts to access resources not hosted on the Web server are indication of trying to run malicious code on Web server to gain access of the server. Identifying and tracing such events through real-time log analysis can help administrators in redesigning security polices of Web servers.

4. Identifying and tracing access request to Web server with HTTP request result status code 403: Attempts to access prohibited documents on the Web server are indication of trying to run malicious code on Web server to gain unauthorized access of the server. Identifying and tracing such events through real-time log analysis can help administrators in redesigning security polices of Web servers.
5. Identifying access requests from poor reputation IPs: Poor reputation IPs are responsible for sending high level of spam and viruses. To identify and block accesses from such IPs can help administrators in preventing malicious activities and enhance security of IT Infrastructure.
6. Identifying Probing on ports of Perimeter Firewall: The perimeter firewall is first level of defence for organization. Identifying open ports is prior step to perform DOS attacks. Identifying and tracing attempts to gain access into organization network through open ports of SSH and telnet through real-time log analysis can help in enhancing security polices of network.

Result analysis presented through this paper also focuses on real-time performance monitoring of Apache Service of application servers through ELK Stack. Result analysis also discusses performance monitoring of Squid Proxy Service through Metricbeat which is a lightweight shipper that can be installed on servers to periodically collect metrics from the operating system and services running on the server. It also focuses on real-time performance monitoring of network traffic collected through Packetbeat which is a lightweight network packet analyser that sends network log data from NIDS Server to ELK Server. Analysis represented through Fig. 4 helps

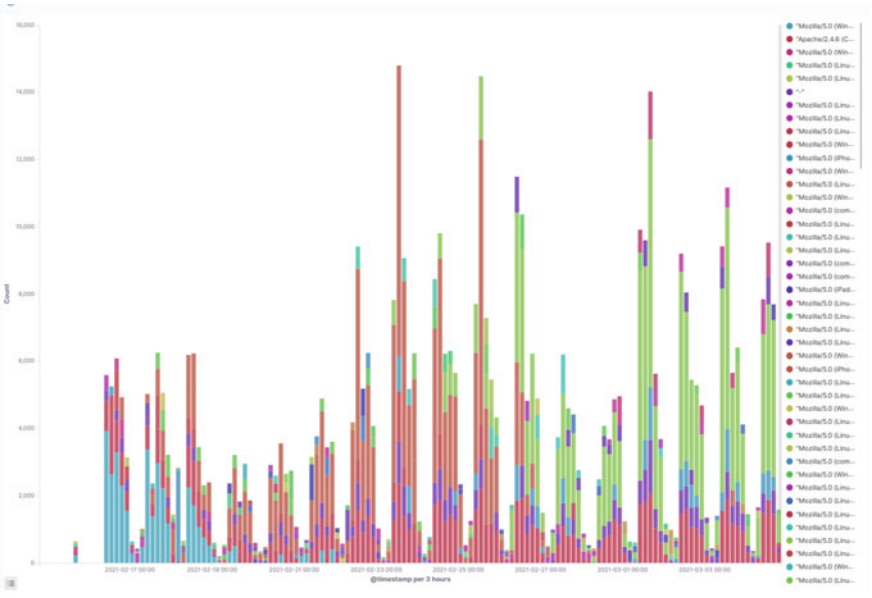

Fig. 4 Analysis of user agent data through batch processing

in identifying user-agents associated with malicious, otherwise unwanted behaviour and blacklisting them such user agents for future access through secure configuration of Apache Service. Analysis also represents the fact that most of user agents that are listed are the user agents white-listed for Apache access through its configuration. It can also help in identifying sources in terms of hosts which are initiating accesses through malicious user agents and categorizing them for further security analysis. Through the analysis depicted through Fig. 5, it seems that Apache Status code 200 was seen most of the time which represents normal working of Apache service on application servers. It also depicts very less percentage of HTTP status codes like 403, 404 which is indication of secure configuration of Web Server. With Apache monitoring, administrators can ensure whether service is configured to sufficiently handle the current scale of access requests. In order to prevent slowdowns which may affect end user experience routine, Apache monitoring can play crucial role in solving performance-related issues in near-real time. Real-time performance monitoring of Apache service deployed on application servers represented through Fig. 6 helps in tracking key Apache monitoring metrics to prevent fatal problems in the scenarios where server load and access requests scale up. With clear and real-time dashboard, represented through [18] Fig. 6 will help administrators to view real-time changes in core metrics and provide stable and efficient server work through proactive actions. Proposed framework also ensures collection, correlation and analysis of squid proxy service logs responsible for providing Internet service over Intranet. Squid logs serve as valuable source of information about Squid workloads and performance. Figure 7

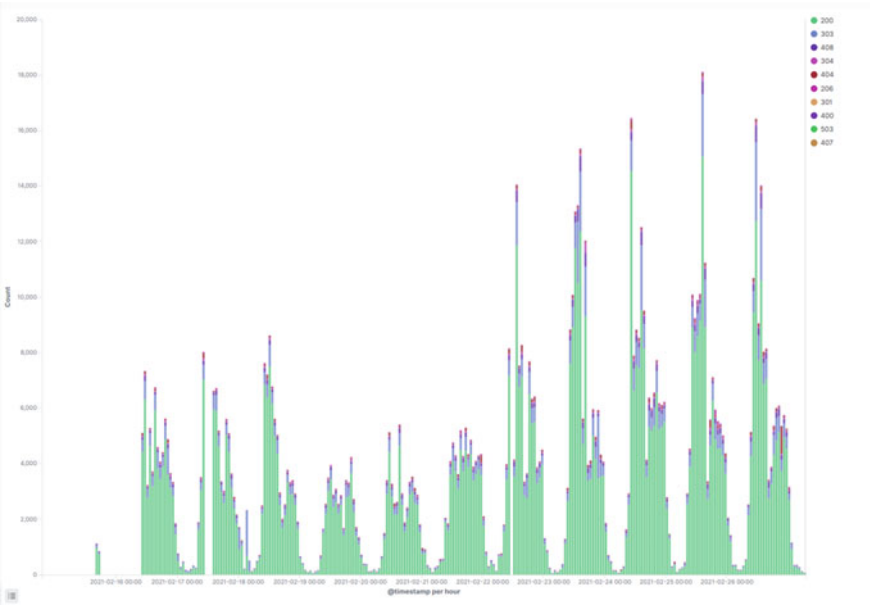

Fig. 5 Analysis of Apache server status code

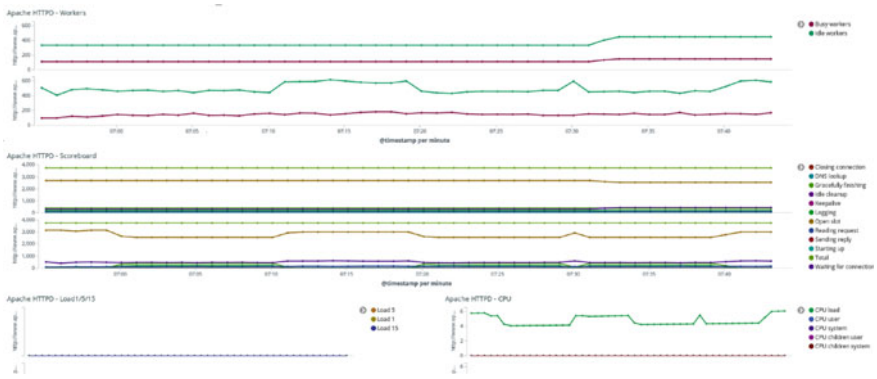

Fig. 6 Real-time performance monitoring of Apache service

Fig. 7 Real-time performance and analysis of Squid Proxy service

[18]represents not only real-time traffic load on Squid Proxy service but also precisely visualize top users usage analysis, top domains accessed, HTTP methods and status code analysis, content types accessed through in-depth analysis of Squid logs.

Application server performance monitoring is really crucial. Using tools like Metricbeat and ELK to monitor, visualise performance metrics makes it convenient to provide stable and error-free working of application servers. Figure 8 [18] represents real-time analysis of application servers in terms infrastructure specific metrics like memory usage, CPU usage, disk usage, network bandwidth and system load through log information shipped through Metricbeat configured on Squid Proxy server.

Packetbeat which is part of the Elastic Stack can be integrated seamlessly with Logstash, Elasticsearch and Kibana in order to transform network data with Logstash,

Fig. 8 Real-time system performance monitoring through Metricbeat

real-time search and analytic in Elasticsearch, review data through precise visualizations in Kibana dashboards [18]. Network performance monitoring can help administrators in ensuring whether network is designed properly and network devices are configured to sufficiently handle the current scale of access. In order to prevent slowdowns in accessing Intranet and Internet contents which may affect end user performing critical tasks, routine real-time network performance monitoring can play crucial role in solving performance related issues in near-real time. Figure 9 [18] depicts real-time Intranet network statistics collected on NIDS server deployed and its analysis by shipping network data collected through Packetbeat to Elasticsearch in terms of precise visualizations through Kibana dashboards. Analysis is represented in terms of connections over time, top Intranet hosts creating and receiving traffic, network traffic statistics between hosts.

IoT devices generate large amounts of logging and events data, monitoring and analysing the same can help both for troubleshooting purposes and as part of predictive maintenance as well. Keeping track of every different activity of IoT devices is impossible without having scalable framework which can address sheer volume of IoT logs. Although the Internet of things (IoT) is getting more and more attraction by researchers, integrating devices and machines to process the data in real time and

Fig. 9 Real-time network performance monitoring through Packetbeat

at scale is a challenge. To reliably operate with IoT setup, a comprehensive view of the device health in aggregate will prove beneficial.

Graphical analysis represented through Fig. 10 showcases analysis of MQTT Broker successful connection requests received over IoT network discussed earlier through ELK stack. Such analysis can help in tracing vulnerabilities like Ping

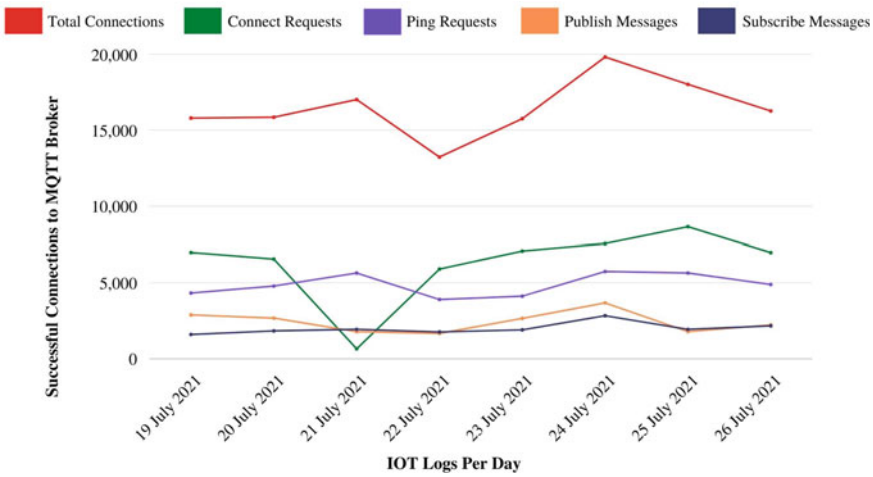

Fig. 10 IoT connection analysis through modelled framework

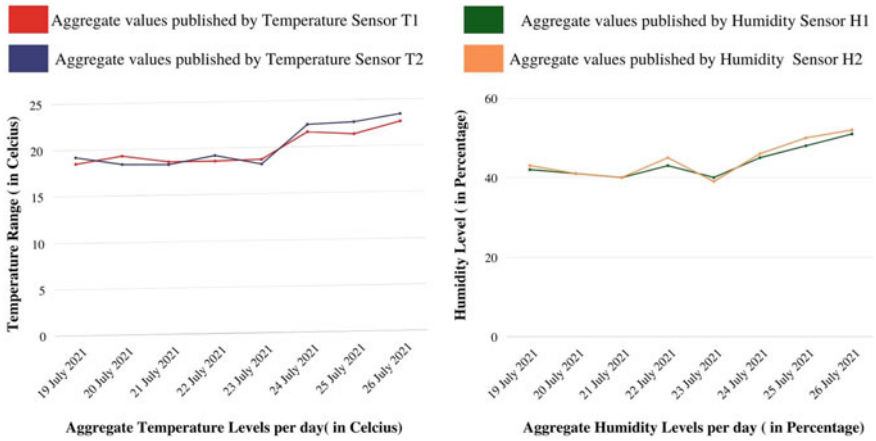

Fig. 11 Aggregate temperature and humidity level analysis through modelled framework

flooding and SYN flooding through which DOS attack is possible. Figure 11 depicts aggregate temperature and humidity level analysis on the values published through IoT network deployed for continuous tracking of temperature and humidity levels required to be maintained in centralized server room which hosts the log analytic setup proposed.

6.3 Performance Evaluation of Framework

This section discusses performance evaluation of proposed framework in analysing voluminous logs generated thorough sources discussed in previous subsection with respect to the search queries listed below. In order to compare the performance 5 million Logs (715 MB), 10 million logs (1.5 GB), 15 million logs (2.1 GB), 20 million logs (2.7 MB) were used for analysis. Time required to search and analyse specific log data pattern from tagged and classified logs is noted down in following table taking into the size of log volumes mentioned earlier. Proposed framework outperforms for different log volume sizes taking into consideration targeted queries listed below.

1. Query I: HTTP Status Code Analysis to get distinct HTTP Status Codes.
2. Query II: Analysing Frequent Hosts to identify and list hosts with poorly rated IPs from access logs.
3. Query III: Analysing the Top ten hosts who have received 404 and 403 HTTP Response Code over month.
4. Query IV: Analysing time period with maximum 404 and 403 HTTP Response Codes per day over month.

Table 1 Visualization time of modelled framework for queries under consideration

Log size	Query I	Query II	Query III	Query IV
5 million	4.7 s	4.7 s	5.7 s	6.7 s
10 million	5.3 s	4.9 s	6.4 s	7.3 s
15 million	5.7 s	5.9 s	7.7 s	8.3 s
20 million	5.9 s	6.1 s	7.9 s	8.6 s

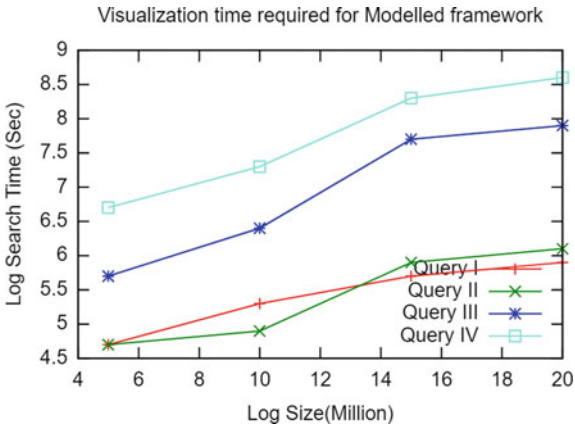

Fig. 12 Visualization time of modelled framework for queries under consideration

5. Query V: Listing probing events on perimeter firewall with respect to ssh port 22 and telnet port 23. Since both these services are configured on nonstandard ports identifying this probing can give important security related insights.

Table 1 lists time required for graphical visualization of specific log data pattern from tagged and classified log data through Kibana over different log sizes. Figure 10 shows graphical analysis of values listed in Table 1. It is observed that Kibana visualization time required for Query III and IV is significantly more to Query I and II which portrait requirement of implementing scalable cluster for Elasticsearch (Fig. 12).

7 Conclusion

Taking into consideration the speed, nature and volume of logs, real-time monitoring of logs is infeasible to get meaningful insights. The setup collects heterogeneous logs to a central location by using Rsyslog server that ingests logs to Apache Kafka and then to ELK Stack for log analysis and correlation. Modelled framework will be helpful in real-time analysis of heterogeneous logs by combining use of open-source tools at different stages of log analysis so that identification of events that might represent threats can be automated. Modelled framework also ensures forensic investigations of historical logs through Apache Spark-based batch processing

framework implemented. Packetbit and Metricbit are used for ingesting network and system logs of NIDS server and Squid server through which real-time network and system performance monitoring can be done. It helps in giving insights of the logged events through graphical visualization which is easy, quick and efficient way to understand and correlate log events. Modelled framework is a much required attempt to unify stream processing implemented through Apache Kafka, ELK Stack and batch processing implemented through Apache Spark. Modelled framework designed and developed using best open-source tools available will provide cost effective, open-source, enterprise-ready platform and solution for online and offline big data log analytics to provide better insights for faster trouble shooting. It can be widely used across multiple enterprises and domains to identify events that might hamper security of their IT services.

Acknowledgements I would like to express our thanks of gratitude to my Guide Prof. Madhuri Rao for guiding me during this work. Lastly, we would like to thank my Research Center Thadomal Shahani College of Engineering for providing me continuous support whenever required.

References

1. S. Yu, Data processing and development of big data system: a survey, in *Advances in Artificial Intelligence and Security. ICAIS 2021*, ed. by X. Sun, X. Zhang, Z. Xia, E. Bertino. Communications in Computer and Information Science, vol. 1423 (Springer, Cham, 2021), p. 34. https://doi.org/10.1007/978-3-030-78618-2
2. M. Harvan, T. Locher, A.C. Sima, Cyclone: unified stream and batch processing, in *2016 45th International Conference on Parallel Processing Workshops (ICPPW)* (2016), pp. 220–229. https://doi.org/10.1109/ICPPW.2016.42
3. H. Nasiri, S. Nasehi, M. Goudarzi, Evaluation of distributed stream processing frameworks for IoT applications in Smart Cities. J. Big Data **6**, 52 (2019). https://doi.org/10.1186/s40537-019-0215-2
4. Z. Lv, H. Song, P. Basanta-Val, A. Steed, M. Jo, Next-generation big data analytics: state of the art, challenges, and future research topics. IEEE Trans. Ind. Inf. **13**(4), 1891–1899 (2017). https://doi.org/10.1109/TII.2017.2650204
5. H. Hu, Y. Wen, T.-S. Chua, X. Li, Toward scalable systems for big data analytics: a technology tutorial. IEEE Access **2**, 652–687 (2014). https://doi.org/10.1109/ACCESS.2014.2332453
6. S. Chaudhari, V.K. Maurya, V. Singh, S.S. Tomara, A. Rajana, A. Rawata, Real time logs and traffic monitoring, analysis and visualization setup for IT security enhancement, in *Next Generation Computing Technologies (NGCT-2019)* (2019)
7. Y. Li, Y. Jiang, J. Gu, M. Lu, M. Yu, E.M. Armstrong, T. Huang, D. Moroni, L.J. McGibbney, G. Frank, C. Yang, A cloud-based framework for large-scale log mining through Apache Spark and elasticsearch. Appl. Sci. **9**(6) (2019)
8. I. Mavridis, H. Karatza, Performance evaluation of cloud-based log file analysis with Apache Hadoop and Apache Spark. J. Syst. Softw. **125**, 133–151 (2017). ISSN 0164-1212. https://doi.org/10.1016/j.jss.2016.11.037
9. X. Lin, P. Wang, B. Wu, Log analysis in cloud computing environment with Hadoop and Spark, in *2013 5th IEEE International Conference on Broadband Network and Multimedia Technology* (2013), pp. 273–276. https://doi.org/10.1109/ICBNMT.2013.6823956

10. J. Therdphapiyanak, K. Piromsopa, Applying Hadoop for log analysis toward distributed IDS, in *Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication (ICUIMC'13)* (Association for Computing Machinery, New York, NY, USA, 2013), Article 3, pp. 1–6. https://doi.org/10.1145/2448556.2448559
11. S. Mehta, P. Kothuri, D.L. Garcia, Anomaly Detection for Network Connection Logs (2018). arXiv:1812.01941
12. C. Yang, M. Yu, F. Hu, Y. Jiang, Y. Li, Utilizing cloud computing to address big geospatial data challenges. Comput. Environ. Urban Syst. **61**, Part B, 120–128 (2017). ISSN 0198-9715
13. C. Yang, Q. Huang, Z. Li, K. Liu, F. Hu, Big data and cloud computing: innovation opportunities and challenges. Int. J. Digital Earth **10**(1), 13–53 (2017). https://doi.org/10.1080/17538947.2016.1239771
14. S. Salloum, R. Dautov, X. Chen et al., Big data analytics on Apache Spark. Int. J. Data Sci. Anal. **1**, 145–164 (2016). https://doi.org/10.1007/s41060-016-0027-9
15. https://spark.apache.org/
16. https://kafka.apache.org/
17. S. Chhajed, *Learning ELK Stack* (Packt Publishing Ltd., Birmingham, UK, 2015)
18. https://www.elastic.co/
19. https://flume.apache.org/
20. T. Kolajo, O. Daramola, A. Adebiyi, Big data stream analysis: a systematic literature review. J. Big Data **6**, 47 (2019). https://doi.org/10.1186/s40537-019-0210-7
21. W. Haoxiang, S. Smys, Big data analysis and perturbation using data mining algorithm. J. Soft Comput. Paradigm (JSCP) **3**(01), 19–28 (2021)
22. D.D. Mishra, S. Pathan, C. Murthy, Apache Spark based analytics of Squid Proxy Logs, in *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, vol. 2018 (2018), pp. 1–6. https://doi.org/10.1109/ANTS.2018.8710044
23. B.H. Park, S. Hukerikar, R. Adamson, C. Engelmann, Big data meets HPC Log analytics: scalable approach to understanding systems at extreme scale, in *IEEE International Conference on Cluster Computing (CLUSTER)*, vol. 2017 (2017), pp. 758–765. https://doi.org/10.1109/CLUSTER.2017.113
24. M. Bajer, Building an IoT data hub with elasticsearch, Logstash and Kibana, in *2017 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)* (2017), pp. 63–68. https://doi.org/10.1109/FiCloudW.2017.101
25. I.Y.M. Al-Mahbashi, M.B. Potdar, P. Chauhan, Network security enhancement through effective log analysis using ELK, in *International Conference on Computing Methodologies and Communication (ICCMC)*, vol. 2017 (2017), pp. 566–570. https://doi.org/10.1109/ICCMC.2017.8282530
26. J.C. Liu, C.T. Yang, Y.W. Chan et al., Cyberattack detection model using deep learning in a network log system with data visualization. J. Supercomput. (2021). https://doi.org/10.1007/s11227-021-03715-6
27. L. Chen, J. Liu, M. Xian, H. Wang, Docker container log collection and analysis system based on ELK, in *International Conference on Computer Information and Big Data Applications (CIBDA)*, vol. 2020 (2020), pp. 317–320. https://doi.org/10.1109/CIBDA50819.2020.00078
28. S.J. Son, Y. Kwon, Performance of ELK stack and commercial system in security log analysis, in *2017 IEEE 13th Malaysia International Conference on Communications (MICC)* (2017), pp. 187–190. https://doi.org/10.1109/MICC.2017.8311756
29. S. Sanjappa, M. Ahmed, Analysis of logs by using Logstash, in *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications*, ed. by S. Satapathy, V. Bhateja, S. Udgata, P. Pattnaik. Advances in Intelligent Systems and Computing, vol. 516 (Springer, Singapore, 2017). https://doi.org/10.1007/978-981-10-3156-4
30. Y.T. Wang, C.T. Yang, E. Kristiani, Y.W. Chan, The implementation of Wi-Fi Log analysis system with ELK Stack, in *Frontier Computing. FC 2018*, ed. by J. Hung, N. Yen, L. Hui. Lecture Notes in Electrical Engineering, vol. 542 (Springer, Singapore, 2019). https://doi.org/10.1007/978-981-13-3648-528

31. B. Debnath et al., LogLens: a real-time log analysis system, in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)* (2018), pp. 1052–1062. https://doi.org/10.1109/ICDCS.2018.00105
32. P. He, J. Zhu, S. He, J. Li, M.R. Lyu, Towards automated log parsing for large-scale log data analysis. IEEE Trans. Dependable Secure Comput. **15**(6), 931–944 (2018). https://doi.org/10.1109/TDSC.2017.2762673
33. R. More, A. Unakal, V. Kulkarni, R.H. Goudar, Real time threat detection system in cloud using big data analytics, in *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology (RTEICT)*, Bangalore (2017), pp. 1262–1264
34. T. Prakash, M. Kakkar, K. Patel, Geo-identification of web users through logs using ELK stack, in *Proceedings of the 2016 6th International Conference Cloud System and Big Data Engineering (Confluence)*, Noida, India, 14–15 Jan 2016, pp. 606–610
35. S. Bagnasco, D. Berzano, A. Guarise, S. Lusso, M. Masera, S. Vallero, Monitoring of IaaS and scientific applications on the cloud using the elasticsearch ecosystem. Proc. J. Phys. **608**, 012016 (2015)
36. Y. Li, Y. Jiang, F. Hu, C. Yang, Armstrong, T. Huang, D. Moroni, C. Fench, Leveraging cloud computing to speedup user access log mining, in *Proceedings of the OCEANS 2016 MTS/IEEE Monterey*, Monterey, CA, USA, 19–23 Sept 2016
37. C.T. Yang, E. Kristiani, Y.T. Wang et al., On construction of a network log management system using ELK stack with Ceph. J. Supercomput. **76**, 6344–6360 (2020). https://doi.org/10.1007/s11227-019-02853-2
38. M. Podhoranyi, A comprehensive social media data processing and analytics architecture by using big data platforms: a case study of twitter flood-risk messages. Earth Sci. Inform. **14**, 913–929 (2021). https://doi.org/10.1007/s12145-021-00601-w
39. F. Firouzi, B. Farahani, Architecting IoT cloud, in *Intelligent Internet of Things*, ed. by F. Firouzi, K. Chakrabarty, S. Nassif (Springer, Cham, 2020), p. 4. https://doi.org/10.1007/978-3-030-30367-9
40. W. Xie, P. Li, H. Xu, Architecture and implementation of real-time analysis system based on cold chain data, in *Complex, Intelligent, and Software Intensive Systems. CISIS 2018*, ed. by L. Barolli, N. Javaid, M. Ikeda, M. Takizawa. Advances in Intelligent Systems and Computing, vol. 772 (Springer, Cham, 2018), p. 44. https://doi.org/10.1007/978-3-319-93659-8
41. https://hive.apache.org/
42. http://hadoop.apache.org/