International Journal on Advanced Science Engineering Information Technology

Vol.8 (2018) No. 4
ISSN: 2088-5334

PPS-ADS: A Framework for Privacy-Preserved and Secured Distributed System Architecture for Handling Big Data

Mohd Abdul Ahad[#], Ranjit Biswas[#]

[#]*Department of Computer Science and Engineering, School of Engineering Sciences and Technology,*
Jamia Hamdard, New Delhi-110062, India
E-mail: itsmeahad@gmail.com, ranjitbiswas@yahoo.com

Abstract— **The exponential expansion of Big Data in 7V's (velocity, variety, veracity, value, variability, and visualization) brings forth new challenges to security, reliability, availability, and privacy of these data sets. Traditional security techniques and algorithms fail to complement this big gigantic data. This paper aims to improve the recently proposed Atrain Distributed System (ADS) by incorporating new features, which will cater to the end-to-end availability and security aspects of the big data in the distributed system. The paper also integrates the concept of Software Defined Networking (SDN) in ADS to effectively control and manage the routing of the data item in the ADS. The storage of data items in the ADS is done based on the type of data (structured or unstructured), the capacity of the distributed system (or coach) and the distance of coach from the pilot computer (PC). In order to maintain the consistency of data and to eradicate the possible loss of data, the concept of "*forward positive*" and "*backward positive*" acknowledgment is proposed. Furthermore, we have incorporated "*Twofish*" cryptographic technique to encrypt the big data in the ADS. Issues like "data ownership," "data security, "data privacy" and data reliability" are pivotal while handling the big data. The current paper presents a framework for a privacy-preserved architecture for effectively handling the big data.**

Keywords— **ADS; r-train; SDN; Twofish; OAuth 2.0; PC; DC; coach**

I. Introduction

This paper aims at providing a refinement over the classical ADS proposed by [1]. The core area of refinement deals with the security, reliability, and availability of the big data. ADS make use of the "r-train" and "r-atrain" data structures for organizing homogeneous and heterogeneous big data respectively [1]-[3]. The work in [1] presented the architectural overview of ADS and had explained how big data elements could be added, deleted or modified in the distributed system. It also discussed and presented two novels "network topologies" exclusively for big data by the name "*Multi-Horse Cart Topology*" and "*Cycle Topology.*" However, the work did not discuss the security and availability aspects of the big concerned data. The fundamental concern about big data is its end-to-end security and immediate availability as and when required by the users.

Furthermore, since the data in the classical ADS is stored in linked distributed systems at multiple sites, there are obvious chances of network failure leading to data loss or node unavailability. This paper addresses these important issues of the classical ADS by providing a secured framework for big data storage and retrieval in an efficient way. The proposal incorporates security features considering the issues like data ownership, reliability and privacy protection of the end users.

II. Material and Method

The current work is divided into four (4) sections. Section II talks about preliminaries and background of classical ADS. The brief Architecture of ADS and related topologies are explained here. The section also provides a brief description of the three prominent "distributed file systems" viz. "Quantcast File System (QFS)," "Cassandra File System (CFS)" and "Hadoop Distributed File System (HDFS)." Finally, some of the latest works in the area of big data security, storage, and retrieval mechanisms are discussed here. It also provides the details about the security issues in the classical ADS and suggests measures to overcome these issues. Section III presents the detailed description of the proposed PPS-ADS architecture. The various components of the PPS-ADS framework and their working details are also provided here. The questions like "how to store the data" and "how to retrieve the data" in a secured, effective and efficient manner are answered here. Section IV presents the conclusion of the paper wherein the effectiveness of the proposed Secured ADS has been highlighted.

A. Preliminaries about the Distributed File Systems for Handling Big Data

This section presents a brief introduction to some of the prominent distributed files systems for handling big data including QFS, CFS, HDFS, and ADS. Furthermore, some of the recent developments in the area are also highlighted here in the form of related works. Before proceeding for the actual work, we need to present some necessary preliminaries from [1] in brief about the “Atrain Distributed System (ADS)” and the corresponding network topologies for ADS like “'Multi-horse Cart Topology” and “Cycle Topology.”

1) A Brief Introduction of ADS: A distributed computer system may be defined as a collection of two or more independent computers, which may be present in same or different locations and are connected via some medium. This medium can be a network of wired or wireless connections created with the help of a middleware. The main aim of the distributed system environment is the optimal sharing of available resources among all the participating entities to give a perception of a single working unit.

The author in [1] proposed a new kind of distributed system, which consists of a single and unique “Pilot Computer (PC)” connected to ‘k’ number of other computers, termed as “Distributed Computers (DC)” and are named as “DC_1, DC_2, DC_3... DC_k”. ADS is a highly scalable system which allows the addition of more computers as and when required (but only at the end) in breadth and depth both. There is connectivity from the PC to every DC. In addition, all the DCs are connected to each other using unidirectional or bidirectional connections. There is a provision wherein the last DC of the breadth level can be connected to the first DC to make it a circular structure. A distributed system following the above-stated connectivity structure is termed as an “Atrain Distributed System (ADS)” [1]. Fig. 1 given below depicts the structure of a typical Uni-tier ADS [1] and Fig. 2 gives the structure of a Two-Tier ADS [1]. The nondirectional arrows (connections) between the DCs depicts that there can be unidirectional or bidirectional connections between them.

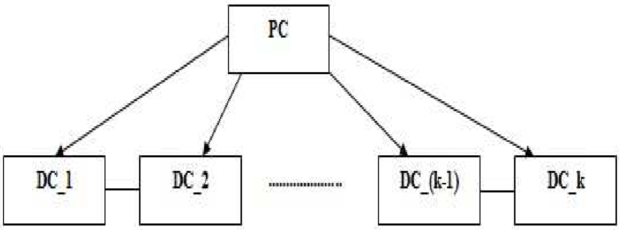

Fig. 1 Uni-tierthe arrangement of PCthe and DCs in the ADS

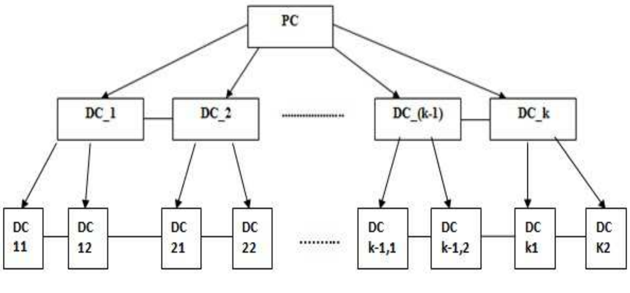

Fig. 2 Two-tier arrangement of PC and DCs in the ADS

2) “Multi-Horse Cart” Topology and “Cycle” Topology: The arrangement of PC and DCs in ADS looks similar to a tree topology wherein the PC acts as the root node and DCs acts as child nodes. However, in reality, they do not follow the definition of typical tree topology. It can be noticed here that this kind of arrangement of participating devices in an ADS does not follow the conventional definitions of “tree/bus/mesh/hybrid” topologies [4]. If the last DC of the ADS stores an invalid address (which signifies that it is the last DC of the ADS), then this type of arrangement of computers (or nodes) is termed as “Multi-horse, Cart” topology [1]. Fig. 3 shows a typical Multi-horse cart topology [1]. Furthermore, if the last DC of the ADS stores the address of the first DC (“making it circular”), then this type of arrangement of computers (or nodes) is termed as “Cycle topology” [1]. Fig. 4 presents a simple cycle topology.

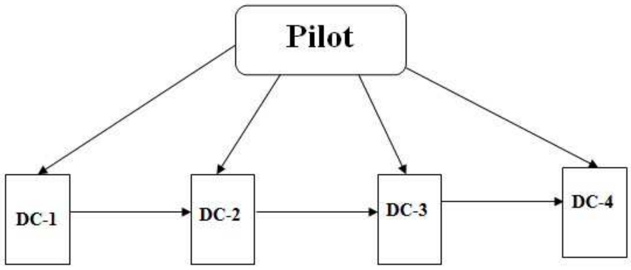

Fig. 3 Multi-horse cart topology with 1 PC and 4 DCs nodes

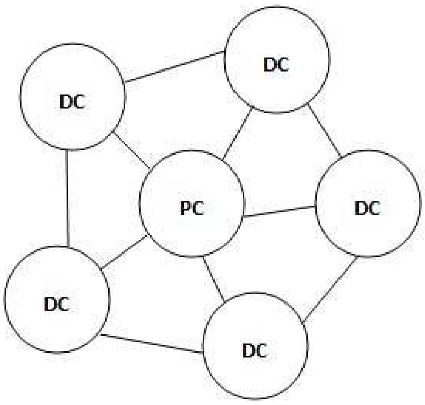

Fig. 4 A Five node cycle topology

Consequently, it can be concluded that a distributed system can be termed an ADS if it follows “multi-horse cart” topology or “cycle” topology explained above. Otherwise, it will not be an ADS. Furthermore, it, can be observed in Fig. 3 that the above combination of DCs and PC gives a uni-tier structure of ADS, and in the same way, we can also have a multi-tier structure for ADS.

For the implementation of ADS, the author of [1] proposed the “r-train” and “r-atrain” data structures for storing homogeneous and heterogeneous data respectively. Here every “r-train/r-atrain” consists of some coaches who are connected to each other via some linked addresses. Also, there is a direct link to each coach with the PC. For further details about “r-train” and “r-atrain” data structures, one could see [1]-[3].

3) Quantcast File System (QFS): QFS is an “open-source” file system, which is used to process big data effectively. QFS is seen as an alternative to the conventional Hadoop system as it outperforms Hadoop in severaa l performance parameters and delivers more results that are efficient. Some of the key features of QFS includes "Reed-Solomon (RS) Error Correction," Higher and faster read/write capacity with faster disk access capabilities," "higher reliability." Further details about QFS can be found in [5]-[6].

4) Cassandra File System (CFS): The CFS was developed to overcome the "single-point-of-failure" problem of the classical Hadoop system and to provide support for the integration of the Cassandra database with Hadoop. CFS follows "peer to peer" structure as compared to Hadoop's "master-slave" structure. CFS is primarily suitable for handling real-time requests. Further details about CFS can be found in these study [7]-[10].

5) Hadoop Distributed File System (HDFS): HDFS is one of the prominent "distributed file system" used for effective handling of big data. HDFS follow master-slave architecture. It consists of two core components. These components are "Namenode" and "Datanodes." The Namenode is the "master," and the Datanodes are the "slaves." The primary task of Namenode is to store the metadata about the data stored in the HDFS. It is also responsible for deciding "data-replication factors." The Datanodes are responsible for storing the big data. The detailed description about HDFS can be found in [11]-[13].

B. Related Works

Extensive research is going around the globe with an aim to find new and improved tools and techniques for "effective handling" of the big data. The researchers in [14] reviewed the various platforms available for performing big data analytics by providing the feature wise evaluation and highlighting the related issues and challenges. The researchers in [15] discussed the storage techniques for big data. They gave the overview of "Cassandra," "MongoDB," "Big tables," "Dynamo" and "Voldemort" technologies that are used for effectively storing big data. In [16], the authors discussed the challenges and issues in storing big data in a secured and "privacy-preserved" manner. The authors in [17] proposed a new prototype named "large capacity storage device" using the "network direct connection storage" device. The authors claim that the performance of their system is dependent on the number of disks in the "NDAS" device. The researchers of [18] introduced and implemented a novel "Preference Aware HDFS (PAHDFS)" system in order to improve the performance of the classical Hadoop system in hybrid environments. The model works by tracking the "read/write" characteristics of the data and optimally chooses the most suitable device by the matching "read/write" capacities of the device. In [19], the authors provided a layered architecture for managing big data wherein the legal issues related to infrastructure (platform), information; IPR ("intellectual property rights"), etc. are described. The researchers in [20] highlighted the "data life cycle" for big data. The primary concerns that were highlighted include privacy and security of the data at rest and in motion. They also highlighted and justified the requirement of third-party trust centers for big data privacy.

The authors in [21] discussed the security threats and challenges of big data and highlighted the need for data anonymizing and encryption. They also presented the access control and governance mechanisms. Finally, some of the best practices were introduced concerning big data security. The author in [23] discussed the challenges and issues in securing big data regarding privacy, legitimacy, and confidentiality. They also highlighted the issue of data ownership and access control mechanisms. The paper [24] highlights the various methods of unstructured big data analytics for catering text, audio, and video formats. The researchers in [25] investigated the various operational and strategic impacts of big data with the help of an extensive review and case study. They further highlighted the future research prospective of big data. The authors of [26] reviewed the correlation between big data and cloud computing. They also highlighted the issues and challenges in big data storage. The researchers in [22] and [27] proposed a secured architecture by the name "Meta Cloud Data Storage" for big data storage on the cloud. The proposed architecture was developed to secure the data and the applications deployed on the cloud. The authors of [28] and [31] proposed a novel architectural model that works by splitting the files into multiple small parts and storing them on distinct storage servers on the cloud. They named their approach as "Security-Aware Efficient Distributed Storage (SAEDS)" model. In [29], the authors highlighted the architecture by the name "Meta Cloud-Redirection (MC-R)" for storing big data generated from the sensors concerning healthcare. This architecture alerts the patients and the doctors in case the vital parameters of the patients exceed the predefined thresholds. The researchers in [30] proposed a framework of "predictive manufacturing cyber-physical system" wherein the big data analysis was carried out to propose improved business decisions. The authors in [32] proposed a "Dynamic Data Encryption Strategy (D2ES)" approach for selectively encrypting the big data using "privacy classification." The researchers of [33] introduced "Dynamic Key-Length-Based Security Framework (DLSeF)" approach for providing "end-to-end" security to the big data. They claim to improve the processing efficiency by reducing the delay in conventional security mechanism. The authors in [34] introduced an improved approach by using "hash algorithm," "weight table" and "sampling method" for visualizing "KDD99" data set. The researchers of [35] provided a secured framework for big data storage and retrieval. Their approach used "Kerberos network authentication protocols" for accessing big data.

C. Security Issues in the Existing ADS

There are several security issues in the current ADS which makes it vulnerable to data theft and leaks. These issues are listed below.

1) The absence of End-to-End Security of Data: In ADS, the data is transferred in its original form from source to destination (and vice-versa) without any encryption. The absence of an appropriate security mechanism makes it vulnerable and brings forth chances that the eavesdropper (if any) can alter the content of the data. This may raise serious concerns in case of highly sensitive data, which must be end-to-end secure and accessible only to the authorized entities.

2) No Authentication of Users: Absence of well-defined authentication mechanism makes it prone to unauthorized access. Without the presence of a proper authentication mechanism, anyone can access the system and cause devastating damages to the system and organizations.

3) Non Availability of Validation and Filtering Mechanism: The current ADS system has no provision for input validation and filtering, which is otherwise critically important due to the emergence of “BYOD (bring your device) model.”

4) No Access Control Policies: ADS lack the tracking of its systems, users and the data being accessed by them. The questions like *“Who is accessing the data from which resource at what time”* are not answered in the existing ADS framework. The proposed version of Secured ADS improves this shortcoming of the existing ADS by incorporating “*Twofish”* cryptographic technique [36-37], “OAuth 2.0” [38-40] authentication mechanism and *“Software Defined Networking (SDN)”* [41-44] framework in the existing ADS system. Twofish is used to encrypt the big data coming for storage in the ADS. OAuth 2.0 framework is used for providing time-based access to the legitimate users and SDN is used for optimal routing of data in the ADS.

5) Data Ownership Issues: Since the data is stored at multiple sites in the distributed systems (DCs), there are chances that any eavesdropper may alter the content of the data without the knowledge of the user or the administrator. This altered data may be harmful for the users or the organization as a whole. With proper authentication mechanism in place, the ownership of the data can be prefixed, and this situation can be controlled. Furthermore, in the case of IoT systems, the pervasive nature of data brings forth critical privacy issues since the data movement and pre-processing is very fast. In addition, the context of the “sensitivity of data” varies from person to person and situation to situation. Therefore, predefined data ownership, role-based access, and time-based authorization must be provided to the users, which is lacking in the current ADS.

6) Data Protection Issue: A Legal Perspective: The term “Data Protection” may be defined as the set of “rules, policies, protocols, procedures and laws” aimed at securing the privacy of an individual or an organization. Any data or information that may reveal the identity or demographics of an individual needs to be protected. The legal issues related to the management of big data include every aspect of big data like storage, processing, analysis, modification, compliance, etc. The questions like “How to legally store and access the data,” “How to legally process and analyze the data,” “Data compliance and liabilities issues” should be addressed in a good big data management system. The techniques like contract documents, privacy by design architecture, consented and informed information collection could prove to be vital. The successful implementation and monitoring of these sections are critical for a secured and privacy-preserved system.

In order to protect the individual’s privacy and sensitive information, techniques like “privacy-by-design” and “Pseudonymization of data” are adopted in recent times. The “accountability,” “liability” and “enforcement” of these rules and regulations are pivotal for providing a secured and privacy-preserved system. The most vulnerable and important phase in which the privacy and security of the information can be compromised is the “information linkage” phase wherein the information from varied participating devices are infused together to deliver a requested service.

In our approach, since the data is end-to-end secured and the users are well authenticated and authorized, the chances of security breaches are at bare minimal.

III. Result and Discussion

The primary objective of this proposal is to overcome the data security, availability and reliability issues in the existing ADS by incorporating new security and authentication features using “Twofish” cryptographic algorithm [36]-[37] and OAuth 2.0 framework [38]-[40]. In addition, in order to optimally route the data to and from the storage destination, Software Defined Networking (SDN) framework [41]-[44] is integrated with the existing ADS. Fig. 5 presents the architecture of the proposed PPS-ADS approach. This section presents the proposed PPS-ADS architecture along with the tools and technologies incorporated in PPS-ADS in order to provide a secure, reliable and robust distributed storage system.

A. PPS-ADS: The Proposed Architectural Framework

The proposed approach can be divided into three parts. The first part deals with the authentication of the users. Here, the users are required to fill in the credentials and based on that; access is granted to them. OAuth 2.0 works behind the authentication mechanism [38]-[40]. The second part deals with the storage of data and the third part deals with the data retrieval. Furthermore, for the sake of understanding, we have the following assumptions in our proposed approach. These assumptions are based on the working of the existing ADS.

- A single coach can store more than one copy of the same data element in different data blocks.
- Each coach has a predefined capacity called by cardinality (the total number of data elements that can be stored in a coach at any point in time).
- Each coach stores the information about the number of empty spaces in it.

B. Components of the Proposed PPS-ADS

There are five core components in the secured ADS architecture. These are listed below:

1) Pilot Computer (PC): The Pilot Computer (PC) is the main control unit of the proposed PPS-ADS architecture. It is responsible for taking vital decisions like “where to store the data”, “how to store the data”, “how many copies of the data to be created”, “where to store the copies of the data”. Furthermore, it stores all the metadata about the data and coaches of the “r-train” or “r-atrain” [1]-[2].

2) User Interface: The end user interacts with this unit. Whenever a user wants to store or retrieve the data, the user interface appears. The user interface authenticates the user and provides access to the system only to the legitimate users. It makes use of OAuth 2.0 framework for providing authorization (valid authorization codes) for a time-based access to the legitimate users.

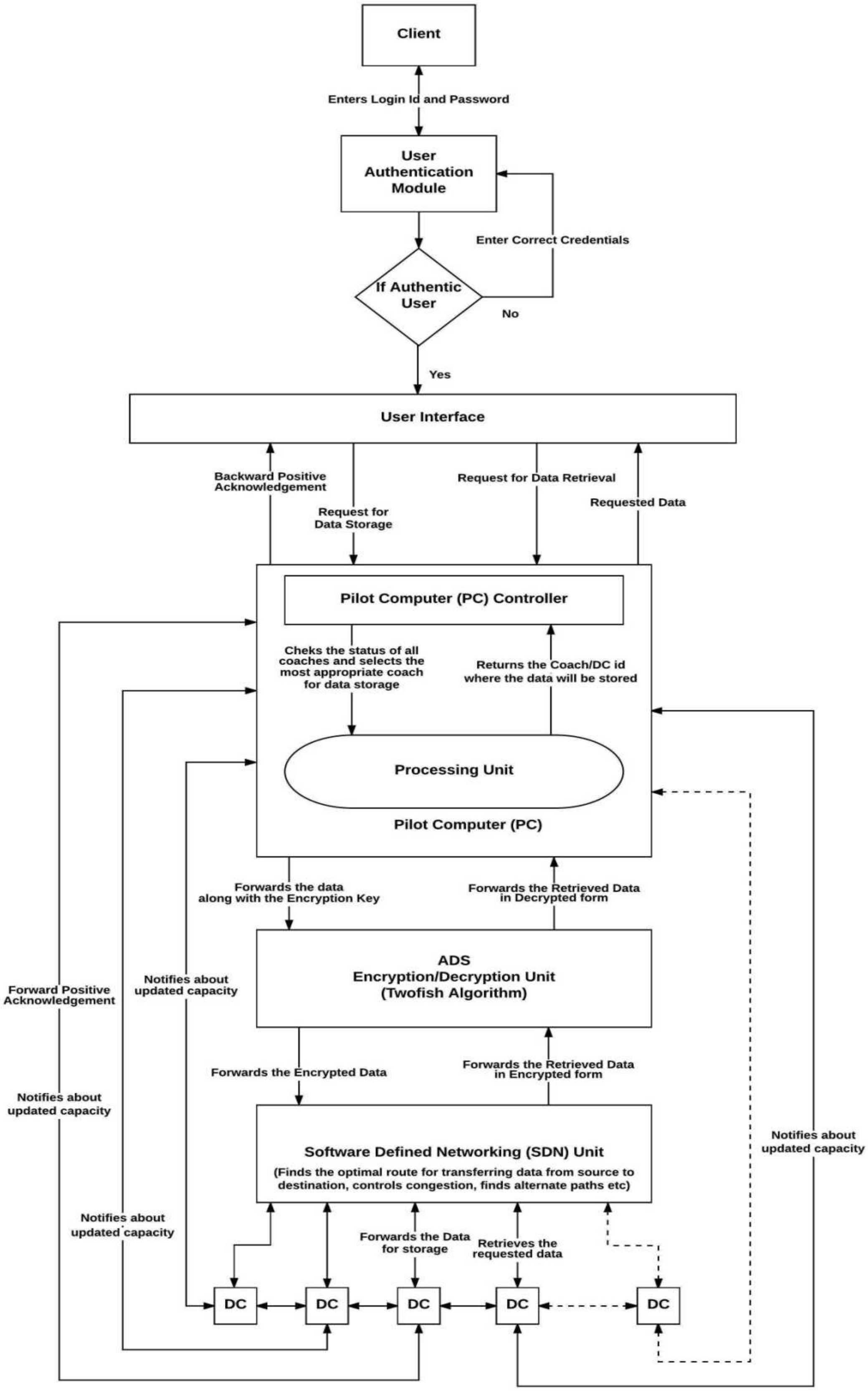
Client
Enters Login Id and Password
User Authentication Module
Enter Correct Credentials
If Authentic User
No
Yes
User Interface
Backward Positive Acknowledgement
Request for Data Storage
Request for Data Retrieval
Requested Data
Pilot Computer (PC) Controller
Cheks the status of all coaches and selects the most appropriate coach for data storage
Returns the Coach/DC id where the data will be stored
Processing Unit
Pilot Computer (PC)
Forwards the data along with the Encryption Key
Forwards the Retrieved Data in Decrypted form
ADS Encryption/Decryption Unit (Twofish Algorithm)
Forward Positive Acknowledgement
Notifies about updated capacity
Notifies about updated capacity
Notifies about updated capacity
Notifies about updated capacity
Notifies about updated capacity
Forwards the Encrypted Data
Forwards the Retrieved Data in Encrypted form
Software Defined Networking (SDN) Unit
(Finds the optimal route for transferring data from source to destination, controls congestion, finds alternate paths etc)
Forwards the Data for storage
Retrieves the requested data
DC
DC
DC
DC
DC

Fig. 5 Architecture of the proposed PPS-ADS approach

3) Encryption and Decryption Unit (EDU): The EDU make use of Twofish cryptographic technique in order to encrypt and decrypt the data. Twofish is an "open-source" cryptographic technique proposed by "Bruce Schneier et al." [36-37]. It is a 128-bit "symmetric-key" based block cipher which is capable of accepting variable length keys ranging from 128-bits up to 256 bits. The Twofish algorithm follows a Feistel network structure. The core components in Twofish algorithm includes "*Input/output whitening, S-Boxes, MDS (Maximum Distance Separable) matrices, PHT (pseudo-Hadamard Transform) and Addition mod 2^{32}*" [36]-[37].

For encrypting the big data, the input is divided into 128-bits blocks and is passed through the above components and the operations like XOR, left, and right rotations, interchanging of bits are performed on the input to give the encrypted data as the output. Further details about the working of the Twofish algorithm can be found in [36]-[37].

4) Software Defined Networking (SDN) unit: SDN is an approach for creating a dynamic network wherein the "control plane" is separated from the "data plane." With SDN, a programmable network can be created (using simple high-level language programming constructs) that interacts with the underlying data plane devices. There are primarily three main components in SDN namely SDN controller, southbound API, and northbound API. Fig. 6 shows the architecture of the SDN framework [41]-[44].

SDN Controllers: They are termed as the brain of the network. They are responsible for deciding the routes of the data items from source to destination, managing network topologies, congestion and flow control, packet forwarding, discarding or blocking, scaling the network as and when required. They give the flexibility to the user to control the commodity network devices (like routers, switches, hubs) by using simple programming constructs making the overall network more agile, scalable and dynamic [41]-[44].

Southbound APIs: They are used facilitate communication between SDN controller and the devices (routers, switches, etc) in the lower infrastructure layer. Primarily Open Flow is used as an interfacing protocol [41]-[44].

Northbound APIs: SDN makes use of northbound APIs to interact with the applications and business logic. These APIs facilitates network administrators to dynamically control the network traffic, network topologies and deploy services using simple programming constructs [41]-[44].

SDN plays a vital role in PPS-ADS by selecting the most appropriate path for the transfer of data and requests from one component to the other in the system. It dynamically controls the network traffic and congestions in real time to select the optimal route for data transfer. The network devices in SDN simple work as a forwarding device and the decisions like when to transfer, how to transfer, where to transfer the data is taken by the SDN controller (administrator) as opposed to conventional firmware architecture which has fixed path for data transfer hardcoded into the network devices itself [41]-[44]. Therefore the absolute control of network configuration and management lies directly in the hands of the administrator. This makes PPS-ADS a dynamic distributed system which is self-reliant in handling complex network issues. Another advantage of using SDN is that is can help reduce the overall operational costs and improve the efficiency since with SDN it becomes easier to diagnose and rectify the network faults and errors [41]-[44].

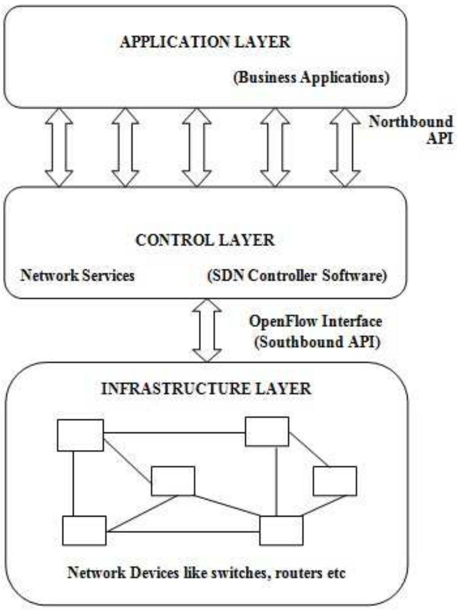

Fig. 6 Architecture of SDN

5) Distributed Computers (DC): They are the computers wherein the actual big data is stored in a distributed manner. These DCs are placed at multiple sites and are connected via a linked address. On receiving the message from the pilot computer regarding the inquiry for their status, the DCs respond to the PC by sending notifications about their updated capacities with every insertion or deletion of the data elements.

C. Working of PPS-ADS

The following section describes the working of the proposed PPS-ADS framework. More specifically, it describes how the data is stored in and retrieved from the system. Furthermore, the concept of maintaining data authentication, authorization, reliability, and availability in PPS-ADS is also discussed here.

1) Data Storage in PPS-ADS: Whenever a new data comes to the pilot computer (PC) for storage, the pilot computer decides in which coach computer (DC) it will be stored. This is to be done by the following three parameters:

- Type of data (structured or unstructured)
- The capacity of coach/DC
- The distance of coach/DC from the Pilot Computer (PC)

Once the data arrives for storage, the pilot computer (PC) checks the status of all the coaches in the DC. It (PC) then decides the coach in which this data will be stored. The decision is taken by the processing unit of the PC and the SDN controller by the distance of the coach from the PC (to make faster data transfer) and the capacity of the coach. Since there are multiple routes for reaching the same destination, the SDN controller selects the optimal route for the data transfer. This selection of route is made dynamically in real time. After the selection of the coach, the data is forwarded to the ADS Encryption-Decryption unit, where it is encrypted using the *"Twofish"* block cipher technique. There are several reasons for the adoption of Twofish cryptographic technique in our proposed approach for performing the encryption and decryption of the data items [36]-[37]. These are listed below:

- Simple and crisp design.
- Support for variable length key sizes (from 128 to 256 bits).
- Highly Optimized for 32 bits CPU.
- Open source.
- Very fast and secure.
- Compatible with smart cards, microprocessors, and other dedicated hardware.
- Supports all standard modes.
- Can be implemented on a wide variety of platforms.
- Allow various performance tradeoffs on encryption-speed, hardware gate count and memory usage.

ALGORITHM I
ALGORITHM FOR INSERTION OF NEW DATA ITEM IN THE SECURED ADS

```
Algorithm 1: Data Storage in PPS-ADS
The various steps involved in data storage are given below:
Input: Data Item
FOR each new data element, DO
    Check the status of all the Coaches/DCs
    IF the Status of the DC/Coach is not Full
        Select the nearest DC/Coach whose status is not Full
        Encrypt the data element
        Insert the encrypted data element in selected DC/Coach
        Update the status of the DC/Coach
    ELSE
        Create a new DC/Coach
        Encrypt the data element
        Insert the encrypted data element in this DC/Coach
        Update the status of this newly created DC/Coach
Output: Success/Failure Notification
```

2) Data Reliability in PPS-ADS: To perform secure transfer and reliable delivery of data, we introduce two types of acknowledgments. On successful arrival and storage of the data item in a coach, the coach sends an acknowledgment to the PC. This type of acknowledgment is known as "forward-positive acknowledgment." On the other hand, the PC in-turn acknowledges the client about the successful storage of data. This type of acknowledgment is known as "backward-positive acknowledgment". These acknowledgments have to reach their respective destination within a stipulated amount of time, otherwise, the data is assumed to be corrupted, and the client is required to resend the data. This 'stipulated time' is not prefixed but depends upon the type of data being transferred. Along with the acknowledgment about the successful storage of data, the coach (wherein the data is being stored) also sends the information about its updated capacity, i.e. about its own latest status (number of empty data-blocks in it).

Furthermore, by incorporating SDN, PPS-ADS ensures a highly agile, fast and effective transfer of data from source to destination and vice-versa. With SDN, the control of information flow, routing and congestion-control directly lies in the hands of the PC. In a conventional network, the switches present in the network are responsible for routing the data packets. These switches always route the data packets going to the same destination along the same route irrespective of the type of data being transferred [4], [41]-[44]. The routing decision in switches is being taken by rules built into the switch's "proprietary firmware" [4], [41]-[44]. However, when SDN is used, this routing can be controlled in real time by using appropriate coding constructs. Thus the PC can have greater control over the routing of the data packets in the proposed PPS-ADS approach.

3) Data Availability in PPS-ADS: To maintain high availability of the data at all times, the data-blocks in the coaches are replicated and are stored at different sites (locations). This process is known as data replication [11]-[13]. The purpose of this data replication is to keep the system up and running in cases of any data losses or network failure. The replication ensures all time availability of data even if one or more coaches are failed. The information about the replication is stored in the PC and the respective DCs (coaches). The PC decides about the exact location for the data to be stored upon receiving the status report from each coach of the DCs.

The data in the coaches are stored in the form of data-blocks, which is analogous to Hadoop [11]-[13]. The deviation in our method is that the data blocks can be of variable length here. Also, each data-block consists of three essential header fields which are helpful in retrieving the stored data at later stages. Fig. 7 shows the fields of the typical data-block in the proposed PPS-ADS architecture and Fig. 8 shows a pure Coach/DC with multiple data-blocks (of variable sizes).

Coach_Id	Coach_Sub_Id	Block_Id	Data Item	Length	Offset

Fig. 7 A simple data-block

The various fields of Fig. 7 are:

Coach_Id: The Id of the coach in which the data is stored
Coach_Sub_Id: Id of the coach, wherein the replication of the data item is stored
Block_Id: The Id of the block wherein the data is being stored inside a coach
Length: The size or length of the data item
Offset: an integer value that gives the distance with respect to the base address in the block.
The Coach_Id, Coach_Sub_Id, and Block_Id in the proposed ADS take two bytes each.

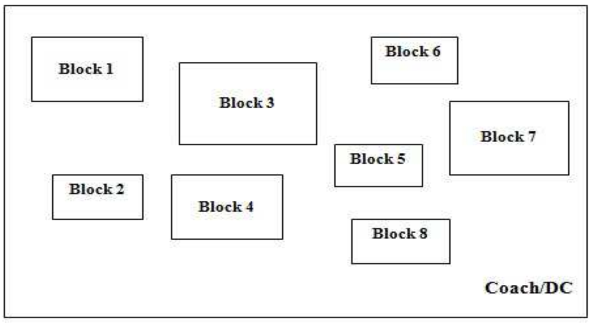

Fig 8. A Coach/DC with variable size data-blocks

If the Coach_Id and Coach_Sub_Id are same for any data block, it means that the particular coach itself has multiple copies of the same data element. The coach consists of the information about the data element which is stored in it and in which block it is stored. In addition, it has the information about the number of empty blocks in it.

In order to ensure all time availability of the data, we proceed as follows:

All the DCs send information about the data stored in their respective data-blocks to the PC after a prefixed quantum of time. The PC analyses this information and decides on the number of copies to be created for any data block. At any point in time, there must be at least three copies of a particular data item to ensure its high availability, which is in fact, analogous to Hadoop [11]-[13]. This replication ensures data availability at all times irrespective of any failures as the same data is being stored at multiple locations in different DCs. In case, any DC goes down or fails (temporarily or permanently), the client's requests can be served by any other DC consisting of the relevant information about the request. Also, any update on the data of any DC gets reflected in its copies after a prefixed quantum of time.

- *User Authorization and Authentication in PPS-ADS:* The user authorization and authentication in PPS-ADS are governed by OAuth 2.0 framework [38]-[40]. Whenever the user requests for using PPS-ADS system, they are required to fill in their credentials on the PPS-ADS User interface. As soon as the user enters the credentials, it gets crosschecked and verified by the system by matching it with the credentials already stored in the system. If there is a match, the system assigns a secure login token to the user. This token is a time-based limited-access grant to the user. That is access taken is valid only for a limited period, and the user can use only the allowed services of the system up to this period only. After the period expires, the system prompts the user to request for re-issuing the token or otherwise the access to the system is revoked. This kind of authorization and authentication mechanism is generally used in cases where access needs to be granted to third part service from within the system or outside the system. The formal steps involved in OAuth 2.0 are given below [38]-[40].

- The user request for an access token from the system
- The system identifies the User by its credentials
- If found authentic, the system issues an Access token to the user based on the credential details (level of access provided)
- Once the user gets the Access token, they can use the system by providing their access token. (This access token is a kind of time-bound pass that enables the user to use only the allowed services of the system with this period)
- As soon as the period of the access token expires, the system prompts the user to request for re-issuing it.
- If the user wants to use the system further, it can request for re-issue of the access token or extend the time-period of the current token.
- The system analyses the request from the user and if deemed fit, extends the period of the current access token or instructs the user to request for a fresh access token.
- Once the user finished working on the system before the expiry of the access token, it notifies the system
- The system for receiving this request from the user revokes the access grant and closes the connection.

4) Data Encryption in PPS-ADS: The Encryption-Decryption unit of the proposed approach utilizes the concept of Twofish cryptography algorithm [36]-[37]. The following steps are adopted:

- Divide the input from the pilot computer into data blocks of 128 bits.
- Each of these 128-bit blocks is further divided into four Segments S1, S2, S3, and S4 of 32-bits each.
- These parts (S1, S2, S3, S4) along with the key 'K' are fed as the input to the encryption unit (where the length of the key can be between 128 to 256 bits).
- The Key "K" is also divided into four parts K1, K2, K3, K4 and is XORed with S1, S2, S3 and S4 respectively. This process is known as input whitening.
- Similarly, other steps of the Twofish cryptographic techniques are followed to get the final data in encrypted format. The detailed working of the Twofish can be found in [36]-[37].

After all the blocks of input data are encrypted, they are merged to get the final encrypted data, which is then passed to the DCs for storage. For the decryption of the data items, the above steps are performed in reverse order. Fig. 9 shows the building blocks of the encryption technique using the Twofish algorithm [36]-[37].

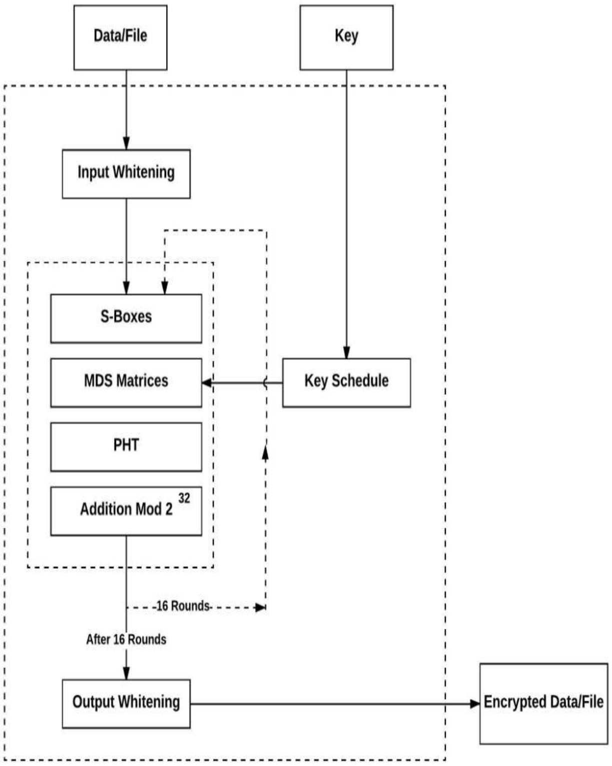

Fig. 9 Building blocks of Twofish cryptographic technique

5) Data Retrieval in PPS-ADS: Whenever the request for retrieval of any data item comes to the Pilot Computer (PC), the PC broadcasts a message to all the connected DCs/ Coaches, requesting the location of the DC/Coach where the data in question is stored. On receiving the reply from the DCs/Coaches, it selects the nearest DC where the data item is stored. After fetching the data item, it is decrypted to convert it into its original format and finally sent to the user. The steps for data retrieval are given below:

ALGORITHM II
ALGORITHM FOR RETRIEVAL OF DATA ELEMENT FROM THE COACH/DC

```
Algorithm 2: Data Retrieval in PPS-ADS

Input: Request for Data Item

1.  FOR every Retrieval request, DO
2.      Find the location of the DC/Coach wherein the data is stored
3.      Find the location of the nearest Data-Block (in a DC/Coach)
        wherein the requested data item is stored
4.      Fetch the data Item from that Data-Block
5.      Decrypt the data item
6.      Send the data item to the client
Output: Requested Data Item
```

D. Discussions

The proposed PPS-ADS is a robust, reliable, secured and privacy-preserved architecture, which is capable of handling gigantic datasets. The proposed PPS-ADS overcomes the limitations and shortcomings of the conventional ADS in a well-coordinated and effective manner. The concept of 'forward positive acknowledgments and 'backward positive acknowledgments' was serving the purpose of data reliability and consistency while 'Twofish' acts as a lightweight cryptographic technique to ensure security and privacy of the user and the data. In order to implement PPS-ADS the data structures r-train and r-atrain are used [1]-[3]. The various network components in PPS-ADS follow Multi-Horse-Cart topology.

IV. CONCLUSIONS

With the enormous amount of data being generated at such fast pace, there is a need for optimized tools and techniques for its secure and synchronized storage and retrieval as there could be chances of data being lost in between. ADS is an appealing architecture to handle the big data as it can physically support both structured and unstructured big data using "r-train" and "r-atrain." However, the classical ADS in its current form do not have a security mechanism in place for the data being stored. In this work, a privacy-preserved and secured architecture is proposed. The proposal uses the concepts of Twofish, OAuth 2.0 and SDN to incorporate security, reliability and availability features in the existing ADS architecture. The Twofish algorithm facilitates efficient encryption and decryption of the data ensuring a secured data storage and retrieval. OAuth 2.0 framework ensures authorized access to the system by issuing authorization codes to the users. The SDN controller efficiently delivers the data items to the destination by optimally controlling the flow, congestion, and routing of the individual data items. Another advantage of the PPS-ADS approach is that the block sizes here are dynamic and depends on the "size of the data item" to be stored. Consequently, this technique does not suffer from the issues of "small size files handling" problem like in classical Hadoop.

Finally, it can be concluded that every such software or system which deals with the private and sensitive information of the individuals or an organization must identify the primary (direct threats) and secondary (indirect/ third party) privacy threats and should implement all the possible and feasible security and privacy-preserving mechanism related to the context for which the data or information is stored in these systems. The context of data storage and protection is important because the privacy and security of data and information in one context may not be relevant to the other context and vice-versa.

REFERENCES

[1] R. Biswas, Atrain distributed system (ads): An infinitely scalable architecture for processing big data of any 4V, Computational Intelligence for Big Data Analysis Frontier Advances and Applications: edited by D.P. Acharjya, Satchidananda Dehuri and Sugata Sanyal, Springer International Publishing. Switzerland. 2015, 3-53.

[2] R. Biswas, r-train (train): A new flexible, dynamic data structure, INFORMATION: An International Journal (Japan) 14 (4) (2011) 1231-1246.

[3] R. Biswas, Heterogeneous data structure r-atrain, INFORMATION: An International Journal (Japan) 15 (2) (2012) 879-902.

[4] B. A. Forouzan, Data Communication, and Networking, 4th Edition, McGraw-Hill, 2007.

[5] Ovsiannikov M, Rus S, Reeves D, Sutter P, Rao S, Kelly J. The Quantcast file system. Proceedings of the 39th International Conference on Very Large Scale Databases (VLDB Endowment); Trento. 2013. p.1092-1101.doi:10.14778/2536222.2536234

[6] Ahad, M. A, & Biswas, R. (2017). Comparing and Analyzing the Characteristics of Hadoop, Cassandra and Quantcast File Systems for Handling Big Data. Indian Journal Of Science And Technology, 10(8). doi:10.17485/ijst/2017/v10i8/105400

[7] Lakshman A, Cassandra MP. A decentralized structured storage system. Proceeding of ACM SIGOPS Operating Systems; USA. 2010. p. 35-40.

[8] Dede E, Sendir B, Kuzlu P, Hartog J, Govindaraju M. An Evaluation of Cassandra for Hadoop. IEEE Proceeding of 6th International Conference on Cloud Computing; USA.2013. p. 494-501. doi:10.1109/cloud.2013.31

[9] Dr. Kalpesh U. Gundigara , Ms. Vibha H., Mehta, (2017), Cassandra as a Big data Modeling Methodology for Distributed Database System, International Journal of Engineering Development and Research(IJEDR), Volume 5, Issue 3, pp 957-965.

[10] Comparing the Hadoop Distributed File System (HDFS) with the Cassandra File System (CFS), White Paper, By Datastax Corporation. 2016. Available from: https://www.datastax.com/wp-content/uploads/2012/09/WP-DataStaxHDFSvsCFS.pdf

[11] Shvachko K, Kuang H, Radia S, Chansler R. The Hadoop distributed file system. IEEE Proceeding of the 26th Symposium on Mass Storage Systems and Technologies (MSST); USA. 2010. p. 1-10.

[12] Borthakur D. HDFS Architecture Guide, Apache Foundation.2016. Available from: https://hadoop.apache.org/ docs/r1.2.1/hdfs_design.pdf

[13] Gupta L. HDFS – Hadoop Distributed File System Architecture Tutorial. Available from: http://howtodoinjava.com/big-data/hadoop/hdfs-hadoop-distributed-file-systemarchitecture-tutorial/

[14] D Singh and C K Reddy, A survey on platforms for big data analytics, Journal of Big Data, 2014, 1:8
http://www.journalofbigdata.com/content/1/1/8

[15] M. Chen et al., Big Data: Related Technologies, Challenges and Future Prospects, Springer Briefs in Computer Science, 2014. DOI 10.1007/978-3-319-06245-7__4

[16] Martin Strohbach, Jorg Daubert, Herman Ravkin, and Mario Lischka, Big Data Storage, Chapter 7, J.M. Cavanillas et al. (eds.), New Horizons for a Data-Driven Economy 2016.

[17] J.K. Park, J. Kim, Big data storage configuration and performance evaluation utilizing NDAS storage systems, AKCE International Journal of Graphs and Combinatorics (2017), https://doi.org/10.1016/j.akcej.2017.09.003.

[18] Wei Zhou, Dan Feng, Zhipeng Tan, Yingfei Zheng, Improving Big Data Storage Performance in Hybrid Environment, Journal of Computational Science, (2017) http://dx.doi.org/10.1016/j.jocs.2017.01.003

[19] R. Kemp, Legal aspects of managing big data, Computer Law and Security Review, 30 (5) (2014), pp. 6482-491. Elsevier. https://doi.org/10.1016/j.clsr.2014.07.006

[20] Jakóbik A. (2016) Big Data Security. In: Pop F., Kołodziej J., Di Martino B. (eds) Resource Management for Big Data Platforms. Computer Communications and Networks. Springer, Cham

[21] Guillermo Lafuente, The big data security challenge, Network Security,Volume 2015, Issue 1, 2015, Pages 12-14, https://doi.org/10.1016/S1353-4858(15)70009-7.

[22] Gunasekaran Manogaran, Chandu Thota, M. Vijay Kumar, MetaCloudDataStorage Architecture for Big Data Security in Cloud Computing, Procedia Computer Science, Volume 87, 2016, Pages 128-133, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2016.05.138.

[23] E. Bertino, "Big Data - Security and Privacy," 2015 IEEE International Congress on Big Data, New York, NY, 2015, pp. 757-761. doi: 10.1109/BigDataCongress.2015.126

[24] Amir Gandomi, Murtaza Haider, beyond the hype: Big data concepts, methods, and analytics, International Journal of Information Management, Volume 35, Issue 2, 2015, Pages 137-144, ISSN 0268-4012, https://doi.org/10.1016/j.ijinfomgt.2014.10.007.

[25] Samuel Fosso Wamba, Shahriar Akter, Andrew Edwards, Geoffrey Chopin, Denis Gnanzou, How ‘big data’ can make big impact: Findings from a systematic review and a longitudinal case study, International Journal of Production Economics, Volume 165, 2015, Pages 234-246, ISSN 0925-5273, https://doi.org/10.1016/j.ijpe.2014.12.031.

[26] Ibrahim Abaker Targio Hashem, Ibrar Yaqoob, Nor Badrul Anuar, Salimah Mokhtar, Abdullah Gani, Samee Ullah Khan, The rise of “big data” on cloud computing: Review and open research issues, Information Systems, 2015, Volumes 98-115, ISSN 0306-4379, https://doi.org/10.1016/j.is.2014.07.006.

[27] Chandu Thota, Daphne Lopez, Gunasekaran Manogaran, Vijayakumar V, Chapter 12, Big Data Security Framework for distributed cloud data centers, Cybersecurity Breaches and Issues Surrounding Online Threat Protection ed. Moore, Michelle,pp 288-310, 2017, IGI Global

[28] K. Gai, M. Qiu and H. Zhao, "Security-Aware Efficient Mass Distributed Storage Approach for Cloud Systems in Big Data," 2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), New York, NY, 2016, pp. 140-145. doi: 10.1109/BigDataSecurity-HPSC-IDS.2016.68

[29] Manogaran G., Thota C., Lopez D., Sundarasekar R. (2017) Big Data Security Intelligence for Healthcare Industry 4.0. In: Thames L., Schaefer D. (eds) Cybersecurity for Industry 4.0. Springer Series in Advanced Manufacturing. Springer, Cham

[30] Radu F. Babiceanu, Remzi Seker, Big Data and virtualization for manufacturing cyber-physical systems: A survey of the current status and future outlook, Computers in Industry, Volume 81, 2016, Pages 128-137, ISSN 0166-3615, https://doi.org/10.1016/j.compind.2016.02.004.

[31] Yibin Li, Keke Gai, Longfei Qiu, Meikang Qiu, Hui Zhao, Intelligent cryptography approach for secure distributed big data storage in cloud computing, Information Sciences, Volume 387, 2017, Pages 103-115, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2016.09.005.

[32] K. Gai, M. Qiu, H. Zhao and J. Xiong, "Privacy-Aware Adaptive Data Encryption Strategy of Big Data in Cloud Computing," 2016 IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud), Beijing, 2016, pp. 273-278. doi: 10.1109/CSCloud.2016.52

[33] Deepak Puthal, Surya Nepal, Rajiv Ranjan, and Jinjun Chen. 2016. DLSeF: A Dynamic Key-Length-Based Efficient Real-Time Security Verification Model for Big Data Stream. ACM Trans. Embed. Comput. Syst. 16, 2, Article 51 (December 2016), 24 pages. DOI: https://doi.org/10.1145/2937755

[34] Zichan Ruan, Yuantian Miao, Lei Pan, Nicholas Patterson, Jun Zhang, Visualization of big data security: a case study on the KDD99 cup data set, Digital Communications and Networks, Volume 3, Issue 4, 2017, Pages 250-259, ISSN 2352-8648, https://doi.org/10.1016/j.dcan.2017.07.004.

[35] P. Johri, A. Kumar, S. Das and S. Arora, "Security framework using Hadoop for big data," 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, 2017, pp. 268-272.
doi: 10.1109/CCAA.2017.8229813

[36] Bruce Schneier, John Kelsey, Twofish: A 128-bit block cipher, AES Round 1 Technical Evaluation CD-1: Documentation, National Institute of Standards and Technology.

[37] B. Schneier, J. Kelsey, N. Ferguson, The Twofish Encryption Algorithm, A 128-Bit Block Cipher, John Wiley & Sons, 1999.

[38] https://oauth.net/2/

[39] Ryan Boyd, Getting Started with OAuth 2.0 2012, Published by O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

[40] Argyriou M., Dragoni N., Spognardi A. (2017), Security Flows in OAuth 2.0 Framework: A Case Study. In: Tonetta S., Schoitsch E., Bitsch F. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2017. Lecture Notes in Computer Science, vol 10489. Springer, Cham

[41] S. A. Diego Kreutz, Fernando M. V. Ramos, S. Uhlig, Software-defined networking: A comprehensive survey, Proceedings of the IEEE 103 (1) (2015) 14-76.

[42] D. B. Rawat and S. R. Reddy, "Software Defined Networking Architecture, Security and Energy Efficiency: A Survey," in IEEE Communications Surveys & Tutorials, vol. 19, no. 1, pp. 325-346, Firstquarter 2017. doi: 10.1109/COMST.2016.2618874

[43] W. Braun, M. Menth, , Software-defined networking using openflow: Protocols, applications and architectural design choices, Future Internet 6 (2014) 302-336.

[44] Software-defined networking: The new norm for networks, Open Networking Foundation (ONF),White Paper (2012) 1-12.