Journal of Information Systems Engineering and Management
2025, 10(5s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Improving and Protecting the Privacy of Data Security Using Blake2 Algorithm

Ghsuoon B. Roomi 1 , Intisar N.manea *2, Auhood mukr dayish3
¹R1 Department of Computer, College of Education for Pure Sciences, University of Thi-Qar, Thi- Qar-, Iraq.
GHSUOON.BADR@utq.edu.iq.

2*Department of Accounting Techniques, Thi-Qar technical College, Southern Technical University, Thi-Qar, Iraq.
intisar.neamah@stu.edu.iq

3 Department of Computer, College of Education for Pure Sciences, University of Thi-Qar, Thi- Qar-, Iraq.

Eu-m@utq.edu.iq

ARTICLE INFO

Received: 10Oct 2024

Revised: 12Dec 2024

Accepted:24Dec 2024

ABSTRACT

Several characteristics of data open up many new avenues for exploitation, and as such, new privacy and security models are required to address these emerging challenges. The magnitude of datasets generated make it near impossible for data managers to capture all the contextual semantics relevant to a unit of data, and this causes novel difficulties in privacy and security of these data systems using BLAKE2 Algorithm. At the micro-level, a datum in any software system transitions between several phases, from inception to deletion; this is known as the data life cycle, which provides a concrete guide for tracking the many states of a unit of data. The usefulness of such a model only increases with the complexity of the system that it represents, so it is useful to examine privacy and security from this perspective, as it provides a precise framework for discussion. This paper provides practical privacy and security recommendations for every step of the data life cycle, examining prominent infrastructures and their features that relate to their data management policies using BLAKE2 Algorithm with 98.97% secure and reliable all the time. BLAKE2 is a cryptographic hash function faster than MD5, SHA-1, SHA-2, and SHA-256. The most glaring issue with SHA is that its data storage system is completely unencrypted by default. Like MD5, SHA1, and many encryption algorithms were designed to be run within a trusted environment. In the case of unencrypted data, this means that malicious parties that have physical or virtual access to the file system can extract information as they please. Fortunately, BLAKE2 is largely reliant on the client code for encryption of any sensitive information before it saves to the database. Some mitigation techniques for this scheme include implementing proper file system permissions and file system level encryption.

Keywords: Data, Contextual Privacy, Contextual Security, Data Life Cycle, BLAKE2 Algorithm, Hadoop, MongoDB, NoSQL, Encryption.

INTRODUCTION

Data represents a paradigm shift in computing, with data sets becoming too large, diverse, and unpredictable for traditional computer models to handle effectively. As a result, new computing frameworks are being rapidly developed and adopted, with the focus shifting from centralized platforms to more distributed ones, as discussed in [1]. Of course, big data provides insights into new types of information that are more detailed and extensive than any previous computing paradigm. Big data is great in theory, but in the context of big data, whether intentional or not, there are certainly more opportunities to exploit it.

In a world increasingly reliant on data, it is an ethical obligation for companies to be proactive about privacy and security and understand the potential risks to their customers. In one study, the average stock price loss in the three days following news of a security breach was nearly 5.6%, equivalent to $17 million to $28 million for some companies, as reported in [2]. Therefore, understanding these risks is a good motivator as it is a financially sound investment for the organization. Contextual privacy and

security are fundamental standards that need to be adhered to in big data systems, but can be very difficult in existing implementations. The purpose of this document is to analyze the privacy and security issues at each step of the data life cycle based on the contextual characteristics of the relevant data.

Contributions. This paper makes the following contributions:

- Explores the importance of contextual privacy and security in data systems for data protection.

● Creates a concrete guide with a set of best practices that are of practical use to developers of data systems, which ensures that the privacy and security of users are preserved.

● Identifies existing big data and normal data software systems, and their potential weaknesses in relation to contextual privacy and security and improving their procedures.

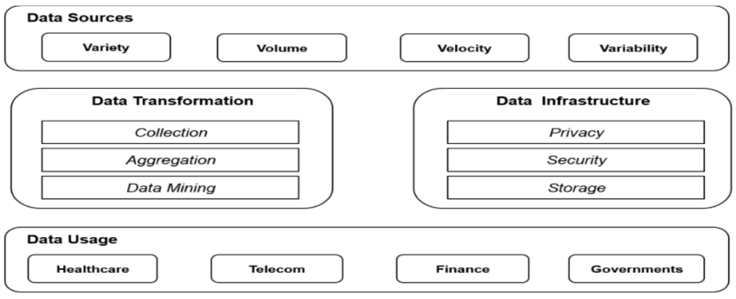

Figure 1. An analysis of privacy and security issues at each step of the data life cycle [3].

1. LITERATURE REVIEW

There have been many suggestions by the data community relevant to the goals of this paper. Some existing solutions that preserve privacy on data systems include:

1.1. Privacy-preserving, attribute-based, fine-grained access control [4].

One way of protecting confidentiality of data is by using encryption. However, this alone may not be sufficient to protect users" privacy, as data handlers often need to enforce things such as identity attributes like user roles, and preferences. These systems are known as attribute-based systems, where fine-grained access control is a requirement, and based on policies specified using identity attributes. The gist of this idea is that the system can process data relative to sensitive information without actually knowing the semantics of the data itself, thereby preserving the confidentiality of certain details.

Researchers provide a context-based publish subscribe model for distributed communication commonly found in big data systems as mentioned in [5]. Their architectural pattern is an improvement on the existing publish-subscribe systems, because it performs an evaluation of blinded subscriptions against blinded notifications without decryption. They do note that there is a hit to routing efficiency.

1.2. Policy languages.

There has been a lot of work in extending rudimentary access control systems, providing responses beyond

a simple “yes/no” [6]; these are known as policy languages, and they provide more expressive power than traditional mechanisms. Some authors believe that policy languages such as XACML represent good solutions for fine granular access control as given in [7]. XACML is a standard for fine-grained access control and authorization systems. It defines a policy language for access control, accompanied by an architecture and a processing model that follows the rules defined in the policies. XACML is based on that principle that authorization features should be based not only on the request context attributes like sub- jects/users, but also on the structured data content. A promising direction is a design that applies attribute based access control mechanism with policies that incorporate data granularity. These policies may contain complex logical expression based on the attributes.

1.3. Access control in NoSQL databases.

The recently popular NoSQL implementations such as Cassandra, MongoDB, and provide varying levels of security and access control as given in [8]. Unfortunately, most implementations have coarse-grained authorization features. Out of the three, however, provides the most support, however it is still at its early development stage, and has poor support for distributed and multi-domain environments as given in [9].

1.4. Open source middleware systems.

Middleware is any software that sits between the machine"s operating system and its applications, while providing services to the latter. Some existing initiatives such as the European Union"s platform, FI-WARE, address privacy in data systems with optional authorization and authentication mechanisms that include a policy language to define the attributes and credentials required to access resources as mentioned in [10]. They also include a data handling policy language and it is typical that it defines how data attributes and credentials are handled, and to whom they are passed on, which is crucial for contextual privacy. In addition, they provide the means to release and verify the given attributes and credentials.

1.5. Encryption for data at rest.

The aforementioned mechanisms are sufficient to ensure no violations of sensitive information occur at several stages of the life cycle, such as the data access, transfer and processing stages as mentioned in [11]. However, the issues with regular access control systems remain: the setup is that a trusted server mediates the access control. Unfortunately, if this trusted server is compromised, then the confidentiality of the data will be as well. These techniques ensure that in the worst-case scenario where a server is compromised, the data itself is ensured to preserve integrity. These schemes are great for data at rest, and is especially relevant to many use cases in big data, especially healthcare or distributed sensor networks. A potential solution to this is to use encryption enhanced access control policies that provide an extra layer on top of traditional access control schemes, i.e. attributes based encryption. These extra features ensure that data decryption only occurs for the targeted subject or attribute owner, which is a sufficient condition for contextual policy protection as mentioned in [12].

1.6. Data Anonymization

There are some suggestions from the community that data anonymization is a potential solution for protecting contextual privacy as mentioned in [13]. The idea is that in the worst-case, if context violations were to occur, the damage would be minimized since the data is obfuscated. Some researchers believe that this path is futile. In their experiments, a mobility dataset of 1.5 million users was collected over a 15-month period and even after obfuscating personal information such as phone numbers, unique IDs, and names, they still managed to identify a person with 95% accuracy using spatial-temporal data points. In another scenario, chunks of the Netflix challenge dataset were re-identified using an external data source (the IMDB database) as mentioned in [14].

2. PRELIMINARIES DATA SECURITY

The concept of big data is quite vague and evolving very rapidly, given the vast amount of research that the area has seen in recent times. As a result, there have been several proposed definitions from the

community since its coinage.

The most widely-cited definition comes from a Meta (now Gartner) report which introduced three prominent V"s associated with big data: volume, velocity, and variety as mentioned in [15]. These terms refer to the increasing size of datasets, the increasing rate at which data is produced, and the increased heterogeneity of data formats, respectively. This classic definition is concrete, however we feel that to a first-time reader, the essence of the idea is not sufficiently communicated. In a survey of definitions, Laney identifies and explores several interpretations of big data as mentioned in [16]. He also recognizes the classic definition however concludes that the majority of big data definitions comprise a mixture of the following elements:

- *Size*, which refers to the volume of the data.
- *Complexity*, which refers to the structure, behavior and permutations of the data.
- *Technologies*, which are the tools and techniques that are used to process large and/or complex datasets.

Laney concludes by extrapolating the aforementioned factors and postulates that big data always describes "the storage and analysis of large and or complex datasets using a series of techniques including, but not limited to: NoSQL, MapReduce and machine learning". We find this definition best captures the essence of big data, however for clarity, and precision, this paper will examine big data against the classic V"s.

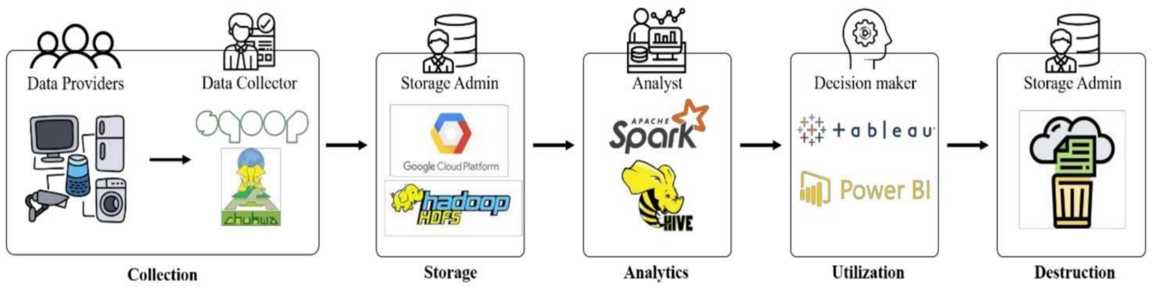

Figure 2. Diagrammatic approach to represent the increased heterogeneity of data formats [16].

2.1. Data Context

The definition we find most intuitive, by [17] describes context as "any information that can be used to characterize the situation of an entity", where an entity is "a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"situation".

The research suggeststhat simply, context is "any information that can be used to characterize the :Furthermore, context can be divided into four categories

1. ***Computing context***, which encompasses things like network connectivity, communication cost, and communication bandwidth.
2. ***User context***, which captures details about the user, such as location, surroundings, and social situation.
3. ***Physical context***, which involves the physical surroundings, e.g. noise, temperature, traffic conditions, etc.
4. ***Temporal context***, which, as it suggests, deals with time,

e.g. the time of day, the current month, the current season, etc.

Researchers in [17] suggests that the relevant user contexts for privacy, are things such as canonical activities, roles, relationships, power structures, norms (or rules), and internal values.

2.2. Data Privacy

One of the first discussions around consumer data privacy and data protection was by Alan Westin, in 1968 who described privacy as "the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others" [18]. Viewed in terms of the relation of the individual to social participation, privacy is the voluntary and temporary withdrawal of a person from the general society through physical or psychological means, either in a state of solitude or small group intimacy or, when among large groups, in a condition of anonymity or reserve as mentioned in [19]. Due to a heavier reliance on computing technologies, the amount of sensitive data collected about individuals has reached unprecedented levels, and will only continue to grow in the future. Naturally, with larger amounts of sensitive data, comes a greater risk for exploits by third parties, which include governments, but also corporations, law enforcement agents, family and friends as well as strangers as mentioned in [20].

2.3. Data Security

Security, on the other hand, is quite different from privacy. Security is defined as the prevention or protection against access to information by unauthorized recipients, or unauthorized (but intentional) destruction or alteration of that information itself as mentioned in [21]. Another concept of security is that it is the ability of a system to protect information and system resources to ensure the preservation of three factors: confidentiality, integrity and availability as mentioned in [22]. These are known as the three pillars of security is defined as follows:

- **Confidentiality** - Unauthorized access of information is prevented
- **Integrity** - Information isn‟t modified in any way by unauthorized users in any way, such that an authorized user will not be able to detect this modification
- **Availability** - Ensuring the availability of a system, should this be violated, it would be said that there‟s a "denial of service" attack taking place.

In some cases, it is argued that authentication, instead of availability is the third pillar of security, however this is argued to be incorrect in many cases as Authentication is technically part of the Confidentiality pillar. It is possible to have a system that is not able to guarantee one‟s privacy despite being mathematically/theoretically "secure", however this is not very common, in many cases a violation of security will violate one‟s privacy as well as potentially sensitive data is now accessible to someone who isn‟t supposed to be able to access it.

2.4. Issues with Data

In the context of data systems, security is something that is pretty tough to ensure. To understand the difficulty of this, we take a look at the 3 V‟s of big data systems: volume, velocity and variety as mentioned in [23].

- **Volume:** In a traditional data system, a handful of machines may be sufficient to process the volume of data. However, big data systems require much larger clusters of computational resources with potentially hundreds of thousands of machines networked across the globe and datasets on the scale of petabytes. For some context, in 2014, Amazon Web Services, one of the premier cloud systems in the world was estimated to have 1.5 to 2 million servers as mentioned in [24]; no doubt, this number is probably much larger now. Figures on this scale were unprecedented in the context of traditional data systems.
- **Velocity:** Because of the increased number of sources of data, there are significantly larger amounts of new data processed every second, day in, day out. Furthermore, the rate at which data is collected is accelerating and has a multiplicative effect on the total volume of data.

For example, consider that in 2014, Amazon AWS data centers were allocated up to 102 Tb/sec of

incoming bandwidth (this data was back in 2014 and it would be fair to say that this has likely increased) and within the data center itself, there is a significantly higher bandwidth. Cumulatively, this velocity of data represents a magnitude of volume at unprecedented levels (even assuming that only 50% of the bandwidth being used) as mentioned in [25]. Ensuring the preservation of security while maintaining this velocity of data transfer in and out is one of the toughest challenges.

- **Variety:** The other thing that should be considered is the variety of the data in big data systems. Unlike traditional data systems, which are generally designed for specific kinds of data, big data systems have data incoming in a variety of formats, from various sources as mentioned in [26]. As a result, security solutions for such heterogeneous datasets need to take into account the variability of the data that the systems handle.

When large scale privacy and security violations occur, the after effects are somewhat similar to what happens in small scale systems, however, the effects are magnified. While in small scale systems, only a small fraction of the population of users may be affected, in big data systems, the number of users affected can run into the millions, or billions as mentioned in [27]. Technologies is a good example in this case, as they were the leading provider for phone services for many of the prisons and jails in the United States as mentioned in [28]. In 2015, there was a massive security breach that violated the privacy of inmates, as over 70 million records of phone calls, complete with links to the recordings of the phone calls itself, spanning a period of slightly over 2 years.

3. CONTEXTUAL PRIVACY AND SECURITY

Contextual security, or context-aware security, is the idea where the context of the data is used to make security decisions as mentioned in [29]. It allows for better, more tailored security decisions to be made. In our context, this allows us to tailor the security measures to be put into place as data moves along the life cycle. Big data systems will require different security protocols given that data will go through many transformations and be distributed between traditional security domains. Contextual privacy to some extent has a similar definition to contextual security; the extent of which is that information under one context should not be shared in the same way as information under a different one. As such, it should be handled as mandated by the necessary legal regulations and organizational policies. The key concern for con- textual privacy is data misappropriation, which is a type of misconduct that is the result of acts that modify the context of data to an inappropriate one as mentioned in [30]. Some researchers suggest that big data systems should include the enforcement of a data policy in order for them to be processed on trusted systems, while complying with other external requirements, i.e., context as mentioned in [31].

4. METHODOLOGY

The data life cycle is an iterative process which describes the phases that a unit of data will pass through, from its inception to the time of destruction. Data life cycle models are important because they provide a rigid structure for considering the various operations a unit of data may undergo throughout its existence.

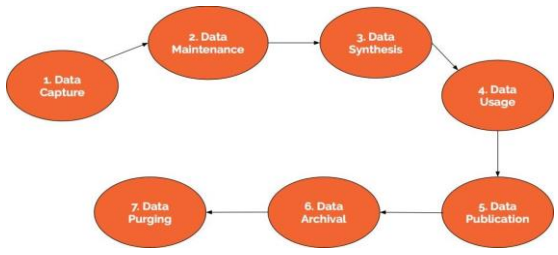

Figure 3. Diagrammatic approach to the data life cycle.

Interestingly, it has been argued that the term "life cycle" is not actually the most appropriate term, as unlike in a regular life cycle of say, a butterfly, data will not reproduce itself to ensure continuity. There have been multiple definitions of the data life cycle stages, with different authors having different views. This has resulted in some cases, different life cycle stages. We feel that the data life cycle proposed by Chisholm is one of the most comprehensive, hence we will use this version of the data life cycle in our paper.

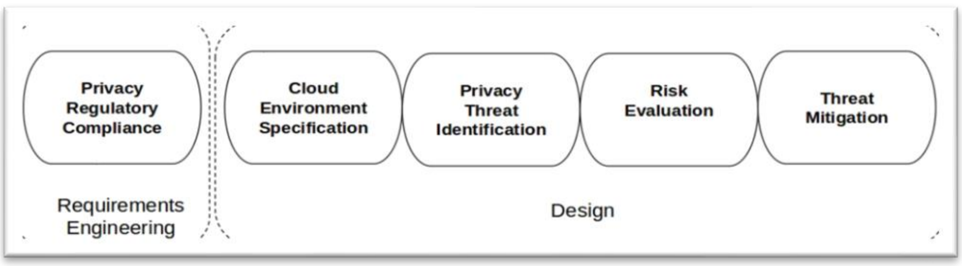
Privacy Regulatory Compliance
Cloud Environment Specification
Privacy Threat Identification
Risk Evaluation
Threat Mitigation
Requirements Engineering
Design

Figure 4. Diagrammatic approach to depict the requirements and designs of system.

4.1. Data Capture using BLAKE2 Algorithm

This is the beginning of the data life cycle, and encapsulates the various processes involved in the creation of a unit of data. There a number of ways this happens in a system:

- ***Data Acquisition***

In this method, existing data from an external system is introduced to the system under consideration. This is still considered as part of the data capture process as in the past, it didn"t exist in our system; instead it only existed in an external system. This is common especially when migrating data from one system to another, or when an organization forms a joint-venture/is acquired by another.

- ***Data Entry***

This method involves the manual entry of data into the system, where new data is created with the help of human operators or devices that generate data for the enterprise automatically. This is the most frequently used method in day-to-day operations.

- ***Signal Reception***

In this method, the data is created by the devices as opposed to people (for example, via sensors) and have this data fed into the system for storage and processing at a later point in time. This is typically important in control systems, but with more Internet of Things devices, this form of data acquisition has become especially popular too. This process allows for data pre-processing and involves actions like cleaning, enrichment, integration, movement, as well as things like extract-transform-load (ETL) processes. The ETL process is essentially the process where data is read, then converted from its previous form into the form it needs to be in, then written in under it"=s new format. An important idea of this phase is that no value is derived from the data itself; it is merely processed.

4.2. Data Synthesis using BLAKE2 Algorithm

This stage is where semantics are extracted from the data. Deriving value from the data is extremely important, especially in big data systems due to the fact that the extremely large amounts of data allow us to identify useful and interesting patterns that may not be immediately obvious -which is something that was never possible in small data systems in the past. It is important to note that in the data synthesis stage, inductive reasoning, as opposed to deductive, is used when deriving value from the data. For example, a case of deductive reasoning would be calculating the net sales for a business with the formula:

Blake2 = Auth-Key x Generated-Key – Private-Key x Public-Key

This particular case of deductive reasoning is trivial to solve, since as long as one knows the underlying equation, and the value of *Auth-Key* and *Public-Key*, one would be able to easily derive the *Generated* value - this is not really considered as deriving value from the data. In the case of inductive logic, modeling is needed, which takes in the large amounts of data and identifies things like patterns, which based on judgment and opinion, allows new data to be derived (for example, in the case of credit score calcu- lations).

4.3. Data Usage using BLAKE2 Algorithm

The data usage stage in the data life cycle is relatively self-explanatory. In this stage, the raw, as well as the derived data is used in the operations of a particular enterprise for it to run and manage itself. However, one thing to note in particular is that in this stage, the usage of the data is contained within the enterprise itself and there is no data leakage in any form outside the bounds of the enterprise.

4.4. Data Publication using BLAKE2 Algorithm

This is where the data actually leaves the owning entity and is transferred to a location external to the entity. For example, in the case of a financial institution, it may send out reports of its financial standings to its investors, or in the case of a hospital, reporting the mortality rates to a national health care database. The one thing that has to be taken into consideration is that once the data has left the owning entity, it is impossible to recall it, and a consequence is that data values cannot be amended once they have left the entity.

4.5. Data Archival using BLAKE2 Algorithm

Eventually, a unit of data will reach a point where it is no longer in active use, and it is appropriate for more up-to-date information to replace it. This is especially apparent in today"s data-rich environment, where in many cases, concrete data may quickly become superseded with newer values in a matter of seconds. It should be noted, though, that it is uncommon for data to be immediately deleted once it has been superseded with more up to date values. In this phase, it is typical that a unit of data is stored for a number of years but this can vary a lot depending on many other factors. One example is in the case of historical analysis, where several datasets from many different time periods need to be compared. The data is moved over to an environment external to the active production environment in this phase. No further maintenance, usage or publication of the data takes place when the data reaches the archived environment;in the case where there is a need for any of these three actions, the data would be restored to the appropriate environment where the corresponding action takes place.

4.6. Data Purging using BLAKE2 Algorithm

When a data item is no longer needed, it is considered to be useless as it has reached the end of its life, and needs to be removed. This process is known as the data purging process. This is in contrast to the process of data deletion, which is often seen as a temporary effect, however the core idea of purging is that data that is no longer needed is deleted permanently to free up some computing resources. As can be observed here, the data life cycle is indeed a long process, with a number of intricacies in every step of the way; therefore, every step of the phase needs to be well-planned and well-thought-out. One of the core issues that needs to be taken into consideration is that the data life cycle handles sensitive and confidential data, hence it is critical that measures are taken at every step of the way to ensure the security of the data and the privacy of the users (or the related entities) are preserved.

5. CONTEXTUAL RESULTS

Data handlers should be aware that there is no specific point in the data life cycle that can be attacked; instead, data is vulnerable to attacks at every step of the way. Hence, we propose a set of guidelines that should be followed every step of the way to ensure contextual privacy and security is preserved. In order to further preserve the privacy of users, as well as the security of the data stored itself, measures should be taken to ensure that the data at rest is encrypted once the necessary maintenance tasks are complete. In terms of the maintenance tasks themselves, care should be taken in order to ensure that the privacy of the users who have contributed their data to the system by removing personally identifiable information from the data being processed. This could be done as part of the extract-transform-load process, where personally identifiable information is removed from the data in the system.

In some cases, organizations still need personally identifiable information for certain tasks. For example, in a bank, they would need to still have the credit card transaction records that can be linked back to a client for billing purposes, but at the same time, they would want to use the data of all the transactions conducted by their clients to learn the most common transactions their clients conduct in order to be able to better cater their credit card offerings depending on the time of the year, for example. In a case like this, instead of deleting the personal information wouldn"t make sense, nor would making a copy of all the data then deleting the personal information from one copy (space complexity would be an issue here, especially considering the extremely large amounts of data in big data systems). Instead it would be possible to have a map table that maps the unique identifier with a particular client and replacing the personal information in the transaction records with a unique identifier that doesn"t reveal any information to an outside party without having access to the map table. One thing to note is that the map table has to be kept securely, so as long as only authorized personnel are allowed to access this file, and the file is kept encrypted until used, this should be adequate.

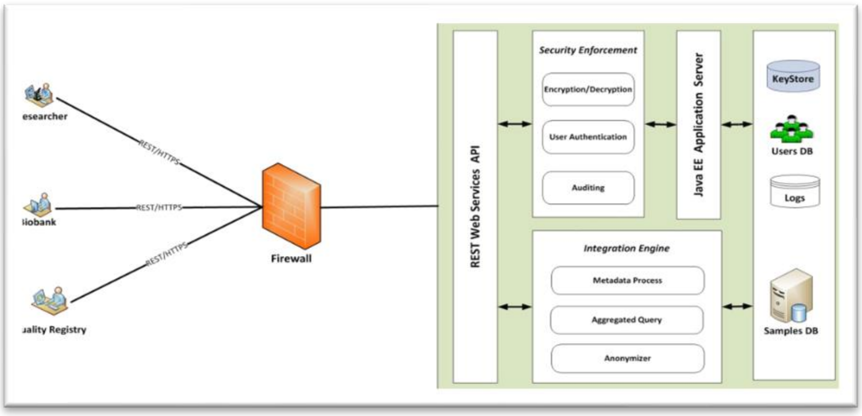

Figure 5. Diagrammatic approach to show the privacy policy that disallow the transfer of the data over to a different system.

The data capture phase is when new data is introduced into the system. This one of the best ways we could introduce potentially malicious code or data files into the system which could affect the operations of the system. One of the most important things in this phase, regardless of the capturing method is ensuring that some form of sanitization is executed on the dataset being introduced into the system. The idea is to ensure that pure data that isn"t potentially malicious is going to be introduced into the system (for example, inputting database query statements or commands in text fields, or opening executable files when data is copied over from a different system with the use of files). More specifically, there are additional measures that can be put in place for the various categories of data capture methods:

- *Data Acquisition using BLAKE2 Algorithm*

When acquiring data from a different system, it is important to check the current privacy policy from the existing system; in some cases, there may be clauses in the privacy policy that disallow the transfer of the data over to a different system. For this reasoning, migration may need the explicit consent from the users/entities that have contributed their data to the old system. In the case that such clauses don"t exist, it should be ensured that during the migration process, the data being copied over is encrypted in order to prevent eavesdropping parties from exploiting the raw data. It is also important to ensure that the data integrity isn"t compromised in any way during transit. This can be solved by digitally signing the data prior to it being sent over, and verifying the signature on the data set on the receiving end to ensure the integrity of the data. If the signatures don"t match up, then is an indication that there was some form of modification that occurred in tran- sit. In this scenario, the data is likely untrustworthy and should be dropped on the receiving end and a request for re-transmission should be sent back to the sender system.

- *Data Entry using BLAKE2 Algorithm*

In the data entry method, it is good practice to also consider the parties that are authorized to enter the data into the system. Implementing some form of an access control list would be useful in this case, where only users with the appropriate credentials would be able to access the system for data entry. From a privacy standpoint, it should be ensured that the necessary privacy notices have been presented to the entities that data was collected from, in order to ensure that the privacy of those entities aren"t violated. Sanitization of input will help in this case, as it would disallow accidental entries, and ensure that there are no deliberate attempts to input invalid commands through command injections in the data fields (like SQL injections).

● *Signal Reception using BLAKE2 Algorithm*

The capture of data through signal receptions also needs some additional measures to ensure that the data capture process is safeguarded, as well as the central data stores of the system. For one, the system should be designed in such a way that only approved signals are accepted, where all other unverified signals are automatically dropped. Since signals are received from devices like sensors, approved sensors (for example, only sensors of particular MAC addresses - if they are running over a network, or only sensors of certain universally unique identifiers - basically some form unique value that would describe a particular sensor) would be an accepted sensor that the system will listen to. If an unauthorized sensor attempts to connect to the system, it would automatically be dropped even if it tries to send signals to the system. Naturally, it should be ensured that only authorized users are allowed to modify the list of approved sensors.

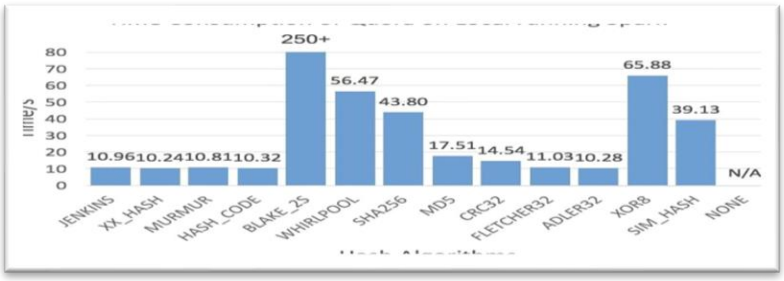

Figure 6: Time consumption on local Spark shows the similar pattern as on a remote cluster on EMR which proves the running mode of Spark won"t change the security and privacy of hash algorithms.

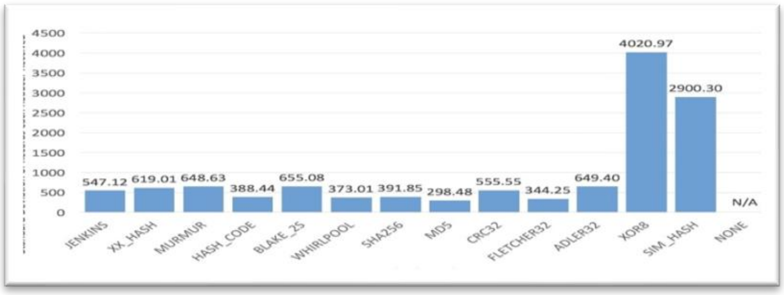

Figure 7: Distribution of local spark is able to get the data of hashing, which proves the cryptographic hashes „performances are dominated by throughput regardless of the tolerable spread.

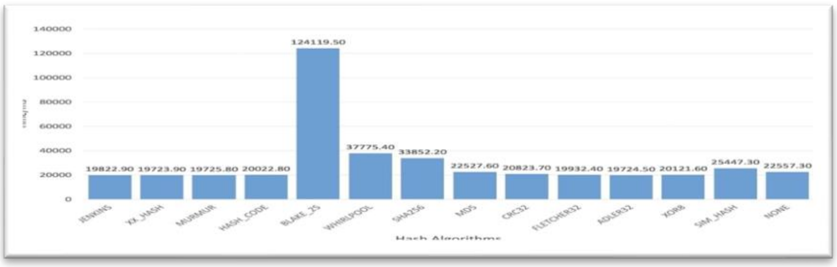

Figure 8: The main discrepancy between Spark and Hadoop is the performance of XOR, which can be an excellent proof of the impact of distribution on the task speed.

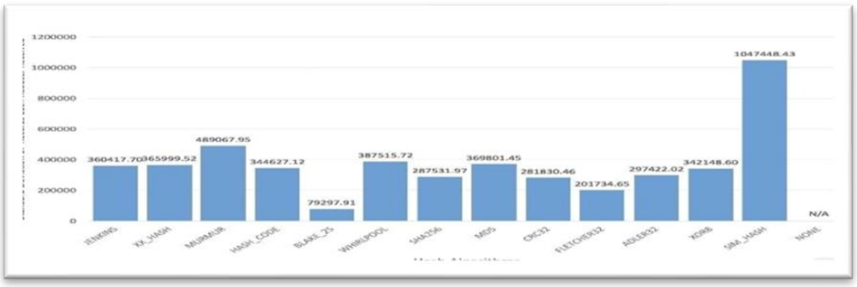

Figure 9: Distribution conditions of XOR are greatly improved this time, and combining with its outstanding throughput, the speed increase in Fig. 8 is completely within our expectation.

6. DICUSSION

In traditional database management systems, large datasets were becoming increasingly inconvenient to store, so new systems emerged to fulfill this requirement as mentioned in [32]. Hadoop is a software framework intended for use in distributed storage and processing of large datasets. It was originally de- signed to unify the processing power of commodity machines with local storage to process vast amounts of data, while minimizing the costs of computing. Hadoop implements a file system called Hadoop Distributed File System (HDFS) that provides scalability and reliability of storage, by spanning large clusters of computers. Three common issues are typically associated with Hadoop implementations:

1. *Weak security at the core*

One drawback about many big data systems is that they were not designed with security in mind; they were largely intended to be run within trusted environments, and Hadoop is no exception. At its inception, Hadoop had not authentication for users, nor was there any concept for preservation of data privacy. Although Hadoop has recently adopted best-practice security standards, its weak foundation in this area is still worrying for data managers.

2. *Fragmented data*

In Hadoop, redundancy and resiliency are encouraged, which are both achieved via replication through slave nodes. Unfortunately this allows the dataset as a whole to succumb to fragmentation issues as there may be delays in the replication process. As a result, there is a lot of complexity in the system, and security issues arise due to the absence of a robust security model.

3. *Distributed computing paradigm*

Since the availability of resources is necessitated by parallel processing on multiple machines of data, there simply exist more points of weakness in the entire system. Due to the eager use of data replication, the entire security of the system is reduced to the weakest node.

4. *Communication between nodes*

Since Hadoop was meant to be run in a trusted environment, node to node communication does not implement any secure communication protocols, so data is transferred in the clear and thus susceptible for exploitation.

5. *Access control in its infancy*

In older versions of Hadoop, for seamless integration, one of the characteristics was that all users with access to the trusted environment also had full control of the system; they could submit arbitrary code for execution. File permissions and access control lists were used in earlier distributions to mitigate this weakness, however they were relatively weak, as the access control mechanism was easily evaded; users could impersonate any other, and exploit privileged actions. In addition, users of the system had the same level of access across clusters, and any user could access and modify any dataset, which obviously yielded opportunities for data misappropriation.

Luckily as SHA-256 grew in popularity, security professional began to express concerns about the insider threats looming over the entire system and implemented a robust authentication system. However, since Kerberos only provides authentication services, it is not a solution for data at rest. Sharding is a method for distributing data across machines, while replication refers to copying data across machines to increase redundancy and data availability.

Table 1. The comparison between the existing literature and proposed technique for securing and

protecting the data.

Article	Technique	Security and Reliability
[33]	Message-Digest Algorithm	94.62%
[34]	Secure Hashing Algorithm (SHA-1)	97.31%
Proposed	BLAKE2 Algorithm	98.97%

6. CONCLUSION

With the emergence of the data paradigm, context can be exploited in a myriad of novel ways as the relevant datasets transition between states. Security and privacy in such systems are more important than ever, and as such this paper surveyed the key challenges and recommendations in big data systems against the data life cycle model. Although many existing implementations were built on shaky foundations of se- curity and privacy, they have since seen rapid and tremendous improvements in this regard. We explored some successfully adopted solutions, such as access control systems, middleware, and policy languages

that are sufficient for preserving the confidentiality of a unit of data in the myriad of states it transitions through using BLAKE2 Algorithm. The adoption of these new security and privacy techniques in real world implementations is comforting, however should be approached with caution as they are still in their infancy using BLAKE2 Algorithm. This paper provides practical privacy and security recommendations for every step of the data life cycle, examining prominent infrastructures and their features that relate to their data management policies using BLAKE2 Algorithm with 98.97% secure and reliable all the time. Upon surveying the state of solutions relevant to this paper"s goal, our suggestion to the community is that the best defense against data misappropriation is awareness. For privacy in modern big data systems, due diligence about the semantics of the data is key, as well as robust access control systems, which becomes more important as the number of users in the system increases. When it comes to security, the support is mostly there. For systems where it ceases to exist, there is no doubt they will reach that state eventually, as vendors have been very proactive in this area recently using BLAKE2 Algorithm. Unfortunately, the default state of many implementations is not geared towards contextual security, thus our recommendation to readers is to be cognizant of the many configurations available for the employed technologies, and to stay informed about the inherent weaknesses in these systems.

REFERENCES

[1] Defining Privacy - Santa Clara University, 2013 (accessed July 06, 2022). [Online]. Available: https://www.scu.edu/ethics/privacy/defining-privacy/

[2] Data Life Cycle - Boston University Libraries, 2015. [Online]. Available: https://www.bu.edu/datamanagement/background/data-life-cycle/

[3] 'One billion' affected by Yahoo hack, 2016. [Online]. Available: http://www.bbc.com/news/world-us-canada- 38324527

[4] "Secure deletion guideline - information security and policy -uc berkeley," 2016. [Online]. Available: https://security.berkeley.edu/secure-deletion-guideline

[5] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in International Symposium on Handheld and Ubiquitous Computing. Springer, 1999, pp. 304–307.

[6] D. Borthakur, "Hdfs architecture guide," HADOOP APACHE PROJECT http://hadoop. apache. org/common/docs/current/hdfs design. pdf, p. 39, 2008.

[7] G. Chen, D. Kotz et al., "A survey of context-aware mobile computing research," Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, Tech. Rep., 2000.

[8] M. Chisholm, 7 Phases of a Data Life Cycle - Bloomberg for Enterprise, 2015. [Online]. Available: https://www.bloomberg.com/enterprise/blog/7-phases-of-a-data-life-cycle/

[9] J. Clark, 5 Numbers That Illustrate the Mind-Bending Size of Amazon's Cloud - Bloomberg, 2016. [Online]. Available: https://www.bloomberg.com/news/2014-11-14/5-numbers-that-illustrate-the-mind-bending-size-of- amazon-s-cloud.html

[10] G. Cormode and D. Srivastava, "Anonymized data: generation, models, usage," in Proceedings of the 2009 ACM SIGMOD International Conference on Management of data. ACM, 2009, pp. 1015–1018.

[11] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The privacy bounds of human mobility," Scientific reports, vol. 3, 2013.

[12] Y. Demchenko, C. Ngo, C. de Laat, P. Membrey, and

[13] D. Gordijenko, "Big security for big data: addressing security challenges for the big data infrastructure," in

[14] Workshop on Secure Data Management. Springer, 2013, pp. 76–94.

[15] Y. Demchenko, C. Ngo, P. Grosso, C. de Laat, and P. Membrey, "Cloud based infrastructure for data intensive e- science applications: Requirements and architecture," Cloud Computing with e-Science Applications, p. 17, 2015.

[16] Y. Demchenko, Z. Zhao, P. Grosso, A. Wibisono, and C. de Laat, "Big data challenges for e-science infrastructure."

[17] M. Franklin, A. Halevy, and D. Maier, "From databases to dataspaces: a new abstraction for information management," ACM Sigmod Record, vol. 34, no. 4, pp. 27–33, 2005.

[18] A. Garg, J. Curtis, and H. Halper, "Quantifying the financial impact of it security breaches," Information Management & Computer Security, vol. 11, no. 2, pp. 74–83, 2003.

[19] D. Gewirtz, Volume, velocity, and variety: Understanding the three V's of big data - ZDNet, 2016. [Online]. Available: http://www.zdnet.com/article/volume-velocity-and-variety-understanding-the-three-vs-of-big-data/

[20] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in Proceedings of the 13th ACM conference on Computer and communications security. Acm, 2006, pp. 89–98.

[21] M. Haughn and S. Gibilisco, what is confidentiality, integrity, and availability (CIA triad)? - WhatIs, 2014. [Online]. Available: http://whatis.techtarget.com/definition/Confidentiality-integrity-and-availability-CIA

[22] R. Hull, B. Kumar, D. Lieuwen, P. F. Patel-Schneider, Sahuguet, S. Varadarajan, and A. Vyas, "Enabling context- aware and privacy-conscious user data sharing," in Mobile Data Management, 2004. Proceedings. 2004 IEEE International Conference on. IEEE, 2004, pp. 187–198.

[23] J. Katz, CMSC 858K Advanced Topics in Cryptography -Efficient Public Key Cryptography, 1st ed. University of Maryland, 2004. [Online]. Available: http://www.cs.umd.edu/~jkatz/gradcrypto2/NOTES/lecture4.pdf

[24] A. Kohgadai, what is Information Rights Management (IRM)?

- Skyhigh Networks, 2016. [Online]. Available: https://www.skyhighnetworks.com/cloud-security-blog/what-is- information-rights-management-irm/

[25] B. Lublinsky, K. T. Smith, and A. Yakubovich, Professional Hadoop Solutions. John Wiley & Sons, 2013.

[26] P. Membrey, E. Plugge, and D. Hawkins, The definitive guide to MongoDB: the noSQL database for cloud and desktop computing. Apress, 2011.

[27] T. Morgan, A Rare Peek into The Massive Scale of AWS -EnterpriseTech, 2014. [Online]. Available: http://www.enterprisetech.com/2014/11/14/rare-peek-massive-scale-aws/

[28] Oudah KH, Najm MA, Roomi AB, Al-saidy HA, Awadallah FM. THE RECENT PROGRESS OF SULFONAMIDE IN MEDICINAL CHEMISTRY. Systematic Reviews in Pharmacy. 2020 Dec 1;11(12).

[29] Roomi AB, AL-Salih RM, Ali SA. The effect insulin therapy and metformin on osteoporosis in diabetic postmenopausal Iraqi women. Indian Journal of Public Health. 2019 Apr;10(4):1479.

[30] H. Nissenbaum, Privacy in context: Technology, policy, and the integrity of social life. Stanford University Press, 2009.

[31] Nori W, Hamed RM, Roomi AB, Akram W. Alpha-1antitrypsin in pre-eclampsia; from a clinical perspective. J Pak Med Assoc. 2021 Dec 1;71(12):S53-56.

[32] Fenjan MN, Jarullah BA, Abdulrahman SJ, Roomi AB. Molecular identification and phylogenetic analysis of rotavirus in children suffered from diarrhea under five years old in Thi-Qar Province. Iraqi J Edu Pure Sci. 2019 Mar 1;9(1):1.

[33] T. Peterson, Whos Keeping Score? How Contextual Security Can Work in Your Organization - Dell TechCenter, 2015. [Online]. Available:

http://en.community.dell.com/techcenter/iam/b/weblog/ archive/2015/04/13/who-s- keeping-score-how-contextual-security-can-work-in-your-organization#&panel1-1

[34] A. Ron, A. Shulman-Peleg, and A. Puzanov, “Analysis and mitigation of nosql injections,” IEEE Security & Privacy, vol. 14, no. 2, pp. 30–39, 2016.

[35] Neyole, Jacob. (2015). Vulnerability of data security using MD5 function in php database design. international journal of science and engineering. 1. 11-15.

[36] Rao, Siddhartha. (2015). Advanced SHA-1 Algorithm Ensuring Stronger Data Integrity. International Journal of Computer Applications. 130. 25-27. 10.5120/ijca2015907056.