Performance Degradation and Cost Impact Evaluation of Privacy Preserving Mechanisms in Big Data Systems

Safia Kalwar[1], Eugenio Gianniti[1], Joas Yannick Kinouani[2],

Youssef Ridene[2], Danilo Ardagna[1]

[1]Politecnico Di Milano, Italy
[2]Netfective Technology, France

Abstract. Big Data is an emerging area and concerns managing datasets whose size is beyond the ability of commonly used software tools to capture, process, and perform analyses in a timely way. The Big Data software market is growing at 32% CAR, almost four times more than the whole ICT market, and the quantity of data to be analyzed is expected to double every two years.

Security and privacy are becoming very urgent Big Data aspects that need to be tackled. Indeed, users share more and more personal data and user generated content through their mobile devices and computers to social networks and cloud services, losing data and content control with a serious impact on their own privacy. Privacy is one area that had a serious debate recently, and many governments require to data providers and companies to protect the users' sensitive data. To mitigate these problems, many solutions have been developed to provide data privacy, which, however, introduce some computational overhead when data is processed.

The goal of this paper is to quantitatively evaluate the performance and cost impact of multiple privacy protection mechanisms. A real industry case study concerning tax fraud detection has been considered. Many experiments have been performed to analyze the performance degradation and additional cost (required to provide a given service level) for running applications in a cloud system.

Keywords: Big Data, Privacy, Performance Impact, Cost Impact.

1 Introduction

Today, data is accumulating at tremendous rates by click streams from web visitors, supermarket transactions, sensor readings, video camera footage, GPS trails. It is really becoming a challenge to store and process it all in a meaningful way. Every day, 2.5 quintillion bytes of data are created. This is so much that the 90% of the data in the world today were produced within the past two years [1]. That is why the embracement of Big Data is steadily increasing, it has moved from experimental projects to mission-critical, enterprise-wide deployments providing new insights, competitive advantage, and business innovation.

In addition, security and privacy are becoming very urgent Big Data aspects that need to be tackled [14]. Users share more and more personal data and user generated content through their mobile devices and computers to social networks and cloud services, losing data and content control with a serious impact on their own privacy, then Big Data technology should provide solid solutions to support users' privacy [15].

Big Data not only increases the scale of the challenges related to privacy and security as they are addressed in traditional security management, but also create new ones that need to be approached in a new way [2]. As more data is stored and analyzed by organizations or governments, more regulations are needed to address these concerns. Achieving security in Big Data has, therefore, become one of the most important barriers that could slow down the spread of technology; without adequate security guarantees, Big Data will not achieve the required level of trust. According to authors in [3], "Big Data brings big responsibility."

The goal of this paper is to quantitatively evaluate the performance and cost impact of multiple privacy protection mechanisms. In particular, masking and encryption techniques are applied to a real industry case study in the tax fraud detection application domain and an extensive experimental campaign has been performed to analyze the efficiency of a Spark cloud based clusters against these techniques. Moreover, relying on our D-SPACE4Cloud tool [4] (which supports the capacity planning of Spark cloud clusters providing deadlines guarantees) also the cost impact of the privacy preserving mechanisms is evaluated.

The structure of the paper is as follows. Section 2 describes the privacy protection mechanisms considered in our work. Section 3 describes the industry case study introducing the logical database model being used throughout for experimental purpose. This section also presents the queries designed for performance degradation and cost impact analyses. Section 4 illustrates D-SPACE4Cloud, the tool employed for Big Data clusters capacity planning. Section 5 presents the comparative analyses for the privacy techniques while Section 6 concludes the paper.

2 Data Privacy Solutions

This section reviews the anonymization techniques that we have considered in our performance benchmarking and cost impact analyses. Data anonymization, also known as de-identification, consists of techniques that can be applied to prohibit the recovery of individual information. Information systems usually store user data in records and each record includes a number of attributes, which can be classified into three categories [23]:

- Key attributes: attributes that uniquely identifies individuals (e.g., ID, name, social security number);
- Quasi-identifiers: attributes that can be combined with external information to expose some individuals, or to reduce uncertainty about their identities (e.g., birth date, ZIP code, position, job, blood type);

- Sensitive attributes: attributes that contain sensitive information about individuals (e.g., salary, medical examinations, credit card releases).

There are several anonymization techniques that can be applied on data before or along the process of mining, in order to protect the privacy of individual. Some of these existing and most used techniques are generalization, suppression, encryption and perturbation/masking [6]. In particular, in our work we considered perturbation/masking and encryption.

Perturbation/Masking consists of the replacement of the actual data values for dummy data, usually for masking databases testing or training. The general idea is to randomly change the data to mask sensitive information while preserving the critical data for data modelling. Some of the masking techniques are:

- *Replacement*: random replacement for similar content, but with no relation to the real data.
- *Shuffling*: random replacement similar to the previous technique, with the difference that the data itself is derived from the table columns.
- *Blurring*: this technique is applied to numerical data and dates. The technique changes the values of the data for some percentage of their random real value;
- *Redaction/Nulling*: this technique replaces sensitive data for null values.

The masking data is used to provide information that seem real but do not reveal information about anyone. This technique protects the privacy of the personal data in the information system as well as other sensitive information that cannot be disclosed.

On the other end, encryption uses cryptographic schemes based on public key or symmetric key to replace sensitive data (key_attributes, quasi-identifiers and sensitive attributes) for encrypted data. It transforms data to make it unreadable to those who do not have the authorization.

In the remainder of the paper we will evaluate the performance degradation and cost impact of masking where IDs are randomly replaced and encryption with AES at 128 and 256 bit.

3 Netfective Technology Case Study

Privacy preservation impact analyses have been performed on a real industry case study coping with fraud detection developed by Netfective Technology (NETF) within the frame of the DICE H2020 European project. Big Data technologies have already proven how much they are valuable to industries. Many businesses that have taken advantage of Big Data management and processing systems have increased their effectiveness and efficiency; whether it be for healthcare, advertising, or retail. Fraud recognition requires a holistic approach, and a combined use of tactical or strategic methods and state-of-the-art Big Data solutions. Traditional fraud detection practices have not been particularly successful largely because they come into play after the fact. Big Data intelligence software can perceive the deviant behavior at real

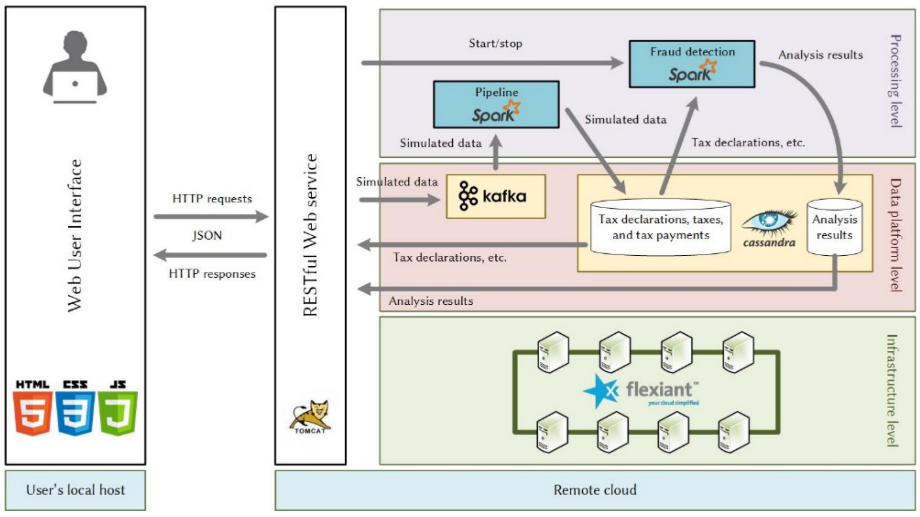

Fig. 1. NETF's Big Blu Architecture

time, thereby enabling fiscal agencies to get better outcomes. Big Blu, the minimal viable product (MVP) developed by NETF sends an alert whenever a suspicious tax declaration enters the information system.

Big Blu has been implemented by relying on open source state-of-the-art Big Data frameworks such as Kafka—to manage high rate flows of events—, Cassandra—to store and query massive amount of data—, and Spark—to process huge volumes of data. It is made of three main parts (Figure 1):

- A graphical user interface, which is a Web application developed in HTML, CSS, and JavaScript. It lets fiscal agents interact with Big Blu. Via this interface, they can simulate the entry of tax returns into the system, and launch fraud indicators to detect potentially fraudulent declarations.
- A RESTful Web service, which makes use of the Big Data frameworks to implement the data analysis. It sends back its results to the user interface for visualization.
- Cassandra databases are filled with taxpayers' details, historical tax declarations, tax payments, and so on.

The application will be performing computation on all data including new generated inputs. These data have to be processed using fraud indicators, which are a set of rules described by a domain expert. In the case of a new fraud indicator, the software has to proceed to a new batch processing phase on all data. It must also to be able to answer any query using a merge between old batch results and new real-time computations. The user will be notified on the graphical user interface with the taxpayers who may be fraudulent. In order to avoid any privacy and/or confidentiality issue, this paper reports results on a synthetic but realistic data set. NETF integrated to the

RESTful Web service a piece of software, the Taxpayers Random Generator Module (TRGM), able to generate, according to its needs, information describing millions of taxpayers. The TRGM produces realistic data using three main inputs:

- Various kinds of configuration parameters, such as the number of taxpayers, the percentage of single or married taxpayers, data encodings, data structures, database locations, and so forth.
- A model of tax declarations (Figure 2).
- A model of fraud indicators, which is actually a list of known fraudulent behaviors. (For example: a huge change in incomes compared to last years.) This model shall be extensible with any new identified fraud patterns.

Based on these three inputs, the TRGM generates millions of data to fill the Cassandra cluster. These generated data are the "raw material" for the whole application. They are based on a dedicated relational model of tax declarations (Figure 2) conceived specifically to be realistic (features could apply to a real system), generic (features and data models could apply to multiple government agencies), and neutral (data are generated in order to protect the privacy of citizens and businesses).
In the following, we consider three reference queries (see Figure 3), which will be the baseline for evaluating the performance and cost overhead introduced by the proposed privacy techniques. Query 1 accesses two tables to perform its analysis: Declare and TaxPayer.

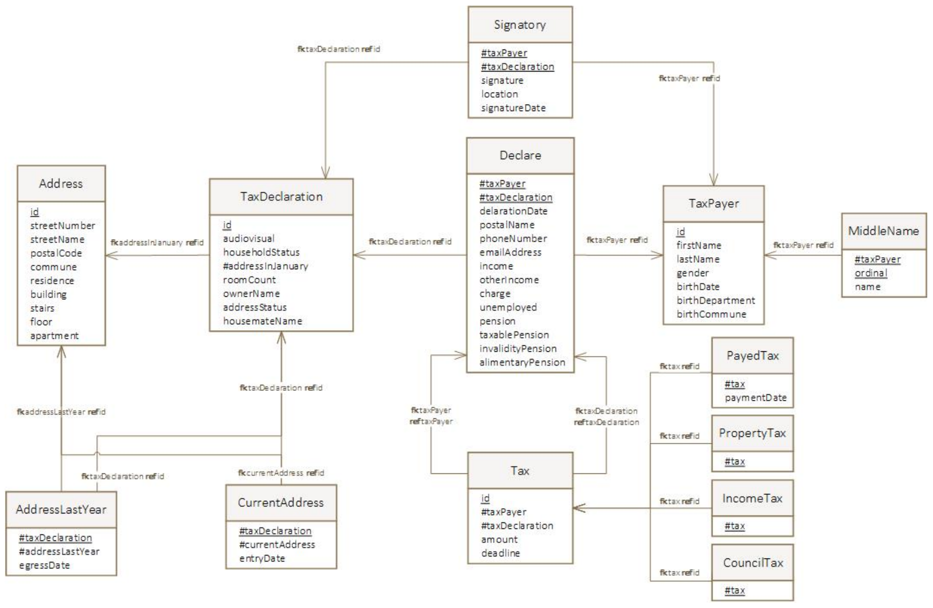

Fig. 2. Relational model of tax declarations with primary keys (underlined) and foreign keys (arrows)

It intends to measure the difference between incomes earned by a taxpayer during two successive years.
This is carried out to detect fraudsters by comparing the two incomes according to some criteria. For instance, if the income received a certain year is less than 20% (this percentage can be set as a parameter) than the one received the previous year, then the taxpayer is suspect. Since incomes are stored in the table Declare, Query 1 executes two joins: the first to make sure that the two tax declarations relate to the same taxpayer; and the second to obtain the full set of information about her/him. The result of this query is saved and used by other queries by passing the expected arguments, such as the percentage of income decrease, and the number of years to be taken under consideration. Query 5 involves only the table Declare. This table contains all the information needed to know every individual income and other credentials helpful to justify the amount of tax to be payed. Query 7 involves three tables: TaxPayer, TaxDeclaration, and Signatory. Each tax return must be signed by the taxpayers before they submit it to the fiscal agency.

Query 1:
```
SELECT  tp.id, , tp.gender, tp.birthdate,
tp.birthdepartment, tp.birthcommune, d1.taxdeclaration,
d1.declarationdate, d1.income,
d2.taxdeclaration AS D2TAXDECLARATION,
d2.declarationdate AS D2DECLARATIONDATE, d2.income AS D2INCOME
FROM Declare d1
INNER JOIN Declare d2 ON d1.taxpayer = d2.taxpayer
INNER JOIN taxpayer tp ON d1.taxpayer = tp.id
```

Query 5:
```
SELECT  *
FROM Declare d1
```

Query 7:
```
SELECT  tp.id,s.location, td.roomcount
FROM taxdeclaration td, signatory s, taxpayer tp
WHERE s.taxpayer = tp.id AND s.taxdeclaration = td.id
```

Query 3:
```
SELECT  dic.und_id, , tp.gender, tp.birthdate,
tp.birthdepartment, tp.birthcommune, d1.taxdeclaration,
d1.declarationdate, d1.income,
d2.taxdeclaration AS D2TAXDECLARATION,
d2.declarationdate AS D2DECLARATIONDATE, d2.income AS D2INCOME
FROM Declare d1
INNER JOIN Declare d2 ON d1.taxpayer = d2.taxpayer
INNER JOIN taxpayer tp ON d1.taxpayer = tp.id
INNER JOIN dictionary dic ON dic.id=tp.id
```

Fig. 3. NETF case study reference queries

This query retrieves, among other things, the place of signature. According to masking, we introduced a dictionary table that mplements a one-to-one mapping between a clear ID and a masked ID. In other words, the tables in Figure 2 store masked IDs for the taxpayers, while the clear ID is available only in the dictionary table which has a restricted access. Query 3 is derived from Query 1 and adds a join to read the clear IDs. Similarly, Query 6 and Query 8 are derived from Query 5 and Query 7 respectively (but are omitted for space limitation). From a performance evaluation perspective, these additional joins introduce a system overhead, which might lead to a performance degradation or to an additional cost if the queries needs to be run within an a-priori fixed deadline. As discussed previously the second privacy mechanisms considered is encryption with AES at 128 and 256 bit. In this case, the IDs and sensitive data stored in the Cassandra tables are encrypted and decryption is performed contextually while running Query 1, 5 and 7. Our goals is to evaluate the encryption overhead and its corresponding cost impact, if any, in the cluster capacity planning.

4 D-SPACE4Cloud

D-SPACE4Cloud [4] is the tool we used to evaluate the cost impact of privacy mechanisms implementation. D-SPACE4Cloud supports the capacity planning process of shared Hadoop Cloud clusters for MapReduce or Spark applications with deadline guarantees. In a nutshell, the tool implements a search space exploration able to determine the optimal virtual machine (VM) type, possibly from different providers, and the optimal number of instance replicas.

The underlying optimization problem is demonstrated to be NP-hard and it is solved heuristically, whereas job execution times are estimated via queueing network (QN) or Stochastic Well Formed Net (SWN) models.

The tool implements an optimization mechanism that efficiently explores the space of possible configurations, henceforth referred to as solution space. An initial solution is identified by relying on the solution of a Mixed Integer Non-linear Programming (MINLP) problem where the job duration is expressed by a machine learning model, which predicts application execution times given the total number of available cores (see [7] for further details). The fast MINLP model is exploited to determine the most cost effective VM type. Yet the quality of the returned solution can still be improved, since the MINLP problem is just an approximate model. For this reason, a more precise QN or SWN model is adopted to get a more accurate execution time assessment: the increased accuracy leaves room for further cost reduction. However, since QN or SWN simulations are time-consuming, the space of possible cluster configurations has to be explored in the most efficient way, avoiding evaluating unpromising configurations. In the light of such considerations, a heuristic approach has been adopted. A parallel Hill Climbing (HC) technique has been implemented to optimize the number of replicas of the assigned resource for each application; the goal is to find the minimum number of resources to fulfill the deadline requirements.

In particular, HC is a local-search-based procedure that operates on the current solution performing a change (more often referred to as move) in the structure of the solution in such a way that the newly generated solution could possibly show an improved objective value. If the move is successful it is applied again on the new solution and the process is repeated until no further improvement is possible.

The HC move consists in increasing/decreasing and changing the VM type for each big data application. This task is executed in parallel where possible. Parallelism is particularly important as each solution spawned during the optimization is evaluated via simulation and this is a time-consuming task. Therefore, in order to make the D-SPACE4Cloud usable, we focused on increasing the level of parallelism as much as possible. In [4] we have shown that, in the worst case, the relative error between performance prediction models and real applications execution can reach up to 32.97%, which is perfectly in line with the expected accuracy in the performance prediction field [8] while the average relative error is 14.13% overall.

5 Experimental Analysis

To evaluate the performance of the NETF application against different privacy preserving mechanisms the queries reported in Figure 3 are implemented in Spark 2.0 and run on Microsoft Azure HDInsight. We ran multiple experiments considering two different virtual machine types, i.e., D12v2 with 8 cores and 28GB of memory and D4v2 with 4 cores and 28GB. The number of executors per VM were either two or four. We set the maximum executor memory 8 GB as for the driver memory while executors were assigned two or four cores. The number of VMs varied between 3 and 13. Overall, we ran experiments using up to 52 cores. Runs were performed considering the default HDInsight configuration. In the following, baseline queries and anonymized queries will be compared considering the same configuration. Each query for each configuration was run 10 times to evaluate the average execution time. In the following we use the number of records in the TaxDeclaration table to express the dataset size. Experiments were performed in the range between 1 and 30 millions records. We first evaluate the performance degradation, if any, of the privacy mechanisms. In case of performance degradation, then we evaluate the cost impact through the D-SPACE4Cloud tool. Section 5.1 describes the experimental settings and the performance degradation analysis. Section 5.2 reports cost impact evaluation. Many experiments have been performed but for space limitation here we listed the most representative ones.

5.1 Performance Degradation Analysis

To analyze the performance of each query running on a different dataset, multiple configurations were considered by varying the number of Spark worker nodes and executors. For 1 million dataset three and four VMs were considered with two executors each. For datasets from 5 up to 30 millions, we used 5, 6, 10, and 12 VMs. Some runs took hours and some were running for very long time and finally failed for stage timeout. The most problematic query was Query 3, introducing an additional join with

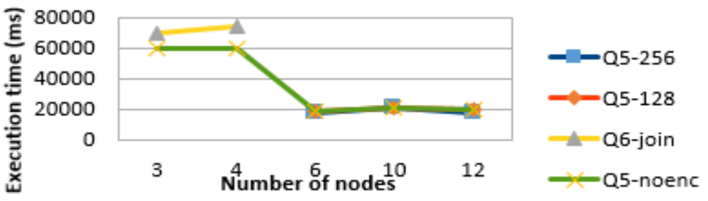

Fig. 4. Performance degradation analysis of Query 5,6 for 1.5 million entries data set

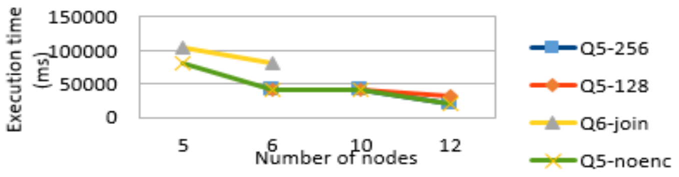

Fig. 5. Performance degradation analysis of Query 5-128 bit encrypted for 13 million entries data set

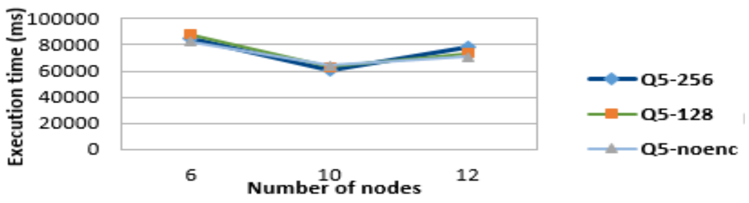

Fig. 6. Performance degradation analysis of Query 5 for 30 million entries data set

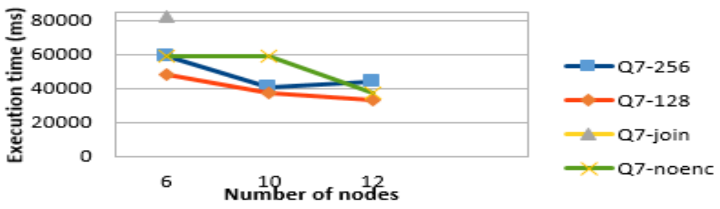

Fig. 7. Performance degradation analysis of Query 7 and 8 for 10 million dataset

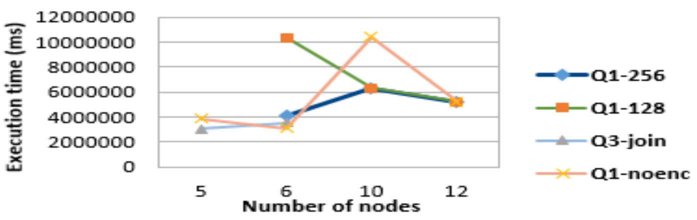

Fig. 8. Performance degradation analysis of Query 1 and 3 for 10 million dataset.

the dictionary table, which led to a huge traffic due to shuffle operation among stages and caused I/O and network saturation.

After analyzing the above fact, we decided to increase the number of nodes for larger datasets (which allowed to reduce at least node I/O pressure). 13 VMs were considered for recurring failing configurations. Unfortunately, we did not obtain any results for masking with dataset size greater than 13 millions.

Figure 4 summarizes the results we obtained for 1.5 millions of taxpayers, more realistic in a real context for analyses at least at a regional or local level. This type of analyses can be useful especially in order to get details about taxpayers in a specific geographic area. We can also perform analyses on a limited number of taxpayers regarding their age combined to their location. For instance, many cities in Europe are known to be very attractive for retired persons, thus they can be targeted by tax agents investigation. Additionally, it is common in Big Data to focus on a subset of the data to be processed in order to build and set-up the models/architectures to be applied in the future to a larger number of taxpayers.

Figure 6 compares the different privacy mechanisms for Query 5 on 1.5 millions data set (recall that Query 6 introduces an additional join on Query 5 when masking is applied). Unfortunately, Query 6 was always failing when more than 6 nodes were used. D12v2 were used in the runs and the number of cores varied between 12 and 48. From Figure 6 it is evident that masking/join has the maximum effect on the performance than other privacy techniques, while encryption has negligible performance effects. Masking introduces around 28% performance degradation while the lines characterizing encryption overlap the plain Query 5 execution time. Encryption has only 3% overhead at maximum. Query 5 behavior is also independent on the data set size considered.

Figures 8 and 9 report results for 13 and 30 millions data sets ran on D12v2. At 13 millions, experiments show that masking causes around 50% performance degradation while encryption has only 4% impact on performance. The plot in Figure 9 compares unencrypted and encrypted version of Query 5 on 30 millions data set. All three lines in plot closely overlaps one another, which depicts there is no significant overhead due to encryption on system performance. In some situations, encryption resulted even in a performance improvement, as reported in Figure 9 for Query 7 ran on D12v2. This was due to the fact that encrypted data resulted in lower shuffle time.

For Query 7, there is no significant overhead due to masking and encryption although we could see in plot at nodes 10 Query 7 (plain) is taking maximum time. The cause behind this is shuffling of additional executors, which causes delay than the actual time taken by the plain query.

Figure 10 reports the results for Query 1 and Query 3 at 10 million data set. As the plot shows, results are very noisy. Query 1 is very complex and takes significantly larger time compared to other queries. For these reasons, performance results are more affected by resource contention, which characterizes cloud environments. In such scenario Query 1 was always failing for all datasets with size more than 10 million. Query 3 was more time consuming so runs were taken against two different configurations that is 10 and 12 executors. The above analysis (see Figure 10) shows that overhead on performance due to masking is around 12%. The green line shows time for 128 bit encryption, initially it is taking more time than 256 bit encryption, but then they both closely overlaps. We guess that this anomalous behavior was due to cloud resource contention and, given also the limited impact, the data we measured was affected by noise. This experiment shows the overall impact due to encryption on system performance is only 2%.

5.2 Cost impact analysis

In case of performance degradation, we used D-SPACE4Cloud to evaluate the cost impact of the implementation of privacy mechanisms when the two versions of the same query (i.e., plain and anonymized) need to be executed within the same deadline. The initial deadline was set according the time measured on the real system with the smallest configuration. The deadline is decreased iteratively with a step whose range is [5, 500] seconds. The difference was not fixed and varied according to the data size and privacy technique being considered. This way, multiple optimization instances are considered for each experiment.

Initially we considered Query 5 and 6 at 1.5 millions dataset with 85s initial deadline. Then, the deadline was iteratively decreased by 20s in order to consider 10 optimization instances. The results are reported in Figure 12. From the experimental results, one can see that above 45s and for 20s no extra costs are incurred while for 25 and 15s deadline the cost overhead due to the masking technique is in between 50 and 66%. Deadlines lower than 15s resulted to be too strict and D-SPACE4Cloud did not find any feasible solution.

Figure 13 reports the results of for Query 1 and 3, when 10 millions entries data set is considered for performance profiling. The initial deadline is set to 3,500s that is maximum of the execution time of both queries registered on the real cluster and which is iteratively reduced by 500s. The results show that cost overhead due to masking technique is between 33 and 40%, and no extra costs are incurred for deadlines larger than 2500s. Deadlines lower than 1500s were too strict and no feasible solutions were found.

Further experiments were targeted for encryption. We selected data set 10 million and we evaluated Query 7, AES 256 bit encrypted and unencrypted for which we registered the largest performance degradation that is 5%. In this way, the results we achieve will be conservative. 80s was set as initial deadline, which was then iteratively reduced by 5s. The results are reported in Figure 12 which shows that 50% cost overhead is achieved at 40s, which is also the minimum deadline that can be supported by the system (otherwise no feasible solution can be found).

Finally, we report the results we achieved by considering the largest data set, i.e., 30 million, for Query 5. 80s was set as initial deadline, which then was iteratively reduced by 20s. Figure 12 reports the cost for AES 256 bit encryption. The experiment shows that cost ratio due to encryption is only 13% at maximum. While below 40 seconds D-SPACE4Cloud could not find any feasible solution.

From the results we achieved we can conclude that masking causes more overhead than encryption while there is no significant difference between 128 and 256 bit encryption. The cost overhead due to masking in the worst case was around 66%, while for encryption was 50%.

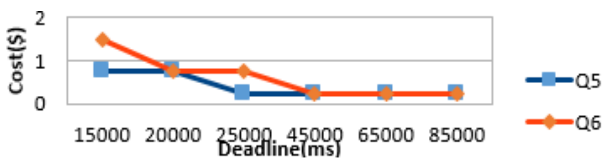

Fig. 9. Cost evaluation of Query 5 to 6 by varying deadlines for 1.5 million data set

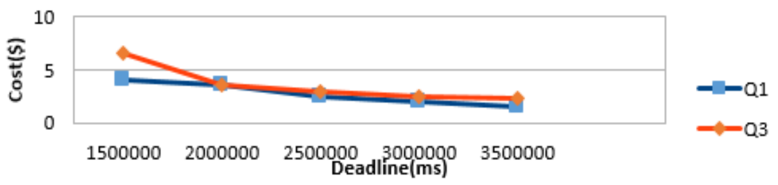

Fig. 10. Cost ratio evaluation of Query 1 to 3 by varying deadlines for 10 million dataset

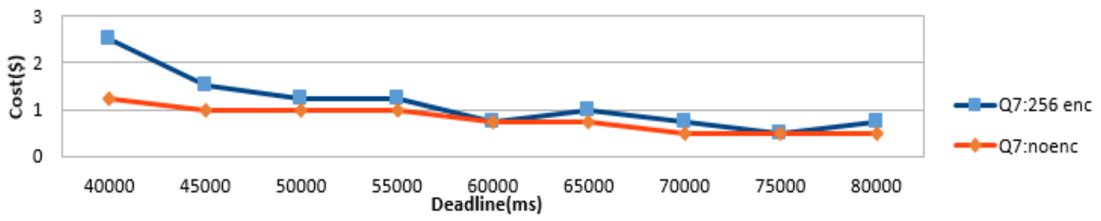

Fig. 11. Cost evaluation of Query 7, 256 bit encrypted to unencrypted by varying deadlines for 10 million dataset

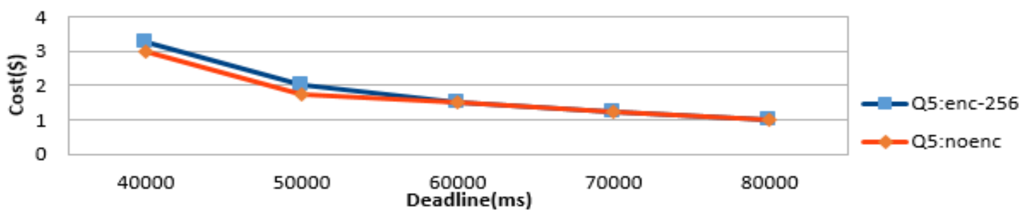

Fig.12. Cost ratio evaluation of Query 5, 256 bit encrypted to unencrypted by varying deadlines for data 30 million

6 Related work

In Big Data systems, security and privacy issues are magnified by velocity, volume and variety. Therefore, traditional security mechanisms, which are tailored to securing small-scale security and privacy challenges, are inadequate. Authors in [5] overviews the most relevant security threats for Big Data infrastructures while the work in [10] provides an analysis of Big Data privacy models. Privacy issues and current technological solutions are surveyed in [9]. To the best of our knowledge, this paper is one of the first attempt to quantitatively evaluate the performance overhead of privacy implementation and its impact on cloud operational costs.

From the side of the capacity planning literature, architecture design space exploration is an important topic [11,12]. High level models and tools to support software architects (see, e.g., Palladio Component Model and PerOptirex design environment [16,17], or stochastic process algebra [18] and the PEPA Eclipse plugin [20]) have been proposed for identifying the best configuration given a set of quality of service requirements. Unfortunately, such works neither support Cloud-specific abstractions nor consider Big Data applications. On the other side, capacity management and cluster sizing for Big Data applications has received also a widespread interest by both academia and industry. The starting point is the consideration that Hadoop often requires an intense tuning phase in order to exhibit its full potential. Starfish, a self-tuning system for analytics on Hadoop, has been proposed [19]. The resource provisioning problem, instead, has been faced in [21]. The goal is the minimization of the execution cost for a single application. Authors present a cost model that depends on the dataset size and on some characteristics of the considered application. In [22], the ARIA framework is presented. This work is the closest to D-SAPCE4Cloud and focuses on Map Reduce clusters dedicated to single user classes aiming at avoiding as much as possible over-provisioning costs. All the above-mentioned works are based on Hadoop 1.0, where CPU slots are statically allocated to Map and Reduce tasks and the basic FIFO scheduler is considered. To the best of our knowledge, D-SPACE4Cloud is one of the first tool coping with Hadoop 2.x and Spark applications.

7 Conclusions

We presented an extended experimental campaign aimed at evaluating the performance degradation and cost impact of masking and encryption privacy mechanisms in a real industry case study. Our results have shown that encryption has a minor impact on performance and costs whilst performance degradation does not always result in additional costs. Future work will integrate the D-SPACE4Cloud tool within a framework that automatically modifies queries and the underlying data representation to obtain informed decisions on privacy impact in a pre-production environment.

References

1. https://whatsthebigdata.com/2016/03/07/amount-of-data-created-annually-to-reach-180-zettabytes-in-2025/ Access Date:15/5/2017, Time: 4.30 pm CET
2. Lekkas, D.; Zissis, D. (2012). “Addressing cloud computing security issues.” Department of Product and Systems Design Engineering, University of the Aegean, Syros 84100, Greece, V.28, Issue.3, Dec, 2010, pp.538-592.

3. Buyya,R.; Yeo.S.C.; Venogopal,S. (2009). “Market-Oriented Cloud Computing: Vision, hype, and reality for Delivering IT services as Computing Utilities.” In HPCC Proc.
4. Ciavotta, M; Gianniti, E.; Ardagna, D. (2016), “D-SPACE4Cloud: A Design Tool for Big Data Applications.”, in ICA3PP Proc.
5. Moura, J.; Serrao, C. (2015). “Security and privacy issues of Big Data.” In EDBT/ICDT Proc.
6. Vieria, M.; Madeira, H. (2015). “Towards a Security Benchmark for database Management Systems.”, In DSN Proc.
7. Ataie, E; Gianniti, E; Ardagna, D; Movaghar, A. (2017). “A combined Analytical Modeling machine learning Approach for Performance Prediction of MapReduce Jobs in Cloud environment. SYNASC 2016 Proc.
8. Lazowska, D, E. et al. (1984). “Quantitative system performance: computer system analysis using queueing network models,” Prentice-Hall,Inc.
9. Jain, P.; Gyanchaandani, M.; Khare, N. (2016). “Big data privacy: a technological perspective and review.” Journal of Big Data, pp. 3-25.
10. Soria-Comas, J.; Domingo-Ferrer, J. (2015). Big Data privacy: “Challenges to Privacy Principles and Models”, in Data Science and Engineering, V.1, Issue , pp. 21-28.
11. Aleti, A.; Buhnova, B.; Grunske, L.; Koziolek, A.; Meedeniya, I. (2013). “Software architecture optimization methods: A systematic literature review.” Software Engineering, IEEE Transactions on 39 (5), pp. 658-683.
12. Brosig, F.; Meier, P.; Becker, S.; Koziolek, A.; Koziolek, H.; Kounev, S. (2015). “Quantitative evaluation of model-driven performance analysis and simulation of component-based architectures”. Software Engineering, IEEE Transactions on 41(2), pp. 157–175.
13. Elmasri, R.; Navathe, S. B. (2011). “Database Systems". Pearson - Addison Wesley, 6th. Edition
14. Agrawal, D.; Das, S.; Abbedi, E, A. (2011). “Big Data and Cloud Computing: Current State and future Opportunities.” In EDBT/ICDT Proc.
15. Marques,J;Serrão. (2013). “Improving Content Privacy on Social Networks Using Open Digital Rights Management Solutions”, Procedia Technology, pp. 405-410.
16. Becker, S.; Koziolek, H.; Reussner, R. (2009). “The Palladio component model for modeldriven performance prediction.” Journal of Systems and Software 82(1), 3–22.
17. Koziolek, A.; Koziolek, H.; Reussner, R. (2011). “PerOpteryx: Automated application of tactics in multi-objective software architecture optimization.” In QoSA 2011 Proc.
18. Tribastone, M.; Gilmore, S.; Hillston, J. (2012). “Scalable differential analysis of process algebra models.” IEEE Transactions on Software Engineering 38(1), pp. 205–219.
19. Herodotou, H.; Lim, H.; Luo, G.; Borisov, N.; Dong, L.; Cetin, F.B., Babu, S. (2011). “Starfish: A self-tuning system for Big Data analytics.” In: CIDR Proc.
20. OMG: PEPA: Performance evaluation process algebra (2015), http://www.dcs.ed.ac.uk/pepa/tools/
21. Tian, F.; Chen, K. (2011). “Towards optimal resource provisioning for running MapReduce programs in public Clouds.” In: CLOUD Proc.
22. Verma, A.; Cherkasova, L.; Campbell, R.H. (2011). “ARIA: Automatic resource inference and allocation for MapReduce environments.” ICAC Proc.
23. Basso, T.; Moraes, R.; Antunes N.; Vieira, M.; Santos, W.; Meira W. Jr. (2017). “PRIVAaaS: privacy approach for distributed cloud-based data analytics platforms.” In CCGrid Proc.