Securing Big Data and the GPG Method

|  |  |  |
| --- | --- | --- |
| Gerardo Garza  [Mail.garza@gmila.com](mailto:Mail.garza@gmila.com) | Ekaterina Pirogova  [epirogova@smu.edu](mailto:epirogova@smu.edu) | Scott Gozdzialski  [Sgozdzialski@smu.edu](mailto:Sgozdzialski@smu.edu) |

***Abstract*—This paper presents the current systems for securing big data and a new approach for securing big data. The new method, the GPG method, focuses on processes and the data itself for security.  The world today is looking for faster and cheaper ways to handle big data, but they have not focused on the risks involved. A simple breach can expose confidential business information or even millions upon millions of individuals to risk, as was the case in the Equifax break exposing 140 million individuals' information.  For all of this, it is becoming more important that as we grow, we develop security procedures for big data and we build security into future tools and systems**.

.

**I. Introduction**

Reading the news today it is not hard to see violations of computer and network security everywhere. Whether it is an article about Kapersky Antivirus and Security Suite having a relationship with the Russian intelligence agency, the FSB, or the thousands of cell phones infected by fake apps on the Android play store. Perhaps most worrying, the security breach at Equifax which exposed 140 million individuals’ social security numbers, addresses, and other personally identifiable information. These breaches show the dangers to individuals caused by untrusted relationships with seemingly trusted agents, nefarious individuals finding loop holes in the system, and lax security practices at large corporations. What if there were some systems that had inherent security issues? What if they were prominent technologies in their infancy but with such explosive growth that newer security could not be implemented? One of these technologies is big data, whose growth has been so pervasive that most of the technology has focused on how to keep up and not how to secure.

The exponential growth of big data into and beyond exabytes [1,2] is reminiscent of the growth of computers in the late 1970s and early 1980s [3], as well as the growth of the internet in the late 1990s and early 2000s [4]. There is a global explosion of technology and need, with everyone from individuals, small companies, and large corporations, to ever larger governments using big data. As such, there is a need for many different technologies to handle these security problems.

The different technologies all have different foundations in the control and manipulation of the data. There is no underlying standard to secure each instance of different security risks and fixes. The nature of the differences in tech build in securities risks. This will be one of the focuses of this paper: what security has been baked into the technology and where can we build from there.

We are in a time where everyone is watching everyone else. We have television shows where we watch other peoples’ lives, called reality television, and we live stream everything. As creatures, we are both voyeur and exhibitionist, but we like our privacy all the same. When we hear things like our phone company has super cookies that track everything we do [5], we protest in the street because this violation of privacy is not granted but taken. With big data, there is also concern of anonymity being taken. If I have enough data, we can in theory parse that data down to a single person and take the privacy away from the individual.

As big data becomes more and more used in individuals’ lives by the companies we trust, companies we don’t trust, and even governments, the risk to individuals is being removed from their control to the control of other entities. With this hand-over of control of individuals’ information, the risks are not always controlled in a proper fashion. Minor infractions in security policies can cause large incidents, as is seen in the US government Office of Personnel Management losing millions of members’ information with security classification and in what organizations they work [6]. The Equifax incident was caused by delaying the implementation of a security patch for a known issue [7].

One of the major points will be a look at the different systems out there to store and manipulate big data, as well as some of the security concerns with these systems. Finally, we will be laying on an innovative approach to security in Big Data. Big Data has some inherent challenges associated with it [8]. When data is so big it bogs down a business’s IT resources it is hard to imagine a way to properly secure it from intruders while maintaining confidentiality. We will do this with a three-prong approach, the Garza, Pirogova, Gozdzialski (GPG) method, with a focus on storage, transmission, and finally processing of the data.

**II. Big Data systems and Security**

Current Big data systems have built in security systems, most of these systems focus on security with different tools for each aspect of the security process. Most systems focus on access control, auditing, authentication, and encryption. As example we will discuss the basic security tools for some popular big data systems. In particular we will discuss the security layers of Cassandra, MongoDB, and SQL. Cassandra [8] for example uses role based control for users [9] and object based permissions for files. It uses SSL encryption for any connections between the servers and the users. Cassandra also closes any port not needed on the firewall to limit access to the outside world. MonoDB [10] uses almost the same exact methods as Cassandra for authorization of user and securing connections. They also use “WiredTiger” storage encryption to protect data at rest. SQl focuses more on the protection of the database [11] [12]. SQL uses database level, schema level, and user level permission to grant access. It leaves network protection and storage protection up to the user to piece together. These set ups work fine to secure our data, but we propose an added security layer by using the nature of the data itself to impart security to the system.

**III. Garza Pirogova Gozdzialski (GPG) method**

**a. GPG method Set theory and explanation.**

The focus of the GPG method is for the protection of big data. Data of petabytes or greater size. The GPG method will focus upon data at rest or storage, data transmission, and finally data processing. It will be doing this by focusing on the whole dataset, while continuing to use the securities put in place by the systems that run the data.

The reason this entire method is simple, as data gets bigger it bogs down resources [13], whether this is storage, processing, or even transmission capabilities. Thus, the need for new approaches or systems to protect our information. With data growing at such a fast rate we need to be able to protect it.

To do this, we need to change our thoughts of security, security is currently thought of as a locked box where a key opens the entire system to compromise. This can be seen in many of the examples above. We need to think of it more like a fortress. In a fortress you have many rooms, some are more secure, like armories, while others are open, like the barracks or kitchen, finally, some areas are open and not as protect, like the courtyards and outer moat. But, all has some expected over-arching security. There are some locations where security is set up in the layer process, so wee need to add another layer to our security model.

This next layer is to think of security as a prison. We need to understand just as important as keeping adversaries out of our computer resources, we need to keep our data in. If we can control the flow of data out of our network, even if an adversary gets in they cannot retrieve any actionable information.

The key to this is the data, call the entire dataset, no matter how large, set A. Set A has some subset B, where if B is secure A is expected to be reasonably secure. Without B the content and context of A cannot reasonably be ascertained through analysis or brute force. Set B can be any subset of Set A up to the entire set of set A.

= *A secure*

Set B can also be split up into subset C, D, E, F and so on, so that with only part of each subsets of B (called set CDEFtoN from now on) you cannot determine the nature of set B. Finally, without complete knowledge of any one subset CDEFtoN you cannot determine the content or context of the subset. As we continue the foundation upon which this is built will become clearer.

B⊇ {C, D, E, F, ….N}

*Where without C+D+E+F..+N the nature of B is indeterminable.*

And

*Without all of Set C, D, E, F, …N we can not determine the content and context of the subset C, D, E, F, ….N itself.*

Finally, if you do not know the relationship of subsets CDEFtoN you cannot construct the content or context of subset B and therefore cannot determine content and context of set B, securing set A. A of this is displayed in figure 1.

Without

*CrD+CrE+CrF+…CrN+DrE+D+F+……N-1 rN*

*Then No Content or Context of B = A secure*

Fig 1


Fig 1.

**b. The GPG Method steps**

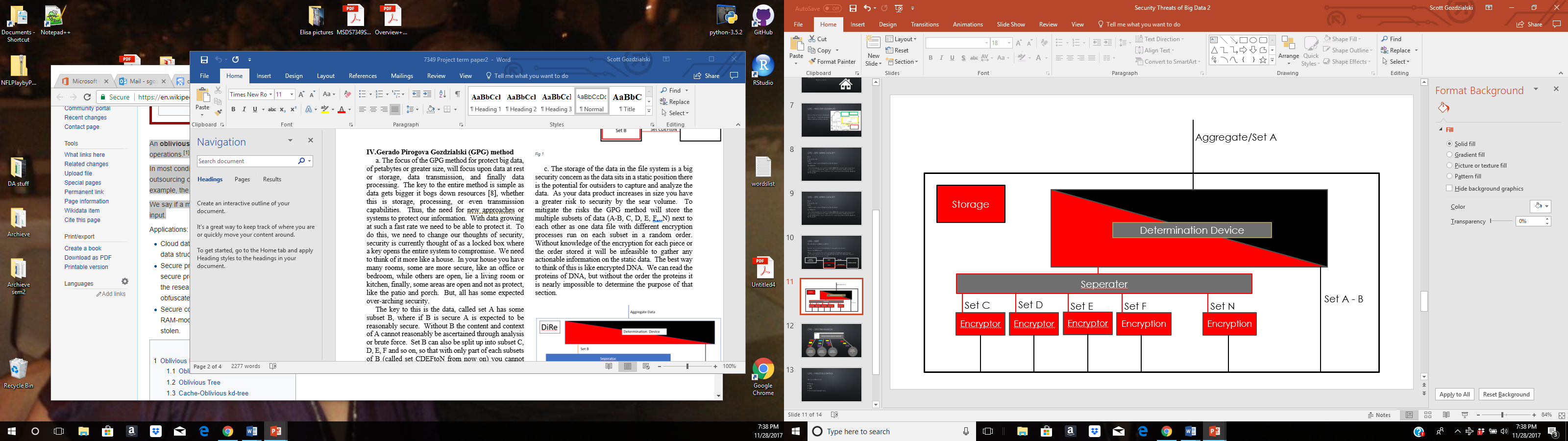
With the GPG method we will first make some assumptions. The first being, the tools we use to secure our data at not compromised in any way. The second being, the tools and methods we talk about work in the prescribed way, meaning they are effective.

While discussing the GPG method of securing big data we will be explaining the process while using the example of securing medical records. We will show if the medical records are open to all facilities to read and update, we can still secure them with the GPG method. We will use a small medical facility that has roughly 1500 patient records. We will focus on a patient who is roughly 30 years old who is of average health and goes to the doctor about once a year.

I. The first step with the GPG method is to split the data into different subsets as discussed above. We analyze the data and determine where we need to protect the information in order to secure the entire dataset. The dissection of the data is based upon the data itself and not a predetermined algorithm. The fact we control where we split our dataset and how many splits our dataset has reduces this risk. So, for our example, we will split our data into individual doctor visits. We will tie these visit to a patient identifying subset. Admin note, how we verify the patent will not be discussed.

Looking at our subset information above we have to decide where the data lies in the set A, Subset B, subset CDEFtoN construct. The identifier subset is part of B, in particular part of subset CDEFtoN and has the distinction of being the key feature. The metadata on the files including Doctor, time /date, and location are considered relationship information between the subsets CDEFtoN and need to also be secure. The actual record of the visit can be either part of set CDEFtoN or even set A-B, as you will see going through the rest of the process.

II. Next, we encrypt the different subsets CDEFtoN uniquely. This is done by some device or process that dissect the data into the different data streams, figure 2.



*Fig 2*

This is done by first determining the split based upon some rule set by the use based upon the unique dataset. A determination device uses this ruleset to split the dataset A. Dataset A-b gets set out to be encrypted regularly. Set B then gets sent to a separator that separates set B further into set CDEFtoN. Then sets CDEFtoN get separated uniquely and passed through, where they get double encrypted by the encryption run by the overall system.

III. Next, we to step out of order, and talk about the actual system used to process and store our data. We need to secure those systems since there is no guarantee it is secure. To protect our different processing systems the GPG method will first use a dedicated system for the processing of our data. This system will only be used to process our data and nothing else. With the importance of big data growing and the security risks associated with it, having a dedicated processing system only makes sense. On this dedicated system we will also run processor and network whitelists to protect from malicious code being run. By running whitelist even if malicious code does get inserted into our system, it will not run, protecting our data. We will also, limit the locations our data can be transmitted to the network. This way if an adversary gets into our systems they can not transmit any information to an untrusted location. For our example of the medical records we will have a dedicated facility for each region that houses all the records. This facility is used only for the processing and transmitting of the medical records. This location uses whitelisting on the processors. It will also only transmit the records to trusted medical facilities. Each facility has unique key for accessing the datacenter, we will not go into the facility login process as we are limited on our space.

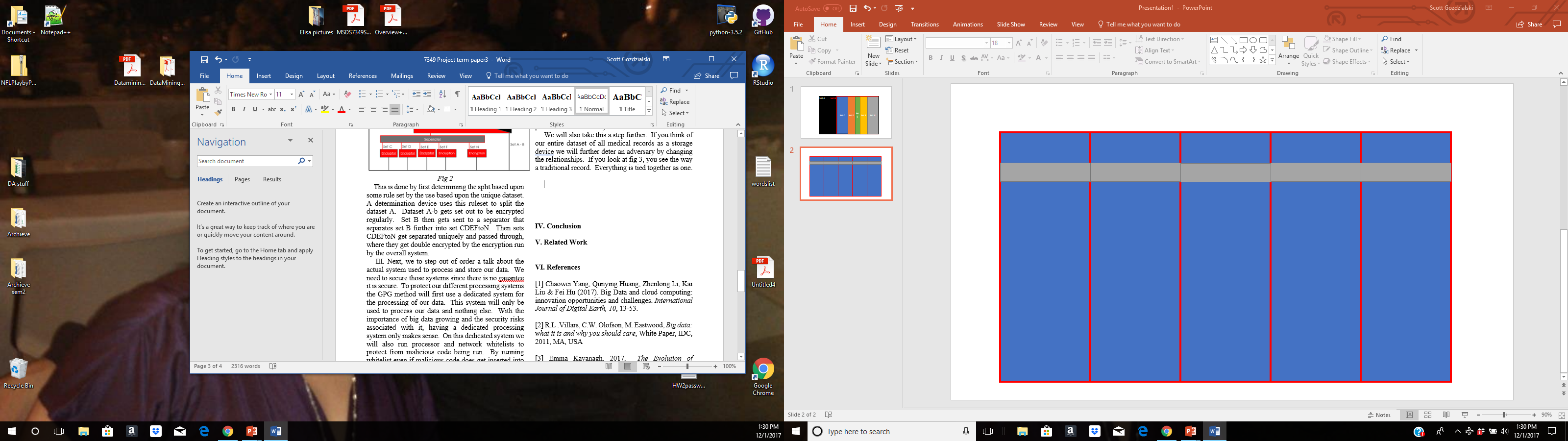
For the next part you will need to understand Oblivious data storage techniques. Oblivious data storage is data storage in a private cloud network based upon the oblivious data structure. It uses a technique where it obstructs the location of storage that is accessed for a piece of data by constantly moving the data to new locations each time the data is accessed. In other words, if you read a piece of data from the private cloud, you pull multiple pieces of data obstructing the actual data needed. When the data is then rewritten, it’s written in a new location, obscuring access pattern of our data. For our example with the medical records this will be used in our storage facility to secure our access patterns for an adversary.

IV. To protect the transmission of the information we will use the system described before in Fig 2. With each file subset getting a unique encryption and double wrapped with the IPSEC key from the network we secure our data in transmission. We further protect our data by using oblivious data storage techniques, like OblivStore [100] or TaoStore [100] to hide our access to the private cloud system.

We will also take this a step further. If you think of our entire dataset of all medical records as a storage device, we will further deter an adversary by changing the relationships. If you look at fig 3, you see the way a traditional record. Everything is tied together as one.

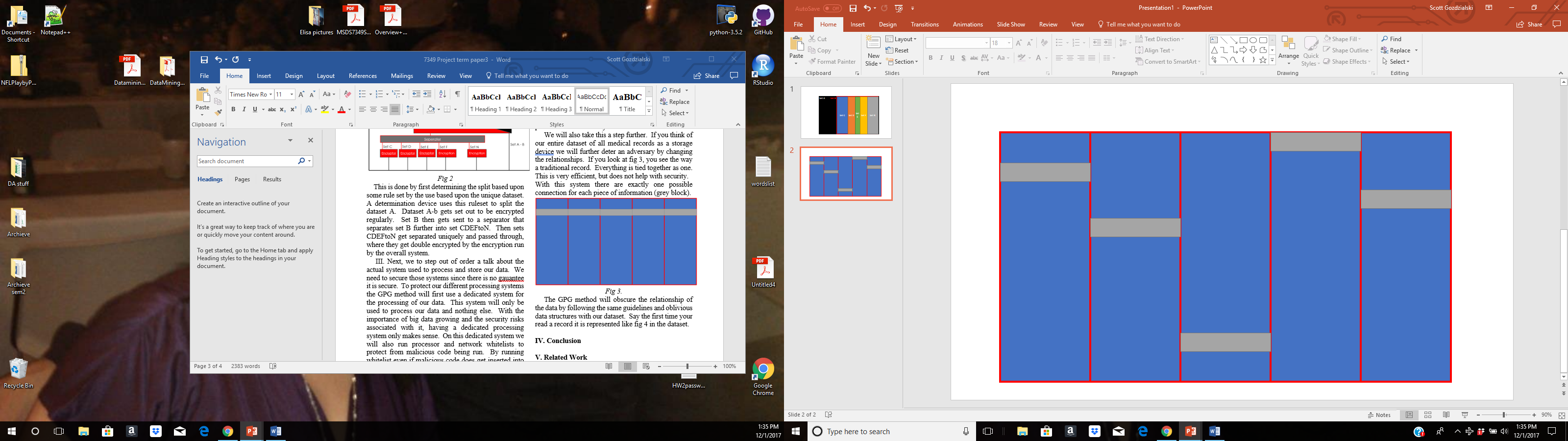
This is very efficient, but does not help with security.

With this system there are exactly one possible connection for each piece of information (grey block).



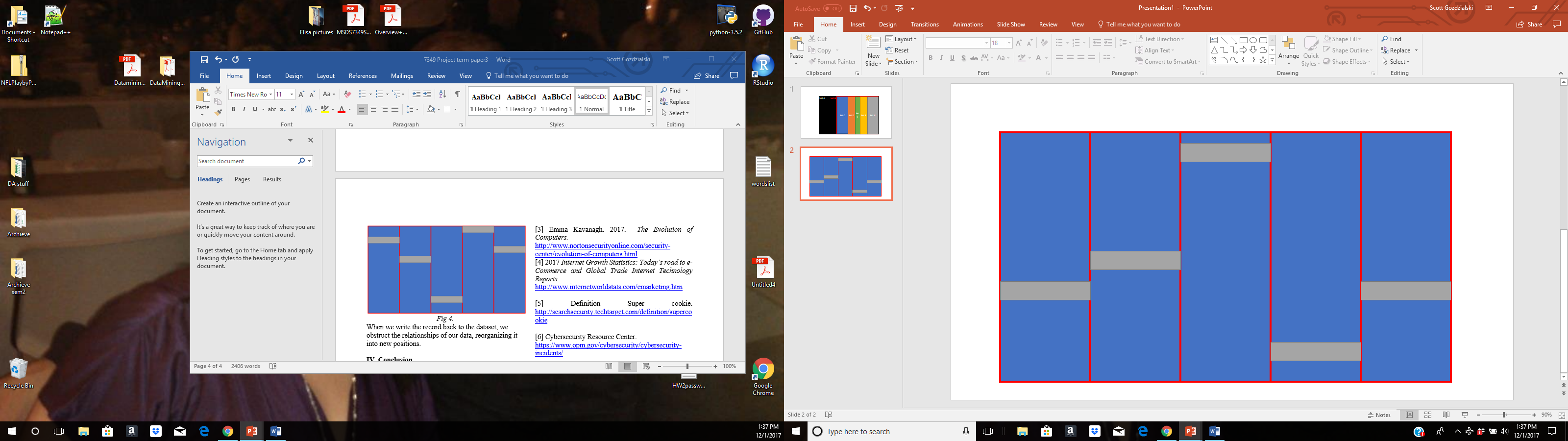
*Fig 3.*

The GPG method will obscure the relationship of the data by following the same guidelines and oblivious data structures with our dataset. Say the first time your read a record it is represented as fig 4 in the dataset.



*Fig 4.*

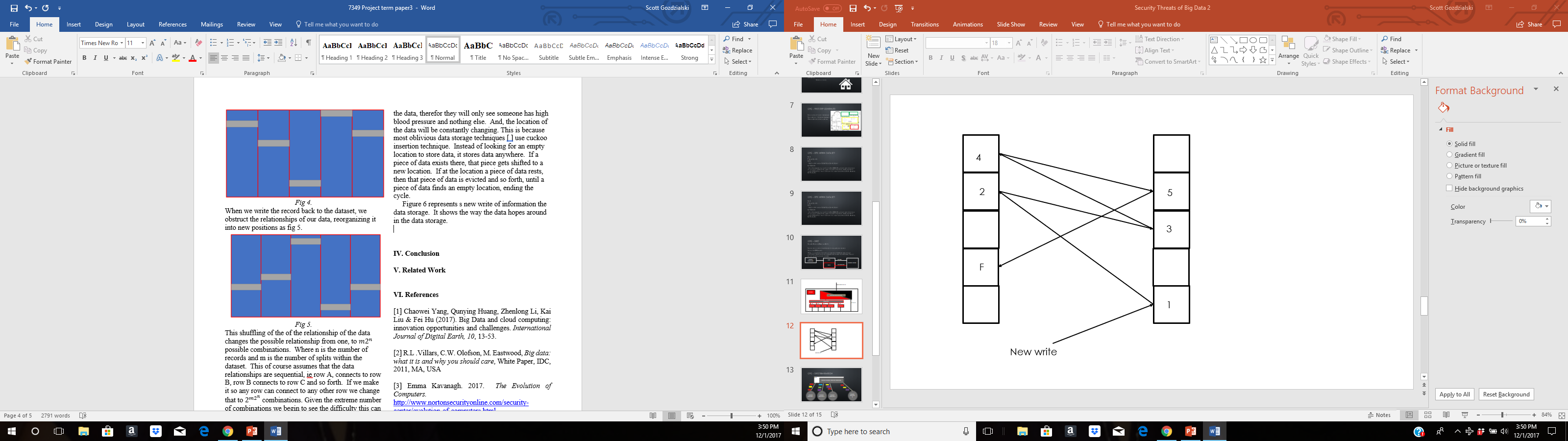
When we write the record back to the dataset, we obstruct the relationships of our data, reorganizing it into new positions as fig 5.



*Fig 5.*

This shuffling of the of the relationship of the data changes the possible relationship from one, to possible combinations. Where n is the number of records and m is the number of splits within the dataset. This of course assumes that the data relationships are sequential, i.e. row A, connects to row B, row B connects to row C and so forth. If we make it so any row can connect to any other row we change that to combinations. Given the extreme number of combinations we begin to see the difficulty this can bring for an adversary. For our example of the medical records, we use this technique to store the actual medical records in the clear, while encrypting and protecting the relationships between the different records. If we use the numbers discussed above, this gives us a total of or 1.052 \* combinations. If we pull a piece of a medical record and someone sees our information, they see multiple records pulled at random. They see someone broke an arm, but will not be able to determine the relationship of any records. When we add an appointment information we will not have to add all the data back, only the additional information back using the oblivious data storage technique. This will start a data shift of all records. This changes the storage system completely, securing the data at rest. The storage is secure because an adversary will not be able to read data off the storage system and see the relationship of the data, therefor they will only see someone has high blood pressure and nothing else. And, the location of the data will be constantly changing. This is because most oblivious data storage techniques [100] use cuckoo insertion technique. Instead of looking for an empty location to store data, it stores data anywhere. If a piece of data exists there, that piece gets shifted to a new location. If at the location a piece of data rests, then that piece of data is evicted and so forth, until a piece of data finds an empty location, ending the cycle.

Figure 6 represents the way we write information to the data storage using oblivious data storage technique. It shows the way the data hops around in the data storage.



*Fig 6*

**IV. Conclusion**

In this paper we have covered different ways to secure your data in a big data environment. We have discussed the different ways each of the current systems use to protect our information. We discussed the security of Cassandra, MongoDB, and SQL. How they focus on using different tools for each portion of security. The GPG method was discussed, starting with different concepts needed to start to understand our proposed process. We talked about how using set theory we can secure a dataset by dissecting the data and encrypting subsets and protecting the relationships. We showed how we can protect our data while being processed by using whiltelists to limit the processes run. We protected the data during transmission by using Oblivious data storage techniques to vary the location and time data is sent and by adding a second layer of encryption to the data based upon the subset it came from of the whole dataset. We showed how using the unique encryption for each subset and the Oblivious data storage again we can protect our data at rest. Finally, we used the example of locally stored medical records to show how the GPG method can be used to our dataset.

**V. Related Work**

The closest related work discussing the use of set theory to secure big data is the discussion of anonymizing data. Bhand and Chaudhari [14] discuss using map-reduce to anonymize data, but do not talk about using it to secure the dataset. Li, Qaraji, and Su [15]use *k-*anonymization to hide in a large crowd is feasible but not the implications on security. But, the discussion of securing big data is not new. Adrian Lane [16] discusses different shortfalls of Hadoop some recommended techniques to help secure they system. While Jonathan Buckley [17] talks more about general ideas and principle for securing you Hadoop system, but these ideas can be used for other systems. The discussion of Oblivious data storage is a new take on a old problem. Oblivious data storage is the use of Oblivious data structures, used for RAM access, used in a new way. Oblivious data storage does have some shortfalls. These are discussed by Emil Stefanov and Elaine Shi [18] in their paper “ObliviStore: High Performance Oblivious Cloud Storage” describing the technique to transfer this to a larger system. And Sahin, Zakhary, Abbadi, Kin, and Tessaro [19] look at securing Oblivious storage from timing analysis in their paper “TaoStore: Overcoming Asynchronicity in Oblivious Data Storage.”

**VI. Future work**

We discussed how set theory can secure our data. Also, we showed how using Oblivious storage tools and set encryption along with whitelisting can secure our data. We explained this using single records in a big dataset to secure the data. The next step is to show how set theory and these tools can secure large batches of data, terabytes and bigger, while limiting the impact on the resources.

**VII. References**

[1] Chaowei Yang, Qunying Huang, Zhenlong Li, Kai Liu & Fei Hu (2017). *Big Data and cloud computing: innovation opportunities and challenges*. International Journal of Digital Earth, 10, 13-53.

<http://www.tandfonline.com/doi/full/10.1080/17538947.2016.1239771?utm_source=TrendMD&utm_medium=cpc&utm_campaign=International_Journal_of_Digital_Earth_TrendMD_0>

[2] R.L .Villars, C.W. Olofson, M. Eastwood, *Big data: what it is and why you should care*, White Paper, IDC, 2011, MA, USA

<http://www.tracemyflows.com/uploads/big_data/idc_amd_big_data_whitepaper.pdf>

[3] Emma Kavanagh. 2017. *The Evolution of Computers.*

<http://www.nortonsecurityonline.com/security-center/evolution-of-computers.html>

[4] 2017 *Internet Growth Statistics: Today’s road to e-Commerce and Global Trade Internet Technology Reports.* <http://www.internetworldstats.com/emarketing.htm>

[5] Definition Super cookie. <http://searchsecurity.techtarget.com/definition/supercookie>

[6] Cybersecurity Resource Center.

<https://www.opm.gov/cybersecurity/cybersecurity-incidents/>

[7] Cybersecurity Incident & Important Consumer Information.

<https://www.equifaxsecurity2017.com/consumer-notice/>

[8]<https://docs.datastax.com/en/cassandra/3.0/cassandra/configuration/secureIntro.html>

[9] Robin Schumacher. 2013.

<https://www.datastax.com/dev/blog/a-quick-tour-of-internal-authentication-and-authorization-security-in-datastax-enterprise-and-apache-cassandra>

[10] MongoDB documentation

<https://docs.mongodb.com/manual/administration/security-checklist/>

[12] Danil Hutmacher. 2014. The SQL Server Security Model, part 1: principles

<https://sqlsunday.com/2014/07/20/the-sql-server-security-model-part-1/>

[12] Gregory A Larson. 2007. SQL Server Security Model . *Database Journal*

<https://www.databasejournal.com/features/mssql/article.php/3713116/SQL-Server-Security-Model.htm>

[13] Avita Katal, Mohammad Wazid, RH Gourder, 2013. *Big data: Issues, Challenges, Tools and Good Practices.*

<https://pdfs.semanticscholar.org/34ad/09cda075101dc4ce3c04006ff804aca3ebf8.pdf>

[14] Shweta Sunil Bhand, Prof J.L Chaudhari. 2014.

*A Scalable Two-Phase Top-Down Specialization Approach for Data Anonymization Using Map Reduce On Cloud.* International Journal of Innovative Research in Computer and Communication Engineering Vol. 2, Issue 11. <https://pdfs.semanticscholar.org/bfa8/6c2c70ee51f4482d4c61603d56b48afd9915.pdf>

[15] Ninghi Li, Wahbeh Qardaji, Dong su. 2010.

*Provably Private Data Anonymization: Or, k-Anonymity Meets Differential Privacy.*

<https://www.cerias.purdue.edu/assets/pdf/bibtex_archive/2010-24.pdf>

[16] Adrian Lane. 2016. *Securing Hadoop: Security Recommendations for Hadoop Environments.*

<https://securosis.com/assets/library/reports/Securing_Hadoop_Final_V2.pdf>

[17] Jonathon buckley. 2015. *Keeping Big data Safe: Common Hadoop Security Issues and best Practices.*

<https://www.qubole.com/blog/hadoop-security-issues/>

[18] Emil Stefanov, Elaine Shi. No date. *ObliviStore: High Performance Oblivious Cloud Storage.*

<http://elaineshi.com/docs/ObliviStore.pdf>

[19] Cetin Sahin, Victor Zakhary, Amr El Abbadi, Huijia Lin, Stefano Tessaro. No date. *TaoStore: Overcoming Asynchronicity in Oblivious Data Storage*

<http://www.cs.ucsb.edu/~rachel.lin/papers/TaoStore-CameraReady.pdf>