INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume3, Issue3)
Available online at www.ijariit.com

Implementing Secure and Efficient Auditing Protocol for Cloud Storage

Manisha L. Narad
G.H. Raisoni College of Engineering and MGMT, CHAS, Ahmednagar
Manishanarad14@gmail.com

Prof. Amruta Amune
G.H. Raisoni College of Engineering and MGMT, CHAS, Ahmednagar
amune.amruta@raisoni.net

Abstract: In cloud storage environment, data owners host their data on cloud servers and users can access the data from cloud servers. Due to the data outsourcing, this process of data hosting service introduces new security challenges, which requires an independent auditing service to check the data integrity in the cloud. Because owner needs to be convinced that the data are correctly stored in the cloud. Two-Party storage auditing system could not be guaranteed to provide proper auditing result thus Third-Party Auditing is the better choice for the storage auditing in cloud computing. In this, we try to evaluate ECC algorithm for encryption and decryption. A Third-Party Auditor is capable of doing a more efficient work and convinces both the cloud service providers and the owner. There are chances of data being lost or get misplaced in cloud storage environment. For this, we propose replication mechanism to third-party auditing such that it will enhance the data availability. We divide a data file into fragments and replicate the fragmented data over the cloud nodes. Each of the nodes stores only a single fragment of a particular data file that ensures that even in the case of a successful attack, no meaningful information is revealed to the attacker. Furthermore, the nodes storing the fragments are separated by certain distance to prohibit an attacker of guessing the locations of the fragments. Hence user will get the belief that his data is safely stored in the cloud and could retrieve data without any modification. To better protect data security, the first attempt to formally address the problem of authorized data deduplication.

Keywords: Cloud Storage Auditing, Network Coding, Security, User Anonymity, Third-Party Public Auditing, Deduplication, Confidentiality

I. INTRODUCTION

Cloud storage refers to saving data to an off-site storage system which is maintained by the third party. The information is stored on computer's hard drive or another local storage device, and it is the saved to the remote database through Internet connection between your computer and the database. Cloud storage has several advantages over traditional data storage. For example, if we store the data on a cloud storage system, it is easy to get that data from any location that has Internet access. We need not carry any physical storage device or use the same computer to save and retrieve your information. With the right storage system, we could even allow other people to access the data, turning a personal project into a collaborative effort. Cloud storage is convenient and offers more flexibility. Cloud storage provides benefits of greater accessibility and reliability, rapid deployment, strong protection for data backup and disaster recovery and also reduces the cost as no purchase is required to manage and maintain the expensive hardware. Data Integrity is the basic requirement of the information technology. As Data Integrity is an essential in databases similarly integrity of Data Storages is an essential in the cloud, it is a major factor affecting the performance of the cloud.

The data integrity provides the validity of the data, assuring the consistency or regularity of the data. It is the complete mechanism of the writing of the data in a reliable manner to the persistent data storages which can be retrieved in the same format without any changes. As described above, in the cloud, the complete storage of data provided by the end-user is done at the data centers or data storages, and the security and integrity of the data lie on the vendor storing data in the data centers but not the cloud hosts. Data deduplication is one of important data compression techniques for eliminating duplicate copies of repeating data and has been widely used in cloud storage to reduce the amount of storage space and save bandwidth. Cloud Storage is gaining popularity for the outsourcing of the day-to-day management of data.

Therefore integrity monitoring of data in cloud storages is as essential for any data center, to avoid any data corruption or data crash. Data corruption or data failure can occur at any storage level. Therefore just storing data at cloud data storages or data centers doesn't ensure the integrity of data, but some mechanisms need to be implemented at each storage level to ensure the data integrity. Data Integrity is most important of all the security issues in cloud data storages because it not only ensures completeness of data but also
ensures that the data is correct, accessible, and consistent and of high quality. To better protect data security, this paper makes the first attempt to formally address the problem of authorized data deduplication.

Secure cloud storage was firstly studied by Juels and Kaliski [4] and Ateniese et al [5]. In cloud computing, data owners host their data on cloud servers and users can access these data from cloud servers. That means data moves or outsource from its local computing system to the cloud. This data outsourcing introduces new security challenges. The Fig. 1 shows the Secure Cloud Storage System. This system consists of two entities Cloud and its user. Cloud could be any Cloud Service Provider such as Amazon's S3, Dropbox, Google Drive, etc. and user could be any individual

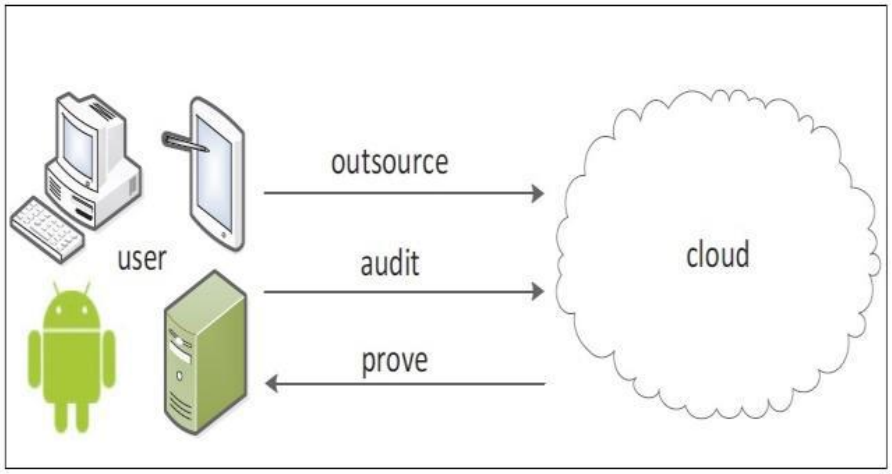

Fig 1. A Secure Cloud Storage System

or company or an organization that uses PC or mobile phone. To extend this model, a third-party auditor could be introduced [4] to shift the auditing task from the user to the third-party auditor. A secure cloud storage system that enables a user to check the integrity of its data is expected to have the following properties:

- *Correct*: If the cloud indeed stores the complete data of the user, then cloud can always prove to the user that the data remains intact.
- *Secure*: If the user's data is changed, the user can detect it as an abnormal event with high probability in the audit query even if the cloud tries to cover the event.
- *Efficient*: The computation, storage and communication cost of both the user and the cloud should be as low as possible. In the traditional approach, owners can check the data integrity based on two-party storage auditing protocols. In cloud storage system, it is inappropriate to let the cloud or user to conduct such auditing because they could not be guaranteed to provide auditing result. In this situation, third-party auditing is a natural choice for the storage auditing in cloud computing. A third party auditor has some capabilities to provide a more efficient work and convince both cloud service providers and owners. For the third-party auditing in cloud storage systems, there are several important requirements.

The auditing protocol should have the following properties:

- *Confidentiality*: The auditing protocol should keep owner's data confidential against the auditor.
- *Dynamic Auditing*: The auditing protocol should support the dynamic updates of the data in the cloud.
- *Batch auditing*: The auditing protocol should also be able to support the batch auditing for multiple owners and multiple clouds. There is some existing remote integrity checking methods which can only serve for static archive data and therefore they cannot be applied to the auditing service because the data in the cloud can be dynamically updated. Thus, an efficient and secure auditing protocol is desired to convince data owners that the data are correctly stored in the cloud. To verify whether the cloud lies to an audit query, the user needs to have some secret information on its side, which is computed according to a certain security level parameter using the probability of successful cheating. A secure cloud storage (SCS) protocol, a keyed protocol used for the user to generate data to be outsourced and subsequently query for auditing.

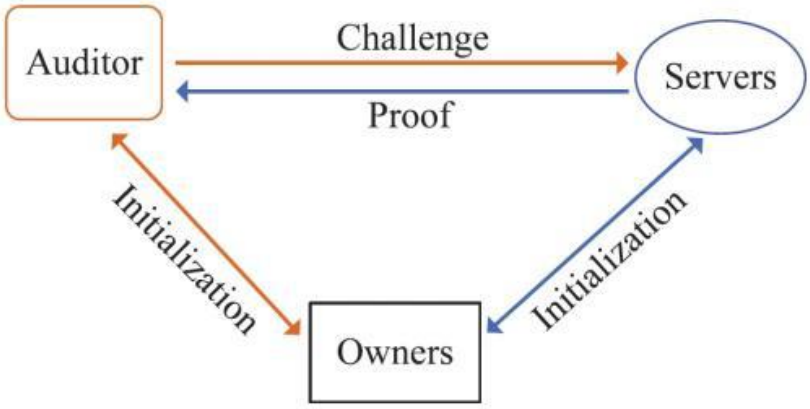

Fig.2 System model of the data storage auditing

II. LITERATURE SURVEY

A. System Model and Security Model

Fig 2 shows the data hosting process. This process involves communication among three entities Cloud server, Owners, and Auditor. The data verification is done by creating challenges and proofs for data integrity. We assume that the auditor is honest but strange. It performs honestly during the whole auditing procedure, but it is curious about the received data. But the server could be dishonest and may launch the following attacks:

- *Replace attack:*

The server may choose another valid and uncorrupted data block to replace the challenged data block when it already discarded the message.

- *Forge attack:*

The server may forge the data tag of the data block and deceive the auditor; if the owner's secret tag keys are reused for the different versions of data.

- *Replay attack:*

The server may generate the proof from the previous proof or other information, without retrieving the actual owner's data.

B. Review of Existing System

In 2014 K. Yang and X. Jia[3], An efficient and secure dynamic auditing protocol for data storage in cloud computing, This paper proposed an efficient auditing framework for cloud storage systems and privacy preserving auditing protocol for data storage in cloud computing. Then this protocol is extended to support dynamic operations. This new paradigm also introduces new security challenges. Owners would worry that the data could be lost in the cloud. Therefore, owners need to be convinced that the data are correctly stored in the cloud.

In 2014 Danan Thilakanathan, Shiping Chen, Surya Nepal and Rafael A. Calvo [4], Secure Data Sharing in the Cloud, Springer-Verlag Berlin Heidelberg. This paper proposed a model and protocol which gives the data owner greater control when sharing data via the Cloud. The proposed algorithm incorporating TPM devices that prevent illegal redistribution of data when sharing data with dishonest users in the distributed computing environment. This paper proposed a secure data sharing model in the cloud.

In 2007 A. Juels and B. Kaliski Jr [5], Pors: Proofs of retrievability for large files, in this paper the proofs of retrievability is defined and explored. A POR scheme enables or backup service to produce a concise proof that a user can retrieve a target file F that is that the archive retains and reliably transmits file data sufficient for the user to recover F in its entirety. The goal of a POR is to accomplish these checks without users having to download the files themselves. It also provides quality of service guarantees. In this variations on basic POR scheme with different sets of tradeoffs is explored.

In 2007 G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, Provable data possession at untrusted stores, In this paper a model for provable data possession is introduced which allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server which drastically reduces I/O costs. The Client maintains a constant amount of metadata to verify the proof. Two provably secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees.

In recent paper Chen, Xiang, et al [2], have designed a general construction of secure cloud storage protocol based on any secure network coding protocol. There have been a number of reviews on security and privacy in the Cloud. Xiao and Xiao [7] identifies the five concerns of Cloud Computing; confidentiality, integrity, availability, accountability, and privacy. Chen and Zhao [8] outline the requirements for achieving privacy and security in the Cloud and the requirements for secure data sharing in the Cloud.

Oza et al. [9] gave a survey on a number of users to determine the user experience of Cloud computing and found that the main issue of all users was trust and how to choose between different Cloud Service Providers. There are many examples [6] of insider attacks such as Google Docs containing a flaw that inadvertently shared user documents, MediaMax going out of business in 2008 after losing 45 of stored client data due to administrator error, Salesforce.com leaking a customer list and falling victim to phishing attacks on a number of occasions. It's clear from many of the reviews, that the Cloud is very susceptible to privacy and security attacks and currently there is on-going research that aims to prevent and or reduce the likelihood of such attacks. In 2012 Xiao Z, Xiao Y Security and privacy in cloud computing.

III.SYSTEM OVERVIEW

This section presents the system model of storage auditing protocol which gives the solution for data integrity checking and security model for a storage auditing system. As shown in Fig 3, upon receiving the file, the cloud manager system performs:

- Fragmentation-
- First cycle of nodes selection and stores one fragment over each of the selected node,
- The second cycle of nodes selection for fragments replication.
- The cloud manager keeps a record of the fragment placement and is assumed to be a secure entity. The fragmentation threshold of the data file is specified to be generated by the file owner. The file owner can specify the fragmentation threshold in terms of either percentage or the number and size of different fragments.
- As we are not considering secure network coding protocol, data checking process will be done at the receiver end which will help to reduce the load of network-level coding. In cloud storage when owner wants to share and maintain his data files on cloud server securely and efficiently, the flow will be
- At the sender end data to be shared, gets divided into a number of chunks and then encrypted along with the owner name, last modification and owner signature if required.
- On the auditing server machine, data get processed to check whether it is not being modified.
- At the receiver end when receiver access required data from the cloud, he decrypts the data by combining divided data chunks together to get the original data file.
- In this scenario, there are chances of data loss due to system crashes or any disaster. In such situation, owner's data should be protected from this and should be kept safely on the cloud server. For this purpose, we divide a data file into fragments and replicate the fragmented data over the cloud nodes. Each of the nodes stores only a single fragment of a particular data file that ensures that even in the case of a successful attack, no meaningful information is revealed to the attacker. Furthermore, the nodes storing the fragments are separated by certain distance to prohibit an attacker of guessing the locations of the fragments. Hence user will get the belief that his data is safely stored in the cloud and could retrieve data without any modification. Although the replication does not improve the retrieval time to the level of full-scale replication, it significantly improves the security.
- So that owner would remain unaware about such data loss situations and get his original data. Thus it helps to achieve data integrity.

This section describes the proposed algorithm (ECC algorithm).It shows flow auditing and security [fig.3] (module wise) for personal storage application.

- ECC Key Generation

1. Initialize an elliptic curve prime finite field with prime order q = Large Number with bit length of q
2. Initialize an elliptic curve with coefficients example a=Large Number and b=Large Number
3. Initialize a point on elliptic curve as G=Large Number and N=Large Number
4. Initialize a Secure Random key generator of elliptic curve points from above parameters
5. Generate system public key (db) and system security key (db)

- User Registration

1. Initialize random value r
2. Calc and mail user security key=private key EXOR r from curve using generator using random analysis
3. Initialize system hash key= (a) common for all documents (static)
4. During user registration, user hash key (db + mail) initialized as system hash key EXOR r
5. Store random value r (db)

- For the First upload

1. Get User hash key from user
2. Calculate system hash key = user hash key EXOR r
3. Calculate file hashtag h using system hash (HMAC) key
   Encrypt user uploaded data in ECC using system security key as below

4. Read generated public key Stored in Database and private key send to user for ECC
5. Initialize key specification for ECC-DES with (user security key EXOR r) and public key generated
6. Calculate secret key using user private key + user public key
7. Initialize ECC-DES encryption model using secret key
8. Convert input message to encrypted bytes using ECC-DES
9. Store hashtag with encrypted data
10. Store filename with hashtag

- For next upload:

1. Calculate system hash key = user hash key EXOR r
2. Calculate file hashtag h' using system hash key
3. Compare h' with h
4. If similar then
5. Store filename2 with hashtag h
6. Else
7. Calculate secret key using (user security key EXOR r) + user public key
8. Convert input message to encrypted bytes using ECC-DES
9. Store hashtag h' with encrypted data
10. Store filename2 with hashtag
11. End if

- Data Retrieval:

1. Read generated public key from database and private key from user for ECC
2. Initialize key specification for ECC-DES with (user security key EXOR r) and public key generated
3. Generate ECC-DES secret key using above ECC keys
4. Initialize ECC-DES decryption model using secret key
5. Identify location of originally encrypted bytes using deduplication concept above
6. Convert encrypted bytes to original message using ECC-DES

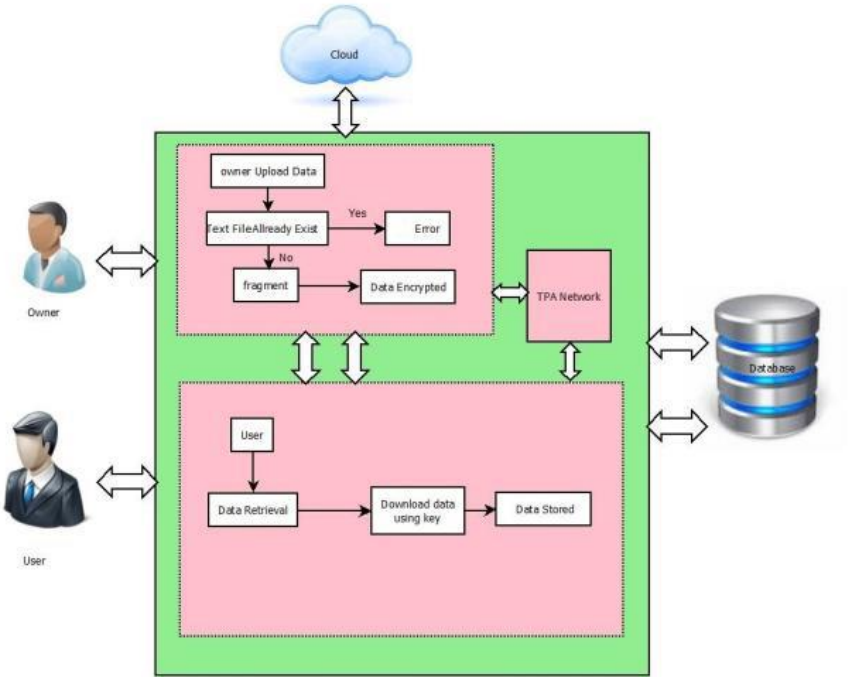

Fig.3 System architecture

This Section also describes the modules and their implementation.

- Data Owner Module:
  Owner register, at the time of registration private key and the public key generated and private key emailed to user mail id and a public key stored in DB.
- Upload File(.txt , .doc):

In this the file is uploaded by users. For uploaded file system generates HashTag and stored data in encrypted format and that specification is stored in the database.

- Deduplication Checking:
  This is part of upload module in this when the user is uploading the file. It checks whether the file is uploaded earlier or not. Then it compares the hashtag of uploading a file with already uploaded files. If its match is found then that file is not uploaded into the system.

- Encryption mechanism:
  The data will be encrypted using encryption ECC (Proposed Algo), the encrypted module allows users to encrypt the file that they have selected in the preceding module to store data. The file is divided into three places one is the main server and other is two fragment and each fragment is divided into two places and each fragment has its unique hashtag value.

- Data Retrieval module:
  Data uploaded by each user will be visible to each user. The retrieval module allows the user to download the data with the help of secret key sent by the owner to their respective E-mail id and public key store in DB. If is hacked then TPA does the access denied to a particular file then original user access file from the main server using a secret key and public key.

- TPA module:
  TPA will log in with his fixed credentials Hashtag value generated for each fragment will be used for verification TPA will verify each fragment stored in the environment and identify store validity status.

IV.SYSTEM ANALYSIS

This section describes the performance evaluation of the system. In this, we built a prototype of our newly constructed, the first publicly verifiable secure cloud storage protocol using Java NetBeans 7.2.1..For performance indicators storage cost, communication cost, and computation cost for both user and cloud is considered. These experiments are done on a PC with pentium IV and 40 GB memory. The protocol is run multiple times and average the performance indicators to obtain the predicted results.

- Storage Cost: For the user, it only needs to store the secret key; thus, the storage cost is two long integers p and q. For the cloud, it needs to store both the data and the authentication information. The additional storage cost of our protocol is for the authentication information. We can see that the storage cost linearly depends on the total number of blocks m. Thus, it is reasonable that we should increase the block size n to increase the data size. The storage cost can be very small if the block size is well chosen.
- Communication Cost: For the user, the communication cost depends on the length of the audit query, which is a constant and thus trivial. For the cloud, the communication cost consists of two parts: one is the linear combination of the queried data blocks and the other is the authentication information. The former is the same as the block size; the latter depends on the size of authentication information. The experimental result is also shown in Table 1, which only focuses on the additional authentication cost since the block size is fixed.
- Computation Cost: For the user and the cloud, the computation cost is composed of four parts, namely, the time for outsourcing, auditing, proving, and verifying the data. Outsourcing the data takes much longer time than other operations. The time cost grows higher when the data size becomes larger. In theory, it depends on the data size, the block size, and the total number of blocks.

TABLE I
ADDITIONAL STORAGE AND COMMUNICATION COST

Benchmark	n	M	Storage	Communication
1	2KB	120	0.08KB	378K
2	2KB	349	0.50KB	378K
3	4KB	4999	2.333MB	378K
4	4KB	100	0.70KB	378K
5	4KB	150	0.80KB	378K

RESULTS

The results show that the protocol is practically usable. The efficiency of the protocol is better than the previous protocol and that efficiency could be further optimized. Fig.4.shows the graphical representation of the result i.e. encryption performed on uploaded a file using ECC encryption algorithm and it is then compared with encryption performed using AES encryption algorithm for testing the response time.Fig.5. Represents the output for decryption performed using ECC algorithm and compared with the output for the same by applying AES algorithm.

Fig 4. Difference while encrypting uploaded file

Fig. 5. Difference while decrypting uploaded file

CONCLUSION

In this work, we proposed an efficient and secure storage auditing protocol. It protects the data privacy against the various security concerns. It also assures the data integrity as we are taking backup of this data into slave cloud server. As most of the computation is processed on auditing server, the load on cloud server gets reduced. It presents the replication mechanism to the third party auditing that it will enhance the data availability. The data file was fragmented and the fragments are dispersed over multiple nodes. The nodes were separated by certain distance. The fragmentation and ensured that no significant information was obtainable by an adversary in the case of a successful attack. Hence, the owner would remain unaware about such data loss situations and get his original data. Thus it helps to achieve data integrity, data availability as well its confidentiality. It is strategic to develop an automatic update mechanism that can identify and update the required fragments only. The future work will save the time and resources utilized in downloading, updating, and uploading the file again.

ACKNOWLEDGMENT

I Miss. Manisha Laxman Narad would like to thank all the researchers and the Department of Computer Engineering G.H.Raisoni College Of Engineering And Management, Chas,

REFERENCES

[1] Fei Chen,Tao Xiang,Yuanyuan Yang,Sherman S. M. Chow, "Secure Cloud Storage Meets with Secure Network Coding", IEEE INFOCOM 2014 - IEEE Conference on Computer Communications.
[2] Jin Li, Yan Kit Li, Xiaofeng Chen, Patrick P. C. Lee, Wenjing Lou, " A Hybrid Cloud Approach for Secure Authorized Deduplication" DOI 10.1109/TPDS.2014.2318320, IEEE Transactions on Parallel and Distributed Systems
[3] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 9, pp. 17171726, 2013.
[4] Danan Thilakanathan, Shiping Chen, Surya Nepal, and Rafael A. Calvo,"Secure Data Sharing in the Cloud", Springer-Verlag Berlin Heidelberg 2014.
[5] A. Juels and B. Kaliski Jr, "Pors: Proofs of retrievability for large files, "in ACM Conference on Computer and Communications Security (SP), 2007, pp. 584597.
[6] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in ACM Conference on Computer and Communications Security (CCS), 2007, pp. 598609.
[7] Huang R, Gui X, Yu S, ZhuangW (2011) Research on privacy-preserving cloud storage framework supporting cipher text retrieval. International conference on network computing and information security 2011:9397
[8] Xiao Z, Xiao Y (2012) Security and privacy in cloud computing. IEEE Commun Surveys Tutorials 99:117
[9] Chen D, Zhao H (2012) "Data security and privacy protection issues in cloud computing". International conference on computer science and electronics, engineering, pp 647651.
[10] Oza N, Karppinen K, Savola R (2010) User experience and security in the cloud-An empirical study in the Finnish cloud consortium. IEEE second international conference on cloud computing technology and Science (CloudCom) 2010:621628.
[11] Sarathy R, Muralidhar K (2006) Secure and useful data sharing. Decis Support Syst 204220.
[12] Butler D Data sharing threatens privacy, vol 449(7163).Nature Publishing, Group, pp 644645.
[13] Feldman L, Patel D, Ortmann L, Robinson K, Popovic T (2012) Educating for the future: another important benefit of data sharing. Lancet 18771878.

