ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Implementation of Key Exposure Resistance Method in Cloud Storage

Huma Kausar, B.E., MCM, M.Tech (CSE)
Assistant Professor,
Dept. of ECM (Electronics & Computer Engineering),
JB Institute of Engineering & Technology,
Hyderabad.

D Sai Krishna
Student,
Dept. of ECM (Electronics & Computer Engineering),
JB Institute of Engineering & Technology,
Hyderabad.

P.Neeraj
Student,
Dept. of ECM (Electronics & Computer Engineering),
JB Institute of Engineering & Technology,
Hyderabad.

Puneeth V
Student,
Dept. of ECM (Electronics & Computer Engineering),
JB Institute of Engineering & Technology,
Hyderabad.

ABSTRACT

Auditing is an important service to verify the data in the cloud. Most of the auditing protocols are based on the assumption that the client's secret key for auditing is secure. The security is not fully achieved, because of the low security parameters of the client. If the auditing protocol is not secured means the data of the client will exposed inevitably. In this paper a new mechanism of cloud auditing is implemented.

And investigate to reduce the damage of the client key exposure in cloud storage auditing. Here the designing is built upon to overcome the week key auditing process. The auditing protocol is designed with the help of key exposure resilience. In the proposed design, the binary tree structure and the pre-order traversal technique is used to update the secret keys of the client. The security proof and the performance shows the cloud storage auditing with key exposure resilience is very efficient.

INTRODUCTION

Cloud storage auditing is used to verify the integrity of the data stored in public cloud, which is one of the important security techniques in cloud storage. In recent years, auditing protocols for cloud storage have attracted much attention and have been researched intensively these protocols focus on several different aspects of auditing, and how to achieve high bandwidth and computation efficiencyis one of the essential concerns. For that purpose, theHomomorphic Linear Authenticator (HLA) technique that supports blockless verification is explored to reduce the overheadsof computation and communication in auditing protocols, which allows the auditor to verify the integrity of the data in cloud without retrieving the whole data. Many cloud storage auditing protocols like have been proposed based on this technique. The privacy protection of data is also an important aspect of cloud storage auditing. In order to reduce the computational burden of the client, a third-party auditor (TPA) is introduced to help the client to periodically check the integrity of the data in cloud.

However, it is possible for the TPA to get the client's data after it executes the auditing protocol multiple times. Auditing protocols are designed to ensure the privacy of the client's data in cloud. Another aspect having been addressed in cloud storage auditing is how to support data dynamic operations. An n get al. has proposed an auditing protocol supporting fully dynamic data operations including modification, insertion and deletion. Auditing protocols can also support dynamic data operations. Other aspects, such as proxy auditing, user revocation and eliminating certificate management in cloud storage auditing have also been studied. Though many research works about cloud storage auditing have been done in recent years, a critical security problem—the key exposure problem for cloud storage auditing, has remained unexplored in previous researches. While all existing protocols focus on the faults or dishonesty of the cloud, they have overlooked the possible weak sense of security and/or low security settings at theclient.In fact, the client's secret key for cloud storage auditing maybe exposed, even known by the cloud, due to several reasons. Firstly, the key management is a very complex procedure which involves many factors including system policy, user training, etc. One client often needs to manage varieties of keys to complete different security tasks. Any careless mistake or fault in managing these keys would make the key exposure possible. It is not uncommon to see a client choosing tousle cheap software-based key management for economical factors, which may only provide limited protection and make the sensitive secret keys vulnerable to exposure.

Secondly, the client himself may be the target and vulnerable to many Internet based security attacks. For an ordinary client, the sense of security protection can be relatively weaker, compared with the case of enterprises and organizations. Hence, it is possible for a client to unintentionally download malicious software from Internet or to overlook the timely security patch to their computer system. Both of these cases could give the hacker easy access to their secret keys. Last but not the least, the cloud also has incentives to get clients' secret keys for storage auditing, e.g., through trading with the aforementioned hackers. Specifically, if the cloud gets these keys, it can regenerate the fake data and forge their authenticators to easily hide the data loss incidents, e.g., caused by Byzantine failures, from the client, while maintaining its reputation. In the malicious case, it can even discard the client's data that are rarely accessed to save the storage space, without worrying about failure to pass the auditing protocol initiated by the client. Obviously, the auditing secret key exposure could be disastrous for the clients of cloud storage applications. Therefore, how to deal with the client's secret key exposure for cloud storage auditing is a very important problem. Unfortunately, previous auditing protocols did not consider this critical issue, and any exposure of the client's secret auditing key would make most of the existing auditing protocols unable to work correctly. In this paper, we focus on how to reduce the damage of the client's key exposure in cloud storage auditing. Our goal is to design a cloud storage auditing protocol with built-in key-exposure resilience. How to do it efficiently under this new problem setting brings in many new challenges to be addressed below. First of all, applying the traditional solution of key revocation to cloud storage auditing is not practical. This is because, whenever the client's secret key for auditing is exposed, the client needs to produce a new pair of public key and secret key and regenerate the authenticators for the client's data previously stored in cloud. The process involves the downloading of whole data from the cloud, producing new authenticators, and re-uploading everything back to the cloud, all of which can be tedious and cumbersome. Besides, it cannot always guarantee that the cloud provides real data when the client regenerates new authenticators. Secondly, directly adopting standard key-evolving technique is also not suitable for the new problem setting. It can lead to retrieving all of the actual files blocks when the verification is preceded. Thesis partly because the technique is incompatible with block less verification. The resulting authenticators cannot be aggregated, leading to unacceptably high computation and communication cost for the storage auditing.

EXISTING SYSTEM:

These protocols focus on several different aspects of auditing, and how to achieve high bandwidth and computation efficiency is one of the essential concerns. For that purpose, the Homomorphic Linear Authenticator (HLA) technique that supports blockless verification is explored to reduce the overheads of computation and communication in auditing protocols, which allows the auditor to verify the integrity of the data in cloud without retrieving the whole data.

The privacy protection of data is also an important aspect of cloud storage auditing. In order to reduce the computational burden of the client, a third-party auditor (TPA) is introduced to help the client to periodically check the integrity of the data in cloud. However, it is possible for the TPA to get the client's data after it executes the auditing protocol multiple times.

Wang et al. have proposed an auditing protocol supporting fully dynamic data operations including modification, insertion and deletion.

DISADVANTAGES OF EXISTING SYSTEM:

- Though many research works about cloud storage auditing have been done in recent years, a critical security problem—the key exposure problem for cloud storage auditing, has remained unexplored in previous researches. While all existing protocols focus on the faults or dishonesty of the cloud, they have overlooked the possible weak sense of security and/or low security settings at the client.
- Unfortunately, previous auditing protocols did not consider this critical issue of how to deal with the client's secret key exposure for cloud storage auditing, and any exposure of the client's secret auditing key would make most of the existing auditing protocols unable to work correctly.

PROPOSED SYSTEM:

In this paper, we focus on how to reduce the damage of the clients key exposure in cloud storage auditing. Our goal is to design a cloud storage auditing protocol with built-in key-exposure resilience. How to do it efficiently under this new problem setting brings in many new challenges to be addressed below. First of all, applying the traditional solution of key revocation to cloud storage auditing is not practical. This is because, whenever the client's secret key for auditing is exposed, the client needs to produce a new pair of public key and secret key and regenerate the authenticators for the client's data previously stored in cloud.

Our goal is to design a practical auditing protocol with key-exposure resilience, in which the operational complexities of key size, computation overhead and communication overhead should be at most sub-linear to T. In order to achieve our goal, we use a binary tree structure to appoint time periods and associate periods with tree nodes by the pre-order traversal technique. The secret key in each time period is organized as a stack. In each time period, the secret key is updated by a forward-secure technique.

The auditing protocol achieves key-exposure resilience while satisfying our efficiency requirements. As we will show later, in our protocol, the client can audit the integrity of the cloud data still in aggregated manner, i.e., without retrieving the entire data from the cloud.

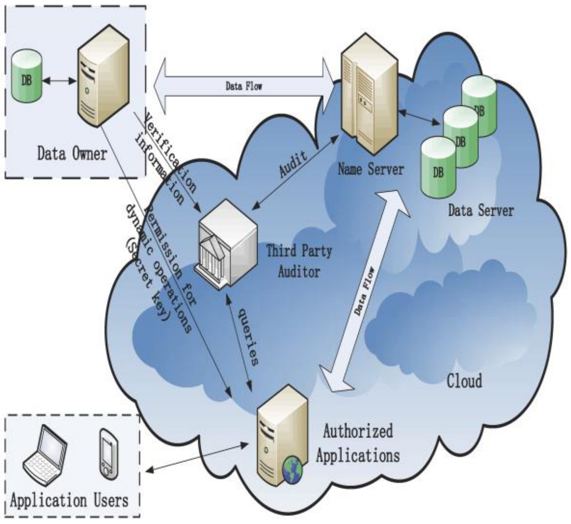
ADVANTAGES OF PROPOSED SYSTEM:

- We initiate the first study on how to achieve the key-exposure resilience in the storage auditing protocol and propose a new concept called auditing protocol with key-exposure resilience. In such a protocol, any dishonest behaviors, such as deleting or modifying some client's data stored in cloud in previous time periods, can all be detected, even if the cloud gets the client's current secret key for cloud storage auditing.
- This very important issue is not addressed before by previous auditing protocol designs. We further formalize the definition and the security model of auditing protocol with key-exposure resilience for secure cloud storage.
- We design and realize the first practical auditing protocol with built-in key-exposure resilience for cloud storage. In order to achieve our goal, we employ the binary tree structure, seen in a few previous works on different cryptographic designs, to update the

secret keys of the client. Such a binary tree structure can be considered as a variant of the tree structure used in the HIBE scheme. In addition, the pre-order traversal technique is used to associate each node of a binary tree with each time period. In our detailed protocol, the stack structure is used to realize the pre-order traversal of the binary tree. We also design a novel authenticator supporting the forward security and the property of blockless verifiability.

- We prove the security of our protocol in the formalized security model, and justify its performance via concrete asymptotic analysis. Indeed, the proposed protocol only adds reasonable overhead to achieve the key-exposure resilience. We also show that our proposed design can be extended to support the TPA, lazy update and multiple sectors.

SYSTEM ARCHITECTURE:



MODULES DESCRIPTION:

Client:

The client produces files and uploads these files along with corresponding authenticators to the cloud. The client can periodically audit whether his files in cloud are correct. The client will update his secret keys for cloud storage auditing in the end of each time period, but the public key is always unchanged.

TPA:

In order to reduce the computational burden of the client, a third-party auditor (TPA) is introduced to help the client to periodically check the integrity of the data in cloud. However, it is possible for the TPA to get the client's data after it executes the auditing protocol multiple times.

Auditing protocols are designed to ensure the privacy of the client's data in cloud. Another aspect having been addressed in cloud storage auditing is how to support data dynamic operations.

Cloud:

Cloud computing is a model for enabling ubiquitous, convenient, on-demand access to a shared pool of configurable computing resources. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers. It relies on sharing of resources to achieve coherence and economies of scale, similar to a utility (like the electricity grid) over a network. At the foundation of cloud computing is the broader concept of converged infrastructure and shared services.

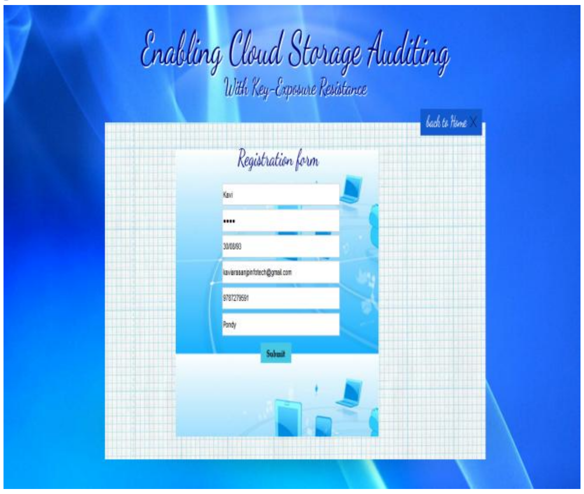
Key Exposure Resistance:

The client needs to produce a new pair of public key and secret key and regenerate the authenticators for the client's data previously stored in cloud. There is a one-time public key sharing for each file and a Time Stamp based secret key Generation. For each instance the timestamp based key exposure will be vary according to the current time stamp.

SCREEN SHOTS:

Home:

Registration:



Kavi

••••

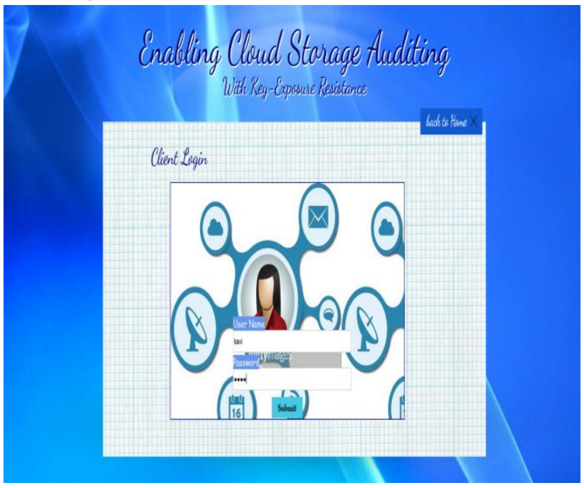
30/08/93

kaviarasanpinfotech@gmail.com

9787279591

Pondy

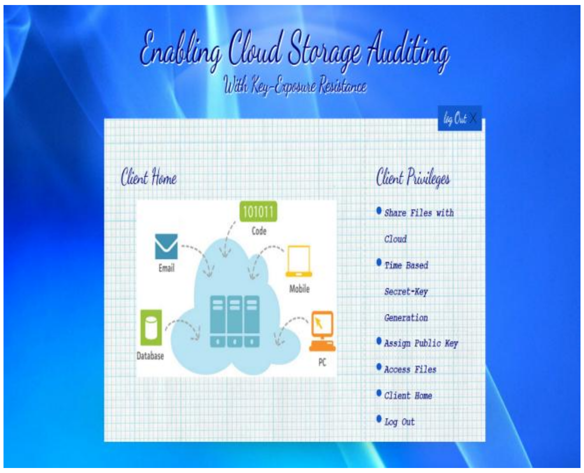
Client Login:



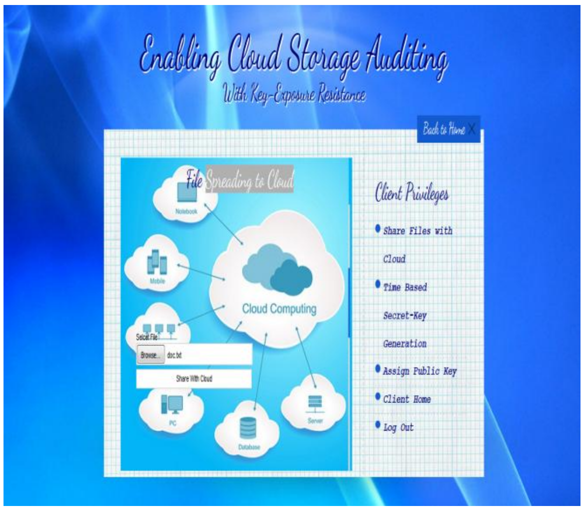
User Name: kavi

Password: ••••

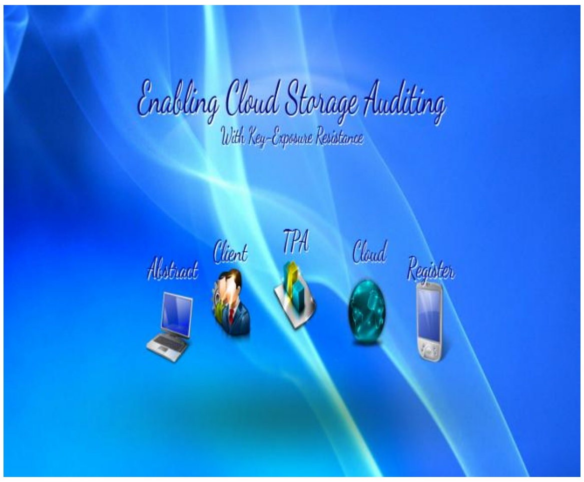
Client Home:



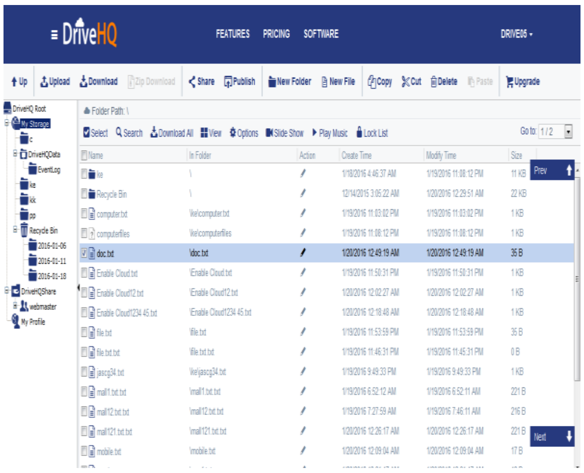
Share File in Cloud:



Cloud:



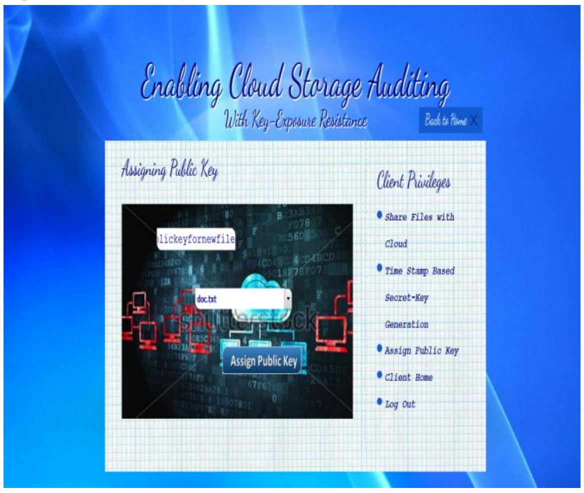
File Stored into Cloud:



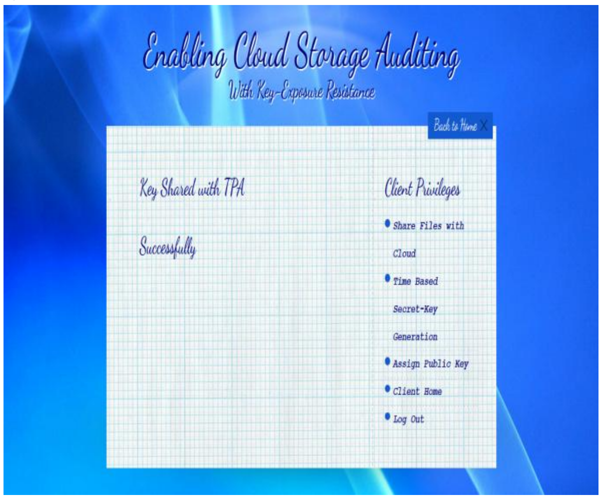
Folder Path: \

Name	In Folder	Action	Create Time	Modify Time	Size
ke	\		1/18/2016 4:46:37 AM	1/19/2016 11:08:12 PM	11 KB
Recycle Bin	\		12/14/2015 3:05:22 AM	1/20/2016 12:29:51 AM	22 KB
computer.txt	\ke\computer.txt		1/19/2016 11:03:02 PM	1/19/2016 11:03:02 PM	1 KB
computerfiles	\ke\computerfiles		1/19/2016 11:08:12 PM	1/19/2016 11:08:12 PM	1 KB
doc.txt	\doc.txt		1/20/2016 12:49:19 AM	1/20/2016 12:49:19 AM	35 B
Enable Cloud.txt	\Enable Cloud.txt		1/19/2016 11:50:31 PM	1/19/2016 11:50:31 PM	1 KB
Enable Cloud12.txt	\Enable Cloud12.txt		1/20/2016 12:02:27 AM	1/20/2016 12:02:27 AM	1 KB
Enable Cloud1234 45.txt	\Enable Cloud1234 45.txt		1/20/2016 12:18:48 AM	1/20/2016 12:18:48 AM	1 KB
file.txt	\file.txt		1/19/2016 11:53:59 PM	1/19/2016 11:53:59 PM	35 B
file.txt.txt	\file.txt.txt		1/19/2016 11:46:31 PM	1/19/2016 11:45:31 PM	0 B
jascg34.txt	\ke\jascg34.txt		1/19/2016 9:49:33 PM	1/19/2016 9:49:33 PM	1 KB
mail1.txt.txt	\mail1.txt.txt		1/19/2016 6:52:12 AM	1/19/2016 6:52:11 AM	221 B
mail12.txt.txt	\mail12.txt.txt		1/19/2016 7:27:59 AM	1/19/2016 7:46:11 AM	216 B
mail121.txt.txt	\mail121.txt.txt		1/20/2016 12:26:17 AM	1/20/2016 12:26:17 AM	221 B
mobile.txt	\mobile.txt		1/20/2016 12:09:04 AM	1/20/2016 12:09:04 AM	17 B

Assign Public Key:



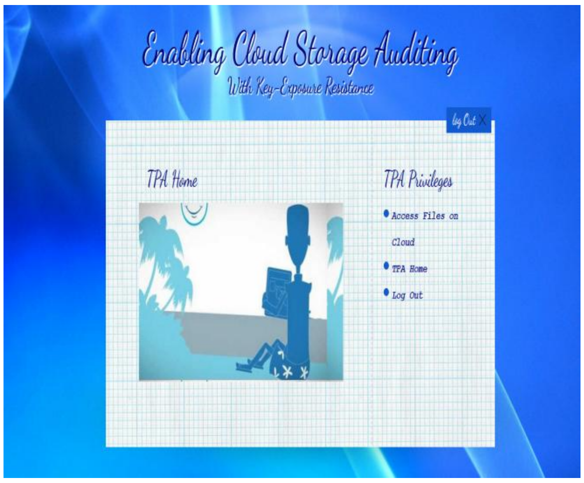
Key Shared with TPA:


Enabling Cloud Storage Auditing
With Key-Exposure Resistance
Back to Home
Client Privileges
Share Files with Cloud
Time Based Secret-Key Generation
Assign Public Key
Client Home
Log Out

Key Shared with TPA

Successfully

TPA Home:



CONCLUSION

In this paper, we study on how to deal with the client's key exposure in cloud storage auditing. We propose a new paradigm called auditing protocol with key-exposure resilience. In such a protocol, the integrity of the data previously stored in cloud can still be verified even if the client's current secret key for cloud storage auditing is exposed. We formalize the definition and the security model of auditing protocol with key-exposure resilience, and then propose the first practical solution. The security proof and the asymptotic performance evaluation show that the proposed protocol is secure and efficient.

REFERENCES

[1] Jia Yu, Kui Ren, Senior Member, IEEE, Cong Wang, Member, IEEE, and Vijay Varadharajan, Senior Member, IEEE, "Enabling Cloud Storage Auditing With Key-Exposure Resistance", IEEE Transactions On Information Forensics And Security, Vol. 10, No. 6, June 2015.

[2] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. 4th Int. Conf. Secur. Privacy Commun. Netw., 2008, Art. ID 9.

[3] F. Sebe, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," IEEE Trans. Knowl. Data Eng., vol. 20, no. 8, pp. 1034–1038, Aug. 2008.

[4] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiplereplica provable data possession," in Proc. 28th IEEE Int. Conf. Distrib. Comput. Syst., Jun. 2008, pp. 411–420.

[5] H. Shacham and B. Waters, "Compact proofs of retrievability," in Advances in Cryptology—ASIACRYPT. Berlin, Germany: Springer-Verlag, 2008, pp. 90–107.

[6] C. Wang, K. Ren, W. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," IEEE Netw., vol. 24, no. 4, pp. 19–24, Jul./Aug. 2010.

[7] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Efficient provable data possession for hybrid clouds," in Proc. 17th ACM Conf. Comput. Commun. Secur., 2010, pp. 756–758.

[8] K. Yang and X. Jia, "Data storage auditing service in cloud computing: Challenges, methods and opportunities," World Wide Web, vol. 15, no. 4, pp. 409–428, 2012.

[9] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 9, pp. 1717–1726, Sep. 2013.

[10] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacypreserving public auditing for secure cloud storage," IEEE Trans Comput., vol. 62, no. 2, pp. 362–375, Feb. 2013.

[11] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 5, pp. 847–859, May 2011.

[12] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for outsourced storages in clouds," IEEE Trans. Services Comput., vol. 6, no. 2, pp. 227–238, Apr./Jun. 2013.

[13] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in Proc. 16th ACM Conf. Comput. Commun. Secur., 2009, pp. 213–222.

[14] H. Wang, "Proxy provable data possession in public clouds," IEEE Trans. Services Comput., vol. 6, no. 4, pp. 551–559, Oct./Dec. 2013.

[15] B. Wang, B. Li, and H. Li, "Public auditing for shared data with efficient user revocation in the cloud," in Proc. IEEE INFOCOM, Apr. 2013, pp. 2904–2912.

[16] H. Wang, Q. Wu, B. Qin, and J. Domingo-Ferrer, "Identity-based remote data possession checking in public clouds," IET Inf. Secur., vol. 8, no. 2, pp. 114–121, Mar. 2014.

[17] T. Stewart. (Aug. 2012). Security Policy and Key Management: Centrally Manage Encryption Key. [Online]. Available: http://www.slideshare. net/Tina-stewart/security-policy-and-enterprise-key-management-fromvormetric

[18] Microsoft. (2014). Key Management. [Online]. Available: http://technet.microsoft.com/en-us/library/cc961626.aspx

[19] FBI. (2012). Is Your Computer Infected with DNSChanger Malware?. [Online]. Available: http://www.fbi.gov/news/news_blog/is-yourcomputer-infected-with-dnschanger-malware