Efficient Data Storage Security with Multiple Batch Auditing in Cloud Computing

P. Sukumar[1]
Department of Computer Science
Sri Venkateswara College of Engineering, Chennai

B. Sathiya [2]
Department of Computer Science
Sri Venkateswara College of Engineering, Chennai

K. Venu [3]
Department of Computer Science
Kongu Engineering College
Perundurai

***Abstract*— Cloud Computing has been visualized as the next-generation architecture of IT endeavor. It moves the storage databases to the centralized large data centers, where the direction of the data and services may not be trustworthy. This fetches many new security challenges, which have not been well understood properly. This work deals with the problem of ensuring the integrity of data stored in the cloud. In the proposed system, we consider the task of allowing a third party auditor to verify the integrity of the data stored in the cloud on behalf of the client. The entry of third party auditor eliminates the participation of the client through the auditing of whether his/her data stored in the cloud are indeed not damaged, which can be important in cloud computing for achieving economies of scale. The support for data dynamic operations such as block modification, insertion, and deletion are also a significant step toward practicality, since services in cloud computing are not restrict to file away or backup data only. While prior works on assuring remote data integrity often miss the support of either public auditability or data dynamic operations, this work achieves both services. These difficulties and potential security problems of direct propagations with fully dynamic updates from existing works has been identified and then we show how to construct a refined verification scheme for the unlined integration of these two characteristics in our proposed protocol design. To attain efficient data dynamics, we amend the existing proof of storage models by manipulating the classic Merkle Hash Tree generation for authentication. To support effective manipulation of multiple batch auditing tasks, the technique called bilinear aggregate signature has been used and third party auditor can perform multiple auditing tasks instantly. The security and performance analysis shows that the proposed system is efficient and provably more secure.**

Keywords—cloud storage; data dynamics ; third party auditor; batch auditing.

I. INTRODUCTION

Several technologies has been proposed for the epoch of cloud computing, which is an internet based evolution and use of computer system. The ever cheap and more powerful processors, collectively with the Software as a Service (SaaS) computing architecture, are transforming data centres into kitty of computing services. The rapid increase in network bandwidth and reliable yet flexible network connections makes it even possible that clients can subscribe high calibre services from data and software that reside on hostile data centres.

Although cloud computing visualized as a promising service platform for the Internet, this new data storage epitome in cloud brings about many challenging issues which have fundamental tempt on the security and performance of the overall computer system. One of the biggest interest with cloud storage is that of data integrity confirmation at remote servers. For example, the service provider, which undergo Byzantine failures from time to time, may decide to conceal the data errors from the clients for the profit of their own. The more serious is that for preserving money and storage space the service provider might ignore to keep or intentionally delete rarely accessed data files which belong to an average client. It is impossible for the client to check the integrity of their data with own periodically.

Considering the role of the verifier in the model, all the methods proposed before fall into two categories such as private auditability and public auditability. Although method with private auditability can yield higher efficiency, public auditability permit anyone, not just the data proprietor, to challenge the cloud server for rightness of data storage while keeping no secret information. The clients are able to depute the rating of the service performance to an independent third party auditor without idolatry of their computation resources. The clients themselves are treacherous or may not be able to afford the overhead of performing frequent integrity verification in the cloud. Thus, for virtual use, it appears more rational to fit out the verification protocol with public auditability, which is anticipated to play a more important role in achieving economies of scale for cloud computing. For efficiency consideration, the out sourced data themselves should not be required by the verifier for the verification purpose.

In perspective of the key role of public auditability and data dynamics for cloud data storage, we proposed an effective construction for the unlined integration of these two components in the protocol design. The following contributions are done through this research work:

- The proposed system actuate the public auditing system of data storage security in cloud computing,

and suggested a protocol supporting for fully dynamic data operations, particularly to support block insertion, which is missing in most existing schemes.

- We extend our method to support scalable and efficient public auditing in cloud computing. This method achieves batch auditing where multiple depute auditing tasks from different users can be performed instantly by the third party.
- We prove the security of our proposed system and vindicate the performance of our method through concrete implementation and comparisons with the state of the art.

II. Related Work

Recently, much of growing interest has been chased in the context of remotely stored data confirmation. In their methods, they utilize RSA-based homomorphic tags for auditing outsourced data to achieve public auditability. Nevertheless, Ateniese et al. do not study the case of dynamic data storage and the direct extension of their method from static data storage to dynamic case may endure design and security problems. In their subsequent developments, Ateniese et al. proposed a dynamic version of the prior Provable Data Possession (PDP) scheme. On the other hand, the system enforce a priori bound on the number of queries and does not support fully dynamic data operations where spot-checking and error-correcting codes are used to ensure both ownership and recoverability of data files on file away service systems. In case of that some special blocks called sentinels are indiscriminately embedded into the data file F for sensing purpose, and F is further encrypted to protect the places of these special blocks. By contrast, the number of queries a client can pose is also a specified priori, and the introduction of pre computed sentinels forestall the development of realizing dynamic data updates and public auditability is not supported in their method. Shacham and Waters designed an improved Proof of Retrievability (PoR) scheme with full proofs of security in the security model.

They proposed publicly verifiable homomorphic authenticators built from Boneh–Lynn–Shacham (BLS) signatures. Based on signatures the proofs can be aggregated into a small authenticator value and public retrievability is achieved. But still the authors only consider static data files. Later on, Erway et al. proposed constructions for dynamic provable data ownership. They broaden the PDP model in to support demonstrable updates to stored data files using rank-based authenticated skip lists. This method is fundamentally a fully dynamic version of the PDP solution.

To support updates, especially for block insertion, they existing work eliminates the index information in the tag computation. Ateniese et al. PDP model employ authenticated skip list data structure to authenticate the tag information of challenged or updated blocks first before the verification steps. On the other hands, the efficiency of their scheme remains ill-defined.

III. Proposed System

A. Problem Statement

With the rapid development of modern technology, we face many security challenges in the storage of data in online especially in the cloud. The problem is to find efficient protocol which avoids unnecessary overhead to the user in checking the integrity of their own data and also it should support dynamic data operations instead of just achieve and back up the data files in the cloud.

B. Proposed System Framework

In this section, we presented our security protocols for cloud data storage service with the aforementioned research goals in mind. We first started with some basic results aiming to provide integrity assurance of the cloud data and discuss their demerits. Then we presented our protocol which supports public auditability and data dynamics. We also showed how to extent our main method to support batch auditing for TPA upon deputation from multiusers. An overall network architecture for cloud data storage is presented in Figure. 1

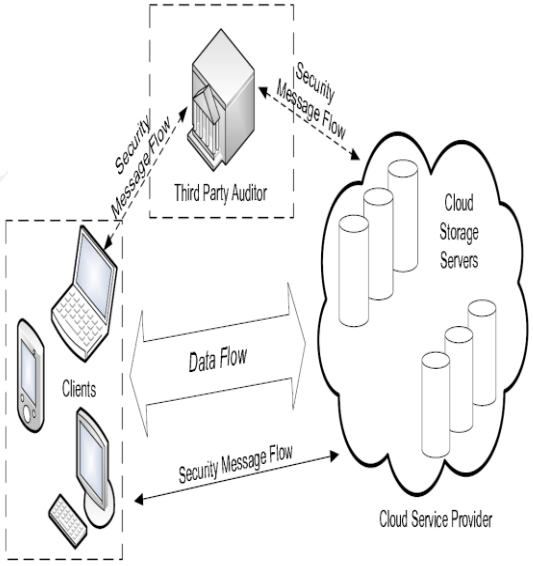

Figure 1: Overall Framework for Cloud Storage

C. Construction of a Network Model

In this module the basic network model for the cloud data storage is developed. Three different network entities can be identified as follows: First element is client, an entity which has large data files to be stored in the cloud and trust on the cloud for data maintenance and computation. It can be either individual consumers or organizations. Secondly, Cloud Storage Server (CSS) is an entity which is managed by Cloud Service Provider (CSP) has important storage space and computation resource to maintain the client data and finally Third Party Auditor (TPA) is an entity, which has potentiality that clients do not have and it is trusted to evaluate and expose risk of cloud storage services on behalf of the clients upon request. In the cloud epitome, by putting the large data files on the hostile servers the clients can be alleviate the burden of storage and calculation. As the clients no longer possess their

data locally, it is of vital important for the clients to guarantee that their data are being correctly stored and maintained. That is clients should be fitted out with certain security means so that they can sporadically verify the rightness of the remote data even without the existence of local copies. In case of those clients do not inevitably have the time, they can depute the monitoring task to a trusted TPA to monitor their data.

In the proposed system, we only consider verification methods with public auditability: any TPA in possession of the public key can act as a verifier. We assume that TPA is indifferent while the server is untrusted. For application intent, the clients may interact with the cloud servers via CSP to get at or retrieve their pre stored data. More significantly, the client may often perform block-level operations on the data files. The general forms of operations that we consider in this work are modification, insertion, and deletion. But we don't address the issue of data privacy since the topic of data privacy in cloud computing is extraneous to the problem we study here.

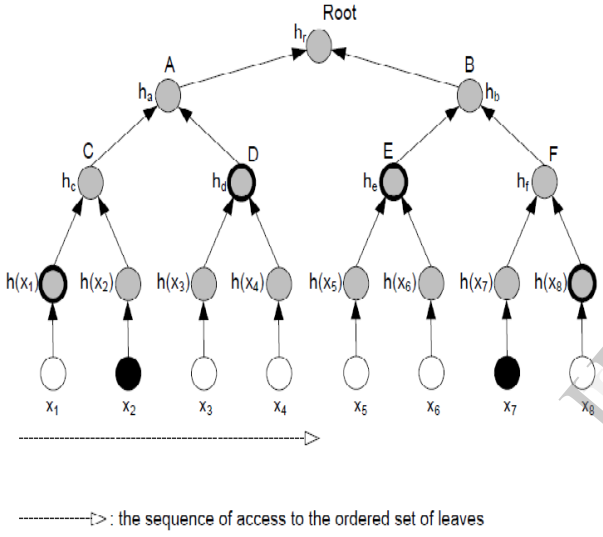

Figure 2: Merkle Hash Tree

D. Implementation of Protocol Based on Merkle Hash Tree

The figure 2 shows the sequence of access carried out in case of the Merkle Hash tree. In this module we presented the basic protocol based on Merkle Hash Tree which supports public auditability and data dynamics. A Merkle Hash Tree (MHT) is a well analyzed authentication structure which is meant to efficiently and securely prove that a set of elements are undamaged and unaltered. It is constructed as a binary tree where the leaves in the MHT are the hashes of authentic data values. The verifier with the authentic user requests and requires the authentication of the received blocks and then the provider provides the verifier with the subsidiary authentication information. After that the verifier can verify and then checking if the calculated user is the same as the authentic one. Merkle Hash Tree is commonly used to authenticate the values of data blocks.

E. Default Integrity Verification

Here in this module we presented the default integrity verification process. The client or third party can verify the integrity of the outsourced data by challenging the server. Before challenging the TPA first uses public key to verify the signature. If the verification fails it rejects by emitting false message otherwise it recover the data.

F. Dynamic Data Operation

In this module we perform the dynamic data operations. Proposed method can explicitly and efficiently handle fully dynamic data operations including data modification (M), data insertion (I), and data deletion (D). Note that in the following descriptions, we consider that the file F and the signature Φ have priory been generated and properly stored at server. Once the authentication process is finished one can select the particular file on the cloud and the user has to enter public key and then private key to perform the deletion and modification operation. We can say this system is more robust since it is very hard to decrypt the blocks by hackers which is scattered based on the size of the file. Each separated data block must be decrypted while the authentic user is requesting the file for accessing in their cloud storage.

G. Batch Auditing for Multi-client Data

Here in this module we present the batch auditing for multi-client data. We broaden our method to allow for provable data updates and verification in a multi-client system. In the BLS based structure, the congeries signature method allows the creation of signatures on arbitrary distinguishable messages. Furthermore, it supports the aggregation of multiple signatures by distinct signers on distinct messages into a single short signature and it greatly reduces the communication cost while giving efficient verification for the authenticity of all messages.

IV. Experimental Results

The cloud setup has been created and users can register with the cloud after getting authentication from the cloud server. The figure 3 shows the login form of the cloud server. The figure 4 shows the various task that server can perform on the cloud space and the figure 5 shows the registration form where new user can enter the details and get approval from the cloud server. The figure 6 shows the user tasks after successfully completing the registration process. The figure 7 shows the form where the user can upload the files he/she wanted to store in the cloud. The figure 8 shows the download and deletion operation form and the figure 9 displaying the file stored in the cloud after downloading completed.

Figure 3: Login Page

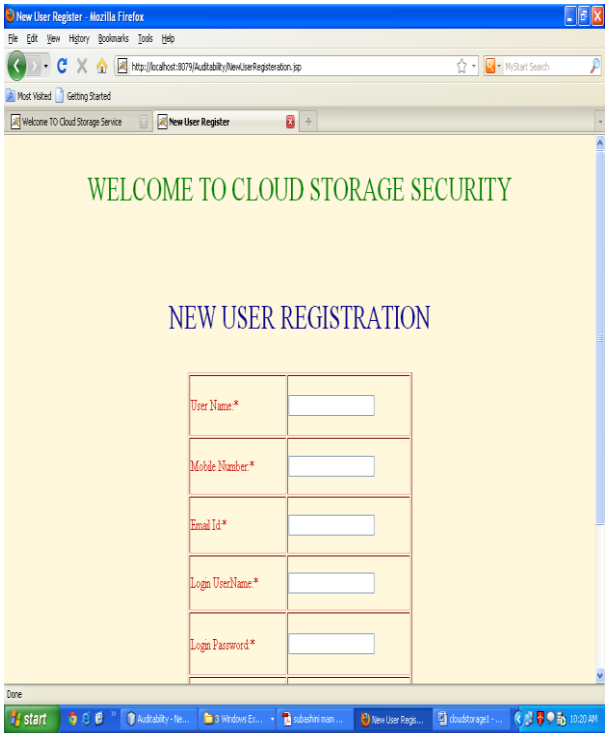

Figure 5: User Registration Form

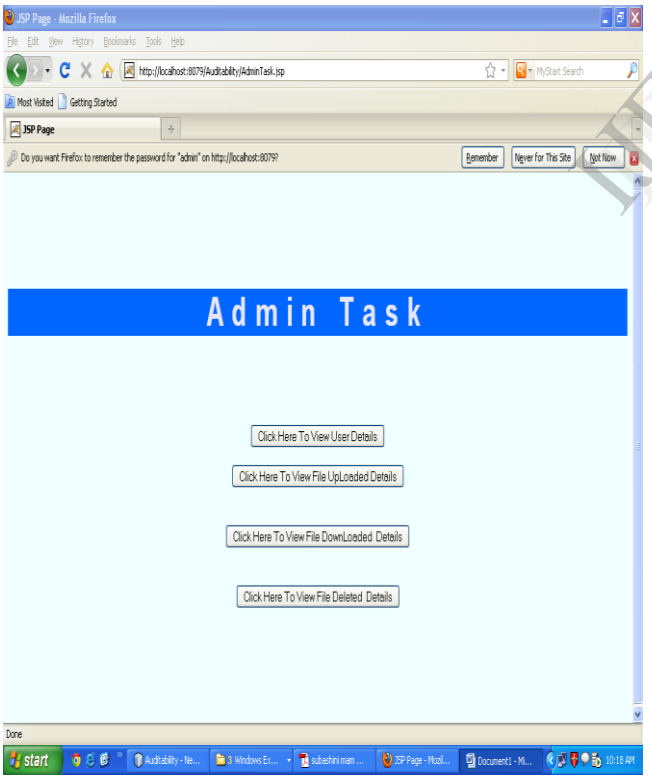

Figure 4: Admin Task

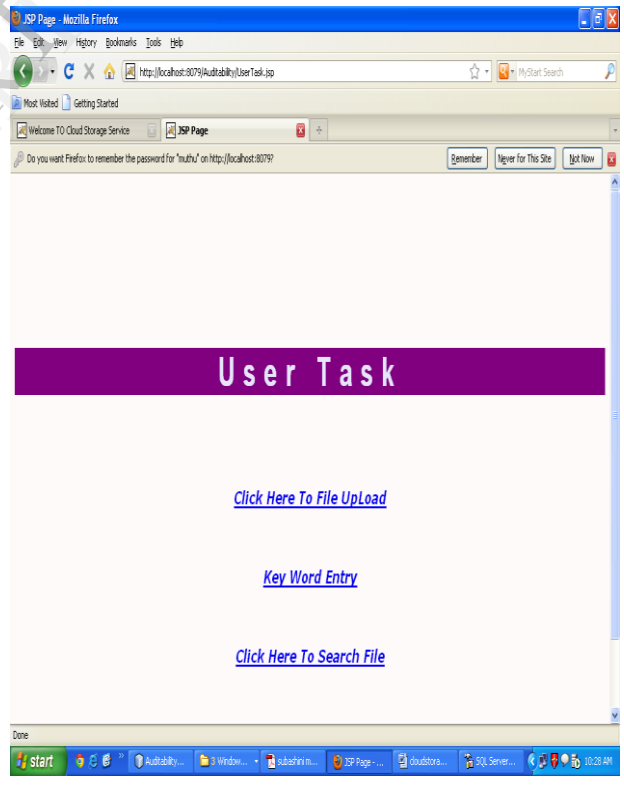

Figure 6: User Task

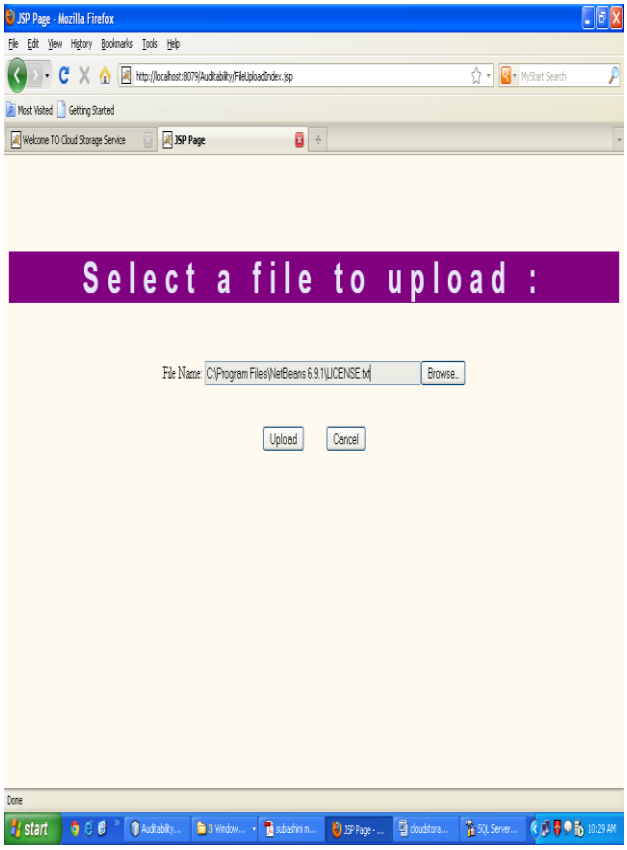

Figure 7: File Uploading

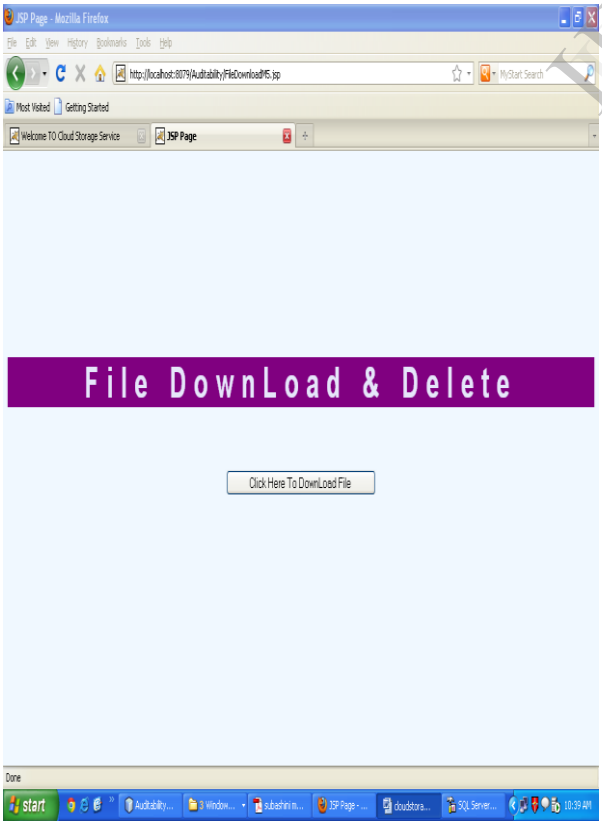

Figure 8: File Downloading and Deletion

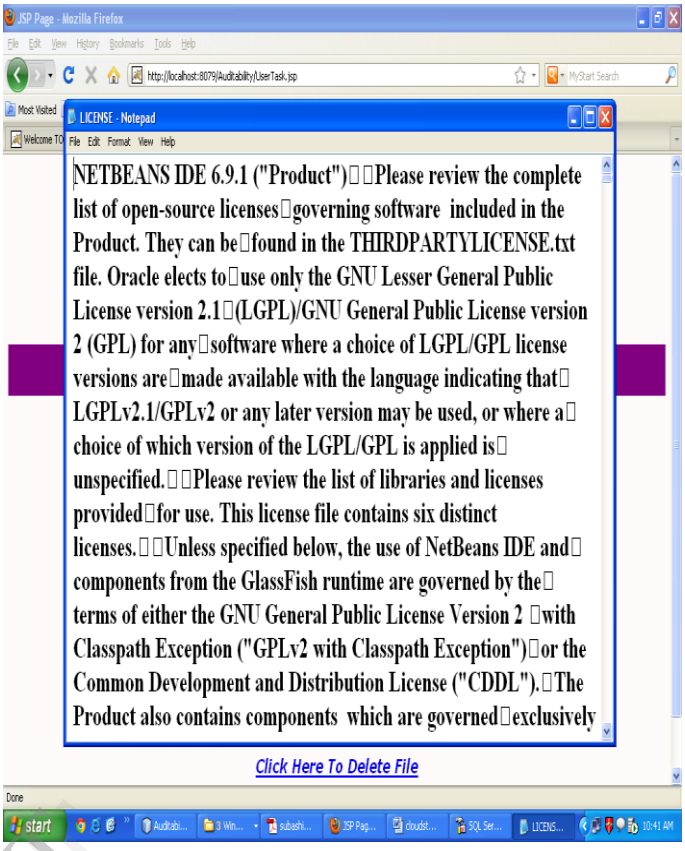

Figure 9: File Display

V. CONCLUSION

To guarantee cloud data storage security, it is decisive to enable a TPA to evaluate the service quality from an objective and independent view. Public auditability also allows clients to depute the integrity verification tasks to TPA while they themselves can be treacherous or not be able to commit necessary computation resources performing continuous auditing process. Another greater concern is how to construct verification protocols that can suit dynamic data files. In this proposed system, we examined the problem of providing instant public auditability and data dynamics for remote data integrity check in cloud computing. Proposed construction is intentionally designed to meet these two important goals while efficiency being kept closely in mind. To attain effective data dynamics we improved the existing system of storage models by manipulating the classic Merkle Hash Tree building for block tag authentication. The support of handling multiple auditing tasks efficiently, we explore the technique of bilinear aggregate signature to broaden our main result into a multiuser setting. Here, TPA can perform multiple auditing tasks simultaneously. Extended security and performance analysis show that the proposed method is highly efficient and provably secure.

REFERENCES

[1] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing", Proc. 14th European Symp. Research in Computer Security (ESORICS '09), pp. 355-370, 2009.

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores", Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 598-609, 2007.

[3] A. Juels and B.S. Kaliski Jr., "Pors: Proofs of Retrievability for Large Files", Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 584-597, 2007.

[4] H. Shacham and B. Waters, "Compact Proofs of Retrievability", Proc. 14th Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT '08), pp. 90-107, 2008.

[5] K.D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation", Report 2008/175, Cryptology ePrint Archive, 2008.

[6] M. Naor and G.N. Rothblum, "The Complexity of Online Memory Checking", Proc. 46th Ann. IEEE Symp. Foundations of Computer Science (FOCS '05), pp. 573-584, 2005.