Asian Journal of Engineering and Applied Technology
ISSN: 2249-068X (P) Vol. 4 No. 2, 2015, pp.27-33
© The Research Publication, www.trp.org.in
DOI: https://doi.org/10.51983/ajeat-2015.4.2.2877

Managing Cloud Security Using Energetic Information Flow Control

S. Babu Renga Rajan[1], D. Pushpa Ranjini[2], B.Shanmuga Sundari[3], A. Haseena Beevi[4] and R. Meenakshiammal[5]
[1]HOD & Professor, [2]Professor, Dept. of CSE, [3&4]Assistant Professor, Dept. of IT,
PET Engineering College, Vallioor, Tamil Nadu, India
[5]Associate Professor, Dept. of CSE, Rohini College of Engineering & Technology,
Kanyakumari, Tamil Nadu, India
E-mail: it.shanmugasundari@petengg.ac.in

(Received 22 August 2015; Revised 11 September 2015; Accepted 28 September 2015; Available online 6 October 2015)

***Abstract* - Most of the cloud computing solutions used today does not ensure security. Although methods like access control list, firewall and cryptography impose limits on information that is released by the system, they provide no guarantee about information propagation. We propose a Decentralized Information Flow Control (DIFC) that is integrated into the PaaS cloud model. This enhances security by associating labels with the data it protects. This data-centric security mechanism tracks and limits data propagation.**

***Keywords:* Decentralised information flow control (DIFC), multi-tenancy, mandatory access control, security, data isolation.**

I. INTRODUCTION

Cloud computing relies on sharing computing resources rather than having local servers or personal devices to handle applications. Here the applications are provided and managed by the cloud server and data is stored remotely in the cloud configuration. Organizations use cloud in a variety of different service models like SaaS, PaaS, IaaS and deployment models like private, public, hybrid. There are a number of security issues associated with cloud computing. The security issues are faced both by the cloud providers and by the customers. The clients must trust the cloud providers. The use of virtualization in implementing cloud infrastructure brings unique security concerns for customers. The virtualization layer must be properly configured.

Many models have been evolved for securing the cloud such as access control list, Chinese wall [2], homomorphic encryption. Access control lists[1] prevent unauthorized file access but do not control how the data is used afterwards. Similarly, homomorphic encryption[3] provides a means to exchange information privately across a non-secure channel, but no guarantee about the confidentiality of the data. Using these methods clients are unable to trust the cloud.

Thus, in order to enhance security the data must be protected at the time of creation itself.

Protecting data in the cloud, providing authenticity, controlling unauthorized access, maintaining integrity are some of the data protection techniques. Cryptography is one of the efficient method for data security in cloud computing. Here data is encrypted into cyphertext using secret key and then decrypted using the same secret key inorder to get back the plain text. This includes the design and implementation of an efficient encryption and decryption algorithms.

In cloud computing, data owners outsource their sensitive data. These data moves across various platform and hence it must be protected. Hence a secure cloud infrastructure [5] is essential. In order to ensure security, encryption and decryption algorithms are used. These method provide security to some extent but does not guarantee security during transmission. The data owners are unaware of the changes that are being made in their original data. Hence it is essential to provide a mechanism which ensures security in cloud and also intimates the data owners about the changes.

Data centric security mechanism such as IFC increases security by tracking information flow. DIFC improves security by giving developers the ability to coordinate with the cloud provider. It also controls the propagation of user data. It also improves multi-tenancy that shares services between cloud tenants. The cloud platform can impose checks to enforce security policies despite flaws in the services themselves. It also tracks data flow across different services and to improve accountability, it offers the cloud provider a way to log sensitive operations on tenant data.

II.RELATED WORK

Various methods have been proposed to ensure security in cloud. There must be some mechanism which ensures security so that both client and owners may trust the cloud provider. Many methods have been developed in the recent years, which has resulted in various approaches to the design of secured information flow.

Biba (1977) formulated a formal state transition system [2] of computer security policy that describes a set of access control rules designed to ensure data integrity. Data and subjects are grouped into ordered levels of integrity. The model is designed so that subjects may not corrupt objects in a level ranked higher than the subject, or be corrupted by objects from a lower level than the subject. Users can only view content at or above their own integrity level. But if an external threat posed by one system, attempting to change the behaviour of another by supplying false data will improperly invoke the functions of it's own behaviour. Improperly performed modification behaviour can sabotage subsystem function.

J. A. Goguen & J. Meseguer (1982) introduced a simple and general automaton theoritic approach [10] to modelling secure systems. It shows how to use abstract capabilities to model the dynamic security aspects of system. The approach can be applied not only to computer operating systems but also to secure message systems, and to database systems.It uses multi level security, capability passing, multi-user/multi key access, automatic distribution and authorisation chains. But the passing of capabilities among users can lead to situations in which it is difficult to determine whether or not security can be violated.

J. Bacon, et al., in 2010 proposed that security engineering must be integrated with all stages of application specification and development. It's aim is to guarantee that above a small trusted code base- data cannot be leaked by buggy or malicious software components [5]. Cloud-hosted services that have end-to-end information flow control preempts worries about security and privacy violations retarding the evolution of large-scale cloud computing. But the stored data and hosted services all have different owners whose interests are not aligned.

Qian Wang, et al., (2011) enabled public auditability and data dynamics in which a third party auditor (TPA), on behalf of the cloud client verifies the integrity of the dynamic data stored in the cloud. The introduction of TPA eliminates the involvement of the client through the auditing of whether his data stored in the cloud is indeed intact, which can be important in achieving economies of scale for Cloud Computing. The support for data dynamics via the most general forms of data operation, such as block modification, insertion and deletion, is also a significant step towards practicality, since services in Cloud Computing are not limited to archive or backup data only.

Hsiao-Ying (2012) proposed a secure erasure code where a threshold proxy re-encryption scheme is integrated with a decentralized erasure code. This supports secure and robust data storage and retrieval. It also lets a user forward his data in the storage servers to another user without retrieving the data back. This scheme supports encoding operations over encrypted messages as well as forwarding operations over encoded and encrypted messages. The drawback is that the storage server must do more computations. The storage server must also manage the cryptographic keys which became a bottleneck.

Junzuo et al., (2013) proposed an Attribute Based Encryption(ABE) and verifiable data decryption method to provide data security in cloud based system. They have designed the data decryption algorithm based on the user requested attributes of the out sourced encrypted data. One of the main efficiency drawbacks of this method is cloud service provider has more computational and storage overhead for verification of user attributes with the outsourced encrypted data.

A. Information Flow Control

A mechanism for IFC is one that enforces information flow policies. Several methods to enforce information flow policies have been proposed. Run-time mechanisms that tag data with information flow labels have been employed at the operating system level and at the programming language level. Static program analyses have also been developed that ensure information flows within programs are in accordance with policies.

Secure data access can be classified into two models; Mandatory Access Control (MAC) and Discretionary Access Control (DAC). In DAC, the owner of the object specifies which subjects can access the object. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission on to any other subject. Most operating systems such as all Windows, Linux, and Macintosh and most flavors of Unix are based on DAC models. In these operating systems, when a file is created, the owner decides what access privileges can be given to other users; when they access the file, the operating system will make the access control decision based on the access privileges created by the owner. Access Control Lists (ACL's), Role Based Access Control (RBAC) and capability systems comes under DAC.

In mandatory access control (MAC), the system specifies which subjects can access specific data objects. The MAC model is based on security labels. Subjects are given a security clearance (secret, top secret, confidential, etc.), and data objects are given a security classification (secret, top secret, confidential, etc.). The clearance and classification data are stored in the security labels, which are bound to the specific subjects and objects. When the system is making an access control decision, it tries to match the clearance of the subject with the classification of the object. For example, if a user has a security clearance of secret, and he requests a data object with a security classification of top secret, then the user will be denied access because his clearance is lower than the classification of the object. IFC is based on MAC approach.

IFC is a data centric approach developed from military information management methodologies. It achieves protection by grouping security labels with data to track and limit data propagation. Labels are in turn grouped with principles in the system. IFC policy states that data protection policy checking can be based on comparing the labels. In other words, permitted relationships between the labels of data and the labels of principals requesting access to data is provided. It is just like permitting unpriviledged users to pass information to priviledged users but not read privileged information with matching restriction on the priviledged users.

B. System Design

IFC system considers data as events. As illustrated in Fig1,The Event Processing backend which encapsulates data in events, consists of *Event Processing Engine* where the *event dispatcher* dispatches the events to the Event Processing unit (EPU). The *event dispatcher* usually acts as a broker to distribute events to EPUs in order to get processed. The *event processing unit* is responsible for generating events which are processed and filtered according to the functional requirements. It also associates labels with the events which are being generated by EPU and generate result events which are exported to web front end. Thus response is being provided for user’s request. The labels are tracked so that to ensure dataowners that their data is protected.

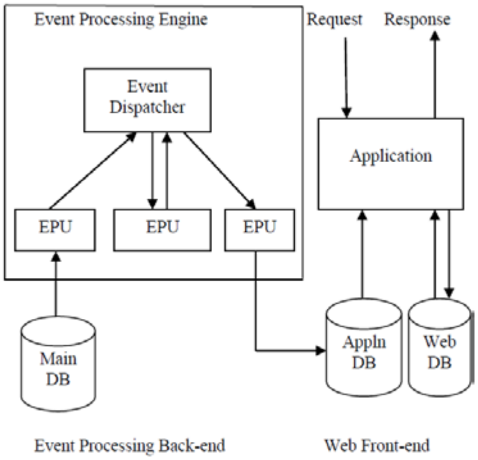


Fig.1 System Architecture

C. Cloud Security Conserns

Cloud provider is usually responsible for providing security in cloud. The extensive use of virtualization in implementing cloud infrastructure brings unique security concerns for customers. There are some current regulatory issues like multi-tenancy, access control and accountability.

1) Multi-Tenancy: Multi-tenancy is an architecture in which a single instance of a software application serves multiple customers or tenants. A tenant is a group of users sharing the same view on the software they use. With a multi- tenant architecture, a software application is designed to provide every tenant a dedicated share of the instance including its data, configuration, user management, tenant individual functionality and non-functional properties. Multitenancy contrasts with multi-instance architectures where separate software instances operate on behalf of different tenants.

The exact degree of multi-tenancy, as it's commonly defined, is based on how much of the core application, or SaaS, layer is designed to be shared across tenants. The highest degree of multi-tenancy allows the database schema to be shared and supports customization of the business logic, workflow and user-interface layers. In other words, all the sub-layers of SaaS offer multi-tenancy in this degree. In the lowest degree, multi-tenancy is limited to the IaaS and PaaS layers, with dedicated SaaS layers for each tenant. And in the middle degree of multi-tenancy are clusters of homogenous tenants that share database schemas and other application layers. In the middle level, each cluster of users has its own version of database schema and the application itself.

2) Access Control: Access control is a selective restriction of access to the resources in cloud. It is a fundamental aspect of information security that is directly tied to the primary characteristics such as integrity, confidentiality and availability. Cloud computing [6] service providers should provide the following functionalities from the perspective of cloud based on the specified policies, Control access to a consumer’s data from other consumers in multi-tenant environments, Control access to both regular user functions and privileged administrative functions, Maintain accurate access control policy and up to date user profile information.

3) Accountability: Accountability is about developing a holistic approach to achieve trust and security in the cloud encompassing legal, regulatory and technical mechanisms. Accountability is a set of approaches to addresses two key problems: Lack of consumer trust in cloud service providers and difficulty faced by cloud service providers with compliance across geographic boundaries. The Emphasis is on data protection, but the notion of accountability encompasses more than just privacy . Accountability ensures transparency, assurance, user trust, responsibility and policy compliance. There is a lack of accountability on the operations performed on tenants’ data in the cloud.

D. Public Auditing

The ring signature is the type of digital signature which is used for privacy preserving. It can be performed by any group member. Therefore, a message signed with a ring signature is endorsed by someone in a particular group of people. The best characteristic of a ring signature is that it should be difficult to identify which of the group members' keys was used to produce the signature. In this, the signature is computed using one of the group member's private key, but the verifier is not able to determine which one. This property can be used to preserve the identity of the signer from a verifier. The ring signatures concept is used to hide the identity of signer on each block so that the private and sensitive information of the group can never be seen by the TPA. To reduce time and long verification, we are extending the ring concept by homomorphic authenticable ring signatures. This homomorphic authenticable ring signature not only maintains the identity but also reduce long verification with supporting to blockless verification.

The name of the ring signature comes from the ring like structure of the signature algorithm because the ring is formed by the private key of the data owner and the public key of the all the users or the group key. The privacy-preserving public auditing using signatures consist of three algorithms as mentioned here: KeyGen, RingSign, and RingVerify. In KeyGen algorithm each user in the group generates their public key and private key. In RingSign algorithm user in the group is related to sign a block with their private key and all group members' public keys. In Ring verify algorithm the verifier is used to check whether the given block is signed by the group member. The ring signatures for public auditing consist of following steps for auditing:

1. Each user generates its public and private key
2. A user in the group signs a block with his private key and all group members' public key. Pk1 is public key of the user; Sk1 is private key of the user; (Pk1 ……Pkd) is ,d number of users of data block m Є ZP
3. User randomly selects data block m, Let id is identifier of data block m
4. User ui encrypts with all user's public key, so only private key of the group user's i Є [1, d] would be able to decrypt it. This ensures privacy of data.
5. To ensure auditing by third-party user (ui), where i Є [1, d] signs the data block using his private key.
6. TPA (Third-party auditor) using a Pk1…Pkd Where d is number of users in the group.

TPA calculates signature of data blocks but unaware of who sign it .Therefore calculates signature using each given public key (Pk1 ……Pkd) from this set. Gsign = signature set for (Pk1 ……Pkd) If Gsign = {sign1, sign2 …signd} matches with original sign then data block is intact. By using this scheme user can also do the data dynamic operation. As there is group of users which share their data to each other, they can do modification on data of CS.

IV. IMPLEMENTATION

A. Static and Runtime IFC Methods

Even though not directly relevant to the runtime enforcement of IFC in cloud, static method can be used for the safety of the cloud by certifying that cloud software components and their interactions are safe before their deployment. Static methods require taint analysis and security- Typed Language. Taint analysis is a key method for performing the illicit use of untrusted data. It prevents potentially damaging data by malicious users. It decides whether the data is safe to run on the system using techniques similar to those of source code data flow analysis. The limitation of taint analysis is the lack of runtime information.

Considering the security-Typed Language, traditionally language type system is augmented with data flow annotations to express confidentiality and integrity data flow policies by compilers. An example for this type of language is Jif which is a hybrid system that explicitly declares data flow requirements as part of the type of each variable. This enables the enforcement of non-interference where data belonging to one security category cannot interfere with another.

Runtime taint tracking [11] is a form of IFC. IFC is a technique for analyzing and enforcing data flow in applications. It can accept either a tainted or an untainted data which might be secret or public, untrusted or trusted. If a particular data is untrusted then it must not be used in the secretive sections of the code. Such sensitive and secretive information should not be leaked out of the application and should flow only through well drained channels. Because of its use to prevent cross-site scripting and SQL injection, it is used in research community. A system generates report if such a flow is detected.

Runtime Label Tracking is a simple form of IFC which manages many different and possibly orthogonal notions of data security in metadata. Fetching code at runtime at remote locations can be done. The program statements that an application runs, as well as their execution order are known for both taint and label tracking.

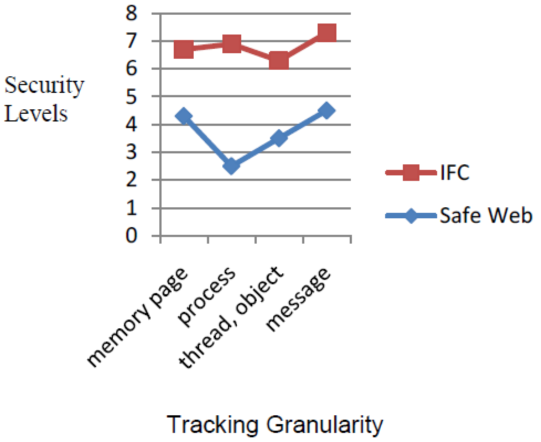


Fig.2 Comparison chart

Fig 2 shows the comparison between IFC and Safe Web. The advantage of IFC is that it provides high level of security for all tracking granularities while considering memory page, process, thread and message. The comparison is done on the basis of security level and tracking granularity.

B. Isolation of Data

Data isolation is essential for effective data flow tracking. It includes the exchange of data by application that are controlled by runtime IFC system. Cloud specific isolation [8] not only isolates tenants' use of resources but support tenants to provide multi-client services. Data isolation is performed by identifying events as compartments so that processing is simplified and to avoid interference with other applications. Hardware-Assisted Isolation by the OS and isolation via Virtualisation are ways in which isolated compartments can be created.

C. Tracking Data Flow

Tracking the flow of data is needed after isolation of data. The cloud based application needs data flow tracking for checking how the data propagates throughout the application. It is offered by cloud hosting providers as a service to their tenants, as well as to the users of the tenants' services. The service providers can easily integrate data flow tracking in their services and mark sensitive user data that needs to be protected. End users can then monitor the propagation of their data directly through the cloud hosting provider, ensure that all sensitive data is treated as expected, and spot any deviations. Service providers can also take advantage of data flow tracking for enabling an additional layer of protection against data leaks, by preventing the propagation of marked data beyond a set of specified network and file system locations, as well as for protecting their own digital assets .

IFC system usually classifies isolated component based on the security properties. Data moves between the isolated components and the information regarding this is maintained accurately by tracking system and this information is provided to other parts and they react accordingly. The main aspect of data flow tracking system is maintaining the granularity at which the tracking operates. IFC systems are designed for a certain granularity of tracking defined by the IFC model, and hence this granularity cannot be altered.

The data flow tracking system normally intercepts the exchange of data, inspects it and then updates the information flow metadata. Thus using data flow tracking, a cloud provider offers added value to both its tenants and their users, potentially leading to a larger customer base.

D. Enforcing Data Flow

The data flow tracking part is followed by the enforcement part of IFC systems that checks IFC policy[10]. If a particular data flow violates IFC policy then action is taken and this checking needs more resources. Inially the policy is specified and attempts are taken to enforce them properly. The policy is enforced at a particular level of granularity. It declassifies data to lower levels of secrecy, or endorse data to higher levels of integrity.

E. Data Sharing

The data sharing includes the privacy preserving mechanism. Here whenever a data is uploaded in a group, it can be shared by other members in the same group. But, when a member from another group wants that data, he sends request to that group and can use that data only when all its members grant access permission. This method also hides the identity of data owner. The main advantage of this method is that it enhances security.

Homomorphic authenticators are basic tools to construct public auditing mechanisms. Besides unforgeability (i.e., only a user with a private key can generate valid signatures), should satisfy the following properties:

1. Block less Verifiability
2. Non-malleabilty

Blockless verifiability allows a verifier to audit the correctness of data stored in the cloud server with a special block, which is a linear combination of all the blocks in data. If the integrity of the combined block is correct, then the verifier believes that the integrity of the entire data is correct.

Non-malleability denotes the members of the group only denote valid signatures which denote the unforgeability of the group members.

F. Experimental Results

In this section, we present the experimental results of IFC. IFC mainly ensures the data owners that their data flow is tracked at every point in its lifetime. As illustrated in Fig 3, when the dataowner uploads his file in the cloud, the client who is in need of the file must request the cryptographic key from the owner. The client can access the file once the owner sends the cryptographic key. Any modification done by the client is intimated to the owner.

Admin process plays a vital role in maintaining proper data flow. It administers the overall flow of the system. It verifies the file details and checks the status. It can see any pending approvals for tracking data flow and summary data regarding the state of virtual machine deployments. Each file is being tracked. All information including file type, status, size, date of creation are maintained. The status of the file includes both verification status and download status. It provides details regarding whether the file has been verified or not and whether it is allowed to get downloaded or blocked. The main part of admin process is to intimate dataowners about the modification in their file by the clients. Fig 4 represents the experimental result which provides label for each file that contains information about the file. The information includes verification status, modification details, file type, size, etc.. Using this, data flow can be tracked and hence security is ensured.

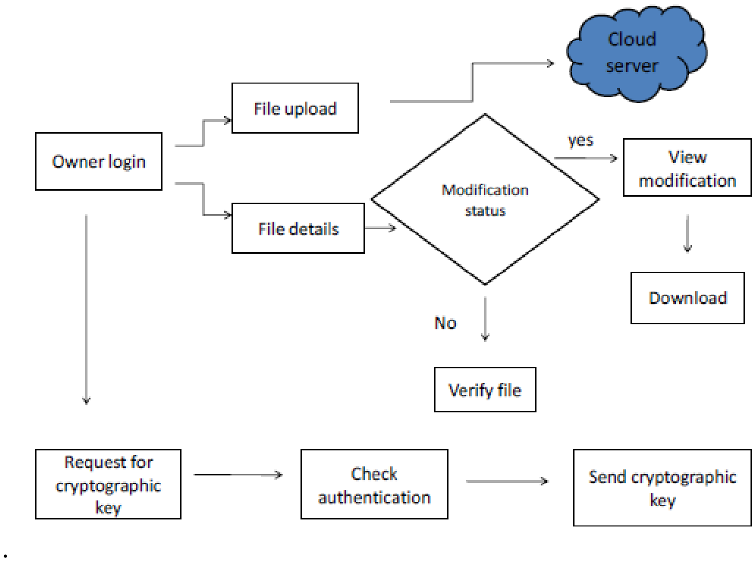
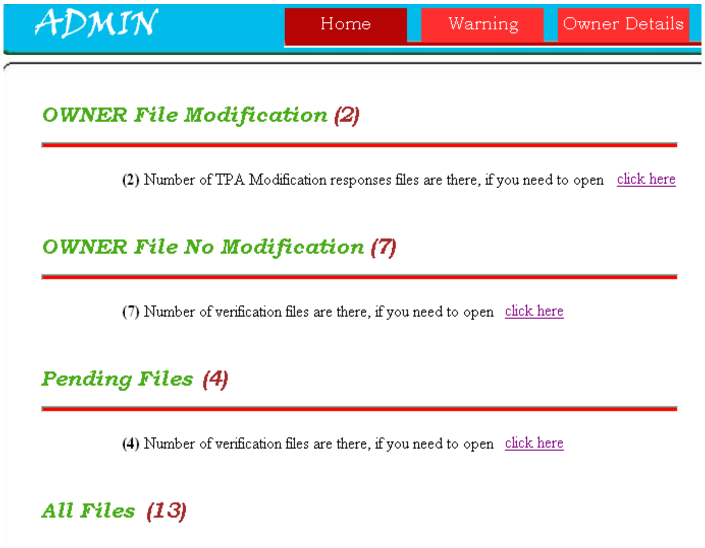


Fig 3 Data Flow Model in Owner Process



OWNER File Modification (2)

(2) Number of TPA Modification responses files are there, if you need to open click here

OWNER File No Modification (7)

(7) Number of verification files are there, if you need to open click here

Pending Files (4)

(4) Number of verification files are there, if you need to open click here

All Files (13)

Fig. 4 File Details

V. CONCLUSIONS AND FUTURE ENHANCEMENT

In this paper, we proposed a Decentralised Information Flow Control mechanism in which security is associated with the data it protects. IFC is data-centric, and achieves protection by associating security labels with data, in order to track and limit data propagation. This data-centric security mechanisms, which track or enforce information Furtherly, the efficiency and effectiveness of our design will be improved by a real implementation with a small-scale cluster environment that is highly desirable to enhance the security of the storage data in cloud. Besides, DIFC should not impose an unacceptable performance overhead and it is important that application developers using cloud-provider IFC are aware of the trust assumptions inherent in the IFC provision.

flow, improves cloud security. Developers are given the ability to coordinate with the cloud provider and control how user data propagates in a cloud platform. This facilitates compliance with regulatory frameworks. Tracking data flows across different services offers the cloud provider a way to log sensitive operations on tenant data rigorously, thus improving accountability. It overcomes the threat that curtails security in a centralized environment.

REFERENCES

[1] D. Denning, *Cryptography and Data Security*. Addison-Wesley Longman, 1982.

[2] Biba, “Integrity considerations for secure computer systems,” MITRE Co., technical report ESD-TR 76-372, 1977.

[3] R. Wu, G.-J. Ahn, H. Hu, and M. Singhal, “Information flow control in cloud computing,” in *CollaborateCom*, 2010.

[4] H. Hacig¨um¨us¸, B. Iyer, *et al.*, “Executing SQL over encrypted data in the database-service-provider model,” in *Proc. 2002 ACM SIGMOD*, pp. 216–227.

[5] J. Bacon, D. Evans, *et al.*, “Big ideas paper: enforcing end-to-end application security in the cloud,” in *2010 ACM/IFIP Middleware*.

[6] P. Mell and T. Grance, “The NIST definition of cloud computing,”2011.

[7] I. Foster and C. Kesselman, *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 2003.

[8] P. Barham, B. Dragovic, *et al.*, “Xen and the art of virtualization,” in *2003 ACM SOSP*.

[9] T. Ristenpart, E. Tromer, *et al.*, “Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds,” in *Proc.2009 ACM CCS*, pp. 199–212.

[10] J. A. Goguen and J. Meseguer, “Security policies and security models,” in *Proc. 1982 IEEE SOSP*, pp. 11–20.

[11] E. Chin and D. Wagner, “Efficient character-level taint tracking for Java,” in *Proc. 2009 ACM SWS*, pp. 3–12.