Enhancing Cloud Security using Decentralized Information Flow Control

Priyanka S. Mane
ME Student, Dept. of Computer Engineering
Padmabhooshan Vasantdada Patil Institute of Technology. Savitribai Phule Pune University
Maharashtra,India.

Yogesh B. Gurav
Professor, Dept. of Computer Engineering
Padmabhooshan Vasantdada Patil Institute of Technology. Savitribai Phule Pune University,
Maharashtra, India.

ABSTRACT
There is major demand to introduce cloud computing in many organizations today. The reason is cloud's sharing infrastructure, multi-tenancy and huge storage facilities ensures increase in computing efficiency, flexibility, generality and cost effectiveness. But with this, organizations want that the computing platform should be secured and should satisfy all the important rules and regulations. So security is the key point for the success of cloud computing. It is examined that cloud computing is less satisfactory in providing security due to its heterogeneity. In this paper a solution named - Decentralized Information Flow Control (DIFC) is defined to solve the problem of security specifically of Software as a Service (SaaS) level. DIFC is a Mandatory Access Control method which is able to provide better security and integrity than is provided by other approaches available today. DIFC enforce general policies by using proper labeling and checking methods. DIFC gives a way to control and monitor the flow of data continuously according to the policy. Hence we believe that DIFC is a powerful tool to enhance SaaS cloud security and to help cloud providers to satisfy rules and regulations and audit this compliance with easy in future.

General Terms
Cloud computing, SaaS, Security, Information Flow Control, Access control.

Keywords
Decentralized information flow control, information flow control, access control, secure cloud computing, data security, labelling.

1. INTRODUCTION
Cloud computing is a proven technology to meet the current needs of Information Technology field. Because of its fast, easy and on demand access to computing resources organizations are moving their data over cloud. But with this, organizations want that the computing platform should be secured and should satisfy all the important rules and regulations[8]. So security is the key point for the success of cloud computing. Security is the challenging task in cloud computing[4]. It stems from the fact that cloud infrastructure is combination of mixed tools and applications which are designed and developed by multiple teams with no integrated approach for assuring data security[3]. For example, some providers may use virtualization[6] concept to isolate the data. Similarly, a data store may provide some other facilities for data isolation. Traditional security methods such as cryptography[10] and Chinese wall[13] are used in cloud computing but does not meet security and are unable to provide efficiency, generality and flexibility required by cloud providers and tenants.To give better security, a solution, a data centric security method known a Decentralized Information Flow control(DIFC)[1] in particular for Software as a Service (SaaS) cloud level is proposed. DIFC ensures high data security and data integrity. DIFC is a type of Mandatory Access Control (MAC)[1] model in which security policy (i.e. labels) is defined at all levels in the system, usually specified by the administrators. DIFC is a MAC model which is originally developed from military information management methods. Such data centric security method gives security in many ways by controlling and tracking information flow. First, the data is stored in secret form to protect from leakage of confidential or sensitive information.Second, controlling the flow of information using access control by imposing policies or rules in the form of labels on the data which are usually specified by the administrator. Third, providing multi-tenancy with data integrity by sharing of resources and services, which is achieved by imposing checks to enforce policies. Fourth, accountability by tracking the flow of information across all services over the cloud which provide a way to log sensitive operations.

In this paper we proposed a Decentralized Information Flow Control model to enhance cloud security particularly for Software as a Service (SaaS) level. We describe the proposed DIFC system with its architecture and implementation. Performance of the DIFC system results in better security. Our contribution is despite of number of challenging issues in cloud environment our DIFC system leads in more secured and practical cloud computing. Thus DIFC is the most appropriate and most suitable model for enhancing cloud security.

2. LITERATURE SURVEY
This survey describes previous methods of information flow control mechanisms, let see in detail. In this section we are going to give brief overview of previous IFC based techniques.

FlowK [12]

Paper describes how, FlowK can be integrated with cloud software. We have designed and evaluated a framework for deploying IFC-aware web applications, suitable for use in a PaaS cloud. Our design based on "policy-mechanism separation", in that the enforcement of IFC in FlowK separated from any knowledge of principals, users and the management of privileges. This separation ensures maximum flexibility for higher levels of software; this work contributes: (1) without modifying monitored access of standard OS it includes IFC within it like other systems. (2) To achieve requirements for isolated processing it supports conflicts of interests in IFC model. (3). Idea of FlowK is based on decentralized IFC model (DIFC) introduced in 1997. Decentralized model has been outlined for varying

needs like static, global, hierarchical levels of security to fluid systems and capable of satisfy this needs of different applications. In this, model every entity having two labels: integrity label and secrecy to catch confidentiality/privacy and reliability of source data, these labels have security tags. By above description we can conclude that FlowK does not require any changes to system calls so unmonitored processes not affected by the existence of FlowK as an OS module, apart from a small performance overhead. The security context manipulation done through a small, well defined set of API calls. The FlowK kernel module is concerned only with enforcement of IFC, following policy-mechanism separation. Regarding this concept, In future, we will explore application policies in more detail and their enforcement via IFC.

Integrating Messaging Middleware and Information flow control [13]

To secure data flows within virtual machine author proposed kernel level protection by applying IFC enabled middleware. Paper describes IFC enables messaging middleware. This approach decides IFC constraints on each data flow within virtual machine. This messaging middleware approach apply IFC across systems which secures data travelling between services (storage) and applications, which is local to virtual machine. To achieve this author introduced concept SBUS middleware enabled by IFC (i.e. SBUS-IFC).SBUS-IFC messaging middleware strongly supports range of communication paradigm- broadcasts, stream based and request-reply; typed messages, security like encryption and access control. Dynamic reconfiguration supported by SBUS mainly, where it provides facility for third parties to manage application's communication as application itself, which add simplification at deployment and development to application. Author mainly emphasis on two things: (1) Describes efficiency of integrated IFC enabled middleware carry in cloud context. (2).Related performance overhead measured.We can draw conclusion that in this paper, IFC enabled middleware shown practically. This work describes IFC control in services, containers, virtual machines, providers, for users, across applications. Proposed mechanism able to separate services and applications from their code, which helps to preserves security in cloud. Author successfully integrated IFC middleware with local mechanism (FlowK) provides protection across applications. In this paper aim was to maintain end-to-end information flow control but assignment of global unique names to tags of application not considered here which will include in future frameworks. Here IFC considers channels and bytes so there is need to extend this work for fine-grained IFC policy. In future, we can apply IFC mechanism to 'Internet of Things'.

Information Flow Control for Cloud Computing [14].

In this paper, we propose an approach to enforce the information flow policies at Infrastructure-as-a-Service (IaaS) layer in a cloud-computing environment. Especially, we adopt Chinese wall policies to address the problems of insecure information flow. We implement a proof-of-concept prototype system based on Eucalyptus open source packages to show the feasibility of our approach. This system facilitates the cloud management modules to resolve the conflict-of-interest issues for service providers in clouds. Several key challenges need to address like Selection of Appropriate Service Layer and Definitions for Policy Components. We can conclude that author first identified the information flow problem, which could raise conflict-of-interest issues in cloud computing environments. In addition, we have articulated challenges in specifying and enforcing information control policies in cloud computing. To address the identified problem and challenges, we proposed an approach to enforce the Chinese wall security policy at the laaS layer of a cloud. We also implemented a prototype system based on Eucalyptus open-source software to prove the feasibility of our approach. In future for instance, we would investigate how laaS management can be complied with both PaaS and SaaS. In addition, a user may wish to delegate his cloud instance access privileges to others. A practical delegation mechanism is another essential component for cloud computing.

Silver Lining: Enforcing Secure Information Flow at the Cloud Edge [15].

In this paper author, proposed policies for Java computations on commodity, data processing, platform –as-a- service cloud by Aspect-Oriented programming (AOP) and In-Lined Reference Monitoring (IRMS).This method provide in-lines secure information flow tracking code into un-trusted Java job binaries in cloud. This facilitates efficient enforcement of large and mandatory access policies without any customize cloud. Silver Lining makes no changes to the cloud infrastructure, which is fully transparent to Java job author with no changes to Java bytecode or API. Result shows the efficiency and scalability of silverLine with low overhead. This technique adds mandatory access control policies as well as secured information flow policies for Hadoop clouds on non-trustworthy java job binaries [16] but execution is completely distinct to rest cloud. Information flow graph (IFG) maintains distributed data resources within loud and tracks information flow. After detailed observation, we can conclude that, SliverLine is first development based on Hadoop cloud information flow, which does not require any changes in cloud infrastructure. In addition, this is solely apparent for java users, which does not require any change in java byte code or API. The work proposed in this paper based only on mandatory access control policies of information flow between resources. SliverLine Efficiency and scalability achieved through low overhead. Verification algorithms used here are smaller than the code-rewriting environment so more trustworthy. In future workflow computations would expressed in other languages such as native code, which is wholesome platform for such extension. There is scope to extend security policy languages and classes on larger and expensive manner. We can investigate feasibility of verification algorithms to endorse IRMs in cloud.

Information flow control for strong protection with flexible sharing in PaaS [18]

This is data-centric method. Paper depicts how information flow control, mechanism is suitable approach for data centric environment. IFC based cloud platforms are able to provide fine-grained control while data sharing. Idea behind IFC is to control information leakage while data exchange. In this paper author proposed, model which tender with common IFC assurance. IFC have security tags named as token, which mentions security concerns, each entity like messages, application instances, sockets and any data exchange platform having this tags. Each entity assigned two labels: secrecy label and integrity label. Linux kernel module used for execution for an IFC based web service framework. IFC based messaging middleware integrated with FlowK.

In this paper author contributed isolation approach to manage data sharing. This technique introduces mechanisms as structured messages having individual label for attributes for middleware integration. Here we aimed to maintain maximum transparency for IFC. In future IFC mechanism can implemented in various levels in cloud stack, like the one-structured objects of policy operate at higher level than operating system. IFC can be enforcing to execute at end-to-end, and possible to apply global naming scheme to tags.

2.1 Merits And Demerits Of Different Approaches Of Ifc

After reviewing different techniques of Information Flow Control the pros and cons are tabulated.

IFC models	Mechanism	Merits	Demerits
FlowK	It enforces information flow control with "policy-mechanism separation".	• It gives maximum flexibility. • It supports conflict-of-interests in IFC. • Small performance overhead.	• Low performance. • Application policies are not taken into consideration.
Integrating Messaging Middleware and Information flow control	It enforces IFC by messaging middleware approach. This approach decides IFC constraints on each data flow within virtual machine.	• It provides high security and supports dynamic reconfiguration. • Good efficiency and end – to-end information flow control.	• Does not support global unique names to tags of application. • It does not provide fine-grained IFC policy.
Information Flow Control for Cloud Computing	It adopts Chinese wall security policy for information flow control at IaaS layer in cloud.	• It ensures feasibility. • It facilitates the cloud management modules to resolve the conflict of interest issues for service providers in cloud.	• It consider IaaS cloud service layer and not complied with PaaS and SaaS cloud. • It lacks in proving practically.
Silver Lining: Enforcing Secure Information Flow at the Cloud Edge	It defines policies for JAVA computations on commodity, data processing, PaaS service cloud by Aspect-Oriented programming and In-Lined ReferenceMonitoring.	• It facilitates efficient enforcement of large and mandatory access policies. • It gives good efficiency and scalability. • It makes no changes the cloud infrastructure. • Low performance overhead.	• Low performance. • It is solely apparent for JAVA users.
Information flow control for strong protection with flexible sharing in PaaS	It uses data-centric method. It depicts how information flow control mechanism is suitable approach for data centric environment.	• It gives maximum transparency. • It provides fine-grained control while data sharing.	• It lacks in integrity. • Does not enforce end-to end IFC.

In previous access control and security systems there is a lack of tracking i.e., accountability on what operations are performed on data in the cloud. A tenant should have the right to know if his data has been misused, mishandled by the provider, or transmitted to third parties without its consent. But unfortunately this is not considered in the previous system. The cloud interface may have a less subtle and comprehensive view of access control than is required for the application.

3. PROPOSED SYSTEM

A. Problem statement

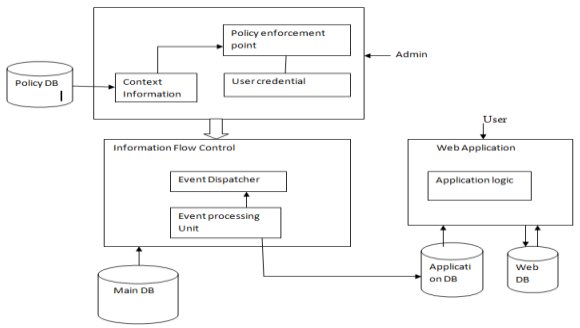
To address cloud security and to meet the requirements of increasing the cloud computing security specifically at Software-as-a-Service level, we propose a data-centric security method named "Decentralized Information Flow Control (DIFC)".

B. Objectives of the proposed system

1. A simplified decentralized information flow control for secure cloud specially for SaaS cloud service layer.
2. Increase data security and maintain integrity of sensitive data.
3. Data isolation with multi-tenancy.
4. Information tracking and accountability.

Structure of the proposed DIFC system

Block Diagram Of The DIFC System



Decentralized Information flow control is type of a Mandatory Access Control system in which the security policies are defined for the overall system, usually done by the administrators. In DIFC policies are enforced by using proper labeling and checking methods. It gives protection to the data by assigning security policies (labels) with the data, in order to control and observe the flow of the data. The labels are also associated with the principals i.e. the users of the system. DIFC security policy allows relations which are true or satisfy, between the labels of data and the labels of principals requesting to access the data. That is, data protection security policy checking schemes are based on comparing the label(s) assigned to the data and with the labels assigned to the principals. The labels are used to provide both data confidentiality and integrity with "secrecy" and "quality" of data.

Functions of the system

1. Data Encryption – Data is stored in secret form to protect from leakage of confidential or sensitive information.
2. Access Control Management - Controlling the flow of information using access control by imposing policies or rules in the form of labels on the data which are usually specified by the administrator.
3. Multi-tenancy - providing data integrity by sharing of resources and services, this is achieved by imposing checks to enforce policies.
4. Data flow Tracking – provides accountability by tracking the flow of information across all services over the cloud which provide a way to log sensitive operations.

Project Contribution

1. In previous methods only encryption and decryption key is saved in database for access. But for more security as it can be easily available separate access key is generated for every request or operation on data.

2. Data centric security mechanism is used by providing both security policies at privilege level and using access key.

C. Algorithm's used

1. A Hash-Key message authentication code (HMAC) algorithm is used for authentication.

2. Data is stored in encrypted form on the cloud by using Cryptographic algorithms.

3. Policy combination algorithms are used to combine all the policies(labels) imposed on

4. data and users and enforce them effectively at runtime.

Policy Combination algorithm for PermitOverrides.

The following specification defines the "Permit Overrides" policy combining algorithm of a policy set.

1. In the entire set of policies to be evaluated, if any policy evaluates to Permit, then the result of the policy combination shall be Permit. In other words, Permit takes precedence, regardless of the result of evaluating any of the other policy in the combination.

2. If all policies are found not to be applicable to the request, the policy combination returns NotApplicable.

3. If there is any error evaluating the target of a policy, a reference to a policy is considered invalid, or the policy evaluation results in Indeterminate, then the result of the combination shall be Indeterminate only if no other policies evaluate to Permit or Deny.

Policy Combination algorithm for DenyOverrides.

The following specification defines the "Deny Overrides" policy combining algorithm of a policy set.

1. In the entire set of policies to be evaluated, if any policy evaluates to Deny, then the result of the policy combination shall be Deny. In other words, Deny takes precedence, regardless of the result of evaluating any of the other policy in the combination.

2. If all policies are found not to be applicable to the request, the policy combination returns NotApplicable.

3. If there is any error evaluating the target of a policy, or a reference to a policy is considered invalid, or the policy evaluation results in Indeterminate, then the result of the combination shall be Deny.

D. Mathematical Model

Definition 1: [Cloud Instance] A cloud instance is a virtual machine running on the cloud infrastructure. It stores customers' data and hosts various kinds of cloud services. Let I denote the set of cloud instances, I = {i1,...,in}.

Definition 2: [Security Group] A security group is a named domain containing several cloud instances on an as need basis .The instances in the same security group are usually dedicated to serving for the same company. Let G denote the set of security groups, G = {gl,.., gn}.

Definition 3: [Conflict-of-Interest (COl) Class] A Col class contains several security groups. Security groups belonging to the same COl class provide services for competing companies. Let C denote the set of COl classes, C = {CI' ... ,cn}.

Based on the above definitions, we give the definition of objects as follows:

Definition 4: [Objects] An object of the DIFC security policy in the SaaS cloud computing environment is a cloud instance. Let 0 denote the set of objects, 0 = {Objl',... , objn} and an object obji.

We further derive two properties associated with objects:

1) Any two objects which belong to the same security group belong to the same conflict of interest class.

2) Any two objects which belong to different conflict of interest classes belong to different security groups.

We formally define the above two properties as follows:

Definition 5: [Object Properties]

• OG i.e O× G is a many-to-one cloud instance object to security group assignment relation. (obj. g) € OG means an object obj belongs to a security group g;

• GC C G × C is a many-to-one security group-to-Col class assignment relation. (g. c) E € GC means the security group g belongs to the COl class c;

• 0 → G is a function that maps a cloud instance object to a security group. SG(Obji) = {g € G I (Obji' gi) €OG}; and

• 0 --→C is a function that maps a cloud instance object

to a COl class. COI(obji) = {c E C I (Obji. gi) E OG^(9i, c) E GC}. Therefore, object properties are defined as:

1) SG(Obj1) = SG(Obj2) =} COI(objI) = COl(obj2)

2) COI(objI) i- COI(obj2) =} SG(objI) ≠ SG(Obj2)

[Subjects] A subject of the security policy in the cloud computing environment is a user

who accesses to the data or services hosted in the cloud instance. Let S denote the set of subjects. S = {81 .... 8n}.

[Access Operations] An access operation includes reading and writing data and using services hosted in the cloud instance by a subject. Let

• ACC i.e. S× 0 be a many-to-many subject-to-object access relation. A subject-to-object access relation can be represented by (sub, obj) €ACC, which means the subject sub has accessed the object obj,

• ACC → Boolean be a function that maps a subject-to object access relation to a boolean value, where

- Access(sub, obj) = {true I (sub,obj) € ACC},

- Access(sub, obj) = {false I (sub, obj) €ACC}.

[Policy Specification]

Let OA is a function mapping each subject to a set of objects, OA(subi) = {obj € OAccess(subi, obj) = true}. Then a subject sub € S can access an object obj € O if and only if any of the following requirements holds:

1) There is an object obj' € 0 such that Access(sub, obj')= true and SG(obj') = SG(obj);

2) For all objects obj', obj' € OA(sub) => COI(obj') ≠COI(obj);

3) obj = objo.

where, initially OA(sub) ≠ø, and the initial access request is assumed to be granted.

4. RESULTS ANALYSIS

A. Experimental results:

1. User credential (Access policy) and Database policy – ensuring efficient information flow control and security.
2. Information tracking i.e. logs all the activities in detail which are carried on the cloud data by the users.
3. User information for authentication and trust.
4. Data is stored in encrypted form on the cloud enforcing data confidentiality.

B. Performance Overhead of DIFC system

To determine DIFC overheads, we compared the DIFC implementation with non-IFC implementation. Our concern is the relative performance of the system. Using a workload of 300 requests, sent in immediate succession, we measured:

1. The DIFC-write measurement represents storage of data in secret form. This prevents unauthorized data leakage i.e. protection of sensitive data.
2. The DIFC-read measurement represents authenticated and authorized access to data by providing security policies(lables) both at data and principals.
3. Non-IFC represents common implementation. As there is no IFC enforcement, there is leakage of sensitive data i.e. on read and write.

Figure 2.1 shows the overhead of DIFC enforcement. The results shows that DIFC enforcement gives 13% overhead in performance time for the workload over Non-IFC, which is normal to handle and does not affect much on performance of the system. The DIFC scenario prevented leakage of sensitive data resulting in better security.

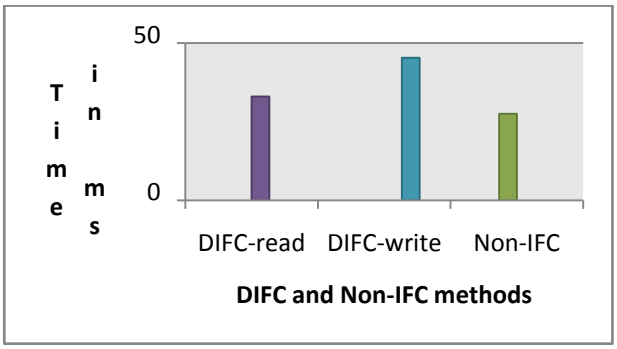


Figure 2.1 Performance evaluation between DIFC system and Non-IFC system for entire 300 request message workload (x-axis time in ms).

In order to validate the DIFC model, we compare it with other IFC models. The comparison is present in table 2.1. It also compared with a number of the information flow control models for cloud computing. The comparison is based on security parameters that either the DIFC or other models can offer. Before proposing the model, we have reviewed almost every proposed information flow control system for secure cloud. Most of them have not been validated or applied in a real cloud computing environment. Hence from the above comparison we can conclude that DIFC is the most appropriate and most suitable model for enhancing secure SaaS cloud computing.

Table 2.1 DIFC against other IFC models.

No.	Comparison parameters	Flow K	SilverLine	Message Middleware-IFC	Flow R	DIFC
1.	Privilege principle	N	N	N	N	Y
2.	Accountability	N	Y	Y	Y	Y
3.	Policy management	Y	Y	N	Y	Y
4.	Integrated with authentication	Y	Y	Y	N	Y

	functions					
5.	Dealing with heterogeneity	Y	Y	Y	Y	Y
6.	Scalability	Y	N	Y	Y	Y
Y = Yes, N = No and N/A = Not applicable.						

5. CONCLUSION

DIFC is able to provide ability for the developers to coordinate with the cloud provider and to control how user's sensitive data propagates on the cloud platform. DIFC is most suitable data centric mechanism for enhancing cloud security.

6. FUTURE SCOPE

In future for this system we will consider that, DIFC should not impose an unacceptable performance overhead and it is important that application developers using cloud-provided IFC are aware of the trust assumptions inherent in the IFC provision.

7. REFERENCES

[1] Jean Bacon, David Eyers, Thomas F. J.-M. Pasquier, Jatinder Singh, Ioannis Papagiannis, and Peter Pietzuch, Information Flow Control for Secure Cloud Computing, IEEE Transactions On Network And Service Management, Vol. 11, No. 1, March 2014.

[2] David Schultz, Barbara Liskov, IFDB: Decentralized Information Flow Control for Databases, ACM , Eurosys'13 April 15–17, 2013.

[3] l. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration. In Open Grid Service infrastructure WG, Global Grid Forum, volume 22, pages 1-5. Edinburgh, 2002.

[4] l. Foster, Y. Zhao, l. Raicu, and S. Lu. Cloud computing and grid computing 360-degree compared. ArXiv e-prints, 901:131, 2008.

[5] T. Ert. Service-oriented architecture: concepts, technology, and design. Prentice Hall PTR Upper Saddle River, NJ, USA, 2005.

[6] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In Proceedings of the nineteenth ACM sympOSium on Operating systems principles, page 177. ACM, 2003.

[7] M. Vouk. Cloud computing Issues, research and implementations. In 30th International Conference on Information Technology Intel!aces, 2008. ITI 2008, pages 31-40, 2008.

[8] C. J. Millard, Ed., Cloud Computing Law. OUP, 2013.

[9] T. F. J.-M. Pasquier, J. Bacon, and D. Eyers, "FlowK: Information Flow Control for the Cloud," in 6th International Conference on Cloud Computing Technology and Science (CloudCom). IEEE, Dec 2014.

[10] J. Singh, T. Pasquier, J. Bacon, H. Ko, and D. Eyers, "20 Cloud Security Considerations for Supporting the Internet of Things," under review.

[11] J. McLean. Security models and information flow. 1990.

[12] Jatinder Singh, Thomas F. J.-M. Pasquier, Jean Bacon, "Integrating Messaging Middleware and Information Control", IEEE , 2015

[13] Ruoyu Wu, Gail-Joon Ahn, Hongxin Hu, Mukesh Singhal, " Information Flow Control For Cloud Computing", IEEE Conference publications , 2010.

[14] Safwan Mahmud Khan, Kevin W. Hamlen, Murat Kantarcioglu, "Silver Lining: Enforcing Secure Information Flow at the Cloud Edge" , IEEE, march 2014, pp. 37-46.

[15] "ApacheHadoop",http://hadoop.apache.org,2013.

[16] Thomas F. J.-M. Pasquier, Jatinder Singh, Jean Bacon, "Information flow control for strong protection with flexible sharing in Paas", IEEE, 2015.

[17] Abdulrahaman A.Almutairi and Muhammad I. Sarfraz, saleh Basalamah, walid g. Aref Ghafoor, "A distributed access control architecture for cloud computing " ,IEEE, 2012, pp. 36-44.

[18] Thomas F. J.-M. Pasquier, J Bacon, " FlowR: Aspect Oriented Programming for Information Flow Control In Ruby", IEEE, 2014, pp. 37-47

[19] Abdulrahaman A.Almutairi and Muhammad I. Sarfraz, saleh Basalamah, walid g. Aref Ghafoor, "A distributed access control architecture for cloud computing " ,IEEE, 2012, pp. 36-44.