Information Flow Control to Secure Data in the Cloud

Fahad Alqahtani
Department of Computer Science
University of Idaho
Moscow, ID, USA
Department of Computer Science
Prince Sattam Bin Abdulaziz University
Al- Kharj, KSA
Alqa0199@vandals.uidaho.edu

Salahaldeen Duraibi
Department of Computer Science
University of Idaho
Moscow, ID, USA
Computer Science Department
Jazan University
Jazan, KSA
dura6540@vandals.uidaho.edu

Predrag T. Tošić
Department of CSEE,Eastern Washington University
Spokane,WA,USA
Department of Mathematics and Statistics, *Washington State University*, Pullman,WA, USA
predrag.tosic@ieee.org

Frederick T. Sheldon
Department of Computer Science,
University of Idaho
Moscow, ID, USA
sheldon@uidaho.edu

***Abstract*— Data security remains a major concern for organizations considering the use of cloud services to store their confidential, business-critical data. In this paper, we investigate how *information flow control* can be used in the cloud to enhance the confidence of enterprises, so they can safely and securely adopt cloud solutions for their data storage needs. We discuss how different techniques can be used with the CloudMonitor tool to guarantee the protection of data in the cloud. We then give an overview of how centralized and decentralized information flow control systems operate, and the comparative advantages and disadvantages of each approach. Our analysis suggests that CloudMonitor can achieve better data security with the use of *decentralized* information flow control. We then discuss different decentralized information flow tracking tools applied to monitoring data in the cloud. CloudMonitor *enables* the consumers and the providers of cloud services to agree on acceptable security policies as well as their implementation, to ensure secure data storage in the cloud.**

Keywords*— *cloud security, information flow control, data tracking, cloud consumers, cloud service providers

I. INTRODUCTION

Cloud services are, in most cases, offered over the Internet. Based on the definitions by the National Institute of Standards and Technology (NIST), the consumer, in this context, is an organization that rents cloud services for its company's use. In contrast, the cloud is built by a cloud service provider (CSP), which is the owner of the IT infrastructure. CSPs offer cloud services to consumers for their business needs [1]. Hence, consumers must access cloud services remotely. But consumer organizations and providers may not be held liable to the same legal degree. For example, if the consumer and the CSP are both based in the same country, it is evident that they need to follow the same legal terms: i.e., the laws of their country. However, if the consumer and the provider are situated in different countries, then each will be liable under the laws of their respective country. Consequently, the data privacy regulation that the consumer and the provider are obligated to follow may be different [3]. The CSPs are managers of most cloud service resources; although, the exact boundaries of who gets to manage which resources depend on the cloud deployment model. In public cloud services, there may exist several organizations that use the same service owned by a single CSP. Such organizations are referred to as *co-tenants*. In general, these different organizations sharing a public cloud need not trust each other and do not want other co-tenants to have access to their organization's data.

Cloud computing technologies pose significant security risks to consumers and service providers alike. Like most other Internet-based technologies, cloud services suffer from several security issues. Key challenges identified in cloud security are that consumers do not have a say in the security of their data once uploaded to the cloud. One common consequence of this is that companies that deal with sensitive data are reluctant to adopt cloud services to share storage space with unknown tenants [2]. Securing data both at the consumer end and in the cloud is one of the critical challenges yet to be fully solved [3].

Conventional security measures have already been considered in cloud environments. However, those measures cannot achieve the security needs of some consumers [4]. We propose decentralized information flow control for securing data in the cloud. In particular, we suggest the use of information flow control to attach security policies to consumer data. These policies should be used at runtime to control where the data flows. In our data-centric model, the proposed information flow control can locally monitor

978-1-7281-7624-6/20/$31.00 ©2020 IEEE
DOI 10.1109/CSCI51800.2020.00241

employees' actions transparently, while also reinforcing consumer confidence in the safety of their data by allowing them to audit it within the cloud.

II. BACKGROUND AND RELATED WORK

The different cloud service models outline standard threat models. Subsequently, typical methods followed to protect the cloud from those threats are discussed; finally, an overview of information flow control mechanisms and security concerns are provided.

A. Cloud computing

The NIST defines cloud computing as: "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management efforts or service provider interaction" [5].

Cloud computing is not a new technology; it merely combines already known and established technologies, such as storage and server virtualization, with infrastructure management technologies to provide and control on-demand services. The cloud computing reference architecture defines five major actors, including cloud consumer, cloud provider, cloud carrier, cloud auditor, and cloud broker. The cloud consumer is the principal stakeholder that uses cloud computing services. A cloud consumer represents a person or an enterprise that maintains a business relationship with and uses the benefit of a cloud provider [6]. A cloud consumer browses the service catalog from a cloud provider, requests the appropriate service, sets up service contracts with the cloud provider, and uses the service [7]. The cloud consumer may be billed for the service provisioned and needs to arrange payments accordingly.

Cloud consumers use *Service-Level Agreements* (SLAs) for specifying the technical performance requirements to be fulfilled by a cloud provider [7-12]. SLAs can cover terms regarding the quality of service, security, and remedies for performance failures. A cloud provider may also list in the SLAs a set of restrictions or limitations and obligations that cloud consumers must accept [13]. In a mature market environment, a cloud consumer can freely choose a cloud provider with better pricing and more favorable terms. Typically a CSP's public pricing policy and SLAs are non-negotiable, although, in practice, a cloud consumer who has heavy usage might be able to negotiate for better contracts [14].

Cloud service models differ in the level of abstraction of the service the provider delivers to the consumer, as shown in Table 1. For instance, in IaaS, consumers enjoy the highest level of control over the cloud infrastructure compared to PaaS and SaaS. Conversely, the provider's capability to control diminishes from SaaS to PaaS to IaaS. Table 2 describes different activities that can be performed by consumers and providers concerning cloud delivery models. Naturally, consumers can choose one or more services of available delivery models to fulfill their requirements [5].

B. Cloud deployment models

There are four major cloud deployment models in everyday use; private cloud, community cloud, public cloud, and hybrid cloud [1]. A private cloud is a cloud with an infrastructure dedicated to a specific organization. The infrastructure can be on- or off-premise. Examples include major financial institutions such as big banks and central government agencies that do not want to share cloud resources with anyone else and are willing to pay the premium for the exclusive use of private cloud infrastructure and resources.

In a public cloud, the infrastructure is shared by different organizations, which in general, are various cloud consumers that do not mutually trust each other. This mistrust implies that many organizations will only store their non-confidential or less business-critical data in a public cloud. The public cloud infrastructure is usually off-premise (besides, different organizations using a public cloud service, in general, are not co-located with each other) [5].

A community cloud is an infrastructure that is accessed by a particular set of consumer organizations that share similar interests. The cloud infrastructure itself can be on or off-premise. The critical difference between the community cloud and the public cloud is that in a community cloud scenario, the different organizations sharing the cloud resources in general trust each other [5].

Last but not least, the hybrid cloud is a composition of two or more other types of clouds; the most common hybrid model involves some combination of private and public cloud services. Due to its flexibility concerning different consumers' needs, hybrid cloud solutions are becoming increasingly popular recently [5].

C. Cloud security concerns

Once consumers adopt cloud services and submit or upload their data, it will be difficult for them to monitor the data for security reasons. Hence, consumer organizations must entrust the security of

their data to the cloud provider. Providers are reluctant to share the security posture of the cloud with consumers in fear of a damaged reputation [15]. The trust issue might be the main reason why a company dealing with sensitive data might not choose to adopt cloud services [16].

There is a need for methodologies that enable consumers to monitor the security status of their data in the cloud. Consumer organizations, however, generally do not need the same level of protection for all of their data [17]. Some consumer organizations will adopt cloud services for some part of their data. But it does not leverage them from security issues that come with the cloud, especially the problems that may arise from insiders, that is, an organization's employees who may breach company security policy.

There are several essential cloud security requirements reported in the literature: authentication, authorization, accountability, and privacy [18]. These are commonly referred to as confidentiality, availability, and integrity of the data in the cloud. The confidentiality entails not exposing the consumer's data privacy to a third party without the consent of the consumer. Integrity is about data manipulation; it refers to whether or not the data residing in the cloud has been compromised and changed by unauthorized parties [19]. Availability is related to the pledge that when a consumer needs his data, the data should be available without delay. Authentication is to verify the ownership of data whenever claimed by a person to prove whether a person has the right to access or perform an activity on data residing in the cloud [20]. Accountability is holding whoever acts on the data accountable. It is sometimes also referred to as non-repudiation [21]. Therefore, before employing the cloud, an organization has to verify the security status of the cloud service and must be aware of the risks that may come with the adoption of cloud services. Likewise, an organization should know the security benefits that may come with the adoption of any cloud services. Hence, the decision of either moving data to the cloud or keeping it at the local premises (such as an organization owning its own data center) depends on whether the risks of adopting the cloud outweigh the expected benefits of cloud adoption [22].

D. Information flow control (IFC)

Secure access control models can be Mandatory Access Control (MAC) or Discretionary Access Control (DAC) systems [23]. IFC is a MAC model that uses security labels attached to data to control data propagation [24]. It is data-centric to track or limit data propagation. It differs from DAC models that give only focusing on where access control is happening in the code of an application. For MAC-based systems using the IFC model, the security policy is defined for the whole system. Another option that makes IFC more useful in data security is that it allows data to be more restrictively or less restrictively labeled [25].

TABLE 1: CLOUD DELIVERY MODELS AND LEVELS OF CONTROL BY CONSUMER AND PROVIDER

Cloud Delivery models	**Level of control granted to consumers**	**Functionality made available to consumers**
SaaS	Usage and usage-related configuration	Access to the front end and user interface
PaaS	Limited administration	Moderate level of administration, control over IT resources relevant to consumer's usage of the platform
IaaS	Full administration	Full access to virtualized infrastructure-related IT resources, and possibly, to underlying physical IT resources

There are centralized and decentralized IFC systems [26]. A centralized IFC system has a fixed set of labels and a central authority that control the labeling of the data. On the other hand, the decentralized IFC system dynamically introduces new tags into the runtime system, with mutual distrust and decentralized authority [27]. Decentralized IFC allows the owners of the data to create labels and to control data propagation inside a single application or across applications. Information flow control can be used in preventing data leakage within an application and across different applications [28].

IFC can be enforced at the language level or the protection domain level [29]. For example, tools proposed in [30-32] add DIFC related annotations in the source code and analyze the information statically. Such enforcement can provide flexibility, portability, and fine granularity at the byte level.

Existing tools enforcing DIFC at the protection-domain level include Asbestos [33], HiStar [34], and Flume [35] making annotations at the granularity of processes. Asbestos [33] and Flume [35] can enforce operating systems based IFC. Asbestos is an operating system that can fully enforce IFC, while Flume is a software that runs atop of Linux OS. Asbestos [33] employs an abstraction of events processes to reuse the protection domain, and a base process can spawn multiple event processes that use together most of the memory pages with the base process. That is to reuse resources across several event processes to provide efficient sharing between base processes and event processes. In Flume [35], the DIFC works at the domain protection level, including rules that allow or forbid inter-domain information flow and regulations that update security labels when the flow is allowed. However, the performance of using DIFC at the process level is significant when there is massive inter-process communication in an application. That is, Flume typically imposes more than 30% of performance overhead on real applications [29]. In many contexts, such overhead may be considered prohibitively high.

TABLE 2: CONSUMER VS. PROVIDER ACTIVITIES ACROSS DIFFERENT CLOUD DELIVERY MODELS

Cloud Delivery models	**Cloud consumer activities**	**Cloud provider activities**
SaaS	Uses and configures cloud	Implements, manages, and maintains cloud services. Monitors usage by cloud consumers
PaaS	Develops, tests, deploys, and manages cloud services and cloud-based solutions	Pre-configures platforms and provisions middleware and other needed IT resources as required. Monitors usage by the cloud consumer
IaaS	Sets up and configures bare infrastructure and installs, manages, and monitors any needed software	Provisions & manages the physical processing storage networking and hosting required. Monitors usage by cloud consumers.

An improved user-space DIFC proposed in duPro [29] has an efficient framework by allowing applications to control information flow between components to enhance their security. In duPro, protection domains are put in place using software-based fault isolation for protection domains. The use of information flow rules in DIFC systems is to provide security to a class of insecure behaviors the system prevents and transparency to a level of security programs that the system executes with unmodified semantics [36].

Related work

Accessing data in the cloud needs to flow secure and resilient access protocols. Some researchers have contributed towards accomplishing secure access in the cloud. However, only a few have employed IFC in their studies. One example is *FlowK*, which provides a continuous security mechanism to cloud using IFCs [37]. The technique enforces fine-grained security policy at the application level. A framework designed for deploying IFC-aware web applications in the cloud tests the system. In [38], researchers investigate an IFC system to relieve the burden of understanding the particulars of the data protection from tenants and providers of PaaS. The study proposes that DIFC is suitable for the protection of data integrity and secrecy in PaaS applications.

In [23], the study focuses on the protection of data in the cloud concerning its geographical location and the jurisdiction under which data in the cloud is liable to be one of the main concerns of cloud consumers. Likewise, in [39], Awani et al. explore IFC to monitor the information flow of the data exchange between different components or applications in the cloud. Remarkably, the study focuses on labeling or tagging the data owned by other users for traffic isolation.

In [40], the study provides a thorough discussion of how to protect data shared between different applications with the use of IFC. These researchers argue that IFC – enabled cloud would ensure that policies are enforced as data flows across all applications without requiring unique sharing mechanisms. [41] proposes a cloud service architecture that isolates user's activities in the cloud. The architecture employs DIFC to prevent vulnerabilities from the cloud by protecting malicious users from gaining unauthorized access to the applications in the cloud. The system limits individual user's operations or access to a particular cloud service. Shyamasundar et al. [42] have used IFC for building secure and privacy-aware of hybrid cloud services. The researchers have proven that their

system proposed in this study is forensic-ready by design because it provides necessary forensics information from hybrid services. Secure-ComFlow [26] is another system that employs IFC to secure cloud environments. The study specifically focuses on the data migration to the cloud from local infrastructures of companies. The system is user-oriented because it gives users the opportunity of specifying the IFC policy about their data.

III. CLOUDMONITOR

The CloudMonitor model is published in one of our previous works [48]. The CloudMonitor secures consumer data in the cloud. Fig. 1 illustrated that the model consists of two parts. The model helps consumers to ensure independently the security of their data once uploaded in the cloud. The second part of the model is delivered to the consumer as a service by the provider. The model uses information flow control to accomplish its job. The focus of the model will be on the data Storage as a Service model (IaaS model).

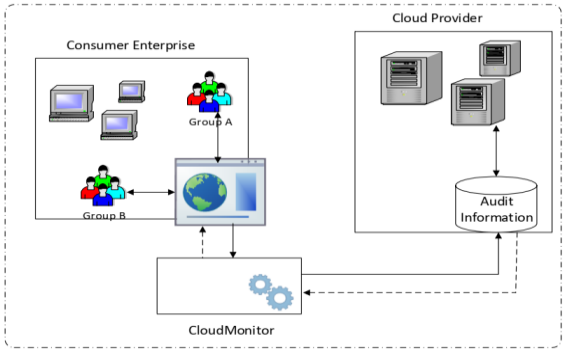

Fig 1. CloudMonitor

Between the two main types of IFC systems, DIFC remains most relevant for cloud solutions, since the cloud itself happens to be a very complex distributed system. In particular, for the IaaS model, IFC can manage and secure information flow both within a single virtual machine and among different virtual machines. In the model, dynamic data flow tracking of IFC in the cloud is used. Dynamic data flow tracking (DDFT) or runtime taint tracking is a minimal form of IFC. DDFT usually analyzes and enforces data flow in applications. Similarly, to accomplish data isolation in the cloud, the model isolates data via virtualization technology. Once the isolation is achievable, how the data propagates throughout the application is tracked. Consequently, the granularity at which the data is tracked is significant. Commonly, data can be tracked at the domain level, process level, variable level, or message level. The proposition here works at the message level. Worth mentioning is that by now, side channels of communication are not considered. Finally, the data flow enforcement is performed on crucial isolation boundary crossings.

The design principles of our model are to be elastic, easily deployable, and usable for information flow tracking. The model is expected to provide a variety of security requirements needed both from consumer and provider sides of the cloud environment.

The system based on our CloudMonitor will be compatible with existing applications and operating systems. It will also have to be parsimonious in how it propagates taint yet maintains soundness. In addition, the model should have the ability to conduct data isolation in the cloud; data isolation in this context means preventing data exchange between different applications. Data coming from cloud to consumer premises needs to be tainted.

IV. MODEL IMPLEMENTATION AND TECHNIQUE SELECTION

There are hardware- [43, 44] and software-based IFC systems. Hardware-based IFC implementations are out of our scope. Software-based IFC systems include IFC enforced by operating systems. In an operating system based IFC, data tracking is done at the process level. Here, processes and persistent data are labeled. Hence, whenever persistent data is accessed, and when inter-process communications happen then, the labels are propagated. DStar [45] translates the security labels between instances to enable IFC in distributed systems. And Aeolus [46] provides IFC tracking cross-host communication. But to accomplish the work, it runs on Asbestos across a distributed system. Nevertheless, Flume suffers from security issues inherent in systems on which it runs. DStar is appropriate for the cloud and, by its nature, can operate over a range of operating systems such as Flume and Linux. Aeolus has a trusted computing base but on Asbestos. It extends Asbestos to run distributed communication. That is, applications run on an authorized basis (filtering I/O, inter-thread, and external communications), enforcing the policy associated with the data.

Apart from the hardware and operating system based IFC systems, there are middleware level IFC systems. These include DEFcon [38] and SafeWeb [47]. SafeWeb mitigates against policy violations in multi-tier applications. Besides, using IFC to track data flow ensures end-to-end data confidentiality and integrity overall web application tiers. Coming back to the storage as service that our model is intended to work in, we now explore how these discussed IFC

schemes relate to our work. Our work mainly focuses on the IaaS service model. Therefore, we will be tracking the data flow residing on cloud virtual machines (VMs). Any fine-grained distributed IFC implementation that can track data flow across VMs is of utmost importance here. Such IFC implementation can be run by the consumer organizations over their adopted IaaS infrastructure. And this is where the first part of our model comes in its implementation.

Based on the reviewed IFC implementations, the unit of isolation is done based on the tracking granularity, which may happen at the process, thread, or object level. The second part of the model needs cloud provider participation in the IFC. The VMs extended to the consumer are exposing labels. In this part of the model, the provider is unlikely to modify the security policies of consumers adopted IaaS service. However, the provider can indirectly influence the flow exchange between different VMs, from the same consumer or those belonging to different consumers, through IFC at the network level. Our prototype will use some popular mechanisms found in the existing DIFC systems. The implemented model will adopt its characteristics from Flume and Asbestos. ownCloud will be used for the implementation of the model. ownCloud is a suite of client-server software for creating and file hosting services. Therefore, ownCloud will ensure the enforcement of isolation among users. The model not only provides data isolation but also policy for data sharing. We will develop a prototype for the demonstration of the feasibility of our Distributed IFC model.

V. CONCLUSIONS

In this paper, we have identified that decentralized IFC is suitable for the CloudMonitor model of protecting data in the cloud. Decentralized information flow control has been shown to protect consumer data integrity and secrecy. We have also discussed how DIFC works, highlighting different techniques that can support our model to be implemented in its intended cloud environment (SaaS).

The policy specification, translation, and enforcement that can support the model are explained in some detail. Likewise, the security status of consumer data in the cloud has been highlighted via an investigation of the audit logs. We point out that DIFC must not create an excessive burden and computational overhead for the cloud service provider; this topic will be elaborated upon in detail in our future work.

REFERENCES

[1] Mell, P. and T. Grance, *The NIST definition of cloud computing.* 2011.

[2] Arregoces, M., N. Bagepalli, and S. Chandrasekaran, *Hybrid cloud security groups*. 2019, Google Patents.

[3] Jakóbik, A., *Stackelberg game modeling of cloud security defending strategy in the case of information leaks and corruption.* Simulation Modelling Practice and Theory, 2020. **103**: p. 102071.

[4] Wang, H., et al., *A high-level information flow tracking method for detecting information leakage.* Integration, 2019. **69**: p. 393-399.

[5] Bohn, R.B., et al. *NIST cloud computing reference architecture*. in *2011 IEEE World Congress on Services*. 2011. IEEE.

[6] Felici, M. and S. Pearson, *Accountability for Data Governance in the Cloud*, in *Accountability and Security in the Cloud*. 2015, Springer. p. 3-42.

[7] Doelitzscher, F., *Security Audit Compliance For Cloud Computing.* 2014.

[8] Thorpe, S., et al. *Towards a Forensic-based Service Oriented Architecture Framework for Auditing of Cloud Logs*. in *Services (SERVICES), 2013 IEEE Ninth World Congress on*. 2013. IEEE.

[9] Ruan, K. and J. Carthy, *Cloud Computing Reference Architecture and Its Forensic Implications: A Preliminary Analysis*, in *Digital Forensics and Cyber Crime*. 2013, Springer. p. 1-21.

[10] Archer, J. and A. Boehm, *Security guidance for critical areas of focus in cloud computing.* Cloud Security Alliance, 2009. **2**: p. 1-76.

[11] Liu, F., et al., *NIST cloud computing reference architecture.* NIST special publication, 2011. **500**: p. 292.

[12] Reilly, D., C. Wren, and T. Berry. *Cloud computing: Forensic challenges for law enforcement*. in *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*. 2010. IEEE.

[13] Kearney, K.T. and F. Torelli, *The SLA model*, in *Service Level Agreements for Cloud Computing*. 2011, Springer. p. 43-67.

[14] Macías, M. and J. Guitart, *SLA negotiation and enforcement policies for revenue maximization and client classification in cloud providers.* Future Generation Computer Systems, 2014. **41**: p. 19-31.

[15] Hou, H., J. Yu, and R. Hao, *Cloud storage auditing with deduplication supporting different security levels according to data popularity.* Journal of Network and Computer Applications, 2019. **134**: p. 26-39.

[16] Moussa, A.N., N.B. Ithnin, and O.A. Miaikil. *Conceptual forensic readiness framework for infrastructure as a service consumers*. in *2014 IEEE Conference on Systems, Process and Control (ICSPC 2014)*. 2014. IEEE.

[17] Wang, Z., et al., *An empirical study on business analytics affordances enhancing the management of cloud computing data security.* International Journal of Information Management, 2020. **50**: p. 387-394.

[18] Kumar, R. and R. Goyal, *On cloud security requirements, threats, vulnerabilities and countermeasures: A survey.* Computer Science Review, 2019. **33**: p. 1-48.

[19] Walia, M.K., et al., *Cloud Computing Security Issues of Sensitive Data*, in *Handbook of Research on the IoT, Cloud Computing, and Wireless Network Optimization*. 2019, IGI Global. p. 60-84.

[20] King, N.J. and V. Raja, *Protecting the privacy and security of sensitive customer data in the cloud.*

Computer Law & Security Review, 2012. **28**(3): p. 308-319.

[21] Alassafi, M.O., et al., *A framework for critical security factors that influence the decision of cloud adoption by Saudi government agencies.* Telematics and Informatics, 2017. **34**(7): p. 996-1010.

[22] Ramachandra, G., M. Iftikhar, and F.A. Khan, *A comprehensive survey on security in cloud computing.* Procedia Computer Science, 2017. **110**: p. 465-472.

[23] Pasquier, T.F.-M. and J.E. Powles. *Expressing and enforcing location requirements in the cloud using information flow control*. in *2015 IEEE International Conference on Cloud Engineering*. 2015. IEEE.

[24] Dontov, D. and M. Klymenko, *Decentralized Access Control for Cloud Services*. 2019, US Patent App. 16/183,575.

[25] Han, J., et al., *Fine-grained information flow control using attributes.* Information Sciences, 2019. **484**: p. 167-182.

[26] Khurshid, A., et al., *Secure-CamFlow: A device-oriented security model to assist information flow control systems in cloud environments for IoTs.* Concurrency and Computation: Practice and Experience, 2019. **31**(8): p. e4729.

[27] Gollamudi, A., S. Chong, and O. Arden. *Information flow control for distributed trusted execution environments*. in *2019 IEEE 32nd Computer Security Foundations Symposium (CSF)*. 2019. IEEE.

[28] Chou, S.-C., *An agent-based inter-application information flow control model.* Journal of Systems and Software, 2005. **75**(1-2): p. 179-187.

[29] Niu, B. and G. Tan. *Efficient user-space information flow control*. in *Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*. 2013.

[30] Myers, A.C. and B. Liskov, *Protecting privacy using the decentralized label model.* ACM Transactions on Software Engineering and Methodology (TOSEM), 2000. **9**(4): p. 410-442.

[32] Zheng, L., et al. *Using replication and partitioning to build secure distributed systems*. in *2003 Symposium on Security and Privacy, 2003.* 2003. IEEE.

[33] Vandebogart, S., et al., *Labels and event processes in the Asbestos operating system.* ACM Transactions on Computer Systems (TOCS), 2007. **25**(4): p. 11-es.

[34] Zeldovich, N., et al., *Making information flow explicit in HiStar.* Communications of the ACM, 2011. **54**(11): p. 93-101.

[35] Krohn, M., et al., *Information flow control for standard OS abstractions.* ACM SIGOPS Operating Systems Review, 2007. **41**(6): p. 321-334.

[36] Alpernas, K., et al., *Secure serverless computing using dynamic information flow control.* arXiv preprint arXiv:1802.08984, 2018.

[37] Pasquier, T.F., J. Bacon, and D. Eyers. *Flowk: Information flow control for the cloud*. in *2014 IEEE 6th International Conference on Cloud Computing Technology and Science*. 2014. IEEE.

[38] Bacon, J., et al., *Information flow control for secure cloud computing.* IEEE Transactions on Network and Service Management, 2014. **11**(1): p. 76-89.

[39] Joshi, A., P. Purohit, and R. Jain, *A Simplified Rule Based Distributed Information Flow Control for Cloud Computing.*

[40] Pasquier, T.F.-M., J. Singh, and J. Bacon. *Information flow control for strong protection with flexible sharing in PaaS*. in *2015 IEEE International Conference on Cloud Engineering*. 2015. IEEE.

[41] Sun, Y., et al. *Pileus: protecting user resources from vulnerable cloud services*. in *Proceedings of the 32nd Annual Conference on Computer Security Applications*. 2016. ACM.

[42] Shyamasundar, R., N.N. Kumar, and M. Rajarajan. *Information-flow control for building security and privacy preserving hybrid clouds*. in *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. 2016. IEEE.

[43] Vachharajani, N., et al. *RIFLE: An architectural framework for user-centric information-flow security*. in *37th International Symposium on Microarchitecture (MICRO-37'04)*. 2004. IEEE.

[44] Suh, G.E., et al., *Secure program execution via dynamic information flow tracking.* ACM Sigplan Notices, 2004. **39**(11): p. 85-96.

[45] Zeldovich, N., S. Boyd-Wickizer, and D. Mazieres. *Securing Distributed Systems with Information Flow Control*. in *NSDI*. 2008.

[46] Cheng, W., et al. *Abstractions for usable information flow control in Aeolus*. in *Presented as part of the 2012 {USENIX} Annual Technical Conference ({USENIX}{ATC} 12)*. 2012.

[47] Hosek, P., et al. *S afeWeb: A Middleware for Securing Ruby-Based Web Applications*. in *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*. 2011. Springer.

[48] F. Alqahtani and F. Sheldon, "CloudMonitor: Data Flow Filtering as a Service," 2019 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2019, pp. 1454-1457, doi: 10.1109/CSCI49370.2019.00271