DOI: 10.3969/j. issn. 1673-5188. 2013. 04. 004
http://www.cnki.net/kcms/detail/34.1294.TN.20131127.1347.001.html, published online November 27, 2013

Preventing Data Leakage in a Cloud Environment

Fuzhi Cang, Mingxing Zhang, Yongwei Wu, and Weimin Zheng
(Department of Computer Science and Technology, Tsinghua National Laboratory for Information Science and Technology Tsinghua University, Beijing 100084, China)

Abstract

Despite the multifaceted advantages of cloud computing, concerns about data leakage or abuse impedes its adoption for security-sensitive tasks. Recent investigations have revealed that the risk of unauthorized data access is one of the biggest concerns of users of cloud-based services. Transparency and accountability for data managed in the cloud is necessary. Specifically, when using a cloud-host service, a user typically has to trust both the cloud service provider and cloud infrastructure provider to properly handling private data. This is a multi-party system. Three particular trust models can be used according to the credibility of these providers. This paper describes techniques for preventing data leakage that can be used with these different models.

Keywords

cloud computing; data leakage; data tracking; data provenance; homomorphic encryption

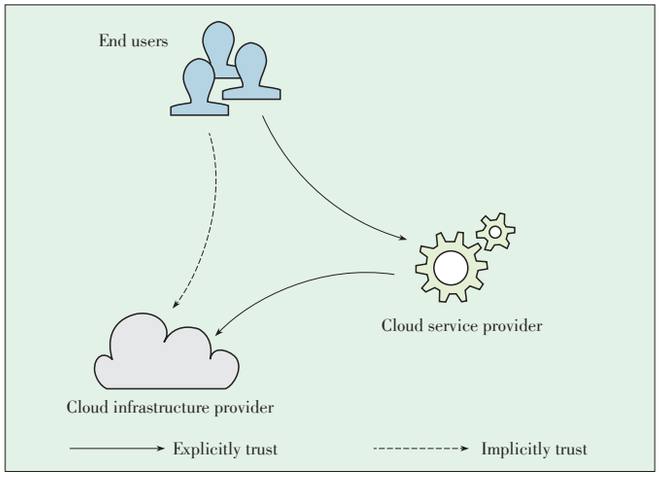
1 Introduction

Data protection is a top priority in the business world, especially in today's electronic era. In prior times, critical information, such as accounts and treaties, were recorded on paper and kept in safes. Such documents could not be stolen without great effort. However, such documents are now saved as digital bits and stored, transferred, and even traded using computers. As hacking methods become more advanced and electronic technologies become more complicated, the risk of important records being compromised is growing.

Compounding the problem is the fact that cloud computing has various benefits that have led to it being adopted at rapid speed by both companies and individuals. This quick uptake further weakens the user's control over their data. In the cloud-computing model, customers plug into the cloud via a network and access a shared pool of computing resources (e.g. networks, servers, storage, and applications) to process their data or hosting their own services. Although on-demand pricing and scaling are attractive, the fact that private customer data is handled and stored on systems outside the customer's control should not be forgotten. This is especially true for companies, such as Dropbox, whose entire business is built upon infrastructure as a service (IAAS). Such companies depend on the honesty and competency of the IaaS provider for data security. This honesty and competency has been called into question with a number of past security incidents and data breaches [1]-[3]. However, even if a service provider is considered trusted and follows best security practices, unauthorized access due to software bugs or misconfiguration is still a threat. A great deal of work has been done on the prevention, detection, and mitigation of such risks. Data owners want to have transparency and accountability so that they can see how their data is being handled by the cloud-hosted service and ensure their data is not being abused or leaked.

An investigation by M. Armbrust et al. [4] revealed that data confidentiality was the third-biggest obstacle to faster adoption of cloud computing. In 2010, Fujitsu Research Institute conducted a survey on potential cloud customers [5] and found that 88% of these potential customers were worried about who could access their data and demanded more awareness of what goes on in the back end.

In recent years, many techniques have been proposed to address these challenges, which, in our survey, divided into three main categories. There are three distinct participants in a typical cloud computing scenario: end users, cloud service provider, and cloud infrastructure provider (**Fig. 1**). In this work, we refer to both IaaS and PaaS providers as cloud infrastructure providers because they all provide base infrastructures for online services that are hosted on them. In this scenario, users need to implicitly trust the service provider to secure data. Users also need to implicitly trust the cloud infrastructure provider that hosts these services. Thus, three trust models can be established according to the credibility of these providers. In the first model, both the cloud service provider and cloud infrastructure provider are trusted (e.g. when using a private cloud). In the second model, the cloud infrastructure provider is usually a large and reputable company, like Google or Amazon), and is trusted. However, the cloud service providers are not trust-



▲Figure 1. Participants in cloud Computing.

ed. In the third model, Neither the cloud infrastructure nor the cloud service provider is trusted. In the remainder of this paper, we discuss current mechanisms that can be used when these different trust models are applied.

2 Tracking Data When Services are Trusted

If both the cloud service provider and cloud infrastructure provider are trusted, or when a private cloud is used, the trust scenario is not much worse than if a cloud were not being used. However, unauthorized access caused due to software vulnerabilities or bugs in the cloud service are could still cause data leakage. To close all loopholes, a data-tracking technique is proposed. With a standalone data-tracking system, all data access is traced and recorded. The administrator can prevent unauthorized data access by checking the operator's privilege before an operation is actually executed. An administrator can also audit a user's behavior by checking the logs afterward. This is an added security measure that alleviates user concerns about security in industries such as banking.

Although this tracking can be done within various layers of abstraction, the best choice is the file abstraction layer. Specifically, in a file-level tracking system, access to every file is intercepted before any action is taken, and relevant information is recorded for further analysis after the execution. This technique creates low additional overhead, is highly effective, and has already been adopted by several popular cloud providers. For example, Rackspace offers a feature called Access Log Delivery, which allows a user to enable logging for their private cloud files containers. Rackspace's blog states, "Customers sharing an account with multiple users can track which users are accessing their data, which are uploading the most content, etc. This gives IT departments more information for better serving their customers and possibly identifying problem users." [6]

Fig. 2 shows a typical architecture for those who want to implement a file-level tracking system from scratch. In this system, clients are only permitted to access cloud utilities indirectly via a portal. When a remote procedure call (RPC) is invoked by the client, the portal queries a database to determine whether a user has the authority to perform an operation. If the user does not have the authority, the request is denied. If the user does have the authority, subsequent operations are executed in the cloud via the application programming interfaces (APIs) provided by the service, and results are returned to the client. The portal also records information such as the user's IP address and operation type in a log database for further reference. If a smaller granularity, such as byte-level granularity, is desired, the range of read/write operations should also be recorded.

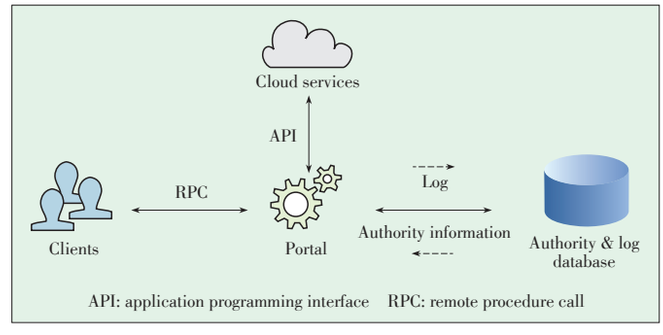
However, tracking data access may not be enough. Even applications are authorized to read a user's data and access output channels, leakage is still possible because an authorized application may be malwares or have been hijacked by another unauthorized program. In an attempt to prevent such leakage, much research has been done on information flow tracking (IFT). Privacy Scope is a novel system for tracking the movement of sensitive user data as it flows through applications [7]. It empowers users to run applications in their own environment and pinpoints information leaks, even when the data is encrypted. Privacy Scope systems are usually based on dynamic taint analysis, which is beyond the scope of this paper.

3 Auditing a Cloud Service Provider when the Cloud Infrastructure Provider is Trusted

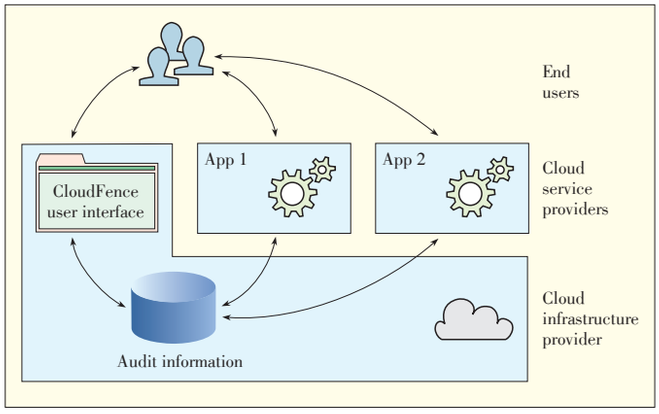
Compared with lesser-known companies that provide online services, big cloud infrastructure providers, such as Amazon, Google, and Microsoft, are well-known and highly reputable. Customers usually feel more comfortable handing control of their data to such providers. By enabling both direct and indirect users to inspect data uploaded to the cloud infrastructures, these companies create a direct relationship between the end user and themselves. This increases user confidence.

3.1 CloudFence

Much research has been done in this area. **Fig. 3** shows a high-level overview of CloudFence [8] as well as the main in-



▲Figure 2. The architecture of a file-level tracking system.



▲Figure 3. Overview of CloudFence.

teractions between parties involved in a CloudFence service. First, a user acquires a universally unique ID from the cloud infrastructure provider. Then, the user's sensitive data is tagged by the service provider with the supplied user ID and is tracked throughout the cloud infrastructure. At the same time, audit information is gathered and stored in the cloud infrastructure provider's database. CloudFence relies on data flow tracking (DFT) techniques to track the flow of data within all participating services hosted by the same cloud provider and to generate audit logs. Each cloud-hosted service has a separate audit log that can be correlated with other logs to create an audit report for the data owner. At any time, a user can monitor their data by using a user-friendly web interface or API. The main component of CloudFence is a DFT subsystem, which provides explicit, fine-grained, byte-level data flow tracking, and no modification of applications or underlying OS is needed. The subsystem can handle 232 different tags at the same time. It uses Intel Pin, a dynamic binary instrumentation toolkit, to instrument all instructions that operate tagged data, and it injects the tag propagation logic before the corresponding instructions. Both the original and additional instrumentation codes (i.e. the data tracking logic) are re-translated using Pin's just-in-time compiler.

The CloudFence system offers service providers an extra feature that reinforces the trust relationship between them and their users. It can also be used as the basis of a tag-based security system that prevents inadvertent leaks or unauthorized data access. The cloud infrastructure provider can improve transparency for their users by integrating CloudFence into their infrastructure, and this potentially increases the customer base.

3.2 Provenance

Besides the tags used in CloudFence, there are several other systems that provide similar functionality (called provenance). In [9], provenance is also referred to as lineage and pedigree and is defined as "the information that helps determine the derivation history of a data product, starting from its original resources." The provenance of a data block mainly includes information about where the data is derived and how the root ancestors transformed into the current data. Provenance-aware Storage System (PASS) [10] is one of the most-cited early provenance systems. In automatic provenance collection and maintenance, provenance is treated as a first-class object. The second PASS prototype [11] allowed the original system to operate one level of abstraction (which better supports a user's need to determine their data's movement) and answer questions that require an integrated view of the provenance. To facilitate cloud computing, Macko et al. extend the second PASS prototype and modify the Xen hypervisor to collect provenance from running guest kernels [12].

Adding provenance mechanisms to cloud computing and storage is essential to improve transparency and ensure accountability for data managed in a cloud. Provenance mechanisms are crucial for further adoption of cloud services [9].

4 Protecting Data Without Trusting a Cloud Service Provider

In light of past security incidents and the disclosure of the NSA's PRISM program, even big, reputable online services cannot be fully trusted. Thus, the most sensitive data, such as passwords, should not be devolved to cloud providers. The question becomes: Can we still leverage the benefits of cloud computing without needing to trust the cloud provider?

4.1 Fully Homomorphic Encryption

The most straightforward answer to this question is encryption. If all data is encrypted before being uploaded to the cloud, and the key is kept with the user, there is no need to worry about the risk presented by the cloud provider. However, even though encryption helps secure data stored in the cloud, it does not necessarily protect the data when it is being processed on remote infrastructure. That is, although the data can be stored in an encrypted form in the cloud, it generally has to be decrypted before being processed. The data is vulnerable when processed in a cloud. To address this challenge, fully homomorphic encryption, has emerged. In the future, this technique might allow encrypted data to be directly processed in the cloud.

The basic idea of this technique is that a specific encryption function allows a defined set of operations (e.g. addition, multiplication) to be performed directly on the encrypted data to produce an encrypted result. When this result is decrypted, it is the same as that obtained by operating on plain-text data.

In other words, for the numbers a and β, the suitable encryption function is $Encrypt(x)$, and the corresponding decryption function is $Decrypt(x)$. If $a \oplus \beta = \gamma$ and $Encrypt(a) \oplus Encrypt(\beta) = \gamma^*$, then $Decrypt(\gamma^*) = \gamma$, where $\oplus$ is a binary function supported by the encryption function. Therefore, if addition is supported, two encrypted values are

added to get a sum that, when decrypted, is the sum of the original two numbers. To be fully homomorphic, the code must allow a third party to add and multiply numbers contained within it without the need for decryption.

This may sound like a magic. In fact, until 2009, no one was sure whether homomorphic encryption was even possible. Then, a Stanford student named Craig Gentry proved its practicability in his PhD thesis [13]. This was a step in the right direction, but Gentry only provided an existence proof that showed that homomorphic encryption was no longer impossible. He did not implement the encryption in practice. Gentry estimated that performing a Google search with encrypted keywords would increase the amount of computing time a trillion times, which is unacceptable.

However, since Gentry's work, there have been a number of improvements that have made the scheme practical. At the end of 2009, M. Dijk et al. presented a simplified, fully homomorphic system called Brakerski-Gentry-Vaikuntanathan (BGV) homomorphic encryption scheme. This scheme uses integers only [14].

Moving forward, Victor Shoup and Shai Halevi, working at the IBM T J Watson Research Center, have just released an open source (GPL) C++ library called Helib. The code incorporates many optimizations to make the encryption faster [15]. As described in its package description, "HElib is a software library that implements homomorphic encryption (HE). Currently available is an implementation of the Brakerski-Gentry-Vaikuntanathan (BGV) scheme, along with many optimizations to make homomorphic evaluation run faster, focusing mostly on effective use of the Smart-Vercauteren ciphertext packing techniques and the Gentry-Halevi-Smart optimizations."

4.2 Trusted Cloud Computing

Besides advanced encryption, many other methods have been proposed for achieving trusted cloud computing. These methods involve a trusted cloud provider attesting service to end users. Although cloud service providers are making substantial efforts to secure their systems, insiders that manage these software systems at the provider's backend ultimately still have the technical means of accessing customer VMs [16]. Thus, there is a pressing need for a technical solution that guarantees confidentiality and integrity during processing and that can be verified by the customers.

To this end, the Trusted Computing Group (TCG) has proposed a set of hardware and software technologies that enable the construction of trusted platforms [17]. Specifically, the TCG proposed a standard for the design of the trusted platform module (TPM) chip, which is often bundled with commodity hardware. The TPM contains a private key that uniquely identifies the TPM, and it also contains some cryptographic functions that cannot be modified. Just like secure sockets layer (SSL), some manufacturers sign the corresponding public key to guarantee the correctness and validity of the chip. Then, a trusted platform can be implemented by leveraging the features of TPM chips to build a TPM attestation chain. This enables trustworthy remote attestations. Typically, this mechanism works as follows [18]: At boot time, the host (a physical machine) computes a measurement list (ML) comprising a sequence of hashes of the software involved in the boot sequence (i.e. BIOS, bootloader, and the software implementing the platform). This ML is securely stored inside the host's TPM, which cannot be modified even by the administrator of the machine. Then, a remote client can verify the platform in the following three steps:

1) The client sends the platform running at the host with a nonce N
2) The platform asks its local TPM to create a message containing both the stored ML and the received nonce N. Then, the platform encrypts the message with the TPM's private key.
3) The host sends the message back to the remote client, which can decrypt the message using the TPM's corresponding public key. This authenticates the host. This is achieved by checking that the nonce matches and the ML corresponds to a configuration that deems to be trusted.

Now the remote client can reliably identify the platform on an untrusted host, and this platform secures the users' data. For example, the trusted platform Terra implements a thin virtual machine manager (VMM) that enforces a closed-box execution environment [18]. This means that a guest VM running on it cannot be inspected or modified by a user with full privileges over the host. To put it simply, TPM allows a user to reliably detect whether or not the host is running a platform that can be trusted by the remote party. This trusted platform effectively secures the VMs running on it.

This attesting chain can be broken if the administrator can migrate the VM from one tested host to another untrusted host in the cloud environment. Therefore, Santos et al. [16] proposed the Trusted Cloud Computing Platform (TCCP), which provides a closed-box execution environment by extending the trusted platform to an entire IaaS back end. The TCCP guarantees confidentiality and integrity for a user's VM and allows a user to determine up front whether the IaaS enforces these.

5 Conclusion

This paper describes current mechanisms for preventing data leakage, one of the biggest concerns with using cloud-hosted services. Transparency and accountability for data managed in the cloud needs to be improved so that users can be more confident in the safety of their data stored in the cloud.

Usually, a user has to trust both the third-party cloud service provider and the cloud infrastructure provider to properly handle the user's data. This is a multiparty system. Three trust models can be used according to the credibility of these providers. There are techniques that can be used for each of these models.

When both the cloud services provider and cloud infrastructure provider are trusted, the main cause of data leakage is unauthorized access. However, this can be prevented by using a file-level tracking system. With a standalone tracking system, all data access attempts must be authorized. The administrator is empowered to audit user behaviors by checking recorded information.

If the cloud infrastructure provider is trusted but the cloud service provider is not, the infrastructure provider can be more transparent by offering auditing systems that can be used by indirect users to inspect the movement of their data. This kind of method often involves additional meta information, such as tags or provenance, that is propagated with the data.

It is still possible to use the cloud services without handing over control of data. Fully homomorphic encryption is an encryption technique that allows encrypted data to be directly processed in the cloud without decryption. Much effort has been put into achieving trusted cloud computing, which leverages the TPM to build a chain of trust that allows users to remotely check on trustworthiness.

Preventing data leakage must not only be limited to a single cloud service provider solution. Instead, data movement within multiple clouds or between the internet and cloud should also be investigated.

Acknowledgements

This Work is supported by National Basic Research (973) Program of China (2011CB302505), Natural Science Foundation of China (61373145, 61170210), National High - Tech R&D (863) Program of China (2012AA012600, 2011AA01A203), Chinese Special Project of Science and Technology (2012ZX01039001).

References

[1] Computerworld. (2010 Dec). *Microsoft BPOS cloud service hit with data breach* [Online]. Available: http://www.computerworld.com/s/article/9202078/Microsoft_BPOS_cloud_service_hit_ with_data_breach
[2] Sophos. (2011 Jun). *Groupon subsidiary leaks 300k logins, fixes fail, fails again* [Online]. http://nakedsecurity.sophos.com/2011/06/30/groupon- subsidiary-leaks-300k-logins-fixes- fail-fails-again/
[3] The Wall Street Journal. (2009 Mar). *Google Discloses Privacy Glitch* [Online]. Available: http://blogs.wsj.com/digits/2009/03/08/1214/
[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, pp. 50-58, 2010. doi: 10.1145/1721654.1721672.
[5] Fujitsu Research Institute. (2010). *Personal data in the cloud: a global survey of consumer attitudes* [Online]. Available: http://www.fujitsu.com/downloads/SOL/fai/reports/fujitsu personal- data-in-the-cloud.pdf
[6] Rackspace. (2012). *Logging for private cloud files containers* [Online]. Available: http://www.rackspace.com/blog/logging-for-private-cloud-files-containers/
[7] D. Zhu, J. Jung, D. Song, T. Kohno, and D. Wetherall, "Privacy scope: a precise information flow tracking system for finding application leaks," Department of Computer Science, UC Berkeley, Tech. Rep. EECS-2009-145, 2009.
[8] V. Pappas, V. P. Kemerlis, A. Zavou, M. Polychronakis, and A. D. Keromytis, "CloudFence: data flow tracking as a cloud service," *Proc. 16th Int. Symp. Research in Attacks, Intrusions and Defenses (RAID)*, Saint Lucia, October 2013. doi: 10.1007/978-3-642-41284-4_21.
[9] OQ Zhang, M Kirchberg, et al., "How to track your data: The case for cloud computing provenance," *3rd IEEE Int. Conf. Cloud Computing Technology and Science*, Athens, Greece, 2011, pp. 446-453. doi: 10.1109/CloudCom.2011.66.
[10] K.-K. Muniswamy-Reddy, et al., "Provenance-aware storage systems," *Proc. USENIX Ann. Technical Conf.(General Track)*, Boston, MA, USA, 2006, pp. 43-56. doi: 10.1.1.110.1442.
[11] K.-K. Muniswamy-Reddy, U. Braun, D. A. Holland, P. Macko, D. Maclean, D. Margo, M. Seltzer, and R. Smogor, "Layering in provenance systems," *Proc. USENIX Ann. Technical Conf.*, San Diego, CA, USA, 2009, pp. 129-142. doi: 10.1109/SSDBM.2004.23.
[12] P. Macko, M. Chiarini, and M. Seltzer, "Collecting provenance via the Xen hypervisor," *3rd USENIX Workshop Theory and Practice of Provenance*, Heraklio, Greece, June 2011.
[13] C. Gentry, "A fully homomorphic encryption scheme," Dissertation, Stanford Univ., CA, USA, 2009.
[14] M. V. Dijk, et al., "Fully homomorphic encryption over the integers," in *Advances in Cryptology-EUROCRYPT 2010*, Germany: Springer Berlin Heidelberg, 2010, pp. 24-43. doi: 10.1007/978-3-642-13190-5_2.
[15] HElib. [Online]. Available: https://github.com/shaih/HElib
[16] N. Santos, K. P. Gummadi, and R. Rodrigues, "Towards trusted cloud computing," *Proc. 2009 conf. Hot topics in cloud computing*, San Diego, CA, USA, June 2009, Article no. 3. doi: 10.1.1.149.2162.
[17] *Trusted Computing Group* [Online]. Available: https://www.trustedcomputinggroup.org
[18] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpa ch, I. Pratt, and A. Warfield, "Live migration of virtual machines," *Proc. 2nd ACM/USENIX Symp. Networked Systems Design and Implementation (NSDI)*, Berkeley, CA, USA, 2005, pp. 273-286. doi: 10.1.1.59.6685.

Manuscript received: October 10, 2013

Biographies

Fuzhi Cang (cfz05@mails.tsinghua.edu.cn) received his BE degree in computer science and technology from Tsinghua University in 2009. He is currently a MS candidate in Department of Computer Science and Technology, Tsinghua University, China. His research interests include distributed systems and cloud security.

Mingxing Zhang (zhangmx12@mails.tsinghua.edu.cn) received his BS degree in computer science and technology from Beijing University of Posts and Telecommunications in 2012. He is currently a PhD student in computer science at Tsinghua University, China. His research interests include distributed and parallel systems.

Yongwei Wu (wuyw@tsinghua.edu.cn) received his PhD degree in applied mathematics from the Chinese Academy of Sciences in 2002. He is currently a professor in computer science and technology at Tsinghua University, China. His research interests include parallel and distributed processing, and cloud storage. Dr. Wu has published more than 80 research papers and has received two Best Paper Awards. He is currently on the editorial board of the *International Journal of Networked and Distributed Computing and Communication* of China Computer Federation. He is an IEEE member.

Weimin Zheng (zwm-dcs@tsinghua.edu.cn) is a professor of computer science and technology at Tsinghua University, China. He received his BS degree and MS degree from Tsinghua University in 1970 and 1982. He is currently the director of the Chinese Computer Society. His research interests include computer architecture, operating systems, storage networks, and distributed computing. He is a senior member of the IEEE.