The State of Public Infrastructure-as-a-Service Cloud Security

WEI HUANG, AFSHAR GANJALI, BEOM HEYN KIM, SUKWON OH, and DAVID LIE,
University of Toronto

The public Infrastructure-as-a-Service (IaaS) cloud industry has reached a critical mass in the past few years, with many cloud service providers fielding competing services. Despite the competition, we find some of the security mechanisms offered by the services to be similar, indicating that the cloud industry has established a number of "best-practices," while other security mechanisms vary widely, indicating that there is also still room for innovation and experimentation. We investigate these differences and possible underlying reasons for it. We also contrast the security mechanisms offered by public IaaS cloud offerings and with security mechanisms proposed by academia over the same period. Finally, we speculate on how industry and academia might work together to solve the pressing security problems in public IaaS clouds going forward.

CCS Concepts: • **Security and privacy** → **Virtualization and security; Distributed systems security**; • **Networks** → *Cloud computing*

Additional Key Words and Phrases: Public Infrastructure-as-a-Service Cloud

ACM Reference Format:
Wei Huang, Afshar Ganjali, Beom Heyn Kim, Sukwon Oh, and David Lie. 2015. The state of public infrastructure-as-a-service cloud security. ACM Comput. Surv. 47, 4, Article 68 (June 2015), 31 pages.
DOI: http://dx.doi.org/10.1145/2767181

1. INTRODUCTION

Cloud computing has experienced a great deal of interest in both academia and industry in recent years. With an estimated industry size of $131B,[1] there is little doubt that it is both a successful technology and business model. By combining technologies such as virtualization, web APIs, and fast networks, cloud technology enables the provisioning and rental of compute infrastructure over the Internet. Similarly, by offering business advantages such as elasticity, the ability to defer capital expenditures, and the ability to outsource IT administration costs, cloud businesses provide many customers a valuable service.

Cloud computing services can be broadly classified into three categories based on the level of abstraction of computing resources they provide. At the highest level are Software-as-a-Service (SaaS) clouds, which provide complete software applications; Platform-as-a-Service (PaaS) clouds, which provide language runtimes and support libraries; and, finally, Infrastructure-as-a-Service (IaaS) clouds, which provide generic

[1]See http://www.gartner.com/newsroom/id/2352816.

This research is supported in part by funding from an NSERC Discovery Grant and an ORF-RE grant from the Ontario Ministry of Economic Development and Innovation.
Authors' address: W. Huang, A. Ganjali, B. H. Kim, S. Oh, and D. Lie, 10 King's College Road, Toronto, ON, Canada M5S 3G4; emails: wh.huang@mail.utoronto.ca, {soh, bhkim, soh}@cs.toronto.edu, lie@eecg.toronto.edu.

DOI: http://dx.doi.org/10.1145/2767181

computing infrastructure resources such as virtual machines and key-value object storage. Cloud computing infrastructure can be *public*, meaning that it is shared among multiple, mutually distrustful tenants, or it can be *private*, meaning that all resources are reserved for one tenant exclusively. Public clouds serve a much larger market and thus are generally available at a lower cost. However, the multitenant nature of public clouds means that they face many more security challenges than private clouds do.

In this article, we focus on the security of public IaaS clouds. Inherent to using the cloud is the assignment of trust, by the customer, to the Cloud Service Provider (CSP) to honestly and correctly provide services. In addition, since public cloud customers are mutually distrustful, customers also trust the CSP to protect their data from other CSP customers. This shift in trust and responsibility leads to the two new threats that cloud customers face: threats from a malicious CSP and threats from other customers of their CSP. These security threats can be directed at any of the three traditional security properties—confidentiality, integrity, and availability—of the customer's data. In addition, cloud computing also adds a new, fourth property, which is the security of the contractual relationship between the customer and the CSP. For example, the contract between the CSP and customer may specify certain properties of the service, such as constraints on the location where the data should be stored or promises of a certain level of performance. A compromise of contractual security would occur if the CSP provides a lower level of service than the customer paid for or if the customer were able to steal more service than was paid for.

In the past, a number of surveys on cloud computing security have been published in the literature [Armbrust et al. 2010; Chen et al. 2010b; Cloud Security Alliance 2011; Takabi et al. 2010; Barker et al. 2014]. However, the cloud computing industry has very recently reached a critical mass, with many CSPs introducing public IaaS services in only the past few years. For example, HP's public cloud just reached general availability at the end of 2012, Google's Compute Engine became generally available in the summer of 2013, while Verizon's public cloud completed its beta phase in late 2014. Of the CSPs listed in the Gartner's study of the IaaS industry [Gartner 2013], all have only been generally available for less than 5 years, with six joining in only the last 3 years (as shown in Table II). Yet, despite the fierce industry competition, we find many commonalities among the security properties of the CSP offerings, which indicates that the industry is on its way to establishing a set of standards or "best-practices" that define what security properties a public IaaS service should offer.

The main goal of this survey is to identify those IaaS industry best-practices and compare them with the research contributions from academia from the same period of time when this industry critical mass arose. Specifically, we set the following goals for this survey:

(1) Identify the security mechanisms used in the IaaS industry and identify cases where the industry has come to a consensus on best-practice solutions.
(2) Identify the IaaS security problems described by academia and the mechanisms proposed to solve those problems.
(3) Understand how industry best-practices may influence proposed academic solutions and how academic research may help industry further establish more best-practices.

Public IaaS services rely on a wealth of computing systems, including web interfaces, specialized virtualization hardware, and systems software. To make this survey tractable, we restrict ourselves to considering security mechanisms and academic work that is specific to public IaaS clouds, as opposed to computer systems security in

general. As a result, generic security measures such as software hardening, mandatory access control, and intrusion detection are out of scope. We instead focus on security measures that are specifically designed to improve the security of public IaaS cloud infrastructure and the customers of such infrastructure.

We begin in Section 2 by defining some of the terminology used in this article, as well as defining the attack model public cloud customers face. Section 3 summarizes the major security mechanisms that CSPs provide and identifies industry best-practices. Section 4 then summarizes major research work on IaaS cloud security in academia and reconciles this with the products and services that CSPs currently provide. Finally, we conclude with Section 5.

2. PRELIMINARIES

2.1. Definitions

IaaS services. An IaaS cloud provides compute and storage resources. Compute resources are provided in the form of Virtual Machines (VMs), which are rented by customers on an hourly basis. VMs are sometimes called "instances" in cloud literature, and the two are essentially equivalent. Typically, CSPs offer a menu of preconfigured VMs that customers can select from. VMs with more resources (i.e., faster CPUs or more memory) or specialized software will have a higher hourly rate than more basic VMs. CSPs may also bill for data transferred between VMs and hosts external to the CSP (i.e., external network traffic). Many CSPs also provide services through which customers may rent VMs or machines on a monthly basis (as opposed to hourly), but we exclude these because they are similar to traditional hosting services and outside the scope of this survey.

In general, CSPs provide two types of storage services: block storage, which is used to store virtual block devices for the VMs, and object storage, which provides a key-value store interface that can be used to store arbitrary objects. Block storage generally provides higher performance than object storage but can only be accessed by VMs in the compute service and offers a lower level of durability in the face of failures. In contrast, object storage, although slower, can be accessed by any client with an Internet connection and provides highly available and durable storage. For example, Amazon estimates the durability of their block store, called the Amazon Elastic Block Service (EBS) to between 99.5% and 99.9%, while they estimate the durability of their object store, called the Amazon Simple Storage Service (S3) to be 99.999999999% [Amazon AWS 2014]. CSPs typically charge fees based on the amount of storage space consumed and the volume of data transferred out of the object storage services.

Hypervisor. A hypervisor is a low-level software component that allows commodity compute hardware to be virtualized and partitioned into VMs, which can then be rented to customers by a CSP. Some examples of well-known commodity hypervisors are Xen, KVM, VMware, and Hyper-V.

Cloud control-stack. A hypervisor on its own cannot be used to implement an IaaS service. Customers typically interact with a web interface that allows customers to remotely create, provision, pause, resume, stop, and destroy VMs over the Internet. In addition, they also need an interface to access and manage data stored in block and object storage services. A cloud control-stack implements these interfaces, as well as the logic that links the customer-facing interface to low-level components such as the hypervisor, networking components, and storage technologies. Mature cloud control-stacks will offer a variety of web interfaces, including a human usable interface accessible via a web browser as well as programmatic interfaces accessible over hypertext protocols such as SOAP or REST.

OpenStack is an example of an open-source cloud control-stack.[2] It contains modules that implement interfaces and management logic for compute resources, networking, metering, block storage, and object storage. In addition, it also implements a federated identity service, called Keystone, which can authenticate humans using passwords and issues "tickets" that act as capabilities. The tickets can then be used by humans over a web session or embedded in application code. OpenStack also provides an interface called the Dashboard (Horizon) that humans can use via a web browser. OpenStack itself does not include a hypervisor, but it provides support for most commodity hypervisors. Similarly, it can support a variety of networked or local storage backends.

Customers and users. We distinguish a customer, who is an entity that has a business relationship with the CSP, from a user, which is an identity recognized by the CSP and that can be given certain privileges to customer-owned resources on the CSP. For example, an enterprise may establish a paying customer relationship with a CSP and make its employees users of the CSP. Each user will be given CSP privileges commensurate with his or her role within the enterprise. Another example is a customer who deploys an application on a CSP and distributes the application to end users. The customer will then make its deployed application a user of the CSP, thus giving it access to components and data hosted on the CSP.

2.2. Attack Model

Security literature from industry sources generally has two purposes. First, it serves to market products and services to potential customers by informing them of the security mechanisms in those products or services. Second, it provides documentation on how to use the security mechanisms for users of the products and services. Although the latter is often much more detailed than the former, there is no explicit requirement to outline the actual type of attack or threat the mechanism is supposed to protect against—this is usually left for the customer to infer.

In contrast, academic papers in security generally identify specific threats and attackers that they will analyze or develop solutions to defend against. As a result, when outlining the attacks and threats customers face in the cloud, we draw from the academic literature. Although each paper only deals with a specific threat, by surveying the threats and attack model of each paper, we can construct an *aggregate attack model*, which can serve as an attack model for which to evaluate any security mechanism, whether proposed in academic or implemented by industry.

To construct this attack model, we begin by first defining the cloud security properties of the customer that an attacker may wish to compromise. These properties include traditional security properties such as confidentiality, integrity, and availability, as well as a new one that is specific to the cloud business model, which we call *contractual security*.

Confidentiality. Customer confidentiality naturally encompasses the content of customer data that are used and stored on the CSP. In addition, we also include information such as data access patterns and existential information under confidentiality. For example, Google's Cloud Platform explicitly tells customers that the names of objects stored on its service exist in a global namespace, so that any customer can tell whether an object of some particular name exists and is being used by some other customer (although not necessarily which one).[3]

[2]See http://www.openstack.org.

[3]See https://developers.google.com/storage/docs/bucketnaming.

Integrity. Customer integrity also includes the content of customer data. However, since customers often move to the cloud because they would like to access their data and applications in a distributed setting, we also include the consistency of data under the integrity security property. Applications may depend on the consistency guarantees of the CSP to work properly [Terry et al. 2013; Kim and Lie 2015], and subversion of those guarantees by the cloud can result in subtle vulnerabilities, such as those caused by time-of-check to time-of-use errors.

Availability. Availability normally defines whether CSP resources are available when the customer wants to access them. In addition, we also include durability, which defines whether the data can be recovered by the customer if the data become unavailable for any reason. Another related concern is "lock-in," where data may be accessible, but customers are either unable to, or face great difficulty, if they want to retrieve their data and move them to a different CSP.

Contractual Security. Finally, CSPs typically provide a Service Level Agreement (SLA) that governs performance levels and availability. Furthermore, many promise additional security guarantees such as restricting the storage data to a certain region or legal jurisdiction or cloud-side encryption of data at rest. In exchange for these promises, customers agree to pay fees, thus constituting a contract between the CSP and customer. When either the CSP or the customer attempts to subvert the contract so that more or less service is received for the same amount of money, this is an attack on the contractual security between the CSP and the customer. Customers may attempt to attack the CSP to get more service or may attack other customers in an attempt to reduce the service they receive. CSP billing usually operates by multiplying a rate by an *easily verifiable* measure, such as the number of hours used or number of bytes transferred. However, the rates themselves are often determined by *difficult to verify* properties, such as performance, location, or the level of data replication. As a result, the security goal is often to ensure that these difficult to verify properties are properly delivered from the CSP to the customer as agreed.

Compromising one or more of these properties is generally the goal of an attacker on a cloud service. The security literature also deals with attackers with differing capabilities. However, we find that we can broadly categorize all cloud attackers into two categories: those that have compromised the CSP and those that have not.

Malicious CSP. A benign CSP that has been compromised by attackers is indistinguishable from a CSP that is inherently malicious, so we label any such CSPs as a "malicious CSP." By using a cloud service, cloud customers must trust and depend on the CSP to some extent. As a result, these scenarios are often the most difficult to defend against and therefore, the focus of a great deal of academic interest. However, the capabilities of malicious CSP attackers generally do not extend to components belonging to the customer, such as customer-owned machines and devices, which the customer can still use to protect data stored on the cloud (i.e., encrypt and verify signatures) or to detect misbehavior.

Malicious Cloud User. An attacker who hasn't compromised the cloud service is more limited because he only has the capabilities of a normal cloud customer. While such attackers are weaker, such capabilities are easier to gain because they do not rely on the presence of privilege escalation vulnerabilities in the CSP infrastructure. However, there may still be some limited capabilities that such attackers are assumed to have, such as the ability to co-locate themselves with the victim customer.

Given the capability of an attacker (control or non-control of the CSP) and goal (C:Confidentiality, I:integrity, A:availability and/or CS:contractual security), the works

in the literature divide themselves into several major attack techniques, which we show in Table I. We now describe the various techniques in detail.

Storage manipulation. An attacker who has privileged access to components of the cloud infrastructure has a much wider array of attack techniques available to him. To attack data stored in the cloud, an attacker who controls the block or object storage component of a cloud service could read, corrupt, or manipulate the data stored by the cloud. Arbitrary control over the storage layer gives an attacker great latitude to target the confidentiality, integrity, and availability of customer data. As a result, we see in the literature a variety of goals for various attack models that use this approach.

Storage monitoring. Rather than manipulating data, a stealthier attacker may simply just observe the access patterns of a victim to data they've stored on the cloud. Standard use of encryption will not hide these access patterns; thus, in this group, the goal of the attacker is to compromise the confidentiality of the data accessed by customers.

VM image sharing. Cloud services permit users to post and share VM images with other users. This can be used both by the sharer to trick victims into using malicious VM images, as well as by an attacker who harvests sensitive information inadvertently left on images by sharers. Thus, this attack techniques impacts the integrity of the victim user in the former case and the confidentiality of the victim sharer in the latter. Typically, both threats are treated together as a single attack model.

Compromised hypervisor. A significant threat to a user is an attacker who has compromised the hypervisor, the component that provides isolation between cloud user VMs. An attacker who compromises the hypervisor has a great deal of power. Similar to an attacker who controls the storage layer, such an attacker has both the greatest amount of visibility into customer data and resources, as well as the greatest ability to stealthily manipulate customer data. As a result, it is not a surprise that a great number of papers try to address this threat. Because control of the hypervisor gives an attacker control over essentially every aspect of a user's VM, making the attacker very strong, many papers do not explicitly state the attacker's goal. In such cases, we assume that such work implies that the attacker will compromise any of the victim's confidentiality, integrity, or availability after gaining control of the hypervisor. Some papers explicitly exclude or focus on only certain security properties. For example, some papers explicitly focus on verifying that computations in the cloud are performed correctly, or they may explicitly exclude availability, which is very difficult to guarantee with a compromised hypervisor.

Storage dishonesty. To save costs or hide failure, a malicious cloud service could claim to a customer that it has stored his data when in fact it hasn't. Similarly, a malicious CSP may claim to replicate data up to a certain level, but in reality provide a lower level of replication, making customers unable to recover from the corruption or loss of data. By concealing the actual storage status of data, the dishonest storage not only can harm the integrity of data, but also puts data availability for clients in jeopardy. To detect this, customers may selectively audit the cloud service in a way that can efficiently and with high probability detect if the cloud service has untruthfully claimed that the customer's data are stored when in reality they are not.

Location dishonesty. Many cloud providers enable users to choose the location where their data are stored or where their VMs run. In some cases, the CSP may charge more for certain locations that have higher costs or are perceived to be better for data storage.

Table I. Aggregate Attack Model

	Technique	Papers	C	I	A	CS
Malicious CSP	Storage manipulation	[Mahajan et al. 2010]		✗	✗	
		[Feldman et al. 2010]	✗	✗		
		[Shraer et al. 2010]		✗		
		[Puttaswamy et al. 2011]	✗			
		[Bessani et al. 2011]	✗	✗	✗	
		[Popa et al. 2011]	✗	✗	✗	
		[Kim and Lie 2015]	✗	✗	✗	
	Storage monitoring	[Islam et al. 2012]	✗			
		[Blass et al. 2014]	✗			
		[Williams et al. 2012]	✗			
		[Goldreich and Ostrovsky 1996]	✗			
		[Stefanov and Shi 2013]	✗			
		[Dautrich et al. 2014]	✗			
	VM Image sharing	[Wei et al. 2009]	✗	✗		
		[Bugiel et al. 2011]	✗	✗		
	Compromised hypervisor	[Santos et al. 2009]	✗	✗	✗	
		[Schiffman et al. 2010]	✗	✗	✗	
		[Santos et al. 2012]	✗	✗	✗	
		[Haeberlen et al. 2010]		✗		
		[Azab et al. 2010]	✗	✗	✗	
		[Keller et al. 2010]	✗	✗	✗	
		[Murray et al. 2008]	✗	✗	✗	
		[Butt et al. 2012]	✗	✗	✗	
		[Colp et al. 2011]	✗	✗	✗	
		[Wu et al. 2013]	✗	✗	✗	
		[Vu et al. 2013]		✗		
		[Parno et al. 2013]		✗		
		[Gentry 2009]	✗			
		[Zhang et al. 2011a]	✗	✗		
		[Szefer and Lee 2012]	✗	✗		
	Storage dishonesty	[Ateniese et al. 2007]		✗	✗	
		[Juels and Kaliski Jr 2007]		✗	✗	
		[Shi et al. 2013]		✗	✗	
		[Bowers et al. 2009b]		✗	✗	
		[Stefanov et al. 2012b]		✗	✗	
		[Bowers et al. 2009a]		✗	✗	
		[Curtmola et al. 2008a]		✗	✗	
		[Ateniese et al. 2011]		✗	✗	
		[Chen et al. 2010a]		✗	✗	
		[Halevi et al. 2011]		✗	✗	
	Location dishonesty	[Gill et al. 2010]				✗
		[Watson et al. 2012]				✗
		[Benson et al. 2011]				✗
	SLA dishonesty	[van Dijk et al. 2012]	✗			✗
		[Zhang et al. 2011b]	✗			✗
		[Juels and Oprea 2013]	✗	✗		✗
		[Bowers et al. 2011]			✗	✗
		[Chen and Chen 2014]				✗

(Continued)

Table I. Continued

	Technique	Papers	C	I	A	CS
Malicious Cloud User	Cache-based leakage channels	[Raj et al. 2009]	✗			
		[Vattikonda et al. 2011]	✗			
		[Aviram et al. 2010]	✗			
		[Ristenpart et al. 2009]	✗			
		[Mowery et al. 2012]	✗			
		[Kim et al. 2012b]	✗			
		[Zhang et al. 2012]	✗			
		[Zhang and Reiter 2013]	✗			
		[Varadarajan et al. 2014]	✗			
	General leakage channels	[Farley et al. 2012]			✗	✗
		[Azar et al. 2014]	✗			
		[Pattuk et al. 2014]	✗			
Total			39	33	25	9

A CSP may either maliciously violate user choices to reduce its own costs of physical server maintenance, data transfer, or other expenses, or may inadvertently violate user choices due to software bugs or misconfiguration.

SLA dishonesty. Works in the literature have examined other diverse ways that malicious CSPs may violate the security guarantees that cloud providers commonly claim to provide. For example, if the cloud provider claims that data are encrypted when they are not, it may harm confidentiality. Similarly, the CSP could claim to replicate data to a certain level when it hasn't, which violates data integrity and/or availability of the client's data.

Cache-based leakage channels. One technique that an attacker lacking privileged control of the CSP can use is to co-locate himself onto the same CPU or even the same core as a victim and observe the cache timing channel. In this attack, the confidentiality is the attacker's only target. He will run a specific workload on the cloud service whose speed will be highly dependent on the state of the cache. This exploits a timing channel between the victim's VM and the attacker's VM through the cache that can leak confidential information.

General leakage channels. This technique generalizes the concept of channels created in multitenant environments through the sharing of physical resources between resources controlled by the attacker and victim. Although these resources are logically isolated, timing and other side channels can allow the attacker to learn information about the victim or even affect the victim's performance. Like cache-based leakage, most of these techniques also focus on violating the victim's confidentiality. However, one also focuses on stealing resources from the victim [Farley et al. 2012]. In general, we find that an attacker who does not have privileges on the cloud infrastructure is generally limited to monitoring or manipulating side channels that are not explicitly closed by cloud infrastructure.

At the bottom of Table I, we show the total number of times each of the cloud user security properties of confidentiality, integrity, availability, and contractual security is examined in the literature. As one might expect, the majority of the literature focuses on examining threats to and protecting the confidentiality and integrity of cloud users because these are the main properties that cloud users are most concerned about. A fair amount of the literature also considers availability, notably when it comes to being able to access data stored on the cloud. Finally, it is interesting to note that very little

Table II. Cloud Service Providers Surveyed

Service Provider	Launch Year	Hypervisor(s)
Verizon Cloud	2014	Xen/VMware
Google	2013	KVM
Savvis Direct	2012	Xen/VMware
HP Public Cloud	2012	KVM
Dimension Data	2011	VMware
Tier 3	2011	VMware
Microsoft Azure	2010	Custom(Hyper-V)
Fujitsu Trusted Public S5	2010	Xen
GoGrid Cloud Platform	2009	Xen
Joyent Compute/Manta Storage	2009	SmartOS
Amazon EC2/S3	2008	Xen
Rackspace Public Cloud	2008	Xen
SoftLayer	Unknown	Xen

literature considers contractual security; that is, that cloud providers actually provide the level of service they promise to their customers.

3. INDUSTRY CLOUD SECURITY PERSPECTIVE

In this section, we summarize the mechanisms and approaches that major industrial CSPs use to protect both their customers and themselves from malicious attacks. To ensure that we have a representative view of the cloud computing industry, we examine the top 13 CSPs identified in Gartner's study of industry-leading IaaS providers [Gartner 2013], which are listed in Table II. The launch date given indicates the year the service became generally available and does not include any closed betas that the service may have had. We note that Gartner states that it did not include Google Cloud Platform in its 2013 report because it had not reached general availability at the time the Gartner report was written, and the Gartner report only included the VMware-based private cloud offering of Verizon.

In many of these cases, the surveyed CSPs also had other lines of cloud-related business such as traditional hosting services and services to rent sections of their data centers out as "private clouds," which are not included in this study—the information given here pertains only the public IaaS cloud component of their offerings. We restrict ourselves to only surveying the set of services defined by our definition of IaaS services given in Section 2.

Some CSPs only provide limited public details about their cloud infrastructure, operations, and capabilities. Although much information (and speculation) could be found via secondary sources, we restricted ourselves to only official marketing materials and documentation published by the CSP. We note that Savvis Direct, Rackspace, HP, and SoftLayer (Object Storage service only) are all based on OpenStack, and, in these cases, we used OpenStack documentation as a proxy if CSP documentation was unavailable.

Our goal is not to try to compare CSPs in an effort to determine the best one [Li et al. 2010], but instead to identify security mechanisms that have been implemented across a large portion of the CSP industry and thus could be considered standardized into a "best-practice." Similarly, we wish to identify the mechanisms whose utility are still under debate in the industry and are actively marketed by some cloud providers as differentiators against their competitors. We categorize the mechanisms by whether they protect customer security, CSP security, or contractual security between customer and the CSP. We explicitly exclude generic security mechanisms, such as software

hardening or mandatory access control, that are either not unique to public IaaS clouds or not mentioned in any of the CSP documentation.

3.1. Customer Security

Mechanisms that protect customers can be broken down into three subcategories: systems mechanisms, cryptographic mechanisms, and authentication and access control mechanisms.

3.1.1. Systems Mechanisms. Systems mechanisms comprise security implemented in software and hardware components within the CSPs infrastructure, such as hypervisors, firewalls, operational procedures, dedicated VMs, and corporate segregation.

Hypervisor. Hypervisors are a technology that has been around since the 1970s, and, as such, their design and implementation is fairly well understood [Goldberg 1974]. As a result, the use of hypervisors and the security guarantees they offer are very similar across CSPs. CSPs depend on hypervisors to isolate VMs from each other, both in terms of security and in terms of performance. Hypervisors also play a crucial role in protecting the CSP from malicious customers by confining customer code inside a VM. To implement hypervisor security, almost every CSP uses a mature commodity hypervisor such as Xen, KVM, or VMware. There are only two exceptions. One is Microsoft, which uses a specialized hypervisor influenced by their commercial Hyper-V product [Kaufman and Venkatapathy 2010]. The other is Joyent Compute, which uses a port of KVM to OpenSolaris called SmartOS.[4] Some CSPs have also made custom modifications to the hypervisor to make it more suitable for public cloud use [Amazon AWS 2014]. Thus, it is a CSP industry standard to either use a mature commodity hypervisor or one based on a commodity hypervisor in their public cloud.

Firewalls. Every CSP offers customer-controlled firewalls that allow customers to restrict traffic to and from their VMs. The granularity of such firewalls can be a single VM or a group of VMs. For example, Amazon uses the concept of a "security group," in which all VMs within the security group are subject to the same network firewall policy. A security group can consist of a single VM or a group of VMs. In addition, all CSPs implement firewalls outside of VMs, so that an attacker who compromises a VM does not automatically gain the ability to change its firewall settings. Thus, even if the firewall itself is implemented in software, the security offered is similar to that of a physically separate hardware firewall in a traditional, unvirtualized infrastructure. Some of the literature suggests that another rationale for this design choice may have been to facilitate certification of compliance to industry security standards for customers [Amazon AWS 2013]. The implementation of this mechanism, as well as the interface, is nearly identical in form and function across all CSPs.

Data center operations. The data center operation procedures that CSPs implement to protect customer data are identical across the industry and follow industry best-practices for implementing secure data centers in noncloud settings. All CSP data centers are physically secured with locks, alarms, guards, and cameras, and can only be accessed by a limited number of authorized personnel. When storage equipment is retired, it is disposed of in a way that prevents recovery of any data on the equipment, also according to industry standards. To protect customer durability and availability, we also note that all CSP data centers also contain environmental control systems and electrical backup that protect data from unavailability or destruction by fire, overheating, and power outages. In addition, data is generally replicated within the data

[4]See http://www.joyent.com/technology/smartos.

center to protect against equipment failure. Some CSPs also automatically replicate across data centers to protect against the failure of a data center, while others require customers to do the replication themselves.

Dedicated VMs. Dedicated VMs give customers the option, for a premium in cost, to provision their VMs on hardware that is not shared with any other customer. There is considerable disagreement among CSPs as to whether to offer dedicated VMs, as well as the premiums to charge. Of the CSPs surveyed, only Amazon, Fujitsu, and SoftLayer offer dedicated VMs to customers. Amazon charges a $2 flat rate plus a roughly 10% rate premium over similar, nondedicated VMs, whereas SoftLayer charges a premium that varies between 120% to 15% depending on the size of the VM. Fujitsu does not post any pricing information. In addition, SoftLayer also offers a "bare-metal" option, in which customers are given access to an unvirtualized machine on which they can install and run their own software.

We believe there are two reasons why dedicated VMs are not a standard offering by CSPs. First, if one completely trusts the isolation of the hypervisor and CSP infrastructure, then dedicated VMs are redundant since the existing infrastructure should provide all the security and performance isolation that is necessary. Thus, one reason behind the variety in offerings may be how much CSPs believe their customers trust the ability of their infrastructure to provide security and performance isolation. For example, Amazon markets its dedicated VMs as having the advantage of "ensuring that your Amazon EC2 compute VMs will be isolated at the hardware level."[5] Second, instead of offering dedicated VMs through the cloud, other CSPs may offer private cloud services or consulting services that help customers to set up private clouds on their own premises.[6] Thus, in these cases, these CSPs may not offer dedicated VMs to avoid product duplication rather than for any technical reason.

Corporate segregation. Corporate segregation describes the institutional procedures with which a CSP separates its corporate data and networks from that of its customers. All CSPs implement segregation and limit the ability of their employees to access infrastructure that contains customer data to the minimum that is required. For example, Amazon strictly separates all infrastructure between its cloud services product and other operations. In addition, access to cloud infrastructure is reviewed every 90 days. Although most CSPs are similar to Amazon, Google is unique in that it mixes infrastructure supporting its cloud products with that of its other products, implementing no clear separation. Google markets this as an advantage since cloud customers automatically gain the security measures that Google uses to protect its other products, as well as its own data. Although Google strays from the pack in this regard, we believe that it has a uniquely credible reason to do so since the vast majority of its other products also contain sensitive customer data; thus, it makes sense to apply the same security controls to its cloud product [Google 2012].

3.1.2. Cryptographic Mechanisms. Cryptographic mechanisms can be divided into those used to protect customer data while in transit and those used to protect customer data while at rest.

Cryptography while in transit. As is standard in web security, all CSPs use encrypted channels such as SSL or TLS to protect the confidentiality and integrity of customer data and commands while in transit. Internet-facing CSP servers are authenticated using certificates issued by well-known certificate authorities to prevent

[5]See http://aws.amazon.com/dedicated-instances/.
[6]See http://www.microsoft.com/en-us/server-cloud/solutions/virtualization-private-cloud.aspx.

man-in-the-middle attacks. Authentication of clients is done via other means inside the SSL or TLS channel, as described later in this section.

Cryptography at rest. Although CSPs all use the same mechanisms for protecting client data while in transit, they differ on how and whether they provide cloud-side encryption of client data at rest. Amazon encrypts and signs objects stored in its object storage service, but leaves images stored on its block storage service in plain text. OpenStack supports encryption and signing for its block storage service, but not for its object storage service,[7] Joyent Compute does not encrypt any data at all, stating that since the keys are stored by the CSP, cloud-side encryption does not increase customer security in its view. Instead, it recommends that customers always encrypt data themselves before storing it to the cloud, thus making cloud-side encryption useless.[8]

We believe Joyent Compute's argument may indicate why CSPs have not established an industry standard best-practice for cloud-side encryption. Indeed, Amazon gives customers the ability to disable cloud-side encryption if the customer has already encrypted the data himself before uploading it.

3.1.3. Authentication and Access Control Mechanisms. Although all CSPs perform authentication on customers and programs acting on behalf of a customer, not all CSPs implement access control mechanisms for customers. Thus, we discuss them separately here.

Authentication. All CSPs surveyed use passwords for authenticating humans. In addition, some offer the option of using a second factor, usually a soft or hard security token.

All CSPs also support programmatic access to cloud resources via a web-based API, usually REST or SOAP over SSL. However, CSPs differ heavily on how they perform authentication in their API, ranging from basic HTTP authentication, to cookie-based authentication, to certificate-based authentication, to tickets issued by single-sign-on (SSO) or federated ID systems. In a system using SSO, users authenticate to a single authentication service and obtain a ticket that acts as a capability, which can be given to other humans or embedded in programs to grant them access to cloud resources. These tickets are sometimes also referred to as "security tokens" in the literature, but we use the term "ticket" to distinguish them from the tokens used by humans in multifactor authentication. A variety of SSO standards are in use by CSPs: Google uses Google account credentials with the OAuth 2.0 protocol [Sun and Beznosov 2012; Hammer-Lahav et al. 2012], OpenStack CSPs use OpenStack's Keystone identity service, Microsoft uses a proprietary protocol called Storage Access Key (SAK) [Kaufman and Venkatapathy 2010], and Amazon uses AWS Identity and Access Management (IAM). Because SSOs issue capabilities and not credentials, they simultaneously provide both authentication and access control.

User creation and access control. Some CSPs allow customers to create users and assign privileges to them as described in Section 2. However, there is some disparity across CSPs with the level of support for user creation and delegation of privileges. CSPs range from having no support for user creation and access control to providing comprehensive support, including the ability to import and synchronize CSP users with a customer's LDAP or Active Directory service. Most CSPs that provide access control capabilities allow customers and users to specify file system-like policies, which allow object owners to grant users read, write, or complete access to objects. Amazon

[7]OpenStack supports third-party plugins that can provide encryption for its object storage service. See http://docs.openstack.org/security-guide/content/data-encryption.html.

[8]See http://www.joyent.com/blog/the-four-keys-of-cloud-security-confidentiality.

goes a bit further and gives object owners the ability to grant or deny access based on request time, whether the request was sent using SSL, the requester's IP address, or based on the requester's client application [Amazon AWS 2014]. CSPs that use SSO also, through that mechanism, allow users to delegate different privileges to programs by giving them tickets with different privileges. Finally, some CSPs also provide the ability for customers to create expiring temporary URLs that grant access to specific objects in object storage. These can be used to grant access to users who do not have an account or identity with the CSP. There are some indications that CSPs who do not support delegation do intend to add this capability in the future, but, at this point, support for these features remains inconsistent across the CSP industry.

3.2. CSP Security

In general, CSPs do not publicly document the measures they take to protect themselves from malicious customers except where it can impact customer functionality. In most cases, documented restrictions have to do with the type of network traffic customer VMs can send out of the cloud network. Customer VMs run customer code, but use IP addresses belonging to the CSP. A customer who sends malicious traffic can cause a CSP IP address to be banned or blacklisted. Since IPs may be rotated among CSP customers, a customer who causes an IP to be banned could adversely affect the next customer who later uses the same IP. As a result, CSPs generally check for and prevent customers from sending spoofed network traffic from VMs using CSP IP addresses. In addition, to prevent spam, Google and GoGrid also prevent outgoing connections to TCP ports associated with sending e-mail (i.e., ports 25, 465, and 587), whereas other CSPs do not explicitly state such restrictions on outgoing traffic. Amazon, for example, does not restrict traffic by destination port, and there is evidence [Ezor 2010] indicating that some Amazon IPs have appeared on spam blacklists as a result.

It is unfortunate that CSPs reveal so little about their internal security practices while heavily marketing customer-visible security features in an effort to convince customers that their data will be secure on the CSP's infrastructure with their approach following certain security standards (e.g., ISO-27000 [International Organization for Standardization 2014]). On one hand, customers do not really need to know about security mechanisms that are not visible to them, so one can argue that it is better that neither they nor anyone else know. On the other hand, the security of customer data on a CSP is entirely dependent on the security of the CSP's own infrastructure, so it may help instill greater customer confidence if CSPs are more forthcoming about their own security mechanisms. This dilemma bears some similarity to the classic "security through obscurity debate" in cryptography, but, as history has shown [Schneier 1999], better security can generally be achieved through greater transparency.

3.3. Contractual Security

Contractual security covers mechanisms and features that give customers legal or economic protections from threats. For example, location constraints may allow customers to mitigate threats from laws in foreign jurisdictions, whereas SLAs protect customers economically in the case of a CSP failure. Finally, billing integrity ensures that both CSP and customers have a fair exchange of services for payment.

Location constraints. CSPs have varying numbers and locations of data centers, and all offer the ability to constrain storage to certain data centers. Those with globally distributed cloud data centers may also allow customers to specify jurisdictional constraints, whereas those that only have operations in one jurisdiction (i.e., the United States) do not. Nevertheless, it appears that all CSPs offer locational controls for customers to the degree that they are able to given their data center locations.

SLAs. All CSPs offer SLAs on the availability of their services. If violated, the SLAs specify that the CSP will refund a portion of the fees paid by the customer to the CSP over some period of time (usually 12 months) based on the severity of the violation. Other than availability, CSP SLAs do not provide financial compensation for any other type of service failure, such as failures of performance, durability, or security. In fact, most explicitly disclaim any warranty or guarantees on their services and indemnify themselves of all damages caused by failures of their services [Popa et al. 2011].

Billing integrity. For the most part, CSPs do not document how they implement the metering of services for billing purposes [Jellinek et al. 2014]. Fortunately, current cloud billing models are, for the most part, based on verifiable service properties such as hours of usage and number of bytes transferred. Although not always trivial for the customer to duplicate the exact same method of metering used by the CSP, it is still fairly easy for the customer to measure these quantities and check that his measurements are within some delta to the amounts he is being billed for.

However, as mentioned in Section 2, hours and network traffic are often multiplied by rates that are based on difficult-to-verify properties, which can still have a large influence on the costs that customers are charged. One difficult-to-verify property that has a large effect on price is the performance level of VMs. VMs can be one of several standard VM types, which have varying numbers of CPU cores, memory, disk, and network bandwidth. Some VM types may also have specialized hardware such as GPUs attached to them. Amazon uses an abstract computational measure called an "Elastic Compute Unit" (ECU) to rate and price its VMs. However, empirical studies have shown that there can be large variances in the performance of VMs even if they have the same number of ECUs. Farley et al. [2012] show that these variances can be exploited by a malicious customer to gain up to 5% improvement for CPU-intensive workloads and 34% improvement for network-intensive workloads.

There are other areas where CSP metering may be inaccurate or non-intuitive. For example, Google states that traffic that traverses VMs external to the network interface will be billed as external even if it is directed to another customer VM on the same network in Google Cloud.[9] Thus, traffic can be billed as external even if it never leaves Google's cloud, which is not something that customers may expect.

3.4. Discussion

As shown by Table II, the public IaaS cloud industry is extremely young, with the vast majority of players running services that are less than 4 years old. However, our analysis, summarized in Table III, shows that CSPs have established best-practices for a slight majority of the security mechanisms we identified. Of the 13 mechanisms discussed, seven showed broad agreement while six showed major disparities across the surveyed CSPs.

A closer look reveals some underlying differences that may point to the reasons behind why some mechanisms have reached some level of best-practice agreement while others have not. The mechanisms where there is agreement include the use of commodity hypervisors (except Joyent), firewalls, similar data center operations, strong segregation between customer and corporate systems, using cryptography to protect information in transit, offering location constraints, and providing a performance SLA. It turns out that these mechanisms are already best practice for the hosting service and data center provider business, from which many current CSPs emerged. For example, many current hosting providers already implement industry standard data center operations, customer configurable firewalls, corporate segregation, and performance

[9]See https://cloud.google.com/compute/docs/networking.

Table III. Summary of Areas where the IaaS Industry has Established Best-Practices

CSP	Hyper-visor	Fire-wall	Data-center Op.	Dedicated VM	Segre-gation	Crypto-Transit	Crypto-Rest	Auth	Delegation/ ACL	CSP security	Location Constraints	SLA	Bill
Amazon	✓	✓	✓	✓	✓	✓	Object	2 factor passwords SSO	Role-based	No spoofing	✓	✓	✗
Verizon	✓	✓	✓	✓	✓	✓	Object	2 factor passwords, PIN	Role-based	Not mentioned	✓	✓	✗
SavvisDirect	✓	✓	✓	✗	✓	✓	Object	Passwords Keystone (SSO)	Access with tokens	Not mentioned	✓	✓	✗
Rackspace	✓	✓	✓	✗	✓	✓	Block	Passwords Keystone (SSO)	Role-based	Not mentioned	✓	✓	✗
Azure	✓	✓	✓	✗	✓	✓	Block	Passwords Keystone (SSO)	Rest-API	Some	✓	✓	✗
DimensionData	✓	✓	✓	✗	✓	✓	Object	Passwords	Role-based	Not mentioned	✓	✓	✗
Tier 3	✓	✓	✓	✗	✓	✓	Object	Passwords REST	Create users	Not mentioned	✓	✓	✗
Joyent	✗	✓	✓	✗	✓	✓	✗	Password SSH Key	Sub users	Not mentioned	✓	✓	✗
Fujitsu	✓	✓	✓	✓	✓	✓	Block	Certificate based	Sub users	Mentioned as risk	✓	✓	✗
GoGrid	✓	✓	✓	✓	✓	✓	Block	Password/ API key	✗	Prevent spam	✓	✓	✗
SoftLayer	✓	✓	✓	✗	✓	✓	Block	Password, API key	Sub users Sub accounts	Not mentioned	✓	✓	✗
HP	✓	✓	✓	✗	✓	✓	Block	Password Keystone (SSO)	Sub users Sub accounts	Not mentioned	✓	✓	✗
Google	✓	✓	✓	✗	✓	✓	Block	Passwords, OAuth2	via OAuth2	Prevent spam	✓	✓	✗

SLAs. Using cryptography to protect data transmitted over the network is standard in many industries, not just in hosting or cloud computing services. Similarly, most hosting providers, by their nature, allow customers to select the geographical location at which their machines will reside to meet latency and jurisdiction requirements. The only new component the cloud provider business requires is the use of a hypervisor to create VMs to rent to customers as opposed to physical machines. However, creating a hypervisor is a huge and difficult engineering task, so it is not surprising that all CSPs except Joyent Compute use an existing commodity hypervisor. Furthermore, these hypervisors all existed before the birth of the public IaaS business model and were already heavily in use in privately owned and run data centers.

On the other hand, the mechanisms in which there is a large amount of disagreement are components that are new and unique to the CSP business. In these cases, CSPs do not have mechanisms that they can import from the hosting business, nor do they have mature commodity software implementations they can use or purchase. For example, there are no commodity implementations of the cloud storage layer: many CSPs use home-grown solutions in which each implementation applies encryption slightly differently. Similarly, there currently exist a plethora of standards and methods for web authentication, delegation, and access control, and each CSP selects the one based on what it believes its customers desire and the amount of implementation effort involved on its part. While the use of a commodity hypervisor is common, there is no known optimal strategy for provisioning and allocating customer VMs on physical nodes, making the availability of dedicated VMs inconsistent across CSPs. Surveyed CSPs provide little information on how they protected themselves from malicious customers. Most seemed to imply that their security rests on the security of the hypervisor and good software engineering practices for the control-stack. A few CSPs explicitly mention that customers should be aware of the risk that other customers using the same hypervisor could be malicious and attack the hosts. Similarly, CSP documentation was also scant on how they ensure billing integrity, with some indications that, in a few cases, CSPs were unable or chose not to be as accurate as possible with billing.

In summary, we find that the current best practices of the public IaaS industry are all best practices that were imported from other industries. As a result, our findings indicate that the CSP industry is still very young and that CSPs have yet to converge on best practices that can address the newer, CSP-specific security threats. One trend that is likely to increase the rate of convergence is the emergence of the OpenStack project, which is currently used by a number of CSP providers to provide a complete public IaaS service (Rackspace, HP, Savvis Direct). OpenStack has the potential to provide a full-featured, open-source commodity cloud computing stack, much the same way that Xen and KVM provide commodity hypervisors. If widely adopted, this will naturally pave the way for broad standardization of security mechanisms by virtue of a large number of CSPs using the same cloud platform. Current industry indicators suggest that OpenStack will likely become the dominant cloud hosting platforms, indicating that organizations with a major development role[10] in OpenStack will have a big influence on which security mechanisms the public IaaS clouds of the future will support.

4. ACADEMIC CLOUD SECURITY PERSPECTIVE

In this section, we survey work published in academic security conferences, journals, and technical reports on IaaS cloud security. As in the previous section, we restrict ourselves to work dealing with IaaS compute, block storage, and object storage services. Like CSP industry security solutions, most academic work focuses on security

[10]http://www.openstack.org/foundation/companies/.

from the customer's perspective. However, unlike industry security solutions, academic work explores both the feasibility of attacks, as well as solutions to those attacks. In addition, academic work considers both attacks by malicious customers and malicious CSPs, whereas industry largely ignores the latter. As a result, academic research adds value by rigorously identifying the real threats to cloud customers. We categorize the academic literature into three broad areas: examinations of attacks and defenses to data confidentiality and information leakage; examination of attacks and defenses on the integrity, availability, and durability of customer data in the cloud; and techniques for securing and verifying contractual guarantees from the CSP.

4.1. Confidentiality and Information Leakage

One of the greatest concerns that customers face when deciding whether to move privately owned infrastructure to the public cloud is the threat of loss of confidentiality of their data and computations. A large body of work has emerged studying the possibility of information leakage due to the multitenancy of customer VMs on shared hardware has emerged. The leakage channels that researchers have studied include shared caches, storage channels, covert channels, and image sharing. In addition, researchers have also explored mechanisms that transform applications to prevent leakage of sensitive information to the cloud. We discuss the major work in these areas here.

Caches timing channels. First demonstrated by Bernstein in 2005, an attacker who is able to run code on the same processor as a victim can use the shared cache as a timing channel to infer information about data being used in computation by the victim [Bernstein 2005]. The attack consists of alternating "prime" and "probe" phases. In the prime phase, the attacker fills the shared cache with her data, thus evicting all the victim's data from the cache. She then lets the victim execute his code, which uses the shared cache. Loads by the victim will cause the attacker's data to be evicted from the cache. The attacker then runs the probe phase, in which she reads her data from the cache and times how long each read takes. Some accesses will take longer because they will miss in the cache and go to memory, and so the attacker can infer which cache lines the victim accessed between the prime and the probe phases. Bernstein's attack demonstrates that with enough observations of encryption operations by the victim, this channel leaks enough information to allow an attacker to recover an AES key used by the victim. The recent "FLUSH+RELOAD" technique [Yarom and Falkner 2014] uses a L3 cache side-channel attack and demonstrates that the adversary does not even need to reside on the same execution core and is still able to recover a high percentage of secret keys from a victim VM. This technique works as long as the adversary is on the same processor and thus shares the L3 cache with the victim VM.

These proof-of-concept attacks have motivated researchers to propose various methods for preventing cache timing attacks. The most straightforward is to simply remove the channel. Raj et al. [2009] propose that CSPs prevent cache leakage by placing mutually distrustful customers on different processors or by allocating memory such that there is no overlap in cache lines used by different customers. Stealthmem [Kim et al. 2012b] allocates memory so that cache lines that contain sensitive information cannot be evicted from the cache and thus do not affect the timing of the attacker's memory accesses. Because recovering information from a timing channel requires access to an accurate timing reference, another approach involves preventing attackers from gaining access to such a reference. Vattikonda et al. [2011] propose degrading the resolution of the `RTDSC` instruction used to measure the timing of events thus depriving the attacker of a way to accurately measure the timing of events. Aviram et al. [2010] propose using deterministic execution to remove timing information from executions. Varadarajan et al. [2014] find that side-channel attackers need to frequently measure

the state of the cache, so they propose a method to change the frequency of interruptions and cross-VM interactions by modifying the hypervisor scheduler. Alternatively, Azar et al. [2014] propose an even higher level solution that makes VM assignment unpredictable, thus making it harder for an adversary to co-locate her VM with that of an intended victim.

Finally, since most demonstrated attacks are in controlled conditions, researchers have raised the question as to whether cache timing attacks are even feasible in reality. One question asks how difficult it is for the attacker to have the CSP place her VM co-resident with the victim VM. Ristenpart et al. [2009] show how to map Amazon's EC2 infrastructure, identify where a victim VM is likely to reside, and then keep starting new VMs until one is placed co-resident with the victim VM. Another question is the feasibility of extracting the information from the victim VM itself. Mowery et al. [2012] find that timing attacks are difficult to mount in practice due to noise from the execution of other code that is not the code the attacker wants to probe, modern memory pre-fetching hardware, private low-level caches, architectural features such as AES-NI, and virtually tagged caches. However, Zhang et al. [2012] have recently pushed the attack even further by using a novel interprocessor-interrupt attack that crosses VM boundaries to create more opportunities for the attacker to probe and by using support vector machines—a machine learning mechanism—to eliminate the effects of noise in the channel. Their results, demonstrated against ElGamal encryption, were able to reduce the key space of the victim to roughly 10,000 keys after 6 hours of probing by the attacker. As a result, it still remains an open question for the academic community as to whether cache timing attacks are a real threat because all current methods require a long period of intense probing in which the attacker needs to remain co-located with the victim and the victim needs to continuously execute the code being probed.

Storage channels. At first glance, it may appear that simply encrypting data before sending it to cloud storage can prevent even a malicious CSP from learning the contents of the data. However, a CSP may still deduce information from the access patterns that the customer makes to the data stored on the CSP because only encrypting data content might not prevent malicious parties from probing physical locations that the customer accesses. From example, with statistical inference, up to 80% of search inquiries to an encrypted email repository can be leaked [Islam et al. 2012] by eavesdropping on access patterns.

To mitigate this, academic researchers have explored adapting the concept of Oblivious RAM (ORAM) [Goldreich and Ostrovsky 1996] to hide access patterns to cloud storage. The main goal of the ORAM algorithms is to confuse the server about the sequences of accessed locations from customers, such that access patterns like "requiring same location twice" and "requiring two adjacent locations" could not be distinguished by the server. With ORAM approaches, the customer accesses data while reshuffling and re-encrypting the data, so that, even if malicious parties obtain the information about access locations, the true access patterns are hidden. In this way, the customer can access encrypted data on an untrusted storage server while not having to reveal what data are requested or where the data are located.

The main challenge behind using ORAM is the performance overhead due to the extra storage accesses it introduces to hide the true access patterns. PrivateFS [Williams et al. 2012] improves the performance of ORAM by parallelizing accesses, while Williams and Sion [2012] improve performance by reducing the number of round trips necessary to access an object to a single round trip. ObliviStore [Stefanov and Shi 2013] goes on to improve performance by another 10× compared to PrivateFS. The system uses SSS ORAM [Stefanov et al. 2012a] to make I/O operations in ORAM both parallel and asynchronous. Path ORAM [Stefanov et al. 2013] further advances

this result using a construction that forces accesses to always be in the form of tree paths and provides an analysis of the asymptotic bounds. Write-only Oblivious RAM is another approach for increasing performance. HIVE [Blass et al. 2014] is a ORAM system showing that if only writes are hidden, the communication complexity can be lowered to $O(1)$ with poly-logarithmic memory cost. Burst ORAM [Dautrich et al. 2014] aims to reduce response times and bandwidth consumption. It is designed for bursty workloads and lowers response times in comparison with baseline ORAM algorithms.

Covert channels. Covert channels differ from previously discussed side channels in that both sender and receiver are actively trying to communicate despite not being permitted to by the CSP. One scenario for this is an attacker who compromises a victim VM and tries to covertly exfiltrate information without alerting the victim, who might be monitoring the network the VM is using, knowing. Normally, the attacker might have to use a network covert channel, but compromising a VM in the cloud gives the attacker an option of exfiltrating information across a covert channel in the cloud to another VM under the control of the attacker. This makes a cloud covert channel easier to use than a network covert channel since the latter often requires control of an intermediate router near the victim, while the former only requires placing a VM co-resident with the victim, which has already been demonstrated in other work [Ristenpart et al. 2009]. Xu et al. [2011] measure the capacity of cache-based covert channels showing that they can transmit between 2 and 10 bits per second. Wu et al. [2012] go on to show that with more advanced methods and use of the memory bus, they can achieve channels of 100 bits per second on Amazon's EC2 cloud. With hard disk contention between co-located VMs on OpenStack, CloudStack [Lipinski et al. 2014] demonstrates a 0.1 bit per second bandwidth covert channel. It is hard to make a direct comparison of this channel capacity to that of network covert channels because advanced network covert channels transmit at a rate based on the number of packets of background traffic, which is heavily dependent on link utilization. Smith and Knight [2008] report rates of about 1 bit per approximately 300 packets of background traffic. Assuming an average packet size of 1,300 bytes [Sinha et al. 2007] and a 1Gb/s link, this gives an upper bound capacity of roughly 100Kb/s, but could be far less if the link is lightly utilized. Thus, although not strictly faster than network covert channels, cloud covert channels also have the additional advantage that the capacity is mostly independent of the utilization level of the victim VM.

Image sharing leakage. One result of widespread cloud adoption is the creation of secondary markets where cloud users can create and publish VM images that other users can purchase or use for free. However, two studies have shown that this seemingly innocuous and useful development has also become a significant channel for the leakage of sensitive information. Wei et al. [2009] show that in creating the machine image, the publisher may use authentication credentials, which the publisher may then forget to remove before publishing the image. Even if the publisher does delete the credentials, VM images generally contain a disk image of the guest OS, which could still hold the deleted data because file systems generally unlink deleted files instead of actually deleting them from the disk. Other sensitive information that could be leaked in this way might include browsing history and the browser cache. In addition to leaking private information, the authors also identify image sharing as a potential attack vector for malware because the publisher could introduce malware into the image or leave a backdoor that would allow it to gain access to the image after it is deployed by another user.

Bugiel et al. [2011] go further and implement a tool called AmazonIA, which can detect some image leaks of private information as well as some backdoors. The authors

run their tool on 1,225 publicly available images on Amazon's EC2 service and find that 31% of images contain SSH backdoors. The authors could retrieve leaked host keys from 29% of the images and found leaked SSH user keys, Amazon API credentials, and SVN credentials in roughly 3% of images. This demonstrates that loss of privacy through image leakage is a real problem in practice.

Leak prevention. Since using a CSP is a potential vector for loss of confidentiality, researchers have also worked on automatically modifying applications to prevent the leakage of sensitive information to a CSP. Silverline [Puttaswamy et al. 2011] demonstrates how to automatically identify and detect sensitive data in a web application and encrypt it before sending it to the cloud. Sedic [Zhang et al. 2011] automatically splits map reduce jobs between private and public cloud nodes, keeping the jobs that handle sensitive data on the private cloud nodes and sending the jobs that don't handle sensitive data to the public cloud nodes. Both of these solutions illustrate the utility of hybrid cloud use, in which customers maintain a private cloud that they use to handle sensitive data and use the public cloud to handle non-sensitive data. Düppel [Zhang and Reiter 2013] is a system residing on a tenant VM that could, without requiring additional changes to the underlying hypervisor, detect and prevent cache-based side-channel attacks. Similarly, focusing specifically on preventing leakage of cryptographic keys, HERMES [Pattuk et al. 2014] partitions the keys into random shares, which are stored in different VMs, and uses periodic resharing to prevent partial extraction.

4.2. Integrity, Availability, and Durability

Research on the integrity, availability, and durability of customer data in the cloud can be classified into two categories: security in storage clouds and security in compute clouds. Research on the security of data in storage clouds can be further categorized as approaches using Proof of Retrieval (POR) and Provable Data Possession (PDP) and systems approaches to guarantee the integrity, availability, and durability of customer data. Research on the security of data in compute clouds involves securing the underlying hypervisor and can be classified into proposals that detect a compromised hypervisor, proposals that harden the hypervisor against compromise, and proposals that protect customer VMs against a compromised hypervisor.

POR and PDP. The goal of both POR and PDP schemes is to give the customer the ability to prove with high probability that a CSP has maintained the integrity, availability, and durability of customer data stored on the CSP. In addition, these schemes seek to prove that the stored data can also be retrieved at any time. In the degenerate case, the customer could simply query the CSP for every piece of data the customer has stored. Although this would prove possession and retrievability, it is very costly in terms of network bandwidth. Instead, both POR and PDP propose probabilistic schemes where customers make specially constructed queries on their data, which, if answered correctly by the CSP, prove with high probability that the CSP has maintained the integrity, availability, and durability of the customer data.

In a PDP scheme [Ateniese et al. 2007], the customer needs to preprocess a file that is transmitted to the server for storage. The customer only has to keep a piece of metadata that is generated from the original file at the local side. The storage server must then respond to challenges from the customer who holds the metadata to ensure its possession of the original data. The POR protocol [Juels and Kaliski Jr 2007] requires the customer to encrypt the file to be uploaded. During the encryption process, a set of blocks are randomly inserted and hidden among the encrypted file. The storage server in the POR protocol also has to respond to the verification requests from the customer that holds a cryptographic key.

The two approaches have different security guarantees in theory. With a verified response from the CSP, the PDP protocol can only indicate that the cloud is currently in possession of most of the data, but not all the data unless the CSP reads over the whole file in order to respond. In contrast, for POR, the CSP does not have to read through all parts of the encrypted data to give a guarantee that the file can be fully retrieved by the customer [Shi et al. 2013]. The main practical difference, then, between POR and PDP is in the initial casting and formulation of the problem. POR schemes require the CSP to store customer data in a redundantly encoded format using an erasure code [Juels and Kaliski Jr 2007; Shacham and Waters 2008; Bowers et al. 2009b]. Random queries are then performed over the erasure code to prove that the original data can be reconstructed with high probability.

A file system prototype called Iris [Stefanov et al. 2012b] uses POR to support efficient integrity checks while supporting a throughput of up to 260MB per second for 100 clients. PDP schemes can store the data in plain text on the CSP along with a set of customer-generated tags [Ateniese et al. 2007; Erway et al. 2009; Ateniese et al. 2008]. The customer then sends random queries that require the CSP to access the requested blocks and tags to generate the correct result.

POR and PDP only verify retrievability and possession, but do not guarantee recovery from an attack. More recently, there have been advances that combine POR and PDP with the ability to recover from a malicious CSP. HAIL [Bowers et al. 2009a] spreads data across independent CSPs such that a failure of some number of CSPs will still enable data to be retrieved from the remaining CSPs, and, similarly, MR-PDP [Curtmola et al. 2008b] stores data redundantly according to reputation. Remote Data Checking (RDC) adds forward error correction [Curtmola et al. 2008a; Ateniese et al. 2011] or network coding [Chen et al. 2010a] to enable the customer to detect and recover from arbitrary amounts of data corruption. Halevi et al. [2011] present a fix to a vulnerability in the DropBox protocol by using a POR-like scheme to allow the customer to efficiently prove that he possesses the data he is claiming ownership of. Finally, Chen and Curtmola [2013] propose an RDC approach with server-side repair.

Systems approaches to storage integrity. Systems approaches use fairly basic cryptographic techniques, such as hashes and signatures, combined with peer-to-peer techniques to enable customers to maintain the integrity, availability, and durability of their data in the face of a malicious or faulty CSP. Depot [Mahajan et al. 2010], Venus [Shraer et al. 2010], SPORC [Feldman et al. 2010], DepSky [Bessani et al. 2011], Unity [Kim et al. 2012a], and Caelus [Kim and Lie 2015] all use standard cryptographic hashes, such as SHA1, to verify that data have not been tampered with by the CSP. These systems all assume that the customer accesses the data from multiple clients, and they verify both the integrity and consistency of the retrieved data. A malicious CSP can subvert the consistency guarantees it advertises by hiding writes made by some clients from a different set of clients [Li et al. 2004] or by breaking guarantees on the ordering of writes and reads to mount a time-of-check to time-of-use attack. The proposed research detects attacks on consistency by having clients share their perceived ordering of operations with each other, thus allowing them to detect inconsistent orderings or missing operations. Depot uses a gossip protocol with direct peer-to-peer communication to do this, whereas Venus and SPORC use e-mail as a delay-tolerant network to communicate among clients. Unity has each writer pass the ordering of writes it had perceived to the next writer via a lease mechanism, thus preventing any inconsistent writes. Depot and Unity provide durability by replicating data among clients, whereas DepSky replicates data across different CSPs. SPORC and Venus to not explicitly provide durability against a malicious CSP. A different approach is taken by CloudProof [Popa et al. 2011], which instead of checking in

real-time for integrity violations, has clients and the CSP sign each operation they perform with a nonrepudiable attestation. A third party can then review these attestations to prove if either a client or the CSP has acted maliciously. Finally, Caelus [Kim and Lie 2015] assumes battery-powered clients and provides a battery-efficient, near real-time consistency violation detection for several popular consistency models.

Detecting a compromised hypervisor. The majority of approaches that try to detect a compromised hypervisor rely on a Trusted Platform Module (TPM) [The Trusted Computing Group 2013], which is a secure co-processor on the motherboard of a computer system. The TPM can be used to sign a measurement (usually a cryptographic hash) of the software running on a system at boot and make it available to a remote party to certify what software was running at boot. This signed measurement is called an *attestation* and was first proposed to be used by a CSP to prove the integrity of a hypervisor to a customer by Santos et al. [2009]. One challenge with this approach is that the customer needs to have a direct network connection with the physical machine, but CSPs often want to hide the IP addresses of the machines and the topologies of their internal networks from customers. One solution, proposed by Schiffman et al. [2010], is to have a publicly visible central verifier, which attests its integrity directly to customers and then iteratively verifies the attestations of internal nodes. Another solution, Excalibur [Santos et al. 2012], gets rid of the need to perform attestations altogether. Instead, Excalibur uses attribute-based encryption to seal keys used to encrypt VMs, so that only hypervisors whose attributes match the policy used to encrypt the VMs will be able to decrypt and run the VMs. Yang et al. also propose a fine-grained attribute-based access control mechanism [Yang et al. 2013] for use in cloud storage systems. Accountable Virtual Machines (AVM) [Haeberlen et al. 2010] use hash chains to record actions performed by a VM in a non-repudiable log to detect tampering by a malicious hypervisor. Although AVM does not rely directly on TPMs, it does need a way to attest to the customer that the CSP is running a hypervisor that supports AVM—a requirement that the authors state could be fulfilled with an attestation provided by a TPM. HyperSentry [Azab et al. 2010] does not use a TPM at all. Instead, it uses Intel's System Management Mode (SMM) to stealthily and repeatedly measure the integrity of the hypervisor to detect if the hypervisor has been compromised.

Hardening the hypervisor. Since the hypervisor is the component that enforces isolation between customer VMs, a compromise of the hypervisor by one VM would allow it to escalate privileges and attack other VMs. Thus, approaches to making the hypervisor harder to attack either try to reduce the attack surface of the hypervisor or try to reduce the size of the Trusted Computing Base (TCB) of the hypervisor. NoHype [Keller et al. 2010] reduces the attack surface by only invoking the hypervisor during VM creation. Afterward, NoHype has the VM run on bare hardware, thus minimizing the interaction between the hypervisor and VMs. Murray et al. [2008], self service computing [Butt et al. 2012] and Xoar [Colp et al. 2011] minimize the TCB of Xen by disaggregating its shared management VM (Dom0) into customer VM-specific parts. Murray et al. take a first step toward disaggregation by extracting the domain builder, which is responsible for starting VMs, from Dom0. Self-service computing splits Dom0 into a system-wide Dom0 and a VM-specific Dom0, thus allowing customers access to the VM-specific Dom0 to access hypervisor functionality without breaking the isolation between VMs. Xoar goes further and splits the Dom0 into individual "service VMs" for each customer VM. Unlike customer VMs, service VMs are not under the control of customers, but still benefit from the isolation that being in a separate VM confers. Xoar minimizes the shared TCB between VMs to just the hypervisor, the XenStore service, and shared device drivers. Xoar also mitigates the effects of potential compromises by periodically micro-rebooting service VMs back to a known good state. Finally, Wu et al.

[2013] reduce the TCB of KVM by refactoring code in the KVM kernel module into Linux user space.

Protecting customers against a compromised hypervisor. As a last line of defense, if the hypervisor is compromised, one may try to retain the integrity and confidentiality of customer VMs despite a compromised hypervisor. Cryptographic methods such as verifiable computation [Vu et al. 2013; Parno et al. 2013] and fully homomorphic encryption [Gentry 2009] provide very strong guarantees of integrity and confidentiality but have very high overhead. More practical systems approaches can incur less overhead but may require specialized hardware. CloudVisor [Zhang et al. 2011a] nests a small hypervisor beneath a commodity hypervisor. CloudVisor intercepts and encrypts the memory pages of customer VMs if they are accessed by the hypervisor, thus protecting the memory contents of VMs if the commodity hypervisor is compromised. However, the interception and cryptographic operations result in an overhead of 4.5–54.5% for I/O intensive applications. Hyperwall [Szefer and Lee 2012] avoids the cost of cryptography by using specialized processor extensions to isolate customer VMs from the hypervisor. However, these extensions require specialized features that are not available on commodity processors. To be secure, both systems will need to be combined with remote attestation to prove to the customer that the CSP is using them to protect customer VMs.

4.3. Contractual Security

Cloud customers pay the CSP for a certain level and type of service. For example, the CSP may promise a certain level of performance or a certain amount of resources. The CSP may also make promises about the type of service, such as the location where the data are stored or constraints on how the data are stored. When the customer gets less than what they paid for, or the cloud provider is tricked into giving up more than what was promised, the security of the contract between them is compromised. To protect customers, researchers have explored how to audit CSPs to ensure that contracts are fulfilled.

As discussed in Section 3, CSPs currently offer the ability for clients to specify the location of their data, whether the CSP should encrypt data at rest, and whether VMs should run on a dedicated hardware node or not. The challenge is that these are all *difficult-to-verify* properties of the CSP service. For example, how does a customer actually check that their data are stored in the location that the CSP has promised it to be stored? Gill et al. [2010] show that a malicious cloud provider can subvert even the most advanced geo-location techniques to fool a customer into thinking their data and VMs are in one location when they are actually in another. Benson et al. [2011] use geo-location in combination with POR to audit whether a CSP is storing data at an agreed-upon location. Similarly, LoSt [Watson et al. 2012] uses POR in combination with ping latencies to audit the location of data stored by a CSP. To audit cloud-side encryption at rest, Van Dijk et al. [2012] employ an hourglass scheme that prevents the CSP from responding to specially constructed queries by the client in time unless the CSP has the corresponding cipher text already computed for the client's data. To ensure that a VM is running on dedicated hardware, Zhang et al. [2011b] use the same techniques that attackers use to recover data leaked through cache timing channels to detect the presence of another VM. Juels and Oprea [2013] propose a general framework to combine with POR to perform remote auditing. Bowers et al. [2011] show how a customer can remotely audit a CSP to check if his data has been stored on multiple drives by measuring the speed of CSP responses to specially constructed queries. Finally, Chen and Chen [2014] propose BitBill as a decentralized system that can verify billable events in the cloud using a Bitcoin-like mechanism [Nakamoto 2008].

Table IV. Summary of Comparison between Academic and Industry Cloud Security Solutions

Problem	Current Solutions and Results	Summary
Malicious or compromised CSPs	**Academia:** POR/PDP, hypervisor integrity, auditing **Industry:** Marketing	CSPs assume they are trusted and disclaim liability for damages due to security compromises. Academia serves an important role for identifying weaknesses in CSP security that would be hard for CSPs themselves to talk about publicly.
Cross-VM leakage, cache timing channels, covert channels	**Academia:** Evaluation of feasibility of attack, memory placement, deterministic execution **Industry:** Dedicated instances	Industry proposes dedicated instances. Main question is whether the benefits of dedicated instances justify their cost. Academia could help by establishing the feasibility of such attacks and the associated costs with mounting the attack.
Malicious VM images and leakage through VM images	**Academia:** Demonstrated the size and scope of the problem through measurement. **Industry:** Provide documentation and warnings to users.	A simple tool to detect leakage and malicious VMs has been constructed in academia. There is potential for a more comprehensive solution to solve this problem.
Hypervisor integrity and compromised hypervisors	**Academia:** Hypervisor hardening, using untrusted hypervisors, TPM-based remote attestation by customers. **Industry:** Hypervisor customizations and keeping the hypervisor patched.	Direct TPM-based remote attestation by customers is of limited use due to industry use of customized hypervisors. Protecting against an untrusted hypervisor is the only viable solution against a malicious CSP.
Billing integrity	**Academia:** Demonstrate attacks against billing integrity, auditing techniques to ensure that difficult-to-verify security properties of cloud service are upheld. **Industry:** Bill based on easily verifiable quantities.	Unfortunately, industry billing practice multiplies easily verifiable quantities with rates based on difficult-to-verify properties. There is potential for research into better auditing and metering techniques.

4.4. Discussion

In this section, we reconcile academic research work with CSPs offerings in the IaaS industry. We briefly summarize our findings in Table IV and give details here.

In comparing academia and industry, the most striking difference is that academic work considers attacks from both malicious customers and malicious CSPs, whereas industry mostly ignores the possibility of the latter. For the most part, CSPs assume trust from the customer, even though CSP terms of service generally say that although they are responsible for maintaining the security of customer data, CSPs are explicitly released from liability for damage due to compromises of their terms of service. Because CSPs need to project an image of trustworthiness, academia provides an important counterpoint for identifying weaknesses in CSP security that would be hard for CSPs themselves to publicly advertise. In the case of threats from malicious customers, industry and academia have identified some of the same threats but have come up with different solutions. For example, there has been a great deal of academic work on the leakage of confidential information across VMs via side channels and covert channels. Academics have proposed some novel and sophisticated solutions. However, most CSPs assert that well-engineered hypervisors should be sufficient to isolate mutually distrustful customers. Some CSPs take further measures to convince customers that they are protected from malicious customers, but these measures are still technically simple when compared to those in academia. Joyent's SmartOS

hypervisor offers two independent isolation mechanisms—a SmartOS Zone (similar to a BSD jail), as well as the isolation provided by QEMU-KVM—which it markets as a differentiator from other CSPs. Amazon, Fujitsu, and SoftLayer offer customers dedicated VMs, which for a premium in cost, are provisioned on hardware that is not shared with any other customer. It would appear that industry has independently developed several other solutions to the problem of cross-VM leakage but is undecided on how much the solution is worth. This suggests that research efforts might be best directed at trying to establish the practicality of attacks that exploit cross-VM leakage and the related costs associated with mounting those attacks. On one hand, if such attacks turn out to be relatively cheap and easy to mount, then all CSPs will have to offer dedicated VMs as an option, and some may even offer it exclusively. This wide availability of dedicated VMs should have the effect of driving down the premium that a dedicated VM can fetch. On the other hand, if such attacks turn out to be very expensive or impractical to mount, then dedicated VMs might only be used by niche customers with extremely valuable information, thus justifying a higher premium.

One way the academic literature can have an influence over industry practice is to demonstrate the feasibility of certain attacks. For example, one attack that academia has shown to be practical and occurring in the wild is the threat of malicious VM images, as well as the threat of confidential information leakage via shared VM images. These threats have been confirmed both by measurement performed by academics on images in Amazon's Marketplace and by Amazon itself, which acknowledges them in its documentation.[11] Another example is the aforementioned leakage of sensitive information over various side channels that exist between physically co-located VMs, which has partially resulted in a number of CSPs offering dedicated VM instances for a small premium.

There exist a number of academic proposals that rely on TPM-based remote attestation to allow customers to verify the integrity of the hypervisor that a CSP is running. In theory, such a defense can protect against both a malicious CSP that uses a malicious hypervisor, as well as a malicious customer who tampers with the hypervisor binary without the CSP knowing. From Section 3, we have seen that some CSPs use customized hypervisors for their public cloud. As a result, the known-good hash for any hypervisor would also have to come from the CSP itself, meaning that TPM-based measurements are ineffective for detecting a malicious CSP (unless some trusted third party is relied upon to provide known-good hashes). Furthermore, if the CSP is trusted, then it is not clear what value allowing customers to directly attest the hypervisors has over having the CSPs perform the integrity checks internally. Thus, remote attestation via TPMs appears to have limited use for verifying hypervisor integrity in public clouds. If customers are unwilling to trust the CSP, then the only workable solutions involve guaranteeing integrity even if the hypervisor is malicious. Recent developments, such as Intel's SGX extensions [Mckeen et al. 2013; Anati et al. 2013; Hoekstra et al. 2013] are used in systems like Haven [Baumann et al. 2014] to protect applications from a malicious operating system. In addition, trusted execution systems like Flicker [McCune et al. 2008] might serve as building blocks for such solutions.

Section 3 also documents cases where industry may not always be able to bill customers accurately, or where charges may be levied unexpectedly. This raises the question as to whether there could be other cases where it is difficult or impractical to accurately meter customer usage. Such inaccuracies present opportunities for customers to game cloud providers or for cloud providers to overcharge customers (either knowingly or unknowingly). A key question is whether such inaccuracies are widespread or whether these inaccuracies are nothing but rare exceptions. Although academics

[11]See http://aws.amazon.com/articles/530.

have identified such inaccuracies [Farley et al. 2012] and even shown how they can be exploited to steal resources from victim VMs [Varadarajan et al. 2012], there has not been much other academic or industrial research on this matter. As a result, we feel there is potential for exciting research into better cloud auditing and metering techniques.

Going forward, an interesting question is how academic research may interact with industrial IaaS cloud deployments. So far, the main interaction has been academia identifying and demonstrating the feasibility of threats and industry reacting to provide solutions to a number of those demonstrated threats. However, there has been little tangible transfer of solutions explored by academia to industrial IaaS cloud deployments. To achieve better cooperation, one might assume that problems should also be identified and characterized by industry, so that more guidance and data can be provided to academics looking to work on relevant problems in cloud security.

Unfortunately, with fierce competition between cloud providers, it is likely that many CSPs see it as a competitive disadvantage to publicize any security threats they may be the targets of because they may fear scaring customers away from them into the hands of their competitors. In many ways, analogies can be made between them and other security-sensitive industries like banking or insurance. However, in mature industries such as these, rather than individual companies voicing concerns, a trade association exists to speak on behalf of all industry members. For example, in the United States, banks are represented by the American Banker's Association, and insurance companies are represented by American Insurance Association. Such entities allow public statements to be made about the state of an industry without the statements being attributable to any one member. We believe that, as the IaaS industry matures, it is likely that such an association will begin to fill this role for IaaS providers. For example, the Cloud Security Alliance[12] is a potentially strong candidate to fill this role and already boasts a membership that includes a number of the major cloud providers surveyed in this paper.

5. CONCLUSION

We have surveyed security mechanisms developed by industry and academia for public IaaS clouds. We find that the security mechanisms that have been standardized across the IaaS industry are mostly comprised of well-known security mechanisms such as firewalls, physical security, corporate segregation for customer data, and network encryption that have been imported from the hosting industry. On the other hand, at this point, there exists an array of offerings for security mechanisms for new threats that are unique to IaaS clouds, such as dedicated VMs for cross-VM leakage, cloud-side encryption for confidentiality, and authentication and authorization methods, thus indicating the potential for future research and innovation. Industry and academia also appear to diverge in a couple of key areas, such as the utility of TPM-based remote attestation by customers to verify hypervisor integrity and the need for sophisticated solutions to cross-VM information leakage. There are also acknowledged problems for which there are currently no solutions in industry or academia, such as identifying and dealing with malicious VM images.

Ultimately, we believe that two trends going forward will drive research in public IaaS cloud security and the relationship between academia and companies in this industry. The first is that there is likely going to be greater standardization as the commodity implementations of cloud control stacks such as OpenStack emerge. Like previous software components, such as operating systems and hypervisors, once one or a few implementations reach critical mass, their widespread use in and of itself causes

[12]https://cloudsecurityalliance.org/.

standardization. The second is the emergence of one or several trade associations that will represent the public IaaS cloud industry. It will be easier and less risky for such associations to speak about security concerns in IaaS clouds when statements made by an industry association are not attributable to any one member. Such associations can become a source of industry wide data on security threats and issues for academic researchers.

ACKNOWLEDGMENTS

The authors would like to thank Zhen Huang, Michelle Wong, Mariana D'Angelo, and Dhaval Miyani for the feedback and the anonymous reviewers for their invaluable suggestions.

REFERENCES

Amazon AWS. 2013. Amazon Web Services Risk and Compliance. https://media.amazonwebservices.com/AWS_Risk_and_Compliance_Whitepaper.pdf, Last accessed: June 2015.

Amazon AWS. 2014. Amazon Web Services Overview of Security Processes. https://media.amazonwebservices.com/pdf/AWS_Security_Whitepaper.pdf, Last accessed: June 2015.

Ittai Anati, Shay Gueron, Simon Johnson, and Vincent Scarlata. 2013. Innovative technology for CPU based attestation and sealing. In *Proceedings of the Workshop on Hardware and Architectural Support for Security and Privacy*.

Michael Armbrust, Armando Fox, Rean Griffith, Anthony Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. 2010. A view of cloud computing. *Communications of the ACM* 53, 4, 50–58.

Giuseppe Ateniese, Randal Burns, Reza Curtmola, Joseph Herring, Osama Khan, Lea Kissner, Zachary Peterson, and Dawn Song. 2011. Remote data checking using provable data possession. *ACM Transactions on Information and System Security (TISSEC)* 14, 1, 12.

Giuseppe Ateniese, Randal Burns, Reza Curtmola, Joseph Herring, Lea Kissner, Zachary Peterson, and Dawn Song. 2007. Provable data possession at untrusted stores. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*. 598–609.

Giuseppe Ateniese, Roberto Di Pietro, Luigi V. Mancini, and Gene Tsudik. 2008. Scalable and efficient provable data possession. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Networks (SECURECOMM'08)*.

Amittai Aviram, Sen Hu, Bryan Ford, and Ramakrishna Gummadi. 2010. Determinating timing channels in compute clouds. In *Proceedings of the 2010 ACM Workshop on Cloud Computing Security (CCSW'10)*. 103–108.

Ahmed M. Azab, Peng Ning, Zhi Wang, Xuxian Jiang, Xiaolan Zhang, and Nathan C. Skalsky. 2010. HyperSentry: Enabling stealthy in-context measurement of hypervisor integrity. In *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS'10)*. 38–49.

Yossi Azar, Seny Kamara, Ishai Menache, Mariana Raykova, and Bruce Shepard. 2014. Co-location-resistant clouds. In *Proceedings of the 2014 ACM Workshop on Cloud Computing Security (CCSW'14)*. 9–20.

Adam Barker, Blesson Varghese, Jonathan Stuart Ward, and Ian Sommerville. 2014. Academic cloud computing research: Five pitfalls and five opportunities. In *Proceedings of the 6th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud'14)*.

Andrew Baumann, Marcus Peinado, and Galen Hunt. 2014. Shielding applications from an untrusted cloud with Haven. In *Proceedings of the 11th Symposium on Operating Systems Design and Implementation (OSDI'14)*. 267–283.

Karyn Benson, Rafael Dowsley, and Hovav Shacham. 2011. Do you know where your cloud files are? In *Proceedings of the 2011 ACM Workshop on Cloud Computing Security (CCSW'11)*. 73–82.

Daniel Bernstein. 2005. Cache-timing attacks on AES. http://cr.yp.to/antiforgery/cachetiming-20050414.pdf, Last accessed: April 2015.

Alysson Neves Bessani, Miguel P. Correia, Bruno Quaresma, Fernando André, and Paulo Sousa. 2011. DepSky: Dependable and secure storage in a cloud-of-clouds. In *Proceedings of the 2011 European Conference on Computer Systems (EuroSys'11)*. 31–46.

Erik-Oliver Blass, Travis Mayberry, Guevara Noubir, and Kaan Onarlioglu. 2014. Toward robust hidden volumes using write-only oblivious RAM. In *Proceedings of the 21th ACM Conference on Computer and Communications Security (CCS'14)*. 203–214.

Kevin Bowers, Marten van Dijk, Ari Juels, Alina Oprea, and Ronald Rivest. 2011. How to tell if your cloud files are vulnerable to drive crashes. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*. 501–514.

Kevin D. Bowers, Ari Juels, and Alina Oprea. 2009a. HAIL: A high-availability and integrity layer for cloud storage. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09)*. 187–198.

Kevin D. Bowers, Ari Juels, and Alina Oprea. 2009b. Proofs of retrievability: Theory and implementation. In *Proceedings of the 2009 ACM Workshop on Cloud Computing Security (CCSW'09)*. 43–54.

Sven Bugiel, Stefan Nürnberger, Thomas Pöppelmann, Ahmad-Reza Sadeghi, and Thomas Schneider. 2011. AmazonIA: When elasticity snaps back. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*. 389–400.

Shakeel Butt, H. Andrés Lagar-Cavilla, Abhinav Srivastava, and Vinod Ganapathy. 2012. Self-service cloud computing. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*. 253–264.

Bo Chen and Reza Curtmola. 2013. Towards self-repairing replication-based storage systems using untrusted clouds. In *Proceedings of the Third ACM Conference on Data and Application Security and Privacy*. 377–388.

Bo Chen, Reza Curtmola, Giuseppe Ateniese, and Randal Burns. 2010a. Remote data checking for network coding-based distributed storage systems. In *Proceedings of the 2010 ACM Workshop on Cloud Computing Security (CCSW'10)*. 31–42.

Li Chen and Kai Chen. 2014. BitBill: Scalable, robust, verifiable peer-to-peer billing for cloud computing. In *Proceedings of the USENIX Workshop on Hot Topics in Cloud Computing*. 20.

Yanpei Chen, Vern Paxson, and Randy Katz. 2010b. *What's New about Cloud Computing Security*. Technical Report UCB/EECS-2010-5. Dept. Electrical Eng. and Comput. Sciences, University of California.

Cloud Security Alliance. 2011. Security guidance for critical areas of focus in cloud computing v3.0. https://cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf, Last accessed: June 2015.

Patrick Colp, Mihir Nanavati, Jun Zhu, William Aiello, George Coker, Tim Deegan, Peter Loscocco, and Andrew Warfield. 2011. Breaking up is hard to do: Security and functionality in a commodity hypervisor. In *Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP'11)*. 189–202.

Reza Curtmola, Osama Khan, and Randal Burns. 2008a. Robust remote data checking. In *Proceedings of the 4th ACM International Workshop on Storage Security and Survivability*. 63–68.

Reza Curtmola, Osama Khan, Randal Burns, and Giuseppe Ateniese. 2008b. MR-PDP: Multiple-replica provable data possession. In *Proceedings of the 28th International Conference on Distributed Computing Systems*. 411–420.

Jonathan Dautrich, Emil Stefanov, and Elaine Shi. 2014. Burst ORAM: Minimizing ORAM response times for bursty access patterns. In *Proceedings of the 23rd USENIX Security Symposium*. 749–764.

Chris Erway, Alptekin Küpçü, Charalampos Papamanthou, and Roberto Tamassia. 2009. Dynamic provable data possession. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09)*. 213–222.

Jonathan I. Ezor. 2010. Busting Blocks: Revisiting 47 USC Sec. 230 to address the lack of effective legal recourse for wrongful inclusion in spam filters. *Richmond Journal of Law and Technology* 17, 1.

Benjamin Farley, Ari Juels, Venkatanathan Varadarajan, Thomas Ristenpart, Kevin Bowers, and Michael Swift. 2012. More for your money: Exploiting performance heterogeneity in public clouds. In *Proceedings of the 3rd ACM Symposium on Cloud Computing*. 20:1–20:14.

Ariel J. Feldman, William P. Zeller, Michael J. Freedman, and Edward W. Felten. 2010. SPORC: Group collaboration using untrusted cloud resources. In *Proceedings of the 9th Symposium on Operating Systems Design and Implementation (OSDI'10)*.

Gartner. 2013. Magic Quadrant for Cloud Infrastructure as a Service. Retrieved form http://www.gartner.com/technology/reprints.do?id=1-1IMDMZ5&ct=130819&st=sb, Last accessed: June 2015.

Craig Gentry. 2009. *A Fully Homomorphic Encryption Scheme*. Ph.D. Dissertation. Stanford University.

Phillipa Gill, Yashar Ganjali, Bernard Wong, and David Lie. 2010. Dude, where's that IP?: Circumventing measurement-based IP geolocation. In *Proceedings of the 19th USENIX Security Symposium*. 16–32.

Robert P. Goldberg. 1974. Survey of virtual machine research. *IEEE Computer Magazine* 7, 6 (June 1974), 35–45.

Oded Goldreich and Rafail Ostrovsky. 1996. Software protection and simulation on oblivious RAMs. *Journal of the ACM* 43, 3 (May 1996), 431–473.

Google. 2012. Google's approach to IT security: A Google white paper. https://static.googleusercontent.com/media/www.google.com/en/US/work/pdf/whygoogle/google-common-security-whitepaper.pdf.

Andreas Haeberlen, Paarijaat Aditya, Rodrigo Rodrigues, and Peter Druschel. 2010. Accountable virtual machines. In *Proceedings of the 9th Symposium on Operating Systems Design and Implementation (OSDI'10)*. 119–134.

Shai Halevi, Danny Harnik, Benny Pinkas, and Alexandra Shulman-Peleg. 2011. Proofs of ownership in remote storage systems. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*. 491–500.

Eran Hammer-Lahav, David Recordon, and Dick Hardt. 2012. The OAuth 2.0 authorization protocol. IETF Q51214 Draft v2.22. https://tools.ietf.org/html/draft-ietf-oauth-v2-22, Last accessed: June 2015.

Matthew Hoekstra, Reshma Lal, Pradeep Pappachan, Carlos Rozas, Vinay Phegade, and Juan del Cuvillo. 2013. Using innovative instructions to create trustworthy software solutions. In *Proceedings of the Workshop on Hardware and Architectural Support for Security and Privacy*. 11:1–11:1.

International Organization for Standardization. 2014. Information security management systems. ISO/IEC 27000:2014.

Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. 2012. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *Proceedings of the Network and Distributed System Security Symposium (NDSS'12)*.

Robert Jellinek, Yan Zhai, Thomas Ristenpart, and Michael Swift. 2014. A day late and a dollar short: The case for research on cloud billing systems. In *The USENIX Workshop on Hot Topics in Cloud Computing*. 21.

Ari Juels and Burton S. Kaliski Jr. 2007. PORs: Proofs of retrievability for large files. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*. 584–597.

Ari Juels and Alina Oprea. 2013. New approaches to security and availability for cloud data. *Communications of the ACM* 56, 2 (Feb. 2013), 64–73.

Charlie Kaufman and Ramanathan Venkatapathy. 2010. Windows Azure™ security overview, version 1.01. http://go.microsoft.com/?linkid=9740388, Last accessed: June 2015.

Eric Keller, Jakub Szefer, Jennifer Rexford, and Ruby B. Lee. 2010. NoHype: Virtualized cloud infrastructure without the virtualization. In *Proceedings of the 37th International Symposium on Computer Architecture (ISCA'10)*. 350–361.

Beom Heyn Kim, Wei Huang, and David Lie. 2012a. Unity: Secure and durable personal cloud storage. In *Proceedings of the 2012 ACM Workshop on Cloud Computing Security (CCSW'12)*. 31–36.

Beom Heyn Kim and David Lie. 2015. Caelus: Verifying the consistency of cloud services with battery-powered devices. In *Proceedings of the 2015 IEEE Symposium on Security and Privacy*.

Taesoo Kim, Marcus Peinado, and Gloria Mainar-Ruiz. 2012b. STEALTHMEM: System-level protection against cache-based side channel attacks in the cloud. In *Proceedings of the 21st USENIX Security Symposium*. 11.

Ang Li, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. 2010. CloudCmp: Comparing public cloud providers. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC'10)*. 1–14.

Jinyuan Li, Maxwell Krohn, David Mazières, and Dennis Shasha. 2004. Secure untrusted data repository (SUNDR). In *Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI'04)*.

Bartosz Lipinski, Wojciech Mazurczyk, and Krzysztof Szczypiorski. 2014. Improving hard disk contention-based covert channel in cloud computing. In *Proceedings of the 2014 IEEE Security and Privacy Workshops*. 100–107.

Prince Mahajan, Srinath T. V. Setty, Sangmin Lee, Allen Clement, Lorenzo Alvisi, Michael Dahlin, and Michael Walfish. 2010. Depot: Cloud storage with minimal trust. In *Proceedings of the 9th Symposium on Operating Systems Design and Implementation (OSDI'10)*. 1–12.

Jonathan M. McCune, Bryan J. Parno, Adrian Perrig, Michael K. Reiter, and Hiroshi Isozaki. 2008. Flicker: An execution infrastructure for TCB minimization. In *Proceedings of the 3rd European Conference on Computer Systems (EuroSys'08)*. 315–328.

Frank Mckeen, Ilya Alexandrovich, Alex Berenzon, Carlos Rozas, Hisham Shafi, Vedvyas Shanbhogue, and Uday Savagaonkar. 2013. Innovative instructions and software model for isolated execution. In *Proceedings of the Workshop on Hardware and Architectural Support for Security and Privacy*. 10:1–10:1.

Keaton Mowery, Sriram Keelveedhi, and Hovav Shacham. 2012. Are AES x86 cache timing attacks still feasible? In *Proceedings of the 2012 ACM Workshop on Cloud Computing Security (CCSW)*. 19–24.

Derek Gordon Murray, Grzegorz Milos, and Steven Hand. 2008. Improving Xen security through disaggregation. In *Proceedings of the 4th International Conference on Virtual Execution Environments (VEE'08)*. 151–160.

Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. http://bitcoin.org/bitcoin.pdf, Last accessed: June 2015.

Bryan Parno, Craig Gentry, Jon Howell, and Mariana Raykova. 2013. Pinocchio: Nearly practical verifiable computation. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*. 238–252.

Erman Pattuk, Murat Kantarcioglu, Zhiqiang Lin, and Huseyin Ulusoy. 2014. Preventing cryptographic key leakage in cloud virtual machines. In *Proceedings of the 23rd USENIX Security Symposium*. 703–718.

Raluca Ada Popa, Jacob R. Lorch, David Molnar, Helen J. Wang, and Li Zhuang. 2011. Enabling security in cloud storage SLAs with CloudProof. In *Proceedings of the 2011 Annual Usenix Technical Conference*. 355–368.

Krishna P. N. Puttaswamy, Christopher Kruegel, and Ben Y. Zhao. 2011. Silverline: Toward data confidentiality in storage-intensive cloud applications. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*. 10:1–10:13.

Himanshu Raj, Ripal Nathuji, Abhishek Singh, and Paul England. 2009. Resource management for isolation enhanced cloud services. In *Proceedings of the 2009 ACM Workshop on Cloud Computing Security (CCSW'09)*. 77–84.

Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. 2009. Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09)*. 199–212.

Nuno Santos, Krishna P. Gummadi, and Rodrigo Rodrigues. 2009. Towards trusted cloud computing. In *Proceedings of the 1st USENIX Workshop on Hot Topics in Cloud Computing (HotCloud'09)*.

Nuno Santos, Rodrigo Rodrigues, Krishna P. Gummadi, and Stefan Saroiu. 2012. Policy-sealed data: A new abstraction for building trusted cloud services. In *Proceedings of the 21st USENIX Security Symposium*.

Joshua Schiffman, Thomas Moyer, Hayawardh Vijayakumar, Trent Jaeger, and Patrick McDaniel. 2010. Seeding clouds with trust anchors. In *Proceedings of the 2012 ACM Workshop on Cloud Computing Security (CCSW)*. 43–46.

Bruce Schneier. 1999. DVD Encryption Broken. Retrieved from https://www.schneier.com/essay-193.html, Last accessed date: June 2015.

Hovav Shacham and Brent Waters. 2008. Compact proofs of retrievability. In *Advances in Cryptology-ASIACRYPT 2008*. Springer, 90–107.

Elaine Shi, Emil Stefanov, and Charalampos Papamanthou. 2013. Practical dynamic proofs of retrievability. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS'13)*. 325–336.

Alexander Shraer, Christian Cachin, Asaf Cidon, Idit Keidar, Yan Michalevsky, and Dani Shaket. 2010. Venus: Verification for untrusted cloud storage. In *Proceedings of the 2010 ACM Workshop on Cloud Computing Security (CCSW'10)*. 19–30.

Rishi Sinha, Christos Papadopoulos, and John Heidemann. 2007. *Internet Packet Size Distributions: Some Observations*. Technical Report ISI-TR-2007-643. USC/Information Sciences Institute.

Ronald Smith and G. Scott Knight. 2008. Predictable design of network-based covert communication systems. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*. 311–321.

Emil Stefanov and Elaine Shi. 2013. ObliviStore: High performance oblivious cloud storage. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*. 253–267.

Emil Stefanov, Elaine Shi, and Dawn Song. 2012a. Towards practical oblivious RAM. In *Proceedings of the 19th Symposium on Network and Distributed System Security (NDSS)*.

Emil Stefanov, Marten van Dijk, Ari Juels, and Alina Oprea. 2012b. Iris: A scalable cloud file system with efficient integrity checks. In *Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC'12)*. 229–238.

Emil Stefanov, Marten van Dijk, Elaine Shi, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. 2013. Path ORAM: An extremely simple oblivious RAM protocol. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS'13)*. 299–310.

San-Tsai Sun and Konstantin Beznosov. 2012. The devil is in the (implementation) details: An empirical analysis of oauth SSO systems. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*. 378–390.

Jakub Szefer and Ruby Lee. 2012. Architectural support for hypervisor-secure virtualization. In *Proceedings of the 17th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'12)*. 437–450.

Hassan Takabi, James B. D. Joshi, and Gail-Joon Ahn. 2010. Security and privacy challenges in cloud computing environments. *IEEE Security & Privacy* 8, 6, 24–31.

Douglas Terry, Vijayan Prabhakaran, Ramakrishna Kotla, Mahesh Balakrishnan, Marcos Aguilera, and Hussam Abu-Libdeh. 2013. Consistency-based service level agreements for cloud storage. In *Proceedings of the 24rd ACM Symposium on Operating Systems Principles (SOSP'13)*. 309–324.

The Trusted Computing Group. 2013. Homepage. Retrieved from https://www.trustedcomputinggroup.org, Last accessed: June 2015.

Marten van Dijk, Ari Juels, Alina Oprea, Ronald L. Rivest, Emil Stefanov, and Nikos Triandopoulos. 2012. Hourglass schemes: How to prove that cloud files are encrypted. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*. 265–280.

Venkatanathan Varadarajan, Thawan Kooburat, Benjamin Farley, Thomas Ristenpart, and Michael M. Swift. 2012. Resource-freeing attacks: Improve your cloud performance (at your neighbor's expense). In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*. 281–292.

Venkatanathan Varadarajan, Thomas Ristenpart, and Michael Swift. 2014. Scheduler-based defenses against cross-VM side-channels. In *Proceedings of the 23rd USENIX Security Symposium*. 687–702.

Bhanu C. Vattikonda, Sambit Das, and Hovav Shacham. 2011. Eliminating fine grained timers in Xen. In *Proceedings of the 2011 ACM Workshop on Cloud Computing Security (CCSW'11)*. 41–46.

Victor Vu, Srinath Setty, Andrew Blumberg, and Michael Walfish. 2013. A hybrid architecture for interactive verifiable computation. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*. 223–237.

Gaven J. Watson, Reihaneh Safavi-Naini, Mohsen Alimomeni, Michael E. Locasto, and Shivaramakrishnan Narayan. 2012. LoSt: Location based storage. In *Proceedings of the 2012 ACM Workshop on Cloud Computing Security (CCSW)*. 59–70.

Jinpeng Wei, Xiaolan Zhang, Glenn Ammons, Vasanth Bala, and Peng Ning. 2009. Managing security of virtual machine images in a cloud environment. In *Proceedings of the 2009 ACM Workshop on Cloud Computing Security (CCSW'09)*. ACM, 91–96.

Peter Williams and Radu Sion. 2012. Single round access privacy on outsourced storage. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*. 293–304.

Peter Williams, Radu Sion, and Alin Tomescu. 2012. PrivateFS: A parallel oblivious file system. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*. 977–988.

Chiachih Wu, Zhi Wang, and Xuxian Jiang. 2013. Taming hosted hypervisors with (mostly) deprivileged execution. In *Proceedings of the 20th Symposium on Network and Distributed System Security (NDSS'13)*.

Zhenyu Wu, Zhang Xu, and Haining Wang. 2012. Whispers in the hyper-space: High-speed covert channel attacks in the cloud. In *Proceedings of the 21st USENIX Security Symposium*.

Yunjing Xu, Michael Bailey, Farnam Jahanian, Kaustubh Joshi, Matti Hiltunen, and Richard Schlichting. 2011. An exploration of L2 cache covert channels in virtualized environments. In *Proceedings of the 2011 ACM Workshop on Cloud Computing Security (CCSW'11)*. 29–40.

Kan Yang, Xiaohua Jia, and Kui Ren. 2013. Attribute-based fine-grained access control with efficient revocation in cloud storage systems. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*. 523–528.

Yuval Yarom and Katrina Falkner. 2014. FLUSH+RELOAD: A high resolution, low noise, L3 cache side-channel attack. In *Proceedings of the 23rd USENIX Security Symposium*. 719–732.

Fengzhe Zhang, Jin Chen, Haibo Chen, and Binyu Zang. 2011a. CloudVisor: Retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization. In *Proceedings of the 23rd ACM Symposium on Operating Systems Principles (SOSP'11)*. 203–216.

Kehuan Zhang, Xiaoyong Zhou, Yangyi Chen, XiaoFeng Wang, and Yaoping Ruan. 2011. Sedic: Privacy-aware data intensive computing on hybrid clouds. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*. 515–526.

Yinqian Zhang, Ari Juels, Alina Oprea, and Michael K. Reiter. 2011b. HomeAlone: Co-residency detection in the cloud via side-channel analysis. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy*. 313–328.

Yinqian Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. 2012. Cross-VM side channels and their use to extract private keys. In *Proceedings of the 2012 ACM Workshop on Cloud Computing Security (CCSW'12)*. 305–316.

Yinqian Zhang and Michael K. Reiter. 2013. Düppel: Retrofitting commodity operating systems to mitigate cache side channels in the cloud. In *Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS'13)*. 827–838.

Received May 2014; revised April 2015; accepted April 2015