Computing (2011) 91:93–118
DOI 10.1007/s00607-010-0140-x

Locking the sky: a survey on IaaS cloud security

Luis M. Vaquero · Luis Rodero-Merino · Daniel Morán

Received: 12 August 2010 / Accepted: 2 November 2010 / Published online: 24 November 2010

Abstract Cloud computing is expected to become a common solution for deploying applications thanks to its capacity to leverage developers from infrastructure management tasks, thus reducing the overall costs and services' time to market. Several concerns prevent players' entry in the cloud; security is arguably the most relevant one. Many factors have an impact on cloud security, but it is its multitenant nature that brings the newest and more challenging problems to cloud settings. Here, we analyze the security risks that multitenancy induces to the most established clouds, Infrastructure as a service clouds, and review the literature available to present the most relevant threats, state of the art of solutions that address some of the associated risks. A major conclusion of our analysis is that most reported systems employ access control and encryption techniques to secure the different elements present in a virtualized (multitenant) datacenter. Also, we analyze which are the open issues and challenges to be addressed by cloud systems in the security field.

Keywords Cloud computing · Security · IaaS · Multitenancy

Mathematics Subject Classification (2000) 68M01 · 68U35

L. M. Vaquero (✉)
Hewlett-Packard Labs, Bristol, UK
e-mail: lmvaquero@ieee.org

L. Rodero-Merino
LIP ENS Lyon, Graal/Avalon Group, INRIA, Villeurbanne, France
e-mail: luis.rodero-merino@ens-lyon.fr

D. Morán
U.N.E.D, Madrid, Spain
e-mail: dmoran@uned.es

1 Introduction

Cloud computing has recently being paid an important attention and dubbed as the "next-best-thing" in Information and Communication Technologies (ICT). Cloud computing is about moving services, computation and/or data off-site to an internal or external, location-transparent facility or contractor. This is not, indeed, new and infrastructure outsourcing has been used since the 1980s [1].

In spite of the many different existent cloud definitions, they all agree that this paradigm aims at offering every network-accessible computing resource "as-a-Service" (XaaS) [2]. Thus, the major cloud resource-exposure models highlighted so far have been: (1) Infrastructure (storage, computation and network capabilities) as a Service (IaaS), basically consisting on the deliverance of virtual machines (VMs) to an IaaS provider, who can rise or shrink the number of VMs so as to offer fast and easy scalability according to variable workloads. (2) Platform (development, service lifecycle, etc. support tools) as a Service (PaaS), offering facilities to develop software products and host them in third-party infrastructures. (3) Software as a Service (SaaS), where the user buys a subscription to some on-line software. Among these three, IaaS is, arguably, the most established cloud service model, already offering a wide variety of products and advanced capabilities: automated scalability, pay-per-use, and on-demand provisioning are some of the most relevant.

In spite of these advantages, security is the most relevant inconvenience for fastening the widespread adoption of the cloud for many business-critical computations [3,4]. IaaS clouds inherit many security concerns associated to the technologies they use. However, the blend of these technologies and the large amounts of users sharing resources create new security concerns. The key for enabling cloud operators to provide seemingly unlimited scaling potential is resource sharing. This enables maximum utilization of the available assets, but introduces new multitenancy concerns. The resources shared are not only the physical machines, but also the network (conventional links and even the network interface on the machine). Virtualization eases computational resources sharing and it is broadly used in current clouds (e.g. Amazon employs a tuned version of Xen). But virtualization embodies a series of threats and vulnerabilities due to the transformation from a dedicated infrastructure into a multitenant scenario where machines and networks are totally shared.

While some cloud security works (see [5]) focus on legal and jurisdictional risks, some institutions provide valuable recommendations for increasing cloud security [6,7] from a more technical point of view. For example, they introduce good policies for operation and management of datacenter facilities. However, the recommendations given by those works are fairly general, including those referred to virtualization. None of them contains a comprehensive analysis of the specific threats (e.g. possible attacks) to be addressed by cloud providers, which are introduced by the special characteristics of clouds (scale, combination of virtualization and multitenancy, etc.). We deem such analysis to be essential for providers to be able to build safe cloud environments. Here, we present already reported threats relating them to the underlying infrastructure to ease their comprehension. Also, we focus on state of the art securitization technologies and challenges that have proven or may prove

useful/required in real clouds. Such solutions would enable the application of the recommendations found in [6,7].

In fact, many of the threats that clouds must face are already known in other fields. This could be expected as IaaS cloud implementations combine already existing technologies, each one with its own potential menaces. But clouds also have certain characteristics that bring new security concerns. First, they must handle a potentially huge demand for scalability. Also, clouds are by nature multitenant environments where tenants share resources. The cloud platform and cloud users must so be protected against attacks from malicious tenants.

This paper presents the most widely accepted cloud threats (Sect. 2) and relates these with three different areas or domains of the IaaS model: machine virtualization domain, network virtualization domain, and physical domain. The machine virtualization domain is discussed in Sect. 3, which deals with IaaS-specific security concerns derived from the multitenant environment induced by virtualization technologies. Next, Sect. 4 copes with security in the network domain, as complete network sharing is an unavoidable companion effect to virtualization and infrastructure sharing. Then, Sect. 5 is devoted to the physical domain briefly introduces some available measurements for securing the physical infrastructure. This is not the major aim of the paper, but it is included here for the sake of completeness. Finally, we wrap up major lessons learned, recommendations and challenges to "lock" the cloud in Sect. 6.

2 Main security threats for the cloud

The list of cloud threats is determined by the large number of technologies it comprises. The most relevant threats for the cloud have recently been shown in [8]:

- **Threat 1:** *Abuse and Nefarious Use of cloud computing:* IaaS providers bring the illusion of unlimited compute, network, and storage capacity. However, IaaS offerings have hosted the botnets, trojan horses, and software exploits.
- **Threat 2:** *Insecure Interfaces and APIs:* a set of APIs to manage and interact with cloud services are typically exposed (provisioning, monitoring, etc). Cloud services' security is dependent upon the security of these basic APIs.
- **Threat 3:** *Malicious Insiders*: this is amplified in the cloud by "the convergence of IT services and customers under a single management domain, combined with a general lack of transparency into provider process and procedure".
- **Threat 4:** *Shared Technology Issues*: IaaS vendors deliver their services in a scalable way by sharing infrastructure. Monitor environment for unauthorized changes/activity.
- **Threat 5:** *Data Loss or Leakage*: This threat increases in the cloud, due to the architectural or operational characteristics of the cloud environment (e.g. insecure APIs, shared environment, etc.).
- **Threat 6:** *Account or Service Hijacking*: phishing, fraud, and exploitation are well-known issues in IT. The cloud adds a new dimension to this threat: "if an attacker gains access to your credentials, they can eavesdrop on your activities and transactions, manipulate data, return falsified information, and redirect your clients to

Table 1 Mapping of VMs lifecycle to threats (as described in Sect. 2) applicable at that stage and dependencies on the IaaS infrastructure

Stage	Lifecycle name	Potential cloud threats	IaaS infrastructure dependencies
1	Image definition	NA	Local to user
2	Image creation	NA	Local to user
3	Image customization	NA	Local to user
4	Transportation	Threats 4 and 5	Shared network
5	Storage	Threats 1–5	Shared network and storage
6	Deployment	Threats 1, 2, 3 and 5	Shared VMM, storage and network
7	Contextualization	Threats 1, 2, 3, 4, 5, 6	Shared physical node, VMM/cluster/datacenter and network
8	Runtime	Threats 1, 3, 4, 5, 6 and 7	Shared physical node, VMM/cluster/datacenter and network
9	Undeployment	Threats 2, 3 and 5	Shared VMM, storage and network

Table 2 Mechanisms to protect VMs from different threats at different stages of their lifecycle

Lifecycle name	Protection/prevention techniques/frameworks	Newly introduced
Storage/transportation	Cryptographic protection (threats 2, 4, 5) [9] (F), [10] (t), [11] (t)	No
	Access control (threat 2) [12] (t)	No
Deployment	Encrypted boot and data partition (threats 3, 5) [14] (t) , [15] (F)	No
	"Linkable" VMs (threats 3, 5) [15] (F)	Yes
Contextualization	Custom binding (threats 3, 4, 5) [13] (t)	No
Runtime	Deviation from "normal" behavior (threats 1, 6) [16] (t)	No
	VM introspection (VM memory consistency checks) (threats 1, 6) [17] (t), [18] (t), [19] (t)	No

F means complete framework; *t* stands for technique

illegitimate sites. Your account or service instances may become a new base for the attacker".

– **Threat 7:** *Unknown Security Profile*: the reduction of cost of ownership induced by the cloud also resulted in more complex analysis of a company's security posture. More tenants imply increased complexity in detecting who and how is using the infrastructure.

These threats will be used throughout the paper to help illustrate where the dangerous points lurk at every level in the typical IaaS cloud architecture: virtualization, network, and physical domains (see Tables 1, 2, 3, and 4). These Tables relate the general threats above to more specific attacks and relevant protection mechanisms.

Table 3 Mechanisms to protect VMMs from different threats

VMM vulnerability	Protection/prevention techniques/frameworks	Newly introduced
Timing channels	Determinism in guest OS (threats 3, 4, 5) [29] (t), [28] (t)	No
Side channels	Hardware support for cryptography operations (e.g. Intel's AES support) (threats 3, 4, 5)	No
Other inter-VM communication	Non-memory lock down (threats 3, 4, 5) [31] (F)	No
Unauthorized software execution	Reduce privileged VM control instructions (threats 1, 3, 4) [32] (t)	No
	Access control and security policies (threats 1, 2, 4, 5, 6) [35] (F), [33] (t), [22] (t), [36] (t), [37] (t)	Yes
Contextualization	Custom binding (threats 3, 4, 5) [13] (t)	No
Runtime	Deviation from "normal" behavior (threats 1, 6) [16] (t)	No
	VM introspection (VM memory consistency checks) (threats 1, 6) [18] (t), [19] (t), [17] (t)	No

F means complete framework; *t* stands for technique

Table 4 Datacenter threats and published protection mechanisms

Datacenter vulnerability	Protection/prevention techniques/frameworks	Newly introduced
Physical machine	Hardware verification via TPM	No
Secure data storage	TPM's sealed storage (threats 2, 3, 4, 5) [42] (t), [43] (F)	No
VM authentication	vTPM (threats 4, 5, 6)	No
	TVMM(threats 4, 5, 6) [35] (F)	No
VM aggregation/grouping	TVDc (threats 3, 4) [45] (F)	Yes
	Access control (threats 2, 4, 5, 6)	No
Hardware/software integrity	vTPM (threats 1, 4, 6)	No
	TCCP (threats 1, 4, 6) [44] (F)	Yes

F means complete framework; *t* stands for technique

3 Security in the machine virtualization domain

Being virtualization one of the key enablers of cloud scalability, it is important to introduce virtualization-derived vulnerabilities at all levels of the "virtualization stack": VM level, introduced in Sect. 3.1, and Virtual Machine Monitor (VMM) level, addressed in Sect. 3.2. Also, the risks that involves the inclusion of several datacenters are discussed in Sect. 3.3.

3.1 Securing VM images

Several threats are there to VMs like eavesdropping of the information sent to the disk or to the network, changes in the VM image itself, taking the whole VM over, etc.

Thus, there are some security issues that need to be taken into account that concern VM management as a software packet whenever they are being created and transferred to the VMM environment they will dwell into.

Some approaches do not consider the previous lifecycle stages of VMs before they are in the VMM.[1] As shown in the sequence of stages in the VM lifecycle shown by the first two columns in Table 1, VM definition (define operating system (OS) version), VM creation (packaging to build a VM image), VM customization (add and configure the needed software stack and configuration procedures, also known as contextualization), transportation to the VMM, storage in a repository, deployment, running and undeployment can also be deemed as risky instants in the VM lifecycle. VM transportation, storage, deployment/contextualization and runtime can be specially prone to attacks, since VMs are more exposed.

Software cryptographic protection of individual files could prevent data modification/leakage during transportation or storage. Previous software attempts to secure VMs on a Grid of servers [9] or desktop computers [10] also pointed in this direction, trying to establish trust for VMs, keeping data files encrypted on disk, or distributing images to many clients assuring data integrity using HTTP-FUSE CLOOP[2] [11], respectively. On the other hand, hardware-based disk encryption mechanisms, such as Intel's Danbury, often restrict compatibility, transparency and manageability.

As for image storage, Jinpeng et al. [12] recently proposed an image management system controlling access to images, tracking the origin of the images, and providing users and administrators with efficient image filters and scanners that detect and repair security violations.

Once an image has been transported and locally stored, it has to be booted and context information acquired to make it work properly (e.g. connecting a web server to a previously deployed database even across different IaaS cloud providers). This is usually accomplished by using different images for different providers, which reduces portability. A recent approach defines a binding protocol (VMs need a pre-installed agent in charge of its part of the protocol) [13]. Constandache et al. establish a secure channel (SSH/SSL/TLS) with mutual authentication between a VM and an automated controller for "post-install actions by the controller to adapt the VM to its local environment, join it to an application service, and/or monitor and control its execution".

Regarding secure VM deployment, Descher et al. provide a simple encryption mechanism. VMs are, thus, composed of a boot system (containing the logic to verify and access the encrypted partition) and together with an encrypted data partition [14]. A general framework for easing encryption key management has recently been proposed in which usage policies indicating the type of trusted system and the VMs that can be connected to the service are employed [15].

Once the machine is running, the VMM has little means for detecting and correcting an anomalous behavior. Raj and Schwan [16] developed a protection mechanism by including the notion of trust-based mandatory access control (MAC) mechanisms to VMs. This approach assumes that each guest VM has an expected behavior, and

[1] http://www.opennebula.org/doku.php?id=documentation:tp2:ug.

[2] HTTP-FUSE CLOOP uses a network loopback block device and saves each compressed block to a file.

deviations from it may reduce the "trust" on the VM. Sala et al. [17] employed VM introspection techniques (based on psyco-virt) to evaluate a series of consistency tests on kernel data structures in the memory of the VMs [18,19].

These mechanisms for protecting a VM in the most relevant points of its lifecycle are summarized in Table 2. It is worth mentioning that the VMM is responsible for enforcing VM runtime protection mechanisms. The last column reveals that most protection techniques/frameworks have not been designed to treat cloud-exclusive attacks.

3.2 VMM

Hypervisor protection on the confidentiality and integrity of tenant software and data have been the subject of a great deal of research. Some literature supports the idea that the VMM is a secure environment, assuming a VMM is safer since it has much less code than an OS [20] and easier to verify using formal tools.[3] But the drive towards features has pushed commodity VMMs to include more code, which increases the risk that a serious bug will appear since VMMs can themselves have more access to hardware resources than conventional applications (they typically execute on the most privileged ring,[4] ring 0, although some exceptions are also available [21]). Salier et al. [22] point out that these "actual isolation" concerns are backed by plenty of examples of bugs in hypervisors that were exploited to compromise security [18] or several recent attacks.[5, 6] The fear derived from these and many other examples has provoked a backwards drive towards simpler VMMs. For example, Flicker reduces the size of hypervisors by exploiting new CPU features designed to make writing hypervisors easier [23].

The first task to secure a VMM is hiding the fact that the machine is a VM to avoid some available exploits. VMs offer additional mechanisms to create a pristine uninfected picture, observe infection's impact and easily restore the system to the pristine state [24]. Thus, attackers have a significant stake in detecting VMs. For instance, when it comes to VMware, malware typically relies on two techniques to detect VMs: (1) *Communication channels*: VMware allows for communication between host and guest OS through a custom hard-coded communication channel. Carpenter et al. [24] discovered a code snippet which behaved differently whenever it run in a VM; this code attempted to detect the presence of the communication channel to determine the presence of a VM. (2) *Memory differences*: the location of the interrupt, global and local descriptor tables (IDT, GDT, and LDT, respectively) vary between host OSs and guest machines. For instance, it has been observed that in VMware the IDT is typically located at `0xFFXXXXXX`, whereas it is far lower in memory in a host OS. Although

[3] Garfinkel et al. [18] describe an architecture for embedding intrusion detection directly inside a hypervisor.

[4] On the x86 architecture, the kernel runs in ring 0 (the highest privilege level) and user processes run in ring 3 (the lowest privilege level).

[5] http://wiki.whmcs.com/HyperVM.

[6] http://www.blackhat.com/presentations/bhusa09/KORTCHINSKY/BHUSA09KortchinskyCloudburstSLIDES.pdf.

these memory differences occur also in multiprocessor architectures, making it a less reliable VM detection method [24].VMMs typically employ a container which is a virtual CPU for an unmodified OS to run in a shared hardware. VM entries load processor state or inject an interrupt or exception. The CPU effects this injection using the guest IDT, just as if the injected event had occurred immediately after the VM entry. VM exits save processor state into the guest-state area and load processor state from the host-state area. All VM exits use a common entry point to the VMM. These exists cause a trap into the VMM which should mimic the right behavior to keep VMs unaware of the shared CPU environment.

The rest of this section studies the kind of attacks that can try to exploit the multi-tenant nature of clouds' VMMs: covert channels.

3.2.1 Covert channels

VMMs are specially vulnerable to covert channel attacks. According to the Trusted Computer Security Evaluation Criteria, TCSEC, a covert channel is created by a sender process that modulates some condition (such as free space, availability of some service, wait time to execute) that can be detected by a receiving process. The TCSEC defines two types of covert channels: (1) *Storage channels*: direct or indirect writing to a storage location by one process and the direct or indirect reading of the storage location by another process; (2) *Timing channels*: one process signals information to another process by modulating its own use of system resources (e.g., central processing unit time) in such a way that this manipulation affects the real response time observed by the second process. When they do not imply any proactive attacking mechanism, but mere passive observation, timing channel attacks are also included in the so-called *side channel attacks*. A *side channel attack* is any attack based on information gained from the physical environment, as opposed to brute force or theoretical weaknesses. For example, timing information, power consumption or electromagnetic leaks can bring extra information.

Covert channels are used to communicate an attacker VM with some other VMs in the VMM [25], bringing additional information to the attacker who can be more efficient in his/her attack. A common timing channel attack is deriving secret keys used in cryptographic operations through legitimate activities (inferring keys from cache power consumption, taking into account the total number of cache misses or CPU timing usage, which reveal characteristic patterns when performing cryptographic operations). Examples of timing attacks to cryptographic keys are given by [26] and [27]. For instance, in shared environments such as multi-core CPU, a sensitive computation sharing a CPU core with an attacker enables the attacker to analyze usage patterns and reconstruct and steal information from the victim via the shared L1 data cache, shared functional units, the branch target cache, or the instruction cache.

Software based techniques to homogenize this side information are handy, but often result in unacceptable slowing. Controlling timing channels implies, for instance, limiting the rate at which one user's demand for a shared resource may visibly affect the resource's availability to another user, either by statically partitioning the resource or by injecting noise into scheduling decisions. These two latter methods reduce the statistical multiplex implemented on the shared hardware. Aviram et al. propose

an enhanced architecture in which the basic idea is to "eliminate all internal reference clocks" [28]. These authors rely on a gateway ("determinator") that employs a reference clock plus a second adjustable one that ensures that guest code can do nothing to make its results depend on internal timing, thus preventing timing channel attacks. Thus, determinism in guest operations results in elimination of timing channels by means of: strict inter-process communication and process lifetime constrains. For instance, a process cannot outlive its parent, a process can communicate directly only with its immediate parent and children, each process has its own registers and address space, and processes cannot share read/write access to the same physical memory. Building this deterministic cloud depends on the existence of a deterministic VMM. A deterministic VMM enforces determinism [29] by recording and replaying a previous (nondeterministic) execution, which imposes a high performance cost.

It is worth mentioning the efforts that hardware vendors are doing to avoid side channel attacks by implementing some instructions in hardware. Intel has included AES instructions[7] in its 2010 Core series to avoid timing attacks such as the ones presented in [26] and [27].

Beyond covert or side channel attacks, there are other "multitenancy-derived" vulnerabilities that allow for executing undesired code. Typically, a hypervisor attack exploits a vulnerability, such as buffer overflows, to inject malicious code into the hypervisor. In Blue Pill attacks, for instance, a rootkit (i.e. a tool to gain administrator-level control over a system without) is installed so that the host OS remains unaware that anything (including hardware interrupts, requests for data or system time) can be intercepted and a fake answer given. An example of vulnerability enabled by rootkits is that a rootkit could theoretically be inserted via a virtual device asking for I/O by performing a write in a special-purpose stored-program computer optimized for I/O (also known as a channel). This program causes the start I/O instruction (SIO), which is a privileged one, to trigger a trap to the VMM, which can easily rewrite the instructions to reference only those I/O and memory locations authorized for a given VM [30]. The results of a recent work could be used to mitigate some of the risks associated to such indirection layers; Wang's and Jiang's "non-memory lock-down" secures page table memory so that it can only be altered by the hypervisor administrator. This means that no new executable code can be exerted, nor could existing code be altered [31].

3.2.2 Securing modern VMMs

In modern I/O architectures secure virtualization has become even more complex, exposing new inter VM communication/control "gaps" that could be exploited. For

[7] Intel Advanced Encryption Standard (AES) instructions are a new set of instructions for all new Intel's based on 32 nm Westmere. These instructions enable fast and secure data encryption and decryption, using the AES which is defined by FIPS Publication number 197. The architecture consists of six instructions that offer full hardware support for AES. Four instructions support the AES encryption and decryption, and other two instructions support the AES key expansion, offering significant increases in performance compared to pure-software implementations.

example, in Xen a single VM, *Dom0*, is the management VM that controls the rest since they "trust" in it.

Xen provides mechanisms for communication to be used by Dom0 to control the other VMs. These mechanisms may be a target for malicious attackers. The simplest one is similar to a system call in traditional OSs, the hypercall, a call from the hosted VMs (DomU) to the hypervisor (software trap to transfer control to the hypervisor). The creation of new VMs (known as domain building), done by Dom0, requires such Dom0-DomU communication. Dom0 uses its special privileges. Xen must have full control over and protection from the VMs that it hosts, so it runs in ring 0 and hosted VM kernels are paravirtualized in order to run in ring 1. in order to access the address space and processor state of the new VM. Murray et al. [32] reduce the risks associated to executing in a privileged ring by creating a domain builder with the necessary privileges in order to build new VMs and an inter-VM communication mechanism that enables developers to create lightweight code that communicates between Xen guest VMs without using a networking stack.

More generally, a number of secure VMMs proposals has been made: sHype,[8] and others [33]. Similarly, SeVMM joined MAC policies, inner-domain communication and secure policy management to build an implementation of a secure VMM [34]. sHype allows secure resource sharing by implementing *access control mechanisms* among groups of related VMs [22,35,36]. Terra [35] includes a trusted VM monitor (TVMM) that partitions a tamper-resistant hardware platform into multiple, isolated VMs. Both, the hardware and TVMM can act as a trusted party to allow some VMs to cryptographically identify the software they run to remote parties. Each VM runs in its own hardware protection domain, providing strong isolation between VMs.

Yet, even thee secure VMM environments lack appropriate mechanisms to configure flexible security policies for VMM management. Rueda et al. recently analyzed the complexities related with the existence of a *hierarchy of MAC* mechanisms representing the interaction between different policy layers (VMM and VM). These MAC mechanisms [37] may prove weak against some recent attacks by which users on the cloud infrastructure may snoop on traffic via timing channels [38]. By placing a VM intentionally on the same physical machine as another customer's VM, the former is allowed to estimate the cache usage, traffic load and keystroke timing of the latter. In this case, hiding "cloud cartography" is essential to protect customers.

Finally, given the number of vulnerabilities induced by VMMs (summarized in Table 3), recent approaches [39] aim at eliminating it from cloud infrastructures by implementing system per core VM usage with appropriate memory partition, dedicated I/O devices and moving VM networking to network switches, such as NoHype.

[8] It offers strong isolation, mediated sharing and communication between VM, implemented by an access control enforcement engine. This engine can enforce mandatory policies such as Multi-level Security (MLS), Role-based Access Control (RBAC).

3.3 Datacenter securitization

Of course, cloud providers do not manage a single VMM but many (even thousands) of them [40], possibly spread over geographically distributed datacenters. Thus, in addition to securing individual elements in the virtualization layer, there are other initiatives aimed at a more integrative effort, dealing with one whole hypervisor, one datacenter or several datacenters.

The first step towards a secure datacenter is relying on secured platforms (software and hardware combinations). TPM (Trusted Platform Module) is useful in this regard since it delivers an integrated circuit that provides basic security-related functions to the software utilizing it. It is normally installed on the motherboard of a PC or laptop, and communicates with the rest of the system via a hardware bus.

TPM-enabled systems are used to create cryptographic keys and encrypt them so that they can only be decrypted by the TPM (often referred to as "wrapping" or "binding" a key).[9] TPM has been proposed as a mechanism for providing process isolation (users can hide pages of main memory so that it can be assured that pages are not modified or observed by any other application or even the OS), secure I/O paths (the information flowing between input/output devices and the system is both encrypted and cryptographically signed), and a Direct Memory Access (DMA) exclusion vector (an in-memory table that may be cached and ensures that DMA devices do not read or write to a secure area of memory).

It is obvious that a TPM is not a device that was designed to be accessed by multiple systems at the same time: TPM requires having *a* dedicated hardware module. In order to overcome this limitation IBM developed the *virtual TPM*, implementing TPM functionality in a software-based virtualized TPM capable of spawning multiple virtual TPM instances [41]. IBM implemented a TPM front- and back-end driver through which all guest VMs can transparently communicate with a virtual TPM-hosting VM where a custom implementation of an extended V1.2 TPM resides.

Getting back for a moment to Sect. 3.1, TPM has also been applied to securing VMs throughout their full lifecycle but transportation and runtime. Gebhardt and Tomlinson employed it to deliver integrity assurance to virtual disk images on the Grid [42].[10] Other approaches also joined trusted computing (TC) concepts with virtualization technologies [35,43].

Mechanisms such as those employed by Terra [35] (see previous subsection on Securing Modern VMMs), reliably detect whether or not the host is running a platform

[9] The master "wrapping" key is called the Storage Root Key (SRK) and is stored within the TPM itself, so that its private part is never exposed. Indeed, private keys are kept separated from memory controlled by the operating system. TPM allows for creating a wrapped key that is also linked to certain hardware platform so that the key can only be unwrapped when those hardware measurements have the same values that they had when the key was created. This process is called "sealing". Hardware is more resistant to attack than software alone, especially regarding cryptographic key management; this is the strength of TPM. Also, since the TPM uses its own internal firmware and logical circuits for processing instructions, it does not rely upon the OS and is not subject to external software vulnerabilities.

[10] Gebhardt and Tomlinson also proposed a system in which the operations are aggregated so as to reduce the cryptography-derived overhead (e.g. writable sections are prone to re-hashing, while read-only sections are not).

implementation that the remote party trusts. These platforms can effectively secure a VM running in a single host. However, many datacenters comprise several hundreds of machines where any VM can be dynamically scheduled. Going beyond a single VMM, Santos et al. [44] propose a trusted cloud computing platform (TCCP) for IaaS. By using TPM, TCCP verifies the hardware and software integrity of a computing node (host) in IaaS before it offers services to cloud users, guaranteeing that no cloud provider's privileged administrator can inspect or tamper with its content. There exists an external trusted entity like Verisign that provides a service which keeps track of the trusted VMs in a cluster.

Other approaches, such as IBM's Trusted Virtual Data-center (TVDc), also address infrastructure and management issues introduced by data-center virtualization [45]. TVDc groups VMs and resources that collaborate towards a common purpose into workloads called Trusted Virtual Domains (TVDs), which provide strong isolation between workloads by enforcing a MAC policy throughout a data-center. This policy defines which VMs can access which resources, and, by extension, which VMs can communicate with each other. Thus, one can infer that MAC policies and TC technologies are the predominant trends towards securing current data-centers.

Table 4 summarizes this subsection by emphasizing the most relevant threats and protection mechanisms published against them at the datacenter level.

3.4 Audition of datacenters

TPM lets cloud providers include several layers of authentication, where the machine is known by the cloud and the machine has a relationship with the user, significantly enhancing cloud security. The TPM's hardware-based security can be easily integrated with software security identification (ID) standards for federated ID management such as OpenID, Security Assertion Markup Language (SAML), WS (Web Services)—Federation and others. TPM can also be used to protect tenants data from malicious hardware components with access to memory or disks, data can be stored in memory and on disk in encrypted form, with keys leveraging the sealed storage feature of TPMs. However, memory encryption is not yet available on commodity CPUs and current TPMs provided limited protection against hardware attacks, such as the sand-and-scan attack used by Tarnovsky to extract secret keys from a TPM [46].

Overall, using TPM-enabled elements, a tenant, a cloud provider or an auditor could verify that an application is indeed running on the contractually specified hardware and software (BIOS, hypervisor, OS, up on) [5]. Similarly, Iliev and Smith propose logs that utilize a security co-processor, such as the IBM 4758, to achieve tamper evidence [47]. Their work followed on the PacketVault project [48], which aimed at capturing and recording every packet over a 10 MBps link indefinitely on commodity disk storage. For a survey on technologies for remotely verifying properties of hardware and software, see the work by Parno [49].

4 Security in the network virtualization domain

Networks are the "glue" sticking cloud applications together. VMs of a single customer may be deployed in different VMMs in multiple physical hosts, yet they need

to be connected while intrusions should be avoided. The network has always been a shared environment, thus facing multitenancy challenges. However, while physical machine virtualization impact on computing is a well known issue, its consequences for networking are less explored: traffic isolation and network scalability are the most prominent ones. The increase in the number of co-located servers may dramatically increase the traffic supported by the same physical hardware. Virtual network requirements on scale and isolation for the cloud are by far exceeding those in current network architectures.

Beyond scaling, there are also some security concerns for IaaS clouds that have to do with the network. Previous solutions also offered network virtualization avoiding overlay networks; spawning networks employ nested programmable networks to build a hierarchical net of virtual networks in which parent networks can spawn child networks to use the parent's resources (children are allowed to use a more specific environment while still inheriting some basic network functionality from their parents) [50]. This approach requires the installation of a spawning kernel in every involved domain, implying extensive changes in already existent network infrastructures which carriers tend to refuse [45].

As of today, networking in virtualized networks is typically performed in two different manners: (1) a simple L2 software switch within the VMM mapping data from the virtual interfaces of every VMs either to the physical interfaces (connecting with VMs in other physical machine) or to the virtual interfaces (connecting to VMs in the same physical machine) (here we will also refer to it as "ethernet virtualization") [51]. (2) overlay networks and TCP/IP virtualization. These techniques are respectively focused in the usage of virtual local area network (VLAN) tags (L2) to separate traffic or public key infrastructures to build L2/L3 overlays [52–55]. Both rely on encryption techniques, but have different scope of usage:

4.1 Ethernet virtualization approaches

Ethernet virtualization is designed for conveying multiple Ethernet connections over a single physical network. There is a broad background on tunneling protocols [56]. Being IEEE802.1Q-2003 VLAN standard one of the most representative ones, it adds VLAN tags to each network segment in order to separate distinct traffics. Since these tags are usually lost in WANs, ethernet encapsulation (wrapping ethernet packets into IP packets) has been proposed to overcome this limitation [45,52,57,58].

4.2 L2/L3 overlays

Network overlays (an aggregate of physical and virtual hosts, routers and tunnels) allow for secure network virtualization among participating hosts. Several overlays have been described in the available literature [52–54]. For instance, VNET implemented a virtual local area network spread over a wide area using *layer 2 tunneling* [59]. Session establishment among VNET servers and clients is performed by using a text-based protocol, while the clients present a password or an SSL certificate.

While VNET provided L2 tunneling features, VIOLIN creates a virtual and isolated internetworking environment (called VIOLIN) on top of PlanetLab's infrastructure. A VIOLIN is a small-scale virtual network with virtual routers, LANs and end-hosts arranged to confine communications inside it a VIOLIN. Each VIOLIN has, thus, its own address space [60]. The VIOLIN model is limited in its capabilities for communicating VIOLINs belonging to different organizations. In practice it would be desirable to enable policy-based flow control to allow for appropriate VIOLIN intercommunication [45]. In this line, VINI [55] relies on a custom combination of these two works (VIOLIN+VNET), to connect VIOLIN virtual networks through VNET-provided tunnels.

Other overlay efforts have focused in managing virtual private execution infrastructures (VPXI), *combining system and network virtualization with cryptographic identification of the virtual network resources*. This Simple Public Key Infrastructure (SPKI) was employed to generate the cryptographic identifiers of the computing resources [61]; these are automatically translated to IP addresses (using the Host Identity Protocol, HIP). VPXI sends a private key-signed SPKI certificate of every involved resource to the user, this SPKI certificate carries itself a signed authorization associated to the resource (identified by its public key) and granted to a previously registered receiver (identified by its public key). The user then sends the received certificate to initiate the connection with the resource. Cryptographic identification of resources eases security and allows for across inter-organizational communications (unlike VIOLIN).

These systems use predetermined and fixed IPs, which imposes some limitations for the required dynamism and scalability of cloud infrastructures. For instance, IaaS IPs are reused promptly as they become ready for usage).The potential persistence or released IPs in DNS caches (or MAC addresses in ARP tables) may cause some troubles [62].

4.3 Other approaches

Amazon also got into L3 connectivity between the cloud and users' facilities. Thus, Amazon released its Virtual Private Cloud (VPC) service, which integrates remote, virtual resources with users' physical computers by using Internet Protocol Security (IPsec). Other options make VMs share all the networking subsystem, but filter their binding to a set of available IPs [63]. Similar encapsulation mechanisms over Multiprotocol Label Switching (MPLS) are available, such as virtual routing and forwarding (VRF)-lite. This specification allows multiple instances of a routing table to co-exist within the same router simultaneously, identifying the customer and resolving redundant IP usage by relying on MPLS' inherent labeling.

The most relevant two approaches above: ethernet encapsulation and overlays, were put together resulting in a so called trusted virtual domain (TVD). TVDs employed Ethernet encapsulation, VLAN tagging, and VPNs to experimentally automate the instantiation and deployment of the appropriate security mechanism and network virtualization technologies based on an input security model that specifies the required level of isolation and permitted network flows [45].

Table 5 Virtual network threats and published protection mechanisms

Network vulnerability	Protection/prevention techniques/frameworks	Newly introduced
Denial of service (poor scalability)	Appropriate segmentation and dynamic bandwidth usage	Yes
Shared network	Hierarchical isolation [50] (t)	No
	L2 tunneling [59] (t)	No
	Virtual network with virtual routers and LANs with individual address spaces (VIOLIN/VINI) [60] (F), [55] (t)	No
	Cryptographic identification of resources [61] (F)	No
	Ethernet encapsulation + overlays (TVD) [45] (F)	No

F means complete framework; *t* stands for technique

Finally, Table 5 sums up the most relevant concerns with regards to security in the virtual network domain.

5 Other general issues

While Denial of Service, DNS Hacking, Routing Table Poisoning XDoS attacks (Economic Denial of Sustainability, EDoS or instance flooding), or Man in the Middle attacks, IP Spoofing, port scanning, and ARP cache attacks are possible, they are not exclusive to cloud settings [64]. For instance, Amazon S3 suffered data corruption due to a flaky border gateway router.[11]

The major differences with regard to security have to do with scale when building large cloud infrastructures: revealed by the inability of state firewalls, Syn flood protection load balancers or access control lists (ACLs) to scale up to the level required for cloud environments. This scalability need is further exemplified by the presence of requirements on controlling every new data flow by a network-wide policy enforcement [65]. Also, firewalls and intrusion detection systems cannot handle load bursts in large clouds. Other networking mechanism that can suffer from scalability problems are Netfilter/IP Tables. These could be regarded as a software mechanism to isolate traffic of a VM from others' in the same physical server. A problem with Netfilter is that each packet introspection must be done by the host CPU, consequently reducing the system throughput, requiring hardware acceleration for layer 4 firewall capability [66]. Scalability is, thus, a major problem with regard to security introduced by the cloud to current physical infrastructures: overflows may cause IaaS system failure.

Apart from these scalability related issues IaaS clouds also present some relevant vulnerabilities at the physical level that can be used to perform some important attacks. For instance, Ristenpart et al. [38] show how to exploit cloud provider IP assignment procedures to set up other attacks, such as VM side-attacks (see Sect. 3). The important point raised by this paper is that by building a "cloud cartography", some other attacks

[11] http://developer.amazonwebservices.com/connect/thread.jspa?threadID=22709.

can be issued more easily. Several techniques such as prevention of static assignation of IP addresses and association of different allocation requests to machine types or regional zones, and isolating each user's view of the internal cloud address space (e.g bridging and VLANs), could prove to be useful mechanisms for slowing down this type of attacks.

6 Discussion and conclusions

Table 6 wraps up the principal attacks and vulnerabilities reported in the reviewed literature and maps these to the seven threats reported by [8]. As can be observed, most of the reported attacks and vulnerabilities correspond with more than a single threat at the same time, responding to a multi-factorial nature. We should remark that threat 7 is more related to trust and, thus, it is not directly addressed by the technical approaches herein reported. However, this fact does not diminish the importance of risk quantification by means of auditing and trust chain-establishment (see discussion below and Sect. 3.4).

Figure 1 summarizes the most relevant lessons derived from the analysis of the state of the art at the different layers of the proposed cloud architecture at the IaaS level. As can be observed, scalability is a paramount problem for securing current clouds at the level of the physical infrastructure. Virtualization is one of the major concerns for typical clouds, the tools used for "multiplexing" several VMs into a single physical machine are not invulnerable and some exploits are available in a multitenant environment. In addition, VM images management is an issue since VM images need to be moved from in-house trusted facilities to a cloud provider through unsecured networks. Moreover, the vulnerabilities can be performed at deployment and runtime also in multitenant environments such as public clouds. Virtual networks are also subject to some security concerns, most of them regarding the secure and dynamic establishment of data paths for communicating distant VMs. A traditional solution has been the usage of PKI-enabled VPNs, overlays or VLAN tagging. These are feasible at edge networks, but it poses a very significant overhead for the carrier in backbones. In addition, encrypted overlays do not prevent from disclosing the encrypted traffic by its attributes observable to an eavesdropper, for example, packet sizes, sequences, inter-packet timings, etc. such as the side and covert channels we mentioned above. Virtual networks call for scalable public key techniques capable of managing the huge amount of VMs and communication channels that need to be dynamically turned on/off as needed. These could be based on a profound renovation of the underlying network infrastructure, hosting virtual switches, firewalls, and managing domains rather than security zones.

Thus, locking an IaaS cloud involves careful control on the employed VMM and whether it is, or not, a secure one. Also, automated mechanisms for controlling VM's security are required (specially at those phases where they are beyond a company's organizational boundaries). vTPM technologies are proving essential to increase security at all levels of the virtualization stack. TPM has also been criticized since process management has been translated from OS sandbox to application authentication support. Also, TPM owner passwords can be remotely changed and,

Table 6 Grouping of the main attacks in accordance with the threats presented in [8]

Threat #	Attack/vulnerability
Threat 1	Deviations from normal behavior[16–19]
	Hypervisor exploits [21,22]
	Execution in privileged ring [32]
	Failures to check hardware/software integrity [44]
Threat 2	Modifications of images at storage [12]
	Context shifting [13]
	Corrupted boot [14,15]
	Insecure storage [42,43]
Threat 3	Deviations from normal behavior[16–19]
	Revealed virtual nature of the VM [24]
	Corrupted boot or data partitions [14,15]
	Covert channels [25–31]
	Execution in privileged ring [32]
	Shared workspace [35]
	Insecure storage [42,43]
	Shared workload [45]
Threat 4	Revealed virtual nature of the VM [24]
	Modifications at transportation time of the images [9–11]
	Covert channels [25–31]
	Execution in privileged ring [32]
	Shared workspace [35]
	Insecure storage [42,43]
	Shared workload [45]
	Failures to check hardware/software integrity [44]
	Insecure network
Threat 5	Modifications at transportation time of the images [9–11]
	Context shifting [13]
	Corrupted boot [14,15]
	Covert channels [25–31]
	Shared workspace [35]
Threat 6	Deviations from normal behavior[16–19]
	Shared workspace [35]
	Failures to check hardware/software integrity [44]
Threat 7	None

thus, do not offer enough guarantees. This could be overcome by running parallel networks offering differentiated access to classified material depending on the user's authorization.

Additional means have to be provided in order to hide as much information as possible to potential attackers. Heuristics to make a cartography of the cloud and induce side attacks are relatively easy to obtain and can be dangerous. The network

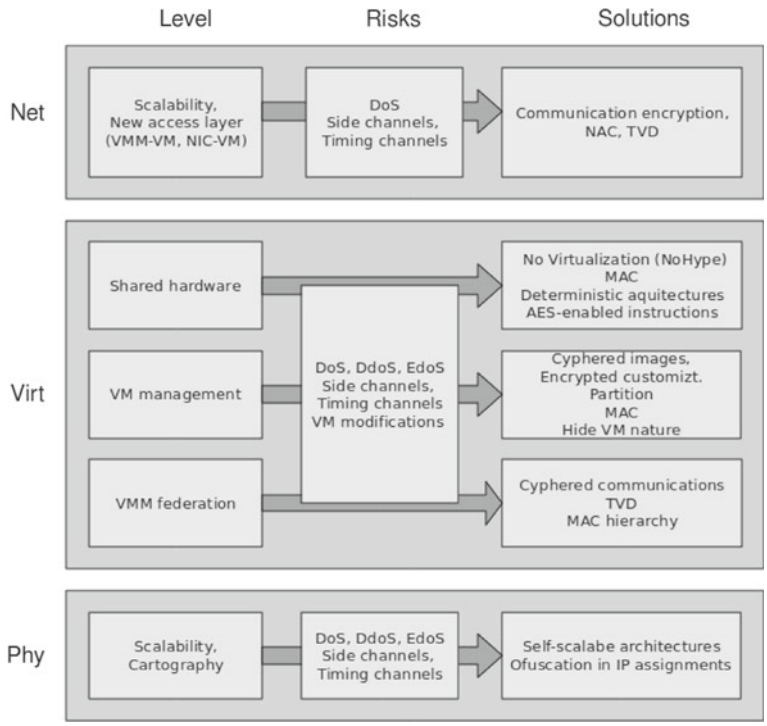

Fig. 1 IaaS security summary

is a traditionally shared environment that deserved further consideration in current IaaS clouds. Mechanisms to simplify/automate PKI management and reduce the huge cryptographic overhead are needed. Again, side attack channels gaining unauthorized access by observing patterns should be reduced by enforcing pseudo-random behaviors. Malfunctioning systems (either networks or VMMs) should be promptly detected by using formal methods, thus promoting early attack detection/abortion.

In spite of the fact that we have been indicating cloud-specific elements throughout the paper, arguably many of the "cloud security" issues just reflect traditional web application and data-hosting problems. These remain well-established challenges such as phishing, downtime, data loss, password weaknesses, and compromised hosts running botnets [3]. Cloud security manifests as a blend of existing topics and most available papers keep a separate focus on specific issues as shown in Tables 1, 2, 3, 4, and 5. These Tables reveal that most of the reported techniques/frameworks do not directly arise to solve cloud-specific needs. At the VM level we could not find any cloud exclusive technique or threat. Since it is believed that the cloud should manage the notion of whole applications (rather than collections of VMs), indicating the type of trusted system and the VMs that can be connected to the service can be considered as a cloud-only technique, in which a chain of trust is needed between different stakeholders as we already mentioned. At the

VMM level, the novelty point comes from the need to integrate several hypervisors across a datacenter (and even across datacenters) and managing the massive scale (huge number of users and services with large numbers of VMs involved). Integrative access control and security policies are a novel element appeared with the advent of the cloud. Also, setting groups of collaborating VMs and resources into workloads (see TVDs above), enforcing strong isolation by MAC mechanisms or TCCPs, where external hardware and software verification were enforced to the cloud's physical elements by an external third party, can be considered as cloud exclusive due to their integrative approach of the different technologies used to build the cloud.

Cloud computing offers a potentially more trustworthy alternative to botnets since cloud botnets are easier to shut down than traditional botnets [3]. Also, since the cloud relies on shared resource, unexpected side channels (passively observing information) and covert channels (actively sending data) can arise, such as those reported in [38]. Thus, a single user cannot disrupt many users even when the cloud provider is supposed to offer better security practices than traditional environments.

Cloud environments introduce novel elements such as its multitenancy which forces one to protect and hide activity pattern, not just data and software or hardware. This multitenancy is enabled by the huge scaling potential of the cloud; there where there was a single server or user, there is now a bunch of virtual servers and services for the same user over the same infrastructure. Thus, scalability is a major weakness that can easily be exploited. Access control mechanisms are there to help controlling these potential attacks, but a trade-off is required between scalability and business profitability and monitoring granularity. The needed scaling potential for the cloud is a new factor that previous systems did not need to dealt with before.

6.1 Challenges ahead

6.1.1 Security at hardware level

At the processor and chip-set level, acceleration of security and virtualization features is expected. The trend to let Symmetric Multi-Processors behind and integrate heterogeneous multicore processors will also accelerate the inclusion of TC means and TMPs into the processor.

Having heterogeneous hardware can help to increase efficiency. NoHype and similar approaches are signaling to solving the challenge of a maximum exploitation of the hardware parallelism to reduce shared usage of the underlying hardware.

6.1.2 Image management

Regarding VM image management, VM encryption techniques are hampered by the fact that VMs are usually large files. A workaround for this problem is the usage of a combination of concatenated files, which improves security (it is harder to get several files than a single one).

6.1.3 VMMs securitization

Some techniques for runtime protection have been developed, such as hiding the fact that the intruder is dealing with a VM, VM introspection or behavioral monitoring to detect anomalous actions that could be pointing to a compromised VM. However, monitoring the activity of all the VMs in a datacenter implies massive computational power and resources. More work is needed to enhance behavioral and introspection techniques without hampering datacenter operation. The ease of cloning and distribution of VMs between physical servers could result in the propagation of configuration errors and other vulnerabilities. Proving the security state of a system and identifying the location of an insecure VM will be challenging. Regardless of the location of the VM within the virtual environment, the intrusion detection and prevention systems will need to be able to detect malicious activity at VM level. The co-location of multiple VMs increases the attack surface and risk of virtual machine-to-virtual machine compromise.

At the hypervisor level, MAC and trusted computing techniques are the major bet to help build future secure cloud systems. vTPM represents a relevant effort towards delivering all the capabilities of TC to VMs. TPM's secure storage and cryptographic functions are made available to OSs and applications running in VMs. But as hypervisors mature, recursive virtualization technologies can be required that pose new security threats.

The inherent nature of a datacenter, in which many VMMs need to be managed and secured poses the need for sophisticated mechanisms. For instance, advanced access control and policies integrating local VMM policies with global datacenter strategy, dynamic management of the keys employed for securing the datacenter and TVDc techniques could help mitigate the associated risks.

6.1.4 VMs isolation

A VMM should provide *isolation*, so that no VM can access or modify the software running in the VMM or in a separate VM; *inspection*, meaning that the VMM has to control the status and behavior of every VM (CPU state (e.g. registers), memory, and I/O devices state; and *interposition* VMMs act as proxies of some VM operations (e.g. executing privileged instructions). While interposition is an inherent feature of VMM, VM isolation is less of an achievement (as we mentioned above) and active inspection mechanisms are still under way in commercial settings (some research efforts are underway, though).

Another important research area is determining granularities for isolation: isolate by virtual or physical machines, LANs, VMM, or datacenters. As of today, we lack understanding of the tradeoffs between security and performance for each of these options [3]. This constitutes an important novel research challenge in IaaS security. The analysis of the required isolation level is essential al the IaaS layer to ensure high level data protection techniques will work: e.g. resource isolation to ensure security of data during processing, by isolating the processor caches in VMs, and isolating those virtual caches from the hypervisor cache or virtual Transaction Lookaside Buffer, or

securing VMs disk drivers from other VMs in hypervisors to prevent access to stored data, encryption of data during VM migrations, etc.

6.1.5 Networking in virtual networks

Covering different levels of the IaaS cloud model leads us to changes at the network level. The cloud leads to a drop in security as the traditional controls such as VLANs (virtual local-area networks) and firewalls prove less effective during the transition to a virtualized environment and denial. Pfaff et al. [67] also indicated how while techniques to secure dynamic establishment of communication channels is a pending issue in datacenters (more so if the application components, VMs, dwell in separate datacenters across IaaS cloud providers), techniques for isolating traffic have been there for a good while.

6.1.6 Scalability issues

Denial of service attacks are easier in an environment with such a high number of users if not appropriately manager. Thus, workload management to control cloud scalability in massively used datacenters is a pending security issue. In the cloud, administrative access is done via the Internet rather than the restricted on-premises connection in the traditional data center model. This increases risk, thus requiring more demanding monitoring for changes in system control and access control restriction.

6.1.7 Trust and auditability

In the rush to take advantage of the benefits of cloud computing, not least of which is significant cost savings, many corporations are like to rush into cloud computing without a serious consideration of the security implications. To establish zones of trust in the cloud, the virtual machines must be self-defending, effectively moving the perimeter to the virtual machine itself. Enterprise perimeter security (i.e., firewalls, demilitarized zones, network segmentation, intrusion detection and prevention systems [IDS/IPS], monitoring tools, and the associated security policies) only controls the data that resides and transits behind the perimeter. In the cloud computing world, the cloud computing provider is in charge of customer data security and privacy.

Throughout the entire paper we have assumed full trust in the IaaS provider, which is what happens in current hosting or connectivity providing approaches (we fully rely on our hosting or Internet service provider). This, however, does not need to be true. The interplay with world-renowned entities acting as certification authorities will be essential in minimizing threat 7, which cannot be solved from a technical point of view only. The complex chain of trust introduced by the different cloud stakeholders can be further complicated by the federation of an application's component across different cloud providers [68]. Some cloud-specific approaches [44], introduced a third external trusted authority to guarantee that the cloud provider herself could not gain access into the deployed VMs. In these scenarios, securing the network connection between datacenters is a desirable goal yet to be accomplished. Achieving auditability without impairing performance is yet to be accomplished. Auditor are independent third party,

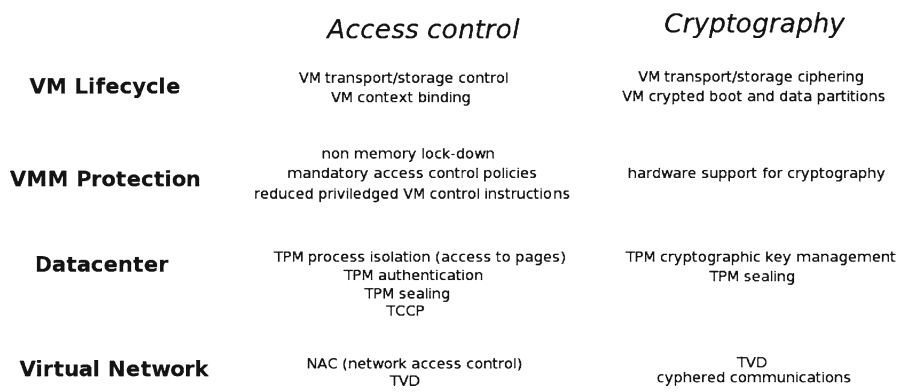

Fig. 2 Access control and cryptographic approaches in the literature

and a third-parties; more so in a complex trust chain as presented above. This is different than todays practice, in which cloud providers record and maintain their own audit logs. Several attempts are already under way. For instance, the CloudAudit working group is working on a specification for an API focused on"audit, assertion, assessment, and assurance for cloud providers.[12] The goal is to generate machine-readable details on which security features a provider offers. Potential users can then programmatically decide whether to purchase resources from a provider for their application given their own security requirements. Enterprises are often required to prove that their security compliance is in accord with regulations, standards, and auditing practices, regardless of the location of the systems at which the data resides. Data is fluid in cloud computing and may reside in on-premises physical servers, on-premises VMs, or off-premises virtual machines running on cloud computing resources, and this will require some rethinking on the part of auditors and practitioners alike.

Although a number of security metrics were proposed [69], we still need reliable ways for assessment of security. A final challenge in security in general has to do with the unclear validity of most security quantification methods [70], in this work the author claims that "quantified security is a weak hypothesis because a lack of validation and comparison between such methods against empirical data". Also, we lack a widely-accepted and unambiguous definition which defines what it means that one system or technique is more secure than another [71]. The cloud introduces new security situations, like a massively shared environment with several stakeholders and even competitor business sharing the same infrastructure, which makes it even harder to find proper terms for general comparison. This is a pending issue that certainly deserves further exploration to help users decide which cloud (or which security technique or framework) is more secure.

As a conclusion, after reviewing some of the most relevant threats for IaaS clouds and the paramount solutions presented by the community, one can ascertain that most of the proposed solutions at all the presented levels (from physical nodes and networks to the VM itself) deal with appropriate access control and data obfuscation

[12] http://www.Cloudaudit.org.

(via encryption or other mechanisms) either implemented in hardware or in software. Figure 2 helps to illustrate this point by grouping initiatives in the literature into one of these two securitization mechanisms. Integrative initiatives avoiding such as variety of proposed mechanisms, e.g. keys to be stored for the network, for the VMs, for the VMM, etc. and holistic access control mechanisms that ease a coherent securitization of the whole datacenter are very much needed. Having a locked IaaS cloud is an essential pre-requisite for building secure higher level clouds (like platform as a service ones dealing, for instance, with important issues such as data privacy).

References

1. Owens CD (2010) Securing elasticity in the cloud. Commun ACM 53(6):46–51. http://10.1145/1743546.1743565
2. Vaquero L, Rodero-Merino L, Caceres J, Lindner M (2009) A break in the clouds: towards a cloud definition. ACM Comput Commun Rev 39(1):50–55
3. Chen Y, Paxson V, Katz RH (2010) Whats new about cloud computing security. Tech. Rep. UCB/EECS-2010-5, EECS Department, University of California, Berkeley
4. Viega J (2009) Cloud computing and the common man. Computer 42:106–108
5. Molnar D, Schechter S (2010) Self hosting vs. cloud hosting:accounting for the security impact of hosting in the cloud. In: Workshop on the economics of information security
6. CSA: Cloud security guide (2009) Tech. rep., cloud security alliance. http://www.cloudsecurityalliance.org/csaguide.pdf
7. ENISA: Cloud computing: Benefits, risks and recommendations for information security (2009) Tech rep., European Network and Information Security Agency
8. Archer J, Boheme A, Cullinarie D, Puhlmann N, Kurtz P, Reavis J (2010) Top threats to cloud computing. Tech. rep., Cloud Security Alliance. http://www.cloudsecurityalliance.org/topthreats
9. Lu W, Keahy K, Freeman T, Siebenlist F (2005) Making your workspace secure: establishing trust with vms in the grid super computing. In: Supercomputing. Poster
10. Calder B, Chien AA, Wang J, Yang D (2005) The entropia virtual machine for desktop grids. In: VEE '05: Proceedings of the 1st ACM/USENIX international conference on virtual execution environments, pp 186–196. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1064979.1065005
11. Suzaki K, Yagi T, Iijima K, Quynh NA (2007) Os circular: internet client for reference. In: LISA'07: Proceedings of the 21st conference on large installation system administration conference, pp 1–12. USENIX Association, Berkeley, CA, USA
12. Jinpeng W, Xiaolan Z, Glenn A, Vasanth B, Peng N (2009) Managing security of virtual machine images in a cloud environment. In: CCSW '09: Proceedings of the 2009 ACM workshop on Cloud computing security, pp 91–96. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1655008.1655021
13. Constandache I, Yumerefendi A, Chase J (2008) Secure control of portable images in a virtual computing utility. In: VMSec '08: Proceedings of the 1st ACM workshop on virtual machine security, pp 1–8. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1456482.1456484
14. Descher M, Masser P, Feilhauer T, Tjoa AM, Huemer D (2009) Retaining data control to the client in infrastructure clouds. In: Availability, reliability and security, international conference on 0:9–16. http://doi.ieeecomputersociety.org/10.1109/ARES.2009.78
15. Baldwin A, Dalton C, Shiu S, Kostienko K, Rajpoot Q (2009) Providing secure services for a virtual infrastructure. SIGOPS Oper Syst Rev 43(1):44–51. http://doi.acm.org/10.1145/1496909.1496919
16. Raj H, Schwan K (2009) Extending virtualization services with trust guarantees via behavioral monitoring. In: VDTS '09: Proceedings of the 1st EuroSys workshop on virtualization technology for

dependable systems, pp 24–29. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1518684.1518689
17. Baiardi F, Sgandurra D (2007) Building trustworthy intrusion detection through vm introspection. In: IAS '07: Proceedings of the third international symposium on information assurance and security, pp 209–214. IEEE Computer Society, Washington, DC, USA. http://dx.doi.org/10.1109/IAS.2007.25
18. Garfinkel T, Rosenblum M (2003) A virtual machine introspection based architecture for intrusion detection. In: Proceedings on network and distributed systems security symposium, pp 191–206
19. Sala G, Sgandurra D, Baiardi F (2007) Security and integrity of a distributed storage in a virtual environment. In: Proceedings of 4th international IEEE security in storage workshop, pp 58–69
20. Perez R, van Doorn L, Sailer R (2008) Virtualization and hardware-based security. IEEE Secur Privacy 6(5):24–31. http://dx.doi.org/10.1109/MSP.2008.135
21. Aoyagi S, Oikawa S (2008) Ixiv vmm: a vmm on 2-level ring architecture. In: Computer and information technology, IEEE 8th international conference on 0:533–538. http://doi.ieeecomputersociety.org/10.1109/CIT.2008.Workshops.62
22. Sailer R, Jaeger T, Valdez E, Caceres R, Perez R, Berger S, Griffin JL, Doorn Lv (2005) Building a mac-based security architecture for the xen open-source hypervisor. In: ACSAC '05: Proceedings of the 21st annual computer security applications conference, pp 276–285. IEEE Computer Society, Washington, DC, USA. http://dx.doi.org/10.1109/CSAC.2005.13
23. McCune JM, Parno BJ, Perrig A, Reiter MK, Isozaki H (2008) Flicker: an execution infrastructure for tcb minimization. In: Eurosys08: Proceedings of the 3rd ACM SIGOPS/EuroSys European conference on computer systems 2008, pp 315–328. ACM, New York, NY, USA
24. Carpenter M, Liston T, Skoudis E (2007) Hiding virtualization from attackers and malware. IEEE Secur Privacy 5:62–65
25. Okamura K, Oyama Y (2010) Load-based covert channels between xen virtual machines. In: SAC '10: Proceedings of the 2010 ACM symposium on applied computing, pp 173–180. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1774088.1774125
26. Tromer E, Osvik DA, Shamir A (2009) Efficient cache attacks on aes, and countermeasures. J Cryptol 23(1):37–71
27. Kocher PC (1996) Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In: CRYPTO '96: Proceedings of the 16th annual international cryptology conference on advances in cryptology, pp 104–113. Springer, London, UK
28. Aviram A, Hu S, Ford B, Gummadi R (2010) Determinating timing channels in statistically multiplexed clouds. CoRR **abs/1003.5303**
29. Dunlap GW, Lucchetti DG, Fetterman MA, Chen PM (2008) Execution replay of multiprocessor virtual machines. In: VEE '08: Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on virtual execution environments, pp 121–130. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1346256.1346273
30. Karger PA, Safford DR (2008) I/o for virtual machine monitors: Security and performance issues. IEEE Secur Privacy 6(5):16–23. http://dx.doi.org/10.1109/MSP.2008.119
31. Wang Z, Jiang X (2010) Hypersafe: A lightweight approach to provide lifetime hypervisor control-flow integrity. In: 31st IEEE symposium on security and privacy
32. Murray D, Milos G, Hand S (2008) Improving xen security through disaggregation. In: 4th ACM SIGPLAN/SIGOPS international conference on virtual execution environments, pp 151–160
33. Karger PA (2005) Multi-level security requirements for hypervisors. In: ACSAC '05: Proceedings of the 21st annual computer security applications conference, pp 267–275. IEEE Computer Society, Washington, DC, USA. http://dx.doi.org/10.1109/CSAC.2005.41
34. Wen-Zhi C, Hong-Wei Z, Wei H (2008) Sevmm: Vmm-based security control model. In: Proceedings of cyberworlds, international conference on 0:820–823. http://doi.ieeecomputersociety.org/10.1109/CW.2008.110
35. Garfinkel T, Pfaff B, Chow J, Rosenblum M, Boneh D (2003) Terra: a virtual machine-based platform for trusted computing. In: Proceedings of the nineteenth ACM symposium on operating systems principles, pp 193–206. ACM Press
36. Hirano M, Shinagawa T, Eiraku H, Hasegawa S, Omote K, Tanimoto K, Horie T, Kato K, Okuda T, Kawai E, Yamaguchi S (2008) Introducing role-based access control to a secure virtual machine monitor: security policy enforcement mechanism for distributed computers. In: Asia-Pacific conference on services computing. 2006 IEEE 0:1225–1230. http://doi.ieeecomputersociety.org/10.1109/APSCC.2008.14

37. Rueda S, Vijayakumar H, Jaeger T (2009) Analysis of virtual machine system policies. In: SACMAT '09: Proceedings of the 14th ACM symposium on access control models and technologies, pp 227–236. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1542207.1542243
38. Ristenpart T, Tromer E, Shacham H, Savage S (2009) Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: ACM conference on computer and communications security
39. Keller E, Szefer J, Rexford J, Lee R (2010) Nohype: virtualized cloud infrastructure without the virtualization. In: ISCA '10: Proceedings of the international symposium on computer architecture
40. Ruan A, Shen Q, Yin Y (2008) A generalized trusted virtualized platform architecture. In: Young computer scientists, international conference for 0:2340–2346. http://doi.ieeecomputersociety.org/10.1109/ICYCS.2008.508
41. Berger S, Cáceres R, Goldman KA, Perez R, Sailer R, van Doorn L (2006) vtpm: virtualizing the trusted platform module. In: USENIX-SS'06: Proceedings of the 15th conference on USENIX security symposium. USENIX Association, Berkeley, CA, USA
42. Gebhardt C, Tomlinson A (2008) Secure virtual disk images for grid computing. In: APTC '08: Proceedings of the 2008 Third Asia-Pacific trusted infrastructure technologies conference, pp 19–29. IEEE Computer Society, Washington, DC, USA. http://dx.doi.org/10.1109/APTC.2008.17
43. Lioy A, Ramunno G, Vernizzi D (2009) Trusted-computing technologies for the protection of critical information systems. J Inform Assur Secur (4):449–457
44. Nuno Santos Krishna P, Gummadi RR (2009) Towards trusted cloud computing. In: Hot Cloud. http://www.usenix.org/event/hotcloud09/tech/full_papers/santos.pdf
45. Cabuk S, Dalton CI, Ramasamy H, Schunter M (2007) Towards automated provisioning of secure virtualized networks. In: CCS '07: Proceedings of the 14th ACM conference on computer and communications security, pp 235–245. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1315245.1315275
46. Tarnovsky C (2010) Deconstructing a secure processor. In: Black hat briefings federal. http://www.blackhat.com/presentations/bhdc10/Tarnovsky_Chris/BlackHat%DC2010TarnovskyDASPslides.pdf
47. Iliev A, Smith SW (2005) Protecting client privacy with trusted computing at the server. IEEE Secur Privacy 3(2):20–28. http://dx.doi.org/10.1109/MSP.2005.49
48. Antonelli CJ, Undy M, Honeyman P (1999) The packet vault: secure storage of network data. In: ID'99: Proceedings of the 1st conference on workshop on intrusion detection and network monitoring, pp 11–11. USENIX Association, Berkeley, CA, USA
49. Parno B (2008) Bootstrapping trust in a "trusted" platform. In: HOTSEC'08: Proceedings of the 3rd conference on hot topics in security, pp 1–6. USENIX Association, Berkeley, CA, USA
50. The genesis kernel: a virtual network operating system for spawning network architectures (1999)
51. Barham P, Dragovic B, Fraser K, Hand S, Harris T, Ho A, Neugebauer R, Pratt I, Warfield A (2003) Xen and the art of virtualization. In: SOSP '03: Proceedings of the nineteenth ACM symposium on operating systems principles, pp 164–177. ACM, New York, NY, USA. http://doi.acm.org/10.1145/945445.945462
52. Sundararaj AI, Dinda PA (2004) Towards virtual networks for virtual machine grid computing. In: VM'04: Proceedings of the 3rd conference on virtual machine research and technology symposium, pp 14–14. USENIX Association, Berkeley, CA, USA
53. Touch J (2001) Dynamic internet overlay deployment and management using the x-bone. Comput Netw 36(2–3):117–135. http://dx.doi.org/10.1016/S1389-1286(01)00172-4
54. Andersen D, Balakrishnan H, Kaashoek F, Morris R (2001) Resilient overlay networks. In: SOSP '01: Proceedings of the eighteenth ACM symposium on operating systems principles, pp 131–145. ACM, New York, NY, USA. http://doi.acm.org/10.1145/502034.502048
55. Bavier A, Feamster N, Huang M, Peterson L, Rexford J (2006) In vini veritas: realistic and controlled network experimentation. In: SIGCOMM '06: Proceedings of the 2006 conference on applications, technologies, architectures, and protocols for computer communications, pp 3–14. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1159913.1159916
56. Davoli R (2005) Vde: Virtual distributed ethernet. In: TRIDENTCOM '05: Proceedings of the first international conference on testbeds and research infrastructures for the DEvelopment of NeTworks and COMmunities, pp 213–220. IEEE Computer Society, Washington, DC, USA. http://dx.doi.org/10.1109/TRIDNT.2005.38
57. Dalton C (2005) Xen virtualization and security. Tech. rep., HP Security Office
58. Housley R (2002) Rfc 3378. etherip: Tunneling ethernet frames in ip datagrams. RFC. http://www.faqs.org/rfcs/rfc3378.html

59. Sundararaj AI, Gupta A, Dinda PA (2004) Dynamic topology adaptation of virtual networks of virtual machines. In: LCR '04: Proceedings of the 7th workshop on workshop on languages, compilers, and run-time support for scalable systems, pp 1–8. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1066650.1066665
60. Jiang X, Xu D (2003) Violin: Virtual internetworking on overlay infrastructure. In: Proceedings of the 2nd international symposium on parallel and distributed processing and applications, pp 937–946. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.2.4260
61. Primet PVB, Gelas JP, Mornard O, Koslovski G, Roca V, Giraud L, Montagnat J, Huu TT (2009) A scalable security model for enabling dynamic virtual private execution infrastructures on the internet. In: CCGRID '09: Proceedings of the 2009 9th IEEE/ACM international symposium on cluster computing and the grid, pp 348–355. IEEE Computer Society, Washington, DC, USA. http://dx.doi.org/10.1109/CCGRID.2009.76
62. Mather T, Kumaraswamy S, Latif S (2009) Cloud security and privacy: an enterprise perspective on risks and compliance. O'Reailly Media, Sebastopol, CA, USA
63. Soltesz S, Potzl H, Pluczynski M, Bavier A, Peterson L (2007) Copntainer-based operating system virtualization: A scalablem hig-performance alternative to hypervisors. In: Eurosys, pp 275–287
64. Jensen M, Schwenk J, Gruschka N, LoIacono L (2009) On technical security issues in cloud computing. Cloud Computing. In: IEEE international conference on 0:109–116
65. Casado M, Freedman MJ, Pettit J, Luo J, McKeown N, Shenker S (2007) Ethane: taking control of the enterprise. SIGCOMM Comput Commun Rev 37(4):1–12. http://doi.acm.org/10.1145/1282427.1282382
66. Bernstein D, Ludvigson E (2009) Networking challenges and resultant approaches for large scale cloud construction.In: Grid and pervasive computing conference, workshops at the 0:136–142. http://doi.ieeecomputersociety.org/10.1109/GPC.2009.10
67. Pffaf B, Pettit J, Koponen T, Anidon K, Casado M, Shenker S (2009) Extending networking into the virtualization layer. In: ACM SIGCOMM's hot topics in networks (HotNets) workshops. http://conferences.sigcomm.org/hotnets/2009/papers/hotnets2009-final143.pdf
68. Rodero-Merino L, Vaquero LM, Gil V, Galán F, Fontán J, Montero RS, Llorente IM (2010) From infrastructure delivery to service management in clouds. Future Gen Comput Syst 26(8):1226–1240
69. Jaquith A (2007) Security metrics: replacing fear, uncertainty, and doubt. Addison-Wesley Professional, Reading
70. Verendel V (2009) Quantified security is a weak hypothesis: a critical survey of results and assumptions. In: NSPW '09: Proceedings of the 2009 workshop on new security paradigms workshop, pp 37–50. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1719030.1719036
71. Krautsevich L, Martinelli F, Yautsiukhin A (2010) Formal approach to security metrics.: what does "more secure" mean for you? In: ECSA '10: Proceedings of the fourth European conference on software architecture, pp 162–169. ACM, New York, NY, USA. http://doi.acm.org/10.1145/1842752.1842787