State of The Art and Hot Aspects in Cloud Data Storage Security
SICS technical report T2013:01

Nicolae Paladi [1†], Christian Gehrmann [2†] and Fredric Morenius [3*]

[†]Swedish Institute of Computer Science
[*]Ericsson Research

March 3, 2013

[1]nicolae@sics.se
[2]chrisg@sics.se
[3]fredric.morenius@ericsson.se

Abstract

Along with the evolution of cloud computing and cloud storage towards maturity, researchers have analyzed an increasing range of cloud computing security aspects, data security being an important topic in this area. In this paper, we examine the state of the art in cloud storage security through an overview of selected peer reviewed publications. We address the question of defining cloud storage security and its different aspects, as well as enumerate the main vectors of attack on cloud storage. The reviewed papers present techniques for key management and controlled disclosure of encrypted data in cloud storage, while novel ideas regarding secure operations on encrypted data and methods for protection of data in fully virtualized environments provide a glimpse of the toolbox available for securing cloud storage. Finally, new challenges such as emergent government regulation call for solutions to problems that did not receive enough attention in earlier stages of cloud computing, such as for example geographical location of data. The methods presented in the papers selected for this review represent only a small fraction of the wide research effort within cloud storage security. Nevertheless, they serve as an indication of the diversity of problems that are being addressed.

Contents

Nomenclature

ASE	Asymmetric searchable encryption, page 12
C	Client, page 4
CBDG	Constraints-Based Data Geolocation, page 18
CE	Cloud Environment, page 4
CKA-2	Adaptive chosen keyword attacks, page 12
CP	Cloud Provider(s), page 5
CPABE	Ciphertext Policy Attribute-Based Encryption, page 10
CSP	Cloud Storage Provider, page 3
DMA	Direct Memory Access, page 15
DO	Data Owner, page 3
ESE	Efficient asymmetric searchable encryption, page 12
IaaS	Infrastructure-as-a-Service, page 3
IOMMU	Input/output memory management unit, page 15
KMIP	Key Management Interoperability Protocol, page 19
MWMR	Multiple Write Multiple Read, page 5
NIST	National Institute of Standards and Technology, page 16
NVRAM	Non-volatile random access memory, page 15
PaaS	Platform-as-a-Service, page 3
PDP	Provable Data Possession, page 12
PUF	Physical Unclonable Function, page 16
RAFT	Remote Attestation of Fault Tolerance, page 17
SaaS	Software-as-a-Service, page 3
SCI	System Call Interposition, page 15
SSE	Symmetric searchable encryption, page 12
SWMR	Single Write Multiple Read, page 5
SWSR	Single Write Single Read, page 5
TCB	Trusted Computing Base, page 15
TCG	Trusted Computer Group, page 15
TPM	Trusted Platform Module, page 15
TXT	Trusted Execution Technology, page 16
VM	Virtual Machine, page 3
VMM	Virtual Machine Manager, page 3

1 Introduction

Following a period of establishing and early adoption, cloud computing is now present in the product portfolio of most large technological companies, in one of the thee basic incarnations described by NIST: Infrastructure-as-a-Service (IaaS) , Platform-as-a-Service (PaaS) or Software-as-a-Service (SaaS) [1]. Another sign that the field has reached the early stages of its maturity is the emergence of legal frameworks that regulate the provisioning and usage of cloud computing services [2] and notably data protection in cloud storage [3].

While security concerns were long cited as barriers to wider adoption of cloud services, emerging regulation will likely require cloud providers to operate with an even wider set of tools to safeguard, when needed, the confidentiality, integrity, authenticity or even geolocation of data stored in the cloud.

This calls for a reassessment of tools and mechanisms available for protection of data in CE and we address this need with a survey of select peer-reviewed publications in the area of cloud storage security. To narrow our review domain, we focus on protection of data in IaaS with a special focus on storage of privacy-sensitive data. In this review, we have focused on papers that are either themselves addressing protection of sensitive data in CE or introduce tools that can be used to create effective, encompassing solutions for the protection of data in CEs.

Considering that storing data such as medical health records, personal financial data and other personally identifiable information is a privacy- and security-wise sensitive topic, the data owner (DO) must be able to obtain guarantees regarding data integrity, confidentiality as well as location within a specific jurisdictional area and can not trust the cloud storage provider (CSP) without such guarantees.

2 Defining Cloud Data Storage Security

In order to get an overview of the state of the art of "IaaS storage security", it is worth to first review what the term actually implies.

Concerns regarding data confidentiality [4, 5, 6, 7], as well as data access authorization aspects [8, 9, 10, 11] and the multitude of attack vectors that will be discussed below all point to the fact that "cloud storage security" is not only a non-trivial problem, but is not fully defined either.

We consider the following life-cycle of data in a cloud, partly illustrated in Figure 1:

1. User input,
2. Virtual Machine Manager (VMM) mediation,
3. Virtual Machine (VM) processing,
4. Data storage and replication,
5. Data replica management,
6. Data retrieval,
7. Data destruction,

not to mention the involved network traffic.

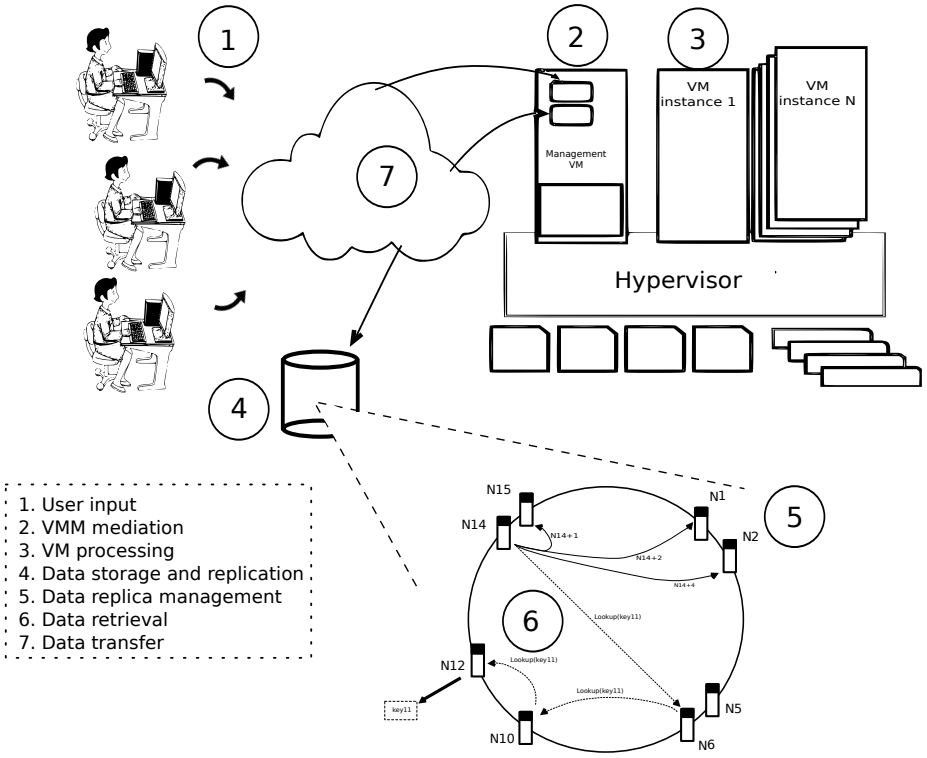

Figure 1: Data flow in the cloud

This sequence reflects the non-triviality of preserving an uninterrupted secure chain at all stages, while also ensuring high availability of the data at all times. At a minimum, in the context of IaaS storage security, such an uninterrupted secure chain would have to ensure the "traditional" confidentiality-integrity-availability requirements typical for data security. Next come cloud-specific or data storage-specific aspects that must be considered when establishing such a secure chain. These aspects can be derived from the concerns listed in papers focusing on IaaS storage security and fall into four distinct categories presented below.

2.1 Structured encryption and controlled disclosure

In the IaaS storage security context "structured encryption[1] and controlled disclosure" is concerned with both secure preservation and management of encryption keys for the data handled by the client's VM instance, as well as allocation, reallocation and revocation of data access by the DO to authorized third parties, further referred as (third party) clients (C). Both topics focus on data flows within the cloud environment(CE) , with the slight difference that while out-of-band communication between DO and instances of TPC is theoretically

[1]Among other publications, structured encryption is defined in [12]. A clarification of the concept is provided in subsection 3.2

possible, the interaction between DO and the components of the CE is only possible through the interfaces made available by the cloud provider (CP) , e.g. graphical or application programming interfaces. This latter aspect places special constraints on the capability of the DO to protect and manage the data encryption keys. Among other work, structured encryption and controlled disclosure is addressed in [6, 8, 9, 10, 12, 13, 14, 15] and somewhat in [7, 16].

2.2 Secure operations on encrypted data

While an area of continuous research (as shown by the selection of a new encryption standard to succeed AES conducted by NSA [17]) symmetric encryption already provides a substantial and versatile selection of refined algorithms and implementations. However, while encryption of Single Write Single Read (SWSR) static data may be a valid use case for IaaS storage settings, one of the essential characteristics of cloud environments – broad network access [1] – points to the fact that the data access patten in CE is more likely to be Single Write Multiple Read (SWMR) or even Multiple Write Multiple Read (MWMR). Furthermore, current data access patterns are typically characterized by *access through search* rather than file system navigation [7], so in order to qualify for a CE, a secure solution for cloud data storage should support search, indexing, updating and versioning of encrypted data stored in the CE. Some promising solutions are introduced in [7, 16, 18] and will be described below.

2.3 Protecting data in the compute host environment

An essential prerequisite for building a secure data protection chain in a CE is ensuring that the compute host where the DO's VM instance is deployed runs a trusted computing base that does not leak information or does not allow side channel attacks as described in [19]. This places the VMM in focus since it is both an attack vector and a component that could be used to enforce data encryption policies and protect it from being leaked. Several contributions [4, 5, 20, 21] propose a range of approaches to implement hypervisor-based encryption that will be introduced in the following subsections.

2.4 Cloud placement and geolocation verification

An emerging and less studied issue is control and enforcement of *cloud data storage location*. On the one hand, this aspect addresses the need for a suitable replication degree to ensure data availability in the event of network and disk failures [22]. On the other hand, a related issue is determining the location of data placed with the CSP [23, 24]. This latter issue is important in the face of an accidental or deliberate misplacement of data by the IaaS storage provider which may lead to a breach of SLAs or government regulations that impose restrictions on the placement and transfer of privacy-sensitive data [3].

2.5 Cloud data storage attack vectors

The distributed and modular architecture of IaaS storage makes it susceptible to multiple attack vectors. We list some of them below, in order to allow for an understanding and overview of current research in IaaS storage security.

- *Vulnerabilities in data transfer interfaces* – highlighted in, among other work, [25, 26] this attack vector focuses on the data transfer interface between the client and the CSP. Such interfaces may contain either vulnerabilities or intended or unintended functionality that can be misused to reveal ancillary information about the infrastructure of the CSP or even about other users' data stored by the CSP [25].

- *Breach of data confidentiality due to VMM vulnerabilities* is primarily a concern in the case of collocated malicious VMs. While an essential value proposition of virtualization is data isolation, earlier research has shown that collocated virtual machines can leak sensitive data [19]. Furthermore, VMMs could be both modified by a malicious CSP or simply misconfigured, jeopardizing the confidentiality of the data stored with the CSP. A number of solutions – ranging from VMM client attestation to VMM elimination have been proposed in the literature and will be discussed below.

- *Breach of data integrity* is considered plausible for example in scenarios where a CSP wishes to save costs by performing only a subset of the operations on data requested by the client, performs incorrect operations and thus breaches computational integrity [27], or discards the rarely used stored data [28]. This aspect has been considered and discussed, among others, in [4, 7, 10, 15].

- *Increased risk of accidental loss of data availability or confidentiality.* This may occur both in the case of a careless CSP due to vulnerabilities or misconfiguration, as well as in the case of subpoenas issued by law enforcement authorities. In certain cases, such events can even lead to permanent data loss [29]. The latter are a risk new to "IaaS storage", where CSP may be required by law enforcement authorities to provide a certain client's data and might even be prevented from notifying the customer [16, 27]. Furthermore, a subpoenaed IaaS storage provider can by negligence or due to legal circumstances be forced to reveal or lose data belonging to other customers collocated on the same hosts as the target organization [29]. Finally, anecdotal cases demonstrate that cloud data storage availability may be temporarily lost even on a global scale due to rather trivial issues, such as failure to replace expiring SSL certificates [30].

3 Dissecting current research within IaaS storage security

In this overview of data protection in CE we focus on recently published papers that address one, several or all of the categories mentioned above.

3.1 Key management of encrypted data

Key management is an important aspect of secure data storage, especially when a scalable and highly available "cloud" storage is intended. While symmetric encryption alone is perhaps enough to protect the confidentiality of a piece of data, many types of data (depending on the domain) are not static, but rather

represent an evolving set of information that can be updated, deleted on the one hand and shared or restricted from sharing on the other hand. As mentioned in [9], trivial approaches such as symmetric data encryption only, as well as data sharing based on shared secrets do not support evolving sets of accessing clients to common data. An immediate problem in this case is the distribution of new shared secrets to all authorized clients in the case of data re-encryption or routine key change.

One of the approaches for tackling encrypted data management is described in [6], where the authors propose a scheme where each information block is encrypted with a different symmetric key, thereby achieving a flexible, cryptography-based access control. An 'information block' is an abstract concept and can be of arbitrary size.

The scheme aims to protect data from a 'curious but not malicious' (*sic*) service provider, i.e. relaxing integrity requirements in favour of access control and data confidentiality. Integrity requirements are relaxed by assuming that the CSP would not attempt to modify or corrupt the stored data. Furthermore, the attack model assumes that the service provider correctly verifies the credentials of the clients and rejects data access in case of a revoked credentials, something which allows a malicious or misconfigured CSP to send even updated data to clients with revoked access rights.

The paper assumes a lazy revocation model, where the user with revoked rights can continue to access the data that she had access to prior to revocation (regardless of whether the user has actually accessed them before their privileges have been modified).

The scheme supports updates to data and a flexible, per user access management. One of the motivations is to also define a processing power-efficient schema, in order to be deployable on mobile devices with constrained HW resources. The scheme aims to optimize reads and updates, by avoiding re-encryption on the client side and focuses on providing fine-grained access control. However, re-encryption is in fact performed on the server side, increasing the cost in the pay-per-use cloud model.

Unfortunately, the approach makes several important assumptions which limit its applicability, especially in a high-scale and highly available system. Thus, the approach assumes:

1. an SWMR model, without advocating the choice of this somewhat limited model in any way. Authors touch only lightly on the applicability of the scheme for the MWMR model.

2. only *one* copy of each data block. This limitation sidelines data replication, which directly imposes a scalability limit on the system.

3. pre-distributed secrets between DO and CP, as well as DO and all end users.

4. a pseudo-random bit sequence generator P() for re-encryption operations shared between the DO and the CP.

Another implicit assumption is that the model is focused on large scientific data with a low update frequency, which once again, is quite a limited use case.

The authors [6] propose a key derivation hierarchy (presented in Figure 2), where every key in the hierarchy can be derived by combining its parent node

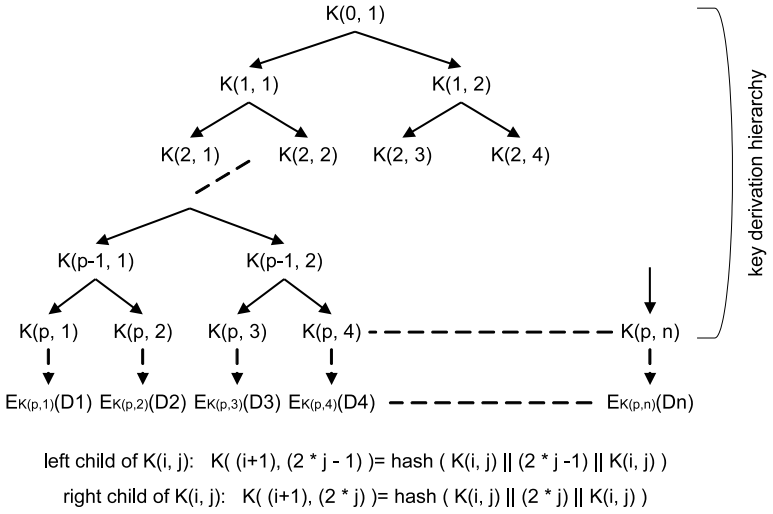

Figure 2: Key derivation hierarchy by Wang et al

and some public information. In essence, the authors propose to use hashes as encryption keys; however, they acknowledge that hashes might not offer substantial protection and propose a different method which would incur an undisclosed higher communication overhead during information access.

A different approach, [9] focuses on managing access rights upon shared versioned encrypted data on cloud infrastructure for a restricted, flexible group. The authors base their model on several components, namely a Key Graph, encrypted updates to the Key Graph (denoted as Key Trails) as well as actual versioned data, where the latter two are stored in the untrusted cloud and the first one is stored with a trusted third party. Key Trails are used to both adapt on the fly the Key Graph based on the granted or revoked data access, as well as format for deltas between two versions of the Key Graph. This model focuses on key management, leaving the encryption and decryption operations to the clients of the cloud storage.

The approach builds on earlier research in the area, namely VersaKey [31] and utilizes the generation of encrypted key updates by storing Key Trails on highly available and scalable but untrusted cloud infrastructures parallel to the encrypted data. All keys are versioned equivalent with the data, in order to allow a rather granular data access control, where the client can access a certain version or all previous versions of the data. The authors also describe a potential extension of the scheme that would allow a granular, per version client access to the data.

3.2 Controlled disclosure of encrypted data

Taking one step closer to performing computations on encrypted data, Chase and Kamara point to the fact that a large proportion of currently generated

data is not text data and focus on the issue of controlled disclosure by the DO of certain data aspects of a dataset [12]. The authors address this topic by applying *structured encryption*, which allows encrypting data while in the same time maintaining the ability to query and retrieve it.

In a nutshell, for a data structure σ and a sequence of data items $\mathbf{m}$= $(m_1...m_n)$ such that σ encodes the structure of the data and $\mathbf{m}$ represents the data, a structured encryption scheme takes as input structured data (σ, $\mathbf{m}$) and outputs an encrypted data structure γ and a sequence of ciphertexts $\mathbf{c}$ = $(c_1..c_n)$.

Considering the example of a data graph σ with n nodes and ith element of $\mathbf{m}$ is the data associated with node i, an efficient query involves querying σ to recover a set of pointers $I \subseteq [1, n]$ and then retrieve items in $\mathbf{m}$ indexed by I. In the cryptographic scheme proposed in [12] for encrypted data sets, it is possible, using the private key, to construct a token τ to any query such that pointers to the encryptions of $(m_i)_{i \in I}$ can be recovered from γ and τ.

There are several different definitions of structured encryption, namely:

- **(1) Structure-only**, which only considers the encryption of data structures, not the data items associated with it.
- **(2) Ciphertext-output**, where the scheme takes γ, τ and $\mathbf{c}$ as inputs according to the model described above. However, different from the above model, the ciphertext-output scheme returns a set of ciphertexts (containing the results of the query) rather than a set of pointers.
- **(3) Plaintext output** is strictly least secure than types **(1), (2)**, since the algorithm, while taking γ, τ and $\mathbf{c}$ as inputs, returns data items in plaintext, exposing the data to the party performing the query.

Structured encryption can be used both in the cases when the DO wishes to run private queries on encrypted data stored in a CE, to perform some computation over data in the CE, without disclosing the scope of the computation, as well as for "cloud-based data brokerage services" [2] [12].

The authors envision applications of structured encryption for private queries on encrypted data and controlled disclosure for local algorithms and provide a detailed description of a structured encryption scheme. One limitation of the proposed scheme is that it has only been considered for static data.

The authors present several queries suitable for different data models:

- **Lookup Queries of Matrices**, address structured encryption schemes for matrix-structured data. The proposed encryption process has an additional property of hiding the access pattern by inducing a pseudo-random permutation between $\mathbf{m}$ and $\mathbf{c}$ (variables from the above model apply).
- **Search Queries on Labeled Data** allows to apply structured encryption and encrypted search queries on labeled data, such as for example a key-value store. This is particularly relevant for a cloud storage environment, since key-value stores are currently offered by large CPs (e.g. Amazon's NoSQL storage service, based on DynamoDB [32]).

[2] According to Gartner, cloud service brokerages refer to third-party assistance for setting up cloud services: http://www.gartner.com/technology/research/cloud-computing/cloud-services-brokerage.jsp

- **Neighbor Queries on Graphs** addresses encryption of graph-structured data for graphs that support neighbor queries. The authors propose encoding the graph as labeling and applying the same approach as for labeled data. [12]

The paper by Chase and Kamara lays a solid foundation with regard to controlled disclosure and operations on encrypted data. It is particularly relevant for cloud storage in the attack model of an untrusted or partly trusted cloud storage providers.

In a different approach, Santos et al combine the use of trusted computing and Ciphertext Policy Attribute-Based Encryption (CPABE) to protect the confidentiality of data in IaaS environments while allowing access to certain parts of the dataset according to a predefined policy [14]. The paper introduces a new trusted computing abstraction - *policy-sealed data* - which allows customer data to be encrypted according to a customer-chosen policy and guarantees that only nodes satisfying the policy can access the plaintext version of the data.

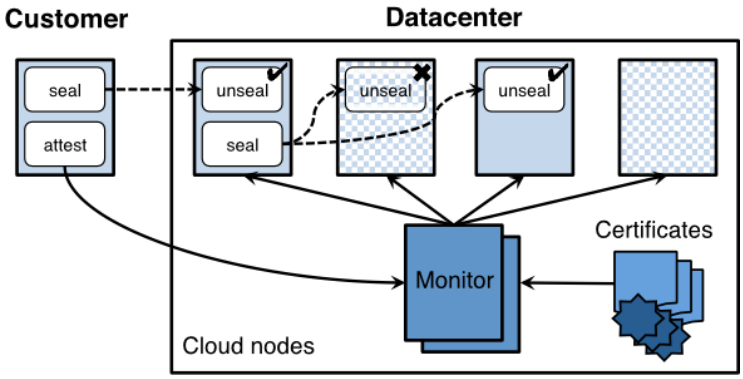

Figure 3: Deployment model for the excalibur CPABE framework

CPABE offers two cryptographic primitives, namely *seal* and *unseal*. Thus, data can be *sealed* on any node taking plaintext data and an attestation policy $\mathcal{AP}$ as input; it can be *unsealed* only if the node's configuration satisfies customer's attestation policy requirements.

In essence, the DO entrusts a *monitor* to attest the nodes and to TPM-seal the client's encryption keys for the data stored on the nodes. The monitor maintains its persistent state in a certificate store and a key store, both as XML files TPM-sealed to a trusted configuration of the monitor. A major drawback of this approach, common to many cloud data storage protection mechanisms is the inefficient key revocation scheme. Thus, in order to revoke a key from an "attested" node, the data would have to be resealed, which might not be the optimal solution, given that scalability of storage is one of the advantages of cloud-based storage.

3.3 Secure operations on encrypted data

Capability to execute operations on encrypted data without revealing neither the content of the operations nor the data on which operations are performed

offers many advantages in the context of cloud computing. In such a scenario, the DO would be able to transfer data to the CSP in ciphertext and operate with the data without disclosing it or having to download it in full. Even though no such capabilities currently exist in practice, they are much closer due to recent advances in cryptography. In the following subsections, we briefly review the main areas of cryptography that focus on operations on encrypted data. We explicitly exclude functional encryption, introduced in [33], which even though seems very promising, still presents many challenges and its practical applications are yet to be identified. An explanation of the differences between functional encryption and structured encryption can be found in [12].

3.3.1 Homomorphic encryption

The notion of "privacy homomorphisms" has been introduced in 1978 by Rivest et al in a context a loan company which stores sensitive information in a remote "data bank" mediated by a computer system. In the scenario, the loan company questions the information protection techniques of the data bank as the systems programmers would presumably have access to the sensitive information [34]. Needless to say, such a scenario is very relevant to today's challenges of protecting data in a cloud storage. However, in the original paper Rivest et al only assumed the existence of privacy homomorphisms, leaving the challenge to demonstrate the existence of highly secure privacy homomorphisms with a large set of operations to future research.

A landmark in this field is the fully homomorphic encryption scheme presented by Gentry, who devised an encryption scheme that allows to compute encrypted results of the data suitable for complex functions of the data without the need for the computing agent to possess the decryption key [35].

Considering the benefits of such an encryption scheme for protecting data in a cloud storage, more practical applications of homomorphic encryption to cloud storage data protection can be expected.

3.3.2 Structured encryption

Kamara and Lauter examine in "Cryptographic Cloud Storage" [16] principles of building a secure IaaS storage on top of an IaaS storage provided by an untrusted IaaS infrastructure. The model aims to provide confidentiality and integrity, while retaining main benefits of IaaS storage, namely availability, reliability, efficient retrieval and data sharing.

The authors focus on ensuring the requirements through cryptographic guarantees rather than administrative controls. The solution assumes 4 components to be deployed on the client side, namely: data processor, data verifier, credential generator and token generator. On a high level, the authors describe the use case for information upload and sharing in the following steps: **(1)** deployment of client applications, **(2)** generation of a Master Key, **(3)** upon upload, the data processor adds metadata and encrypts the data. To retrieve data **(4)**, the user uses the token generator to retrieve a token and then submit the token to the CP. Search queries are performed using the structured encryption schemes introduced in [12] and subsection 3.1

Several schemes of searchable encryption are examined, namely:

- *Symmetric searchable encryption* (SSE) , appropriate in any setting where the party that searches over the data is also the one who generates it (appropriate for SRSW settings). Important to note is that, symmetric searchable encryption requires an additional data structure, namely the index of the documents, to be created and uploaded to the server – also called a "secure index" which assists the server in the search process.

- *Asymmetric searchable encryption* (ASE) is appropriate for MWSR; however, it offers weaker security guarantees, as the server can mount a dictionary attack against the token and learn the search terms of the client.

- *Efficient asymmetric searchable encryption* (ESE) appropriate in MWSR scenarios where the search terms are hard to guess. This scheme offer more efficient search but is also vulnerable to dictionary attacks.

- *Multi-user SSE* appropriate for SWMR settings. In this scheme, the owner of the data is able to besides encrypting indexes and generating tokens to also revoke user's search privileges over data. The paper also introduces attribute-based encryption and proofs of storage, that are necessary for a secure IaaS storage solution.

Based on the principles outlined in [16], Kamara et al expand the idea of a cryptographically-secured cloud data store in [7]. The authors present CS2, a cryptographic storage system that allows to securely search, add and delete files in cloud data store deployed by an untrusted or partially trusted IaaS provider.

The proposed cryptographic storage system is based on three building blocks:

- Symmetric searchable encryption, presented in [12]

- Proofs of data possession, where a Provable Data Possession (PDP) is a protocol executed between the DO and the CE that provides data integrity guarantees regarding the files stored by DO without the need to download the proof of data possession.

- Search authenticators, which enable a server to provide correctness proofs regarding the answer to a search query submitted by the client.

Based on the above building blocks, CS2 provides six client operations, namely `SETUP, STORE, SEARCH, CHECK, ADD, DELETE`. The original paper [7] contains a detailed explanation of the above operations and of the protocol as a whole.

Another contribution of the paper is the implementation of a SSE scheme which runs in sub-linear time and is resistant to *adaptive* chosen-keyword attacks (CKA-2) (more information on CKA-2 in [36]).

An alternative and somewhat simpler approach adopted in related work [37] is to relax the security definition and leak the access pattern. However, the trade-off in that case is that the cloud service provider can eventually derive sensitive information based on the search and access patterns displayed by the clients.

3.4 Protecting data in a fully virtualized environment

Traditional OS-based encryption, while offering some degree of protection for end-user devices, does not live up to the security requirements of cloud environments where the communication between OS and the underlying hardware is mediated by a hypervisor.

An alternative approach is background encryption, which is performed by the hypervisor itself. However, installing hypervisor-based encryption modules into existing systems may potentially require conversion of OS images and modification of OS configurations. Another major problem is that commodity hypervisors may only provide limited support for background encryption, making the solution impractical.

Below is a review of several research efforts towards designing of minimalistic hypervisors that could support background disk encryption, in the same time providing the essential security and functionality requirements that traditional hypervisors do.

3.4.1 BitVisor

BitVisor (introduced in [38] and further in [20]) is a thin hypervisor based on Intel VT-x and AMD-V designed to enforce I/O device security of virtualized guests. BitVisor is non-intrusive in the sense that it can be injected into existing systems and does not require modification of OS images. In order to have a minimal impact on the performance of the guest VM instances (24% overhead on sequential disk access), the VMM uses a parapass-through architecture that allows to forward a subset of the I/O instructions (keyboard and mouse actions) without modification (depicted in Figure 4).

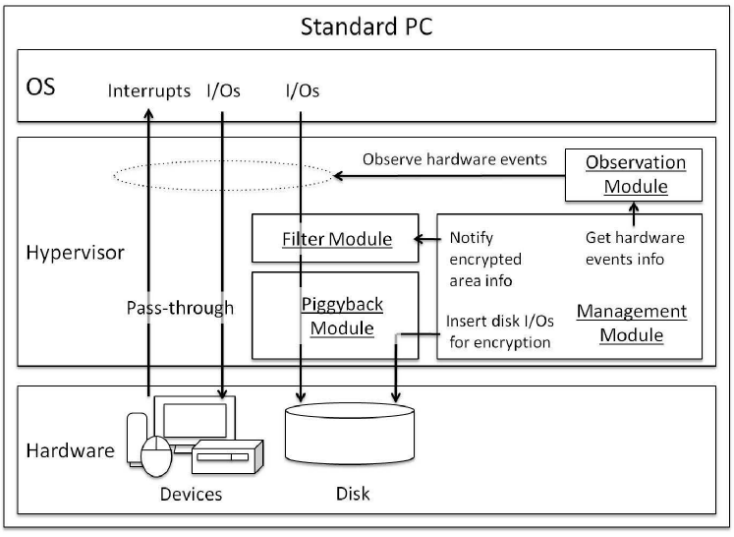

Figure 4: High-level architecture of the Bitvisor VMM model

The approach describes a method to offer access to both encrypted and unencrypted areas of the disk in a transparent manner for the VM image. Different

from other approaches, BitVisor assumes a minimal hypervisor functionality which facilitates deployment efforts.

While it introduces several novel ideas and has a code implementation that has also been extended by other researchers, BitVisor trades the ability to have concurrently executing VM instances for simplicity and ease of installation. This limitation severely reduces the applicability of BitVisor in virtualized IaaS environments, where hardware multiplexing is one of the important advantages of virtualization in general.

3.4.2 TCVisor

In [39] the authors propose a full disk background encryption model by introducing TCVisor, a BitVisor-based hypervisor with a parapass-through architecture that introduces a novel key-management approach and TPM support.

Support for TPM is added in order to store parts of cryptographic keys and whole-disk checksums for integrity checking. The authors use Merkle trees for integrity verification and store the root value "inside TPM" (*sic*). However, the exact procedure of storing or sealing the root value of the Merkle tree hash is not discussed. AES "with some tweaks added" ("sic") is used for data encryption, however the undisclosed modifications to AES raise concerns about the necessity of modifying a standard verified encryption algorithm and about the effects the of introduced modifications. The authors also examine the topic of key revocation and propose an aggressive key revocation triggered by user input. The approach suggested in the paper does indeed address some aspects of protecting privacy-sensitive data in IaaS storage. However, by building TCVisor on top of BitVisor, the model inherits the limitations of BitVisor, e.g. can only support one guest VM. In addition, the above-mentioned poorly explained modifications to AES employed by the authors are at odds with common cryptographic practices and can potentially introduce new vulnerabilities.

3.4.3 CloudVisor

Zhang et al introduce CloudVisor, a hypervisor aimed to protect the integrity and privacy of the guest virtual machines even in the event of a total compromise of the VMM by the untrusted cloud service provider [5]. In [40], the authors address the fact that software stacks in multi-tenant clouds are non-trivially large and complex, something which leads to compromise and abuse from adversaries.

In their paper, the authors introduce a security monitor underneath the commodity VMM using nested virtualization and provides protection to the hosted VMs. As a result, CloudVisor runs in host mode while the commodity VMM is "pushed out" into guest mode. However, one immediate drawback is that this approach would expose the VMM to the attacks from malicious VMs and would facilitate VM side-channel attacks.

CloudVisor interposes interactions between a VMM and its guest VMs for privacy protection and integrity checking. To protect memory owned by a VM, CloudVisor tracks memory pages of a VM and encrypts page content upon unauthorized mappings from the VMM and other VMs. Furthermore, CloudVisor encrypts the data exchange between a VM and the VMM and verifies the integrity, freshness and ordering of disk I/O data.

The model assumes a 'non-intentional' mismanagement of the IaaS environment and aims to prevent a malicious VMM from inspecting or modifying a tenant's VM states, i.e. provides secrecy and integrity to a VM's states. The design considerations of [5] are: whole-VM protection, non-intrusive with commodity VMMs and a minimized Trusted Computing Base (TCB). Some additional design considerations follow:

- Transparent system call interposition [3] using nested virtualization;
- VM-based memory ownership tracking
- I/O protection through encryption, ie. to protect virtual disks,CloudVisor transparently encrypts and decrypts data during each disk I/O access by a VM, including both port-based I/O and Direct Memory Access (DMA)
- Late Launch to Reduce CloudVisor Complexity, i.e. a cloud tenant may first authenticate the cloud platform by using TCG's attestation protocol with TPM to know if the platform is running a known version of CloudVisor.

The authors state that the storage AES key is always maintained "inside CloudVisor", without specifying what that exactly means. In certain interpretations, this opens the possibility for cold boot attacks. However, such attacks are not applicable if the assumption of physical security holds.

3.4.4 TreVisor

TreVisor [41] extends the previously mentioned BitVisor project with a patch that ensures the block storage encryption keys and context are not exposed outside the CPU in order to prevent cold boot attacks. This is achieved by extending BitVisor with TRESOR, which is a Linux kernel patch for the x64 architecture. TRESOR allows to avoid RAM usage for key management and AES cryptographic operations and instead run them entirely on the microprocessor [42]. In addition, the authors utilize VT-d/IOMMU to restrict direct memory access and thus prevent DMA attacks. In essence, while TreVisor inherits the weaknesses of BitVisor such as support for a single VM guest, it extends the BitVisor model with protection against physical attacks and could be suitable in conditions where such a trade-off is acceptable.

3.5 Novel trusted computing principles for data integrity protection

In [43], Costan and Devadas introduced the principles of augmenting cloud servers with a Trusted Computing Base (TCB) for secure high-performance computations, implemented in two paired chips. Based on the Trusted Computer Group (TCG) model and in order to increase the throughput of the Trusted Platform Module (TPM), the authors propose splitting the functionality of a TPM between a state chip with non-volatile random access memory (NVRAM) ($\mathcal{S}$) and a processing chip with high processing power and volatile memory, $\mathcal{P}$

[3] A system call interposition (SCI) support tracks all the system service requests of processes. Each system request can be modified or denied.

(see figure 5 for more details). In the proposed model, $\mathcal{P}$ is responsible for security-sensitive computations and $\mathcal{S}$ securely stores the system state and a Physical Unclonable Function (PUF) is used to bind $\mathcal{P}$ to $\mathcal{S}$ at manufacture time.

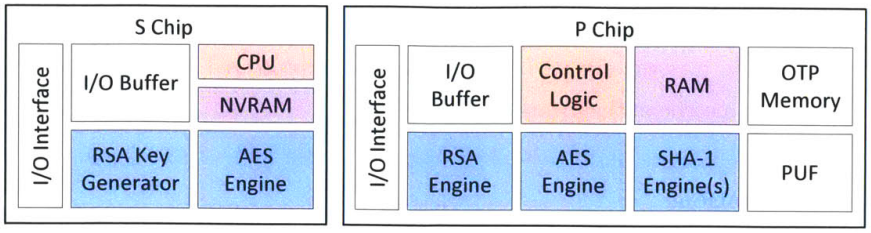

Figure 5: Functional units in S and P chips

Based on the model suggested by Costan and Devadas, Yang presents an implementation of a trusted cloud storage [44]. The author addresses the integrity and freshness requirements towards a cloud storage by using the high-throughput $\mathcal{P}$ chip to construct Merkle tree containing hashes of data blocks over the storage disk. The collision resistance property of hash functions allows the Merkle tree's root to represent the value of the entire disk. Thus, while the whole tree is stored on the untrusted server, the root hash is stored and protected in the $\mathcal{S}$ chip, while $\mathcal{P}$ calculates the initial hash value, as well as updates and stores the root hash when the system is powered.

While the approach proposed by Yang addresses cloud storage integrity and freshness requirements in a novel way, it remains an experimental work, as well as principles put forward in [43]. While both papers follow the TCG model and the principles of TPM operation, the suggested principles are not readily applicable in existing cloud environments.

3.6 Data placement and geolocation verification

The massive scale of data kept in cloud storage shed a new light on the issue of personal data mobility across jurisdictional boundaries. While legal frameworks for the European Union and the United States in this field are in an early development and adoption phase [2,3], there are a number of research initiatives that focus on both ensuring and verifying the geographical placement of data in cloud storage.

The National Institute of Standards and Technology (NIST) has described in a draft report the proof of concept implementation for trusted geolocation in the cloud [45]. The proof of concept relies on the combination of TC, Intel Trusted Execution Technology (TXT) and a set of manual audit steps in order to verify and enforce data location policies. Related research on this topic addresses specific aspects of data geolocation with somewhat more relaxed attacker models.

The authors of [22] start off with the premise that IaaS storage clients have no means of verifying that their files are not vulnerable to e.g. hard drive crashes and explore the challenge of determining whether data is stored in the cloud in a fault tolerant way. The principles presented in [22] have also been used in

later research focused in data geolocation in cloud storage. According to the model, the CSP must prove that:

- File $\mathcal{F}$ is distributed across a certain number of drives;
- $\mathcal{F}$ has been stored with a certain amount of redundancy;
- Distribution of $\mathcal{F}$ across drives is well balanced

To address these requirements, the authors propose a protocol for Remote Attestation of Fault Tolerance (RAFT) . This protocol relies on a combination of coding theory, cryptographic techniques and remote hardware profiling in order to reliably determine that a certain data is replicated across distinct drives (rather than stored in one location, vulnerable to hardware failures). RAFT's key approach is to measure response time from a cloud storage service to a read request from a special collection of file blocks.

Assuming a storage configuration of m blocks over c drives that should be able to tolerate t failures, RAFT involves an initial step where a certain file $\mathcal{F}$ is expanded into an n-block erasure code representation, where $n = mc/(c-t)$.

The client $\mathcal{C}$ verifies the file replication degree by measuring the correctness and promptness of the server's replies to $\mathcal{C}$'s challenges. A challenge in this case would consist of a read request for a subset s of block indices (where $s \subseteq n$) and expects the answer to be consistent with the initial file $\mathcal{F}$ and to be delivered within a bounded time $\mathcal{T}$.

The RAFT protocol assumes that and the DO and CSP agree on a certain mapping of file blocks to drives, in order to be able to retrieve one block per drive according to the agreed-upon layout. Block retrieval queries $\mathcal{Q}$ are based on a *lock-step* challenge generation, where the client initially specifies $\mathcal{Q}$ consisting of a set of challenge blocks with a length equal to c. Subsequent $\mathcal{Q}$s are generated adaptively based on the already accessed file blocks.

For a practical implementation, the authors rely on empirical evidence that network latency is stable enough for the purposes of RAFT on representative Internet routes. Another assumption founded on empirical studies is that the read-response time follows a probability distribution stable across time and physical file positioning on disk, "for a calibrated file-block size" [22], without specifying the choice of the block size.

The proposed adversarial model assumes an economically rational service providers that use fewer drives to reduce costs, provide substandard file redundancy or do not ensure a balanced data replication among different drives. In the same time, the Byzantine failures, i.e. purposefully induced hard drive failures are explicitly excluded from the model.

A different implication of fault tolerance and distribution of IaaS storage (both network and geographical distribution) is examined in [23, 24], which address the question of determining the physical location of data in a distributed IaaS storage.

Benson et al propose a method for determining the location of data in IaaS storage with a per-datacenter granularity [23]. The authors rely on three assumptions, namely:

- Locations of all datacenters where the CSP stores data are known;

- The CP does not have any exclusive Internet connection between the datacenters;
- For each datacenter, there is a TTP node located geographically close to it, relative to the distance between the datacenters.

The proposed method uses the Haversine distance [4] as a passive distance measurement between the datacenters to determine the location of the data center(s) where a certain piece of data is stored. In addition, the paper discusses techniques to determine the location without having the list of data centers disclosed and also detecting the changes within a location. Apart from the proposed method itself, the authors contribute with a solid overview of the cloud data geolocation approaches.

Gondree and Peterson propose a Constraints-Based Data Geolocation (CBDG) solution for determining the location of data and its "binding" to specific locations[5] [24]. The authors have similar assumptions as stated in [23] and consider the same adversary model (economically rational adversary aiming to reduce costs through data migration in spite of contractual agreements. The suggested approach combines probabilistic provable data possession with geolocation in a CBDG protocol, which builds closely on a MAC-based PDP scheme used in [22] with some additional steps. In particular, the protocol assumes an initial model building stage, where a set of landmarks $\mathcal{L}$ throughout the analyzed geographical region each build a latency-distance estimation model. Furthermore, PDP challenges (similar to the challenge queries $\mathcal{Q}$) in [22] are divided between $\mathcal{L}$. Using its latency-distance model, each landmark generates a circular constraint of a radius $r_{\mathcal{L}}$ centered on $\mathcal{L}$. The geolocation step of the protocol uses the intersection of geolocation constraints $[r_{\mathcal{L}}]$ to determine the region where the data resides.

Unfortunately, both the approaches discussed in [23] and [24] assume that the CSP does not have dedicated communication channels between the datacenters but do not provide any empirical studies that would support such an assumption.

4 Potential research topics

As mentioned in the introduction, this review included a set of papers that present relevant solutions or tools that can be used to protect data integrity, authenticity and confidentiality at different levels of a cloud data store. To complement this incomplete review, we present a selection of potential and emergent research topics in the area of cloud storage security identified based on the relevant and recent developments within cloud storage.

- **Key management and data access management** in a cloud environment receives significant interest from the research community. Some approaches also consider trusted computing mechanisms for encryption key protection and management in large-scale IaaS settings. However, integration of novel proposed key and data access management approaches

[4] The Haversine formula ($\text{haversin}(\theta) = sin^2(\frac{\theta}{2}) = \frac{1-cos(\theta)}{2}$) is used to calculate great-circle distances between two points on a sphere from their longitudes and latitudes.

[5] Binding is here used in the sense of detecting occurrences of data misplacement, rather than data binding in the meaning common in trusted computing

with industry standard protocols, such as Key Management Interoperability Protocol (KMIP) [46] could increase their relevance and should, in our opinion, receive more attention. Similarly, efficient and scalable dynamic key revocation schemes suitable for IaaS environments are an important part of key management protocols and require more attention. Finally, integration of standard key management protocols with with lightweight hypervisors supporting background encryption, such as the ones described in [38, 40, 41].

- While protection of data on block storage level and in key-value stores is examined in several papers, **identifying and developing protection mechanisms for relational database storage** requires a closer look. Given the widespread use of relational data storage and the need to minimize the TCB, there is a need to develop solutions that would ensure protection of data in an untrusted, virtualized cloud environment (maintaining in the same time the functionality of traditional RDBMS and supporting the advantages offered by the cloud storage model).

- **Scalability of data storage** is one of the cornerstone requirements of a cloud data storage. Unfortunately, even though many approaches claim a 'scalable' solution, the effects of the proposed data protection approaches on storage scalability, as well as scenarios of arbitrary host failure typical for IaaS models are not examined enough.

- **Cloud storage data location control** is yet another perspective research area, considering the advances in the field of cloud data geolocation [22, 23, 24], as well as the latest evolution of the legal framework with respect to handling and placement of privacy-sensitive data in IaaS storage environments [3]. Combining trusted computing methods with cloud data geolocation could allow to consider stricter attacker models and develop mechanisms to enforce and verify data storage policies in IaaS storage environments.

5 Conclusion

In this report we review a set of sample topics covering IaaS storage security. We do not claim a detailed overview of the field, given the diversity and volatility of topics related to IaaS storage security.

In many cases, protecting the confidentiality of data involves limiting the types of operations that can be performed on that data as well as the efficiency of data replication and synchronization, not to mention the performance impact caused by the time required for cryptographic operations. Furthermore, data access control and controlled disclosure of certain fragments or version of a data set is still a research challenge and an active research field.

This review covers several distinct topics within cloud storage security, although none of them is encompassing enough to exclusively satisfy the requirements for data protection in a CE. When discussing structured encryption and controlled disclosure of data we examined several approaches to securely disclose a part of the dataset in a cloud environment, while keeping the rest confidential. Secure operations on encrypted data address the need to go beyond symmetric

encryption of data and enable the DO to use the data processing capabilities of CE to securely perform addition, deletion or search operations on data in CS. Protection of data in host virtualized environments examines the challenges to secure data processed on virtualized compute hosts before it is transferred to remote data stores in an arbitrary location. Finally, data placement and geolocation verification deals with the emergent need to verify and ensure that data is stored within certain geographical or political boundaries in a fault-tolerant manner.

As a result of this review, we have identified a set of four high-level research directions. Among other things, they call for a closer integration between proposed Key Management Protocols and the ones used in industry, in order to facilitate creation of protection solutions that would be applicable to a wide variety of data stores while maintaining functionality. Finally, in developing data protection mechanisms in CE it is important to take into consideration the scalability, fault-tolerance and distribution principles of IaaS platforms.

References

[1] P. Mell and T. Grance, "The nist definition of cloud computing (draft)," *NIST special publication*, vol. 800, p. 145, 2011.

[2] 112th Congress (2012), "Cloud computing act of 2012, s. 3569 (112th)," 2012.

[3] E. Commission, "Proposal for a regulation of the european parliament and of the council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (general data protection regulation), com (2012) 11 final, 25 january 2012," 2012.

[4] M. Hirano, T. Shinagawa, H. Eiraku, S. Hasegawa, K. Omote, K. Tanimoto, T. Horie, S. Mune, K. Kato, T. Okuda, *et al.*, "A two-step execution mechanism for thin secure hypervisors," in *Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on*, pp. 129–135, IEEE, 2009.

[5] F. Zhang, J. Chen, H. Chen, and B. Zang, "Cloudvisor: Retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization," in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pp. 203–216, ACM, 2011.

[6] W. Wang, Z. Li, R. Owens, and B. Bhargava, "Secure and efficient access to outsourced data," in *Proceedings of the 2009 ACM workshop on Cloud computing security*, pp. 55–66, ACM, 2009.

[7] S. Kamara, C. Papamanthou, and T. Roeder, "Cs2: A searchable cryptographic cloud storage system," tech. rep., Technical Report MSR-TR-2011-58, Microsoft, 2011.

[8] S. Zarandioon, D. Yao, and V. Ganapathy, "K2c: Cryptographic cloud storage with lazy revocation and anonymous access," *Security and Privacy in Communication Networks*, pp. 59–76, 2012.

[9] S. Graf, P. Lang, S. Hohenadel, and M. Waldvogel, "Versatile key management for secure cloud storage," *Submitted at EuroSys*, vol. 11, no. 11.4, pp. 2012–13, 2012.

[10] A. Basu, I. Sengupta, and J. Sing, "Secured cloud storage scheme using ecc based key management in user hierarchy," *Information Systems Security*, pp. 175–189, 2011.

[11] Y. Tang, P. P. Lee, J. C. Lui, and R. Perlman, "Fade: Secure overlay cloud storage with file assured deletion," *Security and Privacy in Communication Networks*, pp. 380–397, 2010.

[12] M. Chase and S. Kamara, "Structured encryption and controlled disclosure," *Advances in Cryptology-ASIACRYPT 2010*, pp. 577–594, 2010.

[13] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communication security*, pp. 89–98, ACM, 2006.

[14] N. Santos, R. Rodrigues, K. Gummadi, and S. Saroiu, "Policy-sealed data: A new abstraction for building trusted cloud services," in *USENIX Security*, 2012.

[15] R. Popa, J. Lorch, D. Molnar, H. Wang, and L. Zhuang, "Enabling security in cloud storage slas with cloudproof," *Microsoft TechReport MSR-TR-2010*, vol. 46, pp. 1–12, 2010.

[16] S. Kamara and K. Lauter, "Cryptographic cloud storage," *Financial Cryptography and Data Security*, pp. 136–149, 2010.

[17] T. Robertazzi, "Advanced encryption standard (aes)," *Basics of Computer Networking*, pp. 73–77, 2012.

[18] S. Sedghi, *Towards Provably Secure Efficiently Searchable Encryption*. PhD thesis, University of Twente, 2012.

[19] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "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*, pp. 199–212, ACM, 2009.

[20] Y. Omote, Y. Chubachi, T. Shinagawa, T. Kitamura, H. Eiraku, and K. Matsubara, "Hypervisor-based background encryption," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pp. 1829–1836, ACM, 2012.

[21] P. Dewan, D. Durham, H. Khosravi, M. Long, and G. Nagabhushan, "A hypervisor-based system for protecting software runtime memory and persistent storage," in *Proceedings of the 2008 Spring simulation multiconference*, pp. 828–835, Society for Computer Simulation International, 2008.

[22] K. Bowers, M. van Dijk, A. Juels, A. Oprea, and R. Rivest, "How to tell if your cloud files are vulnerable to drive crashes," in *Proceedings of the 18th ACM conference on Computer and communications security*, pp. 501–514, ACM, 2011.

[23] K. Benson, R. Dowsley, and H. Shacham, "Do you know where your cloud files are?," in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pp. 73–82, ACM, 2011.

[24] M. Gondree and Z. N. Peterson, "Geolocation of data in the cloud," in *Proceedings of the third ACM conference on Data and application security and privacy*, pp. 25–36, ACM, 2013.

[25] M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. Weippl, "Dark clouds on the horizon: Using cloud storage as attack vector and online slack space," in *USENIX Security*, vol. 8, 2011.

[26] T. Pulls, "(more) side channels in cloud storage," *Privacy and Identity Management for Life*, pp. 102–115, 2012.

[27] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina, "Controlling data in the cloud: outsourcing computation without outsourcing control," in *Proceedings of the 2009 ACM workshop on Cloud computing security*, pp. 85–90, ACM, 2009.

[28] C. Wang, K. Ren, W. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," *Network, IEEE*, vol. 24, no. 4, pp. 19–24, 2010.

[29] Y. Chen, V. Paxson, and R. Katz, "What's new about cloud computing security?," *University of California, Berkeley Report No. UCB/EECS-2010-5 January*, vol. 20, no. 2010, pp. 2010–5, 2010.

[30] Microsoft, "Windows azure service disruption from expired certificate." `http://blogs.msdn.com/b/windowsazure/archive/2013/02/24/windows-azure-service-disruption-from-expired-certificate.aspx`, February 2013.

[31] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner, "The versakey framework: Versatile group key management," *Selected Areas in Communications, IEEE Journal on*, vol. 17, no. 9, pp. 1614–1631, 1999.

[32] S. Brunozzi, "Big data and nosql with amazon dynamodb," in *Proceedings of the 2012 workshop on Management of big data systems*, pp. 41–42, ACM, 2012.

[33] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," *Theory of Cryptography*, pp. 253–273, 2011.

[34] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[35] C. Gentry, "Computing arbitrary functions of encrypted data," *Communications of the ACM*, vol. 53, no. 3, pp. 97–105, 2010.

[36] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in Cryptology-Eurocrypt 2004*, pp. 506–522, Springer, 2004.

[37] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, pp. 253–262, IEEE, 2010.

[38] T. Shinagawa, H. Eiraku, K. Tanimoto, K. Omote, S. Hasegawa, T. Horie, M. Hirano, K. Kourai, Y. Oyama, E. Kawai, *et al.*, "Bitvisor: a thin hypervisor for enforcing i/o device security," in *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, pp. 121–130, ACM, 2009.

[39] M. Rezaei, N. Moosavi, H. Nemati, and R. Azmi, "Tcvisor: A hypervisor level secure storage," in *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*, pp. 1–9, IEEE, 2010.

[40] F. Zhang, J. Chen, H. Chen, and B. Zang, "Cloudvisor: Retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization," in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pp. 203–216, ACM, 2011.

[41] T. Müller, B. Taubmann, and F. Freiling, "Trevisor," in *Applied Cryptography and Network Security*, pp. 66–83, Springer, 2012.

[42] T. Müller, F. C. Freiling, and A. Dewald, "Tresor runs encryption securely outside ram," in *Proceedings of the 20th USENIX conference on Security*, pp. 17–17, USENIX Association, 2011.

[43] V. Costan and S. Devadas, "Security challenges and opportunities in adaptive and reconfigurable hardware," in *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on*, pp. 1–5, IEEE, 2011.

[44] H. Yang, *Efficient trusted cloud storage using parallel, pipelined hardware*. PhD thesis, Massachusetts Institute of Technology, 2012.

[45] E. Banks, M. Bartock, K. Firftal, D. Lemon, K. Scarfone, U. Shetty, M. Souppaya, T. Williams, and R. Yeluri, "Draft trusted geolocation in the cloud: Proof of concept implementation," *NIST special publication*, vol. 7904, p. 42, 2012.

[46] OASIS Key Management Interoperability Protocol Technical Committee, "Key management interoperability protocol," *Draft 0.98*, vol. 3, 2009.