ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 4, Issue 10, October 2017

Providing User Security Guarantees in Public Infrastructure Clouds

[1] Amtul Quddoos [2] Dr.Md Ateeq-Ur-Rahman
[1]CSE Department, professor, Department of CSE, Shadan College of Engineering
[2] professor, Department of CSE, Shadan College of Engineering

***Abstract:*-- One such mechanism is platform integrity verification for figure hosts that support the virtualized cloud infrastructure. Several giant cloud vendors have signaled sensible implementations of this mechanism, primarily to safe guard the cloud infrastructure from corporate executive threats and advanced persistent threats. We tend to see 2 major improvement vectors regarding these implementations. First, details of such proprietary solutions aren't disclosed and may therefore not be enforced and improved by alternative cloud platforms. Second, to the most effective of our information, none of the solutions provides cloud tenants a symbol concerning the integrity of figure hosts supporting their slice of the cloud infrastructure. To address this, we tend to propose a group of protocols for sure launch of virtual machines, which give tenants with a symbol that the requested instances were launched on a number with associate degree expected code stack.**

***Keywords:*- Security; Cloud Computing; Storage Protection; Trusted Computing.**

I. INTRODUCTION

Cloud computing has progressed from a bold vision to massive deployments in various application domains. However, the complexity of technology underlying cloud computing introduces novel security risks and challenges. Threats and mitigation techniques for the IaaS model have been under intensive scrutiny in recent years [1], [2], [3], [4], while the industry has invested in enhanced security solutions and issued best practice recommendations [5]. From an end-user point of view the security of cloud infrastructure implies unquestionable trust in the cloud provider, in some cases corroborated by reports of external auditors. While providers may offer security enhancements such as protection of data at rest, end-users have limited or no control over such mechanisms. There is a clear need for usable and cost-effective cloud platform security mechanisms suitable for organizations that rely on cloud infrastructure.

One such mechanism is platform integrity verification for compute hosts that support the virtualized cloud infrastructure. Several large cloud vendor shave signalled practical implementations of this mechanism, primarily to protect the cloud infrastructure from insider threats and advanced persistent threats. We see two major improvement vectors regarding these implementations. First, details of such proprietary solutions are not disclosed and can thus not be implemented and improved by other cloud platforms. Second, to the best of our knowledge, none of the solutions provides cloud tenants a proof regarding the integrity of compute hosts supporting their slice of the cloud infrastructure. To address this, we proposes set of protocols fort rusted launch of virtual machines (VM) in IaaS, which provide tenants with a proof that the requested VM instances were launched on a host with an expected software stack.

Another relevant security mechanism is encryption of virtual disk volumes, implemented and enforced at compute host level. While support data encryption at rest is offered by several cloud providers and can be configured by tenants in their VM instances, functionality and migration capabilities of such solutions are severely restricted. In most cases cloud providers maintain and manage the keys necessary for encryption and decryption of data at rest. This further convolutes the already complex data migration procedure between different cloud providers, disadvantaging tenants through a new variation of vendor lock-in. Tenants can choose to encrypt data on the operating system (OS) level within their VM environments and manage the encryption keys themselves. This approach allows easy migration of encrypted data volumes and withdraws the control of the cloud provider over disk encryption keys. In addition, DBSP significantly reduces the risk of exposing encryption keys and keeps a low maintenance overhead for the tenant – in the same time providing additional control over the choice of the compute host based on its software stack.

We focus on the Infrastructure-as-a-Service model – in a simplified form, it exposes to its tenants a coherent platform supported by compute hosts which operate VM guests that communicate through a virtual network. The system model chosen for this paper is based on requirements identified while migrating a currently deployed, distributed electronic health record (EHR) system to an IaaS platform [6].

1.1 Contribution We extend previous work applying Trusted Computing to strengthen IaaS security, allowing tenants to place hard security requirements on the infrastructure and maintain exclusive control of the security critical assets. We propose a security framework consisting of three buiding blocks:

- Protocols for trusted launch of VM instances in IaaS;
- Key management and encryption enforcement functions for VMs, providing transparent encryption of persistent data storage in the cloud;
- Key management and security policy enforcement by a Trusted Third Party (TTP);

We describe several contributions that enhance cloud infrastructure with additional security mechanisms:

1. We describe a trusted VM launch (TL) protocol which allows tenants – referred to as domain managers – to launch VM instances exclusively on hosts with an attested platform configuration and reliably verify this.
2. We introduce a domain-based storage protection protocol to allow domain managers store encrypted data volumes partitioned according to administrative domains.
3. We introduce a list of attacks applicable to IaaS environments and use them to develop protocols with desired security properties, perform their security analysis and prove their resistance against the attacks.
4. We describe the implementation of the proposed protocols on an open-source cloud platform and present extensive experimental results that demonstrate their practicality and efficiency.

1.2 Organization The rest of this paper is organized as follows. In Section 2 we describe relevant related work on trusted virtual machine launch and cloud storage protection. In Section 3 we introduce the system model, as well as the threat model and problem statement. In Section 4 we introduce the protocol components, and the TL and DBSP protocols as formal constructions. In Section 5, we provide a security analysis and prove the resistance of the protocols against the defined attacks, while implementation and performance evaluation results are described in Section 6. We discuss the protocol application domain in Section 7 and conclude in Section 8.

2. RELATED WORK

We start with a review of related work on trusted VM launch, followed by storage protection in IaaS.

2.1 Trusted Launch Santos et al. [1] proposed a “Trusted Cloud Compute Platform” (TCCP) to ensure VMs are running on a trusted hardware and software stack on a remote and initially untrusted host. To enable this, a trusted coordinator stores the list of attested hosts that run a “trusted virtual machine monitor” which can securely run the client’s VM.
Trusted hosts maintain in memory an individual trusted key used for identification each time a client launches a VM. The paper presents a good initial set of ideas for trusted VM launch and migration, in particular the use of a trusted coordinator. A limitation of this solution is that the trusted coordinator maintains information about all hosts deployed on the IaaS platform, making it a valuable target to an adversary who attempts to expose the public IaaS provider to privacy attacks.

A decentralized approach to integrity attestation is adopted by Schiffman et al. [2] to address the limited transparency of IaaS platforms and scalability limits imposed by third party integrity attestation mechanisms. The authors describe a trusted architecture where tenants verify the integrity of IaaS hosts through a trusted cloud verifier proxy placed in the cloud provider domain. Tenants evaluate the cloud verifier integrity, which in turn attests the hosts. Once the VM image has been verified by the host and countersigned by the cloud verifier, the tenant can allow the launch. Our protocol maintains the VM launch traceability and transparency without relying on a proxy verifier residing in the IaaS. Furthermore, the TL protocol does not require additional tenant interaction to launch the VM on a trusted host, beyond the initial launch arguments.

Platform attestation prior to VM launch is also applied in [7], which introduces two protocols – “TPM-based certification of a Remote Resource” (TCRR) and “Verify My VM”. With TCRR a tenant can verify the integrity of a remote host and establish a trusted channel for further communication. In “Verify My VM”, the hypervisor running on an attested host uses an emulated TPM to verify on-demand the integrity of running VMs. Our approach is in many aspects similar to the one in [7] in particular with regard to host attestation prior to VM instance launch. We overcome this limitation and generalize the solution by adding a verification token, created by the tenant and injected on the file system of the VM instance only if it is launched on an attested cloud host.

In [8], the authors described a protocol for trusted VM launch on public IaaS using trusted computing techniques. To ensure that the requested VM instance is launched on a host with attested integrity, the tenant encrypts the VM image (along with all injected data) with a symmetric key sealed to a particular configuration of the host reflected in the values of the platform configuration registers (PCR) of the TPM placed on the host. The proposed solution is suitable in trusted VM launch scenarios for enterprise tenants as it requires that the VM image is pre-packaged and encrypted by the client prior to IaaS launch. However, similar to [7], this prevents tenants from using commodity VM images offered by the cloud provider to launch VM instances on trusted cloud hosts. Furthermore, we believe that reducing the number of steps required from the tenant can facilitate the adoption of the trusted IaaS model. We extend some of the ideas proposed in [8], address the above limitations – such as additional actions required from tenants – and also address the requirements towards the launched VM instance and required changes to cloud platforms.

2.2 Secure Storage

Cooper at al described in in [9] a secure platform architecture based on a secure root of trust for grid environments – precursors of cloud computing. Trusted Computing is used as a method for dynamic trust establishment within the grid, allowing clients to verify that their data will be protected against malicious host attacks. The authors address the malicious host problem in grid environments, with three main risk factors: trust establishment, code isolation and grid middleware. We follow a similar approach in terms of interacting with a minimal TCB for protocol purposes following host attestation. However, in order to adapt to the cloud computing model we delegate the task of host attestation to an external TTP as well as use TPM functionality to ensure that sensitive cryptographic material can only be accessed on a particular attested host.

In [10], the authors proposed an approach to protect access to outsourced data in an owner-write-users-read case, assuming an “honest but curious service provider”. Encryption is done over (abstract) blocks of data, with a different key per block. The authors suggest a key derivation hierarchy based on a public hash function, using the hash function result as the encryption key. While this solution is similar to our model with regard to information blocks and encryption with different symmetric keys, we propose an active revocation model, where the keys are cached for a limited time and cannot be retrieved once the access is revoked.

The “Data-Protection-as-a-Service” (DPaaS) platform [11] balances the requirements for confidentiality and privacy with usability, availability and maintainability. DPaaS focuses on shareable logical data units, confined in isolated partitions (e.g. VMs of language-based features such as Caja, Javascript) or containers, called Secure Execution Environments (SEE). Data units are encrypted with symmetric keys and can be stored on untrusted hardware, while containers communicate through authenticated channels. The authors stress the verifiability of DPaaS using trusted computing and the use of the dynamic root of trust to guarantee that computation is performed on a “secure” platform. The authors posit that DPaaS fulfills confidentiality and privacy requirements and facilitates maintenance, logging and audit; We address the above shortcomings, by describing in detail and evaluating protocols to create and share confidentiality protected data blocks. We describe cloud storage security mechanisms that allow easy data migration between providers without affecting its confidentiality.

Grafet al. [12] presented an IaaS storage protection scheme addressing access control. The authors analyse access rights management of shared versioned encrypted data on cloud infrastructure for a restricted group and propose a scalable and flexible key management scheme. Access rights are represented as a graph, making a distinction between data encryption keys and encrypted updates on the keys and enabling flexible join/leave client operations, similar to properties presented by the protocols in this paper. Despite its advantages, the requirement for client-side encryption limits the applicability of the scheme in [12] and introduces important functional limitations on indexing and search. In our model, all cryptographic operations are performed on trusted IaaS compute hosts, which are able to allocate more computational resources than client devices.

Santos et al. [13] proposed Excalibur, a system using trusted computing mechanisms to allow decrypting client data exclusively on nodes that satisfy a tenant-specified policy. Excalibur introduces a new trusted computing abstraction, policy-sealed data to address the fact that TPM abstractions are designed to protect data and secrets on a standalone machine, at the same time over-exposing the cloud infrastructure by revealing the identity and software fingerprint of individual cloud hosts. The authors extended TCCP [1] to address the limitations of binary-based attestation and data sealing by using property-based attestation [14]. The core of Excalibur is 'the monitor', which is a part of the cloud provider, which organises computations across a series of hosts and provides guarantees to tenants. Tenants first decide a policy and receive evidence regarding the status of the monitor along with a public encryption key, and then encrypt their data and policy using ciphertext policy attribute based encryption [15]. Todecrypt, the stored data hosts receive the decryption key from the monitor who ensures that the corresponding host has a valid status and satisfies the policy specified by the client at encryption time. Our solution is similar to the one in [13], with some important differences: 1) In contrast with [13] our protocols were implemented as a code extension for Open stack. Furthermore, the presented measurements were made after we deployed the protocols for a part of the Swedish electronic health records management system in an infrastructure cloud. Thus, our measures are considered as realistic since the experiments were done under a real electronic healthcare system; 2) Excalibur is totally missing a security analysis. Instead authors only present the results of Pro Verif (an automated tool) regarding the correctness of their protocol. In addition to that, through our security analysis we introduced a new list of attacks that can be applied to such systems. This is something that is totally missing from related works such as [13] and it can be considered as a great contribution to protocol designers since can avoid common pitfalls and design even better protocols in the future; In [16] the authors presented a forward-looking design of a cryptographic cloud storage built on an untrusted IaaS infrastructure. The approach aims to provide confidentiality and integrity, while retaining the benefits of cloud storage – availability, reliability, efficient retrieval and data sharing – and ensuring security through cryptographic guarantees rather than administrative controls. The solution requires four client-side components: data processor, data verifier, credential generator, token generator. Important building blocks of the solution are: Symmetric searchable encryption (SSE), appropriate in settings where the data consumer is also the one who generates it (efficient for single writer-single reader (SWSR) models); Asymmetric searchable encryption (ASE), appropriate for many writer single reader (MWSR) models, offers weaker security guarantees as the server can mount a dictionary attack against the token and learn the search terms of the client; In such a scheme, the message can only be decrypted only if the policy matches the key used to encrypt it; finally, proofs of storage allow a client to verify that data integrity has not been violated by the server.

The concepts presented in [16] are promising – especially considering recent progress in searchable encryption schemes [18]. Indeed, integrating searchable and attribute based encryption mechanisms into secure storage solutions is an important direction in our future work. However, practical application of searchable encryption and attribute based encryption requires additional research.

Earlier work in [19], [20] described interoperable solutions towards trusted VM launch and storage protection in IaaS. We extend them to create an integrated framework that builds a trust chain from the domain manager to the VM instances and data in their administrative domain, and provide additional details, proofs and performance evaluation.

3. SYSTEM MODEL AND PRELIMINARIES

In this section we describe the system and threat model, as well as present the problem statement.
3.1 System Model We assume an IaaS system model (e.g. Open Stack, a popular open-source cloud platform) as in [21]: providers expose a quota of network, computation and storage resources to its tenants – referred to as domain managers (Figure 1). Domain managers utilize the quota to launch and operate VM guests.

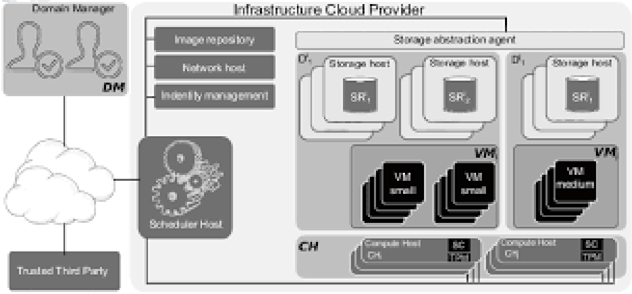


Fig. 1. High level view of the IaaS model introduced in Section 3.

Requests for operations on VMs (launch, migration, termination, etc.) received by the IaaS are managed by a scheduler that allocates (reallocates, deallocates) resources from the pool of available compute hosts according to a resource management algorithm. We assume in this work compute hosts that are physical – rather than virtual – servers. We denote the set of all compute hosts as CH =

{CH1,...,CHn}. We denote a VM instance vmi l running on a compute host CHi by vmi l 7→ CHi and its unique identifier by idvmi l.

The Security Profile (SP) , defined in *19+, is a function of the verified and measured deployment of a trusted computing base – a collection of software components measurable during a platform boot. Measurements are maintained in protected storage, usually located on the same platform. We expand this concept in Section 4.

I/O virtualization enables device aggregation and allows to combine several physical devices into a single logical device (with better properties), presented to a VM [22]. Cloud platforms use this to aggregate disparate storage devices into highly available logical devices with arbitrary storage capacity (e.g. volumes in Open Stack). VMs are presented with a logical device through a single access interface, while replication, fault-tolerance and storage aggregation are hidden in the lower abstraction layers. We refer to this logical device as storage resource (SR); as a storage unit, an SR can be any unit supported by the disk encryption subsystem.

3.2 Threat Model

We share the threat model with [1], [19], [20], [8], which is based on the Dolev-Yao adversarial model [23] and further assumes that privileged access rights can used by a remote adversary ADV to leak confidential information. ADV ,e.g. a corrupted system administrator, can obtain remote access to any host maintained by the IaaS provider, but cannot access the volatile memory of guest VMs residing on the compute hosts of the IaaS provider. This property is based on the closed-box execution environment for guest VMs, as outlined in Terra [24] and further developed in [25], [26].

3.3 Problem Statement

The introduced ADV has far-reaching capabilities to compromise IaaS host integrity and confidentiality. We define a set of attacks available to ADV in the above threat model. Given that ADV has full control over the network communication within the IaaS, one of the available attacks is to inject a malicious program or back door into the VM image, prior to instantiation. Once the VM is launched and starts processing potentially sensitive information, the malicious program can leak data to an arbitrary remote location without the suspicion of the domain manager. In this case, the VM instance will not be a legitimate instance and in particular not the instance the domain manager intended to launch. We call this type of attack a VM Substitution Attack:

A more complex attack involves reading or modifying the information processed by the VM directly, from the logs and data stored on CH or from the representation of the guest VMs' drives on the CH file system. This might be non-trivial or impossible with strong security mechanisms deployed on the host; however, ADV may attempt to circumvent this through a so-called CH Substitution Attack – by launching the guest VM on a compromised CH.

Depending on the technical expertise of DMi, ADV may still take the risk of deploying a concealed – but feature rich – malicious program in the guest VM and leave a fall back option in case the malicious program is removed or prevented from functioning as intended. ADV may choose a combined VM and CH substitution attack, which allows a modified VM to be launched on a compromised host and present it to DMi as the intended VM:.

Denote by Divm the set of storage domains that vm ∈VM, vm 7→ CHi can access. We define a successful storage compute host substitution attack as follows1:

If access to the data storage resource is given to all VMs launched by DMi, ADV may attempt to gain access by launching a VM that appears to have been launched by DMi. Then, ADV would be able to leak data from the domain owned by DMi to other domains. This infrastructure-level attack would not be detected by DMi and requires careful consideration. A formal definition of the attack1 follows.

4. PROTOCOL DESCRIPTION

We now describe two protocols that constitute the core of this paper's contribution. These protocols are successively applied to deploy a cloud infrastructure providing additional user guarantees of cloud host integrity and storage security. For protocol purposes, each domain manager, secure component and trusted third party has a public/private key pair (pk/sk). The private key is kept secret, while the public key is shared with the community. We assume that during the initialization phase, each entity obtains a certificate via a trusted certification authority. We first describe the cryptographic primitives used in the proposed protocols, followed by definitions of the main protocol components.
4.1 Cryptographic Primitives The set of all binary strings of length n is denoted by{0,1}n, and the set of all finite binary

strings as ,0,1-∗. Given a set U, we refer to the ith element as vi. Additionally, we use the following notations for cryptographic operations throughout the paper:

4.2 Protocol Components

Disk encryption subsystem: a software or hardware component for data I/O encryption on storage devices, capable to encrypt storage units such as hard drives, software RAID volumes, partitions, files, etc.
We assume a softwarebased subsystem, such as dm-crypt, a disk encryption subsystem using the Linux kernel Crypto API.

4.3 Trusted Launch Construction

We now present our construction for the TL, with four participating entities: domain manager, secure component, trusted third party and cloud provider (with the 'scheduler' as part of it). TL comprises a public-key encryption scheme, a signature scheme and a token generator. Figure 2 shows the protocol message flow (some details omitted for clarity). TL.Setup : Each entity obtains a public/private key pair and publishes its public key. Below we provide the list of key pairs used in the following protocol:

5. SECURITY ANALYSIS

We now analyse the TL and DBSP protocols in the presence of an adversary. We prove the security of both schemes 4 Key retrieval is currently not covered in the security analysis due to space limitations through a theoretical analysis, showing that our protocols are resistant to the attacks presented in Section 3.3. Option a can only succeed if ADV can break the mutual authentication in the secure channel setting. Given that the selected secure channel scheme is sound and τ is sufficiently long and selected using a sound random generation process, the ADV fails to break the last protocol step. Hence, as long as the secure channel protocol is sound, the overall protocol construction is also sound against this attack option. Option b can only succeed if the adversary either managestoguessavalue$\tau0 = \tau$ when launching vm or manages to either obtain τ when DMi launches vmi l or replace the association between τ and vmi l with an association between τ and vm when DMi launches vmi l, by attacking any of the protocol steps preceding the final mutual authentication step. A successful attack in this case has the probability $\tau0 = \tau$ equals to 1/2n, where n is the length of the token value and is infeasible if n is large enough. Below, we show why the adversary also fails with respect to the last option.

6. IMPLEMENTATION AND RESULTS

We next describe the implementation of the TL and DBSP protocols followed by experimental evaluation results.

6.1 Test bed Architecture

We describe the infrastructure of the prototype and the architecture of a distributed EHR system installed and configured over multiple VM instances running on the test bed.

6.2 Performance evaluation

DBSP Processing time: Table 1 shows a breakdown of the time required to process a storage unlock request, an average of 10 executions. Processing a volume unlock request on the prototype returns in≈2.714 seconds; however, this operation is performed only when attaching the volume to a VM instance and does not affect the subsequent I/O operations on the volume. A closer view highlights the share of the contributing components in the overall overhead composition. Table 1 clearly shows that the TPM unseal operation lasts on average ≈2.7 seconds, or 99.516% of the execution time. According to Section 4.2, in this prototype we use TPMs v1.2, since a TPM v2.0 is not available on commodity platforms at the time of writing. Given that the vast majority of the execution time is spent in the TPM unseal operation, implementing the protocol with a TPM v2.0 may yield improved results.

7. APPLICATION DOMAIN

The presented results are based on work in collaboration with a regional public healthcare authority to address some of its concerns regarding IaaS security. We have deployed the prototype described in Section 6, further extended by integrating a medication database, and evaluated it through end-user validation and performance tests. Our results demonstrate that it is both possible and practical to provide strong platform software integrity guarantees to IaaS tenants and efficiently isolate their data using established cryptographic tools. Platform integrity guarantees allow tenants to take better decisions on both workload migration to the cloud and workload placement within IaaS. This contrasts with the current, "flat" trust model, where all IaaS hosts declare the same–but unverifiable for the tenant–trust level.

On the practical side, specifically regarding the role of the TTP, we envision two scenarios. The TTP could either be managed by the tenant itself (for organizations with enough resources and expertise), or by an external organization (similar to a certificate authority). The first scenario allows the tenant to retain the benefits of cloud services along with additional security guarantees. Similarly, in the second scenario, smaller actor scan obtain the same benefits without the need to invest into own attestation infrastructure. In both scenarios, in order to protect the cloud provider the TTP would only operate on a physical slice of the resources (i.e. a subset of compute hosts) that correspond to the respective tenant domains.

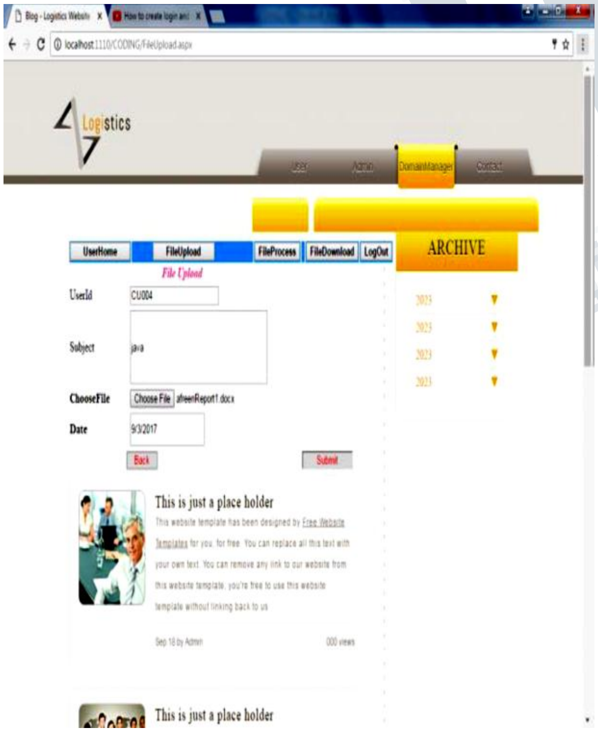
8. RESULTS

GENERAL

Snapshot is nothing but every moment of the application while running. It gives the clear elaborated of application. It will be useful for the new user to understand for the future steps.

VARIOUS SNAPSHOTS

FILE UPLOAD



localhost:1110/CODING/FileUpload.aspx

File Upload

UserId: CU004

Subject: java

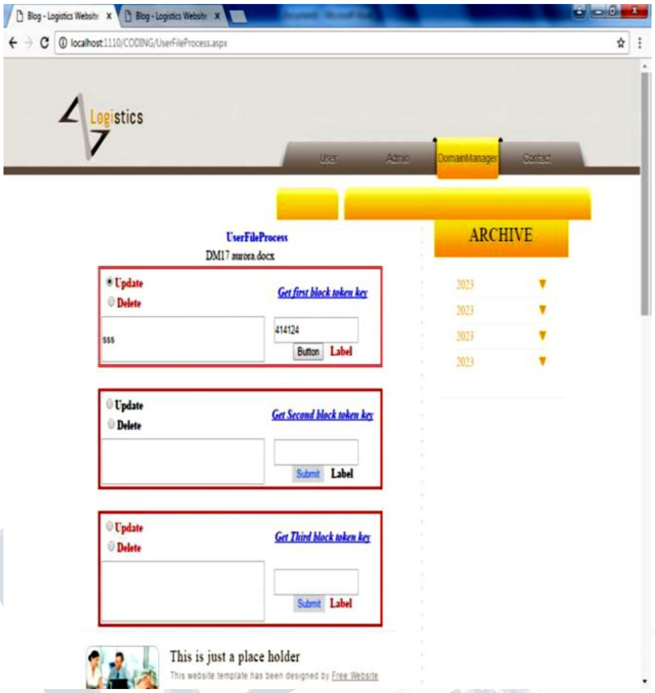
ChooseFile: Choose File afreenReport1.docx

Date: 9/3/2017

Blog - Logistics Website
How to create login and
Logistics
User
Admin
DomainManager
Contact
UserHome
FileUpload
FileProcess
FileDownload
LogOut
ARCHIVE
2023
2023
2023
2023
Back
Submit
This is just a place holder
This website template has been designed by Free Website Templates for you, for free. You can replace all this text with your own text. You can remove any link to our website from this website template, you're free to use this website template without linking back to us
Sep 18 by Admin
000 views
This is just a place holder

The above fig shows the design of image cropping.

FILE PROCESS



localhost:1110/CODING/UserFileProcess.aspx

UserFileProcess

DM17 aurora.docx

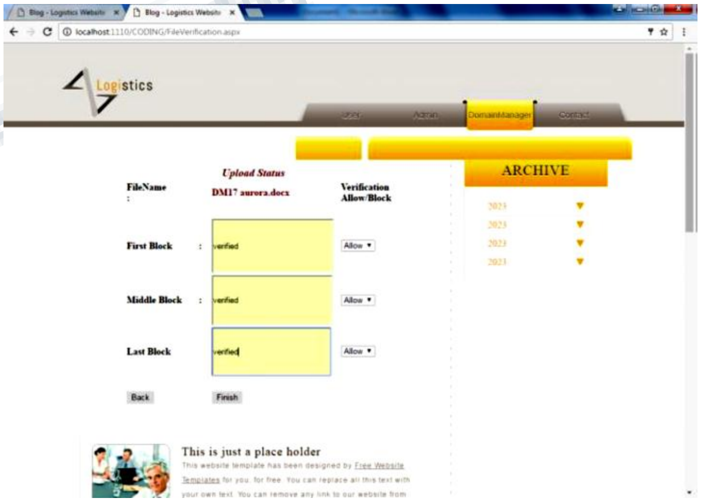
● Update
○ Delete
Get first block token key
sss
414124
Button Label

○ Update
○ Delete
Get Second block token key
Submit Label

○ Update
○ Delete
Get Third block token key
Submit Label

The above fig shows File process.

MANAGER FILE VERIFICATION



localhost:1110/CODING/FileVerification.aspx

Upload Status

FileName : DM17 aurora.docx — Verification Allow/Block

First Block : verified — Allow

Middle Block : verified — Allow

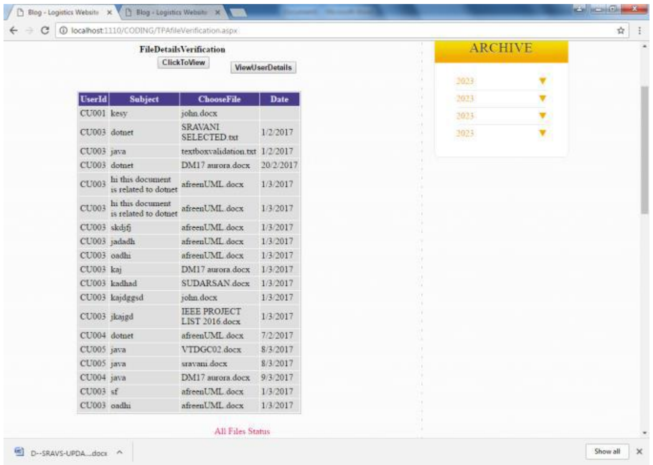
Last Block verified — Allow

Back Finish

Blog - Logistics Website
Blog - Logistics Website
Logistics
User
Admin
DomainManager
Contact
ARCHIVE
2023
2023
2023
2023
This is just a place holder
This website template has been designed by Free Website Templates for you, for free. You can replace all this text with your own text. You can remove any link to our website from

The above fig shows Manager verification.

FILE DETAILS



FileDetailsVerification

UserId	Subject	ChooseFile	Date
CU001	kesy	john.docx	
CU003	dotnet	SRAVANI SELECTED.txt	1/2/2017
CU003	java	textboxvalidation.txt	1/2/2017
CU003	dotnet	DM17 aurora.docx	20/2/2017
CU003	hi this document is related to dotnet	afreenUML.docx	1/3/2017
CU003	hi this document is related to dotnet	afreenUML.docx	1/3/2017
CU003	skdjfj	afreenUML.docx	1/3/2017
CU003	jadadh	afreenUML.docx	1/3/2017
CU003	oadhi	afreenUML.docx	1/3/2017
CU003	kaj	DM17 aurora.docx	1/3/2017
CU003	kadhad	SUDARSAN.docx	1/3/2017
CU003	kajdggsd	john.docx	1/3/2017
CU003	jkajgd	IEEE PROJECT LIST 2016.docx	1/3/2017
CU004	dotnet	afreenUML.docx	7/2/2017
CU005	java	VTDGC02.docx	8/3/2017
CU005	java	sravani.docx	8/3/2017
CU004	java	DM17 aurora.docx	9/3/2017
CU003	sf	afreenUML.docx	1/3/2017
CU003	oadhi	afreenUML.docx	1/3/2017

All Files Status

The above fig shows File details.

9. CONCLUSION

From a tenant point of view, the cloud security model does not yet hold against threat models developed for the traditional model where the hosts are operated and used by the same organization. However, there is a steady progress towards strengthening the IaaS security model. In this work we presented a framework for trusted infrastructure cloud deployment, with two focus points: VM deployment on trusted compute hosts and domain-based protection of stored data. We described in detail the design, implementation and security evaluation of protocols for trusted VM launch and domain-based storage protection. The solutions are based on requirements elicited by a public healthcare authority, have been implemented in a popular open-source IaaS platform and tested on a prototype deployment of a distributed EHR system. In the security analysis, we introduced a series of attacks and proved that the protocols hold in the specified threat model. To obtain further confidence in the semantic security properties of the protocols, we have modelled and verified them with Pro Verif [32]. Finally, our performance tests have shown that the protocols introduce a insignificant performance overhead.

This work has covered only a fraction of the IaaS attack landscape. Important topics for future work are strengthening the trust model in cloud network communications, data geo location [33], and applying searchable encryption schemes to create secure cloud platform software integrity guarantees for tenants and efficiently isolate their data using established cryptographic tools. With reasonable engineering effort the framework can be integrated into production environments to strengthen their security properties. storage mechanisms. Our results show that it is possible and practical to provide strong.

REFERENCES

1. N. Santos, K. P. Gummadi, and R. Rodrigues, “Towards trusted cloud computing,” in Proceedings of the 2009 Conference on Hot Topics in Cloud Computing, HotCloud’09, (Berkeley, CA, USA), USENIX Association, 2009.

2. J.Schiffman, T.Moyer, H.Vijayakumar, T.Jaeger, and P.McDaniel, “Seeding Clouds With Trust Anchors,” in Proceedings of the 2010 ACM Workshop on Cloud Computing Security, CCSW ’10, (New York, NY, USA), pp. 43–46, ACM, 2010.

3. N. Paladi, A. Michalas, and C. Gehrmann, “Domain based storage protection with secure access control for the cloud,” in Proceedings of the 2014 International Workshop on Security in Cloud Computing, ASIACCS ’14, (New York, NY, USA), ACM, 2014.

4. M. Jordon, “Cleaning up dirty disks in the cloud,” Network Security, vol. 2012, no. 10, pp. 12–15, 2012.

5. Cloud Security Alliance, “The notorious nine cloud computing top threats 2013,” February 2013.

6. A. Michalas, N. Paladi, and C. Gehrmann, “Security aspects of e-health systems migration to the cloud,” in the 16th International Conference on E-health Networking, Application & Services (Healthcom’14), pp. 228–232, IEEE, Oct 2014.

7. B. Bertholon, S. Varrette, and P. Bouvry, “Certicloud: a novel tpm based approach to ensure cloud IaaS security,” in Cloud Computing, 2011 IEEE International Conference on, pp. 121–130, IEEE, 2011. 10

8. M. Aslam, C. Gehrmann, L. Rasmusson, and M. Bj¨orkman, “Securely launching virtual machines on trustworthy platforms in a public cloud - an enterprise’s perspective.,” in CLOSER, pp. 511– 521, SciTePress, 2012.

9. A. Cooper and A. Martin, “Towards a secure, tamper-proof grid platform,” in Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on, vol. 1, pp. 8–pp, IEEE, 2006.

10. W. Wang, Z. Li, R. Owens, and B. Bhargava, “Secure and efficient access to out sourced data,”inProceedingsofthe2009ACMworkshop on Cloud computing security, pp. 55–66, ACM, 2009.

11. D. Song, E. Shi, I. Fischer, and U. Shankar, “Cloud data protection for the masses,” IEEE Computer, vol. 45, no. 1, pp. 39–45, 2012.

12. S. Graf, P. Lang, S. A. Hohenadel, and M. Waldvogel, “Versatile key management for secure cloud storage,” in Proceedings of the 2012 IEEE 31st Symposium on Reliable Distributed Systems, pp. 469– 474, IEEE Computer Society, 2012.

13. N. Santos, R. Rodrigues, K. P. Gummadi, and S. Saroiu, “PolicySealed Data: A New Abstraction for Building Trusted Cloud Services,” in Presented as part of the 21st USENIX Security Symposium (USENIX Security 12), (Bellevue, WA), pp. 175–188, USENIX, 2012.

14. A.-R. Sadeghi and C. St´uble, “Property-based attestation for computing platforms: Caring about properties, not mechanisms,” in Proceedings of the 2004 Workshop on New Security Paradigms, NSPW ’04, (New York, NY, USA), pp. 67–77, ACM, 2004.

15. A. Sahai, “Ciphertext-policy attribute-based encryption,” in In Proceedings of the IEEE Symposium on Security and Privacy, 2007.

16. S. Kamara and K. Lauter, “Cryptographic cloud storage,” in Financial Cryptography and Data Security, vol. 6054 of Lecture Notes in Computer Science, pp. 136–149, Springer Berlin Heidelberg, 2010.

17. A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in Advances in Cryptology–EUROCRYPT 2005, Springer, 2005.

18. S.Kamara and C.Papamanthou,“Parallel and dynamic searchable symmetric encryption,” in Financial Cryptography and Data Security, pp. 258–274, Springer, 2013.

19. N. Paladi, C. Gehrmann, M. Aslam, and F. Morenius, “Trusted Launch of Virtual Machine Instances in Public IaaS Environments,” in Information Security and Cryptology (ICISC’12), vol. 7839 of Lecture Notes in Computer Science, pp. 309–323, Springer, 2013.

20. N.Paladi,C.Gehrmann,andF.Morenius,“Domain-Based Storage Protection (DBSP) in Public Infrastructure Clouds,” in Secure IT Systems, pp. 279–296, Springer, 2013.

21. P. Mell and T. Gance, “The NIST Definition of Cloud Computing,” tech. rep., National Institute of Standards and Technology, 2011.

22. C. Waldspurger and M. Rosenblum, “I/O virtualization,” Communications of the ACM, vol. 55, no. 1, pp. 66–73, 2012.

23. D. Dolev and A. C. Yao, “On the security of public key protocols,” Information Theory, IEEE Transactions on, vol. 29, no. 2, 1983.

24. T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh, “Terra: A virtual machine-based platform for trusted computing,” in ACM SIGOPS Operating Systems Review, vol. 37, ACM, 2003.

25. A. Seshadri, M. Luk, N. Qu, and A. Perrig, “SecVisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity OSes,” ACMSIGOPSO perating Systems Review,vol.41,no.6,2007.

26. F. Zhang, J. Chen, H. Chen, and B. Zang, “Cloud visor: 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.

27. G. Greenwald, “How the NSA tampers with US-made Internet routers,” The Guardian, May 2014.

28. S.Goldberg, “Why is it taking so long to secure internet routing?,” Communications of the ACM, vol. 57, no. 10, pp. 56–63, 2014.11

29. Trusted Computing Group, “TCG Specification, Architecture Overview, revision 1.4,” tech. rep., 2007.

30. B. Parno, J. M. McCune, and A. Perrig, Bootstrapping Trust in Modern Computers, vol. 10. Springer, 2011.

31. P. Eronen and H. Tschofenig, “Pre-shared key ciphersuites for transport layer security (TLS),” 2005.

Nicolae Paladi is currently a PhD student at Lund University and researcher in the Security Lab at SICS. His research interests include distributed systems security with a special focus on cloud computing, infrastructure security, Internet security, virtualization and mobile platform security, trusted computing, as well as selected topics on privacy, anonymity and personal data protection.

Christian Gehramann Christian Gehrmann is heading the Security Lab at SICS. The security lab has around 12 members and is performing applied research in security in virtualized systems, security for IoT, cryptography, authentication theory, security in cellular networks and on the Internet.Christian Gehrmann has conducted communication and computer security research for more then 20 years. He holds a PhD from Lund University and is also an associate professor at Lund University.

Antonis Michalas Antonis Michalas received his PhD in Network Security from Aalborg University, Denmark. He is currently a postdoctoral researcher it the the Security Lab at SICS. His research interests include private and secure e-voting systems, reputation systems, privacy in decentralized environments, cloud computing and privacy preserving protocols in participatory sensing applications. He has published a significant number of papers in field-related journals and conferences.