TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol. 15, No. 1, July 2015, pp. 151 ~ 161
DOI: 10.11591/telkomnika.v15i1.8073

Platform-as-a-Service (PaaS): Model and Security Issues

Devi T*[1], Ganesan R[2]
School of Computing Science and Engineering, VIT University, Chennai, India
*Corresponding author, e-mail: devi.janu@gmail.com[1], ganesan.r@vit.ac.in[2]

Abstract

Cloud computing is making a big revolution in the field of information technology thereby reducing capital expenditures spent. Computing is delivered as a service enabling effective utilization of computational resources. Certain security issues exist which prevents individuals and industries from using clouds despite its advantages. Resolving such problems may increase the usage of cloud thereby reducing the amount spent for resources. The paper focuses on one of the three service delivery models, Platform-as-a-Service (PaaS). PaaS model, layers in PaaS and PaaS providers are described along with the security issues encountered in PaaS clouds. The issues along with solutions discussed provide an insight into PaaS security for both providers and users which may help in future PaaS design and implementation.

Keyword: *encryption, interoperability, multi-tenancy, trusted computing base, virtualization*

1. Introduction

Outsourcing of computational resources is possible with the advent of cloud computing [1]. Sharing of resources reduces capital expenditure making it foreseen [2] and can be observed as rising trend. Such sharing of resources may cause certain security issues despite of vast advantages of cloud like better utilization of resources, least time taken in deploying new services and so on. Three ways [3] to deliver cloud computing capabilities (Figure 1) are Software as a Service (SaaS), Platform as a Service(PaaS) and Infrastructure as a Service (IaaS).

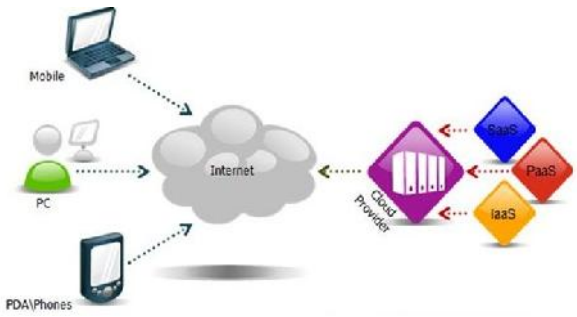

Figure 1. Service delivery models in Cloud

Table 1. Deployment models in Cloud

Deployment Model	User	Managed By
Private Cloud	Private organization	An organization or a third party
Public Cloud	General Public	An organization or selling cloud services
Community Cloud	Shared by several organizations and supports a specific community	An organization or a third party
Hybrid Cloud	An organization	Large organizations

Received April 2, 2015; Revised May 13, 2015; Accepted May 30, 2015

Characteristics of cloud are on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service. The various deployment models include private, public, hybrid and community cloud (Table 1).

PaaS plays a major role in cloud as it brings custom software development to the cloud. NIST [5] defines PaaS as *"The capability provided to the consumer to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages and tools supported by the provider".* In 1990s, desktop platforms (operating systems) and development tools catapulted the sale of PCs by empowering developers and making PCs easier to use. In near future, PaaS will drive demand for the cloud in similar ways. PaaS is important as it speeds development and saves a lot of money. According to NIST, *"PaaS consumers employ the tools and execution resources provided by cloud providers to develop, test, deploy and manage the operation of PaaS applications hosted in a cloud environment. PaaS consumers can be application developers who design and implement application software; application testers who run and test applications in a cloud-based environment; application developers who publish applications into the cloud; and application administrators who configure, monitor and manage applications deployed in a cloud. PaaS consumers can be billed according to the number of PaaS users; the processing, storage and network resources consumed by the PaaS application; and the duration of the platform usage."*

PaaS is collection of related services for creating and deploying software on cloud, so it is not a single technology. PaaS offerings manage user subscriptions, security, resource metering, role-based security and other share services. Attributes that characterize PaaS have been shown in Table 2 along with their functionalities.

Table 2. PaaS characteristics

Attributes	Functionality
Multi-tenant architecture	Common technical resources and code instance for multiple client companies.
Customizable user interface	Support the creation of flexible user interfaces without the need of writing complex code.
Unlimited database customizations	Provide the ability to easily modify or extend the data model through "point and click declarative" environment.
Robust workflow capabilities	Engender process automation by providing "point and click" tools to easily define workflow processes and specify business rules.
Granular permissions model	Multi-level control over security or sharing within applications and platform components.
Integrated content library	Common elements that extend the core application feature set, improve information sharing and speed-up go to market time.
Flexible services - enabled Integration model	Enables seamless integration of cloud application and functionality through a flexible web services enabled integration model.
Analytics layer	Enhanced ability to leverage aggregated data across companies and applications for analytics.

Advantages of PaaS are mentioned below:

a) Can edit operating system features and easily upgrade.
b) Major focus is on security, storage and database integration.
c) Services can be used from variety of international sources.
d) Serves as a application hosting, testing, deployment and development environment.
e) Total expenses is reduced.
f) PaaS is based on application development platforms that allow the use of external resources to create and host applications. Examples of PaaS offerings are as follows:
g) *CloudBees*: platform to build, deploy, and manage Java applications.
h) *Engine Yard*: platform to build and deploy Ruby and PHP applications that can be extended with add-ons.
i) *Google App Engine*: platform to develop and run Java, Python, and Go applications on Google's infrastructure.

j) *Heroku*: platform to deploy Java, Ruby, Python, Clojure, node.js, and Scala applications that can be extended with add-on resources.

k) *Microsoft Windows Azure*: on-demand compute and storage services as well as a development and deployment platform for applications that run on Windows.

l) *Salesforce Force.com*: platform to build and run applications and components bought from AppExchange or custom applications.

Prominent PaaS clouds are Microsoft Windows Azure(MWA), Google App Engine(GAE) and GroundOS(GOS). MWA and GAE are proprietary clouds whereas GOS is open source cloud. Comparison of these three clouds shown in Table 3 provides an insight into what current cloud providers need to offer in terms of cloud offerings and security in their clouds. Based on the comparison results, customers can choose their own PaaS solutions on-demand, which make the process of application development faster.

Table 3. Comparison of PaaS Clouds

Security Attributes	GAE	MWA	GroundOS
Availability	No SLA, No mention of guaranteed uptime	Provided by SLA	Problem of User
Integrity	Encryption Authentication	Encryption Authentication	Problem of User Encryption
Confidentiality	Privacy Policy Encryption Authentication	Privacy Policy Encryption Authentication	Problem of User Encryption
Authentication	Single sign-on Username & Password	Username & Password	Username & Password
Service Level Agreement	No	Yes	No

The paper focuses on basic PaaS model and also in identifying the security in PaaS environment along with the solutions. Related work and security considerations are being discussed in Section 2. Section 3 gives a deep insight into PaaS model, layers and PaaS providers. Section 4 discusses the security issues along with appropriate solutions. The paper is concluded in Section5.

2. Related Work

Security evaluations measure the effectiveness of security controls [4]. Security checklist for evaluating security in cloud has been discussed. However there is no big concentration on PaaS security options like what component of cloud system it should implement and how the set of controls will meet security challenges.

NIST Special Publication 800-53 offers security control baselines which assist organizations in choosing necessary security controls based on risk assessment and security plan for cloud systems. The guidelines serve as building blocks in evaluation and selection of technical security controls and help cloud service providers in implementing security controls. The security issues are discussed in three service delivery models along with appropriate solutions [3, 8].

A framework was proposed [9] that enable cloud service providers and customers to manage their cloud platform security. This is bit complicated as the customers need to specify in advance the type of security controls they need without adequate knowledge of best controls or effective security parameters they can get from the providers. Risk assessment framework [10] suggested for cloud security assesses and identifies risks based on scale of high, moderate and low. But the components prone to risk are not specified. Steps towards a security assessment framework were illustrated [11] by using various procedures to evaluate cloud security and discussed implications to security issues and security related regulatory compliance in the cloud. They however did not specify the best tools and controls that will handle identity and access management concerns.

A risk management methodology [12] suggested in assessing information security in cloud environments from the deployment phase to operational phase of the infrastructure providers cloud lifecycle. This assessment did not provide details on what components of the

cloud are subject to threats and specific controls to be applied in mitigating such threats. Security architecture proposed to extract security parameters [13] that meet cloud security requirements from the service user's side and also from virtual users' side as a way of cloud security management. A transparency assessment [14] was provided of cloud vendors to help businesses assess the transparency of a cloud provider's security, privacy, auditability, and service-level agreements via self-service Web portals and publications. The scorecard however revealed that a significant amount of cloud providers did not give detailed information on the type of security controls and policies that address multi tenancy issues in their various cloud delivery models.

Such an investigation of related work clearly depicts the work carried in cloud security field, but security challenges in PaaS delivery model remains to be investigated and addressed. Section 4 describes security issues with appropriate solutions in PaaS delivery model.

3. PaaS Model

3.1. Basic Model of PaaS

The two different constructs of PaaS model are Control space and App space (Figure 2) serve different purposes. App space is fully wrapped within the Control space. Since control space operates on same infrastructure like the app space, control space shares some characteristics of app space. Control Space components are definitives built from primitives and sophisticates to provide the prescriptive approach to the App Space that makes PaaS an attractive alternative to traditional software builds, configuration and deployments.

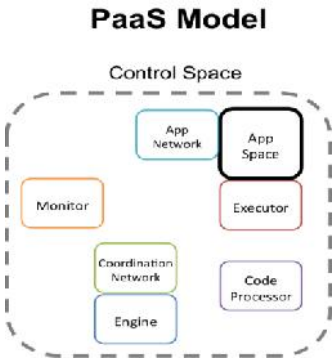

Figure 2. PaaS Model – Control Space

3.1.1. Control Space

The functions of control space include automation, management and provisioning. It interacts to lo-level components with the help of API abstractions. The Control space (Figure 3) determines what elements are exposed to App Space thereby maintaining coherency and dependencies of App Space. Several separate functions of control space can be combined in various manners based on PaaS implementation.

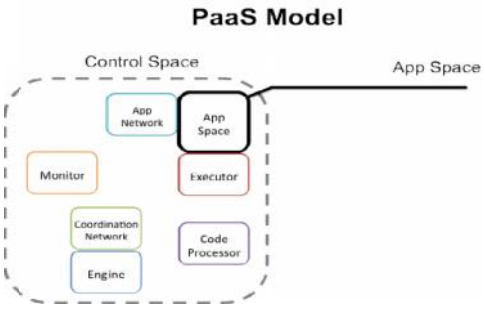

Figure 3. Control Space

3.1.2. App Space

The applications of end-user/customer are deployed, updated and run in App Space which is being controlled by Control Space. Exposure of PaaS Element types by Control Space to App Space is one of the key differentiating factors between different PaaS implementations. App Space characteristics are controlled by how the Control Space is built/designed along with what PaaS Elements were used to build the Control Space.

a) App network: It exposes network connectivity to App Space. This is actually the path through which applications communicate with app and services exposed to apps.

b) Executor: Application bootstrapping mechanism for apps being deployed. The main function is to provide compute/memory resources to App Space.

c) Code Processor: The significant function of code processor is to examine codes, libraries and dependencies before sending to Engine or Executor. It is actually like a pre-processor.

d) Coordination Network: The co-ordination of control space components takes place in this network. Based on the need, App network and the Coordination network can be combined to work together.

e) Engine: The major function of engine is to co-ordinate the distribution and provisioning of code, services and their dependencies. Decisions regarding the activities occurring inside the App space are also taken by engine. It can coordinate with Orchestration layer or automation tools outside Control space to offer resources to both App and Control Space.

f) Monitor : The functions of monitor includes keeping track of the status of App space and Control space and also helps in signaling other Control space components to resolve conflicts. In case of a web application development project, traditional software development involves many processes which make the entire phase costly and complicated. On the other hand, PaaS provider provides this platform for development.

3.1.3. PaaS Elements

These are constructs required to build a PaaS (Figure 4). They are abstractions on top of different layers of resources. Abstractions are done through service based abstractions. PaaS elements are further classified as three types as defined in the figure.

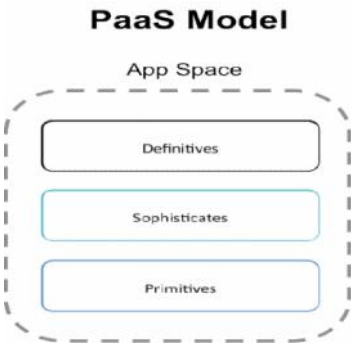

Figure 4. Elements of PaaS

All these elements in PaaS have an ability to interact through a pass-through interface or abstraction to any other PaaS element type which are all provided by Control space. Combination of the elements makes up an application in App space. An important thing to note is that the control space is entirely made up of PaaS elements.

Primitives: They are the core building block of resources. In case of operating system as a primitive, the OS has not actually been instantiated i.e., it is not a running OS. PaaS eliminates all direct ties between code and OS.

Sophisticates: Sophisticates are composites or combinations or extensions of building blocks (primitives). Combination of another sophisticate and a primitive can build a sophisticate. For example, RDBMS will likely leverage a RuntimeVM, an Operating System, Processes, an Interface, Block Store, Cache, and a File System. This complexity is hidden by RDBMS

interface and PaaS exposes such combination as a service which can be consumed by API call or database connection.

Definitives: Instantiations (Running) of Primitives and Sophisticates either directly in use or wrapped in Services/APIs creating easily leverage abstractions form the definitives. These abstractions allow complex configurations of Primitives and Sophisticates coupled with application logic and dynamic configuration capabilities. They are live and fully implemented abstractions as they consume resources. This could be a schema in a RDBMS or a collection with documents in it in a document store and this is where all of specificity occurs in PaaS.

Three major PaaS models are as follows:

a) *Comprehensive PaaS*: These providers support wide range of languages and enable IT to easily switch among these services. Eg.Windows Azure.

b) *Specific-Stack PaaS*: These providers target enterprise-focused stacks for companies that need to use the advantages of PaaS without rewriting an application or reimplementing the stack. Eg. IBM SmartCloud.

c) *Proprietary PaaS*: These providers promise an extremely capable managed platform and service in exchange for embracing a proprietary scripting language and configuration with significant lock-in risk. Eg. WorkXpress

3.2. Layers in PaaS

Figure 5 clearly depicts the two major layers of PaaS as Cloud OS and Cloud Middleware. PaaS providers such as Microsoft Windows Azure allows existing .Net developers to create their own scalable SaaS and Google App Engine enable Java and Python developers to easily develop cloud apps. Other best PaaS providers such as OrangeScape and Wolf PaaS are on-demand browser based platforms for rapidly designing and delivering applications. Business analysts who are non-developers with good analytical ability and domain knowledge can also use to launch their SaaS apps.

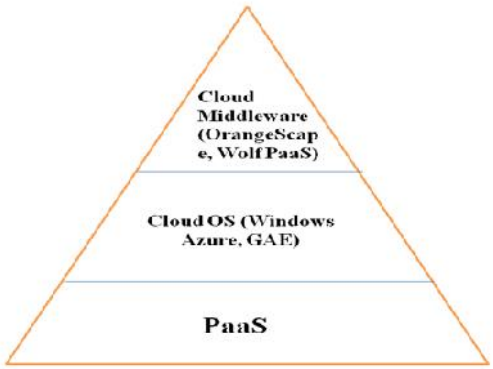

Figure 5. PaaS Layers

3.3. Evaluation criteria in choosing PaaS providers

The seven major factors to be considered while choosing a PaaS provider are as follows:

a) Programming language and frameworks: In proprietary PaaS, customer is can use whatever language is required. Salesforce.com is one of the best example which uses proprietary language.

b) Databases : Migration to different database server is made easier and many PaaS providers support "next-generation" databases such as Xeround that provide an interface similar to MySQL but are provided as a service. Database security features offered by PaaS vendors need to be verified.

c) Availability: In case of failure of server or software, service-level agreements show their importance. This is because vendors' roles and responsibilities have to be specified in SLAs.

d) Security: Security and regulatory compliance are critical when selecting PaaS vendors who drive down costs and maintain high availability by spreading applications and data across a large number of shared servers.

e) Services: Extra services are provided by PaaS vendors through third-party add-ons. Code repository integration, caching services, logging services and payment services are provided by PaaS vendors.

f) Customer care: PaaS vendors build layers between and around various services (such as application-to-database transactions) that impose a much closer relationship between developer and vendor than with other hosting options.

g) Price: Selection is also based on cost factor. Some vendors even provide free trials.

3.4. PaaS for SMBs

In case of a web application development project, traditional software development involves many processes which make the entire phase costly and complicated. On the other hand, PaaS provider provides this platform for development and the platform can be consumed using browser. It eliminates the necessity of downloading any software. These features like cost efficiency empowers small and medium-sized businesses to launch their own SaaS. SMBs can leverage the power of platform providers without any initial investment. Development of applications quickly through PaaS and agile methodologies lowers operational costs for SMBs. Iterative approach of agile methodologies and certain components in PaaS guarantees that best solution is delivered to customers in least duration. This may improve customer responsiveness, process automation and also overlay efficiencies for internal operations for SMBs. Figure shows agile framework for PaaS which the SMBs can use to develop applications faster. With PaaS, developers and project managers use iterative approach where the requirements are gathered based on stages of development. The solutions shall be delivered as a prototype with the available set of data and then the iterations can be taken from there to arrive at the best solution. This is possible with PaaS with its ready to use pre-configured components. So, PaaS can be called as "agile-ready" framework.

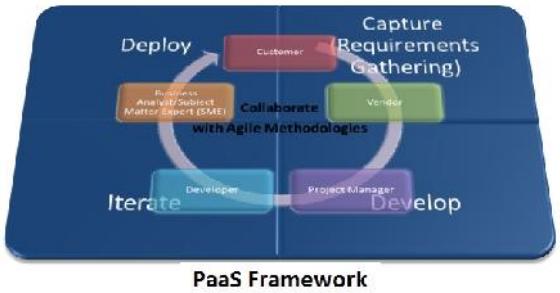

Figure 6. Agile framework of PaaS

4. PaaS Security

4.1. PaaS Security Elements

Elements that characterize PaaS security platform are as follows:

a) *Information processing*: This is the stage where one is creating data and rest of the web uses it. Creation of data may happen live on remote server. So the document can be intercepted. PaaS provides security when this data is in stored format, which clearly states that the problem is during processing stage.

b) *Information interactivity*: This is the process of sharing data across the board. Interaction can go with personal computers, networks, devices like phones and so on. This interaction connects confidential data in local network with the web where most of them can access and hence security issue comes in.

c) *Data Storage*: This specifies the hosting aspect of Cloud. Several mechanisms in PaaS allow multiple applications to be encrypted to prevent data leakage. Verification is hard as data is in shared servers.

4.1. PaaS Security Issues

4.1.1. Interoperability

Interoperability is the ability for different cloud to talk to each other at three different levels (SaaS, PaaS and IaaS). It is actually the ability to write code that works with more than one cloud provider simultaneously, regardless of the differences between the providers [15]. Application written to use specific services from a vendor's PaaS will require changes to use similar services from another vendor's PaaS. Efforts are taken on development of open and proprietary standard API's to enable cloud management, security, and interoperability. Common container formats like DMTF'S Open Virtualization Format (OVF) can be used. Application written to those standards is far more likely to be interoperable and portable. Interoperability can be maintained by providing common interfaces to objects for resource access.

Trusted Computing Base (TCB) is the solution for the above mentioned issue. TCB [16] is collection of executable code and configuration files that are considered to be secure which is installed as a layer over the operating system and provides a standardized application programming interface(API) for user objects. TCB code is minimized to avoid complexity in codes and security flaws. TCB is built by resource classification, setting resource assignment rules and evaluating resource access requests. Interoperability can be achieved by installation of TCB on each and every host and assignment of resource through TCB. Attacks from objects to hosts can be prevented as every resource assignment is checked by TCB.

4.1.2. Host Vulnerability

Vulnerability may be described in terms of resistance to a certain type of attack. Multi-tenancy [17] allows user objects to be spread over interconnected multi-user hosts. Hosts have to be protected from attacks in such an environment. If this protection fails, an attacker can easily access the resources of host and also tenant objects. Provider has to take necessary security measures. TCB serves as solution for host vulnerability also.

4.1.3. Object Vulnerability

Service providers can access and modify user objects [18]. Three ways by which security of an object can be breached in PaaS clouds are:

a) Provider may access any user object that resides on its hosts. A fully homomorphic encryption [19] can be employed as a cryptographic defense for user objects during execution, but it is computationally expensive [20]. Hence, this type of attack is unavoidable and can be avoided to some extent by trust relations between user and provider.

b) Users may mutually attack each other's objects that are tenants of same host because tenant objects synchronously share the same resources.

c) Third party may directly attack a user object. Secure coding enables objects to defend themselves.

Encrypting objects is the solution for object vulnerability. It is the responsibility of provider to protect the integrity and privacy of user object on a host. Certain problems like malicious provider and host being breached by malicious party can make objects to be deleted or modified or inaccessible. Cryptographic methods symmetric and asymmetric encryption, hashing and signatures help to protect object contents.

4.1.4. Access Control

Network communications must be confidential and access of remote entities should be controlled. Three major concepts of access control are: authentication, authorization and traceability. Some of the attacks in such cloud-based environments are impersonation, phishing attacks, brute force attacks and password reset attacks. Two-factor authentication like smart cards and biometric mechanisms can protect from such attacks.

Authentication: It needs parties to prove the authenticity of their identities during an interaction. Authentication fails when unprivileged entities try to access objects. Current authentication methods are enough for PaaS clouds.

Authorization: This mechanism determines who can access objects based on predefined policies. Unprivileged access result due to lack of authorization. Role-based access control and federated access control [21] are employed for managing authorization. But, still problems persist in PaaS as objects migrate and the difficulty exists in keeping up the policies during host reconfiguration. Each user can manage a central database for policies of his

objects. But, the centralized approach burdens the user. Best way is to carry policies along with the objects so that when an object is moved to a new host, policies that are effect on previous host must also be effective on current host.

Traceability: This is achieved through keeping records of events occurred in a system. Event records are necessary for measuring service characteristics. The users are billed by service providers based on the amount of usage. In turn users monitor the state of their applications and audit access to their data. In case of conflicts between these two, the jurisdiction can investigate the logs. HIPAA has been framed for access against personal health records [22].

Solutions to access control problems are as follows:

a) Encapsulation

Encapsulating access control policies with objects can be one of the solutions to resolve access control conflicts. Advantages of such approach are:

(1) Settings are ensured to be carried together with objects.

(2) Additional connection or component is not necessary.

(3) No need of reconfiguration during object migration.

b) Policy enforcement points (PEPs)

A Policy Enforcement Point (PEP) is the logical entity or place on a server that makes admission control and policy decisions in response to a request from a user wanting to access a resource on a computer or network server. Access control architectures often distinguish policy enforcement points and policy decision points. Policy enforcement points intercept access to protected application resources (1) and request authorization decisions from a policy decision point (2). A policy decision point evaluates authorization decision requests relative to a security context (3) and returns the evaluation result to the policy enforcement point (4). If the evaluation result indicates sufficient privileges the policy enforcement point allows the initial requestor to access the protected resource (5), otherwise access is blocked.

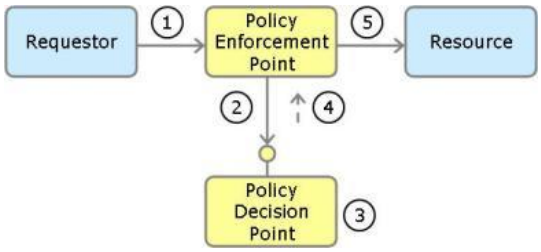

Figure 7. Policy Enforcement Points

Undeniable logging protocol: The protocol aims at solving log integrity problems and allows access requests and respective responses not to be sent directly to related parties [23]. These parties follow the messages published on online bulletin board. Prevention of misuse like false requests and response is also an added feature of the protocol.

Coming to the case of traceability, logging systems assume logger to be trustworthy. Event records are stored as simple text messages in logging systems. Integrated undeniable logging protocol can be employed for protecting logging systems against all interacting parties.

4.1.5. Privacy-Aware Authentication

For authentication, user reveals most of the details regarding him. Proxy certificates help to reduce the risk associated with revealing of these attributes. Proxy certificate is actually an electronic certificate that includes only the required attributes of the corresponding identity. Requirements to be met during privacy-aware authentication with proxy certificates are:

a) Based on access control policies defined by both service providers and users, hosts and objects should not request more attributes than the required amount. If more attributes are requested, then the service is negotiated.

b) With the help of trusted third party, easily configurable credentials which reveal data that the identity owners permit can be achieved.

Proxy certificate authorities (PCAs) handle dynamic generation of attribute based credentials based on genuine certificates and links proxy and genuine certificates in case of conflicts. They keep track of proxy certificates issued along with corresponding genuine certificates. Proxy certificates can be reused and PCAs are services provided in cloud. Hierarchical PCAs can be utilized to achieve scalability.

4.1.6. Continuity of Service and Tolerance of Fault

Service may be disrupted due to malfunction or attack and hence the interaction stops for a period of time. Content of object is changed or deleted completely. Byzantine quorum approach [25] can be employed to attain fault-tolerance and service continuity. Byzantine quorum system [26] offers read and write services to its clients on a set of replicated data items. A read operation retrieves data from a quorum of correct replicas and a write operation applies the update to a quorum of correct replicas. So, modifications on an object will be detected or inaccessible systems can be recovered in such systems. This fault tolerant system provides protection against unavailability of service in cloud environments. Survey of such systems with extensive examples is discussed [27].

5. Results and Discussion

Applications are deployed in PaaS without the necessity of purchasing and maintaining the hardware and software thereby depending on a secure browser. PaaS application security includes the security of application deployed on PaaS as well as the PaaS platform security itself. It is the responsibility of the PaaS provider to protect the runtime engine which runs the client applications. Table 4 shows the summary of security issues in PaaS along with their solutions.

Table 4. Summary of security issues with their solutions

Security Issues	Solutions
Interoperability	Trusted Computing Base
Host Vulnerability	Trusted Computing Base
Object Vulnerability	Encryption
Access Control	Encapsulation
	Policy Enforcement Points
	Undeniable Logging Protocol
Privacy Aware Authentication	Proxy Certificates
Service Continuity and Fault Tolerance	Byzantine Quorum System

4. Conclusion

Secure PaaS cloud can be achieved by understanding the PaaS model, its types and the issues related to security as described in the paper. The various features of PaaS can be utilized in an efficient manner based on the deeper understanding of PaaS environment in cloud. The characteristics of PaaS along with the evaluation criteria in choosing a provider for PaaS has also been identified along with PaaS security elements. Finally, security issues in PaaS with their appropriate solutions have been discussed to provide a clear insight in data security issues and other challenges while running application on PaaS platform. With the solutions and also by knowing these issues, customers can be precautionary while using PaaS.

Acknowledgements

The author wishes to thank the management and Dean of Computer Science department, VIT University, Chennai for their constant encouragement and support given to carry out research work.

References

[1] LM Kaufman. Data security in the world of cloud computing. *IEEE Security & Privacy.* 2009; 7(4): 61-64.
[2] D Catteddu, G Hogben. Cloud computing: Benefits, risks and recommendations for information security. Technical report, ENISA. 2009.

[3] S Subashini, V Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications.* 2011; 34(1): 1-11.
[4] VJR Winkler. Securing the Cloud: Cloud Computer Security Techniques and Tactics. Waltham: Syngress. 2011.
[5] National Institute of Standards and Technology (NIST). http://www.nist.gov.
[6] Google app engine. 2012. https://developers.google.com/ appengine/.
[7] Windows azure platform. 2012. http://www.windowsazure.com/ en-us/.
[8] D Zissis, D Lekkas. Addressing cloud computing security issues. *Future Generation Computer Systems.* 2012; 28(3): 583-592.
[9] M Almorsy, J Grundy, AS Ibrahim. *Collaboration-Based Cloud Computing Security Management Framework*. IEEE 4th International Conference on Cloud Computing. 2011: 364-371.
[10] P Saripalli, B Walters. *QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security.* Proceedings of IEEE 3rd International Conference on Cloud Computing. 2010: 280-288.
[11] S Sengupta, V Kaulgud, VS Sharma. *Cloud Computing Security-Trends and Research Directions.* IEEE World Congress on Services. 2011: 524-531.
[12] AU Khan, M Oriol, M Kiran, M Jiang, K Djemame. *Security Risks and their Management in Cloud Computing.* IEEE 4th International Conference on Cloud Computing Technology and Science. 2012: 121-128.
[13] V Fusenig, A Sharma. *Security Architecture for Cloud Networking*. International Conference on Cloud Computing and Networking Symposium. 2012: 45-49.
[14] WA Pauley. Cloud Provider Transparency: An Empirical Evaluation. Copublished By The IEEE Computer And Reliability Societies.
[15] David Cunha, Pedro Neves, Pedro Sousa. *Interoperability And Portability Of Cloud Service, Enablers In A Paas Environment.* 2nd International Conference on Cloud Computing and Services Science. 2012: 432-437.
[16] U Steinberg, B Kauer. *Nova: a microhypervisor-based secure virtualization architecture*. In Proceedings of the 5th European conference on Computer systems, EuroSys '10. New York, NY, USA. 2010: 209-222. 2010.
[17] JH Saltzer. Protection and the control of information sharing in multics. *Commun, ACM*. 1974; 17(7): 388-402.
[18] F Rocha, M Correia. *Lucy in the sky without diamonds: Stealing confidential data in the cloud*. In Proceedings of the 2011 IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops, DSNW '11. Washington, DC, USA. 2011: 129-134.
[19] A L´opez-Alt, E Tromer, V Vaikuntanathan. *On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption*. In Proceedings of the 44th symposium on Theory of Computing, STOC '12. New York, NY, USA. 2012: 1219-1234.
[20] M van Dijk, A Juels. On the impossibility of cryptography alone for privacy-preserving cloud computing. *IACR Cryptology ePrint Archive*. 2010: 305.
[21] EE Mon, TT Naing. *The privacy-aware access control system using attribute-and role-based access control in private cloud*. In Broadband Network and Multimedia Technology (IC-BNMT), 2011 4th IEEE International Conference on. 2011: 447-451.
[22] PP Gunn, AM Fremont, M Bottrell, LR Shugarman, J Galegher, T Bikson. The health insurance portability and accountability act privacy rule: a practical guide for researchers. *Medical Care.* 2004; 42(4): 321-327.
[23] Sandikkaya, Harmanci. *Security Problems of Platform-as-a-Service (PaaS) Clouds and Practical Solutions to the Problems*. International Symposium on Reliable Distributed Systems, IEEE. 2012: 463-468.
[24] Rfc 3820: Internet x.509 public key infrastructure (pki) proxy certificate profile. Dec 2004. http://tools.ietf.org/html/rfc3820.
[25] Lorenzo Alvisi, Member, Dahlia Malkhi, Evelyn Pierce, Michael K Reiter. Fault Detection for Byzantine Quorum Systems. *IEEE Transactions on Parallel And Distributed Systems.* 2001; 12(9): 996-1007.
[26] D Malkhi, M Reiter. Byzantine Quorum Systems. The Journal of Distributed Computing. 1998; 11(4): 203-213.
[27] A Bessani, M Correia, B Quaresma, F Andr´e, P Sousa. *Depsky: dependable and secure storage in a cloud-of-clouds*. In Proceedings of the sixth conference on Computer systems, EuroSys '11. New York, NY, USA. 2011. 31-46.