Volume 4, Issue 7, July 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

The Anatomy of Platform as a Service (PaaS) in the Cloud: A Detailed Look into the Challenges

Kudakwashe Zvarevashe[1], Nicholas N Karekwaivenane[2], Wilson Bakasa[3]
[1, 2, 3] M Tech Students, Dept of CSE, Jawaharlal Nehru Technological University, Hyderabad, India

Abstract*— Cloud computing has been one of the fasted travelling technical term which has a lot of tongues on fire since its inception. Software, programming platforms and the infrastructures on which the software and data will reside are now being offered as services painting a clear picture of Service Oriented Architecture framework. The cloud has really made a lot of promises on the convenience, availability and many other issues although there have been some hiccups on issues concerning security. The architecture of cloud computing has broken down into layers which include Software as a Service(SaaS), Platform as a Service(PaaS) and Infrastructure as a Service(IaaS). This paper has narrowed down the bigger piece of cloud computing reference model and focused more on Platform as a Service. Here we have exposed the layers that are within Platform as a Service and also classified their challenges according to the research done by other fellow researchers.

Keywords: Cloud computing, SaaS ,PaaS, IaaS, SOA, services.

I. INTRODUCTION

There is an English proverb which says "Necessity is the mother of all inventions". The world has seen a plethora of massive technological advancements over the last twenty years and this has lead to an outcry for the undeniable need of different types of services for hire. This has lead to the timely birth of cloud computing and today's proponents and other observers of this technology have good reason to feel that indeed these are interesting times in this unfolding story.
Lately the term Cloud Computing has become a buzzword that is easily misused to revamp existing technologies and ideas for the public. Cloud computing is a utility-oriented and Internet-centric way of delivering IT services on demand. These services cover the entire computing stack: from the hardware infrastructure packaged as a set of virtual machines to software services such as development platforms and distributed applications[1].
The cloud is associated with the following characteristics[2]:

1. *On-Demand self service:* The clients which are the cloud users in this case won't be needing human aid or human interaction whatsoever.
2. *Broad Network access:* Cloud resources are available over the network and are easily accessible.
3. *Resource Pulling:* Computing resources are shared among multiple users and assigned relative to the demand.
4. *Rapid elasticity:* The provided resources scale rapidly(they appear to be unlimited) according to the demand.
5. *Measured Service:* The usage of resources is measured and controlled.

The characteristics of the cloud can be diagrammatically represented as shown Figure 1.

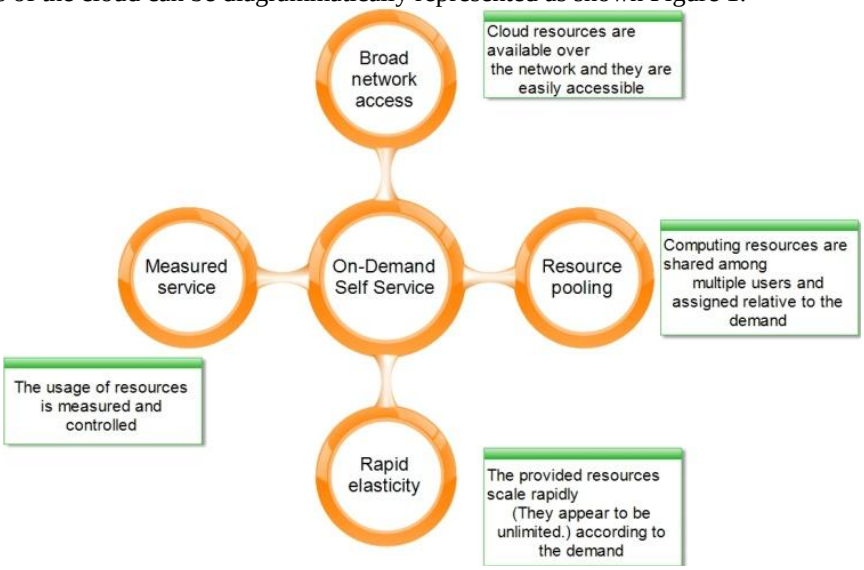

Figure1:Characteristics of The Cloud

The cloud basically provides three types of services which are sometimes known as the SPI mode and these are[3]:

(a). Software as a Service (SaaS)- The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser.

(b). Platform as a Service (PaaS) - The capability provided to the consumer is to deploy onto the cloud infrastructure his own applications without installing any platform or tools on their local machines. PaaS refers to providing platform layer resources, including operating system support and software development frameworks that can be used to build higher-level services.

(c). Infrastructure as a Service (IaaS)- The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The three models are illustrated in Figure 2.

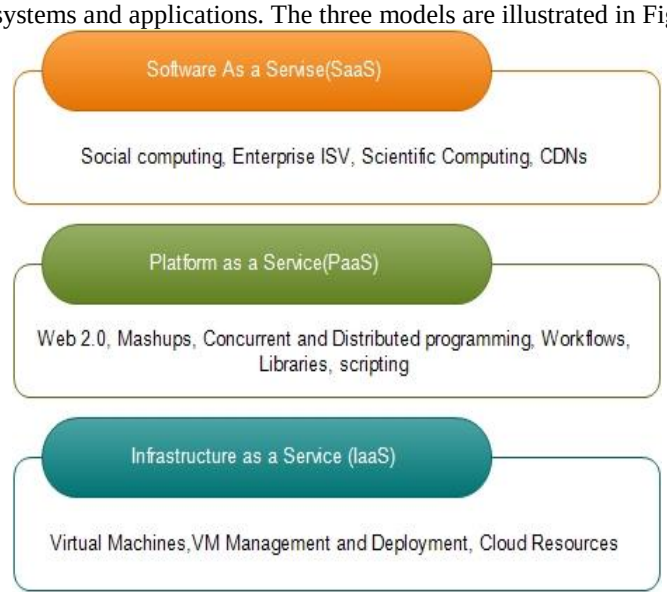

Figure2: Cloud Computing models

II. PLATFORM AS A SERVICE BUILDING BLOCKS

Platform as a Service (PaaS) provides a development and deployment platform for running applications in the cloud. They consist of middleware on which applications are built[3]. In other words PaaS makes a complete software platform including - infrastructures, application servers, development tools, databases and storage- available over the internet[4]. A general overview of the features characterising the PaaS approach is given in Figure3[1].

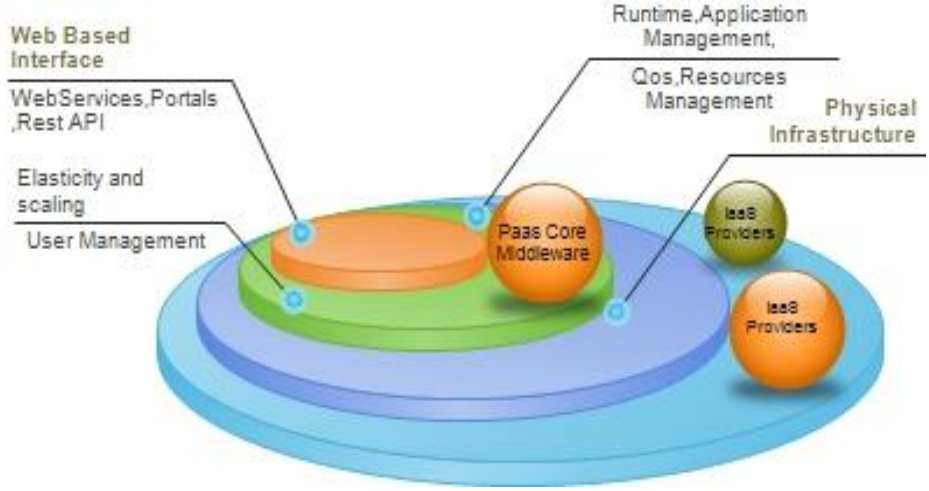

Figure3: PaaS reference model

PaaS core middleware

Application management is the core functionality of the middleware. PaaS implementations provide applications with a runtime environment and do not expose any service for managing the underlying infrastructure. They automate the process of deploying applications to the infrastructure, configuring application components, provisioning and configuring supporting technologies such as load balancers and databases, and managing system change based on policies set by the user[1].

The core middleware is in charge of managing the resources and scaling applications on demand or automatically, according to the commitments made with users. From a user point of view, the core middleware exposes interfaces that allow programming and deploying applications on the cloud. These can be in the form of a Web-based interface or in the form of programming APIs and libraries[1].
The PaaS model also consists of a sub layer which is called PurePaas[3]. This is where users are provided with software that is installed on the user premises. The other part of the equation is where users are provided with the middleware for developing applications together with the infrastructure and we have decided to call this the BasicPaas.

Modes of PaaS
The constant evolution of PaaS notwithstanding, the consensus is that there are two dominant modes of PaaS now in use[4]:

a)Model-driven PaaS involves higher-level programming languages, or even template-based software building programs that enable users with little coding experience to create business applications.

b)Deployment PaaS is newer. It refers to platforms in the cloud that can host applications that were created with standard programming languages such as Java or PHP. The assumption is that the development is being done off of the PaaS solution on integrated development environments (IDEs) such as Eclipse. This may sound a lot like IaaS, but it's different. With deployment PaaS, the developer does not have to worry about architecting, managing, or scaling the virtual machines that underlie the application

III. THE TAXONOMY OF PAAS IMPLEMENTATIONS

According to Rajkumar Buyya[1], PaaS can be classified into three categories of implementations: PaaS-I, PaaS-II and PaaS-III.

a) PaaS-I
This category identifies PaaS implementations that completely follow the cloud computing style for application development and deployment. They offer an integrated development environment hosted within the Web browser where applications are designed, developed, composed, and deployed. This is the case of Force.com and Longjump. Both deliver as platforms the combination of middleware and infrastructure[1].

b) PaaS-II
In this class we can list all those solutions that are focused on providing a scalable infrastructure for Web application, mostly websites. In this case, developers generally use the providers' APIs, which are built on top of industrial runtimes, to develop applications. Google AppEngine is the most popular product in this category. It provides a scalable runtime based on the Java and Python programming languages, which have been modified for pro- viding a secure runtime environment and enriched with additional APIs and components to support scalability[1].

c) PaaS-III
The third category consists of all those solutions that provide a cloud programming platform for any kind of application, not only Web applications. Among these, the most popular is Microsoft Windows Azure, which provides a comprehensive framework for building service- oriented cloud applications on top of the .NET technology, hosted on Microsoft's datacenters. Other solutions in the same category, such as Manjrasoft Aneka, Apprenda SaaSGrid, Appistry Cloud IQ Platform, DataSynapse, and GigaSpaces DataGrid, provide only middleware with different services[1].

IV. CHARACTERISTICS OF PAAS

As pointed out by Sam Charrington, product manager at Apistry.com, there are some essential characteristics that identify a PaaS solution and these are shown in Figure4.

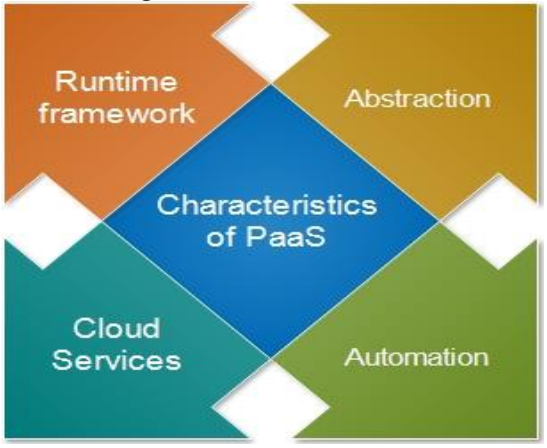

Figure4:Characteristics that identify a PaaS Solution

a)Runtime Framework

This framework represents the "software stack" of the PaaS model and the most intuitive aspect that comes to people's minds when they refer to PaaS solutions. The runtime framework executes end-user code according to the policies set by the user and the product

b)Abstraction

PaaS solutions are distinguished by the higher level of abstraction that they provide. Whereas in the case of IaaS solutions the focus is on delivering "raw" access to virtual or physical infrastructure, in the case of PaaS the focus is on the applications the cloud must support. This means that PaaS solutions offer a way to deploy and manage applications on the cloud rather than a bunch of virtual machines on top of which the IT infrastructure is built and configured[1].

c)Automation

PaaS environments automate the process of deploying applications to the infrastructure, scaling them by provisioning additional resources when needed. This process is performed automatically and according to the SLA made between the customers and the provider. This feature is normally not native in IaaS solutions, which only provide ways to provision more resources[1].

d)Cloud services

PaaS offerings provide developers and architects with services and APIs, helping them to simplify the creation and delivery of elastic and highly available cloud applications. These services are the key differentiators among competing PaaS solutions and generally include specific components for developing applications, advanced services for application monitoring, management, and reporting[1].

V. CHARACTERISTICS OF PAAS

The challenges of PaaS can be classified into four distinct categories as shown in Figure5.

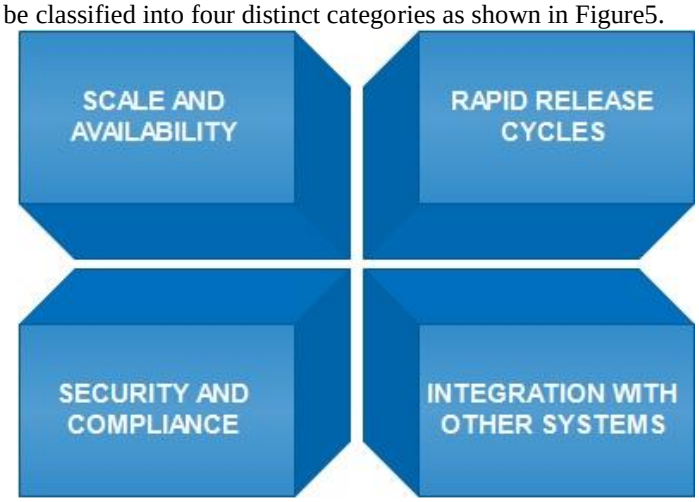

Figure5:Challenges of PaaS

a) Scale and Availability

Availability should be addressed early with PaaS, especially if you've built an application that can go viral. The "as-a-service" trend is part of the broader cloud computing and mobile device revolutions. You have to think through the implications of the new world of IT as you adopt PaaS. Application load levels can be difficult to predict, both for your application and others that are hosted at the PaaS provider. With social media and mobile apps, your PaaS creation could be hit with requests from a huge crowd of new users overnight. That could be an amazing thing or it could cause outages and all kinds of disruptions. You must ensure that your PaaS solution can scale easily in response to demand. If your PaaS application is integrated with back-end systems behind the firewall, an ungoverned runaway success could disrupt your business. The problem doesn't even have to be with your application. "Bad Neighbor Syndrome" occurs with some PaaS providers. Bad neighbors are other PaaS customers who share cloud infrastructure with your application. Even though all PaaS tenants are isolated from each other, tenants who consume more than their fair share of IaaS resources can cause your application to perform poorly[4].

b) Rapid Release Cycles

Compared to traditional software development, the release cycle for PaaS is like playing a movie on fast forward. This is especially true for model-based PaaS but it also applies to deployment PaaS. In both situations, code can be pushed into production more quickly than in the conventional developer-to-operations cycle. In traditional software development, a new feature request involves multiple stakeholders in a clumsy, partially manual review process. A series of handoffs ensues as the feature goes from request to development, test, security review, and installation. The process can take a long time. The new feature would be developed and tested, then put into production in an overall application update. The whole process might last several months or at very minimum, a couple of weeks.

Model-based PaaS can make the creation of a new feature possible within a couple of hours. Generally speaking, this a good thing. Business ideas can be quickly translated into working software. Agile methodology and new streamlining techniques such as "continuous integration" of new code help provide a disciplined framework for aligning business and IT stakeholders with PaaS' rapid development tooling. Yet, these methods cannot create this alignment on their own. The organization has to be ready and willing to change its approach to software development. The reality is that most organizations will have software projects moving at different speeds. As PaaS becomes one of the accepted modes of development, there has to be a parallel organizational commitment to keeping up with the new speed levels in the processes of requirements gathering, development, review, test, approval and release[4].

c) Integration with other systems

Connections between cloud-based applications built on PaaS and other enterprise systems present security, operational and governance challenges. PaaS-based software is inherently service-oriented. It has the ability to call on application programming interfaces (APIs) exposed on numerous systems. These include APIs that use Simple Object Access Protocol (SOAP) as well as the increasingly popular Representational State Transfer (REST).

Without adequate controls, systems can be exposed through APIs along with the business processes they support. Of course, few organizations simply leave an API totally open to the world. However, the difference in development and change cycles between legacy systems and PaaS software can lead to a "Tortoise and Hare" syndrome where the legacy system cannot keep up with new PaaS features. If external users can access internal business process through APIs that are out of sync, that can cause operational and compliance difficulties. Alternatively, if an API is not available because a change in the PaaS solution has broken the connection, that is also bad for business[4].

d) Security and Compliance

Concerns about cloud security are not new, but PaaS can bring risk exposure to a whole new level. The cloud tends to blur the security perimeter in general. In the old days, you had a good idea of where your infrastructure ended and the rest of the universe began. With PaaS, your business extends to multi-tenant servers in vague geographical areas. The ability for non-technologists to set up software on their own, as might be the case with LOB users on model-based PaaS, can wreak havoc on controls over data access, authentication and authorization. If the PaaS application connects to other enterprise systems, it can become an inadvertent conduit for improper access and potential mischief.

Responsibility and accountability are major factors to consider when making the move to PaaS, especially if there is a "shadow IT" effort in the works. Business stakeholders tend to complain about all the "red tape" and slow-moving processes in IT, but those processes exist for a reason. Anyone who has ever been asked to find an email subpoenaed in a lawsuit will know the value of rigorous controls and data governance. A shadow IT project hastily thrown together with PaaS can accidentally create a security and compliance minefield[4].

VI. CLASSIFICATION OF PAAS SECURITY PROBLEMS

The security problems in PaaS can be classified into the following two classes as shown in figure 6:

- Resource pulling and Rapid elasticity Issues
- Broad network access and Measured service Issues

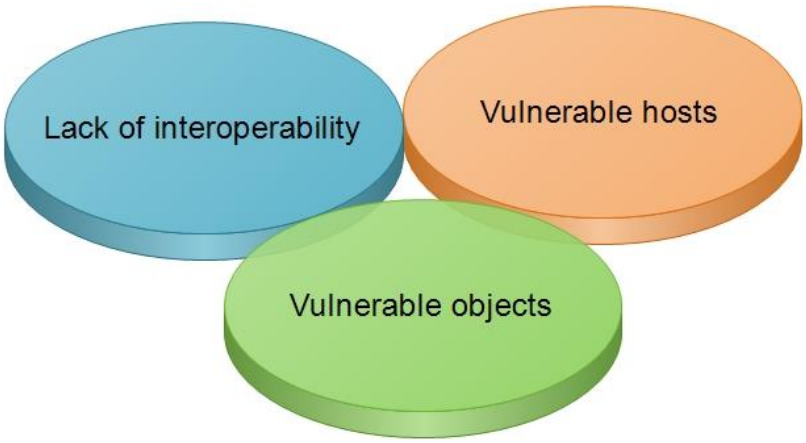

Figure 6:Resource pulling and Rapid elasticity Problems

a)Resource pulling and Rapid elasticity Issues

1) Lack of interoperability:

Diverse computational resources may lead to security breaches if objects' access to the resources cannot be handled in a standard way. This may cause a set of resources to halt or a setting that is proven to be secure for a specific resource to be a breach for another. A simple example is a security breach that is caused by different case sensitivity defaults. A person who is authorized for a file named "file" can gain access to a secret file named "FILE" by mistake. Interoperability can be maintained by providing common interfaces to objects for resource access. Resource interfaces must be designed carefully to support all possible access
scenarios[2].

2) Vulnerable hosts:

Multi-tenancy has been studied since the earliest multi-user operating systems [5]. Today, the concept covers a wider perspective where the user objects are spread over interconnected multi-user hosts. Not only objects but also hosts must be protected from possible attacks in a multi-tenant environment. Such a protection can be achieved by evaluating resource access requests of every single object on the host. If the security of a host is breached, an attacker can access the host's resources as well as all of its tenant objects. Therefore, protection against third parties is also a necessity for the host. Taking the essential network security measures is the responsibility of the provider.

3) Vulnerable objects:

The security of an object can be breached in one of the following three ways in PaaS clouds. **First**, service provider may access any user object that reside on its hosts. **Second**, users may mutually attack each other's objects that are the tenants of the same host. Finally, a **third** party may directly attack a user object. Service provider access to the objects is natural and it is required for the most basic function of a cloud: executing an object. An object is eventually executed unless it is put into the cloud just to be stored. Theoretically, the service provider is capable of accessing and modifying the user objects [6]. Only appropriate cryptographic defense for the user objects during execution is the fully homomorphic encryption schemes [7] which are computationally expensive and may be insufficient[8]. Therefore, the first kind of attack is inevitable and its solution probably lies in the trust relations of the user and the provider. Still, some parts of the objects may be irrelevant during execution and may be encrypted. The second kind of attack may occur if tenant objects synchronously share the same resources. These kind of attack are expected to be prevented by the mechanisms used to evaluate objects' resource access. Third kind of attack must be defended by the object itself. Typically, using a secure program logic and secure coding is the responsibility of the programmer.

b)Broad network access and Measured service Issues

1) Communication confidentiality:

Confidential communication is the key factor of any networked computer system and PaaS clouds are not an exception in this sense. Objects must communicate in order to interoperate and to be accessed by their owners. The communication channel can be eavesdropped to extract valuable information as objects reside on remote hosts. Therefore, communication confidentiality must be obtained.

Figure 7: Broad network access and Measured Service

2) Authentication:

Authentication is the first element of access control. It requires parties to prove the authenticity of their identities during an interaction. Unprivileged entities may access objects if authentication fails. Current authentication mechanisms are mature enough [9] and PaaS clouds probably do not require novel mechanisms. Still, the standards for the authentication mechanisms must be clear.

3) Authorization:

Authorization is the second element of access control. Authorization mechanisms determine who can access to the objects based on predefined policies. Lack of authorization mechanisms may lead to unprivileged access. Today's authorization systems generally define access control for static (or slowly evolving) content in predefined domains. Role-based access control and federated access control help administrators manage authorization up to some extent [10]. Still, they may not cover all problems in a PaaS cloud where objects migrate. Moreover, the hosts may be reconfigured in time and it may be difficult to keep up with the policies during these reconfiguration periods. A host that hosts numerous objects at a given time must know the policies of each object to apply them. The dynamic nature of PaaS clouds does not let the host keep these policies up-to-date locally. Alternatively, the host may try to conduct online queries to fetch the policies of each object. This the policies of his objects. This centralized approach may be a burden for the cloud user. Each object is likely to have its own access control requirements as an object may reside on any host at a given time. For ex ample, when an object is moved to a new host, the policies that are effective on the previous host must also be effective on the current host[11].

4) Traceability:

Traceability is the last element of access control. It is achieved through keeping records of the events occurred in a system. In addition to access control needs, measured service characteristic requires fairly kept event records. Event records are important for each cloud stakeholder. Service providers bill the users according to the amount of use; users monitor their applications' state and audit access to their data; jurisdiction investigates logs in case of conflicts in between parties. Besides, legislation may especially require the logs of some specific sorts of actions, such as access to personal health records [11]. However, many of today's logging systems assume that the logger is trustworthy. Event records are transmitted and stored by logging systems as simple text records. However, a PaaS cloud must have an integrated undeniable logging mechanism and the logging system must be secure, protected against all interacting parties including the system administrators.

VII. CONCLUSION

We have managed to incorporate the challenges associated with PaaS as observed by other authors in this paper. We have also managed to dissect and expose the inner and outer parts of PaaS by clearly showing the different modes of PaaS, the classification of PaaS and many more. This paper has also managed to further expose and uncover the research areas that are still to be looked at solving the challenges of Platform as a Service.

REFERENCES

[1] Rajkumarr Buyya, Mastering Cloud Computing Foundations and Application Programming, Morgan Kaufman, 20013.

[2] Mehmet Tahir Sandikkaya, Ali Emre Harmancı," Security Problems of Platform as a Service(PaaS) Clouds and Practical Solutions to the Problem", 2012 31st International Symposium on Reliable Distributed Systems

[3] Ruchita D.Londhe, Swati S. Sherekar, V. M. Thakare, "Imperial Analysis of Threats and Vulnerabilities in Cloud Computing" Volume 5, Number 4, 2014, International Journal of Advanced Research in Computer Science

[4] Michael Benedict," Coming to (your) Terms With Platform-as-a-Service (PaaS)", 2013, Progress Software Corporation, Whitepaper

[5] J.H. Saltzer. Protection and the control of information sharing in multics. Commun. ACM, 17(7):388–402, 1974.

[6] F. Rocha and 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, pages 129–134, Washington, DC, USA, 2011.IEEE Computer Society.

[7] A. L´opez-Alt, E. Tromer, and 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, pages 1219–1234, New York, NY, USA, 2012. ACM.

[8] M. van Dijk and A. Juels. On the impossibility of cryptography alone for privacy-preserving cloud Archive,2010:305, 2010.

[9] M. Clerc, Rfc 5246: The transport layer security (tls) protocol version 1.2, Dec 2010. http://tools.ietf.org/html/rfc5246.

[10] E.E. Mon and T.T. 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, pages 447–451, oct. 2011.

[11] P.P. Gunn, A.M. Fremont, M. Bottrell, L.R. Shugarman, J. Galegher, and T. Bikson. The health insurance portability and accountability act privacy rule: a practical guide for researchers. Medical Care, 42(4):321–327,2004

