Analysis of Centralized and Decentralized Cloud Architectures

Renuka Prasad Pasupulati, M.S.
Graduate Student
School of Computing
University of South Alabama
Mobile, AL, USA
rp1423@jagmail.southalabama.edu

Dr. Jordan Shropshire, PhD.
Associate Professor
School of Computing
University of South Alabama
Mobile, AL, USA
jshropshire@southalabama.edu

***Abstract*-This research analyzes cloud computing systems from a design perspective. Specifically, the research investigates the impact of distributed and centralized cloud architectures on security and performance. It begins by establishing a generic terminology for comparing cloud computing systems. Next, four architectural components of clouds are compared. The components are: compute, storage, networking, and VM registry. These subsystems are core features which collectively dictate the scope and composition of the cloud's attack surface. For each subsystem, the implications of distributed and centralized architectures are observed. Specific examples and four leading cloud computing platforms are referenced. The selected cloud platforms are vCloud, OpenStack, Eucalyptus, and Cloudstack. These platforms were selected because they present variations of distributed or centralized architectures. The results describe the relative strengths and weaknesses of both types of architectures. Finally, this research concludes that neither architectural approach is more secure, since both incorporate risk tradeoffs in their core components.**

Keywords-cloud computing; architecture; centralization; performance; security

I. INTRODUCTION

Cloud computing systems have emerged as a mainstream ecosystem for scalable, low-cost, on-demand computing [1, 2]. Individuals and organizations have migrated to cloud computing to meet their business, individual, social, and enterprise computing needs [3]. Given this high level of interest, it is surprising that relatively little research has focused on cloud architecture from a high-level of abstraction. For instance, considering the interrelations among various components within a given cloud computing system can yield valuable information about system resilience [4].

Defining characteristics, such as the degree of component centralization or decentralization, have significant implications on the security and performance of cloud-based services [5]. Cloud components, which are centralized and integrated, often support enhanced processing speeds but increase the risk of catastrophic failure in the event of a zero-day virus [6]. Decentralized components are slower but offer compartmentalization of risk. Such tradeoffs must be considered before adopting a cloud computing system [7].

Thus, the purpose of this research is to analyze cloud architectures and consider the degree to which their components are centralized or decentralized. It is assumed that cloud architectures are rarely compared at a high-level because their technical documentation is wrought with proprietary terms and marketing concepts [1, 8]. Although this allows for product differentiation, it inhibits apples-to-apples comparisons of system architectures. Therefore, the background section defines four basic cloud computing system components. The components are: compute, storage, networking, and VM registry operations.

The components are defined in platform-neutral terminology. In the analysis section, distinctions between centralized and decentralized components are drawn. These distinctions are illustrated with diagrams and summary descriptions of the components of leading cloud computing systems. The cloud computing systems used in the analysis include: OpenStack, Cloudstack, Eucalyptus and vCloud. These platforms were selected because they collectively account for the plurality of clouds in existence [9]. Following the analysis, implications for research and practice are discussed. Finally, concluding comments are shared.

II. BACKGROUND

For the purposes of this research, the architecture of a cloud computing system is presented in terms of five conceptual components. The components are based on a meta-analysis of existing models [9]. They are: compute, storage, networking, VM registry, and dashboard. The compute component controls the cloud computing fabric [10]. It has several responsibilities. It defines and controls interaction with underlying virtualization mechanisms. This includes compute nodes or servers which are configured with bare metal hypervisors. The hypervisors host the virtual machines under the direction of the compute controller. The compute component also performs authentication and authorization services. This not only secures client and administrator sessions, but also secures communications between compute nodes and compute controllers. One of the most crucial tasks of the compute component is resource scheduling. The compute controller is responsible for assigning virtual machines to compute nodes. Placement decisions are made using various metrics and policy considerations. Finally, the role of load balancing is also assigned to the compute component. The cloud storage component is responsible for retaining permanent copies of virtual machines and backend data storage [1]. It performs several key tasks. It offers object storage for cost-effective, scale-out storage. Object storage is unique in that it do not split data files up into pieces if raw data. Besides object storage, it also offers block storage. This option provides increased performance and easier integration with enterprise storage platforms. Blocks are split into evenly sized segments of data. Each segment or block has its own address but no metadata to provide context about what it is.

The storage component also provides storage targets for virtual machine instances. Finally, the storage target can be configured to replicate data across storage arrays or distributed file systems. The primary role of the networking component is to provide connectivity to cloud-hosted virtual machines [10]. These virtual machines require access to other cloud resources and ultimately to the internet, so they can provide services to end users. The networking component has several primary responsibilities. It manages networks and IP addresses. Cloud networks can be flat or segmented using virtual LANS. The networking component receives input from administrators and provisions the network as expected. The virtual network can be provisioned manually or automatically using orchestration tools. The networking component creates virtual networks through the use of virtual hardware. It supports fixed IP addresses and floating IP addresses. Fixed IP addresses are manually assigned to devices. They are used for public networks in which external entities require a target IP address. The virtual machine registry component provides discovery, registration, and delivery services for virtual machines [1, 4]. It performs several tasks. It provides the ability to copy or snapshot virtual machine images. A snapshot is a composite backup which records various aspects of an instance's state at a point in time. The virtual machine registry manages storage of virtual disks and virtual machines using a variety of back-ends. It is able to accommodate a variety of storage architectures, such as internal file systems, storage area networks, and network attached storage arrays.

The virtual machine registry offers base templates from which users can start new service stacks. Rather than replicate the same basic elements when provisioning new services, build templates can be used to recreate basic configurations. Finally, the virtual machine registry tracks the location of stored virtual machine instances and snapshots. The registry maps each virtual machine to its storage point and associates each snapshot with a particular instance record.

The dashboard component provides users and administrators with a graphic user interface to access, provision, and automate cloud resources [10]. It has several core functions. It provides web access to users. Individuals may access a cloud using a desktop, laptop, tablet, or smartphone. The web interface must be capable of accommodating all formats. The dashboard must provide an overall view of the size and state of the cloud. Performance metrics, usage rates, spare capacity, and problematic sections must be communicated over the dashboard. The dashboard component must support administrative tasks such as creating, modifying and closing user accounts. The location of the dashboard component and all of other major components within the basic cloud architecture are shown below (Fig. 1).

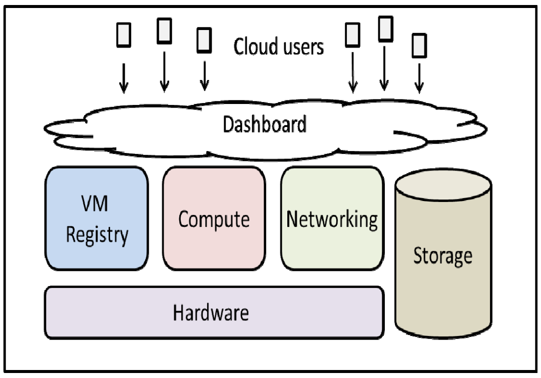

Fig. 1. Main Cloud Components

III. ANALYSIS

This section analyzes four architectural components of cloud computing systems. For each of the four elements, comparisons between centralized and federated are noted. The analysis focuses on differences in the subsystem footprint, distribution or centralization of control, and impact on scope of attack surface.

A. Compute

Cloud compute component is classified as centralized if resource scheduling [11] is performed by cloud controller [11] and decentralized if resource scheduling is not performed by cloud controller (Fig. 2). In the centralized version, the resource scheduling [11] in the hosts is maintained by the cloud controller [11]. It allocates the nearby resources to the hosts when the instances are to be launched and also the operations are managed by the single server in the cloud controller. In the decentralized version, the resource scheduling is done by the cluster controller [11] where it is operated by the cloud controller server [11]. In this version, a group of clusters is maintained by the cloud controller, and the clusters operate the group of hosts. The scheduling is done based on the resources available in the cluster level, and it is called as cluster-level scheduling [11]. The task of the resource scheduling is to accept the requests from the management console or interfaces and to allocate the resources to the hosts or physical nodes. Resource scheduling service determines which physical node or host a virtual machine should be launched on. Resource scheduling also manages the virtual machines execution and service level agreements for the cluster. For instance, cloud platforms with decentralized compute include Eucalyptus and vCloud.

The Eucalyptus Cloud controller (CLC) [12] is one of the top level component in the Eucalyptus which is a Java program that offers EC2-compatible SOAP and Query interfaces as well as web interfaces [12] to the outside world. Cloud controller performs high level resource scheduling, system accounting and in addition handles incoming requests [12]. CLC collect resources from multiple clusters such as nodes sharing a LAN segment [12]. CLC accepts user API requests from command-line interfaces like euca2ools or GUI-based tools like the Management Console [12] and manages the underlying compute resources. A cluster is equivalent to zone, and a single Eucalyptus cloud can have multiple clusters and acts as a front end for the particular cluster. Each cluster consists of one cluster controller for cluster-level scheduling and network control [12]. vCloud compute layer consists of CPU, memory and hypervisor technology components [13]. A three-host cluster [13] is used to support vCloud management components, and it is sufficient for typical vCloud environments. To provide availability for management components, vSphere High Availability (HA) and Distributed Resource Scheduling (DRS) [13] are performed on a management cluster. In terms of performance, centralized compute components require fewer steps than the decentralized approach, as there are not mid-level agents to step through. This translates into faster processing speeds, easier implementation, and a reduced likelihood of "ghost" or unregistered virtual machines. The decentralized compute design presents a reduced risk of catastrophic failure. Decentralized compute components isolate resource scheduling from the control node. A cluster controller performs scheduling for its respective cluster. If an isolated cluster controller is compromised or otherwise brought offline, the other cluster controller would still function. This translates into a more resilient cloud.

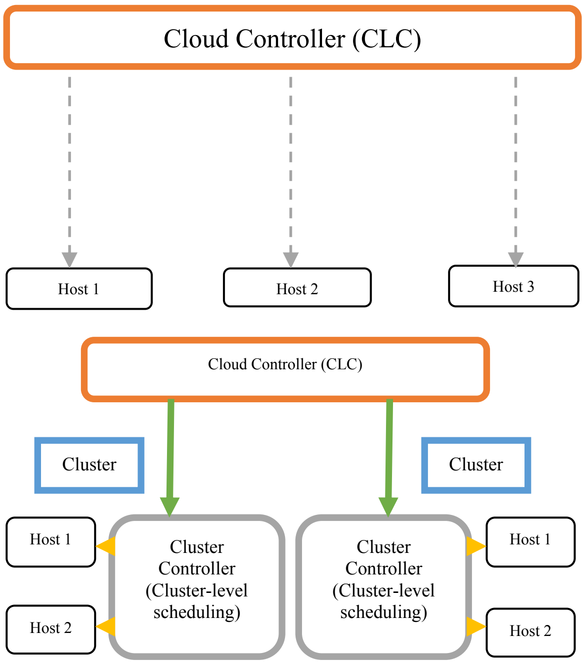

Fig. 2. Centralized & Decentralized Versions of the Cloud Compute Component

B. Storage

Cloud component storage is classified as decentralized if the storage system is distributed and centralized if the storage system is singularly controlled (Fig. 3). In the centralized version, the storage system is maintained by the cloud controller, and it is operated by the central server in the cloud controller. All the hosts are allocated with only one storage system for storing the images, files and data. In the decentralized version, the storage system is maintained at the cluster level. This version consists of a group of clusters with different storage systems for different clusters. Therefore, the group of hosts in the cluster will use the storage system related to that cluster but not the other storage system of the other cluster, which means that the storage system is not centralized and distributed at the cluster level. Storage controller operates based on the type of storage system because some of the storage systems have the central server, and some of the storage systems have their own standard servers. In the distributed storage system, the storage controller allows users to store the static data such as virtual machine images which provides more scalability, redundancy, and durability [13].

For instance, cloud platforms with decentralized storage architecture include Eucalyptus and Cloudstack. They tend towards distributed blob and file systems. Eucalyptus consists of two components for the storage. One is walrus [14] at the cloud level, and the other is storage controller [14] at the cluster level. Storage controller implements the elastic block storage [15], which allows instances to use the volumes. Volumes can be created, deleted, attached to running instances, detached and to snapshot using EC2 interface [15], and all the command operation on the volumes will be performed by the euca2ools [12]. CloudStack consists of two types of storage. One is primary storage associated with the cluster; the other is secondary storage associated with the pods and regions [16]. Primary storage stores all the virtual disks [16] of the virtual machines running on hosts in that cluster. CloudStack [16] plays the key role in allocating the virtual disks to the primary storage devices. Multiple primary storage servers [16] can be added to the cluster or zone [16]. However, one is required for one host, and it is very close to the host to increase the performance.

Decentralized storage clusters are relatively easy to modify. They support best-of-breed configurations, in which independent storage solutions are integrated with a cloud platform in order to achieve a specific goal (e.g., support storage in proprietary data formats). They also provide security via compartmentalization [16]. Even if attackers are able to infiltrate an isolated storage array, they cannot launch a rogue virtual machine without also hacking the cloud's compute component. These advantages come at the cost reduced speed.

The centralized storage configuration is comparatively more streamlined. It eliminates costly, performance-draining middleware by integrating storage with cloud-level controls. The price of this increased performance is increased risk of failure. If the cloud controller is hacked, the storage array should be considered compromised as well.

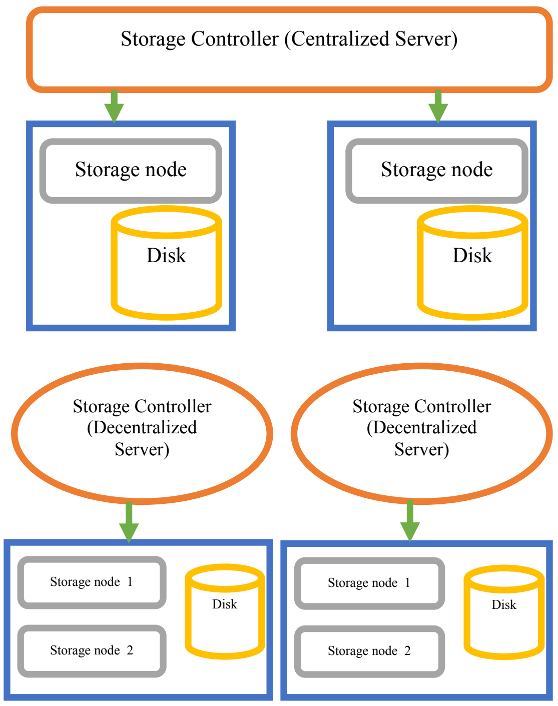
Storage Controller (Centralized Server)
Storage node
Disk
Storage node
Disk
Storage Controller (Decentralized Server)
Storage Controller (Decentralized Server)
Storage node 1
Storage node 2
Disk
Storage node 1
Storage node 2
Disk

Fig. 3. Centralized and Decentralized Versions of the Cloud Storage Component

C. *Networking*

Cloud computing systems rely on connectivity within cloud substrates and between the cloud and the exterior environment (Fig. 4). Most cloud networks are highly centralized. They are typically controlled by a single, top-level element which cannot be bypassed. Functions such as route selection and management of the switch fabric [17] cannot be distributed to third party software. However, with the increasing growth in Software-Defined Networking (SDN) [17], there is growing pressure to decentralize part or all of the networking functions to independently-managed entities. In this case, the ultimate goal is to allow the SDN backplane [17] to control the cloud network. This effort is currently manifested as support for independent virtual network devices (such as vSwitches) [17]. Thus, cloud networks, which are tending toward decentralization, will support independently-managed virtual network devices. These architectural differences mirror a larger trend in cloud computing – responsibility for management of cloud networks. Cloud networks have traditionally been managed by cloud/virtualization teams but are beginning to be passed off to network administration groups. This has implications for security. Network administrators tend to view security in terms of authentication, authorization, and accounting (AAA) [17].

Organizational networking groups have deep expertise in the application of the AAA principles. However, they trade their focused approach for a less global view of the cloud. For instance, OpenStack implements networking as a centralized service. OpenStack networking is a standalone component within the cloud architecture OpenStack networking has an extension framework to allow additional services such as intrusion detection systems (IDS), load balancing, firewalls, and virtual private networks (VPN) to be deployed and managed [18]. Although it is configured to use a centralized network engine, OpenStack supports switching platforms [18] from third-party providers. Thus, it retains control of network functionality while sacrificing compartmentalization.

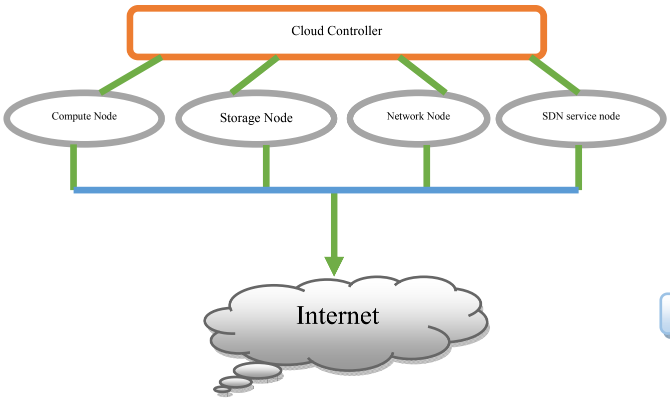

Fig. 4. Centralized Networking

D. *VM Registry*

The purpose of the virtual machine registry component is to support registration and initialization services for virtual machines. It also supports VM discovery and lifecycle management. The registry enables virtual machine snapshot or cloning [19]. It tracks storage of virtual disks and virtual machines using a variety of back-ends. It is able to accommodate a variety of storage architectures, such as internal file systems, storage area networks, and network attached storage arrays. The virtual machine registry offers base templates from which users can start new service stacks. Cloud VM registry component [19] is classified as centralized if the common database is used by all the virtual machine servers to store the data and the status of the virtual machines and decentralized if the virtual machines are using the different database for different servers (see Fig. 5. below).

For instance, OpenStack has a centralized registry component called Glance [20]. This component acts as a client-server [20] feature through which user requests are performed via REST API [20]. Glance domain controller [20] is divided into layers manage internal server operations, and each layer implements its own task. Glance store maintains interactions between glance and various data stores. It includes an optional registry layer which establishes secure communication between the domain and DAL [20] by using a separate service. Eucalyptus also makes use of a centralized registry feature called Walrus [15]. Users can store the data with the Walrus and organize them in the form of buckets and objects [21]. Walrus is able to create, delete, and list buckets and objects as well. It is also set to access control policies. Eucalyptus cloud [21] is able to access Walrus by which end-users can access both from outside and inside the cloud. This component often provides key management. The role of user and device authentication is a critical function in cloud component. Separation marks a significant improvement in security. A cloud with an isolated registry limits attackers' ability to implement rogue virtual machines or commandeer existing images. However, the process of coordinating key exchange among disparate component can be difficult. Programmatic inconsistencies among components may inhibit authentication and authorization operations.

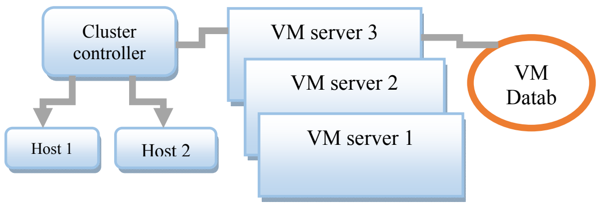

Fig. 5. Centralized VM Registry Component

IV. IMPLICATIONS

In analyzing the major architectural components of cloud computing systems, a number of important differences are apparent. Some of these differences have significant implications for security and performance. One such difference is the degree of flexibility. Distributed or decentralized cloud components are highly customizable. This is because the cloud computing system developers are forced to release an API in order to ensure compatibility and functionality among components. This also translates into support for third-party components, enabling best-of-breed cloud designs [22]. By contrast, centralized cloud components offer relatively few configuration options. This is because the components are often integrated into a series of related binaries which are installed and configured with the use of a package manager (e.g., apt, yum, etc.). The difference has implications for security. Cloud with relatively fewer configuration options and smaller APIs presents a smaller target to attackers. The reduced attack surface provides fewer options for exploitation. A second difference between centralized and decentralized cloud components is the degree of compartmentalization. Decentralized components provide enhanced security to cloud functionality. The services they provide are performed within isolation. If a decentralized component is compromised or encounters a runtime error which is unrecoverable, worst case scenario is that the component goes offline, while the rest of the cloud continues to function normally [23]. This design presents a failsafe which increases cloud stability [24]. On the other hand, centralized cloud designs interlace core functions within the same space. If an error or attack manages to destabilize a centralized component, the entire cloud is at risk. A third major distinction between clouds with centralized and decentralized components is performance. All cloud computing systems require several layers of overhead to support on-demand services. These layers provide resource management and abstraction at the cost of performance. By nature, virtualization is slower than native performance. On top of this, decentralized clouds require additional layers of software to maintain component independence [25]. The cost of this independence is performance. The additional processes and communications take time to complete. Considering the size and scope of public clouds, this performance drain is nontrivial [3]. This tendency is more likely to manifest in clouds with higher hardware utilization levels [2].

V. CONCLUSION

Cloud computing systems have become mainstream ecosystems for supporting all modes of computing. Their importance cannot be understated. Before selecting a cloud platform to implement or adopt, organizations would do well to consider the tradeoffs inherent in each design. Towards this goal, the present study analyzes high-level architectural differences in leading cloud platforms. It focuses on four basic architectural components. For each component, it considers the impact of centralized and decentralized design on cloud security and performance. From this analysis, a number of high-level differences have emerged. This analysis will prove valuable to both IT professionals and software engineers. Enterprise architects and information security professionals are often forced to make acquisition decisions with less than complete information. This research identified a number of factors to consider before implementing new technologies. However, the present study should not be considered comprehensive. Other design factors should be elicited in future research. This study should also be of interest to software engineers. Often, security is considered late in the software development cycle. By analyzing security implications at the architectural level, this research attempts to incorporate security concerns earlier in the design process. Future studies should focus on approaches to compensating for security weaknesses in each type of architecture.

ACKNOWLEDGMENT

This research is funded in part by a grant from the National Science Foundation, Division of Computer and Network Systems, Secure & Trustworthy Cyberspace Program (Award 1318399).

REFERENCES

[1] L. Youseff, M. Butrico, and D. Da Silva, "Toward a Unified Ontology of Cloud Computing," in *Grid Computing Environments Workshop*, Austin, TX, 2008, pp. 1-10.

[2] G. Gallizo, R. Kuebert, K. Oberle, A. Menychtas, and K. Konstanteli, "Service level agreements in virtualized service platforms," in *eChallenges e-2009*, Istanbul, Turkey, 2009, pp. 9-17.

[3] J. Gray and D. Siewiorek, "High-availability computer systems," *IEEE Computer,* vol. 24, pp. 39-48, 1991.

[4] F. Lombardi and R. Di Pietro, "Secure Virtualization for Cloud Computing," *Journal of Network and Computer Applications,* vol. 34, pp. 1113-1122, 2008.

[5] R. Nickelson, J. Muntermann, and U. Varshney, "Taxonomy Development in Information Systems: A Literature Survey and Problem Statement," in *America's Conference on Information Systems* Mina, Peru, 2010, pp. 178-189.

[6] S. Fu, "Failure-aware resource management for high-availability computing clusters with distributed virtual machines," *Journal of Parallel and Distributed Computing,* vol. 70, pp. 384-393, 2010.

[7] L. Schwartz. (2015) Five Common Misconceptions About Cloud Platforms. *Forbes*.

[8] S. Scott, C. Engelmann, C. Leangsuksun, and X. He, "Active/active replication for highly available HPC system services," in *The First International Conference on Availability, Reliability and Security. ARES 2006.* , Vienna, Austria, 2006, pp. 143-151.

[9] F. Lium, J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger*, et al.*, "NIST Cloud Computing Reference Architecture," National Institute of Standards and Technology2011.

[10] D. Nurmi, R. Wolski, C. Grzegorczyk, and G. Obertelli, "The Eucalyptus Open-Source Cloud-Computing System," in *9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, Shanghai, CN, 2009, pp. 124-131.

[11] OpenStack Docs Retrieved from http://docs.openstack.org/juno/config-reference/content/section_compute-scheduler.html

[12] https://region-b.geo-1.objects.hpcloudsvc.com/v1/10625742765718/generated-pdfs/Eucalyptus_1.0/eucalyptus-admin-guide-1.6.pdf

[13] http://www.vmware.com/files/pdf/VMware-Architecting-vCloud-WP.pdf

[14] https://en.wikipedia.org/wiki/Eucalyptus_(software)

[15] https://github.com/eucalyptus/eucalyptus/wiki/Storage

[16] http://docs.cloudstack.apache.org/en/master/concepts.html

[17] https://www.vmware.com/files/pdf/products/vcns/vmware-vcloud-networking-and-security-overview.pdf

[18] https://en.wikipedia.org/wiki/OpenStack#Networking_.28Neutron.29

[19] http://docs.cloudstack.apache.org/projects/cloudstack-administration/en/4.5/systemvm.html

[20] http://docs.openstack.org/developer/glance/architecture.html

[21] https://region-b.geo-1.objects.hpcloudsvc.com/v1/10625742765718/generated-pdfs/Eucalyptus_3.3/admin-guide-3.3.2.pdf

[22] J. Staten, "Cloud Computing Enters its Second Stage, Hypergrowth Ensues," presented at the Forrester Research, http://www.zdnet.com/article/cloud-computing-enters-its-second-stage-hypergrowth-ensues/, 2014.

[23] R. Ko, S. Lee, and V. Rajan, "Understanding Cloud Failures," *IEEE Spectrum,* vol. 2012, pp. 131-139, 2012.

[24] K. Bowers, A. Juels, and A. Oprea, "HAIL: a high availability and integrity layer for clouds," in *16th ACM Conference on Computer and Communications Security*, Chicago, ILL., 2009, pp. 187-198.

[25] C. Engelmann and S. Scott, "Concepts for High Availability in Scientific High-end Computing," in High Avilability and Performance Computing Workshop, Sante Fe, NM, 2005, pp. 21-29