Global Information Assurance Certification Paper

Cloud Testing Methodology

GIAC (GSEC) Gold Certification

Author: Edward Zamora, ezamora@mastersprogram.sans.edu
Advisor: Barbara Filkins
Accepted: July 9th 2015

Abstract

The time has come where the society at large is living in the cloud. Many have questioned the security of information in the cloud and many have been told that information is safe there. But how can one be sure that information is indeed safe in the cloud? In this day and age where there is an increased dependence on such complex technology as cloud systems, there are needs for methodologies to test cloud deployments. For organizations that have or seek to implement cloud technology in their environment, this paper will present a brief background on cloud technology and a methodology for assessing the security of their cloud implementation based on penetration testing principles.

1. Introduction

Cloud computing has become ubiquitous in our society and across the world. The average person checks their smartphone 221 times a day looking at emails, browsing the Internet, and using smartphone apps (Tecmark, 2014). According to a recent Gallup poll, 41% of Americans look forward to checking their email every day (Newport & Ander, 2015). Cloud technology is used for hosting popular services including email, social media, and business applications. A recent survey shows that 82% of large enterprises are now using cloud computing (RightScale, 2015). A market analysis of small businesses forecasts 78% of small businesses will fully adopt cloud technology in the next few years (King, Hicks, & Reeves, 2014). Organizations of all sizes have decided to transfer the risk of managing data storage, applications, and even infrastructure to cloud providers (Catteddu, Hogben, Haeberlen, & Dupre, 2012). However, some businesses have decided that the benefits of using cloud technology internal to their organization includes having controlled access to sensitive data, customized implementations that are optimal for an organization, and offering cloud based sharing services to partners (Babcock, 2009). Likewise many other companies have decided that it would beneficial to deploy a mix of cloud technology internal and external to their network (Neovise, 2013).

Even with all the benefits of using cloud technology, in a recent survey 56% of people stated they do not fully understand how the cloud works (Wakefield Research, 2012). There are several public resources that help describe cloud technologies. Yet, the inner working of cloud technology remains complex and difficult to understand. This unfamiliarity and complexity of cloud engineering has prompted the need for methodologies that can be used to test one's security in a cloud implementation (Cloud Security Alliance, 2013).

The information security community will benefit from a methodology that ties together an understanding of cloud technology with the application of penetration testing techniques. It is necessary to begin with an understanding of the characteristics of cloud computing and the different service and deployment models. Armed with this knowledge a security tester can apply penetration testing techniques to validate the defenses and

configuration of cloud components. The use of such a guideline will enable cloud engineers, system administrators, and network defenders to be better prepared to defend their cloud infrastructure and components.

2. Cloud Computing Essentials

2.1. The Characteristics of Cloud Computing

Cloud computing consists of the set of systems and services working in unison to provide distributed, flexible, and measureable resources to consumers of cloud services. The National Institute of Standards and Technology (NIST) defines cloud computing as a model that consists of on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service (Mell & Grance, 2011). Essentially, cloud computing allows consumers to provision for themselves resources available from a cloud services provider. Consumers are able to access their cloud resources from a wide variety of devices including mobile, thin clients, and traditional desktops. The following is a representation of the characteristics of cloud computing.

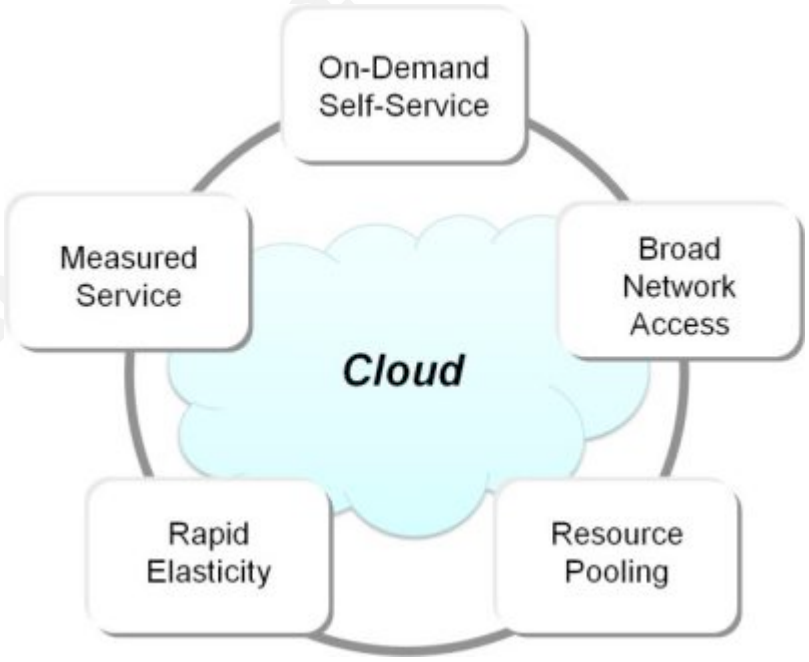

Figure 1. These are the characteristics of cloud computing (The Open Group, n.d.)

Physical and virtual systems are combined to provide consumers with resources dynamically without the user needing to know the details of how it all works. Combined

computing power and dynamic resource provision allows a consumer to quickly provision resources at a large scale in a short amount of time. The cloud service provider is then able to set different cost models according to a variety of metrics consisting typically of CPU processing cycles used, time of use, bandwidth used, and others (Al-Roomi, Al-Ebrahim, Buqrais, & Ahmad, 2013).

Cloud services are generally grouped in three broad categories: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). **Software as a Service (SaaS)** provides users with the ability to access applications over a network (Badger, Grance, Patt-Corner, & Voas, 2012). Organizations and users that use SaaS interact with an application and offload the supporting operating systems and infrastructure to the cloud provider. Examples of SaaS software include email, virtual desktop, and office productivity applications over the Internet.

Platform as a Service (PaaS) provides developers with an environment for creating custom applications without the need to be concerned with managing the underlying operating systems and infrastructure (Eamonn, 2013). Developers are given environments such as .NET, Java, Python, and many other languages through a PaaS provider's website. They are able to write code by using the website interface or upload code they have created using version control software like git. Developers can then publish their code on the PaaS provided website. PaaS providers are able to automatically scale the power of a developed web application according to its use or other metrics (Red Hat, 2014).

Infrastructure as a Service (IaaS) provides the most control over resources of all the service options. IaaS gives system administrators access to storage, networking, virtual machines, and other low level infrastructure. Administrators are able to rapidly provision virtual machines, deploy operating systems, and manage them via their web browsers. The virtual machines can then be used to host different development platforms for developers and software that end users can access. One major difference between IaaS and the other two cloud service types is that the responsibility of the configuration and security of the operating systems do fall on the system administrators. A representation of the cloud service models is shown in the following diagram.

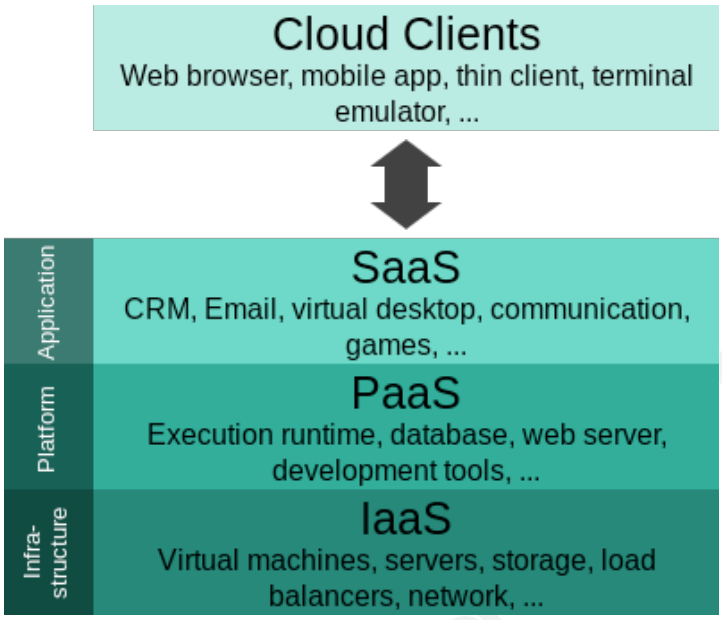

Figure 2. Cloud Service Models (Bikeborg & Wylve, 2013)

Cloud services are deployed in different ways depending on the requirements for sharing cloud resources. An organization's cloud sharing requirement may be to keep the cloud resources available only internally. This is useful when the need is to keep information private (Babcock, 2009). Thus, the deployment model is referred to as a **Private** deployment. A company may choose to share or use cloud services within a community of interest. This form of deployment is referred to as the Community deployment model.

Most users are familiar with cloud services provided to the public and open to anyone such as public email. The deployment model for this type of cloud deployment is the **Public** deployment model. Lastly, the **Hybrid** deployment combines any of the Private, Community, and Public deployment models for a customized implementation. A representation of these deployment models is shown below.

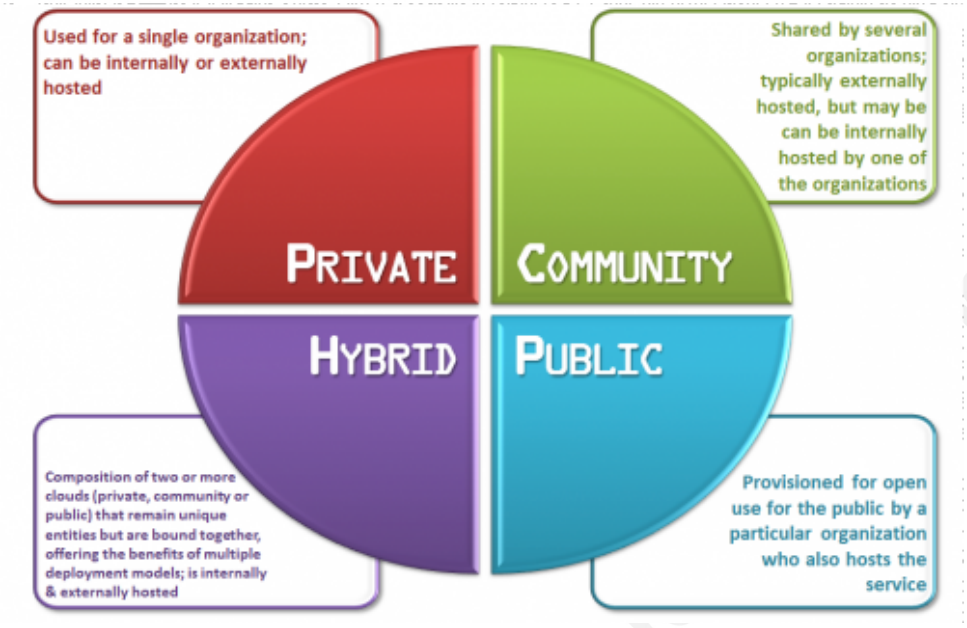

Figure 3. Cloud Deployment Types (Sultana, 2014)

2.2. Cloud Computing Systems

In addition to the essential characteristics of cloud computing, cloud service, and deployment model types there exist the systems that comprise an actual cloud deployment. One of the many benefits of using cloud technology is the flexibility with which one can combine hardware and virtual resources to build out the systems in a cloud deployment (Mell & Grance, 2011). These set of systems can be organized according to different architecture patterns that are optimized for an organization.

There are currently several types of architecture models that engineers can use (Fehling, Arbitter, Schupeck, Retter, & Leymann, 2014). Across these and other architecture models there are common functional components that are foundational to them all. Each cloud deployment at minimum contains a server or cluster of servers dedicated to the functions of computing, networking, storage management, and shared services of a cloud instance (OpenStack, n.d.). The OpenStack project represents an industry accepted standard for implementing cloud technology. A representation of these and other functions in an implementation of OpenStack is shown in the following figure.

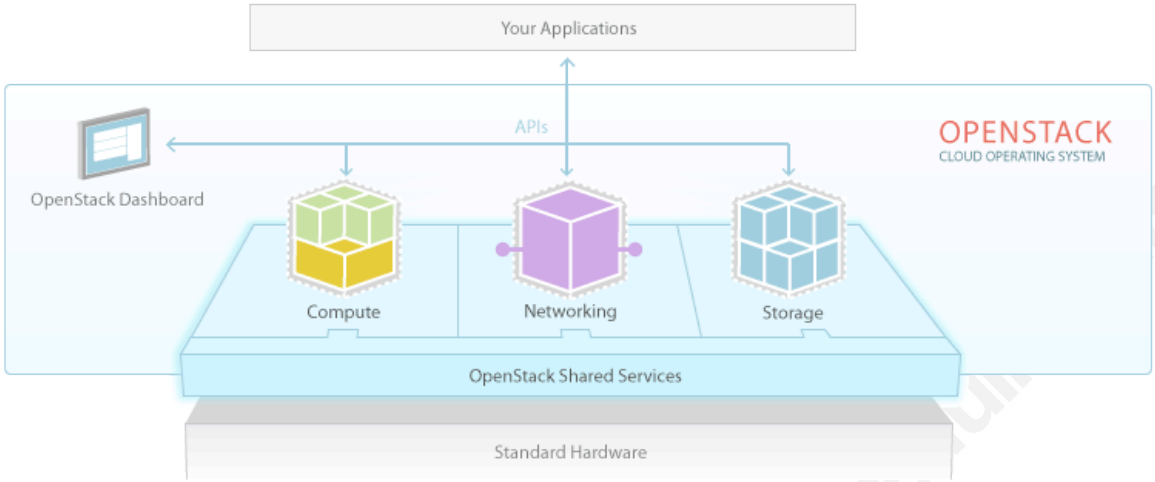

Figure 4. High Level Overview of Essential Cloud Components (OpenStack, n.d.)

In OpenStack, the compute node handles the deployment of virtual machines at a mass scale and computing tasks (Butler, 2014). The networking node is used to facilitate communication between cloud nodes and provisioned resources. Storage nodes are used for the distributed storage of data. Applications would be stored on a file system that is in turn stored in block storage on a storage node. Notable shared services include the identity service used for authentication to cloud resources and the image service used for storing the images used in virtual machines.

Depending on the priorities for deploying a cloud environment and the resources available, these components can be broken down further into sub categories or nodes. For example, OpenStack allows for the separation of the storage node into the block and object storage nodes. This option optimizes the bandwidth usage of object access and I/O performance of block storage (OpenStack, n.d.). Understanding the functional components involved in cloud technology is necessary to develop a good testing methodology. A representation of the different components of a sample OpenStack implementation is shown in the figure below.

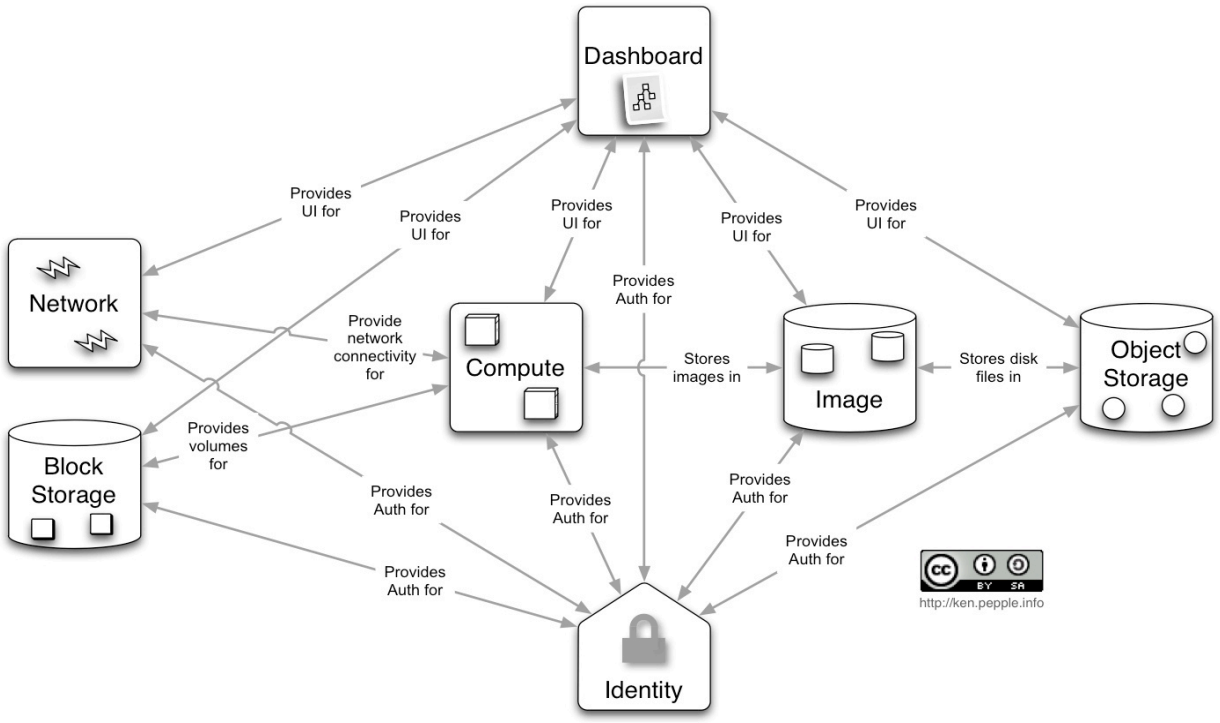

Figure 5. High Level Overview of Essential Cloud Components (Pepple, 2012)

3. Assessment Methodology

3.1. Assessment Approach

Armed with an understanding of cloud technology, a tester can begin the process of planning and scoping the assessment of a cloud deployment. The several components of cloud technology and complex architectural layouts for deployment present a large surface for attack. With such a large area to cover, it is important to know where to begin and what components are the priorities for testing.

As in many scenarios that test security, it is important to test the ability of unauthorized users gaining administrative privileges. Administrative privileges in different parts of a cloud environment will yield different levels of access to information and functionality. Penetration testing techniques are useful in demonstrating the ability to perform reconnaissance, exploit, gain administrative privileges, and compromise sensitive information. The challenge is adapting these techniques to a cloud instance.

SaaS deployments provide users with access to software hosted by cloud providers. PaaS deployments also provide end users with an interface to a web application created or configured by a developer through web browsers. IaaS deployments typically provide users with an interface to the virtual machine through a web browser. The virtual machine can then also be used to host web applications served out to others. In each of these cases, the common denominator is the access to these resources through the web browser. It is because of this that the approach to test the security in each of these deployment types will be similar (Cloud Security Alliance, 2013).

3.2. Access to Software/Applications

The functional component that provides the primary initial entry vector is the dashboard interface used to access cloud resources through a web browser. For SaaS deployments, users have access to the cloud software over a network typically through a web browser. For PaaS deployment, end users have access to the web application created by the developer through their web browser as well. Web application penetration testing techniques are helpful in this scenario for searching exploitable vulnerabilities of software and web applications.

For example, input validation vulnerabilities are a category of issues related to developers not properly validating user input on their web applications (Morana & Nusbaum, 2008). Executing code on backend systems using input validation vulnerabilities may yield access to underlying cloud infrastructure. Unauthorized access to cloud infrastructure may also take place if users are able to bypass authentication mechanisms using input validation vulnerabilities. One of the benefits of cloud computing for SaaS and PaaS is being able to transfer the risk of the cloud infrastructure security to a cloud provider (Catteddu, Hogben, Haeberlen, & Dupre, 2012).

However, in the worst case scenario, execution could take place on the underlying operating system hosting the web application or a cloud node. This ultimately depends on how the SaaS or PaaS cloud provider has configured their cloud implementation. When an organization assumes more control over the cloud infrastructure, they take on more of this risk. A sample architecture model of shared cloud resources is shown below using

the Docker Engine that acts as a hypervisor and provides a layer of abstraction from the underlying operating system in the OpenStack framework.

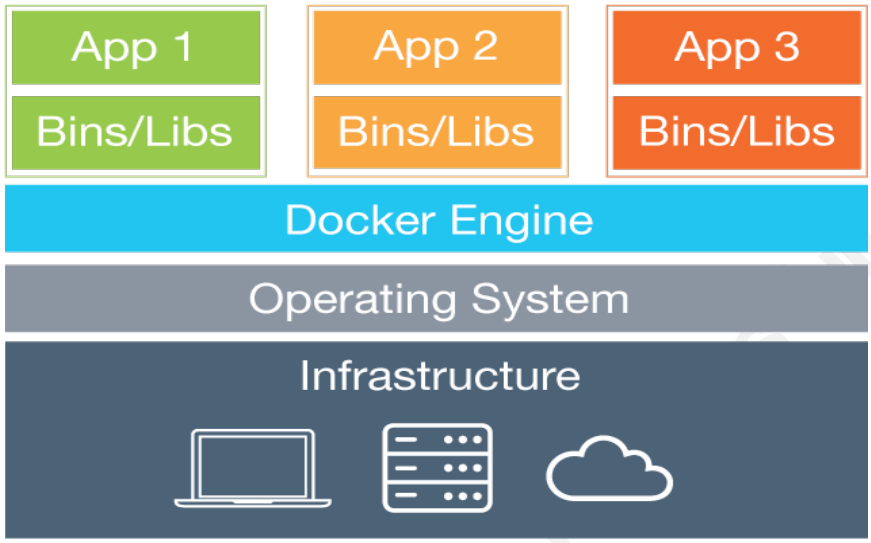

Figure 6. Docker Hypervisor Example in OpenShift (Docker, 2015)

If a published web application allows for a persistent type entry of user input, such as a blog, then additional discovery techniques are possible. This is especially true for cross-site scripting and cross-site request forgery. There are several types of tests that can be run for web application security testing (The OWASP Foundation, 2015). Automated web application scanners help to scan for these and many other kinds of web application related vulnerabilities.

Upon the discovery of vulnerabilities, a tester can attempt to exploit web applications to gain privileged access to the main dashboard or execution on the underlying operating system. This will help determine the effectiveness of any security monitoring and configuration present. The successful exploitation of vulnerabilities in the software or web application in this case will also yield administrative access to the underlying operating system or the cloud dashboard.

Administrative access to the underlying operating system opens up new avenues for malicious attacks and gets closer to the underlying cloud infrastructure. Privileged access to dashboards may allow users to view administrative consoles that contain privileged information related to the cloud implementation configuration, hosted services,

and other tenants. The use of web penetration techniques will be beneficial in assessing the security of the most public facing boundary to cloud resources (Fraser, 2015).

3.3. Access to Virtual Machines

In addition to software and web application access through a web browser, users can access virtual machines through a web browser as well. The dashboard interface displays a virtual machine in the form of a remote desktop or simulated kvm-type access. System administrators are able to interact with a virtual machine as if they were physically in front of the server or workstation. With this type of access, the goal becomes to learn about the environment the virtual machine resides in and the functionality available within the virtual machine. An example of a dashboard is shown below in the OpenStack project that provides access to a virtual machine through a web browser.

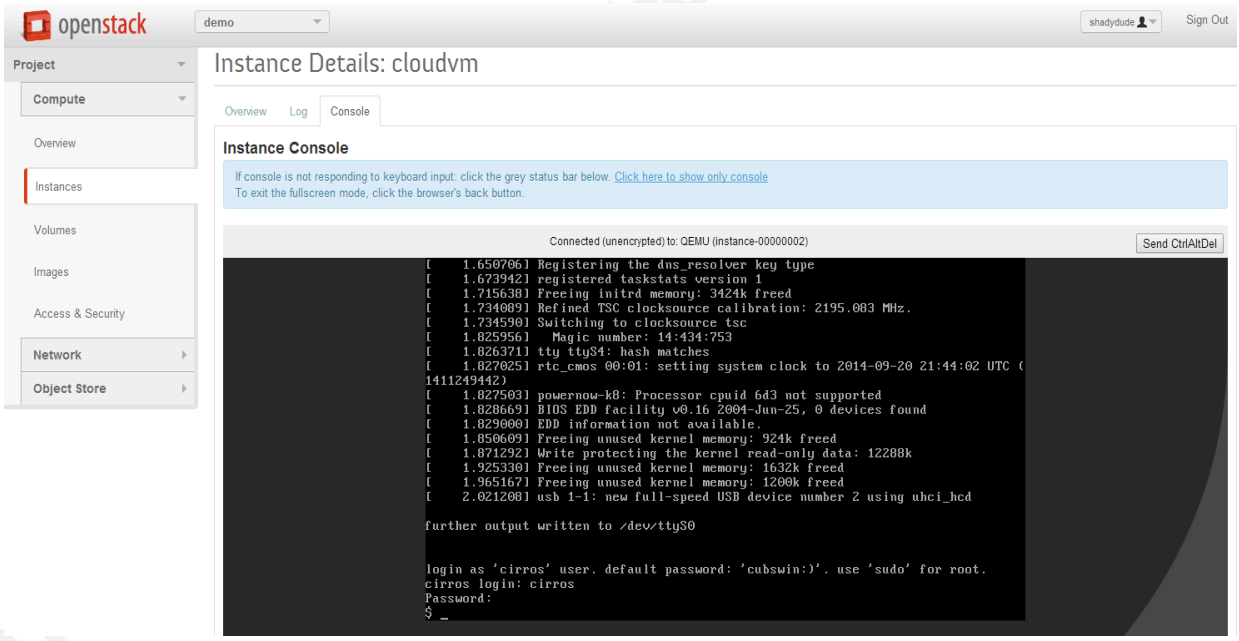

Figure 7. Virtual Machine Access through a Web Browser

3.3.1. Research Functionality of Virtual Machine

Once access to a virtual machine is acquired, a user will be presented with a command prompt or a desktop. The next step is to test the capabilities and level of access granted to default users on the virtual machine. There are many useful methods of doing this that can be borrowed from the penetration testing community. The commands used to

find this information will also vary across operating systems and are not dependent on the type of cloud deployment or framework used.

For example, in UNIX variant operating systems, running the ***id*** command will show the current user's information including group membership. Of particular interest is if the user is a member of a privileged group such as 'root' or 'wheel'. For Windows systems, the ***net user <current username>*** command from a command line will also provide information about the user's group membership. The interest is to verify if membership of the user is in the administrator or some domain level administrator group for privileged access. The image below shows example results.

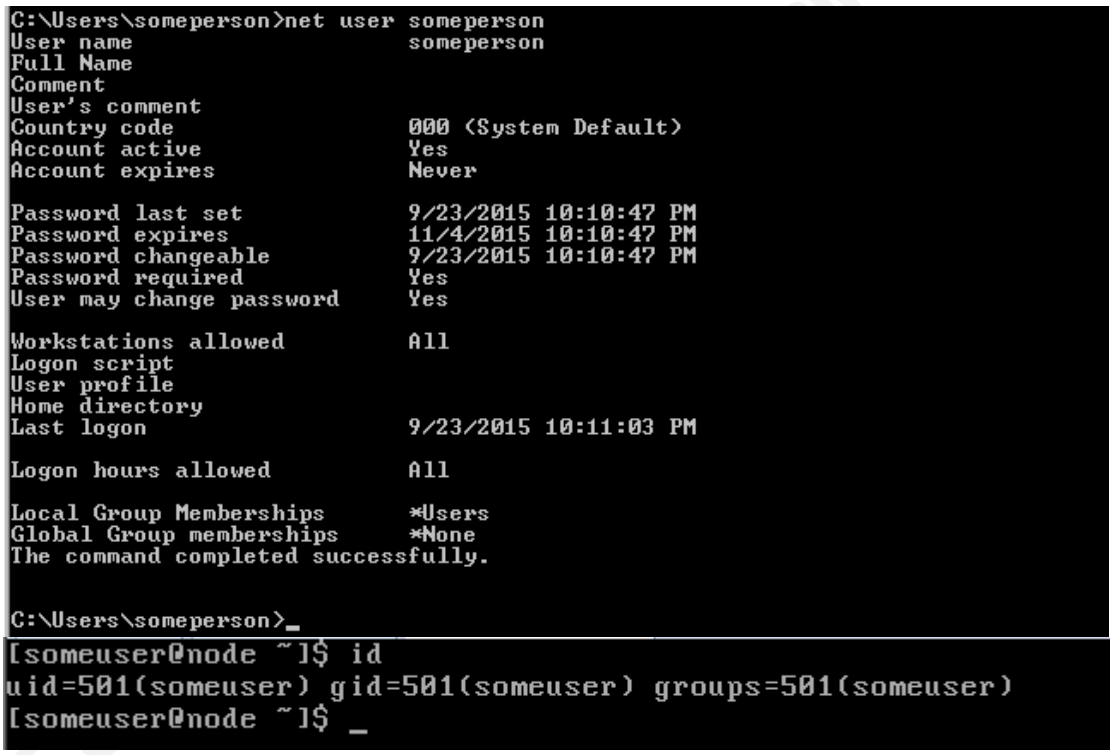

Figure 8. Windows and Linux user information commands

In addition to learning the user's group membership and group membership privileges, it is important to test whether administrator privileged programs are made available to all users. Programs such as ssh, nmap, and remote desktop clients are of particular interest because these programs facilitate connections or surveying other potential systems on the network. The most typical provision of a virtual machine will

provide a user with root or administrator access. This gives a system administrator full control of their virtual machine to create and deploy applications in the cloud.

It will be useful to know all of the installed programs and program versions on a virtual machine. This can be discovered searching the Programs and Features section of the Control Panel in Windows systems. For Linux or UNIX systems, this will vary according to distribution. A couple of the more common methods include ***rpm -qa*** for Red Hat and ***dpkg-query –l*** for Debian Linux variants. Programs that are used for administering systems should be closely monitored. If these programs are not necessary by the common user base, it may even be better that these programs be restricted from common users. Below is an example of checking for programs on a CentOS system.

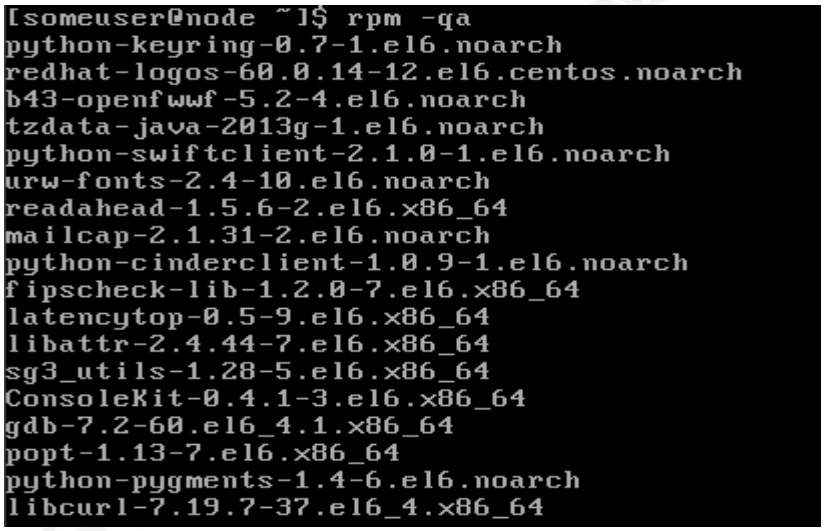

Figure 9. Program query on CentOS Linux variant

It is important to know whether an application repository is available for installing programs to the virtual machine. The ability to search a repository and install programs from it may provide additional functionality not built in by default. Searching for programs including netcat, telnet, and even Metasploit open new opportunities for a user to use a system in an unintended manner. Virtual machine host commands would also be useful in learning more about the virtual environment. Commands such as ***systeminfo*** on Windows and ***virt-what*** on Linux may show the virtual machine's platform. This kind of information is very important for testing potential misconfigurations of an underlying virtual hosting application.

A tester should verify if low level and high level programming applications are also available to use. Virtual machines that can compile programs can also compile exploits. Using low level programming language applications like the Gnu Compiler Collection (gcc) or Microsoft's Visual Studio can be used to develop with such applications. High level or interpreted programming languages have become increasingly versatile and capable for creating programs that can be used for exploits without needing to worry as much about the low level details. Python is such an object oriented, interpreted, high level language (Python Software Foundation, 2015).

3.3.2. Perform Connectivity Research

After learning the available capabilities in the virtual machine, network connectivity should be investigated and tested. A tester will benefit from learning the reach that a virtual machine has in a cloud environment. Several commands exist to learn what the virtual machine is connected to. Commands such as ***ipconfig***, ***arp***, and ***netstat*** will provide information about network interfaces and active connections on Windows machines. The common equivalent commands on Linux variants are the ***ifconfig***, ***arp***, and ***netstat*** commands. Devices that a virtual machine is connected to may be targets for attack that need to be defended and should be tested.

The ***ipconfig*** and ***ifconfig*** commands give the user information about the network interface card (NIC) on the virtual machine. The important information to glean from the configuration includes the complete information about the virtual machine's NIC or NICs. This includes the assigned ip address or addresses, gateway ip, dhcp provider, and dns resolver. Each of these systems are expected to connect to the virtual machine for functionality. These machines present a surface area that should be tested for their potential exposure.

The ***arp*** command will provide information about recent connections to the virtual machine's NIC that are on the same network. What this means is that devices that have communicated with the virtual machine that are on the local broadcast domain will be shown in this output. It is especially important to ensure these devices are properly protected since they are the devices that can directly communicate with the virtual machines. Combining this knowledge with use of the previous programs found, such as

nmap, a tester can search for vulnerabilities on these systems and secure them. The results from the ***netstat*** command will also provide the ports associated with active or recently active connections. This is particularly useful to learn the types of connection to or from other devices. For example, ***netstat*** results that show outbound connections to port 443 are most likely showing https connections using SSL. These types of connections consist of secure browsing to a webpage. Outbound connections to a host on port 445 may likely indicate a connection to an Active Directory domain controller, a file share, or a samba share. Either of these present opportunities to malicious users for attacking connected machines and should be tested.

Inbound listening ports present potential vulnerabilities on the virtual machine itself. For example, listening connections on port 22 shows that the virtual machine is listening for SSH connections. Depending on the configurations of the SSH service and networking infrastructure, this means that it may be possible to connect to the virtual machine over SSH. Inbound listening on port 3389 may also indicate the ability to connect to the virtual machine for remote desktop interaction.

It is useful to scan the range of ip addresses associated with the virtual machine. This will provide a list of devices and services that can be connected to from a virtual machine. A tester that runs different types of nmap scans will observe the responses of devices that are on the network. For example, running a ping sweep scan across the same network as the virtual machine should yield other devices that exist on that network configured to respond to echo requests. The information discovered with ipconfig or ifconfig can be used to run nmap scans on those networks also. In the worst case scenario, there could be other tenant machines or cloud infrastructure nodes that are accessible from the virtual machine.

In the case where name resolution exists for the virtual machine, the ***nslookup*** command is helpful to identify any other machines that resolve hostnames to ip addresses or vice versa. Once having discovered hosts using nslookup, these machines and networks can be added to the list of devices to scan. A security tester could repeat the process of scanning the hosts thoroughly for any vulnerability. An organization would

have to determine whether connectivity to these devices from a tenant virtual machine is absolutely necessary.

3.3.3. Test Exploitation of VM or Connected Devices

After the discovery of connected systems and other virtual machines, testing the ability to compromise those systems from the virtual machine is the next step for security testing. All of the information previously collected can be used to test the defense against exploitation of connected devices from a virtual machine. The installed programs and discovered devices on networks are the components needed for this stage of testing.

If the initial account was not an administrative account the tester should verify the ability to run exploits to escalate privileges. If the ability exists to download exploit programs onto the virtual machine, these programs should be executed to see if the user's privileged access changes from standard user to administrator or root. Many exploits exists online in the form of executable programs and programming code. A comprehensive set of exploits also exist in frameworks such as the Metasploit framework (Rapid7, 2015).

An additional related test would be to verify if a user can download exploit code and compile it on the virtual machine. The compiled executable could then be run and checked for success and detection. Cloud system owners should determine whether this capability for standard users is necessary. If a user's access is administrative on the virtual machine, then there could be little to no restrictions on that local machine. The only limitations would be the availability of programs and connectivity to other devices.

Devices on a network that the virtual machine can communicate with should be tested. One valuable test is using SSH to try to connect to other discovered machines. SSH is a common protocol used to administer networking devices and servers. If a user from the virtual machine is successful in using SSH to connect to other such machines, this would provide the user an opportunity to log into those devices. Likewise, virtual machines may also have the SSH service listening for incoming connections. It would be useful to test whether a user from one virtual machine is able to connect to another tenant machine using SSH. This could be a clear vulnerability.

Along the lines of connecting to other machines, a useful test would be to use remote desktop clients to connect to other servers and desktops. Another method for remotely administering servers and workstations relies on the use of remote desktop. It would be a useful test to determine whether a standard virtual machine is able to connect via this method to unauthorized machines. Cloud system administrators would have to configure access control lists and permissions to tighten the security for this type of connectivity.

The typical goals at this stage are to test how far a user can go with the capabilities and information they have access to. This includes whether a user can use all the previously identified resources to gain unauthorized administrative privileges on local and remote systems. For example, remote exploits could exist for different software versions identified on a host. A tester could test these types of exploits to see if the current defenses detect the exploit attempts. A combination of different methods should be used to test the capability and detection of lateral movement and compromise of devices. In a worst case scenario, an average user would be able to access a cloud infrastructure node with administrative privileges and go undetected.

3.3.4. Use Cloud Resources

The eventual progression of successful penetration testing techniques yields administrative access on local and remote systems. When that system is a cloud infrastructure node, there are different commands that can be run to further test what a user can do with access to cloud infrastructure. Upon successful access to cloud infrastructure nodes, the ability to use the cloud infrastructure backbone to administer, modify, and access cloud resources expands significantly. Administrative access to cloud nodes in essence gives the user all access to data and functionality in that portion of a cloud environment.

The methodology used for learning about the virtual machine environment can be repeated to learn about the cloud node environment. The focus would be on testing the ability to learn about the different components of the cloud's functionality and network connectivity. Cloud components must communicate with each other in order to function

properly. A valuable set of tests involve the ability for a user with access to cloud nodes to learn about or access other cloud nodes.

For example, in the case of OpenStack, there are several key words used in the managing of cloud components that correspond to each node's functionality. An exhaustive list of OpenStack commands is available on their website (OpenStack, 2015). However, the following commands are a good baseline for assessing the functionality on essential OpenStack cloud components.

Administrative access connected to the dashboard's backend will most be on the controller node where the dashboard service runs. The controller node is the central manager of services that keeps all nodes working together in an OpenStack cloud implementation. It would be useful to test the ability to access other cloud nodes from a controller node. Many times, the controller node exists together with other primary nodes such as the compute or networking nodes. The steps to learn about the environment and connectivity could be repeated on this node. This includes running surveying commands on user access and network connectivity.

In the case where the controller and compute nodes are combined, the commands used to manage the compute node all start with ***nova***. Commands with ***nova*** allow users to manage virtual machine instances. This includes starting, stopping, and even adding data to a user's virtual machine instance (OpenStack, 2015). This ability to interact with tenant virtual machines should be tested.

To learn about the cloud implementation's networking configuration, the tester should test the network node. The commands used for network administration all use the ***neutron*** keyword. Using ***neutron*** to learn about the network addressing scheme, subnets, and vlan distributions would be helpful in learning the network topology of the cloud. The ***neutron*** command also provides information about firewall configuration that could also be useful for learning allowed protocols throughout the network.

```
[root@node ~(keystone_admin)]# neutron subnet-list
+--------------------------------------+----------------+----------------+-----
-------------------------------------------+
| id                                   | name           | cidr           | allo
cation_pools                                |
+--------------------------------------+----------------+----------------+-----
-------------------------------------------+
| 20e2883e-8bae-49dd-9e1d-1a41115c4695 | public_subnet  | 172.24.4.224/28 | {"st
art": "172.24.4.226", "end": "172.24.4.238"} |
```

```
[root@node ~(keystone_admin)]# neutron security-group-rule-list | more_

+--------------------------------------+----------------+-----------+----------+
------------------+--------------+
| id                                   | security_group | direction | protocol |
 remote_ip_prefix | remote_group |
+--------------------------------------+----------------+-----------+----------+
------------------+--------------+
| 12b825c1-0d7a-43a3-b07e-89e2c6b4654c | default        | egress    |          |
                  |              |
| 1acd2853-d4e3-4e09-8eaf-026705c45368 | default        | ingress   |          |
                  | default      |
| 22f08f34-ec99-4a9c-814e-0936fa50194d | default        | ingress   |          |
```

Figure 10. Neutron subnet and firewall rule query commands

It would be a good idea to test interaction on the identity node because of its importance in authentication. The keyword used for running commands in the identity node is ***keystone***. Running ***keystone*** commands will provide information about all the users in a cloud environment, their cloud environment variables, and allow the adding and changing user's passwords. One of the cloud variables gathered from the ***keystone*** commands is a user's id number. This is a long hexadecimal value that is stored with every user. Combining the use of nova and this id number, a user with access to these nodes can find any user and their cloud resources in a cloud instance. The identity node is very powerful node and its defense must be ensured.

Two other important nodes that need close attention are the data and image nodes. The keyword used for commands on the date node is ***swift***. Access to this node allows users to browse through the data stores in a default cloud implementation of OpenStack. A user can browse shares on data stores and search for files stored by users on the cloud. For running commands on the image node, the keyword is ***glance***. Access to the image node gives a user access to the virtual machine images that are deployed to users in the cloud. There are several more functions and nodes that exist in an OpenStack cloud implementation, but these all of these commands are good to start testing (OpenStack,

2015). The methodology for every organization should grow and be tailored to their cloud implementation and systems in their environment.

4. Conclusion

The use of the cloud can be seen everywhere in very obvious ways and also in many subtle ways. Cloud technology is very complex in structure and functionality, but can be better understood with helpful background information. The defining characteristics of cloud computing are the quickly provisioned, demand based, resource pooled, broadly networked, and measured service of cloud resources made available to users. There are different service models in cloud computing that correspond to types of resources being provisioned. These service models are the SaaS, Paas, and PaaS service models. In addition to the service models, there are deployment models that represent the type of distribution of cloud resources for a cloud implementation. This includes the private, community, public, and hybrid cloud deployment models.

Many different cloud providers such as Amazon, Google, and Microsoft have complex implementations of cloud technology in their environments. OpenStack provides a good reference implementation of cloud infrastructure that is used in many production environments. Primary functional components in OpenStack include the Computing, Networking, and Storage nodes that have keywords used to interact with the cloud infrastructure. These nodes can also be broken down into further functionality according to the requirements of an organization.

A security tester can perform several reconnaissance, scanning, and exploitation techniques on different components of a cloud implementation. Such a methodology can be expanded on and tailored for an organization's unique environment. Findings from such an assessment will help an organization find the spots in their cloud implementation that need tighter security configurations and monitoring. In spite of the complexity of cloud technology, organizations can apply proven penetration techniques to any cloud environment following a methodology. A summarized checklist of recommended steps for testing is available in Appendix A.

References

Al-Roomi, M., Al-Ebrahim, S., Buqrais, S., & Ahmad, I. (2013). Cloud Computing Pricing Models: A Survey. *International Journal of Grid and Distributed Computing*, *6*(5), 93-106. Retrieved from http://dx.doi.org/10.14257/ijgdc.2013.6.5.09

Babcock, C. (2009, April 9). Why 'Private Cloud' Computing Is Real -- And Worth Considering [Web log post]. Retrieved from http://www.informationweek.com/cloud/software-as-a-service/why-private-cloud-computing-is-real----and-worth-considering/d/d-id/1078460

Badger, L., Grance, T., Patt-Corner, R., & Voas, J. (2012). *Cloud Computing Synopsis and Recommendations* (SP 800-146). Retrieved from National Institute of Standards and Technology website: http://csrc.nist.gov/publications/nistpubs/800-146/sp800-146.pdf

Bikeborg, & Wylve. (2013, June 12). *Cloud computing layers* [Vector Image]. Retrieved from https://commons.wikimedia.org/wiki/File:Cloud_computing_layers.svg

Butler, B. (2014, May 19). OpenStack 101: The parts that make up the project [Web log post]. Retrieved from http://www.networkworld.com/article/2176963/cloud-computing/openstack-101--the-parts-that-make-up-the-project.html

Catteddu, D., Hogben, G., Haeberlen, T., & Dupre, L. (2012). *Cloud Computing Benefits, risks and recommendations for information security* (2.0 Rev.B). Retrieved from European Network and Information Security Agency website: https://resilience.enisa.europa.eu/cloud-security-and-resilience/publications/cloud-computing-benefits-risks-and-recommendations-for-information-security

Cloud Security Alliance. (2013). *The Notorious Nine: Cloud Computing Top Threats in 2013*. Retrieved from https://cloudsecurityalliance.org/group/top-threats/

Docker. (2015). What is Docker? Retrieved from https://www.docker.com/whatisdocker

Eamonn. (2013, August 27). What's The Difference Between SaaS, PaaS, IaaS? [Blog post]. Retrieved from https://www.computenext.com/blog/when-to-use-saas-paas-and-iaas/

Fehling, C., Arbitter, P., Schupeck, W., Retter, R., & Leymann, F. (2014). *Cloud Computing Patterns: Fundamentals to Design, Build, and Manage Cloud Applications*. Vienna, Austria: Springer-Verlag GmbH.

Fraser, G. (2015). *Tunneling, Pivoting, and Web Application Penetration Testing*. Retrieved from The SANS Institute website: https://www.sans.org/reading-room/whitepapers/testing/tunneling-pivoting-web-application-penetration-testing-36117

King, S., Hicks, T., & Reeves, J. (2014). *Small Business Success in the Cloud*. Retrieved from Emergent Research website: http://www.emergentresearch.com/

Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing* (SP 800-145). Retrieved from National Institute of Standards and Technology website: http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

Morana, M., & Nusbaum, S. (2008). *Input Validation Vulnerabilities, Encoded Attack Vectors and Mitigations*. Retrieved from The OWASP Foundation website: https://www.owasp.org/images/6/6c/Encoded_Attacks_Threats_Countermeasures_9_30_08.pdf

Neovise. (2013). *Public, Private and Hybrid Clouds: When, Why and How They are Really Used*. Retrieved from http://www.neovise.com/public-private-hybrid-cloud-computing-research

Newport, F., & Ander, S. (2015). *Four in 10 Americans Look Forward to Checking Mail*. Retrieved from Gallup, Inc. website: http://www.gallup.com/poll/182261/four-americans-look-forward-checking-mail.aspx

OpenStack. (2015). *OpenStack Command-Line Interface Reference*. Retrieved from OpenStack Foundation website: http://docs.openstack.org/cli-reference/content/index.html

OpenStack. (2015). *OpenStack command-line interface cheat sheet* (1.0.0). Retrieved from OpenStack Foundation website: http://docs.openstack.org/user-guide/cli_cheat_sheet.html

OpenStack. (n.d.). *Chapter 6. Storage Decisions*. Retrieved from OpenStack Foundation website: http://docs.openstack.org/openstack-ops/content/storage_decision.html

OpenStack. (n.d.). *OpenStack Open Source Cloud Computing Software*. Retrieved from OpenStack Foundation website: https://www.openstack.org/software/

Pepple, K. (2012, September 25). OpenStack Folsom Architecture [Blog post]. Retrieved from http://ken.pepple.info/openstack/2012/09/25/openstack-folsom-architecture/

Python Software Foundation. (2015). What is Python? Retrieved from https://www.python.org/doc/essays/blurb/

Rapid7. (2015). Metasploit. Retrieved from http://www.metasploit.com/

Red Hat. (2014). OpenShift Online. Retrieved from https://www.openshift.com/products/online

RightScale. (2015). *RightScale 2015 STATE OF THE CLOUD REPORT*. Retrieved from http://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2015-state-cloud-survey

Sultana, A. (2014, April 28). Cloud Computing- How much you know of it (Part-2) [Blog Post]. Retrieved from http://blog.sysfore.com/cloud-computing-introduction-part2/

Tecmark. (2014). *Tecmark Smartphone Usage Survey 2014*. Retrieved from http://www.tecmark.co.uk/smartphone-usage-data-uk-2014/

The Open Group. (n.d.). *Cloud Computing for Business : What is Cloud?* Retrieved from http://www.opengroup.org/cloud/cloud/cloud_for_business/what.htm

The OWASP Foundation. (2015). Web Application Security Testing Cheat Sheet. Retrieved from https://www.owasp.org/index.php/Web_Application_Security_Testing_Cheat_Sheet

Wakefield Research. (2012). *Partly Cloudy – About Cloud Computing*. Retrieved from Citrix website: https://www.citrix.com/content/dam/citrix/en_us/documents/go/wakefield-citrix-cloud-survery-guide.pdf

Appendix A – Summary Checklist

Recommended Testing Checklist:

- Know the cloud type you are working on (private, public, hybrid)
- Know where you are starting from (SaaS, PaaS, IaaS)
- Use web application penetration testing techniques to test for vulnerabilities in software (SaaS), software platforms (PaaS) and virtual machine (IaaS) dashboards made available through web browsers with the goal of accessing the underlying operating system or backend cloud infrastructure
- With access to a virtual machine or underlying operating system, test whether users are able to research the functionality available on the virtual machine by searching other users, installed programs and hardware identifying information
- Test whether users are able to research connectivity information on the virtual machine that would show the ability to connect to other networks, cloud devices, and tenant virtual machines
- Use penetration testing techniques to test if users would be able to exploit vulnerabilities in the local virtual machine and discovered connected devices
- With access to cloud infrastructure, use native cloud commands to test the ability for users to learn about and compromise cloud infrastructure with the level of access gained
- Compile findings and engage the appropriate staff to implement necessary detection, prevention, and remediation capabilities