ISSN 2229-5518

Cloud Security using Honeypot Systems

Nithin Chandra S.R, Madhuri T.M

Abstract— Cloud computing systems fundamentally provide access to large pools of data and computational resources through a variety of interfaces similar in spirit to existing grid and HPC resource management and programming systems. These types of systems offer a new programming target for scalable application developers and have gained popularity over the past few years. However, most cloud computing systems in operation today are proprietary, rely upon infrastructure that is invisible to the research community, or are not explicitly designed to be instrumented and modified by systems researchers. Customers are both excited and nervous at the prospects of Cloud Computing. They are excited by the opportunities to reduce capital costs, for a chance to divest themselves, of infrastructure management and focus on core competencies, they are excited by the agility offered by the on-demand provisioning of computing and the ability to align information technology with business strategies and needs more readily. However, customers are also very concerned about the risks of Cloud Computing, if not properly secured there is loss of direct control over systems for which they are nonetheless accountable.

In this work, we present Cloud Security using Honeypots – Honeypots are an exciting new technology with enormous potential for the security community. The concepts were first introduced by several icons in computer security, specifically Cliff Stoll in the book "The Cuckoo's Egg", and Bill Cheswick's paper " An Evening with Berferd." The purpose of this paper is to explain how honeypots are used for securing cloud systems, their advantages and disadvantages, and their value to the security.

Keywords-Cloud security;Honeypot;Types of Honeypots; Advantages and Disadvantages

1. INTRODUCTION

IT Security instantly becomes an issue for anyone who connects their system to the Internet. Security threats range from hacking intrusions, denial of service attacks to computer worms, viruses and more. We must understand that intrusion to a network or system can never be eliminated but however, can be reduced. Computer crimes are always increasing. Countermeasures are developed to detect or prevent attacks - most of these measures are based on known facts. Security activities range from keeping intruders out of the network or system, preventing the interception of information sent via the Internet to limiting the spread of and damage caused by computer viruses.

The three concepts in IT Security are prevention, detection and respond. There is no end-to-end security equipment or solution that can cover two or all the concepts. For example, firewalls and anti-virus would fall under prevention, intrusion detection system and vulnerability scanners under detection and incident response teams would come under respond. Comprehensive security solutions include a mixture of software and hardware components. But however, honeypots fall under two main categories, Detection and Respond. Honeypots collect as much information as possible on the attack. The honeypot should operate in stealth mode so that the attacker would not know of its presence.

2. SECURITY IN THE CLOUD

2.1 Cloud Security Overview

Cloud service providers are lever-aging virtualization technologies combined with self-service capabilities for computing resources via the Internet. In these service provider environments, virtual machines from multiple organizations have to be co-located on the same physical server in order to maximize the efficiencies of virtualization. Cloud service providers must ensure that their customer's applications and data are secure if they hope to retain their customer base and competitiveness. Today, enterprises are looking toward cloud computing horizons to expand their on-premises infrastructure, but most cannot afford the risk of compromising the security of their applications and data Security is ranked first as the greatest challenge or issue of cloud computing. This paper identifies current security concerns about cloud computing environments and describes the methodology for ensuring application and data security and compliance integrity for those resources that are moving from on-premises to public cloud environments. More important, this discussion focuses on why and how these resources should be protected in the Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infra-structure-as-a-Service (IaaS) environments and offers security "best practices" for service providers and enterprises that are in or are contemplating moving into the cloud computing space. Software-as-a-Service is a model of software deployment in which an application is licensed for use as a service provided to customers on demand. On-demand licensing and use relieves the customer of the burden of equipping a device with every application to be used. Platform-as-a-Service is an outgrowth of the SaaS application delivery model. With the PaaS model, all of the facilities required to support the complete life cycle of building and delivering web applications and services are available to developers, IT managers, and end users entirely from the Internet, without software downloads or installation. PaaS is also some-times known as "cloud ware." PaaS offerings include workflow facilities for application design, application development, testing, deployment, and hosting, as well as application services such as team collaboration, web ser-vice integration and marshalling, database integration, security, scalability, storage, persistence, state management, application versioning, application instrumentation, and developer community facilitation. These services are provisioned as an integrated solution over the web. Infrastructure-as-a-Service is

the delivery of computer infrastructure (typically a platform virtualization environment) as a service. These "virtual infrastructure stacks" are an example of the everything-as-a-service trend and share many of the common characteristics. Rather than purchasing servers, software, data center space, or network equipment, clients buy these resources as a fully outsourced service. The service is typically billed on a utility computing basis, and the quantity of resources consumed (and there-fore the cost) typically reflects the level of activity. Inspired by the IT industry's move toward SaaS, in which software is not purchased but rented as a service from providers, IT-as-a-Service (ITaaS) is being proposed to take this concept further, to bring the service model right to your IT infrastructure. The modern IT organization must run itself as a separate operation and become more strategic in operational decisions. Many organizations are in the process of transforming their IT departments into self-sustaining cost-center operations, treating internal users as if they were customers. This transformation is not trivial and usually involves elements of project portfolio management, workflow reengineering, and process improvement. Many large IT organizations have adopted the Information Technology Infrastructure Library (ITIL) framework to help with this transformation. Organizations can harness their help desks, avoid downtime resulting from unauthorized changes, and deliver better service to their internal customers simply by adopting best practices for managing service requests, changes, and IT assets. The adoption of IT-as-a-Service can help enterprise IT functions focus on strategic alignment with business goals.

. While enterprises cope with defining the details of cloud computing, the single, unifying theme is service. Cloud computing, on-demand applications, and managed security are now perceived as part of an emerging ITaaS paradigm. Current industry buzz seems to reinforce the message that significant investments of capital, time, and intellectual resources are indeed being directed toward offering next-generation information and communication technology (ICT) infrastructure, which may allow enterprises to out-source IT completely and confidently. Many in the industry believe that the advent of developer platforms designed for the cloud will hasten this transition and, as a result, fewer enterprises will need to deploy middleware to manage patchwork-implemented business applications, legacy or otherwise. Infrastructure vendors are also jumping on this bandwagon. Amazon has been a pioneer, with the release of Amazon S3 (Storage-as-a-Service).

There are some key financial benefits in moving to an ITaaS model, such as not having to incur capital costs; having a transparent monthly pricing plan; scalability; and reasonable costs of expansion. Operational benefits of ITaaS include increased reliability because of a centralized infra-structure, which can ensure that critical services and applications are monitored continually; software flexibility, with centrally maintained products that allow for quick rollout of new functionalities and updates, and data security, since company data can be stored on owner-managed premises and backed up using encryption to a secure off-site data center. Another service that is being discussed as we are writing this paper is the concept of Anything-as-a-Service (XaaS), which is also a subset of cloud computing. XaaS broadly encompasses a process of activating reusable software components over the network. The most common and successful example is Software-as-a-Service. The growth of "as-a-service" offerings has been facilitated by extremely low barriers to entry (they are often accessible for free or available as recurring charges on a personal credit card). As a result, such offerings have been adopted by consumers and small businesses well before pushing into the enterprise space. All "as-a-service" offerings share a number of common attributes, including little or no capital expenditure since the required infrastructure is owned by the service provider, massive scalability, multitenancy, and device and location independence allowing consumers remote access to systems using nearly any current avail-able technology. On the surface, it appears that XaaS is a potentially game-changing technology that could reshape IT. However, most CIOs still depend on internal infrastructures because they are not convinced that cloud computing is ready for prime time. Many contend that if you want real reliability, you must write more reliable applications. Regardless of one's view on the readiness of cloud computing to meet corporate IT requirements, it cannot be ignored. The concept of pay-as-you-go applications, development plat-forms, processing power, storage, or any other cloud-enabled services has emerged and can be expected to reshape IT over the next decade. Other concerns plague IT executives. They fear their data won't be safe in the hands of cloud providers and that they won't be able to manage cloud resources effectively. They may also worry that the new technology will threaten their own data centers and staff. Collectively, these fears tend to hold back the cloud computing market. Although there is a significant benefit to leveraging cloud computing, security concerns have led organizations to hesitate to move critical resources to the cloud. Corporations and individuals are often concerned about how security and compliance integrity can be maintained in this new environment. Even more worrying, however, may be those corporations that are jumping into cloud computing that may be oblivious to the implications of putting critical applications and data in the cloud. This paper will answer the security concerns of the former and educate the latter. Moving critical applications and sensitive data to public and shared cloud environments is of great concern for those corporations that are moving beyond their data center's network perimeter defense. To alleviate these concerns, a cloud solution provider must ensure that customers will continue to have the same security and privacy controls over their applications and services, provide evidence to customers that their organization and customers are secure and they can meet their service-level agreements, and that they can prove compliance to auditors. This paper is to define how honeypots add their value to the cloud security community

3. HONEYPOT IN CLOUD

A honeypot is a security resource whose value lies in being probed, attacked, or compromised. This means that whatever we designate as a honeypot, it is our expectation and goal to have the system probed, attacked, and potentially exploited. Honeypot is a detection and response tool, rather than prevention which it has a little value in. Because honeypots cannot prevent a particular intrusion or spread of virus or worm, it merely collects information and detects attack patterns. After doing so, the defenders can respond to this evidence by building better defenses and countermeasures against future security threats. A honeypot is a tool to collect evidence or information, and to gain as much knowledge as possible especially on the attack patterns, hacker's purpose and motivations and the commonly used programs launched by them. From all the information received, we can even learn more about the hacker's ability especially their technical knowledge. I can say this is the main objective of a honeypot.

Honeypots can also be used to catch hackers while they are in the network and to redirect hackers from the actual production systems to the honeypot system. The best personnel to manage the honeypot is one with extensive knowledge in three critical areas – Security, Systems, and Networks. In the right hands, a honeypot can be an effective tool for information gathering – In the wrong, inexperienced hands; a honeypot can become infiltrated machine and an instrument for the blackhat community. Honeypots never assist to upgrade the network security of an organization, but it rather lures hackers into the network. Honeypots are security tools that have no real or production value. It should not be communicated by anyone. If there is an activity or traffic to the honeypot, this can be suspected as an intrusion, unauthorized access or a probing attempt. And rather, if there are any outgoing connections initiated by the honeypot, there is a high possibility that the honeypot has been compromised and taken over. A production honeypot is used to assist an organization in protecting its internal IT infrastructure whereas a research honeypot is used to accumulate evidence and information in order to study hackers' or the black hat criminal attack patterns and motives. Honeypots add very little value in prevention – They WILL NOT keeps the black hat hackers out. The organization still need to depend on their security policies, procedures and best practices such as disabling unused services, patch management, implementing security mechanisms such as firewall, intrusion detection systems, anti-virus and secure authentication mechanisms to keep the blackhat community out of the organization's IT infrastructure. There is a need to ensure that the production honeypot is properly built, as if it is incorrectly implemented, the hacker can easily get into the honeypot. Production honeypots are valuable to the organization especially commercial, as it helps to reduce or mitigate risk that a specific organization faces. Production honeypots secure the organization by policing its IT environment to identify attacks. These production honeypots are useful in catching hackers with criminal intentions. The implementation and deployment of production honeypots are relatively easier than research honeypots. One of the reasons is that production honeypots have less purpose and require fewer functions. Thus production honeypots also provide less evidence about hacker's attack patterns and motives. Using production honeypots, we may know the origin of the hackers such what kind of machine or operating system it uses, which country they are from, the kind of tools they used and the types of exploits the blackhat launches. But, how the hackers interact with each other and how they create their attack weapons might not be revealed from the production honeypot. Production honeypots have less risk. The concept of production honeypots is to let the blackhat community spend time and resource into attacking the honeypots rather than the organization's production systems. The attacker is tricked, deceived and mislead into attacking the honeypot, thus protecting the organization's production systems from the attack. But rather than using deception which also contributes little to prevention, organizations should use their time and resources into better securing their IT infrastructure with best security practices and policies. Nowadays, attacks are becoming more automated and creative. Deception fails against two of the most common attack today; automated toolkits and worms – These automated tools will probe, attack, and exploit anything they can find vulnerable. These tools will definitely be used to attack honeypots. Research honeypots are complex. They are designed to collect as much information as possible about hackers' and their activities. Research honeypots are not specifically valuable to an organization. Their primary mission is to research the threats organization may face, such as who and how the attackers are, how they are organized, what kind of tools they use to attack other systems, and where they obtained those tools. While production honeypots are like the police, research honeypots act as their intelligence counterpart and their mission – To collect information about the black hat community. From the information gathered by research honeypots, it will help the organization to better understand on the hackers' attack patterns, motives and how they function. With all these knowledge about potential threats in grasp, the organization can better prepare to arm itself with the necessary defence mechanisms and processes. Research honeypots are also an excellent tool to capture automated attacks as mentioned in the above paragraphs, such as auto-rooters or worms. From research honeypots, organizations can better understand the three important concepts in Security; Prevention, Detection and Respond. But for the organization to better secure its security infrastructure, they should use production honeypots as it is easier to implement and manage. An example of research honeypot is honeynet. To better utilize research honeypots, organization need to provide the blackhat community with real operating systems, protocols and applications for them to communicate with. But, with added role, research honeypots become a disadvantage as it is difficult and complex to implement higher risk and require skilled personnel with time and effort to manage the honeypot. If an organization wants to protect its IT environment and production systems by detecting and

blocking the attacker, and to prosecute the attacker in court later, they should use a production honeypot. But, if the organization just wants to strengthen the security of its IT infrastructure by learning the hacker's attack techniques, the origin of attack, the kinds of tools and exploits used and his activities on the compromised honeypot, they should use a honeypot with a research touch.

3.1 Types of Honeypots

Honeypots come in many shapes and sizes. We break them down into two general categories, low-interaction and high-interaction honeypots. Low-interaction honeypots (Production honeypots) have limited interaction they normally work by emulating services and operating systems. Attacker activity is limited to the level of emulation by the honeypot. For example, an emulated FTP service listening on port 21 may just emulate a FTP login, or it may support a variety of additional FTP commands. The advantages of a low-interaction honeypot are their simplicity. These honeypots tend to be easier to deploy and maintain, with minimal risk. Usually they involve installing software, selecting the operating systems and services you want to emulate and monitor, and letting the honeypot go from there. This plug and play approach makes deploying them very easy for most organizations. Also, the emulated services mitigate risk by containing the attacker's activity, the attacker never has access to an operating system to attack or harm others. The main disadvantages with low interaction honeypots is that they log only limited information and are designed to capture known activity. The emulated services can only do so much. Also, its easier for an attacker to detect a low-interaction honeypot, no matter how good the emulation is, skilled attacker can eventually detect their presence. Examples of low-interaction honeypot include Specter, Honeyd, and KFSensor.

High-interaction honeypots (Research honeypots)are usually complex solutions as they involve real operating systems and applications. Nothing is emulated, we give attackers the real thing. If you want a Linux honeypot running an FTP server, you build a real Linux system running a real FTP server. The advantages with such a solution are two fold. First, you can capture extensive amounts of information. By giving attackers real systems to interact with, you can learn the full extent of their behavior, everything from new rootkits to international IRC sessions. The second advantage is high-interaction honeypots make no assumptions on how an attacker will behave. Instead, they provide an open environment that captures all activity. This allows high-interaction solutions to learn behavior we would not expect. An excellent example of this is how a Honeynet captured encoded back door commands on a non-standard IP protocol (specifically IP protocol 11, Network Voice Protocol). However, this also increases the risk of the honeypot as attackers can use these real operating system to attack non-honeypot systems. As result, additional technologies have to be implement that prevent the attacker from harming other non-honeypot systems. In general, high-interaction honeypots can do everything low-interaction honeypots can do and much more. Examples of high-interaction honeypots include Symantec Decoy Server and Honeynets.

3.2 Honeyd: Low-interaction honeypot

Developed by Niels Provos, Honeyd is Open Source and designed to run primarily on UNIX systems (though it has been ported to Windows). Honeyd works on the concept of monitoring unused IP space. Anytime it sees a connection attempt to an unused IP, it intercepts the connection and then interacts with the attacker, pretending to be the victim. By default, Honeyd detects and logs any connection to any UDP or TCP port. In addition, you can configure emulated services to monitor specific ports, such as an emulated FTP server monitoring TCP port 21. When an attacker connects to the emulated service, the honeypot detects and logs the activity and also captures all of the attacker's interaction with the emulated service. In the case of the emulated FTP server, we can potentially capture the attacker's login and password, the commands they issue, and perhaps even learn what they are looking for or their identity. Most emulated services work the same way. They expect a specific type of behavior, and then are programmed to react in a predetermined way. If attack A does this, then react this way. If attack B does this, then respond this way. The limitation is, if the attacker does something that the emulation does not expect, then it does not know how to respond. Most low-interaction honeypots simply generate an error message. You can see what commands the emulated FTP server for Honeyd supports by review the source code.

Some honeypots, such as Honeyd can emulate services and actual operating systems. In other words, Honeyd can appear to the attacker to be a Cisco router, WinXP web server, or Linux DNS server. There are several advantages to emulating different operating systems. First, the honeypot can better blend in with existing networks if the honeypot has the same appearance and behavior of production systems. Second, you can target specific attackers by providing systems and services they often target, or systems and services you want to learn about. When an attacker connects to an emulated service, you can have that service behave like and appear to be a specific OS.

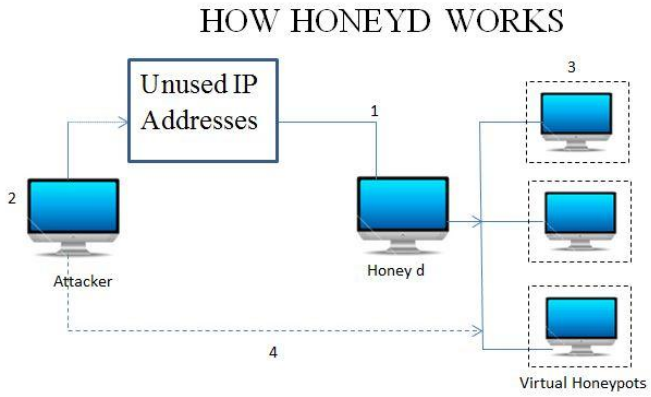

Figure2

How honeyd works can be shown more appropriately as shown in Figure 2. Honeyd monitors unused IP space (1). When an attacker(2) probes an unused IP, Honeyd detects the probe, takes over that IP via ARP spoofing, then creates a virtual honeypot(3) for the attacker to interact with (Honeyd can create multiple virtual honeypots to fool attackers on all unused addresses). The attacker is fooled into thinking he is interacting with a successful hacked system (4). In addition, Honeyd automatically updates its list of unused IPs as systems are added or removed from the network. Honeyd is primarily used for detecting attacks. It works by monitoring IP addresses that are unused, that have no system assigned to them. Whenever an attacker attempts to probe or attack a non-existent system, Honeyd, through Arp spoofing, assumes the IP address of the victim and then interacts with the attacker through emulated services. These emulates services are nothing more than scripts that react to predetermined actions. For example, a script can be developed to behave like a Telnet service for a Cisco router, with the Cisco IOS login interface. Honeyd's emulated services are also Open Source, so anyone can develop and use their own. The scripts can be written in almost any language, such as shell or Perl. Once connected, the attacker believes they are interacting with a real system. Not only can Honeyd dynamically interact with attackers, but it can detect activity on any port. Most low interaction honeypots are limited to detecting attacks only on the ports that have emulated services listening on. Honeyd is different, it detects and logs connections made to any port, regardless if there is a service listening. The combined capabilities of assuming the identity of non-existant systems, and the ability to detect activity on any port, give Honeyd incredible value as a tool to detect unauthorized activity.

3.3 Honeynets:high interaction honeypots

Honeynets are architecture, an entire network of computers designed to attack. The idea is to have an architecture that creates a highly controlled network, one where all activity is controlled and captured. Within this network we place our intended victims, real computers running real applications. The bad guys find, attack, and break into these systems on their own initiative. When they do, they do not realize they are within a Honeynet. All of their activity, from encrypted SSH sessions to emails and files uploads, are captured without them knowing it. This is done by inserting kernel modules on the victim systems that capture all of the attacker's actions. At the same time, the Honeynet controls the attacker's activity. Honeynets do this using a Honeywall gateway (Figure 3). This gateway allows inbound traffic to the victim systems, but controls the outbound traffic using intrusion prevention technologies. This gives the attacker the flexibility to interact with the victim systems, but prevents the attacker from harming other non-Honeynet computers.

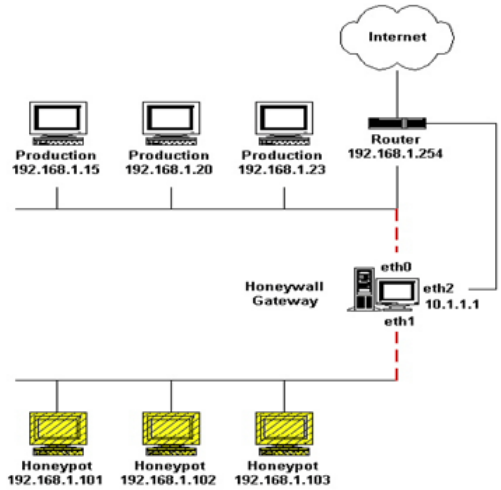

Fig: A honeynet

Advantages: Honeypots are a simple concept.

- Small data sets of high value: Honeypots collect small amounts of information. Instead of logging a one GB of data a day, they can log only one MB of data a day. Instead of generating 10,000 alerts a day, they can generate only 10 alerts a day. Remember, honeypots only capture bad activity, any interaction with a honeypot is most likely unauthorized or

- ion with a honeypot is most likely unauthorized or malicious activity. As such, honeypots reduce 'noise' by collecting only small data sets, but information of high value, as it is only the bad guys. This means its much easier (and cheaper) to analyze the data a honeypot collects and derive value from it.

- New tools and tactics: Honeypots are designed to capture anything thrown at them, including tools or tactics never seen before.

- Minimal resources: Honeypots require minimal resources, they only capture bad activity. This means an old Pentium computer with 128MB of RAM can easily handle an entire class B network sitting off an OC-12 network.

- Encryption or IPv6: Unlike most security technologies (such as IDS systems) honeypots work fine in encrypted or IPv6 environments. It does not

- matter what the bad guys throw at a honeypot, the honeypot will detect and capture it.

- Information: Honeypots can collect in-depth information that few, if any other technologies can match.

- Simplicity: Finally, honeypots are conceptually very simple. There are no fancy algorithms to develop,

state tables to maintain, or signatures to update. Thus there will be mistakes or misconfigurations.

Disadvantages: It is because of disadvantages they do not replace any current technology, but work with existing ones.

- Limited view: Honeypots can only track and capture activity that directly interacts with them. Honeypots will not capture attacks against other systems, unless the attacker or threat interacts with the honeypots also.

- Risk: All security technologies have risk. Honeypots also have risk. Honeypots have the risk of being taken over by the bad guy and being used to harm other systems. Depending on the type of honeypot, it can have no more risk than an IDS sensor, while some honeypots have a great deal of risk.

CONCLUSION

The purpose of this paper was to define what honeypots are and their value to the cloud security community. We identified two different types of honeypots, low-interaction and high-interaction honeypots. Interaction defines how much activity a honeypot allows an attacker. Scalable distributed intrusion detection systems may be implemented based on cloud computing infrastructure. The value of these solutions is both for production or research purposes. Honeypots can be used for production purposes by preventing, detecting, or responding to attacks. Honeypots can also be used for research, gathering information on threats so we can better understand and defend against them.

REFERENCES

[1] Lance Spitzner, "Honeypots Tracking Hackers", 2003, Pearson Education, Inc
[2] RetoBaumann and Christian Plattner,"White Paper: Honeypots", 26 February 2002 URL: http://www.inf.ethz.ch/~plattner/pdf/whitepaper.pdf
[3] Lance Spitzner, "Honeypots Definition and Value of Honeypots", 17 May, 2002, URL: http://www.enteract.com/~lspitz/honeypot.html
[4] Kurt Seifried, "Honeypotting with VMWARE – basics", 15 February 2002, URL: http://www.seifried.org/security/ids/20020107-honeypot-vmware-basics.html
[5] Honeynet Project, "Know Your Enemy: Defining Virtual Honeynets, Different types of Virtual Honeynets", 18 August 2002, URL: http://www.honeynet.org/papers/virtual/
[6]Wikipedia http://en.wikipedia.org/wiki/Honeypot_(computing)[7] F. Berman, G. Fox, and T. Hey. Grid Computing: Making the Global Infrastructure a Reality. Wiley and Sons, 2003.
[8] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wal-lach, M. Burrows, T. Chandra, A. Fikes, and R. Gruber. Bigtable: A Distributed Storage System for Structured Data. Proceedings of 7th Symposium on Operating System Design and Implementation(OSDI), page 205218, 2006.
[9] J. Chase, D. Irwin, L. Grit, J. Moore, and S. Sprenkle. Dy-namic virtual clusters in a grid site manager. High Performance Distributed Computing, 2003. Proceedings. 12th IEEE International Symposium on, pages 90–100, 2003.
[10] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. Proceedings of 6th Sym-posium on Operating System Design and Implementa-tion(OSDI), pages 137–150, 2004.
[11] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: amazon's highly available key-value store. Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles, pages 205–220, 2007.
[12] Enomalism elastic computing infrastructure. http://www.enomaly.com.
[14] I. Foster and C. Kesselman. Globus: A metacomputing infrastructure toolkit. International Journal of Supercom-puter Applications, 1997.
[15] I. Foster and C. Kesselman, editors. The Grid – Blueprint for a New Computing Infrastructure . Morgan Kaufmann,1998.
[16] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The phys-iology of the grid: An open grid services architecture for distributed systems integration, 2002.
[17] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. Int. J. High Perform. Comput. Appl., 15(3):200–222, 2001.
[18] D. Gannon. Programming the grid: Distributed software components, 2002.
[19] Google – http://www.google.com/.
[20] D. Greschler and T. Mangan. Networking lessons in deliv-ering 'software as a service': part i. Int. J. Netw. Manag. , 12(5):317–321, 2002.
[21] D. Greschler and T. Mangan. Networking lessons in deliv-ering 'software as a service': part ii. Int. J. Netw. Manag., 12(6):339–345, 2002.
[22] R. Hiremane. Intel Virtualization Technology for Directed I/O (Intel VT-d). Technology@Intel Magazine , 4(10), May 2007.
[23] Honeypots: Tracking Hackers-http://www.tracking-hackers.com/