Multi-channel Secure Interconnection Design for Hybrid Clouds

Mauro Storch, César A. F. De Rose, Avelino F. Zorzo and Régio Michelin
Computer Science School – Pontifical Catholic University of Rio Grande do Sul
Email: {mauro.storch, regio.michelin}@acad.pucrs.br, {cesar.derose, avelino.zorzo}@pucrs.br

***Abstract*—Since its conception, Cloud Computing elasticity behavior offers a smart option of extension to companies' infrastructure. This extension is also used to create Hybrid Cloud Computing (HCC) environments through connecting private and public cloud instances. The connection channel between the company's private cloud and the public cloud provider is often encrypted due to security reasons. This paper proposes a multi-channel interconnection design for computing services of a hybrid cloud, targeting companies' security profiles. The design prioritizes vital channels, i.e., Quality of Services guarantees, and improves security by applying a different cryptographic cipher to each given channel. Also, our tests using OpenVPN as the channel player show improved communication up to 5 times when splitting the workload into multiple connections.**

Keywords*—*Hybird Cloud Computing; Communication Security; Distributed Systems; Network Communications; Communication Performance; Quality of Services.

I. Introduction

HCC is becoming a scalable and economic solution to increase an in-house infrastructure. This architecture is a composition of public and private cloud models which support computational resources as services. The scalable skill is supported by the well-known feature called Cloud Computing Elasticity [1]. This feature allows companies to rent computational resources on-demand, from Virtual Machines to Storage area. Once a local environment is designed as a private cloud, it applies the Cloud principles and increases computational management quality, reduces the project budget through infrastructure sharing, and also saves power consumption by virtual machine consolidation.

Recently, VMWare vCenter started to offer a hybrid concept, called vCloud Hybrid Service [2], where companies move its virtual infrastructure to a public cloud in a secure and transparent way. However, there are some limitations in terms of heterogeneity in the proposed model. The VMWare's product only supports its own virtualization technology and the extension should be made to their public cloud instance.

On the other hand, Amazon Web Services (AWS), RackSpace, HP Cloud, among others, offer the majority of Cloud services on Internet [3]–[5]. Features such as global zoning design and the competitive price make them attractive for large, medium and even more for start-up companies. Through the global zoning design, users can easily deploy their applications around the world to reach more clients or users, considering short communication paths, local data processing, mobility and so on. Besides, each Cloud provider is priced on demand, the more expensive the resources (electricity, taxes, location, etc.) the higher the price per service unit is (month, hour, CPU cycles, bandwidth usage, etc). A company can arrange its outlays by balancing the elasticity distribution over a public cloud instance, considering aspects such as on-demand economy, availability, scalability, etc. [1].

Based on this scenario, one may consider an environment extension of a company infrastructure to some Cloud provider under the hybrid cloud model. Although a local environment is under secure and controlled rules, companies normally adopt a single communication channel through the Internet between the public and its local private cloud instance. This channel is used for data exchanging of applications, tasks synchronization, monitoring and so on.

Once this channel is created over the Internet, it is susceptible to different types of attacks such as eavesdropping, man-in-the-middle, data modification, and so on [6]. In this case, the application's data or even its integrity could be compromised. The most common communication setup is applying some *ad hoc* encryption algorithm in order to keep the exchanged information secure against these attacks. In order to ensure the security of this information, a strong encryption algorithm may be necessary.

This paper proposes to create multiple communication channels between the public and private cloud, each one for the different types of information that are exchanged between the Clouds. The information sent through these channels has different levels of priority. So, in order to achieve this qualification, it would be reasonable to assign different priority levels to each channel.

By separating the communication profiles (managing, monitoring, database replication, application's data exchange and many others) into different channels with different encryption levels, it is possible to keep information safe and also achieve better performance through load balancing among the created channels. Also, data exchange in the same application could be spread over those channels, after identifying the required priority level [7].

This work considers the HCC concept following the NIST (National Institute of Standards and Technology) [8] point of view and explores a secure multi-channel communication design in this concept.

The paper is organized as follows. Section II presents some related works. Section III describes the Multi-channel secure design for hybrid clouds. Section IV presents the multi-channel design application and the issues to be considered. Section V describes the proof of concept, the experimental evaluation and the achieved results. Finally, Section VI shows the conclusions and future works.

II. Related Work

Researches related to Cloud communication are focused on a Cloud interconnection using IP-VPN techniques.

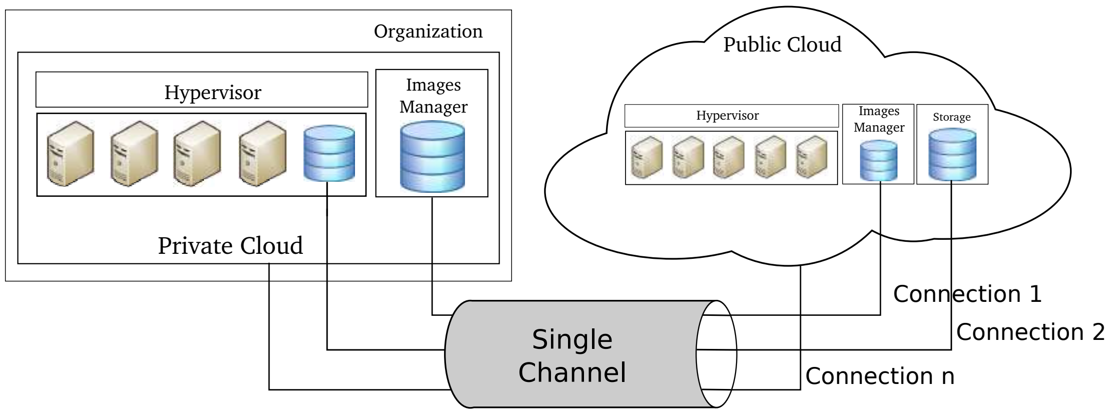

Fig. 1: Single Channel interconnection.

Ishimura [9] proposes a model in which a private company Intranet is connected to different cloud providers. This connection is established through IPSecVPN and is focused on the interconnection architecture (Full-Mesh or Hub). It does not consider the cryptography algorithm applied, neither the processing time involved in this operation.

Hata [10] defines a mechanism to connect different networks to a Cloud. Due to the virtual machine flexibility (it can be migrated between data centers depending on the load condition) he proposes a mechanism called Dynamic IP-VPN. He virtualizes the network, creating an architecture of Dynamic IP-VPN in order to enable users to control network components and equipment resources. In this architecture, he also describes a protocol that allows the user to reach the Cloud server. However, communication is still based on one tunnel connection per server's network interface.

Komu *et. al.* [11] specifies a model of secure communication for administrators, from the Infrastructure-as-a-Service (Iaas) point of view, using the Host Identity Protocol (HIP). The model isolates the multi-tenant network at the public cloud side and offers tunnelled access for external users. However, it does not consider security requirements, applying the same set of rules to the entire communication.

Wood [12] proposes the CloudNet architecture as a Cloud framework consisting of Cloud computing platforms linked with a network infrastructure based on Virtual Private Network (VPN) to provide seamless and secure connectivity between enterprise and Cloud data centers. This framework is optimized for supporting virtual machine live migration between geographically distributed data centers, but it still uses only one channel to migrate all information.

III. Multi-Channel Design

In order to build a hybrid cloud environment, an important component to be considered is the communication channel between the services on public and private clouds. This channel is commonly established through the Internet. The security requirements should be considered in agreement to company's data exposition rules.

Considering the interconnection among services of a private and a public cloud instances, a single channel using a VPN approach is normally adopted. All communication from each component, i.e., application's communication, database replication, service monitoring and management, are made through that single channel. Some improvements in WAN live migration of virtual machines also use VPN to provide network transparency after the migration [12]. Figure 1 shows the connections of some subsystems built through a single channel.

Usually, in common scenarios all data streams share a single VPN using the same physical path and configuration such as routes, network addresses, security strategies (considering authentication method and cryptography algorithms), and so on.

In a different perspective, a strategy that considers splitting isolated streams over different setups could bring new advantages in this scenario. For example, by qualifying the security priority of the communication for different parts of the hybrid cloud system, it is possible to rearrange the cryptography algorithms in order to achieve better results, i.e, by increasing either communication performance or security. For instance, during the application deployment, no sensitive data is transmitted. So, a non-encrypted channel could be used and the application could be verified on destination by comparing the generated MAC (Message Authentication Code). This verification ensures that the transferred data was not altered by any attacker during transmission. Moreover, some communication levels such as monitoring, management and billing, could also be encapsulated by lower encryption due to lack of business data on line.

Once the majority of inter-cloud communication channels are built over the Internet, the security ought to be considered an essential feature in this interconnection. Although, some IT managers penalize the security in order to acquire better performance during the communication. This trade-off relates to the fact that some encryption algorithms increase both the data size after encryption and the execution time in the processing phase (queuing application's communication). If the communication of services were split into several channels it would be possible to allocate different security levels for each

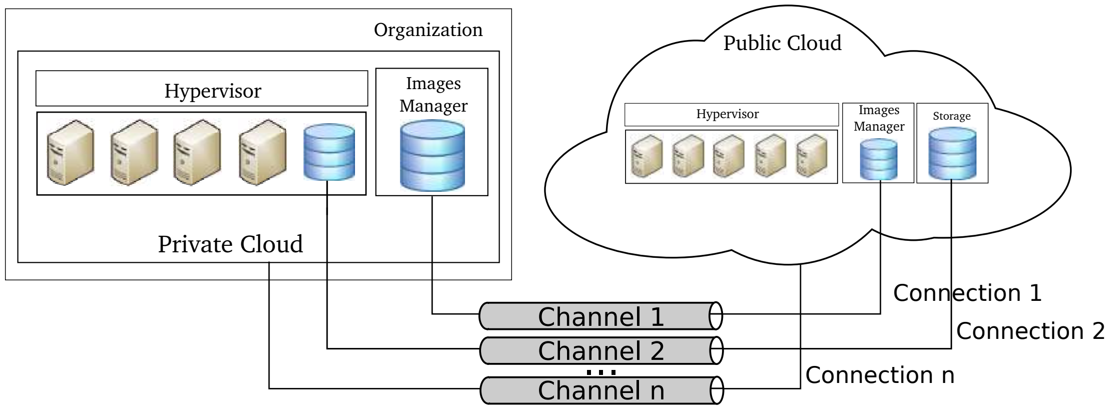

Fig. 2: Multi-Channel interconnection design considering hybrid cloud's communication split according to applications' security requirements, interconnection paths and isolation of communication's performance.

channel matching the security requirements of services (this security level differentiation will not be covered in this work). Figure 2 shows a channel per service connection.

An application defined as a three-layer model (data, logic and presentation) is a typical scenario in companies. The sensitive point would be the data and it should be protected in a more reliable communication channel when transferred outside the company's frontiers. This decision model considers that the logic and the presentation layers have already been deployed on public cloud resources and only data is present during the communication.

Although data is predominantly transferred, the security requirements are diverse for every company's application. Considering the three-layer stack for an application, some performance advantages would be achieved by splitting the communication channels and applying security techniques for the minimal requirements of each layer.

The multi-channel design would be implemented not only due to security issues. It is possible to identify at least two further benefits. A second benefit considers that a channel could be instantiated over different physical networks to balance or achieve better performance in communication. By using a network-layer tunnel protocol, such as IPSec [13], it is possible to attach the channel to a given logical or physical network, under a well-specified security model and drive the packet flow through a known route. In an upper level in the TCP/IP stack, it is possible to instantiate a transport-layer tunnel based on TCP or UDP protocols, by using tools such as the OpenVPN. The tunnel in that layer does not consider the layers below, which could be combined, expanding the security possibilities, or even replaced according to its network compatibilities.

A third benefit can be reached by adopting one channel per one or more connections, it is possible to schedule the total bandwidth by prioritizing a given channel. The balancing is provided by network techniques in each layer, such as filtering packets by origin and destination addresses, or even queueing content transferred such as backup or monitoring. On the one hand, from the hybrid cloud user point of view the communication would be transparent. On the other hand, from the administrative point of view, managers could prioritize critical communication related to services' life-cycle (monitoring, synchronization, replication, etc). Still, the production environment could be prioritized over software development resources, even into the same environment.

IV. Application Scenario

In order to instantiate the proposed interoperability design, this section presents a three-channel scenario considering a deployed application over a HCC environment.

The three channels are mapped in order to support the software communication between a private cloud instance and a public cloud provider.

The three channels are:

- **Application Synchronization:** Application will synchronize content and variables to keep its state;
- **Data Share:** Files and the database will place replicas among Cloud instances to provide fault tolerance, low access latencies, etc;
- **Management and Monitoring:** Access, authentication, monitoring, and billing processes are considered to manage public-side software.

The hybrid cloud scenario, instantiated for this environment, consists of an account in a public cloud provider and a private cloud instance. The public cloud account was created at AWS. It provides standard virtual machines to be connected to the private cloud ones. OpenStack [14] was used as private cloud manager. This tool is an open source initiative to support management of IaaS for common data centers. It is integrated with various hypervisors [15], such as KVM [16], XenServer [17], Hyper-V [18], and VMWare vSphere [19].

OpenStack was designed as a modular set of components that can interact with each other to create a single point of view of the entire data center's hardware. In this work only the Compute, Image and Networking modules are considered for managing the private cloud environment. These three modules support virtual machine creation and network configuration, but, in order to create the multi-channel design we use

OpenVPN [20]. The creation of channels can be a future feature to be added to cloud management tools to support the proposed model. Although OpenStack already supports VPN creation through IPSec, OpenVPN was used due to its firewall bypassing feature.

OpenVPN creates a single TCP or UDP connection in a given transport layer port, which could be set to be handled by the company's firewall. Once this connection is made, OpenVPN creates a virtual interface with a local-private IP address in both sides, i.e., client and server. From the user point of view, the interface is related to a local network connection. Every packet sent or received will be caught by OpenVPN, being encrypted and/or compressed, and delivered to the TCP/UDP connection.

Both cryptography and compression tasks are made in the operating system's user-space. So, the communication will use some CPU cycles both in private and public cloud. However, it does not have a significant impact on communication since OpenVPN creates a pipe during this process. In other words, as the packets are sent, they are one by one encrypted and delivered. Because the CPU is faster than the network communication there is no significant overhead. Yet, by using the Advanced Encryption Systems (AES) instructions provided by new processors, the CPU usage will be reduced as shown in Section V.

After the instantiation of the channels, based on this scenario, some features explore the benefits of the design as mentioned earlier. The first benefit is an increase in security due to the adoption of more than one cryptography key for the hybrid cloud interconnection. It is a straight forward achievement and it is supported through the size of the keys and the cryptography algorithm that will be adopted per channel. The second benefit related to paths and routes of connections were not implemented in this scenario, once this scenario was instantiated over a non-controlled network, the Internet.

The third benefit is the bandwidth scheduling, mentioned in Section III. It was implemented by prioritizing the channels according to its transferring profiles. Although the prioritization could be made without a specialized connection (using common TCP/UDP communications), it could be hard for IT managers to handle it on the fly. By setting up a rule in a given channel they could offer communication profiles for their users or applications transparently. These features are not commonly found on current instances of both public and private clouds, but they could be added to management tools as the following suggestions.

Although it is possible to manage both public and private clouds remotely through Application Programming Interfaces (API), the IT manager should be aware of every configuration in order to create the interconnection. The current OpenStack network module release can create a rich networking topology, including secure channels using VPN. However, there is no automation for setting up certificates and the cryptography level of each channel or even a prioritization behaviour. As a work in progress, we consider a contribution of those features by creating a proof-of-concept based on OpenVPN and OpenStack.

V. Experimental evaluation

In order to validate the multi-channel design, we show three evaluations focused on (i) demonstrating a performance increase by splitting communication in several channels; (ii) comparing the performance with different channel priorities and; (iii) empirically demonstrating an increased security provided by the multi-channel design that will be discussed during this section. The first test aims to show the adoption of a single VPN process (authentication, encryption, transferring, decryption) for each service in a hybrid cloud. By doing so, it will check if it is possible to reduce the overall communication time. The second test presents the results of adding prioritization of each channel. For the tests, Linux HTB [21] was applied to handle each level of the communication. This test is intended to show a bandwidth balance among the hybrid cloud's services.

A. Testbed description

For the tests, a hybrid cloud environment was configured combining a private cloud and a public cloud. The private cloud is composed by a machine powered by two Intel Xeon E7-2850 2.00 GHz (20 cores each, supporting hyper-threading) with the hardware virtualization flag enabled and using AES instructions, making cryptography phase up to five times faster in this scenario. Virtualization is provided by the Xen hypervisor [17] and managed by OpenStack (Havana 2013.2 release). OpenStack is responsible for the creation of virtual machines, local network configurations, storage allocation, and other aspects related to the management of the virtual resources. For the public cloud we use AWS, where a virtual machine was created under the IaaS model. This virtual machine is configured as a one core Intel Xeon CPU E5-2650 2.00GHz with 590MB RAM running Ubuntu Server 12.04 Amazon-image-based.

The three channels connecting the public cloud and the private cloud were implemented using the OpenVPN tool [20]. This tool is a standard open-source VPN player that offers TCP and UDP connections for data transport between peers under a client-server model. The server listens on a TCP/UDP port that is handled by both company's and public cloud's firewall. The transfer process supports cryptography and/or compression algorithms that are set during channel setup. For the tests in this paper, all the channels are configured using Advanced Encryption Standard (AES) [22]. Three encryption levels were evaluated during the experiments: AES128, AES192 and AES256. We also evaluate a channel without an encryption algorithm and the compression flag was disabled in all scenarios. In the public cloud's VM, a VPN service is instantiated in order to accept connections from the private cloud. There is one VPN service for each channel.

For performance evaluation, the communication is made over a TCP connection running in the User-Space domain. This approach enables tunnel creation using a TCP port that can bypass companies' firewall rules without exposing its network. Each connection is handled individually by the pair client-server. For new connections, a new server instance is created on another TCP port. Because the cryptography process of a tunnel runs in single-thread mode, by splitting the communication into several tunnels, each one running on a different operating system process in parallel, we obtain a better performance of the cryptography process and consequently a reduction in the overall communication time.

The setup for QoS tests consider that several prioritization models could be applied for a given set of connections. In some use cases it is common that some connections need more bandwidth or even more priority than others. One can consider, for instance, a real-time application that needs to synchronize data frequently. Instead of turning off other connections to reduce the response time of the synchronization, it is possible to prioritize that channel to handle it without killing other service's communication. In the same way, in a scenario with a production and a test environment, the prioritization of the channel of production against the test environment could guarantee QoS aspects even in a hybrid cloud, without the need for multiple instances of a private cloud, one for each purpose.

The setup also considers that each channel has its own authentication certificates (generating a new cryptography key) and cryptography algorithm. The adoption of different encryption keys for the safe channels makes the HCC interconnection stronger, since an attacker needs to break all three cryptographic keys to intercept the entire information, instead of only one in the case of a single channel.

Since security improvements are straightforward, we have conducted experiments to validate the performance and QoS benefits of the multi-channel approach. Section V-B describes these evaluations and presents our preliminary results.

B. Evaluation and results

The first evaluation was conducted to showcase the performance of the multi-channel design. We first created pairs of client-server instances from two to ten. Payload was one GByte of data split equally among 2 to 10 channels along the horizontal axis. The *Transfer Time* line in the chart, Figure 3, indicates the execution time of the total data transfer. The *CPU Load* line indicates the aggregated CPU load in the client side considering all processes of the OpenVPN tool. In this case, only the tunnel was considered, discarding the application load time. In the chart, for example, 4% of CPU-load represents that the process took 4 seconds using the CPU for a 100 seconds communication. The CPU-load time is not necessarily a blocking operation, since packets have no more then 1500 bytes and are processed like in a queue. In other words, the CPU time has no significant impact during the communication. However, if the AES instructions were turned off, the CPU-load would be increased up to 20%, taking longer for packets to queue, adding more overhead for fast connections. In our scenario we considered all available resources, since AES instructions are commonly found in nowadays processors [23]. The cryptography algorithm applied to all channels was AES-256 and no compression was used. AES-128 and AES-192 were also tested, but no significant difference was found to be considered in the evaluation.

We can clearly see a reduction in transfer time as more channels are used. Although the CPU usage also increased, it is

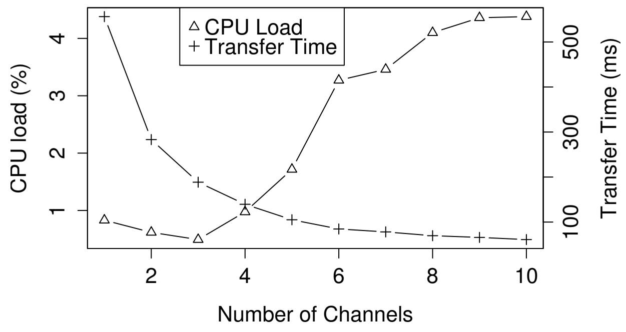

Fig. 3: Multi-Channel Splitting - Time x CPU load

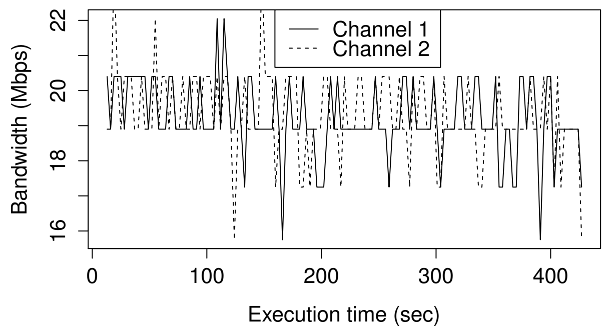

Fig. 4: Non-Priorized channel communication

not significant, not exceeding 5% of the total CPU available in the machine with this technique. Nevertheless, communication time was significantly reduced, up to almost 5 times with 10 channels (from 532 to 108 seconds).

The second evaluation which considered the QoS test was done to evaluate the effectiveness of prioritizing channels. In this case, we use a payload of one GByte per channel, sharing the same physical connection. Figure 4 shows the bandwidth of each channel and the resulting transfer time when both channels have the same priority concurring for the total system bandwidth (100 Mbits/s).

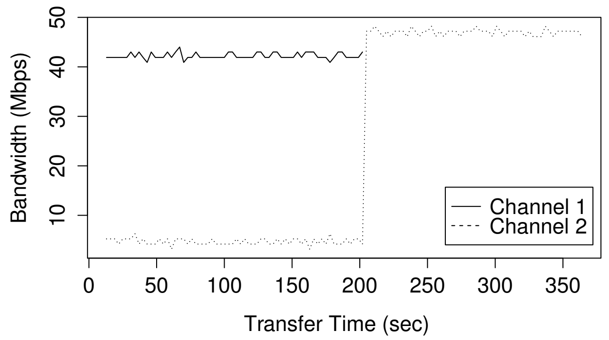

Fig. 5: Priorized channel communication

The possibility of prioritizing one of the channels results in better communication performance for sensitive data. This can be clearly seen in Figure 5 where channel one has priority 0 and channel two priority 5 (lower values represent higher priorities). In this case, Channel 1 gets a much bigger share of the total system bandwidth resulting in a significant reduction of its transfer time (from approximately 420 seconds to 200 seconds). Channel 2, on the other hand, had a much smaller share of the bandwidth when concurring with channel 1, improving only after Channel 1 transfer was finished. This is an important feature of a multi-channel interconnection having several applications in real hybrid clouds use cases.

VI. Conclusions and Future Work

HCC plays an important role combining the advantages of private and public clouds. It allows managers to use local infrastructure in a flexible and efficient way and on demand rent services under a pay-as-you-go fashion from a public cloud provider, reducing acquisition and management costs with a more flexible environment.

The most common interconnection design in hybrid clouds uses a VPN to create a single channel between the private and the public cloud that consolidates all service flows. This channel is often encrypted due to security requirements of a specific traffic.

In this paper, we presented a multi-channel design that uses one channel per service. This allows a much more customizable design, with each channel being configured depending on the characteristics of the different flows. This increases the security and the performance of the communication with the public cloud. By applying different keys in each channel, for example, there is an increase in security with several keys to break in order to acquire information from all flows. Besides that, a stronger key/algorithm could be applied to sensitive data in one specific channel. Concerning the performance, the multi-channel design allows a prioritization model, where a channel with critic data could receive a greater bandwidth share to achieve better performance.

Our preliminary tests with the multi-channel design show also a reduced communication time when splitting the payload transferring in several channels by a factor of 5. The results were validated using OpenVPN tool, a TCP/UDP tunnel player which encrypts data before sending data in a user-space thread. The splitting improvement is related to cryptography parallelism provided by multiple threads. Nevertheless, the CPU load does not overpass 4.4%.

As future work, we will evaluate the impact of the multi-channel design when running a distributed application over a hybrid cloud. We are also interested in security overhead modelling in hybrid clouds.

Acknowledgment

The authors would like to thank CNPq (National Counsel of Technological and Scientific Development - Brazil), CAPES and FAPERGS for financial support.

References

[1] B. Rimal, E. Choi, and I. Lumb, “A taxonomy and survey of cloud computing systems,” in *Fifth International Joint Conference on INC, IMS and IDC, 2009. NCM ’09.*, Aug 2009, pp. 44–51.

[2] VMware, Inc., “VMware vCloud ® Hybrid Service TM Service Description,” http://www.vmware.com/files/pdf/vchs/vCloud-Hybrid-Service-Service-Description.pdf, retrieved: Mar., 2015.

[3] RackSpace US Inc., “RackSpace - The open cloud company,” http://rackspace.com, accessed: Mar, 2015.

[4] Hewlett-Packard Development Company, “HP Public Cloud,” http://hpcloud.com, accessed: Mar, 2015.

[5] Amazon, Inc., “Amazon AWS,” http://aws.amazon.com/, accessed: Mar, 2015.

[6] S. Hansman and R. Hunt, “A taxonomy of network and computer attacks,” *Computers & Security*, vol. 24, no. 1, pp. 31 – 43, 2005.

[7] P. Watson, “A multi-level security model for partitioning workflows over federated clouds,” *Journal of Cloud Computing*, vol. 1, no. 1, pp. 1–15, 2012. [Online]. Available: http://dx.doi.org/10.1186/2192-113X-1-15

[8] P. Mell and T. Grance, “The NIST definition of cloud computing (draft),” *NIST special publication*, vol. 800, no. 145, p. 7, 2011.

[9] K. Ishimura, T. Tamura, S. Mizuno, H. Sato, and T. Motono, “Dynamic ip-vpn architecture with secure ipsec tunnels,” in *2010 8th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT)*, June 2010, pp. 1–5.

[10] H. Hiroaki, Y. Kamizuru, A. Honda, T. Hashimoto, K. Shimizu, and H. Yao, “Dynamic ip-vpn architecture for cloud computing,” in *2010 8th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT)*, June 2010, pp. 15–20.

[11] M. Komu, M. Sethi, R. Mallavarapu, H. Oirola, R. Khan, and S. Tarkoma, “Secure networking for virtual machines in the cloud,” in *Cluster Computing Workshops (CLUSTER WORKSHOPS), 2012 IEEE International Conference on*, Sept 2012, pp. 88–96.

[12] T. Wood, K. K. Ramakrishnan, P. Shenoy, and J. van der Merwe, “CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines,” *SIGPLAN Not.*, vol. 46, no. 7, pp. 121–132, Mar. 2011. [Online]. Available: http://doi.acm.org/10.1145/2007477.1952699

[13] S. Kent and K. Seo, “Security Architecture for the Internet Protocol,” RFC 4301 (Proposed Standard), Internet Engineering Task Force, Dec. 2005, updated by RFC 6040. [Online]. Available: http://www.ietf.org/rfc/rfc4301.txt

[14] OpenStack OpenSource Community, “OpenStack Project,” http://www.openstack.org/, accessed: Mar, 2015.

[15] OpenStack OpenSource, Community, “OpenStack Hypervisors Support Matrix,” https://wiki.openstack.org/wiki/HypervisorSupportMatrix, accessed: Mar, 2015.

[16] Qumranet, “KVM,” http://www.linux-kvm.org/page/Main_Page accessed: Mar, 2015.

[17] P. Barham *et al.*, “Xen and the art of virtualization,” *SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 164–177, Oct. 2003. [Online]. Available: http://doi.acm.org/10.1145/1165389.945462

[18] Microsoft, “Microsoft Hyper-V Server 2012 R2,” http://www.microsoft.com/en-us/server-cloud/hyper-v-server, accessed: Mar, 2015.

[19] VMWare Inc., “VMWare vSphere Hypervisor,” http://www.vmware.com/products/vsphere-hypervisor/, accessed: Mar, 2015.

[20] J. Yonan, “OpenVPN–an open source SSL VPN solution,” https://openvpn.net/, accessed: Mar, 2015.

[21] M. Devera, “Hierarchical token bucket theory,” *URL: http://luxik.cdi.cz/˜devik/qos/htb/manual/theory.htm*, accessed: Mar, 2015.

[22] N.-F. Standard, “Announcing the advanced encryption standard (aes),” *Federal Information Processing Standards Publication*, vol. 197, pp. 1–51, 2001.

[23] S. Gueron, “Intel advanced encryption standard (AES) instructions set,” *Intel White Paper, Rev*, vol. 3, pp. 1–94, 2010.