Transparent migration of virtual infrastructures in large datacenters for Cloud Computing

Roberto Bifulco, Roberto Canonico, Giorgio Ventre
Dipartimento di Informatica e Sistemistica,
Universita' degli Studi di Napoli "Federico II"
Naples, Italy
roberto.bifulco2@unina.it, roberto.canonico@unina.it,
giorgio.ventre@unina.it

Vittorio Manetti
CRIAI
Consorzio Campano di Ricerca per l'Informatica
e l'Automazione Industriale
Portici, Napoli, Italy
v.manetti@criai.it

***Abstract*—Cloud-enabled datacenters need advanced support for an integrated management of platform virtualization technologies. The networking infrastructure of large-scale datacenters is implemented according to redundant multi-tiered architectures whose layers operate at Layer 3 of the networking stack. Splitting the network infrastructure of a datacenter in a number of IP subnets, however, creates limits to the migration of Virtual Machines, reducing the possibility for administrators to efficiently balance the load and reduce the energy consumption of the whole infrastructure. In this paper we propose an innovative solution that allows transparent migration of Virtual Machines across the whole datacenter, based on the coordinated use of NAT rules that need to be consistently managed across the layers of the datacenter networking infrastructure. We describe in details how our approach can be easily implemented with current network devices without any modification to their hardware and present an experimental evaluation of an early prototype of our solution.**

***Index Terms*—network virtualization, Cloud Computing, datacenter management**

I. Introduction

Cloud Computing is "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [1]. Customers find this paradigm convenient as it relieves them from the responsibility of buying and managing a dedicated computing infrastructure, since resources may be dynamically acquired and released, according to their actual needs. Cloud Providers, on the other hand, can take advantage of scale economies to organize and manage large datacenters, whose resources can be efficiently utilized by partitioning and renting them to a number of customers. Depending on the abstraction level of the provided resources, Cloud Computing takes different Service Models: *Infrastructure as a Service* (IaaS), *Platform as a Service* (PaaS) and *Software as a Service* (SaaS). In this paper we focus on the IaaS model, which provides users with the ability to obtain a set of common fundamental computing resources (e.g., Virtual Machines, storage and networking services) that can be composed to create a customized *virtual infrastructure*. Despite its success, the IaaS paradigm poses new challenges in terms of management of the computing infrastructure: Cloud Providers have the responsibility to manage a large infrastructure that hosts a number of highly dynamic virtual infrastructures operated by different users. A typical commercial IaaS offering provides virtual infrastructures composed by a predefined set of VMs. A customer's virtual infrastructure may comprise groups of VMs. For instance, Amazon EC2 [2] allows users to define so-called "security groups". Each VM in a security group can directly communicate with all other VMs in the same security group, while traffic coming from external sources or destined to external hosts must pass through a firewall managed by the Cloud management system.

Modern virtualization technologies play a key role in modern datacenters for Cloud Computing, as they allow an efficient utilization of physical resources. Server virtualization technologies based on Virtual Machines are able to flexibly consolidate the computing workload of a datacenter over a set of available physical servers, through migration of running VMs from a physical server to another. This feature is particularly useful when the number of managed VMs is high, as it provides the ability to dynamically redistribute the computing workload for both load balancing and hardware maintenance. Current virtualization technologies, however, only support *live migration* of running VMs within a single IP subnet. The networking infrastructure of large-scale datacenters is implemented according to redundant multi-tiered architectures, comprising a number of different IP subnets. Splitting the network infrastructure in several IP subnets limits the scope of migration of VMs to portions of the datacenter, and reduces the possibility for administrators to efficiently balance the load and reduce the energy consumption of the whole infrastructure.

In this paper we propose an innovative solution that allows transparent migration of VMs across the whole datacenter by adapting the novel *Service Switching* paradigm, originally proposed for supporting geographic migration of network services [3]. Our solution is based on the coordinated use of NAT rules and ARP proxying that needs to be consistently managed across the layers of the datacenter networking infrastructure. We describe in details how our approach can be easily implemented with current network devices without any modification to their hardware and present an experimental evaluation of an early prototype of our solution.

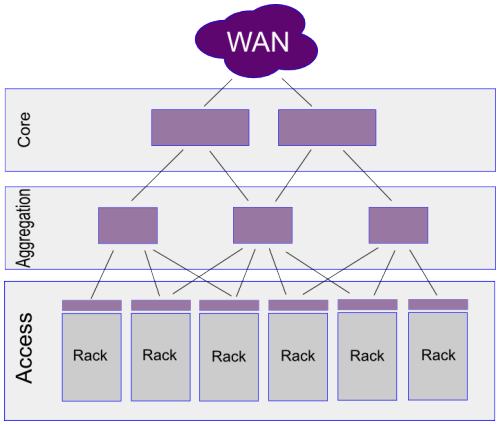

Fig. 1. Datacenter layered architecture

The rest of the paper is organized as follows. Section II illustrates the hierarchical networking infrastructure of modern large scale datacenters. Section III presents the role of virtualization and how live migration of Virtual Machines can be used for an efficient management of physical resources in Cloud-enabled datacenters. Section IV illustrates our solution for transparent migration of VMs across different IP subnets. We discuss all the possible communication patterns and how they are preserved in case of migration of one or more Virtual Machines of the same group. Section V illustrates a prototype of our solution using Xen-based Virtual Machines and Linux-based software routers. Our evaluation demonstrates the feasibility of our approach and its correct behaviour in all the scenarios presented in the previous section. In Section VI we compare our solution against similar works that have been recently published in the literature. Finally, we draw our conclusions and illustrate our future developments.

II. Datacenter networking

Due to the large numbers of connected devices and the huge aggregated communication requirements, the networking infrastructure of a datacenter providing IaaS services is necessarily organized according to a hierarchical design. Figure 1 shows a typical network architecture for a large scale datacenter (adapted from [4]). Commercial datacenter networking solutions typically identify at least three levels of network devices, commonly referred to as *Access*, *Aggregation*, and *Core*. Core level devices connect the datacenter with the Internet, through one or more geographic links.

While the upper levels usually operate at layer 3 of the networking stack, i.e., they act as routers, Access layer devices may be configured to operate at either L2 or L3. The use of L2 devices at all layers of the infrastructure has proven to be unfeasible due to scalability problems deriving from too large broadcast domains. Spanning Tree Protocol (STP), in fact, can take up to 50 seconds to converge in a large network, while the Rapid Spanning Tree Protocol (RTSP) still requires tens of seconds to converge in some topologies [5]. Hence, L3 solutions at the Access layer are recently preferred for large scale datacenters, as they provide faster routing convergence, contain broadcast domains to a limited size, and simplify troubleshooting and management procedures.

In the context of a datacenter for IaaS services it is common practice to use private IP addresses for internal networks. To guarantee public accessibility of services from the Internet, front-end nodes are associated to a limited set of public IP addresses, which are NAT-ted at the datacenter edge. For the purposes of this work, we do not consider such public IP addresses, and assume that, within the datacenter infrastructure, a VM is uniquely identified by one or more private IP addresses.

III. Virtualization

Virtualization is a widely adopted solution for *resource multiplexing* problems. In general terms, virtualization is a technique in which a software layer multiplexes lower-level resources for the benefit of higher level software programs and systems. Virtualization can be applied to either single physical resources of a computing system (e.g. a single device) or to a complete computing system. When applied in this latter sense, (a.k.a. *Platform Virtualization*), it allows the coexistence of multiple "Virtual Machines" in the same computing host. Platform virtualization is implemented by means of an additional software layer, called *Virtual Machine Monitor* (VMM) (or *hypervisor*), that acts as an intermediary between the system hardware resources and the Operating System. Modern VMMs (such as Xen, KVM, VMware vSphere) add support for *live migration* of Virtual Machines. Live migration is a feature that allows the migration of a running VM from one physical host to another, with a downtime limited to a few dozens of milliseconds. Live migration allows Cloud Providers to dynamically reconfigure the allocation of VMs over the available physical resources, so enabling advanced strategies for workload distribution and energy savings in the datacenter [6].

A VM migration process basically consists in suspending the running VM, copying the status from the source VMM to the destination VMM, and finally resuming the VM at its destination. This process may be optimized with several techniques. Xen, for instance, uses an *iterative pre-copy* strategy, as described in [7].

The Xen live migration process described in [7] assumes that (i) source and destination VMMs both reside in the same LAN, and that (ii) a network-attacched storage (NAS) provides a shared storage to VMs, which do not rely on local storage resources.

The problem of migrating virtual machines across the boundaries of a single IP network (or sub-network) is not considered in [7]. Similar constraints also exist for other virtualization technologies [8].

IV. Our solution

In this section we illustrate a novel solution that allows transparent live migration of Virtual Machines within a large scale datacenter infrastructure comprising different IP subnets.

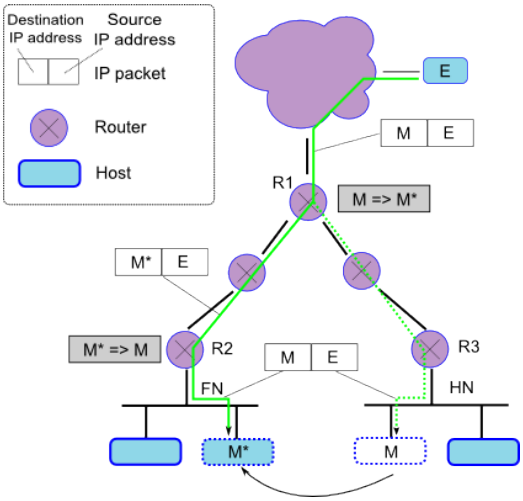

Fig. 2. NAT-based Service Switching

Our solution is based on a modification of the *Service Switching* paradigm, a technique originally proposed in [3] with the aim of supporting geographic migration of virtualized network services.

Our datacenter-oriented implementation of Service Switching relies on a combination of *live migration* mechanisms [7], of the *Mobile IP* model [9], and of the *Network Address Translation* (NAT) technique.

IP version 4 assumes that a node's IP address uniquely identifies its point of attachment to the Internet: a node must be located on the network indicated by its IP address in order to receive datagrams which are destined to it. IP Mobility Support (or Mobile IP) provides a mechanism that allows *Mobile Hosts* to change their point of attachment to the Internet without changing their IP address. This mechanism relies on two intermediary entities: the *Home Agent* and the *Foreign Agent*. The role of the Home Agent is to maintain current location information of the mobile node, and to re-transmit all the packets addressed to the Mobile Host through a tunnel to the Foreign Agent to which the Mobile Host is currently registered. The role of the Foreign Agent, in turn, is to deliver datagrams to the Mobile Host.

In Mobile IP, a Mobile Host interacts with the Foreign Agent to obtain the *Care of Address* (CoA), that is, the Foreign Agent's IP address. This address is notified by the Mobile Host to its own Home Agent, which, in turn, uses this piece of information to establish a tunnel to the Foreign Agent. Hence, the standard Mobile IP model, as described in [9], is not transparent to the Mobile Host, which is required to actively interact with the agents. Mobile Hosts, in fact, need to be able to discover agents and register with them. Agent discovery is performed through special ICMP messages, while host registration is based on the exchange of UDP datagrams sent to a well-known port (434).

As we mentioned before, the Service Switching paradigm is based on a few basic ideas taken from Mobile IP. In our context, a migrated Virtual Machine plays the role of a *Mobile Host*. However, the standard Mobile IP cannot be taken as-is, since one of the requirements for Service Switching is to make the VM migration process completely transparent to the migrated VM, which should be not aware of its migration. The Service Switching paradigm assumes that the VM migration procedure is managed by an external entity that is responsible of orchestrating the *live VM migration* with an *IP address migration* process. While the live VM migration process is strictly dependent on the adopted virtualization technology, the IP address migration process consists in properly configuring the devices forming the datacenter networking infrastructure in order to preserve the reachability of the migrated VM.

In the following we describe the Service Switching *IP address migration* process in the context of a hierarchically organized large scale datacenter. We assume a datacenter with a Top-of-Rack layout, in which the access layer of the networking infrastructure is formed by one single switch per rack. A Top-of-Rack switch usually connects 20-40 servers per rack, typically with 1 Gbps links. The same model, however, can be easily adapted to datacenters with a End-of-Row layout. We also assume that access switches are configured to behave as Layer 3 devices, acting as IP routers and interacting with the upper layers of the networking infrastructure through an Interior Gateway Protocol, such as OSPF, RIP or IS-IS.

In order to solve the transparent live migration problem among IP subnets within a datacenter, we must operate at both layer 3 and layer 2 of the networking stack. Layer 3 operations enable the reachability of the migrated VM through plain IP routing and forwarding. Layer 2 operations are necessary to ensure that the migrated VM is reachable from any other host in the datacenter, regardless of its location, and to avoid any reconfiguration of the VM's network settings (e.g., the default gateway address).

When a Virtual Machine is deployed for the first time, it is allocated in one of the available physical servers in a specific rack. This allocation choice assigns one or more IP addresses to the VM. These IP addresses will be kept for the entire VM lifecycle, even in case of migration. Such IP addresses are referred to as the VM's *Home Addresses*, and the *Home Addresses*' subnet is referred to as the VM's *Home Network*.

Each router involved in the IP address migration process is called a *Service Switch*. In particular, for a given VM, we refer to the router located at the edge of the rack in which resides the physical server where the VM is initially deployed as to the VM's *Home Service Switch*. Likewise, the router located at the edge of the rack hosting the server where the VM is going to be migrated, will be referred to as the VM's *Foreign Service Switch*.

By analogy with Mobile IP, a generic host communicating with a VM will be referred to as *Correspondent Node*. Making the simplistic assumption that a Virtual Machine presents a unique IP address, in order to access a given service implemented by the VM, a Correspondent Node sends packets to the VM's Home Address.

When the *IP address migration* process starts, a Care-of Address is generated in order to identify the new location of

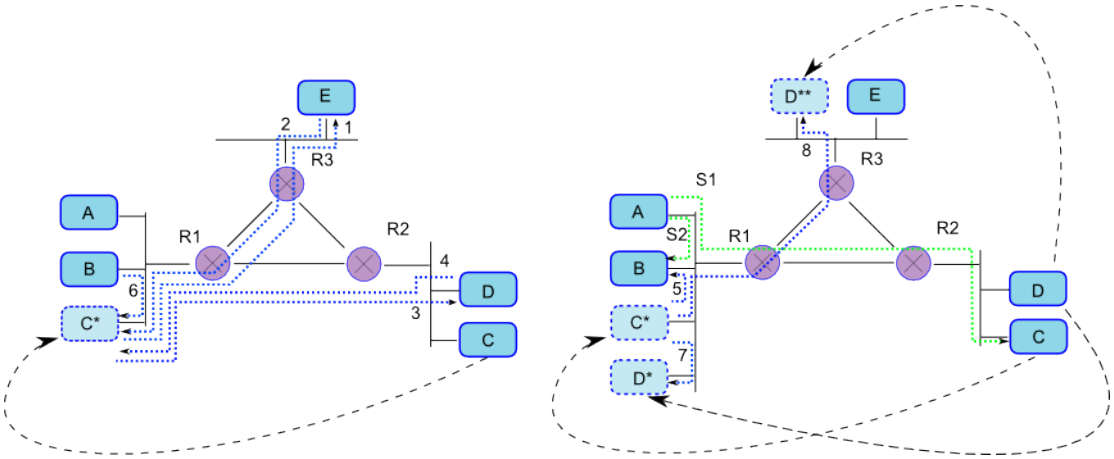

Fig. 3. Transparent VM migration cases

the migrated VM. Such Care-of Address is a new IP address that is assigned to the VM on the new IP subnet (the *Foreign Network*). Notice that this new address is not directly known by the VM, but only the datacenter's Service Switches are aware of it. Making the VM unaware of the Care-of Address enables the services on that VM to run without interruptions and without the need for reconfiguration even in case of migrations among different IP subnets.

Because we consider the problem of VM migration only within the limited scope of a single datacenter, where VMs are identified by private IP addresses, we do not concern about using an additional IP address for a migrated VM. In fact, the use of a whole class-A private IP network (10.0.0.0), combined with subnetting, is largely sufficient even for large scale datacenters. Moreover, by using a VM-specific Care-of Address, we are also able to avoid the use of a tunneling layer to forward network traffic to migrated VMs, by implementing NAT functions at Service Switches level. In the following two subsections we describe in details how a Service Switch should operate at both L2 and L3 in order to guarantee a seamless connectivity for migrated Virtual Machines.

A. Layer 3 operations

In figure 2 we show an example of IP address migration using NAT functions in Service Switches: an external node with IP address E sends packets to the datacenter's VM with IP address M, which is in the VM's Home Network (HN). Once the migration happens, the Care-of Address M* is assigned to the migrated VM. This new address belongs to the VM's Foreign Network (FN), and is only used at the datacenter's Service Switches level, while the VM is totally unaware of it. Simultaneously, NAT rules are added to the R1 (Edge Service Switch), R2 (Foreign Service Switch), and R3 (Home Service Switch) Service Switches: R1 and R3 are instructed to transform M address into M* address, so that packets destined to M are routed into the datacenter using the Care-of Address. When packets reach R2, a dual NAT rule is applied, transforming M* again in M, hence the packet can be put on the FN and the migrated VM can take it, being unaware of the changed IP network. In addition to NAT rules, the R2 routing table is also updated to take into account the new location of the migrated VM, that is now directly reachable from one of its own interfaces.

B. Layer 2 operations

To provide migration transparency we need to take into account both the VM's IP address migration and the VM's network configuration. Layer 3 operations for transparent address migration have been presented in the previous paragragh. Here we present operations needed at layer 2. Assuming that each VM knows its own Home Network (as this can be derived by combining the VM's IP address with the netmask) and knows its default gateway's IP address, we have to solve the connectivity cases shown in figure 3.

In the following we consider eight different communication cases. Each case is identified by a subsection number, that is consistently used in Figure 3 to illustrate the corresponding exchange of packets.

1) C to E:* To enable the communication of a migrated VM (C* in figure 3) with a host E sitting in another subnet (that is different from C*'s Home and Foreign networks), we have to consider the regular IP behaviour in the case of a communication between two hosts living in different IP subnets. The external node resides on a subnet that is different from the VM's one, so the VM's OS, looking at the subnet mask, realizes that the communication must happen through the default gateway. Because of the migration, the VM's default gateway (R2) is actually on a different subnet, and hence unreachable. To solve this problem in a transparent way, we enable the router on the foreign network (R1) to act as Proxy ARP, so that it responds to ARP request on behalf of R2. Hence, packets generated by the migrated VM destined to R2 are taken by R1, that in turn is able to correctly route them.

2) E to C:* The reverse path of the previous case is enabled using layer 3 operations at R3 and R1 as previously described.

3) C to D:* To allow the communication of the migrated VM with an host on the home network, we need again proxy ARP functionalities. The migrated VM tries to send packets destined to home network directly, because it considers home network's hosts as neighbours at layer 2. R1, in this case, must act as proxy ARP for the entire Home Network: each packet destined to the Home Network is taken by R1 and forwarded to the Home Network, using IP routing, where R2 can correctly perform the delivery.

4) D to C:* The communication from an host on the home network with the migrated VM (case 4) is performed configuring R2 to work as proxy ARP for the migrated VM's *home address*. Because R2 is aware of the VM's *Care-of Address*, it performs NAT on the packets and send them to R1 using IP routing.

5) C to B:* Because the migrated VM is able to estabilish a communication with a Generic Host as of case 1, it is also able to communicate with a Host on the Foreign Netwok. If desired, the Service Switch R1 can use ICMP Redirect messages to allow the direct communication of the VM with the host on the foreign network.

6) B to C:* The communication of a foreign network's host with the migrated VM happens as in case 2. Once again, the Service Switch R1 can use ICMP Redirect messages to optimize the communication.

7) C to D*:* Because both VMs are configured to talk directly to each other, the only operation needed at the Service Switch is to avoid its ARP proxying for the migrated VMs'addresses.

8) C to D**:* This case can be resolved as a simple combination of the previous cases.

V. Evaluation

To evaluate the feasibility of our solution, we realized a software implementation using standard GNU/Linux tools such as *NetFilter/iptables* and Linux Kernel 2.6's proxy ARP support. The implementation has been evaluated on the testbed shown in figure 4, comprising four physical servers, one acting as external host, one acting as datacenter edge-router, and two playing the role of virtualization containers. Virtualization containers are physical servers configured with the Xen Hypervisor [10]. Each virtualization container is able to host one or more VMs. One of these VMs acts as a router, with one interface connected to the edge-router and another connected to a virtual LAN. Virtual LANs are realized by means of the GNU/Linux software bridge, and connect all VMs hosted by a physical server.

In order to obtain the transparent migration of the "mobile VM", as shown in Figure 4, from the Home Network 192.168.11.0/24 to the Foreign Network 192.168.10.0/24, we need to configure the testbed with 6 NetFilter rules, (two rules for each of the three routers R1, R2, and R3), and 2 additional IP routes, (one in R2, and another in R3). More precisely, R1 and R3 are instructed to perform:

- destination NAT translating 192.168.11.3 in 192.168.10.3;

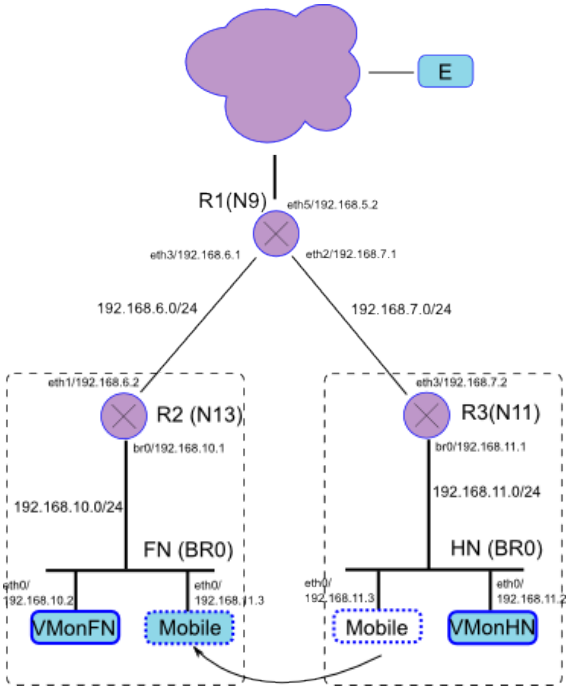

Fig. 4. Testbed

- source NAT translating 192.168.10.3 in 192.168.11.3.

R2 is instructed to perform the opposite translations, with the rules:

- destination NAT translating 192.168.10.3 in 192.168.11.3;
- source NAT translating 192.168.11.3 in 192.168.10.3.

By applying these rules, a migrated IP address totally disappears from packet headers when packets traverse the datacenter's network on the path between access and core layers. It is worth noting that NAT rules applied at R1 are not strictly necessary: applying rules to just R2 and R3 resembles the plain Mobile IP operations (apart from the use of tunnels). Anyhow, by exploiting the full control of the whole datacenter's network, we can optimize the network traffic flow, performing an early redirection of packets, so to avoid packets go firstly to R3 and then to R2. Notice also that because the migrated VM must discover the "new" MAC address for its default gateway, communications may not work until the VM's ARP cache is renewed. (The same happens for hosts on the VM's Home Network, with the VM's MAC address). To make this process faster, i.e., not to wait for ARP cache expiration, one can force the renewal of the ARP cache entries by using gratuitous ARP messages.

To further evaluate our design, we performed two simple tests. In the first one we used ping to measure the mean roud trip time (RTT) between an "external" host and the migrated VM, in three different scenarios: i) before migration; ii) after migration, without configuration of the core layer router (i.e. resembling the "classical" mobile IP configuration); iii) after migration, with core layer router configuration, as previously explained in this paper.

Ping test	Mean RTT
non migrated	2.573ms
migrated (scenario ii)	3.192ms
migrated (scenario iii)	2.034ms
Web Server test	Mean Response Time
non migrated	3.4ms
migrated (scenario iii)	3.1ms

TABLE I
TESTS RESULTS

In the second test the migrated VM runs a web server. We measured the response time before and after the migration. We performed 10 http GET requests per second, for a period of 100 seconds, downloading a 1 MB file. Results for both tests are shown in table I.

Notice that due to some configuration bias among machines used to perform our tests, response times are even better after the migration process. More complex migration scenarios, e.g. involving multiple migrated VMs, also were tested in our prototype, but cannot be described here in more details due to space contraints.

VI. RELATED WORK

The problem of properly re-designing the networking infrastructure of modern datacenters is today under the spotlight of several research groups and big companies. The main challenge and goal is to achieve the ability to assign any server to any service, a property called *agility* in [11]. To this purpose, a few papers have proposed innovative solutions aimed at radically changing the way the network infrastructure of a datacenter is built. For instance, Greenberg et al. propose in [11] an innovative architecture, called *VL2*, that is organized in a flat scheme and operates like a very large switch. VL2 claims to be able to organize any set of servers in the datacenter in a virtual layer 2 isolated LAN. VL2 can be implemented with commodity switches with layer 3 functionality, but it requires modification to the end-system networking stack and a flat addressing scheme, supported by a directory service.

The potential and the costs of live migration of Virtual Machines in Cloud-enabled large scale datacenters has been investigated in [12]. The experimental evaluation conducted by the authors of this paper shows that live migration needs to be carefully managed in SLA-oriented environments requiring more demanding service levels.

In the last few years, other papers have presented similar techniques aimed at allowing live migration of Virtual Machines across different IP subnets [3], [13], [14]. The solution presented in this paper applies to migration within a single datacenter and does not require any modification to the public Internet. Moreover, it does not pose any requirement on the addressing scheme to be used in the datacenter, and does not require the establishment of IP tunnels.

As we pointed out in section V, our solution can be easily implemented with current devices, as it employs standard layer 2 and layer 3 functions, such as IP NAT-ting and ARP proxying. For an efficient implementation of the required behaviour and an easier configuration management of the devices, OpenFlow, a newly proposed open standard API for datacenter devices [15] [16], is an useful tool for the actual deployment of our solutions in real large-scale datacenters.

VII. CONCLUSION

Engineering the networking infrastructure of modern datacenters for Cloud Computing is today a very important problem. Cloud-enabled datacenters, in fact, need advanced support for an integrated management of Virtual Machines. In this paper we propose an innovative solution, based on the coordinated use of NAT rules and ARP proxying, for the problem of transparently migrating Virtual Machines across multiple IP subnets within a single datacenter. Our approach can be easily implemented with current network devices without any modification to their hardware. Our initial prototype is completely implemented in software and makes use of standard layer 2 and layer 3 functions.

REFERENCES

[1] National Institute of Standards and Technology, "NIST Definition of Cloud Computing," 2009. [Online]. Available: http://csrc.nist.gov/groups/SNS/cloud-computing/

[2] "EC2 web site." [Online]. Available: http://aws.amazon.com/ec2/

[3] V. Manetti, R. Canonico, G. Ventre, and I. Stavrakakis, "System-Level Virtualization and Mobile IP to Support Service Mobility," in *Proceedings of the 2009 International Conference on Parallel Processing Workshops, ICPP'09*. IEEE Computer Society, September 2009, pp. 243–248. [Online]. Available: http://dx.doi.org/10.1109/ICPPW.2009.85

[4] Cisco Systems, *Cisco Data Center Infrastructure 2.5 Design Guide*, March 2010.

[5] Juniper Networks, *Data Center LAN Connectivity Design Guide*, 2009.

[6] K. Sato, H. Sato, and S. Matsuoka, "A Model-Based Algorithm for Optimizing I/O Intensive Applications in Clouds Using VM-Based Migration," in *Proceedings of IEEE/ACM CCGRID '09*, May 2009, pp. 466–471.

[7] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd Symposium on Networked Systems Design & Implementation - Volume 2*, ser. NSDI'05. USENIX Association, 2005, pp. 273–286.

[8] "VMWare vMotion," http://www.vmware.com/products/vmotion.

[9] C. Perkins, "RFC 3220 - IP Mobility Support for IPv4," 2002.

[10] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.

[11] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "VL2: a scalable and flexible data center network," *Communications of the ACM*, vol. 54, pp. 95–104, March 2011.

[12] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation," in *Proceedings of the 1st International Conference on Cloud Computing (CloudCom '09)*. Springer-Verlag, 2009, pp. 254–265.

[13] E. Silvera, G. Sharaby, D. Lorenz, and I. Shapira, "IP mobility to support live migration of Virtual Machines across subnets," in *Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference*. ACM, 2009, pp. 13:1–13:10.

[14] Y. Kanada and T. Tarui, "A Network-Paging Based Method for Wide-Area Live-Migration of VMs," in *Proceedings of the 25th International Conference on Information Networking (ICOIN 2011)*, January 2011.

[15] *The OpenFlow Switch Consortium - http://www.openflow.org*.

[16] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communications Review*, vol. 38, pp. 69–74, March 2008.