An Architecture for Adaptation of Virtual Networks on Clouds

Carlos R. Senna, Milton A. Soares Jr., Luiz F. Bittencourt, and Edmundo R. M. Madeira
Institute of Computing - University of Campinas (UNICAMP)
Av. Albert Einstein, 1251, 13083-852, Campinas, São Paulo, Brazil
{crsenna, bit, edmundo}@ic.unicamp.br, milton@lrc.ic.unicamp.br

***Abstract*— Virtual networks are a new research topic advocated to increase flexibility, manageability and isolation in the Internet. However they introduce many open issues to become practical in real scenarios. On the other hand, cloud computing provides elasticity, where availability scales up on demand, with resources being offered frequently as virtualized services over the Internet. The use of virtual networks as a mechanism in cloud computing can aggregate traffic isolation, improving security and facilitating pricing. Also, it allows us to act in cases where the performance is not in accordance with the contract for services between the customer and the provider of the cloud. In this paper we propose an architecture for the deployment of clouds over virtualized networks. In addition, we experimentally evaluate how the virtual network manager can benefit from different virtual network configurations to improve users' quality of service.**

I. Introduction

The cloud computing paradigm provides resources which scale up dynamically and are frequently virtualized as services over the Internet [1]–[5]. The Internet model, based on end-to-end data transfer services and the TCP/IP protocol stack, accelerated its growth, but caused some structural problems like scalability, management, mobility, and security. Simple tasks such as configuration and optimization require the intervention of administrators that may result in service interruption.

Nowadays, the new approach for the future Internet proposes the pluralism of architectures and defines that network providers should be split into service and infrastructure providers and proposes the use of virtualization [6], [7]. This way, the network service providers instantiate virtual networks over the substrate of the infrastructure providers. Each virtual network can have its own protocols and configurations, in accordance with the objectives of the service running over it.

The use of virtual networks as a tool in cloud computing can aggregate traffic isolation, improving security and facilitating pricing. Also, it allows us to act in cases where the performance is not in accordance with the contract for services between the customer and the provider of the cloud. The combined use of the technologies described hitherto opens a new horizon of possibilities, making the management of the cloud much more complex, especially if offering autonomic aspects.

In this paper we present an architecture for management and adaptation of virtual networks on clouds. Our architecture is based on Service Oriented Computing (SOC) [8], and allows users to establish connections among services, organizing them as workflows. Its infrastructure is composed of a network substrate, a set of software tools for creating on demand virtual networks, a computational grid, and a workflow management system.

The proposed infrastructure allows the creation of virtual networks on demand, associated with the execution of workflows, isolating and protecting the execution environment. Also, it provides performance monitoring of virtual networks by acting preemptively in the case of performance dropping below the stated requirements. The management acts autonomously changing routes automatically and without interruption of services. To validate the proposed architecture, we built a prototype on a testbed, which we used to execute image processing workflows utilized in e-Science applications. We show results of real workflow executions in our testbed to evaluate the network performance, the overheads involved when using virtual routers, how virtual network channels behave with data flow transmission, and how the adaptation provided by the virtual network management system can benefit the workflow execution.

This paper is organized as follows. Section II discusses some related works in network virtualization and clouds. In Section III we present the testbed infrastructure deployed to perform our experiments, while Section IV describes the experimental setup, scenarios, and results. The conclusion and future works are presented in Section V.

II. Related Works

Virtual networks are a new research topic advocated to increase flexibility, manageability and isolation in the Internet. However they introduce many open issues to become practical in real scenarios. These perspectives and research challenges are presented in [9]. In this paper we explore the interfacing between infrastructure and service providers. Our scenarios use real services, a workflow application running in computational resources connected by virtual networks. The virtual network management system provides an interface to adjustments required by the workflow manager.

The implementation of virtual networks, its performance issues and trends, are addressed in [10]. We use its virtual machine approach to construct our virtual networks. Although our focus is not on performance, we could evaluate our results and assess if they are factual.

This is a pre-print version.
The final version is available at the publisher's website.

TABLE I
RESOURCES IN THE TESTBED.

Name	Processor	Clock	Cores	RAM
Apolo	Pentium 4	3.00 GHz	2	2.5GB
Nix	Core2 Quad Q6700	2.66 GHz	4	8 GB
Hermes	Core2 Quad Q6700	2.66 GHz	4	8 GB
Artemis	Xeon 3040	1.86 GHz	2	1 GB

In [11], the authors present a resource management framework for VN-based infrastructure providers. In this work, an architecture called Local Resource Manager (LRM) was developed to monitor and control virtual resources in a physical machine and to provide an interface with external clients/agents to do a high-level management. They extend the Xen tools to enable a fine grain, self-adjusting virtual resources control. The evaluation was performed with an implementation of a mechanism for dynamic adjust of CPU resources based on the application requirements of QoS. In our work we are not interested in isolating and controlling the resources in the hosts of the computer environment, but the ones of the network that interconnect then.

Hao et al. [12] propose mechanisms to migrate virtual machines in clouds within different networks. As stated by the authors, this demands network reconfiguration to offer uninterrupted services for the cloud users, which is achieved through network virtualization. However, the authors do not evaluate performance issues when reconfiguring the virtual network.

Our work contributes to the decision on how to reallocate data paths among different virtual networks in order to achieve a satisfiable performance in a cloud computing infrastructure, as the one proposed in [12]. This reconfiguration is important in the virtual network management in order to efficiently use the available links by allocating virtual networks according to the current network usage needs. Such actions can help in obeying SLA contracts, giving priority to flows from users or applications with more strict requirements.

III. THE TESTBED INFRASTRUCTURE

We deployed a testbed to execute our experiments using the virtual network. The infrastructure is composed of a network substrate, a set of software tools for creating on demand virtual networks, a computational grid, and a workflow management system. The testbed receives as input a set of workflows used to evaluate different strategies of network virtualization. Our infrastructure uses the Globus Toolkit (GT) [13] deployment, an OGSA (Open Grid Service Architecture) [14] implementation. In the OGSA, all resources (physical or virtual), are modeled as services, bringing to the grid the concepts offered by Service Oriented Computing (SOC). Our base system is a GT version 4 deployed on 4 resources: *Apolo*, *Artemis*, *Hermes*, and *Nix*, all with Debian Linux connected by the network substrate. Resources characteristics are summarized in Table I.

A. Virtual Networks Testbed

Each virtual network created in our testbed uses two virtual routers. These virtual routers are located at the real hosts *Zeus* and *Dionisio*, as shown on the top of the Figure 1. For example, to bring virtual network A to operation, it is necessary to instantiate the virtual routers *horizonzeusA*, at the real host *Zeus*, and *horizondionisioA*, at the real host *Dionisio*. In our testbed, we instantiated 4 virtual networks to perform the experiments. The bottom part of Figure 1 shows *Apolo* and *Artemis* connected by the virtual network A (IP 10.10*). Similar instantiations were made for the virtual networks B (IP 10.20*), C (IP 10.30*), and D (IP 10.40*). The paths for each virtual network can be mapped in one of two possible physical paths between the real hosts Zeus and Dionisio: an 100Mbps link and an 1Gbps link.

The main tools used to build our testbed are qemu, KVM, and libvirt. Qemu [15] is a processor emulator which can also be used as a virtualization platform. The Kernel-based Virtual Machine (KVM) [16] is a full virtualization hypervisor based on the machine emulator qemu. KVM is a free software under the GPL and open-source, and it allows the use of external tools, such as libvirt, to control it. Libvirt [17] is an API to access the virtualization capabilities of Linux with support to a variety of hypervisors, including qemu, KVM, and Xen, and some virtualization products for other operating systems. It allows local and remote management of virtual machines. With libvirt it is possible for a management agent to use the same code to request information regarding the performance of a virtual link independent of the hypervisor running in the virtual routers.

B. Workflow Management System

In order to enact real workflows in our experiments, the management of the service compositions in our infrastructure is made by the GPO (Grid Process Orchestration) [18], a middleware for service workflows execution in the grid (Figure 2). The GPO allows the creation and management of application flows, tasks, and services. The GPO uses workflows built with the GPOL (GPO Language) [18]. The GPOL is based on concepts of service orchestration from WS-BPEL [19], with added specific directives for grids, such as state maintenance, potentially transient services, notification, data-oriented services, and groups. The language includes variables, lifecycle, fabric/instance control, flow control, and fault handling. Additionally, it allows the user to start task executions, service executions, and workflow executions in sequence or in parallel. The scheduling service (SS) is responsible for distributing the workflow services to be executed in the available resources. To accomplish this, the scheduling service may implement different algorithms with different optimization objectives, and decide which one to use depending on the application or current environment characteristics. Information about the available resources in the grid can be obtained through the resource monitor (RM). The RM operates in a distributed manner, maintaining one instance on each computing resource and providing on demand information for other services. Our

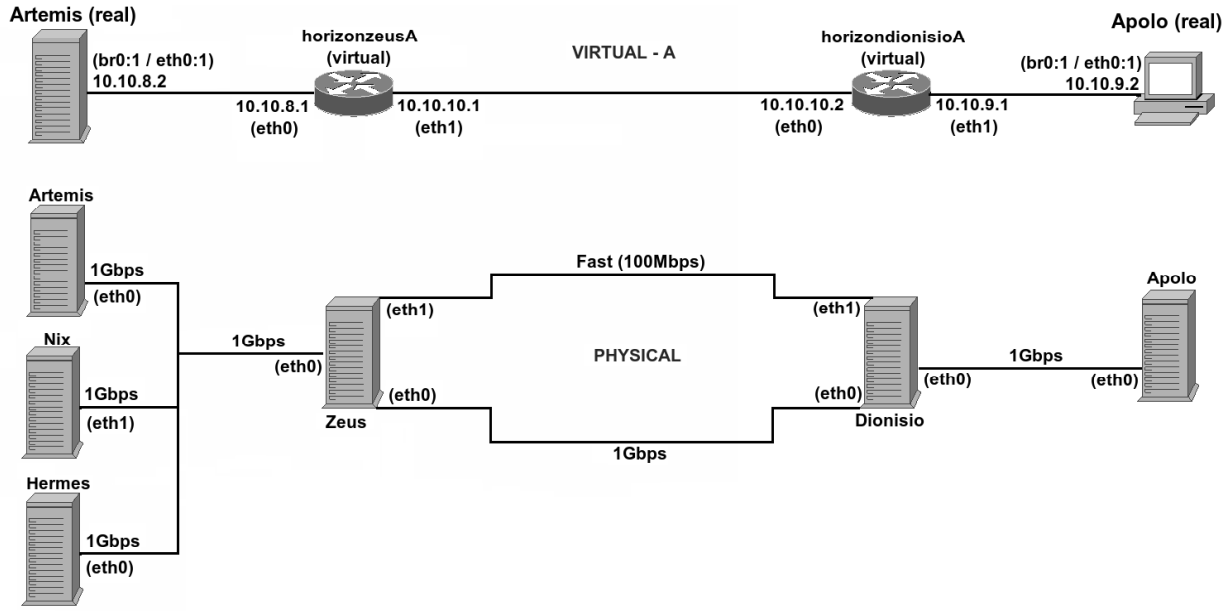


Fig. 1. Network substrate.

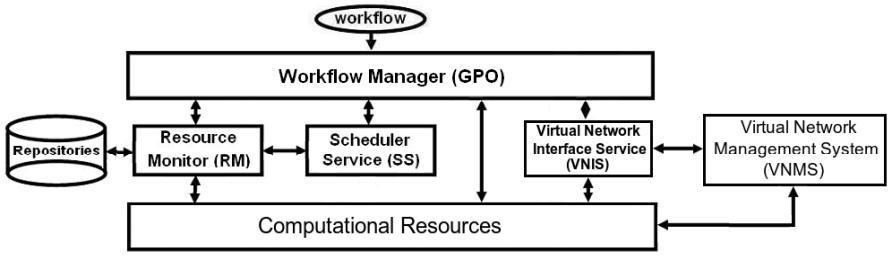


Fig. 2. The Workflow Management Architecture.

middleware provides the monitoring of workflow executions. The GPO monitors the execution times of each section specified in the GPOL workflow, including the time spent on each operation invoked in the process, registering them in a log file exclusive for each workflow.

The integration between the GPO and management of virtual networks is done through the Virtual Network Interface Service (VNIS). Using the VNIS, the workflow manager requests the network to be used for workflow execution. During the workflow execution the RM monitors the performance of virtual network links and can identify problems such as miscommunication or underperforming. In such cases, the GPO notifies the virtual network management system (VNMS) [20], requesting improvements in the performance of a link or the entire network when it is appropriate.

In this paper we implemented and deployed a system using the architecture proposed in Figure 2. Therefore, our experiments comprise the virtual networks, the workflow specification in GPOL, the workflow emulation using the emulator service over the GPO middleware, including workflow data transfers over different configurations of the virtualized network along with the adaptation provided by the VNMS component.

The use of real applications in our testbed is essential, but there exist limitations to implement all the necessary services for all workflows and deploy them on all available resources. Such limitations include personnel and software requirements, which can be conflicting, making it not possible to experiment the necessary service-resource combinations to evaluate performance and strategies of network virtualization. To contour this situation, we created an emulation service which mimics many aspects of the workflows execution [21]. Using our emulator service we built emulation workflows which present a quite similar behavior to the real application workflows. In this paper we used the median filter workflow to perform virtual network evaluations. The median filter is an image processing application [22] that can be executed in parallel by splitting the image into pieces and merging the results back into one single image. The median filter substitutes the value of a pixel by the surrounding values on its neighborhood. Figure 3 shows the file is divided into 5 parts, the slices files processed on parallel way, and the slices files merged on the final filtered file [18].

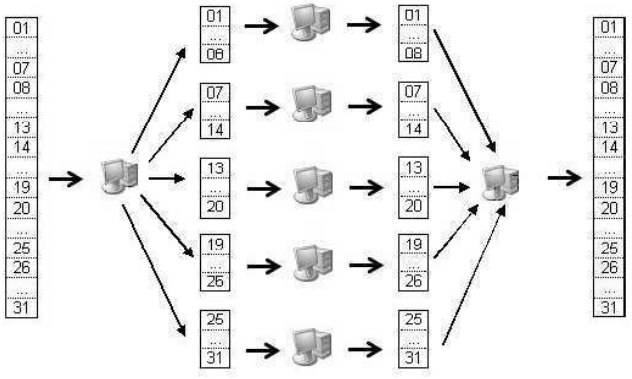


Fig. 3. Median filter workflow example.

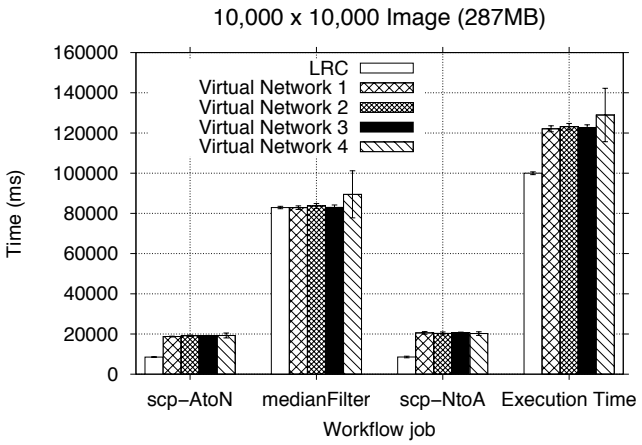


Fig. 5. Results for images of size $10,000 \times 10,000$ in all available networks.

IV. Experimental Results

We performed real workflow executions in our testbed to evaluate the network performance in three aspects:

1) The overheads involved when using virtual routers.
2) How virtual network channels behave with concurrent data flow transmissions.
3) How the adaptation provided by the virtual network management system can benefit the workflow execution.

The evaluations presented in this section are handy for the development of advanced management algorithms for the virtual network substrate. In addition, the experiments can provide background for the development of autonomic management agents capable of switching flows across network links when abnormal behavior is observed.

A. Virtual Network Overheads

We start by evaluating the overhead introduced by the virtual routers when compared to the transmissions without them, i.e., in a switched gigabit ethernet network. In this scenario we measured the times taken to execute a simple workflow which performs a median filter in an image. It uses 3 services and performs 2 data transfers, as shown in Figure 4.

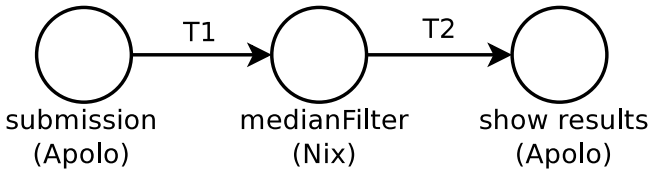


Fig. 4. Workflow used in the virtual network overhead evaluation.

The workflow receives a user submission in *Apolo* and sends the image to be processed in *Nix* (transfer 1 - T1), where the median filter is applied to the image. After that, the image is copied back to *Apolo* (T2), where the resulting image is shown to the user. We executed this median filter workflow for images of $10,000 \times 10,000$ pixels. We compare executions of the workflow in the Computer Networks Laboratory (LRC) gigabit ethernet network with the execution in our testbed using the gigabit links available from the virtual routers. Figure 5 shows the execution times of each workflow step averaged over 5 executions with confidence interval of 95%.

Figure 5 presents the execution times of the workflow services in all available networks in our testbed: 4 virtual networks plus the LRC gigabit network. We can observe that both data transfer services (T1 and T2) double their execution times when using the virtual network to transfer the $10,000 \times 10,000$ pixels (287 MB) images. This impacts the final execution time of the workflow, which is increased by 23%. Therefore, the 4 virtual networks present similar behavior with some overhead over the LRC network.

Not surprisingly, the virtual routers introduce overheads to the file transfers needed by the workflow [10]. This is caused by a variety of factors, such as:

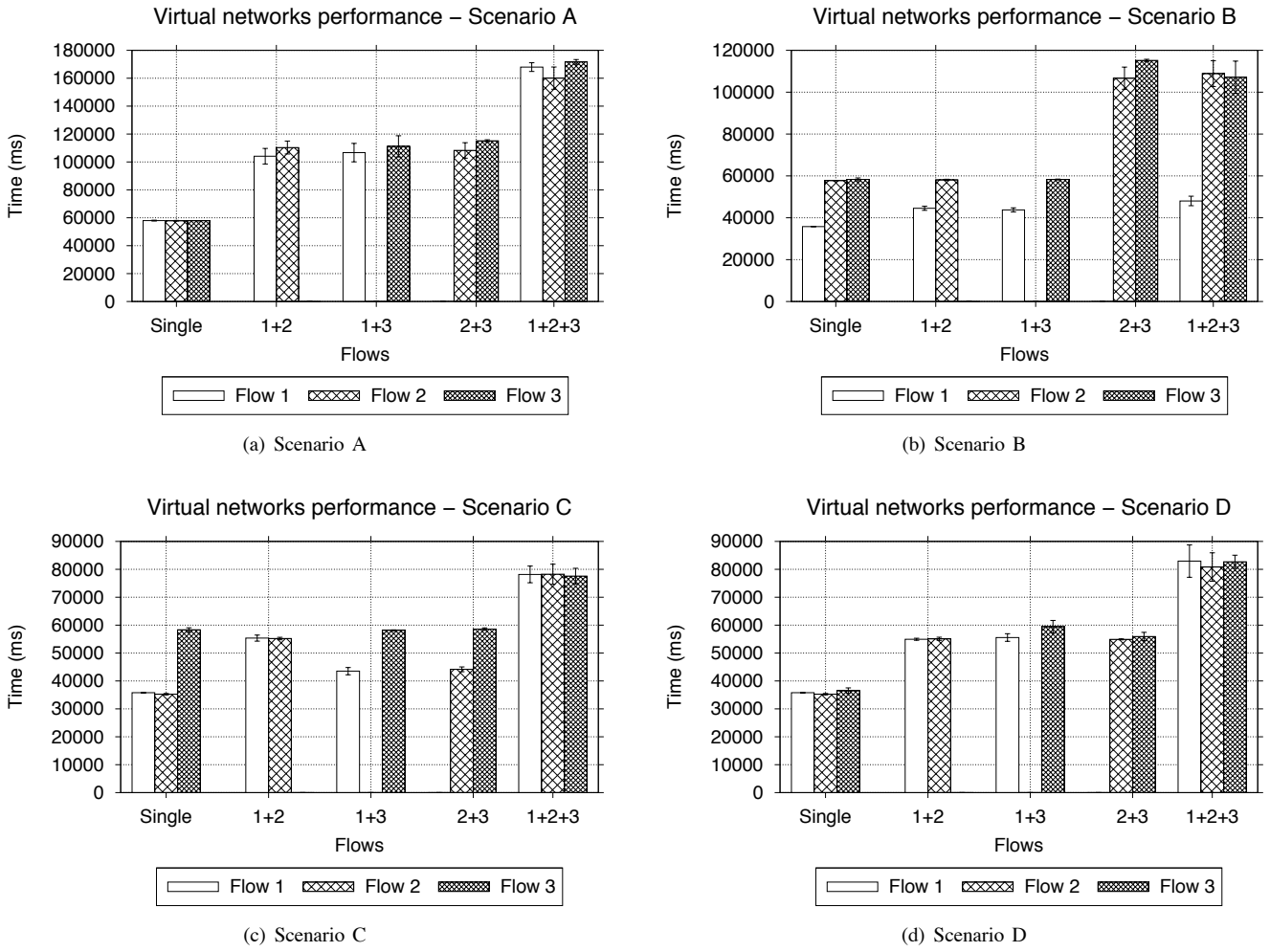
- Complexity in the packet forwarding through virtual machines;
- Multiplexing packets to virtual machines through bridges;
- While in the LRC network the data path between *Apolo* and Nix has a single hop, the virtual network transfer passes through 3 hops, introducing queue/propagation overheads to the data stream; and
- *Zeus* and *Dionisio* introduce overheads when processing the incoming data and forwarding it to the destination.

In this paper we focus on the management of the flows through the available virtual networks, therefore we accept this overhead as part of the virtual network infrastructure.

B. Virtual Network Performance

In the previous section we evaluated the performance of an 1Gbps link in a virtual network with a single data flow. In this section we add an 100Mbps link, and we analyze the performance of both channels using up to 3 data flows. Each data flow is a transfer of a $15,000 \times 15,000$ image file (644 MB). For such evaluation, we consider 4 routing scenarios (A, B, C, and D), as shown in Table II.

In each scenario, we measured the time taken to send all combinations of 3 flows, with each one being the data transfer of a $15,000 \times 15,000$ image file. The flows are as follows.



(a) Scenario A

(b) Scenario B

(c) Scenario C

(d) Scenario D

Fig. 6. Virtual network performance routing up to 3 flows through 1Gbps and 100Mbps links.

TABLE II
SCENARIOS USED TO EVALUATE THE VIRTUAL NETWORK PERFORMANCE.

	100 Mbps			1 Gbps		
	Flow 1	Flow 2	Flow 3	Flow 1	Flow 2	Flow 3
A	**X**	**X**	**X**			
B		**X**	**X**	**X**		
C			**X**	**X**	**X**	
D				**X**	**X**	**X**

- **Flow 1**: TCP data transfer from *Nix* to *Apolo* using the virtual network $10.10.*$.
- **Flow 2**: TCP data transfer from *Hermes* to *Apolo* using the virtual network $10.20.*$.
- **Flow 3**: TCP data transfer from *Artemis* to *Apolo* using the virtual network $10.30.*$.

Figure 6(a) shows results for scenario A (all flows routed through the 100Mbps link), where *Single* is the control measurement, i.e., the time taken for transferring each flow alone. We can note that the transmission of flows 1 and 2 concurrently (labeled "1+2") remains a little below the double of the control time, as expected. The same happens when only flows 1 and 3 and when flows 2 and 3 are transmitted. When the 3 flows are transmitted, all of them have the performance significantly worsened by the concurrency.

When we consider scenario B (Figure 6(b)), the control for flow 1 drops, as expected, since it is now in the gigabit link. Note that flow 1 has a similar transfer time with all combinations of flows, since it is always alone in the 1Gbps link. However, we can observe some overhead in flow 1 when increasing the number of flows due to processing concurrency in physical machines where virtual routers are instantiated. Flows 2 and 3 perform similarly to the control time when transmitted only with flow 1. When flows 2 and 3 are transmitted together, they share the 100Mbps link, what worsens their performance. By comparing scenarios A and B we note that changing flow 1 from the 100Mbps brings benefits to all flows.

In scenario C (Figure 6(c)) we observe that flow 3 is not affected by flows 1 and 2, since flow 3 is the only one in the 100Mbps link, except in the case where all flows are sent together. This makes it clear that the concurrency in the

physical machines where virtual routers are instantiated is also a limiting factor for the virtual network overall performance. In the "$1+2+3$" case, flows 1 and 2 affect each other when sharing the 1Gbps link, making their performance to be closer to the flow 3 alone in the 100 Mbps link.

When all flows are routed through the $1Gbps$ link (scenario D), the results in Figure 6(d) show that any combination of 2 flows results in a higher transfer time when compared to the control measurement, as expected. When all the 3 flows are transmitted, the concurrency makes the transmission time even higher. However, it is important to note that the transmission times of all flows together is smaller than the sum the transmission time of all flows alone. In addition, the transmission of the 3 flows concurrently in the 1 Gbps network, i.e. scenario D, is faster than combinations in scenarios A and B. However, when compared to scenario C, scenario D presents similar performance when all 3 flows are being transmitted. Therefore, scenarios C and D are valid options for transmitting the 3 flows in the fastest manner in our testbed. These results can help in the development of adaptation strategies for executing workflows over virtual networks.

C. Network Adaptation

In this section we present results on how the virtual network management system (VNMS) could adapt the routing of flows during the execution to achieve a better performance. We use the same scenarios names as in the previous section to refer to different distribution of virtual networks over the links.

We executed the median filter workflow splitting a $15,000 \times 15,000$ image file in 3 pieces, sending them to be processed in parallel on different resources, and receiving the 3 pieces back to generate the final resulting image (Figure 7). The workflow steps are as follows.

1) *Apolo* breaks the image in 3 pieces.
2) *Apolo* transfers in parallel one piece to *Nix* using the virtual network $10.10.*$, one piece to *Hermes* using the virtual network $10.20.*$., and one piece to *Artemis* using the virtual network $10.30.*$. At this moment, all flows are routed through the 100Mbps link.
3) *Nix*, *Hermes*, and *Artemis* execute the median filter on their image pieces.
4) *Apolo* gets all the pieces back from *Nix*, *Hermes*, and *Artemis*. Here the flows are routed through different paths, using different scenarios as in the previous section.

Upon the execution of the workflow, the VNMS can choose how to distribute the data flows in the virtual networks depending on the workflow's requirements. For example, if the transmission times for the first three transfers in the 100Mbps link are above the workflow requirements (given by a service level agreement - SLA, for instance), the VNMS can choose to migrate some networks to the 1Gbps link. Figure 8(a) shows potential gains of such adaptation when executing the workflow. We can observe that, if the VNMS chooses to move from scenario A to scenario B, it would improve the transfer times for the returning image pieces (Transfers 4-6). As a consequence, the workflow execution time would also be reduced. However, if the workflow requirements are tighter, the VNMS could choose to move to scenario C or D, achieving a data transfer time for the returning image pieces closer to the non-virtual network.

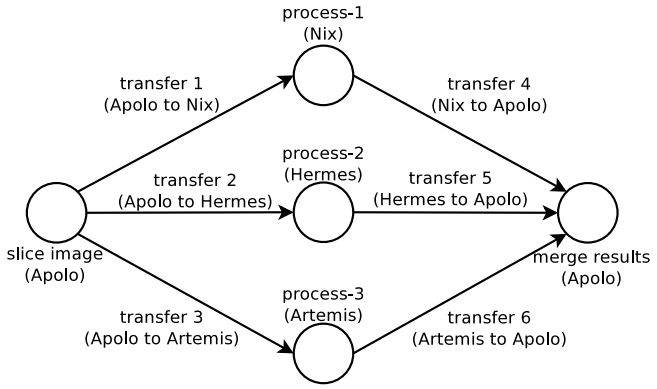


Fig. 7. Workflow used in the virtual network adaptation evaluation.

Figure 8(b) shows the time taken by each transfer and median filter processes. We can observe that transfers 4 to 6 show the same transfer time pattern achieved in the four scenarios from the previous section. For example, *Transfer 4* is moved to the 1Gbps link in scenario B, having a transfer time close to the one presented by flow 1 in the previous section. This would be useful when there exist a priority flow in the network, which could be routed alone through the $1Gbps$ network. In scenarios C and D the transfer times for all transfers are similar, corroborating the experiments from the previous section by showing that both options are valid to more efficiently transfer three data flows.

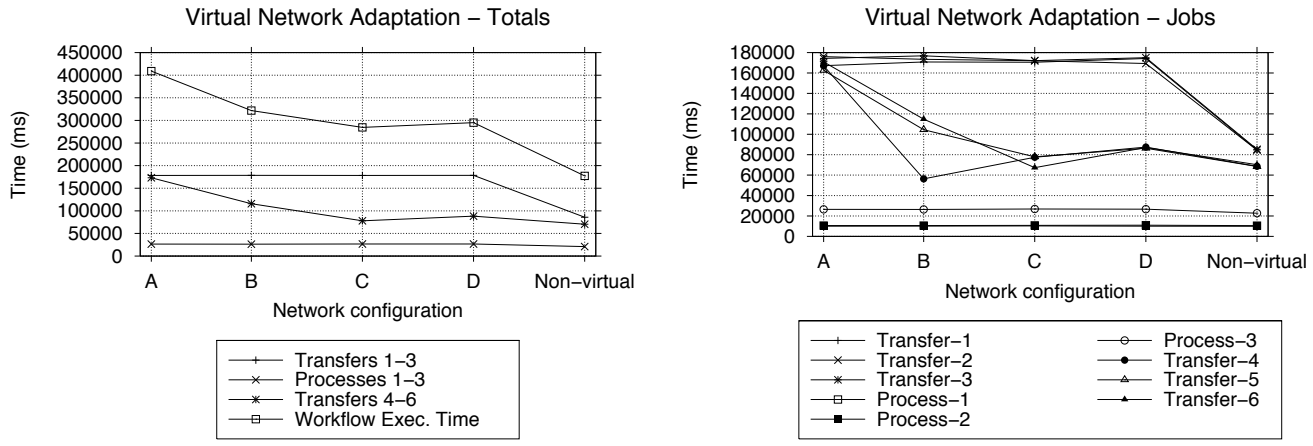
D. Adaptation Case Study

In this evaluation we present results of a case study on flow priority. We consider two workflows: (i) the workflow $W1$ from Figure 9 for a $20,000 \times 20,000$ image, which is split for processing; and (ii) a higher priority workflow $W2$ from Figure 4 with a $10,000 \times 10,000$ pixels image.

The case study was performed as follows. First, W_1 starts its execution, splitting the image file and sending its pieces to *Nix*, *Hermes*, and *Artemis* using the 1Gbps link provided by the virtual network. After the processing, a user W_P is submitted to execution in Apolo to be processed in Hermes. At this point, the network will experience concurrency among 4 data flows: $T2, T4, T5$ from W_1 and the first data dependency from W_P ($T1$) in the gigabit link. At this moment, the GPO requests priority to the VNMS through the VNIS. The objective of the VNMS now is to satisfy the higher priority from W_P. To achieve this, it must reconfigure the virtual networks to provide a faster transfer for *T1* from W_P.

We analyze 3 possible actions to be taken by the VNMS:

- **Action 1**: Take no action. Simply allow all flows to go through the 1Gbps link.
- **Action 2**: Reconfigure the network so that flow $T1$ from W_P can use the 100Mbps link exclusively.



(a) Totals

(b) Jobs

Fig. 8. Case study of virtual network adaptation with the workflow of Figure 7.

- **Action 3**: Reconfigure the network so that flow $T1$ from W_P can use the 1Gbps link exclusively, i.e., changing all the other flows to the 100Mbps link.

Results for data transfers in this concurrent workflow execution are shown in Figure 10. In the Action 1 case, the executions of the whole workflows (i.e., including all the processing times – not shown in the figure for the sake of cleanliness) are $206,038$ms for W_1 and $70,605$ms for W_P. Action 2 has shown to be the worse option for the priority workflow, since it worsens the data transfer time for its data dependencies (W_P/T1 and W_P/T2). In addition, in this case the total execution times for W_1 and W_P are $216,825$ms and $81,368$ms respectively. On the other hand, when Action 3 is taken, data transfer times from the priority workflow W_P are shorter, making the execution time of the whole workflow to drop to $50,598$ms.

This adaptation case study shows that, when a priority flow arrives, the best option is to route it through the gigabit link alone, as expected. However, as a second option, routing it through the gigabit link along with other 3 flows may still be better than routing the priority flow alone in the 100Mbps link.

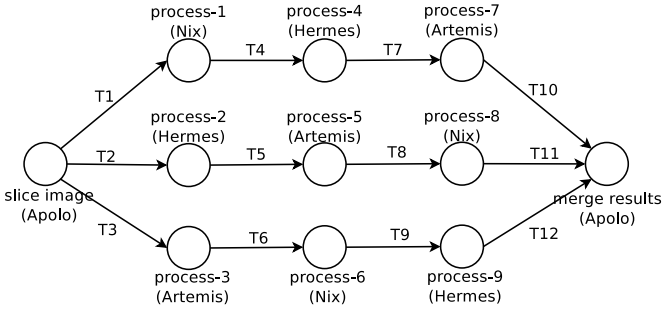


Fig. 9. Workflow to apply three filters sequentially to the image file.

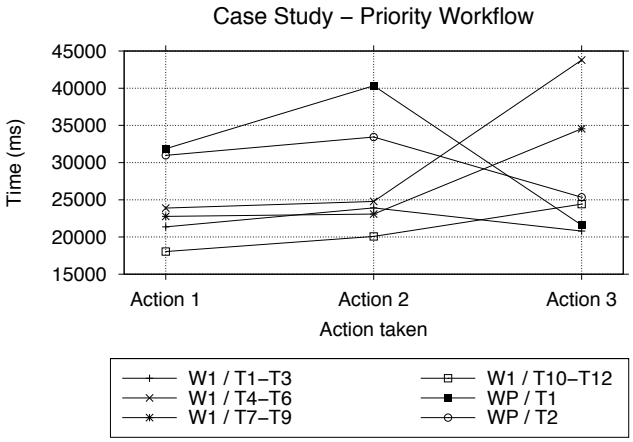


Fig. 10. Data transfer times according to the action taken.

V. Conclusion

The new paradigm of networks as a service (NaaS) based in the network virtualization can bring benefits to cloud computing as aggregate traffic isolation, improving security, and facilitates pricing. This new mechanism permits, for example, to act in cases where the performance is not in accordance with the contract for services between the customer and the provider of the cloud. However, the union of these technologies opens a new horizon of possibilities, making the management of the cloud much more complex.

In this paper we present an architecture for management and adaptation of virtual networks on clouds. Our infrastructure allows the creation of virtual networks on demand, associated with the execution of workflows, isolating and protecting the user environment. The virtual networks used in workflow execution has its performance monitored by our manager which acts preemptively in the case of performance dropping below stated requirements.

To validate the proposed architecture, we built a prototype

of testbed to provide insights on how the virtual network management system can act to offer a better quality of service to the user. The results of image processing workflow executions showed that the management and adaptation of virtual networks is able to improve the data transfer times for the executed workflows.

As future work, we consider the development of algorithms to allocate the virtual networks in the available links depending on the current load/priorities of data transfers in the cloud. This would impact on the quality of service offered to the user as well as in the profit of the cloud provider by offering better SLA contracts.

ACKNOWLEDGMENT

We would like to thank CAPES, FINEP, FAPESP (09/15008-1), and CNPq for the financial support.

REFERENCES

[1] "The nist definition of cloud computing 15," National Institute of Standards and Technology (NIST), Tech. Rep., July 2009. [Online]. Available: http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc

[2] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the internet impasse through virtualization," *Computer*, vol. 38, no. 4, pp. 34 – 41, April 2005.

[3] S. Zhang, S. Zhang, X. Chen, and X. Huo, "Cloud computing research and development trend," in *Proceedings of the 2010 Second International Conference on Future Networks*, ser. ICFN '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 93–97.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, pp. 50–58, April 2010.

[5] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Gener. Comput. Syst.*, vol. 25, pp. 599–616, June 2009.

[6] "Horizon project: A new horizon to the internet," 2011. [Online]. Available: http://www.gta.ufrj.br/horizon

[7] N. Feamster, L. Gao, and J. Rexford, "How to lease the internet in your spare time," *SIGCOMM Comput. Commun. Rev.*, vol. 37, pp. 61–64, January 2007. [Online]. Available: http://doi.acm.org/10.1145/1198255.1198265

[8] F. Curbera, R. Khalaf, N. Mukhi, S. Tai, and S. Weerawarana, "The next step in web services," *Communications of ACM*, vol. 46, no. 10, pp. 29–34, 2003.

[9] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862 – 876, 2010.

[10] N. Fernandes, M. Moreira, I. Moraes, L. Ferraz, R. Couto, H. Carvalho, M. Campista, L. Costa, and O. Duarte, "Virtual networks: isolation, performance, and trends," *Annals of Telecommunications*, vol. 66, pp. 339–355, 2011.

[11] F. Rodrguez-Haro, F. Freitag, and L. Navarro, "Enhancing virtual environments with qos aware resource management," *Annals of Telecommunications*, vol. 64, pp. 289–303, 2009.

[12] F. Hao, T. V. Lakshman, S. Mukherjee, and H. Song, "Enhancing dynamic cloud-based services using network virtualization," *Computer Communication Review*, vol. 40, no. 1, pp. 67–74, 2010.

[13] G. Alliance, "Globus toolkit," 2011. [Online]. Available: http://http://www.globus.org/toolkit/

[14] I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "The physiology of the grid: An open grid services architecture for distributed systems integration," 2002. [Online]. Available: http://www.globus.org/research/papers/ogsa.pdf

[15] F. Bellard, "QEMU, a fast and portable dynamic translator," in *Proceedings of the annual conference on USENIX Annual Technical Conference*, ser. ATEC '05. Berkeley, CA, USA: USENIX Association, 2005, pp. 41–41.

[16] I. Habib, "Virtualization with kvm," *Linux J.*, vol. 2008, February 2008.

[17] M. T. Jones, "Anatomy of the libvirt virtualization library an api for easy linux virtualization," 2010. [Online]. Available: http://www.ibm.com/developerworks/linux/library/l-libvirt/

[18] C. R. Senna, L. F. Bittencourt, and E. R. M. Madeira, "Execution of service workflows in grid environments," *International Journal of Communication Networks and Distributed Systems (IJCNDS)*, vol. 5, no. 1/2, pp. 88–108, 2010.

[19] "Web services business process execution language version 2.0," OASIS Web Services Business Process Execution Language (WSBPEL) TC, Tech. Rep., April 2007.

[20] C. R. Senna, D. M. Batista, M. A. S. Jr., E. R. M. Madeira, and N. L. S. Fonseca, "Experiments with a self-management system for virtual networks," in *II Workshop de Pesquisa Experimental da Internet do Futuro (WPEIF 2011)*. Campo Grande, MS, Brazil: Brazilian Computer Society, 2011.

[21] C. R. Senna, L. F. Bittencourt, and E. R. M. Madeira, "An environment for evaluation and testing of service workflow schedulers in clouds (to appear)," in *International Conference on High Performance Computing & Simulation (HPCS)*, july 2011.

[22] C. A. Lindley, *Practical image processing in C: acquisition, manipulation and storage: hardware, software, images and text*. New York, NY, USA: John Wiley & Sons, Inc., 1991.