Experiments with a self-management system for virtual networks

Carlos R. Senna, Daniel M. Batista, Milton A. Soares Junior, Edmundo R. M. Madeira, and Nelson L. S. da Fonseca

Institute of Computing - University of Campinas (UNICAMP)
Av. Albert Einstein, 1251, 13083-852, Campinas, São Paulo, Brazil

{crsenna, batista, edmundo, nfonseca}@ic.unicamp.br,
milton@lrc.ic.unicamp.br

Abstract. *This paper presents an agent-based platform to support the development of a self-management system for the network architecture proposed by the Horizon Project. The platform consists of a substrate network, a software for creating virtual networks, and a set of agents that support ontology to enable the development of a Piloting Plane. We also show the prototype built to evaluate a simple ontology for the exchange of information between agents. The self-management system proposed is an intermediary between the control and management entities and the context of network and services. The piloting concepts of the system are realised with the help of multi-agents based on the Ginkgo System. Details about the testbed built to evaluate the proposed system are also presented.*

1. Introduction

Autonomic networks [Gaiti 2006] were proposed to deal with the problem of increasing complexity of telecommunications. They represent a specific topic in the area of autonomic computing [Kephart 2003], a term coined by IBM, intended to deal with complexity by enabling systems to self-manage themselves. This concept is bio-inspired by the autonomic nervous system that carries out the regulation of body in the face of changes in the environment without a conscious control. In the scenario of networks, simple tasks of configuration, optimization, disaster recovery, and security are achieved by the network itself, leaving administrators free to more complex tasks such as setting policies and goals. Nowadays it is also advocated the approach of pluralism of architectures for the future Internet over the one-size-fits-all TCP/IP. The new approach defines that network providers should be splitted into service and infrastructure providers and proposes the use of virtualization. Users request network services from the service providers, which instantiate virtual networks over the substrate provided by the infrastructure providers. Each virtual network can have its own protocols and configurations, in accordance with the objectives of the service running on it, and must have isolation, i.e., the operation of virtual networks does not cause interference among them, although they are on the same infrastructure.

The Horizon [Horizon 2011] project aims to define and validate a new network architecture based on the principles of pluralism and the knowledge plane. To achieve these objectives, it is necessary to have a piloting plane where the decisions are made. This paper presents a self-management system proposed by us, which is within the piloting plane, being an intermediary between the control and management entities and

the context of network and services. Our proposal is to apply concepts of autonomic computing in a scenario of virtual networks through a multi-agent system.

2. Platform architecture

The platform consists of a network substrate, a set of software tools for creating on-demand virtual networks, and a set of agent-based tools that support ontology to enable the development of a Piloting Plane. The following subsections describe the main components of the platform.

2.1 Testbed

Before deploying the proposed self-management system in a real network, it is important to evaluate it in a controlled environment. Figure 1 illustrates the testbed built with this objective.

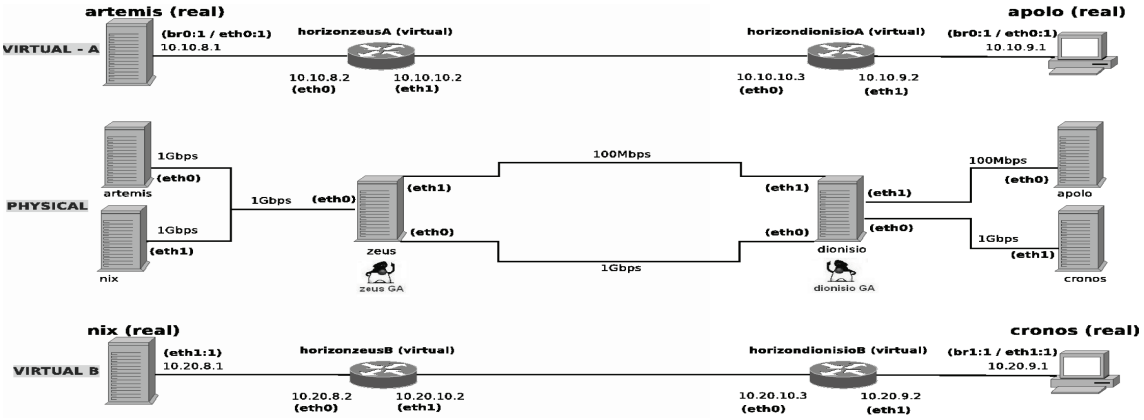

Figure 1. The testbed.

The testbed contains two virtual networks (virtual network A and virtual network B). Each virtual networks contains two virtual routers. The virtual routers are located at the real hosts *zeus* and *dionisio*. To the virtual network A to be instantiated, it is necessary to instantiate the virtual routers *horizonzeusA*, at the real host *zeus*, and *horizondionisioA*, at the real host *dionisio*. Similar instantiations are needed to the virtual network B. Virtual network A is created to interconnect hosts *artemis* and *apolo* through a 2-hop virtual path. Similarly, virtual network B is created to interconnect hosts *nix* and *cronos*. The 2-hop paths of each virtual network can be mapped on one of two possible physical paths between the real hosts *zeus* and *dionisio*: an 100Mbps link and an 1Gbps link. Initially the two virtual paths share the 100Mbps link.

2.2. Tools

The major tools used to build out testbed are qemu, KVM, libvirt and Ginkgo Platform. The qemu is a processor emulator which can also be used as a virtualization platform. The Kernel-based Virtual Machine (KVM) is a full virtualization hypervisor based on the machine emulator qemu. KVM is a free software under the GPL and open-source, and allows the use of external tools to control it, like libvirt. The libvirt 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 that an agent uses the same code to request information regarding the performance of a virtual link independent of the hypervisor running in the virtual routers.

Ginkgo System [Ginkgo 2011] is an agent platform based on autonomic networks. It has the building blocks for the development of a piloting system for computer networks. The framework allows the creation of lightweight and portable agents, which facilitates its implementation in heterogeneous environments: routers, switches, hosts, wired, and wireless networks. The agents play the role of the autonomic manager of autonomic computing. The platform allows to form clusters of agents in neighborhoods. Neighbors exchange information and get a situated view of the network. Thus, besides the local environment, the agent is aware of other network places. This information is stored in the knowledge base that has an information model to facilitate communication between agents. Other data repository is the policy file, which contains application rules. With the environment and application knowledge, the multi-agent system can provide to network the self-knowledge property. The sensing, cognition, and acting of agents are realized by the behaviors. They feed the knowledge base, perceive and predict threatening events, and perform changes on the managed elements. Agents also have a dynamic planner that, with knowledge base information and the rules in the policy file, changes parameters of the behaviors and controls the life cycle of the agent.

3. The multi-agent system and experiment

We implemented a multi-agent system within the testbed to perform the management of virtual networks in context of the Horizon project. The aim of the experiment is to change the mapping of virtual links over paths in the substrate network dynamically and automatically, according to context changing. The resources being monitored are in the virtual networks, not in the physical network. For example, the failure of a resource in the data plane is monitored, which can lead to faults in the management system. In the knowledge base, each network resource is an individual of the information model. Each resource is controlled by a single agent and, therefore, each individual is unique in the knowledge base even after the diffusion. In the information model, the links are directed, i.e., each wire is represented by two links in opposite directions. The agents are located in routers, send data over a directed link, monitor and control this link. The neighborhood consists of all agents of the routers within the domain, so the information is exchanged by broadcast. To make a change in the mapping of the virtual link it is necessary to perform actions on both ends of the physical path. Two agents in separate routers are responsible for a part of the execution (*zeus GA* and *dionisio GA* in Figure 1). Ideally, the two actions occur at the same time to reduce losses in the data plane, so we must create a mechanism to synchronize agents. The Ginkgo platform communication is based on the diffusion of knowledge base, then, we need to extend the information model in order to indicate the status of the resources and agents. With this information, behaviors perform the actions at approximately the same time.

We use the multi-agent system to manage situations that require the exchange of virtual paths during the operation of both virtual networks. The main idea of our experiment is to overload the 100Mbps link and, after that, to migrate just one virtual path to the 1Gbps link. This experiment simulates a scenario where agents located at the routers detect the high utilization of a link and decide to migrate one of the virtual links.

4. Results

In the experiment, the virtual routers are VMs based on KVM. The virtual links are created in the data link layer, with the Ethernet protocol, via virtual interfaces and bridges. The configuration parameters of the agents are in the policy file, and behaviors of the autonomic cycle runs at an interval of 1s in normal state. The use of all physical links in the network are below the threshold set, configured at 50%. When an overload occurs, the cycle frequency runs at an interval of 0.2s.

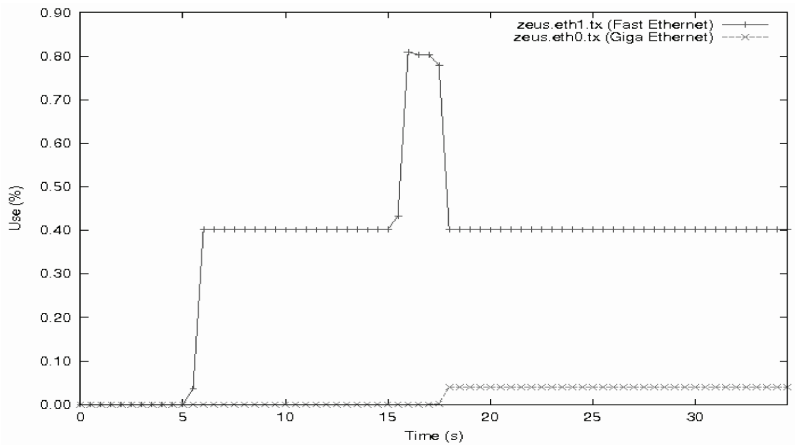

Figure 2. Utilization of the physical links.

The graph of Figure 2 shows the utilization of the physical links throughout the experiment. A flow passing through the virtual network A, with constant rate of 40Mbps generated by the **iperf** application, starts close to 5s. The Fast Ethernet physical link reaches 40% of utilization, not exceeding the threshold. Another flow is started about 15s, through the virtual network B at a constant rate of 40Mbps. The link utilization exceeds the threshold, reaching 80%, and the correction begins to be performed. The ending of the execution occurs near the 18s, when the flow of the virtual network is migrated to the Giga Ethernet link. Therefore, the multi-agent system takes about 3 seconds to prepare and implement the changes.

5. Conclusion

This paper presents an agent-based platform to support the development of a self-management system for the network architecture in the context of the Horizon Project. The proposed system changes the mapping of virtual links if the load of specific physical links increases more than a certain threshold. Although the simplicity of this experiment, the results show the viability of the Piloting Plane.

References

Gaiti, D., Pujolle, G., Salaun, M., and Zimmermann, H. (2006) “Autonomous network equipments”, Springer LNCS, Vol. 3854/2006, pp. 177–185.

Kephart, J. O. and Chess, D. M. (2003) “The vision of autonomic computing”. Computer, vol. 36, no. 1, no. 1, pp. 41–50.

Horizon Project: A New Horizon to The Internet (2011). On line: http://www.gta.ufrj.br/horizon/.

Ginkgo Networks. (2008) “Ginkgo distributed network piloting system. white paper”. On line: www.ginkgo-networks.com/IMG/pdf/WP_Ginkgo_DNPS_v1_1.pdf