Thesis Overview:

An Architecture Model for a Distributed Virtualization System

Pablo Pessolani
National University of La Plata, Argentina
PhD in Computer Science
Advisors: Toni Cortes, Fernando G. Tinetti, Silvio Gonnet
ppessolani@frsf.utn.edu.ar, toni.cortes@bsc.es, fernando@info.unlp.edu.ar,
sgonnet@santafe-conicet.gov.ar

Virtualization technologies are massively adopted to cover those requirements in which Operating Systems (OS) have shown weakness, such as fault and security isolation. They also add features like resource partitioning, server consolidation, legacy application support, management tools, among others, which are attractive to Cloud service providers.

Hardware virtualization, paravirtualization, and OS-level virtualization are the most widely used technologies to carry out these tasks, although each of them presents different levels of server consolidation, performance, scalability, high-availability, and isolation.

The term "*Virtual Machine*" (VM) is used in issues related to hardware virtualization and paravirtualization technologies to describe an isolated execution environment for an OS and its applications. Containers, Jails, Zones are the names used in OS-level virtualization to describe the environments for applications confinement. Regardless of the definition of the virtualization abstraction, ***its computing power and resource usage are limited to the physical machine where it runs.***

The proposed virtualization architecture model breaks this issue, distributing processes, services, and resources to provide distributed virtual environments based on OS factoring and OS containers. The outcome is a Distributed Virtualization System (DVS) which allows running several distributed Virtual Operating System (VOS) on the same cluster. A DVS also fits the requirements for delivering high-performance cloud services with provider-class features as high-availability, replication, elasticity, load balancing, resource management, and process migration. Furthermore, a DVS is able to run several instances of different guest VOS concurrently, allocating a subset of nodes for each instance (resource aggregation), and to share nodes between them (resource partitioning). Each VOS runs isolated within a Distributed Container (DC), which could span multiple nodes of the DVS cluster. The proposed architecture model keeps the appreciated features of current virtualization technologies, such as confinement, consolidation and security, and the benefits of DOS, such as transparency, greater performance, high-availability, elasticity, and scalability.

A DVS allows running multiple Distributed VOSs as guests that can extend beyond the limits of a physical machine. Each DVOS could have more computing power and could provide greater scalability and elasticity in its configuration as a consequence of resource and computing power aggregation. The set of resources (both physical and abstract) and the set of processes that constitute a DVOS can be scattered (and eventually replicated) in the nodes of a cluster. Several related processes (in the same DC) could be executed in different nodes using the same abstract resources as those offered by the DVOS. This feature simplifies application (or library) programming since standard APIs, such as operations on semaphores, message queues, mutexes, etc. can be used. On the other hand, the process location transparency is helpful for application administrators since it avoids dealing with IP addresses, ports, URLs, etc., simplifying applications deployment and management, and reducing costs and implementation times.

A DVS is suitable as infrastructure for this new trend in software development, like applications based on Microservices Architecture (MSA) or Service Oriented Architecture (SOA), because it is inherently distributed. Furthermore, thousands of legacy applications would benefit because they would not require modifications to take advantage of DVS features. Migration of legacy applications from on-premises servers to a Cloud execution environment requires changes in their design and coding. If a standard interface, such as POSIX is available in the Cloud, the migration task is simplified by reducing costs and time.

The main components of the DVS architecture are:

1) *Distributed Virtualization Kernel (DVK):* It is the core software layer that integrates the resources of the cluster, manages and limits the resources assigned to each DC. It provides interfaces for low-level protocols and

services, which can be used to build a VOS, such as InterProcess Communication (IPC), Group Communication System (GCS), synchronization, replication, locking, leader election, fault detection, mutual exclusion, performance parameter sensing, processes migration mechanism, and key-value services. The DVK provides interfaces to manage all DVS resources, such as nodes, DCs and processes. Process management allows the DVS administrator to assign processes to a DC and to allocate nodes for it. The node in which the process runs can be changed, as in case of a migration, or when the process was replaced by another one, such as a backup process. For communication purposes, location changes made by the replacement or migration of a process are hidden from the other processes within the DC.

2) *Distributed Virtualization Management System (DVMS)*: It is the software layer that allows the DVS administrator to both manage the resources of the cluster, providing a DC for each VOS or distributed application, and perform DVS monitoring.

3) *Container*: It is a host-OS abstraction which provides an isolated environment to run the components of a VOS or distributed application. A set of related Containers (one on each node) makes up a Distributed Container.

4) *Distributed Container (DC)*: It is a set of single Containers, each one being set up by the DVMS in the host-OS of each node. There is one DC per VOS or distributed application, and a DC can span from one to all nodes.

5) *Virtual Operating System (VOS)*: Although any kind of VOS can be developed or modified to meet DVS architecture requirements, a DVOS can obtain greater benefits because it is able to distribute its processes in several nodes. Each VOS (single or distributed) runs within a DC. The task of modifying an existing OS to turn it into a VOS is simplified because it does not need to deal with real hardware resources but with virtual ones. Moreover, a VOS needs to manage neither virtual memory nor CPU scheduling because it is done by the host-OS.

6) *VOS applications*: They are applications (single or distributed) running within the same DC, using VOS-provided services.

The resource allocation unit for a DOS is the node as a whole; but for a Distributed VOS (running within a DC) it is each single virtual resource provided by the host-OS on each node. This higher degree of granularity of the infrastructure unit allows a better use of resources and provides greater elasticity and efficiency.

The proposed DVS model combines and integrates Virtualization and DOS technologies to provide the benefits of both worlds, making it suitable to deliver provider-class Cloud services. A DVS prototype was developed to check the design and implementation correctness; and after several testing environments, the feasibility of the proposed model was proved.

This thesis is divided into seven chapters:

- *Chapter 1*: presents an introduction to the topic of the thesis and defines the main goals and contributions.
- *Chapter 2*: is a general description related technologies such as Virtualization, Distributed Operating Systems, Multiserver Operating Systems, Group Communication Systems, Process Migration, Containers and Replication.
- *Chapter 3*: presents the proposed architecture model, the abstractions used to describe it and its components.
- *Chapter 4*: details the DVS prototype used to evaluate the model and its components.
- *Chapter 5*: presents the different tests carried out on the prototype, their results and evaluation.
- *Chapter 6*: proposes future works as a research continuation of those topics related distributed virtualization.
- *Chapter 7*: is a summary of the contributions and benefits of the architecture model.

The main scientific publications that support this thesis are the following:

- Pablo Pessolani, Tony Cortes, Fernando G. Tinetti, Silvio Gonnet, Oscar Jara; "*A Fault-Tolerant Algorithm For Distributed Resource Allocation*"; IEEE Latin America Transactions (Volume: 15, Issue: 11, Nov. 2017); ISSN: 1548-0992.
- Pablo Pessolani, Tony Cortes, Fernando G. Tinetti, Silvio Gonnet; "*Sistema de Virtualización con Recursos Distribuidos*", WICC 2012 - Workshop de Investigadores en Ciencias de la Computación; Posadas, Abril 2012.

- Pablo Pessolani, Tony Cortes, Fernando G. Tinetti, Silvio Gonnet; "*Un mecanismo de IPC de microkernel embebido en el kernel de Linux*"; WICC 2013 - Workshop de Investigadores en Ciencias de la Computación; Paraná, Abril 2013.
- Pablo Pessolani, Tony Cortes, Fernando G. Tinetti, Silvio Gonnet, Diego Padula, Mariela Alemandi; "*A User-space Virtualization-aware Filesystem*"; 3er Congreso Nacional de Ingeniería Informática/Sistemas de Información – CONAIISI 2015; 19 al 20 de Noviembre de 2015, UTN – FRBA.
- Pablo Pessolani, Tony Cortes, Fernando G. Tinetti, Silvio Gonnet, "*Sistema de Virtualización Distribuido*"; WICC 2017 - Workshop de Investigadores en Ciencias de la Computación; Buenos Aires 27 y 28 de Abril de 2017.
- Pablo Pessolani, Tony Cortes, Fernando G. Tinetti, Silvio Gonnet; "*An IPC Software Layer for Building a Distributed Virtualization System*"; CACIC 2017 - Congreso Argentino de Ciencias de la Computación, La Plata, Argentina, 2017.
- Pablo Pessolani, Tony Cortes, Fernando G. Tinetti, Silvio Gonnet; "*An Architecture Model for a Distributed Virtualization System*"; Cloud Computing 2018; The Ninth International Conference on Cloud Computing, GRIDs, and Virtualization; ISSN: 2308-4294, ISBN: 978-1-61208-607-1. Barcelona, España.2018.

This thesis has been developed following the lines of research of three "*Research & Development Projects*" (PID, in Spanish) related to Operating Systems and Virtualization at the National Technological University (Argentina). It had the external support of PhD Toni Cortes (Polytechnic University of Catalonia, Spain), PhD Fernando G. Tinetti (National University of La Plata, Argentina) and PhD Silvio Gonnet (National Scientific and Technical Research Council, Argentina).

Citation: P. Pessolani. *An Architecture Model for a Distributed Virtualization System*. Journal of Computer Science & Technology, vol. 19, no. 2, pp. 183-185, 2019.
DOI: 10.24215/16666038.19.e17