Distributed Virtual Scenarios over Multi-host Linux Environments: Virtual Networks over LinuX (VNX)

David Fernández, Alejandro Cordero, Jorge Somavilla, Jorge Rodríguez
Departamento de Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Madrid, Spain
david@dit.upm.es

Aitor Corchero, Luis Tarrafeta
S21Sec
Pamplona, Spain
aolite@s21sec.com

Fermín Galán
Telefónica I+D
Madrid, Spain
fermin@tid.es

Introduction

Virtualization based testbeds widely used for the creation of network environments needed to test protocols and applications.

However, complexity of present networks and protocols arises the need of very complex network testbeds, made of tenths or hundreds of virtual machines.

Need for tools to support the design, deployment and management of large virtual network scenarios over clusters of servers

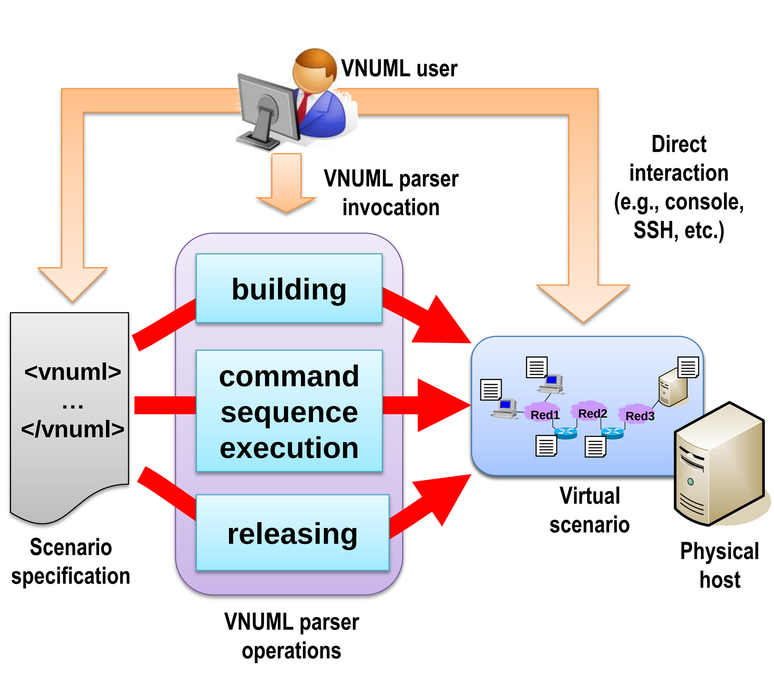
Apart from large scale initiatives, several tools are available for small group or individual use:

- VNUML (www.dit.upm.es/vnuml)
- Netkit (http://wiki.netkit.org)
- MLN (http://mln.sourceforge.net)
- Marionnet (http://mln.sourceforge.net/)
- CORE (http://cs.itd.nrl.navy.mil/work/core)

However, none of them support neither distributed deployment (except VNUML) nor the diversity of virtual machines operating systems needed for complex testbeds.

Previous work : VNUML

Virtual Networks User Mode Linux (VNUML) is a general purpose open-source scenario-based management tool designed to help building virtual network testbeds automatically.



VNUML Operation Workflow

VNUML made of two main components:

1. XML based scenario specification language
2. Language interpreter

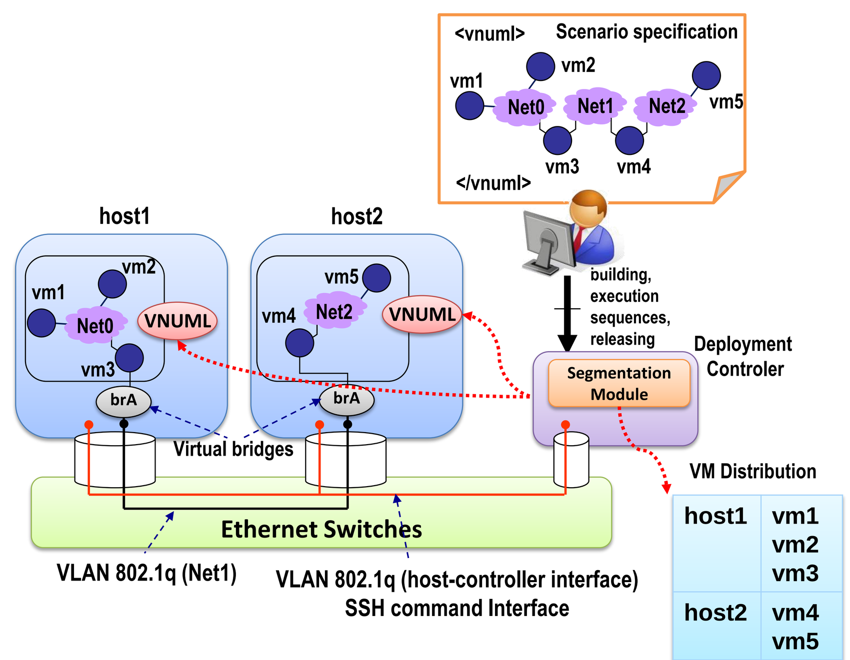
Three basic operations:

- **Scenario building**: virtual scenario specification is processed to create all the virtual machines and interconnect them by virtual networks following the scenario topology
- **Command execution**: users can directly interact with virtual machines or automatize the execution of commands with the help of VNUML
- **Scenario releasing**: virtual machines and networks are released

VNUML development started in 2003 and it has been used in several research projects and university networking laboratories. More details in http://www.dit.upm.es/vnuml.

EDIV: VNUML Distributed Deployment

In EDIV project, a partnership between Telefónica I+D and UPM, a wrapper application to VNUML was developed to allow the distributed deployment of virtual scenarios over clusters of servers.



Distributed Deployment Architecture

EDIV segments virtual scenarios into sub-scenarios deployed to the different servers, interconnecting them by means of VLANs.

EDIV includes basic segmentation algorithms (restrictions, round robin and weighted round robin) as well as an API to allow the user to develop new segmentation algorithms.

VNUML/EDIV Limitations

Limitations of VNUML and EDIV tools:

- Only Linux virtual machines (User Mode Linux limitation). Performance problems.
- Inability to manage virtual machines individually
- Autoconfiguration and command execution limited
- Distributed version (EDIV) limitations: manual network configuration for disperse clusters, lack of monitoring tools, etc

All these limitations led us to create:

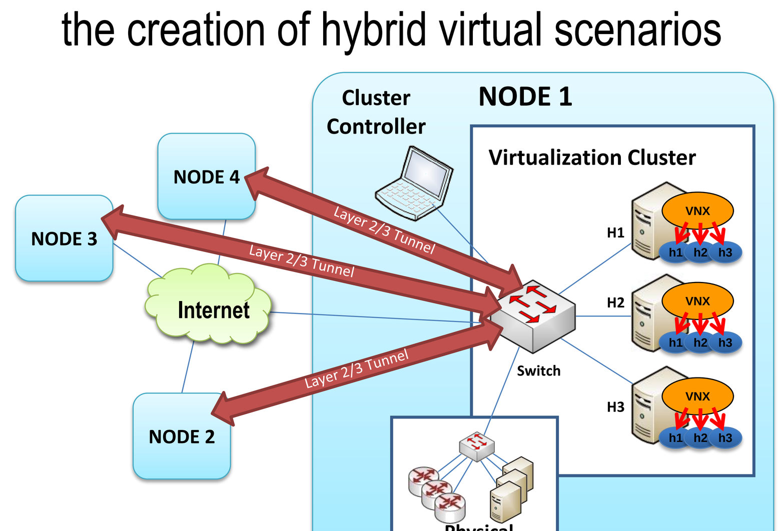
VNX
Virtual Networks over linuX

VNX Architecure

VNX overall goal is the creation of a tool to allow the deployment of large virtual network scenarios over a federated cluster environment made of disperse nodes interconnected by means of layer 2/3 tunnels over Internet.

Each node is composed of:

- Virtualization servers running different types of hypervisors
- Physical (non-virtual) equipment to allow the creation of hybrid virtual scenarios

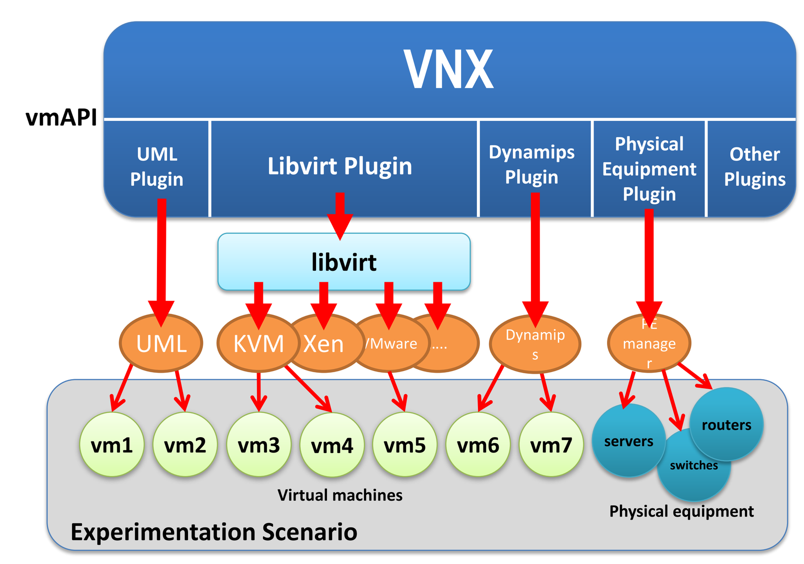


Distributed Cluster Architecture

VNX
Virtual Networks over linuX

VNUML redesign: VNX

VNX (Virtual Networks over LinuX) is a major rewrite of VNUML. A modular architecture based on a virtual machine control API has been defined to accommodate new virtualization platforms:



VNX Architecture

Plug-ins developed:

- **libvirt**, the Linux standard API for virtualization (libvirt.org), provides access to most of virtualization platforms supported in Linux (KVM, Xen, UML, etc)
- **Dynamips**, to support emulated CISCO routers
- **UML**, which includes old VNUML code

The internal API is a simplified version of libvirt API, with the addition of a primitive to execute commands inside virtual machines.

Primitive	Description
defineVM	Defines a new virtual machine
undefineVM	Undefines an existent virtual machine
startVM	Starts a virtual machine
shutdownVM	Shutdowns a virtual machine in an ordered way.
destroyVM	Kills (switches off) a virtual machine
saveVM	Hibernates a virtual machine (saves state to disk)
restoreVM	Restores a virtual machine previously hibernated
suspendVM	Suspends a virtual machine (saves state to memory)
resumeVM	Resumes a previously suspended virtual machine
rebootVM	Reboots a virtual machine (=shutdown+define+start)
resetVM	Resets a virtual machine (=destroy+define+start)
executeCMD	Executes a command inside the virtual machine

VNX Internal API

New autoconfiguration and command execution mechanism created. Based on the OVF approach: a dynamic cdrom offered to virtual machines with an XML file with autoconfiguration parameters and commands to execute.

VNX implementation

First version of VNX available:

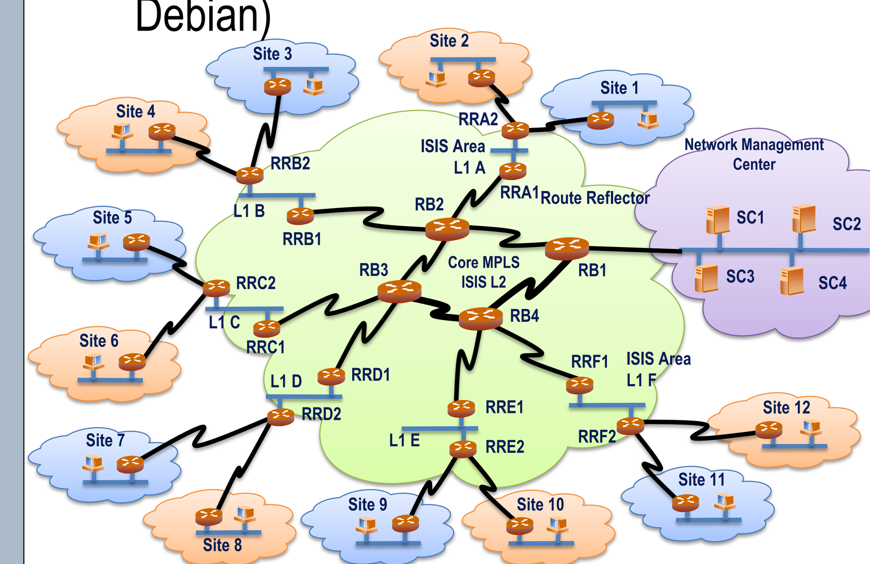
- **libvirt** support. Tested with Linux (Ubuntu, Fedora, CentOS), FreeBSD and Windows (XP and 7).
- **Dynamips and Olive** router emulation support
- Virtual machine individual management (start, stop, restart, reboot, suspend, etc)
- OVF-like autoconfiguration and command execution support
- Plug-in architecture to add extensions to VNX
- Distributed deployment support (EDIV)
- Library of root filesystems available: Ubuntu, Fedora, CentOS, FreeBSD, etc

VNX is mostly written in Perl (around 25000 lines of code); Windows autoconf daemon writen in C++. Around 40% of VNUML code reused with minor modifications.

Validation and Tests

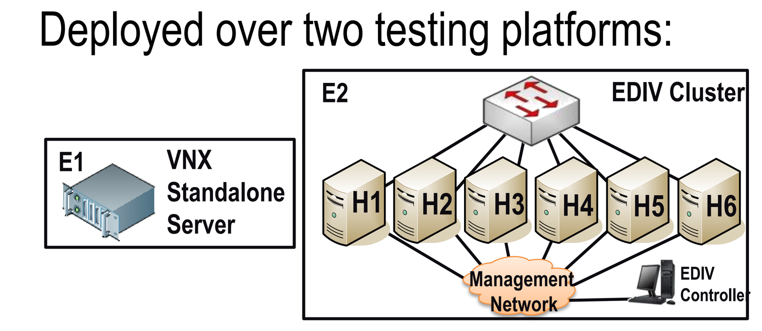
Networking laboratory for dynamic routing tests, resembling the topology of an ISP and involving 44 virtual devices as follows:

- 16 Cisco routers
- 6 Juniper routers
- 6 Linux/Quagga routers
- 12 end user computers
- 4 Servers (WinXP, FreeBSD, Ubuntu, Debian)



Example Validation Scenario

Deployed over two testing platforms:



Deployment scenarios

Conclusions and Future Work

First VNX version succesfully used in university networking laboratories. Development continues, mainly focused on improving distributed version.

Future work includes:

- Full OVF support
- Dynamic scenarios (adding or deleting machines and networks, machine migration
- Graphical user interface
- New virtual machine types (e.g. Android)
- Plug-in to control physical equipment
- Better network emulation capabilities
- Testbeds over the cloud: deploy virtual scenarios over cloud infrastructures

Acknowledgment

This research was partially supported by S21Sec and the Centre for the Development of Industrial Technology (CDTI) as part of the SEGUR@ project (https://www.cenitsegura.es/), within the CENIT program, with reference CENIT-2007 2004, as well as by the Spanish Ministry of Science and Innovation (National Plan for Research, Development and Innovation) grant TEC2008-06539, as part of ARCO Project.

Contact information

David Fernández
Dpto. Ingeniería de Sistemas Telemáticos
Universidad Politécnica de Madrid
Avda. Complutense, 30
28040 MADRID - SPAIN
E-mail: david@dit.upm.es
www: http://www.dit.upm.es

POLITÉCNICA

VNX
Virtual Networks over linuX
http://www.dit.upm.es/vnx