Creating Dynamic Virtual Networks for network isolation to support Cloud computing and virtualization in large computing centers

Davide Salomoni [1], Marco Caberletti [1]

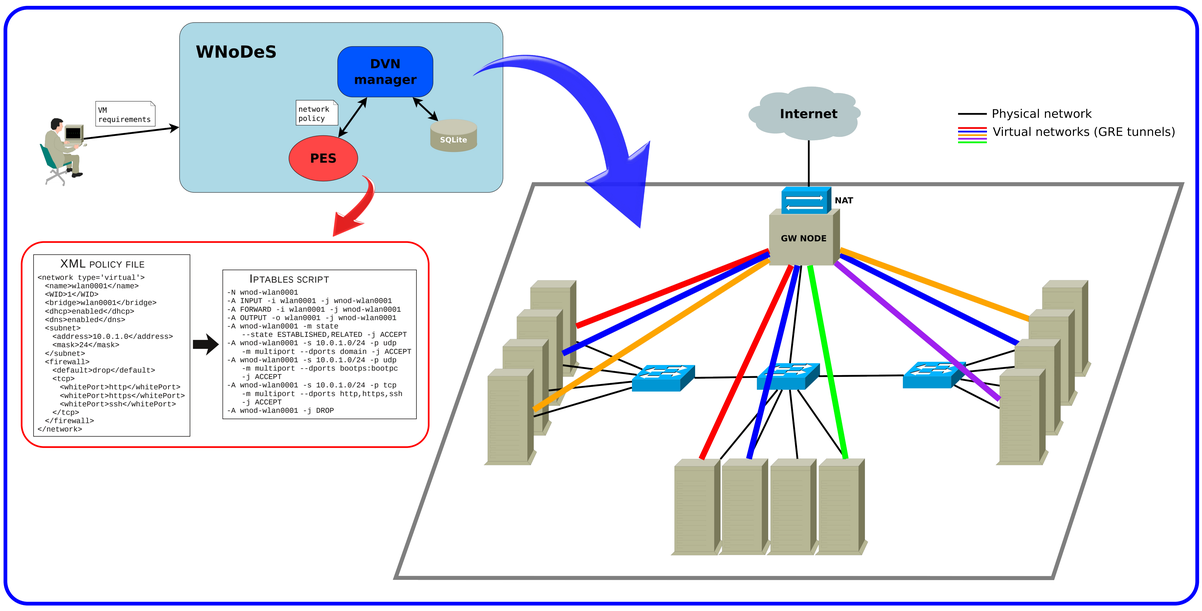
[1] INFN-CNAF, Bologna -- {davide.salomoni, marco.caberletti}@cnaf.infn.it

Goals

- Flexible and dynamic definition of virtual networks for Cloud Computing without changing the physical network configuration.
- Isolation of different tenants in a IaaS Cloud infrastructure.
- Using a tunnelling approach based on the Generic Routing Encapsulation (GRE) protocol for network virtualization.
- Possibility to have dynamic virtual networks spanning multiple sites.
- Integration into the WNoDeS virtualization framework (http://web.infn.it/wnodes).

Architecture

- GRE tunnels connect hosts in the same private virtual network.
- "Hub and Spoke" topology.
- The central node ("GW node"), also provides DHCP, firewall and NAT.
- A component, called DVN Manager, controls virtual network allocations in the datacenter.
- A component, called Policy Enforcement Service (PES), manages traffic control policies for each private virtual network.



Security

- Replication of bridges and tunnels to provide network isolation.
- Each private virtual network is in a separated broadcast domain.
- Each cloud customer has a different virtual network and subnet.
- With traffic control policies you can decide which protocol to enable and if allow outgoing communications.

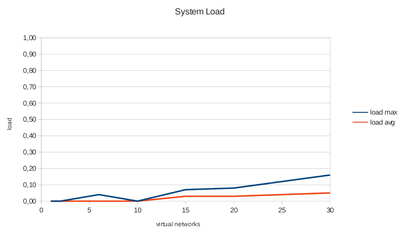
Policy Enforcement Service

- Manages different policies for each private virtual network.
- Policies are defined in an XML-based high level language.
- PES translates policies in suitable Iptables rules for physical hosts.
- PES generates rules to avoid MAC and IP spoofing on VMs.

Results

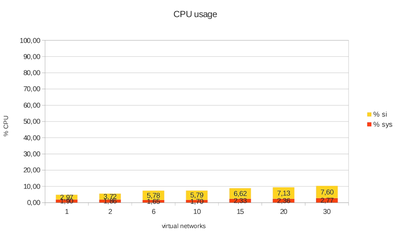
The following plots show results obtained after load tests conducted on a Dynamic Virtual Newtork system. These tests are performed with a single GW node, implemented with a standard Linux server.

Plot P1 reports system load as the number of active virtual networks increases; plot P2 reports total CPU consumption, separating time spent in kernel time (%sys) and time spent handling software interrupts (%si).



Plot P1. System Load

- A software-based GW node is naturally the bottleneck of the entire system.
- A single GW node can easily manage several tens of different active virtual networks.
- Load tests show that most of CPU time is spent handling software interrupts.
- For increased resiliency and scalability, the GW node can be easily replicated or replaced with a network device capable of handling GRE tunnels in hardware (e.g. a router).



Plot P2. CPU consumption

The Dynamic Virtual Network component will be released in a future WNoDeS upgrade as part of the EMI (European Middleware Initiative) stack, see http://www.eu-emi.eu/

More info: http://web.infn.it/wnodes