



# FCoE and Virtualization

T11/07-693v0, December 2007

Ed Bugnion



# Outline

## Context:

- Multiple presentations have used virtualization as a use case
  - None were done by Hypervisor vendors
  - Nuova Systems is also not a Hypervisor vendor
- But ... we need to ensure that the FCoE standard properly supports virtualized environments

## Outline

- Virtualization fundamentals; Ethernet and Storage
- New design option for VM-based storage model for FCoE
- The case for Mapped MAC addresses

# Fundamental attributes

A Virtual Machine is a unit of isolation

A Virtual Machine is hardware-independent

- All identities that are visible to the VM's Guest OS must be independent of the underlying hardware

A Virtual Machine is encapsulated at run-time

- The Hypervisor can externalize the entire state of the VM at any VM instruction boundary (including critical regions, interrupt handlers, ...)
- Applications:
  - Checkpoint/restore
  - Transparent migration (aka VMotion aka Live Migration)
  - VM Replay
  - Continuous H/A

# VM mobility and identity

VM Mobility based on encapsulation attribute

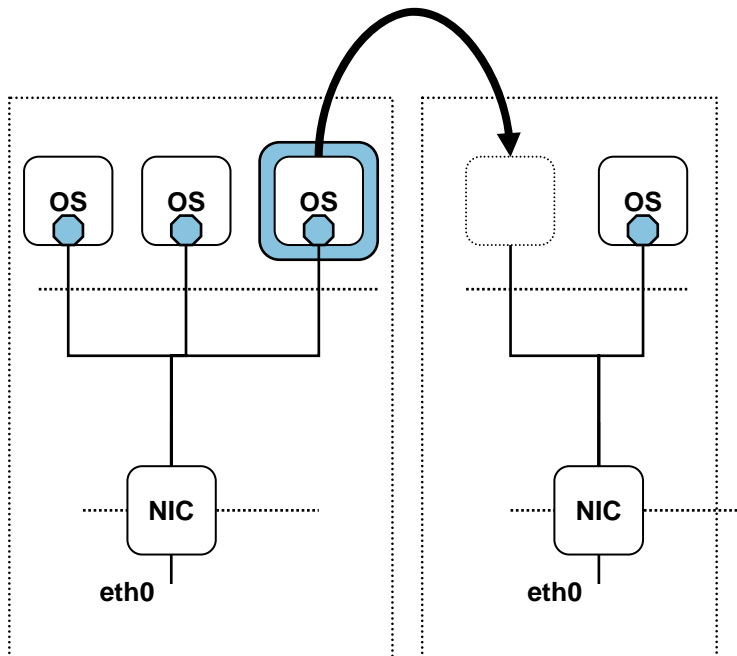
- Can occur on any VM instruction boundary
- The SW running within a VM has no notion of the event

To encapsulate a VM, the Hypervisor

- Stalls new I/O
- Waits for in-progress I/O to complete
- Posts the I/O completion events upon restore
- This is totally transparent to the VM

Implications for Ethernet

- Each VM has its own MAC address
- MAC address follows the VM (Hypervisor sends a gratuitous RARP to notify new location)
- Mobility limited to a L2 subnet



# Current Ethernet model and usage

Each VM can have multiple vNICs

- Each vNIC has its own identity (MAC address)
- Typically, the VM has one vNIC per VLAN accessed

Each Hypervisor instance has a built-in vSwitch or uses an OS-level bridge

Each physical NIC is put in promiscuous mode

- Need to receive frames destined to all VMs
- Not a issue at all on a modern Ethernet network

Burned-in MAC addresses are never used by a VM

- Managed by the Hypervisor or more typically a network-based VM management tool

# Quick clarification

Myth: In VMware the MAC addresses of a vNIC must have the VMware OUI

Counter-evidence – email sent to T11 reflector:

- > Using VMware's products, a VM is configured with one or multiple vNICs:
- >
- > \* Each vNIC has a default MAC address, prefixed with an IEEE OUI registered to VMware.
- >
- > \* The port group policy that the vNIC is attached to specifies whether the Guest is allowed to change the MAC address for that vNICs (by setting the flag "macChanges"; the default is "set" to allow this).
- > This behavior can be disallowed by overriding the default.
- >
- > \* When set, the Guest can change the actual MAC address at run-time with an arbitrarily chosen MAC address. In particular, the Guest can set the OUI bits to an arbitrary value.
- >
- > \* The vNIC does not have to be put in promiscuous mode for this operation to be allowed. Note that our default is not to allow a VM's vNIC to be put into a promiscuous mode, anyway.
- >
- > Key point: there is a difference in our existing ESX Server product between putting a VM's vNIC into promiscuous mode versus just pushing down a different MAC unicast Rx filter; the former is not allowed by default, the latter is. Both policies are part of ESX Server's port group layer 2 security policy. Please refer to page 10 of [http://www.vmware.com/files/pdf/virtual\\_networking\\_concepts.pdf](http://www.vmware.com/files/pdf/virtual_networking_concepts.pdf)
- >
- > Rich Brunner, Andrew Lambeth, Boon Seong Ang
- > VMware Engineering and Office of the CTO

# Scaling pressures

With currently shipping products (e.g., ESX 3.5)

- Support up to 128 VMs
- Each VM has up to 4 vNICs
- Each ESX cluster has up to 32 nodes (for VMotion)

Consider a switch with 32 ESX-facing ports

- Needs to manage up to 16K MAC addresses
- Each VM could be on any port
- VM-specific features must be applied to all ports

Multi-core will make that even denser

# 3 Major storage models

## Virtual Disks

- Mapped by the Hypervisor into the VM as SCSI Direct-Attached Disk
- Layout of the virtual disk is controlled by the Hypervisor
- Virtual disks can be stored on (clustered) filesystems, e.g., ext3, NTFS, VMFS, ...

## Raw Disks

- Mapped by the Hypervisor into the VM as SCSI Direct-Attached disks
- Maps 1:1 onto a FCP LUN, iSCSI LUN or a local device

## VM-based storage

- Hypervisor does not map a disk into the VM or manage the access
- Rather, VM accesses the resource directly from the network
- Primary usage model is through NAS
- Block storage is also available via iSCSI (**but NOT for FC**)

Each model has distinct and valuable use cases and value proposition

# Storage and identity (pre-FCoE)

## In the Virtual Disk model:

- The VM has no network identity (i.e., WWN)
- The Hypervisor has a single identity used to mount the shared file system. That identity is associated with the Hypervisor instance

## In the Raw Disk model

- The Hypervisor can use either a single shared identity or a different identity for each VM.
- Use of FC NPIV is optional and its use is not visible to the VM

## In the VM-based Storage model

- Each VM has a distinct network identity
- Example: iSCSI uses the MAC of the vNIC and the IP address associated with that vNIC. The Hypervisor is only involved at the Ethernet layer (L2)
- Currently not supported with FC

# Requirements for FCoE

The standard should take into consideration all three storage models

- Virtual Disk model
- Raw disk model
  - with or without a distinct identity per VM
- VM-based storage model

These models are complementary, not mutually exclusive, and should not be precluded by the standard

# Conclusion (1)

Virtualization has increased the number of MAC addresses in the network and will continue to do so irrespective of FCoE

All software-visible identities must be unique for each VM

Isolation and encapsulation are fundamental attributes

There are multiple storage use cases



# Storage models and FCoE

Unlike iSCSI, it won't "just work"



# Overview

Virtual disk – FCP terminated in the Hypervisor

- Either Hardware HBA or Software stack

Raw disk – FCP terminated in the Hypervisor

- To support multiple identities, Can use either
  - One MAC address + multiple FDISC
  - Multiple MAC address and multiple FLOGI
  - Equivalent from a Hypervisor perspective

VM-based storage

- Today, works with NAS and iSCSI (Ethernet-based)
- Today, does not work with FC

# VM-based storage with FCoE

## Hardware assumption

- Hardware NIC supports lossless Ethernet

## Hypervisor enhancements

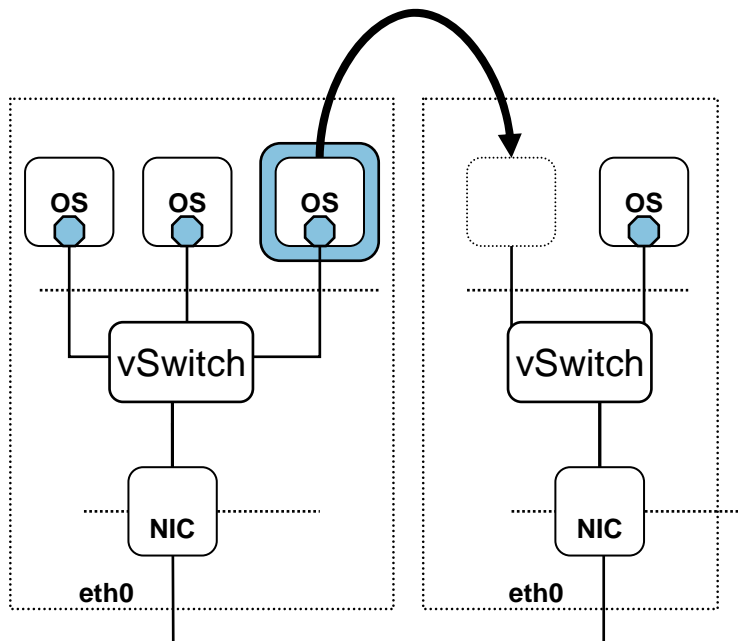
- Support Lossless Ethernet vNIC
- Support Lossless Ethernet vSwitch
- Flap the link to the vNIC as part of the mobility event

## FCoE software stack requirements

- Re-initializes FCoE stack upon link flap
- Fails and retry all outstanding I/O to the SCSI and multipathing layer
- These requirements are not different than in a physical configuration

## This is plausible

- Hypervisor is only involved at Layer 2 (lossless Ethernet)
- Uses common FCoE software stack
- Same model as iSCSI



## Conclusions (2)

For the Hypervisor-based models (virtual disks and raw disks), the Hypervisor terminates FCP sessions

- Two design options to support multiple identities

Plausible future solution for the VM-based model

- Runs FCoE software stack in the VM
- Hypervisor does not to be aware of FCoE
- Hypervisor only needs to support lossless Ethernet (L2) with the appropriate flapping mechanism



# FCoE and Virtualization In Support of Mapped MAC addresses

# FCoE in a VM if you're from Missouri

You too can try this at home:

- Install widely used virtualization software (off-the-shelf)
- Create a Linux VM (default settings are ok)
- Download FCoE stack from [www.open-fcoe.org](http://www.open-fcoe.org)
- Trivial patch – stack checks for TX/RX pause capabilities of the NIC
- Bridge the VM to the LAN
- Connect it to a target (via FCF or directly FCoE target)

Mount your favorite filesystem over FCoE !

It “works” but with obvious fundamental caveats:

- This setup does not meet the prerequisite of a lossless Ethernet environment at the vNIC and the bridge

[www.open-fcoe.org](http://www.open-fcoe.org) uses Mapped addresses !

# MAC address uniqueness

Let's define a VM management domain as a set of Hypervisors that enslaved to a network-based centralized management tool

Today, MAC address uniqueness is guaranteed within a VM management domain only

- Applies to all of the VMs that run within that management domain
- Can be quite large

Best practice: VM management domains should not share L2 domains (VLAN)

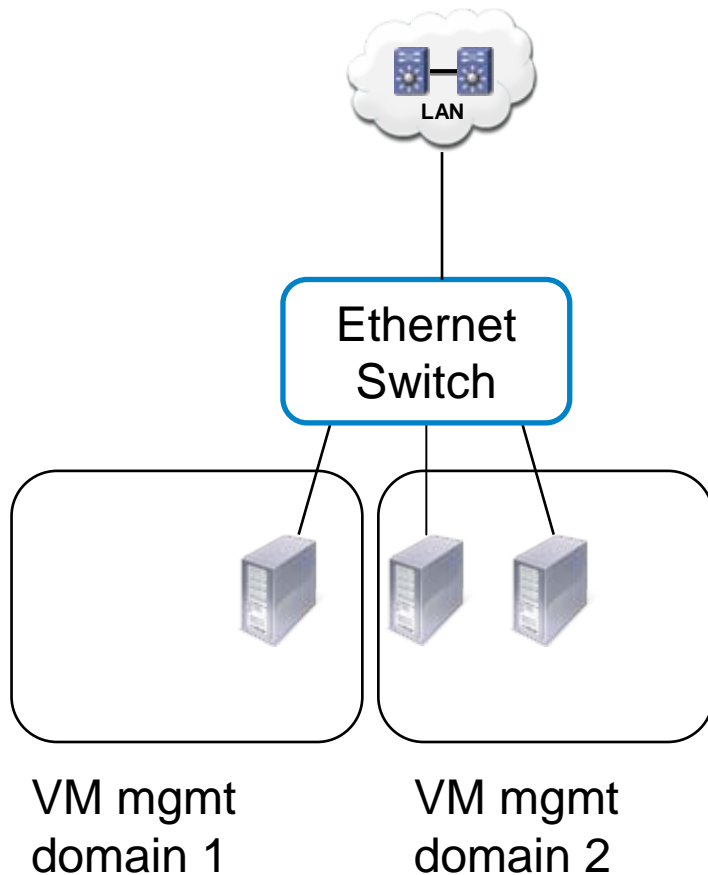
Uniqueness is not guaranteed

# Non-unique MAC address scenarios

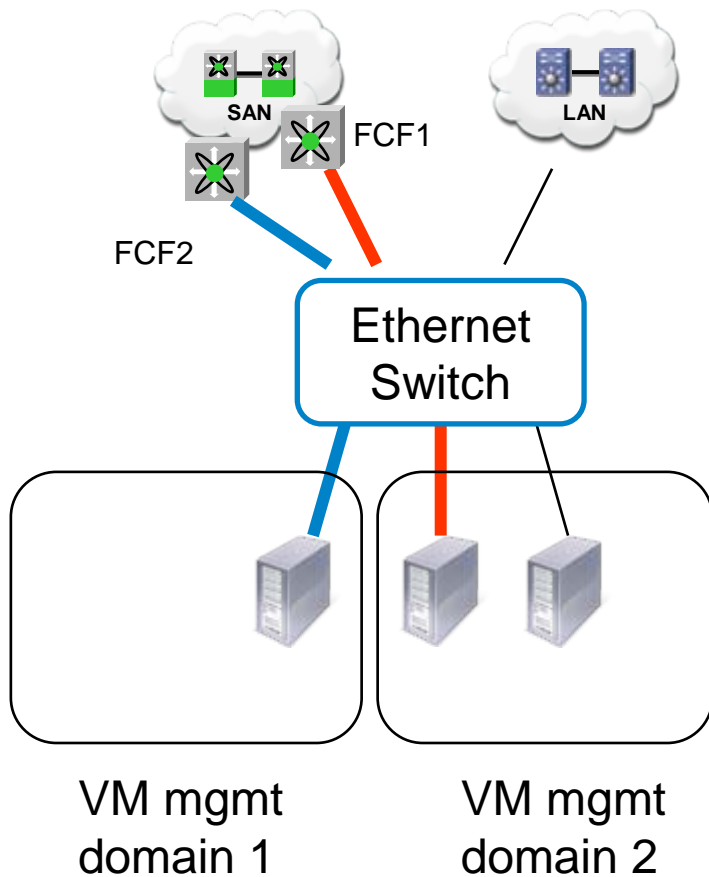
## Situation today

Two VM management domains can share the same L2 switch

- Creates possibility of duplicate MAC addresses
- MAC address conflicts create Ethernet black holes



# Non-unique MAC address scenarios



## Scenario with multiple FCF

- Red and Blue servers have the same MAC addresses
- Each FCF associates a different FCID to each server
- FCF1 and FCF2 don't know this
- Leads to black holes whenever there are concurrent outstanding I/Os from each server
  - Black holes lead to frame loss and SCSI timeouts
  - Can be very hard to diagnose

**Mapped MAC addresses solve the problem**

# Myth: Ethernet relies on burned-in MAC addresses

## Reality – that address can be changed or ignored entirely

- Can be set via the GUI (Windows) or ifconfig (Linux)

## Keep server identity while swapping hardware

- Example – OS and application licensing/activation
- Example - DHCP

## Use a floating MAC address for clustering, bonding, or trunking purposes

## Virtualization

- Relies on MAC addresses that are managed and handed off by a network-based entity

## Ethernet multicasting destination addresses

# Example of Network-mapped MAC addresses

## RFC 1112 (1989) – Host Extensions for IP Multicasting

- An IPv4 Multicast group is specified by its Class D IP address (224.0.0.0 : 239.255.255.255) (28 LSB)
- Locally, IP multicast group addresses are mapped onto Ethernet multicast addresses
  - 01:00:05 || xx.yy.zz (23 LSB)
  - 01:00:05 is an IEEE reserved OUI
- Algorithm: map the 23 LSB of the IP group to form the MAC address

This is an example of network-based MAC address! The Ethernet address is algorithmically derived from the ULP address

# Conclusion (3)

Despite myths, Mapped addresses work with virtualization (see the Missouri test)

Ethernet forwarding is based on learning, not on any MAC address format or scheme

Mapped addresses resolve the lack of uniqueness of virtualization MAC addresses

IP Multicast is an example of mapped MAC addresses based on an ULP

Existing (and successful) products are based on network-driven MAC addresses



**Thank You**

