Automated Ethernet Virtual Bridging

Renato Recio and Omar Cardona

Abstract— **The traditional Data Center (DC) compute model, especially in the x86 space, has consisted of lightly utilized servers running a bare metal OS or a Hypervisor with a small number of Virtual Machines. In this traditional model, servers attach to the network lower bandwidth links, such as 1 Gbps Ethernet and 2 or 4 Gbps Fibre Channel. The physical compute model suffers from two major issues: High capital expenses due to under utilized servers and multiple fabrics; and High operational expenses due to manual administration of many management tools.**

We see the industry moving to a Dynamic Infrastructure Networking model that has highly utilized servers running many VMs per server and uses high bandwidth links to communicate with virtual storage and virtual networks. This paper will describe the problems that must be overcome to provide an automated, optimized virtual system solution. It will describe optimization approaches for directly sharing Ethernet adapters across multiple Virtual Machines (VMs). It will also describe options for automating orchestrating the migration of the port profile associated with a VM, when that VM is migrated across physical systems. It will discuss options for providing local MAC Address uniqueness within a management domain. Options for an adapter Virtual Ethernet Bridge will also be covered. These mechanisms can be used to lower capital & operational expenses by increasing system utilization & automating VM management.

Index Terms—**Virtual Servers, Virtual Machines (VMs), Single-Root IO Virtualization (SR-IOV), Virtual Ethernet Bridging (VEB), Automated Port Profile Migration, Ethernet Virtual Bridging (EVB)**

I. Introduction to the Problem Space

With the advent of PCIe adapters supporting multi-queue, multi-function or Single-Root IO Virtualization[1] (SR-IOV), enterprise class methods for directly sharing IO[2] are becoming available for x86, high volume servers. These virtualization approaches enable a Virtual Machine's device driver to bypass the Hypervisor and thereby directly share a single PCIe adapter across multiple Virtual Machines (VMs). The PCI Special Interest Group (SIG) only standardized the north side IO Virtualization (IOV) mechanisms. The southside mechanisms, such as Ethernet bridging, were outside the PCI SIG scope and weren't standardized.

R. J. Recio (recio us;ibm;com) and O. Cardona (ocardona us;ibm;com) are with IBM, 11400 Burnet Rd., Austin, TX 78758 USA

An Ethernet adapter that is directly shared by multiple VMs has two options for where the virtual Ethernet bridging function[3] is performed: at the server or in the external network. Enterprise use cases exist for both of these options. An example use case for performing VM-to-VM virtual switching at the server, either in the Hypervisor or adapter, is High Performance virtualized Computing. This use case provides very low VM-VM communication overheads[4,5,6,7] for VMs residing in the same physical server, which becomes more and more important as the number of cores per server, and hence the number of VMs per server, increases. An example for performing VM-to-VM virtual switching in the network is multi-tier enterprise environments that want to take advantage of state of the art network access and QoS controls. This paper will describe a server virtual switching approach that satisfies both of these use cases.

An additional problem that must be addressed is the complexity associated with provisioning and orchestrating a VM's network identity. In today's VM models, the MAC Address is used to associate a VM to a specific port profile. Without orchestration between the VM manager and the fabric manager, there is no way for the fabric to distinguish between a re-incarnated[A] and a migrated[B] MAC Address. As a result the MAC Address alone provides insufficient information for the fabric to determine which port profile to use, because a re-incarnated MAC Address may have a different port profile than its use on a previous VM. An automated, orchestration solution to this problem is needed. Also, today's VM management uses MAC Addresses that are local within a VM management domain. A solution is called for that allows a MAC Address to have a wider scope, including spanning multiple Data Centers using layer-2 overlay technologies, such as MPLS and VPLS.

The next section will describe a proposal for selecting where Ethernet Virtual Bridging is performed and options for automating port profile migration. This will be followed by a description of an adapter provided Virtual Ethernet Bridge.

[A] A re-incarnated MAC Address is one that was previously in use by a recently destroyed VM and is now in use by a different VM, which may require a completely different external network port profile.

[B] A migrated MAC Address is one that is associated with a VM that has been migrated across two physical servers in the fabric and retains the same port profile association after the VM migration.

II. VM Automation and Optimization

Today's server virtualization infrastructure (e.g. a Hypervisor) associates a Virtual Ethernet Bridge (VEB) port profile to each Ethernet MAC Address used by a VM to access the network through a VEB's port[8]. The information found in a VEB's port profile, such as: the types of frames allowed on the port (e.g. all frames, Only Virtual LAN tagged frames or untagged frames), the Virtual LAN identifiers that are allowed to be used on egress, and rate limiting attributes (e.g. port or access control based rate limiting). If the VM is migrated from one physical server to another, the VEB's port profile migrates with it[9]. In other words, today's server virtualization infrastructure provides automated port profile migration of the server's VEB port(s) that are associated with a VM.

Today's automated port profile migration approaches are fine, for environments where the server's virtualization infrastructure provides sufficient controls. An example of such an environment is a high performance cluster, constructed with increasingly more cores per node, which uses a layer-2 network which is isolated from external networks through firewalls and security appliances.

Today there is a gap between the access and Quality of Service (QoS) controls supported in external layer 2 switches and server virtualization infrastructure. That is, external layer 2 switches have more advanced controls compared to server VEB implementations. Server virtualization infrastructure is continually adding these controls, but we expect the gap will continue for several reasons, including: deployment of directly shared IO adapters, which cannot provide all the access controls available on external switches due to PCIe adapter power/cost constraints; and many controls implemented in external switches are proprietary.

Some environments prefer the more advanced controls provided by external network switches. An example of such an environment is a multi-tier data center that has several types of applications, each with differing advanced network controls, running over the same layer-2 network. In this type of environment the network administrator often prefers the use of advanced access controls available in external switches. The following sections will explore alternatives for providing these advanced controls and for maintaining port profile associations when a VM migrates across physical servers.

A. Advanced Port Profiles options for VMs

There are two options for providing advanced port profiles used in external switches on the ports used for VMs communication.

1) External and Virtual Ethernet Bridge Homogeneity

Under this option the VEB used by the server's virtualization infrastructure has all the same access and QoS controls as external bridges, because it is provided by the same vendor as the external bridge. An example of this approach is the Nexus 1000v[10]. Similar to a Hypervisor vendor provided VEB, the network vendor provided VEB can be discovered and monitored by external network tools. However, it has the additional access controls that are only available from the network vendor[11]. This allows the network vendor to use the same Network Change and Configuration Management (NCCM) tools for configuring the VEB's port profiles, as are used for the network vendor's external switches.

Though this option is feasible for a Hypervisor based VEB, it is not for a VEB used by a directly shared adapter, due to the adapter has silicon constraints stated earlier.

Homogeneity between port profiles in the external network and the VEB also faces another issue. Each switch vendor today has vendor unique port profiles. Most data centers use Ethernet switches from multiple vendors, especially at the access layer. To reap the rewards of automated port profile migration, network administrators have two choices, either: partition the fabric, and VM migration, into sections that have homogeneity between the external network and VEB provider; or use the lowest common denominator set of port profile attributes, which in essence reverts back to Hypervisor vendor provided access controls.

2) Mechanism for selecting Internal vs External VEB

As covered earlier, use cases exist for whether the server's virtualization infrastructure is configured to use internal or external Virtual Ethernet Bridging. Option 2 simply proposes a standard based mechanism for configuring whether the server's virtualization infrastructure will use internal or external Virtual Ethernet Bridging. Obviously, proprietary mechanisms can also be used to make this selection.

We propose a new Ethernet Virtual Bridging (EVB) Data Center Bridging eXchange (DCBX) protocol Type, Length, Value (TLV) be used to configure whether the server's virtualization infrastructure will be configured to use internal or external Virtual Ethernet Bridging.

In our view network administrators will want a given physical server to have homogeneity in terms of where Ethernet bridging and associated controls (e.g. access, QoS) will be performed. For this reason, we propose that all ports be homogenously configured to use either internal or external

bridging.

If use cases warrant heterogeneity with respect to where bridging is performed, an alternative to the proposed single bit would to enable internal or external bridging on a per port basis. This would be more complex, because it would require setting internal bridging on or off for each VM MAC Address.

Following With is the proposed EVB DCBX TLV structure:

← 32 bit field →

I	Maximum VEB ports
	Active VEB ports

Table 1

If I is "1"b, then the server's virtualization infrastructure uses internal Ethernet Virtual Bridging (EVB). If I is "0"b, then it uses external EVB [e.g. with either a Virtual NIC Tag (VNTag)[12] or Virtual Ethernet Port Aggregator approach[13]].

The "Maximum VEB Ports" field defines the number of VEB ports the server's virtualization infrastructure is allowed to have active at any given point in time.

The "Active VEB Ports" field defines the number of VEB ports the server's virtualization infrastructure currently has active.

As stated above, though the EVB DCBX TLV could be implemented on a per port basis, we expect all VEB ports to have common control requirements. That is, if one of the VEB's ports requires external access controls, all require external access controls.

If the server is configured for external virtual Ethernet bridging, all VM to VM communications within the same physical server are forwarded to the external switch. This takes advantage of all the external's switch's advanced network access controls without being tied to a single vendor's switching infrastructure. By using the proposed EVB DCBX TLV to select external switching for VM-VM communication, the server's virtualization infrastructure becomes network vendor agnostic. The server's virtualization infrastructure is neither tied to a single network vendor's VEB implementation, nor forced to implement, test, support and manage a proprietary VEB for each network vendor.

Section III describes a proposal for the case where the server is configured for internal switching for VM-VM communications. We are working with adapter providers on an additional DCBX TLV for the internal switch's attributes.

B. Automating External Port Profile Migration

When a VM migrates from one physical server to another, the EVB DCBX TLV doesn't migrate the Port Profile(s) associated with that VM. Additional mechanisms are needed to automatically migrate the Port Profile(s) associated with the VM's MAC Address(es) when the VM and the MAC Address(es) it uses migrate. In this paper we will call these mechanisms Automated Migration of a Port Profile (AMPP).

We propose a mechanism that can be used by the server's virtualization infrastructure to associate the MAC Address used by a VM to a specific port profile on an external switch. Figure 6 below depicts the mechanism's ladder diagram.

The mechanism in figure 6 is used between the server's virtualization infrastructure (e.g. Hypervisor) and a network resident Port Profile Manager. The server's virtualization infrastructure passes the Port Profile Manager the VM MAC Address that is to be associated with an existing Port Profile Identifier.

There are two options for how to implement the handshake shown in figure 6. One option is to implement the Associate request and, if the nearest neighbor switch recognizes the Port Profile Identifier, it responds with a success. If it doesn't, it returns an unsuccessful response.

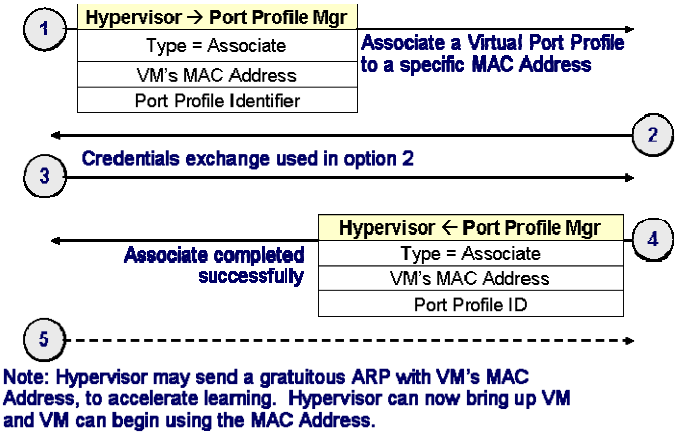

Figure 6

The second option is for the Port Profile Manager to use a challenge handshake to assure the server's virtualization infrastructure has the authority to ask for the specific MAC Address to Port Profile Identifier association. As shown in Figure 6, if the challenge completes successfully, the Port Profile Manager returns an association completed successfully message that includes the MAC Address and associated Port Profile. By consuming this message, the nearest neighbor external switch can use the Port Profile Identifier to look up

the Port Profile that is to be associated with the MAC Address. The nearest neighbor switch can then automatically associate the MAC Address to the Port Profile defined by the Port Profile Identifier. If the challenge doesn't complete successfully, the Port Profile Manager returns an unsuccessful completion message. Note, this case is not shown in Figure 6.

Figure 6 states the server's virtualization infrastructure may issue a gratuitous ARP prior to allowing the VM to use the MAC Address. The purpose of this gratuitous ARP is to start fabric learning (or relearning for a migrated VM) before the VM is brought up. Once the VM is brought up, it too may issue a gratuitous ARP.

Though the AMPP mechanism described in this section is intended for external switches, it can also be used by an internal Virtual Ethernet Bridge (VEB), such as an adapter's VEB implementation. In this case the adapter supports the association of it's internal VEB's port profiles to a VM MAC Address.

The proposed AMPP mechanism can be provided through a new protocol. However, given the AMPP is really a port based network access control mechanism, we propose extending IEEE 802.1x[14] and IETF protocols (EAP[15,16,17]) to carry a new Associate and De-associate Type and IETF's RADIUS[18,19,20] protocol to essentially authorize the association mapping. These new Types would consist of the following:

	1	2	3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	7 8 9 0 1 2 3 4	5 6 7 8 9 0 1 2
Type	Length		
VM's MAC Address			
Port Profile Identifier			

Table 2

The Associated and De-associate Types are each a new (unused) value. The length equals 10 octets. The VM's MAC Address is the MAC Address the server's virtualization infrastructure is asking to be associated with the Port Profile Identifier. The Port Profile Identifier is proposed to be 16 bits.

Figure 7 below depicts a ladder diagram for a successful AMPP completion based on the proposed extensions IEEE and IETF extensions.

The EAP Association Response is used to carry the Associate message that asks for a VM's MAC Address to be associated with a specific Port Profile Identifier. The EAP Association Success is used to communicate successful completion of the MAC Address association with the Port Profile Identifier. The Authenticator Switch consumes the EAP Association Success message, which associates the MAC Address to the Port Profile defined by the Port Profile Identifier. After the EAP Association Success has been consumed, the Authenticator Switch will apply the Port Profile controls to all Ethernet Frames that use the associated VM MAC Address as the Source MAC Address.

To cover error and shut-down scenarios, the semantics for the association can use a "time to live" attribute, which can be either specified as a fixed or administrator variable semantic. If this semantic is used, the Hypervisor would need to perform a re-association prior to expiration of the "time to live" value.

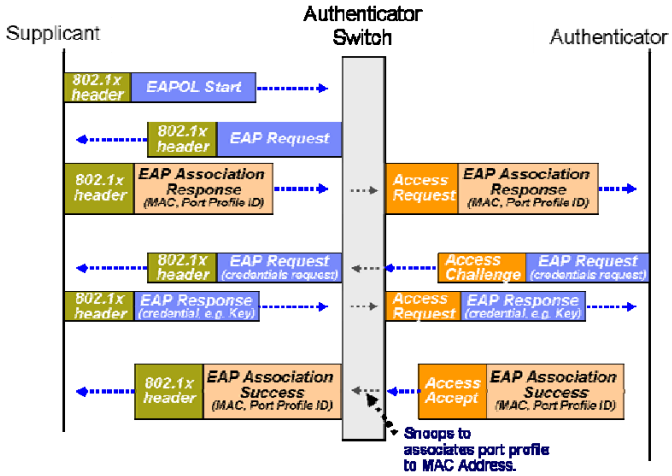

Figure 7

Within a data center the number of different Port Profiles is expected to be relatively small. One intended usage model would have a Port Profile Identifier for each VM application type, such as: an e-mail VM, a web serving VM, a print server VM, an application serving VM, a database VM and so on.

When a VM migrates from one physical server to another, the AMPP mechanism can be used to de-associate the MAC Address from the Port Profile Identifier at the VM's previous location. Similarly, the AMPP mechanism can be used to associate the MAC Address with the Port Profile Identifier at the new VM's location.

The AMPP mechanism described above depends on an out-of-band mechanism for communicating the Port Profile to the server's virtualization infrastructure. That is, a proprietary protocol is used between the network's Port Profile Manager and the tool used to manage each server's virtualization infrastructure. The next section describes a mechanism for standardizing this protocol.

C. Port Profile Management Protocol

This protocol is used to manage the Port Profile Identifiers associated with the various Port Profiles used within a data center layer-2 fabric. These are the Port Profile Identifier used by the AMPP mechanism described earlier.

The Port Profile Management (PPM) Protocol is used to create, change and destroy Port Profile Identifiers. In this paper, the tool used to maintain the Port Profile database is referred to as the Port Profile Resource Manager (PPRM). One use case model would be to have the tool used to manage the server's virtualization infrastructure use the PPM protocol to communicate with the PPRM. Figure 8 below depicts this usage model.

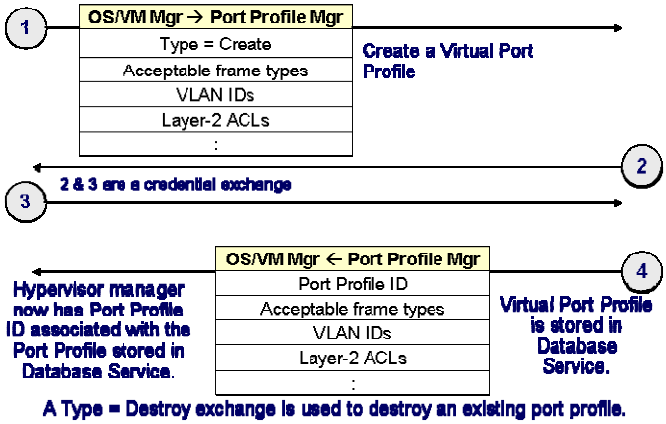

Figure 8

In step 1 of figure 8 above, the tool used to manage the server's virtualization infrastructure uses the PPM protocol to ask that a new Port Profile be created. This message includes a set of standard fields for the new port profile, such as: the types of frames allowed on the port (e.g. all frames, Only Virtual LAN tagged frames or untagged frames), the Virtual LAN identifiers that are allowed to be used on egress, and rate limiting attributes (e.g. port or access control based rate limiting). The message also includes fields that are used to carry vendor unique or experimental port attributes.

A credentials exchange (steps 2 and 3) can be used to assure to assure the PPM creation request is emanated from an authorized source.

The last step of figure 8 depicts a successful creation handshake. After this step is completed, the Port Profile Identifier can be used by a server's virtualization infrastructure in an AMPP handshake.

Given the PMM protocol is associated with port based network access controls, we propose extending IEEE 802.1x and IETF protocols (EAP) to carry three new Types: Create, Change and Destroy.

Table 3 describes the proposed Create and Change Types:

	1	2	3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	7 8 9 0 1 2 3 4	5 6 7 8 9 0 1 2
Type	Length	Standard Length	Exp. Length
Start of TBD Standard Port Profile Attributes			
:			
End of TBD Standard Port Profile Attributes			
Experimental Vendor's OUI (24 bit OUI with 8 bit extension)			
Start of TBD Experimental Port Profile Attributes			
:			
End of TBD Experimental Port Profile Attributes			

Table 3

The Type is two new (unused) values, one for the Create and one for the Change. The length equals is greater than 4. The "TBD Standard Port Profile Attributes" is a list of Port Profile Attributes the Ethernet Virtual Bridging Ad-Hoc group can agree to define in a standard format. Following is an initial starting proposal for that list:

- Acceptable Frame Types: Only VLAN Tagged,' Untagged, and All frames.
- Port VLAN ID
- Egress VLAN IDs
- Priority based Flow Control setting (flow control enabled/disabled)
- MAC Addresses to filter
- Port-based rate limiting value (% of link rate)
- Bandwidth allocation per port

The "TBD Experimental Port Profile Attributes" is a list of vendor unique or experimental Port Profile Attributes.

The Destroy Type would simply be a 2 octet type:

	1	2	3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	7 8 9 0 1 2 3 4	5 6 7 8 9 0 1 2
Type	Length		

Table 4

Though this section has described the direction of the PPM protocol to be from the server management tool to the PPRM, another alternative would be to have the direction be from the NCCM tool to the PPRM.

The PPM protocol standardizes the creation, change and destruction of Port Profiles and their associated Identifiers. If we stopped here, then a proprietary control plane mechanism would be needed to distribute the Port Profiles and the associated Port Profile Identifiers through out the fabric. This is certainly an option, but not one that enables multi-vendor network topologies, which is a requirement for many data

centers. For example, many data centers use a different access switch vendor than the vendor that supplies aggregation and core switches. The next section proposes a mechanism for disseminating the Port Profiles and the associated Port Profile Identifiers through out the fabric.

D. Port Profile Identifier Dissemination Protocol

This section proposes a protocol for standardizing the dissemination of a Port Profile and its associated Port Profile identifier into fabric. The proposal is to define a new Multicast, layer-2 service for use by the Port Profile Identifier Dissemination protocol. Upon reception of a Port Profile Identifier Dissemination multicast message, each layer-2 (physical or virtual) switch in the fabric would store the Port Profile Identifier and the standard Port Profile content. Additionally, for fabrics using a homogeneous vendor, the experimental field would also be used to disseminate vendor unique experimental Port Profile Attributes.

The multicast message would consist of the following:

	1	2	3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	7 8 9 0 1 2 3 4	5 6 7 8 9 0 1 2
Port Profile Identifier		Standard Length	Exp. Length
Start of TBD Standard Port Profile Attributes			
:			
End of TBD Standard Port Profile Attributes			
Experimental Vendor's OUI (24 bit OUI with 8 bit extension)			
Start of TBD Experimental Port Profile Attributes			
:			
End of TBD Experimental Port Profile Attributes			

Table 5

The mechanisms described above enable the automation of port profile migration during VM migration. However, they do not provide a way of guaranteeing the MAC Addresses used by the server's virtualization infrastructure are unique within a data center. To provide unique VM MAC Addresses for use by multiple Hypervisor vendor(s) throughout the data center, an additional mechanism is needed.

E. MAC Address Provider Service Protocol

Within many customer environments, there are a range of usage models that require multiple MAC addresses to be assigned to a given port. For example, vendor-assigned MAC addressing (e.g. local MAC Addresses) faces collision problems as the layer-2 network size increases and the number of virtualization platforms increases. This section proposes a MAC Address Provider Service (MAPS) per customer-defined administrative domain, which is used assign or remove a set of MAC addresses to a specific device, such as a server virtualization manager. Under this approach the assignment has a "time to live" lease period, which can be either specified as a fixed or administrator variable semantic. If the server virtualization manager doesn't renew the lease within the pre-defined time, the MAC Addresses are no longer leased by that virtualization manager and may be given to different device by the MAC Address Provider Service.

Similar to the AMPP protocol, the MAPS protocol can be based on IEEE 802.1x, EAPOL and IETF RADIUS, EAP. However, doing so would constrain the MAC Address Provider Service to a single layer-2 domain.

An alternative solution would be to use a DHCP model for the MAPS protocol. Figure 9 below depicts such a model:

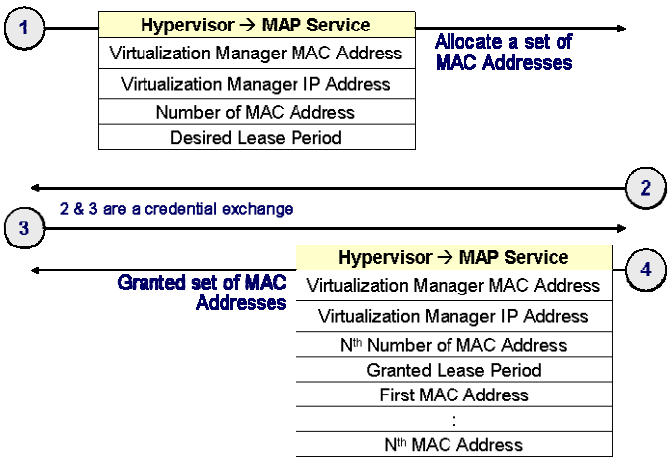

Figure 9

As shown in step 1 of figure 9 above, the server virtualization manager issues MAC Address reservation message to the MAP Service. The message is encapsulated in a UDP datagram, which the server virtualization manager may retransmit if no reply is received within a timeout period. The server virtualization manager asks the MAP Service for a Number of MAC Addresses for use within a given lease period. The server virtualization manager may periodically renew the lease or let it expire.

A credentials exchange (steps 2 and 3) can be used to assure to assure the MAPS allocation advertisement emanates from an authorized source.

In step 4, the MAP Service Success message includes the list of granted MAC Addresses. The MAP Service grants the server virtualization manager the MAC Addresses for the granted lease period included in the MAPS response.

The leased MAC Addresses are assured to be unique within the domain of the MAP Service.

III. Server Ethernet Virtual Bridging Optimization

The PCI SIG defined several enhancements, such as the SR-IOV specification[1], that improve the ability of a PCI adapter to be directly shared by multiple VMs. SR-IOV provides the ability for multiple VMs to directly share IO within a single server. Direct sharing circumvents the resource and processing overhead inherent in Hypervisor based Ethernet bridging. With SR-IOV, communications between the VM device driver and the adapter (e.g. a PCIe adapter doorbell for a transmit queue and adapter DMA operations) are performed at levels equivalent or close to that of a dedicated adapter[7].

However, SR-IOV did not define the Virtual Ethernet Bridging (VEB) mechanisms needed to bridge between multiple VMs in the same server and an external port on the PCIe adapter. For VM to VM communication, an adapter based VEB is required as depicted in Figure 1.

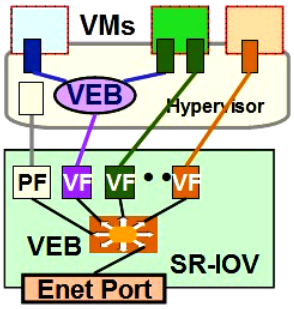

Figure 1

Using an internal PCIe adapter VEB allows VM-to-VM communication to be performed at PCIe bandwidth and latencies, versus the lower bandwidths and higher latencies that would be the case if the VEB were to be moved below the adapter's Ethernet port. In order to achieve efficient and feature rich VM-to-VM communication, a VEB requires a minimum set of interoperability features in provisioning, access controls, security, automation, and management. The following sections will provide an overview of light-weight VEB implementation examples for some of the necessary features.

1) MAC/VLAN Management and Access Controls

In a direct IO sharing model, each VM needs to be assigned a source MAC Address to use when communicating to either other VMs or through the external port. To prevent one VM from spoofing another VM's source MAC Address, the PCIe adapter needs to perform MAC Address Access Controls. There are several options for how to perform this function in the adapter, including:

- Multi-queue approach, where each queue is associated with a VM. In this model the MAC Address is associated with a queue. The adapter validates that the Source MAC Address used by the VM matches the MAC Address associated with the queue.

- Multi-Function approach, where each PCI Function is associated with a VM. In this model the MAC Address is associated with a PCI Function. The adapter validates that the Source MAC Address used by the VM matches the MAC Address associated with the PCI Function.

- SR-IOV based approach, where each PCI Virtual Function is associated with a VM. In this model the MAC Address is associated with a PCI Virtual Function. The adapter validates that the Source MAC Address used by the VM matches the MAC Address associated with the PCI Virtual Function.

The options that follow can be used for any of the above three methods of directly sharing an IO adapter across multiple VMs. To simplify the description, we'll document the usage model for an SR-IOV based approach.

There are a range of options for preventing MAC Address spoofing between VMs sharing the same PCIe adapter. One prudent approach is to have the Virtualization Intermediary (VI), such as a Hypervisor, program an allowed set of MAC Addresses, which a given VM is allowed to use. Under this mode a VM can select a single MAC Address from a subset of MAC Addresses the VI pre-populated in the SR-IOV's Virtual Function (VF) context. The adapter must compare the Source MAC Address used by the VM to the MAC Address stored in the VF context. If the frame's Source MAC Address equals one of the four MAC Addresses in the VF context, the frame is forwarded to its destination. Otherwise the frame is discarded. Figure 2 depicts the VF context associated with this mode of operation.

In addition to the MAC Address access control, each VF's virtual Ethernet port needs to have the ability of being associated with one or more VLAN identifiers. Again there are several options for implementing a VLAN access control. One prudent approach is to have the VI program an allowed set of VLAN identifiers that are associated with a given VM. This approach is depicted in figure 2. Under this mode the adapter ensures egress frames from a VM use one of the

VLAN identifiers stored in the VF context table shown in figure 2. If so, the frame is forwarded towards its destination, otherwise it is discarded. Similarly, on ingress, the adapter ensures the incoming frame has one of the VLAN identifier ID in figure 2. If so, the frame is forwarded to the appropriate VF or VFs (for multicast/broadcast frames), otherwise it is discarded.

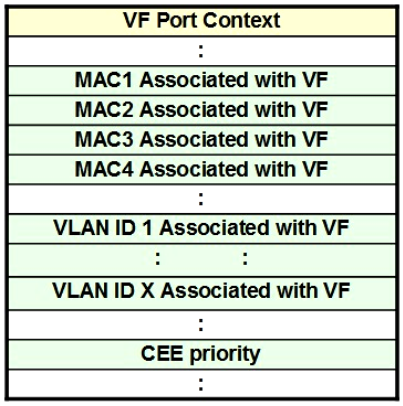

Figure 2

Figure 2 also depicts the CEE priority that the VI associates to a VF. The VI must also have the ability to manage the adapter's Data Center Bridging eXchange[21] protocol's Enhanced Transmission Selection[22] and Priority Based Flow Control[23] configuration.

2) Network Security and Diagnostics

In a physical data center, security appliances are used to provide Intrusion Detection and Prevention (IDP) functions, for example inspection of communication between the tiers in a multiple-tier data center. Integrating VMs from these multiple tiers onto the same physical server requires the communication between the VMs to have the same level of security inspection as performed in the physical fabric. To perform this function a virtual appliance is required to inspect VM-to-VM communications. An adapter based VEB implementation must satisfy this requirement.

In order to support virtual security appliances, we propose that an adapter VEB support two mechanisms: Port-Mirroring and Port-Pass-Through. Port-Mirroring is used to support a virtual intrusion detection appliance by mirroring all frames received by the adapter to the virtual appliance. Port-Pass-Through is used to support a virtual intrusion prevention appliance by forwarding all frames received by the adapter through the virtual appliance. These two mechanisms are advertised through a PCI capability resident in the Function associated with the VI. At adapter initialization time, the VI may enable either of these mechanisms. For an SR-IOV implementation, if either function is enabled, the VI also sets the Virtual Function (or VFs) number used by the virtual security appliance. An analogous approach is used for a multi-queue (i.e. VI sets the queue number used by the virtual security appliance) or multi-function adapter (i.e. VI sets the function number used by the virtual security appliance).

Under the Port-Mirroring mechanism, all VF-to-VF traffic and ingress frames are forwarded to the destination and also mirrored to a Virtual Function assigned to virtual intrusion detection appliance. The intrusion detection appliance inspects the frames. If a frame is benign, it is silently dropped. If it is found to have a possible malignancy, an alert is surfaced through the virtual security appliance's manager. Port-Mirroring increases the amount of traffic processed through the adapter, therefore it must be configured with sufficient resources (e.g. queues, interrupts and other VM processing resources) to handle the additional load. Depending on the workload, the use of a virtual intrusion detection appliance can consume significant server resources (CPU, memory and IO). The performance analysis is outside the scope of this document.

Under the Port-Pass-Through mechanism, all VF-to-VF traffic and ingress frames are redirected to a Virtual Function assigned to virtual intrusion prevention appliance. The intrusion prevention appliance inspects the frames, which if determined to be benign are then forwarded to the destination VM (or VMs for multicast/broadcast) or external network through the VEB. Otherwise the frames are dropped. Similar to Port-Mirroring, Port-Pass-Through must be configured with sufficient resources to handle the additional processing load.

Given an adapter configured for Port-Mirroring or Port-Pass-Through may get overloaded, the VEB needs to maintain a statistics register is accessible by the virtual appliance and logs the number of frames that bypassed Port-Mirroring or Port-Pass-Through.

3) Link Management

VM-to-VM communications occurs through the VEB over a virtual link and does not pass through the external switch. Under this model if the physical link goes down, the VI is notified, but local VMs can continue to communicate through the VEB. In order to track link loss events for failover and notification actions, separate link states should be exposed for the VEB's virtual ports. For an SR-IOV adapter, a VF can receive link state change notifications via interrupts and enable the VM to determine the correct course of action immediately after a Physical or Virtual link state change. The likely use cases are physical link aggregation, link switchover and event tracking.

4) Multicast Management

Forwarding of Multicast (MC) and Broadcast (BC) frames can be performed by the: adapter; or through a special purpose (Physical or Virtual) function assigned for MC/BC forwarding. For the latter, the adapter VEB redirects MC/BC traffic to the special function assigned for MC/BC forwarding. For MC in an SR-IOV adapter, the manager that owns the special function forwards the frame to the external port associated with the special function and to each local function (PCIe VF, PF or Function) associated with the MC address. For BC in an SR-IOV adapter, the manager that owns the special function forwards the frame to the external port associated with the special function and to all functions (PCIe VF, PF or Function). Note, if a MC or BC frame was sent from a local function the frame is not forwarded to that function.

Figure 3 below depicts an outbound MC message processed through the manager that owns the special purpose function. In figure 3, the left most VM (System Image, in PCI SIG parlance) sends an MC frame through its transmit queue. The adapter's VEB forwards the frame to the MC/BC manager, which performs the forwarding look-up described above. The MC/BC manager sends three frames through its transmit queue, one for each of the VMs that are associated with the MC address (both VMs in the example below) and one for the external port.

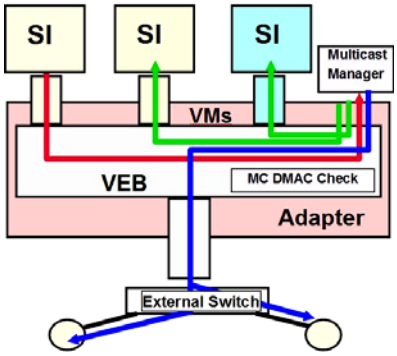

Figure 3

An adapter based MC/BC implementation essentially offloads the forwarding performed by the MC/BC manager. In this case, for MCs, the VI uses a MC address table to determine if an incoming MC frame's destination address exists in the MC table. If not, the frame is forwarded to the MC/BC special purpose VF, so that software based MC processing can be performed. Otherwise, the adapter looks up the list of functions (PCIe VF, PF or Function) associated with the MC Address and forwards the packet to all associated functions. For BC, the adapter forwards the frame to all functions (PCIe VF, PF or Function). Note, if a MC or BC frame was sent from a local function, the adapter does not forward the frame to that function. Figure 4 depicts an example of these tables for an SR-IOV implementation.

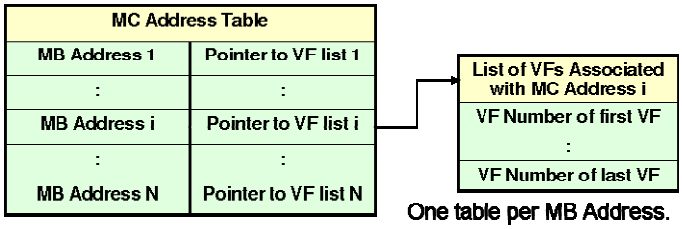

Figure 4

5) Convergence Enhanced Ethernet (CEE) and VEB

Virtual Ethernet Bridging is an integral part of fabric convergence and Data Center Bridging. There are several options for how the server can implement LLDP (802.1AB) and the DCB capability eXchange protocol (802.1az). One is to implement these functions in the VI. Another is to implement these functions in the adapter and allow the VI to configure each adapter port's DCBX capabilities. Following is a description of the latter for an SR-IOV based implementation.

The VI defines which traffic classes are enabled on the adapter's physical port at physical link initialization. The VI may also set desired DCBX settings, such as the bandwidth allocated to a priority group and whether a priority group has priority based flow control enabled. The adapter then uses the DCBX capability exchange to synchronize the configuration with the external switch. The adapter also implements the DCBX state machine and manages renegotiation during a configuration change. After DCBX negotiations are complete, the VI is responsible for assigning one or more priority groups to each VF and the adapter is responsible for assuring that VF only transfers frames using the VI assigned priority group.

6) Traffic Scheduling Across VFs

In addition to CEE's Enhanced Transmission Selection a mechanism is needed to schedule traffic across functions (PCIe VF, PF or Function). Following is a proposed approach for scheduling traffic for an SR-IOV implementation.

Three variables are used to schedule traffic across functions (PCIe VF, PF or Function): a maximum capacity; a weight and a minimum capacity. The maximum capacity defines the maximum percentage of the egress link bandwidth the adapter will make available to the function even if there is no link contention. That is, the function's egress bandwidth will not exceed this value. The minimum capacity defines the minimum percentage of the egress link bandwidth the adapter must make available to the function. The weight defines a

weighted priority at which each function competes for excess capacity on the link. The weight value allows for prioritizing the functions relative to each other such that a higher priority function is favored over a lower priority function by the weight associated with each. Figure 4 depicts a conceptual view of this approach in terms of the function's Virtual Port.

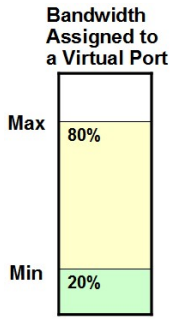

Figure 5

IV. Conclusion

The mechanisms in this paper describe use cases for when to perform Ethernet virtual bridging within the server vs in the external fabric. For the latter, the paper proposes mechanisms for automating migration of an external switch's port profile when a VM is migrated across physical servers. It also proposes a mechanism for providing unique locally administered MAC Addresses within a layer-3 domain. For the use case where Ethernet virtual bridging is performed within the server, this paper describes mechanisms for the necessary VEB functions. We are working with adapter providers on an additional DCBX TLV for the internal switch's attributes.

V. Acknowledgements

The authors are grateful to Parag Bhide, Paul Congdon, Yaron Haviv, Vivek Kashyap, Bruce Klemin, James Macon, Daniel Martin, Vijoy Pandey, Rakesh Sharma, Manoj Wadekar & Suresh Vobbilisetty, for their insightful comments.

References

[1]PCI SIG SR-IOV, www.pcisig.com/specifications/iov/single_root/

[2]Integrated Virtual Ethernet Adapter Technical Overview and Introduction, www.redbooks.ibm.com/redpapers/pdfs/redp4340.pdf

[3] Ko, Mike; Recio, Renato; Virtual Ethernet Bridging, www.ieee802.org/1/files/public/docs2008/new-dcb-ko-VEB-0708.pdf

[4] Ferrer, M. "Measuring Overhead Introduced by VMWare Workstation Hosted Virtual Machine Monitor Network" Subsystem studies.ac.upc.edu/doctorat/ENGRAP/Miquel.pdf

[5] Sugerman, J.; Venkitachalam, G.; Lim, B; "Virtualizing I/O devices on VMWare Workstation's hosted virtual machine monitor" www.usenix.org/events/usenix01/sugerman/sugerman.pdf

[6] Ram, K.K,; Santos, J. R.; Turner, Y.; Cox, A.; Rixner, S.; "Achieving 10 Gb/s using Xen Para-virtualized Network Drivers", www.xen.org/files/xensummit_oracle09/xensummit_networking.pdf and www.xen.org/files/xensummit_4/NetworkIO_Santos.pdf

[7] Raj, H; Ganev, I.; Schwan, K.; Xenidis, J.; "Self-Virtualized I/O: High Performance, Scalable I/O Virtualization in Multi-core Systems" www.cercs.gatech.edu/tech-reports/tr2006/git-cercs-06-02.pdf

[8] W. J. Armstrong, R. L. Arndt, D. C. Boutcher, R. G. Kovacs, D. Larson, K. A. Lucke, N. Nayar, R. C. Swanberg, "Advanced virtualization capabilities of POWER5 systems", IBM J. RES. & DEV. Vol. 49 No. 4/5 July/September 2005

[9] IBM PowerVM Live Partition Mobility, "http://www.redbooks.ibm.com/abstracts/sg247460.html"

[10] Cisco Nexus 1000V Virtual Switch Data Sheet, "http://www.cisco.com/en/US/prod/collateral/switches/ps9441/ps9902/data_sheet_c78-492971.pdf" Cisco 2008

[11] Cisco, Virtual Networking Features of the VMware vNetwork Distributed Switch and Cisco Nexus 1000V Series Switches, www.cisco.com/en/US/prod/collateral/switches/ps9441/ps9902/solution_overview_c22-526262.pdf

[12] J. Pelissier, "Introduction to VNTag", DC CAVES 2009.

[13] P. Congdon, "Virtual Ethernet Port Aggregator Standards Body Discussion", www.ieee802.org/1/files/public/docs2008/new-congdon-vepa-1108-v01.pdf

[14] 802.1x, IEEE Standard for Local and metropolitan area networks, Port-Based Network Access Control, December 2004.

[15] Blunk, L., Vollbrecht, J., "PPP Extensible Authentication Protocol (EAP)", RFC 2284, March 1998.

[16] IETF RFC 3579, RADIUS Support for Extensible Authentication Protocol (EAP), B. Aboba, P. Calhoun, September 2003.

[17] IETF RFC 3748, Extensible Authentication Protocol (EAP), Blunk, L., Vollbrecht, J., Aboba, B., Carlson, J., Levkowetz, H, June 2004.

[18] P.Congdon, IETF RFC 3580, IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines, September 2003.

[19] Rigney, C., Willens, S., Rubens, A. and W. Simpson, "Remote Authentication Dial in User Service (RADIUS)", RFC 2865, June 2000.

[20] Rigney, C., Willats, W. and P. Calhoun, "RADIUS Extensions", RFC 2869, June 2000.

[21] IEEE P802.1Qaz: DCB Capability Exchange Protocol (DCBX), www.ieee802.org/1/files/public/docs2008/azwadekar-dcbx-capability-exchange-discovery-protocol-1108-v1.01.pdf

[22] IEEE P802.1Qaz: Enhanced Transmission Selection (aka Priority Groups), http://www.ieee802.org/1/files/public/docs2008/azwadekar-ets-proposal-0608-v1.01.pdf

[23] IEEE P802.1Qbb: Priority-based Flow Control (PFC), http://www.ieee802.org/1/files/public/docs2008/bbpelissier-pfc-proposal-0508.pdf