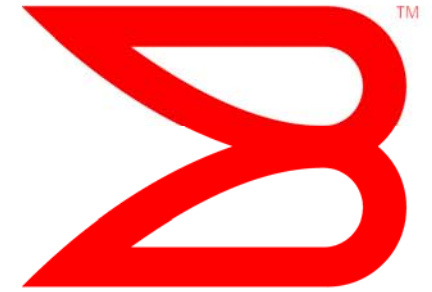


BROCADE



Clarification concerning the use of SPMA

Includes a further look at ACLs required for :
Server Provided MAC Address Environments
and a review of the complete picture

A follow-on presentation to T11/07-656v0

T11/07-690v0

John L. Hufferd
Robert Snively
Anoop Ghanwani
Suresh Vobbilisetty

Remember (From the T11/07-656v0)

We showed how ACLs would work with Server Provided MAC Address (SPMA) environments

We contrasted them with the Network Provided MAC Address (Mapped MAC Address) environments

(See Slides in the Backups)

We showed how individual Servers with Software-FCoE or single “Burnt-In” MAC address FCoE HBAs will be protected with SPMA ACLs

We want to quickly review the SPMA ACLs

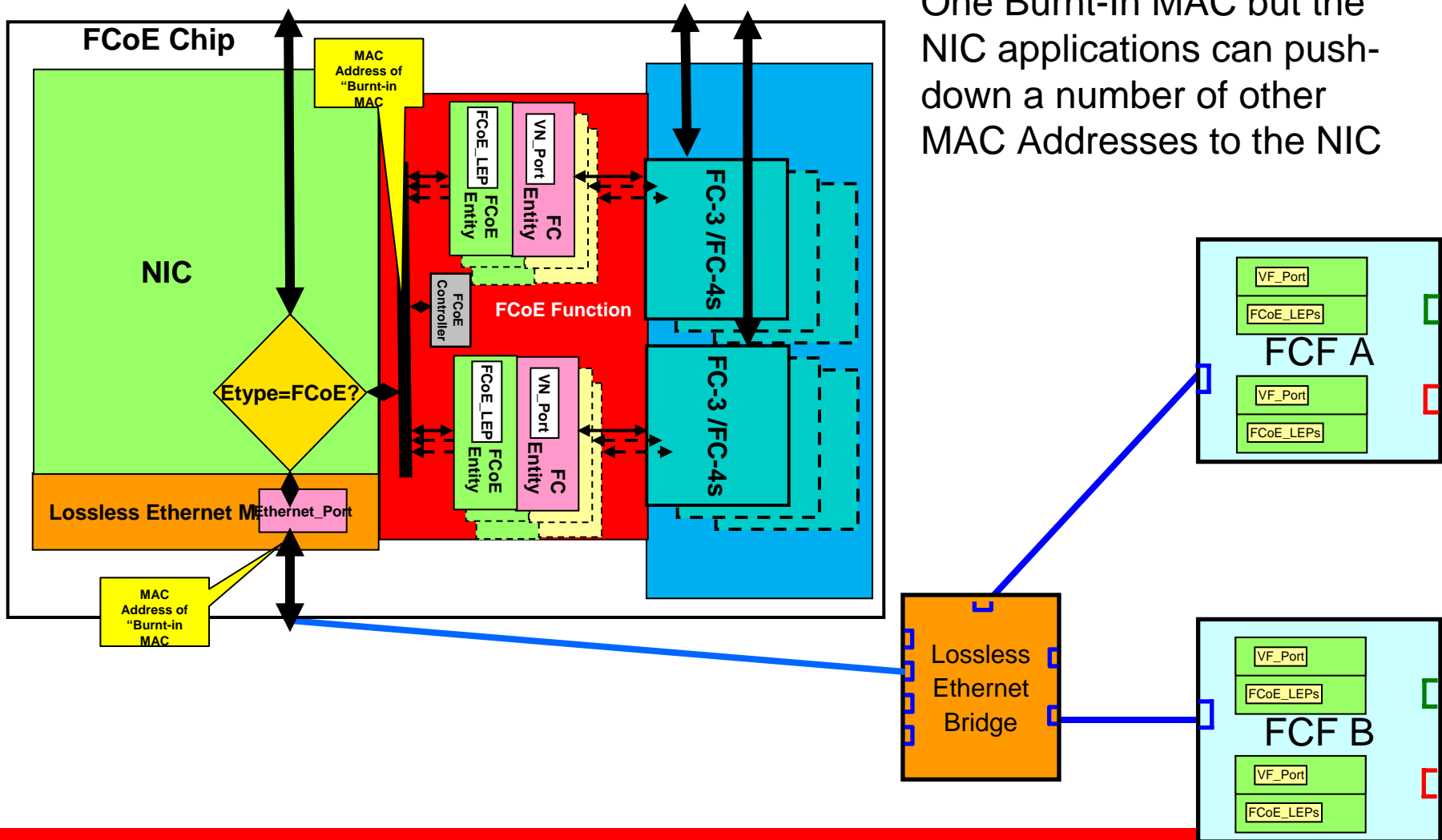
Then continue where we left off,

- We will go over how SPMA ACLs can be used in Virtualized Software FCoE environments
- And then we will show ACL with dual MAC Address FCoE HBAs



Remember the Single Burnt-in MAC FCoE HBA ? (Shown in T11/07-655v0)

One Burnt-In MAC but the NIC applications can push-down a number of other MAC Addresses to the NIC



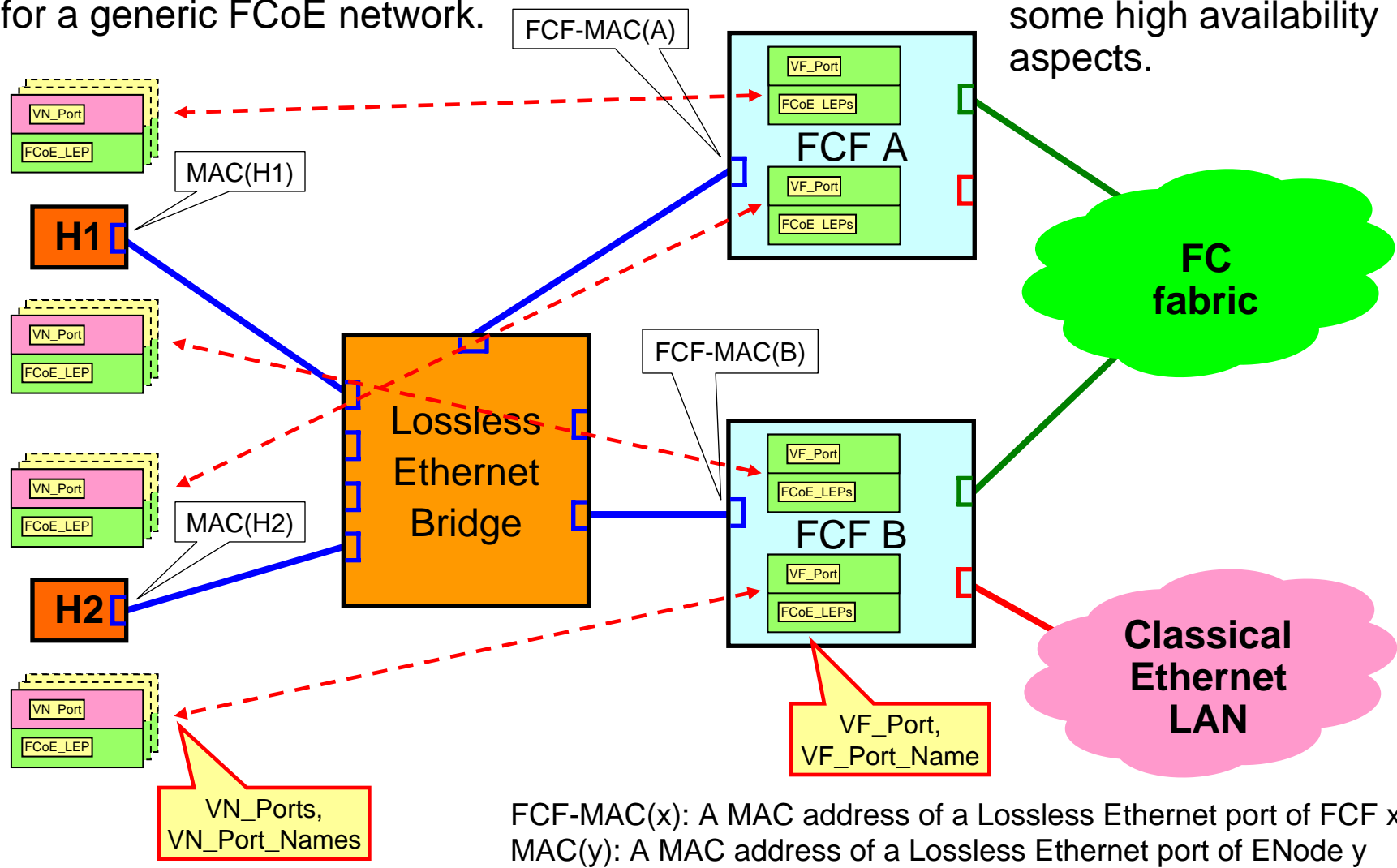
Review



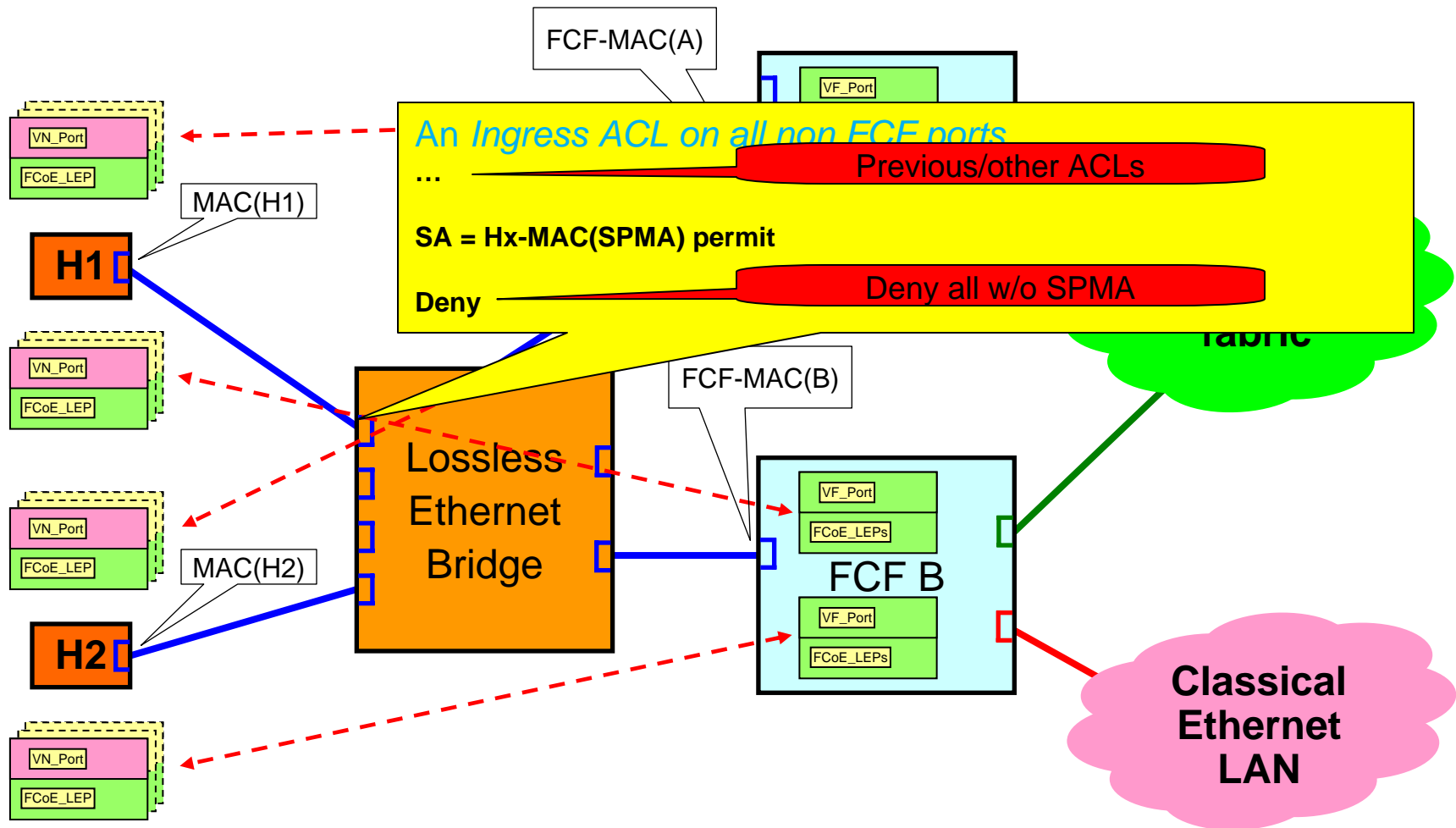
Generic Topology with NPIV stacks

A Discovery protocol is needed for a generic FCoE network.

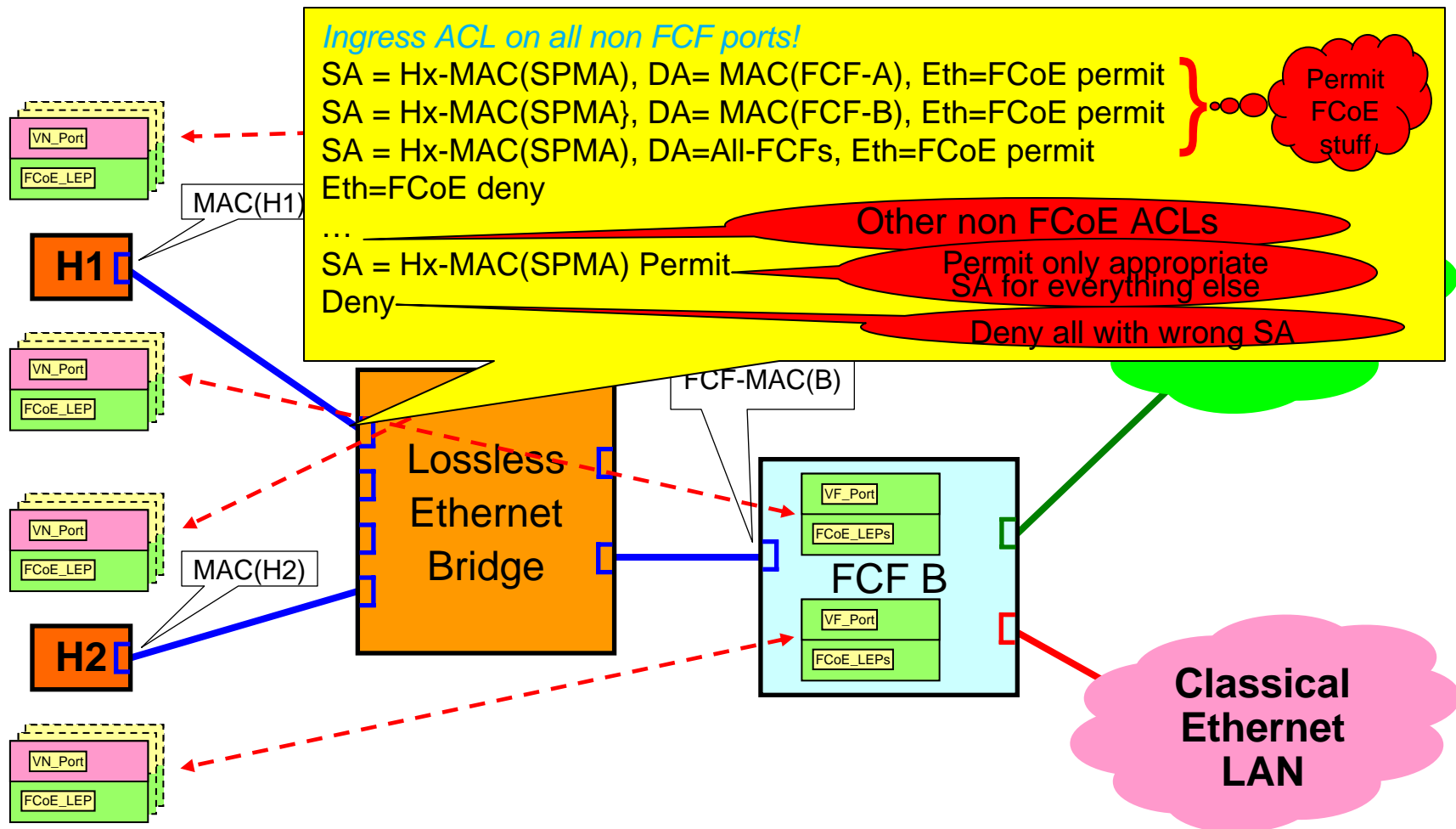
The configuration has some high availability aspects.



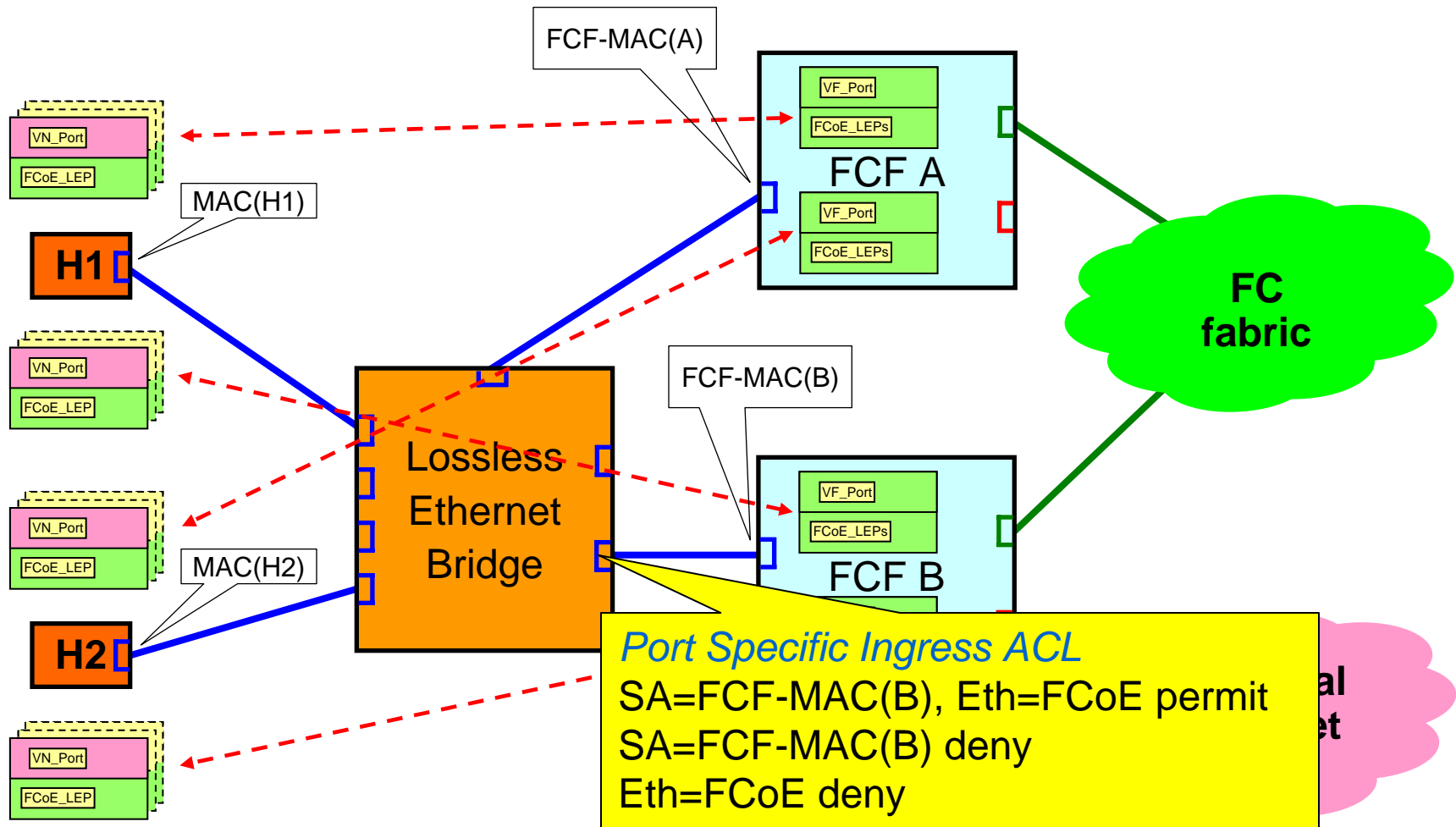
Generic Topology (simple) ACLs (with Server Provided MAC Addresses [SPMA])



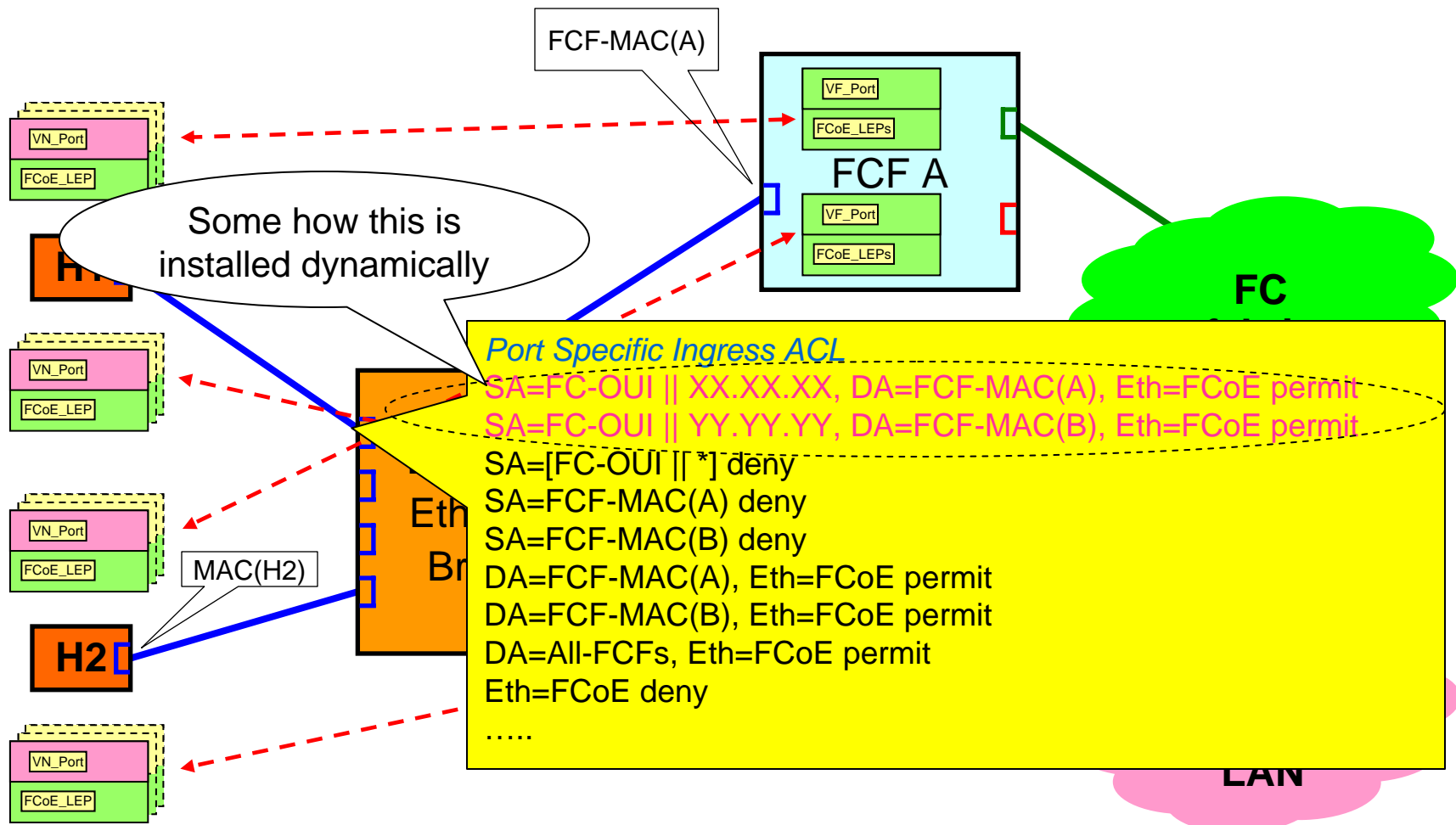
Generic Topology & more sophisticated ACLs (with Server Provided MAC Addresses [SPMA])



Ingress ACLs for FCF Ports (with Server Provided MAC addresses)



Remember the Dynamic ACLs (with Network Provided MAC Addresses)



Advantage of SPMA for planning for new installations

Some server vendors send MAC address of NICs (and FCoE HBAs) before the equipment is sent to the installation

- So that installation can begin work on establishing ACLs, and management processes before equipment arrives
- SPMA is the best fit for this

Installations want to be able to use any vendors Ethernet switches

- Want to use Standard Lossless Ethernet Switches with ACLs to prevent Rogues
- Sometimes want to use 802.1X with RADIUS Servers
- SPMA is the best fit for this need

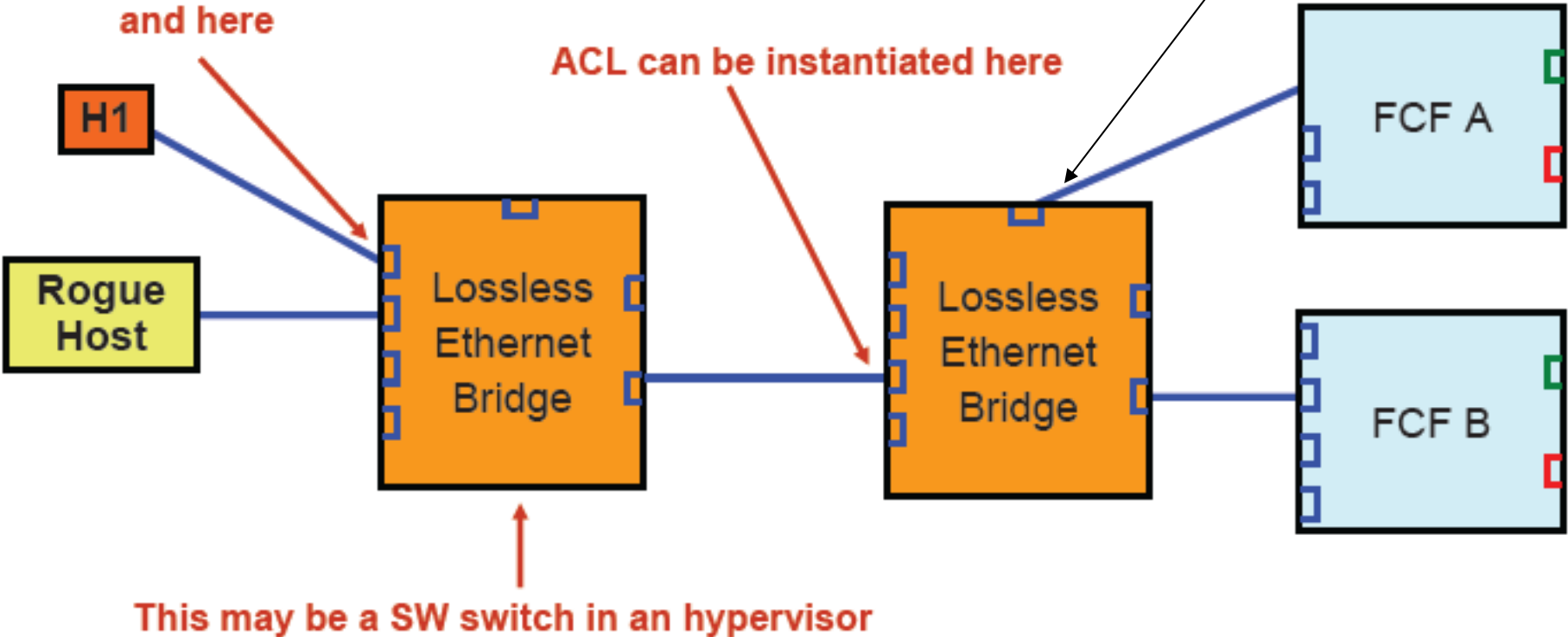
But what about Virtualized environments?



Double Ethernet Bridges (Virtual & Real)

This chart was shown in previous presentations

And of course here for FCFs Ports



NOTICE

Currently Hypervisor Vswitch do NOT have the capability of ACLs

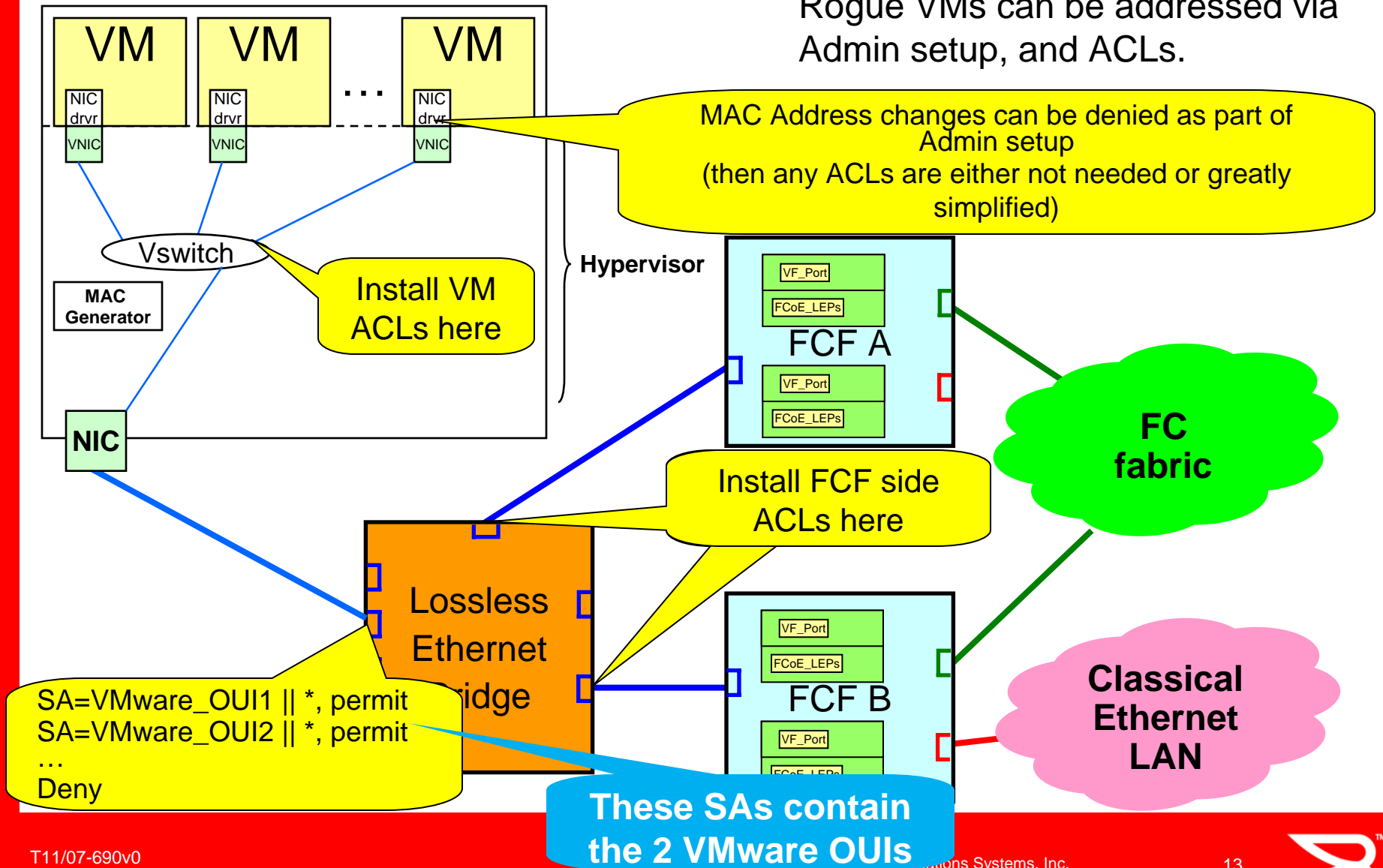
- These slides do NOT imply anything about timing or even whether the capability will ever exist

However, since such capabilities have been claimed to be possible, by members of T11, this presentation assumes that if and when such a capability exists that the ACLs shown herein are the way such protections might be employed



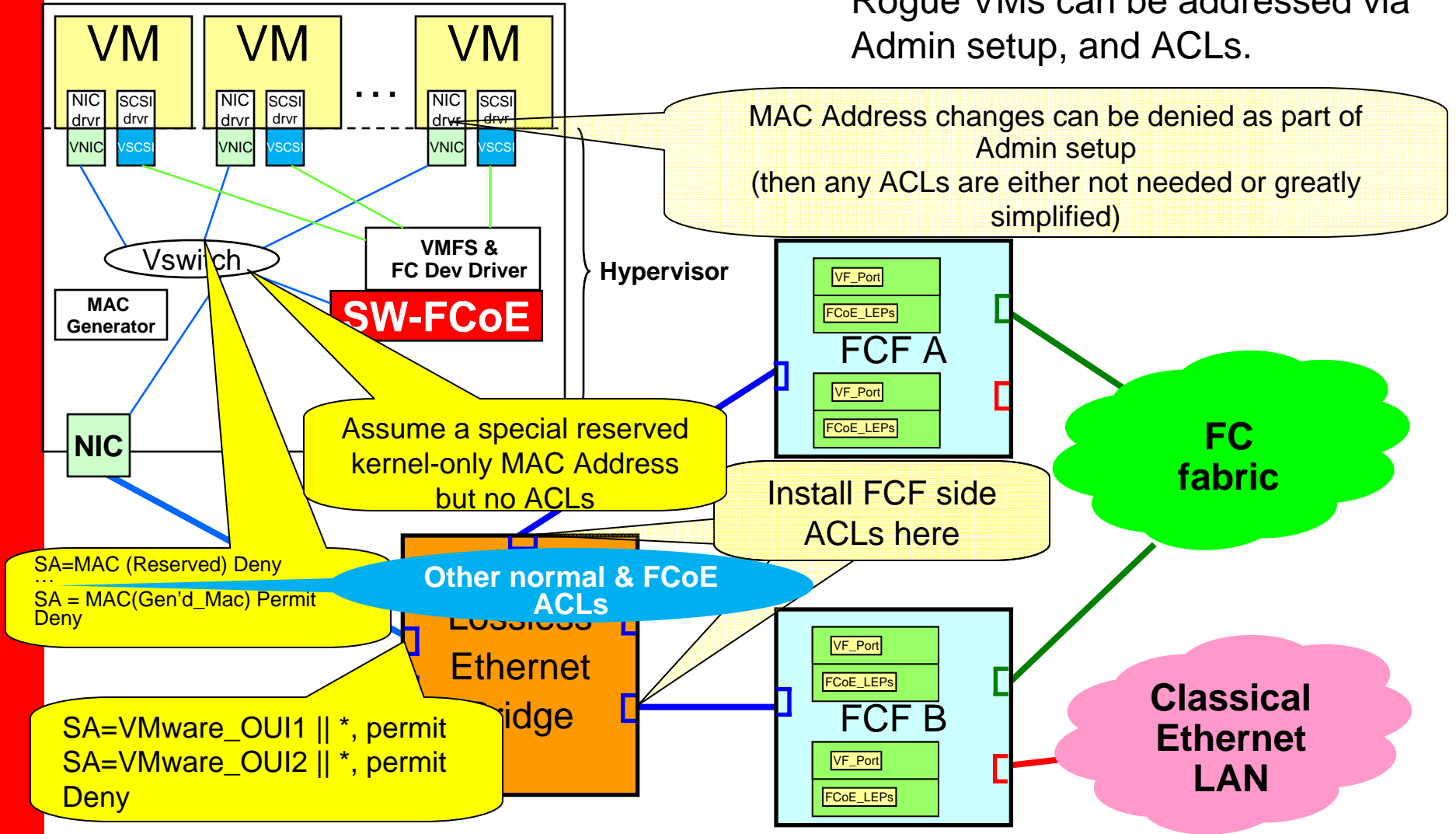
VMware & Double Ethernet Bridges with ACLs Concept

Rogue VMs can be addressed via Admin setup, and ACLs.



Software FCoE in VMware & a shared Vswitch

Rogue VMs can be addressed via Admin setup, and ACLs.



Vswitch ACLs

The ACL shown was intended to show an example of such ACLs

```
SA=MAC (Reserved) Deny {Prevent spoofing Hypervisor MAC Addr}
... {include other non FCoE ACL, and perhaps more FCoE permits}
SA = MAC(Gen'd_Mac) Permit {Permit the Generated MAC Addr}
Deny {Deny all other traffic with some other MAC}
```

The ACLs can be more sophisticated and include statements similar to those shown previously which limit the FCoE DAs, or permit special push down MAC address, etc

See the following slide for such an example



Vswitch ACLs with add'l FCoE tests

The ACL shown below is intended to show another example of such ACLs

SA=MAC (Reserved) Deny {Prevent spoofing Hypervisor MAC Addr}

SA = MAC(Gen'd_Mac), DA= MAC(FCF-A), Eth=FCoE permit {Permits access to FCF-A}

SA = MAC(Gen'd_Mac), DA= MAC(FCF-B), Eth=FCoE permit {Permits access to FCF-B}

SA = MAC(Gen'd_Mac), DA=All-FCFs, Eth=FCoE permit {Permits discovery of FCFs}

Eth=FCoE deny {Disable other FCoE access}

... {include other non FCoE ACLs}

SA = MAC(Gen'd_Mac) Permit {Permit the Generated MAC Addr for other stuff}

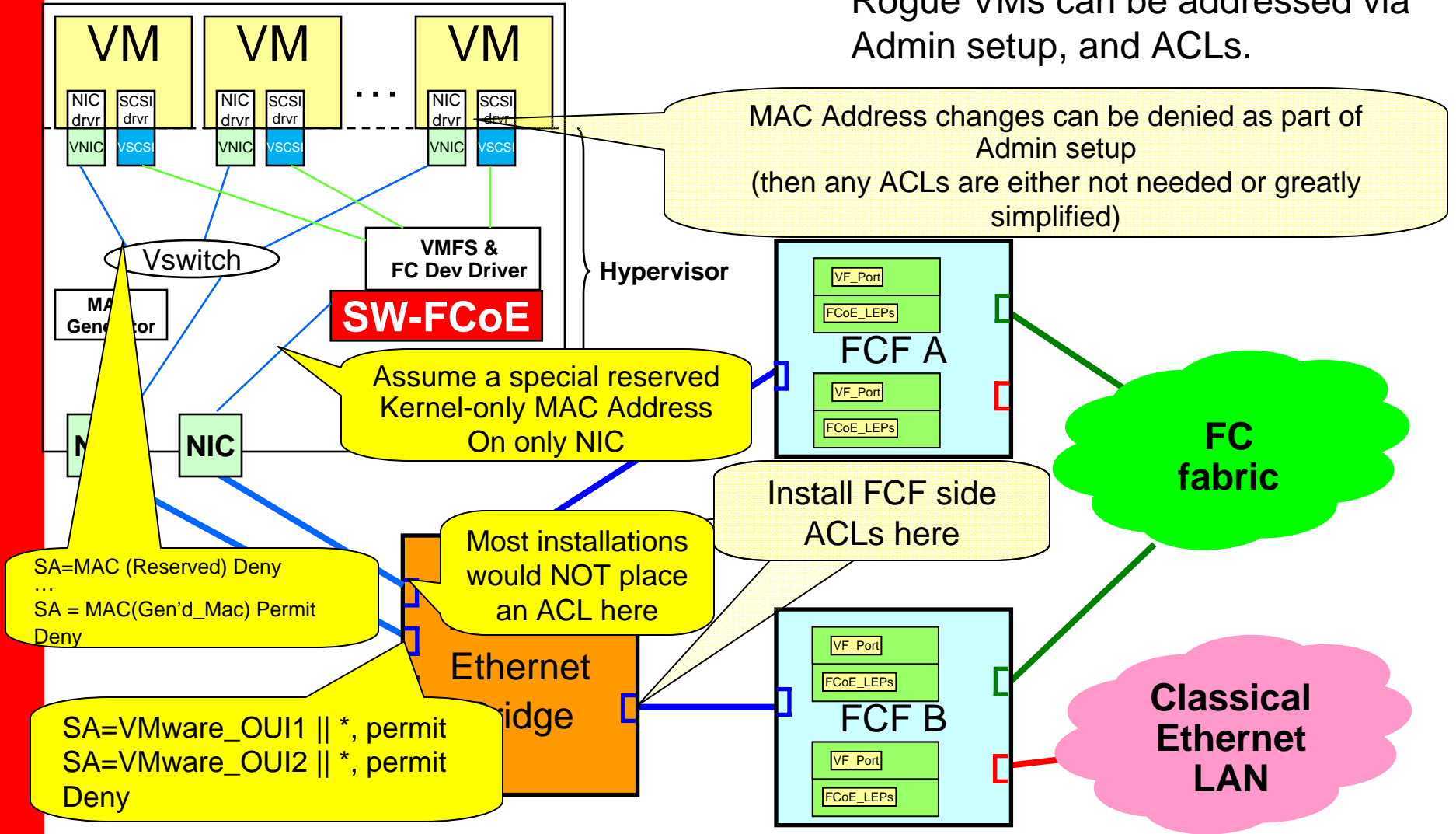
Deny {Deny all other traffic with some other MAC}

The above ACL prevent spoofing the Hypervisor MAC address, and permit a VM system to operate a test software FCoE

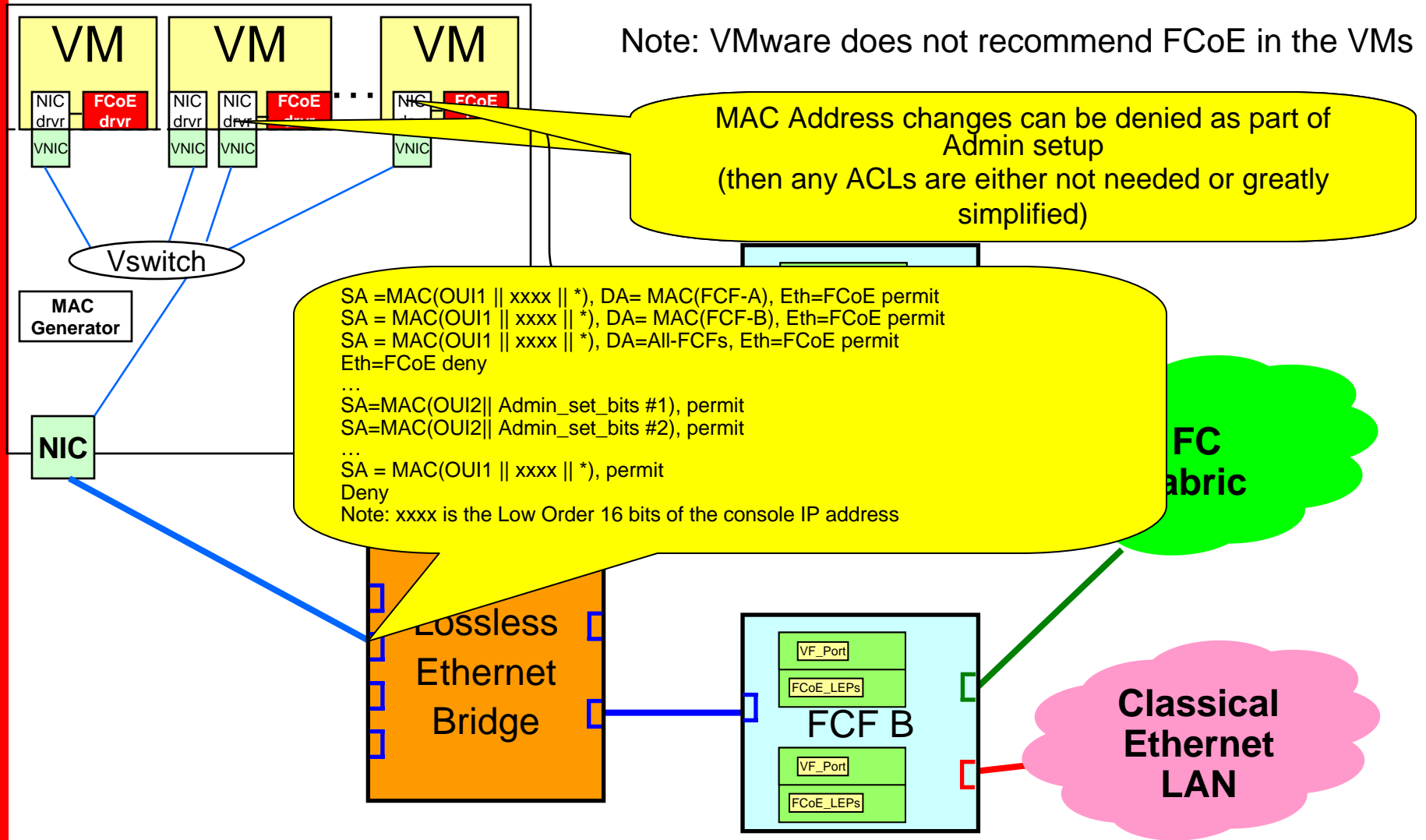


Software FCoE in VMware & Separate NICs

Rogue VMs can be addressed via Admin setup, and ACLs.



Software FCoE within the VMs and ACL in External Switch



External ACLs with VMware servers

The ACL shown below is intended to show another example of such ACLs

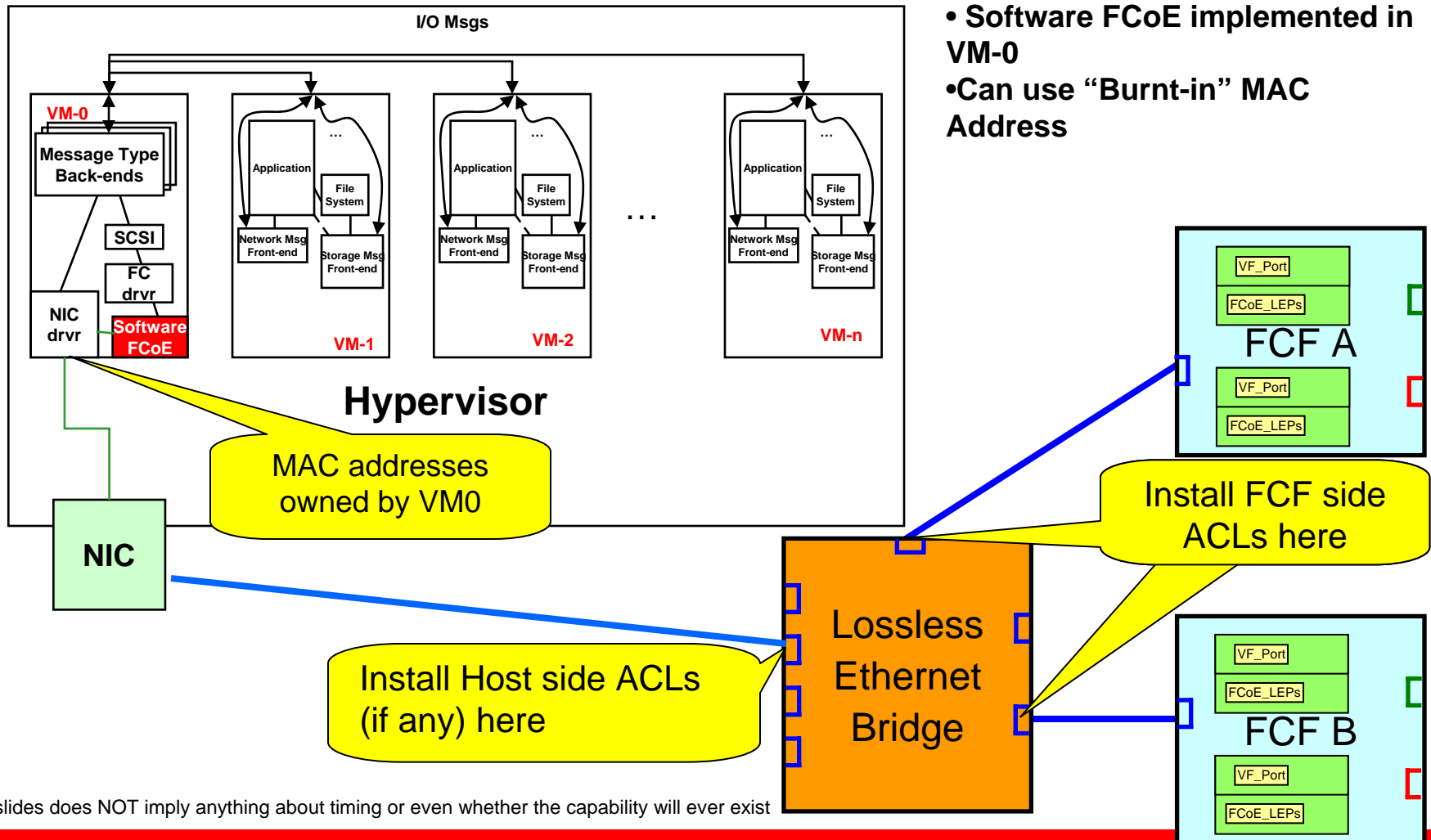
Note that VMware creates unique MAC addr from one of its OUIs and lower 16 bit of the Admin console IP addr and then adds unique values to the rest of the MAC addr

```
SA =MAC(OUI1 || xxxx || *), DA= MAC(FCF-A), Eth=FCoE permit {Permit VMware ports to access the FCF}
SA = MAC(OUI1 || xxxx || *), DA= MAC(FCF-B), Eth=FCoE permit {Permit VMware ports to access the FCF}
SA = MAC(OUI1 || xxxx || *), DA=All-FCFs, Eth=FCoE permit {Permit VMware ports to discover the FCF}
Eth=FCoE deny
... {include other IP ACLs}
SA=MAC(OUI2|| Admin_set_bits #1), permit {Permit Admin set #1 MAC to operate}
SA=MAC(OUI2|| Admin_set_bits #1), permit {Permit Admin set #2 MAC to operate}
... {Permit other Admin set MACs to operate}
SA = MAC(OUI1 || xxxx || *), Permit {Permit VMware ports to do other stuff}
Deny {Deny any Packet that does not have right SA}
```

Note: xxxx is the Low Order 16 bits of the console IP address



Software FCoE as maybe seen in Xen or Microsoft Virtualization *



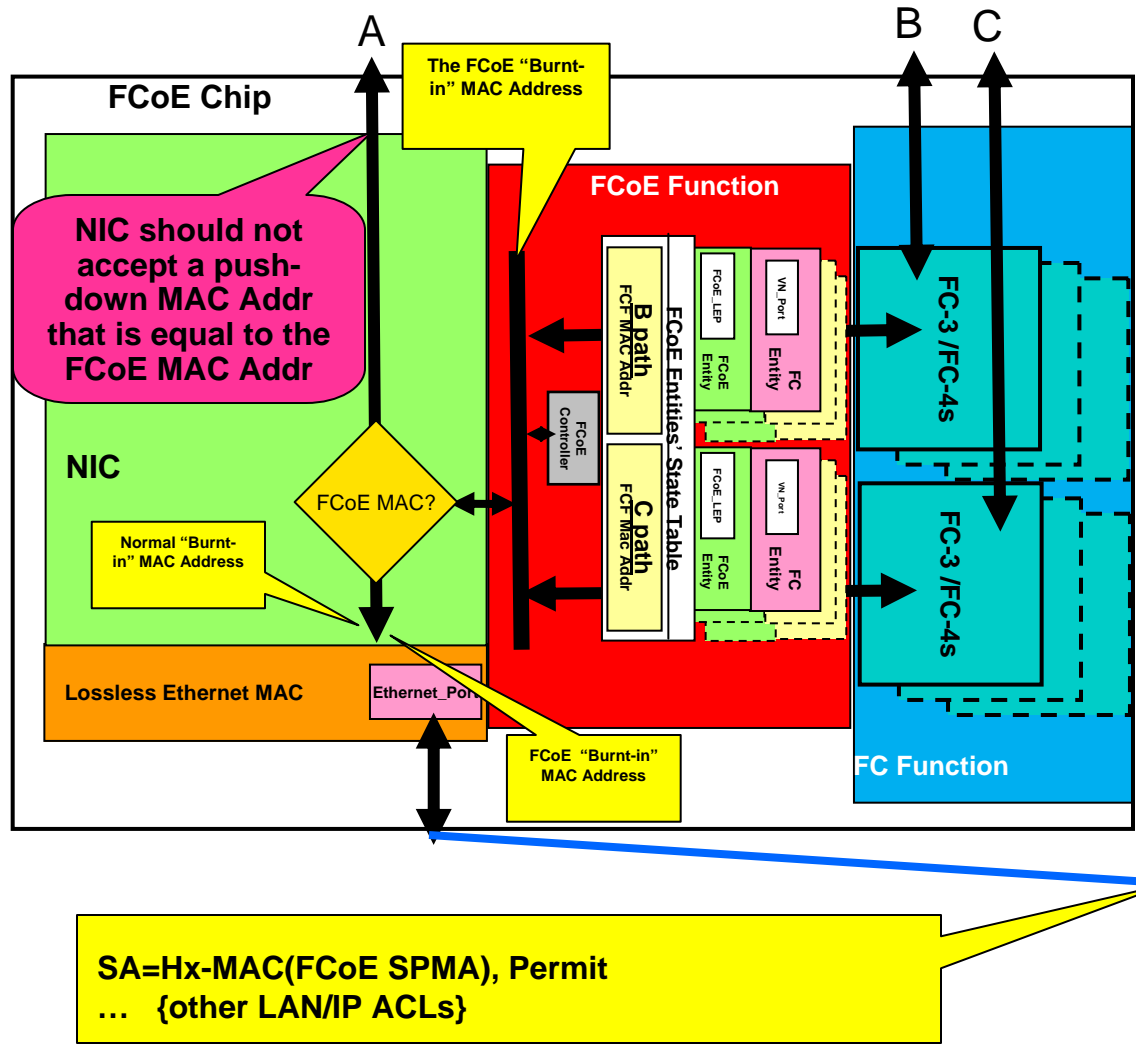
* This slides does NOT imply anything about timing or even whether the capability will ever exist



What about HBAs with a special Burnt-in MAC for just FCoE?



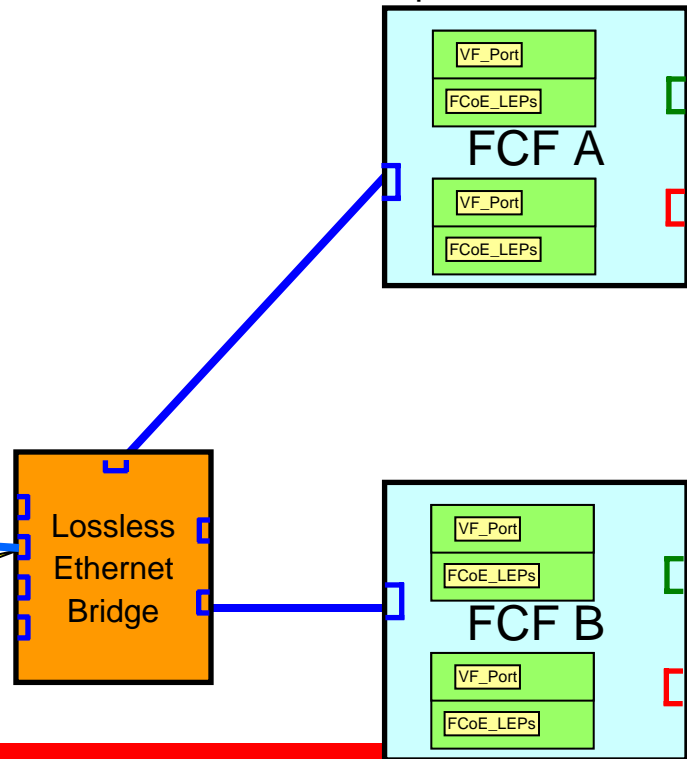
Remember the Dual MAC Address FCoE HBA ? (Shown in T11/07-655v0)



HBA can have Two Burnt-in MAC address

- One for **NON** FCoE
- &
- One for FCoE

Then ACLs can be simplified



ACLs used with a Dual MAC Addr HBA

The ACL shown below is intended to show an example of such ACLs

The ACL below assumes that Burnt-In MAC HBAs are NOT rogue

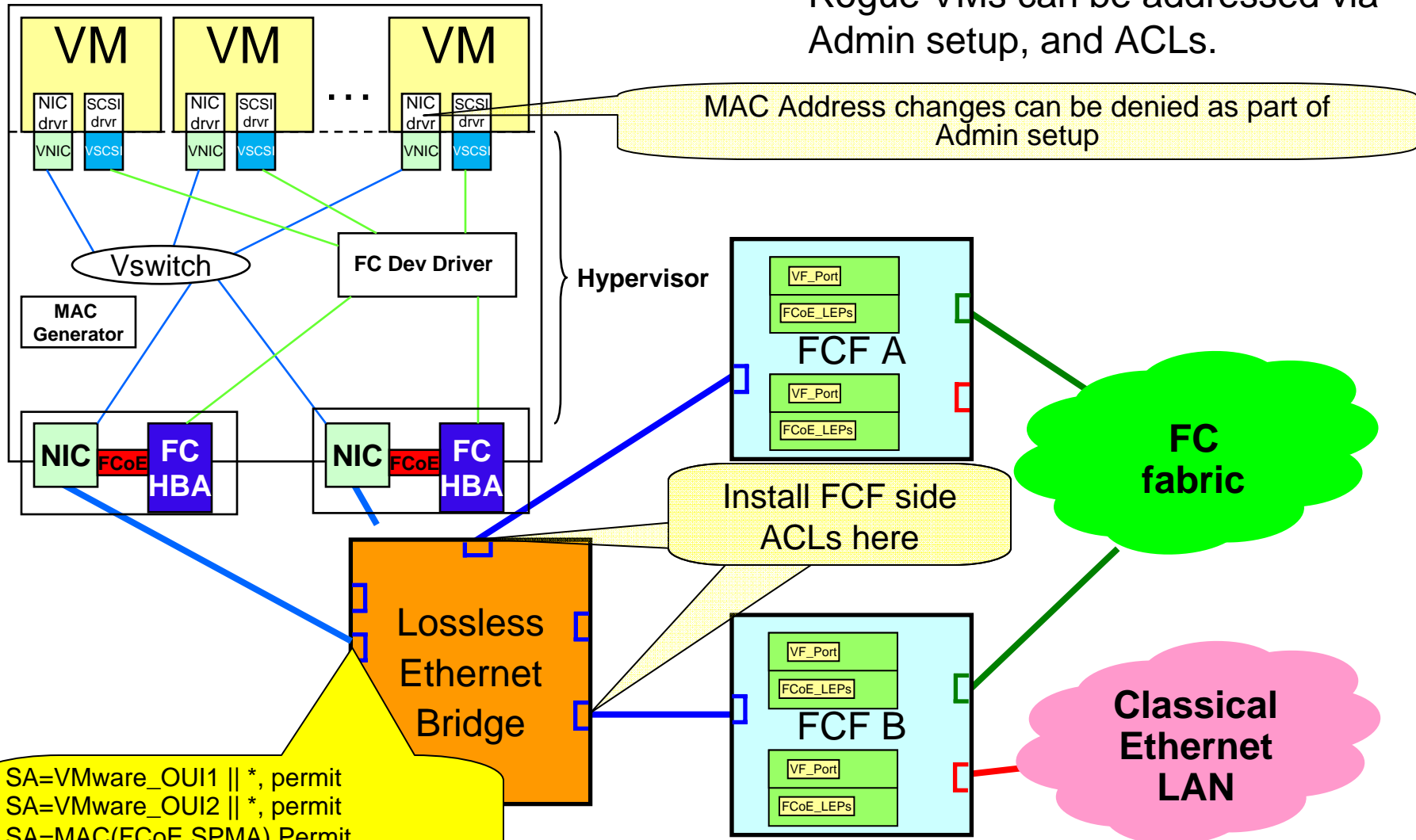
SA=Hx-MAC(FCoE SPMA), Eth=FCoE, Permit {Let the FCoE HBA do what ever it wants}

... {Include other ACLs here for normal LAN/IP traffic}



Hardware Virtualization with Dual MAC Addr FCoE HBAs

Rogue VMs can be addressed via Admin setup, and ACLs.



SA=VMware_OUI1 || *, permit
 SA=VMware_OUI2 || *, permit
 SA=MAC(FCoE SPMA) Permit

 Deny



Analysis

Characteristic	Server-provisioned MAC (“Burnt-in”)	Fabric-provisioned MAC (“Mapped”)
Rogue spoofing of switch learning	Denied	Possible for Discovery and Login
Dynamic ACL modification	Not required	Required, but no standard mechanism
Number of lines of ACL per ingress	Can be a Small number	May be large number with Dynamic updates
Static administration	Yes	Does NOT work (Requires subsequent dynamic modification)
Able to use current Ethernet Switches	Yes	No; needs new ACL protocol, or Snooping for ACL et.al.
Compatible with 802.1X	Yes	No; requires dynamic update to RADIUS Server



ACL Summary

Comparing the Rogue preventing ACLs needed with Server Provided MAC Addresses to the Rogue preventing ACLs needed with Network Provided MAC Addresses, one finds:

The ACLs used with Server Provided MAC Address are:

- Simpler
- Statically defined
 - Can be applied by normal Administration processes
 - Can have Management utilities build and apply the ACL

There is no need to invent a new ACL establishment protocol

There is no need to create a new snooping capability, etc.

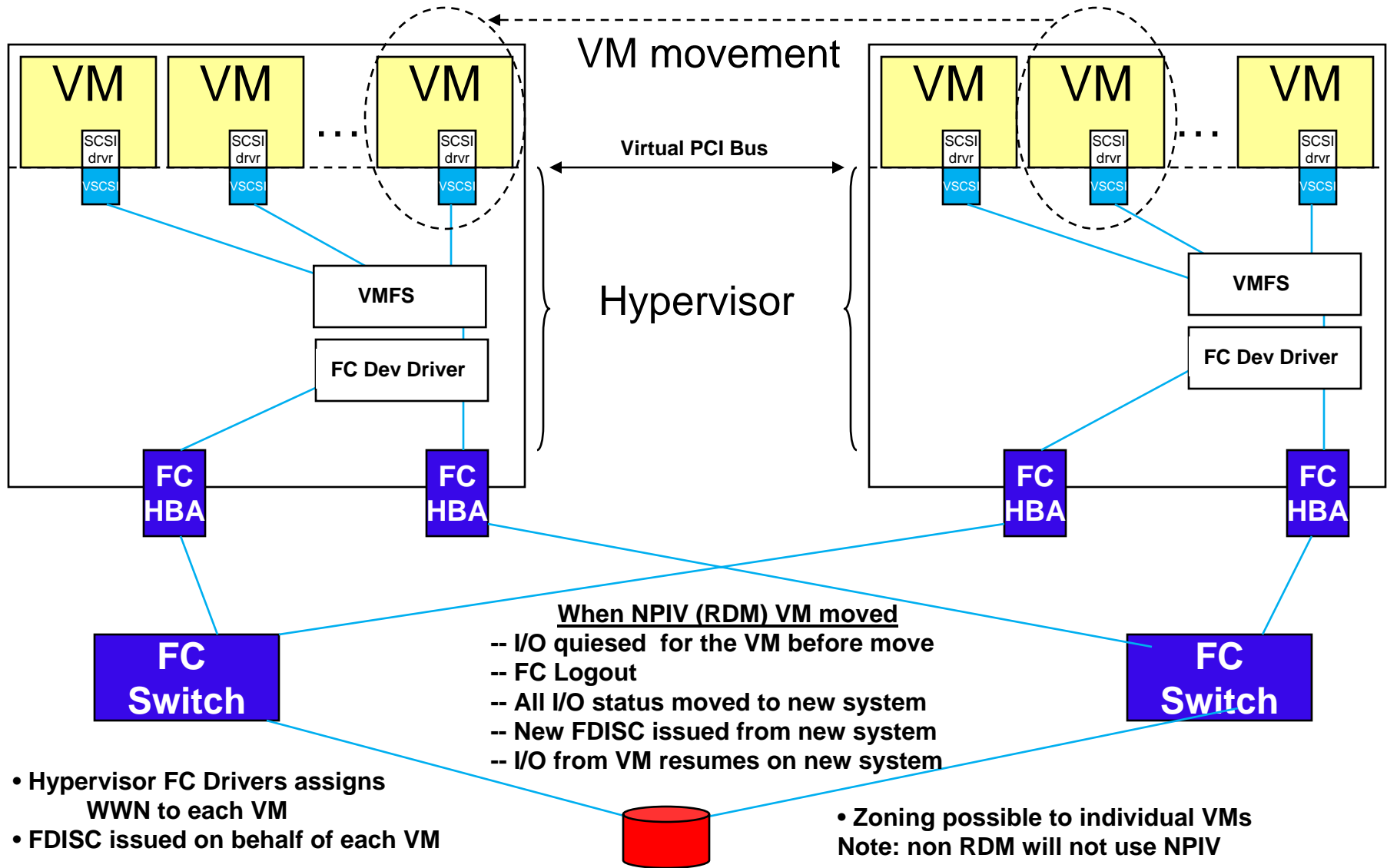
All the Ethernet Switch Processes are the same as today

May apply to Vswitch, in the future, but does apply to External Switches Today

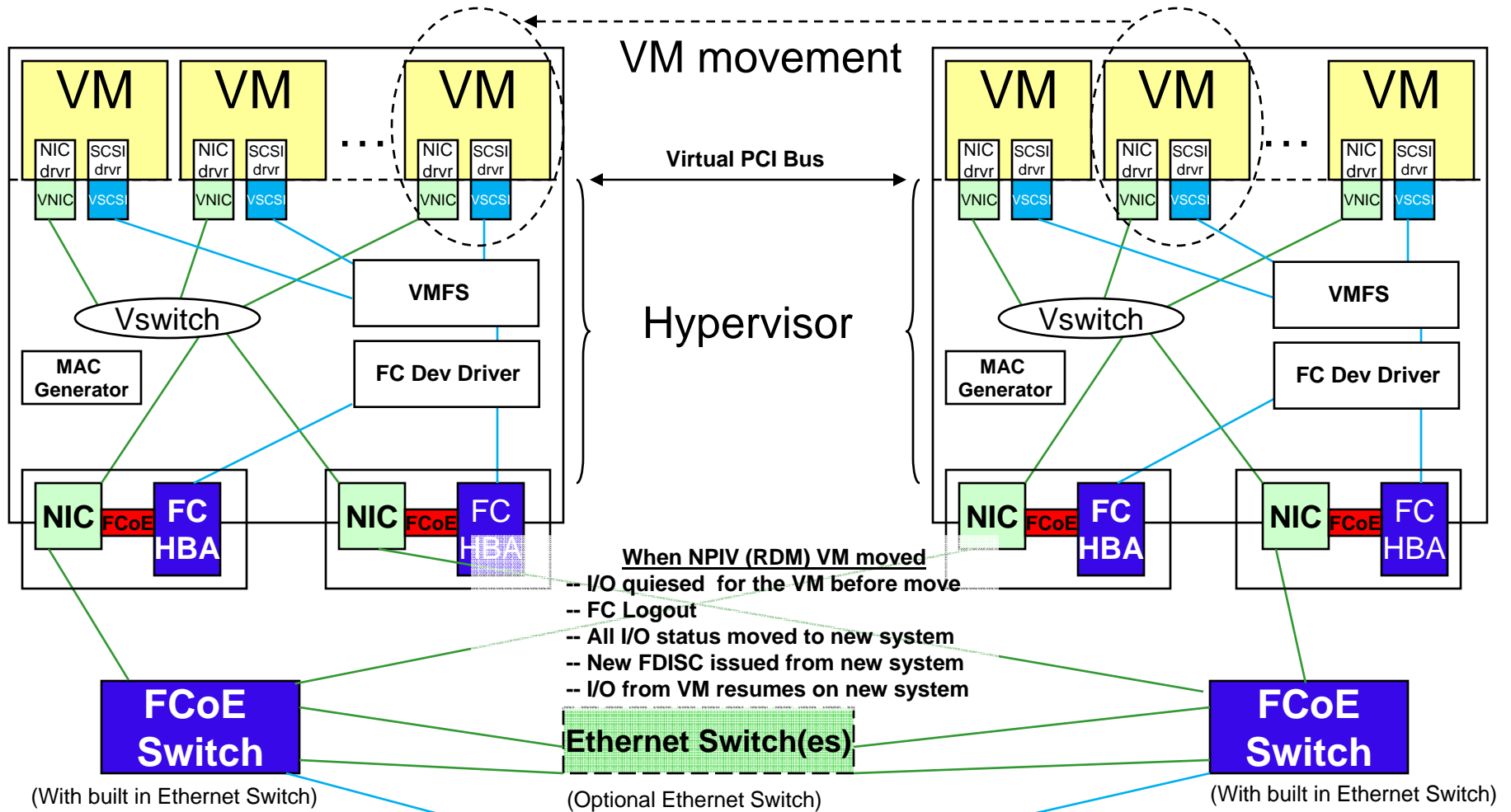
Additional Review



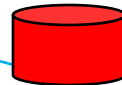
New VMware with Vmotion (NPIV FC based RDM)



VMware with Vmotion with NPIV & FCoE/NIC Adapters



HBA has a FC compatible interface



Uses "Burnt-in" MAC Address

Same configuration for both NON NPIV and NPIV support



The take away for FC and FCoE HBA & Vmotion

VMware uses VMFS (a shared file system) to simulate a LUN in a file for NON RDM LUNs

Whether VMFS LUN or RDM LUN is used, the normal FC interfaces are employed

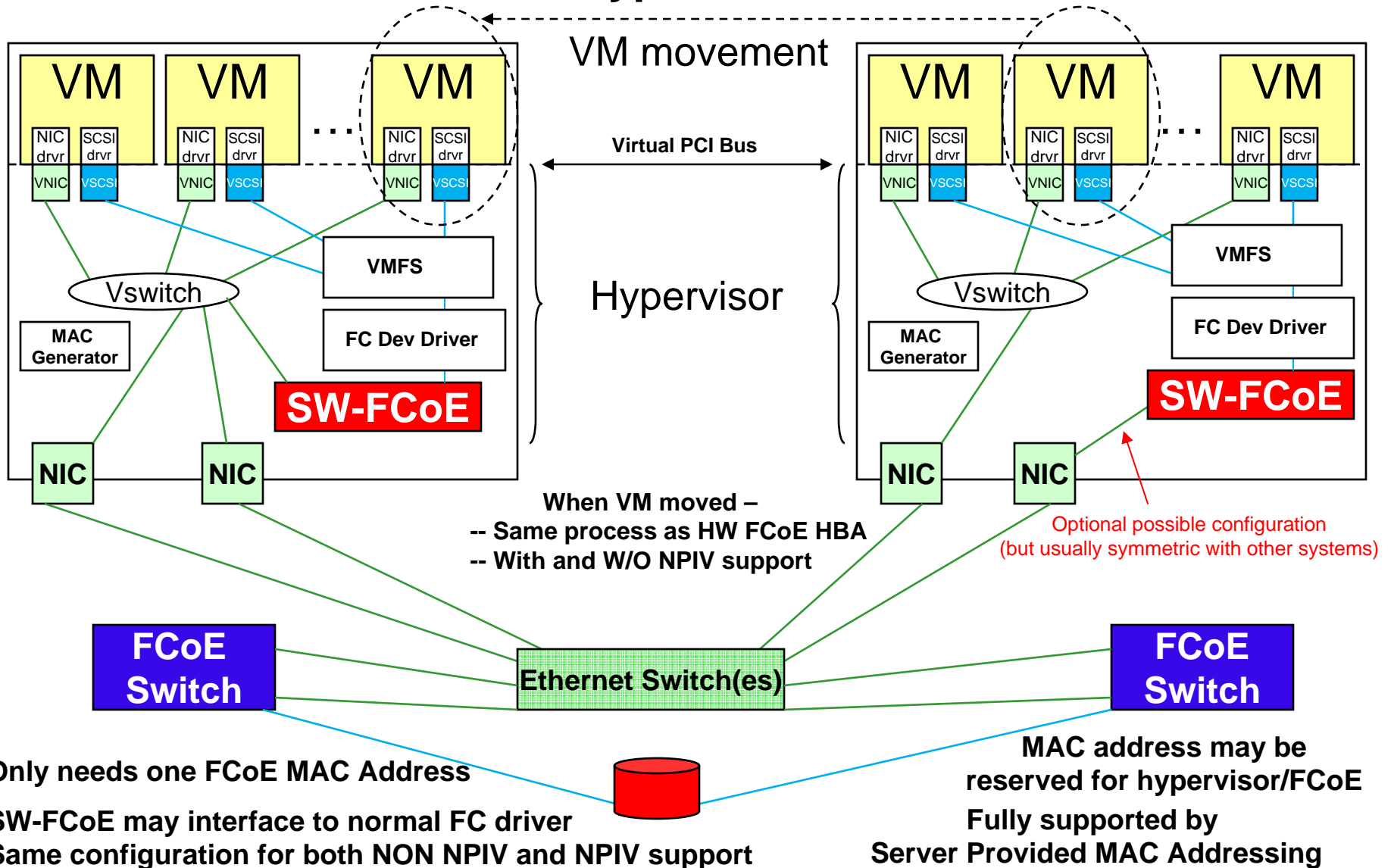
- For RDM -- NPIV support is possible
 - Where each VM is supported by a unique FDISC
- For non RDM no NPIV is used
 - Because the LUN is in the shared file system (VMFS)

SPMA supports the Hypervisor with the same interface as Real FC

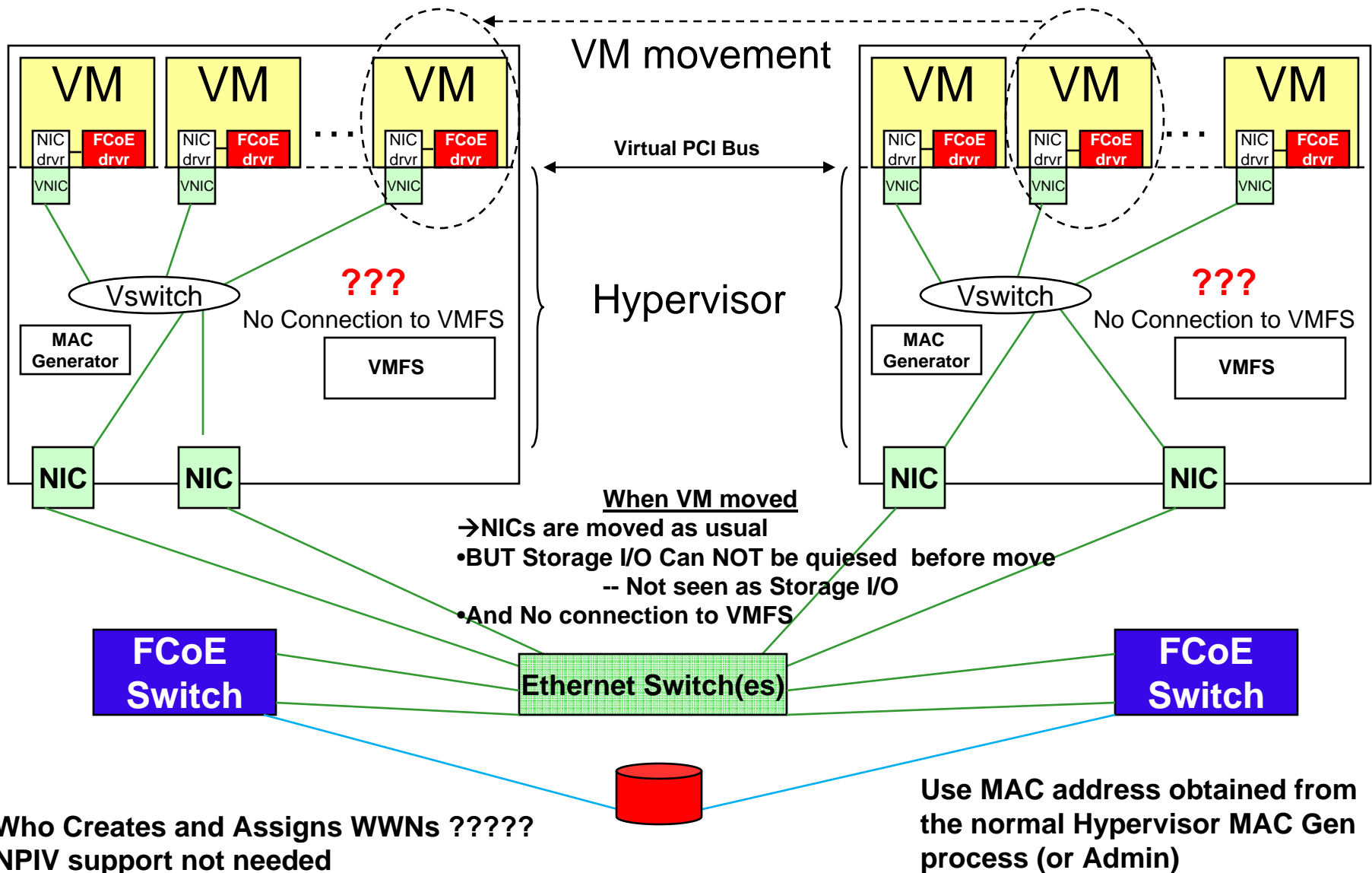
- Only the MAC address allocated to the FCoE function is needed
- The raw FC frame delivered from the FCoE entity has all the info needed (S_ID & D_ID) for routing within the device drivers/Hypervisor
- Additional MAC addresses do not help



Possible VMware with Vmotion & Software FCoE in Hypervisor



Suggested VMware with Vmotion & Software FCoE in the VMs



The Take Away regarding Software FCoE

The Hypervisor version of Software FCoE will be able to support both VMFS simulated LUNs and RDM real LUNs

Hypervisor version of Software FCoE only needs a single MAC address for FCoE functions

- This is an example of SPMA (Server Provided MAC Addressing)
- All the same processes used with Real FC, for Migrating a VM to another system, applies here

The VM based software FCoE is unlikely to be a production model

- Prevents Kernel knowledge of the I/O status
- Not compatible with VMFS (Shared File System)
- Questions on how the WWN will be assigned
- Regardless – SPMA still works by letting each SW FCoE use the VMs' virtual NICs



Overall Summary



Number of MAC addresses

NPIV uses virtual N_Ports in the numbers of hundreds

Today's NICs provide support for multiple MAC address

- But the numbers are no where near this quantity
- Typically just a few

Also, this increases the number of addresses that the LAN switches must handle (and that needs to be managed)



Implementation complexity

Extra steps that Mapped Addressing would require of a VF port during discovery/initialization

- That is, once the FC-MAP is received by the VN port in the FCF Advertisement and the FC_ID is received in the FLOGI response, the VF port must derive the mapped address, register an additional address in the NIC, and optionally send a gratuitous ARP to announce the new MAC address
- Every time the fabric login is removed, the VF port may optionally reclaim that MAC address with an additional gratuitous ARP
- This adds further complication, expense and room for error on the part of every VN port and VF port implementation
- There can be large numbers of these when doing NPIV

The FCoE Entity must be made aware of N_Port IDs received by the HBA FC Entity in an FLOGI/FDISC responses in order to construct the source MAC address for transmitted FCoE packets, rather than simply using the server assigned MAC address



Security concerns

Nuova pointed out the probability of duplicate MAC addresses when 2 fabrics are accidentally joined through an Ethernet connection and presented how the combination of mapped MACs and dynamic Ethernet ACLs can prevent this from causing a problem

- FC-MAP is assigned by an administrator, the likelihood of duplicate MAC addresses significantly increases when 2 fabrics that were never intended to be connected to each other are accidentally joined through an Ethernet fabric
- There is going to be a default value for this FC-MAP and if the fabrics are never "intended" to be connected, then they will likely use the same default
- In the absence of the ability to dynamically change Ethernet ACLs and snoop FLOGI messages to protect against this (not an available function today), the problem can be solved using Server assigned MAC addresses and static ACLs

Security concerns around duplicate MAC address, rogue hosts, etc. already exist in today's LAN environments

- The problems are better understood and many solutions are available
- Used mapped address breaks the security model that is understood since MAC addresses are now dynamically assigned
- Slide set T11/07-688v0 show how normal checking rules eliminate the corruption/Looping issues



Security concerns

(continued)

Implementing security with mapped addresses **requires** FCoE awareness in Ethernet switches which they do not have today (snooping FLOGI messages, and dynamic ACL updates)

Many of the security arguments used by the mapped address assume that security for other protocols is less important; i.e. an administrator would tolerate a rogue host for IP traffic, but not for FCoE

The requirement for new switches, for security, will impact the acceptance of FCoE for maybe years

Denial of Service (DoS) attack on Discovery will shut down all new connections if no Static ACLs are defined for the protection of the Discovery process.

The same ACLs that protect the SPMA, also protects the Discovery process.



FC-MAP: Architecturally limiting

- VF (Virtual Fabric) IDs and FC-MAPs are supposed to be orthogonal
- We will need a protocol/process/procedure to ensure FC-MAP uniqueness and map it to a VF ID – Not yet identified.
- Even if we do that – VF ID is a 12bit field it needs to map to an 8bit field in FC-MAP to identify fabrics. How can uniqueness be enforced across all VF IDs?
- FC-MAP architecturally limits the number of FCoE virtual fabrics. Fewer than FC!!



Use with Virtualizers

Since the FC needs to be supported

- For Vmotion
- For NPIV
- For Vmotion with NPIV
- For SR-IOV
- For SR-IOV with NPIV
- For Vmotion with SR-IOV and NPIV

If we make FCoE work like FC with FC drivers then all of the above will work with FCoE

No new MAC addressing paradigms are needed; SPMA is exactly what is needed



SPMA vs NPMA

Category	SPMA	NPMA
Architecture	Preserves addressing architecture used by all existing L3 protocols. Does not limit the number of Virtual Fabrics	Changes the way addressing is done. Uses local addresses on a large scale. What if some other L3 protocol wanted to use local addresses? Also limits number of VFs
Uniqueness	Usually guaranteed at manufacture time.	Burden is on administrator to ensure that FC-map does not conflict with something else.
Identifying an VN_Port to VF_Port link	A single pair of MAC addresses.	In the presence of NPIV, multiple pairs of MAC addresses represent a single logical FC link.
Number of addresses to be supported in an adapter	1-2; with VMWare it could be up to n+1; n being the number of LAN/IP address for VMs.	Up to ~2x that of SPMA; one extra one for each VM with NPIV (one for each FLOGI & FDISC).
Switch forwarding table requirements	No different than without FCoE.	Requires ~2n address for NPIV or SW-FCoE running in the VMs. (n=# of VMs)
Additional steps during fabric login/logout	None.	Program NIC to receive on each new address after FLOGI and optionally send gratuitous ARP. After logout, address must be removed from NIC, and FCF optionally "reclaims" the address by sending a gratuitous ARP. Additional complexity and room for operational error to high-volume parts (end systems).
Security concerns	Security issues are well understood and are solved by value-add functions. FCoE does not create any new security risks in these environments.	Imposes an entirely new security architecture because addresses of end systems are ephemeral. Static provisioning is ruled out. Delays acceptance of FCoE



Thank You

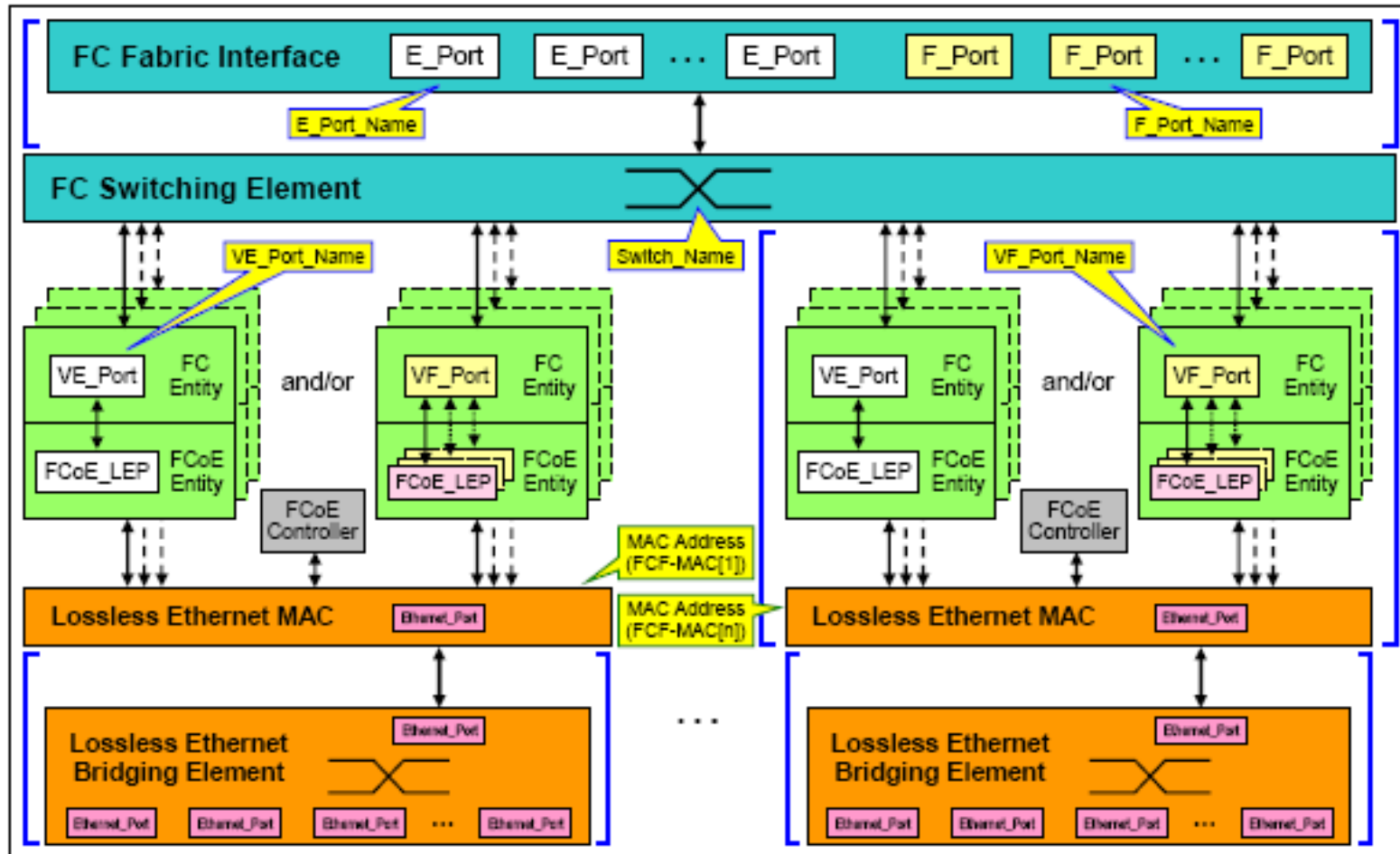


Backups

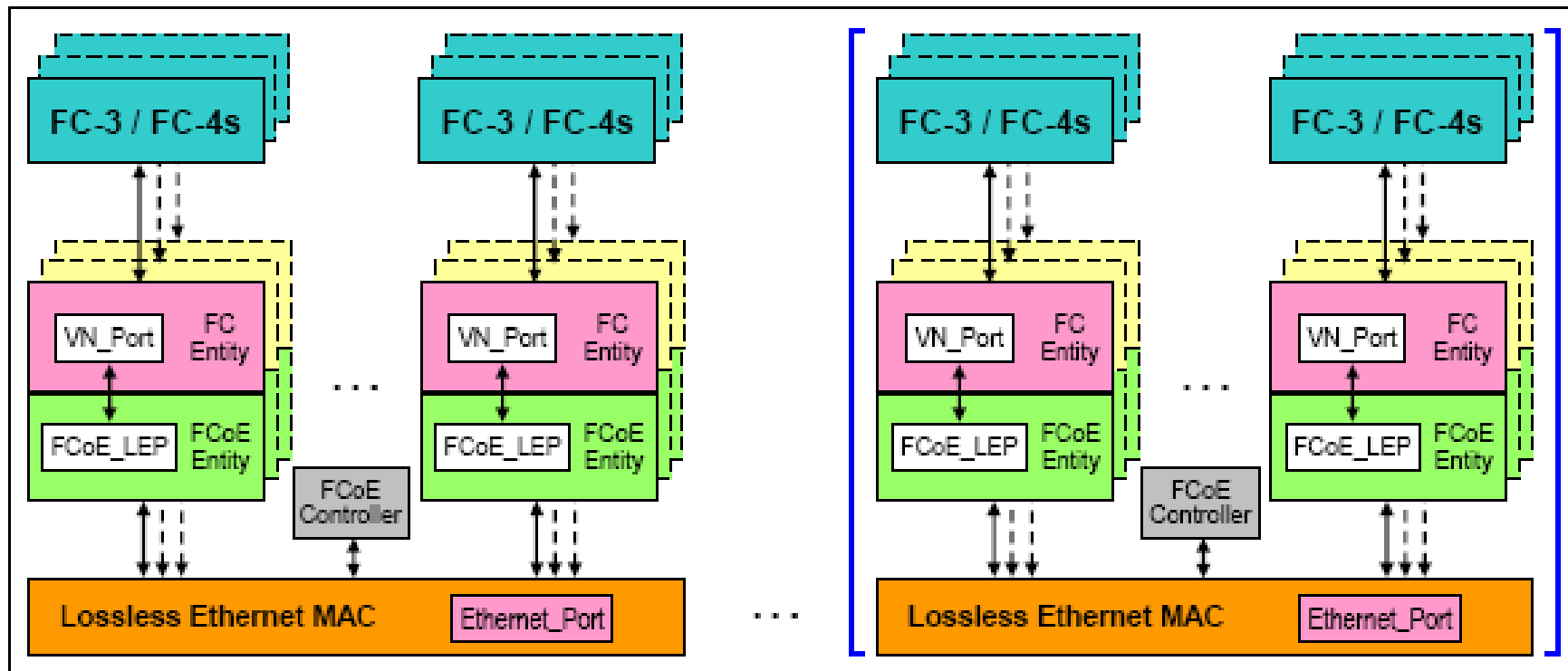
**(Slides from the previous presentations
T11/07-656v0 & T11/07-651v2)**



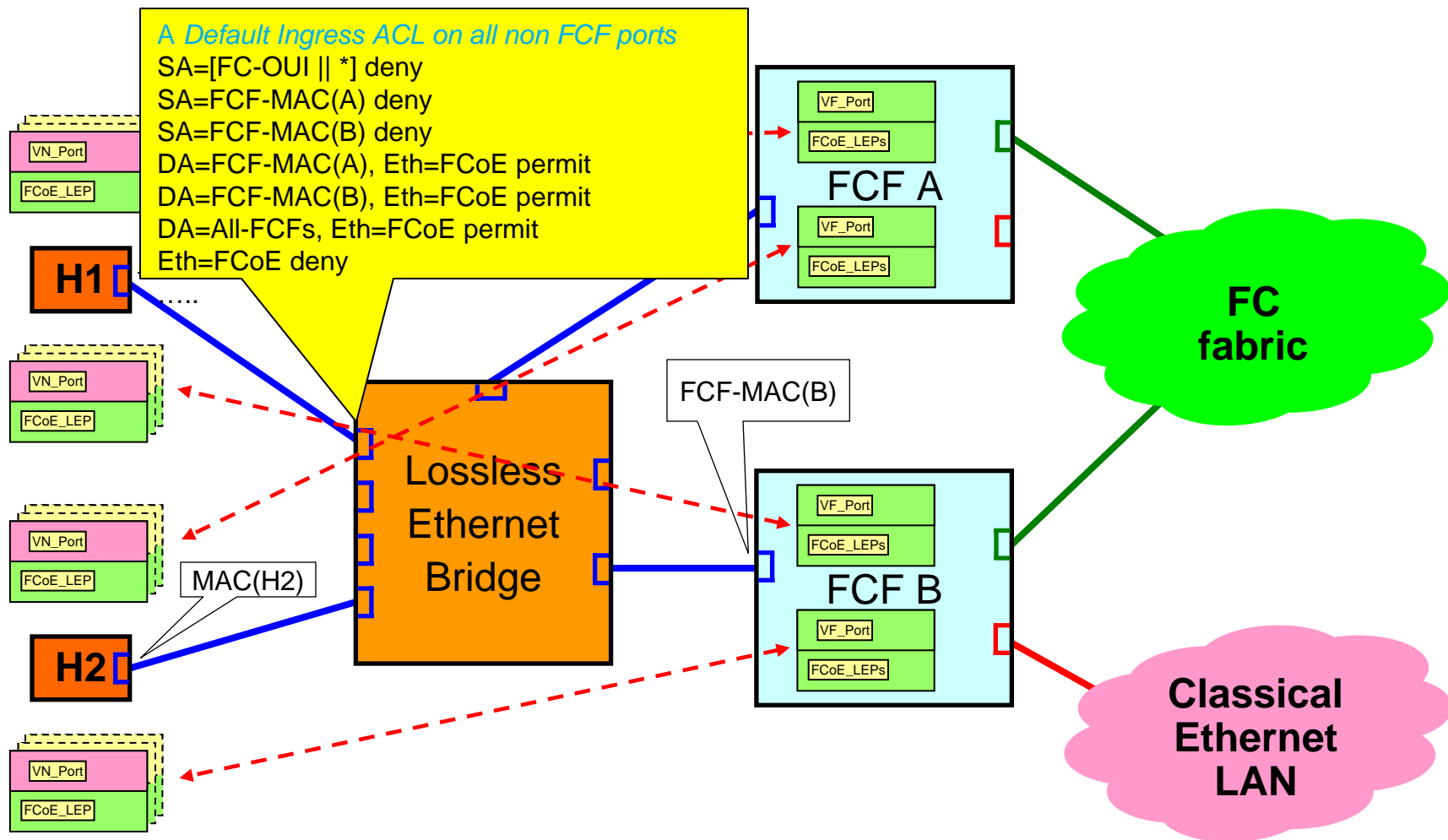
FCoE VE_Port/VF_Port Functional Model



FCoE VN_Port/ENode Functional Model



Generic Topology ACLs (with Network Provided MAC Addresses)



Default MAC layer ACL (with Network Provided MAC Addresses)

Default Ingress ACL on all non FCF ports! (as shown in T11/07-546v0)

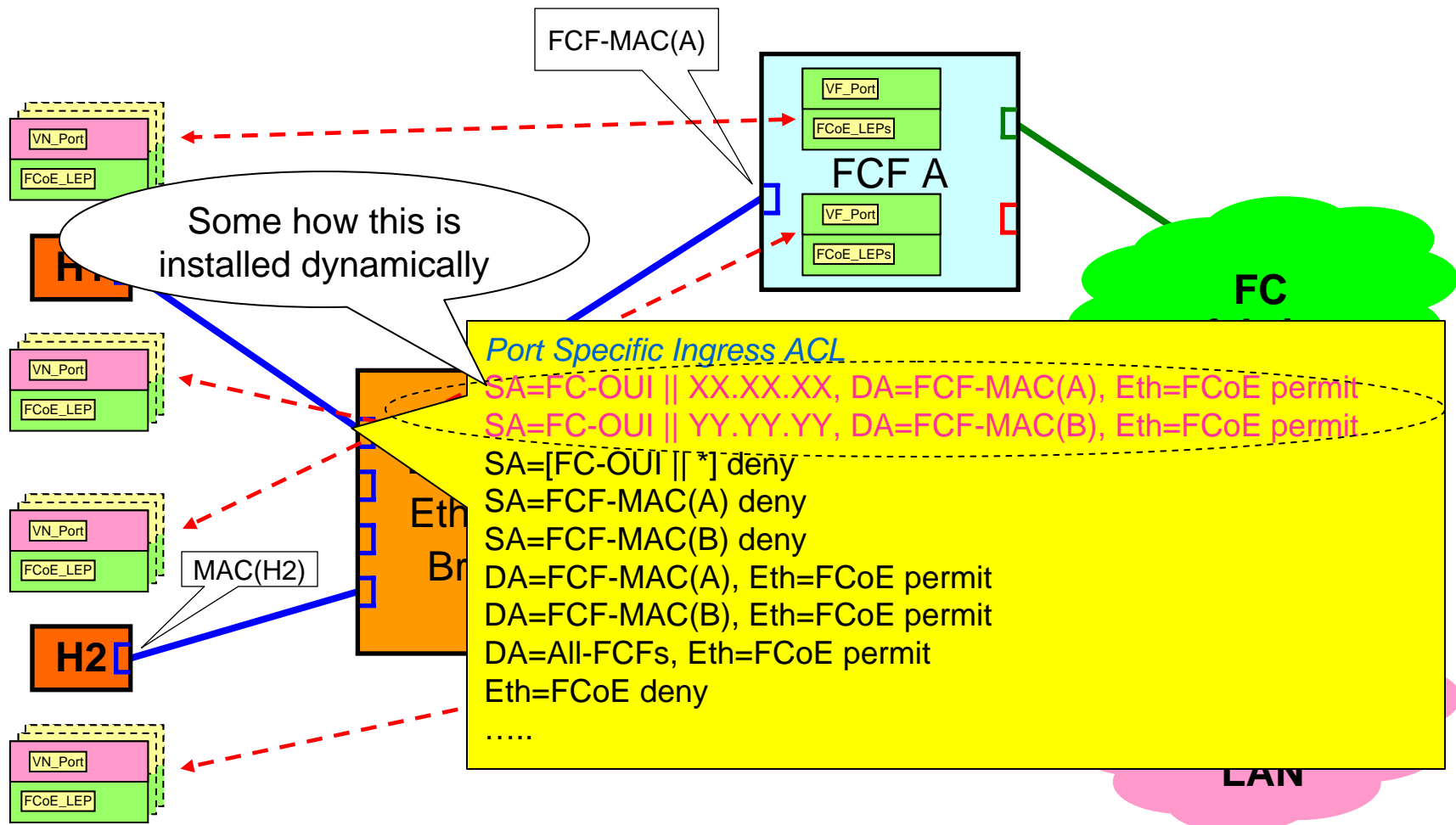
- SA=[FC-OUI || *] deny {Protect Mapped MAC addresses }
- SA=FCF-MAC(A) deny {Protect one of the FCF burned-in MAC addresses}
- SA=FCF-MAC(B) deny {Protect one of the FCF burned-in MAC addresses}

... Note: A Rogue could spoof the Switch Learning on SA & impact Discovery with a normal Ping

- DA=FCF-MAC(A), Eth=FCoE permit {Enables FLOGI }
- DA=FCF-MAC(B), Eth=FCoE permit {Enables FLOGI }
- DA=All-FCFs, Eth=FCoE permit {Enables Discovery Solicitation}
- Eth=FCoE deny {Disable all other FCoE traffic }
- {Non FCoE related entries (e.g. IPv4) }



Generic Topology Dynamic ACLs (with Network Provided MAC Addresses)



Dynamically updated MAC layer ACL (with Network Provided MAC Addresses)

Port specific Ingress ACLs (somehow created dynamically by the FCFs) (as shown in T11/07-546v0)

SA=FC-OUI || XX.XX.XX, DA=FCF-MAC(A), Eth=FCoE permit {Dynamically added to top of ACLs}

SA=FC-OUI || YY.YY.YY, DA=FCF-MAC(B), Eth=FCoE permit {Dynamically added to top of ACLs}

... **Note: there could be 100s of those!!! One for each FLOGI & FDISC**

SA=[FC-OUI || *] deny {Protect Mapped MAC addresses}

SA=FCF-MAC(A) deny {Protect “n” FCF burned-in MAC addresses}

...

SA=FCF-MAC(B) deny {Protect “n” FCF burned-in MAC addresses}

... **Note: A Rogue could spoof the Switch Learning on SA & impact Discovery**

DA=FCF-MAC(A), Eth=FCoE permit {Enables “n” FLOGIs / FDISCs}

...

DA=FCF-MAC(B), Eth=FCoE permit {Enables “n” FLOGIs/ FDISCs}

...

DA=All-FCFs, Eth=FCoE permit {Enables Discovery Solicitation}

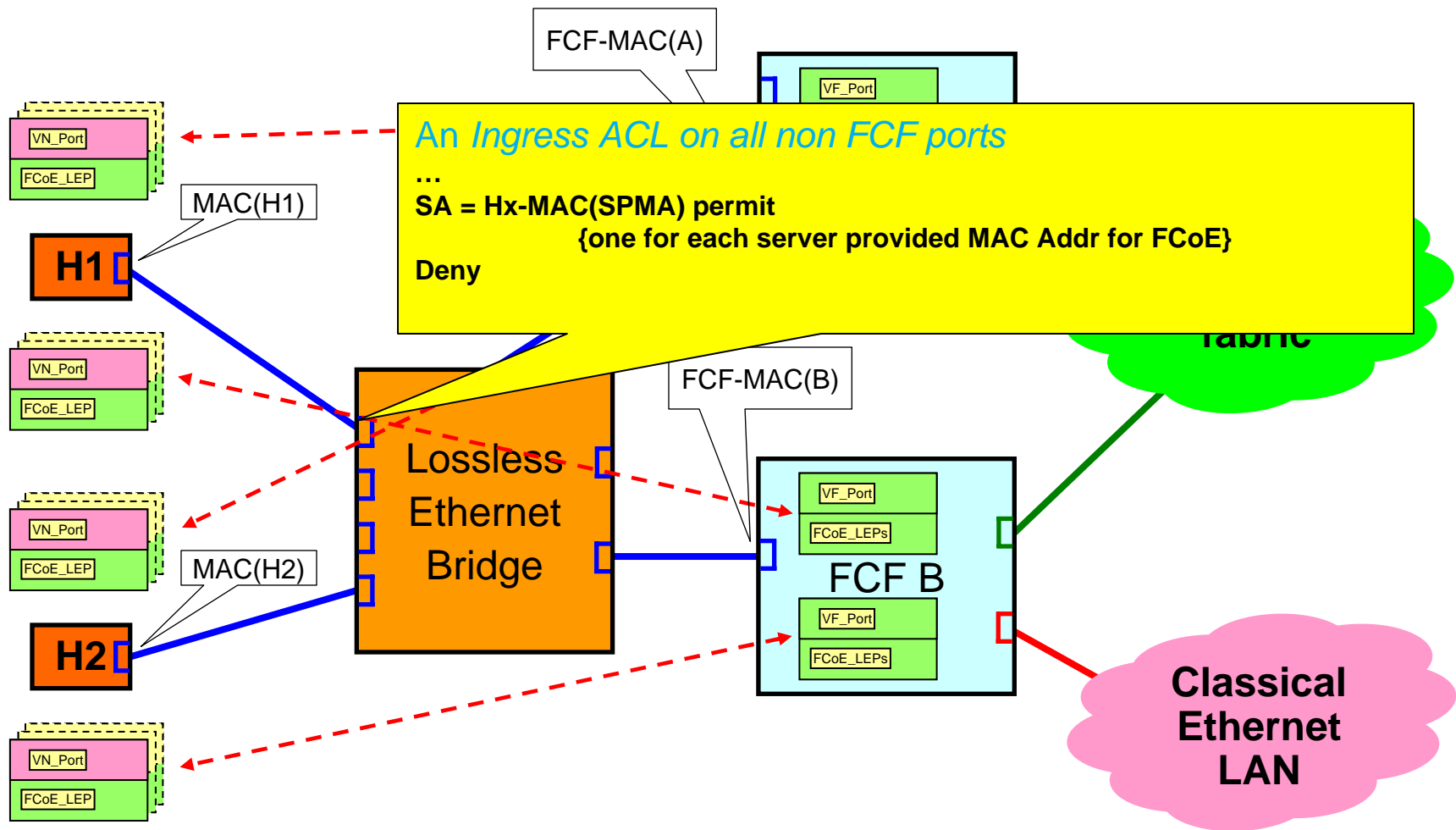
Eth=FCoE deny {Disable all other FCoE traffic}

.....

{Non FCoE related entries (e.g. IPv4)}



Generic Topology (simple) ACLs (with Server Provided MAC Addresses [SPMA])



The really simple Ethernet Switch ACL (with Server Provided MAC Addresses [SPMA])

Ingress ACL on all non FCF ports!

... {Other normal ACLs }

SA = Hx-MAC(SPMA*) permit

{one for each Server Provided MAC Addr (SPMA), for FCoE, assigned to that switch port}

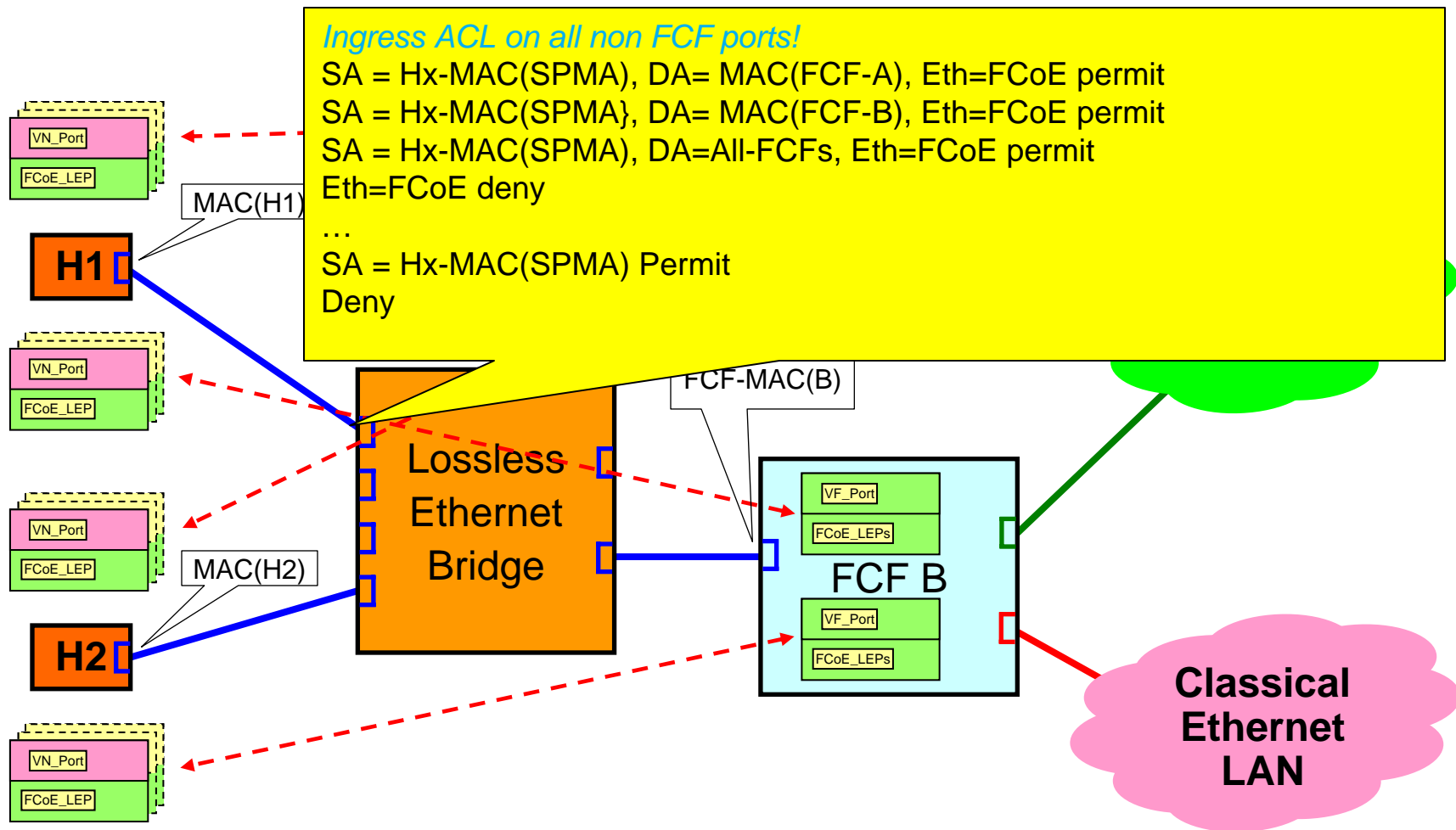
Deny

Note: The above assumes that if the SA is valid, other stuff is OK

***SPMA = Server Provided MAC Addr**



Generic Topology & more sophisticated ACLs (with Server Provided MAC Addresses [SPMA])



A More sophisticated Ethernet Switch ACL (with Server Provided MAC Addresses [SPMA])

Ingress ACL on all non FCF ports!

SA = Hx-MAC(SPMA), DA= MAC(FCF-A), Eth=FCoE permit {Permits access to FCF-A}

SA = Hx-MAC(SPMA), DA= MAC(FCF-B), Eth=FCoE permit {Permits access to FCF-B}

SA = Hx-MAC(SPMA), DA=All-FCFs, Eth=FCoE permit {Permits discovery of FCFs}

Eth=FCoE deny {Disable other FCoE access}

... {Other normal ACLs}

SA = Hx-MAC(SPMA) Permit {Permits any xmits from SA}

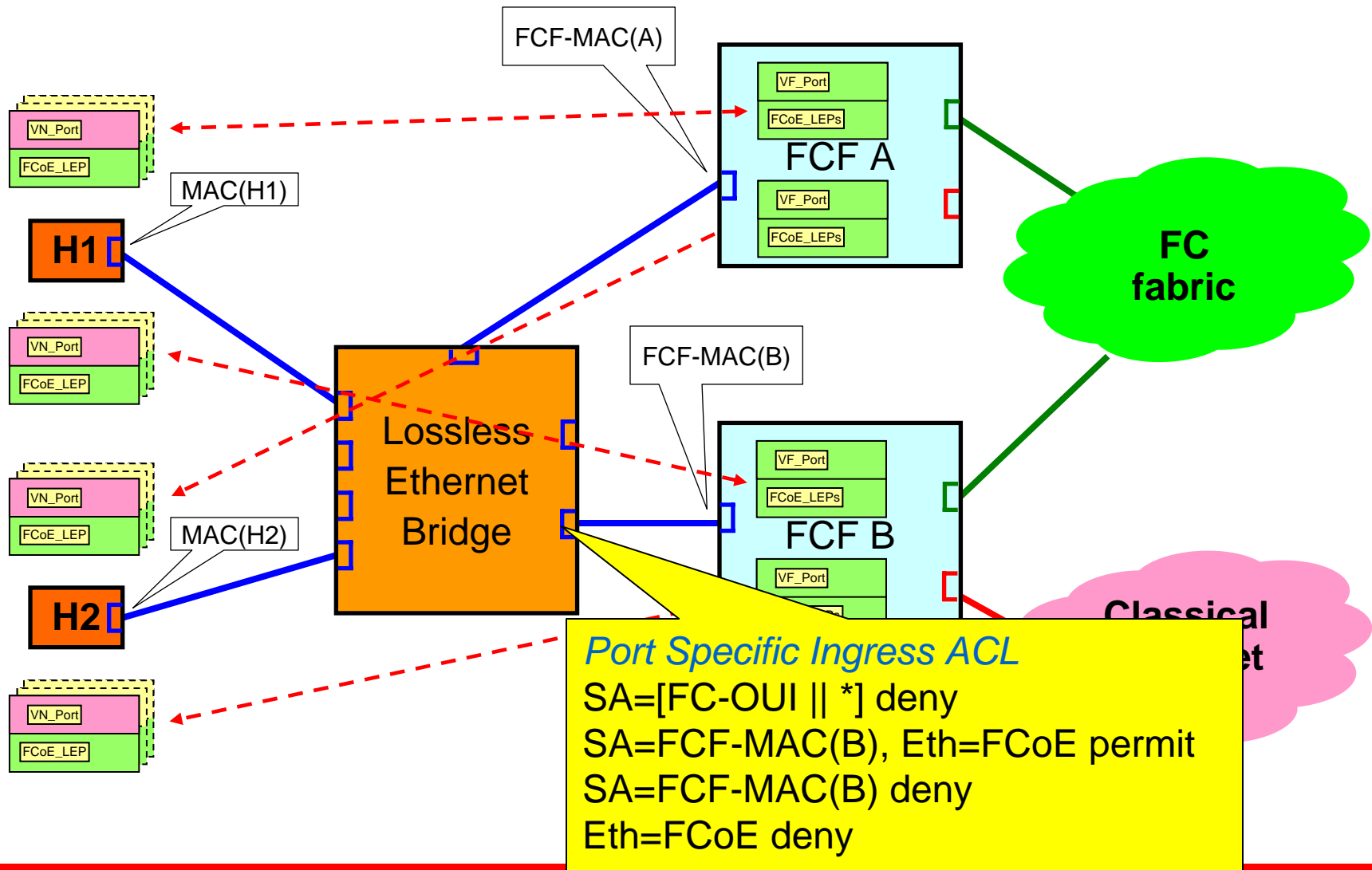
Deny {Prevents use of other SA}

Note: The above makes sure that Ethernet Rogues do not get access to the “Data Center” Ethernet switches and confuse the learning in the switches

Fibre Channel ACLs and Zoning is part of the FC layer in FCoE not Ethernet



Ingress ACLs for FCF Ports (with Network Provided MAC addresses)



Default Ingress ACLs for FCF Ports (with Network Provided MAC addresses)

Port Specific Ingress ACL (as shown in T11/07-546v0)

SA=[FC-OUI || *] deny {Denys all incoming Post FLOGI operations}

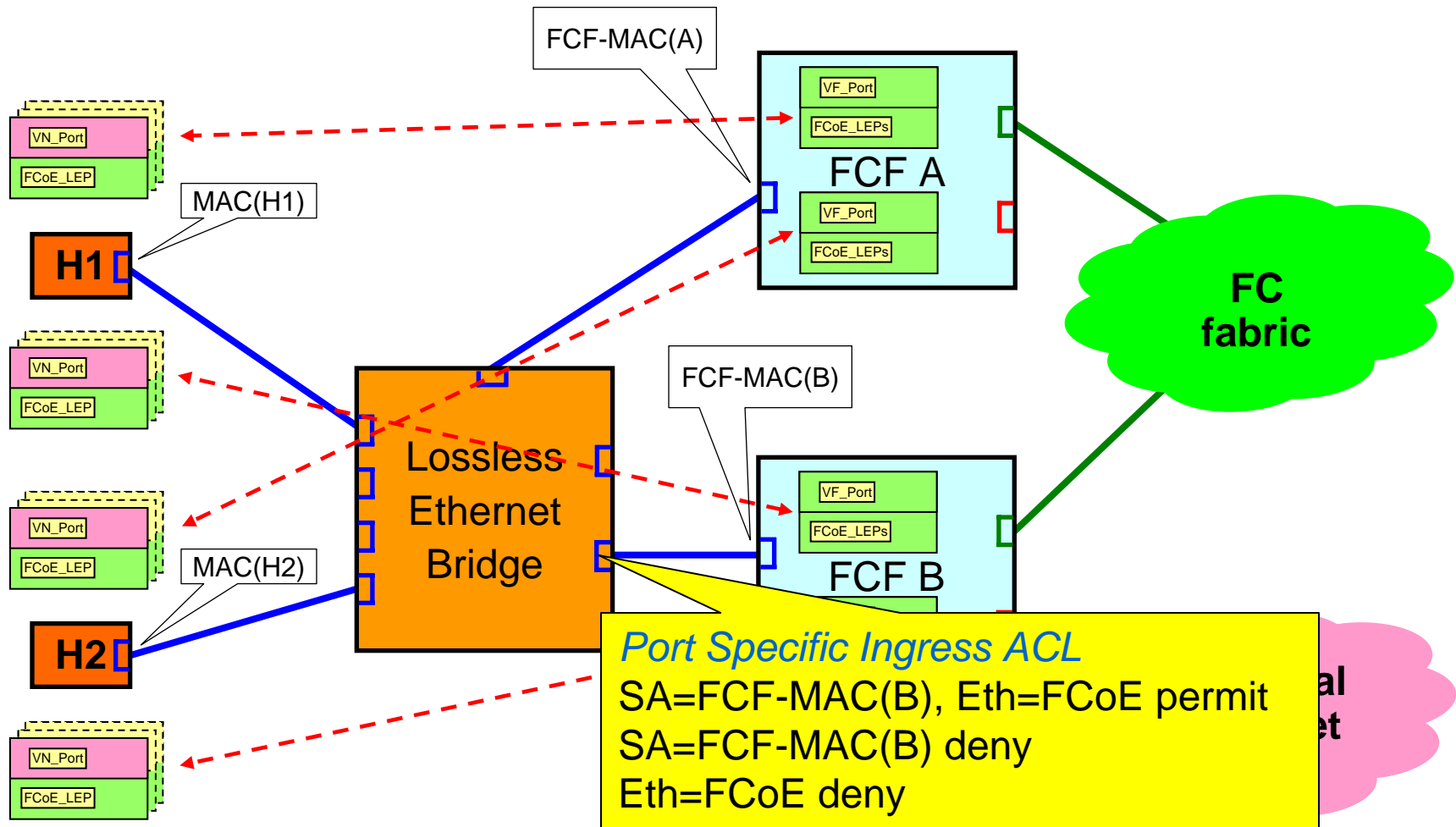
SA=FCF-MAC(B), Eth=FCoE permit {Permits FCoE operation from that SA}

SA=FCF-MAC(B) deny {Prevents any non FCoE with that SA}

Eth=FCoE deny {Prevents any other FCoE operations}



Ingress ACLs for FCF Ports (with Server Provided MAC addresses)



Default Ingress ACLs for FCF Ports (with Server Provided MAC addresses)

Port Specific Ingress ACL

SA=FCF-MAC(B), Eth=FCoE permit

{Permits FCoE operations from that SA}

SA=FCF-MAC(B) deny

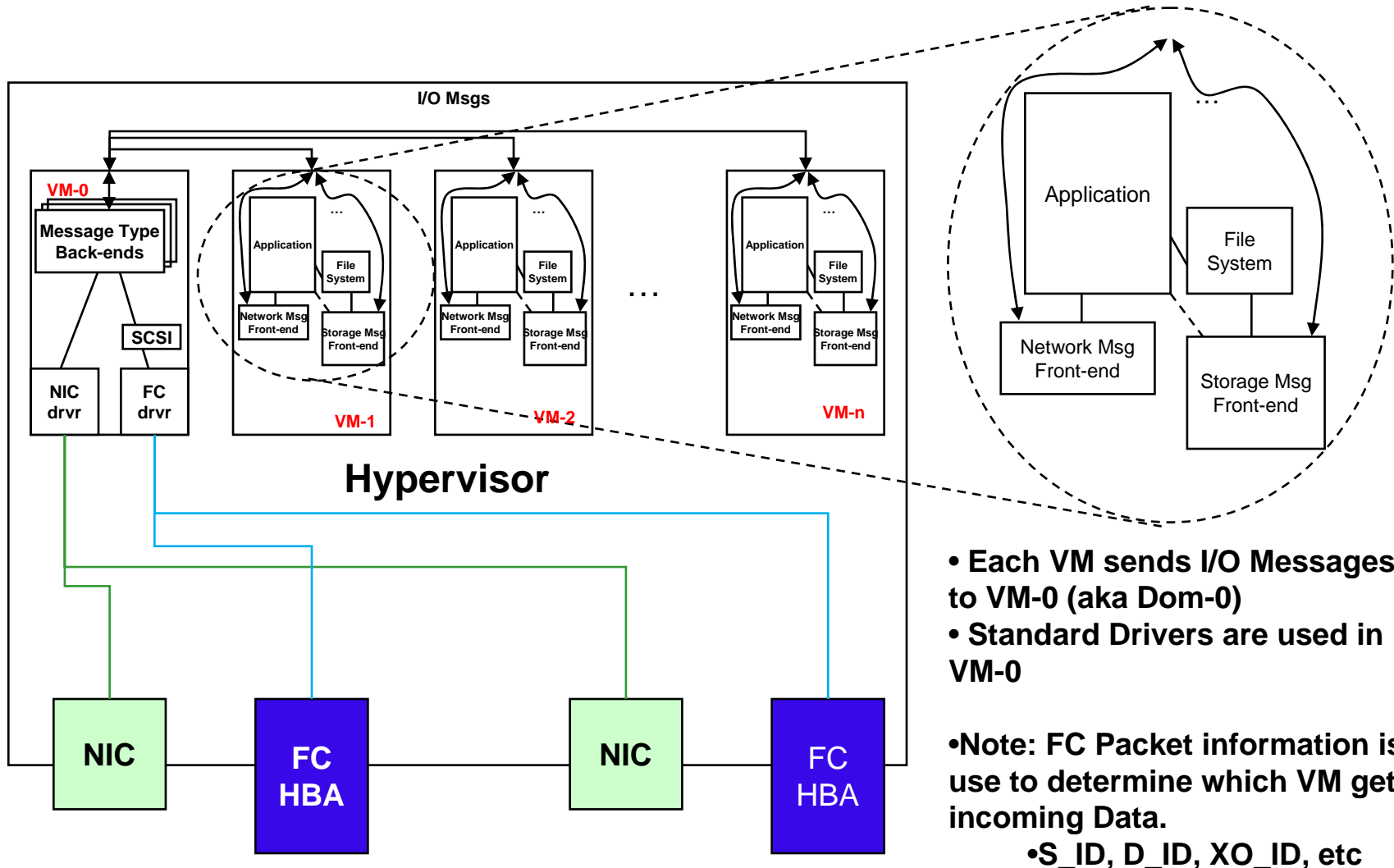
{Prevents any non FCoE with that SA}

Eth=FCoE deny

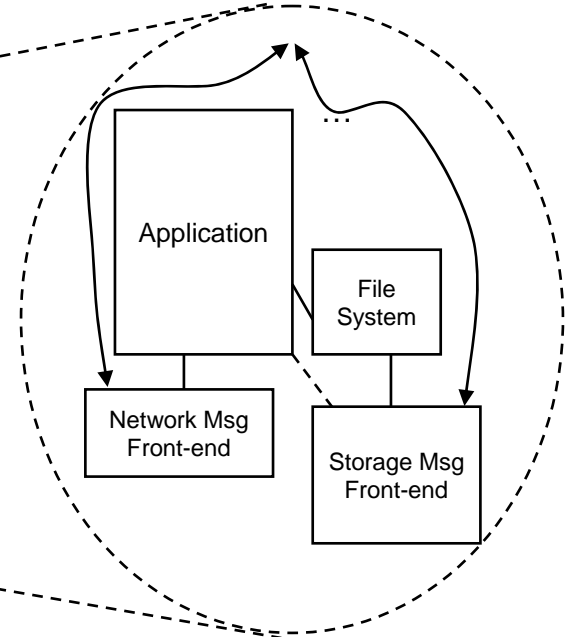
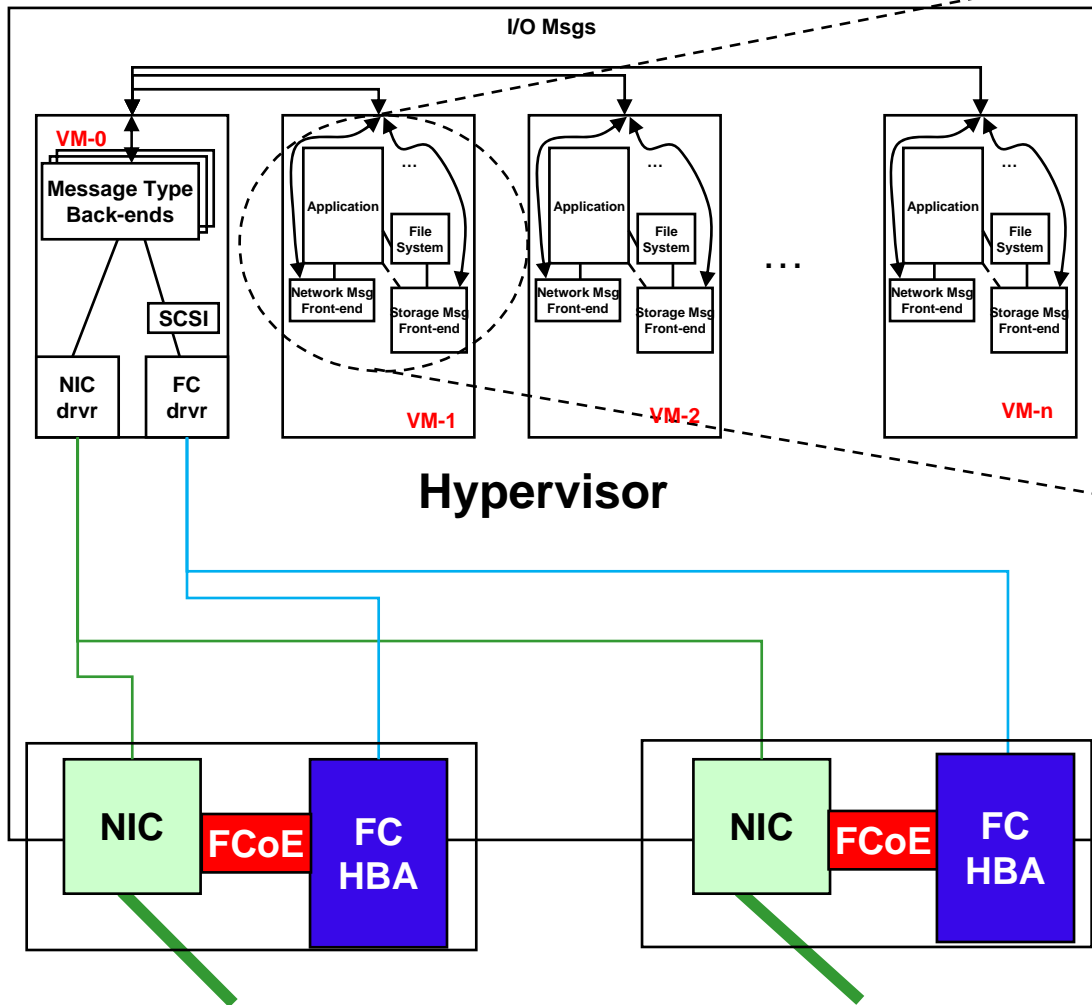
{Prevents any other FCoE operations}



Virtualization as seen in Xen, Microsoft, etc.



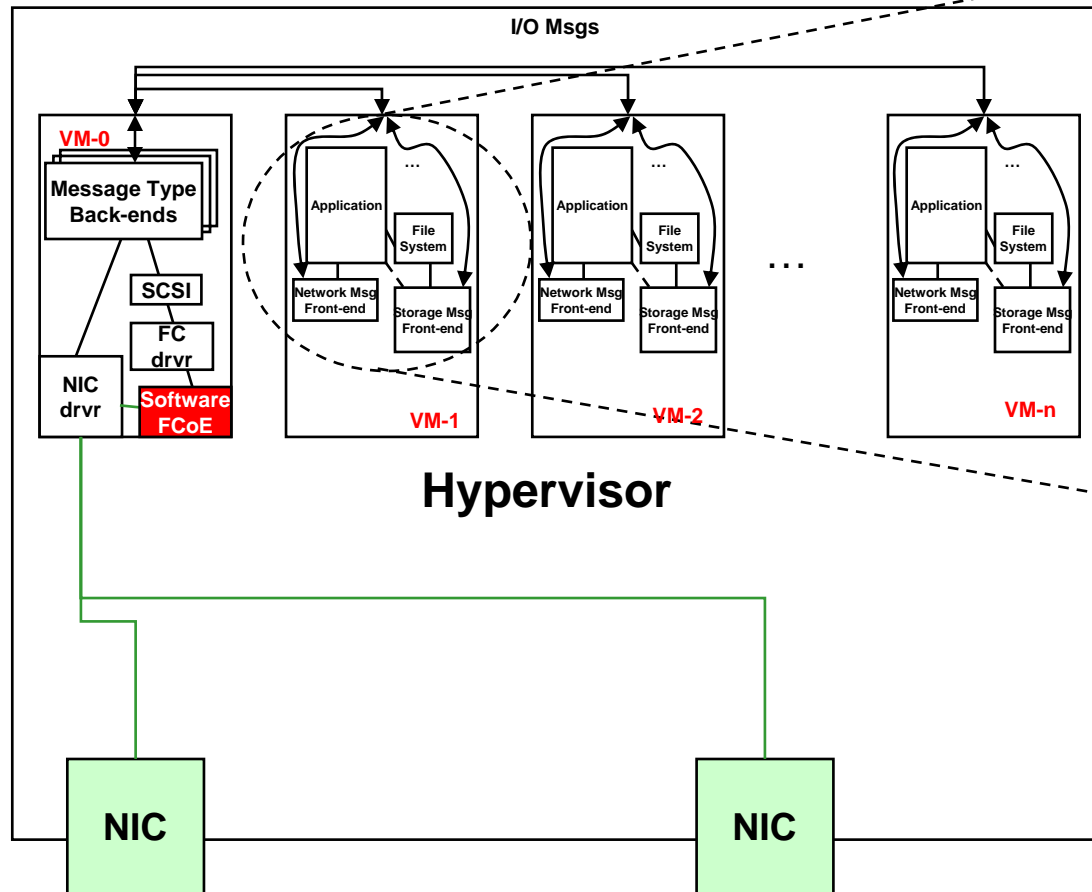
Xen, Microsoft, etc. Virtualization (with FCoE HBAs)



- Continues to use the real FC Device Drivers
- Does not get involved in the use of a “Burnt-in” MAC Address used by the Adapter
- The same real FC NPIV capable drivers are used for FCoE
- S_ID, D_ID, XO_ID, etc is used to route to appropriate VM



Xen, Microsoft, etc. Virtualization (& Software FCoE)



- Software FCoE implemented in VM-0
- Can use a “Burnt-in” MAC Address
- Routing to appropriate VM is the same as real FC
 - Uses S_ID, D_ID, XO_ID, etc is used to route to appropriate VM (Not MAC address)