

Flexible FDF Connectivity

Siamack Ayandeh
Chief Architect (Datacenter)
HP Networking
Sept 21st, 2011
Siamack@HP.Com



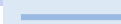
Summary

- What is the goal of flexible FDF connectivity
- Why is this goal important and should be a BB6 requirement
- Dist-FCF Constraints
 - Past attempts in BB6 to deal with these constraints
- Flexible FDF Connectivity as a potential solution

Desired Goal

I/T

Initiator/Target

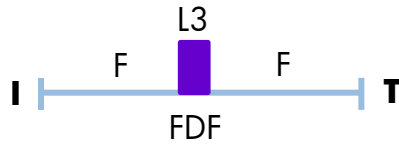


L2 DCB path



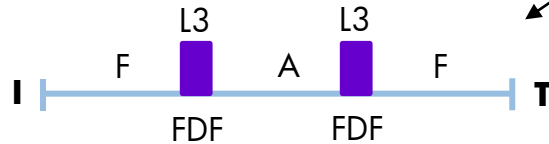
L3 DID lookup and possible hard zoning

I/T on same FDF

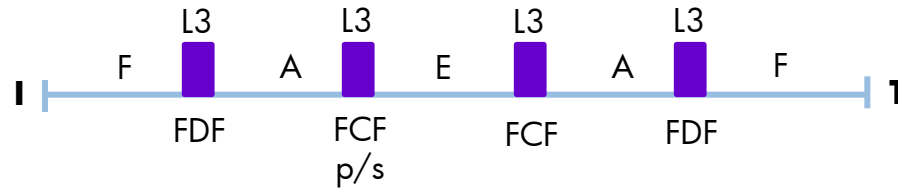


Dist-FCF Today

I/T in same VD



I/T in different VD



I/T in different VD



Flexible FDF Connectivity

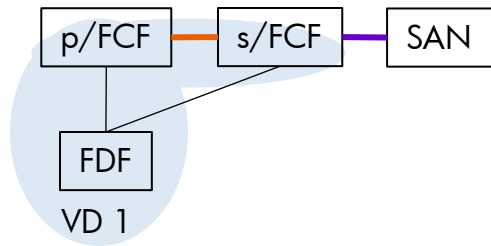
Goal of Flexible FDF Connectivity is to build on what has been done and add capability

FDF Trunking Constraint

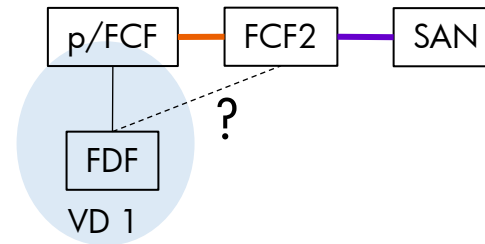
VA_Port — VA_Port

VE_Port — VE_Port

FC SAN —

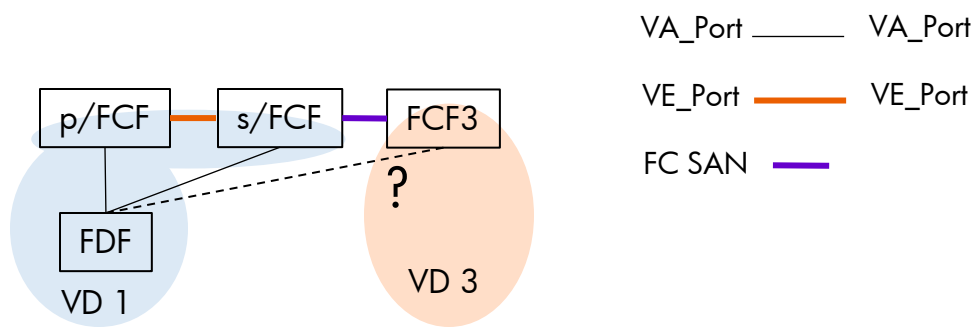


s/FCF absent



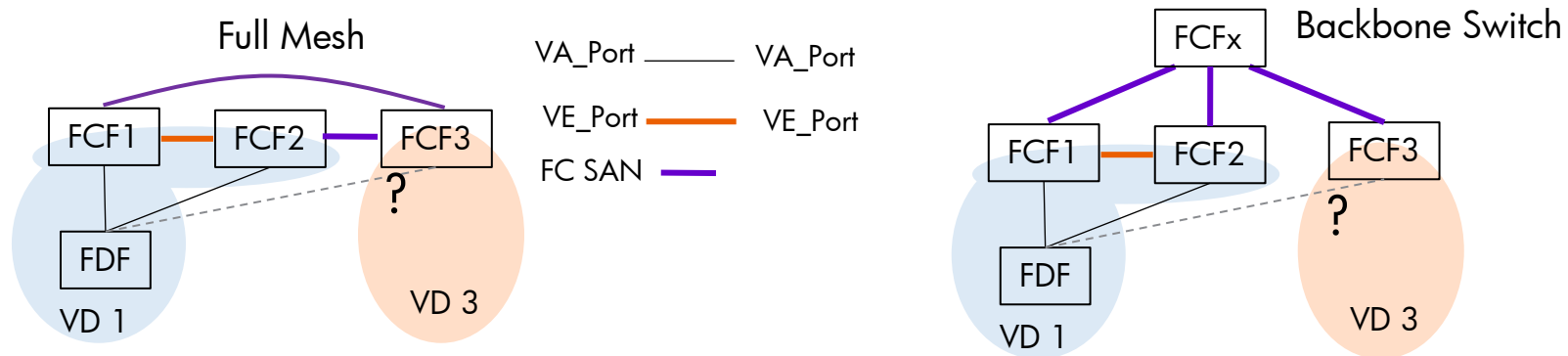
- For a virtual domain VD1; a primary and secondary FCF can be defined; control is active/standby; *forwarding is active/active* through p/s FCF (right?)
- However a s/FCF is optional
- In the absence of a control plane in synch secondary/FCF, a forwarding path in the form of a VA_Link from FDF to FCF2 is still desirable

FDF Trunking Constraint



- Virtual Domain 1 has a p/FCF and s/FCF; FDF can load balance across p/s FCFs
- Can A_Link between FDF and FCF3 form? No, not currently
 - A_Ports only live within a virtual domain and E_Port from FDF is not allowed
- Currently an FDF is constrained to connect to two FCF aggregation devices
 - Current generation of edge devices have anywhere from 6 to 16 trunking ports
- Previous proposals to deal with this constraint were not pursued for good reason:
 - Queue of backup FCF's: this is simply a different form of virtual switch idea i.e. stacking across FCFs (not suitable subject for standardization IMHO); don't want to make HA a crutch for routing
 - E_port light that requires changes to FSPF and I will discuss in more detail in a minute

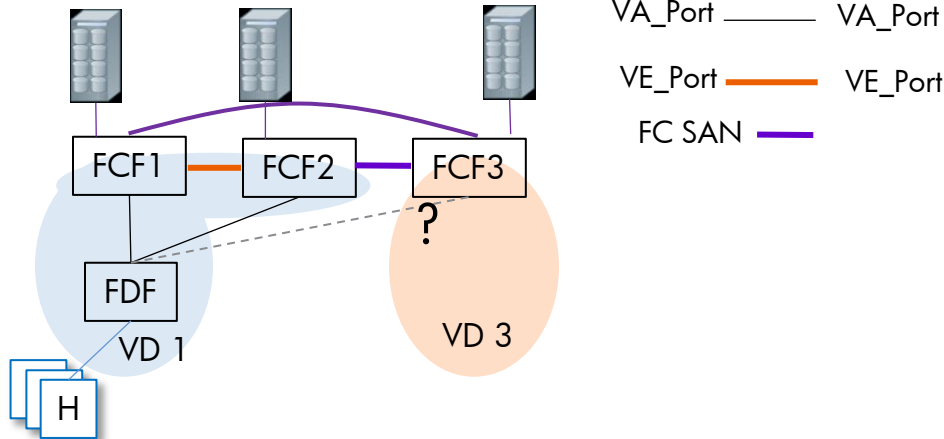
Limitation on FDF trunking adds a layer to the hierarchy



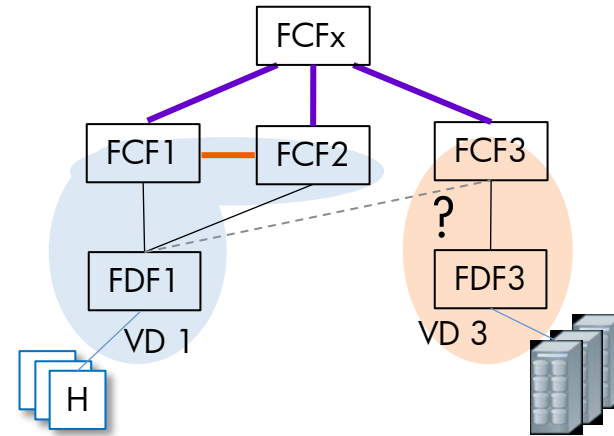
- In the absence of FDF to FCF3 VA_Link, there are two connectivity options:
 - Fully (or partial) Mesh the FCFs
 - And/or add a layer of switching as backbone to FCF's to extend the hierarchy
- These options:
 - Go against flattening of the network;
 - Constrain the topology with FCF choke points & add hops
- This is irrespective of where the storage is placed (please see next slide)

Common Storage Fabric Designs

Edge-Core Storage Fabric

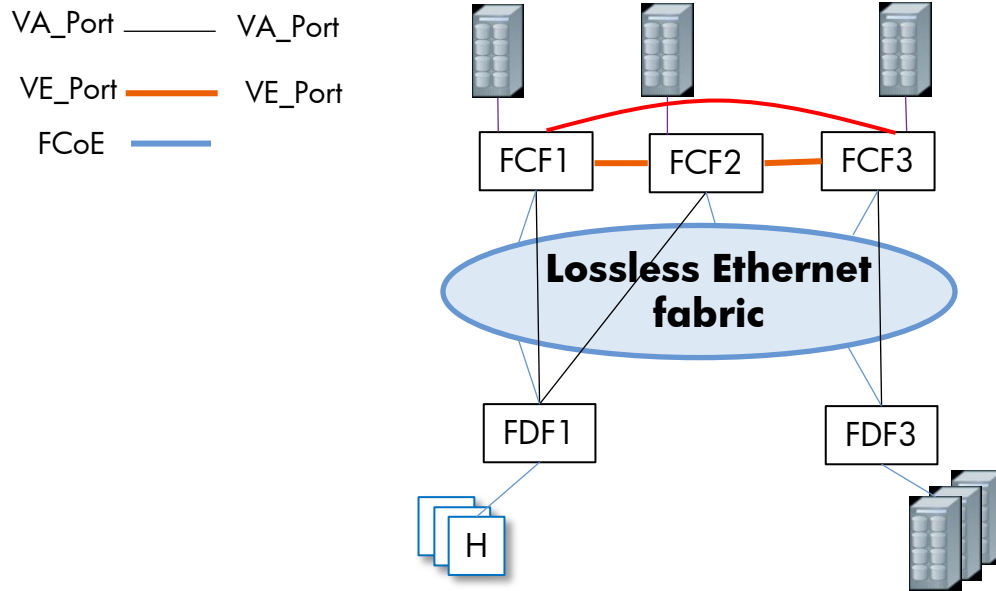


Edge-Core-Edge Storage Fabric



- Common storage fabric designs are variations of edge-core and edge-core-edge design shown above
- Both storage placement options benefit from a more flexible FDF connectivity

Using a Lossless L2 Fabric

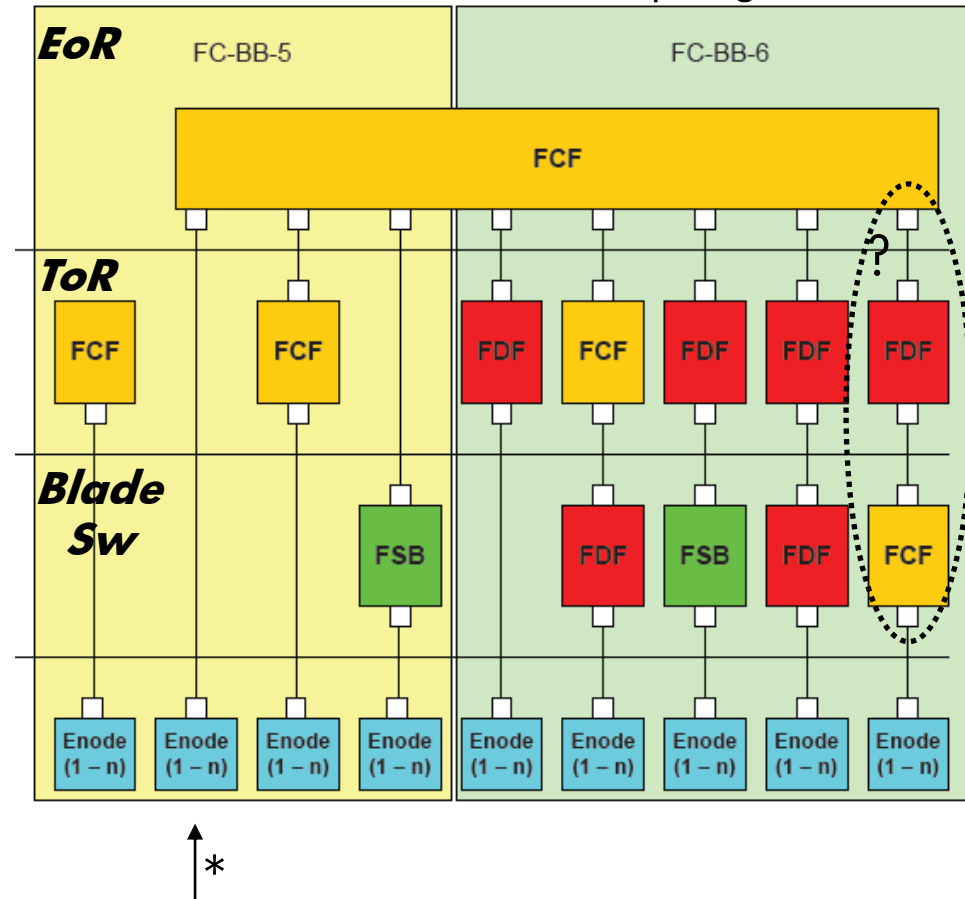


- In edge-core scenario the Ethernet fabric is crossed twice
- In edge-core-edge design the Ethernet fabric is crossed three times
- With Flexible FDF Connectivity the Ethernet fabric is crossed once
- *What is needed is separation of virtual domain controls, for virtual domain setup and HA, from forwarding of storage traffic over links subject to BB6 requirements*
 - FCF wide connectivity is required and remains however for the purpose of fabric controls

BB5 allowed for more flexible trunking options

- With BB5 a Blade-Sw or ToR, e.g. an NPIV device, has option to trunk with more than two aggregation switches
 - Limiting this to two FCFs seems like a step back
- 10-343 showed that:
 - N_Port <-> FCF <-> FDF (c/FCF) or
 - N_Port <-> FC <-> FDF (c/FCF) are NOT supported since a virtual domain only lives at the edge
- 10-343 focused on realistic physical location of FDF/FCF at the edge of the network i.e. focused on a single distributed switch
- Trunking options from FDFs don't seem to have been the center of focus

10-343v0: "Review of Realistic Topologies"



Past suggestion to remove topology constraints -- an E_Port light perhaps?

- Arguments in favor of E_Port light:
 - Ideally a Dist-FCF should be a distributed switch, i.e. from outside the dotted line surrounding FCFs & FDFs, the rest of the network should not care what the internal components behind the physical ports are
 - With FC, a switch physical port can be an F_Port or E_Port etc.
 - Internally an FDF E_Port light can optionally play dumb and forward the control traffic to its cFCF! or alternatively E_Port fabric synchronization can be disabled if E_Port is to an FDF for an E_Port light variation
- But E_Port light would not work without changes to FSPF (Please see next slide)

I am NOT suggesting that we use FSPF or E_Port light to FDFs

FSPF to FDF or not – E_Port light Achilles

- From routing perspective a virtual domain or dist-FCF is a single node, v_x , in the graph $G(V,E,\Psi)$ of a FC fabric
 - Nodes outside the virtual domain have no visibility to the internal links of a virtual domain
 - Hence A_Ports and FDFs are not visible to the fabric FSPF
 - By extension, currently access to Enode's attached to an FDF from outside of the virtual domain is only through primary and secondary FCF
- To make FDF's, routing-visible, to the rest of the network requires that they be identified by a unique label i.e. FSPF should be configured to route based on VD_ID + FDF_IDs;
 - What kind of change does this require to FSPF; depends on implementations?
 - Not having to change FSPF seems to be a key requirement (i.e. backward compatibility)
 - Also legitimate concern is that FSPF is not interoperable (hence limiting FCF/FDF interoperability)
 - Making FDFs visible to FSPF also adds to complexity of FDF control software
- Also a problem seems to be how an FDF would instantiate E_Ports light with FCF's other than p/s FCF using FIP? How is network topology controlled?
 - Not appealing having to configure the entire network topology!

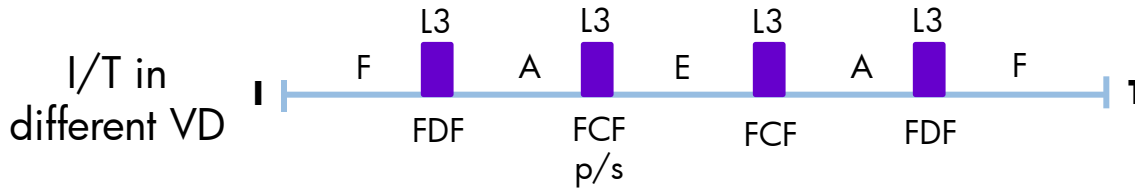
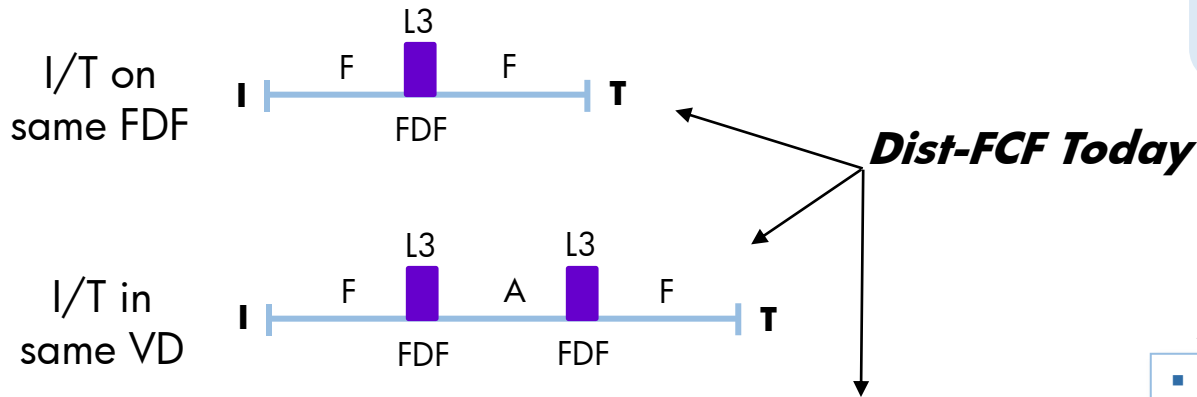
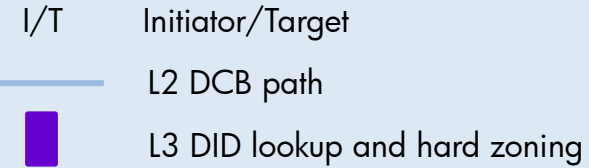
Dilemma FC faces -- Summary

- Dilemma FC faces is:
 - A virtual domain, where FDFs share a domain_ID and centralize fabric control protocols is a useful concept to conserve the domain-ID and scale the fabric port count
 - However a virtual domain masks routing visibility as a label/domain-ID (and associated E_Ports) are required to add granularity and visibility to FSPF
 - To be backward compatible, we can't change FSPF to make internals of the virtual domain visible to routing
 - Hence FDF's can not route to or be routed to from outside of the virtual domain
 - And a new E_Port light makes FDFs more complex and requires changes to FSPF
- Therefore currently the following constraints apply to a dist-FCF port extender model:
 - FDFs have no E_Ports making a dist-FCF an edge switch technology
 - FDF trunk connectivity for forwarding to the SAN, i.e. ASLs, are limited to two FCFs; limiting fabric design options
 - All traffic to/from outside of the virtual domain is through E_Ports of the p/s FCF
 - Edge trunking options are more limited than BB5

Can we use VA_Links to other virtual domains?

- What does an ASL ELP procedure accomplish?
 - Learn the MAC address of the VA_Port
 - *Can also be learnt through other means for FDFs outside of my virtual domain*
 - Specify and confirm jumbo frame size along the path
 - *This can be fixed by configuration for the fabric*
 - Learn the FDF switch name so the ASL & its virtual link cost can be reported to FCF for routing
 - *Not necessary if FDF is not part of MY virtual domain*
 - Other ELP parameters such as class F service parameters, class of service supported, flow control mode and parameters are not used?
 - Keep_Alive's follow to monitor the status of connectivity at layer-3
 - *Can still be sent to monitor connectivity; the first keep alive perhaps can also test for jumbo frame size compliance*
- FDF's already have ASL's to the secondary FCF which shares the virtual domain ID
 - A s/FCF has at least two domain_IDs (the switch DID and the virtual domain ID)
 - FDF's already forward to this other domain_ID via the s/FCF!

Flexible FDF Connectivity



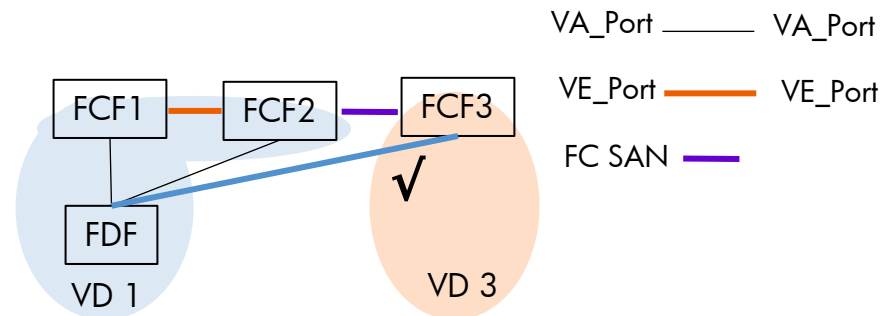
One way to do Flexible FDF Connectivity

- Define a mechanism to piggy back the ingress/egress FDF MAC (e.g. PLOGI rewrite)
- Use MAC-addr at ingress/egress FDF for a L2 VA_Port connectivity
- No change to adapters?
- Dist-FCF controls already defined remain in place
- Hard zoning etc. is applied at FDF
- *No need to expose FDFs to FSPF*
- Forwarding is optimized at L2
- Is more compatible with L2 Ethernet fabric technologies

Goal is for the standard to also allow flexible FDF Connectivity

There is no free lunch

- Downside of flexible FDF connectivity is that the forwarding tables grow
 - Actual numbers depend on how FDF address assignment is done and implementation
- To prevent this, one can optionally limit A_Link connectivity, outside of a virtual domain, only between FDFs and FCFs
 - This would then allow for vendor differentiation
 - Smaller tables allow FDF to any FCF connectivity
 - Larger tables allow FDF to any FDF connectivity



Summary

- Separation of FDF *controls* as simple edge devices, sharing a virtual domain (ID), from their capability to *forward* to any FCF/FDF, while meeting BB6 requirements, is desirable and does not seem to be mutually exclusive
- FDF control remains the same using a primary FCF, a virtual domain concept , VA_Port protocol etc.
 - Domain ID's are preserved using a virtual domain and fabric synchronization remains manageable
 - BB6 requirements are met:
 - Do not change adapters
 - Use trusted fabric/FDF for forwarding based on L3 at the edge where hardware based zoning is applied
- Forwarding and trunking topology becomes more flexible with following benefits:
 - A topologically flatter network saves on cost
 - Fewer L3 hops reduce latency and remove potential choke points
 - And more resilient FDF connectivity enhances HA for two reasons:
 - First; FCoE customers seem to be moving away from dual redundant air-gap fabric design in the access (“Customers do not think a dual fabric configuration is viable for their access tier” 11-164v0)
 - Second: HA design with both P/S FCF will likely be optional (so as a minimum a secondary data path should be available in the absence of s/FCF for load balancing)

Thank You

