vCD-NI Networking and MTU – The most crucial point

Well Network Performance inside a vCloud Infrastructure is very crucial in terms of serving customer without any hiccups. So MTU (read Maximum Transmission Unit) inside a vSwitch or a pSwitch is an extreme important part. We have lot of discussion around the MTU size (read Jumbo Frame) and we have seen a real performance boost around that.

So now we are back with a MTU consideration around vCD-NI Networking and we need to look at it very closely. Let us first look at the vCD-NI Network and what it does.

- A vCloud Director Network Isolated (vCD-NI) network is a proprietary, layer 2 network connection between two (or more) VMware ESX/ESXi hosts based on a MAC-in-MAC encapsulation performed by the VMkernel.
- A vCD-NI network pool can be used as the backing for organization networks and vApp networks.
- Networks are created by the use of tunnelling (encapsulation).
- Traffic moves between ESX/ESXi hosts on network layer 2, using MAC-in-MAC encapsulation.
- vCenter Server creates the required port groups as needed (Automatic Provision).

Below is a pictorial representation of a vCD Tunnel.

So now let us see in a deeper way what a MiM Encapsulation (Mac in Mac) has to offer.

Mac-in-Mac (802.1ah) encapsulates Ethernet frames with a Service Provider MAC header. MAC-in-MAC technology overcomes the inherent scalability limitations of VLAN and Q-in-Q networks that make them impractical for use in larger networks by enabling up to 4000 times as many service instances as supported by traditional VLAN and Q-in-Q networks.

So now we know that vCD-NI packets use MAC-in-MAC encapsulation and it adds 24 bytes to the core packet. Also we know that the default physical network MTU size is 1500 bytes but when a vCD-NI header adds to the core packet it becomes 1524 bytes.

Normally a VM creates a 1500 byte packet and sends it to the vSwitch. After this Cloud layer (vCD-NI module) add the vCD-NI header to it which increases the packet size to 1524 bytes. For a Physical Switch 1524 byte packet size is too large, so the packet will be splitted into two packets. They will be merged into a single 1524 byte packet at the destination and VMkernel will unwrap the packet and pass on a 1500 byte packet to the destination virtual machine. This fragmentation causes a big performance loss.

Ok so now we have figured out what the problem is and how it happens. Now we will look at how to deal with this so that the performance will not take a hit and end customers will get better service.

Increase MTU on dVS and Set network pool setting accordingly to inform the vCD-NI layer:
- Add 24 bytes: 1500 + 24 = 1524 bytes MTU.
- This is not going to be an optimal solution as physical switch still does not accept larger packets.

Decrease MTU size on the virtual NIC (vNIC) of the guest operating system:
- To 1500 – 24 = 1476 bytes.
- Again this is not a viable solution as it must be done on every virtual machine which is a tedious job.

Increase MTU size on the dVS and the physical switch to 1524:
- This is going to be the best solution looking at the feasibility and timeline to implement

- At the same time we need to inform the vCD-NI layer too to adjust it to 1524 byte