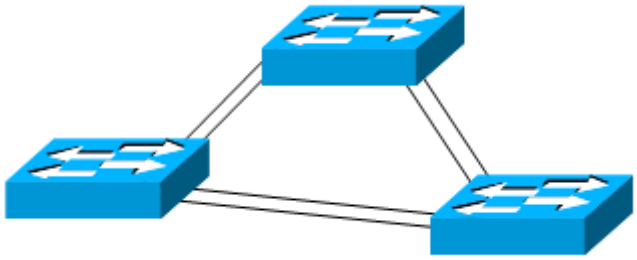
REDUNDANT SWITCH DESIGN WITHOUT SPANNING TREE

And no FabricPath either. This one works without any active protocol involved, and no blocked links. To good to be true? Of course!



Take the above example design: three switches connected by port channels. Let's assume users connect to these switches with desktops.

Using a normal design, spanning tree would be configured (MST, RPVST+, you pick) and one of the three port-channel links would go into blocking. The root switch would be the one connecting to the rest of the network or a WAN uplink, assuming you set bridge priorities right.

Easy enough. And it would work. Any user in a VLAN would be able to reach another user on another switch in the same VLAN. They would always have to pass through the root switch though, either by being connected to it, or because spanning tree blocks the direct link between the non-root switches.

Disabling spanning-tree would make all links active. And a loop would definitely follow. However, wouldn't it be nice if a switch would not forward a frame received from

another switch to other switches? This would require some sort of split horizon, which VMware vSwitches already do: if a frame enters from a pNIC (physical NIC) it will not be sent out another pNIC again, preventing the vSwitch from becoming a transit switch. Turns out this split horizon functionality exists on a Cisco switch: ‘switchport protect’ on the interface, which will prevent any frames from being sent out that came in through another port with the same command.

Configuring it on the port channels on all three switches without disabling spanning tree proves the point: the two non-root switches can’t reach each other anymore because the root switch does not forward frames between the port channels. But disabling spanning tree after it creates a working situation: all switches can reach each other directly! No loops are formed because no switch forwards between the port channels.

Result: a working network with active-active links and optimal bandwidth usage. So what are the caveats? Well…

- It doesn’t scale: you need a full mesh between switches. Three inter-switch links for three switches, six for four switches, ten for five switches,… After a few extra switches, you’ll run out of ports just for the uplinks.
- Any link failure breaks the network: if the link between two switches is broken, those two switches will not be able to reach each other anymore. This is why my example uses port-channels: as long as one link is active it will work. But there will not be any failover to a path with better bandwidth.

Source : http://reggle.wordpress.com/2014/02/26/redundant-switch-design-without-spanning-tree/