Cable Management

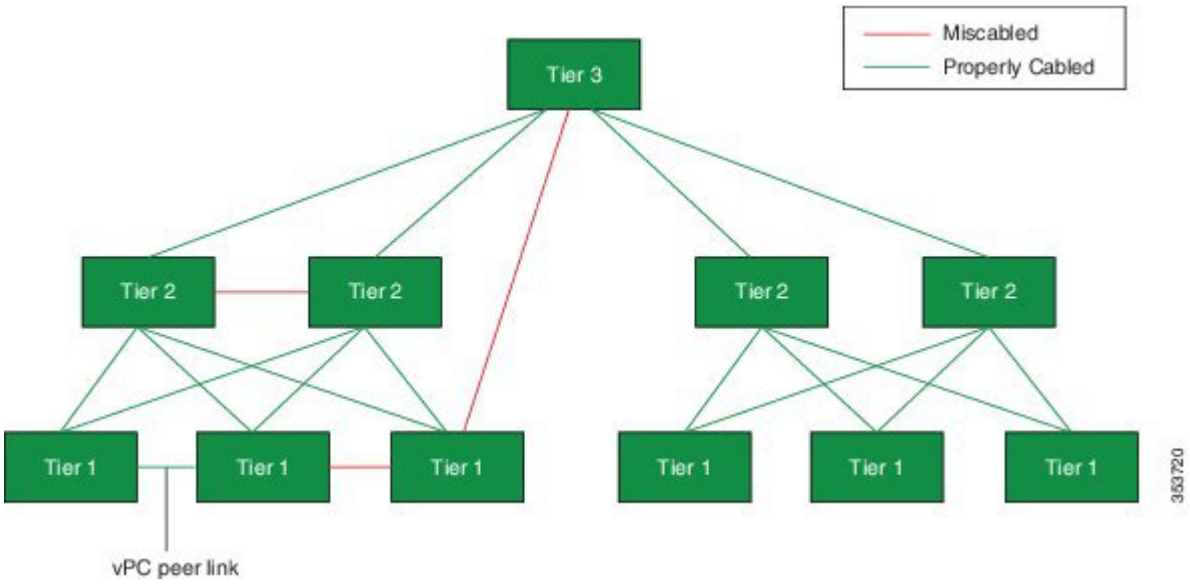
Cable Management

In a highly meshed network such as Clos topology based network fabric, miscabling can be a pragmatic problem leading to painful troubleshooting without sufficient support. The cable management feature calls out for two mechanisms to address the miscabling issues caused due to human errors. The first mechanism is based on the tier-based checks and the second mechanism is based on a user-defined cabling plan.

Tier-Based Miscabling Detection

The figure below shows the properly cabled (in green) and miscabled (in red) links in a hierarchical network designed based on Clos design principle. The cable verification rules based on tier-level is applicable only for Clos topologies. Each box represents a switch in the network.

Figure 1: Tier-Based Miscabling Detection



As shown in the figure, every switch in a stage is associated with the corresponding tier-level number assigned to the stage it is in. That means that all the leaf switches (in the lowest-level Clos stage) are provisioned with a tier-level of 1, the next higher-level stage switches (1st stage of spine switches) are provisioned with a tier-level value of 2, and the next higher-level stage switches (2nd stage of spine switches) are provisioned with a tier-level of 3, so on.

As an example the tier level configuration on a level 2 spine switch is performed as follows:

```
Switch(config)# fabric connectivity tier 2
```

Note You can configure this automatically using the POAP template on DCNM when the switch is designated as a spine or leaf.

You have to enable features 'cable-management' and 'LLDP' and this command enables tier-based checks on the switch.

Every switch in the network advertises its tier-level number (only if one is configured with), as part of the link layer PDUs (of LLDP) in addition to the chassis-id, and physical port-id to their adjacent peers.

The receiver of the link layer PDU performs the following algorithm checks to validate link connections with sender per Clos network design rules:

1 Performed only at leaf switches (switches assigned with a tier-level of 1)

- The received tier level from adjacent link partners must be 2. There will be an exception to this rule. If the receiving link is vPC peer link (detected automatically via LLDP), a remote tier-level of 1 will be allowed.

2 Performed only at spine switches (switches assigned with tier-level > 1)

- The received tier level from adjacent link partners must be either (my_tier_level + 1) or (my_tier_level - 1), where my_tier_level is the tier level assigned to the spine switch performing the check.

If any cabling inconsistency is detected while validating the remote link information from the adjacent switches, then the miscabling action (error disabling the port by default) is performed and a cabling mismatch error is logged. The logged error includes necessary information about the involved local chassis-id, local port-id, local tier-level, and remote switch details such as peer chassis-id, peer port-id, and peer tier-level.

The following example shows the fabric connectivity neighbors with tier details.

```
Switch# show fabric connectivity neighbors all

------------------------------------------------------------------------------
Local System:
Device Tier Config:          Enabled       Device Tier Level:               1
Tier-mismatch Retry Config:  Disabled      Tier-mismatch Retry Timeout:     0
Cable Plan Check:            Enabled
DeviceID:  poap-integ-leaf1                ChassisID:  0003.0111.0008
------------------------------------------------------------------------------
         Codes: (Ok) Normal, (ErrT) Tier error , (ErrC) Cabling Plan error,
                (Ok/VPC) VPC leaf-to-leaf connection, (S) Stale entry
                (Unkn) Unknown Tier, (Enp) Plan Entry not present
Neighbor Table:
------------------------------------------------------------------------------
Local      DeviceID        PortID       Tier  Cable-Plan          Status
Intf

Eth2/1     spine0          Eth2/1       2     spine2,Eth2/1       ErrC
Eth2/6     spine5          Eth2/2       2     spine5,Eth2/2       Ok
Eth2/5     spine4          Eth2/2       2     spine0,Eth2/2       ErrC
Eth2/3     spine6          Eth2/2       2     Enp                 Ok
Eth2/4     spine3          Eth2/2       2     spine7,Eth2/1       ErrC

poap-integ-spine0# show fabric connectivity neighbors errors
------------------------------------------------------------------------------
Local System:
Device Tier Config:          Enabled       Device Tier Level:               2
Tier-mismatch Retry Config:  Disabled      Tier-mismatch Retry Timeout:     0
Cable Plan Check:            Disabled
DeviceID:  poap-integ-spine0               ChassisID:  0003.0210.0008
------------------------------------------------------------------------------
         Codes: (Ok) Normal, (ErrT) Tier error , (ErrC) Cabling Plan error,
                (Ok/VPC) VPC leaf-to-leaf connection, (S) Stale entry
                (Unkn) Unknown Tier
Neighbors with Miscabling Warning/Error:
------------------------------------------------------------------------------
Local           DeviceID             ChassisID        PortID       Tier Status
Interface

Ethernet2/1     poap-integ-leaf0     0003.0110.0001   Eth2/1       2    ErrT,S
Ethernet2/2     poap-integ-leaf1     0003.0111.0001   Eth2/1       2    ErrT,S
Total entries displayed: 2
```

In case switch is configured with a tier level that is connected to another switch that does not support this feature and hence does not send tier level information, then the link will not be checked for miscabling.

Overview of Cable-Plan Miscabling Detection

Cable-plans also rely on LLDP to exchange information, for example switch ID, port number, between two ends of a cable connection. This information is carried in vendor specific LLDP TLVs.

You must first import a valid cable-plan to the switch and check against the information in these received TLVs. Cable-plans are XML files that fully or partially describe the topology of the data center.

Once a plan is imported, checks will happen to make sure that the imported plan matches the received TLV information. If it matches nothing will happen, however if there is a mismatch then the miscabling action will be taken on the port in question (err-disable by default). The user can change the actions taken on a miscabling event.

Administrators will then be warned via SysLog of any such miscablings, and they will take the appropriate action to fix the problem (in most cases like above, they will simply need to fix the miscabling to the correct configuration as described in the cable-plan, and clear the err-disable).

To enable miscabling detection, configure the following feature commands:

```
Switch(config)# feature lldp
Switch(config)# feature cable-management
```

Cable-Plan XML Schema

The following schema of cable-plan XML is for a data center with two spine switches, and three leaf switches:

```
<?xml version="1.0" encoding="UTF-8"?>
<CISCO_NETWORK_TYPES version="1.0" xmlns="http://www.cisco.com/cableplan/Schema2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cisco.com/cableplan/Schema2 nxos-cable-plan-schema.xsd ">
  <DATA_CENTER networkLocation="san-jose" idFormat="hostname">
    <CHASSIS_INFO sourceChassis="spine1" type="n7k">
      <LINK_INFO sourcePort="Eth2/1" destChassis="leaf1" destPort="Eth2/1"/>
      <LINK_INFO sourcePort="Eth2/2" destChassis="leaf2" destPort="Eth2/1"/>
      <LINK_INFO sourcePort="Eth2/3" destChassis="leaf3" destPort="Eth2/1"/>
    </CHASSIS_INFO>

    <CHASSIS_INFO sourceChassis="spine2" type="n7k">
      <LINK_INFO sourcePort="Eth1/1" destChassis="leaf1" destPort="Eth1/2"/>
      <LINK_INFO sourcePort="Eth1/2" destChassis="leaf2" destPort="Eth1/2"/>
      <LINK_INFO sourcePort="Eth1/3" destChassis="leaf3" destPort="Eth1/2"/>
    </CHASSIS_INFO>
  </DATA_CENTER>
</CISCO_NETWORK_TYPES>
```

Following is the breakdown of each line and the XML tags and attributes:

```
<?xml version="1.0" encoding="UTF-8"?>
<CISCO_NETWORK_TYPES version="1.0" xmlns="http://www.cisco.com/cableplan/Schema2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cisco.com/cableplan/Schema2 nxos-cable-plan-schema.xsd ">
```

The previously mentioned example headings are required for XML processing and Cisco specific headers required to denote it as a Cisco cable-plan. These must be exact as mentioned in the previous example for all cable-plans. Failure to adhere to the specified format will result in a rejected cable-plan.

- `CISCO_NETWORK_TYPES` - Parent tag for the entire XML cable-plan. The entire cable-plan must be within this tag.

```
<DATA_CENTER networkLocation="san-jose" idFormat="hostname">
```

- `DATA_CENTER` - Houses all the information for each chassis in the plan.

- networkLocation - States where the data center is for clerical purposes. No affect on miscabling.
- idFormat - States the format in which IDs will be presented in subsequent entries. As of 12/2/2014, the only supported format is "hostname". All cable-plans that do not use “hostname” as the ID format will be rejected.

```
<CHASSIS_INFO sourceChassis="spine1" type="n7k">
```

- CHASSIS_INFO - Describes a single chassis. All interfaces that belong to sourceChassis that administrators want to have checked by cable-plan must be within one, and only one of these tags.
- sourceChassis - The chassis that all subsequent interfaces (described in LINK_INFO tags) belongs to. In this case, all interfaces within the CHASSIS_INFO tags are said to belong to the chassis called "spine1".
- type - The type of chassis. Only the Cisco Nexus Switches are supported, so this attribute must read "n#k", such as "n7k" (non-case sensitive). All cable-plans that do not adhere to the "n#k" format will be rejected.

```
<LINK_INFO sourcePort="Eth2/1" destChassis="leaf1" destPort="Eth2/1"/>
<LINK_INFO sourcePort="Eth2/2" destChassis="leaf2" destPort="Eth2/1"/>
<LINK_INFO sourcePort="Eth2/3" destChassis="leaf3" destPort="Eth2/1"/>
```

- LINK_INFO - Fully describes an interface connection from sourceChassis (as mentioned previously) to destChassis. In this case, we are stating that spine1's port eth2/1 is connected to leaf1's eth2/1, spine1's eth2/2 is connected to leaf2's eth2/1, and so on.
- sourcePort - The sourcePort is the port on the sourceChassis (as mentioned previously). Source ports should be unique per chassis (that is, spine1 should not describe multiple connections coming from port eth2/1). While an import will not fail on cable-plans with non-unique ports, a warning will be printed and only the first entry will be read and checked by cable-plan.
- destChassis - The destination chassis that the sourceChassis is connected to.
- destPort - The port on the destination chassis that has the connection. Like sourcePort, this should be unique to destChassis.

```
</CHASSIS_INFO>

<CHASSIS_INFO sourceChassis="spine2" type="n7k">
<LINK_INFO sourcePort="Eth1/1" destChassis="leaf1" destPort="Eth1/2"/>
<LINK_INFO sourcePort="Eth1/2" destChassis="leaf2" destPort="Eth1/2"/>
<LINK_INFO sourcePort="Eth1/3" destChassis="leaf3" destPort="Eth1/2"/>
</CHASSIS_INFO>
```

Here you close the CHASSIS_INFO tag and we completely describe the connections on all interfaces that spine1 can have. You can open another tag to describe interfaces in the same way, on a different router that you want it checked. There is no limit to the number of unique CHASSIS_INFO tags you can describe in a cable-plan.

```
</DATA_CENTER>
</CISCO_NETWORK_TYPES>
```

Finish closing the tags to complete our XML file. At this point we have a valid XML cable-plan that completely describes a data center containing the following switches: spine1, spine2, leaf1, leaf2, leaf3.

Note that you need not repeat connections from each chassis' perspective. Since you describe that sourceChassis spine2 is connected to destChassis leaf1 via Eth1/1 - Eth1/2, you do not need an entry describing sourceChassis leaf1 and destChassis spine2. While the plan will not fail if you include this redundant information, it will be ignored and add unnecessary length to the plan.

Also ensure that other interfaces may be present that were not described in the cable-plan. Consider a third spine, spine3, connected to all the leaf nodes. Since this is not described in the cable-plan, LLDP TLVs with this information will not be checked and only an indicator warning administrators of its absence from the cable-plan will be noted. As long as the un-described spine is not interfering with ports already described in the cable-plan, then no actions will occur with the missing ports (that is, the new spine3 cannot be connected to leaf1's eth1/2, since you explicitly state that leaf1's eth1/2 is connected to spine2's eth1/1).

Cable-Plan Specific Commands

The cable-plan exec commands are under the **fabric connectivity cable-plan** commands. For example, to import the valid XML cable-plan we can use this CLI:

```
leaf1# fabric connectivity cable-plan import [means]:[location] {update} {verbose}
```

- means - Describes the way you may want to import. Currently the only local means for importing is via bootflash and USB. If you would like to import via remote location, FTP, SCP, SFTP and TFTP protocols are supported for this option.
- location - The location of the file. For instance "`scp://calinfor@171.77.77.7/nobackup/test.xml`" will scp the file located at /nobackup/test.xml on 171.77.77.7.
- update - (Optional) To add the entries in the importing cable-plan to the cable-plan entries that have already been imported in a previous import command. If there is no cable-plan already imported then this keyword has no effect. If this keyword is not specified during the import command, then the previously imported cable-plan (if there is one) will be deleted and it will be replaced with the new cable-plan specified in the import command.
- verbose - (Optional) To print all errors associated with the import. If this option is not specified, only a one line description of success or failure will be printed. If the option is specified, detailed information on why the import failed will be printed to the console.

```
leaf1# fabric connectivity cable-plan export[cable-plan location]
```

The export CLI will take a previously imported cable-plan and write it to a file. Since cable-plans are stored in memory after they are imported, users may have lost or altered the previous XML file containing the cable-plan. This allows these users to export the current cable-plan stored in memory and regain their XML file.

```
leaf1# fabric connectivity cable-plan generate [cable-plan location] [name]
```

Auto generate a cable-plan based on the current LLDP neighbors. This creates a valid cable-plan based on the local switch's perspective. Outputs a time-stamped file in the bootflash when done. Useful for allowing users to quickly create a cable-plan out of current topology without writing it by hand. Users can then either import this cable-plan to enforce the current topology or users can use it as a template of sorts to create a more complex cable-plan. Note, if there are no LLDP neighbors present then only a template that will need to be edited will be created.

Now turn on cable-checks. If this is disabled (by default) then all TLVs received on this switch will not be checked against the cable-plan, regardless of whether or not a plan is imported:

```
leaf1# configure terminal
leaf1(config)# [no] fabric connectivity cable-plan enforce
leaf1(config)# exit
```

To check information regarding the new cable-plan:

```
leaf1# show fabric connectivity cable-plan
```

This command prints the information in the currently stored cable-plan. If no information is stored this will be blank. This will print the local and remote Chassis and Port IDs, and the cabling status of this connection. This is just a printing of the cable-plan and the status of each link.

Switch Links Cable State

The switch learns and stores all the neighbors and their respective ports and states. The following command can be used to display this information.

```
spine1# show fabric connectivity neighbors {errors | interface | tier}
```

- errors - (Optional) Displays only interfaces with errors
- interface - (Optional) Displays information pertaining to the specified interface
- tier - (Optional) Displays only interfaces with the specified tier level
- default (blank) - Displays all interfaces that have received an LLDP TLV since either tier or cable-plan checks have been toggled on. Unlike "show fabric connectivity cable-plan", the show neighbor command will show the actual data being received by the switch via LLDP TLVs

Local Chassis and Port IDs, remote Chassis and Port IDs, the tier levels of the remote Chassis, the expected cable-plan entry and all the status will be displayed.

Status codes include the following:

- Ok - Everything working as intended, the check succeeded
- Unkn - Unable to determine the status, most likely because cable-check and tier-check has been disabled
- ErrC - The port has been err-disabled due to a mismatch (that is the TLV received did not match the specific entry in the cable-plan)
- ErrT - A tier level mismatch error
- S - May be specified at the end of a status. This means the port is "stale" in that the switch has received a purge event from the remote peer. This usually happens when the port has gone err-disabled or when we have not seen LLDP events from this link in a long time.

Examples

The following is a sample output of fabric connectivity:

```
switch# show fabric connectivity neighbors

------------------------------------------------------------------------------
Local System:
Device Tier Config:          Disabled      Device Tier Level:          Unknown
Mismatch Delay Config:       Disabled      Mismatch Delay Timeout:           0
Cable-Plan Enforce:          Enabled
DeviceID:  switch                          ChassisID:  f866.f2d6.37c4
------------------------------------------------------------------------------
         Codes: (Ok) Normal, (ErrT) Tier error , (ErrC) Cable-Plan error,
                (V) VPC Peer connection, (S) Stale entry, (Unkn) Unknown,
                (Enp) Entry not present in Cable-Plan, (Tl) Tier level
Neighbor Table:
------------------------------------------------------------------------------
Local      DeviceID              PortID     Tl   Cable-Plan            Status
Intf                                           Entry

Eth1/13    switch                Eth1/14    Unkn switch2,Eth1/14       ErrC,S
```

```
Eth1/14    switch                Eth1/13    Unkn switch9,Eth1/13       ErrC,S
```

When cable-plan is enabled, the cable-plan column will show the plan based validation status (Ok, Enp - Entry Not Present, or as above the expected value for errored links).

```
Spine11# show interface Eth2/1

Ethernet2/1 is down (errDisabled)
......
......

switch# show interface status err-disabled

--------------------------------------------------------------------------------
Port          Name               Status             Reason
--------------------------------------------------------------------------------
Eth2/1         --                Miscabled          Miscabled
```

Miscabling Error Disable Action Control

By default any tier based or cable-plan based mismatches trigger err disabling of the ports. After the miscabling or topological errors are corrected, the miscabled ports can be brought back live either by manually removing them out of error disabled state or automatically through a configuration CLI. The error disabling action can also be turned off or delayed. The following knobs can control these optional behaviors.

- To remove the default errdisable on error action:
  `Switch(config)# no errdisable detect cause miscabled`
- Miscabled error disabling action can be delayed for specified time (timeout) using the following configurations. Here action will be taken after <timeout> seconds have passed:
  `Switch(config)# fabric connectivity mismatch action delay <timeout>`
- Auto recovery of the miscabled ports out of error disabled state can be enabled using the following command. The below command will attempt to recover all miscabled ports at some set interval.
  `Switch(config)# errdisable recovery cause miscabled`
- To change the interval at which the errdisable recovery command attempt to recovery, the following configure command can be used:
  `Switch(config)# errdisable recovery interval <time>`
- To clear a single entry or all entries from the Clos neighbor cache immediately, the following clear command can be used:
  `Switch# clear fabric connectivity neighbors [interface | stale]`

  This CLI will allow the you to clear a single entry or all error entries or all entries from the Clos neighbor cache immediately. You should manually clear an already secured port in the neighbor cache if recabling is desired to remove old stale entry immediately. The entry will be automatically removed after the hold time otherwise. If you have multiple errors that you just fixed, performing a 'clear neighbors' on the affected switches is the easiest way to bring up all interfaces again if 'error recovery' is not enabled. Also, if you have a switch that was previously in the network, but has gone inactive (may be you removed it or it was taken down), performing a clear command is the only way to completely remove it from the neighbor cache (so it stops showing up in the **show neighbors** command mentioned above).
- To clear the cable-plan from this switch, the following command can be used:
  `Switch# clear fabric connectivity cable-plan`

  Clears the current cable-plan. If you had previously written a cable-plan to the startup config and you want this clearing to persist, use the **copy running-config startup-config** command.