Augmentation Impact on Configuration

YANGsters Discussion (2021-11-30)
Scott Mansfield
Ericsson
Introduction

• Augmentation is an excellent way to add capabilities to a YANG module.
• For example ieee802-dot1q-bridge.yang augments the IETF’s interface capability.
• However there is a potential issue that is driving this discussion in YANGsters

• The work on ieee802-dot1q-sched.yang has introduced an augment that adds constraints that causes configuration errors even when you don’t want to use the scheduled-traffic feature.
A small example to demonstrate the issue

- Two yang modules
- main.yang has a list that has two leafs
- aug1.yang augments main-list with a container that includes a list that has a must constraint
This will be troublesome....
Because, if you include the augment in a solution, then any instances of main-list will have to have an admin-control-list whether they want it or not.
Example Instances...

• running with just main.yang

• running with both main.yang and aug1.yang
Potential Solution

- Create a leaf that will toggle when you want to support the functionality needed by the container, then use a when statement in the container.
Example Results

• There is one list that doesn’t have the admin-control-list and one that does...
Discussion

• This example was created because if you have a configuration that doesn’t use scheduled-traffic and then you include the yang files for scheduled-traffic your configuration will break.
  • The “feature” statement doesn’t help because there are times when you only need some of your bridge-ports to support the feature. When the feature is on, all the YANG in the feature appears in the tree.

• Where else are there must statements that could cause this type of problem with configuration?
• Are there other solutions that work?