Overview

The fundamental operation of the network operating system command-line interface (CLI) has not changed in over twenty years. Created as the primary mechanism for interacting with switches and routers, the CLI has remained an unchanged enigmatic entity since its creation. Once mastered, it allows the user to perform varied tasks from configuration and monitoring to troubleshooting and management. What it does not provide is the ability to support the rapidly changing requirements of today's high performance networks.

The extensible design of Arista's EOS provide the ability to rapidly customize EOS to meet your operational requirements. Instead of creating 'off box' scripts/tools, or waiting year(s) for your chosen vendor to react, you can easily customize EOS.

Getting Started

Arista's EOS makes it possible to provide customization, a word not typically associated with network operating systems, to meet your changing operational needs. In the following scenario we will walk through the steps required to make basic changes to the CLI.

The Scenario

During company XYZ's annual security audit the auditors cited the ability to add a disclaimer banner to the output of 'show port-security' as highly desirable functionality. Using EOS this is not a problem we will walk through the steps below.

First take a look at the output of 'show port-security' and see what the auditors want.

The audit team requested the banner:

"This device is for the authorized access of XYZ employees only!"

```
7148S#sh port-security

Secure Port     MaxSecureAddr  CurrentAddr  SecurityViolation  Security Action
                  (Count)        (Count)        (Count)
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Total Addresses in System: 0
7148S#
```

To perform customization of EOS, enter the “bash” shell and navigate to the directory that holds the CLI Plugin data.

```
7148S>en
7148S#bash
Arista Networks EOS shell
[admin@7148S ~]$ cd /usr/lib/python2.6/site-packages/CliPlugin
[admin@7148S CliPlugin]$
```

Enter enable mode

Enter bash shell

cd to the proper directory

NOTE: *Take a minute and familiarize yourself with the files in this directory. They contain the code that makes up the Arista EOS CLI. You can use ‘ls –la *Cli*py’ to shorten the output and get a list of all of the relevant files. The filenames are typically self-explanatory. The files that are not stored under /mnt/flash (or /mnt/usb1, if present) live in a RAM disk that is created at boot time. All changes that you make to files outside of /mnt/flash are lost on reboot. We will discuss how to make changes persistent later in this paper.*

The CLI is written in Python and named in accordance with the functionality it provides, in this case find the file that contains the code for the port security command, highlighted in **red** below, PortSecCli.py.

```
[admin@7148S CliPlugin]$ ls *Cli*py

AaaCli.py                  IntfCli.py          PimCli.py
AclCli.py                  IntfRangeCli.py     PortSecCli.py
AclCliRules.py             IntfSnmpCli.py      PowerCli.py
ActiveManagementIntfCli.py Ira1p6Cli.py        PowerDiagsCli.py
AgentCli.py                Ira1p6IntfCli.py    RadiusCli.py
AliasIntfCli.py            Ira1pCli.py         RedSupCli.py
BackupIntfCli.py           Ira1pIntfCli.py     ReloadCauseCli.py
BeaconLedCli.py            LagCli.py           ReloadCli.py
…                          …                   …
FileCli.py                 PeerIntfCli.py      VmTracerIntfCli.py
FocalPointCli.py           PfcCli.py           VrrpCli.py
FruCli.py                  PhyAelurosCli.py    WaitForWarmupCli.py
IgmpCli.py                 PhyCli.py
IgmpSnoopingCli.py         PhyConfigCli.py
```

This paper uses the 'vi' editor, but 'zile', an emacs-like editor, is also installed.

Since the files in CliPlugin are read-only, for non-root users, open the file as root and review the content, scrolling through you will find the code ‘show port-security’ command.

```
[admin@7148S ~]$ sudo vi PortSecCli.py
#-----------------------------------------------------------------------
# The "show port-security" command, in "enable" mode.
#-----------------------------------------------------------------------
def doShowPortSecurity( mode ):
  print ("Secure Port      MaxSecureAddr  CurrentAddr  "
        "SecurityViolation  Security Action")
  print "                  (Count)       (Count)      (Count)"
  print ("-------------------------------------------------------------"
        "-------------------")
  format = "     %-7s    %7d    %10d    %9d           Shutdown"
  numAddresses = 0
  for iname in sorted( portSecStatus.intfStatus ):
    intf = portSecStatus.intfStatus[iname]
   config = portSecConfig.intfConfig.get( iname )
    print format %( IntfCli.Intf.getShortname( iname ),
                    config.maxAddrs if config else 0,
                    intf.addrs,
                    intf.violations )
    numAddresses += intf.addrs
  print ("-------------------------------------------------------"
        "-------------------------")
  print "Total Addresses in System:", numAddresses
```

Clearly commented code

Command definition, in Python 'def' is the keyword used to define a function. When doShowPortSecurity is called this code is executed

We want to put our banner here, prior to the initial print statement.

Insert the code into the PortSecCli.py file.

```
[admin@7148S ~]$ sudo vi PortSecCli.py
#-----------------------------------------------------------------------
# The "show port-security" command, in "enable" mode.
#-----------------------------------------------------------------------
def doShowPortSecurity( mode ):
  print("This device is for authorized access of XYZ employees only!")
  print ("Secure Port      MaxSecureAddr  CurrentAddr  "
        "SecurityViolation  Security Action")
  print "                  (Count)       (Count)      (Count)"
  print ("-------------------------------------------------------------"
        "-------------------")
  format = "     %-7s    %7d    %10d    %9d           Shutdown"
```

Insert the print statement with the appropriate text

Now save and exit the editor, launch a new CLI session and look at the show port-security command.

```
[admin@7148S CliPlugin]$
[admin@7148S CliPlugin]$ Cli
7148S>en
7148S#sh port-security
This device is for authorized access on XYZ employees only!
Secure Port      MaxSecureAddr  CurrentAddr  SecurityViolation  Security Action
                 (Count)        (Count)      (Count)
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Total Addresses in System: 0
7148S#
```

Exit the editor and launch a new instance of the CLI

Enter enable mode and issue the 'show port-security' command

Your banner is now displayed as part of the show port-security command

Congratulations, you have made the first changes to the EOS CLI. As you can see you can make modifications to meet your critical operational needs.

Final touch

Now that changes have made you need to ensure that they are persistent between reboots, remember that accept for files in /mnt/flash files and volumes are contained in a temporary file system that is created and mounted at the boot of the device. If you do not copy the file(s) to /mnt/flash any changes you have made will be lost on a reboot of the switch.

```
[admin@7148S CliPlugin]$ cp /usr/lib/python2.6/site-packages/CliPlugin/PortSecCli.py
/mnt/flash/PortSecCliNEW.py

[[admin@7148S CliPlugin]$

[admin@7148S CliPlugin]$ ls -la /mnt/flash/PortSecCliNEW.py

-rwxrwx--- 1 root eosadmin 8752 Oct 21 18:46 /mnt/flash/PortSecCliNEW.py

[admin@7148S CliPlugin]$
```

Using the 'cp' command copy the file to /mnt/flash

Check to ensure the file was copied to /mnt/flash

Now that the file is on persistent storage, it is just a matter of ensuring that the file is automatically copied to /usr/lib/python2.6/site-packages/CliPlugin as part of the boot process. Leveraging some of the built-in feature of EOS, rc.eos, this is an easy process.

NOTE: rc.eos provides the hook to change EOS behavior early in the boot process.

In /mnt/flash, edit/create the file rc.eos, use a bash script to copy the modified PortSecCli.py from /mnt/flash to the original location at /usr/lib/python2.6/site-packages/CliPlugin

```
[admin@7148S flash]$ vi rc.eos
#!/bin/sh
. /etc/swi-version
if [ $SWI_VERSION == 4.5.3 ]; then
 echo SWI version match copying modified parser files
 cp /mnt/flash/PortSecCliNEW.py /usr/lib/python2.6/site-packages/CliPlugin/PortSecCli.py
else
 echo WARNING: SWI version has changed to $SWI_VERSION
fi
```

Open/create rc.eos

Read the /etc/swi-version file. This contains an environmental variable called SWI_VERSION, this indicated which version of software is currently running.

Error checking to ensure the version numbers match. See note below

If the s/w version matches then copy the modified file

CLI modifications should be restricted to the software version they were created, if not it could lead to unexpected results if the underlying CLI has changed. If you are going to perform an upgrade it is recommended you review any customizations and recreate them in the new created Cli Plugins.

NOTE: Arista allow's customers to customize and enable and install new and/or customized agents and services as a benefit. Of note however, is the fact that our support for these customizations is "best effort". If there is a system support issue, we may request that the customizations are temporarily disabled if we believe they are affecting basic system or switch forwarding operations.