SONIC CLI

Contents

[SONIC CLI 1](#_Toc514107807)

[1. Requirements 2](#_Toc514107808)

[2. Major design 2](#_Toc514107809)

[2.1. How to reuse current CLI? 3](#_Toc514107810)

[2.2. Support all commands of Quagga/FRR 3](#_Toc514107811)

[2.3. Different between SONIC CLI and VTYSH 5](#_Toc514107812)

[2.4. Authorization 5](#_Toc514107813)

# Requirements

1. Many questions about how to config.
2. Tacacs+ authorization.
3. Like traditional CLI to support:

Command completion by TAB, ? Help.

Configure mode, Interface mode, BGP mode.

1. No conflict with current SONIC commands and easy to reuse.

# Major design

ConfigDB

Quagga/FRR

Linux CLI

SONIC CLI

Host

BGP Docker

VTYSH

socket

APP Docker

Enforcer

APP

Tacacs+ Server

authorization

1. SONIC CLI is running on host environment.
2. Wrapping on current CLI to reuse it.
3. SONIC CLI is developing based on VTYSH and support all commands of Quagga or FRR.
4. Do command authorization by remote Tacacs+ server.

## How to reuse current CLI?

VTYSH has a good defined framework and easily to create a new command. SONIC CLI just call linux command by a pipe.

Example:

1. call existing SONIC comand

*DEFUN (show\_interface\_counters,*

*show\_interface\_counters\_cmd, ---> function name*

*"show interface counters", ---> command format*

*SHOW\_STR ---> help string*

*INTERFACE\_STR*

*"interfaces counters\n")*

*{*

*execute\_linux\_cmd\_pipe("show interface counters -a");*

*return CMD\_SUCCESS;*

*}*

1. call linux command

*DEFUN (show\_timezone,*

*show\_timezone\_cmd,*

*"show timezone",*

*SHOW\_STR*

*"timezone\n")*

*{*

*execute\_linux\_cmd\_pipe("timedatectl status");*

*return CMD\_SUCCESS;*

*}*

1. call existing SONIC command to config

*DEFUN (cli\_aaa\_auth\_login\_one,*

*aaa\_auth\_login\_one\_cmd,*

*"aaa authentication login (local|tacacs+)",*

*AAA\_HELP\_STR*

*AAA\_AUTHENTICATION\_HELP\_STR*

*AAA\_LOGIN\_HELP\_STR*

*"Local authentication\n"*

*"TACACS+ authentication\n")*

*{*

*EXEC\_CMD\_WITH\_PARAM("sudo config aaa authentication login %s", argv[0]);*

*if(author\_update\_config())*

*vty\_out(vty, "Update TACACS+ configuration fail\n");*

*return CMD\_SUCCESS;*

*}*

## Support all commands of Quagga/FRR

1. Compiling Quagga to generate a file named vtysh\_cmd.c under *src/sonic-quagga/vtysh/*
2. Copy it to *src/sonic-cli/vtysh/*
3. Get all commands of Quagga



What CLI actually has been done with Quagga?

1. CLI use vtysh\_cmd.c to get the command tree from Quagga.

*DEFSH (VTYSH\_ZEBRA, show\_ip\_route\_prefix\_cmd\_vtysh,*

*"show ip route A.B.C.D/M",*

*"Show running system information\n"*

*"IP information\n"*

*"IP routing table\n"*

*"IP prefix <network>/<length>, e.g., 35.0.0.0/8\n")*

1. CLI connects to vtysh server of zebra and bgpd.
2. CLI send command to zebra. zebra will execute the command and return result to CLI.
3. CLI printout the result.

bgpd

SONIC CLI

BGP Docker

VTYSH Server

socket

VTYSH Client

zebra

VTYSH Server

socket

1. Get a command

2. Send command

4. Return result

3. Execute command

5. Print result

## Different between SONIC CLI and VTYSH

1. SONIC CLI is an independent repository and can work without Quagga/FRR.
2. SONIC CLI is running in host linux and easily to call existing python CLI and Linux commands.
3. SONIC CLI supports authentication and authorization.

## Authorization

1. User enters to SONIC CLI by default when logins to switch.

/etc/passwd

yongcan.wyc:x:1003:1000:remote\_user:/home/yongcan.wyc:/usr/bin/cli

1. Get privilege level after do authentication by Tacacs+ server.
2. Create a command tree based on privilege level.

Admin user gets a tree of all commands.

Other user gets a tree without high risk commands. For example, there is no command to enter configure mode, also no command to enter to Linux.

1. Before executing any command, SONIC CLI will send it to Tacacs+ server to see if user has authorization to do it.
2. All users including no admin user join to the sudo group and docker group, as some show commonds need sudo or docker. So if user passes authorization by Tacacs+, it can show anything successfully.