用户指导

SONiC testbed使用指导

# 修改记录

|  |  |  |
| --- | --- | --- |
| 日期 | 版本 | 修改备注 |
| 2019-05-10 | V1.0.0 | 创建文档 |

目录

[修改记录 2](#_Toc1015985218)

[1 安装前准备 1](#_Toc2050832699)

[1.1 硬件 1](#_Toc645361702)

[1.2 软件 1](#_Toc794818157)

[2 Testbed环境安装 2](#_Toc971961026)

[2.1 操作系统安装 2](#_Toc880971793)

[2.2 系统环境准备 2](#_Toc1019017215)

[2.3 Testbed适配 4](#_Toc1663549314)

[2.4 Testbed安装 4](#_Toc1814852982)

[3 测试topo搭建 5](#_Toc2139798253)

[3.1 testbed文件准备 5](#_Toc422010400)

[3.2 topo启动 6](#_Toc1783795074)

[4 逻辑topo说明 7](#_Toc1407272182)

[4.1 t0-2拓扑 7](#_Toc1749268045)

[4.2 t1-4拓扑 8](#_Toc1159564362)

[5 执行测试 9](#_Toc1789367479)

[5.1 执行自动化测试例 9](#_Toc1995789816)

[5.2 手动执行测试用例 9](#_Toc663257653)

[附录 12](#_Toc1949110386)

# 安装前准备

## 硬件

安装系统需要有如下设备

表 1.1‑1

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 数量 | 型号 | 备注 |
| 服务器硬件台 | 1 |  | 至少一个管理网口  一块10G光口网卡  64G以上内存  1T以上硬盘  CPU至少32软核 |
| U盘 | 1 |  | 用于制作系统安装盘 |
| 交换机 | 1 |  | Fanout交换机，需要光口，32口以上 |
| 被测设备 | 1 |  | SONiC交换机 |
| 光模块和光纤线 | 根据需求 |  | 连接fanout交换机、DUT、server的光口 |

## 软件

**系统文件**

Ubuntu16.04的安装镜像（ISO文件），用于制作安装盘

# Testbed环境安装

目标是进入testbed命令行，安装好testbed依赖的组件

## 操作系统安装

物理服务器上安装Ubuntu16.04

**第一步：制作U盘镜像**

下载Ubuntu系统安装镜像文件到电脑

ubuntu@desktop$ ls

ubuntu-16.04.5-desktop-amd64.iso

寻找待烧录U盘，格式化U盘

ubuntu@desktop$ sudo fdisk -l

找到要烧录的U盘盘符（注意：此盘符上的信息会全部清除，不要错找为电脑系统盘符，或者其他盘符,假设我们需要烧录的盘符为/dev/sdb。）,找到盘符后通过下面命令进行格式化和镜像文件的烧录。

ubuntu@desktop$ sudo mkfs.vfat /dev/sdb -I

烧录文件到U盘

ubuntu@desktop$ sudo dd if=ubuntu-16.04.5-desktop-amd64.iso of=/dev/sdb

**第二步：服务器引导顺序设定为USB存储设备优先**

不同的硬件，boot顺序设置不一样，这里就不列出操作方式

**第三步：安装Ubuntu**

按照提示操作即可。网上有教程。安装完成后拔掉U盘，重启

## 系统环境准备

**配置用户**

在服务器上创建测试账户，或者使用已有的账户也行。假设账户和密码是sonic，sonic。

**配置管理桥**

假设服务器的管理口是eno2，管理ip设置为10.250.0.240/24，网关10.250.0.254

1.创建桥br1并把eno2加入br1。需要先安装brctl工具。

sonic@Asterfusion:~$ sudo brctl addbr br1

sonic@Asterfusion:~$ sudo brctl addif br1 eno2

2.给br1配置ip

sonic@Asterfusion:~$ sudo ip addr add 10.250.0.240/24 dev br1

3.修改配置文件,保证开机这些配置不丢失。

sonic@Asterfusion:~$ cat /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)

auto eno2

iface eno2 inet manual

auto br1

iface br1 inet static

bridge\_ports eno2

bridge\_stp off

bridge\_maxwait 0

bridge\_fd 0

address 10.250.0.240

netmask 255.255.255.0

gateway 10.250.0.254

dns-nameserver 8.8.8.8

dns-nameserver 114.114.114.114

**安装docker并导入image**

安装docker

导入sonic-mgmt、ptf的镜像

**veos镜像准备配置管理桥**

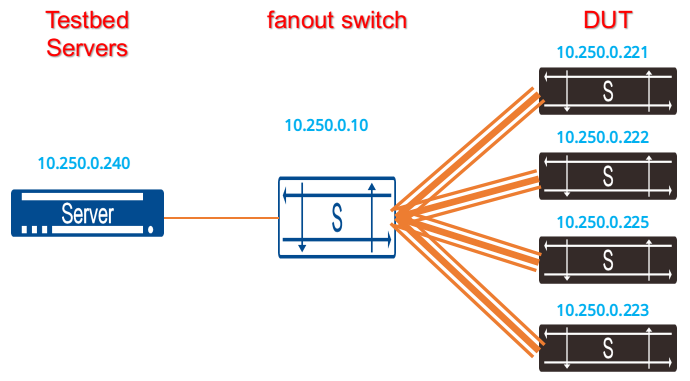
下载veos的image放到～/veos-vm/images/目录下

sonic@Asterfusion:~$ ls veos-vm/images/

Aboot-veos-serial-8.0.0.iso vEOS-lab-4.15.10M.vmdk

**物理topo连接**

当前实验室环境说明:



Server：做测试的发包平台，包括多个自动化测试仪和开发发包测试仪。不同测试仪的不同port的流量通过vlan区分，进行收发

Fanout：用QinQ实现L2VPN，为不同DUT的不同Port的流量提供单独通道，汇聚到Server的一个业务口

DUT ：被测设备，SONiC交换机

## Testbed适配

根据环境修改配置文件：系统环境适配和物理topo适配

物理topo适配：

系统环境适配：

## Testbed安装

**第一步：下载sonic-mgmt的工程**

ubuntu@desktop$ ls

ubuntu-16.04.5-desktop-amd64.iso

寻找待烧录U盘，格式化U盘

ubuntu@desktop$ sudo fdisk -l

找到要烧录的U盘盘符（注意：此盘符上的信息会全部清除，不要错找为电脑系统盘符，或者其他盘符,假设我们需要烧录的盘符为/dev/sdb。）,找到盘符后通过下面命令进行格式化和镜像文件的烧录。

ubuntu@desktop$ sudo mkfs.vfat /dev/sdb -I

烧录文件到U盘

ubuntu@desktop$ sudo dd if=ubuntu-16.04.5-desktop-amd64.iso of=/dev/sdb

**第二步：服务器引导顺序设定为USB存储设备优先**

不同的硬件，boot顺序设置不一样，这里就不列出操作方式

**第三步：安装Ubuntu**

按照提示操作即可。网上有教程。安装完成后拔掉U盘，重启

完成后，可以修改ansibleplaybook，避免后续使用需要重复的联网检查（一般测试环境没有连外网）。

# 测试topo搭建

搭建出一个虚拟topo，准备对DUT做测试

## testbed文件准备

主要是确定虚拟topo的信息：对应的dut，topo类型等等

先从目前testbed.csv文件中已有的testbed中选一个用

hubo@Asterfusion:~$ docker exec -it testbed bash

oxl@52db682e4d9d:~$ cd sonic-mgmt/ansible

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cat testbed.csv

# conf-name,group-name,topo,ptf\_image\_name,ptf\_ip,server,ex\_if,vm\_base,dut,comment

Ps3-oxl1,oxl1,ptf6,docker-ptf,10.250.0.184/24,server\_2,enp130s0f0,,CX306P-48S-3,Tests aster switch

Twz-t12,wz-t12,t1-2,docker-ptf,10.250.0.185/24,server\_1,enp130s0f1,VM0100,48S-wz,Tests aster switcht0-2,

Twz-t14,wz-t14,t1-4,docker-ptf,10.250.0.185/24,server\_1,enp130s0f1,VM0100,48S-wz,Tests aster switch

Twz-t02,wz-t02,t0-2,docker-ptf,10.250.0.185/24,server\_1,enp130s0f1,VM0100,48S-wz,Tests aster switch

Twz-t1la,wz-t1la,t1-lag,docker-ptf,10.250.0.185/24,server\_1,enp130s0f1,VM0100,48S-wz,Tests aster switch

Tlz-t02,lz-t02,t0-2,docker-ptf,10.250.0.186/24,server\_1,enp130s0f1,VM0104,48S-lz,Tests aster switch

Tlz-t14,lz-t14,t1-4,docker-ptf,10.250.0.186/24,server\_1,enp130s0f1,VM0104,48S-lz,Tests aster switch

Tsdd-t02,sdd-t02,t0-2,docker-ptf,10.250.0.187/24,server\_1,enp130s0f1,VM0108,48S-sdd,Tests aster switch

Tsdd-t12,sdd-t12,t1-2,docker-ptf,10.250.0.187/24,server\_1,enp130s0f1,VM0108,48S-sdd,Tests aster switch

Tsdd-t14,sdd-t14,t1-4,docker-ptf,10.250.0.187/24,server\_1,enp130s0f1,VM0108,48S-sdd,Tests aster switch

Tsdd-t1la,sdd-t1la,t1-lag,docker-ptf,10.250.0.187/24,server\_1,enp130s0f1,VM0108,48S-sdd,Tests aster switch

Toxl-t02,oxl-t02,t0-2,docker-ptf,10.250.0.188/24,server\_1,enp130s0f1,VM0112,48Y-oxl,Tests aster switch

Toxl-t02是唯一标志此次要搭建的测试topo；oxl-t02定位本测试top的容器；t0-2测试topo的逻辑topo；

Docker-ptf是所使用的daoker镜像名字；；；；48Y-oxl是被测设备的主机名(这个如果使用交换机不同，要修改成不同的交换机名）。

Toxl-t14,oxl-t14,t1-4,docker-ptf,10.250.0.188/24,server\_1,enp130s0f1,VM0112,48Y-oxl,Tests aster switch

Ttf-t02,tf-t02,t0-2,docker-ptf,10.250.0.184/24,server\_1,enp130s0f1,VM0116,48S-tf,Tests aster switch

Ttf-t14,tf-t14,t1-4,docker-ptf,10.250.0.184/24,server\_1,enp130s0f1,VM0116,48S-tf,Tests aster switch

选择Toxl-t02这个testbed来说：

Toxl-t02 ：唯一标识这个此次要搭建的测试topo

oxl-t02 ：不能超过8个字符，可以定位出本测试topo的ptf容器是ptf\_oxl-t02

t0-2 ：测试topo的逻辑topo，具体情况下一章说明

docker-ptf ：ptf所使用的docker image的name

10.250.0.188/24 ：ptf的docker的管理ip

server\_1 ：测试topo使用的server标识

enp130s0f1 ：测试topo使用server上的哪个网口收发测试流量

VM0112 ：测试topo使的一批虚拟机从哪个开始

48Y-oxl ：被测设备的主机名

## topo启动

**第一步：启动vms**

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ ./testbed-cli.sh start-vms Toxl-t02 password.txt

**第二步：add topo**

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ ./testbed-cli.sh add-topo Toxl-t02 password.txt

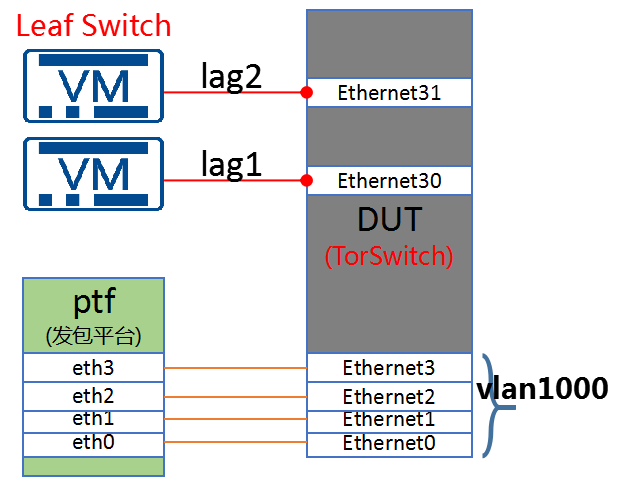
**第三步：deploy 将拓扑部署到交换机上**

./testbed-cli.sh deploy-mg Toxl-t02 lab password.txt

# 逻辑topo说明

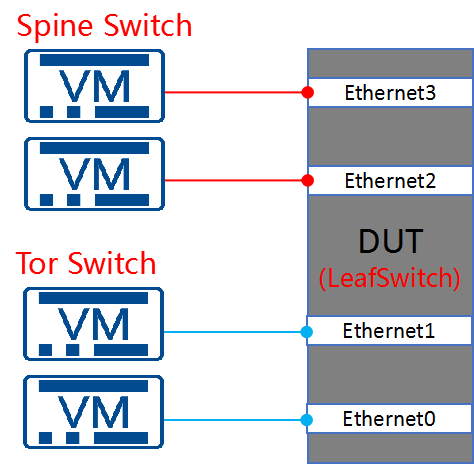
在物理server中，使用veos、ptf等构建虚拟网络，模拟数据中心的交换机和运行的服务器，形成逻辑的拓扑，与DUT对接，进行测试

## t0-2拓扑



这个拓扑模拟：DUT做Tor交换机，向上通过LAG连接Leaf交换机，向下二层连接服务器(ptf)

## t1-4拓扑



这个拓扑模拟：DUT做Leaf交换机，上接Spine交换机，下接Tor交换机

# 执行测试

## 执行自动化测试例

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ ansible-playbook -i lab -l 48Y-oxl test\_sonic.yml -e testbed\_name=Toxl-t14 -e testcase\_name=vlan

说明：

-l 48Y-oxl ：指定被测设备

test\_sonic.yml ：测试脚本入口

testbed\_name=Toxl-t14 ：见3.1

testcase\_name=vlan ：测试用例名字，见文件roles/test/vars/testcases.yml。列出了所有的自动化测 试用例名

## 手动执行测试用例

进入ptf容器，进行报文收发，手动测试。testbed\_name对应的ptf容器名是ptf\_{testbed\_name}，如上面例子中的testbed\_name是Toxl-t14，它的对应的ptf容器名是ptf\_Toxl-t14

hubo@Asterfusion:~$ docker exec -it ptf\_oxl-t02 bash

root@c5b437ade83b:/# ip link show

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

208: mgmt@if209: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group default qlen 1000

link/ether c2:cd:8d:e3:01:cd brd ff:ff:ff:ff:ff:ff

210: eth47@if211: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 9216 qdisc noqueue state UP mode DEFAULT group default qlen 1000

link/ether 22:59:dd:15:13:88 brd ff:ff:ff:ff:ff:ff

212: eth46@if213: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 9216 qdisc noqueue state UP mode DEFAULT group default qlen 1000

link/ether 0a:ec:dd:cc:e8:32 brd ff:ff:ff:ff:ff:ff

220: eth0@if13: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state LOWERLAYERDOWN mode DEFAULT group default qlen 1000

link/ether 68:91:d0:61:cb:35 brd ff:ff:ff:ff:ff:ff

221: eth1@if13: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state LOWERLAYERDOWN mode DEFAULT group default qlen 1000

link/ether 68:91:d0:61:cb:35 brd ff:ff:ff:ff:ff:ff

222: eth2@if13: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state LOWERLAYERDOWN mode DEFAULT group default qlen 1000

link/ether 68:91:d0:61:cb:35 brd ff:ff:ff:ff:ff:ff

223: eth3@if13: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state LOWERLAYERDOWN mode DEFAULT group default qlen 1000

link/ether 68:91:d0:61:cb:35 brd ff:ff:ff:ff:ff:ff

root@c5b437ade83b:/#

ptf内的接口与DUT的接口的对应关系见第3章的逻辑topo说明。接口可以按照自己的需要配置mac和ip等信息。

ptf内可以用scapy进行报文构造和发送，也可以直接使用ptf里面已有的工具发包，如ping

# 附录

### **怎样添加dut到testbed**

确认DUT的信息

1.要添加的DUT设备管理ip是10.250.0.227/24，设备类型是CX306P-48S，设备主机名48S-wz

注意，设备名称格式为设备类型加人名首字母。

2.DUT的Ethernet0～Ethernet3连接fanout switch（设备名str-7260-10）的XGE1/0/1~XGE1/0/4口，Ethernet46～Ethernet47连接str-7260-10的XGE1/0/5~XGE1/0/6口

修改sonic\_lab\_devices.csv和sonic\_lab\_links.csv并生成lab\_connection\_graph.xml

1.编辑sonic\_lab\_devices.csv加上48S-wz信息

oxl@52db682e4d9d:~/sonic-mgmt/ansible/files$ cat sonic\_lab\_devices.csv

Hostname,ManagementIp,HwSku,Type

48Y-oxl,10.250.0.221/24,CX306P-48Y,DevSonic

48S-wz,10.250.0.227/24,CX306P-48S,DevSonic

48S-sdd,10.250.0.222/24,CX306P-48S,DevSonic

48S-lz,10.250.0.227/24,CX306P-48S,DevSonic

48S-tf,10.250.0.223/24,CX306P-48S,DevSonic

str-7260-10,10.250.0.243/24,H3C,FanoutLeaf

str-7260-11,10.250.0.244/24,H3C,FanoutRoot

sonic,10.250.0.240/24,TestServ,Server

2.编辑sonic\_lab\_links.csv加上设备连线信息

oxl@52db682e4d9d:~/sonic-mgmt/ansible/files$ cat sonic\_lab\_links.csv

StartDevice,StartPort,EndDevice,EndPort,BandWidth,VlanID,VlanMode

48S-wz,Ethernet0,str-7260-10,XGE1/0/1,10000,1001,Access

48S-wz,Ethernet1,str-7260-10,XGE1/0/2,10000,1002,Access

48S-wz,Ethernet2,str-7260-10,XGE1/0/3,10000,1003,Access

48S-wz,Ethernet3,str-7260-10,XGE1/0/4,10000,1004,Access

48S-wz,Ethernet46,str-7260-10,XGE1/0/5,10000,1005,Access

48S-wz,Ethernet47,str-7260-10,XGE1/0/6,10000,1006,Access

48S-lz,Ethernet0,str-7260-10,XGE1/0/7,10000,1007,Access

.......

48S-tf,Ethernet47,str-7260-10,XGE1/0/30,10000,1030,Access

str-7260-11,XGE1/0/48,sonic,enp130s0f1,10000,1001-1047,Trunk

str-7260-11,XGE1/0/47,str-7260-10,XGE1/0/48,10000,1001-1047,Trunk

3.生成lab\_connection\_graph.xml

oxl@52db682e4d9d:~/sonic-mgmt/ansible/files$ ./creategraph.py -o lab\_connection\_graph.xm

### **怎样新增一个fanout switch**

...

### **怎样新增一个自定义topo**

topo只能从现有的t0、t1、ptf来拓展。不能超过这三种类型的范畴。因为当前testbed环境的自动化搭建脚本只适配了这三类topo

**拓展t0拓扑**

增加t0-2拓扑，DUT做Tor交换机，上行接2个leaf交换机，下行接4个server host

1.拷贝t0拓扑文件为t0-2，然后修改t0-2，主要是去掉多于的接口和vm。内容如下

oxl@52db682e4d9d:~$ cd sonic-mgmt/ansible

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cat vars/topo\_t0-2.yml

topology:

host\_interfaces:

- 0

- 1

- 2

- 3

VMs:

ARISTA01T1:

vlans:

- 46

vm\_offset: 0

ARISTA02T1:

vlans:

- 47

vm\_offset: 1

configuration\_properties:

common:

dut\_asn: 65100

dut\_type: ToRRouter

swrole: leaf

podset\_number: 200

tor\_number: 16

tor\_subnet\_number: 2

max\_tor\_subnet\_number: 16

tor\_subnet\_size: 128

spine\_asn: 65534

leaf\_asn\_start: 64600

tor\_asn\_start: 65100

failure\_rate: 0

nhipv4: 10.10.246.100

nhipv6: FC0A::C9

configuration:

ARISTA01T1:

properties:

- common

bgp:

asn: 64600

peers:

65100:

- 10.0.0.56

- FC00::71

interfaces:

Loopback0:

ipv4: 100.1.0.29/32

ipv6: 2064:100::1d/128

Ethernet1:

lacp: 1

Port-Channel1:

ipv4: 10.0.0.57/31

ipv6: fc00::72/126

bp\_interface:

ipv4: 10.10.246.29/24

ipv6: fc0a::3a/64

ARISTA02T1:

properties:

- common

bgp:

asn: 64600

peers:

65100:

- 10.0.0.58

- FC00::75

interfaces:

Loopback0:

ipv4: 100.1.0.30/32

ipv6: 2064:100::1e/128

Ethernet1:

lacp: 1

Port-Channel1:

ipv4: 10.0.0.59/31

ipv6: fc00::76/126

bp\_interface:

ipv4: 10.10.246.30/24

ipv6: fc0a::3d/64

2.修改veos文件，topologies变量中增加t0-2

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cat veos

[vm\_host\_1]

STR-ACS-SERV-01 ansible\_host=10.250.0.240

......

[servers:vars]

topologies=['t1','t1-2', 't1-4','t1-lag', 't1-64-lag', 't0','t0-2', 't0-16', 't0-56', 't0-52', 'ptf32', 'ptf64', 't0-64', 't0-64-32', 't0-116']

3.eos的配置模板文件中，增加t0-2的topo下的leaf交换机的配置模板

直接从t0的配置模板拷贝

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cd roles/eos/templates/

oxl@52db682e4d9d:~/sonic-mgmt/ansible/roles/eos/templates$ cp t0-leaf.j2 t0-2-leaf.j2

oxl@52db682e4d9d:~/sonic-mgmt/ansible/roles/eos/templates$ ls

t0-116-leaf.j2 t0-2-leaf.j2 t0-56-leaf.j2 t0-64-leaf.j2 t1-2-spine.j2 t1-4-spine.j2 t1-4-tor.j2cp t1-64-lag-tor.j2 t1-lag-tor.j2 t1-tor.j2

t0-16-leaf.j2 t0-52-leaf.j2 t0-64-32-leaf.j2 t0-leaf.j2 t1-2-tor.j2 t1-4-tor.j2 t1-64-lag-spine.j2 t1-lag-spine.j2 t1-spine.j2

**拓展t1拓扑**

增加t1-4拓扑，DUT做Leaf交换机，上行接2个Spine交换机，下行接2个Tor交换机

1.拷贝t1拓扑文件为t1-4，然后修改t1-4，主要是去掉多于的vm。内容如下

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cat vars/topo\_t1-4.yml

topology:

VMs:

ARISTA01T2:

vlans:

- 0

vm\_offset: 0

ARISTA02T2:

vlans:

- 1

vm\_offset: 1

ARISTA01T0:

vlans:

- 2

vm\_offset: 2

ARISTA02T0:

vlans:

- 3

vm\_offset: 3

configuration\_properties:

common:

dut\_asn: 65100

dut\_type: LeafRouter

nhipv4: 10.10.246.100

nhipv6: FC0A::C9

spine:

swrole: spine

podset\_number: 200

tor\_number: 16

tor\_subnet\_number: 2

leaf\_asn\_start: 62001

tor\_asn\_start: 65501

failure\_rate: 0

tor:

swrole: tor

tor\_subnet\_number: 5

configuration:

ARISTA01T2:

properties:

- common

- spine

bgp:

asn: 65200

peers:

65100:

- 10.0.0.0

- FC00::1

interfaces:

Loopback0:

ipv4: 100.1.0.1/32

ipv6: 2064:100::1/128

Ethernet1:

ipv4: 10.0.0.1/31

ipv6: fc00::2/126

bp\_interface:

ipv4: 10.10.246.1/24

ipv6: fc0a::2/64

ARISTA02T2:

properties:

- common

- spine

bgp:

asn: 65200

peers:

65100:

- 10.0.0.2

- FC00::5

interfaces:

Loopback0:

ipv4: 100.1.0.2/32

ipv6: 2064:100::2/128

Ethernet1:

ipv4: 10.0.0.3/31

ipv6: fc00::6/126

bp\_interface:

ipv4: 10.10.246.2/24

ipv6: fc0a::5/64

ARISTA01T0:

properties:

- common

- tor

tornum: 1

bgp:

asn: 64001

peers:

65100:

- 10.0.0.32

- FC00::41

vips:

ipv4:

prefixes:

- 200.0.1.0/26

asn: 64700

interfaces:

Loopback0:

ipv4: 100.1.0.17/32

ipv6: 2064:100::11/128

Ethernet1:

ipv4: 10.0.0.33/31

ipv6: fc00::42/126

bp\_interface:

ipv4: 10.10.246.17/24

ipv6: fc0a::22/64

ARISTA02T0:

properties:

- common

- tor

tornum: 2

bgp:

asn: 64002

peers:

65100:

- 10.0.0.34

- FC00::45

vips:

ipv4:

prefixes:

- 200.0.1.0/26

asn: 64700

interfaces:

Loopback0:

ipv4: 100.1.0.18/32

ipv6: 2064:100::12/128

Ethernet1:

ipv4: 10.0.0.35/31

ipv6: fc00::46/126

bp\_interface:

ipv4: 10.10.246.18/24

ipv6: fc0a::25/64

2.修改veos文件，topologies变量中增加t1-4

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cat veos

[vm\_host\_1]

STR-ACS-SERV-01 ansible\_host=10.250.0.240

......

[servers:vars]

topologies=['t1','t1-2', 't1-4','t1-lag', 't1-64-lag', 't0','t0-2', 't0-16', 't0-56', 't0-52', 'ptf32', 'ptf64', 't0-64', 't0-64-32', 't0-116']

3.eos的配置模板文件中，增加t1-4的topo下的tor交换机和spine交换机的配置模板

直接从t1的配置模板拷贝

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cd roles/eos/templates/

oxl@52db682e4d9d:~/sonic-mgmt/ansible/roles/eos/templates$ cp t1-tor.j2 t1-4-tor.j2

oxl@52db682e4d9d:~/sonic-mgmt/ansible/roles/eos/templates$ cp t1-spine.j2 t1-4-spine.j2

oxl@52db682e4d9d:~/sonic-mgmt/ansible/roles/eos/templates$ ls

t0-116-leaf.j2 t0-2-leaf.j2 t0-56-leaf.j2 t0-64-leaf.j2 t1-2-spine.j2 t1-4-spine.j2 t1-4-tor.j2cp t1-64-lag-tor.j2 t1-lag-tor.j2 t1-tor.j2

t0-16-leaf.j2 t0-52-leaf.j2 t0-64-32-leaf.j2 t0-leaf.j2 t1-2-tor.j2 t1-4-tor.j2 t1-64-lag-spine.j2 t1-lag-spine.j2 t1-spine.j2

### **怎样新增一个自定义testbed项**

首先确认测试资源：

VM ：每个人都会分配四个vm（由管理员统一分配）。如wangzhui分配的四个vm是VM0100～VM0103

Ptf ip ：一个ptf的ip（由管理员统一分配）。如wangzhui分配的ptf的ip是10.250.0.185/24

DUT ：自己要测试的DUT设备名。如，48S-wz

Topo ：要在哪种逻辑topo下测试。由于测试资源限制，只支持占用的测试口不多于6个的topo。目前有t0-2，t1-2，t1-4

然后确认testbed的各项的值：

|  |  |
| --- | --- |
| conf-name | 格式遵照T{人名拼音首字母}-{topo名去掉横线的缩写}，如Twz-t02 |
| group-name | 格式遵照{人名拼音首字母}-{topo名去掉横线的缩写}，如wz-t02 |
| topo | 上面确认好的Topo |
| ptf\_image\_name | 当前只有docker-ptf。ptf的docker image |
| ptf\_ip | 上面确认好的Ptf ip |
| server | 填server\_1即可。server\_1见veos文件 |
| ex\_if | 表示测试流量从服务器的哪个口出去可以到达DUT。可以看DUT所连的fanout 交换机是跟服务器哪个口连接的 |
| vm\_base | 上面确认的分配给自己的VM里面的第一个vm名。如分配的VM是VM0100～VM0103，vm\_base就是VM0100 |
| dut | 上面确认好的DUT |
| comment | 注释信息，随意填写 |

然后按照testbed.csv文件要求的顺序，在testbed.csv文件填入各项值。如

oxl@52db682e4d9d:~/sonic-mgmt/ansible$ cat testbed.csv

# conf-name,group-name,topo,ptf\_image\_name,ptf\_ip,server,ex\_if,vm\_base,dut,comment

...

Twz-t02,wz-t02,t0-2,docker-ptf,10.250.0.185/24,server\_1,enp130s0f1,VM0100,48S-wz,Tests aster switch

...

### **怎样新增一个testcase**

...

==============================================================

**关于星融元数据**

星融（Asterfusion Data Technologies）作为新一代云网络架构的解决方案提供商，以领先的、真正意义上的软件定义网络方案，帮助用户重新定义云计算的网络基础设施，与生态合作伙伴共同践行“让云网络回归网络”的梦想。凭借所拥有的专利技术，基于高度软件定义的整体架构、完全开放透明的操作系统、突破传统限制的硬件平台，星融为云计算的使用者赋予真正弹性与超高性能的虚拟网络，为云计算的管理者提供纯粹的开放接口和自动部署调度，为云计算的开发者构建可编程和可视化的业务环境。获取更多有关星融公司、产品及解决方案的信息，请登录[www.asterfusion.com](mailto:www.asterfusion.com)，或关注星融官方微信。销售咨询，请发送邮件至[sales@asterfusion.com](mailto:sales@asterfusion.com)。

“星融”、“Asterfusion”、“PICFA”、“Cloud Scalability Matrix”、“Cloud Visibility Matrix”及其徽标均为星融元数据技术（苏州）有限公司在中国的商标或注册商标。其他所有商标为其各自所有者之财产。本文件所包含的信息可能会发生修改，恕不另行通知。未经书面许可，本文件所含内容不作为合同或许可证的一部分。

|  |  |  |  |
| --- | --- | --- | --- |
| **联系方式** | | | |
| **苏州**  0512-62982976  苏州市工业园区星湖街  328号创意产业园  2栋B401 | **北京**  010-62672668  北京市海淀区宝盛南路  1号奥北科技园  20号楼207 | **西安**  029-89834058  西安市新城区幸福南路  109号老钢厂设计创意产业园  5号楼5-212 | **武汉**  武汉市武昌区松竹路28号万达环球国际中心  4号楼2603 |