sonic-buildimage NOTE

注：本文档参考github上sonic-buildimage

构建sonic交换机镜像

概述

以下这些指令是为网络交换机构造了NOS(network operating system)以及在NOS内部运行Docker镜像。注意Sonic镜像构建在每个芯片平台的。同一芯片平台的交换机使用共同的镜像。

硬件平台

任意的服务都能在镜像服务上构建。我们使用1t的硬盘，OS使用Ubuntu16.04.

准备工作

主机上安装pip和jinja，如果j2/j2cli不可用，执行以下指令：

sudo apt-get install -y python-pip

sudo python2 -m pip install -U pip==9.0.3

sudo pip install --force-reinstall --upgrade jinja2>=2.10

sudo pip install j2cli

SAI版本

参考sonic-roadmap

Clone或者获取所有子模块的仓库代码

Clone网址:git clone https://github.com/Azure/sonic-buildimage.git

但是由于公司项目的需要，公司自己是有一个sonic-buildimage项目的git库的，网址是（使用192.168.10.1代理下载）：https://git.teraspek.cn

使用下面的方法做编译：（具体见滕飞文档，使用10.2进行编译，因为需要从外网下载东西）

(Optional) Checkout a specific branch. By default, it uses master branch

git checkout [branch_name]（注意要在201811分支上，不能使用master）

Execute make init once after cloning the repo, or after fetching remote repo with submodule updates

make init

Execute make configure once to configure ASIC

make configure PLATFORM=[ASIC_VENDOR]

Build SONiC image

make all

ARM架构的使用方法

为（ARMHF）平台构造ARM32位【armhf：arm hard float, armhf传参用的是fpu寄存器，浮点运算性能更高】

Execute make configure once to configure ASIC and ARCH、

make configure PLATFORM=[ASIC_VENDOR] PLATFORM_ARCH=armhf

example

make configure PLATFORM=marvell-armhf PLATFORM_ARCH=armhf

构造为平台构造arm64位

Execute make configure once to configure ASIC and ARCH

make configure PLATFORM=[ASIC_VENDOR] PLATFORM_ARCH=arm64

example

make configure PLATFORM=marvell-arm64 PLATFORM_ARCH=arm64

Note：

- 需要预留50G的空间去构建一个平台；
- 如果Docker的工作区（/var/lib/docker)所在的分区没有足够的空间，可能会出现下面的错误。

  /usr/bin/tar: /path/to/sonic-buildimage/<some_file>: Cannot write: No space left on device

  解决方法是转移到一个有足够多空间的分区。
- 使用http_proxy = [your_proxy] https_proxy = [your_proxy] make在构建过程中启用http（s）代理。
- 不支持root或者sudo，将自己的账户加到Docker组中，使用自己的账户去做编译。

> 所有的Docker镜像（镜像生成容器，就和类与实例的关系大概一致）都需要sonic的安装容器。Sonic为一个芯片提供商的所有设备使用一个镜像（就是说一个厂商的芯片对应一个特定的镜像）。支持的芯片有broadcom marvell mellanox cavium centec nephos p4 vs 现在公司的下一个芯片选择的是barefoot，完全可编程芯片。
>
> ---
>
> 对于broadcom芯片，我们构造ONIE和EOS镜像。Arista（这个公司很牛逼，在SDN领域）的设备使用的是EOS镜像。其余所有基于Broadcom芯片的设备使用ONIE镜像。
>
> make configure PLATFORM=broadcom
>
> #build debian stretch required targets
>
> BLDENV=stretch make stretch
>
> # build ONIE image
>
> make target/sonic-broadcom.bin
>
> # build EOS image
>
> make target/sonic-aboot-broadcom.swi
>
> 执行这上面的操作构造生成broadcom的ONIE镜像和EOS镜像。不过前提是make init执行之后。【这样看的话，目前我们是基于cavium芯片所以生成的是sonic-cavium.bin这个镜像文件是基于cavium芯片的，那么这个镜像的作用是基础镜像，所有后面的工作都是在这个镜像上完成的】

Rules/config文件是有自己的功能作用。这个文件中包含着构建过程的配置选项，比　　如添加冗余或者显示依赖、用户名、和基础镜像的密码等。

每个Docker镜像能够构建和保存到target下的目录中，利于只构造Docker-database，只需要执行：make target/docker-database.gz，【单独编译】，deb包的编译也是类似，指定路径和名称就行。

如果需要删除一个目标文件，比如删除swss就可以执行make target/debs/swss_1.0.0_amd64.deb-clean，就可以将这个文件删除。建议使用清理目标文件来清楚一块构造的包，比如dev包。参考下面的**Documentation**来了解相似的构造过程和如何做出修改。【这个documentation很重要,编译的文件架构在其中】

构建调试Docker并调试sonic安装镜像

Sonic构造系统支持使用debug工具和debug符号构造dockers和one-image，可以帮助，以帮助进行实时和核心调试。 有关详细信息，请参阅（SONiC Buildimage指南）。

Note

- 如果你是第一次运行编译，sonic-slave-{USER}的docker镜像会自动创建。会花一点时间，但是这是一次行为，所以保持耐心。
- Root账户不可用，但是可以使用sudo
- 目标路径是Target/，包括NOS的安装镜像和docker镜像。
    - Sonic-generic.bin：sonic交换机的安装镜像（兼容onie）
    - Sonic-aboot.bin:sonic交换机安装镜像（兼容Aboot）
    - Docker-base.gz:其他Docker镜像创建的基础镜像，仅在构造进程中使用（gzip tar存档）
    - Docker-database.gz:为内存键值存储的Docker镜像，当内部进程相互交流时时使用（gzip tar 存档）
    - Docker-fpm.gz:docker-fpm.gz：使用fpm模块开启的Quagga的docker映像（gzip tar存档）【Quagga是一个路由软件套件，为Unix平台提供OSPFv2，OSPFv3，RIP v1和v2，RIPng和BGP-4的实现，特别是FreeBSD，Linux，Solaris和NetBSD。 Quagga是由Kunihiro Ishiguro开发的GNU Zebra的分支。】
    - Docker-orchagent.gz: SWitch State Service(SWSS)的Docker镜像（gzip tar存档）
    - Docker-syncd-brcm.gz：守护进程Docker镜像同步数据库和Broadcom交换芯片（gzip tar存档）
    - docker-syncd-cavm.gz: 守护进程Docker镜像同步数据库和cavium交换芯片（gzip tar存档）
    - Docker-syncd-mlnx.gz:守护进程Docker镜像同步数据库Mellanox交换芯片（gzip tar存档）
    - Docker-syncd-nephos.gz:守护进程Docker镜像同步数据库和Nephos交换芯片（gzip tar存档）
    - Docker-sonic-p4.gz：all-in-one(应该是说所有的东西封装在一个镜像中），用于p4软件交换的多功能一体的Docker镜像（gzip tar存档）
    - docker-sonic-vs.gz：用于软件虚拟交换机的多功能一体机的docker镜像（gzip tar archive）