Building an NVIDIA Verified SONiC Image

NVIDIA is committed to your success when you choose SONiC (Software for Open Networking in the Cloud), the free, community-developed, Linux-based network operating system (NOS) hardened in the data centers of some of the largest cloud service providers. SONiC is an ideal choice for centers looking for a low-cost, scalable, and fully controllable NOS without sacrificing functionality or security. It offers all the standard networking protocols developers should need and is constantly evolving with new features and updates.

While greatly advantageous in many ways, open-source projects bring their own set of disadvantages. SONiC is built using many third-party packages working together, leading to version incompatibility and security concerns. And as with any complex, collaborative project, version control can get chaotic. Many contributors to the SONiC project push commits every day, contributing further to potential mismatching. Updates to the SONiC repository can even cause a build to fail when external dependencies change. When you update SONiC or install it for the first time, you need to ensure that code is well-tested, secure, and configured properly.

Reproducible Builds

Reproducible builds address many of the concerns raised by open-source software. They allow you to build the exact same binary based on source code and external dependencies fully verified by the networking experts at NVIDIA. With a reproducible

SONiC build verified by NVIDIA, you can be confident your build is carefully tested and secured through extensive quality assurance. NVIDIA highly recommends this process when building your own SONiC image.

Security: Reproducible builds add an extra layer of security and trustworthiness within the open-source community, where anyone can view and modify source code. They verify the source matches the distributed SONiC binary.

Stability: A SONiC build may fail even when there are no code changes, often due to changes in external dependencies. Reproducible builds allow you to lock package versions, including slave docker images, host images, Debian packages, and more, so you can expect the same result every time you build SONiC from a verified NVIDIA hash.

They ensure the software can be rebuilt in the future, even if the original build environment or tools are no longer available or have changed.

Development: When you need to trace and fix issues, reproducible builds ease the debugging process by ensuring your SONiC build is consistent.

Accessing NVIDIA Verified Hashes

NVIDIA pushes qualified source code to the SONiC GitHub. Commit hashes, pull request requirements, and external dependency versions are communicated to subscribed users on a regular basis as SONiC GA versions are released. Detailed release notes, user manuals, and links to the build hashes on GitHub for each hash release can be found on the Enterprise Support Portal.

Building your own SONiC image

Any server can be a build server for a SONiC image with enough resources. Current hardware recommendations can be found here. Our build server consists of a virtual Ubuntu 20.04 server with 8 CPU cores, 32GB RAM and 300GB hard disk space. We are using Docker version 27.0.3. See further detailed instructions on the official GitHub readme for building SONiC images.

Let's build a custom ONIE compatible SONiC image with some tweaked parameters based on a qualified hash from NVIDIA. SONiC offers many build configuration options in the rules/config file. You can edit them in the file directly, but for this build we will input them as parameters in the final build instruction.

We will enable ZTP in our build, which is disabled by default as decided by the community. Add this line to the build instruction:

```
ENABLE_ZTP=y
```

We will also tell SONiC to execute build jobs in parallel for a faster build time. Our build server has 8 CPU cores, so we can execute up to 8 jobs in parallel:

```
SONIC_CONFIG_BUILD_JOBS=8
```

Check out the NVIDIA qualified branch and hash after cloning the repo. If you do not specify a branch or commit hash, git will use the master branch and download the most recent dependencies, which is not recommended for production environments.

```
git checkout 202311 156b067c
```

The final build instruction will result in a ONIE compatible binary image by specifying the target target/sonic-vs.bin in the build instruction. You can build other image types if you need:

```
target/sonic-vs.img.gz
target/sonic-vs.raw
target/docker-sonic-vs.gz
```

The Steps

1. Install Docker and configure your system to allow running the docker command without sudo. For installing on an Ubuntu system, follow the specific Ubuntu steps here.
2. Install dependencies.
   sudo apt install -y python3-pip
   pip3 install --user j2cli
3. Clone the repo.
   git clone --recurse-submodules https://github.com/sonic-net/sonic-buildimage.git
4. Build the image.

```
sudo modprobe overlay

cd sonic-buildimage

# Checkout a specific branch. It is highly recommended by NVIDIA to
use a hash qualified by NVIDIA.
# git checkout [branch] [commit hash]
git checkout 202311 156b067c

make init

# Configure your ASIC platform of choice. This may take several
hours.
# make configure PLATFORM=[ASIC_VENDOR]
make configure PLATFORM=mellanox

# Run the build instruction. Build the image with ZTP enabled and 8
parallel build jobs for faster building. This may take several hours.
ENABLE_ZTP=y SONIC_CONFIG_BUILD_JOBS=8 make target/sonic-vs.bin
```

Open Networking

These instructions result in a ONIE compatible, custom SONiC image with ZTP enabled. One could argue this process demonstrates open networking at its best: building an open-source operating system and eliminating vendor dependence for good.