Introduction to Network Emulation and Requirements for Virtual Network Devices

By Damien Garros

December 3, 2019

(Published as a blog post here: https://blog.networktocode.com/post/Network_Emulation/)

Network Emulation consists of (re)creating a network in a virtual environment using virtual devices. Network engineers have been using Network Emulation for training and demos for some time now, but some new use cases are emerging around Continuous Integration and Continuous Delivery (CI/CD) of the network itself or the tools around it.

I've seen a lot of interest from both big and small companies as Network Emulation aligns well with the introduction of DevOps principles in Networking (NetDevOps). Some of the largest barriers are the quality of the virtual images, which are emulating production network devices, obtained from the network vendors and the lack of proper tools available to create large emulated environments.

This is a vast topic with many different use cases and tools–as a result there are often misunderstandings, which can result in unclear specifications and requirements. However, without clear requirements, the need for better Virtual Network Devices is not clearly communicated back to the vendors and there is no path to get out of the current situation.

In this blog, I'll cover the most common use cases for Network Emulation and how Network Emulation differs from Network Virtualization or Network Simulation, which are two different technical solutions that are often conflated with Network Emuation. In the second part of the blog will take a deeper look at the Virtual Network Device: what are their main differences with their physical counterpart and what you should look for when evaluating a Virtual Network Device for Network Emulation.

> *Disclaimer*: My background as a network engineer is largely based in Datacenter Switching and I've been using Network Emulation for 5 years in different roles and environments. During my time at Juniper, I was part of the team who made the virtual QFX publicly available. Due to my background, when I'm thinking about Network Simulation, I'm thinking first about networks made of Routers and Switches but this write up should be applicable beyond datacenter, switches and routers.

3 Main Use Cases for Network Emulation

I categorized the main use cases for Network Simulation into 3 categories:

- **Network Design, Validation, and Testing**: Recreate a network to validate its behavior.
- **Tools development and Validation**: Use virtual devices to validate and develop tools that are dependent on network devices.
- **Training and Demo**: Create a network for training or to demonstrate a new feature.

Network Design, Validation, and Testing

Ideally, everything going into production should be first thoroughly tested, whether it is a change in the design, a standard configuration change, or a new version of your favorite Network Operating System (NOS). Unfortunately, only a few organizations have the resources to have a lab network mirroring the production environment and it goes without mentioning the challenges inherent in managing such an environment.

The ability to emulate a production network in a virtual environment reduces the barrier to entry so that most organizations can have one virtual lab, and it also makes it possible to simulate networks at very large scale.

For a NetDevOps organization, the goal is to be able to test the network automatically for every change. At some point, it will be common to have an Emulated Network simulating the production network created on-demand by the Continuous Integration server, for every change.

Tools for development and Validation

All software that interacts with a network device–whether using the CLI or an API–should be tested not only when the software itself changes but also when the network changes (see first use case : Network Validation and Testing).

Software Testing is a topic on its own with multiple stages (unit, integration, system, etc.) Normally you will start with unit tests by leveraging simulated (mocked) interface but it's recommended to also have some end-to-end testing with something as close as possible to your final environment.

Depending on what you are developing, you might want to even test your software against multiple versions of your NOS to make sure that you are covering all cases.

CI/CD is already the norm in software development, in the application world it's common to recreate a production-like environment on-demand during the testing cycle to do end-to-end testing. With a proper Network Simulation environment it would be possible to introduce networks in the mix as well.

Training and Demonstration

Protocols and architectures in networking are evolving at a very fast pace and it's often challenging to access a a physical lab with the right devices in order to get familiar with this new technologies.

For this use case, it's very convenient to be able to use a virtual lab that will let you explore all sorts of topologies and protocols.

This is probably the use case that people are most familiar with. For some time now, CCIE or JNCIE candidates have been practicing on virtual labs built with GNS3 or EVE-Ng.

Network vendors like Cumulus Networks or Big Switch Networks are providing virtual environments available in the cloud so that everyone can get can easily get access to their technology (Cumulus In the Cloud, Bigswitch Labs).

Network Emulation is different than Network Virtualization and Network Simulation

Often people mix up **Network Emulation**, **Network Virtualization**, and **Network Simulation**. Here are our definitions:

- **Network Virtualization** consists of using a virtual device **in a production network** as a replacement for a physical device.
- **Network Emulation** consists of emulating a production device with a virtual equivalent for testing or training, derived from the same software as the production device.
- **Network Simulation** consists of simulating a production device with a completely different software. (Batfish, Forward networks …)

Network Virtualization has gained popularity in the last 5 years as it allows for better resource control, scaled out solutions and the ability to deploy as you grow that fit very well with Service Provider or Cloud use cases. The most popular Virtual Network Devices (AKA Virtual Network Function, VNF) are virtual router, firewall and load balancer. **Network Emulation** and **Network Simulation** are close since their goal is to reproduce a production device/network in a controlled environment, but there is an important difference. Network Emulation is based on the same code as the production device, while Network Simulation is based on a third party software attempting to replicate the behavior of the network device. Both have a place and both approaches have their pros and cons in a testing strategy. A mature testing environment will often leverage both.

Differences between Physical Network Device and Virtual Network Device

The main component is the Virtual Network Device (VND) that simulates the behavior and the feature of a production network device. Ideally a VND will be at feature parity with the real device. Unfortunately, this is not true for most devices out there. The gap between VND and production devices varies a lot from vendor to vendor or even between devices.

So if we don't have feature parity between a production device and its virtual image, does that mean we can't do Network Emulation? No, it's still possible and there are some benefits but we won't be able to emulate everything and we won't be able to get the most out of it. For now, we need to be aware of these caveats and their workarounds. Hopefully, as more people are starting on this journey and are defining a clear list of requirements they are expecting from their network vendors and are "**voting with their wallets**" we should see the gap shrinking.

Nowadays, all routers or switches are making the distinction between the control-plane traffic (all the network protocols :lldp, lacp bgp, etc ..) and the data-plane traffic (real traffic). In most cases, the control plane traffic is handled by the routing engine in software and the dataplane traffic is handled with the a network processor or ASIC in hardware, in order to process Gbps or Tbps of data. The ASIC market is a topic on its own with lots of players and differences but in a nutshell, most vendors out there are not building there own chipset, they are buying them off the shelf. The goal for an VND is to have full feature parity with the physical device it's emulating, which means that we would need to

emulate both the control plane and the dataplane in order to have a complete emulation. Unfortunately, most ASIC manufacturers are not providing a software emulator for their chipset that can be distributed or they are not providing a solution at all. As a result, some VND can only emulate the control plane and are providing a different solution for the dataplane. The available solutions are disparate–some vendors are able to provide a real emulator for both the control-plane and the data-plane, and some are just providing an emulation of the control plane. As a user, it's important to ask these questions because the architecture of a VND will have a direct impact on its features and ultimately on the level of trust you can place on it regarding some features that are usually executed on the dataplane. Also if the dataplane is not properly emulated using the same code than the production device, you won't be able to reproduce or identify bugs that are specific to this component. It's not necessarily a blocker but it's important to understand the underlying architecture of a VND to understand what is representative of the production environment and what is not.

Network Device Emulation Requirements

Here are 9 points of consideration when evaluating a VND for Network Emulation:

1. **Resource consumption**: Some Virtual Network Device or Virtualization solutions are optimized for production and performance instead of testing and may require a lot of resources to work properly, check the CPU usage, the number of CPU core needed and the memory.
2. **Max Number of Interfaces**: The maximum number of interfaces supported is usually lower than the real device. This factor is important because it may limit the size of the network you will be able to emulate. If your VND only supports 8 interfaces, you won't be able to create a datacenter network with more than 8 racks and only few servers per rack.
3. **List of supported features**: As discussed previously, it's very important to understand which which features are–and even more importantly which ones are not–as this will determine what can be emulated. Some vendors are not clearly publishing a list of supported/not supported features for their VND.
4. **Performance**: It's expected for an VND to be able to carry real packets in order to do end-to-end connectivity tests. The performance varies dramatically between solutions, as some VND can only carry few 100s of packets per second while others can carry 100s of Mbps of traffic.
5. **Release cycle**: The release cycle is an important point that is often forgotten. Once you have integrated Network Emulation into your process and are relying on it to validate a new configuration or a new code before deploying it in production, it will become mandatory to have access to a VND image for every software release available; major, minor and bug fixes releases. Some vendor are still releasing images only for major releases of their software or for some minor releases but not all.
6. **Ability to bootstrap out-of-the box**: Network Emulation benefits are decoupled when you are able to automate the creation and the provisioning of your virtual environment. It opens the door for more use cases, such as easily testing a new version of NOS. In order to automate the creation and the provisioning of your VND, you need to make sure it provide some API or solution to get its configuration at boot time (ZTP, DHCP, cloud-init ..)

7. **License & Support**: What conditions need to be met to access a VND image and what is the level of support provided on the VND itself. Some vendors are still not providing official support and are considering their VND as best-effort since they are not charging for it.
8. **Support for main tools and hypervisor**: Almost each hypervisor or network Emulation tool has specific requirements, so make sure to ask which one is supported out of the box.
9. **Hardware requirements**: As mentioned some VND are optimized for production and performance instead of testing, and may require some specific hardware or options. Check for Hardware Acceleration, Specific type of NIC or kernel version.

> It's important to differentiate VND designed for Network Virtualization from the one designed for Network Emulation, as the requirements are very different. VND designed for Network Virtualization are usually optimized for performance and this usually leads to bigger resource requirements

Conclusion

I hope that this blog will help increase the level of understanding and awareness on this topic and that collectively we'll be able to articulate our requirements to our vendors better. I'm looking forward to the day when we'll be able to truly emulate networks at scale without comprimises and when it will be mainstream to have changes tested automatically in a network emulation as part of the Continuous Integration pipeline.

-Damien (@damgarros)