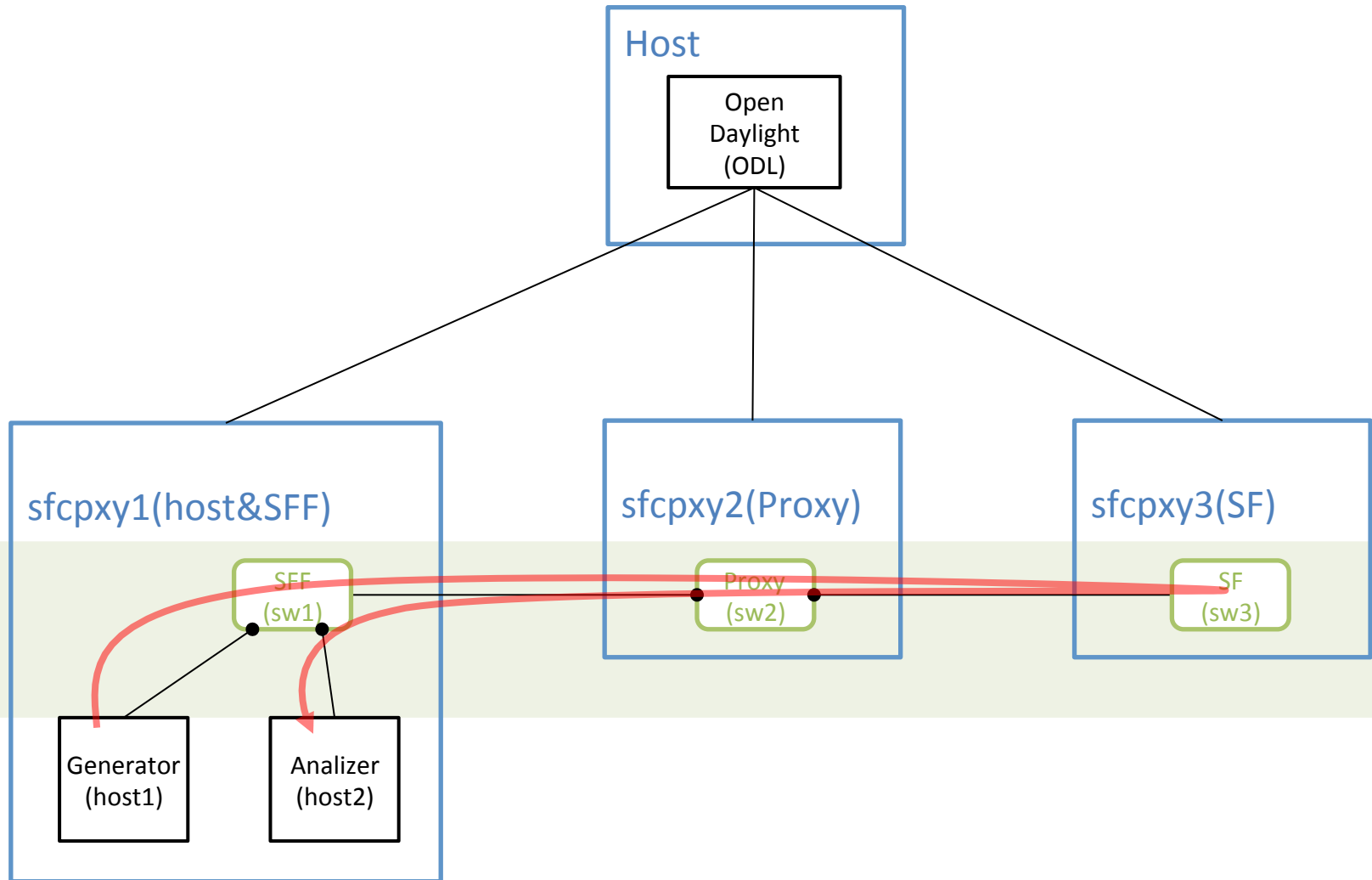
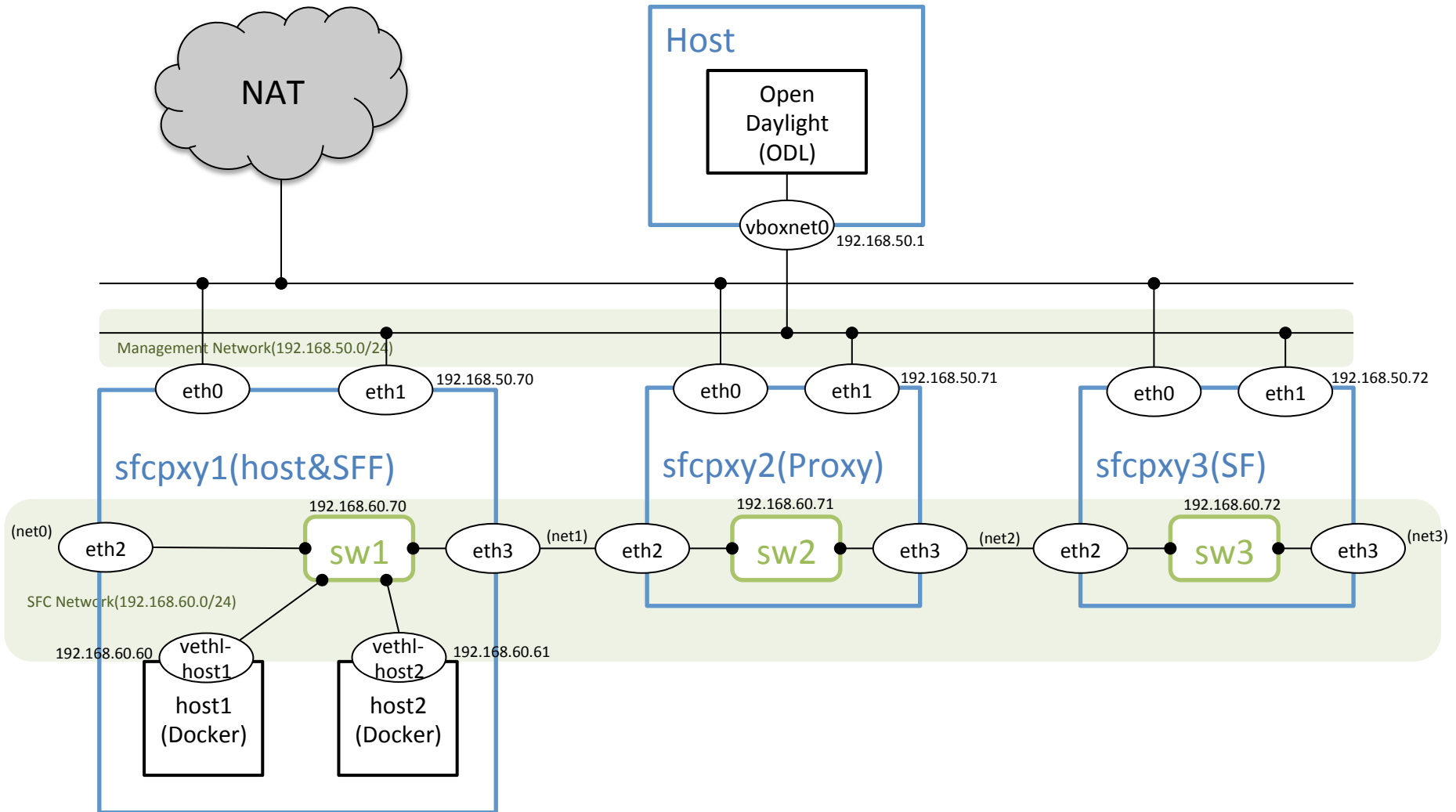


Network Topology(logical)

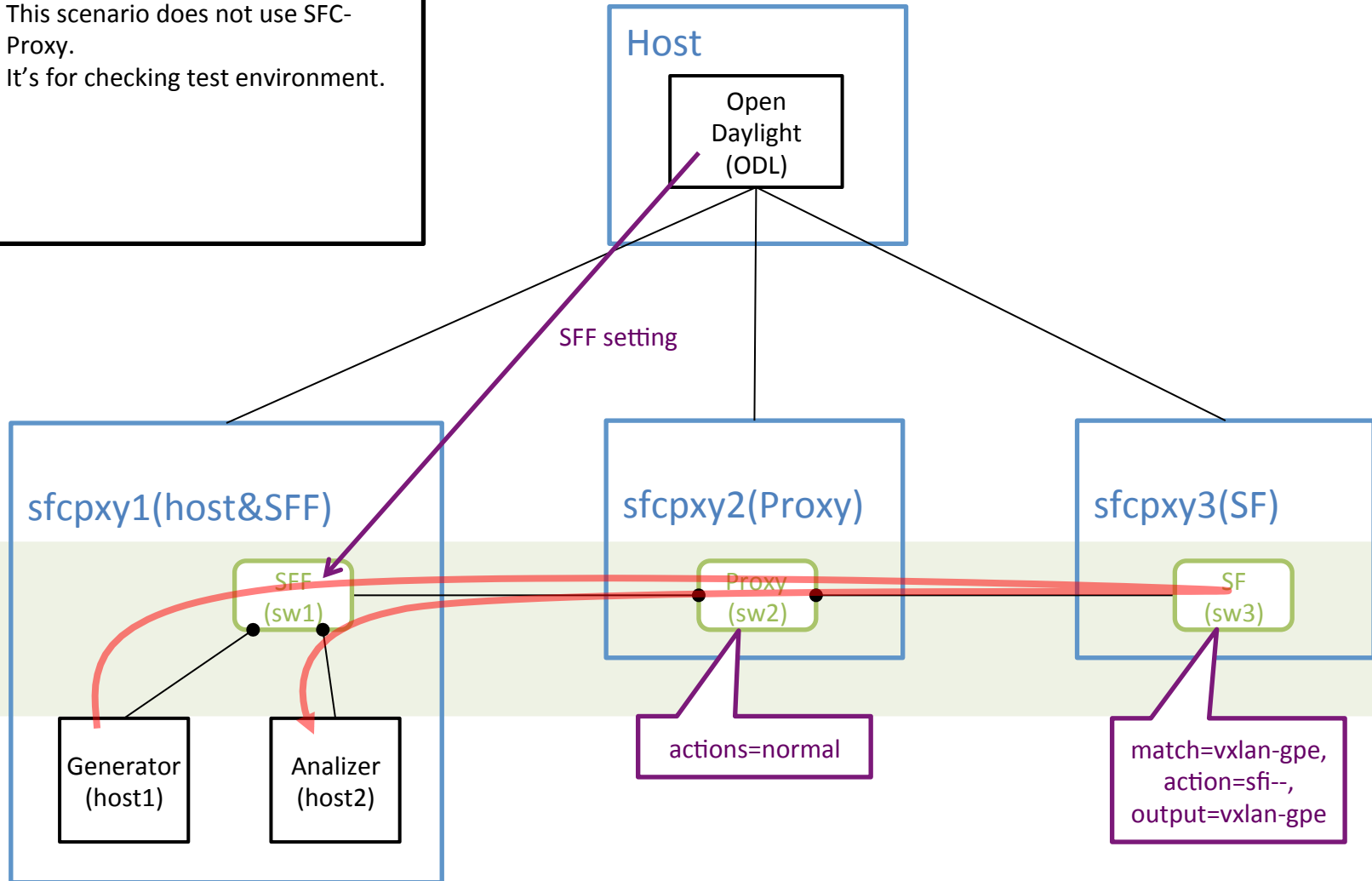


Network Topology(detailed)



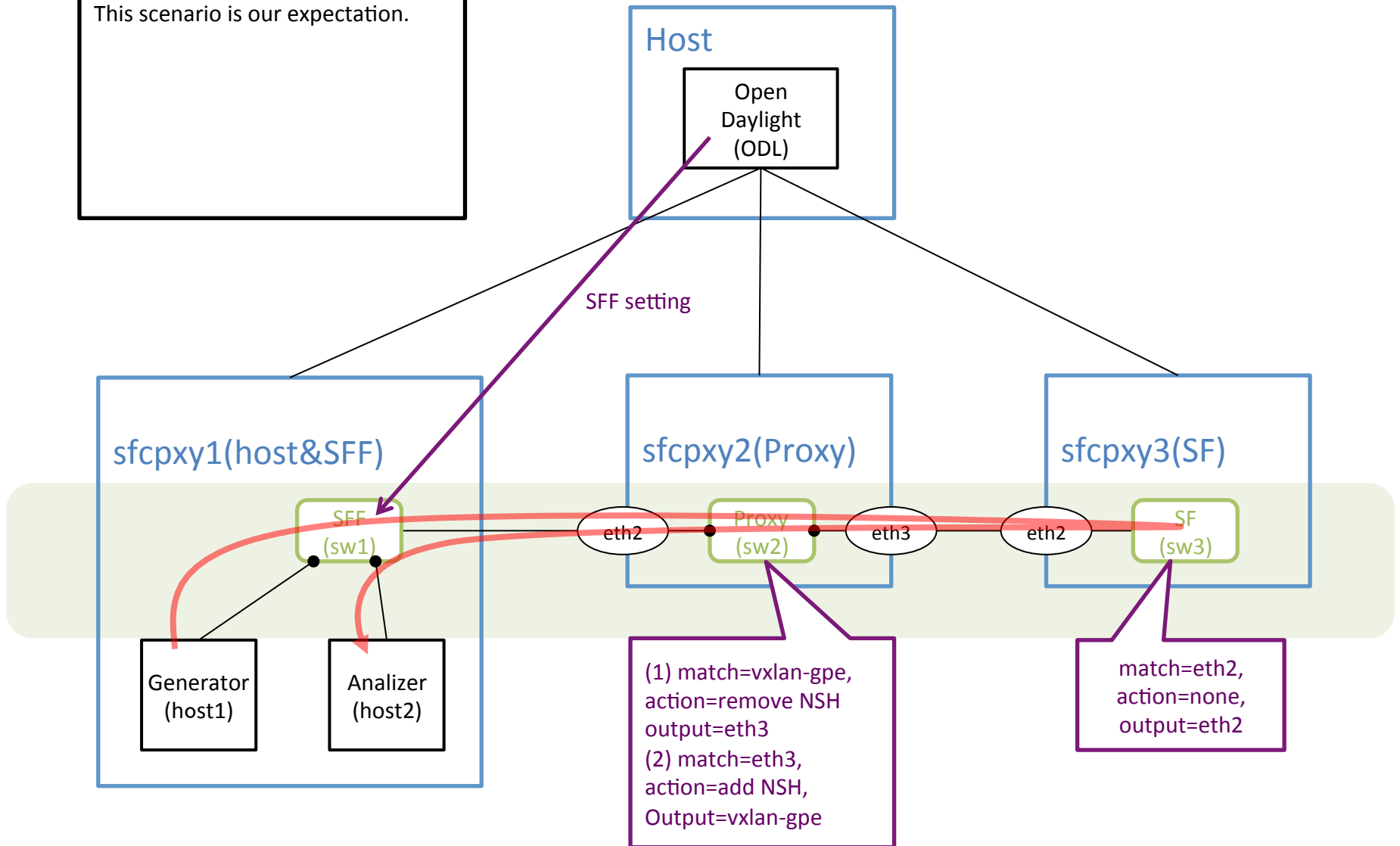
Scenario1: NSH-aware SF

This scenario does not use SFC-Proxy.
It's for checking test environment.



Scenario2: non NSH-aware SF

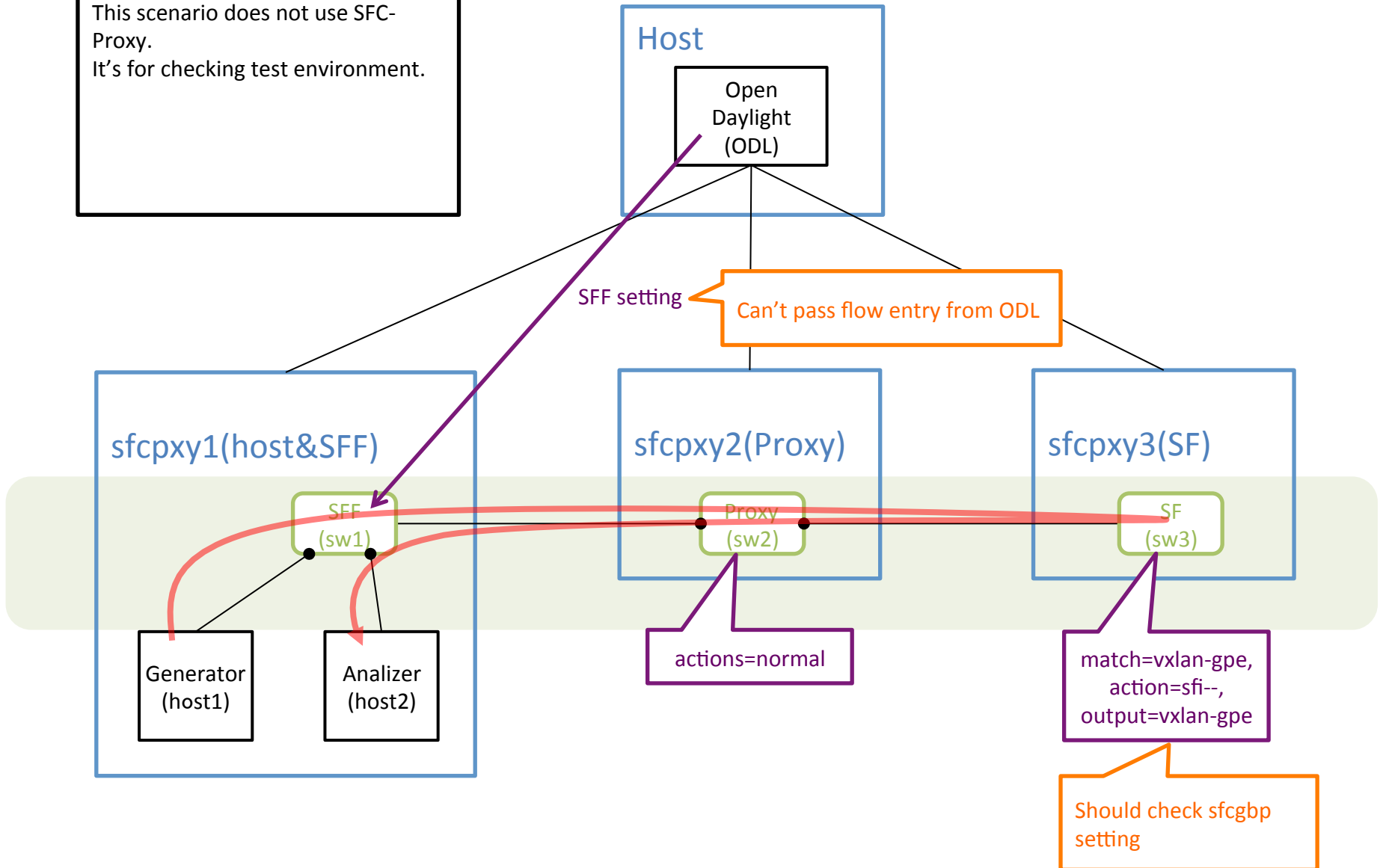
This scenario is our expectation.



Consideration:

Scenario1: NSH-aware SF

This scenario does not use SFC-Proxy.
It's for checking test environment.



Preparation (Before testing)

- Install the applications below.
 - VirtualBox
 - <https://www.virtualbox.org/wiki/Downloads>
 - Vagrant
 - <https://www.vagrantup.com/downloads.html>
- Install the OpenDaylight.
 - <https://www.opendaylight.org/downloads>
- Download VM image file.
 - https://atlas.hashicorp.com/toshirin/boxes/sfcpxy_base/versions/1.0/providers/virtualbox.box
 - Large file about 3.0GB.

Start Network & VMs(1)

- 1) Clone the “sfcpxy” git repository.
 - `git clone https://github.com/toshirin/sfcpxy.git`
- 2) In the repository, modify the “Vagrantfile”.
 - Find “###Your Directory Here###”, and replace to your VM image file path which you have downloaded before.
- 3) Read the configuration, then start VMs.
 - **`source ./env.sh`**
 - `vagrant up`
 - This may take several minutes.

Start Network & VMs(2)

4) You run the OpenDaylight.

- bin/karaf
- If you run the OpenDaylight first, you install the SFC modules below.
 - `feature:install odl-sfc-core odl-sfc-ui odl-sfc-ui odl-sfcowl2`

5) Run the setting scripts.

- `./startdemo.sh nsh-aware`
 - The argument phrase is for selection of test scenarios.
 - As a first step, select “nsh-aware”.

6) If you want to login to the VMs, use “vagrant ssh sfcpxyN”.

- N is a host number.

Operation in the emulated host.

- 1) Login to the “sfcpxy1”.
 - `vagrant ssh sfcpxy1`
- 2) Check the docker status.
 - `docker ps`
- 3) Login to the emulated host.
 - `docker attach host1`
 - The choices are “host1” and “host2”.
 - Don’t forget to **double ENTER**.
- 4) If you want to detach from the emulated host, press “Ctrl-p Ctrl-q”.
 - you can go back to the sfcpxy1’s prompt.

Test procedure

- (All settings are in the emulated host.)
- Generate NSH packet
 - `cd /sfc/sfc-py/sfc`
 - `python3.4 sff_client.py --remote-sff-ip=10.0.35.1 --sfp-id=1 --sfp-index=255 --encapsulate=vxlan-nsh-ethernet-legacy --inner-src-ip=10.1.0.10 --inner-dest-ip=10.2.0.10`
 - adjusting...
- Analyze NSH packet
 - `sudo tcpdump -i eth0`

Stop Networks & VMs

- Stop networks:
 - `./cleandemo.sh`
- Stop VMs temporarily:
 - `vagrant halt`
 - If you want to re-run, you also execute “vagrant up”.
- Remove VMs:
 - `vagrant destroy`

Some tips

- Directory share
 - Host:<vagrant execution dir> = sfcpxyN:/vagrant
= hostN:/vagrant
 - It's useful for sharing the packet capture file.

Trouble shooting

- If you don't go well by the environment, check below.
- 1. Don't you forget "source ./env.sh"?
- 2. Are your VMs running longer?
 - Longer operation may cause dirty status.
 - Clean the current VMs by "vagrant destroy; vagrant up".
- 3. Did you update the vagrant image file?
 - Vagrant may cache the old image file.
 - Delete cache file in "~/.vagrant.d/boxes"(as MacOS).