VSPERF Guides and Installation

Release draft (4e48999)

OPNFV

January 21, 2016

CHAPTER

ONE

GETTING STARTED WITH 'VSPERF'

1.1 Requirements

VSPERF requires a traffic generators to run tests, automated traffic gen support in VSPERF includes:

- IXIA traffic generator (IxNetwork hardware) and a machine that runs the IXIA client software.
- Spirent traffic generator (TestCenter hardware chassis or TestCenter virtual in a VM) and a VM to run the Spirent Virtual Deployment Service image, formerly known as "Spirent LabServer".

If you want to use another traffic generator, please select the Dummy generator option as shown in Traffic generator instructions

Supported OSes include:

- CentOS Linux release 7.1.1503 (Core) host.
- Fedora 21 and 22.
- Ubuntu 14.04

1.2 vSwitch Requirements

The vSwitch must support Open Flow 1.3 or greater. VSPERF supports both:

- OVS
- OVS with DPDK

1.3 VSPERF Installation

Follow the installation instructions to install.

1.4 Traffic Generator Setup

Follow the Traffic generator instructions to install and configure a suitable traffic generator.

1.5 Cloning and building src dependencies

In order to run VSPERF, you will need to download DPDK and OVS. You can do this manually and build them in a preferred location, OR you could use vswitchperf/src. The vswitchperf/src directory contains makefiles that will allow you to clone and build the libraries that VSPERF depends on, such as DPDK and OVS. To clone and build simply:

```
$ cd src
$ make
```

VSPERF can be used with stock OVS (without DPDK support). In this case you have to specify path to the kernel sources when building OVS in src by specifying WITH_LINUX parameter:

```
$ cd src
$ make WITH_LINUX=/lib/modules/`uname -r`/build
```

To build DPDK and OVS in the src directory for PVP and PVVP testing with vhost_user as the guest access method, use:

```
$ make VHOST_USER=y
```

To build all options in src:

- Vanilla OVS
- OVS with vhost_user as the guest access method (with DPDK support)
- OVS with vhost_cuse s the guest access method (with DPDK support)

simply call 'make' in the src directory :

```
$ make
```

The vhost_user build will reside in src/ovs/ The vhost_cuse build will reside in vswitchperf/src_cuse The Vanilla OVS build will reside in vswitchperf/src_vanilla

To delete a src subdirectory and its contents to allow you to re-clone simply use:

```
$ make clobber
```

1.6 Configure the `./conf/10_custom.conf` file

The `10_custom.conf` file is the configuration file that overrides default configurations in all the other configuration files in `./conf` The supplied `10_custom.conf` file **MUST** be modified, as it contains configuration items for which there are no reasonable default values.

The configuration items that can be added is not limited to the initial contents. Any configuration item mentioned in any .conf file in `./conf` directory can be added and that item will be overridden by the custom configuration value.

1.7 Using a custom settings file

If your `10_custom.conf` doesn't reside in the `./conf` directory of if you want to use an alternative configuration file, the file can be passed to `vsperf` via the `--conf-file` argument.

```
$ ./vsperf --conf-file <path_to_custom_conf> ...
```

Note that configuration passed in via the environment (`--load-env`) or via another command line argument will override both the default and your custom configuration files. This "priority hierarchy" can be described like so (1 = max priority):

1. Command line arguments
2. Environment variables
3. Configuration file(s)

1.8 Executing tests

Before running any tests make sure you have root permissions by adding the following line to /etc/sudoers:

```
username ALL=(ALL)       NOPASSWD: ALL
```

username in the example above should be replaced with a real username.

To list the available tests:

```
$ ./vsperf --list
```

To run a single test:

```
$ ./vsperf $TESTNAME
```

Where $TESTNAME is the name of the vsperf test you would like to run.

To run a group of tests, for example all tests with a name containing 'RFC2544':

```
$ ./vsperf --conf-file=<path_to_custom_conf>/10_custom.conf --tests="RFC2544"
```

To run all tests:

```
$ ./vsperf --conf-file=<path_to_custom_conf>/10_custom.conf
```

Some tests allow for configurable parameters, including test duration (in seconds) as well as packet sizes (in bytes).

```
$ ./vsperf --conf-file user_settings.py
    --tests RFC2544Tput
    --test-param "duration=10;pkt_sizes=128"
```

For all available options, check out the help dialog:

```
$ ./vsperf --help
```

1.9 Executing Vanilla OVS tests

If you have compiled all the variants of OVS in ''src/" please skip step 1.

1. Recompile src for Vanilla OVS testing

```
$ cd src
$ make cleanse
$ make WITH_LINUX=/lib/modules/`uname -r`/build
```

2. Update your "10_custom.conf" file to use the appropriate variables for Vanilla OVS:

```
VSWITCH = 'OvsVanilla'
VSWITCH_VANILLA_PHY_PORT_NAMES = ['$PORT1', '$PORT1']
```

Where $PORT1 and $PORT2 are the Linux interfaces you'd like to bind to the vswitch.

3. Run test:

```
$ ./vsperf --conf-file=<path_to_custom_conf>
```

Please note if you don't want to configure Vanilla OVS through the configuration file, you can pass it as a CLI argument; BUT you must set the ports.

```
$ ./vsperf --vswitch OvsVanilla
```

1.10 Executing PVP and PVVP tests

To run tests using vhost-user as guest access method:

1. Set VHOST_METHOD and VNF of your settings file to:

```
VHOST_METHOD='user'
VNF = 'QemuDpdkVhost'
```

2. Recompile src for VHOST USER testing

```
$ cd src
$ make cleanse
$ make VHOST_USER=y
```

3. Run test:

```
$ ./vsperf --conf-file=<path_to_custom_conf>/10_custom.conf
```

To run tests using vhost-cuse as guest access method:

1. Set VHOST_METHOD and VNF of your settings file to:

```
VHOST_METHOD='cuse'
VNF = 'QemuDpdkVhostCuse'
```

2. Recompile src for VHOST USER testing

```
$ cd src
$ make cleanse
$ make VHOST_USER=n
```

3. Run test:

```
$ ./vsperf --conf-file=<path_to_custom_conf>/10_custom.conf
```

1.11 Executing PVP tests using Vanilla OVS

To run tests using Vanilla OVS:

1. Set the following variables:

```
VSWITCH = 'OvsVanilla'
VNF = 'QemuVirtioNet'

VANILLA_TGEN_PORT1_IP = n.n.n.n
VANILLA_TGEN_PORT1_MAC = nn:nn:nn:nn:nn:nn

VANILLA_TGEN_PORT2_IP = n.n.n.n
VANILLA_TGEN_PORT2_MAC = nn:nn:nn:nn:nn:nn

VANILLA_BRIDGE_IP = n.n.n.n

or use --test-param

./vsperf --conf-file=<path_to_custom_conf>/10_custom.conf
         --test-param "vanilla_tgen_tx_ip=n.n.n.n;
                       vanilla_tgen_tx_mac=nn:nn:nn:nn:nn:nn"
```

2. Recompile src for Vanilla OVS testing

```
$ cd src
$ make cleanse
$ make WITH_LINUX=/lib/modules/`uname -r`/build
```

3. Run test:

```
$ ./vsperf --conf-file<path_to_custom_conf>/10_custom.conf
```

1.12 Selection of loopback application for PVP and PVVP tests

To select loopback application, which will perform traffic forwarding inside VM, following configuration parameter should be configured:

```
GUEST_LOOPBACK = ['testpmd', 'testpmd']
```

or use –test-param

```
$ ./vsperf --conf-file=<path_to_custom_conf>/10_custom.conf
      --test-param "guest_loopback=testpmd"
```

Supported loopback applications are:

```
'testpmd'       - testpmd from dpdk will be built and used
'l2fwd'         - l2fwd module provided by Huawei will be built and used
'linux_bridge'  - linux bridge will be configured
'buildin'       - nothing will be configured by vsperf; VM image must
                  ensure traffic forwarding between its interfaces
```

Guest loopback application must be configured, otherwise traffic will not be forwarded by VM and testcases with PVP and PVVP deployments will fail. Guest loopback application is set to ‘testpmd’ by default.

1.13 Code change verification by pylint

Every developer participating in VSPERF project should run pylint before his python code is submitted for review. Project specific configuration for pylint is available at ‘pylint.rc’.

Example of manual pylint invocation:

```
$ pylint --rcfile ./pylintrc ./vsperf
```

1.14 GOTCHAs:

1.14.1 OVS with DPDK and QEMU

If you encounter the following error: "before (last 100 chars): '-path=/dev/hugepages,share=on: unable to map backing store for hugepages: Cannot allocate memoryrnrn" with the PVP or PVVP deployment scenario, check the amount of hugepages on your system:

```
$ cat /proc/meminfo | grep HugePages
```

By default the vswitchd is launched with 1Gb of memory, to change this, modify –socket-mem parameter in conf/02_vswitch.conf to allocate an appropriate amount of memory:

```
VSWITCHD_DPDK_ARGS = ['-c', '0x4', '-n', '4', '--socket-mem 1024,0']
```

CHAPTER

TWO

INSTALLING VSWITCHPERF

The test suite requires Python 3.3 and relies on a number of other packages. These need to be installed for the test suite to function.

Installation of required packages, preparation of Python 3 virtual environment and compilation of OVS, DPDK and QEMU is performed by script **systems/build_base_machine.sh**. It should be executed under user account, which will be used for vsperf execution.

Please Note: Password-less sudo access must be configured for given user account before script is executed.

Execution of installation script:

```
$ cd systems
$ ./build_base_machine.sh
```

Please note: you don't need to go into any of the systems subdirectories, simply run the top level **build_base_machine.sh**, your OS will be detected automatically.

Currently supported operating systems are:

- CentOS 7
- Fedora 20
- Fedora 21
- Fedora 22
- Ubuntu 14.04

Script **build_base_machine.sh** will install all the vsperf dependencies in terms of system packages, Python 3.x and required Python modules. In case of CentOS 7 it will install Python 3.3 from an additional repository provided by Software Collections (a link). Installation script will also use virtualenv to create a vsperf virtual environment, which is isolated from the default Python environment. This environment will reside in a directory called **vsperfenv** in $HOME.

You will need to activate the virtual environment every time you start a new shell session. Its activation is specific to your OS:

CentOS 7:

```
$ scl enable python33 bash
$ cd $HOME/vsperfenv
$ source bin/activate
```

Fedora and Ubuntu:

```
$ cd $HOME/vsperfenv
$ source bin/activate
```

2.1 Working Behind a Proxy

If you're behind a proxy, you'll likely want to configure this before running any of the above. For example:

```
export http_proxy=proxy.mycompany.com:123
export https_proxy=proxy.mycompany.com:123
```

CHAPTER

THREE

'VSPERF' TRAFFIC GEN GUIDE

3.1 Overview

VSPERF supports the following traffic generators:

- Dummy (DEFAULT): Allows you to use your own external traffic generator.
- IXIA (IxNet and IxOS)
- Spirent TestCenter

To see the list of traffic gens from the cli:

```
$ ./vsperf --list-trafficgens
```

This guide provides the details of how to install and configure the various traffic generators.

3.2 Background Information

The traffic default configuration can be found in tools/pkt_gen/trafficgen/trafficgenhelper.py, and is configured as follows:

```
TRAFFIC_DEFAULTS = {
    'l2': {
        'framesize': 64,
        'srcmac': '00:00:00:00:00:00',
        'dstmac': '00:00:00:00:00:00',
        'srcport': 3000,
        'dstport': 3001,
    },
    'l3': {
        'proto': 'tcp',
        'srcip': '1.1.1.1',
        'dstip': '90.90.90.90',
    },
    'vlan': {
        'enabled': False,
        'id': 0,
        'priority': 0,
        'cfi': 0,
    },
}
```

The framesize paramter can be overridden from the configuration files by adding the following to your custom configuration file `10_custom.conf`:

```
TRAFFICGEN_PKT_SIZES = (64, 128,)
```

OR from the commandline:

```
$ ./vsperf --test-param "pkt_sizes=x,y" $TESTNAME
```

You can also modify the traffic transmission duration and the number of trials run by the traffic generator by extending the example commandline above to:

```
$ ./vsperf --test-param "pkt_sizes=x,y;duration=10;rfc2455_trials=3" $TESTNAME
```

3.3 Dummy Setup

To select the Dummy generator please add the following to your custom configuration file `10_custom.conf`.

```
TRAFFICGEN = 'Dummy'
```

OR run `vsperf` with the `--trafficgen` argument

```
$ ./vsperf --trafficgen Dummy $TESTNAME
```

Where $TESTNAME is the name of the vsperf test you would like to run. This will setup the vSwitch and the VNF (if one is part of your test) print the traffic configuration and prompt you to transmit traffic when the setup is complete.

```
Please send 'continuous' traffic with the following stream config:
30mS, 90mpps, multistream False
and the following flow config:
{
    "flow_type": "port",
    "l3": {
        "srcip": "1.1.1.1",
        "proto": "tcp",
        "dstip": "90.90.90.90"
    },
    "traffic_type": "continuous",
    "multistream": 0,
    "bidir": "True",
    "vlan": {
        "cfi": 0,
        "priority": 0,
        "id": 0,
        "enabled": false
    },
    "frame_rate": 90,
    "l2": {
        "dstport": 3001,
        "srcport": 3000,
        "dstmac": "00:00:00:00:00:00",
        "srcmac": "00:00:00:00:00:00",
        "framesize": 64
    }
}
What was the result for 'frames tx'?
```

When your traffic gen has completed traffic transmission and provided the results please input these at the vsperf prompt. vsperf will try to verify the input:

```
Is '$input_value' correct?
```

Please answer with y OR n.

VPSERF will ask you for:

- Result for 'frames tx'
- Result for 'frames rx'
- Result for 'min latency'
- Result for 'max latency'
- Result for 'avg latency'

Finally vsperf will print out the results for your test and generate the appropriate logs and csv files.

3.4 IXIA Setup

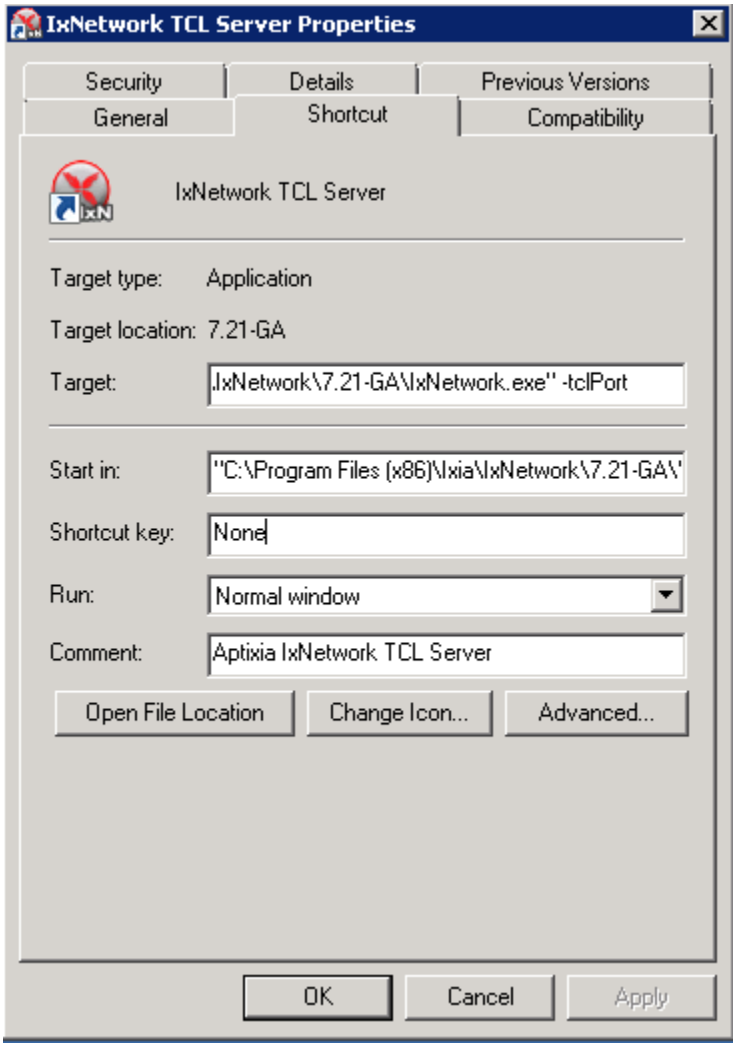
3.4.1 On the CentOS 7 system

You need to install IxNetworkTclClient$(VER_NUM)Linux.bin.tgz.

3.4.2 On the IXIA client software system

Find the IxNetwork TCL server app (start -> All Programs -> IXIA -> IxNetwork -> IxNetwork_$(VER_NUM) -> IxNetwork TCL Server)

Right click on IxNetwork TCL Server, select properties - Under shortcut tab in the Target dialogue box make sure there is the argument "-tclport xxxx" where xxxx is your port number (take note of this port number you will need it for the 10_custom.conf file).



Hit Ok and start the TCL server application

3.5 Spirent Setup

Spirent installation files and instructions are available on the Spirent support website at:

http://support.spirent.com

Select a version of Spirent TestCenter software to utilize. This example will use Spirent TestCenter v4.57 as an example. Substitute the appropriate version in place of 'v4.57' in the examples, below.

3.5.1 On the CentOS 7 System

Download and install the following:

Spirent TestCenter Application, v4.57 for 64-bit Linux Client

3.5.2 Spirent Virtual Deployment Service (VDS)

Spirent VDS is required for both TestCenter hardware and virtual chassis in the vsperf environment. For installation, select the version that matches the Spirent TestCenter Application version. For v4.57, the matching VDS version is 1.0.55. Download either the ova (VMware) or qcow2 (QEMU) image and create a VM with it. Initialize the VM according to Spirent installation instructions.

3.5.3 Using Spirent TestCenter Virtual (STCv)

STCv is available in both ova (VMware) and qcow2 (QEMU) formats. For VMware, download:

Spirent TestCenter Virtual Machine for VMware, v4.57 for Hypervisor - VMware ESX.ESXi

Virtual test port performance is affected by the hypervisor configuration. For best practice results in deploying STCv, the following is suggested:

- Create a single VM with two test ports rather than two VMs with one port each
- Set STCv in DPDK mode
- Give STCv 2*n + 1 cores, where n = the number of ports. For vsperf, cores = 5.
- Turning off hyperthreading and pinning these cores will improve performance
- Give STCv 2 GB of RAM

To get the highest performance and accuracy, Spirent TestCenter hardware is recommended. vsperf can run with either stype test ports.