**Nuage VNS Underlay Health Checker**

SDWAN CPE Pre-deployment underlay connectivity test utility

Document Management

**Document History**

|  |  |  |  |
| --- | --- | --- | --- |
| Release | Date | Content | Author |
| 1.0 | 30/12/2018 | Deployment guidelines and common usage of the tool for underlay characterisation | Yohan Gunasekara |
|  |  |  |  |

About this document

This document provides deployment guidelines for Nuage VNS (SD-WAN) underlay test tool and the way to use to measuring the underlay characteristics.

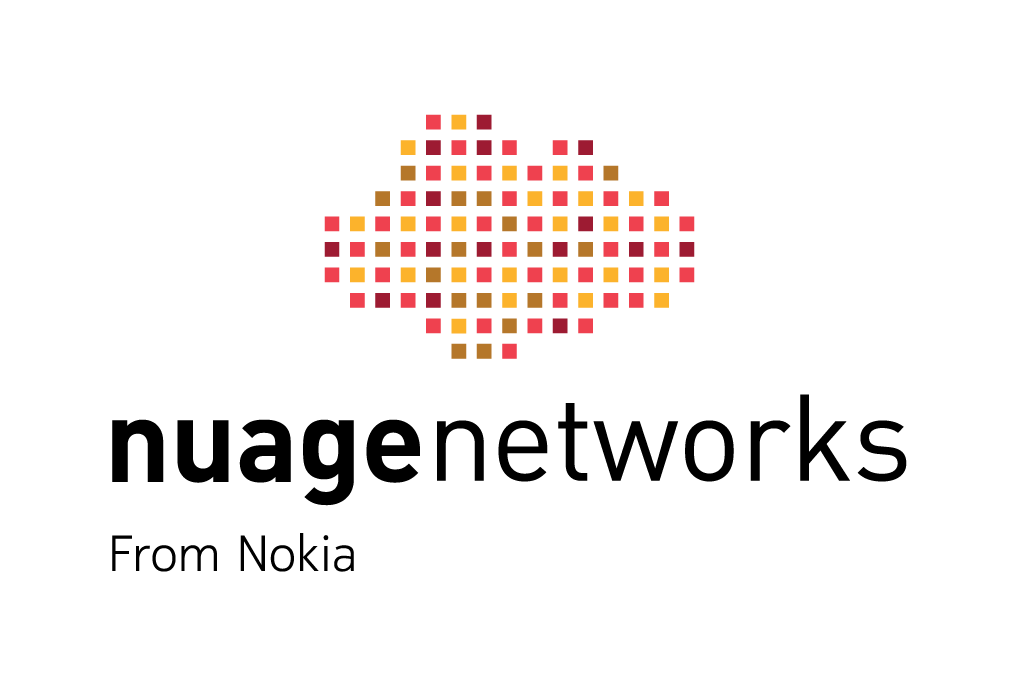


Table of Contents

1 Introduction 3

2 Deployment Guidelines 3

3 Installation and configuration – Server software 4

3.1 Server instance installation 4

3.2 Server instance configurations 11

3.3 Start, stop and upgrades 12

4 Client software installation 13

5 VNS Client Test Tool – Operational Usage 15

5.1 Connectivity testing based on port numbers and round-trip time 15

5.2 Network MTU Self Learn 17

5.3 Long Lived Connections 17

5.4 Local PC IP Addresses and events 18

5.5 MTU Settings 19

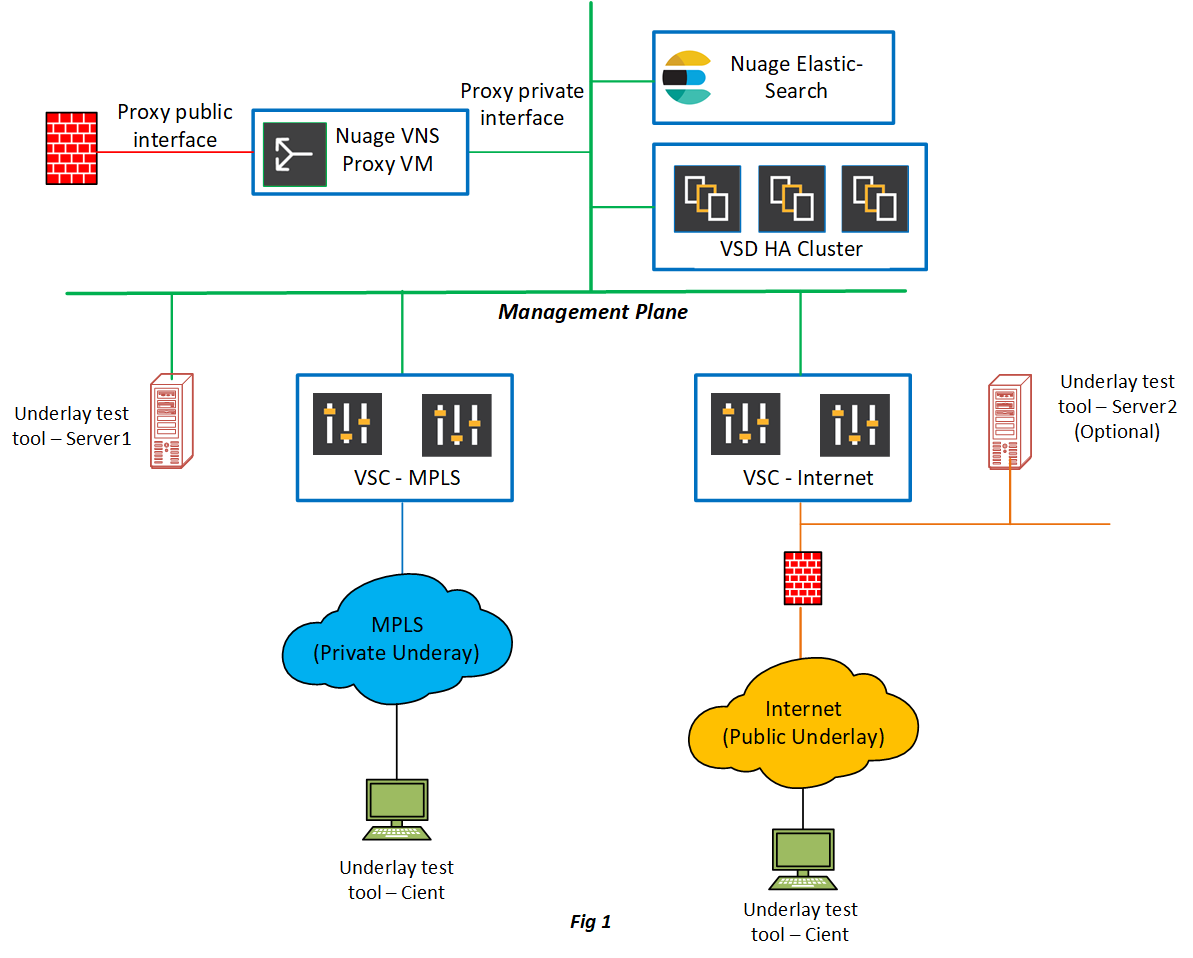
5.6 Bandwidth Monitoring 19

# 1 Introduction

SDWAN CPE requires good connectivity to its controllers and head end for the optimal operations. This communication channel depends on the WAN underlay environment. Different underlays typically have various performance and behaviour matrixes. Before bootstrapping an NSG, it is essential to make sure the selected underlay perform up to a certain standard where SDWAN topology can deploy. Nuage VNS underlay test utility is capable of running necessary tests to validate the performance/behaviour of an underlay and highlight any issues.

## 2 Deployment Guidelines

Sample deployment diagram



Common practices:

* There are two elements to deploy such as underlay test tool server portion and the client software.
* Client software deployment is very straight forward and it can install on installers branch end PC to test the underlay.
* The server software can install on KVM or ESXi hypervisor as a standard virtual machine.
* The installation position of the server software is very important. The client talks to the server and report the health of *n* number of ports to the installer. This client-server communication simulates the NSG-VSC/VNS\_Proxy communication and provide a sample output of the generic underlay behavior.
* Minimum single test server requires and maximum *n* number of server instances can consider to deploy depends on the end customer topology.

***Figure 1 – A sample deployment***

Underlay test tool server 1:

* Strategic installation location is in the management plane. In real world use case, NSG to vns\_proxy communication requires *X* number of ports to be opened via the perimeter firewall. This proxy either could be a HAProxy or a commercial appliance such as F5. Most of the time the proxy is behind a firewall. Usually proxy operates two key interfaces such as public facing interface and private circuit interface. This private interface network usually shares the same as Nuage management plane or at least it could be inter-routable in between the Nuage management plane. When the test tool server locates in the management plane then it is possible for client to check the communication health of the respective vns-proxy related ports.
* It is possible a service provider may have multiple underlays such as internet and private MPLS. A pair of VSC takes care of a particular underlay and the joined NSGs to that underlay. It is important to check the health status of NSG to VSC communication.
* The underlay test tool server 1 can use to test the MPLS underlay circuit health towards MPLS-VSCs as most of the cases MPLS underlay and the management plane is inter routable.

Underlay test tool server 2:

* Internet underlay may treat with additional perimeter firewall. If necessary a second server instance can deploy in the internet underlay as same as the exact internet VSC data plane.

## 3 Installation and configuration – Server software

**Package Contents**

Two main packages are available as below:

* VNS test tool server installation files
* VNS test tool client installation files

### 3.1 Server instance installation

#### 3.1.1 Platform requirements

Supported hypervisors: KVM and ESXi

Resource specifications:

Memory: 4GB

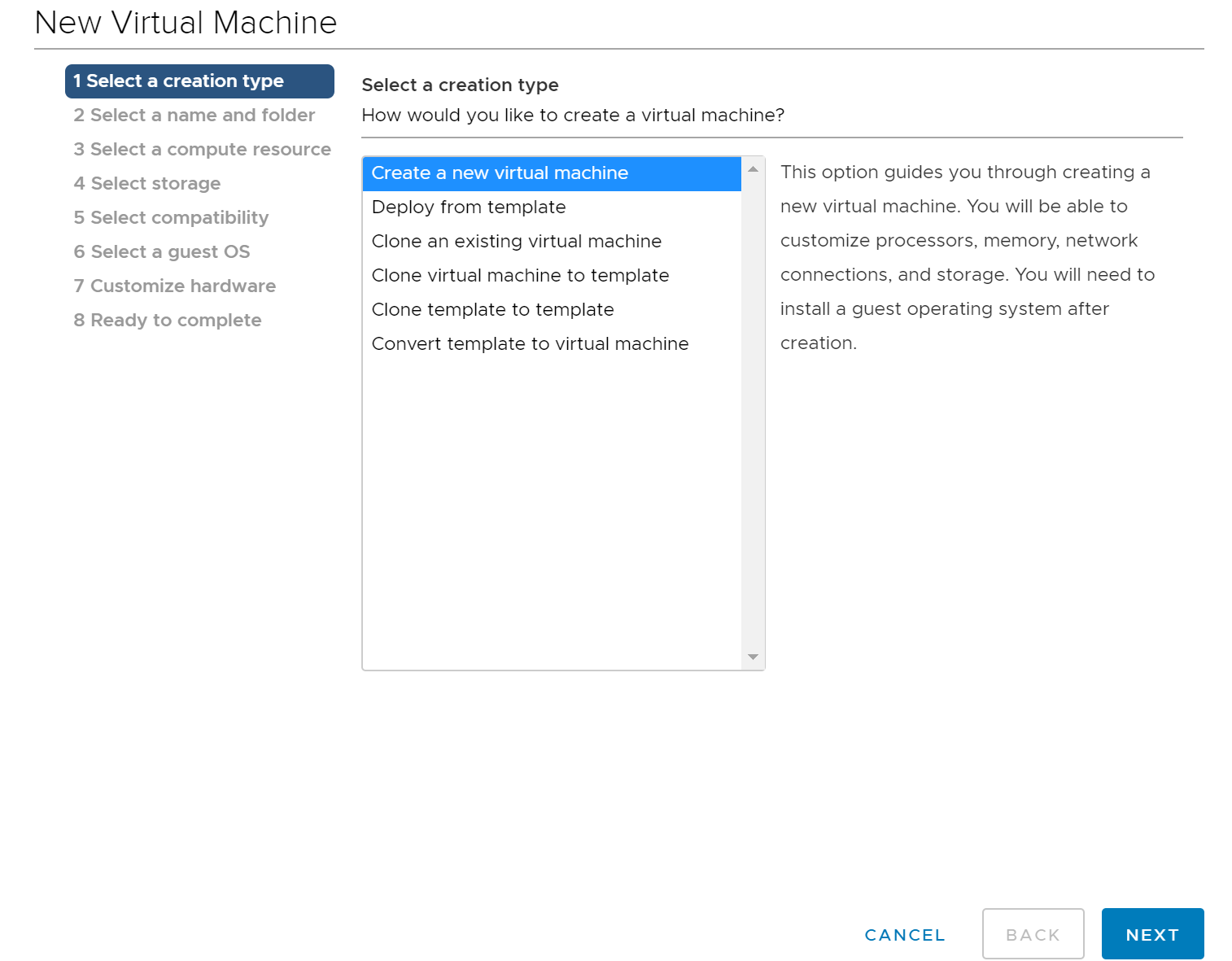
vCPU: 1

Disk size: 100GB

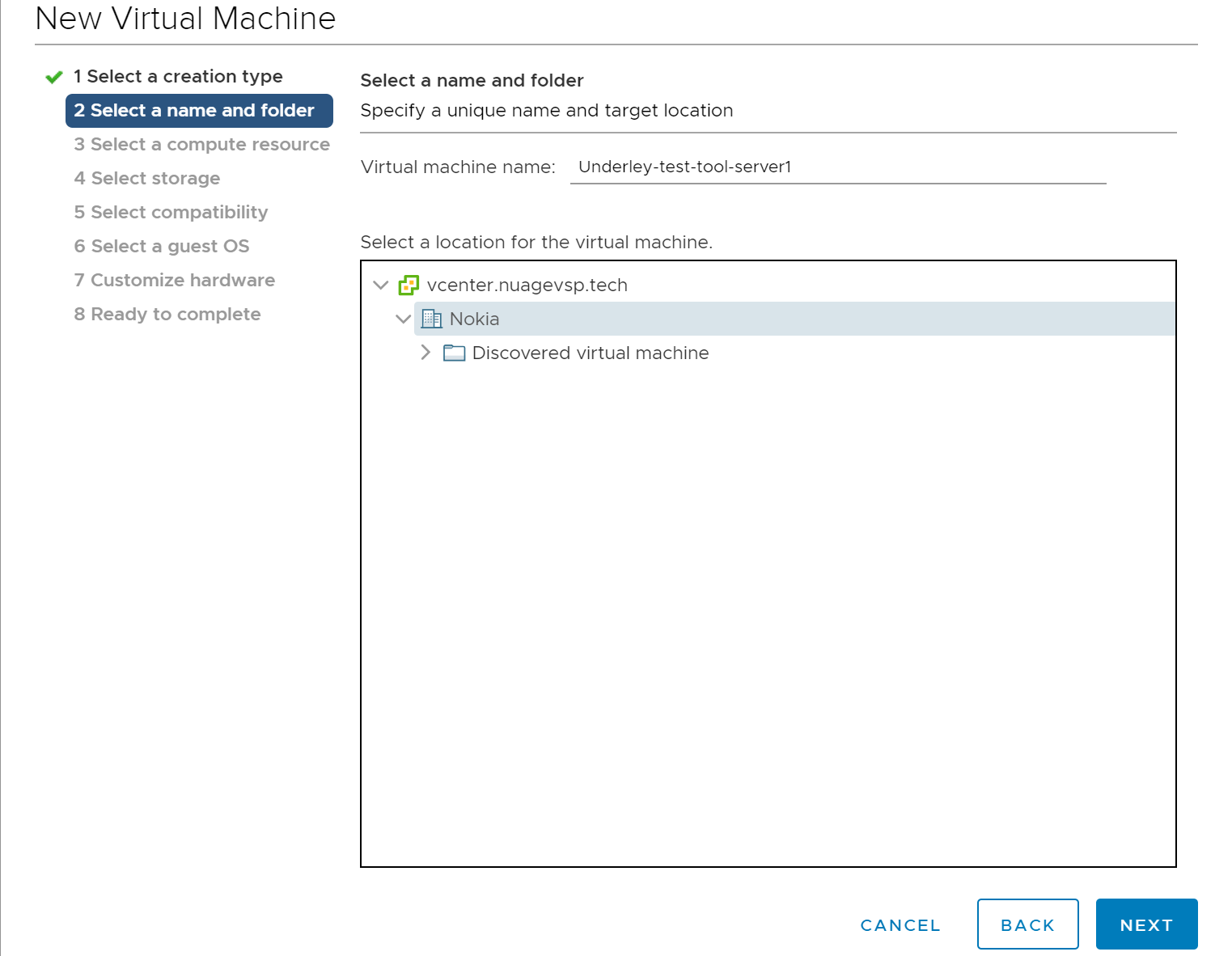
#### 3.1.2 Installation steps – VMWare ESXi

*\*Upload the vns-test-tool-server vmdk file to the ESXi datastore before start deploying the server VM*

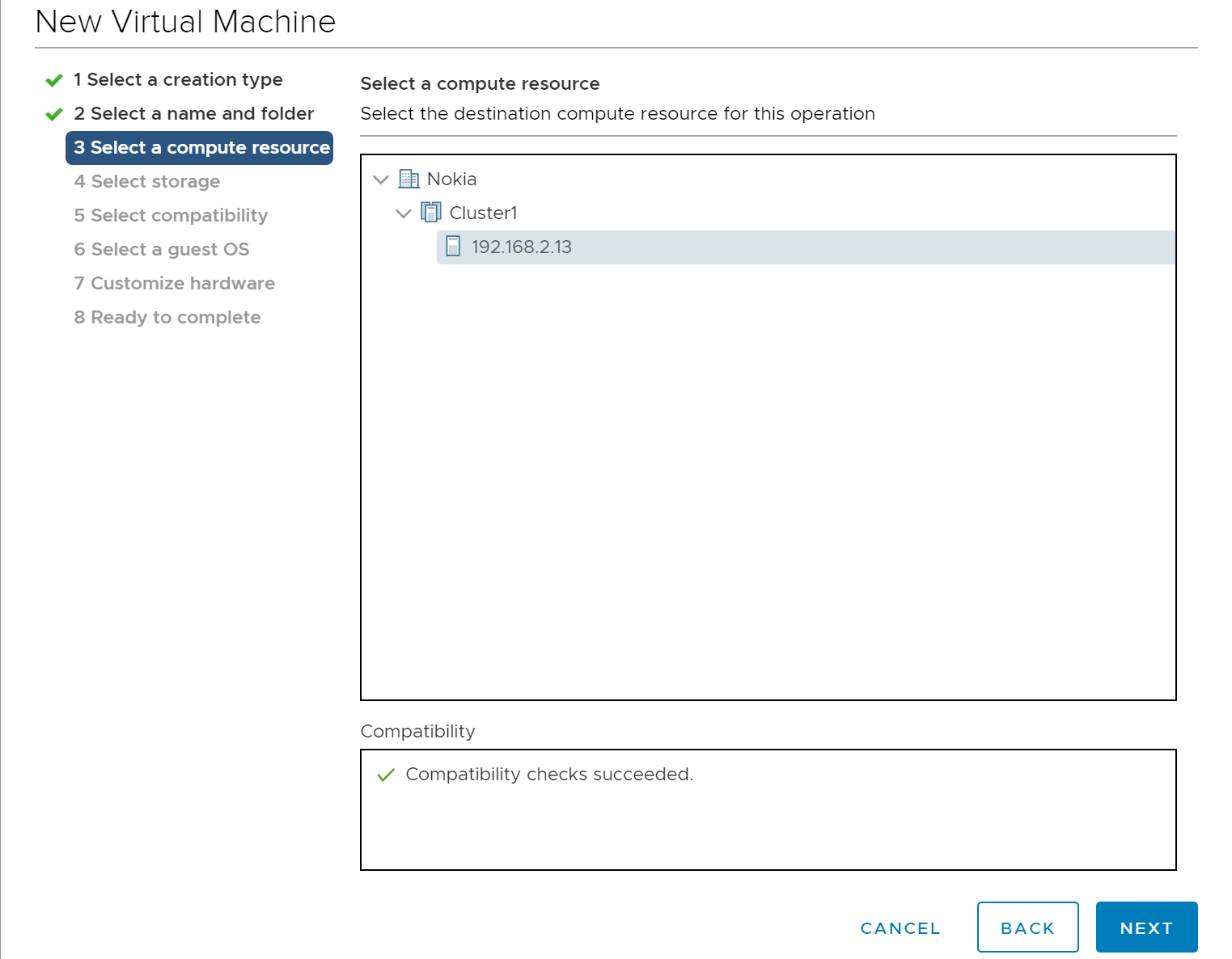
***Step 01: Create a new virtual machine***



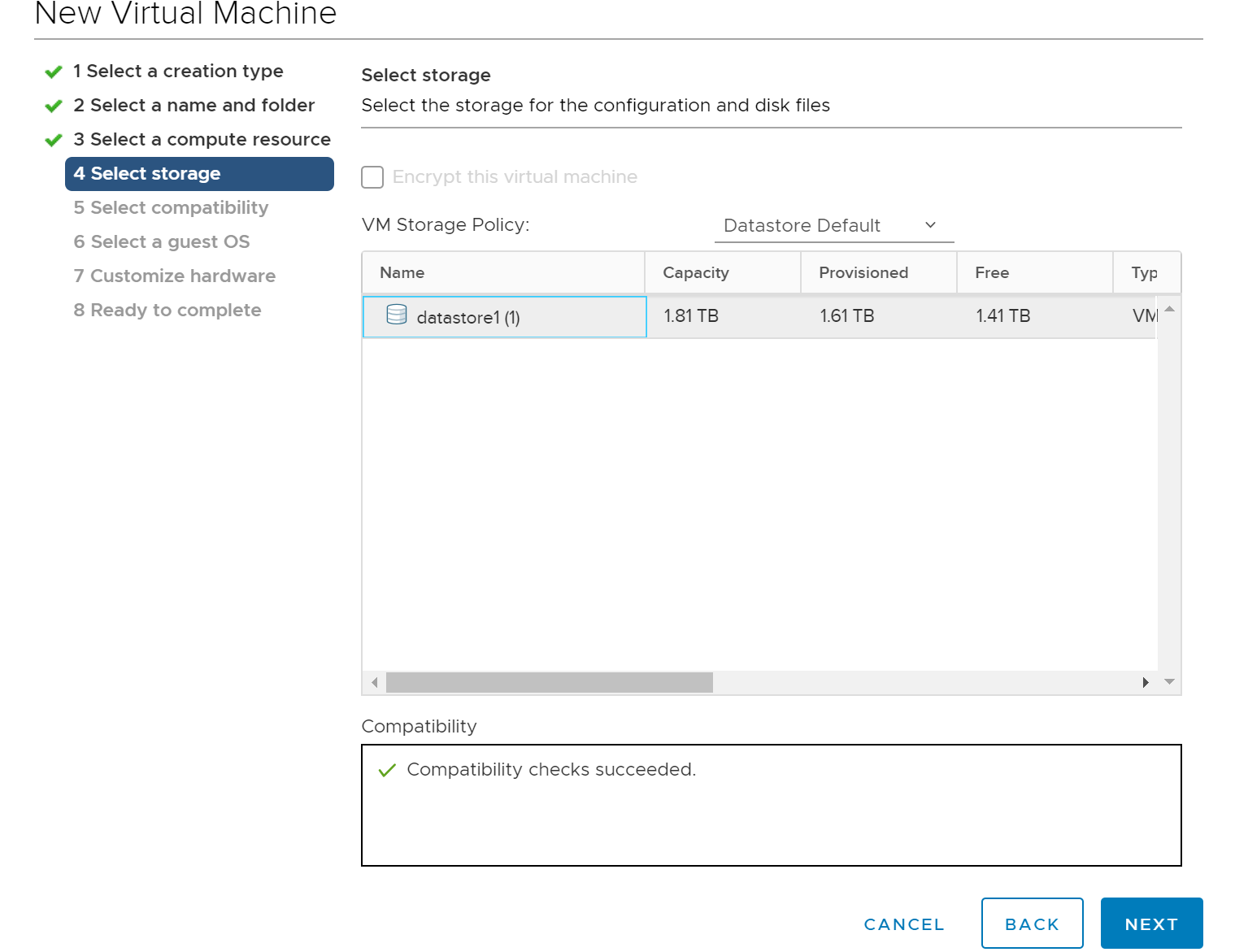
***Step 02: Select a name and folder***



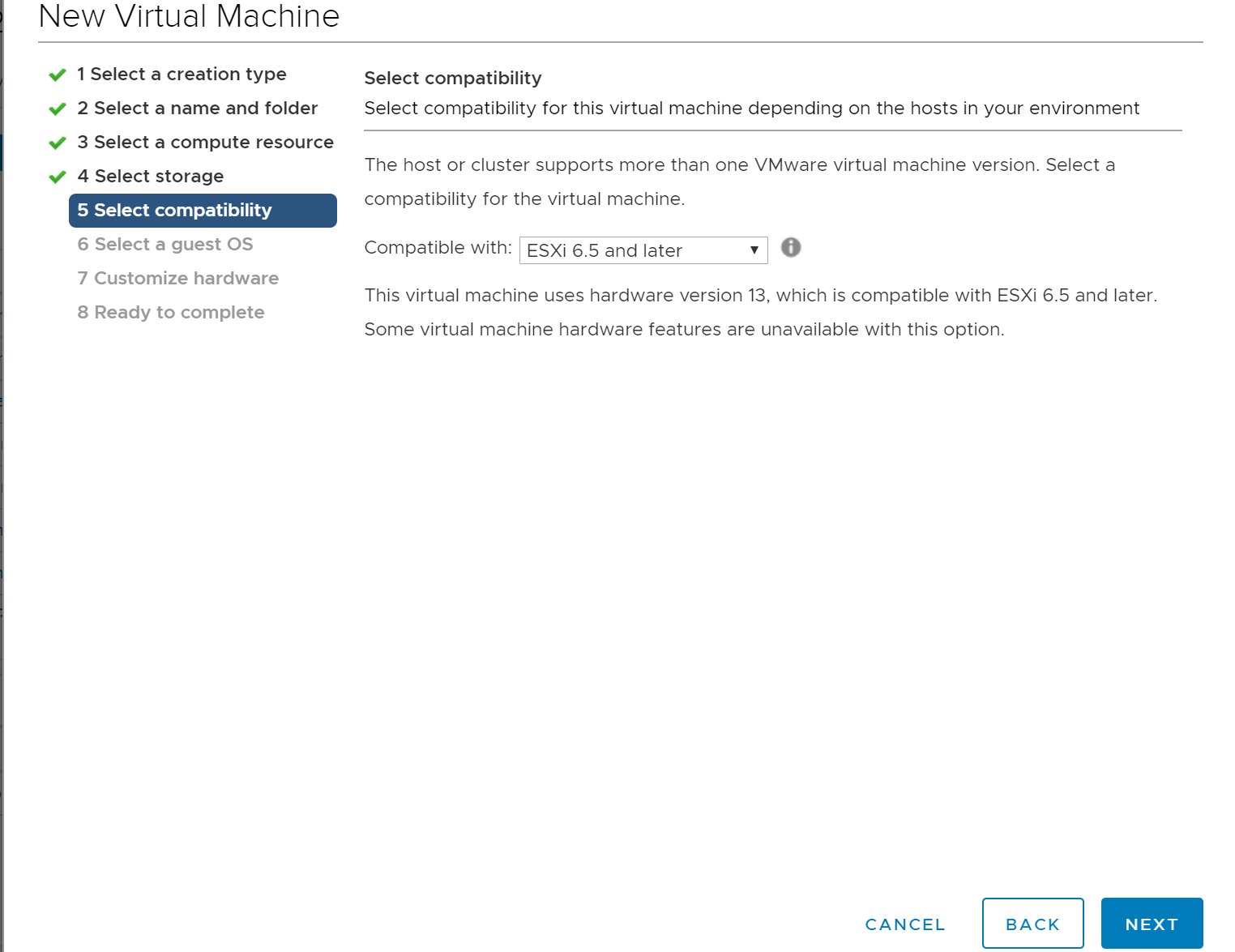
***Step 03: Select a compute host***



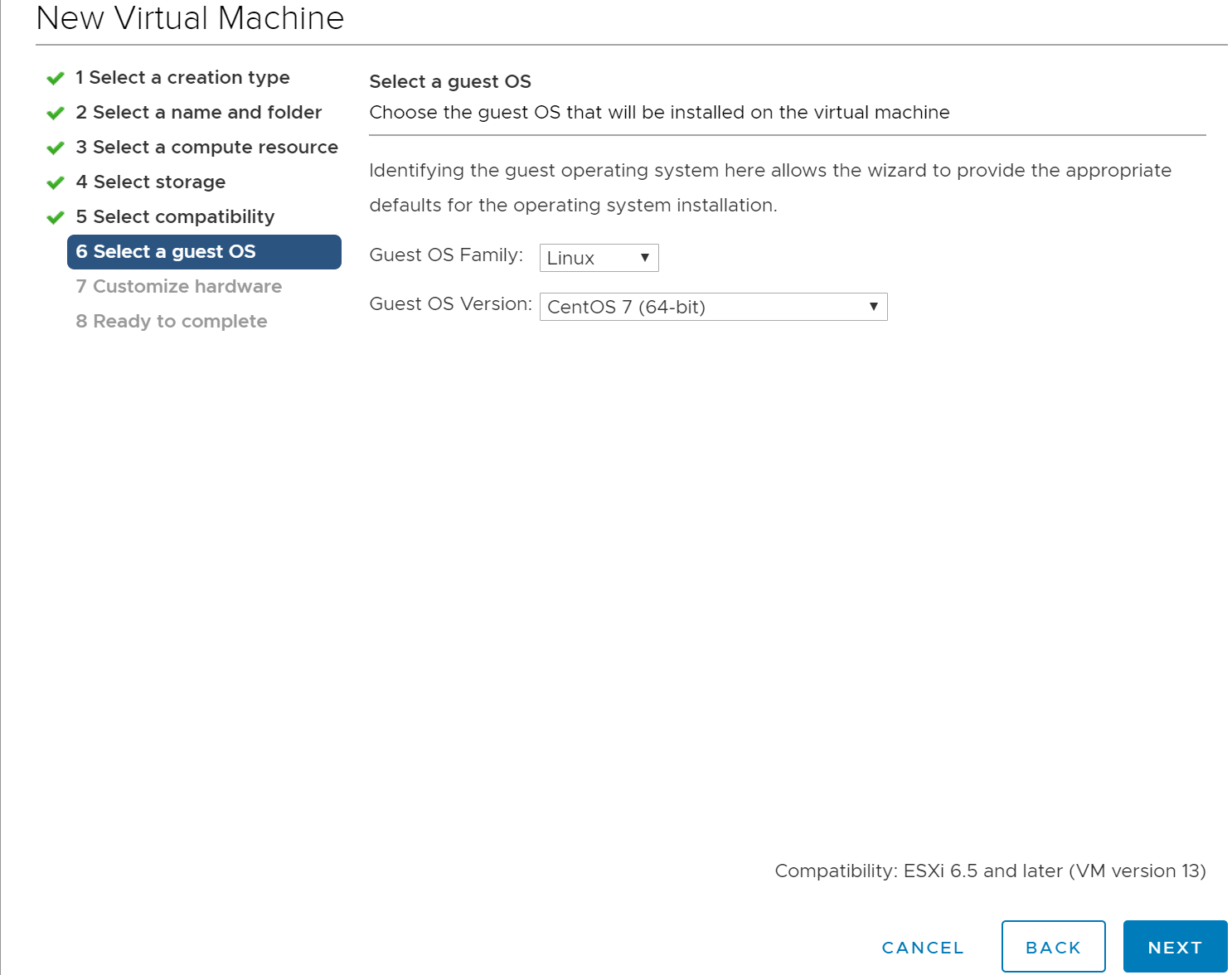
***Step 04: Select the datastore***



***Step 05: Select the ESXi compatibility***

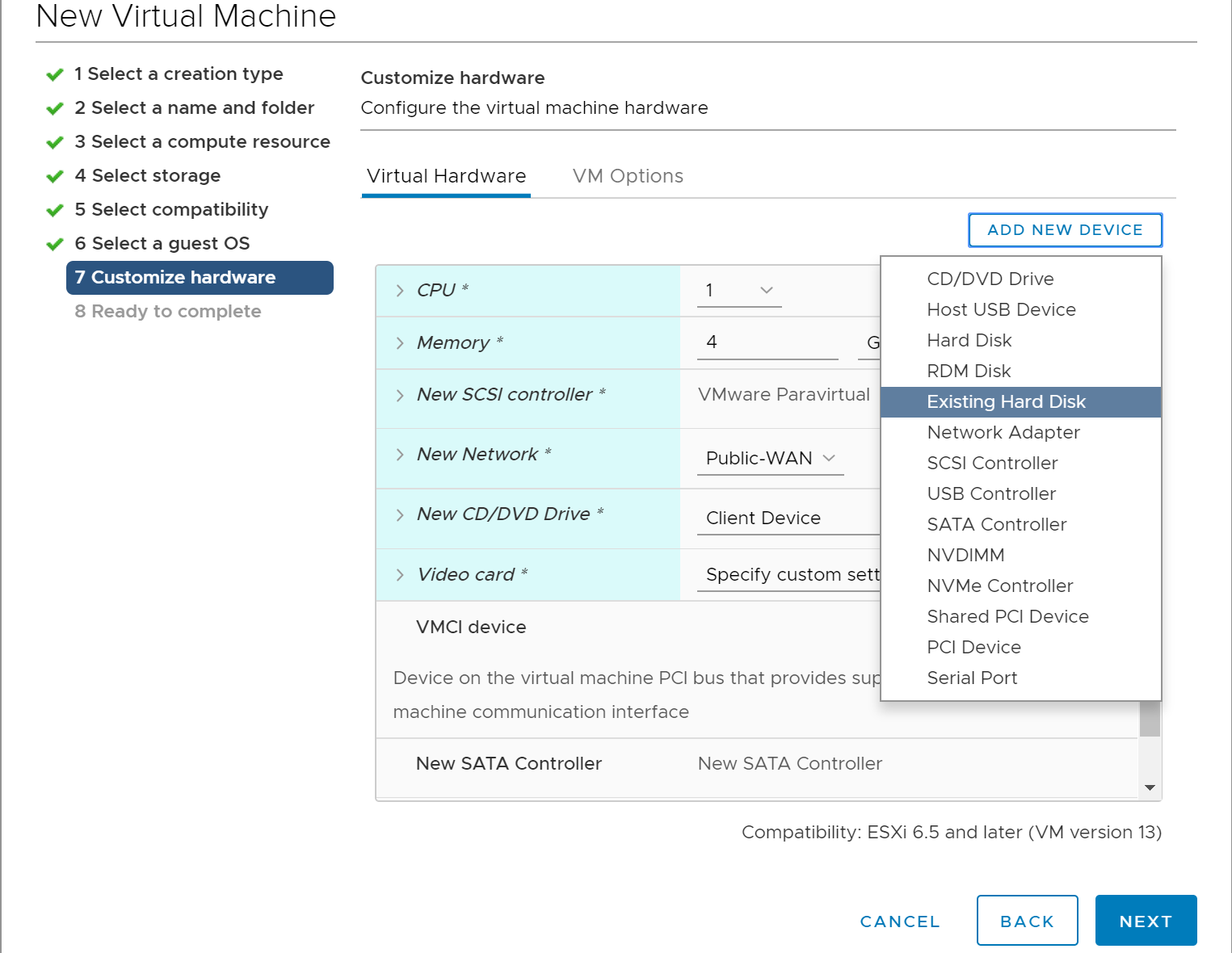


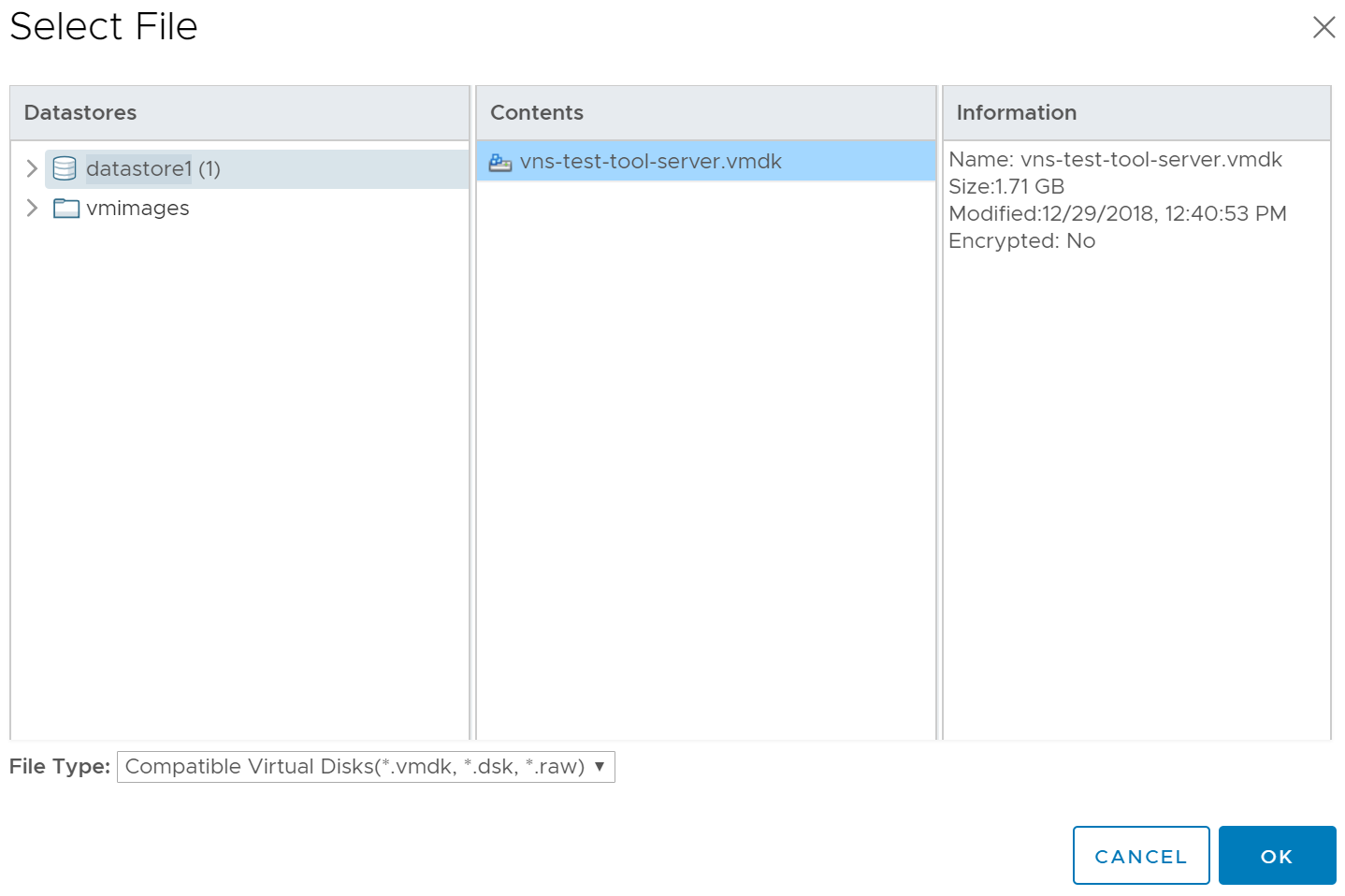
***Step 06: Select guest OS***



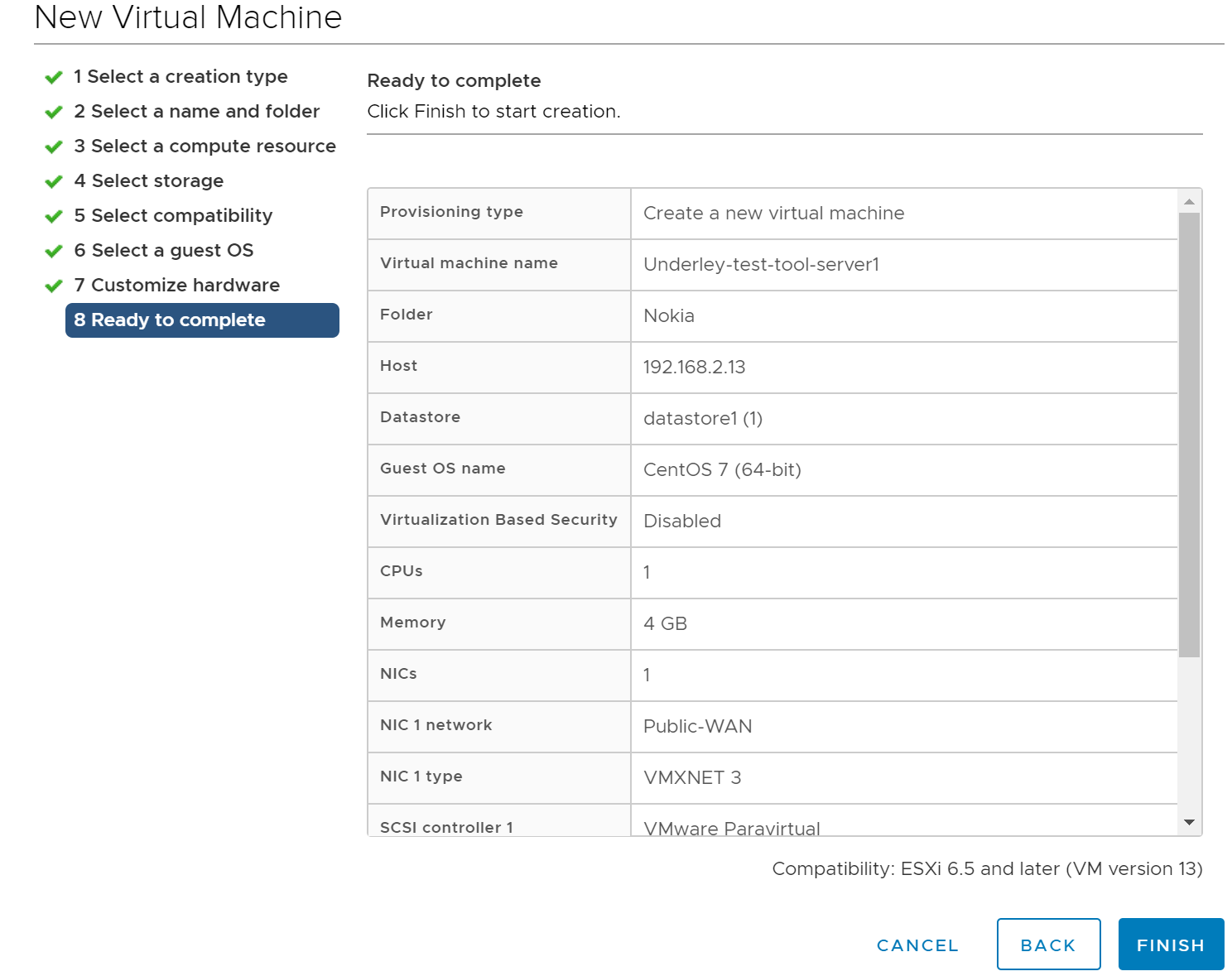
***Step 07: Add test-tool-server vmdk existing disk (remove the default disk and add a new hard disk as below)***

\*Make sure to select disk type driver “IDE”





***Step 08: Finalize and complete the setup.***



**Step 09:** Assign the server VM’s network interface to the appropriate VMWare network port group to locate the VM in the correct network landscape & power on the VM.

#### 3.1.3 Installation steps – KVM

**Step 01:** Copy the “vns-test-tool-server.xml” file in to the KVM host and define a VM.

Make sure to copy the “vns-test-tool-server.qcow2” image into the “/var/lib/libvirt/images/” file path.

virsh define /root/ vns-test-tool-server.xml

*Sample xml file:*

<domain type='kvm'>

<name>vns-test-tool-server</name>

<uuid>80b40d42-bbfc-4ec8-9a92-e000c3bfcb7c</uuid>

<memory unit='KiB'>4194304</memory>

<currentMemory unit='KiB'>4194304</currentMemory>

<vcpu placement='static'>1</vcpu>

<os>

<type arch='x86\_64' machine='pc-i440fx-rhel7.0.0'>hvm</type>

<boot dev='hd'/>

<boot dev='cdrom'/>

</os>

<features>

<acpi/>

<apic/>

<pae/>

</features>

<clock offset='utc'/>

<on\_poweroff>destroy</on\_poweroff>

<on\_reboot>restart</on\_reboot>

<on\_crash>restart</on\_crash>

<devices>

<emulator>/usr/libexec/qemu-kvm</emulator>

<disk type='file' device='disk'>

<driver name='qemu' type='qcow2'/>

<source file='/var/lib/libvirt/images/vns-test-tool-server.qcow2'/>

<target dev='vda' bus='virtio'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>

</disk>

<controller type='usb' index='0' model='piix3-uhci'>

<address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x2'/>

</controller>

<controller type='pci' index='0' model='pci-root'/>

<controller type='ide' index='0'>

<address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x1'/>

</controller>

<interface type='bridge'>

<mac address='52:54:00:1c:bd:92'/>

<source bridge='br0'/>

<model type='virtio'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>

</interface>

<serial type='pty'>

<target type='isa-serial' port='0'>

<model name='isa-serial'/>

</target>

</serial>

<console type='pty'>

<target type='serial' port='0'/>

</console>

<input type='tablet' bus='usb'>

<address type='usb' bus='0' port='1'/>

</input>

<input type='mouse' bus='ps2'/>

<input type='keyboard' bus='ps2'/>

<graphics type='vnc' port='-1' autoport='yes' listen='0.0.0.0'>

<listen type='address' address='0.0.0.0'/>

</graphics>

<sound model='ich6'>

<address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>

</sound>

<video>

<model type='cirrus' vram='16384' heads='1' primary='yes'/>

<address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>

</video>

<memballoon model='virtio'>

<address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>

</memballoon>

</devices>

</domain>

**Step 02:** Start the VM

virsh start vns-test-tool-server

## 3.2 Server instance configurations

**Step 1:** Log in to the server VM with below credentials:

User name: root

Password: asb!234

**Step 2:** Run “ifconfig” command and list down the network interfaces. Record the network interface device name.

[root@vns-test-tool ~]# ifconfig

ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500

…

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536

inet 127.0.0.1 netmask 255.0.0.0

…

Configure the network interface as below:

[root@vns-test-tool ~]# cat /etc/sysconfig/network-scripts/ifcfg-ens160

TYPE=Ethernet

BOOTPROTO=static

DEFROUTE=yes

NAME=ens160

DEVICE=ens160

ONBOOT=yes

IPADDR=192.168.10.35

PREFIX=24

GATEWAY=192.168.10.254

DNS1=192.168.5.10

**Step 03:** Run “/opt/nuage\_vns\_tester\_server/iptables.sh”

[root@vns-test-tool ~]# /opt/nuage\_vns\_tester\_server/iptables.sh

Appending iptables rules, please make sure iptables rules are empty.

The above script makes sure to populate the Linux host iptables for the optimal operation of the platform.

[root@vns-test-tool ~]# iptables --list

Chain INPUT (policy DROP)

target prot opt source destination

ACCEPT all -- anywhere anywhere state RELATED,ESTABLISHED

ACCEPT all -- anywhere anywhere /\* Allow loopback connections \*/

ACCEPT icmp -- anywhere anywhere /\* Allow Ping to work as expected \*/

ACCEPT tcp -- anywhere anywhere multiport dports ssh,bgp,893,targus-getdata1,6633,7407,11443 ,12443,39090,48179

ACCEPT udp -- anywhere anywhere multiport dports domain,ntp,isakmp,ipsec-nat-t,4789,50000,50 001,50002,50003

Chain FORWARD (policy DROP)

target prot opt source destination

Chain OUTPUT (policy ACCEPT)

target prot opt source destination

**Step 04:** Run “/opt/nuage\_vns\_tester\_server/sysctl.sh” to configure sysctl values

[root@vns-test-tool ~]# /opt/nuage\_vns\_tester\_server/sysctl.sh

Config sysctl params

net.ipv4.tcp\_syncookies = 1

net.ipv4.tcp\_fin\_timeout = 5

net.ipv4.tcp\_tw\_reuse = 1

net.ipv4.tcp\_tw\_recycle = 1

## 3.3 Start, stop and upgrades

**Start the server**

[root@vns-test-tool ~]# /opt/nuage\_vns\_tester\_server/vnstester.sh start

Starting Nuage VNS tester server.

**Stop the server**

[root@vns-test-tool ~]# /opt/nuage\_vns\_tester\_server/vnstester.sh stop

Stopping Nuage VNS tester server.

**Server Upgrades**

\*Assuming upgrade requires for the already deployed server instance.

Download the “nuageVnsTesterServer.tar.gz” source code from the latest release package. Copy the source code into the following folder path:

"/opt/nuage\_vns\_tester\_server”

## 4 Client software installation

**Platform requirements:**

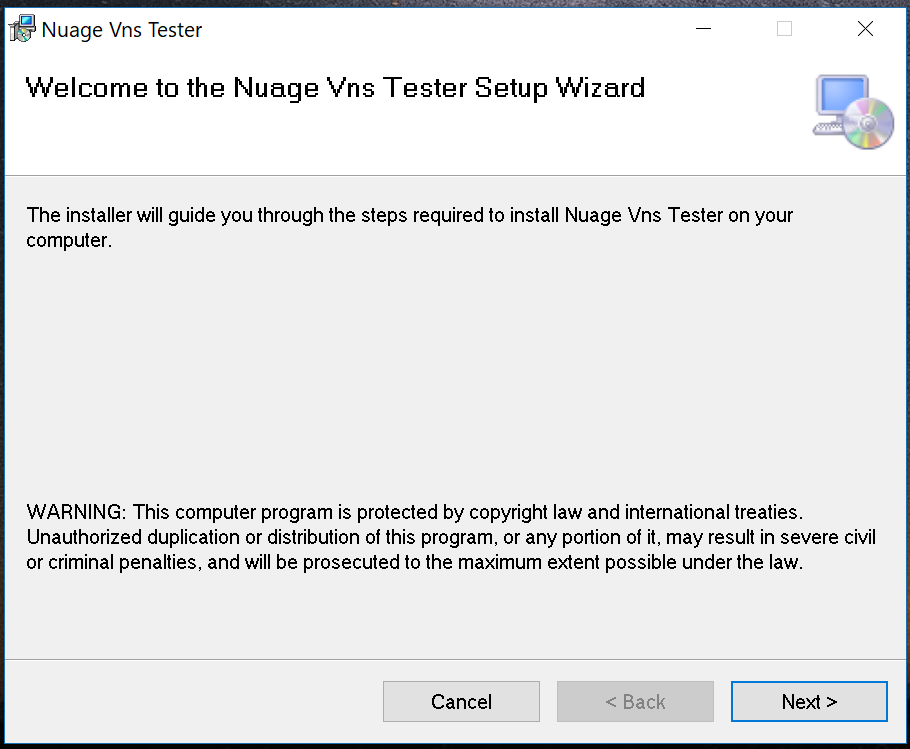
Tested and validated on Windows 7 and Windows 10 operating systems

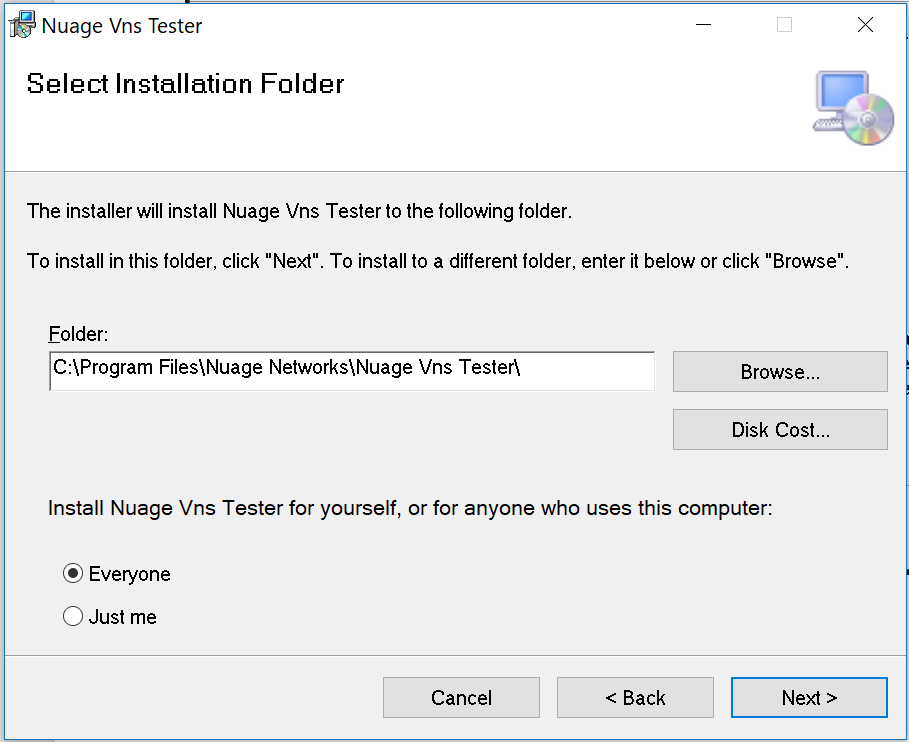
**Step 01:** Download the “vns-test-tool-client” folder into the user’s PC. Install **“nmap-7.70-setup”**. If ncap installation fails during the nmap installation then perform the “**npcap-0.99-r7.exe**” installation directly.

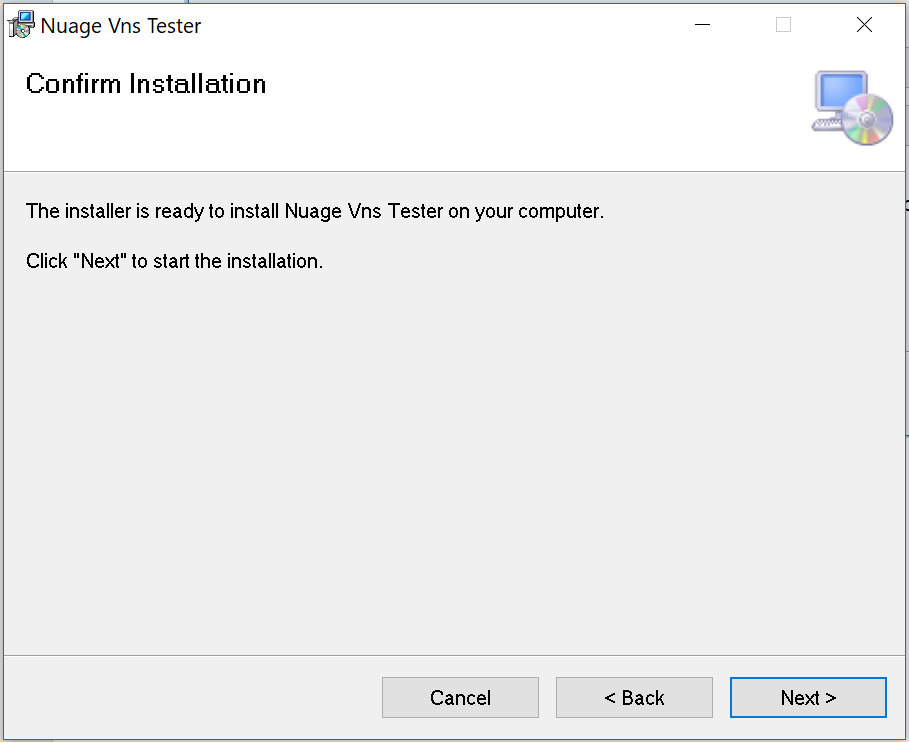
\*Both the nmap and ncap files are available within the “vns-test-tool-client” folder.

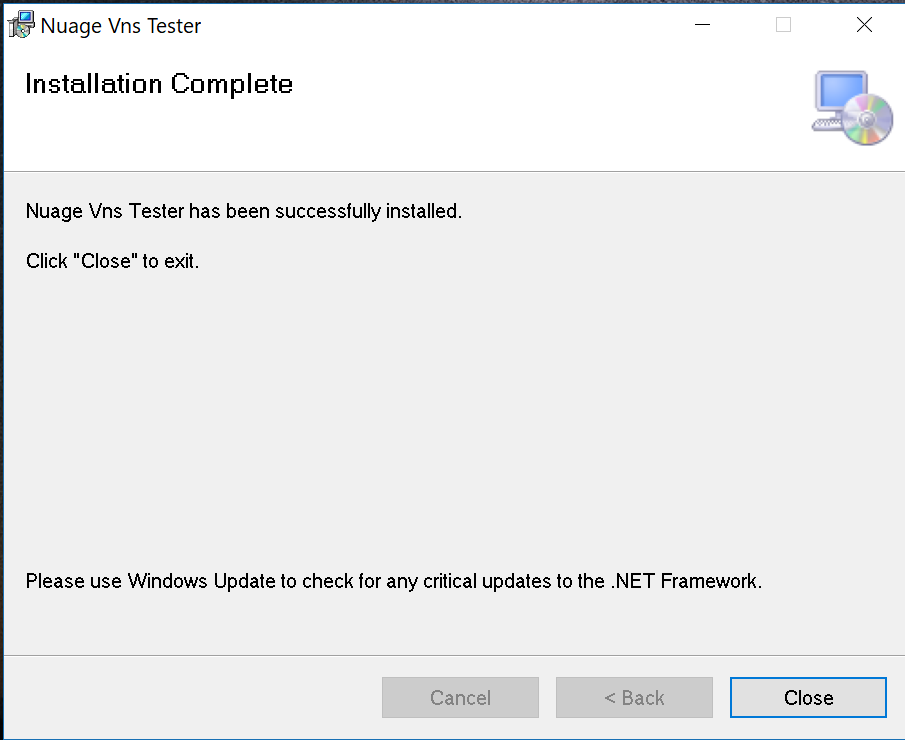
**Step 02:** Execute “NDP461-KB3102436-x86-x64-AllOS-ENU” and install .NET Framework 4.6.1.

**Step 03:** Run “**NuageVnsTesterSetup**” windows installer package and install the client.





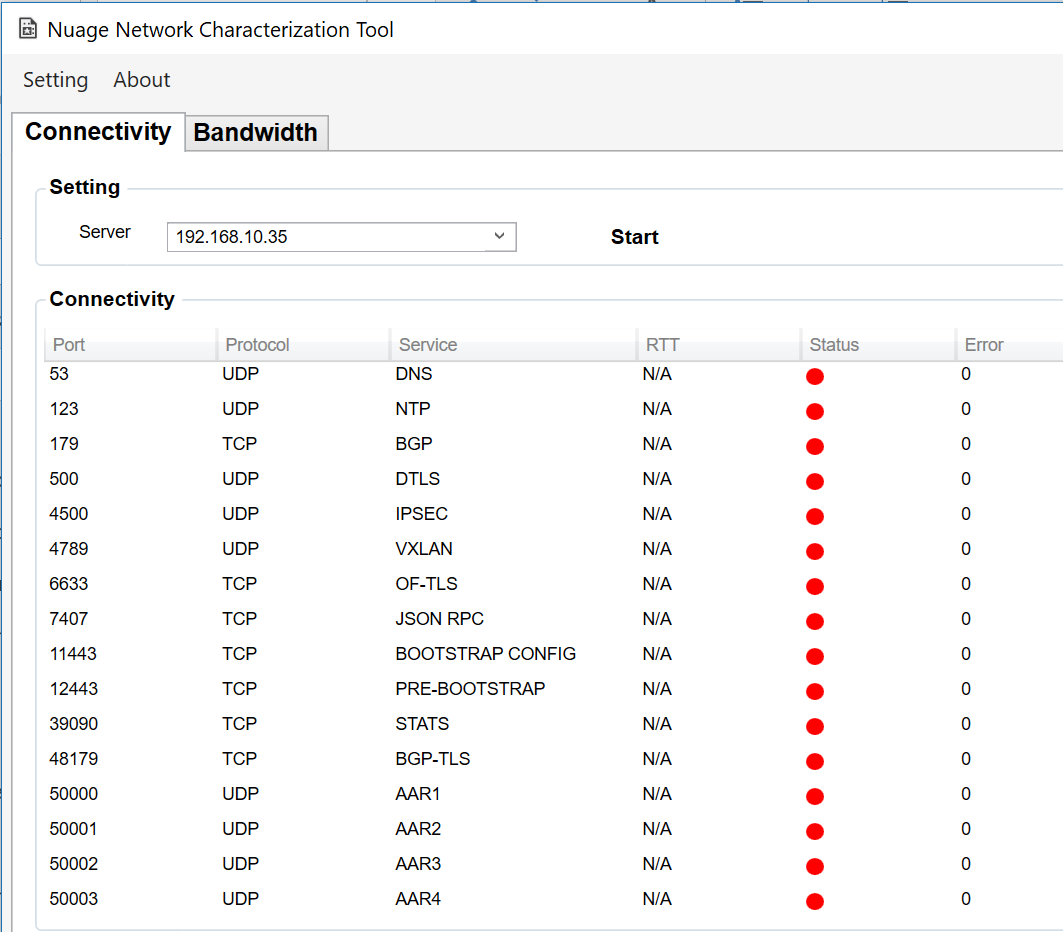




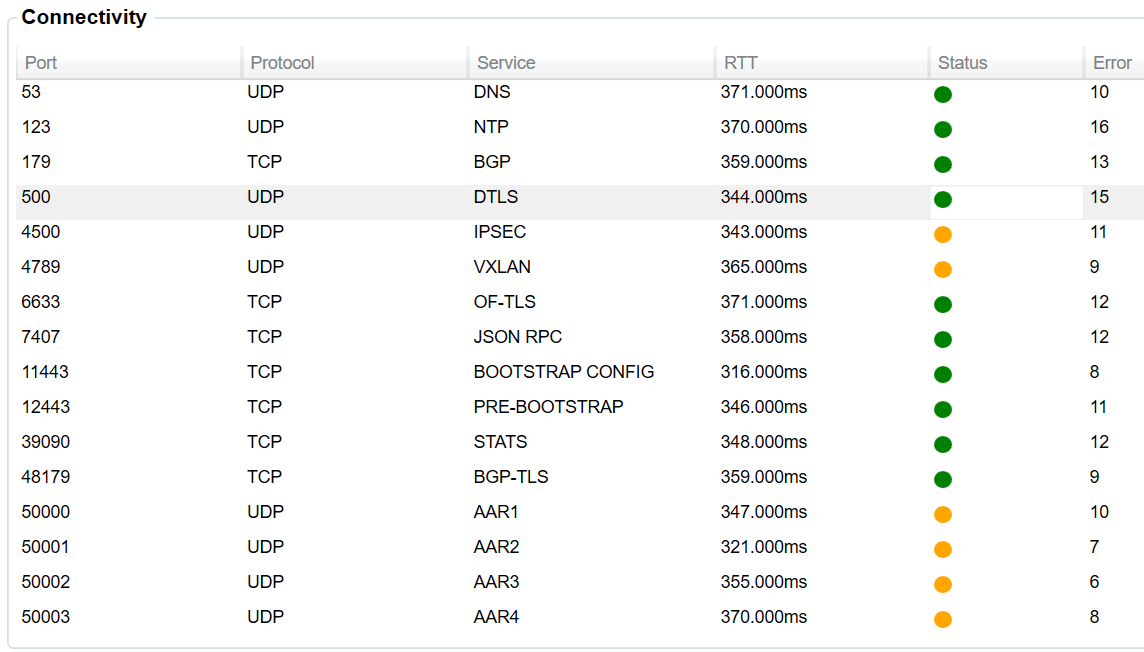
## 5 VNS Client Test Tool – Operational Usage

### 5.1 Connectivity testing based on port numbers and round-trip time

Go to the connectivity section and key in the vns-test-tool-server IP address and start monitoring.



A sample output:



Port Usage:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Protocol** | **Port** | **From** | **To** | **Description** |
| UDP | 53 | NSG | DNS Resolver | When the private MPLS underlay is in use: DNS server may reside within the management plane and still the test tool allows to check the service availability. But for the public internet this port checking may not applicable as the DNS resolver is within the internet space and shall not locate next to the test-server. |
| UDP | 123 | NSG | VSC | NTP |
| TCP | 179 | NSG | VSC | BGP |
| UDP | 500 | NSG | VSC | DTLS for IPSec |
| UDP | 4500 | NSG | VSC |
| UDP | 4789 | NSG | VSC | DTLS for VXLAN |
| TCP | 6633 | NSG | VSC | Open Flow |
| TCP | 7407 | NSG | VSC | JSON RPC |
| TCP | 11443 | NSG | VNS Proxy | NSG post-bootstrap |
| TCP | 12443 | NSG | VNS Proxy | NSG pre-bootstrap |
| TCP | 39090 | NSG | VNS Proxy | Stats collection |
| TCP | 48179 | NSG | VSC | BGP-TLS |
| UDP | 50000 | NSG | VSC | NSG Application Aware Routing [AAR] process probes |
| UDP | 50001 | NSG | VSC |
| UDP | 50002 | NSG | VSC |
| UDP | 50003 | NSG | VSC |

**Monitoring and reporting:**

*Acceptable maximum delay (RTT) and packet loss:*

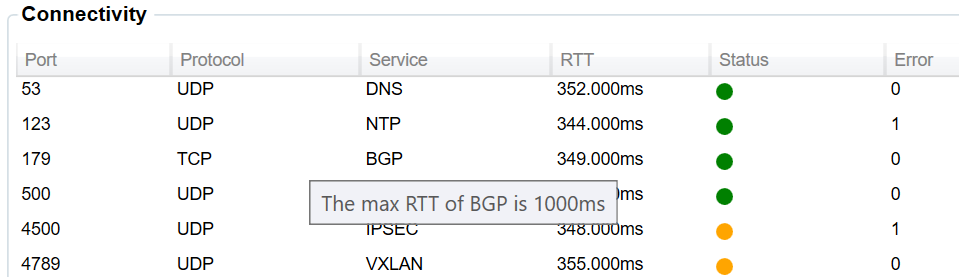
|  |  |  |
| --- | --- | --- |
| **From 🡨🡪 To** | **Packet Loss** | **Delay** |
| NSG To VSC | 2% | 1000 ms |
| NSG To VNS Proxy | 1% | 850 ms |

When the mouse hover on the RTT column it shows the acceptable value in milli seconds. The “***status***” column shows:

GREEN – the RTT is within the acceptable value

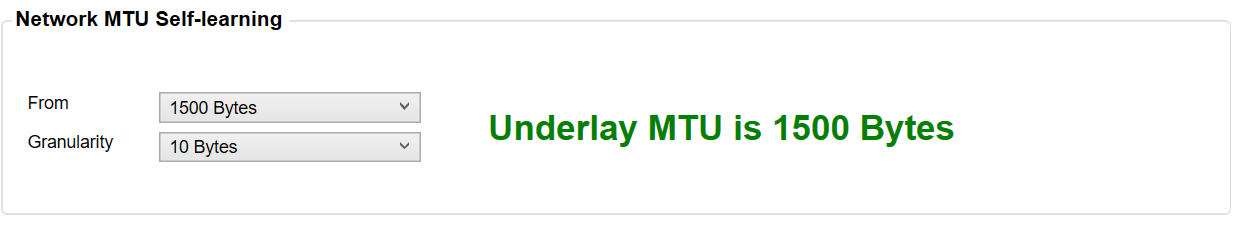
YELLOW – when the maximum RTT is breached

RED - when the port is not reachable



Error – Column shows the number of errors occurred in terms of ???

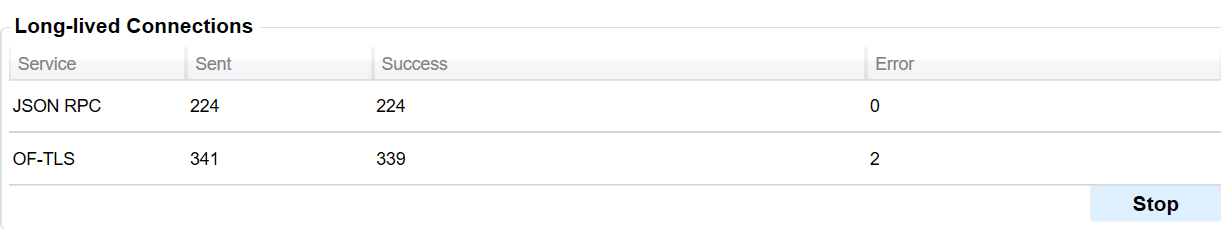
### 5.2 Network MTU Self Learn



Set the MTU “*from-field*” starts from “1300 or 1400 or 1500” bytes to self-learn the underlay MTU size. Set the granularity field start from either “10 or 50 or 100” bytes.

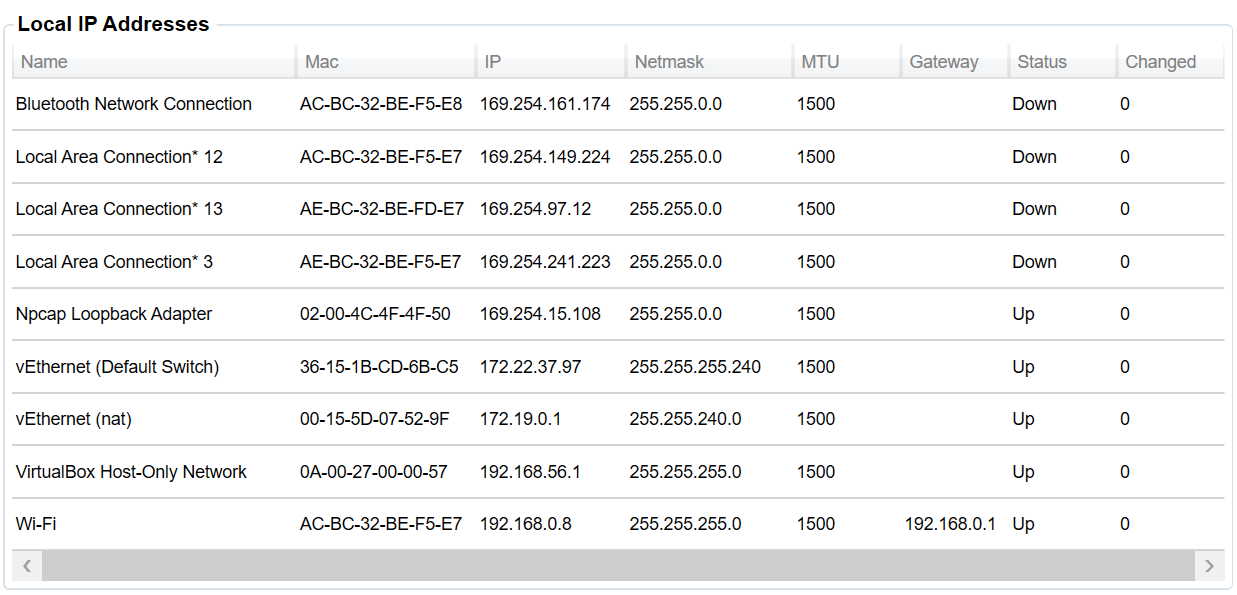
### 5.3 Long Lived Connections

In real world deployment scenarios NSG may resides behind 3rd party modems and routers. In some occasions a firewall or connection controlling mechanisms are within these 3rd party modems. These applications may cause some issues in a long run. A classic example is some modems tear down the Openflow connection in every few minutes. The long-lived connection section allows to monitor these sessions and find out any abnormal behaviour on the upstream underlay.

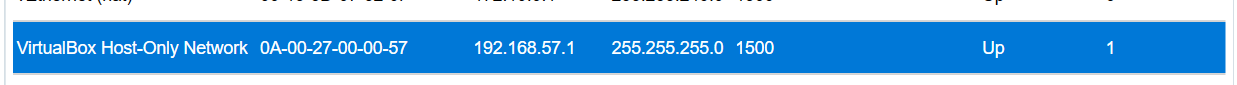


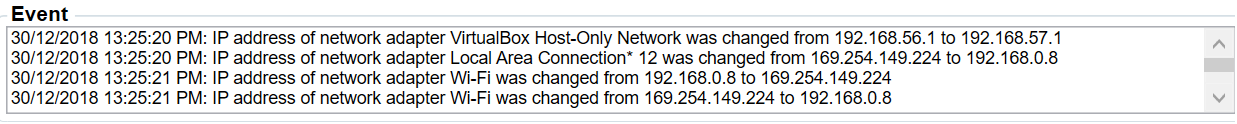
The control panel shows how many packets are being sent and received. If any packet loss detects then it records in the Error column.

### 5.4 Local PC IP Addresses and events

****

This section shows all the local network interface related details including IP addresses and IP changes. If any IP change occurs then it records as an event in the events section.

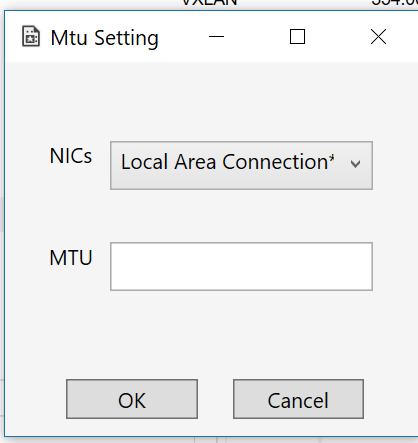




30/12/2018 13:25:20 PM: IP address of network adapter VirtualBox Host-Only Network was changed from 192.168.56.1 to 192.168.57.130/12/2018 13:25:20 PM: IP address of network adapter Local Area Connection\* 12 was changed from 169.254.149.224 to 192.168.0.830/12/2018 13:25:21 PM: IP address of network adapter Wi-Fi was changed from 192.168.0.8 to 169.254.149.22430/12/2018 13:25:21 PM: IP address of network adapter Wi-Fi was changed from 169.254.149.224 to 192.168.0.830/12/2018 13:25:24 PM: IP address of network adapter Local Area Connection\* 12 was changed from 192.168.0.8 to 169.254.149.22430/12/2018 13:25:24 PM: IP address of network adapter Local Area Connection\* 12 was changed from 169.254.149.224 to 192.168.0.830/12/2018 13:25:24 PM: IP address of network adapter Wi-Fi was changed from 192.168.0.8 to 169.254.149.22430/12/2018 13:25:24 PM: IP address of network adapter Wi-Fi was changed from 169.254.149.224 to 192.168.0.8

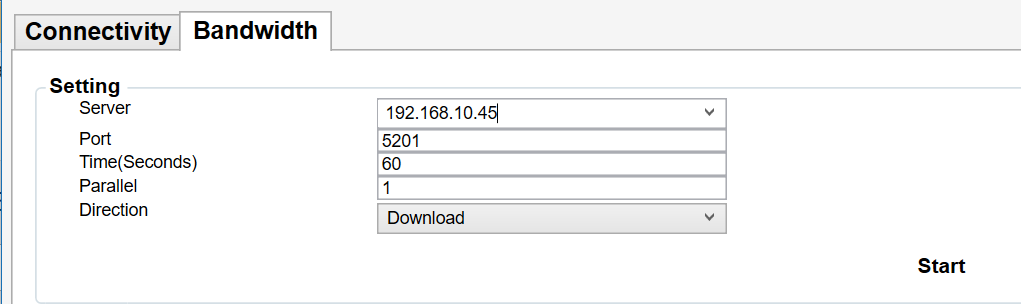
### 5.5 MTU Settings

The client utility allows to change the PC network interface MTU size via settings 🡪 MTU Setting as below.

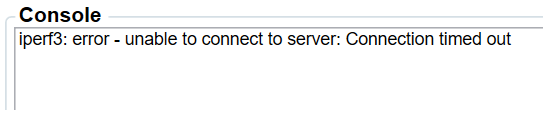


### 5.6 Bandwidth Monitoring

Key in the correct server IP address in the “server” option



Iperf3 is in use under the hood to perform this bandwidth monitoring. If the port 5201 is blocked from the firewall or the server IP is not reachable the error message may appear in the console panel.



**Port**: 5201 is the default port that iperf3 server listen.

**Time** (seconds): default is set to 60. This instructs time in seconds to transmit for

**Parallel**: number of parallel client streams to run

**Direction**: Upload/Download

Sample execution:

Console Output:

***Download***

warning: Ignoring nonsense TCP MSS 0

Connecting to host 192.168.10.45, port 5201

Reverse mode, remote host 192.168.10.45 is sending

[ 4] local 192.168.0.8 port 13746 connected to 192.168.10.45 port 5201

[ ID] Interval Transfer Bitrate

[ 4] 0.00-1.00 sec 99.8 KBytes 0.82 Mbits/sec

[ 4] 1.00-2.00 sec 287 KBytes 2.35 Mbits/sec

[ 4] 2.00-3.00 sec 1.59 MBytes 13.3 Mbits/sec

[ 4] 3.00-4.00 sec 2.04 MBytes 17.1 Mbits/sec

[ 4] 4.00-5.00 sec 2.03 MBytes 17.1 Mbits/sec

[ 4] 5.00-6.00 sec 2.29 MBytes 19.2 Mbits/sec

[ 4] 6.00-7.00 sec 2.46 MBytes 20.7 Mbits/sec

[ 4] 7.00-8.00 sec 2.06 MBytes 17.3 Mbits/sec

[ 4] 8.00-9.00 sec 2.33 MBytes 19.5 Mbits/sec

[ 4] 9.00-10.00 sec 2.21 MBytes 18.5 Mbits/sec

- - - - - - - - - - - - - - - - - - - - - - - - -

[ ID] Interval Transfer Bitrate Retr

[ 4] 0.00-10.35 sec 20.2 MBytes 16.4 Mbits/sec 1 sender

[ 4] 0.00-10.00 sec 17.4 MBytes 14.6 Mbits/sec receiver

iperf Done.

***Upload***

warning: Ignoring nonsense TCP MSS 0

Connecting to host 192.168.10.45, port 5201

[ 4] local 192.168.0.8 port 13832 connected to 192.168.10.45 port 5201

[ ID] Interval Transfer Bitrate

[ 4] 0.00-1.00 sec 256 KBytes 2.10 Mbits/sec

[ 4] 1.00-2.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 2.00-3.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 3.00-4.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 4.00-5.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 5.00-6.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 6.00-7.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 7.00-8.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 8.00-9.00 sec 0.00 Bytes 0.00 Mbits/sec

[ 4] 9.00-10.00 sec 128 KBytes 1.05 Mbits/sec

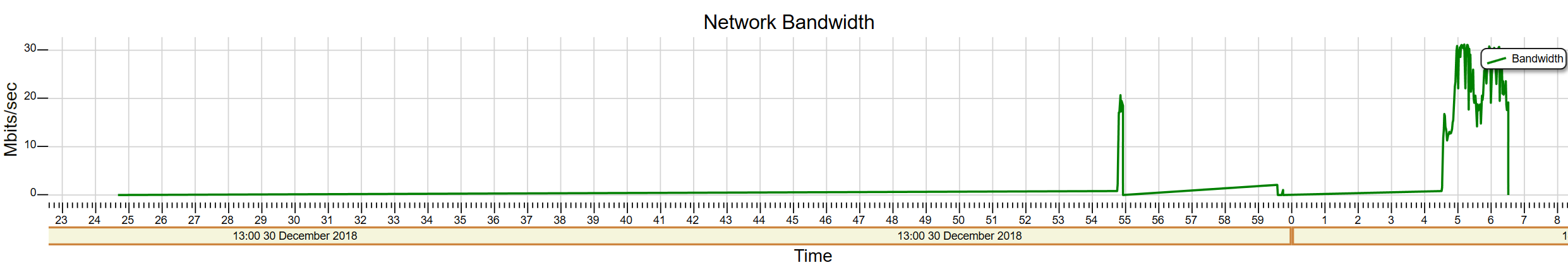
- - - - - - - - - - - - - - - - - - - - - - - - -

[ ID] Interval Transfer Bitrate

[ 4] 0.00-10.00 sec 384 KBytes 0.31 Mbits/sec sender

[ 4] 0.00-10.36 sec 154 KBytes 0.12 Mbits/sec receiver

iperf Done.



The bandwidth usage showcase as above for visualization purpose.