NSFOCUS ADS NX1-VN Installation and Deployment Guide

Version: V4.5R90F06 (2024-12-31)

Confidentiality: RESTRICTED

■ Statement

The purchased products, services, or features are stipulated in the contract made between NSFOCUS and the customer. Part of products, services, and features described in this document may not be within the purchased scope or the usage scope.

All information in this document is provided "AS-IS" without guarantees of any kind, express or implied. The information in this document is subject to change without notice. It may slightly differ from the actual product due to version upgrade or other reasons.

■ Disclaimer

Please read the disclaimer carefully before using the product. Once you use the product, you acknowledge and agree to all the contents of this disclaimer. NSFOCUS shall not assume any responsibility for any loss or damage in the following circumstances:

- Data loss and system availability reduction caused by the negligence or misconduct of the system O&M or management personnel, for example, they do not handle alerts that affect system stability and availability in a timely manner.
- Data loss and system availability reduction caused by the fact that the traffic exceeds the planned hardware capacity.
- Data loss and system availability reduction or unavailability caused by natural disasters (including but not limited to floods, fires, and earthquakes) or environmental factors (including but not limited to network disconnection and power outage).

Contents

Preface

Scope

This document briefly describes NSFOCUS Anti-DDoS System NX1-VN series (vADS) and details how to deploy and install it.

Currently, vADS supports the Kernel-based Virtual Machine (KVM) and VMware ESXi platforms. Users of other host machine types should perform configuration by referring to other related documents.

This document is provided for reference only. It may slightly differ from the actual product due to version upgrade of the virtual platform or other reasons.

Audience

This document is intended for the following users:

- Users who wish to provide anti-DDoS services for users via vADS
- Users who wish to know main features and usage of this product
- System administrator
- Network administrator

This document assumes that you have knowledge in the following areas:

- Virtualization
- Cybersecurity
- Linux operating systems
- TCP/IP protocols
- KVM
- VMware ESXi
- ADS

Organization

Chapter	Description
1 Basic Information	Describes requirements for configuring the host and vADS.
2 Deployment on KVM	Describes how to import and configure vADS on KVM.
3 Deployment on VMware ESXi	Describes how to import and configure vADS on VMware ESXi.
A Default Parameters	Describes default parameters of vADS.
B Terminology	Describes terminologies associated with vADS.

Chapter	Description
C FAQs	Describes frequently asked questions (FAQs).

Change History

Version	Description
V4.5R90F06	Updated the template.
V4.5R90F04	• Updated the structure based on the new template. • Revised the minimum CPU requirements for KVM and VMware configurations.

Conventions

Convention	Description
Bold font	Keywords, names of screen elements like buttons, drop-down lists or fields, and user-entered text appear in bold font.
Italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in italic font.
Note	Reminds users to take note.
Tip	Indicates a tip to make your operations easier.
Caution	Indicates a situation in which you might perform an action that could result in equipment damage or loss of data.
Warning	Indicates a situation in which you might perform an action that could result in bodily injury.
A > **B**	Indicates selection of menu options.

Technical Support

Hardware and Software Support

Email: support@nsfocusglobal.com

Cloud Mitigation Support

Email: cloud-support@nsfocusglobal.com

Phone:

- USA: +1-844-673-6287 or +1-844-NSFOCUS

- UK: +44 808 164 0673 or +44 808 164 0NSF
- Australia: +61 2 8599 0673 or +61 2 8599 0NSF
- Netherlands: +31 85 208 2673 or +31 85 208 2NSF
- Brazil: +55 13 4042 1673 or +55 13 4042 1NSF
- Japan: +81 3-4510-8673 or +81 3-4510-8NSF
- Singapore: +65 3158 3757
- Hong Kong +852 5803 2673 or +852 5803 2NSF
- Middle East: +973 1619 7607

Documentation Feedback

For any query regarding the usage of the documentation, you can contact us:

Email: info-support@nsfocus.com

1 Basic Information

This document describes requirements for configuring the host and vADS.

This chapter covers the following topics:

Topic	Description
Host Configuration Requirements	Describes configuration requirements of the host.
VM Configuration Requirements	Describes configuration requirements of vADS.

1.1 Host Configuration Requirements

vADS should run on a host with virtual machine software installed. Make sure that the host meets all requirements listed in Table 1-1 and Table 1-2.

Table 1-1 Reference configuration of the host

Item	Reference Configuration
CPU	Intel(R) Xeon(R) CPU E5-2687W v4 @ 3.00 GHz
Memory	128 GB (at least 32 GB free space)
Hard drive	1 TB (at least 10 GB free space)

Table 1-2 Reference configuration of NICs

NIC Type	Model	Quantity
1000M	I210, I350, 82571, 82576, and 82580	1–8
10G	82599 and X710/XL710	1–4
Virtual	Virtual NICs other than the models listed above	1–8

Note	vADS does not support use of more than one type of network interface card (NIC). That is to say, vADS can work properly only when NICs configured are of the same type, namely, 1000M, 10G, or virtual.

1.2 VM Configuration Requirements

1.2.1 KVM Configuration Requirements

Table 1-3 lists the requirements for configuring vADS on KVM.

Table 1-3 KVM configuration requirements

Item	**vADS**				
Hypervisor support	QEMU KVM 1.5.3 or later versions				
vCPU number	An even number ranging from 4 to 32				
Storage	At least 10 GB				
Mitigation capacity	(@128bytes)	200M-2G	10G	20G	40G
Minimum requirement	CPU cores	4	6	14	22
	Memory	20G	20G	20G	32G

1.2.2 VMware Configuration Requirements

Table 1-4 lists the requirements for configuring vADS on VMware.

Table 1-4 VMware configuration requirements

Item	**vADS**				
Hypervisor support	VMware ESXi 6.5 or later versions				
vCPU number	An even number ranging from 4 to 32				
Storage	At least 10 GB				
Mitigation capacity	(@128bytes)	200M-2G	10G	20G	40G
Minimum requirement	CPU cores	4	6	10	22
	Memory	20G	20G	20G	32G

2 Deployment on KVM

This chapter describes how to import and configure vADS on KVM.

This chapter covers the following topics:

Topic	Description
Preparations	Describes preparations to be made for installing vADS on KVM.
Installation Procedure	Describes how to install vADS on KVM.

2.1 Preparations

Before installing vADS locally, you must make preparations listed in Table 2-1.

Table 2-1 List of items to be prepared for installing vADS locally

Item		Description
Host	IP address	IP address of the host that can properly connect to the network
	Account	Account with privileges of a system administrator
	Network interface	At least one 1000M interface available
	Operating system	CentOS 7 recommended
vADS	vADS image file	Including **vads.img** and **vads.xml**
	IP address	IP address of the management interface of vADS

2.1.1 Installing and Configuring the Host System

To install and configure the host system, follow these steps:

Step 1 Install CentOS 7.

For details on the installation process, visit https://docs.centos.org/en-US/centos/install-guide/.

Step 2 Install some basic tools.

Run the following commands to install some tools for the use of certain networks and PCI commands:

```
yum -y install net-tools
yum -y install pciutils
yum -y install lshw
yum -y install numactl
```

----End

2.1.2 Installing KVM

To install KVM, follow these steps:

Step 1 Install KVM as **root** from the network.

```
yum install kvm  virt-viewer virt-manager libvirt libvirt-python python-virtinst
libvirt-client qemu-kvm qemu-img bridge-utils -y
```

Step 2 Start KVM.

```
systemctl start libvirtd     #starts KVM.
systemctl enable libvirtd   #sets KVM to start upon system boot.
```

----End

2.1.3 Configuring the Network Bridge Connection

2.1.3.1 Configuration Requirements

Create a bridge interface. By default, vADS's management interface uses the bridge NIC **br0**.

For details on configuration commands and parameters, visit the following link:

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/networking_guide/sec-network_bridging_using_the_command_line_interface

2.1.3.2 Configuration Example

Create a bridge interface br0 on the Ethernet interface em3 and set the IP address of this bridge interface.

Step 1 Perform network configurations.

In **/etc/sysconfig/network-scripts/ifcfg-em3**, configure parameters as follows:

```
DEVICE="em3"
ONBOOT=yes
BRIDGE="br0"
```

In **/etc/sysconfig/network-scripts/ifcfg-br0**, configure parameters as follows:

```
IPADDR="192.168.1.100"
NETMASK="255.255.255.0"
GATEWAY="192.168.1.254"
DEVICE="br0"
ONBOOT="yes"
BOOTPROTO="none"
STP="on"
DELAY="0"
```

```
TYPE="Bridge"
```

Note	
	• The interface em3 should be changed to the actual interface of the server. • The host information, including IPADDR, NETMASK, and GATEWAY, should be configured according to the actual network deployment scenario.

Step 2 Restart the network.

```
systemctl restart network
```

Step 3 Verify that the bridge interface is successfully configured.

```
brctl show
#-----The command output is as follows:--------
bridge name     bridge id               STP enabled     interfaces
br0             8000.246e9660c50c       yes             em3
```

----End

2.1.4 Virtualization

2.1.4.1 Enabling Virtualization

To enable virtualization, follow these steps:

Step 1 Reboot the computer and open the system's BIOS menu.

This can be done by pressing **Delete**, **F1**, or **Alt+F4**, depending on the operating system you use.

Step 2 Enable virtualization extensions in BIOS.

a. Open the **Processor** submenu. The processor settings menu may be hidden in the **Chipset**, **Advanced CPU Configuration**, or **North Bridge** tabs.

b. Enable **Intel Virtualization Technology** (also known as Intel VT-x). AMD-V extensions cannot be disabled in the BIOS and should already be enabled. The virtualization extensions may be labeled **Virtualization Extensions**, **Vanderpool**, or other names, depending on the OEM and system BIOS.

c. Enable **Intel VTd** or **AMD IOMMU**, if these options are available. They are used for PCI device assignment.

d. Select **Save & Exit**.

Note	
	The preceding configurations may vary with your motherboard, processor type, chipset, and OEM. For how to correctly configure your system, see your system's accompanying documentation.

Step 3 Restart the computer.

Step 4 Check whether virtualization is enabled.

Run the following command to check whether CPU virtualization extensions are available. If there is no command output, virtualization extensions are not enabled. In this case, you need to check and modify BIOS settings accordingly.

```
grep -E "vmx|svm" /proc/cpuinfo
```

Run the following command to check whether virtualization extensions are available. If there is no command output, virtualization extensions are not enabled and device assignment cannot be done. If device assignment is required for NICs, you need to check and modify BIOS settings.

```
ls /sys/kernel/iommu_groups/
```

Step 5 Configure the GRUB on the host to enable NIC device assignment.

Edit **/etc/default/grub** by adding the following line:

```
GRUB_CMDLINE_LINUX_DEFAULT=" intel_iommu=on";
```

a. Run the following command to modify the system GRUB:

```
grub2-mkconfig -o $(find / -name grub.cfg | head -1)
```

b. Restart the host (or do this after the CPU isolation configuration is complete).

c. You can use the following command to confirm whether the configuration is successful.

```
cat /proc/cmdline
#---------The command output is as follows:----------
BOOT_IMAGE=/vmlinuz-3.10.0-957.el7.x86_64 root=/dev/mapper/centos-root ro
crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet
intel_iommu=on isolcpus=1-11,13-23
```

----End

2.1.4.2 Example

The following is an example of enabling virtualization:

Step 1 Enable CPU virtualization (Intel Virtualization), as shown in Figure 2-1 and Figure 2-2.

Figure 2-1 Enabling CPU virtualization (substep 1)

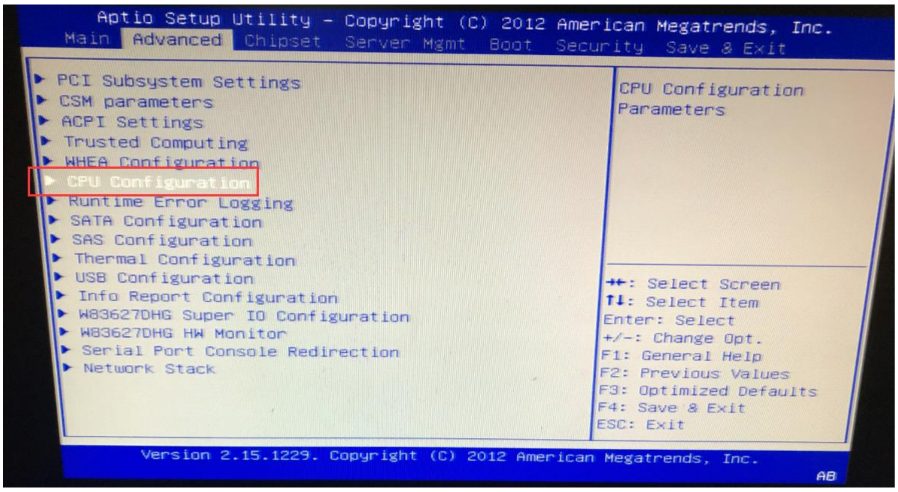
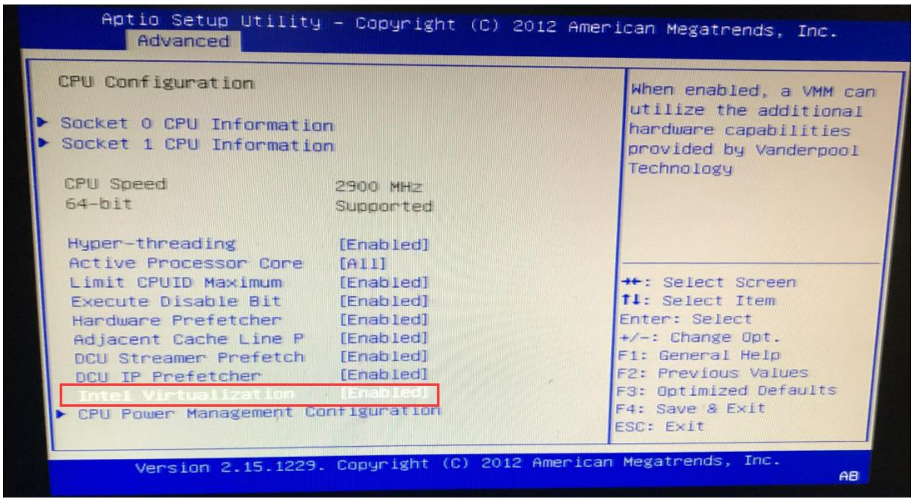


Figure 2-2 Enabling CPU virtualization (substep 2)



Step 2 Enable IOMMU support (Intel(R) VT-d) in the BIOS.

Figure 2-3 Enabling IOMMU support (Intel(R) VT-d) in BIOS (substep 1)

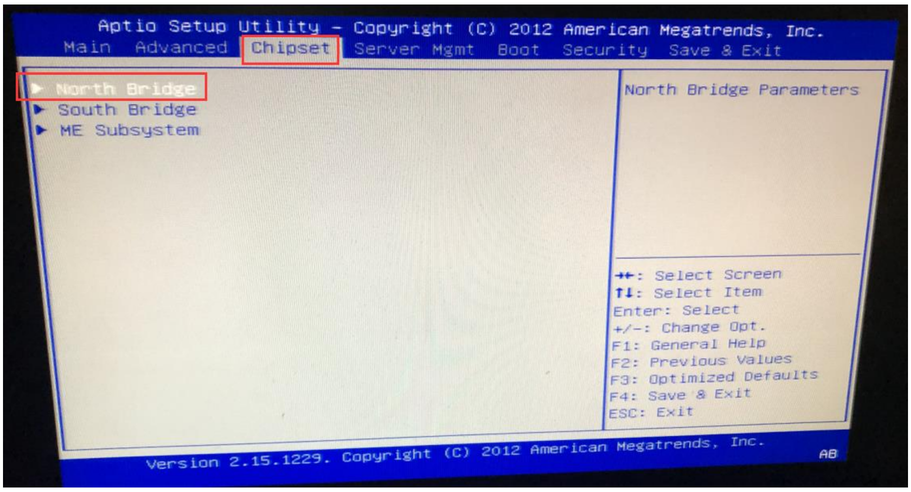


Figure 2-4 Enabling IOMMU support (Intel(R) VT-d) in BIOS (substep 2)

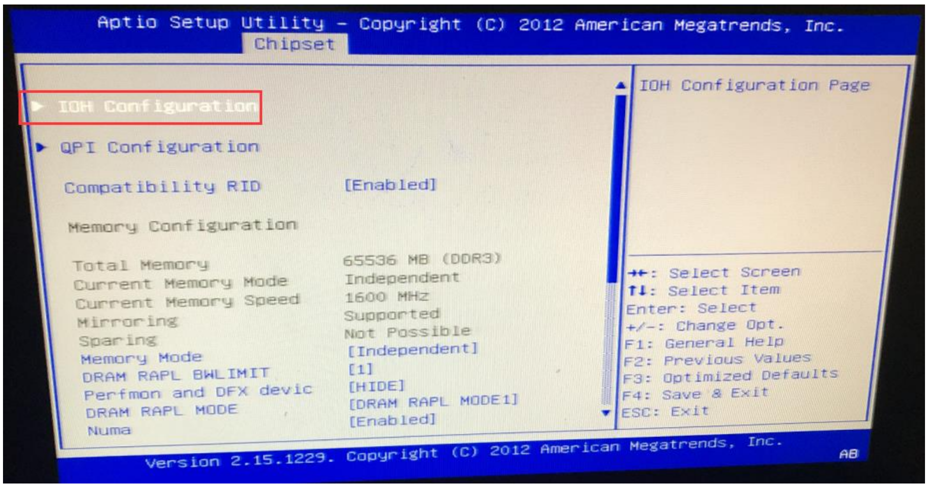


Figure 2-5 Enabling IOMMU support (Intel(R) VT-d) in BIOS (substep 3)

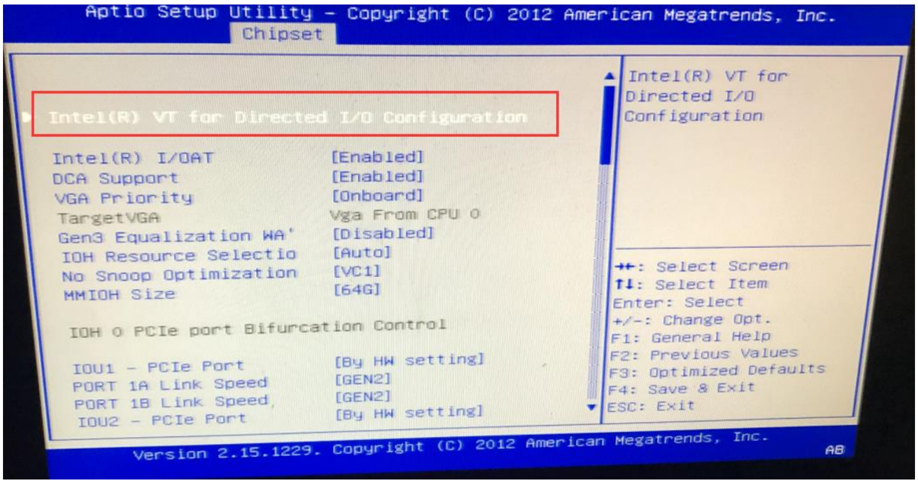
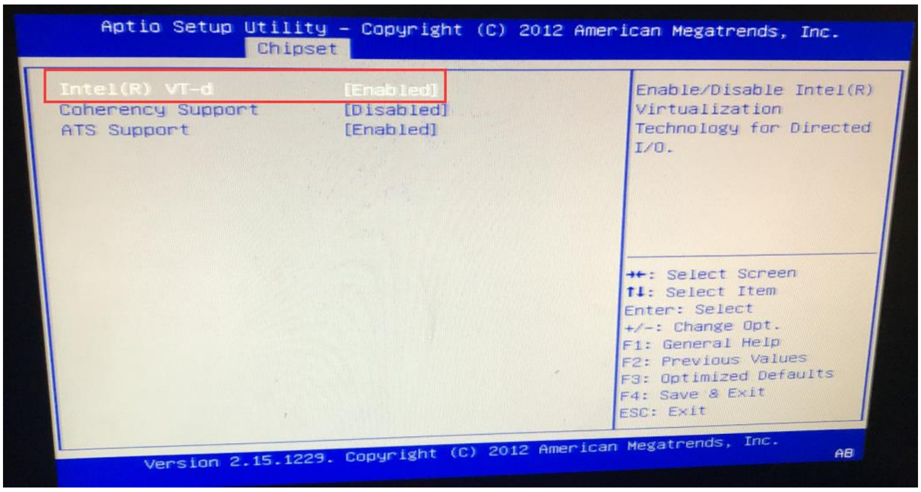
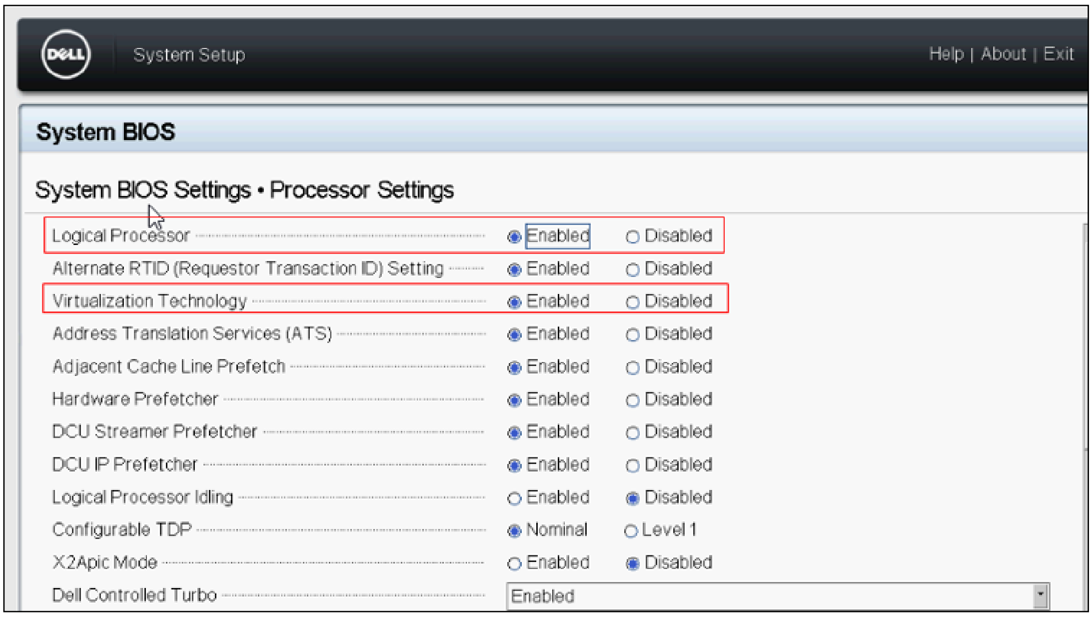


Figure 2-6 Enabling IOMMU support (Intel(R) VT-d) in BIOS (substep 4)



Step 3 Choose **Bios** > **Processor Settings** > **Virtualization Technology** and set Dell R730 BIOS parameters.

Figure 2-7 Setting Dell R730 BIOS parameters



----End

2.2 Installation Procedure

Note	The following operations, namely command executions and file edits, are all done on a Linux host.

2.2.1 Importing the vADS Image

Before importing the vADS image, you need to obtain it, which contains two files: **vads.img** and **vads.xml**.

To import the vADS image, follow these steps:

Step 1 Log in to the host and define the **/home/ADS** directory.

```
mkdir -p /home/ADS
```

Step 2 Put the vADS image file in the **/home/ADS** directory.

Step 3 Run the following command to import vADS.

```
virsh define /home/ADS/vads.xml
```

----End

2.2.2 Configuring CPU Isolation

ADS is a system sensitive to CPU usage. For the use of vADS, you need to isolate a certain number of CPU cores for vADS's exclusive use. This section describes how to implement CPU isolation.

Table 2-2 describes basic concepts. As shown in Figure 2-8, the host has two physical CPUs, each of which has eight cores that have two hyper threads respectively.

Table 2-2 Basic concepts

Concept	Description
CPU(s)	Number of hyper threads
Socket(s)	Number of physical CPUs
Core(s) per socket	Number of cores of each physical CPU.
Thread(s) per core	Number of hyper threads in each CPU core

Figure 2-8 Basic concepts

```
[root@localhost ~]# lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                32
On-line CPU(s) list:   0-31
Thread(s) per core:    2
Core(s) per socket:    8
Socket(s):             2
NUMA node(s):          2
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 45
Model name:            Intel(R) Xeon(R) CPU E5-2690 0 @ 2.90GHz
Stepping:              7
CPU MHz:               2891.857
CPU max MHz:           2900.0000
CPU min MHz:           1200.0000
BogoMIPS:              5799.73
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              20480K
NUMA node0 CPU(s):     0-7,16-23
NUMA node1 CPU(s):     8-15,24-31
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse s
se2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmper
f eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic popcnt tsc_d
eadline_timer aes xsave avx lahf_lm epb ssbd ibrs ibpb tpr_shadow vnmi flexpriority ept vpid xsaveopt dtherm arat pln pts md_c
lear
```

2.2.2.1 Configuration Principle

CPU configurations vary with the actual host. The configuration principle is as follows:

- When a host has multiple physical CPUs, you need to assign processors of only one CPU for vADS's use.
- For a core assigned to vADS, make sure that all of its processors are assigned to vADS.
- Processors of the same core should be assigned to vADS in sequence.
- The number of processors assigned to vADS ranges from 4 to 32.
- (Optional) Make sure that the CPU and NIC assigned to vADS belong to the same NUMA node to boost packet processing performance.

Note	It is not possible to allocate all CPUs to virtual machines, and some CPUs must be reserved for the host. Otherwise, the host will not be able to start.

2.2.2.2 Configuration Procedure

To configure CPU isolation settings, follow these steps:

Step 1 Query CPU information.

Run the following command to query CPU information, including **processor_id**, **physical_id**, **cord_id**, and **numa_node**.

```
cat /proc/cpuinfo
lscpu
```

Step 2 List the number of available NUMA nodes connected to the NIC used by vADS.

Run the following command to query the NUMA node information of the host. This step is required only when a host has more than one NUMA node.

```
numactl --hardware
#---------The command output is as follows:----------
```

```
available: 2 nodes (0-1)
#---Other irrelevant data is omitted.----
```

The above command output indicates that the host has two NUMA nodes.

If em4 serves as the NIC of vADS, run the following command:

```
cat /sys/class/net/em4/device/numa_node
#---------The command output is as follows:----------
1
```

The above command output indicates that em4 belongs to NUMA node 1. If there are multiple NICs, repeat this command several times to query their NUMA nodes.

If this command output is **-1**, see appendix C FAQs for more information.

Step 3 Configure the GRUB on the host for CPU isolation.

Determine which CPU to isolate according to the command output shown in Step 1 and modify GRUB settings.

a. Edit /etc/default/grub.

   Add **isolcpus=1-7,9-15** to the line beginning with GRUB_CMDLINE_LINUX_DEFAULT.

   Note that the isolcpus setting **1-7,9-15** here is just an example for reference and the actual setting depends on the actual host.

b. Run the **grub2-mkconfig -o $(find / -name grub.cfg | head -1)** command to modify the system GRUB.

c. Restart the host.

d. After restarting, you can use the command **cat /proc/cmdline** to confirm whether the configuration is correct.

```
cat /proc/cmdline
#---------The command output is as follows:----------
BOOT_IMAGE=/vmlinuz-3.10.0-957.el7.x86_64 root=/dev/mapper/centos-root ro
crashkernel=auto rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet
intel_iommu=on isolcpus=1-7,9-15
```

Step 4 Modify vADS's related CPU settings.

For an undefined vADS, directly edit **vads.xml**; for a defined vADS, run the **virsh edit vADS** command to modify the related CPU settings.

----End

2.2.2.3 Configuration Example

The following takes a server as an example to describe how to complete the CPU isolation configuration.

Step 1 Query CPU information.

a. Run the following code to obtain the CPU information file (.csv).

```
#!/bin/bash

PROCESSOR_ID_FILE="processor_id"
CORE_ID_FILE="core_id"
PHYSICAL_ID_FILE="physical_id"
```

```
CPU_INFO_FILE="cpu_info.csv"

cpu_data=`cat /proc/cpuinfo`
echo -n "$cpu_data" | grep "processor" | awk 'BEGIN{FS=":"}{print $2}' >
$PROCESSOR_ID_FILE
echo -n "$cpu_data" | grep "core id" | awk 'BEGIN{FS=":"}{print $2}' >
$CORE_ID_FILE
echo -n "$cpu_data" | grep "physical id" | awk 'BEGIN{FS=":"}{print $2}' >
$PHYSICAL_ID_FILE

echo "processor_id, physical_id, core_id" > $CPU_INFO_FILE
paste -d ',' processor_id physical_id core_id >> $CPU_INFO_FILE

rm -f $PROCESSOR_ID_FILE  $CORE_ID_FILE  $PHYSICAL_ID_FILE
```

Figure 2-9 shows the data included in the CSV file.

Figure 2-9 Obtained CSV data

	A	B	C
1	processor_id	physical_id	cord_id
2	0	0	0
3	1	1	0
4	2	0	1
5	3	1	1
6	4	0	2
7	5	1	2
8	6	0	3
9	7	1	3
10	8	0	4
11	9	1	4
12	10	0	5
13	11	1	5
14	12	0	8
15	13	1	8
16	14	0	9
17	15	1	9
18	16	0	10
19	17	1	10
20	18	0	11
21	19	1	11
22	20	0	12
23	21	1	12
24	22	0	13
25	23	1	13
26	24	0	0
27	25	1	0
28	26	0	1
29	27	1	1
30	28	0	2
31	29	1	2
32	30	0	3
33	31	1	3
34	32	0	4
35	33	1	4
36	34	0	5
37	35	1	5
38	36	0	8
39	37	1	8
40	38	0	9
41	39	1	9
42	40	0	10
43	41	1	10
44	42	0	11
45	43	1	11
46	44	0	12
47	45	1	12
48	46	0	13
49	47	1	13

b. Query the relationship between CPUs and NUMA nodes.

```
lscpu
#---------The command output is as follows:----------
#---Other irrelevant data is omitted.----
NUMA node0 CPU(s):
0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,32,34,36,38,40,42,44,46
```

```
NUMA node1 CPU(s):
1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35,37,39,41,43,45,47
#---Other irrelevant data is omitted.----
```

Step 2 List the number of available NUMA nodes of the NIC used by vADS.

```
cat /sys/class/net/em4/device/numa_node
#---------The command output is as follows:----------
1
```

vADS uses em4 as its NIC that belongs to NUMD node 1.

Step 3 Configure the GRUB on the host for CPU isolation.

a. Determine which physical CPU to isolate: CPUs in physical CPU 1 belong to NUMA node 1. Therefore, CPU 1 is selected here.

b. Determine which cores to isolate: All cores of physical CPU 1 are selected here.

Figure 2-10 Determining cores to be isolated

	A	B	C
1	processor_id	physical_id	cord_id
3	1	1	0
5	3	1	1
7	5	1	2
9	7	1	3
11	9	1	4
13	11	1	5
15	13	1	8
17	15	1	9
19	17	1	10
21	19	1	11
23	21	1	12
25	23	1	13
27	25	1	0
29	27	1	1
31	29	1	2
33	31	1	3
35	33	1	4
37	35	1	5
39	37	1	8
41	39	1	9
43	41	1	10
45	43	1	11
47	45	1	12
49	47	1	13

Edit **/etc/default/grub** as follows:

```
GRUB_CMDLINE_LINUX_DEFAULT="intel_iommu=on isolcpus=1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,35,37,39,41,43,45,47"
```

c. Run the **grub2-mkconfig -o $(find / -name grub.cfg | head -1)** command to modify the system GRUB.

d. Restart the host.

Step 4 Modify the vADS's related CPU settings.

a. Sort out data in the CSV file in ascending order of core_id, as shown in Figure 2-11. In the CSV file, the vCPU data is shown in the **processor_id** column.

Figure 2-11 Sorting out data in ascending order of core_id

1	processor_id	physical_id	cord_id
3	1	1	0
5	25	1	0
7	3	1	1
9	27	1	1
11	5	1	2
13	29	1	2
15	7	1	3
17	31	1	3
19	9	1	4
21	33	1	4
23	11	1	5
25	35	1	5
27	13	1	8
29	37	1	8
31	15	1	9
33	39	1	9
35	17	1	10
37	41	1	10
39	19	1	11
41	43	1	11
43	21	1	12
45	45	1	12
47	23	1	13
49	47	1	13

b. Modify vADS settings.

As the current vADS is already defined, run the **virsh edit vADS** command to edit its settings.

Caution	The number of vCPUs should be changed as required, and so should the number of cores and that of threads.

Figure 2-12 Running the "virsh edit vADS" command to edit vADS settings

```
<vcpu placement='static'>24</vcpu>
<cputune>
  <vcpupin vcpu='0' cpuset='1'/>
  <vcpupin vcpu='1' cpuset='25'/>
  <vcpupin vcpu='2' cpuset='3'/>
  <vcpupin vcpu='3' cpuset='27'/>
  <vcpupin vcpu='4' cpuset='5'/>
  <vcpupin vcpu='5' cpuset='29'/>
  <vcpupin vcpu='6' cpuset='7'/>
  <vcpupin vcpu='7' cpuset='31'/>
  <vcpupin vcpu='8' cpuset='9'/>
  <vcpupin vcpu='9' cpuset='33'/>
  <vcpupin vcpu='10' cpuset='11'/>
  <vcpupin vcpu='11' cpuset='35'/>
  <vcpupin vcpu='12' cpuset='13'/>
  <vcpupin vcpu='13' cpuset='37'/>
  <vcpupin vcpu='14' cpuset='15'/>
  <vcpupin vcpu='15' cpuset='39'/>
  <vcpupin vcpu='16' cpuset='17'/>
  <vcpupin vcpu='17' cpuset='41'/>
  <vcpupin vcpu='18' cpuset='19'/>
  <vcpupin vcpu='19' cpuset='43'/>
  <vcpupin vcpu='20' cpuset='21'/>
  <vcpupin vcpu='21' cpuset='45'/>
  <vcpupin vcpu='22' cpuset='23'/>
  <vcpupin vcpu='23' cpuset='47'/>
  <emulatorpin cpuset='0,12'/>
</cputune>
<os>
  <type arch='x86_64'>hvm</type>
  <boot dev='hd'/>
</os>
<features>
  <acpi/>
  <apic/>
</features>
<cpu mode='host-passthrough' check='none'>
  <topology sockets='1' cores='12' threads='2'/>
</cpu>
```

----End

2.2.3 Assigning NICs

vADS supports both passthrough NICs and virtual NICs. A passthrough NIC's performance is nearly as good as a physical NIC's. If a virtual NIC is used, the host needs to send packets to vADS. In this case, packet loss may occur in certain situations due to the limited packet processing capability of the host.

2.2.3.1 Passthrough NIC Assignment

To assign a passthrough NIC, follow these steps:

Step 1 Get the PCI address of the NIC used by vADS.

```
lshw -c network -businfo
```

Step 2 View devices in the IOMMU group.

```
find /sys/kernel/iommu_groups/ -type l
```

Step 3 Add a passthrough NIC for vADS.

a. Edit the vADS configuration file.

```
virsh edit vADS
```

b. Add a passthrough NIC as follows:

```
   <hostdev mode='subsystem' type='pci' managed='yes'>
     <source>
       <address domain='0x0000' bus='0x06' slot='0x00' function='0x0'/>
     </source>
   </hostdev>
```

Note	Each hostdev element specifies a NIC directly attached to the interface used by vADS. The query results in Step 1 show information about the NIC, including the domain name, bus, slot, and function. You can add hostdev elements according to the number of NICs to be used. In the hostdev element, **managed='yes'** indicates that the passthrough NIC is detached from the host when vADS is started, but back to the host when vADS is shut down.

----End

2.2.3.2 Example

The following uses X710 as an example to illustrate how to add four passthrough NICs:

Step 1 Get the PCI addresses of NICs used by vADS.

```
lshw -c network -businfo
#-----The command output is as follows:--------
Bus info         Device         Class        Description
=============================================================
#---Other irrelevant data is omitted.----
pci@0000:06:00.0  p5p1           network      Ethernet Controller X710 for 10GbE
SFP+
pci@0000:06:00.1  p5p2           network      Ethernet Controller X710 for 10GbE
SFP+
pci@0000:06:00.2  p5p3           network      Ethernet Controller X710 for 10GbE
SFP+
pci@0000:06:00.3  p5p4           network      Ethernet Controller X710 for 10GbE
SFP+
#---Other irrelevant data is omitted----
```

Assume that you need to add four passthrough NICs, as listed in Table 2-3.

Table 2-3 PCI addresses of four NICs

Device	**PCI**				
	PCI Address	**Domain**	**Bus**	**Slot**	**Function**
p5p1	0000:06:00.0	0000	06	00	0
p5p2	0000:06:00.1	0000	06	01	1
p5p3	0000:06:00.2	0000	06	02	2
p5p4	0000:06:00.3	0000	06	03	3

Step 2 View devices in the IOMMU group.

```
find /sys/kernel/iommu_groups/ -type l
#-----The command output is as follows:--------
# Focus only on the IOMMU groups that contain devices for passthrough assignment:
/sys/kernel/iommu_groups/18/devices/0000:06:00.0
/sys/kernel/iommu_groups/19/devices/0000:06:00.1
/sys/kernel/iommu_groups/20/devices/0000:06:00.2
/sys/kernel/iommu_groups/21/devices/0000:06:00.3
#---Other irrelevant data is omitted.----
```

An IOMMU group may contain multiple devices. For example, IOMMU group 15 contains em1 and em2 NICs.

```
/sys/kernel/iommu_groups/15/devices/0000:01:00.0
/sys/kernel/iommu_groups/15/devices/0000:01:00.1
```

Note	The smallest unit for passthrough assignment is not a specific device in the IOMMU group, but the entire group. That is to say, the passthrough assignment is done for all devices included in an IOMMU group. Therefore, if em1 is assigned to vADC, em2 should also be assigned to it. Here, devices included in the IOMMU group to which the four X710 NICs belong are assigned to the same vADS.

Step 3 Modify the configuration file of vADS:

```
virsh edit vADS
```

Add four passthrough NICs for vADS:

```
    <hostdev mode='subsystem' type='pci' managed='yes'>
      <source>
        <address domain='0x0000' bus='0x06' slot='0x00' function='0x0'/>
      </source>
    </hostdev>
    <hostdev mode='subsystem' type='pci' managed='yes'>
      <source>
        <address domain='0x0000' bus='0x06' slot='0x00' function='0x1'/>
```

```
    </source>
  </hostdev>
  <hostdev mode='subsystem' type='pci' managed='yes'>
    <source>
      <address domain='0x0000' bus='0x06' slot='0x00' function='0x2'/>
    </source>
  </hostdev>
  <hostdev mode='subsystem' type='pci' managed='yes'>
    <source>
      <address domain='0x0000' bus='0x06' slot='0x00' function='0x3'/>
    </source>
  </hostdev>
```

----End

2.2.3.3 Virtual NIC Assignment

If the current NICs are not supported by vADS or cannot be configured as passthrough NICs, you can assign a virtual NIC to vADS. For the sake of more efficient packet forwarding, the NIC assigned to vADS cannot be used by the host.

To assign a virtual NIC to vADS, follow these steps:

Step 1 Modify the configuration file of vADS:

```
virsh edit vADS
```

Step 2 Add a virtual NIC.

Note that **em4** shown in the following script should be replaced by the actual name of the NIC assigned to vADS.

```
  <interface type='direct' trustGuestRxFilters='yes'>
    <source dev='em4' mode='passthrough'/>
    <model type='virtio'/>
    <driver name='vhost' queues='8'/>
  </interface>
```

Note	An interface element corresponds to a virtual interface to be assigned to vADS. Therefore, you need to add interface elements according to the number of NICs to be used by vADS.

----End

2.2.4 Enabling vADS

To enable vADS, follow these steps:

Step 1 Start vADS.

```
virsh start vADS
```

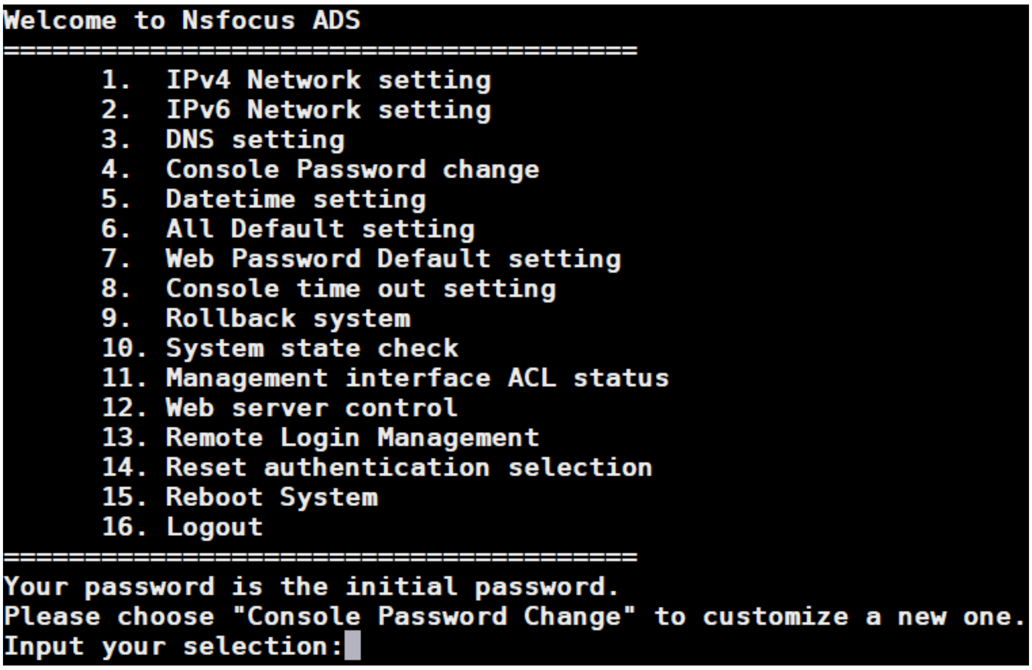
Step 2 Wait for several minutes and then set the IP address of the management interface, subnet mask, and gateway of vADS.

Step 3 Run the following command on the host to connect to the console of vADS.

```
virsh console vADS --force
```

Step 4 After login to vADS as user **admin**, complete network configurations and DNS configurations on the console-based manager as indicated in the *NSFOCUS ADS User Guide*. Figure 2-13 shows the main window of the console-based manager.

Figure 2-13 Console-based manager



Note	If cloud-based authentication is required, you must complete DNS configurations; otherwise, the domain name of the cloud authentication center cannot resolved.

----End

3 Deployment on VMware ESXi

This chapter describes how to import and configure vADS on VMware ESXi.

This chapter covers the following topics:

Topic	Description
Preparations	Describes preparations to be made for installing vADS on ESXi.
Installation Procedure	Describes how to install vADS on ESXi.

3.1 Preparations

Before installing vADS locally, you must make preparations listed in Table 3-1.

Table 3-1 List of items to be prepared for installing vADS locally

Item		Description
Host	IP address	IP address of the host that can properly connect to the network
	Account	Account with privileges of a system administrator
	Network interface	At least one 1000M interface available
	Operating system	VMware ESxi 6.7 or a later version
vADS	vADS image file	**vads.ova**
	IP address	IP address of the management interface of vADS

Installing and Configuring the Host System

For details on installing and configuring the host system, see the VMware vSphere Documentation at https://docs.vmware.com/en/VMware-vSphere/index.html.

For details on enabling BIOS virtualization, see section 2.1.4 Virtualization.

3.2 Installation Procedure

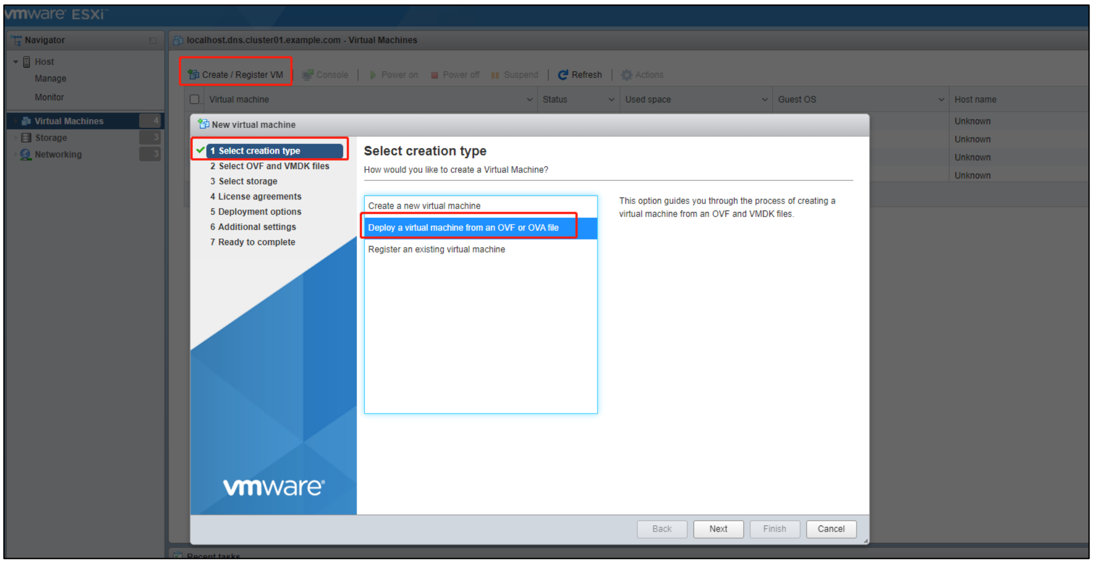
3.2.1 Importing the vADS Image

Before importing the vADS image, you need to obtain the image file **vads.ova**.

To import the vADS image, follow these steps:

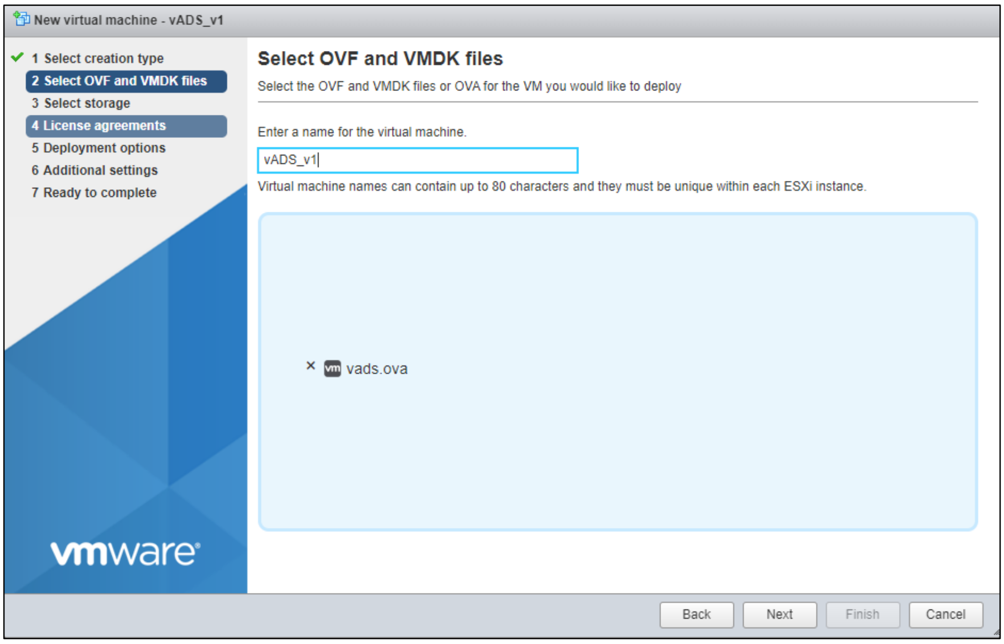
Step 1 Choose to deploy a virtual machine from an OVA file.

Figure 3-1 Selecting the deployment mode



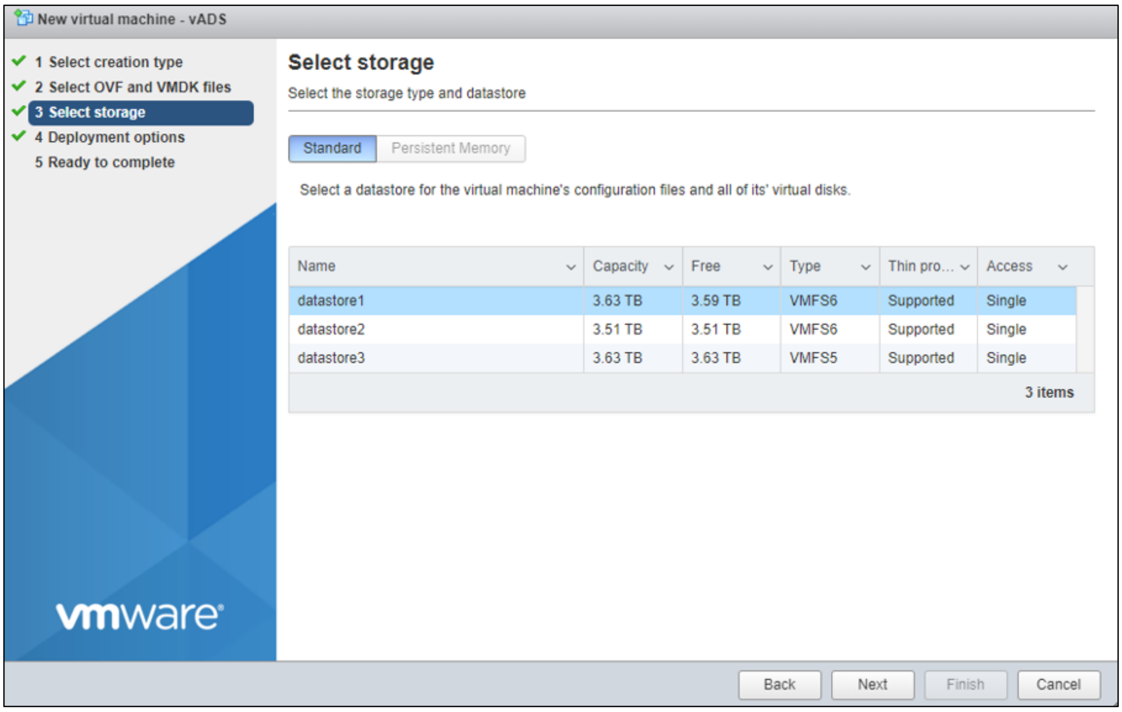
Step 2 Select the image file and enter a name for the virtual machine.

Figure 3-2 Setting the VM name and selecting the image file



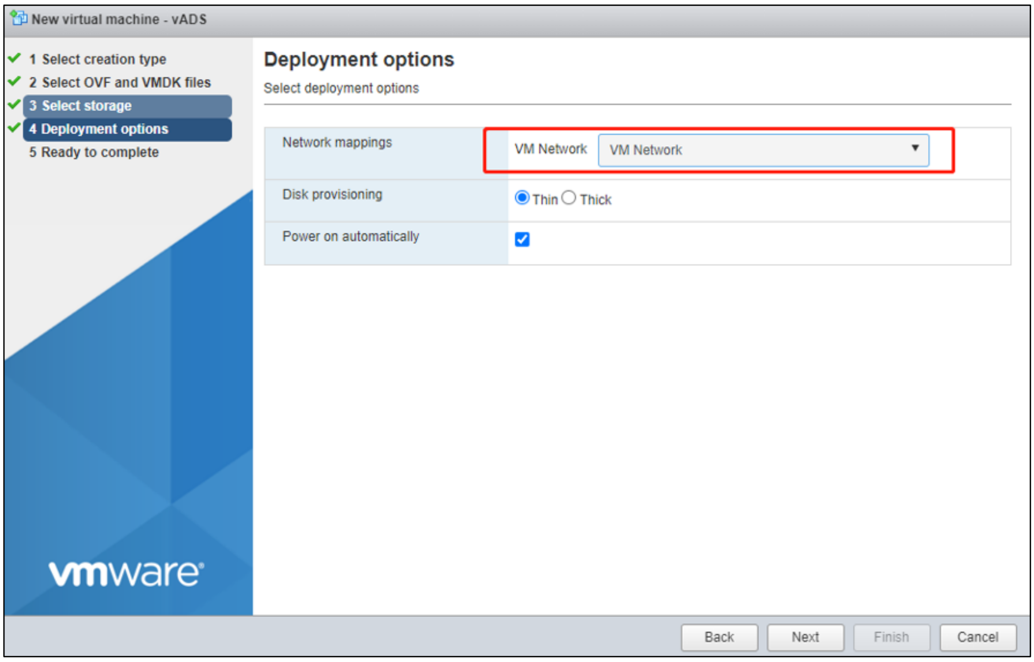
Step 3 Select the storage location for disk files.

Figure 3-3 Selecting a datastore



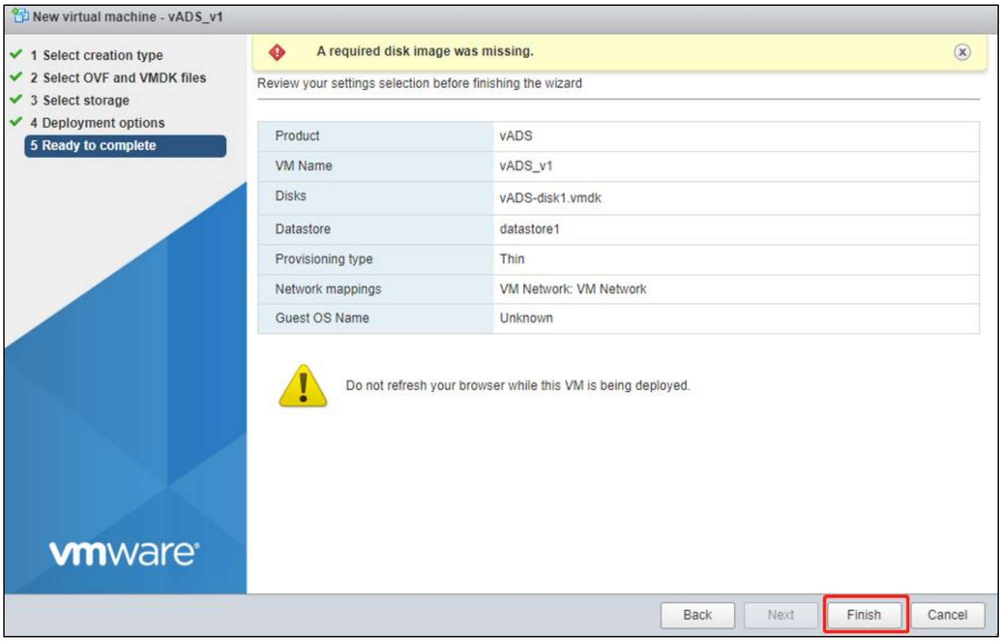
Step 4 Select **VM Network** for **Network mappings**, and keep default settings for other deployment options.

Figure 3-4 Deployment options



The preparations for deployment are completed.

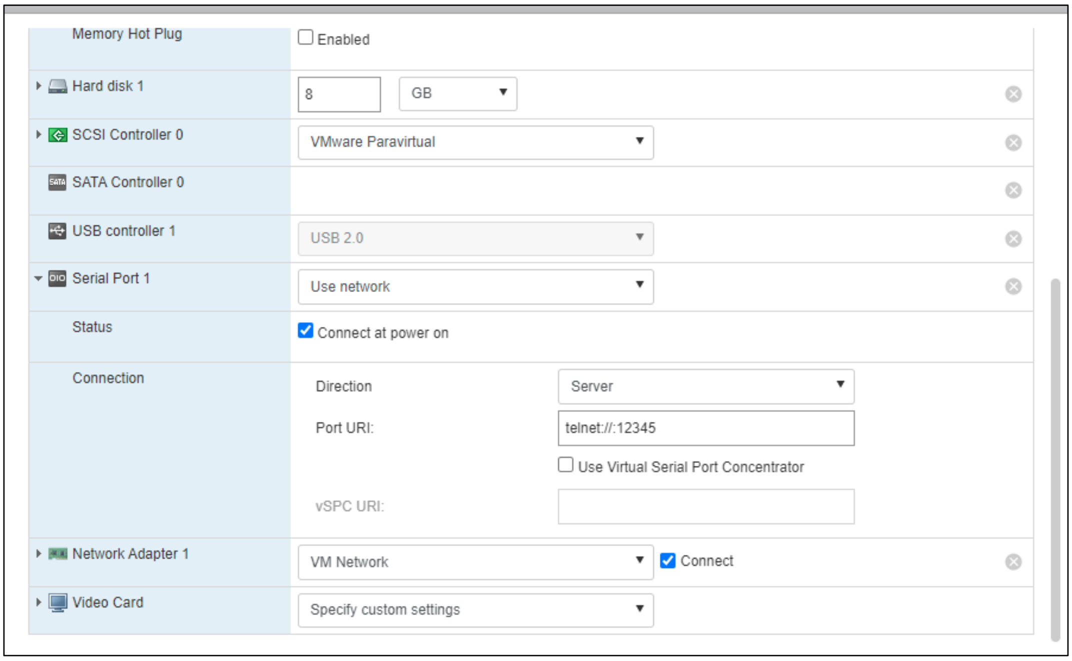
Figure 3-5 Ready to deploy



Step 5 Click **Finish** to upload the OVA file.

Step 6 Configure a virtual serial port.

Figure 3-6 Configuring a serial port



For details on how to configure parallel ports and serial ports, see the *vSphere Virtual Machine Administration* file of the VMware vSphere Documentation.

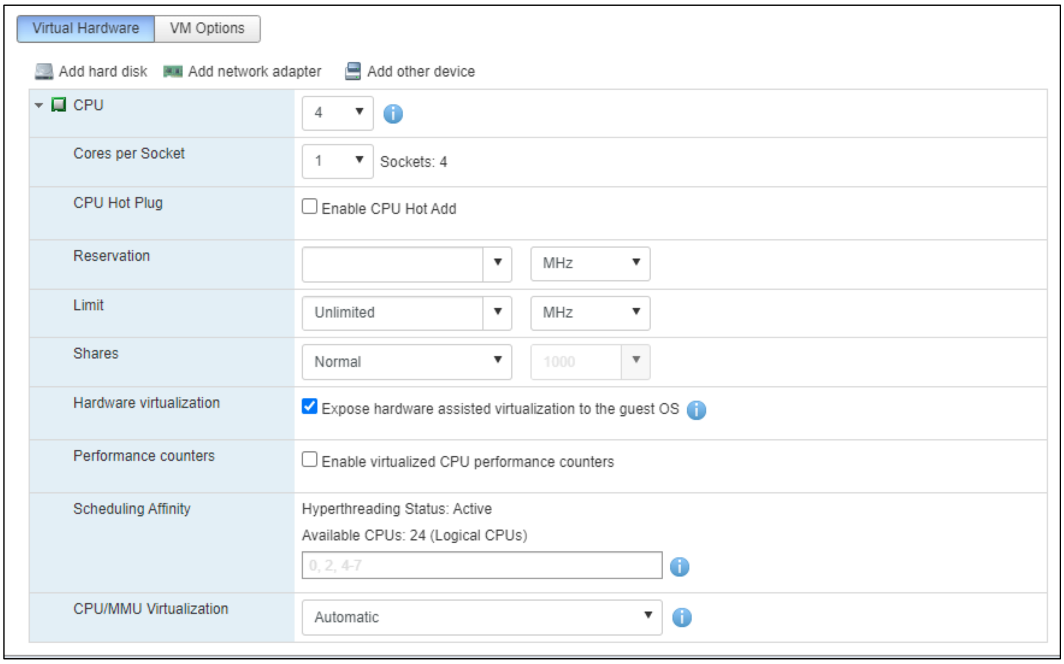
Note	The NIC type of the management interface is set to E1000, which cannot be modified.

----End

3.2.2 Allocating Resources

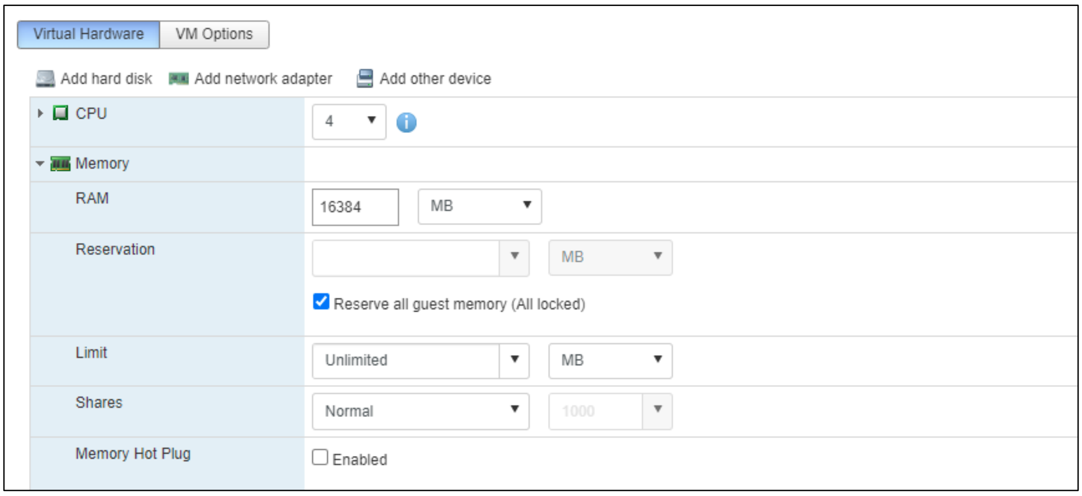
Adjust the CPU and memory resources to meet the requirements stated in Table 1-4. For better performance, enable the hardware virtualization as shown in Figure 3-7.

Figure 3-7 Allocating CPU resources



Select the **Reserve all guest memory (All locked)** check box and select **Unlimited** for the memory limit.

Figure 3-8 Allocating memory resources



3.2.3 Assigning NICs

vADS supports both passthrough NICs and virtual NICs. A passthrough NIC's performance is nearly as good as a physical NIC's. If a virtual NIC is used, the host needs to send packets to vADS. In this case, packet loss may occur in certain situations due to the limited packet processing capability of the host.

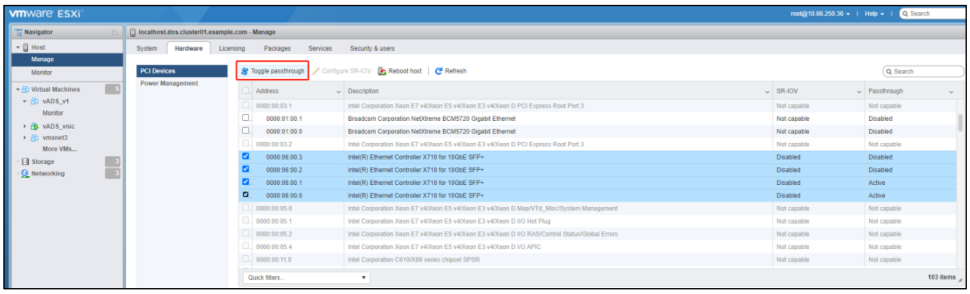
3.2.3.1 Passthrough NIC Assignment

For deployment details, see the "Direct Path I/O" section of the *vSphere Networking* file.

To assign a passthrough NIC, follow these steps:

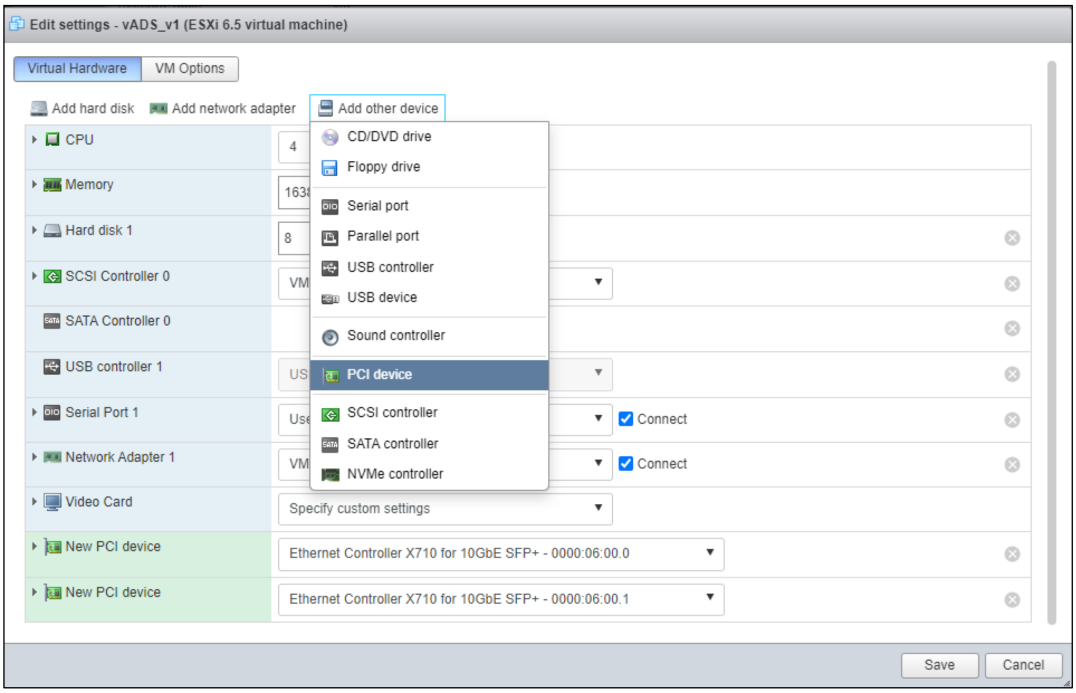
Step 1 Enable the passthrough features for networking devices of the host.

Figure 3-9 Enabling passthrough



Step 2 Add PCI devices to vADS.

Figure 3-10 Adding a PCI device



Note	
	• To assign a passthrough NIC, the memory limit must be set to **Unlimited**. • Snapshots are not supported for vADS with a passthrough NIC.

----End

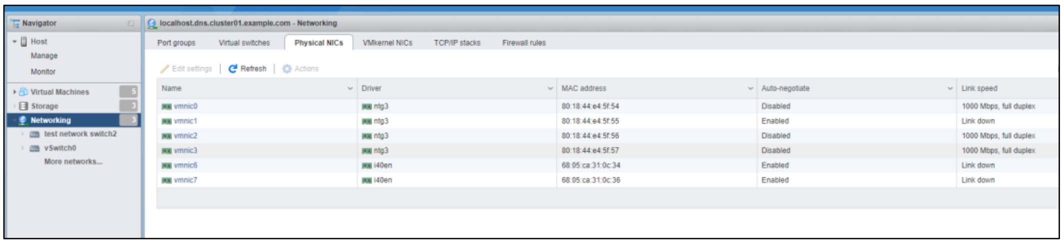
3.2.3.2 **Virtual NIC Assignment**

If the current NICs are not supported by vADS or cannot be configured as passthrough NICs, you can assign a virtual NIC to vADS. For the sake of more efficient packet forwarding, the NIC assigned to vADS cannot be used by the host. When a virtual NIC is used, up to 4 CPU cores can be assigned to vADS.

To create a bridge on a physical NIC, see the VMware documentation. The specific procedure is as follows:

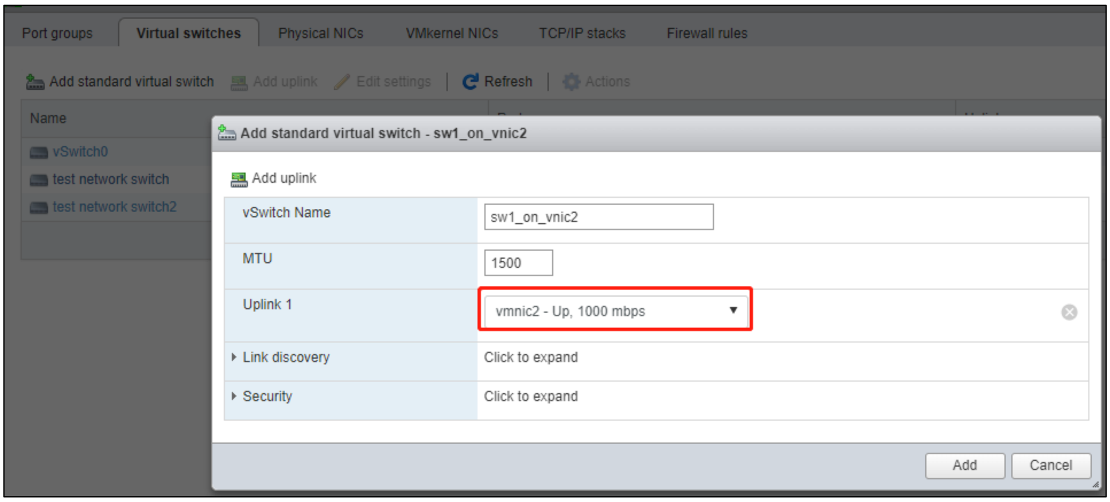
Step 1 Determine the physical NIC to be used.

Figure 3-11 Selecting a physical NIC



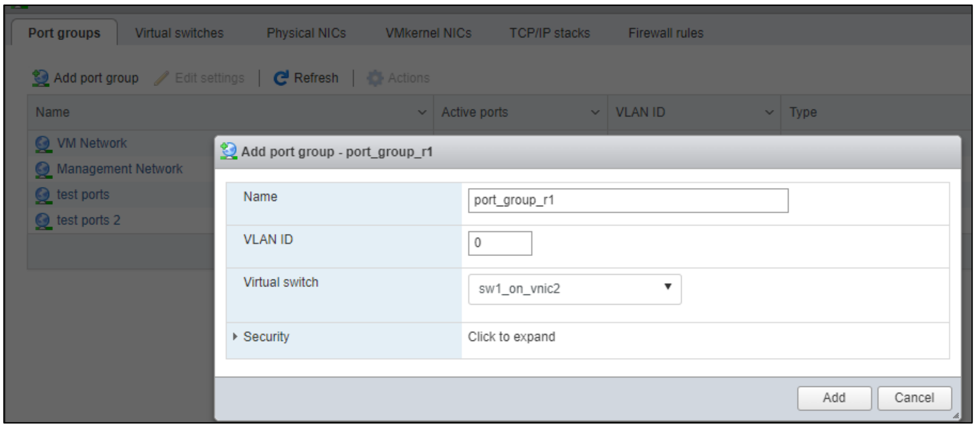
Step 2 Creating a standard virtual switch.

Figure 3-12 Adding a virtual switch



Step 3 Create a port group.

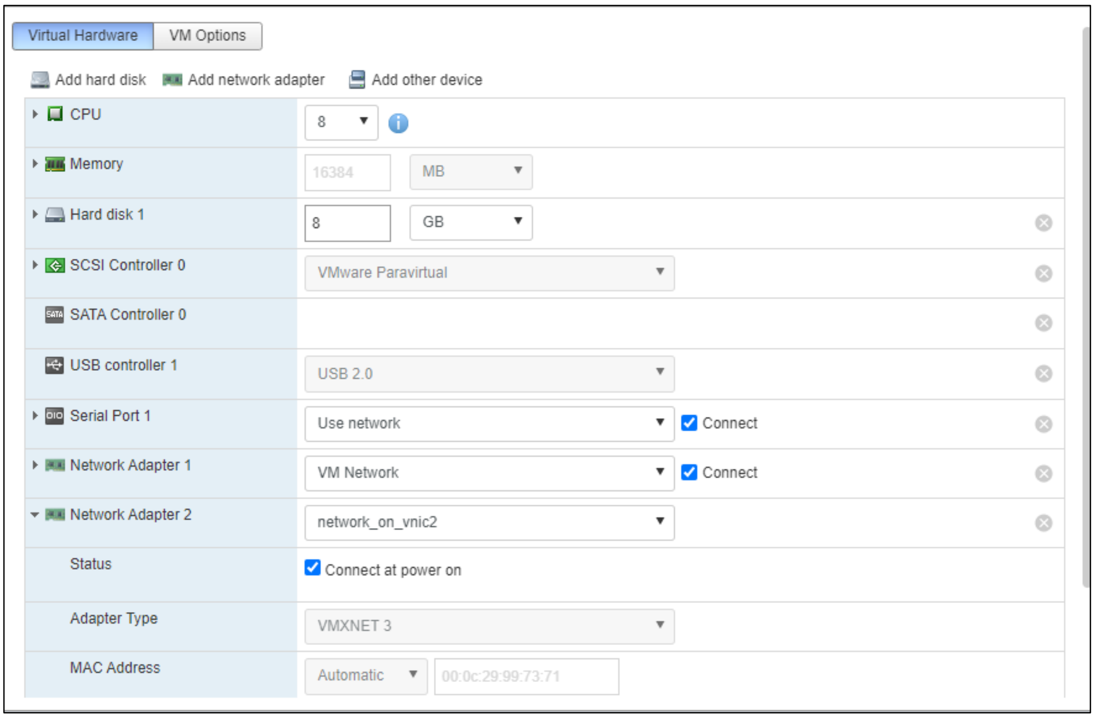
Figure 3-13 Adding a port group



Step 4 Add a virtual NIC.

Click **Add network adapter** to create a virtual NIC and assign it to vADS. The NIC type is set to **VMXNET 3**.

Figure 3-14 Configuring a virtual NIC



----End

3.2.4 Enabling vADS

To enable vADS, follow these steps:

Step 1 Start vADS on VMware.

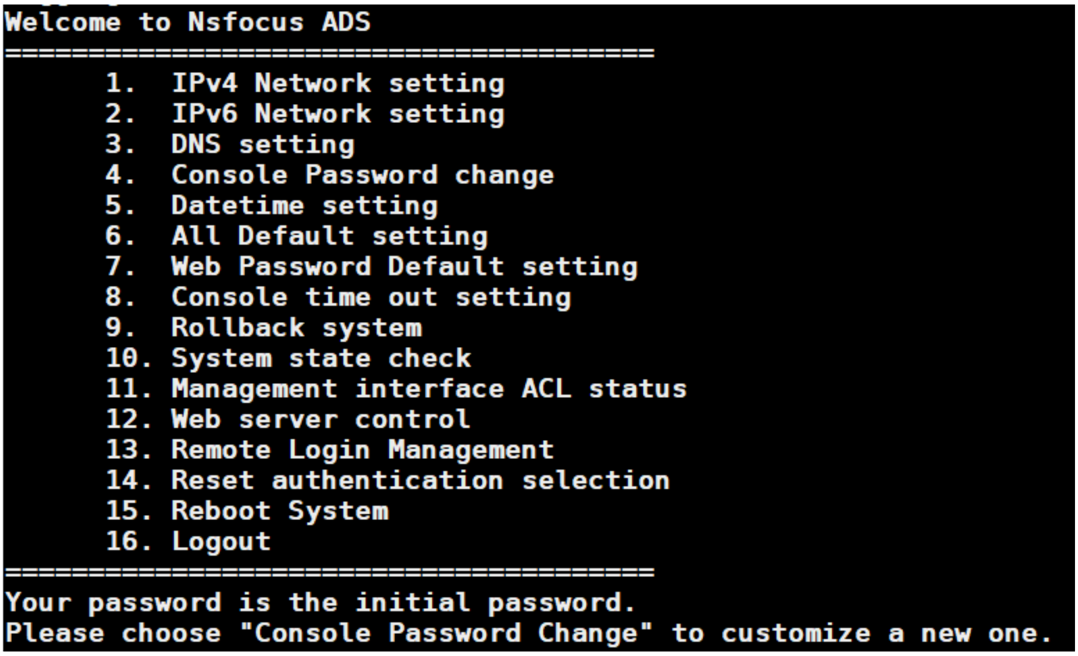
Step 2 Wait for several minutes and then set the IP address of the management interface, subnet mask, and gateway of vADS on the console.

Run the following command on the host to connect to the console of vADS.

```
telnet HOST_IP PORT
#Replace HOST_IP with the management IP address of the host, and replace PORT with
the actual port number configured in the virtual serial port, for example
telnet 10.66.250.36 12345
```

Step 3 After login to vADS as user **admin**, complete network configurations and DNS configurations on the console-based manager as indicated in the *NSFOCUS ADS User Guide*. Figure 3-15 shows the main window of the console-based manager.

Figure 3-15 Console-based manager



Note	If cloud-based authentication is required, you must complete DNS configurations; otherwise, the domain name of the cloud authentication center cannot resolved.

----End

A Default Parameters

A.1 Default Parameters of the Management Interface

Management IP Address	192.168.1.100
Subnet Mask	255.255.255.0
Default Gateway	192.168.1.1
Reserved IP Segment for Internal Communication	172.16.1.0/24

A.2 Default Accounts

Account Type	User Name	Password
Web Administrator	admin	nsfocus
Console Administrator	admin	nsfocus

B Terminology

Term	Description
Host	Physical machine or server that provides the virtual platform (KVM or VMware ESXi).
Guest	Virtual machine hosted on the virtual platform. In this document, vADS is a guest on KVM or VMware ESXi.

C FAQs

C.1 Why Can't a NIC Supported by vADS on KVM Be Configured to Be a Passthrough NIC ?

Step 1 Check whether virtualization and IOMMU are enabled in the system.

Step 2 Get the PCI address of the NIC to be directly attached to vADS.

Assume that the NIC is ens69f0 and you can get its PCI address 0000:07:00.0 by running the following command.

```
lshw -c network -businfo
```

Step 3 View devices in the IOMMU group.

Run the following command to show details of the devices (note that the PCI address should be replaced as required).

```
virsh nodedev-dumpxml pci_0000_07_00_0
```

- The IOMMU group contains non-endpoint devices.

If the following execution result is displayed for the **virsh nodedev-dumpxml pci** command, you can see that IOMMU group contains a series of devices.

```
    <iommuGroup number='1'>
      <address domain='0x0000' bus='0x00' slot='0x01' function='0x0'/>
      <address domain='0x0000' bus='0x01' slot='0x00' function='0x0'/>
      <address domain='0x0000' bus='0x02' slot='0x04' function='0x0'/>
      <address domain='0x0000' bus='0x02' slot='0x05' function='0x0'/>
      <address domain='0x0000' bus='0x02' slot='0x08' function='0x0'/>
      <address domain='0x0000' bus='0x02' slot='0x09' function='0x0'/>
      <address domain='0x0000' bus='0x03' slot='0x00' function='0x0'/>
      <address domain='0x0000' bus='0x04' slot='0x01' function='0x0'/>
      <address domain='0x0000' bus='0x04' slot='0x03' function='0x0'/>
      <address domain='0x0000' bus='0x05' slot='0x00' function='0x0'/>
      <address domain='0x0000' bus='0x05' slot='0x00' function='0x1'/>
      <address domain='0x0000' bus='0x05' slot='0x00' function='0x2'/>
      <address domain='0x0000' bus='0x05' slot='0x00' function='0x3'/>
      <address domain='0x0000' bus='0x06' slot='0x00' function='0x0'/>
      <address domain='0x0000' bus='0x06' slot='0x00' function='0x1'/>
      <address domain='0x0000' bus='0x06' slot='0x00' function='0x2'/>
      <address domain='0x0000' bus='0x06' slot='0x00' function='0x3'/>
      <address domain='0x0000' bus='0x07' slot='0x00' function='0x0'/>
      <address domain='0x0000' bus='0x07' slot='0x00' function='0x1'/>
```

```
</iommuGroup>
```

Run the **lspci** command to query details of the node. It turns out that 00:01.0 is a non-endpoint device. KVM does not support the passthrough assignment of non-endpoint devices to vADS.

```
lspci -vv -s 00:01.0
00:01.0 PCI bridge: Intel Corporation Xeon E3-1200/2nd Generation Core Processor Family PCI Express Root Port (rev 09) (prog-if 00 [Normal decode])
```

As KVM can assign only one IOMMU group to vADS and does not support assignment of non-endpoint devices to vADS, the host does not support the passthrough assignment of this NIC to vADS.

- The IOMMU group contains multiple NICs.

If the following execution result is displayed for the above **virsh nodedev-dumpxml pci** command, you can see that the IOMMU group contains two NICs.

```
   <iommuGroup number='1'>
     <address domain='0x0000' bus='0x07' slot='0x00' function='0x0'/>
     <address domain='0x0000' bus='0x07' slot='0x00' function='0x1'/>
</iommuGroup>
```

Here, both devices in the IOMMU group are NICs. You can assign the two NICs in either of the following ways:

a. Assign both NICs in the IOMMU group to vADS.
b. Assign one NIC to vADS and run the following command to detach the other from the host. Note that the PCI address used in this command is the PCI address (0000:07:00.1) of the device not assigned to vADS.

```
$ lspci -n -s 07:00.1
07:00.1 0200: 8086:10fb (rev 01)
```

Edit the GRUB configuration file by appending the address (0200: 8086:10fb) of the device assigned to vADS to the value of **pci-stub.ids**. Then reboot the host.

```
$ vim /etc/default/grub
        GRUB_CMDLINE_LINUX="rd.lvm.lv=fedora-server/root rd.lvm.lv=fedora-server/swap rhgb quiet intel_iommu=on pci-stub.ids=8086:0152,10de:1401, 0200: 8086:10fb”

$ grub2-mkconfig -o $(find / -name grub.cfg | head -1)

$ reboot
```

----End

C.2 Why Can't I Log In to vADS's Web-based Manager After I Start vADS Following the Process of Deploying vADS on KVM?

On the host, run the **virsh list** command to check whether vADS is in the running state. If no, first start vADS; if yes, run the **virsh console vADS --force** command to log in to the console-based manager as user **admin** and then complete network configurations and check whether those configurations take effect.

C.3 What Are Common Commands for Virtualization on KVM?

Table C-1 lists common commands for virtualization on KVM.

Table C-1 Common commands for virtualization on KVM

Command	Description
virsh autostart vADS	Sets vADS as an automatic startup item.
virsh console vADS	Logs in to the console-based manager.
virsh destroy vADS	Shuts down vADS.
virsh list	Checks the vADS operating status.

C.4 Why Does Serious Packet Loss Occur When a Virtual NIC Is Used for vADS on KVM?

If a virtual NIC is used, packets need to be processed by the host's kernel. The packet processing capability is strongly associated with the host's CPU and NIC performance and configurations. Therefore, you can optimize the CPU and NIC configurations to improve the host's packet processing capability (small packets of about 200 MB). For more efficient packet processing, you are advised to use a passthrough NIC for vADS.

When a serious packet loss issue occurs, do as follows to increase the packet processing capacity:

Step 1 Check how many CPUs are used by the host.

At least four CPUs should be used by the host.

Step 2 Check the number of physical NIC queues.

Eight physical NIC queues are recommended.

Here, the physical NIC named em4 is used as an example. Run the following commands to configure and query physical NIC queues.

```
# Set the number of em4 queues to 8:
ethtool -L em4 combined 8

# Query the number of em4 queues:
ethtool -l em4
--------The query result is as follows:---------
Channel parameters for em4:
Pre-set maximums:
RX:           0
TX:           0
Other:         1
Combined:      8
Current hardware settings:
RX:           0
TX:           0
Other:         1
Combined:      8
```

Step 3 Set the size of the hardware cache queue.

It is recommended that the size of the hardware cache queue be set to the maximum value supported by the NIC.

Here, the physical NIC named em4 is used as an example. Run the following commands to configure and query the hardware cache queue of the em4.

```
# Set the size of em4's hardware cache queue to 4096:
ethtool -G em4 rx 4096
ethtool -G em4 tx 4096

# Query the size of the hardware cache queue of em4:
ethtool -g em4
--------The query result is as follows:---------
Ring parameters for em4:
Pre-set maximums:
RX:             4096
RX Mini:          0
RX Jumbo:         0
TX:             4096
Current hardware settings:
RX:             4096
RX Mini:          0
RX Jumbo:         0
TX:             4096
```

Step 4 Save the ethtool settings permanently in the network device.

The preceding ethtool settings will be missing upon the host reboot. It is recommended that ethtool settings be permanently saved in the network device. For example, you can add the **ethtool** command to **/etc/rc.d/rc.local** for persistency.

----End

C.5 Why Do Query Results Show That the Number of NUMA Nodes Is -1 on KVM?

First, check the number of NUMA nodes in the host. If the number is 1, there is only one NUMA node, i.e., NUMA node0. All NICs belong to this NUMA node.

If there are two NUMA nodes, the NICs with the bus of the PCI address being 8* (such as 86:00:01) generally belong to NUMA node 1, while NICs with other types of PCI address belong to NUMA node 2.

In other scenarios, it is impossible to differentiate NUMA nodes to which NICs belong. You can select any NUMA node and isolate CPUs in it.