NASA HW3

b09902004 郭懷元

Network Administration

1.

1.

> References:
>
> https://www.netadmin.com.tw/netadmin/zh-tw/technology/FFD0629A85794E9BAC2A9173D5B96EDC
>
> https://learningnetwork.cisco.com/s/question/0D53i00000Kt6oFCAR/what-happens-when-an-8021q-trunk-port-receives-a-tagged-frame-with-vlan-id-the-same-as-the-native-vlan

When the frame is sent to `Gi1/0/3`, because this interface is set to `access` mode, the frame will be tagged to be `vlan 307`. However, when it's sent to either `Gi1/0/4` or `Gi1/0/5`, because these two interface are set to `trunk` mode, no tags would be added, despite the fact that they have different native vlan.

2.

> References:
>
> https://www.netadmin.com.tw/netadmin/zh-tw/technology/FFD0629A85794E9BAC2A9173D5B96EDC
>
> https://learningnetwork.cisco.com/s/question/0D53i00000Kt6oFCAR/what-happens-when-an-8021q-trunk-port-receives-a-tagged-frame-with-vlan-id-the-same-as-the-native-vlan
>
> https://community.cisco.com/t5/switching/what-happens-when-switch-trunk-port-connected-to-pc/td-p/3731202

Because `Gi1/0/1` is a trunk port and the destination is in `vlan 424`, since `vlan 424` is not the native vlan of `Gi1/0/1`, the header remains the same after going through `Gi1/0/1`. When the packets goes through `Gi1/0/2`, because it's an access port, the header is stripped.

3.

> References:
>
> https://www.jannet.hk/zh-Hant/post/virtual-lan-vlan-attack/
>
> https://en.wikipedia.org/wiki/VLAN_hopping

`switchport trunk native` can be used against "double tagging", which is a attack technique that can allow an attacker to gain access to a vlan he doesn't belong to. Double tagging exploits the fact that trunk ports will strip down vlan tags that belong to its native vlan, thus the inner tag made by the attacker can be shown to the switch on the other side. By changing the native vlan to an unused one, all frames will preserve its tag, and the inner tag will not be shown.

2.

1.

> References:
>
> https://en.wikipedia.org/wiki/Link_aggregation#Same_link_speed

No. According to the IEEE standard, link aggregation requires all ports to have the same speed. In this case the maxmium bandwidth would be 2*100Mbps.

2.

> References:
>
> Lab 4 slides

`channel-group mode` is preferred to be set to `active` rather than `passive`, because link aggregation would work only if at least one side is set to `active`.

Commands to fix:

```
Switch> ena
Switch# conf t
Switch(config)# int range Gi1/0/1-2
CiscoLab(config-if-range)# channel-group 1 mode active
```

3.

1.

> References:
>
> https://documentation.meraki.com/MS/Layer_3_Switching/Layer_3_vs_Layer_2_Switching
>
> http://www.fiber-optic-equipment.com/layer-3-switch-vs-router-choose.html

A L2 switch uses MAC address to forward frames to other machines, and it can only work within a LAN. A L3 switch is like a L2 switch that can do routing.

Both a router and a L3 switch can route packets. But L3 switches are designed with the purpose of handling traffics between VLAN, unlike routers are for WAN. Therefore L3 switches doesn't support many protocol that routers support, like NAT. L3 switches also tend to be cheaper than routers.

2.

> References:
>
> None

An ARP table shows a IP address's corresponding MAC address. A MAC address table shows the MAC address of the device connected to a port.

3.

> References:
>
> https://www.sikich.com/insight/how-to-find-which-switch-port-a-device-is-plugged-into-via-ip-address/

1. Connect to the core switch.
2. `ping 140.112.30.250` to get the ARP table and MAC address table updated.
3. Type `sh ip arp 140.112.30.250`, we should get a MAC address here.
4. Type `sh mac address-table address <mac address>`, we should get a port.
5. Since the user isn't directly connected to core switch, use `sh cdp neighbor` and we know which switch is connected to the port we get in the previous step.
6. Connect to the switch we found in the previous step.
7. Repeat step 4, and use `sh cdp neighbor` to check if the port is connected to another switch.
8. If it is, we go back to step 6. If not, we find the user.

4.

1.

> References:
>
> https://en.wikipedia.org/wiki/Serial_communication

Serial communication is a means of sending bits, and is contrast to "parallel communication". For example, if the data we are sending is `1011`, in serial communication we would send `1`, `0`, `1`, `1` one at a time. While in parallel communication, we would have 4 channels and each bit it sent in an individual channel.

2.

The upside is that it's almost plug-and-play, very few things need to be configured in order to access the device.

The downside is that you can't remotely control and configure the device.

3.

> References:
>
> https://www.jannet.hk/zh-Hant/post/console-cable/

Assuming the switch hasn't been configured and current settings are unknown. One way is to use normal ethernet cable to DIY a Rollover cable, then use a RJ45 to DB9 (female) adapter.

4.

> References:
>
> https://coolking1206.pixnet.net/blog/post/57616767
>
> https://www.cisco.com/c/en/us/support/docs/smb/switches/cisco-350x-series-stackable-managed-switches/smb5252-what-is-stacking.html

Compared to port trunk, stacking can tolerant not only broken cables but also broken switches.

In this case we have 2 switches to stack, so we need two in total, one cable from `switch 1` to `switch 2`, and the other from `switch 2` to `switch 1`.

5.

> References:
>
> Lab 4 slides
>
> https://smallbusiness.chron.com/disable-dns-lookup-cisco-58863.html
>
> http://www.james-tw.com/cisco/cisco-she-ding-yuan-duan-lian-xian-telnet-ssh

1.

```
Switch> ena
Switch# conf t
Switch(config)# hostname CiscoLab
CiscoLab(config)# exit
CiscoLab# copy running-config startup-config
```

2.

```
CiscoLab# conf t
CiscoLab(config)# no ip domain-lookup
CiscoLab(config)# exit
CiscoLab# copy running-config startup-config
```

3.

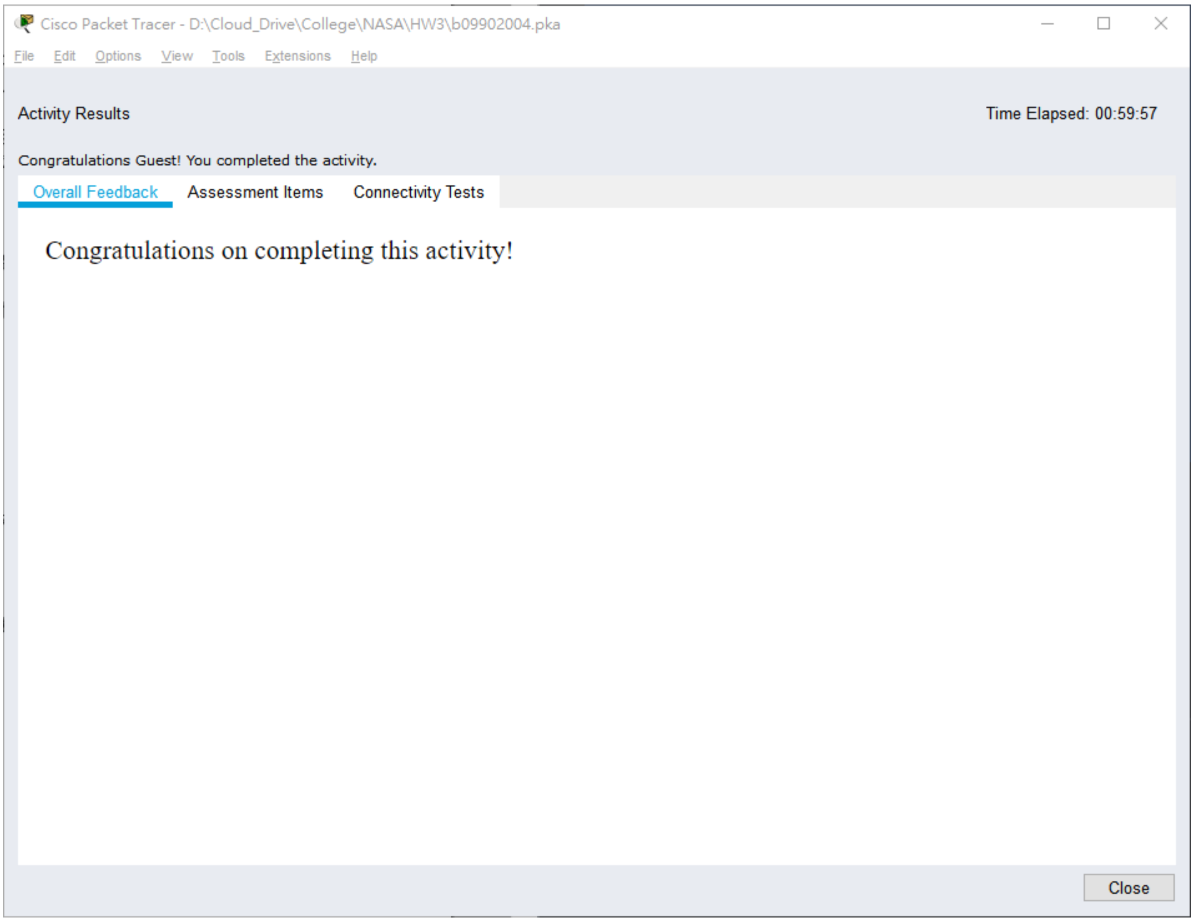
```
CiscoLab(config)# enable password CISCO
CiscoLab(config)# service password-encryption
CiscoLab(config)# exit
CiscoLab# copy running-config startup-config
```

4 ~ 6.

```
CiscoLab# conf t
CiscoLab(config)# int range fa0/1-2
CiscoLab(config-if-range)# switchport mode access
CiscoLab(config-if-range)# switchport access vlan 10
CiscoLab(config-if-range)# exit
CiscoLab(config)# int range fa0/3-4
CiscoLab(config-if-range)# switchport mode access
CiscoLab(config-if-range)# switchport access vlan 20
CiscoLab(config-if-range)# exit
CiscoLab(config)# int fa0/5
CiscoLab(config-if)# switchport mode access
CiscoLab(config-if)# switchport access vlan 99
CiscoLab(config-if)# exit
CiscoLab(config)# exit
CiscoLab# copy running-config startup-config
```

7.

```
CiscoLab# conf t
CiscoLab(config)# int vlan 99
CiscoLab(config-if)# ip address 192.168.99.1 255.255.255.0
CiscoLab(config-if)# no shutdown
CiscoLab(config)# line vty 0 4
CiscoLab(config-line)# password cisco
CiscoLab(config-line)# login
CiscoLab(config-line)# exit
CiscoLab(config)# service password-encryption
CiscoLab(config)# exit
CiscoLab # copy running-config startup-config
```



Cisco Packet Tracer - D:\Cloud_Drive\College\NASA\HW3\b09902004.pka

Activity Results

Time Elapsed: 00:59:57

Congratulations Guest! You completed the activity.

Congratulations on completing this activity!

System Administration

1.

1 ~ 3.

> References:
>
> http://linux.vbird.org/linux_basic/0157installcentos7.php

4.

In VM:

```
ip add
# to get the IP address of the VM
# in my case the IP address is 192.168.244.128
```

In your machine:

```
ping 192.168.244.128
# ping is a convenient tool to check communication status
# just type `ping <ip address>`
```

5.

> References:
>
> https://blog.gtwang.org/linux/centos-7-install-kvm-qemu-virtual-machine-tutorial/

```
sudo yum install virt-install qemu-kvm libvirt
# the default package manager of centos is yum
# so we use `yum install <package name>` to install packages
```

6.

> References:
>
> https://blog.gtwang.org/linux/centos-7-install-kvm-qemu-virtual-machine-tutorial/

```
sudo yum install vim
# vim isn't preinstalled, so intall it
sudo vim /etc/sysconfig/selinux
# change "SELINUX=enforced" to "SELINUX=disabled"
reboot
sudo systemctl start libvirtd
# this line is to manually start libvird
sudo systemctl enable libvirtd
# this line is to auto start libvird at boot
reboot
```

2.

1.

> References:
>
> https://zer931.pixnet.net/blog/post/36975565

When using NAT configuration on a guest OS, the guest OS isn't visible to other machines that are on the same network as the host. The host acts like the guest's gateway and a virtual network is built between host and guest.

When using bridge configuration on a guest, the guest is visible to other machines on the same network. In fact, the guest will look just like all the other machines. Therefore if the network disconnects, the guest and the host won't be able to communicate through network, unlike the NAT case.

2.

> References:
>
> https://www.linux-kvm.org/page/Networking#Public_Bridge
>
> https://www.techotopia.com/index.php/Creating_a_CentOS_KVM_Networked_Bridge_Interface

```
sudo -i
ifconfig
# check current configuration
# in this case, we are creating a bridge called br10 to ens33
ifdown ens33
# bring down the interface that we are bridging to
```

Edit `/etc/sysconfig/network-scripts/ifcfg-ens33`

```
# some configs already here
# ...
# BOOTPROTO=dhcp # comment this line
# ...
BRIDGE=br10 # add this line
```

Create `/etc/sysconfig/network-scripts/ifcfg-br10`

```
DEVICE=br10
BOOTPROTO-dhcp
ONBOOT=yes
TYPE=Bridge
```

Back to command line

```
ifup ens33
ifup br10
# bring both interfaces up
reboot
# make sure everything behave properly
```

3.

> References:
>
> https://serverfault.com/questions/731417/how-do-you-create-a-qcow2-file-that-is-small-yet-commodious-on-a-linux-server
>
> https://docs.fedoraproject.org/en-US/fedora/rawhide/install-guide/appendixes/Kickstart_Syntax_Reference/
>
> https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/virtualization_deployment_and_administration_guide/sect-guest_virtual_machine_installation_overview-creating_guests_with_virt_install
>
> https://wyde.github.io/2017/10/15/How-to-Install-VM-on-Linux-KVM-Virtualization-Host-using-Kickstart-File/
>
> https://forums.centos.org/viewtopic.php?t=46192
>
> https://jamielinux.com/docs/libvirt-networking-handbook/bridged-network.html

1.

```
sudo mkdir -p /data/img
# -p for recursively create directories
```

2.

```
cd /data/img
sudo qemu-img create -f qcow2 nasa.qcow2 10G
# usage:
# qemu-img create -f <format> <filename> <size>
```

3.

```
sudo -i
yum install wget
# wget is used to grab xiaoming's script
cd /data
wget ix.io/1ElA -O xm.cfg
# download xiaoming's script and name it xm.cfg
python -c 'from crypt import crypt; print(crypt("XMishandsome"))' >> xm.cfg
# a encrypted password is needed, so we use python
# to generate one (demonstrated by fedora's document)
vim xm.cfg
```

added lines in `xm.cfg`:

```
# create user "xiaoming" under group "wheel"
# password is encrypted first using python
user --name=xiaoming --groups=wheel --
password=$6$KJDuKLNh7KvcZskn$ptG1.SKzasK9ejG3TmsL1QT/IHsPbjbqtN.ToHKAiAi8AJ59sXVHTJ
OcP1jMyf3meopeY2B1ux83K7A5WDtcV. --iscrypted

# add repo for epel
repo --name=epel --baseurl=http://dl.fedoraproject.org/pub/epel/6/x86_64/

%packages
# some packages already in here
epel-release
vim
sudo
wget

%end
```

4.

```
sudo -i
vim install.sh
chmod 755 install.sh
# enable file execute permission
./install.sh
```

`install.sh`:

```
virt-install \
--name ILoveNASA \ # name
--ram 2048 \ # ram size in MB
--vcpus 2 \ # cpu count
--network bridge=br10 \ # using br10 for network
--disk /data/img/nasa.qcow2 \ # using nasa.qcow2 created earlier for disk
--location http://centos.cs.nctu.edu.tw/7.9.2009/os/x86_64/ \ # url to OS
--initrd-inject /data/xm.cfg \ # kickstart file location
--extra-args="ks=file:/xm.cfg console=tty0 console=ttyS0,115200n8" \ # some configs for console
--nographics # to install without GUI
```

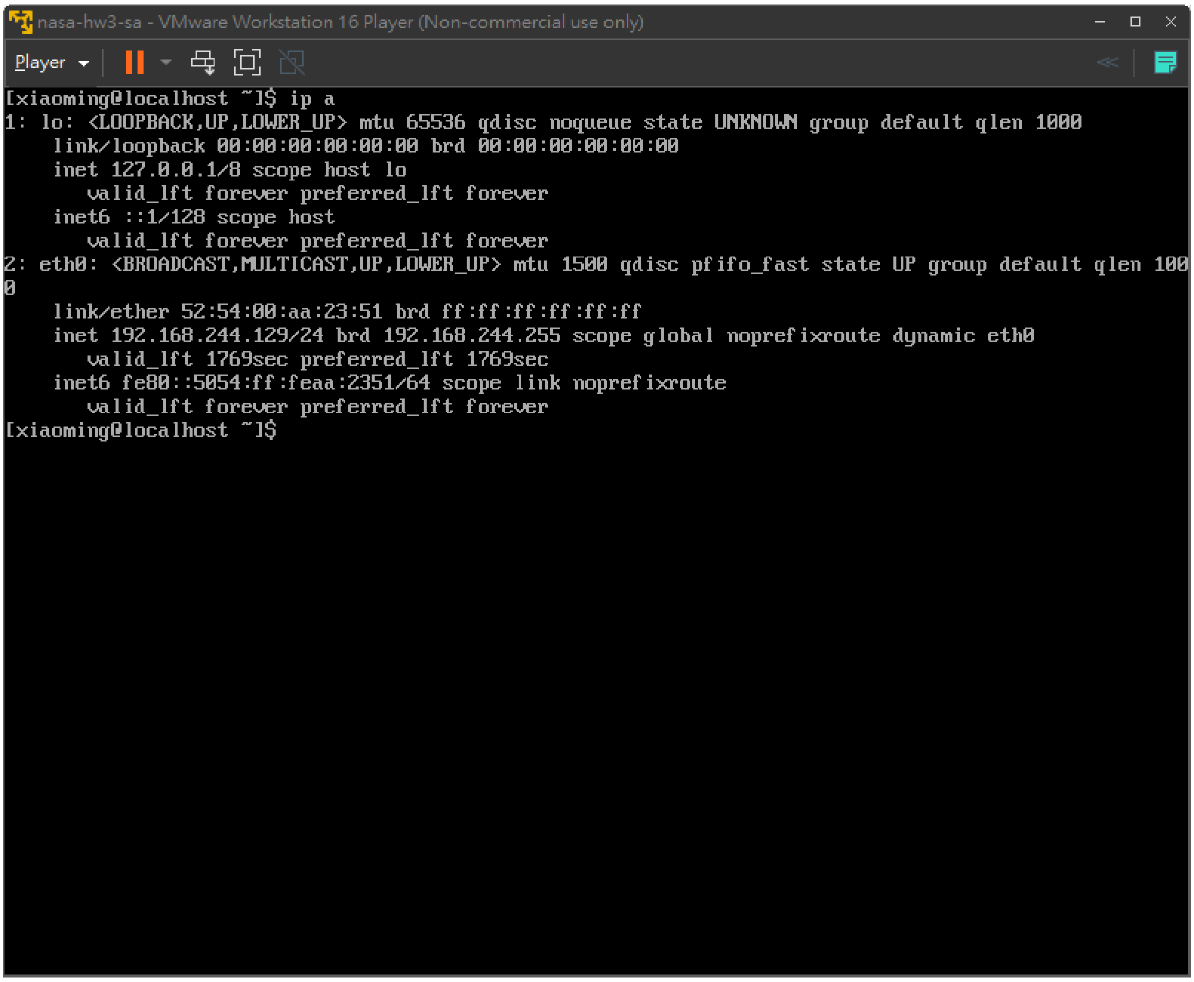
4.

1.

> References:
>
> https://www.cyberciti.biz/faq/centos-stop-start-restart-sshd-command/

```
sudo -i
yum install openssh-clients openssh-server
systemctl start sshd
systemctl enable sshd
```

2.


nasa-hw3-sa - VMware Workstation 16 Player (Non-commercial use only)
Player

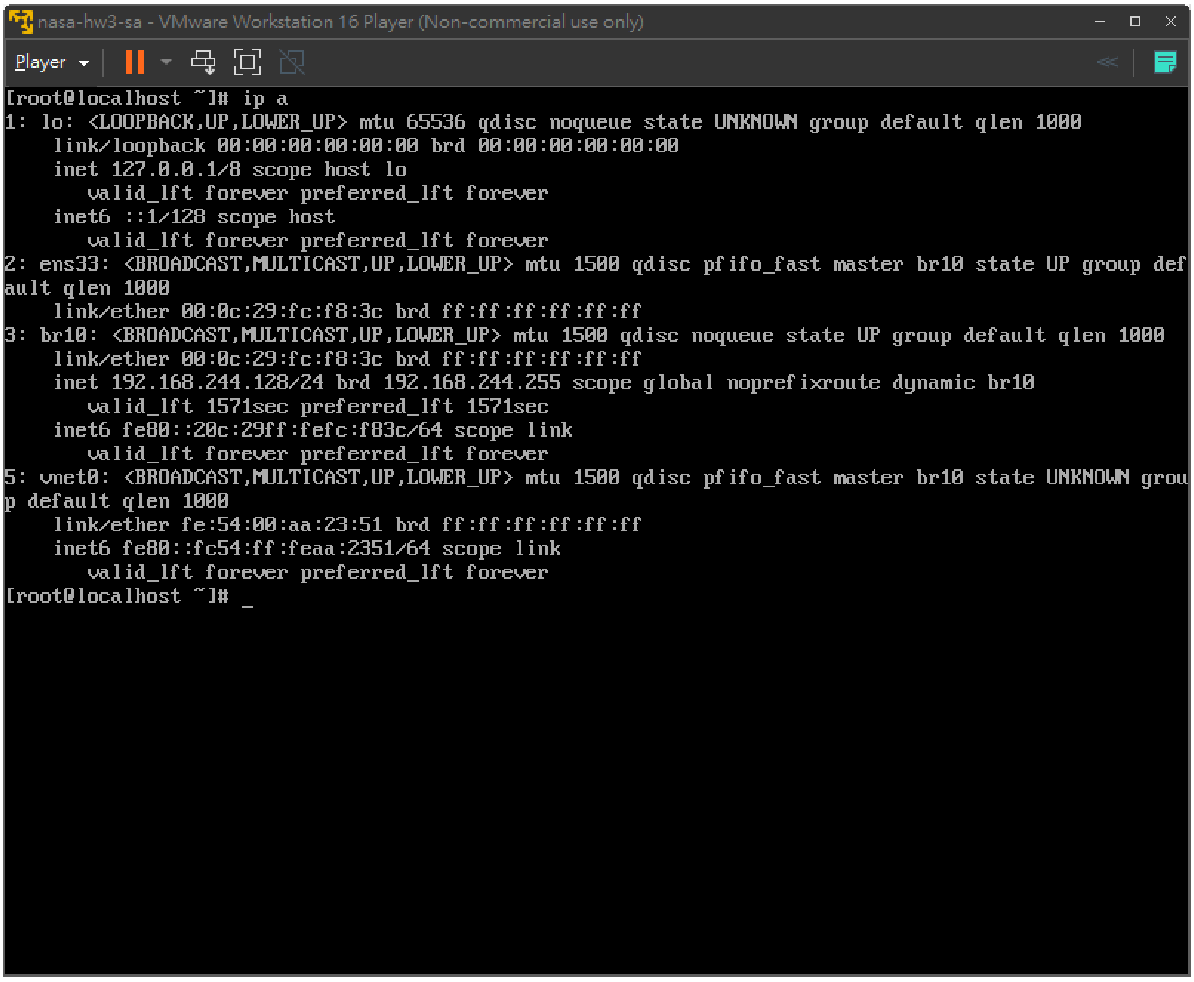
```
[xiaoming@localhost ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 100
0
    link/ether 52:54:00:aa:23:51 brd ff:ff:ff:ff:ff:ff
    inet 192.168.244.129/24 brd 192.168.244.255 scope global noprefixroute dynamic eth0
       valid_lft 1769sec preferred_lft 1769sec
    inet6 fe80::5054:ff:feaa:2351/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[xiaoming@localhost ~]$
```

3.

> References:
>
> https://superuser.com/questions/637669/how-to-exit-a-virsh-console-connection

Press `Ctrl` + `]`

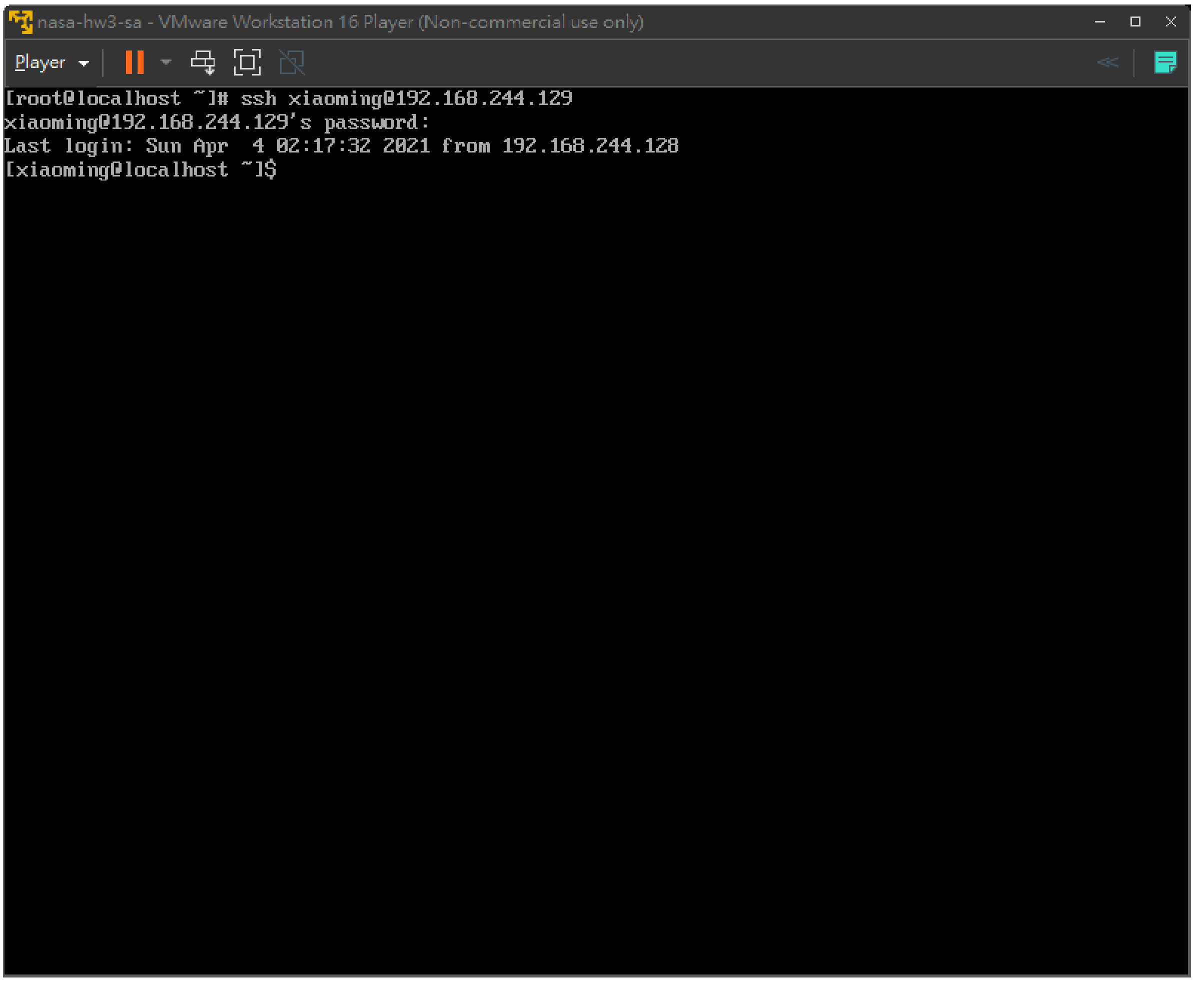
4.


nasa-hw3-sa - VMware Workstation 16 Player (Non-commercial use only)
Player

```
[root@localhost ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master br10 state UP group def
ault qlen 1000
    link/ether 00:0c:29:fc:f8:3c brd ff:ff:ff:ff:ff:ff
3: br10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 00:0c:29:fc:f8:3c brd ff:ff:ff:ff:ff:ff
    inet 192.168.244.128/24 brd 192.168.244.255 scope global noprefixroute dynamic br10
       valid_lft 1571sec preferred_lft 1571sec
    inet6 fe80::20c:29ff:fefc:f83c/64 scope link
       valid_lft forever preferred_lft forever
5: vnet0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master br10 state UNKNOWN grou
p default qlen 1000
    link/ether fe:54:00:aa:23:51 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc54:ff:feaa:2351/64 scope link
       valid_lft forever preferred_lft forever
[root@localhost ~]# _
```

5.

```
ssh xiaoming@192.168.244.129
# usage:
# ssh <username>@<address>
```



nasa-hw3-sa - VMware Workstation 16 Player (Non-commercial use only)
Player

```
[root@localhost ~]# ssh xiaoming@192.168.244.129
xiaoming@192.168.244.129's password:
Last login: Sun Apr  4 02:17:32 2021 from 192.168.244.128
[xiaoming@localhost ~]$
```

5.

1.

> References:
>
> https://docs.fedoraproject.org/en-US/Fedora/18/html/Virtualization_Administration_Guide/ch15s05.html

```
sudo virsh suspend ILoveNASA
```

2.

> References:
>
> https://www.cyberciti.biz/faq/how-to-clone-existing-kvm-virtual-machine-images-on-linux/

```
sudo virt-clone --original ILoveNASA --name ILoveNASA-2 --auto-clone
# usage:
# virt-clone --original <og guest name> --name <new guest name> --auto-clone
```

3.

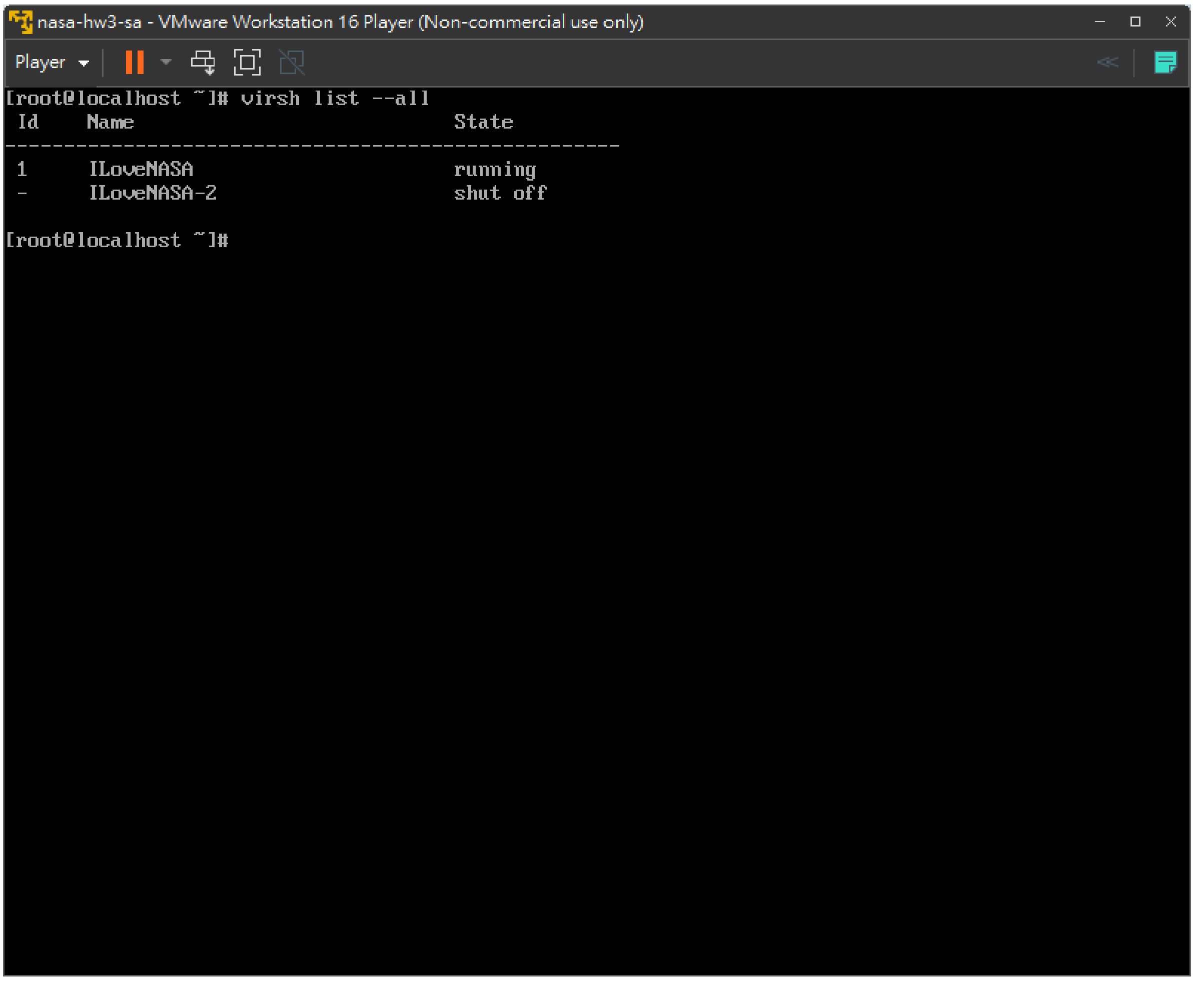
> References:
>
> https://docs.fedoraproject.org/en-US/Fedora/18/html/Virtualization_Administration_Guide/ch15s05.html

```
sudo virsh resume ILoveNASA
```

4.

> References:
>
> https://www.cyberciti.biz/faq/how-to-clone-existing-kvm-virtual-machine-images-on-linux/

```
sudo virsh list --all
```


nasa-hw3-sa - VMware Workstation 16 Player (Non-commercial use only)
Player

```
[root@localhost ~]# virsh list --all
 Id    Name                           State
----------------------------------------------------
 1     ILoveNASA                      running
 -     ILoveNASA-2                    shut off

[root@localhost ~]#
```

5.

> References:
>
> https://serverfault.com/questions/403561/change-amount-of-ram-and-cpu-cores-in-kvm

```
sudo virsh setvcpus ILoveNASA-2 1 --config
# usage:
# virsh setvcpus <guest name> <cpu count> --config
```

6.

> References:
>
> https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/virtualization_deployment_and_administration_guide/sect-statlists

```
sudo virsh domiflist ILoveNASA-2
# usage:
# virsh domiflist <guest name>
```