

Linux networking and firewalls report

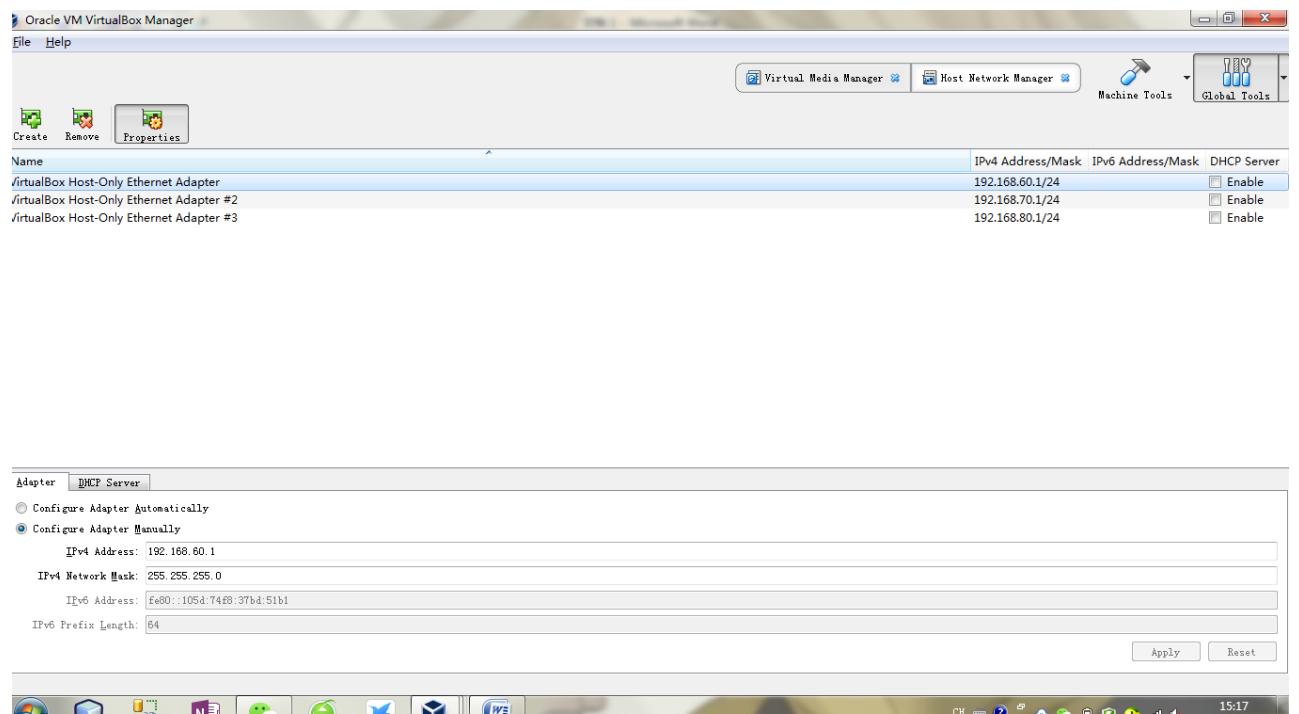
RONG PENG

9511207004

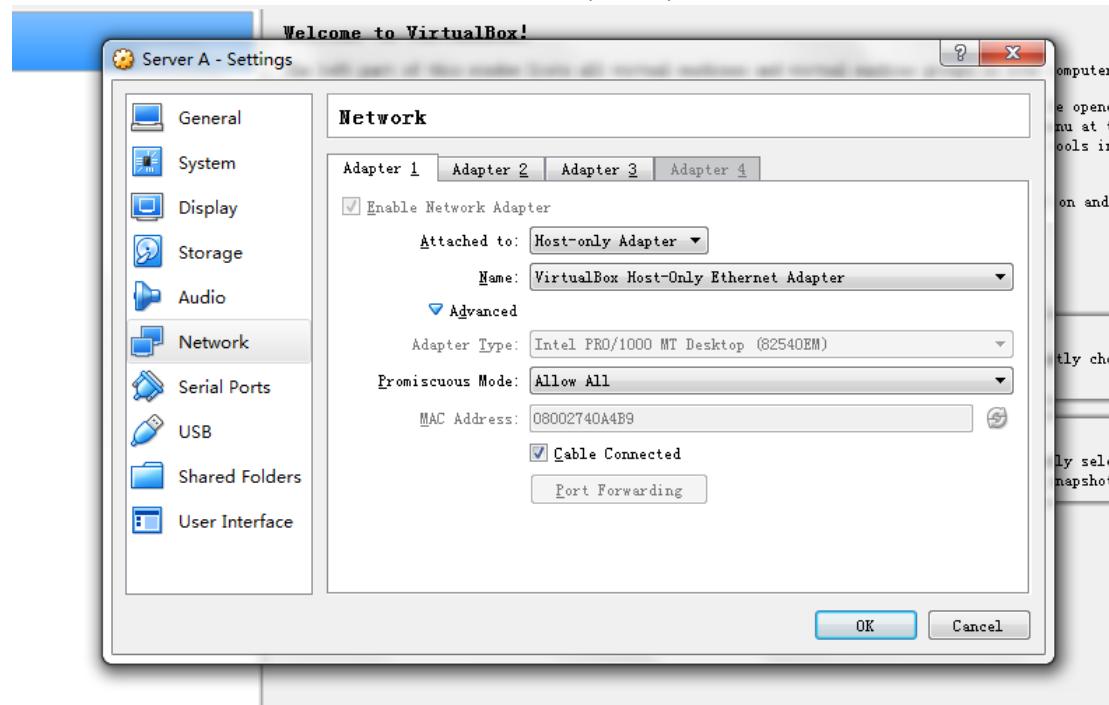
921411446@qq.com

Setting VirtualBox appliance

Firstly I followed the instruction and set three IP addresses following the introduction:



And Then I set Server A to use Host-only Adapter:

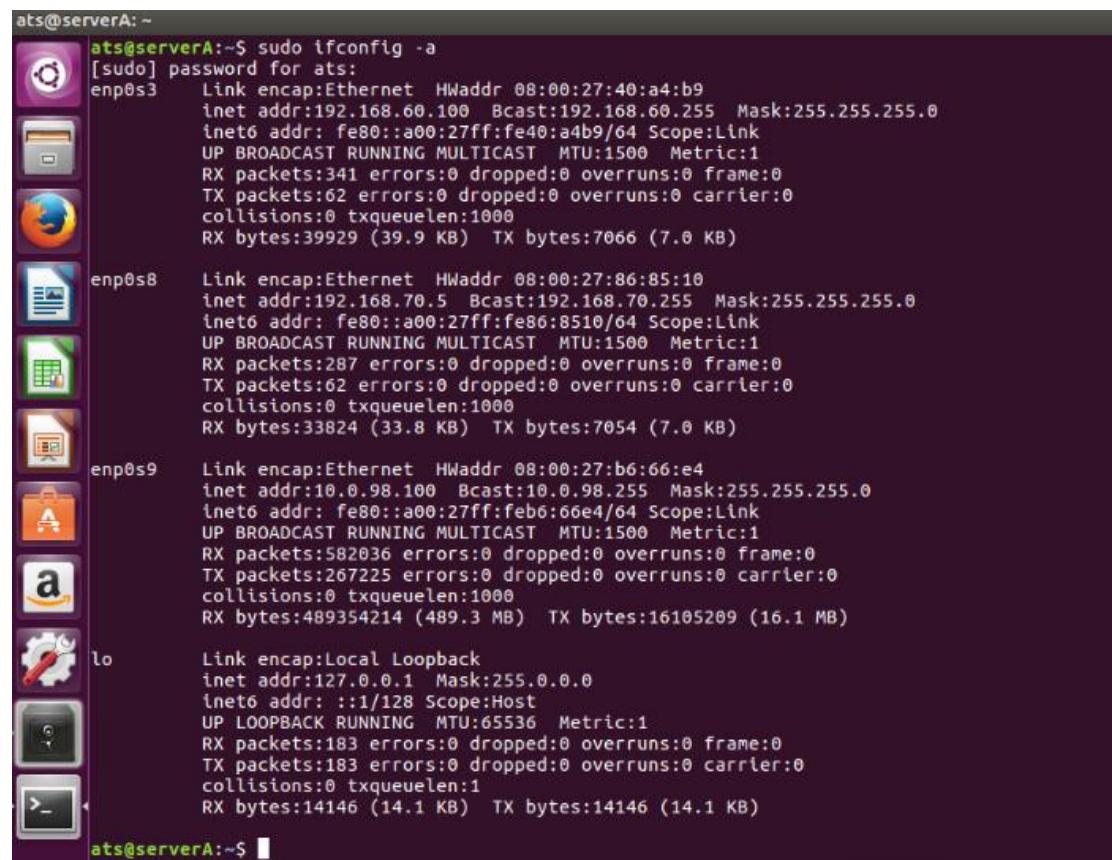


Now, let's start my Server A, and begin the Task1.

Task1:MAC address

Command ifconfig is used to show the network interface configuration.

Command netstat is used to show the information of network connection, router table and network interface.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window has a dark background and light-colored text. It displays the output of the 'ifconfig' command, which shows network interface configurations for 'enp0s3', 'enp0s8', 'enp0s9', and 'lo'. The 'enp0s3' interface is connected to an Ethernet port with an IP of 192.168.60.100. The 'lo' interface is the loopback interface with an IP of 127.0.0.1. Other icons for various applications like file manager, terminal, and system settings are visible on the desktop.

```
ats@serverA: ~
[ats@serverA ~]$ sudo ifconfig -a
[sudo] password for ats:
enp0s3    Link encap:Ethernet HWaddr 08:00:27:40:a4:b9
          inet addr:192.168.60.100 Bcast:192.168.60.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe40:a4b9/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:341 errors:0 dropped:0 overruns:0 frame:0
            TX packets:62 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:39929 (39.9 KB) TX bytes:7066 (7.0 KB)

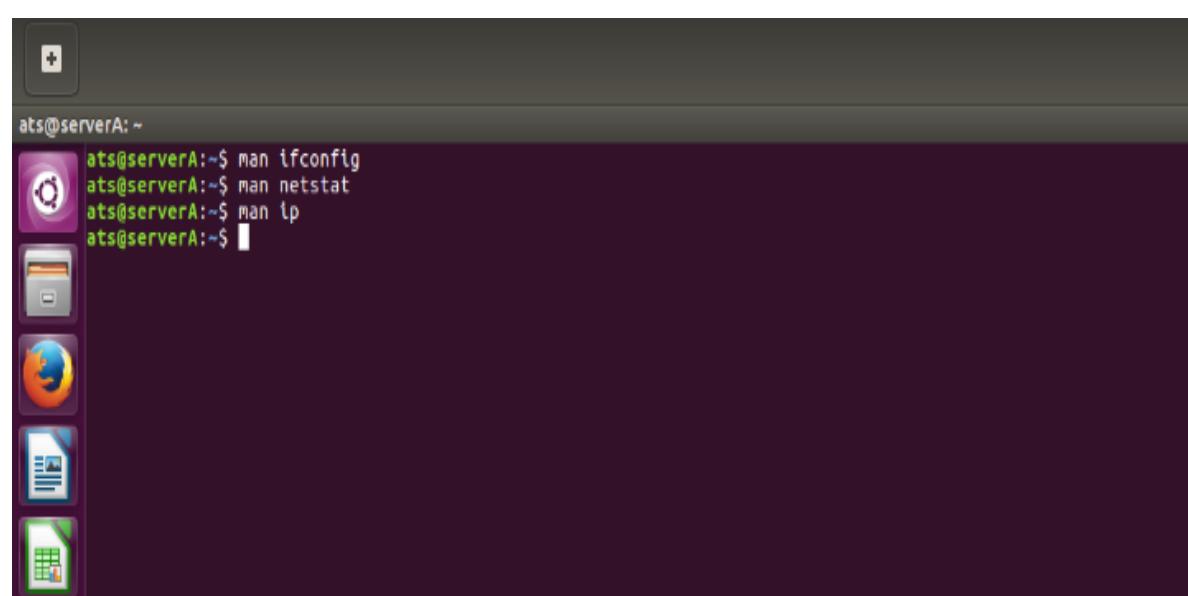
enp0s8    Link encap:Ethernet HWaddr 08:00:27:86:85:10
          inet addr:192.168.70.5 Bcast:192.168.70.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe86:8510/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:287 errors:0 dropped:0 overruns:0 frame:0
            TX packets:62 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:33824 (33.8 KB) TX bytes:7054 (7.0 KB)

enp0s9    Link encap:Ethernet HWaddr 08:00:27:b6:66:e4
          inet addr:10.0.98.100 Bcast:10.0.98.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:feb6:66e4/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:582036 errors:0 dropped:0 overruns:0 frame:0
            TX packets:267225 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:489354214 (489.3 MB) TX bytes:16105209 (16.1 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING MTU:65536 Metric:1
            RX packets:183 errors:0 dropped:0 overruns:0 frame:0
            TX packets:183 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
            RX bytes:14146 (14.1 KB) TX bytes:14146 (14.1 KB)

ats@serverA: ~
```

Linux man command is the abbreviation of manual, its man pages can help to check some reference information.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window has a dark background and light-colored text. It displays the command 'man ifconfig' being entered. The desktop environment includes icons for file manager, terminal, and system settings.

```
ats@serverA: ~
[ats@serverA ~]$ man ifconfig
[ats@serverA ~]$ man netstat
[ats@serverA ~]$ man ip
[ats@serverA ~]$
```

ats@serverA: ~

IFCONFIG(8) Linux Programmer's Manual 15:32

NAME ifconfig - configure a network interface

SYNOPSIS

```
ifconfig [-v] [-a] [-s] [interface]
ifconfig [-v] interface [aftype] options | address ...
```

DESCRIPTION

Ifconfig is used to configure the kernel-resident network interfaces. It is used at boot time to set up interfaces as necessary. After that, it is usually only needed when debugging or when system tuning is needed.

If no arguments are given, ifconfig displays the status of the currently active interfaces. If a single interface argument is given, it displays the status of the given interface only; if a single -a argument is given, it displays the status of all interfaces, even those that are down. Otherwise, it configures an interface.

Address Families

If the first argument after the interface name is recognized as the name of a supported address family, that address family is used for decoding and displaying all protocol addresses. Currently supported address families include **inet** (TCP/IP, default), **inet6** (IPv6), **ax25** (AMPR Packet Radio), **ddp** (Appletalk Phase 2), **ipx** (Novell IPX) and **netrom** (AMPR Packet radio).

OPTIONS

- a display all interfaces which are currently available, even if down
- s display a short list (like netstat -i)
- v be more verbose for some error conditions

interface

The name of the interface. This is usually a driver name followed by a unit number, for example eth0 for the first Ethernet interface. If your kernel supports alias interfaces, you can specify them with eth0:0 for the first alias of eth0. You can use them to assign a second address. To delete an alias interface use ifconfig eth0:0 down. Note: for every scope (i.e. same net with address/netmask combination) all aliases are deleted, if you delete the first (primary).

up This flag causes the interface to be activated. It is implicitly specified if an address is assigned to the interface.

down This flag causes the driver for this interface to be shut down.

[-larp Enable or disable the use of the ARP protocol on this interface.

Screenshot from 2017-11-15 15-32-34.png

ats@serverA: ~

NETSTAT(8) Linux Programmer's Manual 15:32

NAME netstat - Print network connections, routing tables, interface statistics, masquerade connections, and multicast memberships

SYNOPSIS

```
netstat [address family options] [--tcp|-t] [--udp|-u] [--raw|-w] [--listening|-l] [--all|-a] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-users] [--symbolic|-N] [--extend|-e|-extend|-e] [--timers|-o] [-program|-p] [--verbose|-v] [--continuous|-c]
netstat [--route|-r] [address family options] [--extend|-e|--extend|-e] [--verbose|-v] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-users] [--continuous|-c]
netstat [--interfaces|-i] [--all|-a] [--extend|-e|--extend|-e] [--verbose|-v] [--program|-p] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-users] [--continuous|-c]
netstat [--groups|-g] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-users] [--continuous|-c]
netstat [--masquerade|-M] [--extend|-e] [--numeric|-n] [--numeric-hosts] [--numeric-ports] [--numeric-users] [--continuous|-c]
netstat [--statistics|-s] [--tcp|-t] [--udp|-u] [--raw|-w]
netstat [--version|-V]
netstat [--help|-h]
address_family_options:
```

- [-4] [-6] [--protocol=[inet,unix,ipx,ax25,netrom,ddp][,...]] [--unix|-x] [--inet|--ip] [--ax25] [--ipx] [--netrom] [--ddp]

DESCRIPTION

Netstat prints information about the Linux networking subsystem. The type of information printed is controlled by the first argument, as follows:

(none) By default, netstat displays a list of open sockets. If you don't specify any address families, then the active sockets of all configured address families will be printed.

--route , -r Display the kernel routing tables. See the description in route(8) for details. netstat -r and route -e produce the same output.

Screenshot from 2017-11-15 15-32-46.png

```
ats@serverA: ~ IP(8) Linux 15:32 IP(8)

NAME      ip - show / manipulate routing, devices, policy routing and tunnels
SYNOPSIS  ip [ OPTIONS ] OBJECT { COMMAND | help }
          ip [ -force ] -batch filename
OBJECT := { link | address | addrlabel | route | rule | neigh | ntable | tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm |
           netns | l2tp | tcp_metrics }
OPTIONS := { -V[ersion] | -h[uman-readable] | -s[tatistics] | -r[esolve] | -f[amily] { inet | inet6 | ipx | dnet | link } | -o[neline] |
           -n[eigns] name | -a[l] | -c[olor] }
OPTIONS
  -V, -Version
    Print the version of the ip utility and exit.
  -h, -human, -human-readable
    output statistics wth human readable values followed by suffix.
  -b, -batch <FILENAME>
    Read commands from provided file or standard input and invoke them. First failure will cause termination of ip.
  -force Don't terminate ip on errors in batch mode. If there were any errors during execution of the commands, the application return code
    will be non zero.
  -s, -stats, -statistics
    Output more information. If the option appears twice or more, the amount of information increases. As a rule, the information is
    statistics or some time values.
  -d, -details
    Output more detailed information.
  -l, -loops <COUNT>
    Specify maximum number of loops the 'ip address flush' logic will attempt before giving up. The default is 10. Zero (0) means loop
    until all addresses are removed.
```

Task2: Network interfaces

Terminal

```
ats@serverA: ~
ats@serverA:~$ sudo ip address
[sudo] password for ats:
Sorry, try again.
[sudo] password for ats:
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    qlen 1
        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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
    group default qlen 1000
        link/ether 08:00:27:40:a4:b9 brd ff:ff:ff:ff:ff:ff
        inet 192.168.60.100/24 brd 192.168.60.255 scope global enp0s3
            valid_lft forever preferred_lft forever
        inet6 fe80::a00:27ff:fe40:a4b9/64 scope link
            valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
    group default qlen 1000
        link/ether 08:00:27:86:85:10 brd ff:ff:ff:ff:ff:ff
        inet 192.168.70.5/24 brd 192.168.70.255 scope global enp0s8
            valid_lft forever preferred_lft forever
```

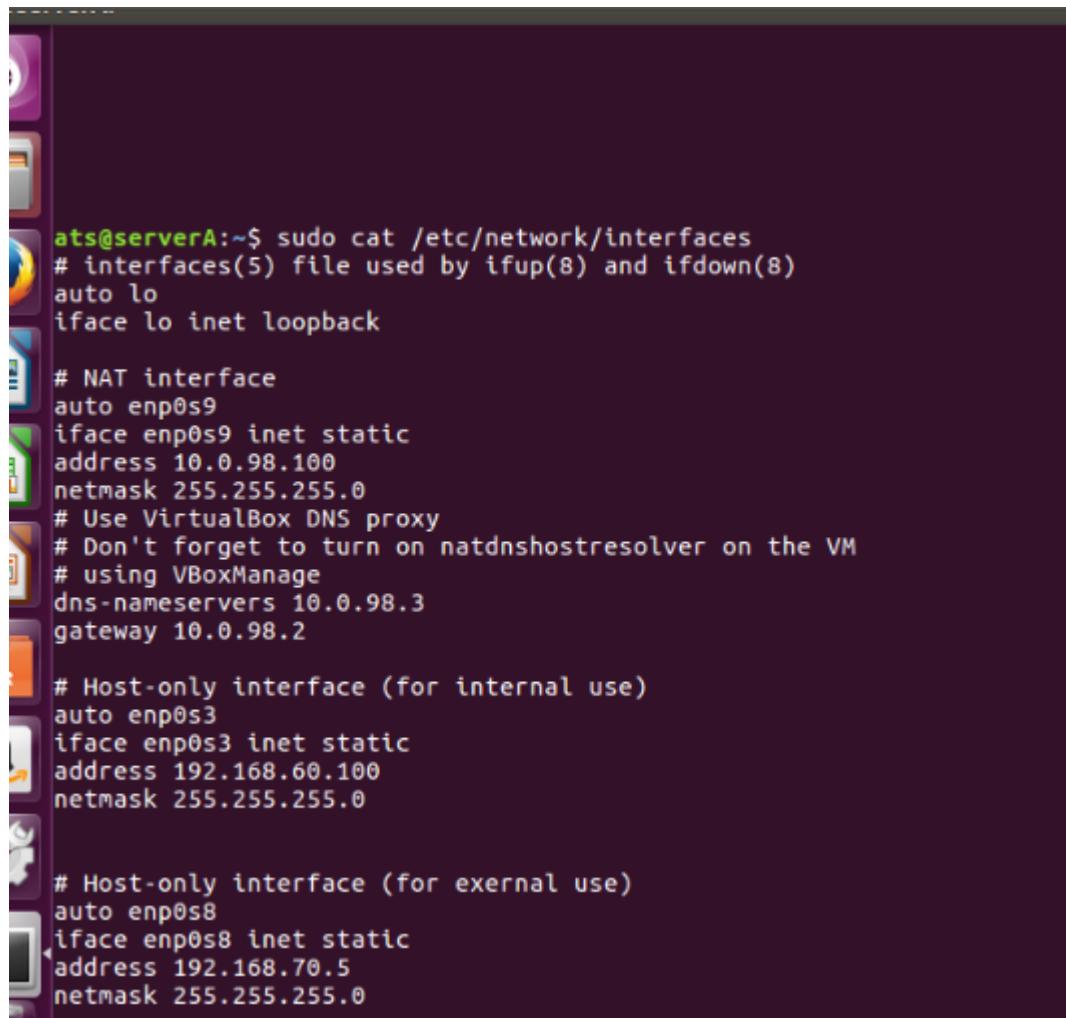
Terminal

```
ats@serverA:~$ ip -4 address
inet 10.0.98.100/24 brd 10.0.98.255 scope global enp0s9
    valid_lft forever preferred_lft forever
inet6 fe80::a00:27ff:feb6:66e4/64 scope link
    valid_lft forever preferred_lft forever
ats@serverA:~$ sudo ip -4 address
sudo: ip-4: command not found
ats@serverA:~$ sudo ip -4 address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 192.168.60.100/24 brd 192.168.60.255 scope global enp0s3
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 192.168.70.5/24 brd 192.168.70.255 scope global enp0s8
        valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 10.0.98.100/24 brd 10.0.98.255 scope global enp0s9
        valid_lft forever preferred_lft forever
ats@serverA:~$
```

Terminal

```
ats@serverA:~$ ip -6 address
inet 192.168.60.100/24 brd 192.168.60.255 scope global enp0s3
    valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 192.168.70.5/24 brd 192.168.70.255 scope global enp0s8
        valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 10.0.98.100/24 brd 10.0.98.255 scope global enp0s9
        valid_lft forever preferred_lft forever
ats@serverA:~$ sudo ip -6 address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 state UNKNOWN qlen 1
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 state UP qlen 1000
    inet6 fe80::a00:27ff:feb6:66e4/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 state UP qlen 1000
    inet6 fe80::a00:27ff:fe86:8510/64 scope link
        valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 state UP qlen 1000
    inet6 fe80::a00:27ff:feb6:66e4/64 scope link
        valid_lft forever preferred_lft forever
ats@serverA:~$
```

Task3:IP addresses, netmasks and subnet



```
ats@serverA:~$ sudo cat /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

# NAT interface
auto enp0s9
iface enp0s9 inet static
address 10.0.98.100
netmask 255.255.255.0
# Use VirtualBox DNS proxy
# Don't forget to turn on natdnshostresolver on the VM
# using VBoxManage
dns-nameservers 10.0.98.3
gateway 10.0.98.2

# Host-only interface (for internal use)
auto enp0s3
iface enp0s3 inet static
address 192.168.60.100
netmask 255.255.255.0

# Host-only interface (for external use)
auto enp0s8
iface enp0s8 inet static
address 192.168.70.5
netmask 255.255.255.0
```

Server A IP:192.168.60.100 , the IP address is assigned to the NAT interface emulated internally by VBox. The NAT engine from VirtualBox is completely decoupled from the any NAT that will configure on the VM.

Task4: Host-only interfaces

The screenshot shows two separate Command Prompt windows. The top window is titled '管理员: C:\Windows\system32\cmd.exe' and displays detailed network configuration for a 'VirtualBox Host-Only Network' interface. It includes fields like 'DHCP 已启用' (Enabled), '自动配置已启用' (Autoconfiguration Enabled), and various IP and DNS settings. The bottom window is also titled '管理员: C:\Windows\system32\cmd.exe' and shows the command 'ipconfig/all' being run, displaying basic network information for the host system.

Because my host is Windows, so I wrote ipconfig /all in the CMD, then it showed information. The above screenshot is the host-only interface in the guest.

Task5: Routing tables in the host OS

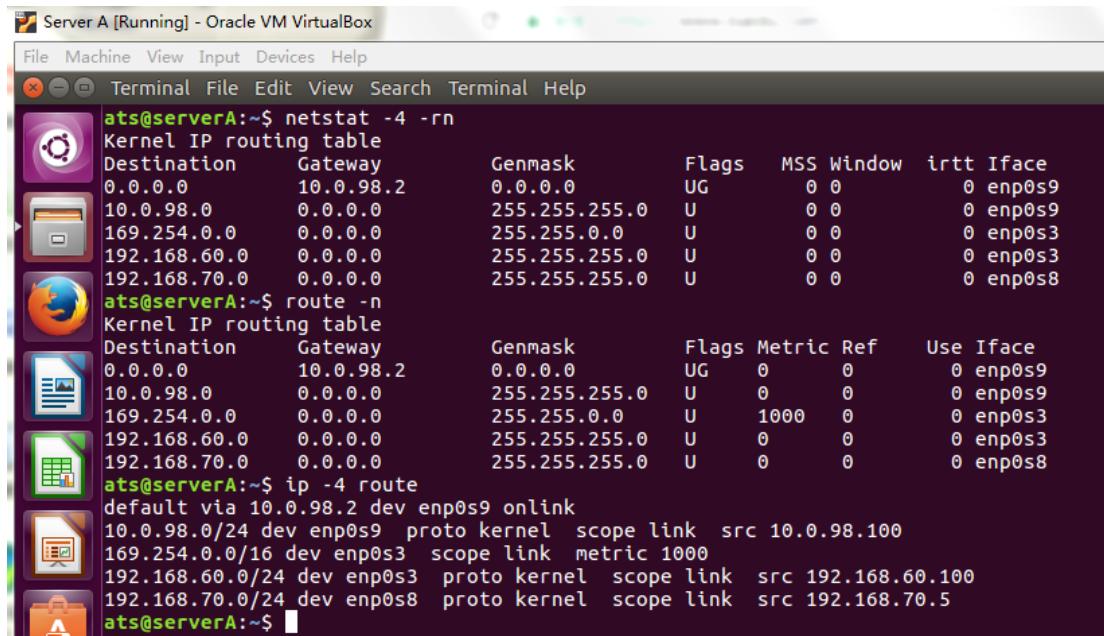
route -4 PRINT

192.168.60.0	255.255.255.0	在链路上	192.168.60.1	266
192.168.60.1	255.255.255.255	在链路上	192.168.60.1	266
192.168.70.0	255.255.255.0	在链路上	192.168.70.1	266
192.168.70.1	255.255.255.255	在链路上	192.168.70.1	266
192.168.80.0	255.255.255.0	在链路上	192.168.80.1	266
192.168.80.1	255.255.255.255	在链路上	192.168.80.1	266

This command is to show the router table's information.

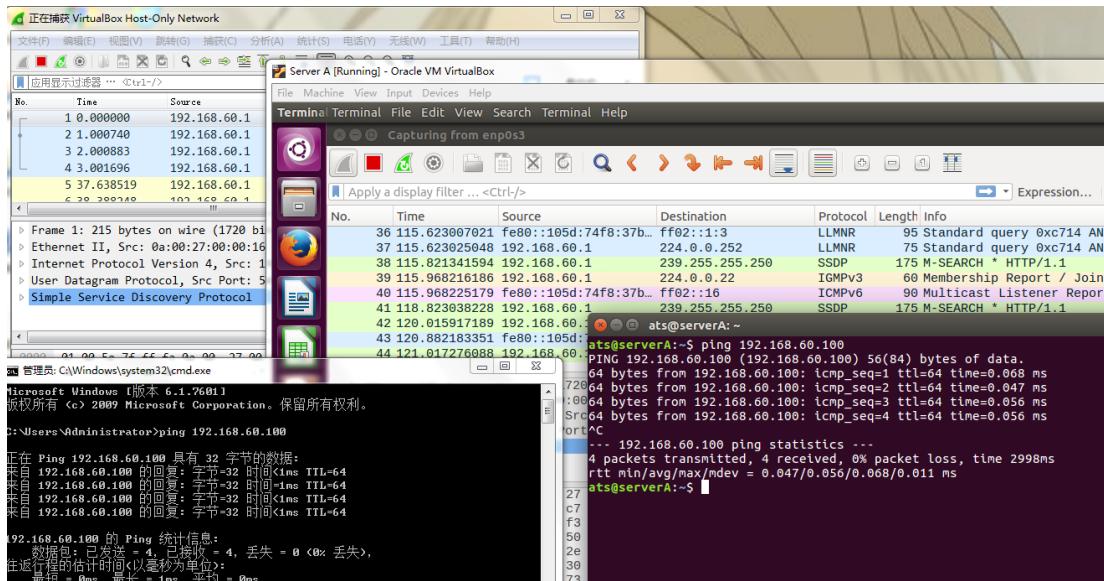
Task6: Routing tables in the guest OS

The gateway is NAT interface.



```
ats@serverA:~$ netstat -4 -rn
Kernel IP routing table
Destination     Gateway         Genmask        Flags MSS Window irtt Iface
0.0.0.0          10.0.98.2      0.0.0.0       UG    0 0        0 enp0s9
10.0.98.0        0.0.0.0        255.255.255.0 U      0 0        0 enp0s9
169.254.0.0      0.0.0.0        255.255.0.0   U      0 0        0 enp0s3
192.168.60.0     0.0.0.0        255.255.255.0 U      0 0        0 enp0s3
192.168.70.0     0.0.0.0        255.255.255.0 U      0 0        0 enp0s8
ats@serverA:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref  Use Iface
0.0.0.0          10.0.98.2      0.0.0.0       UG    0 0        0 enp0s9
10.0.98.0        0.0.0.0        255.255.255.0 U      0 0        0 enp0s9
169.254.0.0      0.0.0.0        255.255.0.0   U      1000 0       0 enp0s3
192.168.60.0     0.0.0.0        255.255.255.0 U      0 0        0 enp0s3
192.168.70.0     0.0.0.0        255.255.255.0 U      0 0        0 enp0s8
ats@serverA:~$ ip -4 route
default via 10.0.98.2 dev enp0s9 onlink
10.0.98.0/24 dev enp0s9 proto kernel scope link src 10.0.98.100
169.254.0.0/16 dev enp0s3 proto kernel scope link metric 1000
192.168.60.0/24 dev enp0s3 proto kernel scope link src 192.168.60.100
192.168.70.0/24 dev enp0s8 proto kernel scope link src 192.168.70.5
ats@serverA:~$
```

Task7: Ping the host-based host-only interface



Obviously, it is identical.

A screenshot of Wireshark showing a capture of ICMP traffic. The table lists 23 entries of ICMP Echo requests and replies between the host (192.168.60.1) and the guest (192.168.60.100). The protocol is ICMP, length is 74 bytes, and the info column shows details like sequence numbers and TTL values.

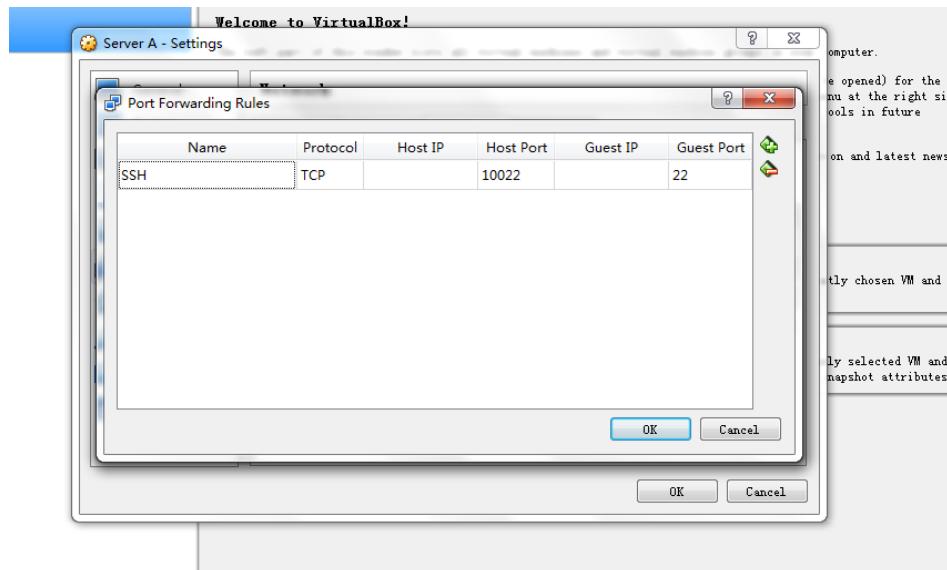
No.	Time	Source	Destination	Protocol	Length	Info
16	110.245632	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=53/13568, ttl=64 (reply in 17)
17	110.246036	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=53/13568, ttl=64 (request in 16)
18	111.248876	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=54/13824, ttl=64 (reply in 19)
19	111.249683	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=54/13824, ttl=64 (request in 18)
20	112.252904	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=55/14080, ttl=64 (reply in 21)
21	112.253463	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=55/14080, ttl=64 (request in 20)
22	113.257075	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=56/14336, ttl=64 (reply in 23)
23	113.257675	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=56/14336, ttl=64 (request in 22)

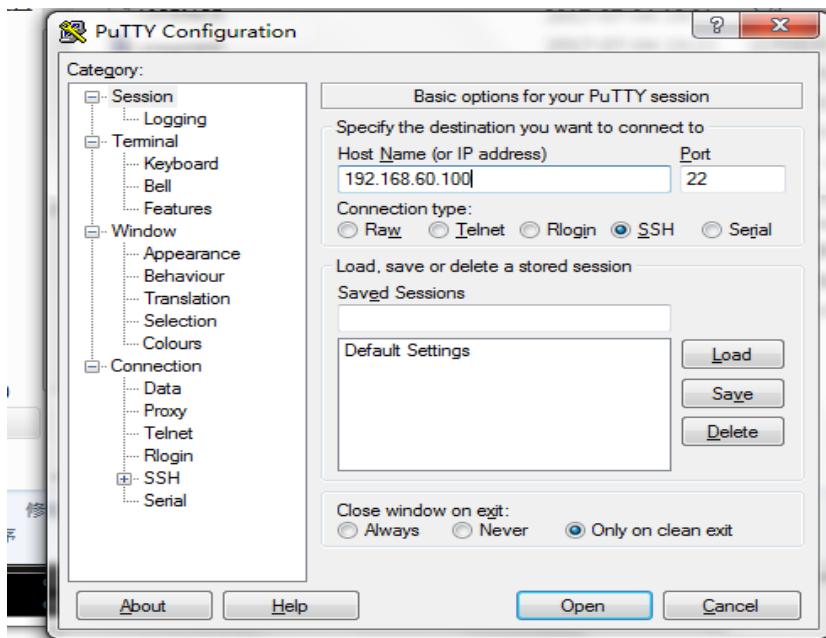
A screenshot of Wireshark showing a capture of ARP traffic. The table lists several entries, including ARP requests for broadcast addresses and ICMP echo requests and replies. One entry is highlighted in yellow, showing an ARP request from CadmusCo_40:a4:b9 to 192.168.60.100.

No.	Time	Source	Destination	Protocol	Length	Info
14	110.249067410	0a:00:27:00:00:16	Broadcast	ARP	60	Who has 192.168.60.100? Tell 192.168.60.1
15	110.249087866	CadmusCo_40:a4:b9	0a:00:27:00:00:16	ARP	42	192.168.60.100 is at 0a:00:27:40:a4:b9
16	110.249511065	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=53/13568, ttl=64 (reply in 17)
17	110.249535242	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=53/13568, ttl=64 (request in 16)
18	111.253076422	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=54/13824, ttl=64 (reply in 19)
19	111.253200734	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=54/13824, ttl=64 (request in 20)
20	112.256950798	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=55/14080, ttl=64 (reply in 21)
21	112.256989082	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=55/14080, ttl=64 (request in 22)
22	112.261196261	192.168.60.1	192.168.60.100	ICMP	74	Echo (ping) request id=0x0001, seq=56/14336, ttl=64 (reply in 23)
23	113.261238540	192.168.60.100	192.168.60.1	ICMP	74	Echo (ping) reply id=0x0001, seq=56/14336, ttl=64 (request in 22)

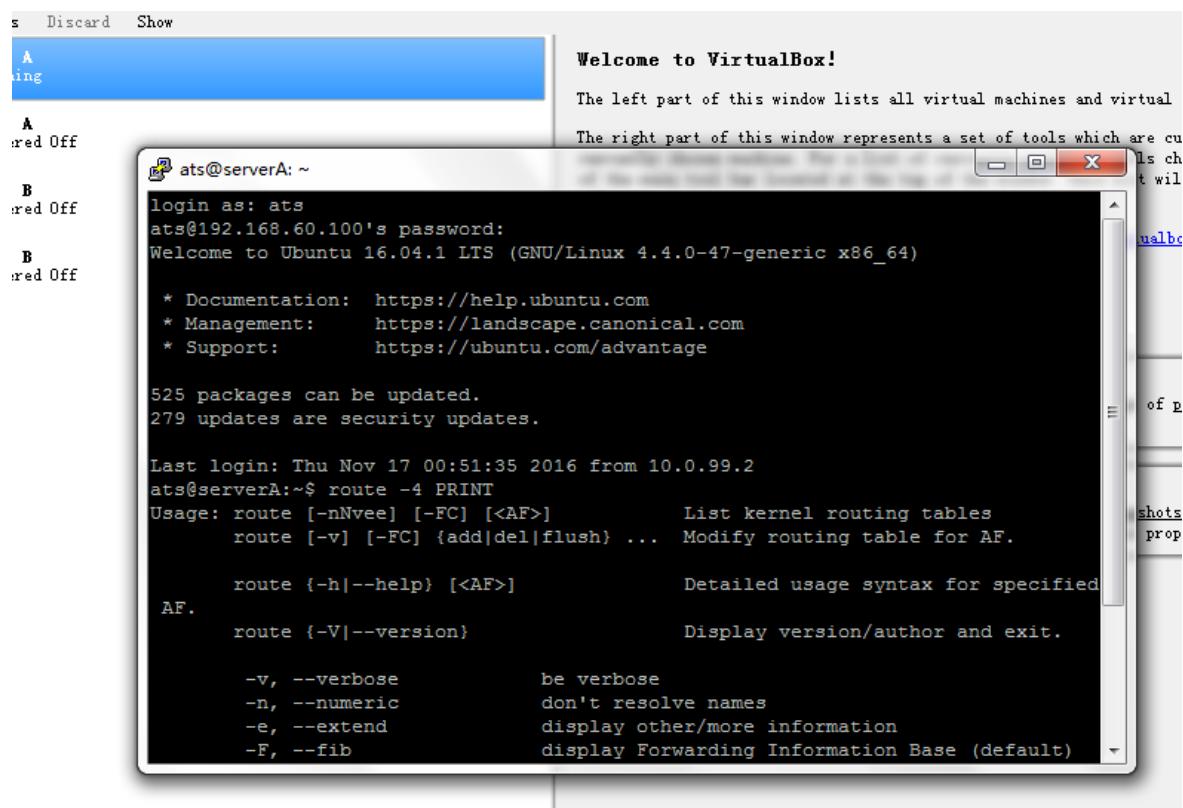
Task8: ssh into VM via localhost

My host is Windows, so I download putty to implement the ssh.

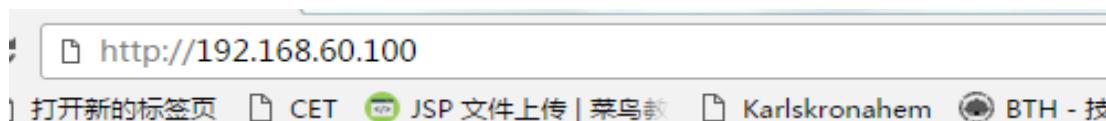
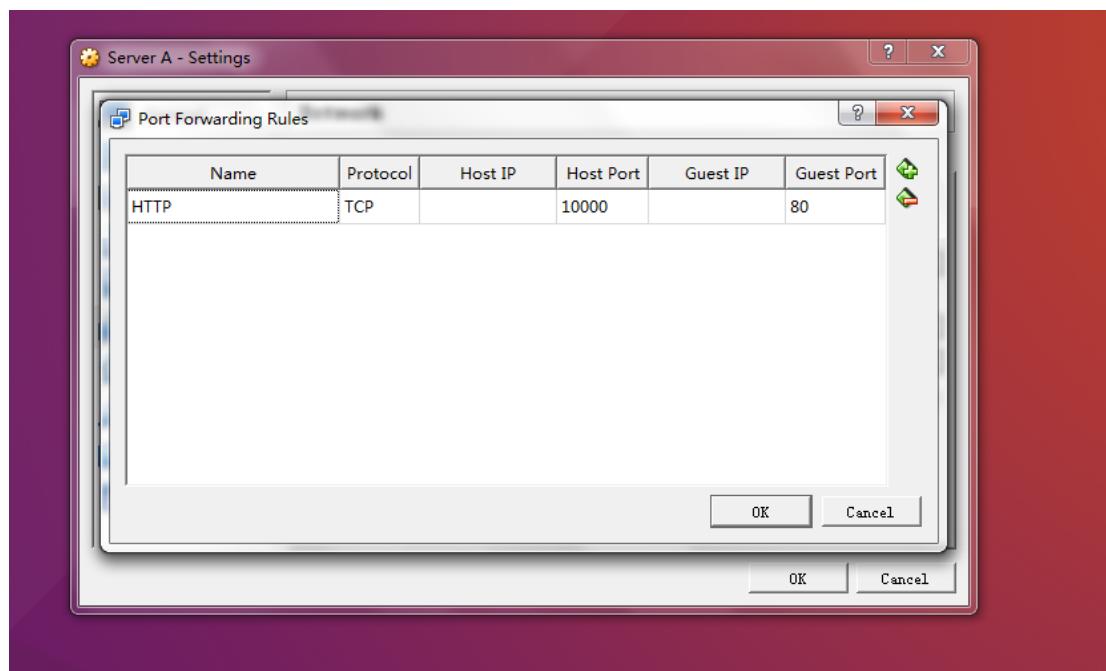


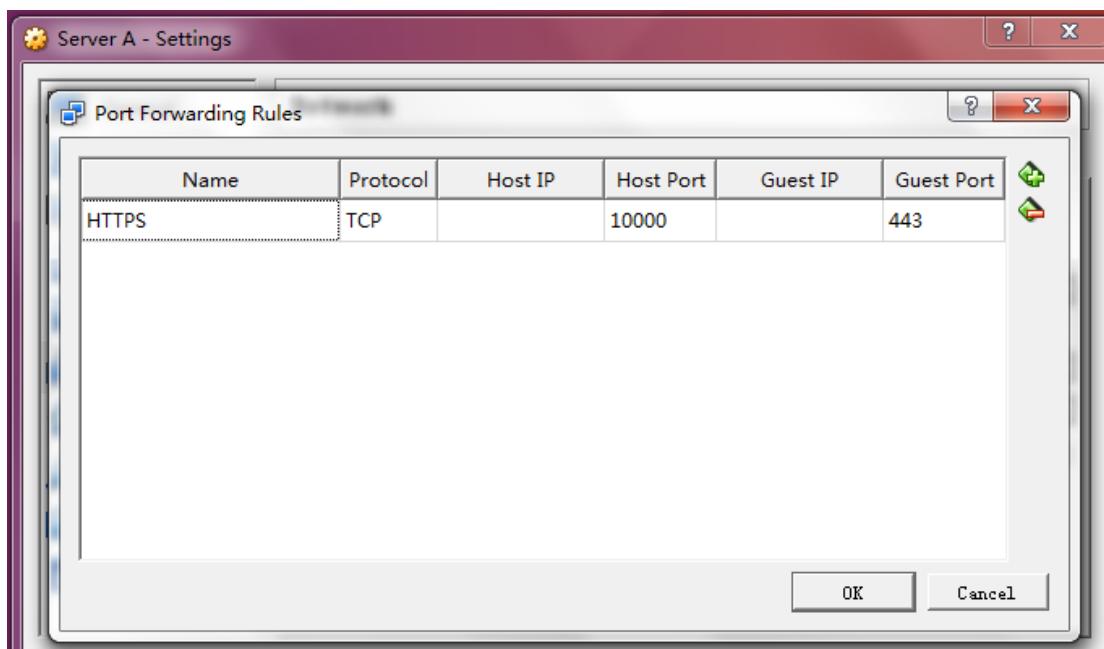


I input the serverA IP, and ssh default port is 22 , then I like the default port, then the black window prompt password, I input the ats password, then successfully login in.



Task9: Add forwarding rules for HTTP and HTTPS in VirtualBox





https://192.168.60.100:443

https://192.168.60.100

https://192.168.60.100:443 - Google 搜索

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

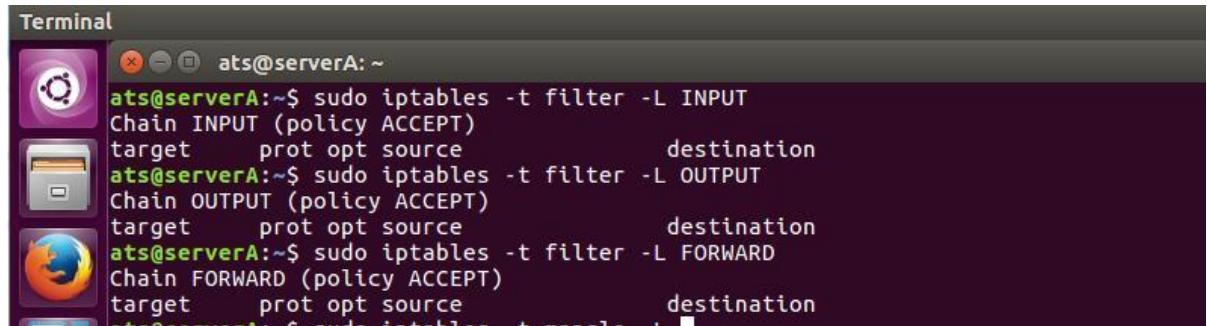
```
/etc/apache2/
├── apache2.conf
│   └── ports.conf
└── mods-enabled
    └── *.load
```

I changed the HTTPS, but Chrome may set something and show unsecure, but it was accessible.

Task10:Default firewall policy and rules

My rule : `sudo iptables -L chain_name`

Default is filter table.



```
Terminal ats@serverA: ~
ats@serverA:~$ sudo iptables -t filter -L INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ats@serverA:~$ sudo iptables -t filter -L OUTPUT
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ats@serverA:~$ sudo iptables -t filter -L FORWARD
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
```

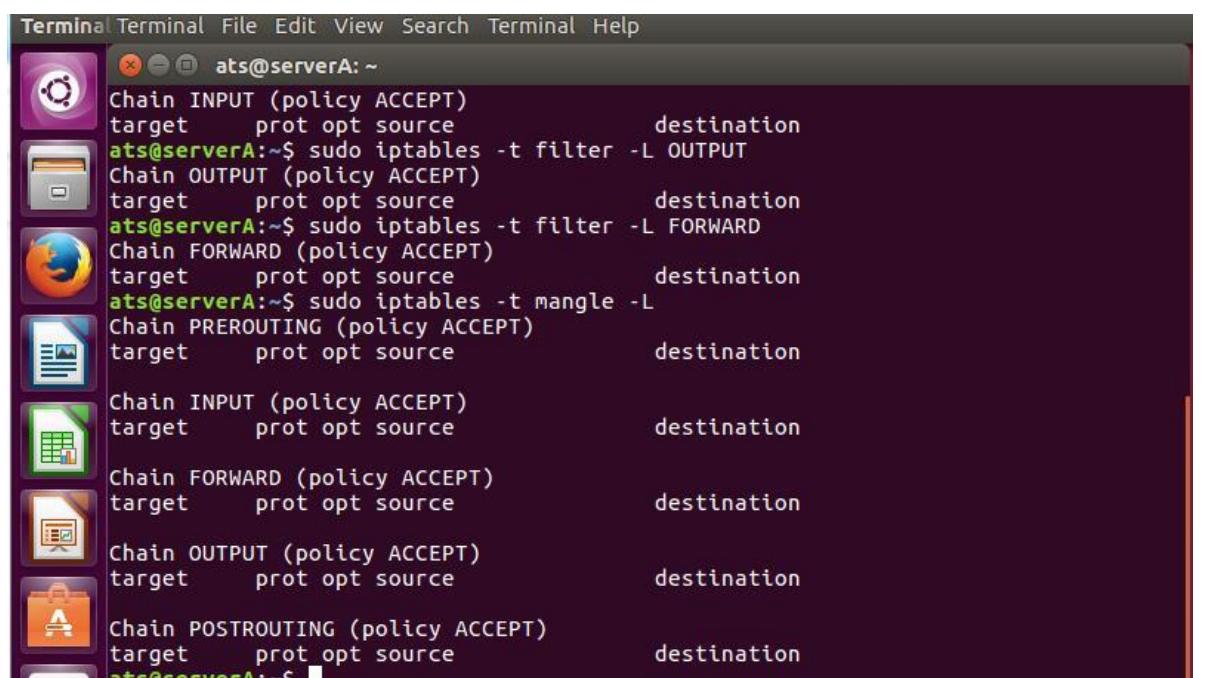
Filter table: Filter packet.

NAT table:nat function(map port/IP).

Mangle:specific packets' modification.

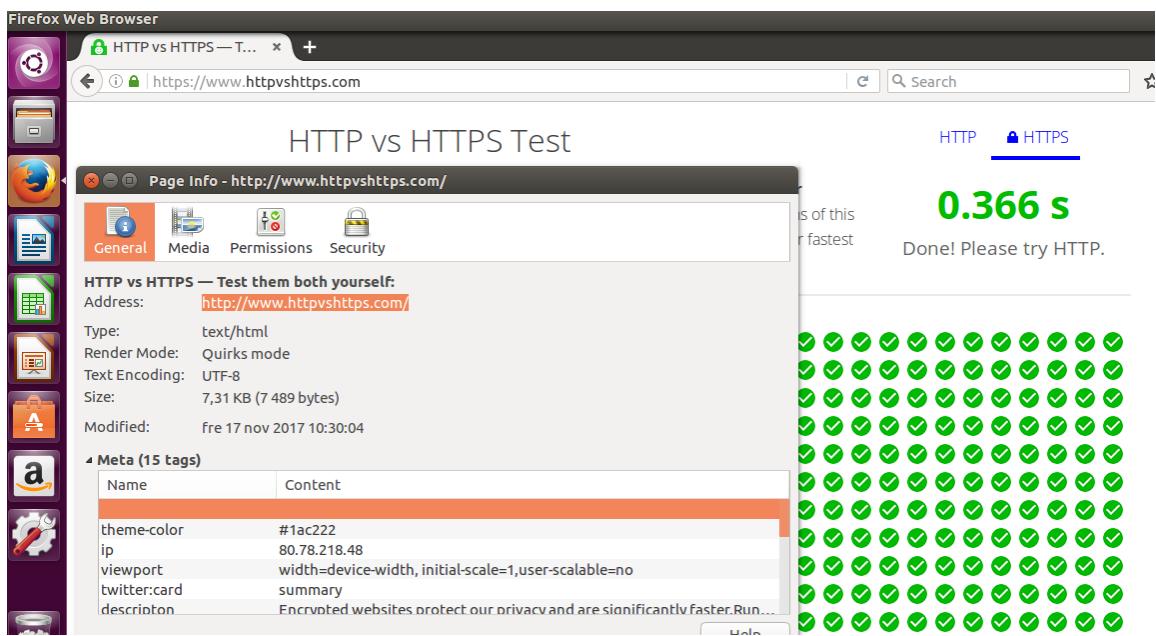
INPUT chain: after through the router table ,its destination is this machine.

OUTPUT chain: generate the packet from the machine and ready to go outside.



```
Terminal Terminal File Edit View Search Terminal Help
Terminal ats@serverA: ~
ats@serverA:~$ sudo iptables -t filter -L INPUT
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ats@serverA:~$ sudo iptables -t filter -L OUTPUT
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ats@serverA:~$ sudo iptables -t filter -L FORWARD
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ats@serverA:~$ sudo iptables -t mangle -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ats@serverA:~$ sudo iptables -t mangle -L
Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
ats@serverA:~$
```

```
ats@serverA:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target    prot opt source               destination
Chain INPUT (policy ACCEPT)
target    prot opt source               destination
Chain OUTPUT (policy ACCEPT)
target    prot opt source               destination
Chain POSTROUTING (policy ACCEPT)
target    prot opt source               destination
ats@serverA:~$
```

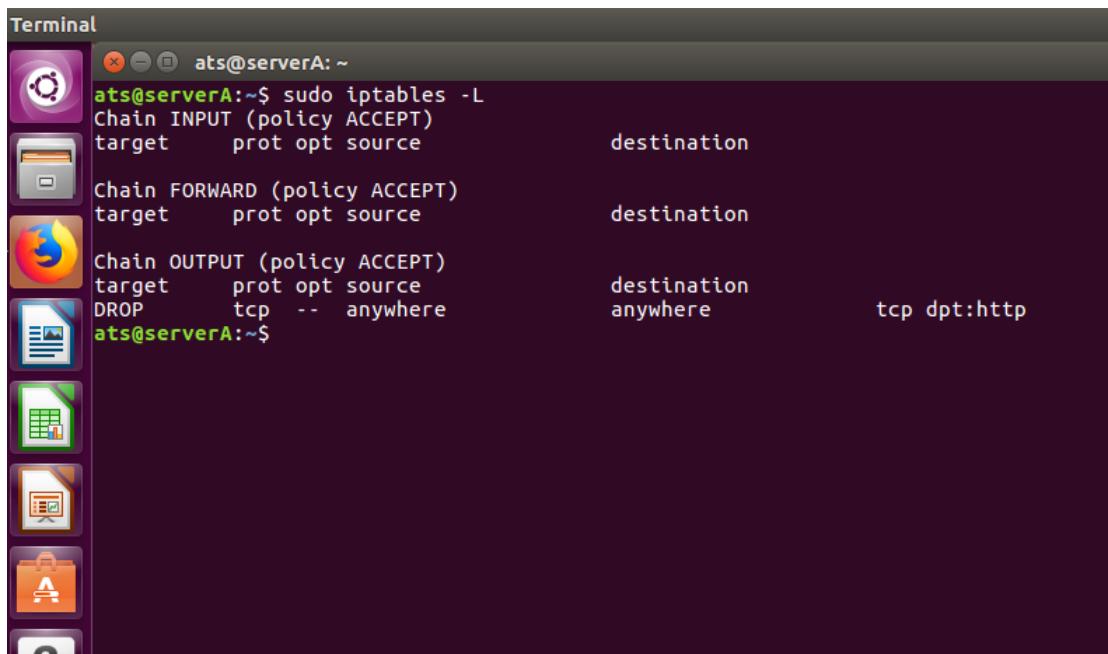


Task11:Block HTTP-browsing in the guest OS

I blocked http-browsing in the guest OS by setting OUTPUT in DROP rule.

```
ats@serverA:~$ sudo iptables -A OUTPUT -p tcp --dport 80 -j DROP  
ats@serverA:~$ sudo iptables -L
```

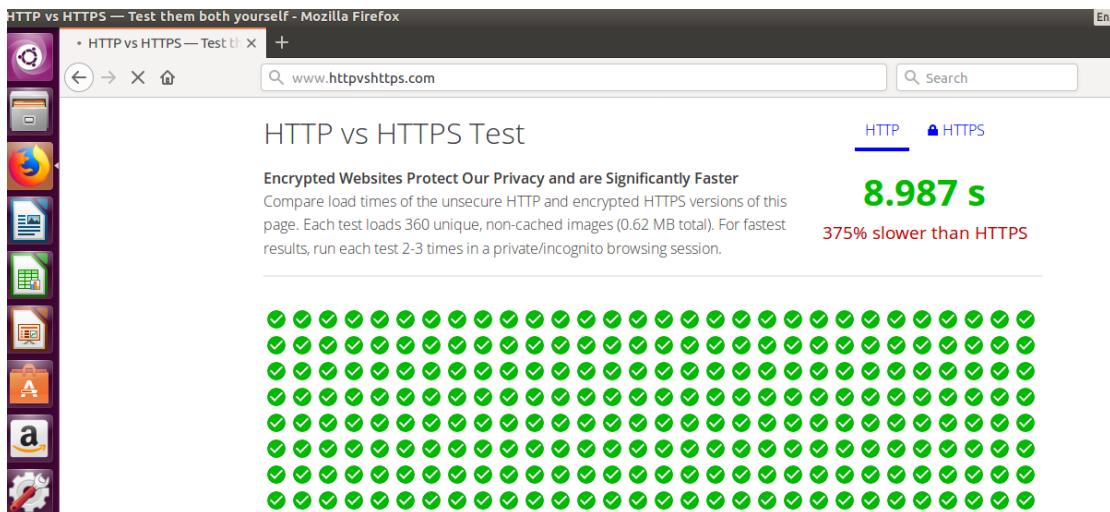
I made sure this rule have added in the chain.



I can browse in my host with HTTP.



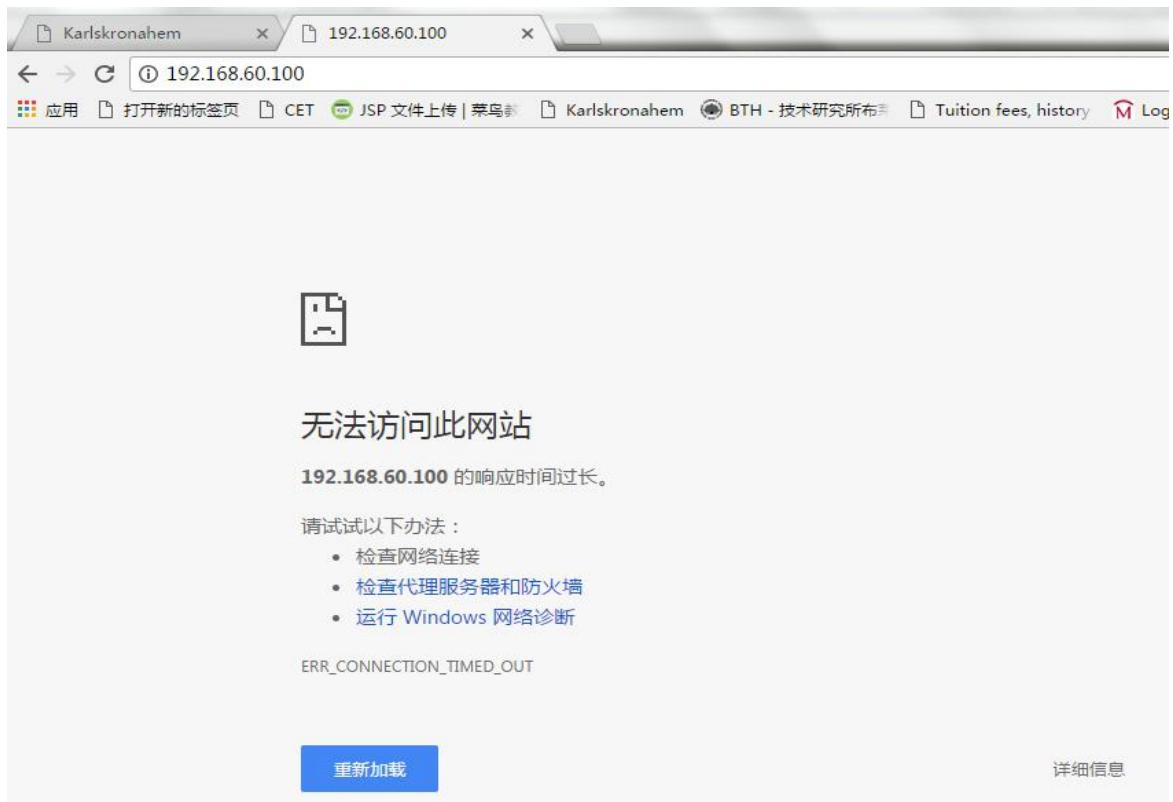
But look at my guest, it shows 375% slower than HTTPS, because it has already blocked by me.

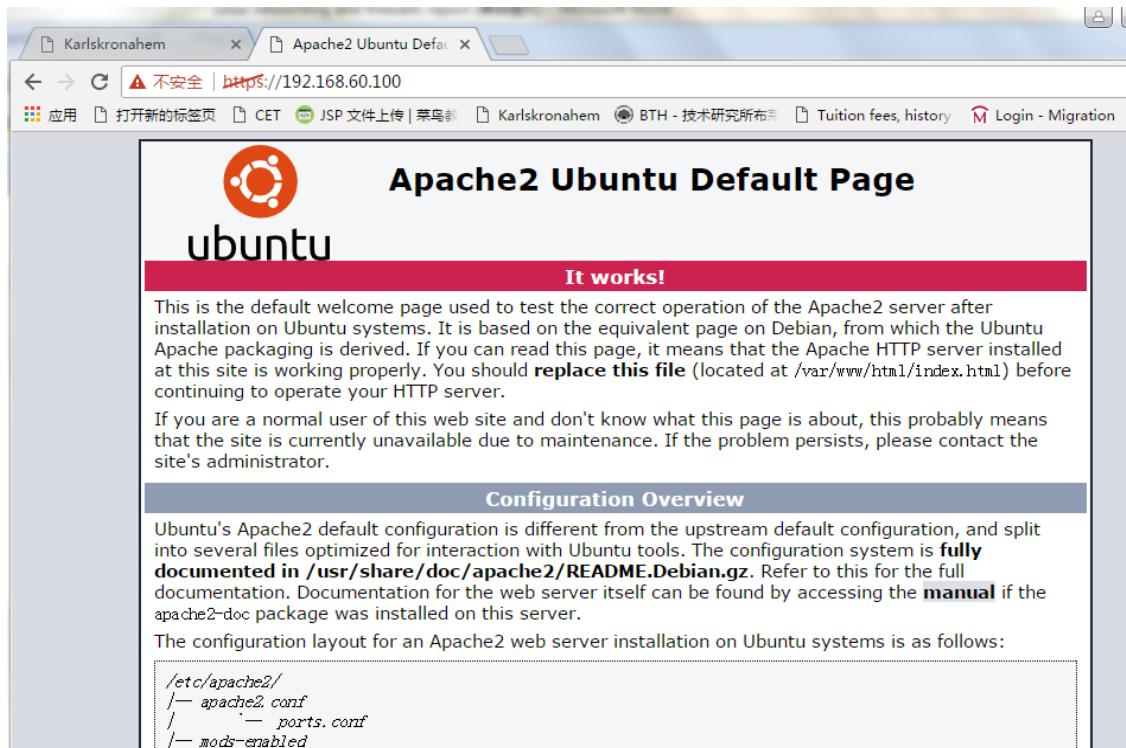


Task12:Block Apache web server from serving content over HTTP

I used DROP in INPUT chain to block only http. Http's port is 80.

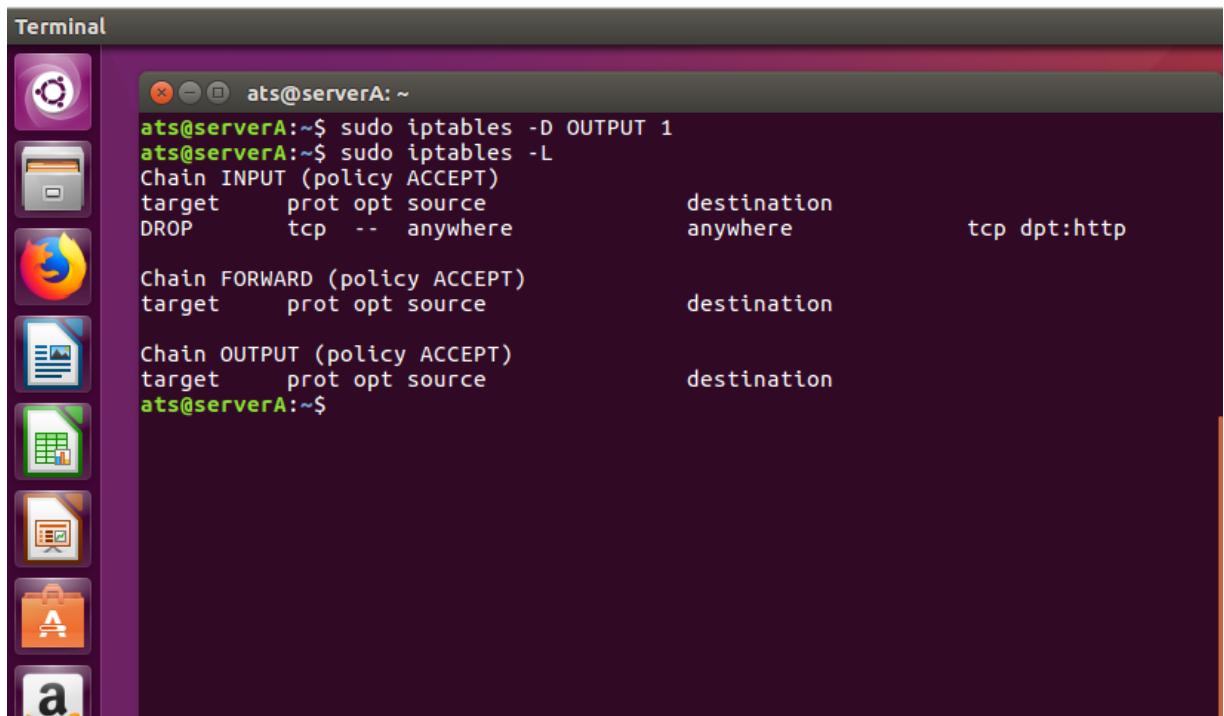
```
ats@serverA:~$ sudo iptables -A INPUT -p tcp --dport 80 -j DROP  
ats@serverA:~$ sudo iptables -L
```

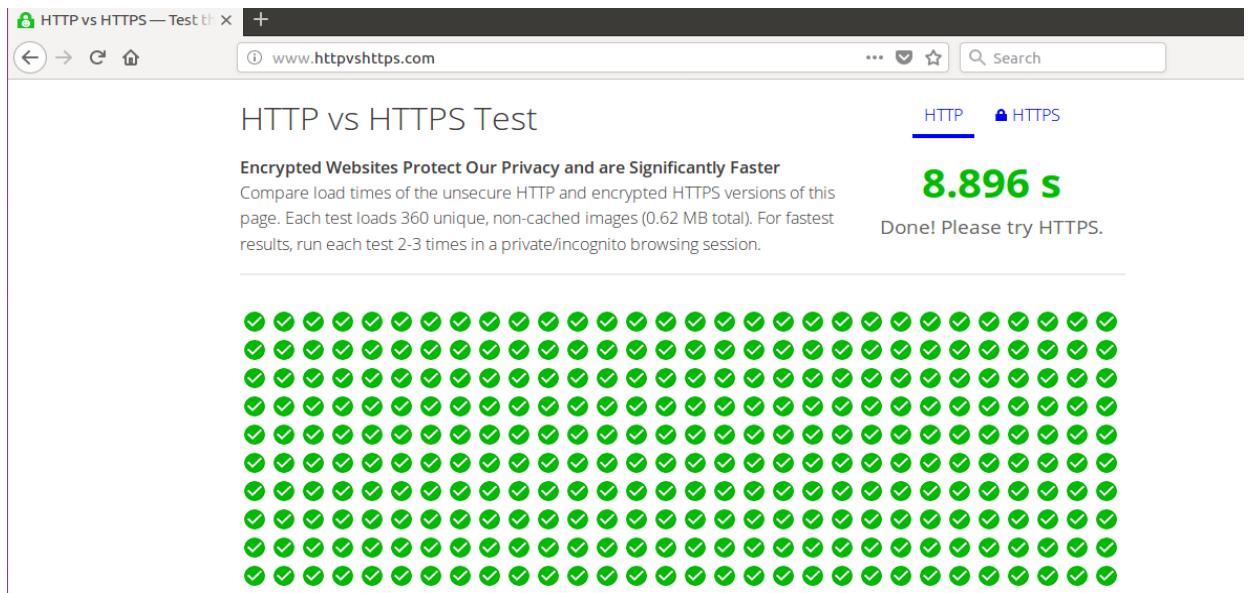




Task13:Unblock HTTP-browsing in the guest OS

Unblock the task11, I used delete -D and it is the first rule, so I wrote -D 1.





Task14:Use firewall.sh to configure the firewall

I copy the `firewall.sh.template` file at the same directory, and named it `firewall.sh`.

```
ats@serverA:~$ cp firewall.sh.template firewall.sh
```

To keep the state I had in Task13, I wrote the rule like following:

```
# Default policy is to send to a dropping chain
$IPT -t filter -P INPUT ACCEPT
$IPT -t filter -P OUTPUT ACCEPT
$IPT -t filter -P FORWARD ACCEPT

# Create logging chains
#$IPT -t filter -N input_log
#$IPT -t filter -N output_log
#$IPT -t filter -N forward_log

# Set some logging targets for DROPPED packets
#$IPT -t filter -A input_log -j LOG --log-level notice --log-prefix "input drop: "
#$IPT -t filter -A output_log -j LOG --log-level notice --log-prefix "output drop: "
#$IPT -t filter -A forward_log -j LOG --log-level notice --log-prefix "forward drop: "
#echo "Added logging"

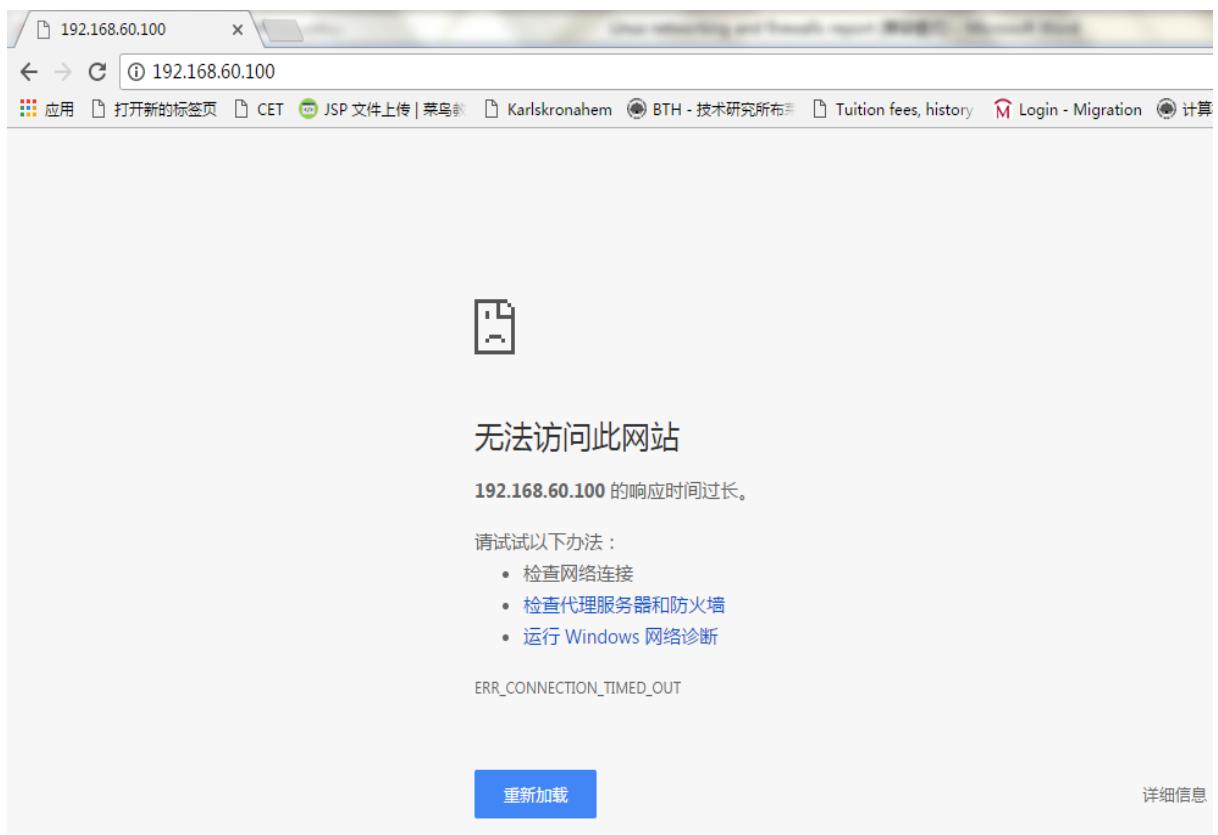
# Return from the logging chain to the built-in chain
#$IPT -t filter -A input_log -j RETURN
#$IPT -t filter -A output_log -j RETURN
#$IPT -t filter -A forward_log -j RETURN

# These rules must be inserted at the end of the built-in
# chain to log packets that will be dropped by the default
# DROP policy
#$IPT -t filter -A INPUT -j input_log
#$IPT -t filter -A OUTPUT -j output_log
#$IPT -t filter -A FORWARD -j forward_log
$IPT -A INPUT -p tcp --dport 80 -j DROP
```

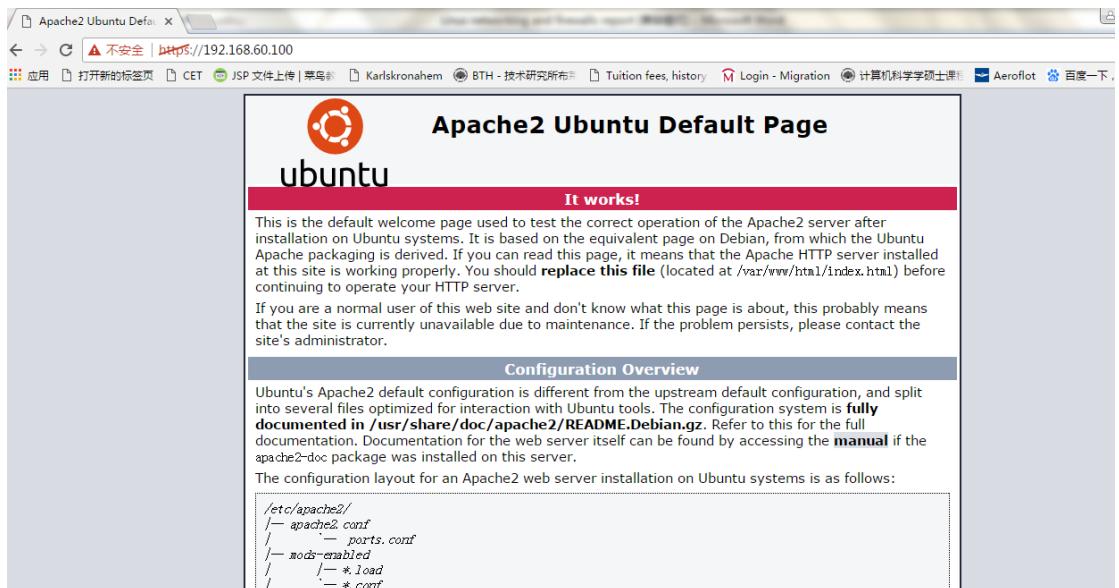
```
ats@serverA:~$ sudo ./firewall.sh
iptables v1.6.0: no command specified
Try `iptables -h` or `iptables --help` for more information.
ats@serverA:~$ sudo ./firewall.sh
ats@serverA:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
DROP      tcp   --  anywhere        anywhere          tcp dpt:http

Chain FORWARD (policy ACCEPT)
target     prot opt source          destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
ats@serverA:~$
```



Then accessed in my host ,and checking!



Task15:Change default firewall policy to DROP

```
# Default policy is to send to a dropping chain
$ IPT -t filter -P INPUT DROP
$ IPT -t filter -P OUTPUT DROP
$ IPT -t filter -P FORWARD DROP
```

A screenshot of a terminal window titled 'ats@serverA:~'. The user runs three commands: 'sudo ./firewall.sh', 'sudo iptables -L', and 'sudo iptables -Z'. The output shows the default policy for all chains (INPUT, FORWARD, OUTPUT) is set to 'DROP'.

```
ats@serverA:~$ sudo ./firewall.sh
ats@serverA:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source          destination
Chain FORWARD (policy DROP)
target     prot opt source          destination
Chain OUTPUT (policy DROP)
target     prot opt source          destination
ats@serverA:~$
```

```
管理员: C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.1.7601]
版权所有 <c> 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>ping 192.168.60.100

正在 Ping 192.168.60.100 具有 32 字节的数据:
请求超时。
请求超时。
请求超时。
请求超时。

192.168.60.100 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 0, 丢失 = 4 <100% 丢失>,
```

C:\Users\Administrator>

```
ats@serverA:~$ sudo ./firewall.sh
ats@serverA:~$ sudo ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
^C
--- 127.0.0.1 ping statistics ---
11 packets transmitted, 0 received, 100% packet loss, time 10081ms
```

Task16:Logging DROPPED packets

```
ats@serverA:~$
ats@serverA:~$ sudo iptables -t filter -L -v -n
Chain INPUT (policy DROP 2 packets, 138 bytes)
pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy DROP 23 packets, 2466 bytes)
pkts bytes target     prot opt in     out     source               destination

ats@serverA:~$
```

```

# Default policy is to send to a dropping chain
$IPT -t filter -P INPUT DROP
$IPT -t filter -P OUTPUT DROP
$IPT -t filter -P FORWARD DROP

# Create logging chains
$IPT -t filter -N input_log
$IPT -t filter -N output_log
$IPT -t filter -N forward_log

# Set some logging targets for DROPPED packets
$IPT -t filter -A input_log -j LOG --log-level notice --log-prefix "input drop: "
$IPT -t filter -A output_log -j LOG --log-level notice --log-prefix "output drop: "
$IPT -t filter -A forward_log -j LOG --log-level notice --log-prefix "forward drop: "
echo "Added logging"

# Return from the logging chain to the built-in chain
$IPT -t filter -A input_log -j RETURN
$IPT -t filter -A output_log -j RETURN
$IPT -t filter -A forward_log -j RETURN

# These rules must be inserted at the end of the built-in
# chain to log packets that will be dropped by the default
# DROP policy
$IPT -t filter -A INPUT -j input_log
$IPT -t filter -A OUTPUT -j output_log
$IPT -t filter -A FORWARD -j forward_log

```

The screenshot shows a terminal window with two sections of output. The top section shows the result of a ping command:

```

ats@serverA:~$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
^C
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 2999ms

```

The bottom section shows kernel log entries for dropped packets:

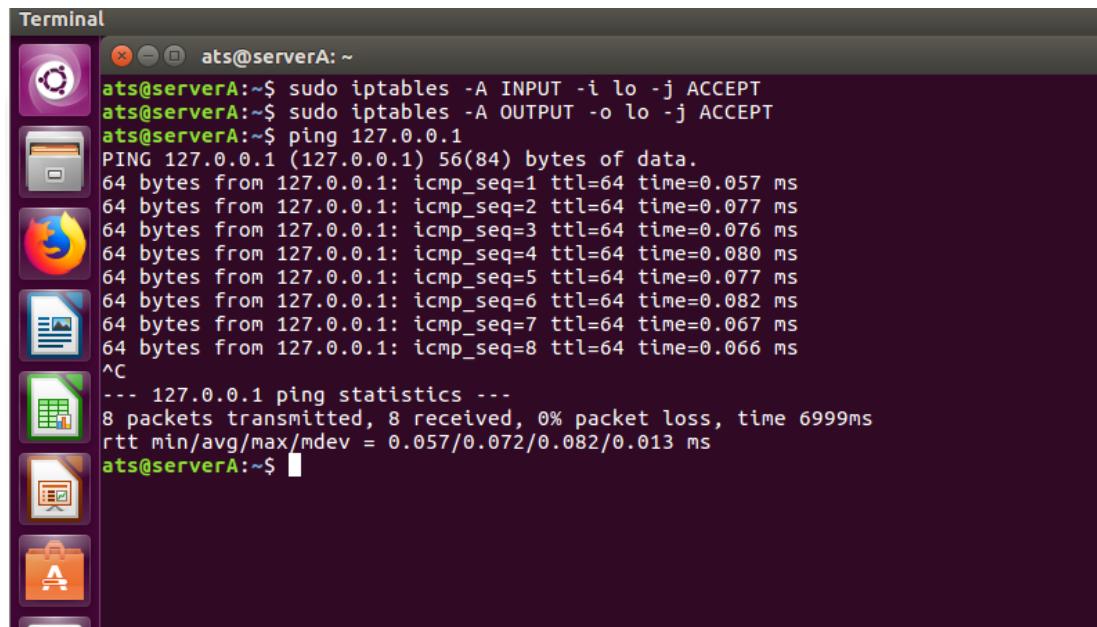
```

ats@serverA:~$ 
DST=224.0.0.251 LEN=146 TOS=0x00 PREC=0x00 TTL=255 ID=27992 DF PROTO=UDP SPT=53
T=5353 LEN=126
3 13:00:48 serverA kernel: [ 748.765974] output drop: IN= OUT=enp0s8 SRC=192.16
3 DST=224.0.0.251 LEN=146 TOS=0x00 PREC=0x00 TTL=255 ID=21300 DF PROTO=UDP SPT=5
T=5353 LEN=126
3 13:00:48 serverA kernel: [ 748.768174] output drop: IN= OUT=enp0s3 SRC=192.16
00 DST=224.0.0.251 LEN=146 TOS=0x00 PREC=0x00 TTL=255 ID=59834 DF PROTO=UDP SPT
DPT=5353 LEN=126
3 13:01:24 serverA kernel: [ 785.006668] output drop: IN= OUT=enp0s9 SRC=10.0.9
DST=10.0.98.3 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=28502 DF PROTO=UDP SPT=36354
3 LEN=40
3 13:01:24 serverA kernel: [ 785.006752] output drop: IN= OUT=enp0s9 SRC=10.0.9
DST=10.0.98.3 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=28503 DF PROTO=UDP SPT=36387
3 LEN=40
3 13:01:24 serverA kernel: [ 785.006792] output drop: IN= OUT=enp0s9 SRC=10.0.9
DST=10.0.98.3 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=28504 DF PROTO=UDP SPT=54794
3 LEN=40
3 13:01:24 serverA kernel: [ 785.006817] output drop: IN= OUT=enp0s9 SRC=10.0.9
DST=10.0.98.3 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=28505 DF PROTO=UDP SPT=37108
3 LEN=40
3 13:02:37 serverA kernel: [ 857.862463] input drop: IN=enp0s3 OUT= MAC=01:00:5
0:fb:0a:00:27:00:00:16:08:00 SRC=192.168.60.1 DST=224.0.0.251 LEN=69 TOS=0x00 P
00 TTL=255 ID=374 PROTO=UDP SPT=5353 DPT=5353 LEN=49

```

I think is the default policy is DROP and the packets are dropped, so it blocking my ping.

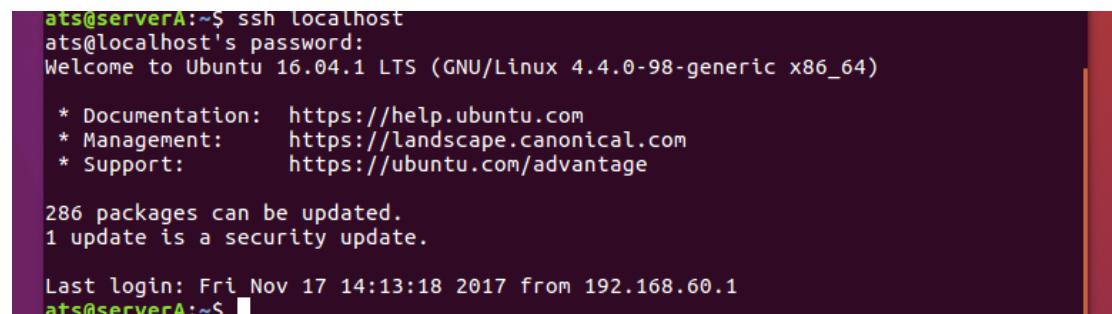
Task17:Enable traffic from loopback interface



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and contains the following command-line session:

```
ats@serverA:~$ sudo iptables -A INPUT -i lo -j ACCEPT
ats@serverA:~$ sudo iptables -A OUTPUT -o lo -j ACCEPT
ats@serverA:~$ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.057 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.077 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.076 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.080 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.077 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.082 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.067 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.066 ms
^C
--- 127.0.0.1 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 6999ms
rtt min/avg/max/mdev = 0.057/0.072/0.082/0.013 ms
ats@serverA:~$
```

```
ats@serverA:~$ sudo iptables -A OUTPUT -p tcp --dport 22 -j ACCEPT
```



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window has a dark background and contains the following command-line session:

```
ats@serverA:~$ ssh localhost
at localhost's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-98-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

286 packages can be updated.
1 update is a security update.

Last login: Fri Nov 17 14:13:18 2017 from 192.168.60.1
ats@serverA:~$
```

Task18:Allow Server A to ping the other interfaces

I added rules and ping the traffic initiated from ServerA. Because the request-echo is 8 and reply echo is 0, so I used the two numbers. And I pointed out the initiated source ip, and accept the icmp protocol.

```
#Task 18 Allow ServerA to ping the other interfaces
$IPT -A OUTPUT -s 192.168.60.100 -p icmp --icmp-type 8 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
echo "add echo ICMP Echo Request...""
$IPT -A INPUT -d 192.168.60.100 -p icmp --icmp-type 0 -m state --state ESTABLISHED,RELATED -j ACCEPT
echo "add echo ICMP Echo Reply...""

[...]
ats@serverA:~$ ping 192.168.60.100
PING 192.168.60.100 (192.168.60.100) 56(84) bytes of data.
64 bytes from 192.168.60.100: icmp_seq=1 ttl=64 time=0.152 ms
64 bytes from 192.168.60.100: icmp_seq=2 ttl=64 time=0.123 ms
64 bytes from 192.168.60.100: icmp_seq=3 ttl=64 time=0.231 ms
64 bytes from 192.168.60.100: icmp_seq=4 ttl=64 time=0.141 ms
64 bytes from 192.168.60.100: icmp_seq=5 ttl=64 time=0.136 ms
64 bytes from 192.168.60.100: icmp_seq=6 ttl=64 time=0.237 ms
64 bytes from 192.168.60.100: icmp_seq=7 ttl=64 time=0.076 ms
64 bytes from 192.168.60.100: icmp_seq=8 ttl=64 time=0.251 ms
64 bytes from 192.168.60.100: icmp_seq=9 ttl=64 time=0.232 ms
64 bytes from 192.168.60.100: icmp_seq=10 ttl=64 time=0.129 ms
64 bytes from 192.168.60.100: icmp_seq=11 ttl=64 time=0.133 ms
64 bytes from 192.168.60.100: icmp_seq=12 ttl=64 time=0.137 ms
^C
--- 192.168.60.100 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 10999ms
rtt min/avg/max/mdev = 0.076/0.164/0.251/0.057 ms
```

Task19:Allow Server A to ping all hosts

To ping the host I need it have DNS service, so I accept port 53. Then accept icmp echo.

```
#Task19 :Allow Server A to ping all host
$IPT -A OUTPUT -p udp --dport 53 -j ACCEPT
$IPT -A INPUT -p udp --sport 53 -j ACCEPT
$IPT -A INPUT -p udp --dport 53 -j ACCEPT
$IPT -A OUTPUT -p udp --sport 53 -j ACCEPT

$IPT -A OUTPUT -p icmp --icmp-type 8 -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type 0 -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type 8 -j ACCEPT
$IPT -A OUTPUT -p icmp --icmp-type 0 -j ACCEPT
```

```

ats@serverA:~$ sudo ./firewall.sh
Added logging
ats@serverA:~$ ping bth.se
PING bth.se (213.52.129.125) 56(84) bytes of data.
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=1 ttl=47 time=113 ms
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=2 ttl=47 time=40.6 ms
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=3 ttl=47 time=42.6 ms
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=4 ttl=47 time=42.7 ms
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=5 ttl=47 time=42.3 ms
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=6 ttl=47 time=47.0 ms
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=7 ttl=47 time=113 ms
64 bytes from c6386.cloudnet.se (213.52.129.125): icmp_seq=8 ttl=47 time=57.4 ms
^C
--- bth.se ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7014ms
rtt min/avg/max/mdev = 40.660/62.545/113.820/29.924 ms
ats@serverA:~$ nslookup bth.se
Server:      10.0.98.3
Address:     10.0.98.3#53

Non-authoritative answer:
Name:   bth.se
Address: 213.52.129.125

ats@serverA:~$ host bth.se
bth.se has address 213.52.129.125
bth.se has IPv6 address 2a01:7e00::f03c:91ff:fe18:15af
bth.se mail is handled by 20 bth-se.mail.protection.outlook.com.

```

Task20:Enable stateful firewall

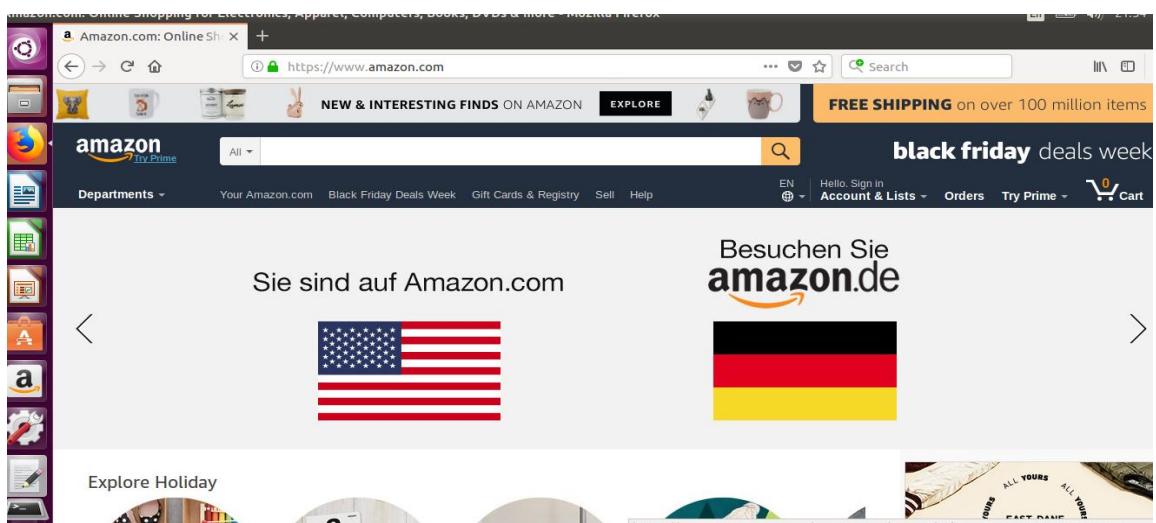
Stateful packet can supply stateful packet inspection. I added rules to enable TCP connection to be established to any destination, so used the sentence “—ctstate NEW,ESTABLISHED,RELATED”.

```

#Task20:Enable stateful firewall
$IPT -A OUTPUT -p udp --dport 53 -j ACCEPT
$IPT -A INPUT -p udp --sport 53 -j ACCEPT|
$IPT -A INPUT -p udp --dport 53 -j ACCEPT
$IPT -A OUTPUT -p udp --sport 53 -j ACCEPT

$IPT -A OUTPUT -p icmp --icmp-type 8 -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type 0 -j ACCEPT
$IPT -A INPUT -p icmp --icmp-type 8 -j ACCEPT
$IPT -A OUTPUT -p icmp --icmp-type 0 -j ACCEPT
$IPT -t filter -A INPUT -p tcp -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT
$IPT -t filter -A FORWARD -p tcp -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT
$IPT -t filter -A OUTPUT -p tcp -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT

```



Task21:Enables SSH and HTTPS content from apache2 server for web browser on host

Just enable ssh and https, then I thought the multiport should be used.

```
#Task21:Enable SSH and HTTPS content from apache2 server for web browser on host
$ IPT -A INPUT -p tcp -m multiport --dport 22,443 -j ACCEPT
$ IPT -A OUTPUT -p tcp -m multiport --sport 22,443 -j ACCEPT
```

```
ats@serverA:~$ sudo ./firewall.sh
Added logging
ats@serverA:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
input_log  all  --  anywhere             anywhere
ACCEPT    tcp  --  anywhere             anywhere            multiport dports ssh,https

Chain FORWARD (policy DROP)
target     prot opt source               destination
forward_log all  --  anywhere             anywhere

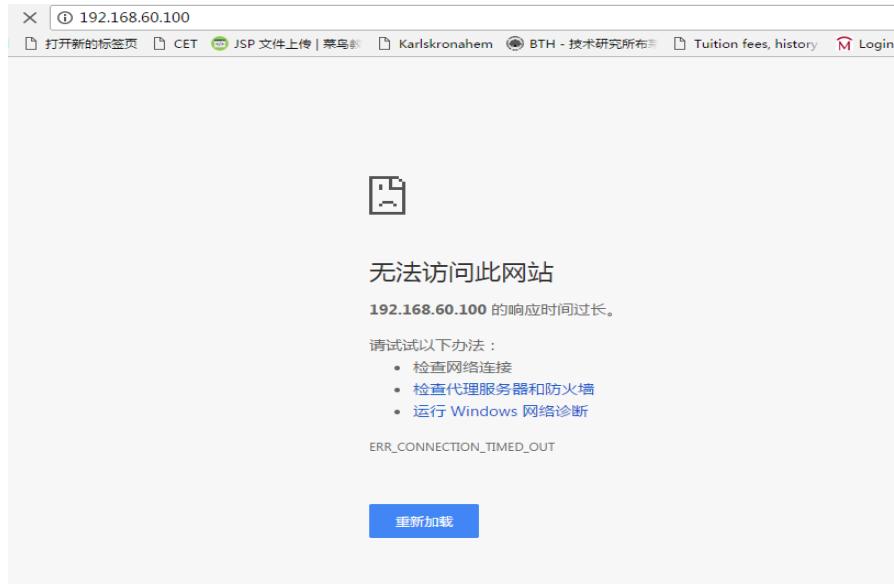
Chain OUTPUT (policy DROP)
target     prot opt source               destination
output_log all  --  anywhere             anywhere
ACCEPT    tcp  --  anywhere             anywhere            multiport sports ssh,https

Chain forward_log (1 references)
target     prot opt source               destination
LOG       all  --  anywhere             anywhere           LOG level notice prefix "forward drop: "
RETURN   all  --  anywhere             anywhere

Chain input_log (1 references)
target     prot opt source               destination
```

Firefox Browser Screenshot:

- Address bar: http://192.168.60.100
- Second tab: http://192.168.60.100 - 192.168.60.100
- Third tab: https://www.facebook.com - (1) Facebook



This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

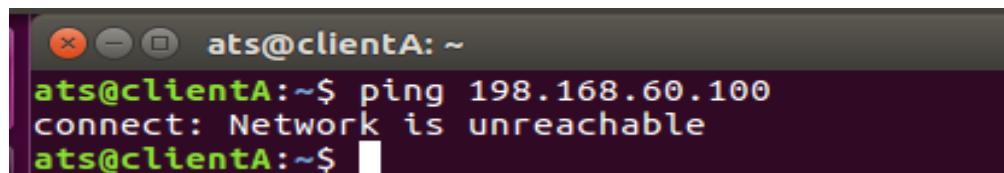
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
└── apache2.conf
    ├── ports.conf
    └── mods-enabled
        ├── *.load
        └── *.conf
    └── conf-enabled
        ├── *.conf
        └── sites-enabled
            └── *.conf
```

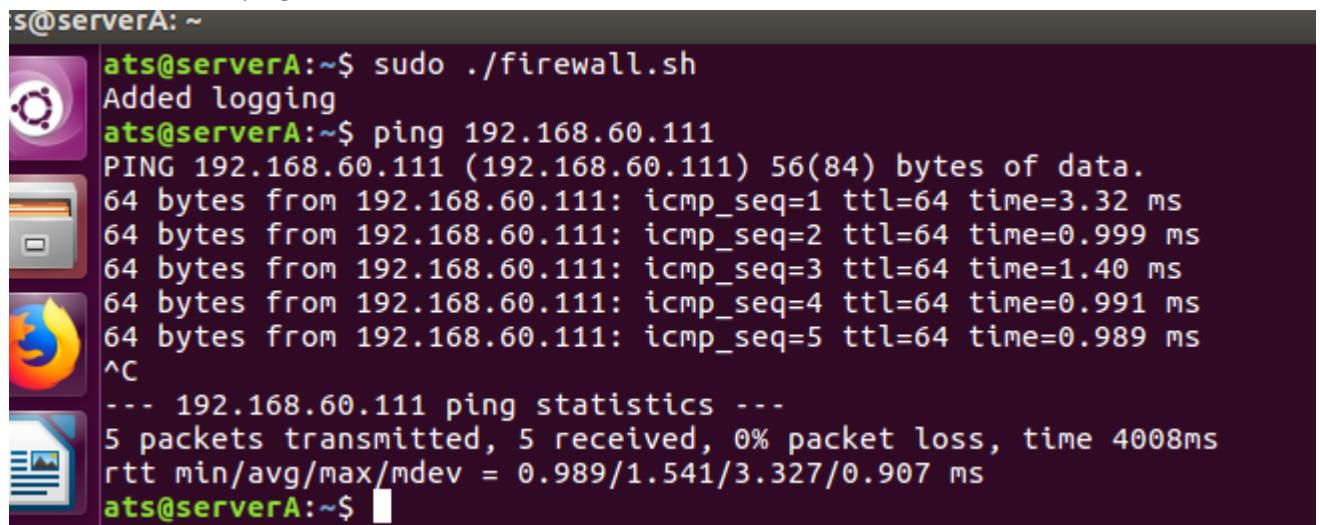
Task22:Ping Server A from Client A

First time, I tried ping serverA from clientA to verified the lab instruction, that's true, it was unsuccessful.



```
ats@clientA:~$ ping 198.168.60.100
connect: Network is unreachable
ats@clientA:~$
```

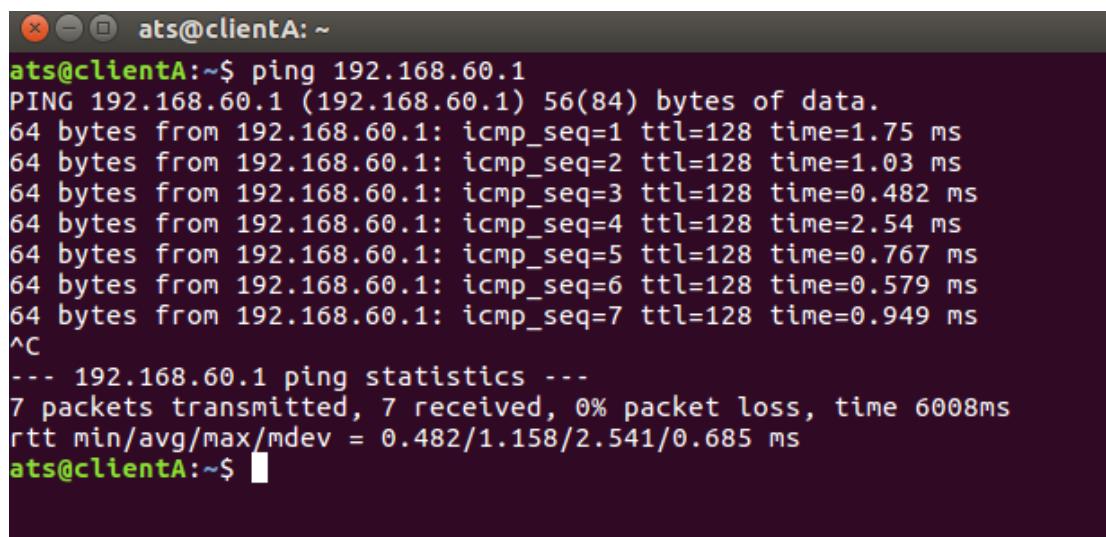
But serverA can ping ClientA.



```
s@serverA:~$ ats@serverA:~$ sudo ./firewall.sh
Added logging
ats@serverA:~$ ping 192.168.60.111
PING 192.168.60.111 (192.168.60.111) 56(84) bytes of data.
64 bytes from 192.168.60.111: icmp_seq=1 ttl=64 time=3.32 ms
64 bytes from 192.168.60.111: icmp_seq=2 ttl=64 time=0.999 ms
64 bytes from 192.168.60.111: icmp_seq=3 ttl=64 time=1.40 ms
64 bytes from 192.168.60.111: icmp_seq=4 ttl=64 time=0.991 ms
64 bytes from 192.168.60.111: icmp_seq=5 ttl=64 time=0.989 ms
^C
--- 192.168.60.111 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 0.989/1.541/3.327/0.907 ms
ats@serverA:~$
```

Then I thought it's the firewall rules in ServerA impeded the ClientA to ping, so I pointed out the specific IP address and tried whether it would work.

```
#Task22:Ping ServerA from ClientA
$IPT -A OUTPUT -s 192.168.60.111 -d 192.168.60.100 -p tcp -j ACCEPT
```



```
ats@clientA:~$ ping 192.168.60.1
PING 192.168.60.1 (192.168.60.1) 56(84) bytes of data.
64 bytes from 192.168.60.1: icmp_seq=1 ttl=128 time=1.75 ms
64 bytes from 192.168.60.1: icmp_seq=2 ttl=128 time=1.03 ms
64 bytes from 192.168.60.1: icmp_seq=3 ttl=128 time=0.482 ms
64 bytes from 192.168.60.1: icmp_seq=4 ttl=128 time=2.54 ms
64 bytes from 192.168.60.1: icmp_seq=5 ttl=128 time=0.767 ms
64 bytes from 192.168.60.1: icmp_seq=6 ttl=128 time=0.579 ms
64 bytes from 192.168.60.1: icmp_seq=7 ttl=128 time=0.949 ms
^C
--- 192.168.60.1 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6008ms
rtt min/avg/max/mdev = 0.482/1.158/2.541/0.685 ms
ats@clientA:~$
```

I learned some information about ClientA.

```
ats@clientA:~$ sudo ifconfig
enp0s3    Link encap:Ethernet HWaddr 08:00:27:f2:5d:0e
          inet addr:192.168.60.111 Bcast:192.168.60.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe2:5d0e/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:318 errors:0 dropped:0 overruns:0 frame:0
            TX packets:280 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:31147 (31.1 KB) TX bytes:35099 (35.0 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING MTU:65536 Metric:1
            RX packets:13460 errors:0 dropped:0 overruns:0 frame:0
            TX packets:13460 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
            RX bytes:995904 (995.9 KB) TX bytes:995904 (995.9 KB)

ats@clientA:~$
```



```
ats@clientA:~$ sudo netstat -4 -rn
Kernel IP routing table
Destination     Gateway         Genmask        Flags   MSS Window irtt Iface
169.254.0.0     0.0.0.0       255.255.0.0   U        0 0          0 enp0s3
192.168.60.0    0.0.0.0       255.255.255.0 U        0 0          0 enp0s3
```

Task23:SSH from Client A to Server A

Yes, ServerA can ssh ClientA.

```
ats@serverA:~$ ssh 192.168.60.111
ats@192.168.60.111's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-47-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

Last login: Thu Nov 17 00:51:35 2016 from 10.0.99.2
ats@clientA:~$
```

How to let clientA ssh serverA, I also figured it's the firewall rules' problem, then I tried to let serverA accept port 22, ssh default port is 22. Then it works.

```

ats@clientA:~$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
ats@clientA:~$ sudo iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT
ats@clientA:~$ ssh 192.168.60.100
The authenticity of host '192.168.60.100 (192.168.60.100)' can't be established.
ECDSA key fingerprint is SHA256:W+LPjhGRAjAU6ZmmVMzlgjvytXF4mC2eXKlDqKC505U.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.60.100' (ECDSA) to the list of known hosts.
ats@192.168.60.100's password:
Permission denied, please try again.
ats@192.168.60.100's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-101-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

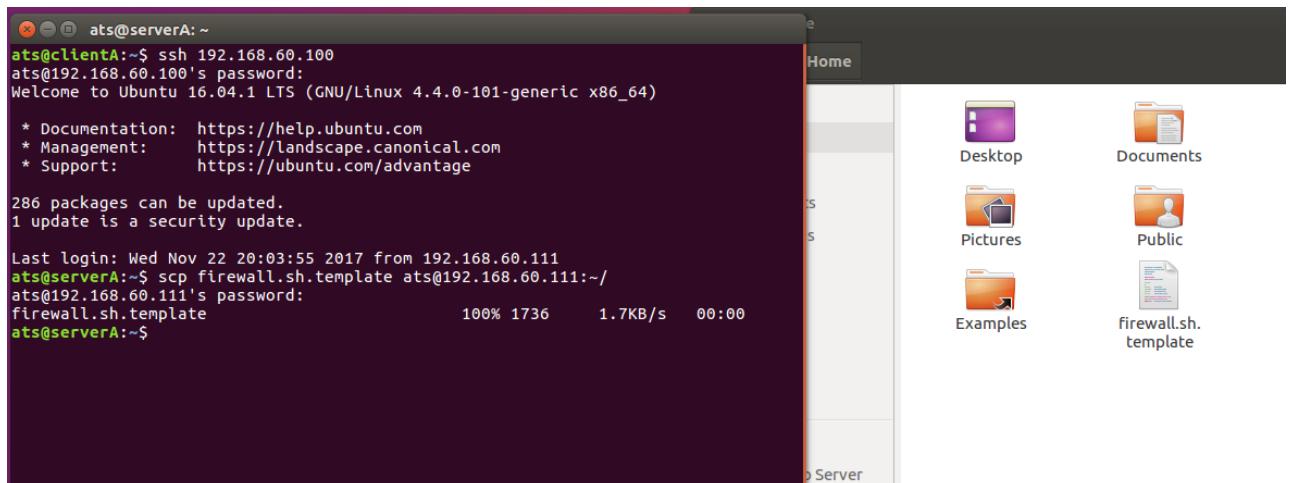
286 packages can be updated.
1 update is a security update.

Last login: Mon Nov 20 22:43:41 2017 from 192.168.60.1

```

截图(Alt + A)

And followed the instruction,I tried the scp command.



Task24:Add gateway and DNS server to Client A

The following screenshot I used gedit command to add gateway and DNS for ClientA. The adding order is really important, at first time I added them before clientA's IP address and netmask, then check in route -n, you will not see the gateway. And such simple problem actually stucked me for several days.

```

ats@clientA:~$ sudo gedit /etc/network/interfaces

** (gedit:2165): WARNING **: Set document metadata failed: Setting attribute met
adata::gedit-position not supported
ats@clientA:~$ 

```

```

# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

# Host-only interface
auto enp0s3
iface enp0s3 inet static
    address 192.168.60.111
    netmask 255.255.255.0
    gateway 192.168.60.100
    dns-nameservers 10.0.98.3

```

Task25:Enable IP forwarding on Server A

```

ats@serverA:~$ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
ats@serverA:~$ sudo sysctl -p
sudo: sysctl: command not found
ats@serverA:~$ sudo sysctl -p
ats@serverA:~$ 

```

I used wireshark to narrow down the problems, it showed "who has 192.168.60.100? Tell 192.168.60.111", and also showed ARP problem. ARP protocol is to analyse the IP address to MAC address which using in DataLink Layer. I configured the ClientA didn't know the ARP of ServerA, so I added the rules in clientA's terminal.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.60.111	10.0.98.3	DNS	74	Standard query 0x3bef A www.google.com
2	5.005655056	192.168.60.111	10.0.98.3	DNS	74	Standard query 0x3bef A www.google.com
3	5.011045229	CadmusCo_f2:5d:0e	CadmusCo_40:a4:b9	ARP	42	Who has 192.168.60.100? Tell 192.168.60.111
4	5.011977216	CadmusCo_40:a4:b9	CadmusCo_f2:5d:0e	ARP	60	192.168.60.100 is at 08:00:27:40:a4:b9
5	10.011800865	192.168.60.111	10.0.98.3	DNS	74	Standard query 0x7270 A www.google.com
6	15.014666622	192.168.60.111	10.0.98.3	DNS	74	Standard query 0x7270 A www.google.com
7	21.155808158	192.168.60.1	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
8	22.155870861	192.168.60.1	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
9	23.156007868	192.168.60.1	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
10	24.156774547	192.168.60.1	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1

► Frame 3: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
 ► Ethernet II, Src: CadmusCo_f2:5d:0e (08:00:27:f2:5d:0e), Dst: CadmusCo_40:a4:b9 (08:00:27:40:a4:b9)
 ► Address Resolution Protocol (request)

```

ats@clientA:~$ route -n
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref  Use Iface
0.0.0.0          192.168.60.100  0.0.0.0        UG    0      0      0  enp0s3
169.254.0.0      0.0.0.0        255.255.0.0    U     1000   0      0  enp0s3
192.168.60.0     0.0.0.0        255.255.255.0  U     0      0      0  enp0s3
ats@clientA:~$ sudo arp -s 192.168.60.100 08:00:27:40:a4:b9
[sudo] password for ats:
ats@clientA:~$ 

```

Task26:Change iptables to forward packets

```
$IF=enp0s3  
$IP=192.168.60.100  
$NF=enp0s9  
$NP=10.0.98.100
```

Let's Accept FORWARD traffic:

```
#Task26:Change iptables to forward packets  
$IPT -t filter -A FORWARD -i $IF -j ACCEPT  
$IPT -t filter -A FORWARD -i $NF -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

Task27:Enable SNAT on Server A

Let all the traffic from ClientA go to ServerA, then ServerA let this traffic go to external network by using external gateway, the both VMs surf net through the host.

```
#Task27:Enable SNAT on Server A  
$IPT -t nat -A POSTROUTING -j SNAT -o $NF --to $NP
```

```
ats@clientA:~$ sudo /etc/init.d/networking restart  
[ ok ] Restarting networking (via systemctl): networking.service.  
ats@clientA:~$ ping www.google.com  
PING www.google.com (216.58.209.132) 56(84) bytes of data.  
64 bytes from arn09s05-in-f132.1e100.net (216.58.209.132): icmp_seq=1 ttl=51 time=224 ms  
64 bytes from arn09s05-in-f132.1e100.net (216.58.209.132): icmp_seq=2 ttl=51 time=16.9 ms  
64 bytes from arn09s05-in-f132.1e100.net (216.58.209.132): icmp_seq=3 ttl=51 time=17.9 ms  
^C  
--- www.google.com ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2003ms  
rtt min/avg/max/mdev = 16.988/86.394/224.230/97.465 ms
```

Then, I can access the www.google.com!

