

第二小题：逻辑隔离（10 分）

一、实验背景

云平台服务器上的不同虚拟服务器，分属于不同的用户。用户远程登录自己的虚拟服务器之后，安全上不允许直接访问同一局域网的其他虚拟服务器。

二、实验目的

搭建简单网络，通过逻辑隔离的方法，实现用户能远程登录局域网内自己的虚拟内服务器，同时不允许直接访问同一局域网的其他虚拟服务器。

三、实验环境搭建

如图 1-1 所示，我们会创建一个基于 OpenFlow Switch 的网络实验平台。两台装有 NetFPGA 的主机 PC-A 和 PC-B 实现 OpenFlow Switch 的功能，图中的 PC-0 是客户端 PC 机，PC-1 和 PC-2 分别作为服务器 A 和服务器 B，跟 OpenFlow Switch 相连，分别连接 NetFPGA 的 nf2c0 和 nf2c3 端口；而 PC-C 则是实现 OpenFlow Switch Controller 的功能，在另一链路上利用 OpenFlow Protocol，与 OpenFlow Switch 进行通信，对 Switch 的 Flow Table 进行控制。表 1-1 为网络实验环境的硬件配置与软件环境以及参数配置。

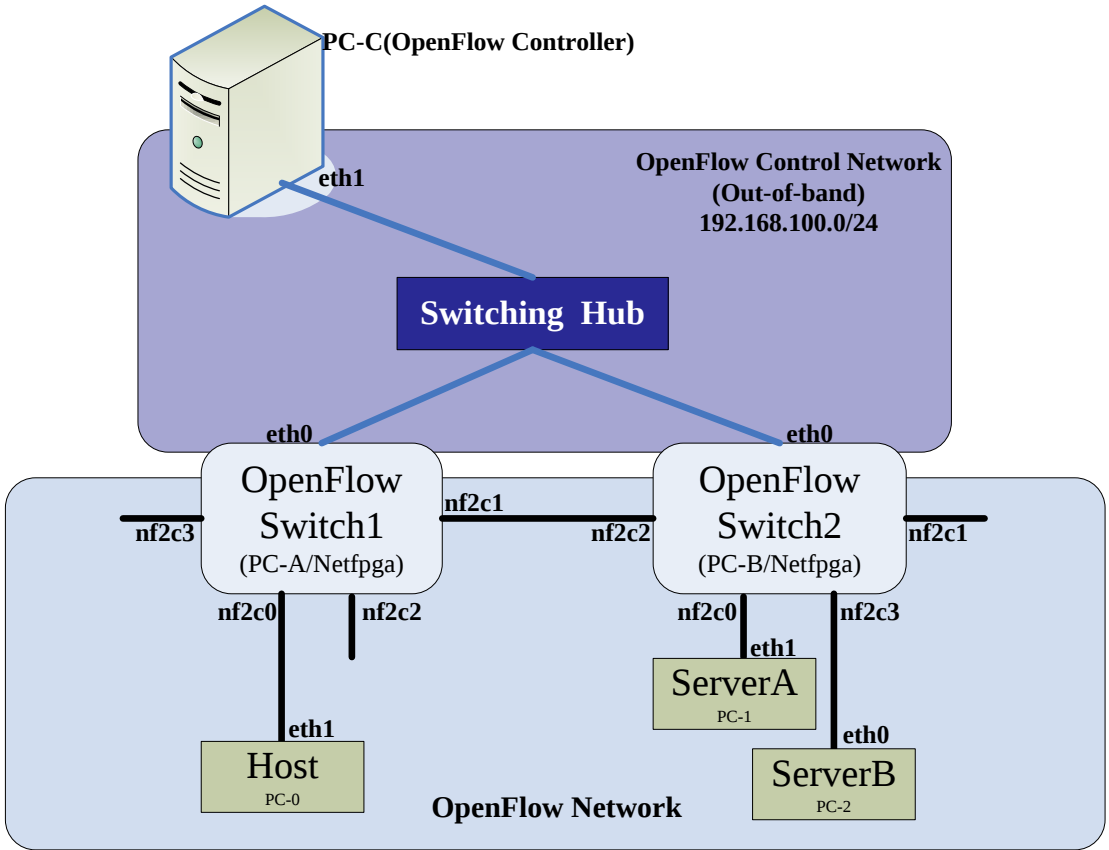


图 1-1 基于 NetFpga 的 Openflow Switch 网络物理实验平台

表 1-1 网络实验环境的硬件配置与软件环境以及参数配置

物理主机	角色	操作系统	内存	IP
PC-A	OpenFlow Switch 1 With Netfpga	Centos 5.6 32 位	2G	Eth1:192.168.100.11/24 Nf2c0:192.168.111.254/24
PC-B	OpenFlow Switch 2 With Netfpga	Centos 5.6 32 位	2G	Eth1:192.168.100.22/24 Nf2c0:192.168.122.254/24 Nf2c1:192.168.133.254/24
PC-C	OpenFlow Controller (采用 POX ¹)	Ubuntu 10.04 32 位	2G	Eth1:192.168.100.100/24
PC-0	Host	Federal 13 32 位	2G	Eth1:192.168.111.201 MAC:68:05:CA:08:1F:C4
PC-1	ServerA	Federal 13 32 位	2G	Eth1:192.168.122.202 MAC:14:CF:92:F9:FF:6B
PC-2	ServerB	Centos 5.6 32 位	2G	Eth0:192.168.133.203 MAC:1C:6F:65:49:FC:3F

四、实验过程及结果

(一) OpenFlow 实验平台初始化。

1.PC-A 初始化

#IP 配置(ifconfig)

ifconfig eth0 192.168.100.11

ifconfig nf2c0 192.168.111.254

#初始化版卡

/usr/local/sbin/cpci_reprogram.pl -all

#下载 OpenFlow bit 文件

nf_download /usr/local/netfpga/bitfiles/openflow_switch.bit

#设置 openflow datapath(路径 /usr/local/netfpga/openflow/udatapath)

cd /usr/local/netfpga/openflow/udatapath

./ofdatapath --detach punix:/var/run/dp0 -d 40169FF344C0 -i nf2c0,nf2c1,nf2c2,nf2c3

#与远程 Controller 进行通信(路径/usr/local/netfpga/openflow/secchan)

cd /usr/local/netfpga/openflow/secchan

./ofprotocol unix:/var/run/dp0 tcp:192.168.100.100:6633

#显示流表信息

cd /usr/local/netfpga/openflow/utilities

./dpctl dump-flows unix:/var/run/dp0

¹ 本实验用的是 POX+POXDesk。POXDesk 是 POX 的一个组件，增加了 web UI，新增显示 switch 和 host 组成的网络拓扑、显示 switch 的流表等功能。下载地址：https://github.com/09zwcubpt/undergrad_thesis

2.PC-B 初始化

#IP 配置(ifconfig)

```
ifconfig eth0 192.168.100.22
ifconfig nf2c0 192.168.122.254
ifconfig nf2c3 192.168.133.254
```

#初始化版卡

```
/usr/local/sbin/cpci_reprogram.pl -all
```

#下载 OpenFlow bit 文件

```
nf_download /usr/local/netfpga/bitfiles/openflow_switch.bit
```

#设置 openflow datapath(路径 /usr/local/netfpga/openflow/udatapath)

```
cd /usr/local/netfpga/openflow/udatapath
./ofdatapath --detach punix:/var/run/dp0 -d 40169FF344C0 -i nf2c0,nf2c1,nf2c2,nf2c3
```

#与远程 Controller 进行通信(路径/usr/local/netfpga/openflow/secchan)

```
cd /usr/local/netfpga/openflow/secchan
./ofprotocol unix:/var/run/dp0 tcp:192.168.100.100:6633
```

#显示流表信息

```
cd /usr/local/netfpga/openflow/utilities
./dpctl dump-flows unix:/var/run/dp0
```

3.PC-C 初始化

#IP 配置(ifconfig)

```
ifconfig eth1 192.168.100.100
```

POX 下发流表环境配置

```
./pox.py samples.pretty_log web messenger \
messenger.log_service messenger.ajax_transport openflow.of_service \
poxdesk openflow.discovery poxdesk.tinytopo \
openflow.of_01 --address=192.168.100.100 --port=6633 py
```

如图 4-1 所示：

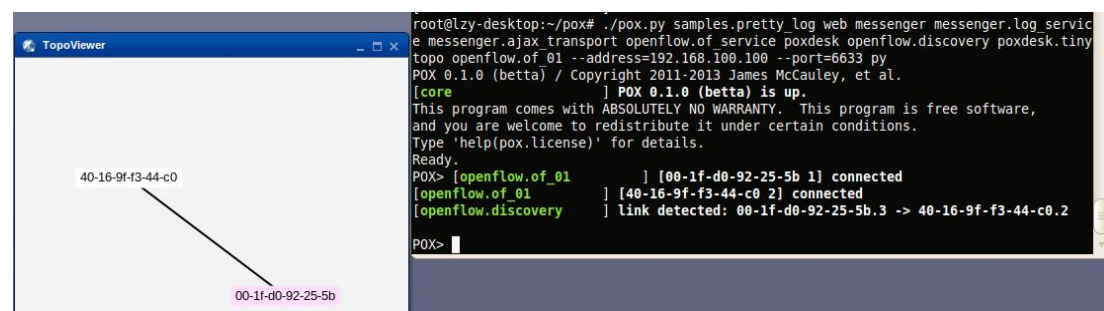
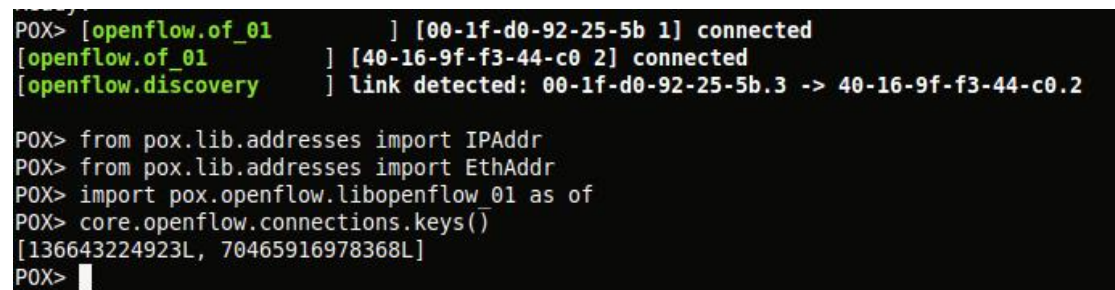


图 4-1 POX 启动并与 switches 连接上

```
#导入 MAC, IP, 核心模块
from pox.lib.addresses import IPAddr
from pox.lib.addresses import EthAddr
import pox.openflow.libopenflow_01 as of

#获取连接控制端的 openflow switch 的 key[Switch2,Switch1]
core.openflow.connections.keys()
[136643224923L, 70465916978368L]
```

如图 4-2 所示:



```
POX> [openflow.of_01] [00-1f-d0-92-25-5b 1] connected
[openflow.of_01] [40-16-9f-f3-44-c0 2] connected
[openflow.discovery] link detected: 00-1f-d0-92-25-5b.3 -> 40-16-9f-f3-44-c0.2

POX> from pox.lib.addresses import IPAddr
POX> from pox.lib.addresses import EthAddr
POX> import pox.openflow.libopenflow_01 as of
POX> core.openflow.connections.keys()
[136643224923L, 70465916978368L]
POX> █
```

图 4-2 获取连接控制端的 openflow switches 的 keys

(二) 实验环境流表配置, 通过 Controller 下发流表给 PC-A 与 PC-B, 使得 PC-0 机可以登陆服务器 A 和 B。

通过 Controller POX 下发流表:

```
#Openflow switch1 流表项
ofs1_msg1=of.ofp_flow_mod(command=0)
ofs1_msg1.priority=3
ofs1_msg1.match.in_port=1
ofs1_msg1.match.dl_type=0x0800
ofs1_msg1.actions.append(of.ofp_action_output(port=2))
core.openflow.connections[70465916978368L].send(ofs1_msg1)

ofs1_msg2=of.ofp_flow_mod(command=0)
ofs1_msg2.priority=3
ofs1_msg2.match.in_port=2
ofs1_msg2.match.dl_type=0x0800
ofs1_msg2.actions.append(of.ofp_action_dl_addr.set_dst("68:05:CA:08:1F:C4"))
ofs1_msg2.actions.append(of.ofp_action_output(port=1))
core.openflow.connections[70465916978368L].send(ofs1_msg2)
```

如图 4-3 所示:

```

POX> [openflow.of_01 ] [00-1f-d0-92-25-5b 1] connected
[openflow.of_01 ] [40-16-9f-f3-44-c0 2] connected
[openflow.discovery ] link detected: 00-1f-d0-92-25-5b.3 -> 40-16-9f-f3-44-c0.2

POX> from pox.lib.addresses import IPAddr
POX> from pox.lib.addresses import EthAddr
POX> import pox.openflow.libopenflow_01 as of
POX> core.openflow.connections.keys()
[136643224923L, 70465916978368L]
POX> ofs1_msg1=of.ofp_flow_mod(command=0)
POX> ofs1_msg1.priority=3
POX> ofs1_msg1.match.in_port=1
POX> ofs1_msg1.match.dl_type=0x0800
POX> ofs1_msg1.actions.append(of.ofp_action_output(port=2))
POX> core.openflow.connections[70465916978368L].send(ofs1_msg1)
POX>
POX> ofs1_msg2=of.ofp_flow_mod(command=0)
POX> ofs1_msg2.priority=3
POX> ofs1_msg2.match.in_port=2
POX> ofs1_msg2.match.dl_type=0x0800
POX> ofs1_msg2.actions.append(of.ofp_action_dl_addr.set_dst("68:05:CA:08:1F:C4"))
POX> ofs1_msg2.actions.append(of.ofp_action_output(port=1))
POX> core.openflow.connections[70465916978368L].send(ofs1_msg2)
POX>

```

在 Controller 上，下发流表到 Switch1 上

图 4-3 通过 POX 向 switch 1 下发流表项

POX 上 Switch1 如图 4-4 所示：

40-16-9f-f3-44-c0 - TableViewer						
Switch		Operations				
port	dl_src	dl_dst	dl_type	nw_src	nw_dst	actions
2			IP			SET_DL_DST,OUTPUT:1
1			IP			OUTPUT:2
		01:23:20:00:00:01	LLDP	0.0.0.0	0.0.0.0	OUTPUT:OFPP_CONTROLLER

图 4-4 POX 上的 Switch 1

PC-A 上 Switch1 上流表如图 4-5 所示：

```

lzy@localhost: /usr/local/netfpga/openflow/utilities
File Edit View Terminal Tabs Help

stats_reply (xid=0x3f07cc08): flags=none type=1(flow)
  cookie=0, duration_sec=176s, duration_nsec=808000000s, table_id=0, priority=3,
  n_packets=386, n_bytes=37828, idle_timeout=0,hard_timeout=0,ip,in_port=2,action
s=mod_dl_dst:68:05:ca:08:1f:c4,output:1
  cookie=0, duration_sec=179s, duration_nsec=156000000s, table_id=0, priority=3,
  n_packets=485, n_bytes=47530, idle_timeout=0,hard_timeout=0,ip,in_port=1,action
s=output:2
  cookie=0, duration_sec=361s, duration_nsec=100000000s, table_id=2, priority=65
000, n_packets=70, n_bytes=4200, idle_timeout=0,hard_timeout=0,dl_dst=01:23:20:0
0:00:01,dl_type=0x88cc,nw_src=0.0.0.0,nw_dst=0.0.0.0,nw_tos=0x00,nw_proto=0,tp_s
rc=0,tp_dst=0,actions=CONTROLLER:65535
[root@localhost utilities]# ./dpctl dump-flows unix:/var/run/dp0
stats_reply (xid=0xda77f041): flags=none type=1(flow)
  cookie=0, duration_sec=584s, duration_nsec=262000000s, table_id=0, priority=3,
  n_packets=1630, n_bytes=161969, idle_timeout=0,hard_timeout=0,ip,in_port=2,acti
ons=mod_dl_dst:68:05:ca:08:1f:c4,output:1
  cookie=0, duration_sec=586s, duration_nsec=610000000s, table_id=0, priority=3,
  n_packets=1771, n_bytes=174508, idle_timeout=0,hard_timeout=0,ip,in_port=1,acti
ons=output:2
  cookie=0, duration_sec=768s, duration_nsec=554000000s, table_id=2, priority=65
000, n_packets=148, n_bytes=8880, idle_timeout=0,hard_timeout=0,dl_dst=01:23:20:
00:00:01,dl_type=0x88cc,nw_src=0.0.0.0,nw_dst=0.0.0.0,nw_tos=0x00,nw_proto=0,tp_
src=0,tp_dst=0,actions=CONTROLLER:65535
[root@localhost utilities]#

```

OpenFlow Switch1 流表项

图 4-5 Switch 1 上的流表

#Openflow switch2 流表项

```
ofs2_msg1=of.ofp_flow_mod(command=0)
ofs2_msg1.priority=3
ofs2_msg1.match.in_port=3
ofs2_msg1.match.dl_type=0x800
ofs2_msg1.match.nw_dst="192.168.122.202/32"
ofs2_msg1.actions.append(of.ofp_action_dl_addr.set_dst("14:CF:92:F9:FF:6B"))
ofs2_msg1.actions.append(of.ofp_action_output(port=1))
core.openflow.connections[136643224923L].send(ofs2_msg1)
```

```
ofs2_msg2=of.ofp_flow_mod(command=0)
ofs2_msg2.priority=3
ofs2_msg2.match.in_port=1
ofs2_msg2.match.dl_type=0x0800
ofs2_msg2.actions.append(of.ofp_action_output(port=3))
core.openflow.connections[136643224923L].send(ofs2_msg2)
```

```
ofs2_msg3=of.ofp_flow_mod(command=0)
ofs2_msg3.priority=3
ofs2_msg3.match.in_port=3
ofs2_msg3.match.dl_type=0x0800
ofs2_msg3.match.nw_dst="192.168.133.203/32"
ofs2_msg3.actions.append(of.ofp_action_dl_addr.set_dst("1C:6F:65:49:FC:3F"))
ofs2_msg3.actions.append(of.ofp_action_output(port=4))
core.openflow.connections[136643224923L].send(ofs2_msg3)
```

```
ofs2_msg4=of.ofp_flow_mod(command=0)
ofs2_msg4.priority=3
ofs2_msg4.match.in_port=4
ofs2_msg4.match.dl_type=0x0800
ofs2_msg4.actions.append(of.ofp_action_output(port=3))
core.openflow.connections[136643224923L].send(ofs2_msg4)
```

如图 4-6 所示：

```

POX> ofs2_msg1=of.ofp_flow_mod(command=0)
POX> ofs2_msg1.priority=3
POX> ofs2_msg1.match.in_port=3
POX> ofs2_msg1.match.dl_type=0x800
POX> ofs2_msg1.match.nw_dst="192.168.122.202/32"
POX> ofs2_msg1.actions.append(of.ofp_action_dl_addr.set_dst("14:CF:92:F9:FF:6B"))
POX> ofs2_msg1.actions.append(of.ofp_action_output(port=1))
POX> core.openflow.connections[136643224923L].send(ofs2_msg1)
POX>
POX> ofs2_msg2=of.ofp_flow_mod(command=0)
POX> ofs2_msg2.priority=3
POX> ofs2_msg2.match.in_port=1
POX> ofs2_msg2.match.dl_type=0x800
POX> ofs2_msg2.actions.append(of.ofp_action_output(port=3))
POX> core.openflow.connections[136643224923L].send(ofs2_msg2)
POX>
POX> ofs2_msg3=of.ofp_flow_mod(command=0)
POX> ofs2_msg3.priority=3
POX> ofs2_msg3.match.in_port=3
POX> ofs2_msg3.match.dl_type=0x800
POX> ofs2_msg3.match.nw_dst="192.168.133.203/32"
POX> ofs2_msg3.actions.append(of.ofp_action_dl_addr.set_dst("1C:6F:65:49:FC:3F"))
POX> ofs2_msg3.actions.append(of.ofp_action_output(port=4))
POX> core.openflow.connections[136643224923L].send(ofs2_msg3)
POX>
POX> ofs2_msg4=of.ofp_flow_mod(command=0)
POX> ofs2_msg4.priority=3
POX> ofs2_msg4.match.in_port=4
POX> ofs2_msg4.match.dl_type=0x800
POX> ofs2_msg4.actions.append(of.ofp_action_output(port=3))
POX> core.openflow.connections[136643224923L].send(ofs2_msg4)

```

在 Controller 上，下发流表到 Switch2 上

图 4-6 通过 POX 向 switch 2 下发流表项

POX 上 Switch2 上如图 4-7 所示：

00-1f-d0-92-25-5b - TableViewer						
Switch Operations						
port	dl_src	dl_dst	dl_type	nw_src	nw_dst	actions
3			IP		192.168....	SET_DL_DST,OUTPUT:4
1			IP			OUTPUT:3
3			IP		192.168....	SET_DL_DST,OUTPUT:1
		01:23:20:00:00:01	LLDP	0.0.0.0	0.0.0.0	OUTPUT:OFPP_CONTROLLER

图 4-7 POX 上的 Switch 2

PC-B 上 Switch1 上流表如图 4-8 所示：

```

lzy@localhost: /usr/local/netfpga/openflow/utilities
File Edit View Terminal Tabs Help

[lzy@localhost ~]$ su
Password:
su: incorrect password
[lzy@localhost ~]$ su
Password:
[root@localhost lzy]# cd /usr/local/netfpga/openflow/utilities
[root@localhost utilities]# ./dpctl dump-flows unix:/var/run/dp0
stats_reply (xid=0x5f9565d1): flags=none type=1(flow)
[root@localhost utilities]# ./dpctl dump-flows unix:/var/run/dp0
stats_reply (xid=0x865e7e56): flags=none type=1(flow)
  cookie=0, duration_sec=86s, duration_nsec=981000000s, table_id=0, priority=3,
  n_packets=87, n_bytes=8526, idle_timeout=0,hard_timeout=0,ip,in_port=3,nw_dst=19
  2.168.133.203,actions=mod_dl_dst:1c:6f:65:49:fc:3f,output:4
  cookie=0, duration_sec=86s, duration_nsec=996000000s, table_id=0, priority=3,
  n_packets=261, n_bytes=25578, idle_timeout=0,hard_timeout=0,ip,in_port=1,actions
  =output:3
  cookie=0, duration_sec=87s, duration_nsec=266000000s, table_id=0, priority=3,
  n_packets=174, n_bytes=17052, idle_timeout=0,hard_timeout=0,ip,in_port=3,nw_dst=
  192.168.122.202,actions=mod_dl_dst:14:cf:92:f9:ff:6b,output:1
  cookie=0, duration_sec=319s, duration_nsec=818000000s, table_id=2, priority=65
  000, n_packets=0, n_bytes=0, idle_timeout=0,hard_timeout=0,dl_dst=01:23:20:00:00
  :01,dl_type=0x88cc,nw_src=0.0.0.0,nw_dst=0.0.0.0,nw_tos=0x00,nw_proto=0,tp_src=0
  ,tp_dst=0,actions=CONTROLLER:65535
[root@localhost utilities]#

```

OpenFlow Switch2 流表项

图 4-8 Switch 2 上的流表

(三) 实验结果以及现象描述

1.Host 登陆 ServerA:

Host (主机名为 lee):

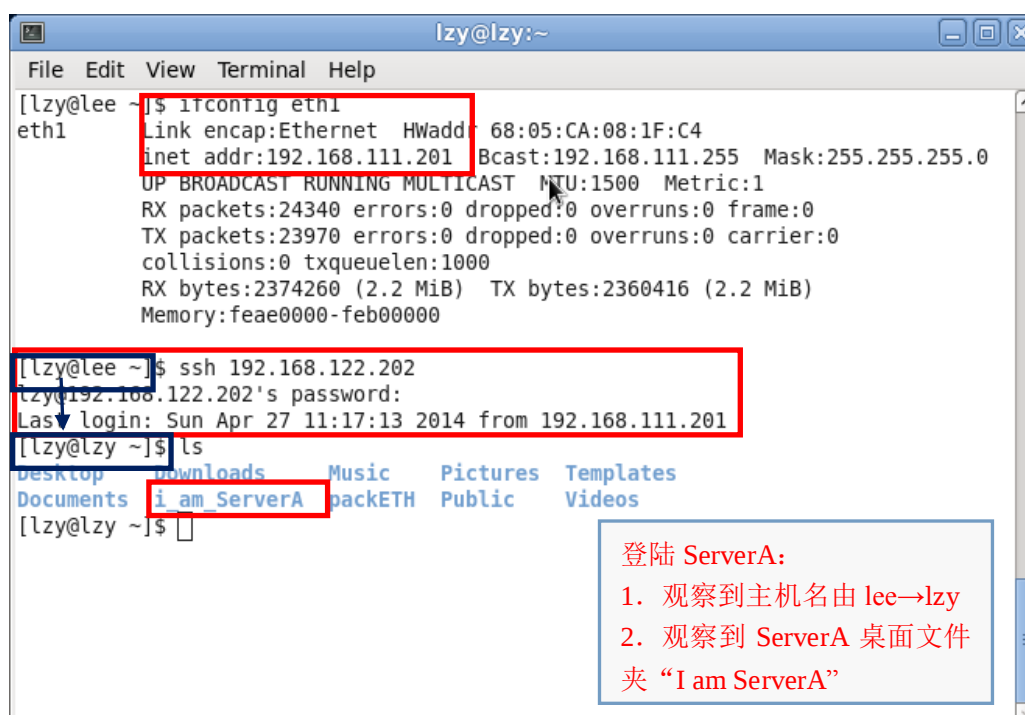


图 4-9 Host 登陆 ServerA

ServerA(主机名字为 lzy):

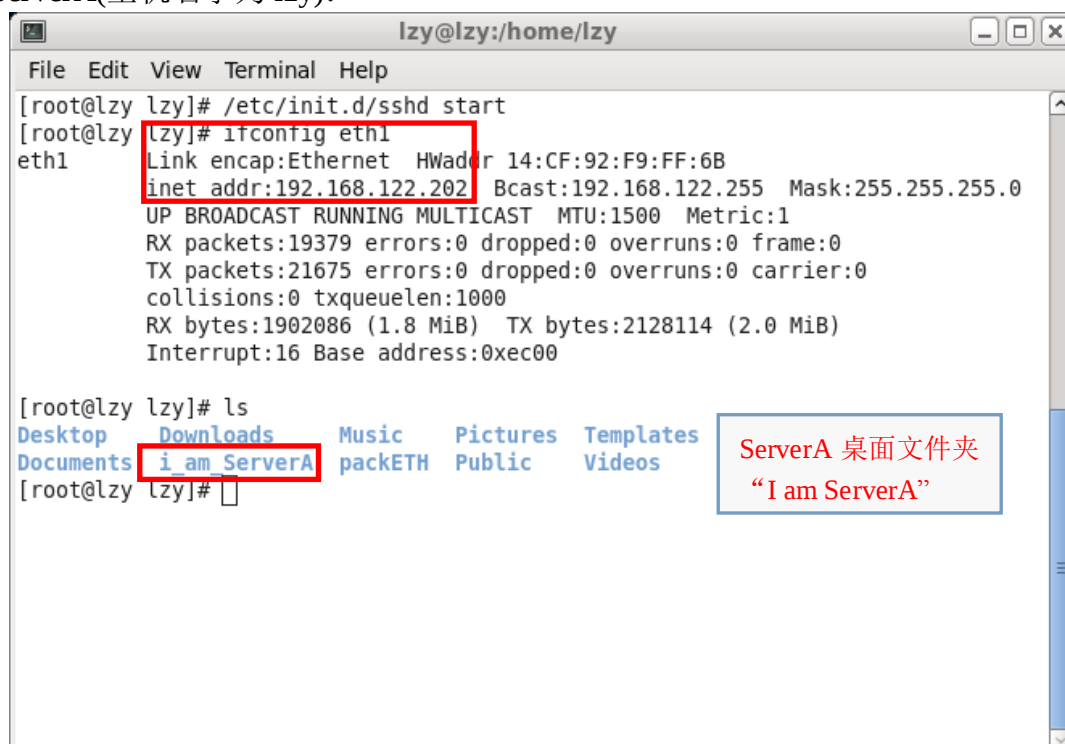
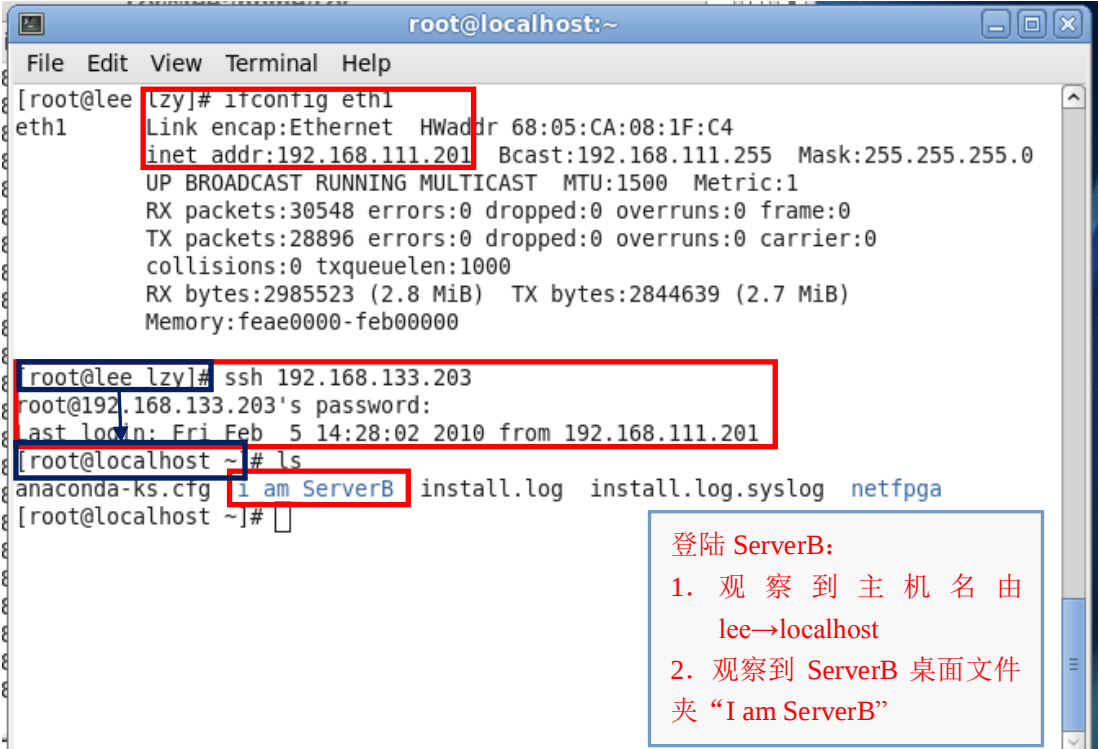


图 4-10 ServerA

2.Host 登陆 ServerB:
Host (主机名为 lee):

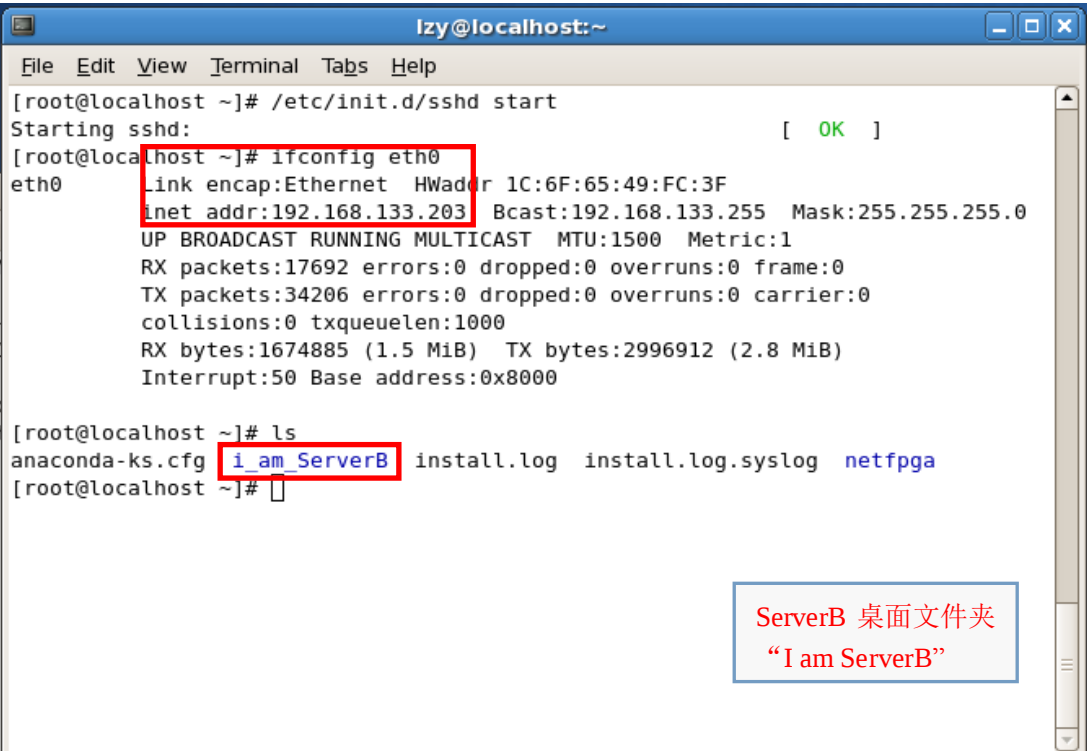


```
root@localhost:~  
File Edit View Terminal Help  
[root@lee lzy]# ifconfig eth1  
eth1 Link encap:Ethernet HWaddr 68:05:CA:08:1F:C4  
      inet addr:192.168.111.201 Bcast:192.168.111.255 Mask:255.255.255.0  
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
      RX packets:30548 errors:0 dropped:0 overruns:0 frame:0  
      TX packets:28896 errors:0 dropped:0 overruns:0 carrier:0  
      collisions:0 txqueuelen:1000  
      RX bytes:2985523 (2.8 MiB) TX bytes:2844639 (2.7 MiB)  
      Memory:feae0000-feb00000  
  
[root@lee lzy]# ssh 192.168.133.203  
root@192.168.133.203's password:  
Last login: Fri Feb  5 14:28:02 2010 from 192.168.111.201  
[root@localhost ~]# ls  
anaconda-ks.cfg i_am_ServerB install.log install.log.syslog netfpga  
[root@localhost ~]#
```

登陆 ServerB:
1. 观察到主机名由 lee→localhost
2. 观察到 ServerB 桌面文件夹 “I am ServerB”

图 4-11 Host 登陆 ServerB

ServerB (主机名为 localhost):



```
lzy@localhost:~  
File Edit View Terminal Tabs Help  
[root@localhost ~]# /etc/init.d/sshd start  
Starting sshd: [ OK ]  
[root@localhost ~]# ifconfig eth0  
eth0 Link encap:Ethernet HWaddr 1C:6F:65:49:FC:3F  
      inet addr:192.168.133.203 Bcast:192.168.133.255 Mask:255.255.255.0  
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
      RX packets:17692 errors:0 dropped:0 overruns:0 frame:0  
      TX packets:34206 errors:0 dropped:0 overruns:0 carrier:0  
      collisions:0 txqueuelen:1000  
      RX bytes:1674885 (1.5 MiB) TX bytes:2996912 (2.8 MiB)  
      Interrupt:50 Base address:0x8000  
  
[root@localhost ~]# ls  
anaconda-ks.cfg i_am_ServerB install.log install.log.syslog netfpga  
[root@localhost ~]#
```

ServerB 桌面文件夹
“I am ServerB”

图 4-12 ServerB

3.ServerA 与 ServerB 互相登陆:

ServerA 登陆 ServerB:

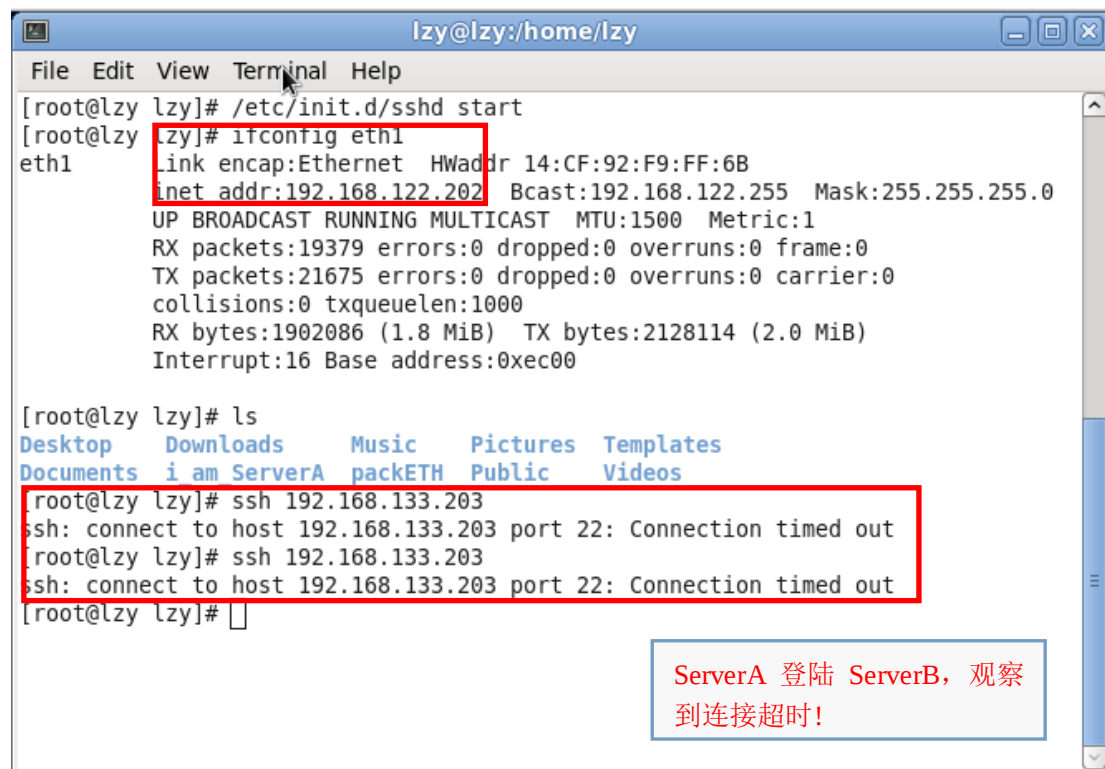


图 4-13 ServerA 无法登陆 ServerB

ServerB 登陆 ServerA:

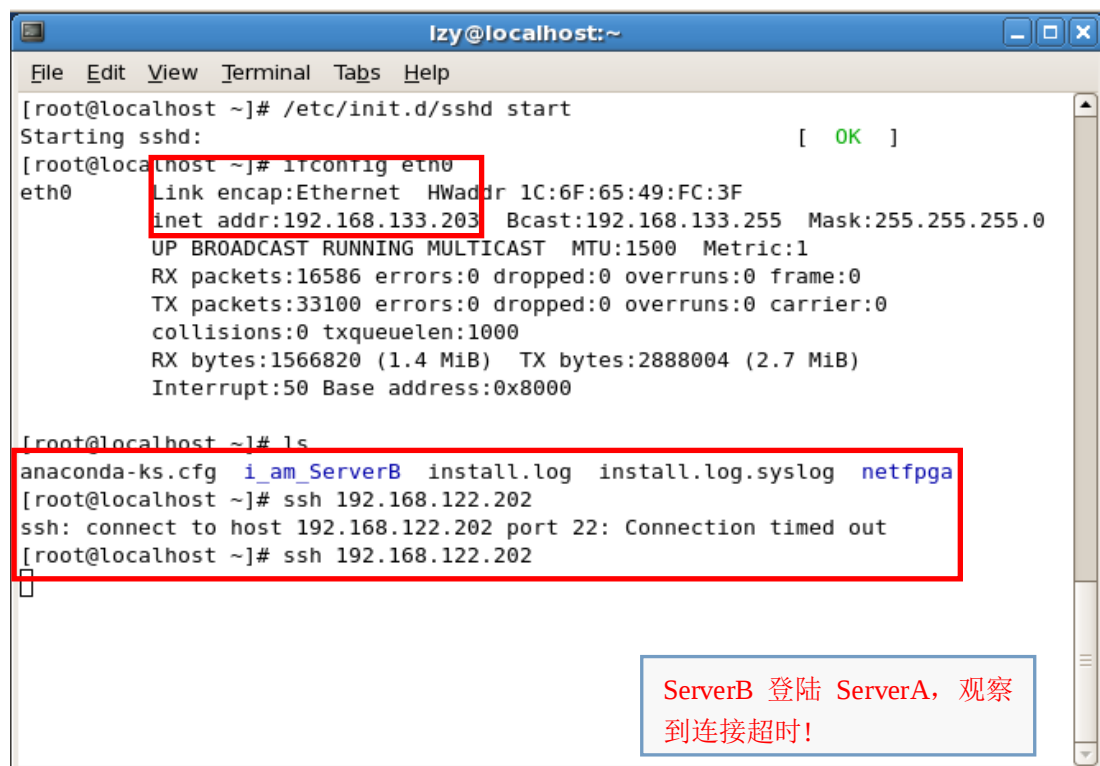


图 4-14 ServerB 无法登陆 ServerA