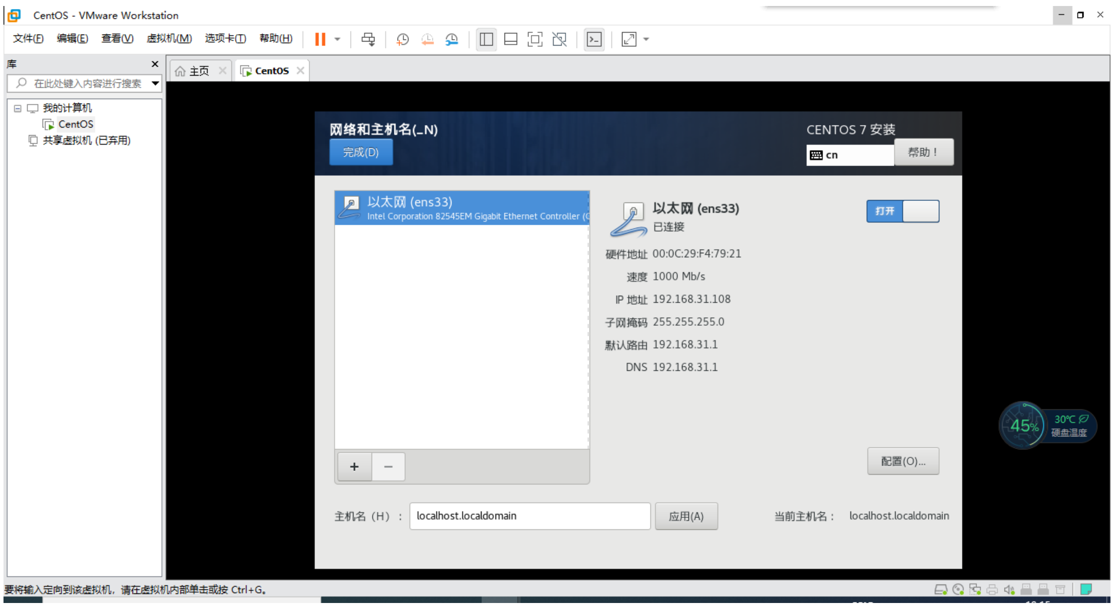
Assignment 1

1. Conduct a traceroute test for an Internet website and analyze the test results. Will the delay of the N-th hop always be longer than that of the (N-1)-th hop? Why?

1、由于每一次设置 TTL 并发送数据包的过程是相互独立的，所以包的到达时间也是不确定的，没有固定的时间先后关系，在测试时，我们发现存在 N-1 hop 比 N hop 时间长。

实验论证：

实验在 CENTOS 系统下进行：
我用的是 VMware，使用的是桥接模式。



以太网 (ens33)
Intel Corporation 82545EM Gigabit Ethernet Controller (

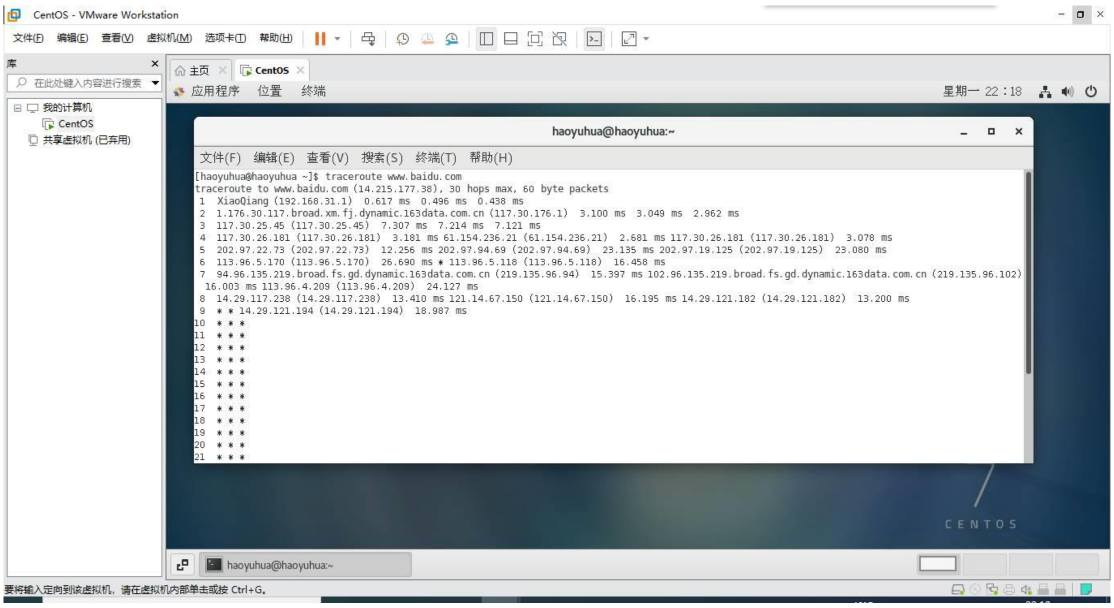
以太网 (ens33)
已连接

硬件地址	00:0C:29:F4:79:21
速度	1000 Mb/s
IP 地址	192.168.31.108
子网掩码	255.255.255.0
默认路由	192.168.31.1
DNS	192.168.31.1

主机名（H）： localhost.localdomain

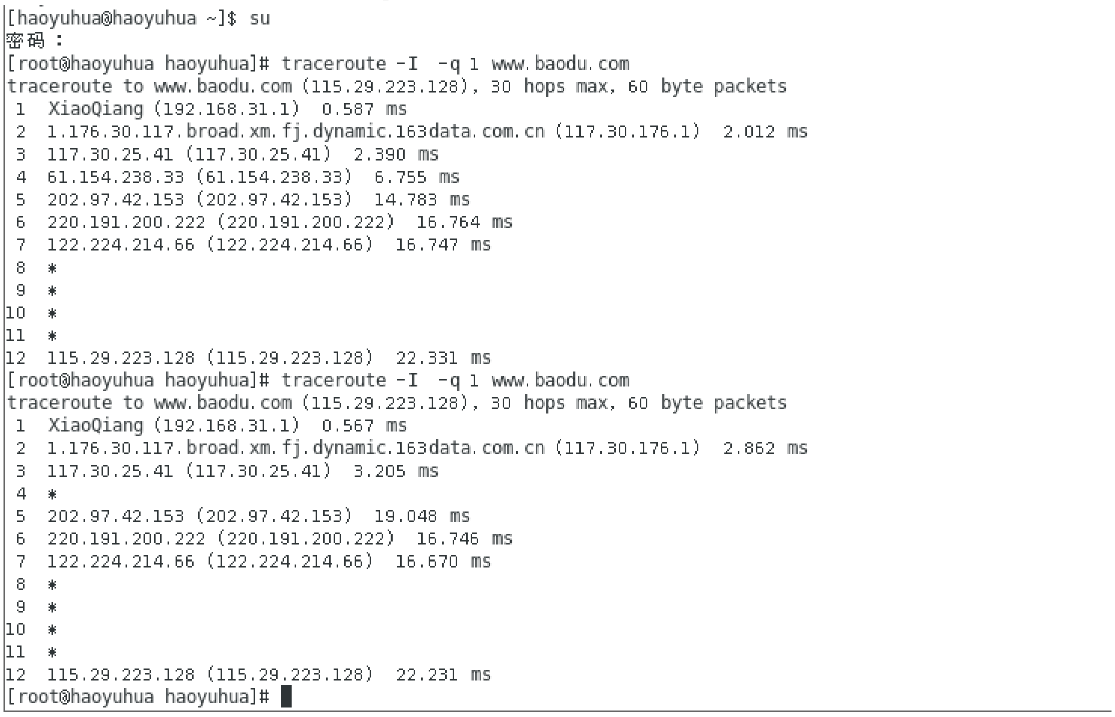
当前主机名： localhost.localdomain

在 CentOS 7 的虚拟机上测试 Traceroute 命令的时候，发现有些包的返回结果为*



```
[haoyuhua@haoyuhua ~]$ traceroute www.baidu.com
traceroute to www.baidu.com (14.215.177.38), 30 hops max, 60 byte packets
 1  XiaoQiang (192.168.31.1)  0.617 ms  0.496 ms  0.438 ms
 2  1.176.30.117.broad.xm.fj.dynamic.163data.com.cn (117.30.176.1)  3.100 ms  3.049 ms  2.962 ms
 3  117.30.25.45 (117.30.25.45)  7.307 ms  7.214 ms  7.121 ms
 4  117.30.26.181 (117.30.26.181)  3.181 ms 61.154.236.21 (61.154.236.21)  2.681 ms 117.30.26.181 (117.30.26.181)  3.078 ms
 5  202.97.22.73 (202.97.22.73)  12.256 ms 202.97.94.69 (202.97.94.69)  23.135 ms 202.97.19.125 (202.97.19.125)  23.080 ms
 6  113.96.5.170 (113.96.5.170)  26.690 ms * 113.96.5.118 (113.96.5.118)  16.458 ms
 7  94.96.135.219.broad.fs.gd.dynamic.163data.com.cn (219.135.96.94)  15.397 ms 102.96.135.219.broad.fs.gd.dynamic.163data.com.cn (219.135.96.102)
 16.003 ms 113.96.4.209 (113.96.4.209)  24.127 ms
 8  14.29.117.238 (14.29.117.238)  13.410 ms 121.14.67.150 (121.14.67.150)  16.195 ms 14.29.121.182 (14.29.121.182)  13.200 ms
 9  * * 14.29.121.194 (14.29.121.194)  18.987 ms
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
```

我们使用使用-I 解决*问题， -q 1 限制发送数据报数目为 1



```
[haoyuhua@haoyuhua ~]$ su
密码：
[root@haoyuhua haoyuhua]# traceroute -I  -q 1 www.baodu.com
traceroute to www.baodu.com (115.29.223.128), 30 hops max, 60 byte packets
 1  XiaoQiang (192.168.31.1)  0.587 ms
 2  1.176.30.117.broad.xm.fj.dynamic.163data.com.cn (117.30.176.1)  2.012 ms
 3  117.30.25.41 (117.30.25.41)  2.390 ms
 4  61.154.238.33 (61.154.238.33)  6.755 ms
 5  202.97.42.153 (202.97.42.153)  14.783 ms
 6  220.191.200.222 (220.191.200.222)  16.764 ms
 7  122.224.214.66 (122.224.214.66)  16.747 ms
 8  *
 9  *
10  *
11  *
12  115.29.223.128 (115.29.223.128)  22.331 ms
[root@haoyuhua haoyuhua]# traceroute -I  -q 1 www.baodu.com
traceroute to www.baodu.com (115.29.223.128), 30 hops max, 60 byte packets
 1  XiaoQiang (192.168.31.1)  0.567 ms
 2  1.176.30.117.broad.xm.fj.dynamic.163data.com.cn (117.30.176.1)  2.862 ms
 3  117.30.25.41 (117.30.25.41)  3.205 ms
 4  *
 5  202.97.42.153 (202.97.42.153)  19.048 ms
 6  220.191.200.222 (220.191.200.222)  16.746 ms
 7  122.224.214.66 (122.224.214.66)  16.670 ms
 8  *
 9  *
10  *
11  *
12  115.29.223.128 (115.29.223.128)  22.231 ms
[root@haoyuhua haoyuhua]# █
```

测试发现，第6跳时间比第5跳短，第7跳时间比第6跳短，论证了自己的结论。

2. Consider that there are four hosts A, B, C, and D, on a network. Initially, their ARP caches are all empty. Consider the following sequence of IP packet transmissions: (1) A-> C, (2) D ->B, (3) C->A, (4) B->D, and (5) A broadcasts locally. At the end of each successful packet transmission, what is the ARP cache content (i.e., which IP host's IP-MAC address binding) in each host? Assume that none of the cache items, if any, expire.

<1>、A→C：主机A在ARP缓存中没有找到C的映射，它将一个ARP Query的请求帧广播到本地网络的所有主机（MAC地址设置为ff-ff-ff-ff-ff），C收到这个询问后，发送reply应答信息，并将{IP_A：MAC_A}增加到自己的ARP Cache中，而网络上的其它主机发现ARP请求的IP地址与自己的不匹配而丢弃，当主机A收到从主机C发来的ARP回复后，将收到的{IP_C，MAC_C}加到自己的Cache中。

此时状态，A：{IP_A，MAC_A，IP_C：MAC_C}

B，{IP_B：MAC_B}

C：{IP_C：MAC_C，IP_A，MAC_A}

D：{IP_D：MAC_D}。

<2> D→B：主机D在ARP Cache中没有找到B的映射，发送ARP Query请求帧到本地网络。B收到询问后，将{IP_D：MAC_D}加入到自己的IP Cache中，并作ARP Reply回复，主机D收到回复，将{IP_B：MAC_B}加入到自己的ARP Cache中。

此时状态：A：{IP_A：MAC_A，IP_C：MAC_C}

B，{IP_B：MAC_B，IP_D：MAC_D}

C：{IP_C：MAC_C，IP_A：MAC_A}

D：{IP_D：MAC_D，IP_B：MAC_B}

<3>、C→A：假设此时本地缓存的生命周期均未结束，此中C根据ARP Cache中的内容 IP_A：MAC_A可以直接访问。

此时状态：不变。

<4>、B→D：B可以根据缓存中的IP_D：MAC_D直接访问。

此时状态：不变。

<5>、A broadcast locally：~~[illegible]~~ MAC地址设置为ff-ff-ff-ff-ff即可广播本地。

此时状态：不变。

3. Consider the following TCP connection (shown in Fig.1) that spans across three data-link networks. Each data-link network provides CRC for error detection. Assume that each CRC can detect the errors with probability p_{CRC} and each 16-bit checksum (for both IP and TCP) can detect the errors with probability p_{IP}. The error detection events are mutually independent. Compute the probability that the errors can be detected in the following two scenarios:

   a) Errors have been introduced to the source IP address when the packet is buffered in R1, and there are no other errors.

3、⑴在 R1 cache 中的 Packet 已经经过了 R1 的 CRC 检验和 IP checksum 计算，同时，之后的 R2 和 D 的 CRC 检验和 IP checksum 计算均不会发现，只有最终 TCP 的 checksum 会发现错误.

概率 $P = P_{IP}$.

   b) Errors have been introduced to the source IP address in the link between S and R1, and there are no other errors.

<2>. 当在 S 和 R1 的链路上出错时，R1，R2 和 D 的 CRC 检验和 IP checksum 计算都有可能发现这处错误，同样 D 处的 TCP 的 checksum 也可发现

概率：$P = 1-(1-P_{IP})^4(1-P_{CRC})^3$

   c) Please explain the functions of the end-to-end checksum in TCP and UDP

<3>、在网络中传输包，为了保证传输数据的正确性，使用了 checksum 来校验数据是否正确，ip 头的 checksum 用于检验 ip 头的数据是否正确，tcp 的 checksum 用于检验 tcp 头．tcp 数据部分是否正确．udp 的 checksum 用于检验 udp 头和 udp 数据。

TCP 的检验和是必需的，而 udp 的校验和是可选的。TCP 和 udp 计算检验和时．都要加上一个 12 字节的伪首部。伪首部的数据都是从 IP 数据报头获取的。其目的是让 TCP 检查数据是否已经正确到达目的地。伪首部共 12 字节（前 96 bits）包含如下信息：源 IP 地址，目的 IP 地址，保留字节（置 0）、传输层协议号（TCP 是 6），TCP 报文长度（报头+数据）。伪首部是为了增加 TCP 验验和的检错能力.

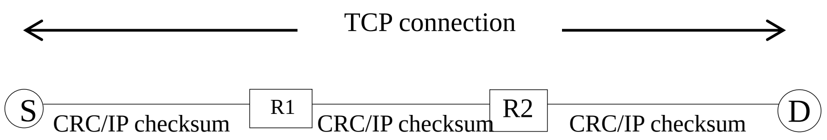


Fig 1

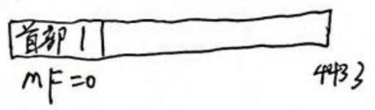
4. Consider the network in Fig. 2 in which **S** sends out a UDP data of 4,433 bytes (without counting the UDP header) to **R**. How many IP fragments is **R** expecting to receive? Explain your answer clearly. A UDP header is 8 bytes in length. Assume that there are no option fields in the IP headers, i.e., the IP header is 20 bytes in length.

4. 分析 udp: 4433 bytes data udp head: 8 bytes IP header 20 bytes

S → R1:

MTU = 4500 bytes = IP MTU

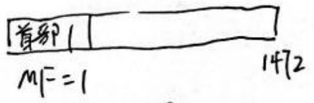
UDP - MSS = 4500 - 28 = 4472 bytes



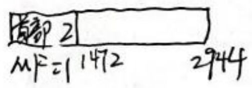
R1 → R2:

MTU = 1500 bytes = IP MTU

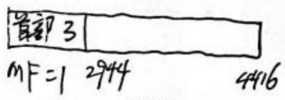
UDP-MSS = 1500 - 28 = 1472 bytes



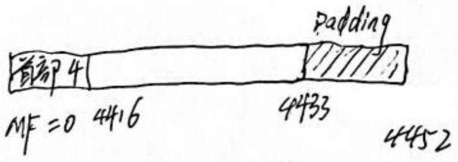
偏移 0/8 = 0



偏移 1472/8 = 184



偏移 $\frac{2944}{8}$ = 268



偏移量 $\frac{4416}{8}$ = 552

最小 MTU = 64 bytes = IP MTU

udp MSS = 64 - 28 = 36 bytes

R2→R ∵ MTU = 532 bytes = IP MTU

UDP-MSS = 532 - 28 = 504 bytes

对于 R1→R2 中 首部1→3 的帧 (MF=1)

会分成

首部1 | MF=1 0 504

偏移 0

首部2 | MF=1 504 1008

偏移 $\frac{504}{8} = 63$

首部3 | MF=1 1008 1472

偏移 $\frac{1008}{8} = 126$

对于 首部4 的帧 (MF=0)

不需分片.

R 一共会收到 3×3+1 = 10 个 fragments.

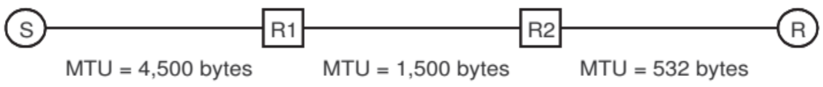


Fig. 2. A three-IP-hop network path.

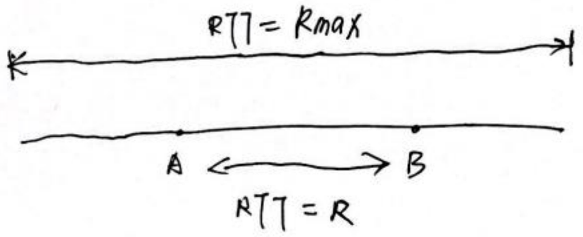
5. This question concerns the CSMA/CD protocol. Consider two nodes, A and B, on the Ethernet segment. The maximum round-trip propagation delay in this Ethernet segment is given by R_{max} seconds. And the transmission time for a frame sent by either A or B is given by R_{max} seconds. Then the round-trip propagation delay between A and B is given by $R \leq R_{max}$ seconds. Consider that node A has a frame to transmit at time *t*. Determine whether a frame collision would occur for each of the following cases:

a) Node B has a frame to transmit at time *t - 0.75R* seconds.

b) Node B has a frame to transmit at time *t - 0.25R* seconds.

c) Node B has a frame to transmit at time *t + 0.75R* seconds.

5.



分片在 AB 间传送需要

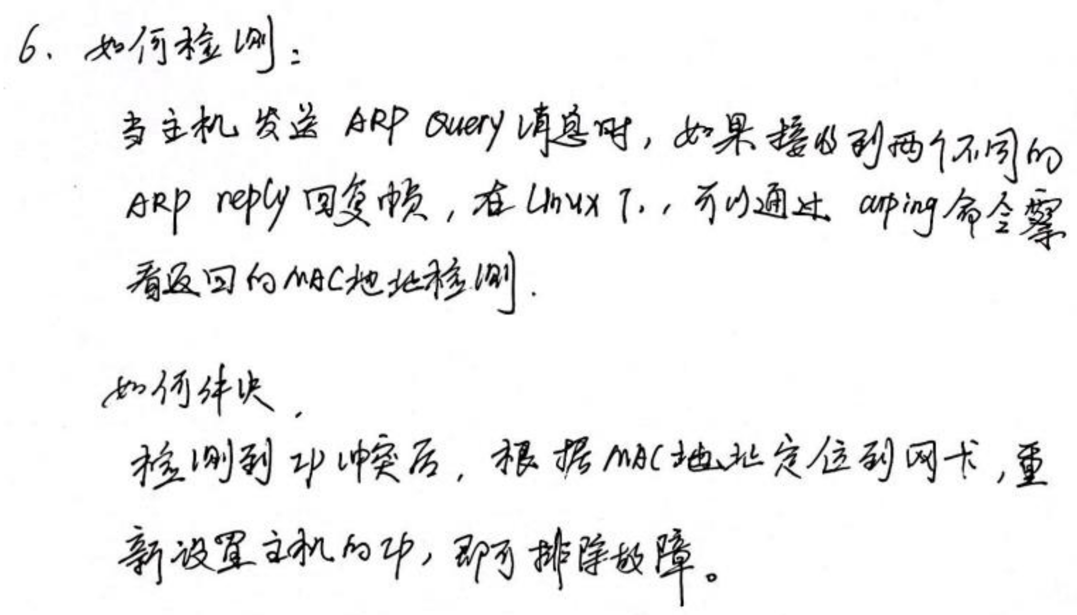
$$\frac{R}{2} + \underbrace{R_{max}}_{\text{trans time}} \geq 1.5R$$

a> B 在 t-0.75R 发送，帧会在 t-0.25R 到达 A，需 Rmax 时间完全接收，在 CSMA/CD 协议中，一个站进行的是半双工通信. 不能同时接收和发送. 由于信道忙，在 t 时刻 A 不会发送.

b> B在 t-0.25R 发送，将在 t+0.25R 到达 A，在 t 时刻此时帧还未发送到 A，A 在t时刻发送帧，会发生碰撞。

c>. A在 t时刻发送帧. 帧将在 t+0.5R 时到达 B，需 Rmax时间完全接收，由于信道忙，B在 0.75R+t 时间不会发送。

6. (Optional) What should we do if detecting duplicate IP Addresses? Please write down your steps and provide the reasons and the results for each step.


6. 如何检测：

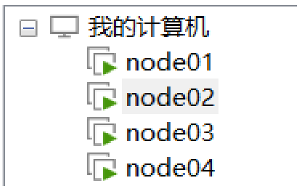
当主机发送 ARP Query 请求时，如果接收到两个不同的 ARP reply 回复帧，在 Linux 下，可以通过 arping 命令察看返回的MAC地址检测.

如何解决。

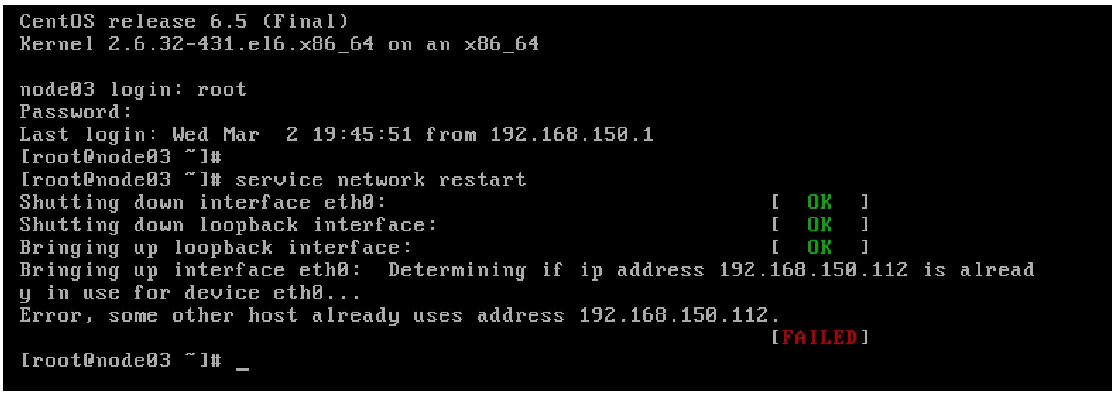
检测到 ip 冲突后，根据 MAC地址定位到网卡，重新设置主机的 ip，即可排除故障。

实验论证：

实验在 CENTOS 系统下进行：
实验使用 4 个 CENTOS 的虚拟机集群做实验。

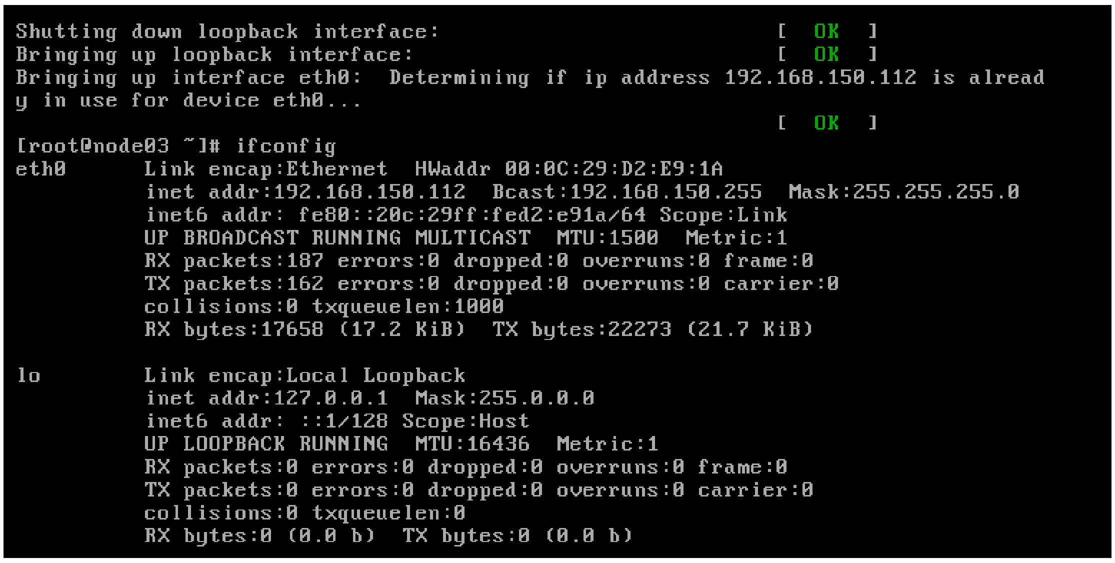

我的计算机
node01
node02
node03
node04

实验网络采用 NAT 模式
配置
Node01 IP: 192.168.150.111
Node02 IP: 192.168.150.112
Node03 IP: 192.168.150.113
Node04 IP: 192.168.150.114
本来想配置 Node02 IP 和 Node03 相同（192.168.150.112），同时开机的情况下失败了


```
CentOS release 6.5 (Final)
Kernel 2.6.32-431.el6.x86_64 on an x86_64

node03 login: root
Password:
Last login: Wed Mar  2 19:45:51 from 192.168.150.1
[root@node03 ~]#
[root@node03 ~]# service network restart
Shutting down interface eth0:                              [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:  Determining if ip address 192.168.150.112 is alread
y in use for device eth0...
Error, some other host already uses address 192.168.150.112.
                                                           [FAILED]
[root@node03 ~]# _
```

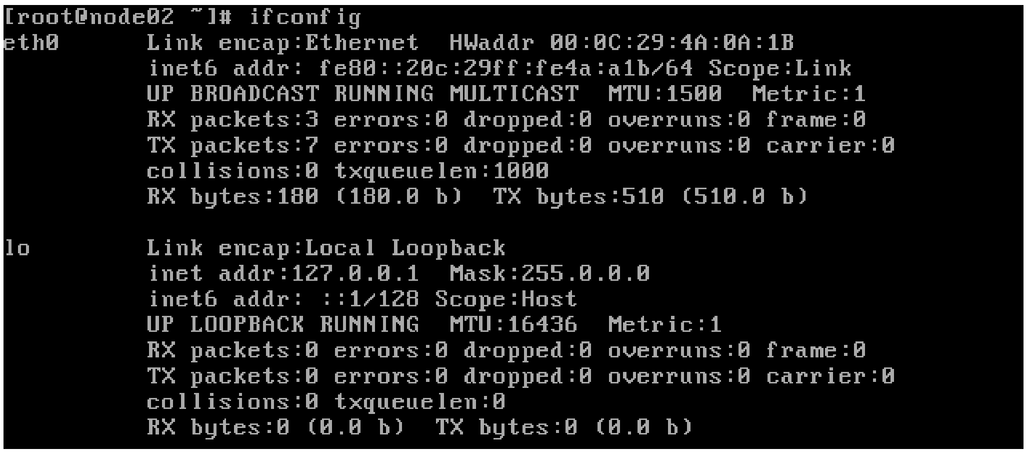
尝试关机一个再重启：

配置 Node02 IP 和 Node03 相同（192.168.150.112），先开启 Node03



```
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:  Determining if ip address 192.168.150.112 is alread
y in use for device eth0...
                                                           [  OK  ]
[root@node03 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:D2:E9:1A
          inet addr:192.168.150.112  Bcast:192.168.150.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fed2:e91a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:187 errors:0 dropped:0 overruns:0 frame:0
          TX packets:162 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:17658 (17.2 KiB)  TX bytes:22273 (21.7 KiB)

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:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

但是开启 Node02 后网卡检测不到 IP，没有解决这个问题，找原因发现是开机时候 Service network start 失败了。

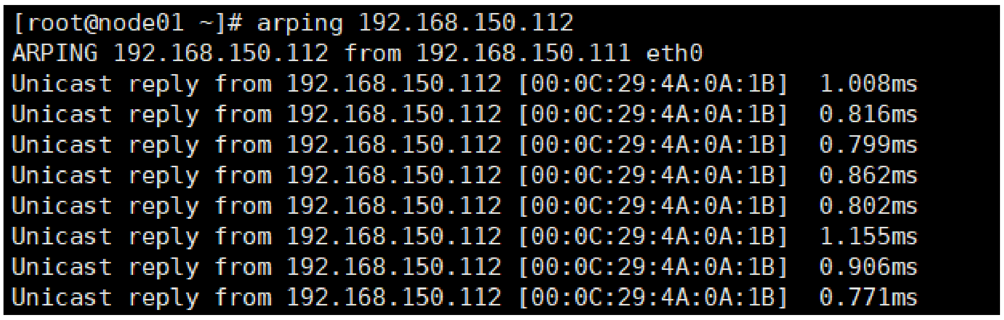


```
[root@node02 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:4A:0A:1B
          inet6 addr: fe80::20c:29ff:fe4a:a1b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:180 (180.0 b)  TX bytes:510 (510.0 b)

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:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

只能提供检测和解决的办法了

首先：

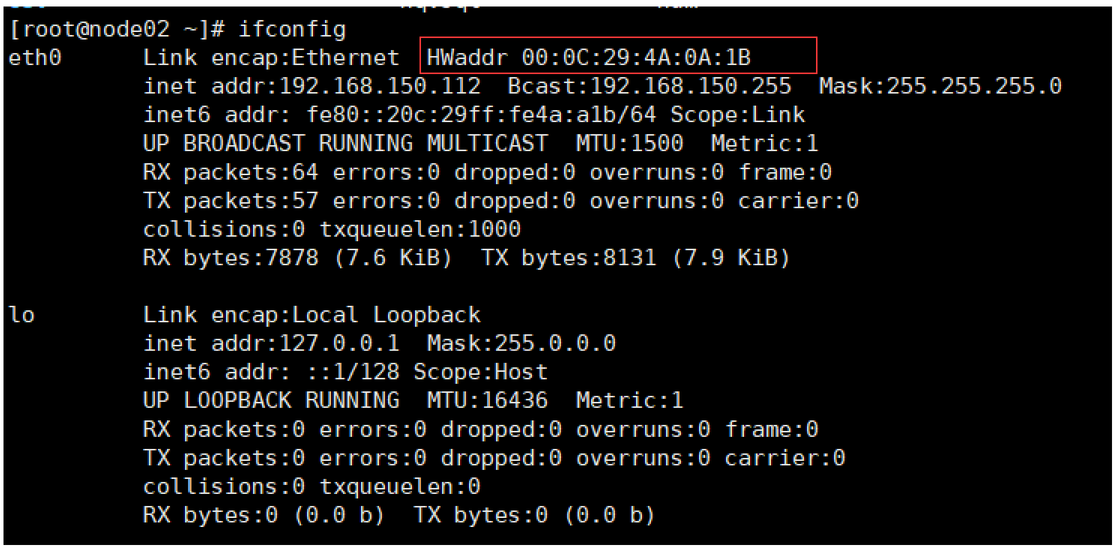
我们需要检测 IP 重复的物理网卡的 MAC 地址，通过 arping 命令



```
[root@node01 ~]# arping 192.168.150.112
ARPING 192.168.150.112 from 192.168.150.111 eth0
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  1.008ms
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  0.816ms
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  0.799ms
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  0.862ms
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  0.802ms
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  1.155ms
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  0.906ms
Unicast reply from 192.168.150.112 [00:0C:29:4A:0A:1B]  0.771ms
```

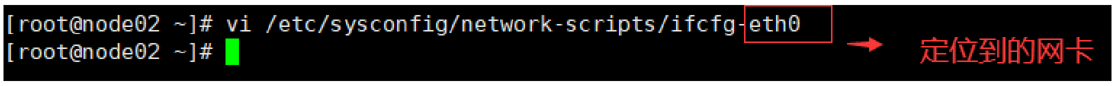
如果重复，通过 MAC 地址定位到物理网卡

使用 ifconfig 查看各个机器的网卡，找到符合的匹配网卡



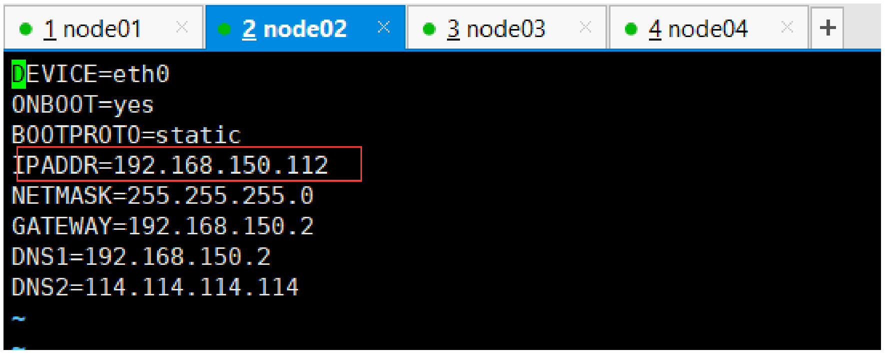
```
[root@node02 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:4A:0A:1B
          inet addr:192.168.150.112  Bcast:192.168.150.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe4a:a1b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:64 errors:0 dropped:0 overruns:0 frame:0
          TX packets:57 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7878 (7.6 KiB)  TX bytes:8131 (7.9 KiB)

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:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

定位到之后，修改当前网卡的 ip



```
[root@node02 ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth0
[root@node02 ~]#
```

修改其中的 IPADDR 即可。



```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.150.112
NETMASK=255.255.255.0
GATEWAY=192.168.150.2
DNS1=192.168.150.2
DNS2=114.114.114.114
~
```

最后

Service network restart 重启网络

尝试设计 bash 脚本，失败了，因为定位网卡部分好像需要字符串的模式匹配，设计了半天失败了。