广播网络实验实现

实验内容

- 实现节点广播的broadcast_packet函数
- 验证广播网络能够正常运行
    - 从一个端节点ping另一个端节点
- 验证广播网络的效率
    - 在three_nodes_bw.py进行iperf测量
    - 两种场景：
        - H1: iperf client; H2, H3: servers （h1同时向h2和h3测量）
        - H1: iperf server; H2, H3: clients （ h2和h3 同时向h1测量）
- 自己动手构建环形拓扑，验证该拓扑下节点广播会产生数据包环路

实验流程及结果

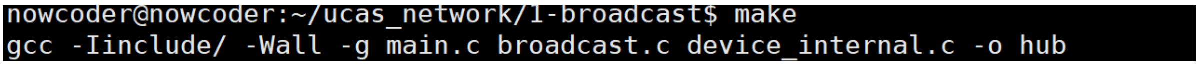
编写`brocast_packet`函数

```
void broadcast_packet(iface_info_t *iface, const char *packet, int len)
{
    iface_info_t *tx_iface = NULL;
    list_for_each_entry(tx_iface, &instance->iface_list, list) {
        if (tx_iface->index != iface->index)
            iface_send_packet(tx_iface, packet, len);
    }
    // TODO: broadcast packet
    fprintf(stdout, "TODO: broadcast packet.\n");
}
```

只要转发接口不是自己就继续发

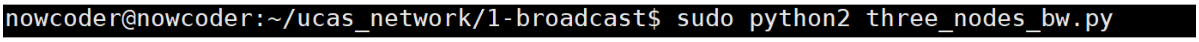
验证网络能否ping通

cd到当前目录下make，编译可执行hub文件



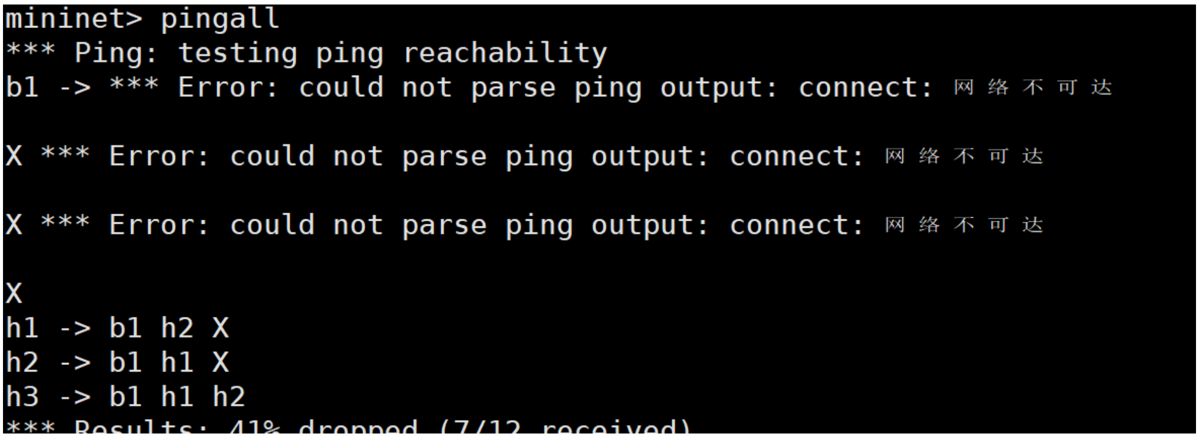
```
nowcoder@nowcoder:~/ucas_network/1-broadcast$ make
gcc -Iinclude/ -Wall -g main.c broadcast.c device_internal.c -o hub
```

执行`three_nodes_bw.py`脚本，构建基本的网络



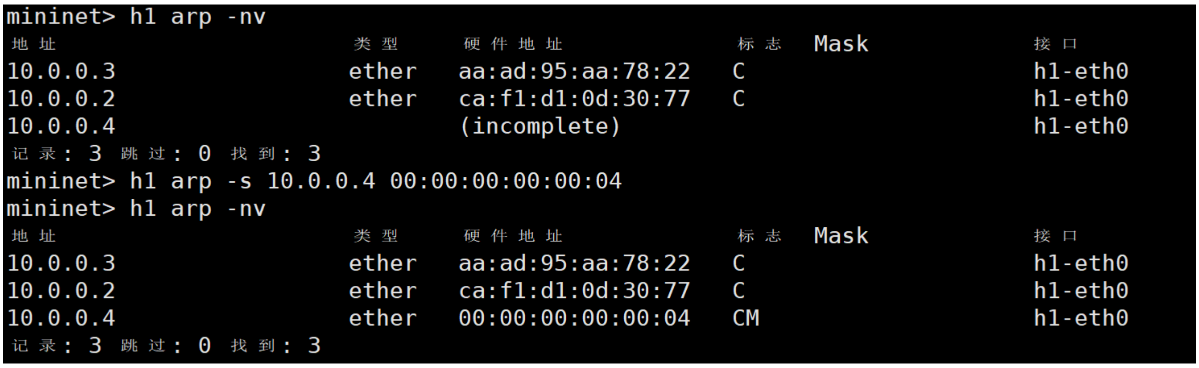
```
nowcoder@nowcoder:~/ucas_network/1-broadcast$ sudo python2 three_nodes_bw.py
```

pingall

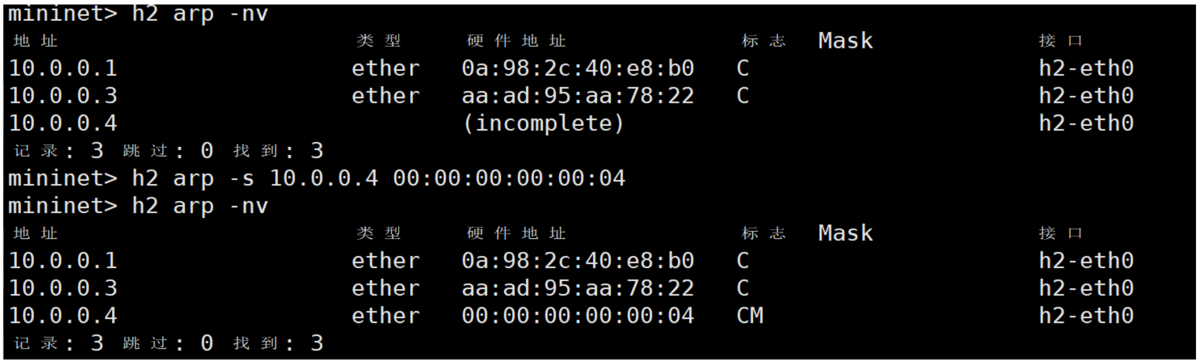


```
mininet> pingall
*** Ping: testing ping reachability
b1 -> *** Error: could not parse ping output: connect: 网络不可达

X *** Error: could not parse ping output: connect: 网络不可达

X *** Error: could not parse ping output: connect: 网络不可达

X
h1 -> b1 h2 X
h2 -> b1 h1 X
h3 -> b1 h1 h2
*** Results: 41% dropped (7/12 received)
```

发现h1，h2都ping不通h3，分别观察mac地址发现缺失，于是分别设置相同mac地址

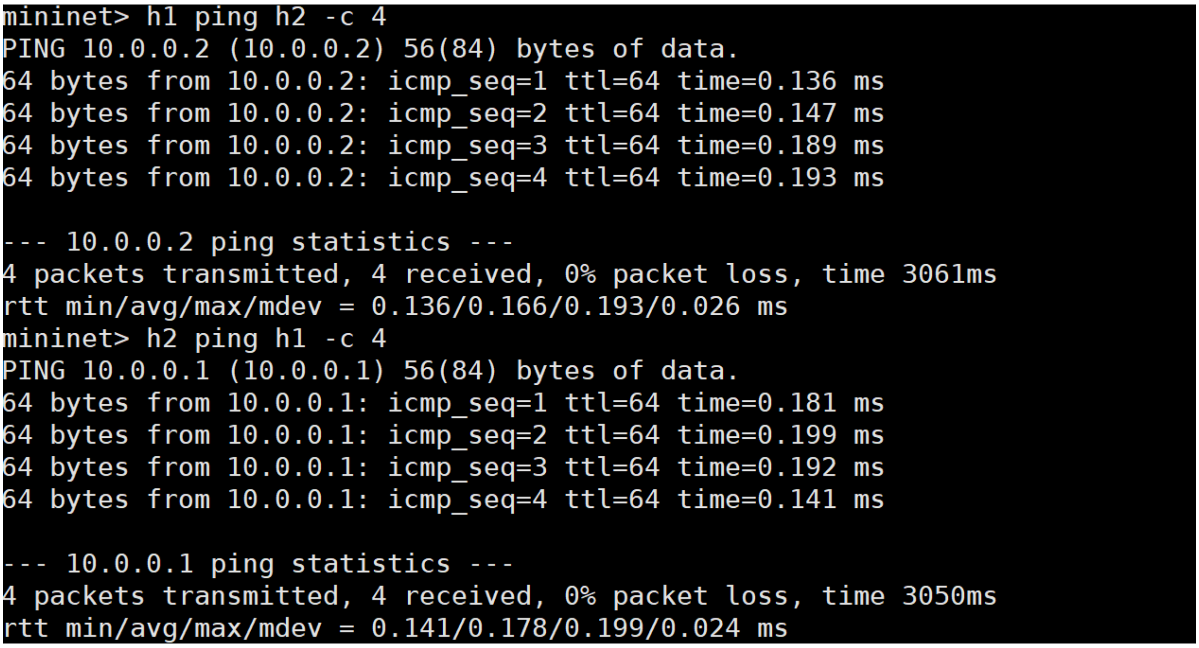


```
mininet> h1 arp -nv
地址                     类型    硬件地址            标志   Mask              接口
10.0.0.3                 ether   aa:ad:95:aa:78:22   C                        h1-eth0
10.0.0.2                 ether   ca:f1:d1:0d:30:77   C                        h1-eth0
10.0.0.4                         (incomplete)                                 h1-eth0
记录：3 跳过：0 找到：3
mininet> h1 arp -s 10.0.0.4 00:00:00:00:00:04
mininet> h1 arp -nv
地址                     类型    硬件地址            标志   Mask              接口
10.0.0.3                 ether   aa:ad:95:aa:78:22   C                        h1-eth0
10.0.0.2                 ether   ca:f1:d1:0d:30:77   C                        h1-eth0
10.0.0.4                 ether   00:00:00:00:00:04   CM                       h1-eth0
记录：3 跳过：0 找到：3
```

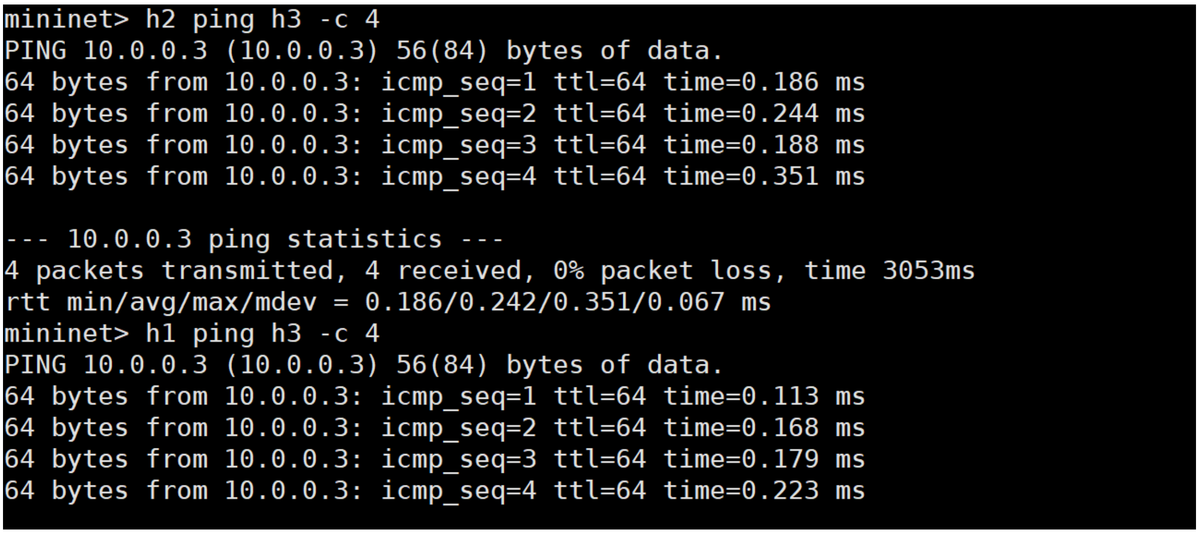


```
mininet> h2 arp -nv
地址                     类型    硬件地址            标志   Mask              接口
10.0.0.1                 ether   0a:98:2c:40:e8:b0   C                        h2-eth0
10.0.0.3                 ether   aa:ad:95:aa:78:22   C                        h2-eth0
10.0.0.4                         (incomplete)                                 h2-eth0
记录：3 跳过：0 找到：3
mininet> h2 arp -s 10.0.0.4 00:00:00:00:00:04
mininet> h2 arp -nv
地址                     类型    硬件地址            标志   Mask              接口
10.0.0.1                 ether   0a:98:2c:40:e8:b0   C                        h2-eth0
10.0.0.3                 ether   aa:ad:95:aa:78:22   C                        h2-eth0
10.0.0.4                 ether   00:00:00:00:00:04   CM                       h2-eth0
记录：3 跳过：0 找到：3
```

再次尝试能否ping通



```
mininet> h1 ping h2 -c 4
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.136 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.147 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.189 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.193 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3061ms
rtt min/avg/max/mdev = 0.136/0.166/0.193/0.026 ms
mininet> h2 ping h1 -c 4
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.181 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.199 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.192 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.141 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3050ms
rtt min/avg/max/mdev = 0.141/0.178/0.199/0.024 ms
```



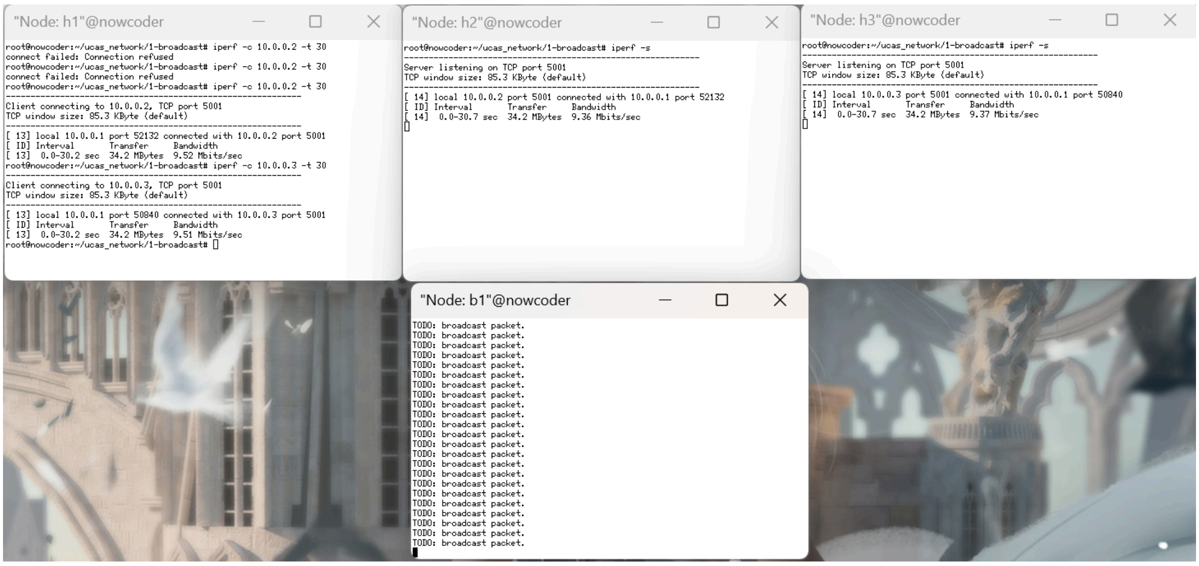
```
mininet> h2 ping h3 -c 4
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.186 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.244 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.188 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.351 ms

--- 10.0.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3053ms
rtt min/avg/max/mdev = 0.186/0.242/0.351/0.067 ms
mininet> h1 ping h3 -c 4
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.113 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.168 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.179 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.223 ms
```

发现都可以ping通

验证传播效率

xterm h1 h2 h3 b1打开节点终端

H1: iperf client; H2, H3: servers （h1同时向h2和h3测量）



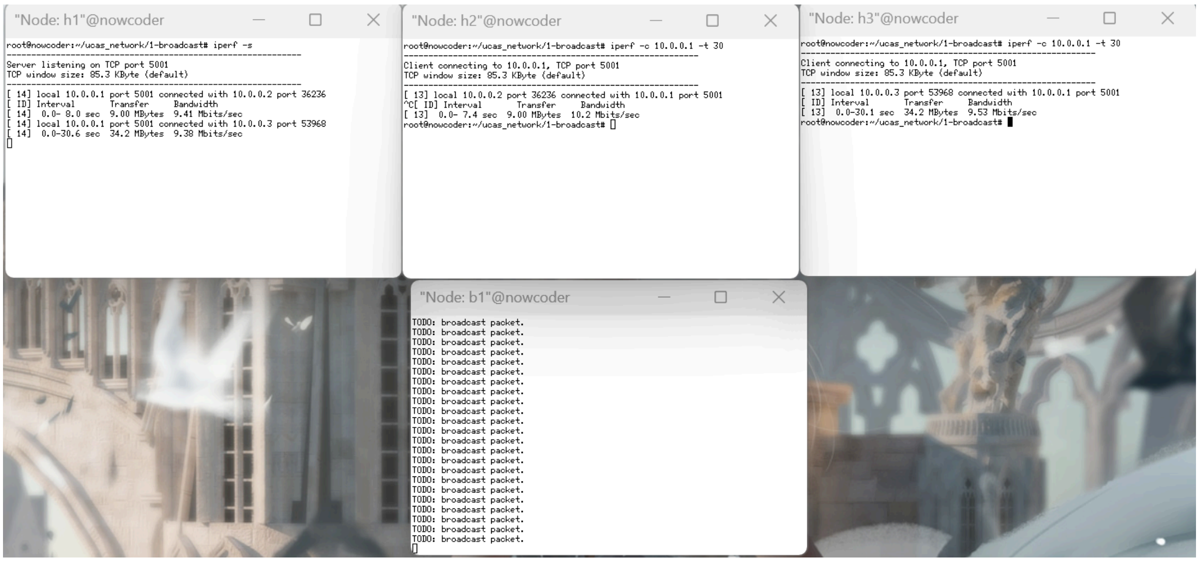
```
root@nowcoder:~/ucas_network/1-broadcast# iperf -c 10.0.0.2 -t 30
connect failed: Connection refused
root@nowcoder:~/ucas_network/1-broadcast# iperf -c 10.0.0.2 -t 30
connect failed: Connection refused
root@nowcoder:~/ucas_network/1-broadcast# iperf -c 10.0.0.2 -t 30
------------------------------------------------------------
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 13] local 10.0.0.1 port 52132 connected with 10.0.0.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 13]  0.0-30.2 sec  34.2 MBytes  9.52 Mbits/sec
root@nowcoder:~/ucas_network/1-broadcast# iperf -c 10.0.0.3 -t 30
------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 13] local 10.0.0.1 port 50840 connected with 10.0.0.3 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 13]  0.0-30.2 sec  34.2 MBytes  9.51 Mbits/sec
root@nowcoder:~/ucas_network/1-broadcast#
```

```
root@nowcoder:~/ucas_network/1-broadcast# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 14] local 10.0.0.2 port 5001 connected with 10.0.0.1 port 52132
[ ID] Interval       Transfer     Bandwidth
[ 14]  0.0-30.7 sec  34.2 MBytes  9.36 Mbits/sec
```

```
root@nowcoder:~/ucas_network/1-broadcast# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 14] local 10.0.0.3 port 5001 connected with 10.0.0.1 port 50840
[ ID] Interval       Transfer     Bandwidth
[ 14]  0.0-30.7 sec  34.2 MBytes  9.37 Mbits/sec
```

```
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
```

H1: iperf server; H2, H3: clients （ h2和h3 同时向h1测量）



```
root@nowcoder:~/ucas_network/1-broadcast# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 14] local 10.0.0.1 port 5001 connected with 10.0.0.2 port 36236
[ ID] Interval       Transfer     Bandwidth
[ 14]  0.0- 8.0 sec  9.00 MBytes  9.41 Mbits/sec
[ 14] local 10.0.0.1 port 5001 connected with 10.0.0.3 port 53968
[ 14]  0.0-30.6 sec  34.2 MBytes  9.38 Mbits/sec
```

```
root@nowcoder:~/ucas_network/1-broadcast# iperf -c 10.0.0.1 -t 30
------------------------------------------------------------
Client connecting to 10.0.0.1, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 13] local 10.0.0.2 port 36236 connected with 10.0.0.1 port 5001
^C[ ID] Interval       Transfer     Bandwidth
[ 13]  0.0- 7.4 sec  9.00 MBytes  10.2 Mbits/sec
root@nowcoder:~/ucas_network/1-broadcast#
```

```
root@nowcoder:~/ucas_network/1-broadcast# iperf -c 10.0.0.1 -t 30
------------------------------------------------------------
Client connecting to 10.0.0.1, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 13] local 10.0.0.3 port 53968 connected with 10.0.0.1 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 13]  0.0-30.1 sec  34.2 MBytes  9.53 Mbits/sec
root@nowcoder:~/ucas_network/1-broadcast#
```

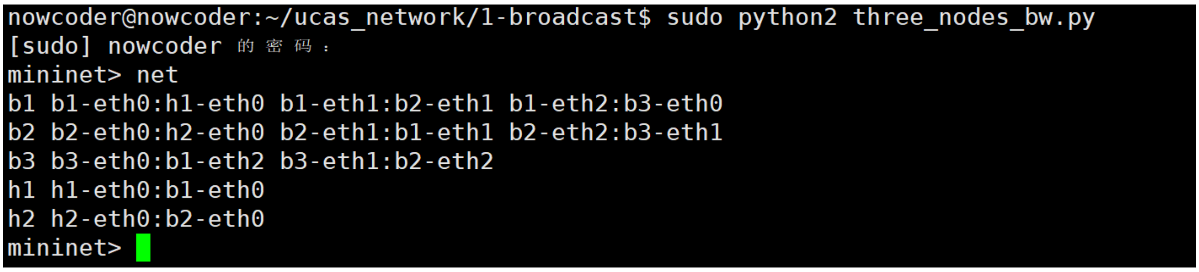
```
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
```

自己动手构建环形拓扑，验证该拓扑下节点广播会产生数据包环路

首先根据环路改写py代码

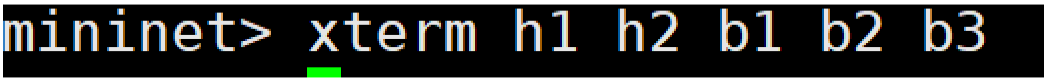
```
class BroadcastTopo(Topo):
    def build(self):
        h1 = self.addHost('h1')
        h2 = self.addHost('h2')
        b1 = self.addHost('b1')
        b2 = self.addHost('b2')
        b3 = self.addHost('b3')

        self.addLink(h1, b1, bw=20)
        self.addLink(h2, b2, bw=10)
        self.addLink(b1, b2, bw=10)
        self.addLink(b1, b3, bw=10)
        self.addLink(b2, b3, bw=10)

if __name__ == '__main__':
    check_scripts()

    topo = BroadcastTopo()
    net = Mininet(topo = topo, link = TCLink, controller = None)

    h1, h2, b1, b2, b3 = net.get('h1', 'h2', 'b1', 'b2', 'b3')
    h1.cmd('ifconfig h1-eth0 10.0.0.1/8')
    h2.cmd('ifconfig h2-eth0 10.0.0.2/8')
    clearIP(b1)
    clearIP(b2)
    clearIP(b3)

    for h in [ h1, h2, b3, b1, b2]:
        h.cmd('./scripts/disable_offloading.sh')
        h.cmd('./scripts/disable_ipv6.sh')

    net.start()
    CLI(net)
    net.stop()
```

重新执行py代码

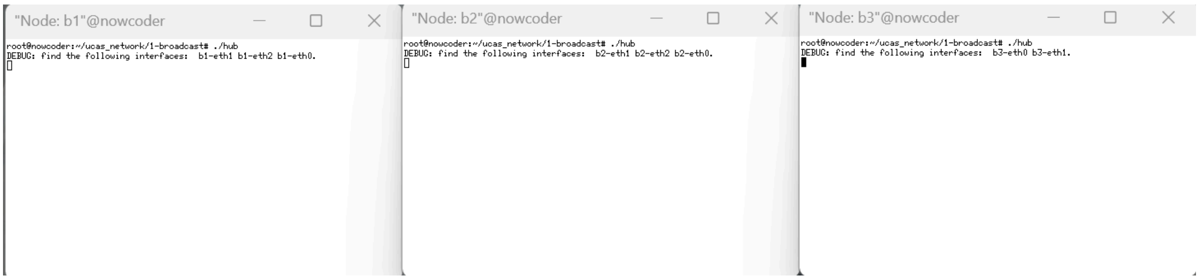


```
nowcoder@nowcoder:~/ucas_network/1-broadcast$ sudo python2 three_nodes_bw.py
[sudo] nowcoder 的密码：
mininet> net
b1 b1-eth0:h1-eth0 b1-eth1:b2-eth1 b1-eth2:b3-eth0
b2 b2-eth0:h2-eth0 b2-eth1:b1-eth1 b2-eth2:b3-eth1
b3 b3-eth0:b1-eth2 b3-eth1:b2-eth2
h1 h1-eth0:b1-eth0
h2 h2-eth0:b2-eth0
mininet>
```

打开五个节点终端并启动集线器



```
mininet> xterm h1 h2 b1 b2 b3
```

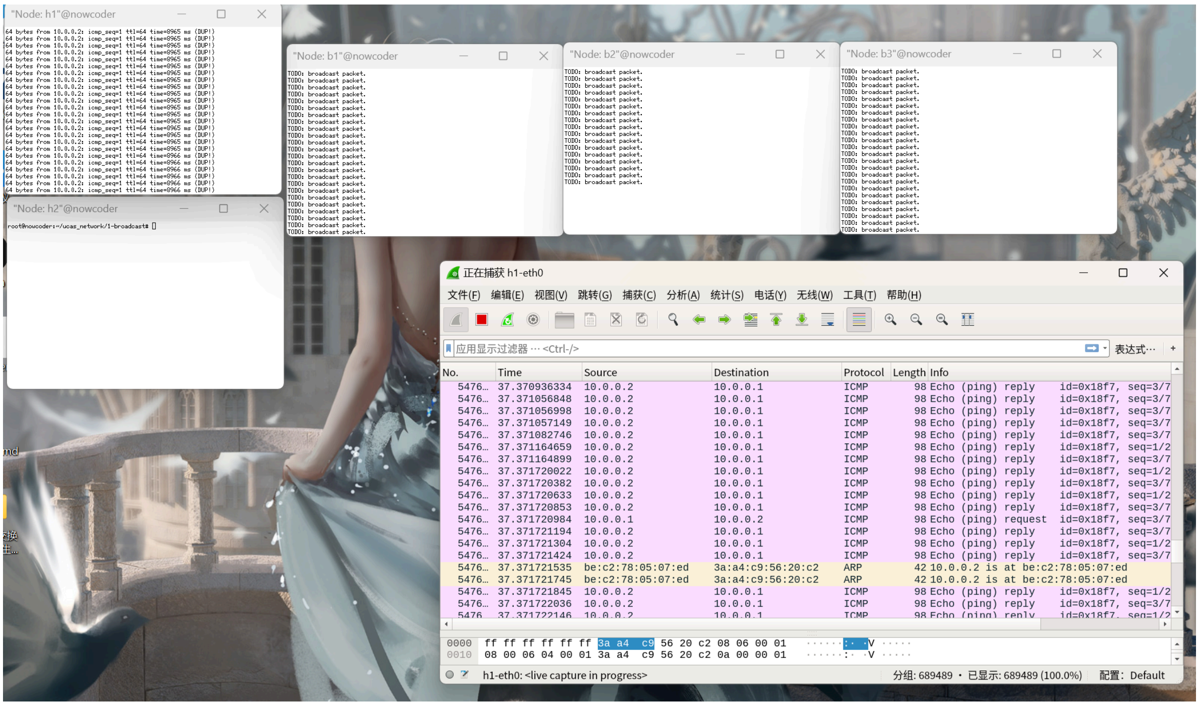


```
root@nowcoder:~/ucas_network/1-broadcast# ./hub
DEBUG: find the following interfaces:  b1-eth1 b1-eth2 b1-eth0.
```

```
root@nowcoder:~/ucas_network/1-broadcast# ./hub
DEBUG: find the following interfaces:  b2-eth1 b2-eth2 b2-eth0.
```

```
root@nowcoder:~/ucas_network/1-broadcast# ./hub
DEBUG: find the following interfaces:  b3-eth0 b3-eth1.
```

在h1ping h2



```
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8965 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8966 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8966 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8966 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8966 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8966 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=8966 ms (DUP!)
```

```
root@nowcoder:~/ucas_network/1-broadcast# 
```

```
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
```

```
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
```

```
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
TODO: broadcast packet.
```

```
No.     Time         Source              Destination         Protocol Length Info
 5476… 37.370936334 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371056848 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371056998 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371057149 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371082746 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371164659 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=1/2
 5476… 37.371164899 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371720022 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=1/2
 5476… 37.371720382 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371720633 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=1/2
 5476… 37.371720853 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371720984 10.0.0.1            10.0.0.2            ICMP        98 Echo (ping) request  id=0x18f7, seq=3/7
 5476… 37.371721194 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371721304 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=1/2
 5476… 37.371721424 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476… 37.371721535 be:c2:78:05:07:ed   3a:a4:c9:56:20:c2   ARP         42 10.0.0.2 is at be:c2:78:05:07:ed
 5476… 37.371721745 be:c2:78:05:07:ed   3a:a4:c9:56:20:c2   ARP         42 10.0.0.2 is at be:c2:78:05:07:ed
 5476… 37.371721845 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=1/2
 5476… 37.371722036 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=3/7
 5476  37.371722146 10.0.0.2            10.0.0.1            ICMP        98 Echo (ping) reply    id=0x18f7, seq=1/2

0000  ff ff ff ff ff ff 3a a4  c9 56 20 c2 08 06 00 01   ······:· ·V ·····
0010  08 00 06 04 00 01 3a a4  c9 56 20 c2 0a 00 00 01   ······:· ·V ·····

h1-eth0: <live capture in progress>          分组: 689489 · 已显示: 689489 (100.0%)     配置: Default
```

可以观察到b1，b2，b3不断输出`brocast packet`,同时抓包循环抓到数据，说明已经形成环路.