一、实验内容

1、 实现节点广播的 broadcast_packet 函数

2、 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

3、 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）

4、 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。

二、实验流程

1、 实验准备

(1) 下载并安装 VirtualBox；

(2) 下载 Ubuntu 镜像并在 VirtualBox 中安装 Ubuntu 操作系统；

(3) 运行 Ubuntu 操作系统；

(4) 启用“共享文件夹”，并将实验代码通过“共享文件夹”复制到 Ubuntu 操作系统中；

(5) 安装 mininet：

```
wasder@WASDER:~$ sudo apt install mininet
```

(6) 安装 xterm：

```
wasder@WASDER:~$ sudo apt install xterm
```

(7) 安装 make：

```
wasder@WASDER:~$ sudo apt install make
```

(8) 安装 gcc：

```
wasder@WASDER:~$ sudo apt install gcc
```

(9) 安装 wireshark：

```
wasder@WASDER:~$ sudo apt install wireshark
```

2、 实现节点广播的 broadcast_packet 函数（broadcast.c）

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

3、 验证广播网络能够正常运行

(1) 执行命令 make，生成可执行程序 hub：

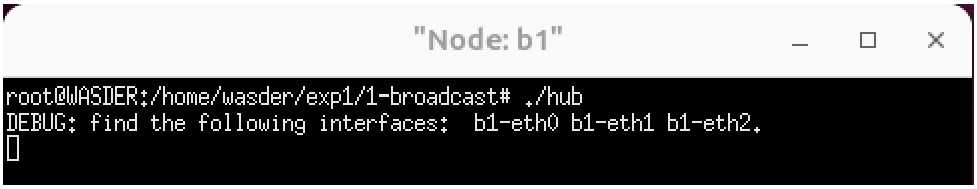
```
wasder@WASDER:~/exp1/1-broadcast$ make
```

(2) 运行拓扑文件 three_nodes_bw.py，启动 Mininet 网络：

```
wasder@WASDER:~/exp1/1-broadcast$ sudo python3 three_nodes_bw.py
```

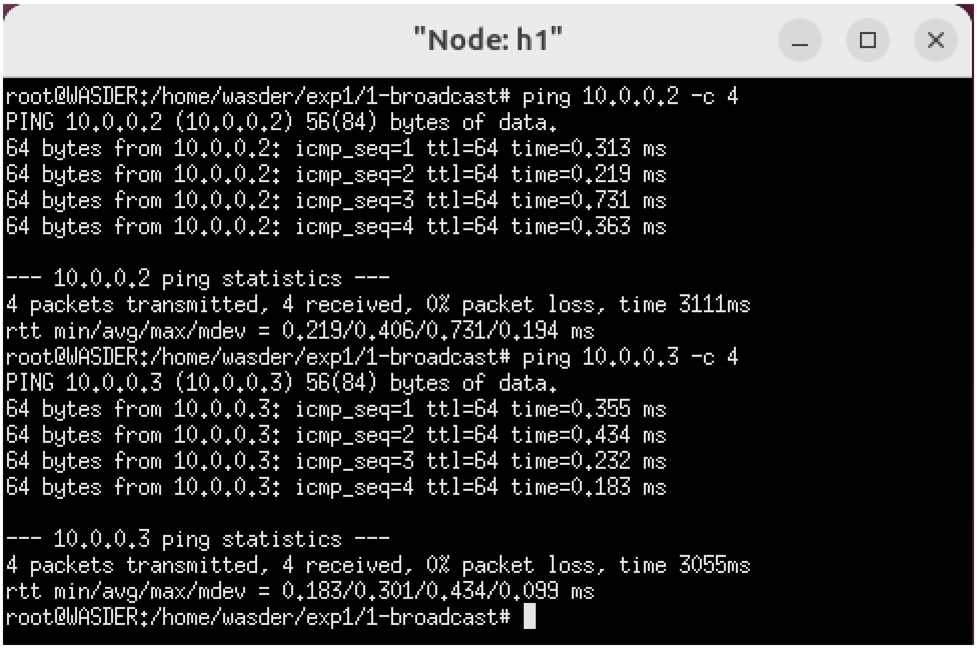
(3) 打开集线器 b1 的终端窗口，启动可执行程序 hub：

```
mininet> xterm b1
```

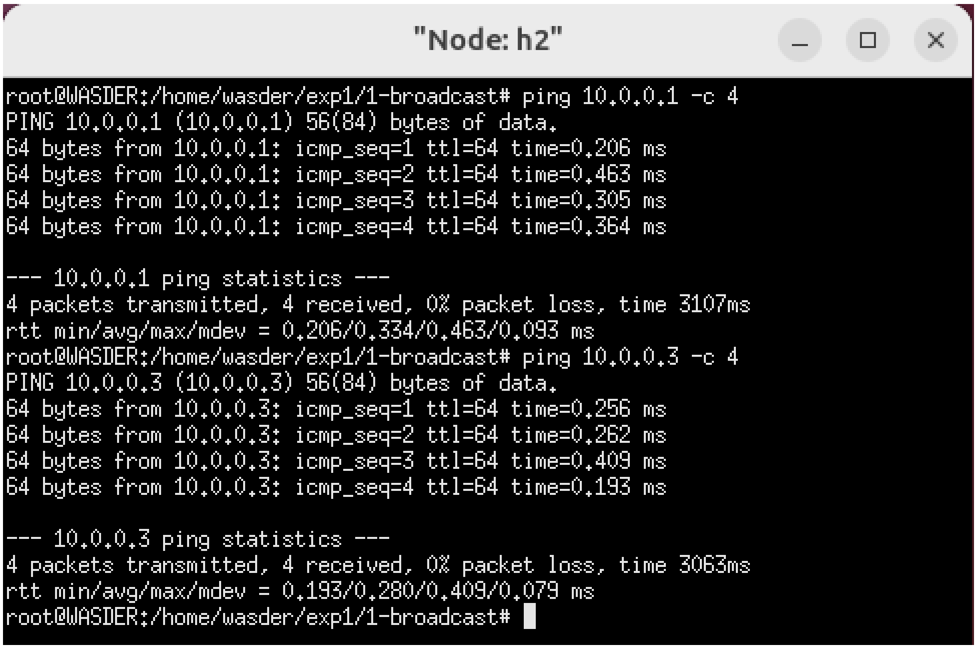


```
root@WASDER:/home/wasder/exp1/1-broadcast# ./hub
DEBUG: find the following interfaces:  b1-eth0 b1-eth1 b1-eth2.
```

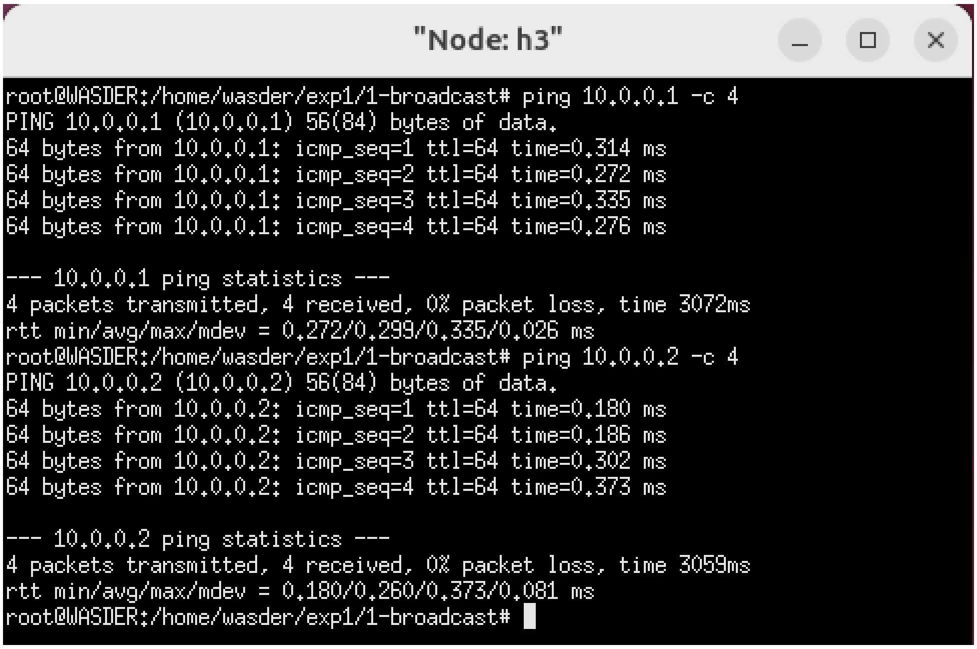
(4) 验证广播网络能够正常运行，即三个节点相互能够 ping 通：



```
root@WASDER:/home/wasder/exp1/1-broadcast# ping 10.0.0.2 -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=0.313 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.219 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.731 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.363 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3111ms
rtt min/avg/max/mdev = 0.219/0.406/0.731/0.194 ms
root@WASDER:/home/wasder/exp1/1-broadcast# ping 10.0.0.3 -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=0.355 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.434 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.232 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.183 ms

--- 10.0.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3055ms
rtt min/avg/max/mdev = 0.183/0.301/0.434/0.099 ms
root@WASDER:/home/wasder/exp1/1-broadcast#
```



```
root@WASDER:/home/wasder/exp1/1-broadcast# ping 10.0.0.1 -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=0.206 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.463 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.305 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.364 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3107ms
rtt min/avg/max/mdev = 0.206/0.334/0.463/0.093 ms
root@WASDER:/home/wasder/exp1/1-broadcast# ping 10.0.0.3 -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=0.256 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.262 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.409 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.193 ms

--- 10.0.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3063ms
rtt min/avg/max/mdev = 0.193/0.280/0.409/0.079 ms
root@WASDER:/home/wasder/exp1/1-broadcast# 
```

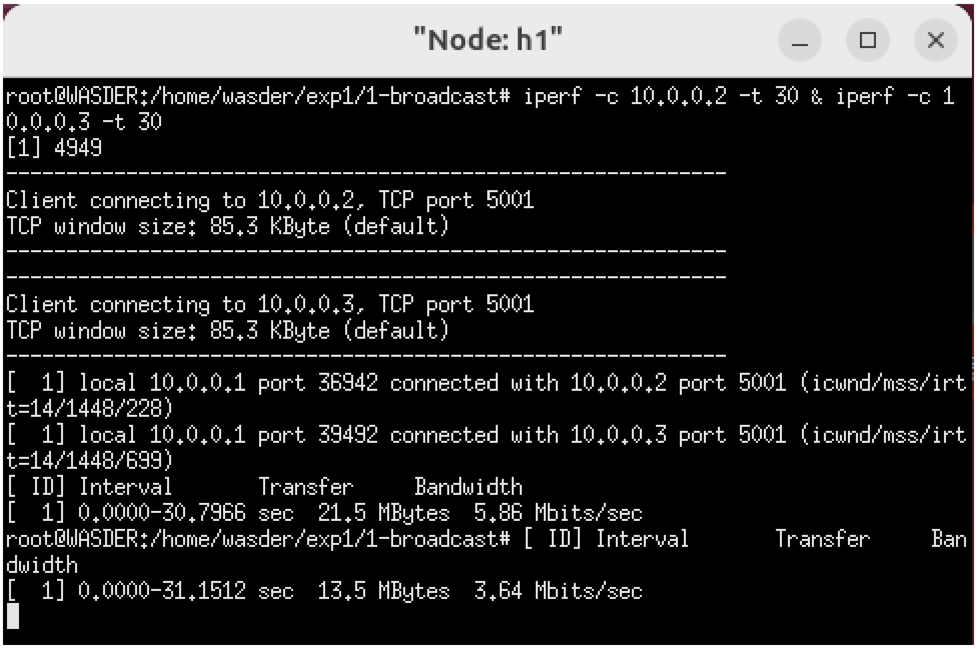


```
root@WASDER:/home/wasder/exp1/1-broadcast# ping 10.0.0.1 -c 4
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.314 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.272 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.335 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.276 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3072ms
rtt min/avg/max/mdev = 0.272/0.299/0.335/0.026 ms
root@WASDER:/home/wasder/exp1/1-broadcast# ping 10.0.0.2 -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=0.180 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.186 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.302 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.373 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3059ms
rtt min/avg/max/mdev = 0.180/0.260/0.373/0.081 ms
root@WASDER:/home/wasder/exp1/1-broadcast# 
```

4、 验证广播网络的效率

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

```
h2# iperf -s
h3# iperf -s
```

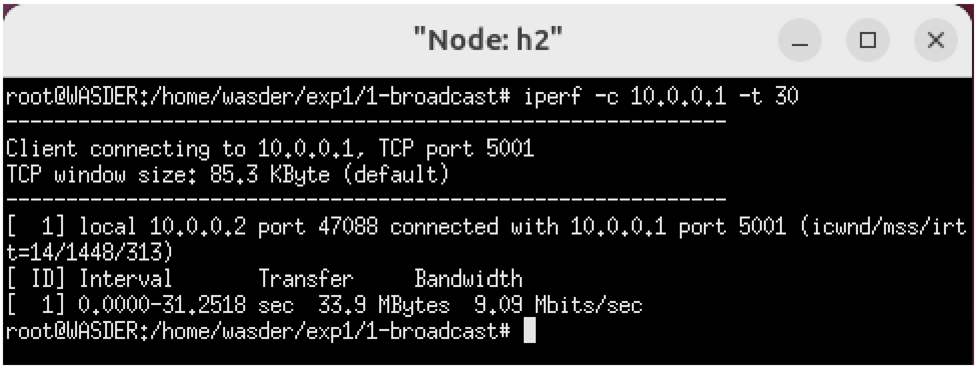


```
root@WASDER:/home/wasder/exp1/1-broadcast# iperf -c 10.0.0.2 -t 30 & iperf -c 1
0.0.0.3 -t 30
[1] 4949
------------------------------------------------------------
Client connecting to 10.0.0.2, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  1] local 10.0.0.1 port 36942 connected with 10.0.0.2 port 5001 (icwnd/mss/irt
t=14/1448/228)
[  1] local 10.0.0.1 port 39492 connected with 10.0.0.3 port 5001 (icwnd/mss/irt
t=14/1448/699)
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-30.7966 sec  21.5 MBytes  5.86 Mbits/sec
root@WASDER:/home/wasder/exp1/1-broadcast# [ ID] Interval       Transfer     Ban
dwidth
[  1] 0.0000-31.1512 sec  13.5 MBytes  3.64 Mbits/sec
█
```

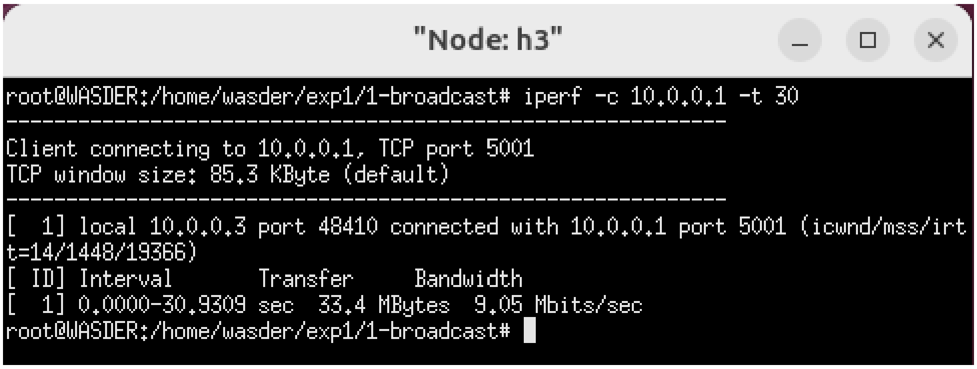
h1 节点向 h2 节点和 h3 节点的发送带宽分别为 5.86Mbps 和 3.64Mbps，而在拓扑文件中，h1->b1 的带宽为 20Mbps，b1->h2 的带宽为 10Mbps，b1->h3 的带宽为 10Mbps，因此 h1 同时向 h2 和 h3 测量得到的集线器广播网络效率约为 47.5%。

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

```
h1# iperf -s
```



```
root@WASDER:/home/wasder/exp1/1-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)
------------------------------------------------------------
[  1] local 10.0.0.2 port 47088 connected with 10.0.0.1 port 5001 (icwnd/mss/irt
t=14/1448/313)
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-31.2518 sec  33.9 MBytes  9.09 Mbits/sec
root@WASDER:/home/wasder/exp1/1-broadcast# █
```



```
root@WASDER:/home/wasder/exp1/1-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)
------------------------------------------------------------
[  1] local 10.0.0.3 port 48410 connected with 10.0.0.1 port 5001 (icwnd/mss/irt
t=14/1448/19366)
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-30.9309 sec  33.4 MBytes  9.05 Mbits/sec
root@WASDER:/home/wasder/exp1/1-broadcast# █
```

h1 节点对 h2 节点和 h3 节点的接收带宽分别为 9.09Mbps 和 9.05Mbps，而在拓扑文件中，h1->b1 的带宽为 20Mbps，b1->h2 的带宽为 10Mbps，b1->h3 的带宽为 10Mbps，因此 h2 和 h3 同时向 h1 测量得到的集线器广播网络效率约为 90.7%。

5、 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

(1) 建立 three_nodes_bw.py 的副本 three_nodes_bw_copy.py:

```
wasder@WASDER:~/exp1/1-broadcast$ cp three_nodes_bw.py three_nodes_bw_copy.py
```

(2) 根据所需环形网络拓扑修改 three_nodes_bw_copy.py:

```
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=20)
        self.addLink(b1, b2, bw=20)
        self.addLink(b1, b3, bw=20)
        self.addLink(b2, b3, bw=20)

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, b1, b2, b3 ]:
        h.cmd('./scripts/disable_offloading.sh')
        h.cmd('./scripts/disable_ipv6.sh')
```

(3) 运行拓扑文件 three_nodes_bw_copy.py，启动 Mininet 网络：

```
wasder@WASDER:~/exp1/1-broadcast$ sudo python3 three_nodes_bw_copy.py
```

(4) 打开集线器 b1、b2、b3 的终端窗口，启动生成可执行程序 hub：

```
mininet> xterm b1 b2 b3
b1# ./hub
b2# ./hub
b3# ./hub
```

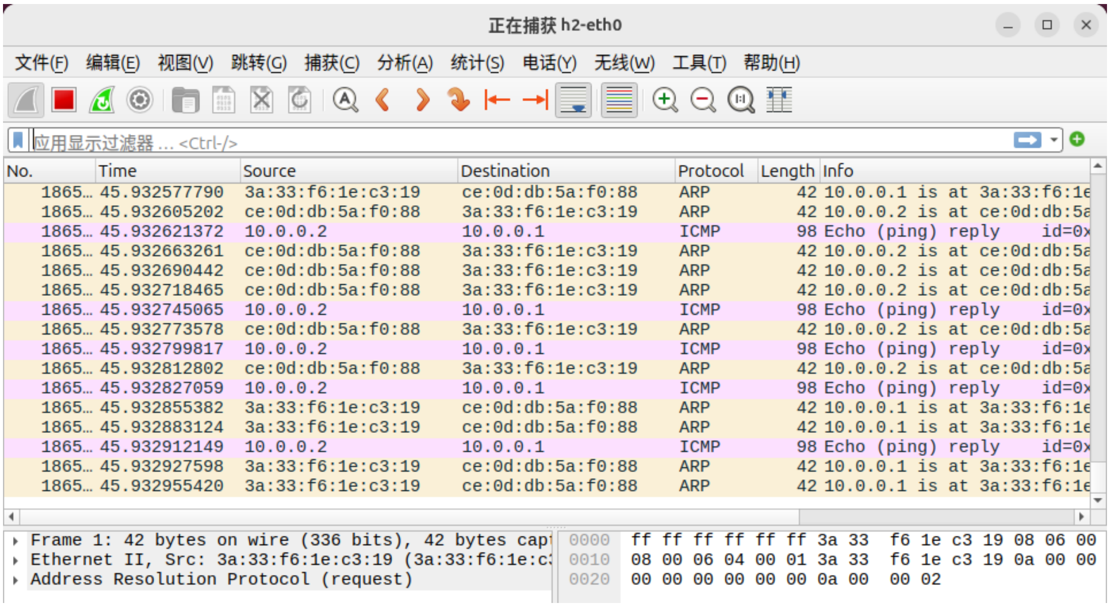
(5) 打开主机 h2 的终端窗口，启动 wireshark，等待捕获主机 h1 发来的数据包：

```
mininet> xterm h2
h2# wireshark
```

(6) 打开主机 h1 的终端窗口，向主机 h2 发送一个数据包：

```
mininet> xterm h1
h1# ping -c 1 10.0.0.2
```

(7) 抓包看到一个数据包不断被广播：



No.	Time	Source	Destination	Protocol	Length	Info
1865…	45.932577790	3a:33:f6:1e:c3:19	ce:0d:db:5a:f0:88	ARP	42	10.0.0.1 is at 3a:33:f6:1e
1865…	45.932605202	ce:0d:db:5a:f0:88	3a:33:f6:1e:c3:19	ARP	42	10.0.0.2 is at ce:0d:db:5a
1865…	45.932621372	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply id=0x
1865…	45.932663261	ce:0d:db:5a:f0:88	3a:33:f6:1e:c3:19	ARP	42	10.0.0.2 is at ce:0d:db:5a
1865…	45.932690442	ce:0d:db:5a:f0:88	3a:33:f6:1e:c3:19	ARP	42	10.0.0.2 is at ce:0d:db:5a
1865…	45.932718465	ce:0d:db:5a:f0:88	3a:33:f6:1e:c3:19	ARP	42	10.0.0.2 is at ce:0d:db:5a
1865…	45.932745065	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply id=0x
1865…	45.932773578	ce:0d:db:5a:f0:88	3a:33:f6:1e:c3:19	ARP	42	10.0.0.2 is at ce:0d:db:5a
1865…	45.932799817	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply id=0x
1865…	45.932812802	ce:0d:db:5a:f0:88	3a:33:f6:1e:c3:19	ARP	42	10.0.0.2 is at ce:0d:db:5a
1865…	45.932827059	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply id=0x
1865…	45.932855382	3a:33:f6:1e:c3:19	ce:0d:db:5a:f0:88	ARP	42	10.0.0.1 is at 3a:33:f6:1e
1865…	45.932883124	3a:33:f6:1e:c3:19	ce:0d:db:5a:f0:88	ARP	42	10.0.0.1 is at 3a:33:f6:1e
1865…	45.932912149	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) reply id=0x
1865…	45.932927598	3a:33:f6:1e:c3:19	ce:0d:db:5a:f0:88	ARP	42	10.0.0.1 is at 3a:33:f6:1e
1865…	45.932955420	3a:33:f6:1e:c3:19	ce:0d:db:5a:f0:88	ARP	42	10.0.0.1 is at 3a:33:f6:1e

```
▸ Frame 1: 42 bytes on wire (336 bits), 42 bytes capt
▸ Ethernet II, Src: 3a:33:f6:1e:c3:19 (3a:33:f6:1e:c3
▸ Address Resolution Protocol (request)
```

```
0000  ff ff ff ff ff ff 3a 33  f6 1e c3 19 08 06 00
0010  08 00 06 04 00 01 3a 33  f6 1e c3 19 0a 00 00
0020  00 00 00 00 00 00 0a 00  00 02
```