实验报告

交换机转发实验

一、实验内容

了解交换机的转发原理和转发表的构建方式，理解交换机如何学习和维护转发表。实现转发表的数据结构，支持转发表的查询、插入、老化操作，完成一个能自动学习转发表的交换机。使用 iperf 和给定的拓扑进行测试，对比交换机转发与之前集线器广播的性能差异。

二、实验流程

1. 根据转发表的结构，实现转发表的查询、插入操作。
2. 利用多线程与互斥，实现转发表的老化操作。
3. 根据交换机的转发原理，完成对数据包的处理函数。
4. 使用 iperf 进行测试，对比交换机转发与集线器广播的性能。

三、实验结果及分析

（一）实现交换机转发

1、转发表查询

转发表为了快速查询了使用了 256 个链表，在查询时先对 MAC 地址 hash，根据 key 值找到所对应链表。然后遍历该链表，查看有无与输入 MAC 地址相同的表项。若有，查询成功，返回该表项中端口结构 iface；若无，查询失败，返回 NULL。具体的代码现实如下：

```c
iface_info_t *lookup_port(u8 mac[ETH_ALEN])
{
    // TODO: implement the lookup process here
    //fprintf(stdout, "TODO: implement the lookup process here.\n");

    int i = hash8(mac, ETH_ALEN);
    mac_port_entry_t *entry = NULL;

    pthread_mutex_lock(&mac_port_map.lock);
    list_for_each_entry(entry, &mac_port_map.hash_table[i], list) {
        if (memcmp(entry->mac, mac, ETH_ALEN) == 0) {
            pthread_mutex_unlock(&mac_port_map.lock);
            return entry->iface;
        }
    }
    pthread_mutex_unlock(&mac_port_map.lock);

    return NULL;
}
```

2、转发表插入

insert_mac_port 函数用于实现插入操作。首先根据源 MAC 地址在转发表中查找表项，如果找到，那么更新表项、更新访问时间；如果没有找到，那么将源地址与端口的映射关系写入转发表。具体的代码现实如下：

```c
void insert_mac_port(u8 mac[ETH_ALEN], iface_info_t *iface)
{
    // TODO: implement the insertion process here
    //fprintf(stdout, "TODO: implement the insertion process here.\n");

    int i = hash8(mac, ETH_ALEN);
    mac_port_entry_t *entry = NULL;
    pthread_mutex_lock(&mac_port_map.lock);
    list_for_each_entry(entry, &mac_port_map.hash_table[i], list) {
        if (memcmp(entry->mac, mac, ETH_ALEN) == 0) {
            entry->iface = iface;
            entry->visited = time(NULL);
            pthread_mutex_unlock(&mac_port_map.lock);
            return;
        }
    }

    mac_port_entry_t *new_entry = (mac_port_entry_t *)malloc(sizeof(mac_port_entry_t));
    bzero(new_entry, sizeof(mac_port_entry_t));
    init_list_head(&new_entry->list);
    memcpy(new_entry->mac, mac, ETH_ALEN);
    new_entry->iface = iface;
    new_entry->visited = time(NULL);

    list_add_tail(&new_entry->list, &mac_port_map.hash_table[i]);
    pthread_mutex_unlock(&mac_port_map.lock);
}
```

3、转发表老化

执行老化操作时遍历整个转发表，查看当前时间与每个表项访问时间之差是否超过 30 秒，如果超过就删除掉该表项。具体的代码现实如下：

```
int sweep_aged_mac_port_entry()
{
    // TODO: implement the sweeping process here
    //fprintf(stdout, "TODO: implement the sweeping process here.\n");

    int num = 0;
    pthread_mutex_lock(&mac_port_map.lock);
    mac_port_entry_t *entry, *q;
    for (int i = 0; i < HASH_8BITS; i++) {
        list_for_each_entry_safe(entry, q, &mac_port_map.hash_table[i], list) {
            if(time(NULL) - entry->visited > MAC_PORT_TIMEOUT){
                list_delete_entry(&entry->list);
                free(entry);
                num++;
            }
        }
    }
    pthread_mutex_unlock(&mac_port_map.lock);

    return num;
}
```

4、交换机处理函数

交换机需要完成 3 种操作，查询操作、插入操作、老化操作。其中老化操作由单独线程处理，基本就是封装调用上面的 sweep 函数，这里不再赘述。handle_packet 函数中需要完成的，就是查询操作和插入操作。

查询操作对目的 MAC 地址进行查询，如果查到相应条目，那么只对相应转发端口转发数据包；如果没查到就广播该数据包。

插入操作对源 MAC 地址进行查询，如果查到相应条目，更新访问时间；如果没有查到，那么将该地址与端口的映射关系写入到转发表。这里只需要调用已经写好的 insert_mac_port 函数。

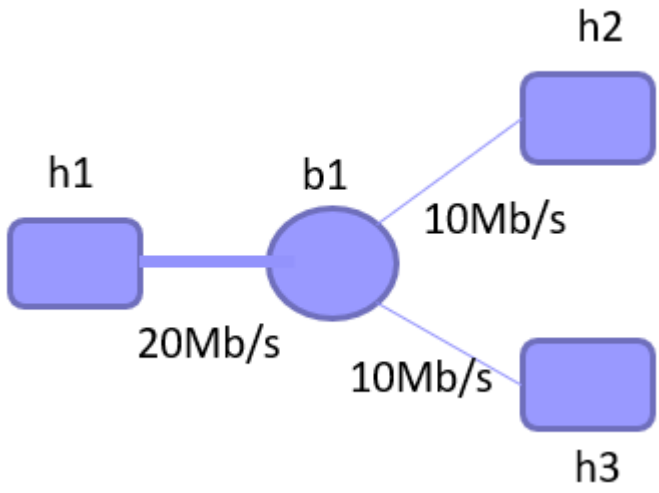
```
// lookup dest
iface_info_t *dest_iface = lookup_port(eh->ether_dhost);
if(dest_iface){
    iface_send_packet(dest_iface, packet, len);
}
else{
    broadcast_packet(iface, packet, len);
}

// update source
insert_mac_port(eh->ether_shost, iface);
```

（二）测量交换机转发性能

注意在测试性能前先注释掉代码中的 debug 打印信息，避免打印占据较多时间而对传输速率产生影响。

1、h1 同时向 h2 和 h3 测量

网络拓扑结构与上次实验一致，如下图所示：



这次由 h1 同时向 h2、h3 发送数据，测试实际传输速率。

"Node: h1"

```
root@joker-linux:/mnt/hgfs/share/05-switching# iperf -c 10.0.0.2 -t 30
------------------------------------------------------------
Client connecting to 10.0.0.2, TCP port 5001
TCP window size:  170 KByte (default)
------------------------------------------------------------
[ 13] local 10.0.0.1 port 59668 connected with 10.0.0.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 13]  0.0-30.0 sec  34.6 MBytes  9.67 Mbits/sec
```

"Node: h1"

```
root@joker-linux:/mnt/hgfs/share/05-switching# iperf -c 10.0.0.3 -t 30
------------------------------------------------------------
Client connecting to 10.0.0.3, TCP port 5001
TCP window size:  170 KByte (default)
------------------------------------------------------------
[ 13] local 10.0.0.1 port 47214 connected with 10.0.0.3 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 13]  0.0-30.0 sec  34.4 MBytes  9.60 Mbits/sec
```

我们可以看到，h1 到 h2 实际传输速率为 9.67Mb/s，而 h1 到 h3 实际传输速率为 9.60Mb/s，都比较接近各自带宽上限 10Mb/s。多测试几组数据如下，取平均值。

	h1 -> h2 传输速率（Mb/s）	h1 -> h3 传输速率（Mb/s）

1	9.67	9.60
2	9.71	9.11
3	9.34	9.72
4	9.72	9.28
5	9.52	9.66
平均值	9.59	9.47

我们可以看到，h1 到 h2 平均传输速率为 9.59Mb/s，而 h1 到 h3 平均传输速率为 9.47Mb/s，都基本接近各自带宽。可以说各自链路都充分利用了各自带宽，这是因为交换机在第一次传输时学到了端口与 MAC 地址的映射关系，之后都可以只向对应端口转发数据包。没有广播那样额外的数据传输，充分利用了带宽，网络效率较高。

2、h2 和 h3 同时向 h1 测量

这次由 h2、h3 同时向 h1 发送数据，测试实际传输速率。

多测试几组数据如下，取平均值。

	h2 -> h1 传输速率（Mb/s）	h3 -> h1 传输速率（Mb/s）
1	9.19	9.31
2	9.30	9.35
3	9.22	9.28
4	9.26	9.28
5	9.43	9.41
平均值	9.28	9.33

我们可以看到，h2-h1 和 h3-h1 的实际传输速率都接近 10Mb/s，几乎完全利用了带宽。

与上一种情况类似，在交换机学到映射关系后，只向对应端口发送数据包，而 b1-s1 链路带宽为 20Mb/s，刚好可以接收 2 个同时满带宽的 10Mb/s 数据。因此在这种情况下，各链路带宽完全利用，网络效率较高。

3、交换机转发与集线器广播的性能对比

对于集线器广播方式，从上次实验中可以得知，h1 向 h2 和 h3 发送时，速率大约为 6.55Mb/s、3.44Mb/s，总体利用率只有带宽的一半。这是因为广播方式会将数据包向所以端口发送，占用其他链路的带宽。

交换机在第一次转发时，也是广播方式。但之后学习到 MAC 地址和端口的对应关系之后，就只向目的端口发送数据包，只占用该链路带宽。因此总体上利用率可以拉满，实际传输速率接近满带宽。

而对于 h2 和 h3 同时向 h1 发送数据时，集线器和交换机没有什么差异，都能完全利用带宽。

综上，交换机的性能较好，它在学习完毕后可以定向发送数据包，没有无用数据挤占带宽。同时它不受传输方向的影响，上下行效率一致。而集线器效率受传输方向影响很大，上下行不对等，坏的情况下效率很低。此外在节点更多后无用数据会更加挤占带宽，性能受到局限。

四、思考题

1. 交换机在转发数据包时有两个查表操作：根据源 MAC 地址、根据目的 MAC 地址，为什么在查询源 MAC 地址时更新老化时间，而查询目的 MAC 地址时不更新呢？

（1）在查询目的 MAC 地址时查到，只说明之前学到过该条对应关系，不能说明当前该对应关系仍然成立，不能更新老化时间。而源 MAC 地址和端口的对应关系是当前成立的，需要添加到转发表中。如果在表中查到，说明以前学到过源地址与端口的对应关系，可以复用该表项，于是更新老化时间。

（2）假设某地址向该目的地址持续发数据，而如果此时目的地址的主机更换了端口，那么交换机会根据查到的转发表项还是往原来的端口发送数据。正常情况下不在查询目的地址时更新老化时间，那么该主机更换端口后一直收不到数据，无回应数据包，该目的地址的表项就会老化消失。这时交换机查不到表项，于是采用广播方式发送数据包。新端口就会受到数据包，之后发送回应包，这时交换机就能学到新的端口映射关系。

但如果查询目的地址时更新老化时间，那每次向目的地址发送数据包都会更新时间，那原来的对应关系就会在转发表中一直存在，持续向旧端口发数据，无法获得新的端口映射关系，导致连接不上目的地址。

2. 网络中存在广播包，即发往网内所有主机的数据包，其目的 MAC 地址设置为全 0xFF ，例如 ARP 请求数据包。这种广播包对交换机转发表逻辑有什么影响？

网络中没有设备的 MAC 为全 0xff，而交换机只根据源 MAC 地址来学习，因此转发表中不存在全 0xff 地址对应的表项。于是对于目的 MAC 地址全 0xff 的广播包，交换机总是会将其广播发送。广播包能正常发送，不会对交换机产生影响。

3. 理论上，足够多个交换机可以连接起全世界所有的终端。请问，使用这种方式连接亿万台主机是否技术可行？并说明理由。

理论上可以，但实际上不可行。首先转发表是有限的，因为内存（硬盘）总是有限的，没有交换机能装下亿万个表项。那么除了老化这样被动地移除，交换机还需要在转发表装满时主动地替换表项。由于实际主机数目远超过转发表表项数目，很可能在很长一段时间内，接收的都是查询不到目的 MAC 地址的数据包，这样交换机只能选择广播发送。这样交换机将退化为集线器，性能大大降低。

此外，面对这样一张庞大的转发表。即使用上 hash 链表，查询也将消耗大量时间。此外老化、主动替换也将耗费大量时间，交换机处理能力将大大降低。

五、实验总结

通过本次实验，我对交换机及其工作原理有了一定的了解。首先，我学到了交换机的工作方式，通过转发表来学习 MAC 地址与端口的对应关系，以此优化转发。然后，我了解到转发表的组织结构，以及表上的一些基本操作。之后，我现实了转发表和交换机，并通过 iperf 进行测试，对比分析了集线器和交换机的性能差异。最后，通过一些思考题，我对交换机的一些细节有了更准确的认识，这让我对交换机有了更深的理解。