《现代交换原理》实验报告

实验名称 **MPLS 编程实验**

班　　级 **2020211310 班**

学　　号 **2020211616**

姓　　名 付容天

指导教师 赵　学　达

一、实验目的

安排的三个编程实验主要用于加强学生对 MPLS 交换中标记请求、标记分配与分发、标记分组转发的理解。

二、实验内容

首先，我们需要了解多协议标记交换（Multiple Protocol Labeled Switching）的概念。MPLS 技术是将第二层交换和第三层路由结合起来的一种 L2/L3 集成数据传输技术。由于 MPLS 是一项面向连接的交换技术，因此有建立连接的过程。各个 MPLS 设备运行路由协议，在标记分发协议 LDP 的控制下根据计算得到的路由在相邻的路由器进行标记分配和分发，从而通过标记的拼接建立起从网络入口到出口的标记交换路径 LSP。

在数据转发的过程中，入口标记路由器 LER 根据数据流的属性比如网络层目的地址等将分组映射到某一转发等价类 FEC，并为分组绑定标记。核心标记交换路由器 LSR 只需根据分组中所携带的标记进行转发即可。出口标记路由器 LER 弹出标记，根据分组的网络层目的地址将分组转发到下一跳。MPLS 节点（MPLS 标记交换路由器 LSR 或 MPLS 边缘路由器 LER）均要创建和维护传统的路由表和标记信息库 LIB。

路由表记录记录路由信息，用于转发网络层分组和标记分发从而建立标记交换路径。LIB 记录了本地节点分配的标记与从邻接 MPLS 节点收到的标记之间的映射关系，用于标记分组的转发。

总的来说，MPLS 技术的核心实质在于：（1）网络中分组基于标记的转发（2）LDP 协议控制下的进行标记分发从而建立标记交换路径 LSP。实验网络的拓扑结构（节点分布示意图）如下所示：

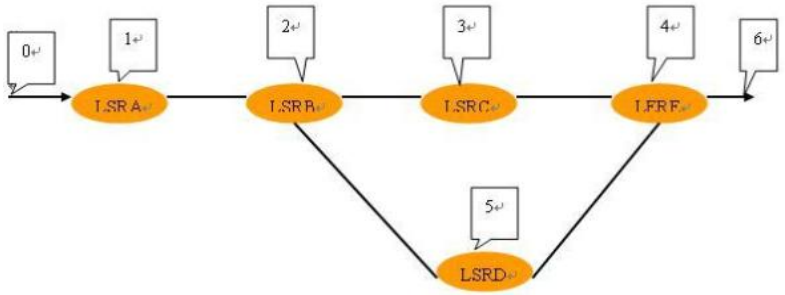

图 1：MPLS 编程实验拓扑图

在本次实验中主要用到的数据结构包括：

（1） 发送的请求信息数据结构 ReqType；

（2） 路由表表项数据结构 routetype；

（3） 标记信息表表项数据结构 libtype；

（4） 发送的标记信息数据结构 LabelPack 和 funcusedtype；

（5） 发送的标记分组信息数据结构 LabelledDataPack；

（6） 标记分组类型数据结构 MessageType。

上述数据结构的详细信息可在实验指导书或软件上查看得到。

在本次实验中，我们需要完成三个部分的实验：

（1） 标记请求实验：编写完整的 req_process 函数，使 MPLS 过程中的标记请求阶段可以正确完成；

（2） 标记分配与转发实验：编写完整的 label_process 函数，使 MPLS 过程中的标记分配与转发阶段可以正确完成；

（3） 标记分组转发实验：编写完整的 pack_process 函数，使 MPLS 过程中的标记分组转发过程可以正确完成。

三、实验过程、源代码及实验结果

对于第一个实验（标记请求实验），我编写的源代码如下所示：

```
#include "mplsconstant.h"
extern "C" _declspec(dllexport) struct ReqType req_process(int idnow,
struct routertype routenow) {
    ReqType someReqType;
    someReqType.iFirstNode = idnow;
    someReqType.iEndNode = routenow.nexthop;
    someReqType.ipaddress = routenow.ipaddress;
    return someReqType;
}
```

在上面的代码中，我将 someReqType 的第一个节点设为了当前的节点号 idnow，并将最后一个节点设为了当前所指的路由表表项 routenow 的下一条节点号 nexthop，然后指定了 ip 地址为 routenow 的 ip 地址。这样，函数 req_process 便能够根据提供的当前节点号和路由表表项值产生标记请求包。

实验结果如下图所示：

图 2-1：建立 E 路由表

图 2-2：建立 C 路由表

图 2-3：建立 B 路由表

图 2-4：建立 A 路由表且传输

图 2-5：传输经过 B

图 2-6：传输经过 C

可以看到，建立从 Local Host_An 到 Local Host_B1 的路由正确（路由表 A、B、C、E 均正确），并且在数据传输阶段成功将信息从源转达到了目的地，完美满足了既定的实验要求！

对于第二个实验（标记分配与转发实验），我编写的源代码如下所示：

```
#include "mplsconstant.h"
extern "C" _declspec(dllexport) struct funcusedtype label_process(struct
routertype routenow, int labelout, int idnow) {
    struct funcusedtype somefuncused;
```

```
    somefuncused.libinfo.ipaddress = routenow.ipaddress;
    somefuncused.libinfo.inpoint = routenow.inpoint;
    somefuncused.libinfo.outpoint = routenow.outpoint;
    somefuncused.libinfo.inlabel = 7;
    somefuncused.libinfo.outlabel = labelout;
    somefuncused.labelinfo.iFirstNode = idnow;
    somefuncused.labelinfo.iEndNode = routenow.lasthop;
    somefuncused.labelinfo.labelvalue = somefuncused.libinfo.inlabel;
    return somefuncused;
}
```

在上面的代码中，我首先定义了标记信息数据 somefuncused（数据结构为 funcusedtype），接着使用当前所指的路由表表项 routenow 的相关数据初始化了 spmefuncused，并自定义 inlabel 为 7。这样，该函数便可以根据提供的路由表当前表项、分配的输出标签号和当前节点号，构造一个 funcusedtype 信息包。

实验结果如下图所示：

图 3-1：建立 E 路由表

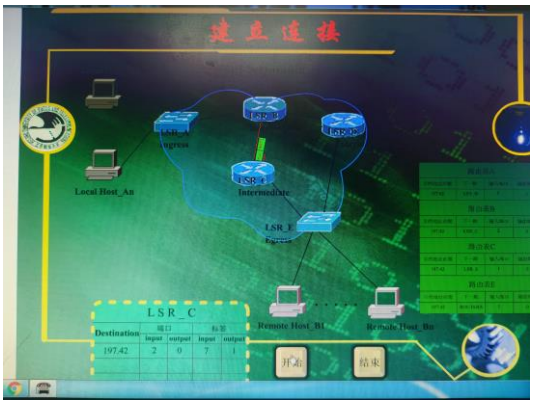

图 3-2：建立 C 路由表

图 3-3：建立 B 路由表

图 3-4：建立 A 路由表且传输

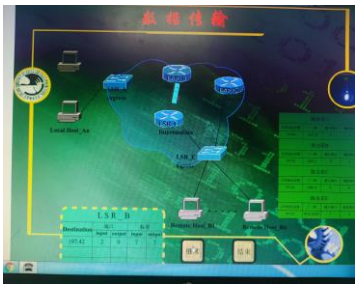

图 3-5：穿过路由器 B

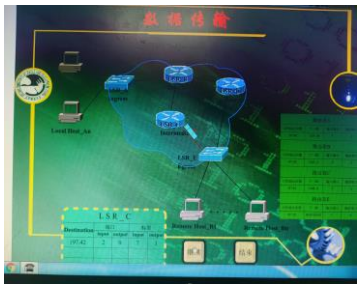

图 3-6：穿过路由器 C

图 3-7：穿过路由器 E

可以看到，建立从 Local Host_An 到 Local Host_B1 的路由正确（路由表 A、B、C、E 均正确），并且在数据传输阶段成功将信息从源转达到了目的地，完美满足了既定的实验要求！

对于第三个实验（标记分组转发实验），我编写的源代码如下所示：

```
#include "mplsconstant.h"
extern "C" _declspec(dllexport) struct LabelledDataPack
pack_process(struct routertype routenow, struct libtype libnow, int
idnow) {
    struct LabelledDataPack someLabelledDataPack;
    someLabelledDataPack.iFirstNode = idnow;
    someLabelledDataPack.iEndNode = routenow.nexthop;
    someLabelledDataPack.DataInfo.ipaddress = routenow.ipaddress;
    someLabelledDataPack.DataInfo.labelvalue = libnow.outlabel;
    return someLabelledDataPack;
}
```

在上面的代码中，我通过设置待转发分组 someLabelledDataPack 的各项属性，使待转发分组可以沿着指定路径、根据相应的路由表信息构造出相应的数据包，实验结果如下图所示：

图 4-1：建立 E 路由表

图 4-2：建立 C 路由表

图 4-3：建立 B 路由表

图 4-4：建立 A 路由表且传输

图 4-5：穿过路由器 B

图 4-6：穿过路由器 C

图 4-7：穿过路由器 E

可以看到，建立从 Local Host_An 到 Local Host_B1 的路由正确（路由表 A、B、C、E 均正确），并且在数据传输阶段成功将信息从源转达到了目的地，完美满足了既定的实验要求！

四、实验心得

在本次实验中，我复习了多协议标记交换（MPLS）的基本知识，动手编写了 MPLS 编程实验的三个函数，完成了本实验的三个部分，在理论知识和具体实践上都收获满满！