**第5章 网络层实验**

**实验1编程：**基于ICMP的ping程序

配置文件关键要点：

无，对方的IP地址以命令行参数的形式提供。

程序运行屏幕输出要点：

参见windows ping命令的输出。

**实验2编程：**基于ICMP的traceroute程序

配置文件关键要点：

无，对方的IP地址以命令行参数的形式提供。

程序运行屏幕输出要点：

参见windows tracert命令的输出。

**实验3 编程：**距离矢量（DV）算法

编写一个基于DV的路由算法，定期更新路由表。要求：

要求：

1，定义路由表（routing table）结构

2，程序启动后，首先读入配置文件中的二维矩阵或数组，初始化每个结点的路由表。

3，**按任意键** 或 **定期** 更新所有结点的路由表。

4，设置输入框，输入A，B，C，D，E或ALL，输出**任何一个指定结点 或 所有结点** 的路由表。

5，每个结点计算收敛次数，即迭代更新了多少次，各个路由表的内容与网络状态一致。

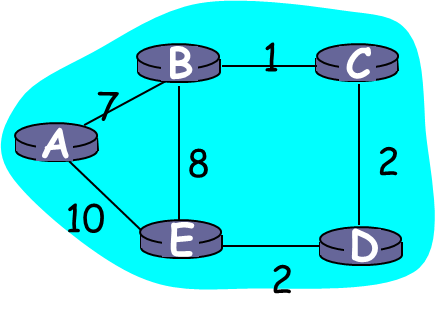
6，手动设置，将某条路由的距离调小（好消息），重新计算收敛次数。

7，手动设置，将某条路由的距离调大（坏消息），重新计算收敛次数。

配置文件关键要点：

二维矩阵（或数组）：表示网络中**各路由结点**与其他路由结点的邻接关系。若两个结点直接相邻，则对应元素值为这两个结点之间的距离（相邻结点之间的最小距离为1）。若两个结点不直接相邻，则对应元素值为空（或16，表示无穷大）。例如，某网络的拓扑结构及Erwin矩阵如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| A | 0 | 7 |  |  | 10 |
| B | 7 | 0 | 1 |  | 8 |
| C |  | 1 | 0 | 2 |  |
| D |  |  | 2 | 0 | 2 |
| E | 10 | 8 |  | 2 | 0 |



程序运行屏幕输出要点：

可以采用二维矩阵或数组输出的各结点的路由表。形式：

A结点路由表：

|  |  |  |
| --- | --- | --- |
|  | Distance | Next-hop |
| A | 0 | - |
| B |  |  |
| C |  |  |
| D |  |  |
| E |  |  |

**实验4 编程：**软路由器（Soft Router）

基于实验3，编写一个基于DV的路由算法的路由器。

要求：

1，使用UDP与邻居路由结点通信。

2，配置一个一维数组，指示每个结点向外通告其路由表时所使用的UDP端口号。

3，设计一个多进程或多线程程序。主程序启动后，应根据配置的结点，分别启动多个进程或线程，每个进程或线程代表一个路由器。每个进程或线程启动后，应读取二维数据，初始化路由表。读入一维数组，确定发送路由表的UDP端口，进而启动监听线程、路由更新线程（包括定时器）、分组转发线程、控制台线程等，分别用于路由的发布、计算更新、分组的转发，和响应管理员在控制台发出的操作命令。

4，控制台支持以下输入命令：

RT：显示输入路由表

SP：发送分组

ST：显示统计信息

3，除了满足实验三的各项要求外，路由器应能实现分组的转发。分组基本结构定义如下：

（1）源地址

（2）目的地址

（3）TTL：最多结点数。每经过一个结点，TTL值减一。若TTL=0，则丢弃该分组，不再转发。

（4）数据（可选）

4，为指定的结点输入待转发的分组，分别输入：源地址，目的地址，TTL值和数据。输入完毕后，启动分组的转发。

5，每个结点记录其所接收、转发、丢弃的分组及其数量。

配置文件关键要点：

|  |  |
| --- | --- |
| 结点 | UDP端口号 |
| A | 52001 |
| B | 52002 |
| C | 52003 |
| D | 52004 |
| E | 52005 |

程序运行屏幕输出要点：

1，路由表：结构如实验三

2，统计信息：如下：

？？？（结点名称）路由器

接收分组数：xxxxxx

分组1：源地址， 目的地址， TTL， 数据

分组2：源地址， 目的地址， TTL， 数据

…

分组n：源地址， 目的地址， TTL， 数据

转发分组数：xxxxxx

分组1：源地址， 目的地址， TTL， 数据

分组2：源地址， 目的地址， TTL， 数据

…

分组n：源地址， 目的地址， TTL， 数据

丢弃分组数：xxxxxx

分组1：源地址， 目的地址， TTL， 数据

分组2：源地址， 目的地址， TTL， 数据

…

分组n：源地址， 目的地址， TTL， 数据