# *Exp 10*：Network Disturbance 网络干扰分析

**目的**：让学生了解实际网络环境中的network disturbance并举例，让学生在下一章节的Error Control能更容易上手。

**摘要**：Packet lost、Packet delay 和Packet errors 是现实的网络传输中最常遇见的问题，所以本实验里我们将模拟实际因特网中的环境。

**时间**：3 hrs。

## 一、网络拓扑

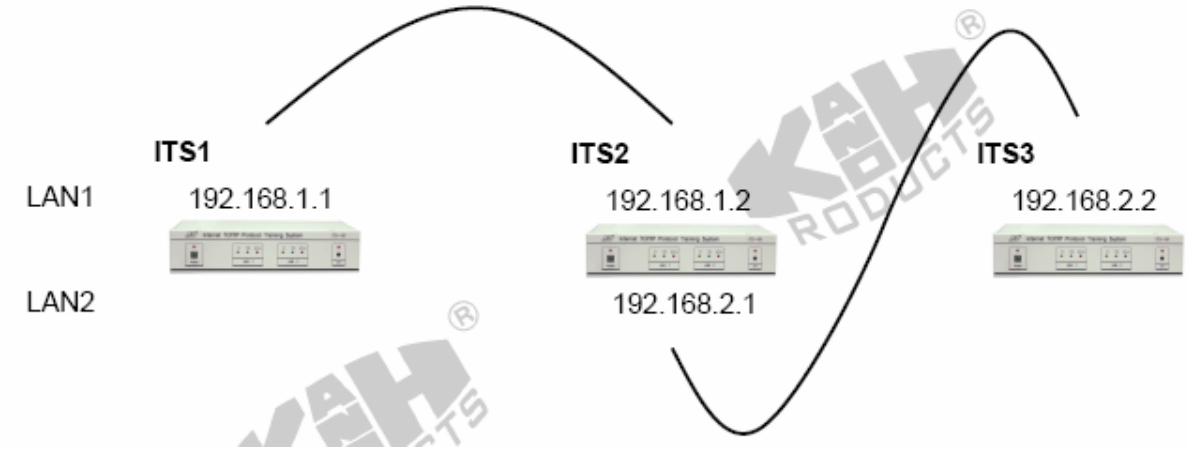


图 10.1

## 二、技术背景

本实验利用以下几个状态来描述一个路由器可能遇到的网络干扰：

- Packet Lost：网络联机时封包遗失。
- Packet Delay：网络联机时封包延迟 (可用观察方法：观察有多少重复的回应封包)。
- Packet Error：网络联机时封包损坏。
- Bandwidth：在单一时间点同一网络联机内可发送的封包数量及大小。

## 三、实验步骤

### 1、网络拓扑连线

1）在Hubox上将网络连线如图10.2所示。

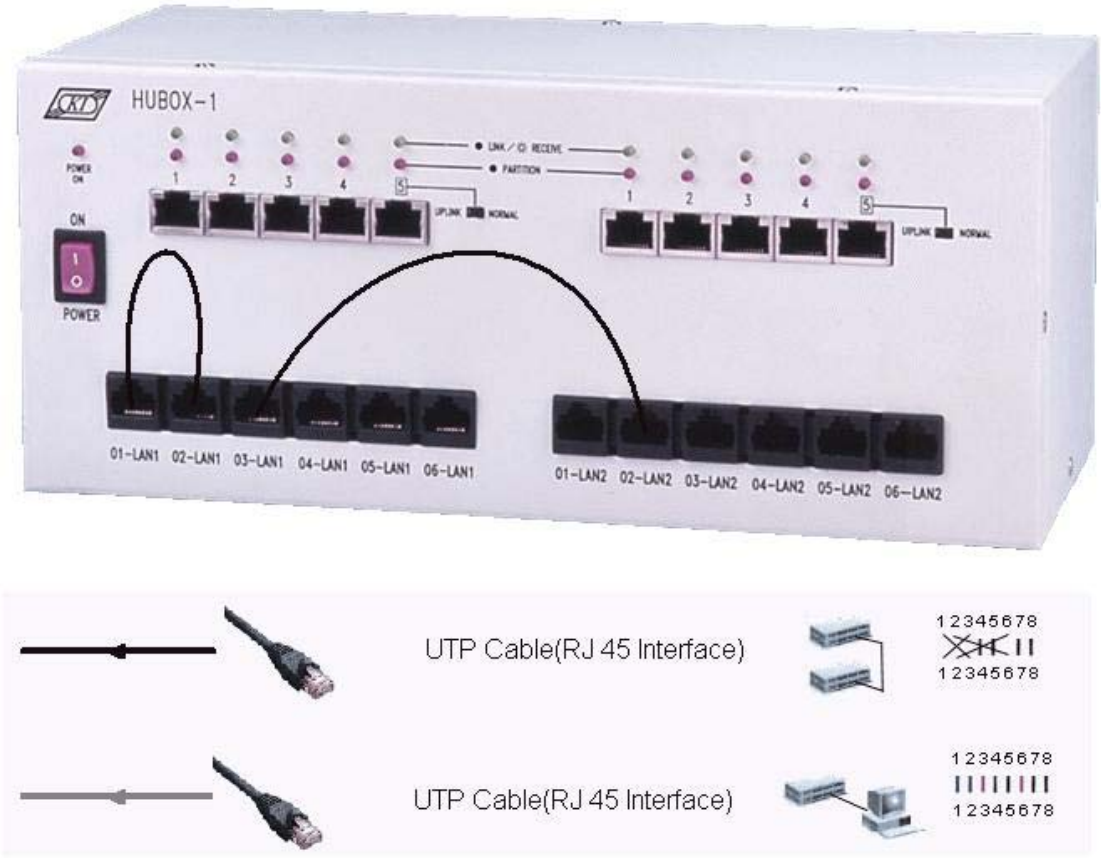


图 10.2

**2、利用MDDL制造网络干扰环境**

A. 设定 Host 和 Gateway

1）运行 **XCLIENT.BAT** 程序，打开 ITS 软件界面（KCodes Network Explorer）。

2）从 Tool menu 中选择 **Network Configuration** ，打开网络属性设置界面。

ITS1 (Host)

3）参照网络拓扑 A, 输入 “**192.168.1.1**” 到 **IP Setting Address of Interface 1** 文本框中，并单击 **Add new routing entry** 按钮，见图 10.3。

4）输入 “**192.168.2.0**” 到 Destination 文本框中, 输入“**255.255.255.0**” 到 Mask 文本框中, 并且输入 “**192.168.1.2**” 到 Gateway 文本框中(见图 10.4), 最后点击 **Update** 按钮。

5）选择 **Host** 模式并且点击 **Set & Close** 按钮。

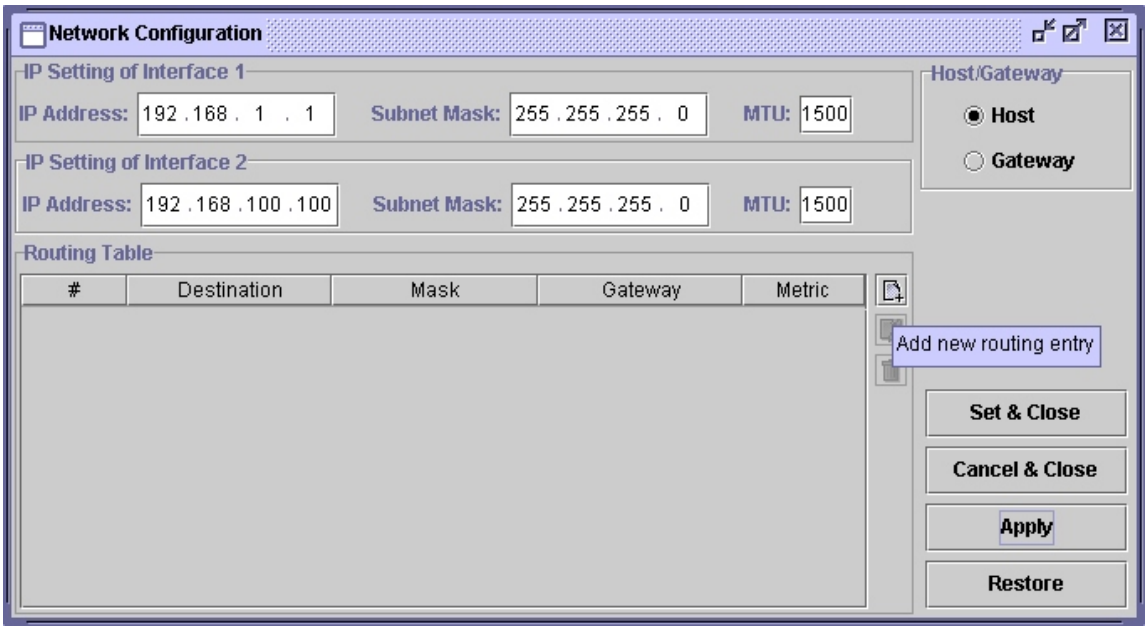


图 10.3

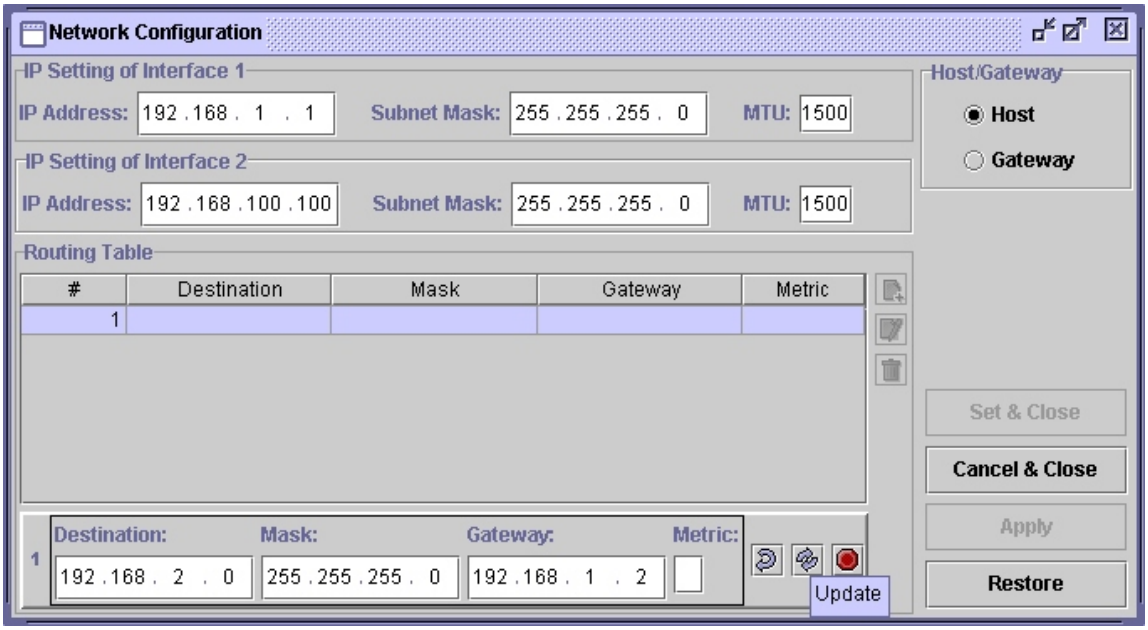


图 10.4

ITS3 (Host)

6）和 ITS1 相同步骤，先输入 “**192.168.2.2**” 到 **IP Setting Address of Interface 1** 文本框中，并点击 **Add new routing entry** 按钮。

7）输入 “**192.168.1.0**” 到 Destination 文本框, 输入“**255.255.255.0**” 到 Mask 文本框, 并且输入“**192.168.2.1**”到 Gateway 文本框，最后点击 **Update** 按钮。

8）选择 **Host** 模式，并且单击 **Set & Close** 按钮。

ITS2 (Gateway)

9）参照网络拓扑 A, 输入“**192.168.1.2**” 到 **IP Setting Address of Interface 1** 文本框

中，输入到“**192.168.2.1**” into **IP Setting Address of Interface 2** 文本框中(见图 10.5)。

10）选择 **Gateway** 模式，并点击 **Set & Close** 按钮。

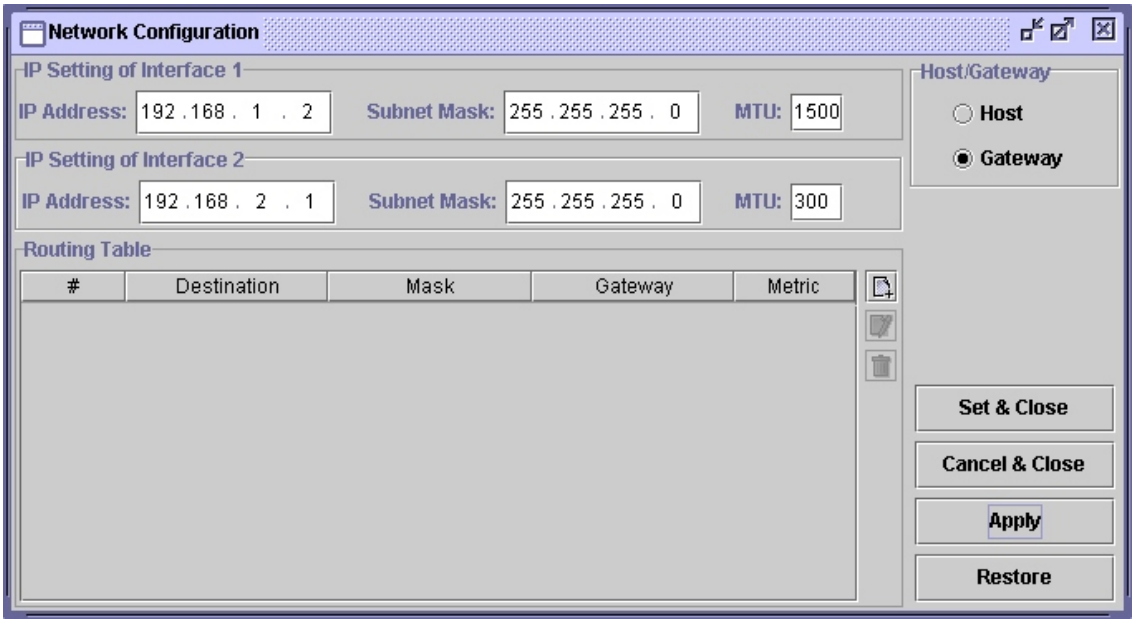


图 10.5

B. 制造网络丢包的环境

ITS2

11）打开一个新的网络信息浏览器（Network Message Browser），勾选 **Listening On**。

12）从 Reactor menu 中选择 **MDDL Reactor Panel** ，打开 mddl 编辑平台。

13）点击 **Load** 按钮。调用 PktLost.mddl 程序（路径为 C: \XClient \Data \Mddl \Tutorial \Ex10 \PktLost.mddl），然后点击 **Upld** 按钮 (见图 10.6)。

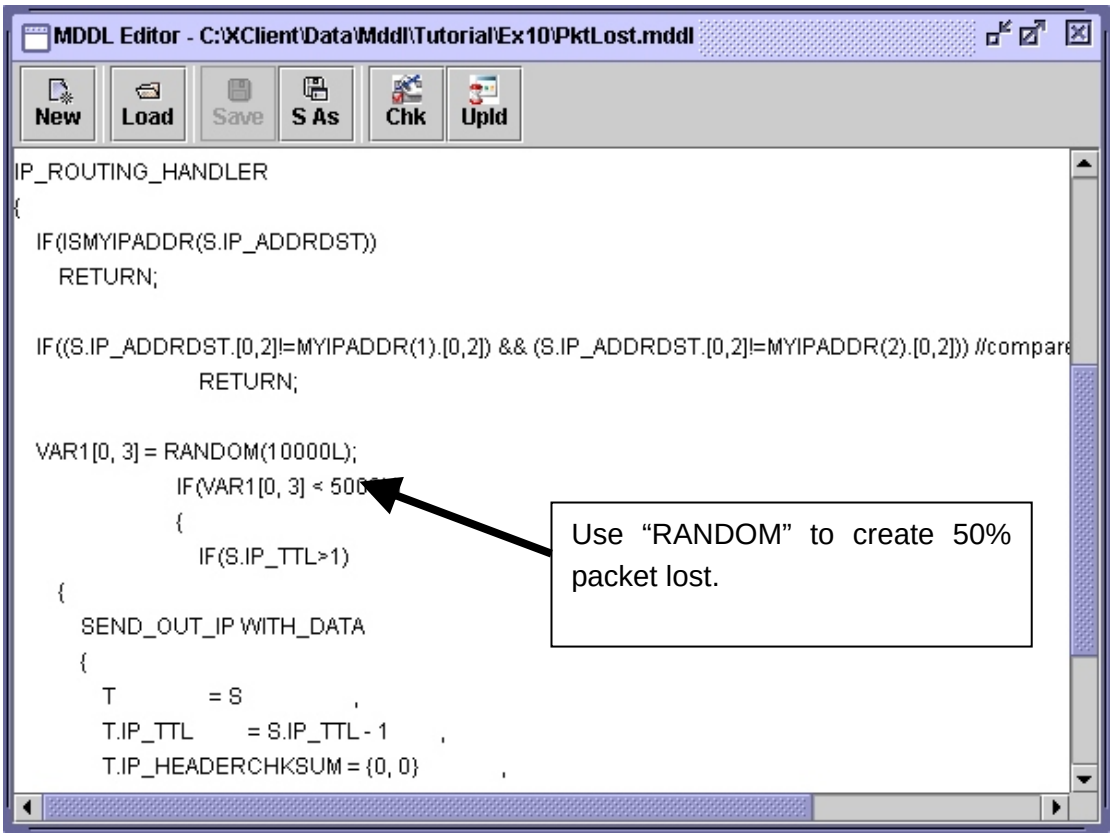


图 10.6

ITS3

14）打开一个新的网络信息浏览器（Network Message Browser），勾选 **Listening On**。

ITS1

15）打开一个新的网络信息浏览器（Network Message Browser），勾选 **Listening On**。

16）参考前面的实验，发送数个 ICMP Echo Request 报文给 ITS3。你会发现有 50%的报文将被 ITS2 丢弃。

C. 制造报文延迟的环境

ITS2

17）复位网络信息浏览器（ Network Message Browser ）。

18）从 Reactor menu 中选择 **MDDL Reactor Panel** ，打开 mddl 编辑平台。

19）点击 **Load** 按钮。调用 PktDelay.mddl 程序（路径为 C: \XClient \Data \Mddl \Tutorial \Ex10 \PktDelay.mddl），然后点击 **Upld** 按钮。此程序运行后，通过 ITS2 的报文将会有 50%的概率被延迟 1 秒。

ITS1 和 ITS3

20）复位网络信息浏览器（Network Message Browser）。

21）参考前面的实验，由 ITS1 发送数个 ICMP Echo Request 报文给 ITS3。我们可以观察到传输过程中的延迟现象。

D. 制造特定的报文丢失环境

ITS2

22）复位网络信息浏览器（Network Message Browser）。

23）从 Reactor menu 中选择 **MDDL Reactor Panel** ，打开 mddl 编辑平台。

24）点击 **Load** 按钮。调用 PktLost4.mddl 程序（路径为 C: \XClient \Data \Mddl \Tutorial \Ex10 \PktLost4.mddl），然后点击 **Upld** 按钮。此程序运行后，从 ITS1 发出每 5 个报文中，第四个报文都将被 ITS2 丢弃。

ITS1 和 ITS3

25）复位网络信息浏览器（Network Message Browser）

26）参考前面的实验，由 ITS1 发送数个 ICMP Echo Request 报文给 ITS3 我们可以观察到传输过程中的丢包现象。

## 四、实验讨论

1、试使用MDDL写一个能够有20%机率让封包遗失的子网络绕送程序。

2、在制造随机数的封包延迟这段实验里，试着改用计算机主机的Command Prompt，使用ping 指令侦测到ITS 2并从网络讯息浏览器观察封包延迟现象。

## *REACTOR PROGRAMS*

### 1、PktLost.mddl

```
IP_ROUTING_HANDLER
{
    IF(ISMYIPADDR(S.IP_ADDRDST))
        RETURN;
  IF((S.IP_ADDRDST.[0,2]!=MYIPADDR(1).[0,2])&&(S.IP_ADDRDST.[0,2]!=MYIPADDR(2).[0,2]))
```

```
                                                    //compare with netmask 255.255.255.0
        RETURN;
    VAR1[0, 3] = RANDOM(10000L);
  IF(VAR1[0, 3] < 5000L)
  {
    IF(S.IP_TTL>1)
      {
          SEND_OUT_IP WITH_DATA
          {
              T = S ,
              T.IP_TTL = S.IP_TTL - 1 ,
              T.IP_HEADERCHKSUM = {0, 0} ,
              T.IP_HEADERCHKSUM = CHECKSUM(T.IP_HEADER)
          }
    }
  }
  DISCARD_MESSAGE;
}
```

**2、PktDelay.mddl**

```
    IP_ROUTING_HANDLER
    {
     IF(ISMYIPADDR(S.IP_ADDRDST))
     RETURN;
    IF((S.IP_ADDRDST.[0,2]!= MYIPADDR(1).[0,2])&&(S.IP_ADDRDST.[0,2]!=MYIPADDR(2).[0,2]))
                                                    //compare with netmask 255.255.255.0
     RETURN;
     VAR1[0, 3] = RANDOM(10000L);
     IF(VAR1[0, 3] < 5000L)
     {
          IF(S.IP_TTL>1)
```

```
        {
            SEND_OUT_IP WITH_DATA
            {
                T = S ,
                T.IP_TTL = S.IP_TTL - 1 ,
                T.IP_HEADERCHKSUM = {0, 0} ,
                T.IP_HEADERCHKSUM = CHECKSUM(T.IP_HEADER)
            }
        }
    }
    ELSE IF(VAR1[0, 3] < 10000L)
    {
            ADD_TO_POOL 22 WITH_LIFETIME 20000 WITH_DATA
            {
                T.[0] = 10 ,
                T.[6, ] = SOURCE
            }
    }
    DISCARD_MESSAGE;
    }
    TIMER_WITH_PERIOD 100
    {
        FOR_EVERY_ELEMENT_IN_POOL 22
        {
         PE[0] = PE[0] - 1;
         IF(PE[0] == 0)
         {
            SEND_OUT_IP WITH_DATA
            {
                TARGET = PE[6, ],
```

```
                    T.IP_TTL = PE.IP_TTL - 1,
                    T.IP_LEN = LENGTH(T),
                    T.IP_HEADERCHKSUM = {0, 0} ,
                    T.IP_HEADERCHKSUM = CHECKSUM(T.IP_HEADER)
                }
                 REMOVE_CURRENT_POOL_ELEMENT;
            }
    }
}
```

### 3. PktLost4.mddl

```
VAR1[0] = 0;

IP_ROUTING_HANDLER
{
 IF(ISMYIPADDR(S.IP_ADDRDST))
  RETURN;

  IF(S.IP_ADDRDST.[0,2]!= MYIPADDR(2).[0,2]) //compare with netmask 255.255.255.0
  RETURN;

  VAR1[0] = VAR1[0]+1 ;

  GENERATE_USER_MSG WITH_DATA
  {
      TARGET = VAR1[0]
  }
  IF(VAR1[0] !=4)
  {
  IF (VAR1[0] == 5)
  {
```

```
            VAR1[0] = 0 ;
        }
        IF(S.IP_TTL>1)
        {
            SEND_OUT_IP WITH_DATA
            {
                    T = S ,
                    T.IP_TTL = S.IP_TTL - 1
                    T.IP_HEADERCHKSUM = {0, 0} ,
                    T.IP_HEADERCHKSUM = CHECKSUM(T.IP_HEADER)
            }
        }
}
 DISCARD_MESSAGE;
}
```