第28卷 第6期 计 算 机 工 程 2002年6月
Vol.28 № 6 Computer Engineering June 2002
· 工程应用技术与实现 · 文章编号：1000—3428(2002)06 —0208—03 文献标识码：A 中图分类号： TP393.09

局域网上动态视频实时传输技术的研究

江 冰，张金波，张学武，奚 吉
(河海大学(常州分校)计算机及信息工程学院，常州 213022)

摘 要：介绍一种在局域网上实时传输视频的策略，并给出实现方法，该方法成功地应用于远程视频集散监控系统中。
关键词：局域网；动态视频；实时传输

Research on Real-time Transmitting of Active Video over LAN

JIANG Bing，ZHANG Jinbo，ZHANG Xuewu，XI Ji
(College of Computer & Information Engineering, Hehai Univ., Changzhou 213022)

【Abstract】 This paper introduces a kind of tactic of real-time transmitting dynamic video over LAN, and provides method of realizing, which is applied on the remote video distributing centre monitoring and controlling system.
【Key words】 LAN；Dynamic video；Real-time transmitting

随着计算机网络和通信技术的发展，监控系统将朝着网络化、智能化的方向发展。在我们研究开发的远程视频集散监控系统中，二级分控主机(作为服务器)与各现场控制主机之间(作为客户机)是利用TCP/IP协议实现视频实时传输。TCP/IP是使用最广泛的协议族，研究在此协议上的动态视频实时传输具有十分重要的意义。

1 关键技术

1.1 基于WindowSocket套接字的种类及选择

首先Windows针对网络套接字提供两个类支持，即CasyncSocket和Csocket。CasyncSocket(异步套接字)是Csocket的基类，提供了数据报和面向流两种方案。异步套接字属于低层套接字，由于它提供了更大的灵活性，在大数据量的多媒体通信中被经常采用。Csocket在CasyncSocket的基础上封装了堵塞的功能。“堵塞”保证了数据在网络上可靠稳定的传输。给一般的编程人员提供了很大的方便。但是在进行堵塞调用的时候，线程将被堵塞，同时其他的工作也不得不停下来等待堵塞调用返回。整个进程不能充分利用CPU时间。其次,像视频信息，其特点是数据量大，监控要求实时性很高，在某些情况下，为了盲目追求数据传输的可靠性而放弃了实时性，往往得不偿失。当然，最理想的方法就是在可靠性和实时性之间寻求平衡点，使系统最大限度地使用CPU，发挥效能。采用的方案是：以CasyncSocket为基类，构造一个派生类，在应用层保证数据的可靠性。

1.2 视频网络传输系统流程

基于TCP/IP协议的动态视频传输是在C/S结构环境下讨论的，运行于典型的10Mbps以太网之下，此类局域网已经很成熟并且很普遍了，图1是最简单的结构，当然局域网有更多的结构，它可以很复杂，但对于套接字编程来说，这一切是完全透明的。

客户机/服务器数据交换流程如图2所示，表面上看和经典的面向流的套接字非常相似。但是具体实现截然不同，此处的套接字请求连接、接收连接请求、发送数据、数据应答等都是通过字定义的帧类型来完成的。

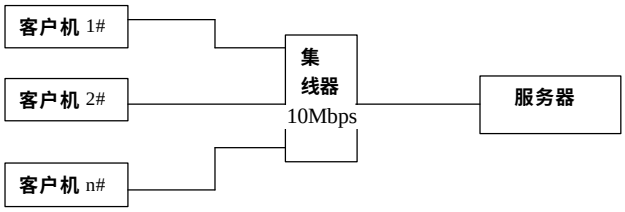

图1 客户机服务器结构

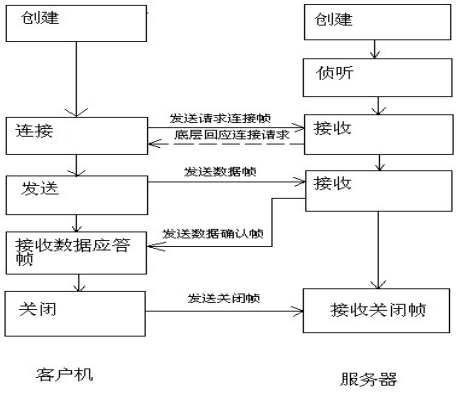

图2客户机服务器通信流程

1.3 套接字的实现方法

1.3.1 客户机/服务器的帧类别、网络状态的定义

我们使用的是面向数据报的套接字，在应用层提供可靠性的保证。如何保证呢？我们考虑在数据报的基础上再提供一层封装，包括帧头标志、帧类别、图像格式、滑动窗口大小、帧号、数据长度(有效数据)、数据区。并定义了几种类型的帧，此外还明确定义了帧结构，如图3所示。

```
//帧类别定义
#define frmData          0      //数据帧
#define frmDataAnswer    1      //数据应答帧
```

基金项目：水利部水利重点科技项目(SZ9816)
作者简介：江 冰(1960～)，女，副教授，主研数字信号处理；张金波，讲师；张学武，助教；奚 吉，硕士生
收稿日期：2001-08-05

```
#define frmPolling          2       //检测帧
#define frmPlAnswer         3       //检测应答帧
#define frmAbort            4       //异常中止帧
#define frmSysError         5       //系统错误帧
#define frmNormalColse      6       //正常关闭帧
```

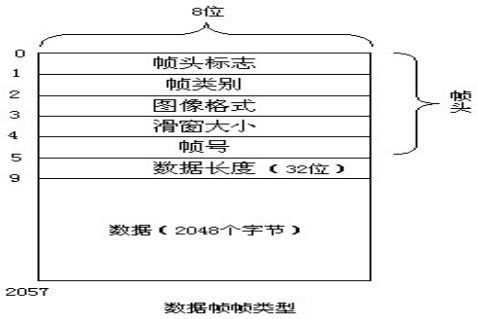

图3 数据帧类型

这个帧结构通用于所有的图像类型：128*96格式的图像，每次传输128个字节；176*144格式的图像，每次传输512个字节；352*288格式的图像，每次传输2048个字节。

这样每种视频格式分别定义最大网络传输单元的目的是为了实时的传输和显示。因为每种格式的视频在单位时间内生成的数据量大小是不同的，而且是成倍的关系，如果三种视频格式的最大传输单元大小相同，相对小的图像，很难填满缓冲区，这样会失去服务端显示的实时性；而对于相对大的图像，由于数据量很大，网络发送的频率很高，容易引起堵塞和缓冲区填满的错误状态。

同时还定义了网络套接字的10个状态，客户端和服务器端的套接字都在这10个状态之间转换。其中初始化状态、接收数据状态进行的速度很快，终端程序很难显示出来，但这些状态在程序运行过程中是存在的。

```
//网络状态
#define Vedio_STAT_NULL          0      //断开
#define Vedio_STAT_INIT          1      //初始化状态
#define Vedio_STAT_COMM          2      //通信状态
#define Vedio_STAT_STOP          3      //堵塞状态
#define Vedio_STAT_DATA          4      //发送数据状态
#define Vedio_STAT_REC           5      //接收数据状态
#define Vedio_STAT_POLLING       6      //检测状态
#define Vedio_STAT_ABORT         7      //中断状态(没有使用)
#define Vedio_STAT_SYSERROR      8      //系统错误状态
#define Vedio_STAT_CLOSE         9      //正常关闭状态
#define Vedio_STAT_WAITFORDATA   10     //等待接收数据
```

1.3.2 客户端套接字的定义及实现

在客户端要求客户能主动向服务器发送连接请求；在接收到连接请求应答帧之后，保存对方套接字的地址和端口号，系统处于通信状态；如果有数据，并且系统处于通信状态，先置客户端套接字为发送数据状态，套接字向网络发送数据帧，然后置客户端套接字为堵塞状态，这时，缓冲区中即使有数据也不能发送，必须等待对方的应答，这样就避免了大量数据淹没网络的发生；如果客户端套接字接收到服务器的数据应答帧，客户端套接字恢复到通信状态。

如果在30s之内接收不到回应则确定网络中断,关闭套接字重新连接。这样客户端套接字处于一个可控的状态转换之中不会引起系统故障。图4是客户端套接字的状态转换。

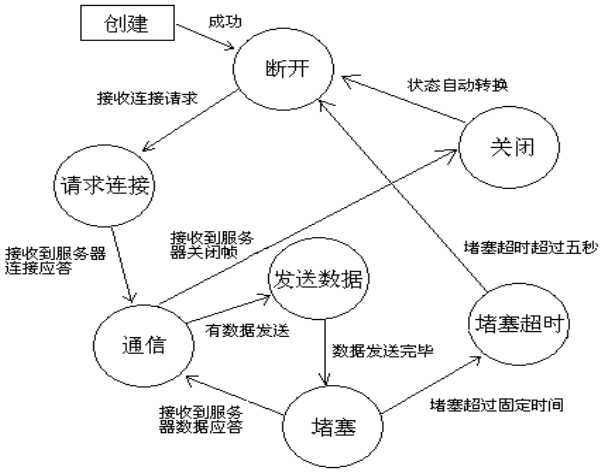

图4 客户机套接字状态转换图

1.3.3 服务器端的套接字的定义及实现

在服务器端，创建主应用程序时创建侦听套接字，系统处于等待连接状态；当有客户端连接请求到时，侦听套接字创建接收套接字来准备接收数据，如果侦听套接字创建成功，发送检测应答帧，服务器接收套接字处于等待接收数据状态；当有数据到，服务器接收套接字处于接收数据状态，服务器端接收套接字验证帧头标志、帧长度、检查帧类型。如果是数据帧就将数据存入缓冲区，向客户端套接字发送数据应答帧，服务器接收套接字处于通信状态；如果服务器接受管理人员的关闭命令，服务器端将发送关闭帧。自动关闭套接字。服务端的网络套接字的状态同样处于有限的几个可控的状态之下。服务端网络接收套接字的系统转换如图5。

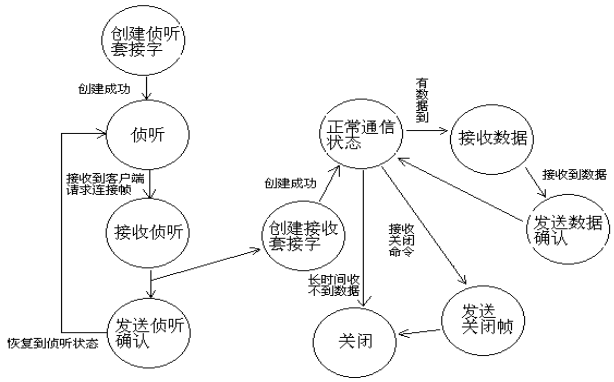

图5 服务器套接字状态转换图

1.4 网络套接字实现的特点

1.4.1自定义滑动窗口协议

CasyncSocket类不提供堵塞的功能，其中有利有弊。为了在应用层保证网络套接字双方的可靠性，上面的很多描述都是针对这个目标而进行的。这种方法与面向连接的流套接字有本质的区别，数据报数据发送以后，无法保证对方一定收到，要可靠传送，对方必须对发送信号表示确认，这种功能数据报套接字是无法实现的，必须由应用层自己完成。而面向连接的流套接字在该层保证数据的有序可靠发送时，在调用堵塞函数时系统效率会下降，若实时性要求很高时，往往可以牺牲一些网络传输的可靠性(容许丢失很少的几帧)，

而且这种要求在面向流的套接字中是无法实现的。

通过发送检测帧来代替流套接字的Connect函数，发送完检测帧后可以做其他任何事，直到收到服务器端的检测应答帧且中间没有任何的堵塞。如果使用Connect函数，由于它是一个堵塞函数，在对方应答之前调用函数的进程将被堵塞，除非连接超时。用定时器实现了超时处理。滑动窗口实现的具体策略是要能准确判断客户端、服务器端连接状态；要保证数据的正确传输，以至于服务器端能正确解压出视频信号；保证视频信号的实时传输。在能保证解压质量图像的前提下，使传输延时最小。在偶尔丢失帧的情况下，服务器端不要求重发该帧，而是让客户端发送当前的视频数据，以保证视频的实时传输。

为此，经过大量测试，发现由于压缩算法的特殊性，在传输过程中，丢掉的数据在传输总数据三分之一范围以内，客户端能正确解压出来，以此确定了传输质量，确定了数据丢失的最大上限；如果传输过程中，数据丢失率在三分之一之内，就认为传输链路正常，数据丢失率等于或大于三分之二，就认为传输链路堵塞。这样传输链路的可靠性、实时性大大增强了。具体实现方法：

(1)在客户机端定义滑动窗口大小为15，初始化为15，客户机套接字每发送一帧数据帧，窗口大小减一。如果客户端套接字在发送过程中接收到数据应答帧，客户端的滑动窗口大小重置为15，它可以继续发送下面的10帧；如果客户机套接字发送完15帧后仍然没有收到应答，客户机套接字滑动窗口大小为零，客户端套接字将处于堵塞状态，如果在接下来的10s内收到数据应答帧，套接字恢复到通信状态，如果在接下来的10s内没有收到数据应答帧，套接字将置为堵塞超时状态，5s以后网络断开。

(2)在服务器端定义滑动窗口大小为10，初始化为零，在服务端接收套接字每收到一帧数据帧，窗口大小增1。如果套接字连续接收到10帧，套接字将发送数据应答以回应，如果2s以后还没有数据收到，套接字将重发数据应答帧，如果在接下来的10s内仍然没有数据收到，网络中断，如果又收到了数据帧，网络恢复到通信状态。如果套接字收到的数据帧一直少于10帧，将导致网络状态超时，网络连接自动中断。

1.4.2 基于数据报套接字传输数据的性能分析

既然是自定义的协议，与普通的数据报、流套接字之间总有一些优缺点：

首先，一般自定义的协议使得双方的套接字在有限的几个状态之间转换，系统不可能由于套接字跑飞(套接字的状态无法捕获)而陷入瘫痪，而套接字本身是稳定的，这是开发这个协议的基础。

其次，如果网络空闲，客户端发送的数据服务端都可以有序收到，那么我们定义的协议和提供堵塞的流套接字以及不提供堵塞的数据报套接字效果是一样的。但是假设网络很忙，以至于有可能丢失数据(设100帧丢失5帧)，将会出现下列情况：

(1)套接字在客户端只要有数据就发送，网络极易瘫痪，接下来客户机、服务器之间失去联系，对方状态无法得知。

(2)流套接字在客户端陷入堵塞，直到数据被正常发送。如果其中丢掉一帧，套接字将重发该帧，套接字依然处于堵塞状态之中。

相比之下，自定义的滑动窗口协议套接字比数据报套接字可靠，比面向流的堵塞套接字更好地实现了实时性，并保证了图像质量。

2 结束语

在局域网的环境，我们成功地完成了视频(压缩之后的视频流)的实时传输，能够满足视频监控的各项指标。整个系统(包括服务器端和客户端)的软件用VC++6.0编程实现，同时在校园网上对该软件进行了测试，效果和在局域网上一样。如果要投入到因特网上使用该软件，需要在传输策略上作修改后，可以达到上述两种环境下的效果。

参考文献

1 Kuo F,Effelsberg W.龙晓苑译多媒体通信协议与实现.北京:清华大　学出版社

2 刘严明,李　鹏.实用网络编程技术西安电子科技大学出版社,1998

3 毕厚杰.多媒体信息的传输与处理北京人民邮电出版社,1999

☆☆

(上接第120页)

的，这些函数可以由Alice等用户用来为其移动代码获得签名。安全机制需要作如下修改：

· 公开密钥：用于验证签名的Alice的公开密钥需要由函数v2, …，v_k确定，这里假定Alice不想公开函数v1。

· 签名程序的构造：Alice随机选择一个有理函数$r:R^l \rightarrow R$，通过计算$f_{signed,i}:=s_i(r,G2of,Gkof),1 \le i \le k$.可以构造出映射$f_{signed}: R^l \rightarrow R^k$,Alice将二元组(f, f_{signed})发送给Bob.

· 签名程序的执行：Bob计算出$y:=f(x)$和$z:= f_{signed}(x)$，(y,z)就是签名结果。

· 签名的验证：计算$G_i(y)$和$v_i(z)$，$2 \le i \le k$,当且仅当$v_i(z)= G_i(y)$时z才是合法的签名。

以上签名机制的安全性增强的一个关键是攻击者不知道r和v_1，由于对r一无所知，因此利用分解攻击由f_{signed}中的第i项分解得到si的难度更大；由于不知道v_1，攻击者也无法计算出s_i的输入/输出二元组，因此分析攻击就可以避免了。

5 结论

通过以上分析，我们得出了一种可以减少移动代理受到恶意主机攻击的方法，从而为移动代码提供基本的软件保护，即保证代理的秘密性和完整性。通过引入有理多项式函数以及相关加密机制，可以在无交互的条件下实现程序的加密执行，以满足移动代理的生存条件。此外还介绍了签名函数隐藏方法，即将数字签名与计算函数结合起来从而保证签名不会被滥用。

参考文献

1 Abadi M, Feigenbaum J. Secure Circuit Evaluation. Journal of Cryptology,1990，2(1): 1-12

2 Chess D, Grosof B. Itinerant Agents for Mobile Computing.Technical Report RC 20010,IBM,1995

3 Feigenbaum J, Merritt M. Open Questions, Talk Abstracts and Summary of Discussions. DIMACS Series in Discrete Mathematics and Theoretical Computer Science,1991, 2:1-45

4 Meadows C. Detecting Attacks on Mobile Agents. Proceedings of the DARPA Workshop on Foundations for Secure Mobile Code, Monterey CA, USA,1997