**Windows平台远程音频**

**用户使用说明书**

版本：1.0

时间：2013年9月16日

目 录

[1. 引言 2](#_Toc367200634)

[1.1编写目的 2](#_Toc367200635)

[1.2项目背景 2](#_Toc367200636)

[1.3定义 2](#_Toc367200637)

[1.4参考资料 2](#_Toc367200638)

[2. 软件概述 3](#_Toc367200639)

[2.1软件介绍 3](#_Toc367200640)

[2.2系统功能 3](#_Toc367200641)

[2.3系统组成 3](#_Toc367200642)

[2.4 总体结构 3](#_Toc367200643)

[2.5 模块结构 5](#_Toc367200644)

[2.5.1 音频截获管理模块 AudioHook 5](#_Toc367200645)

[2.5.2 Dll注入模块 InjectDll 6](#_Toc367200646)

[2.5.3音频编码发送模块 AudioSender 7](#_Toc367200647)

[2.5.4 音频接收解码模块 AudioReceiver 8](#_Toc367200648)

[2.5.5 音频播放模块 LibAudioPlayer 9](#_Toc367200649)

[3. 运行环境 10](#_Toc367200650)

[3.1硬件环境 10](#_Toc367200651)

[3.2支持软件 10](#_Toc367200652)

[4. 使用说明 10](#_Toc367200653)

[4.1 安装 10](#_Toc367200654)

[4.2 使用 10](#_Toc367200655)

[5. 可执行文件和数据文件一览表 11](#_Toc367200656)

[6. 常见问题及其解决方法 11](#_Toc367200657)

[6.1注入dll失败 11](#_Toc367200658)

[6.2 Socket初始化失败 11](#_Toc367200659)

[6.3 客户端无法播放声音 11](#_Toc367200660)

# 1. 引言

## 1.1编写目的

本文档详细说明了Windows平台下远程音频技术，尤其适用于多用户、多进程情况下的各进程音频的分开截取，混合传输。

## 1.2项目背景

现有的音频截取技术大多数为截取系统总缓冲区中的音频进行传输，但是却无法很好地应对多session的问题。本项目通过分别截取每个进程的音频数据，再加以选择地进行混音，从而能将属于所请求session的声音取得并传输。

## 1.3定义

无

## 1.4参考资料

无

# 2. 软件概述

## 2.1软件介绍

Windows远程音频软件是用于取得机器属于某一session的进程的声音并进行混音、编码、传输、播放的软件。

## 2.2系统功能

Windows远程音频技术通过向所需的服务器端所有进程注入dll，来截取这些进程的音频，然后把这些音频数据根据session的不同分开存储在缓冲区，然后将每个客户端对应的session所需的音频数据进行混音，编码，传输，解码，播放。

## 2.3系统组成

Windows远程音频 主要由以下几部分组成：

1. AudioHook

进行音频截取，管理和混音的dll

1. InjectDll

注入dll的应用程序

1. AudioSender

进行音频编码和发送的进程

1. AudioReceiver

进行音频接收和解码的进程

1. LibAudioPlayer

进行音频播放的进程

## 2.4 总体结构

编码发送模块

接收解码模块

音频截获管理模块

（DLL）

音频播放模块

（LIB）

音频进程

音频进程

音频进程

网络传输

获取数据

调用

调用

载入数据

植入

写入数据

植入

植入

写入数据

写入数据

…

图1远程音频总体结构图

## 2.5 模块结构

### 2.5.1 音频截获管理模块 AudioHook

调用GetBuffer函数获取缓冲区指针

向缓冲区内写入数据

调用ReleaseBuffer函数获取缓冲区指针

......

截获缓冲区指针

截获缓冲区内数据

音频进程

DLL文件

图2音频截获流程图

本模块采用微软官方提供的Detours库来完成API Hook的工作，对系统的GetBuffer函数和Release函数进行修改，修改成如上图的操作，从而将进程的音频完成截获。

Struct SessionAudio[0] Struct SessionAudio[1]

Int SessionId

Struct Process Audio[1]

Int SessionId

Int NumOfProcess

Int NumOfProcess

**…...**

Struct Process Audio[0]

Struct Process Audio[1]

Struct Process Audio[0]

, ...... ……

图3音频缓冲区结构图

AudioHook将截获的PCM音频流数据存在一组结构体数组中，这个数组的名称为struct SessionAudio[]， 每个struct SessionAudio中存放了这个Session的sessionid,process的数量，以及一个struct ProcessAudio[]数组，而每个struct ProcessAudio中则存放了该进程的基本信息以及该进程的音频数据。

音频截获后，就找到该音频对应的session结构体以及其中的process结构体(若没有则新建)，然后开始向其中储存音频信息。

而提取音频时，通过找到所需session的struct，然后把其中所有process的音频数据都取出，并进行混音。

至于将这些数据给音频编码发送模块调用的方式，则是通过共享数据段的方式。将这个struct SessionAudio的数组以dll的共享数据段的格式储存，然后AudioSender通过API函数来取出这些PCM音频流数据，其中调度的方式是采用进程间的信号量来管理。

### 2.5.2 Dll注入模块 InjectDll

核心思想是通过函数ProcessIdToSessionId函数寻找所有sessionid符合的进程，然后利用Windows API中创建远程线程的函数，在目标进程中创建线程，同时参数设置为DLL文件的绝对路径。另外，该参数需要在目标进程中可以使用，因此需要在目标进程中分配一定的空间，将DLL文件绝对路径的字符串写入到分配的空间中，创建线程函数才能正确的将DLL文件植入目标进程。

### 2.5.3音频编码发送模块 AudioSender

getAudio

Put into buffer

Data in buffer > 8192B?

No

Yes

Get 8192B out

Wav\_read\_float

faacEncEncode

Put into send buffer

Send

图4 音频编码发送模块流程图

在每一个取出数据进行编码发送的周期中，AudioSender从AudioHook的缓冲区中取出混音完成后的PCM音频流数据，然后放入一个环形缓冲区中。编码采用的是64kbps的aac格式编码，编码模块集成了faac编码器，而faac编码器采用的是每次编码一个ADTS frame，即8192字节的机制，因此进入一个循环，只要环形缓冲区中的数据大于8192字节，就取出8192字节然后以32位float型读取，再调用faac的API函数进行编码，存入一个发送的缓冲区。当不再有数据能进行编码时，将缓冲区的数据通过TCP/IP协议进行发送。

### 2.5.4 音频接收解码模块 AudioReceiver

Receive

Is there data left?

yes

Get one ADTS frame

NeAACDecDecode

Load data into player

Wait for next reciving

图5 音频接收编码模块流程图

每一个接收数据的周期中，AudioReceiver先将数据接收到缓冲区中，然后进入循环：只要缓冲区中有数据，就从中取出一个ADTS frame，并调用集成的faad解码器的解码函数，然后解码成8192字节的PCM流数据并送入播放器处理。直到所有此次接收到的数据都被处理完。

### 2.5.5 音频播放模块 LibAudioPlayer

开始

结束

调用Start函数开始播放

调用GetBuffer函数获取缓冲区指针

向缓冲区写入数据

调用ReleaseBuffer函数释放缓冲区

是否还有数据

是

否

图6 音频播放模块流程图

播放器的核心是一个通过临界区的方式来管理的环形缓冲区。当第一次有PCM音频流数据进入缓冲区后，就开启一个向系统缓冲区传递数据的线程，只要缓冲区有数据，就将这些数据取出播放，从而完成音频的不断流畅播放。

# 3. 运行环境

## 3.1硬件环境

1. 主机CPU：P4 2.8Mhz（推荐Intel Core Duo 2.0GHz以上）
2. 内存：256MB（推荐512MB以上）
3. 磁盘空间：空闲空间100MB以上
4. 网络带宽：100M（推荐1000Mbps以上）

## 3.2支持软件

1. 操作系统

* Windows XP及以上

1. 软件

* 无

# 4. 使用说明

### 4.1 安装

无需安装，直接在当前目录使用

### 4.2 使用

1. 在服务器端运行InjectDll.exe将当前session的所有Process都注入dll
2. 运行AudioSender.exe准备发送数据
3. 在客户端运行AudioReceiver.exe开始接收数据，并且集成在其中的音频播放模块会将接收到的声音播放。

# 5. 可执行文件和数据文件一览表

**表1 可执行文件和数据文件表**

|  |  |
| --- | --- |
| 配置文件 | 具体说明 |
| ./InjectDll/InjectDll.exe | 进行dll注入的服务器端程序 |
| ./AudioSender/AudioSender.exe | 音频编码发送的服务器端程序 |
| ./AudioReceiver/AudioReceiver.exe | 音频接收播放的客户端程序 |
|  |  |
|  |  |
|  |  |

# 6. 常见问题及其解决方法

## 6.1注入dll失败

不存在符合条件的进程和AudioHook.dll路径错误将导致注入dll失败，应确认InjectDll的代码中进程的选取以及AudioHook.dll的绝对路径是否正确。

## Socket初始化失败

确认网络传输发送和接收方的匹配是否正确。

## 客户端无法播放声音

确认客户端有可用的音频播放设备。