

实 验 报 告

# 实验名称： SNMP管理站编程

**姓 名：** **王俊**

**学 号：** **202108060918**

**实验日期：** **2024年4月29日**

**指导老师：** **包博文**

实验5 SNMP管理站编程

一、实验目的

掌握应用SNMP++软件包开发SNMP管理站程序的基本方法。

二、实验内容

1. 编写一个简单的SNMP 管理站程序，能够发送SNMP请求，查询指定设备和对象信息。

2. 支持GETREQUEST, GETNEXTREQUEST, SETREQUEST请求,接收事件报告。

3. 输入的IP、团体名、对象OID等进行信息查询（类似snmputil 工具）

三、实验环境

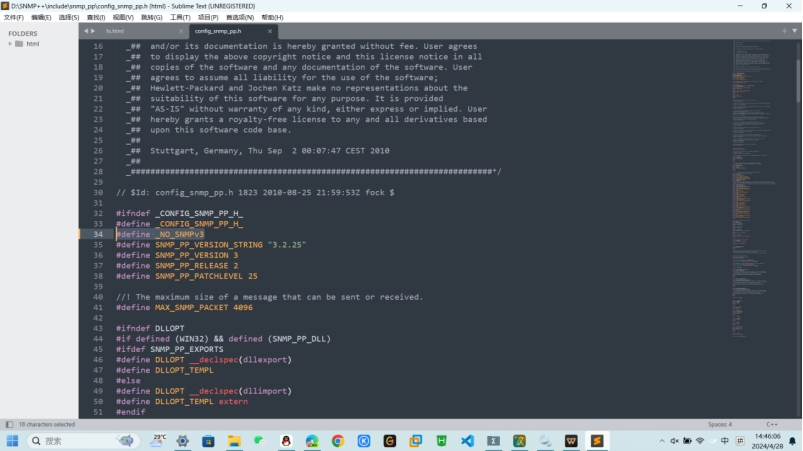
SNMP++ v3.2.25软件包、VC++6.0开发工具。

四、实验步骤

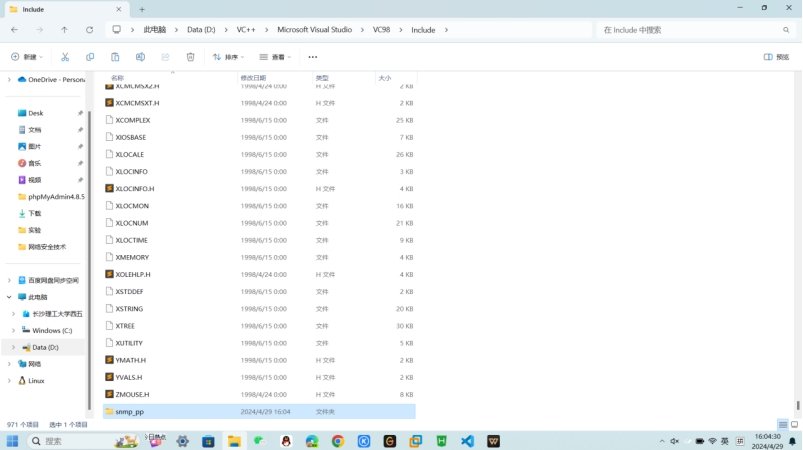
1、编译SNMP++软件包

1)下载SNMP++v3.2.25.tar.gz并解压到D盘根目录下（SNMP++），同时修改文件D:\SNMP++\include\snmp\_pp\config\_snmp\_pp.h，在其中添加下面代码用来去除SNMPv3（如果不添加该行，那么需要添加des库来实现SNMPv3的安全功能,还需要下载libdes-l-4.01a）：

#define \_NO\_SNMPv3 //去掉SNMPv3。

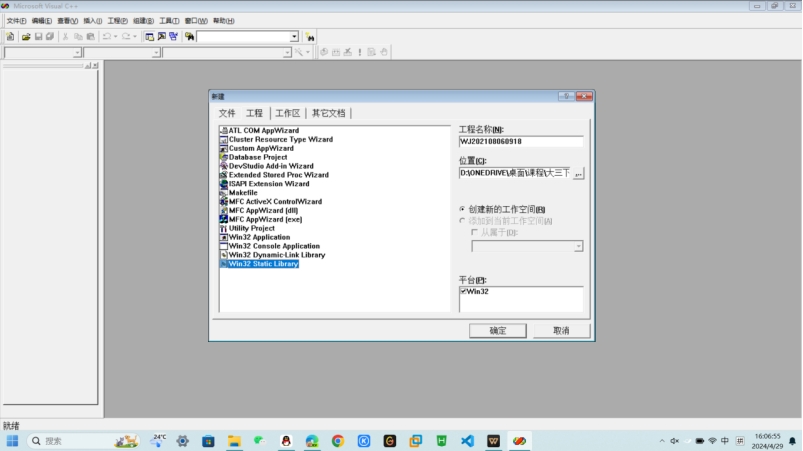


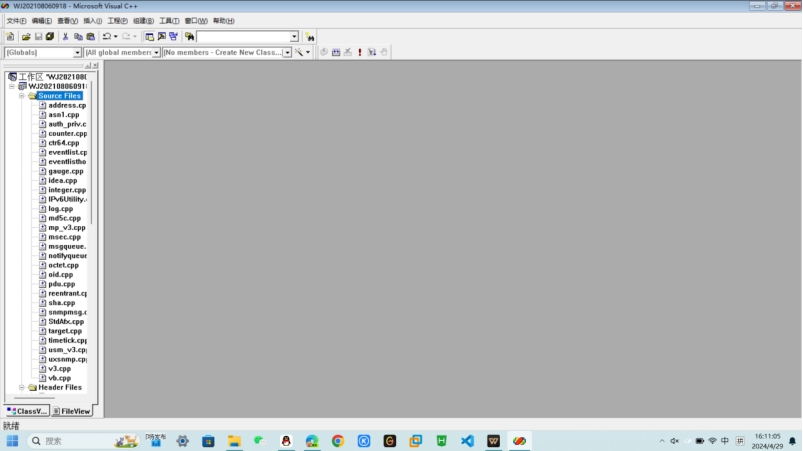
1. 将include下的目录复制到VC6的include目录下（D:\VC++\Microsoft Visual Studio\VC98\Include）

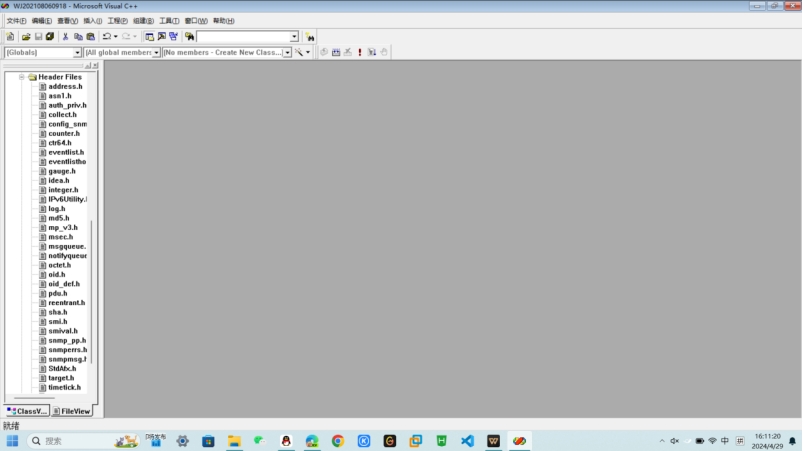


3）在vc6中新建一个win32项目，选择静态库，空项目

将所有需要的头文件和源文件均加入进来，截图如下：

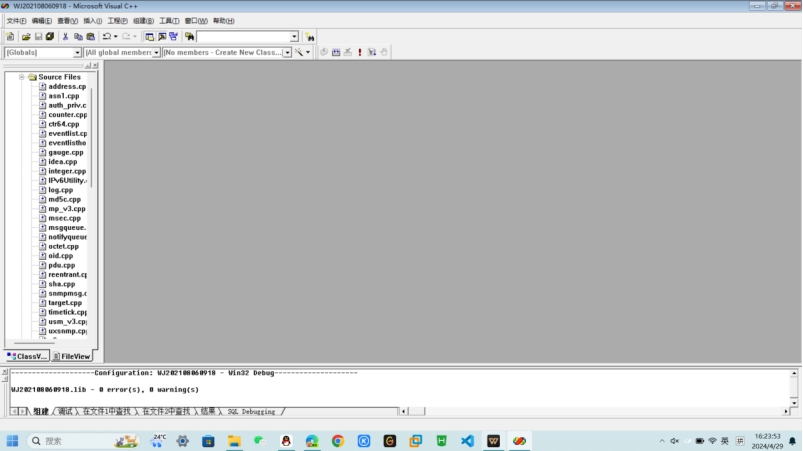


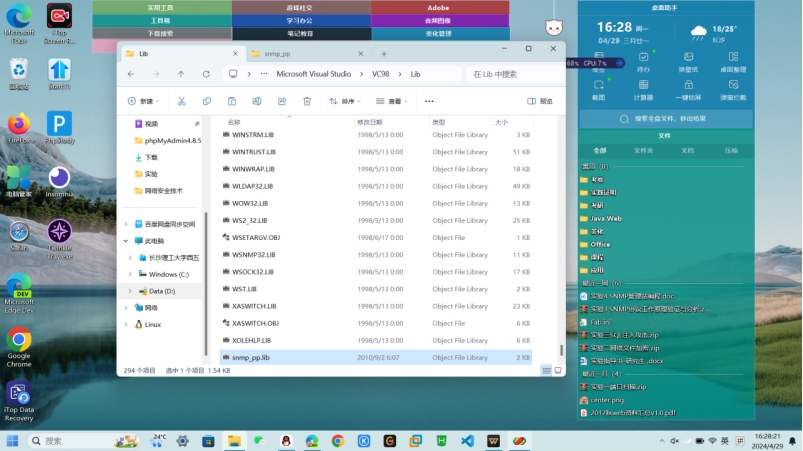




注意：为工程添加的源文件是指D:\SNMP++\src下的所有\*.cpp文件,头文件是指D:\VC++\Microsoft Visual Studio\VC98\Include\snmp\_pp下的所有\*.h文件。

1. 编译、链接，然后将WJ202108060918\Debug目录下WJ202108060918.lib重命名为snmp\_pp.lib复制到vc6的lib目录。

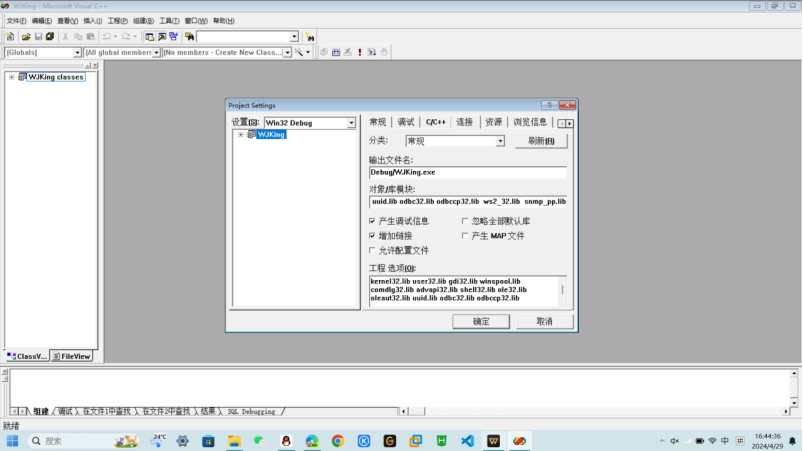




2、编写一个简单的SNMP管理站程序。

1. 首先建立VC++6.0控制台工程

在Project →Settings→ Link →Object /library modules中添加ws2\_32.lib 、snmp\_pp.lib，如下图：（也可使用预处理命令添加#pragma comment(lib," snmp\_pp.lib ")）



1. 参考代码：

#include "iostream"

#include"snmp\_pp/snmp\_pp.h"

using namespace std;

void main()

{

Snmp::socket\_startup(); //初始化Winsocket套接字

int status; //状态信息

Vb vb; //创建Vb对象

Oid oid; //创建Oid对象

oid = ".1.3.6.1.2.1.1.2.0";

vb.set\_oid( oid ); //绑定对象标识符（）

Pdu pdu; // 创建Pdu对象

pdu += vb; //将Vb对象添加到Pdu对象中

UdpAddress udp( "127.0.0.1:161");//初始化IpAddress，同组同学的ip地址

CTarget ctarget( udp );//实例化代理主机

snmp\_version version = version1;

ctarget.set\_version(version); //设置Snmp协议版本

ctarget.set\_readcommunity("public"); //设置只读共同体名

SnmpTarget \*target;

target = &ctarget;

//------------------建立snmp会话---------------------------

Snmp snmp( status, 0, false ); // 创建snmp对象

if ( status != SNMP\_CLASS\_SUCCESS) { //如果创建失败，输出错误信息

cout << snmp.error\_msg( status); //并结束进程

return; }

//-----------------snmp的get操作--------------------------------------------

if ( (status = snmp.get( pdu, ctarget)) != SNMP\_CLASS\_SUCCESS) //如果操作失败

{ cout << snmp.error\_msg( status); }//输出错误信息

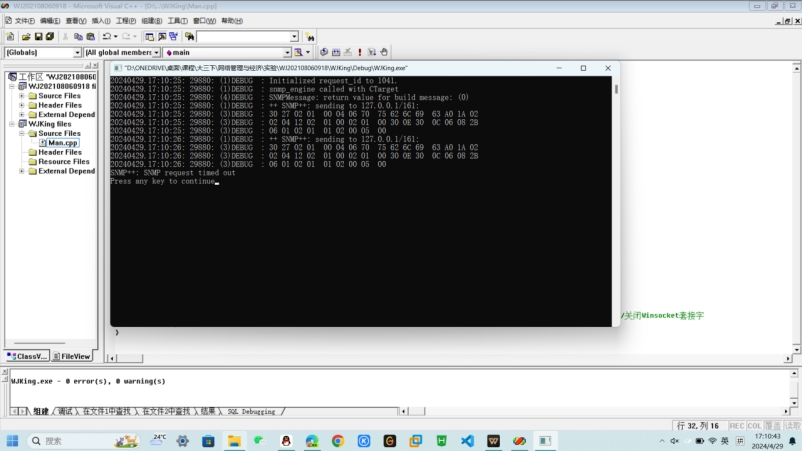
else {

pdu.get\_vb( vb, 0); // 取出应答pdu中的vb对象

cout << "System **sysObjectID** = "<< vb.get\_printable\_value(); } //输出vb中的值 Snmp::socket\_cleanup(); //关闭Winsocket套接字

cout<<"\n";

}

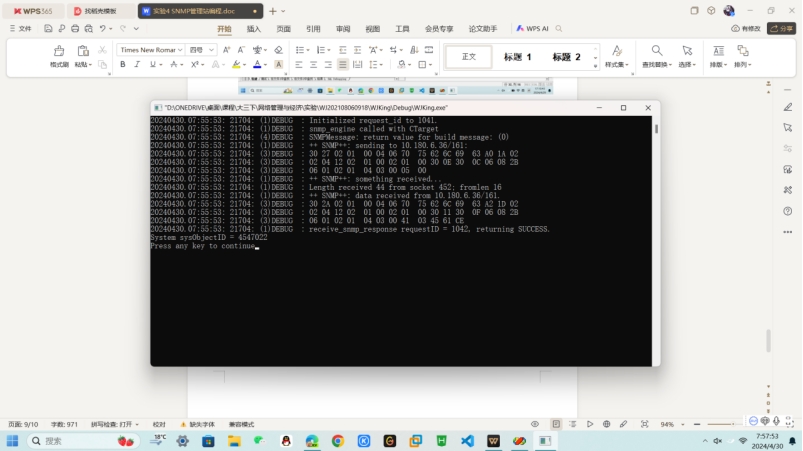


如果未能得到类似的结果，应检查被管设备是否运行SNMP协议和相关的服务、端口是否正确、共同体名是否相符等情况，将原因详细记录。

1.修改程序，查询设备的ipInReceives对象的信息。

修改OID为.1.3.6.1.2.1.4.3.0

完整代码见后。



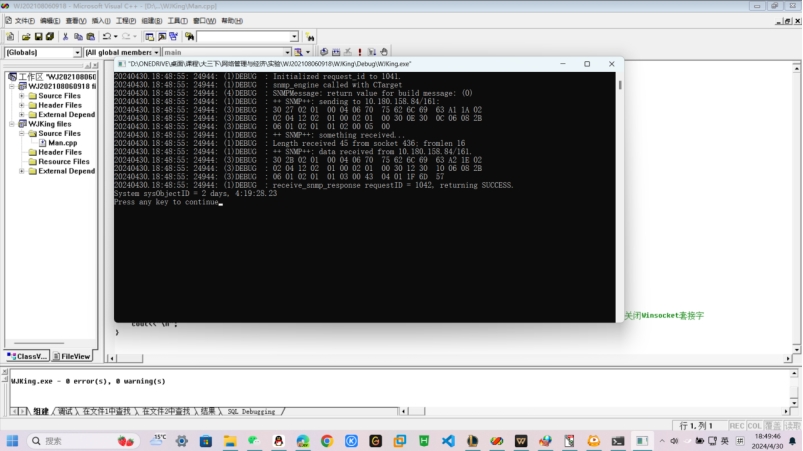
2.修改程序，GETNEXTREQUEST, SETREQUEST请求，接收事件报告。（参见参考资料“Visual C++开发基于SNMP的网络管理软件”）

GETNEXTREQUEST操作

修改PDU类型为GETNEXT请求

将原来的snmp.get(pdu, ctarget)修改为snmp.get\_next(pdu, ctarget)

完整代码见后。



SETREQUEST操作

修改PDU类型为SETREQUEST请求

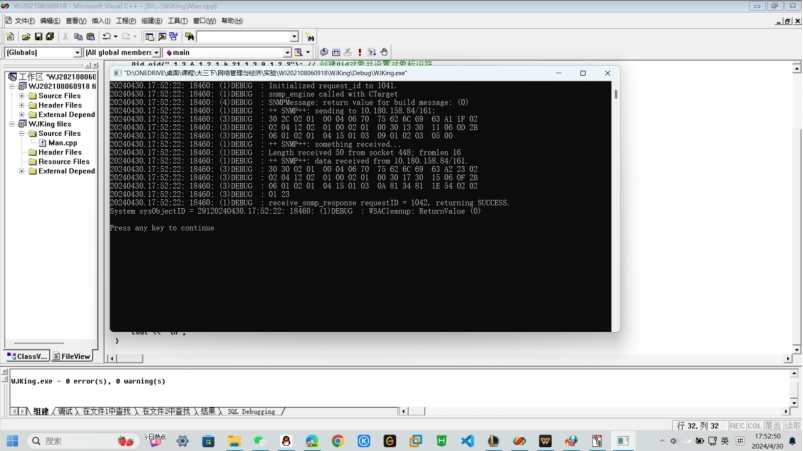
只读共同体改为读写共同体ctarget.set\_writecommunity("public");

将原来的snmp.get(pdu, ctarget)修改为snmp.set(pdu, ctarget)

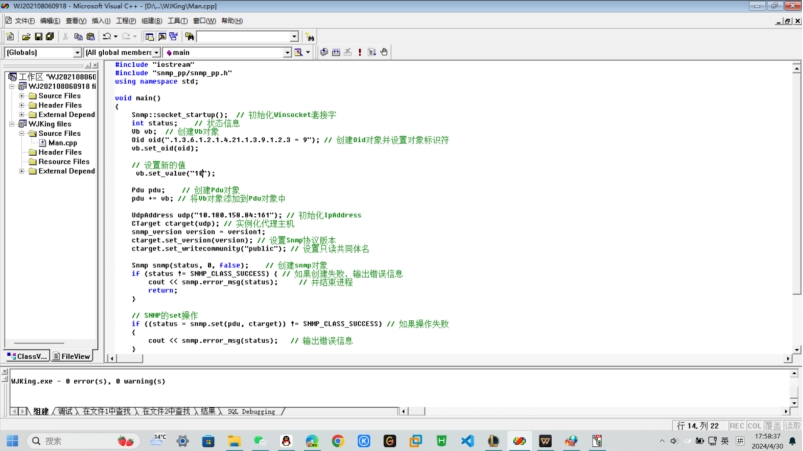
然后需要设置新的值vb.set\_value("10");

完整代码见后。

如图所示，一开始该值为291



使用SETREQUEST命令后，其值变为10.



3.完善程序，使得能根据输入的IP、团体名、对象OID等进行GETREQUEST,GETNEXTREQUEST和SETREQUEST查询，

例如：

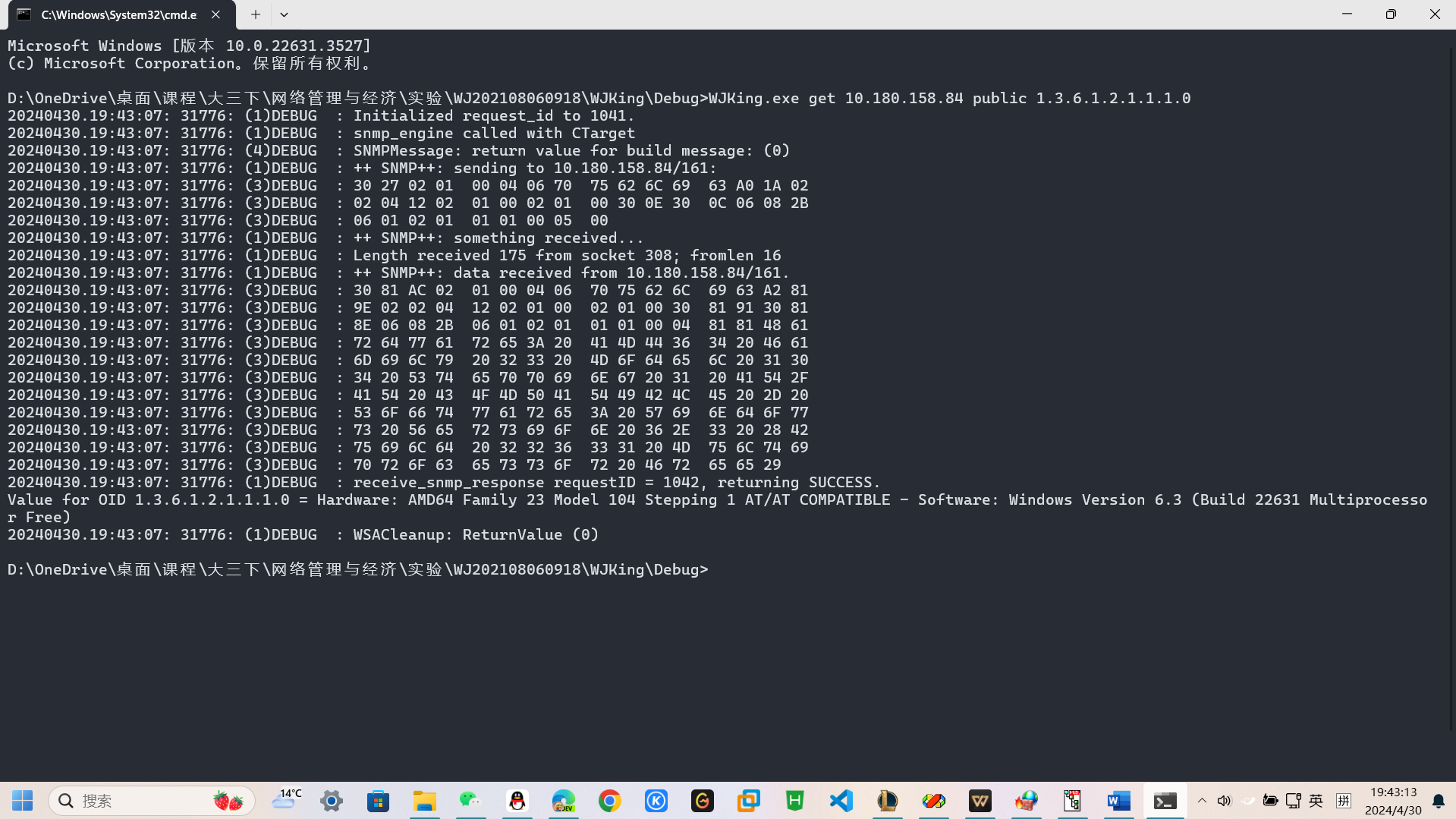
WJKing.exe get 10.180.158.84 public 1.3.6.1.2.1.1.1.0

WJKing.exe getnext 10.180.158.84 public 1.3.6.1.2.1.1.1

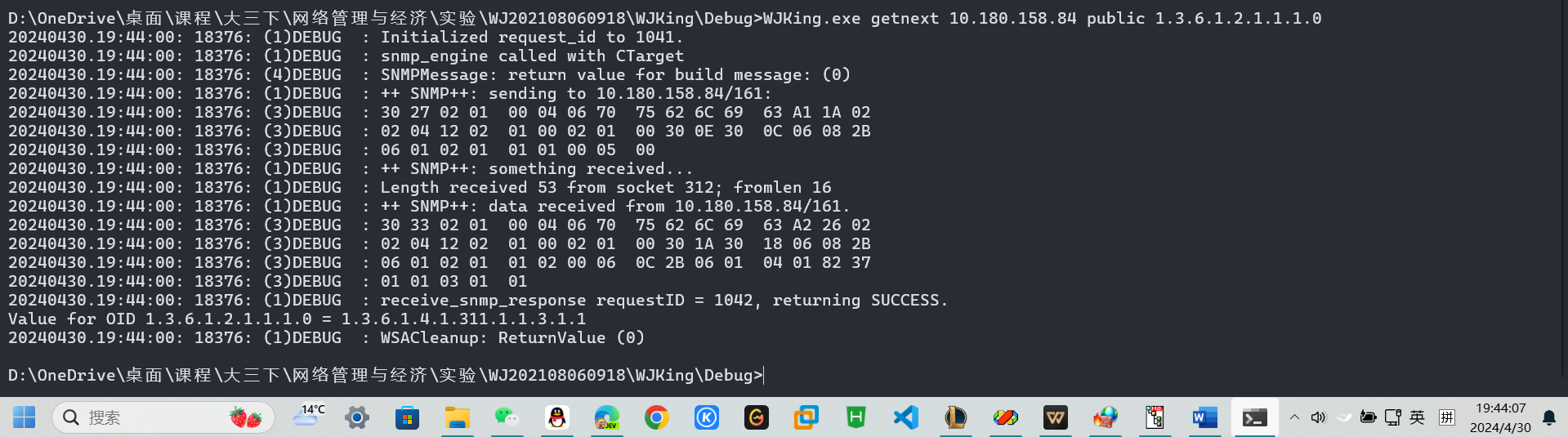
WJKing.exe set 10.180.158.84 public 1.3.6.1.2.1.1.1 “20”

通过在命令行使用可执行文件实现类似snmputil查询方式

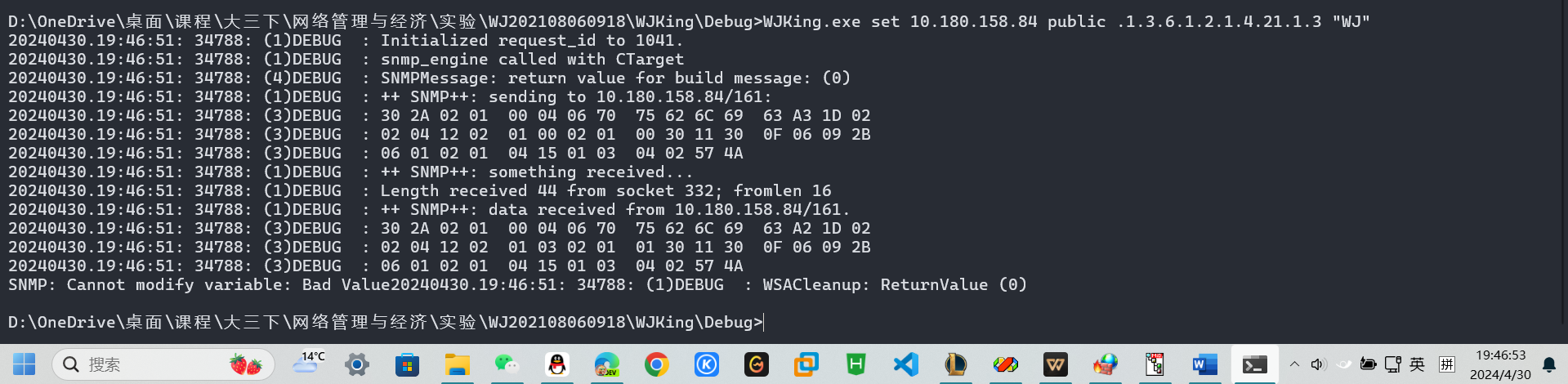
Get:



GetNext:



Set:



五、实验结果与分析

实验非常成功，可以完美实现命令，在SETREQUEST操作中，出现了bad value、cannot modify ，varity not exists以及执行成功多种结果。

1.GETREQUEST代码：

#include "iostream"

#include"snmp\_pp/snmp\_pp.h"

using namespace std;

void main()

{

Snmp::socket\_startup(); //初始化Winsocket套接字

int status; //状态信息

Vb vb; //创建Vb对象

Oid oid; //创建Oid对象

oid = ".1.3.6.1.2.1.1.2.0";

vb.set\_oid( oid ); //绑定对象标识符（）

Pdu pdu; // 创建Pdu对象

pdu += vb; //将Vb对象添加到Pdu对象中

UdpAddress udp( "127.0.0.1:161");//初始化IpAddress，同组同学的ip地址

CTarget ctarget( udp );//实例化代理主机

snmp\_version version = version1;

ctarget.set\_version(version); //设置Snmp协议版本

ctarget.set\_readcommunity("public"); //设置只读共同体名

SnmpTarget \*target;

target = &ctarget;

//------------------建立snmp会话---------------------------

Snmp snmp( status, 0, false ); // 创建snmp对象

if ( status != SNMP\_CLASS\_SUCCESS) { //如果创建失败，输出错误信息

cout << snmp.error\_msg( status); //并结束进程

return; }

//-----------------snmp的get操作--------------------------------------------

if ( (status = snmp.get( pdu, ctarget)) != SNMP\_CLASS\_SUCCESS) //如果操作失败

{ cout << snmp.error\_msg( status); }//输出错误信息

else {

pdu.get\_vb( vb, 0); // 取出应答pdu中的vb对象

cout << "System **sysObjectID** = "<< vb.get\_printable\_value(); } //输出vb中的值 Snmp::socket\_cleanup(); //关闭Winsocket套接字

cout<<"\n";

}

2.GETNEXTREQUEST代码：

#include "iostream"

#include"snmp\_pp/snmp\_pp.h"

using namespace std;

void main()

{

Snmp::socket\_startup(); //初始化Winsocket套接字

int status; //状态信息

Vb vb; //创建Vb对象

Oid oid; //创建Oid对象

oid = ".1.3.6.1.2.1.1.2.0";

vb.set\_oid( oid ); //绑定对象标识符（）

Pdu pdu; // 创建Pdu对象

pdu += vb; //将Vb对象添加到Pdu对象中

UdpAddress udp( "10.180.158.84:161");//初始化IpAddress，同组同学的ip地址

CTarget ctarget( udp );//实例化代理主机

snmp\_version version = version1;

ctarget.set\_version(version); //设置Snmp协议版本

ctarget.set\_readcommunity("public"); //设置只读共同体名

SnmpTarget \*target;

target = &ctarget;

//------------------建立snmp会话---------------------------

Snmp snmp( status, 0, false ); // 创建snmp对象

if ( status != SNMP\_CLASS\_SUCCESS) { //如果创建失败，输出错误信息

cout << snmp.error\_msg( status); //并结束进程

return; }

//-----------------snmp的get操作--------------------------------------------

if ( (status = snmp.get\_next( pdu, ctarget)) != SNMP\_CLASS\_SUCCESS) //如果操作失败

{ cout << snmp.error\_msg( status); }//输出错误信息

else {

pdu.get\_vb( vb, 0); // 取出应答pdu中的vb对象

cout << "System sysObjectID = "<< vb.get\_printable\_value(); } //输出vb中的值 Snmp::socket\_cleanup(); //关闭Winsocket套接字

cout<<"\n";

}

3.SETREQUEST代码：

#include "iostream"

#include "snmp\_pp/snmp\_pp.h"

using namespace std;

void main()

{

Snmp::socket\_startup(); // 初始化Winsocket套接字

int status; // 状态信息

Vb vb; // 创建Vb对象

Oid oid(".1.3.6.1.2.1.4.21.1.3.9.1.2.3 "); // 创建Oid对象并设置对象标识符

vb.set\_oid(oid);

// 设置新的值

vb.set\_value("10");

Pdu pdu; // 创建Pdu对象

pdu += vb; // 将Vb对象添加到Pdu对象中

UdpAddress udp("10.180.158.84:161"); // 初始化IpAddress

CTarget ctarget(udp); // 实例化代理主机

snmp\_version version = version1;

ctarget.set\_version(version); // 设置Snmp协议版本

ctarget.set\_writecommunity("public"); // 设置只读共同体名

Snmp snmp(status, 0, false); // 创建snmp对象

if (status != SNMP\_CLASS\_SUCCESS) { // 如果创建失败，输出错误信息

cout << snmp.error\_msg(status); // 并结束进程

return;

}

// SNMP的set操作

if ((status = snmp.set(pdu, ctarget)) != SNMP\_CLASS\_SUCCESS) // 如果操作失败

{

cout << snmp.error\_msg(status); // 输出错误信息

}

else {

pdu.get\_vb(vb, 0); // 取出应答pdu中的vb对象

cout << "System sysObjectID = " << vb.get\_printable\_value(); // 输出vb中的值

}

Snmp::socket\_cleanup(); // 关闭Winsocket套接字

cout << "\n";

}

4.实现类似snmputil命令行方式代码：

#include "iostream"

#include "snmp\_pp/snmp\_pp.h"

#include <string>

#include <cstdlib> // 用于 atoi 等函数

using namespace std;

void performGet(const string& ip, const string& community, const string& oidStr) {

// 与原queryInfo函数相似，但专注于GET操作

Snmp::socket\_startup();

int status;

Oid oid(oidStr.c\_str());

Vb vb(oid);

Pdu pdu;

pdu += vb;

string udpAddress = ip;

UdpAddress udp(udpAddress.c\_str());

CTarget ctarget(udp);

snmp\_version version = version1;

ctarget.set\_version(version);

ctarget.set\_readcommunity(community.c\_str());

Snmp snmp(status, 0, false);

if (status != SNMP\_CLASS\_SUCCESS) {

cout << snmp.error\_msg(status);

return;

}

if ((status = snmp.get(pdu, ctarget)) != SNMP\_CLASS\_SUCCESS) {

cout << snmp.error\_msg(status);

} else {

pdu.get\_vb(vb, 0);

cout << "Value for OID " << oidStr << " = " << vb.get\_printable\_value() << endl;

}

Snmp::socket\_cleanup();

}

void performGetNext(const string& ip, const string& community, const string& oidStr) {

Snmp::socket\_startup();

int status;

Oid oid(oidStr.c\_str());

Vb vb(oid);

Pdu pdu;

pdu += vb;

string udpAddress = ip;

UdpAddress udp(udpAddress.c\_str());

CTarget ctarget(udp);

snmp\_version version = version1;

ctarget.set\_version(version);

ctarget.set\_readcommunity(community.c\_str());

Snmp snmp(status, 0, false);

if (status != SNMP\_CLASS\_SUCCESS) {

cout << snmp.error\_msg(status);

return;

}

if ((status = snmp.get\_next(pdu, ctarget)) != SNMP\_CLASS\_SUCCESS) {

cout << snmp.error\_msg(status);

} else {

pdu.get\_vb(vb, 0);

cout << "Value for OID " << oidStr << " = " << vb.get\_printable\_value() << endl;

}

Snmp::socket\_cleanup();

}

void performSet(const string& ip, const string& community, const string& oidStr, const string& setValue) {

Snmp::socket\_startup();

int status;

Oid oid(oidStr.c\_str());

// 创建Vb实例

Vb vb;

// 设置OID

vb.set\_oid(oid);

// 将字符串值转换为OctetStr对象并设置Vb的值

OctetStr osSetValue(setValue.c\_str());

vb.set\_value(osSetValue);

Pdu pdu;

pdu += vb; // 将Vb对象添加到Pdu对象中

string udpAddress = ip;

UdpAddress udp(udpAddress.c\_str());

CTarget ctarget(udp);

snmp\_version version = version1;

ctarget.set\_version(version);

ctarget.set\_writecommunity(community.c\_str());

Snmp snmp(status, 0, false);

if (status != SNMP\_CLASS\_SUCCESS) {

cout << snmp.error\_msg(status);

return;

}

if ((status = snmp.set(pdu, ctarget)) != SNMP\_CLASS\_SUCCESS) {

cout << snmp.error\_msg(status);

} else {

cout << "SET operation on OID " << oidStr << " successful." << endl;

}

Snmp::socket\_cleanup();

}

int main(int argc, char\* argv[]) {

if(argc < 5 || argc > 6) {

cout << "Usage: " << argv[0] << " [get|getnext|set] <IP> <Community> <OID> [ValueToSet]\n";

return 1;

}

string operation = argv[1];

string ip = argv[2];

string community = argv[3];

string oidStr = argv[4];

string setValue; // 仅SET操作需要

if(argc == 6) {

setValue = argv[5]; // 获取设置的值

}

if(operation == "get") {

performGet(ip, community, oidStr);

} else if(operation == "getnext") {

performGetNext(ip, community, oidStr);

} else if(operation == "set") {

if(setValue.empty()) {

cout << "SET operation requires a value to set.\n";

return 1;

}

performSet(ip, community, oidStr, setValue);

} else {

cout << "Unsupported operation: " << operation << ". Use 'get', 'getnext', or 'set'.\n";

return 1;

}

return 0;

}

六、实验过程遇到的问题和解决办法

在实验中遇到了很多困难：

1. 无法找到snmp\_pp.lib文件。

**出错原因：**编译生成的lib文件是项目名称。

解决方法：将项目生成的WJ202108060918.lib文件重命名为snmp\_pp.lib。

1. 出现SNMP++: SNMP request timed out 。

出错原因：IP设置错误，UDP161端口未打开。

解决方法：仔细检查IP设置，打开UDP161端口。

1. 在执行SETREQUEST操作时出现SNMP: Cannot modify variable: Bad Value。

出错原因：团体名为只读共同体，set的变量值不符合规范。

解决方法：将团体名设置为读写共同体，仔细检查set的变量值。