信息安全原理课程报告

作业二

王子腾 3180102173

【实验内容】

Try to simulate a computer attack: implement a malicious code, including but not limited to buffer overflow, dictionary attack (the dictionary file could be from Internet or written by yourself) and viruses. Explain how it works and how to defend such kind of attack.

【实验环境】

C++

【攻击方式】

命名为 Virus.cpp 的恶意代码通过诱导用户点击对话框互动，将当前目录下所有后缀名为（.c, .cpp）类型的代码文件全部乱码处理，以达到破坏程序员编写的代码的效果。

【算法设计】

本程序通过交互式界面，在用户不经意间触发破坏程序。程序首先抛出一个对话框，在用户点击发生互动后，触发 Virus 函数，执行破坏任务，并同时无限次弹出对话框，干扰用户视野。

当用户通过任务管理器关闭本程序时，会发现所有代码文件均已被破坏，并且本程序源代码自毁，若无解药代码，用户极难恢复丢失信息。

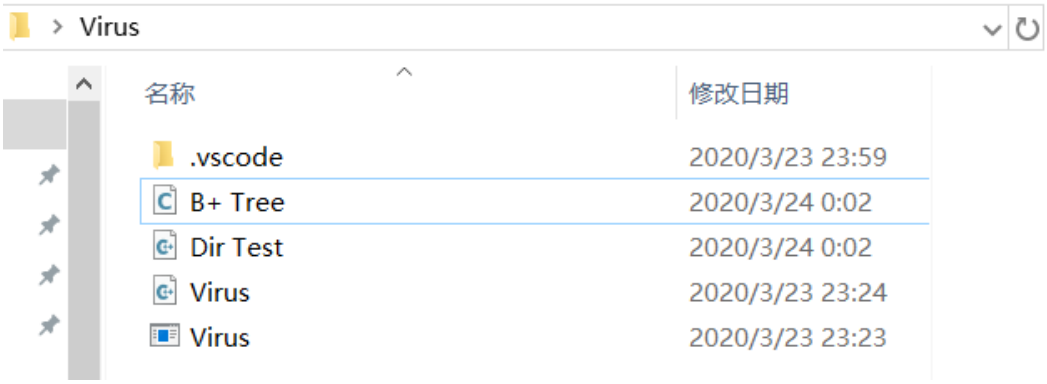
破坏原理：为方便操作文件，同时更大程度造成破坏，本实验通过二进制流读入文件，并通过将每一位读入数据进行加一的操作，造成破坏。在破坏开始前，先创建一个临时文件，用于保存修改后的数据，任务完成后，删除原文件，同时为了迷惑，将临时文件按照原文件命名。

```
InFile.read((char*)&ch, sizeof(char));
while(!InFile.eof())
{
    ch += 1;
    OutFile.write((char*)&ch, sizeof(char));
    InFile.read((char*)&ch, sizeof(char));
}
InFile.close();
```

恢复原理：与破坏原理相近，不过核心操作相反，将数据进行减一操作。

【实验结果】

在当前文件下放入两个用于测试的例程，分别为.c, .cpp 文件



其中“B+ Tree.c”和“Dir Test.cpp”文件部分内容如下：

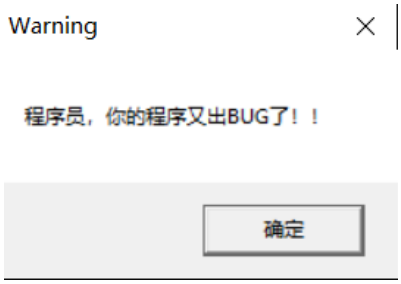
B+ Tree.c > ...

```c
//已裁剪，用于测试病毒文件

#include<stdio.h>
#include<stdlib.h>

typedef struct Node* PtrtoNode;
struct Node{
    int children;
    int capacity;
    int key[4];
    int leaf;
    int layer;
    PtrtoNode parent;
    PtrtoNode son[4];
};

int cmp(const void*a,const void*b)
{
    return *(int*)a-*(int*)b;
}
```

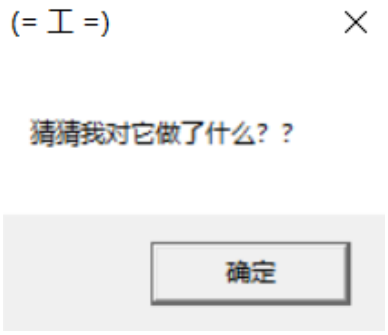
Dir Test.cpp > ...

```cpp
//简单程序，用于测试病毒文件

#include<iostream>
#include<string>
using namespace std;

int main()
{
    string dir = __FILE__;
    cout << dir <<endl;
    return 0;
}
```

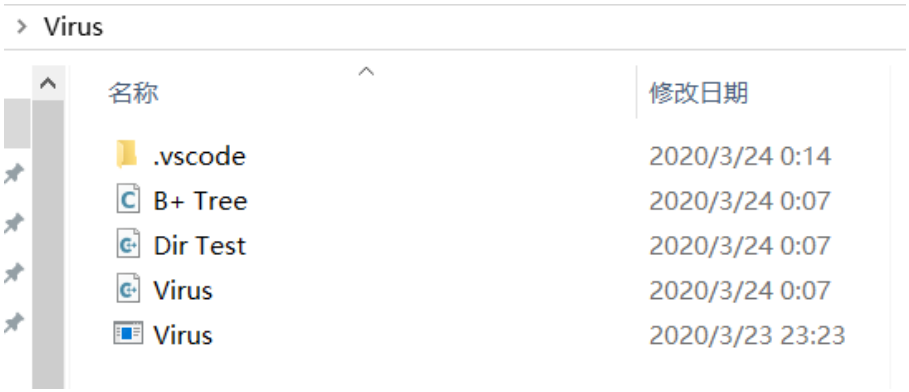
程序运行后，弹出对话框

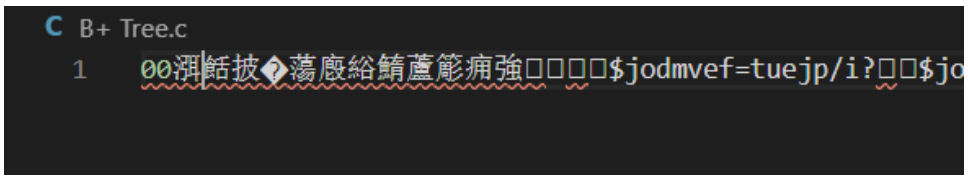


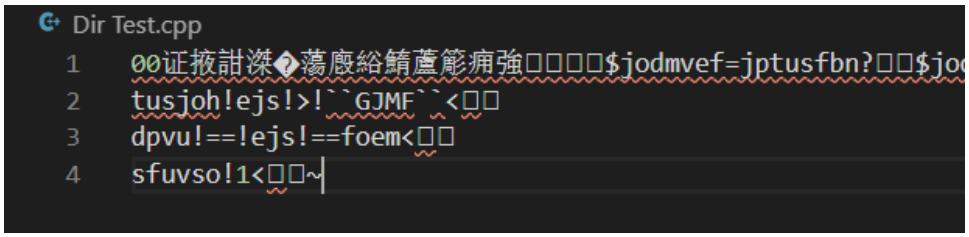
点击确定，之后开始无限弹出另一个对话框

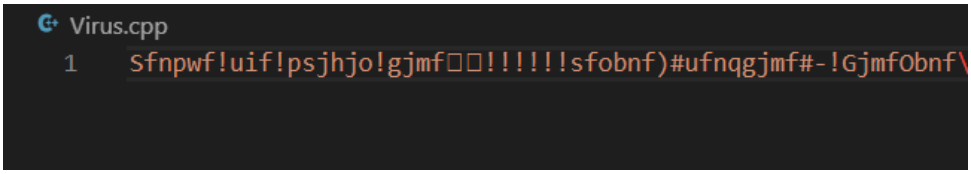


通过任务管理器或关闭控制台，强行关闭程序，再查看时发现，虽然文件依然还在，但是内容都已被破坏，发生乱码



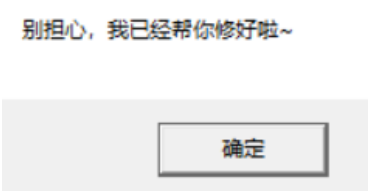






此时再运行“Medicine.exe”解药文件，弹出对话框，发现所有文件均恢复正常。

O(∩_∩)O



B+ Tree.c > Node > leaf

```c
//已裁剪，用于测试病毒文件

#include<stdio.h>
#include<stdlib.h>

typedef struct Node* PtrtoNode;
struct Node{
    int children;
    int capacity;
    int key[4];
    int leaf;
    int layer;
    PtrtoNode parent;
```

Dir Test.cpp > ...

```cpp
//简单程序，用于测试病毒文件

#include<iostream>
#include<string>
using namespace std;

int main()
{
```

Virus.cpp > ...

```cpp
/*
    Program: Virus.cpp
    Programmer: Ziteng Wang
    Date: 2020/3/23

    Used as HW for <Principles of Information Securit
*/
#include<io.h>
#include<cstdio>
#include<windows.h>
#include<string>
#include<string.h>
#include<fstream>
#include<iostream>
#include<vector>

using namespace std;

// Function delaration
void Virus();
```

【防御策略】

1. 最重要的就是：千万不要点击陌生的可执行文件，不要编译执行陌生的代码（操作前先检查代码逻辑），恶意代码就像一个有万能钥匙的小偷，不给小偷指路远远比换一把高级的锁来得实在。
2. 即使你不小心执行了，正如本题的逻辑所示，只要你不按下第一个对话框的按键，恶意代码就不会执行，立马通过任务管理器强制关闭程序，也许为时未晚。
3. 如果你不小心点了按钮，文件被破坏，还有最后一条路可以走，如本题所示，有些恶意代码会给你配备一瓶解药，只要能够得到这样的代码或可执行文件，丢失的信息就还能正常恢复。
4. 另外，这样的事情也告诉我们，日常生活中，对于重要的资料和文件，应当养成备份的好习惯，即是全部丢失，备份的部分也能正常恢复，弥补损失。

【实验心得】

本次实验对我来说收获丰厚，不仅磨合了我的编程技术，熟悉了许多平时不常用的库，比如：“windows.h”中对话框的编写，以及二进制操作文件，根据路径搜索文件等编程功能，更加深了我对恶意代码和攻击程序构建的理解，同时对信息安全技术产生了浓厚的兴趣。