

# 信息安全科技创新 项目结题报告

项目名称：基于特征串匹配的攻击检测系统

小组成员：

赵天航 521021910855

史轩宇 521021910853

陈梓轩 521021910976

郑祥 521021910968

吴铮 521030910252

学院：电子信息与电气工程学院

报告完成日期：7月19日

# 项目摘要

基于现存许多利用网络流量包输入的入侵技术，用于维护网络安全的入侵检测技术应运而生。但传统的入侵检测技术存在误报率高、适应性差和检测率低的问题。该项目组希望基于误用检测技术构建一个基于特征串匹配的攻击检测系统，目标是实现自动的网络数据包抓取与基于自主建立的特征串数据库的特征串匹配，提高入侵检测系统的安全检测能力，降低误报率和漏报率。在此基础上，提供不同的安全等级供用户挑选，并且使系统具备一定的反逃避能力，改进特征串匹配算法提升效率。

本项目工作主要分为以下几个部分：项目目标任务确立、项目初始代码实现、项目功能拓展、图形化实现、结题报告撰写。第一周我们确立了项目实现的目标和分工情况；第二周完成代码初步撰写并可以成功编译，进行简单的特征串匹配检测；第三周至第四周周三我们完善了项目的各个功能，对代码进行优化，实现了图形化，确定代码最终形式；最后时间我们用于撰写项目的结题报告。

项目成功完成了基于特征串匹配的攻击检测系统，满足了以下基本功能：

1. 特征库建立。
2. 网络数据包获取。
3. 特征串匹配。

完成以上几项基本功能后，我组成员又进行了如下额外功能拓展与改进：

1. 考虑相邻报文的合并检测，提升了攻击检测系统的反逃避能力。
2. 改进了特征串匹配算法，针对100%匹配率要求的情形，采用AC自动机算法，极大地提高了匹配效率；对于其他匹配率要求的情形，采用DP算法求得待匹配串与数据包的编辑距离，而后计算相似度，以判断是否达到匹配率要求的阈值。
3. 针对不同用户的需求，给用户可选择权利，用户可以选择不同的level来选择系统报出攻击的阈值
4. 利用QT对系统进行了可视化，图形界面简洁美观，提升了用户的可操作性以及系统的欣赏性。
5. 考虑系统的鲁棒性，检测功能健壮，即使攻击者在数据包里增加了冗余信息，系统依旧可以匹配到有特征串的部分，降低了漏报率。

我们分别对不同相似度的网页攻击检测能力、抗逃避检测能力、多个TCP连接的稳定性以及一个TCP存在大量数据的稳定性进行测试。

通过反复修改网页中输入的字符串，发现系统可以按照预定的相似度阈值设定检测到攻击。

通过把攻击特征串的前半段和后半段分在两个数据包里进行实验，系统依旧检测到了攻击，说明系统具有一定的抗逃避检测的能力。

通过不断建立TCP连接反复实验，发现测试至多可以建立737个TCP连接，已经足以说明系统面对多个TCP连接时的稳定性。不能建立更多TCP连接的原因可能与虚拟机的本身配置有关。

通过在同一TCP连接里传输十万量级个数的数据包检测，发现系统依旧保持稳定，说明系统面对TCP连接中的大量数据依旧保持稳定。

总之，系统在完成了基本功能的基础上，也出色地完成了拓展功能，能够适应绝大多数的入侵检测情况。具体的测试结果详见第五章第3节。

# 目录

## 第一章 需求分析

- 1.1 项目需求分析
- 1.2 项目功能目标
- 1.3 项目用户需求

## 第二章 总体设计

- 2.1 总体设计图
- 2.2 main
- 2.3 widget
- 2.4 set\_parameters
- 2.5 choose\_patternfile
- 2.6 wait
- 2.7 show\_detection

## 第三章 详细设计

- 3.1 模块1——网络模块
  - 3.1.1 初始化
  - 3.1.2 数据报文获取与分析
  - 3.1.3 报文合并
- 3.2 模块2——特征串文件处理模块
  - 3.2.1 特征串文件处理模块概述
  - 3.2.2 重要数据结构的构建
  - 3.2.3 tcp\_callback回调函数
  - 3.2.4 matchpattern的检测
  - 3.2.5 output\_alert函数
- 3.3 模块3——图形化模块
  - 3.3.1 Qt的安装与配置
  - 3.3.2 从命令行到图形化
  - 3.3.3 图形化的发布

## 第四章 系统实现

- 4.1 实现环境及开发工具；
- 4.2 源文件分析及调度关系图
- 4.3 目标程序组成和运行方式

## 第五章 系统测试

- 5.1 测试方法
- 5.2 测试流程图
- 5.3 测试效果、现象、结论

## 第六章 项目小节

- 6.1 项目实施情况
- 6.2 不足、展望
- 6.3 体会、感受
- 6.4 建议、意见

# 第一章 需求分析

---

## 1.1 项目需求分析

恶意软件、网络钓鱼、SQL注入、跨站脚本攻击等都是网络安全领域中一些常见的攻击方式，均可以通过对网络数据包抓取并分析检测到这些攻击。基于特征串匹配的攻击检测系统正是针对这些攻击方式而诞生的。在设计基于特征串匹配的攻击检测系统时，需要考虑到实际网络环境中可能存在的各种攻击手段和技术，以及相应的攻击特征和模式。此外，还需要考虑到系统的实时性、准确性和可扩展性等方面的因素，以便在实际应用中能够有效地保护网络安全。

## 1.2 项目功能目标

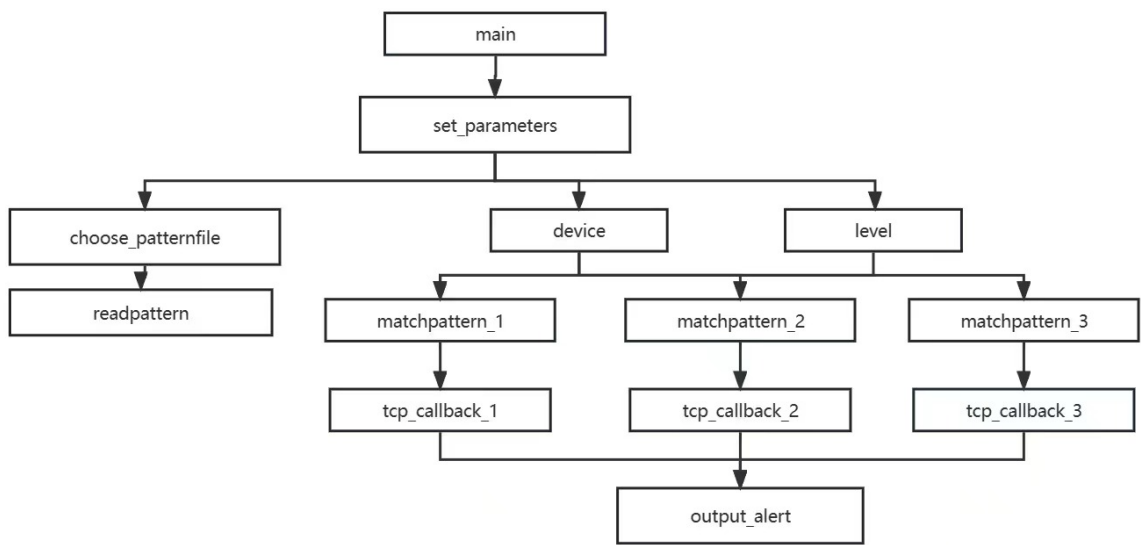
- 自动获取攻击机向目标机传输的报文；
- 可供用户自主选择设备和安全等级；
- 通过特征串文件生成可供报文检测匹配的特征串链；
- 特征串链与报文匹配，提高匹配算法的效率；
- 特征串匹配存在一定的反逃避能力；
- 实现攻击检测结果的输出；
- 实现系统的图形化展示。

## 1.3 项目用户需求

- 自主选择准确率和误报率平衡级别的攻击检测；
- 高效的攻击检测；
- 实时的攻击检测；
- 便于使用且容易理解的图形化界面；
- 全面系统的特征串数据库以保证全面的检测。

## 第二章 总体设计

### 2.1 总体设计图



### 2.2 main

1. 调用窗口显示的函数，实现图形化界面。

### 2.3 widget

1. 实现了一个主窗口的界面，并提供了相应的构造和析构函数以及槽函数来处理用户的操作和界面的转换。

### 2.4 set\_parameters

1. 显示可用设备列表：当用户点击 `device_detection` 按钮时，通过调用 `pcap_findalldevs` 函数找到所有的设备，并将它们显示在界面上的 `device_output` 控件中。
2. 确定参数设置：当用户点击 `parameters_confirm` 按钮时，根据用户选择的设备和安全级别，设置相应的参数。
3. 切换安全级别：用户可以在界面上的 `choose_level` 控件中选择不同的安全级别，通过设置全局变量 `level` 来记录用户选择的安全级别。`level1`，需要用AC树匹配字符串，匹配度100%则告警；`level2`、`3`利用计算字符串相似度函数来比较匹配度，匹配度大于85%和70%则告警。

### 2.5 choose\_patternfile

`level`为2,3时，构建攻击类型和具体特征串内容的链表。

`level`为1时：

1. 构建AC自动机树：通过读取模式文件中的攻击模式串和对应的攻击描述，构建AC自动机树数据结构。
2. 匹配攻击模式串：使用构建好的AC自动机树，对输入的文本进行匹配，以检测是否存在任何匹配的攻击模式串。
3. 读取模式文件：从给定的模式文件中读取攻击模式串和对应的攻击描述，以用于构建AC自动机树。

## 2.6 wait

1. 实现了一个等待对话框的界面。

## 2.7 show\_detection

1. 启动检测：当用户点击 `start_detection` 按钮时，进行攻击检测。根据用户选择的安全级别，注册相应的回调函数，并启动检测线程。
2. 初始化网络：在启动检测之前，进行网络初始化操作。设置网络设备、校验和控制等参数。
3. 输出攻击信息：根据安全级别，通过回调函数对接收到的网络流进行处理，并根据匹配结果输出相应的攻击信息。
4. 显示检测结果：在界面上的 `detection_output` 控件中展示检测到的攻击结果。根据不同的安全级别，展示不同的信息。

项目中主要用到的数据结构有：

- level为2,3时，保存攻击类型和具体特征串内容的链表

```
class Attackpattern{
    char attackdes[256]; // 攻击类别
    char patterncontent[256]; // 特征串内容
    Attackpattern *next;
};
```

- level为1时，AC树与攻击类型链表

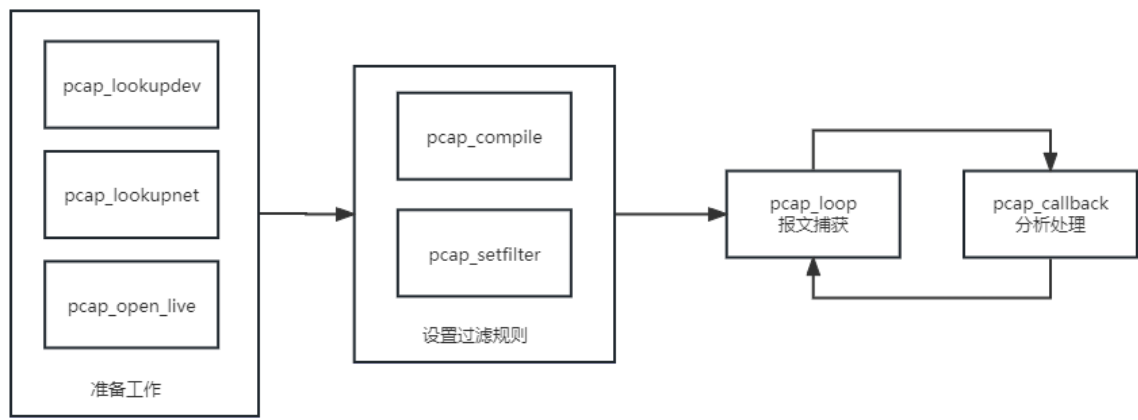
```
class Node {
    int category; // 设定攻击编号
    char attackcontent[256];
    Node* next;
};

class ACNode{
    ACNode *children[256]; // 用于存储的子节点
    int isEnd; // 用于标记是否是模式串的结尾,如果结尾的话是那种类型，为正整数；不是
    结尾则为0
    ACNode *fail; // 用于存储失配指针
};
```

## 第三章 详细设计

### 3.1 模块1——网络模块

初始时，本项目使用libpcap库进行数据报文获取，具体流程为



此时实现了对单个报文进行检测，不考虑相邻报文的合并检测的基础功能。为考虑能够合并多个报文进行分析，达到基本的抗逃避检测能力，考虑使用libnids库进行重新设计。

#### 3.1.1 初始化

为使尽可能多的报文被检测到，同时libpcap库中有默认过滤规则“ip and tcp”，而libnids库则默认不进行过滤，本项目考虑不再设置过滤规则。

libnids库中的初始化功能被集中在函数nids\_init中，仅需在调用nids\_init进行初始化前指定网络设备名nids\_params.device，否则使用默认的网络设备，并指定是否使用多线程进行抓包与数据分析，若nids\_params.multiproc设置为大于零，则使用多线程。

完成初始化后，可注册回调函数，以便在捕获数据报文后进行分析。本项目仅考虑了建立在TCP/IP协议上进行的攻击，故仅需要使用nids\_register\_tcp来注册tcp\_callback。tcp\_callback中包含了matchpattern函数与output\_alert函数，进行特征串比对与发现攻击后的提示。

同时，为防止因计算校验和引起的抓不到包的现象，关闭自动计算校验和，即在nids\_init之前使用：

```
struct nids_chksum_ctl temp;
temp.netaddr = 0;
temp.mask = 0;
temp.action = 1;
nids_register_chksum_ctl(&temp,1);
```

#### 3.1.2 数据报文获取与分析

在数据包捕获上，libnids库依然使用libpcap库的pcap\_loop函数，整个抓包与分析处理过程被封装在函数nids\_run中。

在获取到数据包后，libnids库会依次调用默认的处理函数并根据协议类型，选择之前注册过的处理函数对数据包进行分析，本项目中即为tcp\_callback。若nids\_params.multiproc大于零，则抓包与分析会在两个线程中进行。

### 3.1.3 报文合并

tcp\_stream结构能提供一个TCP连接的所有信息。其中又有两个half\_stream结构，分别保存了client到server方向的数据和server到client方向的数据。每个half stream有：

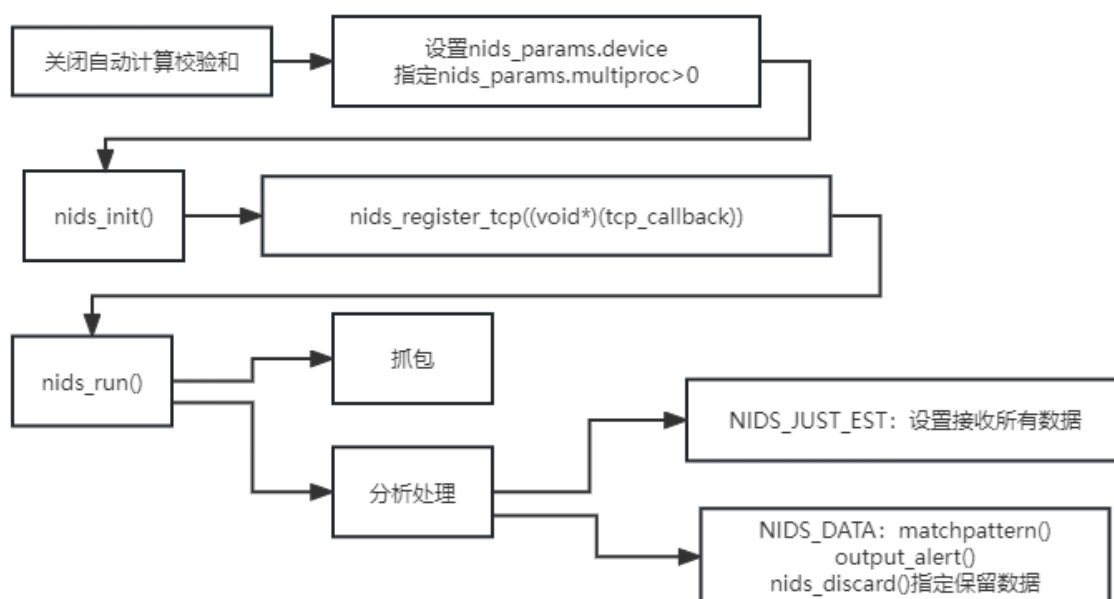
- char \* data，作为正常数据的缓冲区；
  - char collect，如果大于零，数据将被存在data缓冲区中，否则这个方向的数据流会被忽略；
  - int count，记录连接建立以来，添加到data缓冲区的字节数；
  - int offset，记录data缓冲区第一个字节在数据流中的偏移；
  - int count\_new，记录这一次添加到data缓冲区中的字节数，如果为0，则没有新数据到达。
- count-offset标记了data中数据的长度。

tcp\_stream还有一个char nids\_state，记录当前连接的状态，tcp\_callback的行为依赖于它。

- ns->nids\_state为NIDS\_JUST\_EST描述了一个刚刚建立的连接，tcp\_callback必须决定在之后该连接有新数据到达时是否被通知。如果需要考虑该连接，tcp\_callback将通知libnids期望接收的数据，包括data to client，data to server，urgent data to client，urgent data to server；
- ns->nids\_state为NIDS\_DATA表示新数据到达，half\_stream结构包含了数据所在的缓冲区；
- nids\_state为NIDS\_CLOSE，NIDS\_RESET，NIDS\_TIMED\_OUT，NIDS\_EXITING时，表明连接关闭，tcp\_callback应释放分配的资源。

为实现基本的抗逃避检测功能，合并TCP数据包进行分析，本项目考虑在读取特征库文件时，保存出现的特征串的最大长度maxpattern\_len，随后在报文处理时，每次保存之前所有报文合并后的最后maxpattern\_len个字符。使用nids\_discard可指定丢弃TCP数据的字节数，即指定不保留data中的前几位。

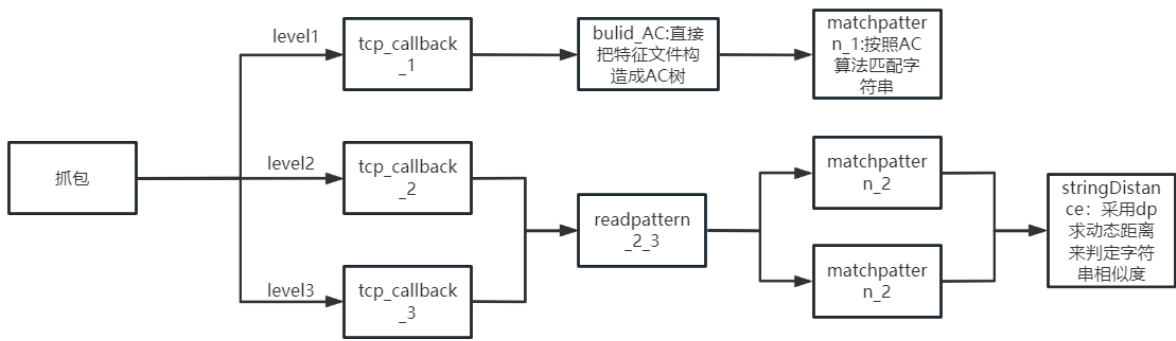
总体流程为：





## 3.2 模块2——特征串文件处理模块

特征串文件处理模块整体流程图如下：



### 3.2.1 特征串文件处理模块概述

对于用户的特征文件需要一定的格式要求，每一行记录一条特征串攻击，“#”前面表示攻击的类别，“#后面”记录具体的特征串。

特征文件的读取在readpattern函数中完成，而readpattern函数在tcp\_callback回调函数中调用。

对于用户不同的准确度要求，程序分别设置了3个不同的精度level(匹配率100%，85%，70%)，以满足用户不同的情形需求。

对于匹配率要求100%的情形(level1)，运行回调函数tcp\_callback1，此时调用bulid\_AC函数来读取特征文件，直接把特征串构造为AC树。

对于匹配率要求 > 85%的情形(level2)，运行回调函数tcp\_callback2，此时调用readpattern\_2\_3函数把特征串逐行读取，每行利用matchpattern\_2函数中的stringDistance函数求得编辑距离，再得到相似度。

对于匹配率要求 > 70%的情形(level3)，运行回调函数tcp\_callback3，此时调用readpattern\_2\_3函数把特征串逐行读取，每行利用matchpattern\_3函数中的stringDistance函数求得编辑距离，再得到相似度。

### 3.2.2 重要数据结构的构建

本模块涉及的主要数据结构有两个：AC树及链表typeofattack。

#### (1)、AC树

a、定义AC树的节点ACNode: ACNode包括存储的子节点children[CHARSET\_SIZE]，是否为模式串的标记isEnd，失配指针fail以及对应的模式串pOnepattern。

b、AC树的相关函数: 本模块中AC树包括以下函数，插入新节点的insert函数，构造失配指针的construct\_fail函数，AC树构造函数bulid\_AC。

c、为了服务于construct\_fail函数失配指针的寻找，本模块还定义了队列结构体，以及队列的创建、进队出队、检查是否为空的函数。

#### (2)、链表typeofattack

本模块构造了链表 `typeofattack` 存储攻击类型对应的具体攻击。

a、定义链表的节点Node：Node包括攻击的种类category，种类的名字字符串attackcontent，以及指向下一个节点的指针next。

b、Typeofattack的相关函数：create\_list函数以创建链表，add\_node函数以增加链表节点，delete\_node函数以删除链表节点，find\_attackcontent函数以查找攻击类别序号对应的具体类别，find\_category函数以查找具体共计类别的对应攻击序号。

### 3.2.3 tcp\_callback回调函数

当TCP连接发生变化时，NIDS会自动调用回调函数。

由于不同level下的匹配方式不同，回调函数也有所区别，以下分别讨论：

(1)、tcp\_callback1：对应用户要求100%匹配度的level1，采取AC算法以加快字符串匹配效率。首先根据nids\_state字段判断TCP连接的状态，然后根据不同的状态进行不同的处理。如果状态是NIDS\_JUST\_EST，表示连接刚刚建立，那么就让服务器和客户端都收集数据和紧急数据。如果状态是NIDS\_DATA，表示有新数据到达，那么就分别对服务器和客户端的新数据进行模式匹配，如果匹配到某个类别，就调用output\_alert\_1函数输出警报，并且用nids\_discard函数丢弃已经处理过的数据。如果状态是其他的值，比如NIDS\_CLOSE、NIDS\_RESET、NIDS\_TIMED\_OUT或者NIDS\_EXITING，那么就不做任何处理。

(2)、tcp\_callback2/ tcp\_callback3：对应用户要求85%以上和70%以上的level2和level3，采用dp算法算出编辑距离后计算相似度。整体逻辑与tcp\_callback1相同，唯一区别在于由于匹配方式的不同，需要增加一个循环语句，把特征文件的每一行都进行matchpattern。

报文合并检测这一功能也是在回调函数中实现的。

```
int num = tcp_connection->client.count-tcp_connection->client.offset-
maxpattern_len;
nids_discard(tcp_connection,num>0?num:0);
```

匹配成功后，该函数通过调用nids\_discard函数来丢弃已处理的数据。nids\_discard函数从连接缓冲区中丢弃指定数量的字节数据。如果要丢弃的字节数num大于0，则调用nids\_discard函数来丢弃这些字节的数据。否则，如果num小于等于0，则不需要丢弃任何数据，因此调用nids\_discard函数时传入0作为参数。该函数可以用于拦截TCP流量并将多个数据包中的数据合并到一起，形成完整的报文。然后，该函数可以对完整的报文进行分析，并查找其中是否包含特定的攻击模式。

### 3.2.4 matchpattern的检测

(1)、matchpatter\_1：检测数据流中是否包含特征串，如果100%精度匹配到，就返回其对应的攻击类别。首先计算剩余长度和剩余内容，然后使用AC自动机算法，快速地在数据流中查找多个模式串。从根节点开始，根据当前字符找到对应的子节点，如果没有找到，就沿着失败指针回溯，直到找到或者回到根节点。如果找到，则检查当前节点是否是一个模式串的结尾，若是，就返回标志位，反之继续匹配下一个字符。如果匹配完所有的字符都没有找到任何模式串，则返回0表示匹配失败。

(2)、matchpatter\_2/ matchpatter\_3：二者之间唯一的区别在于允许的相似度。函数先调用stringDistance函数计算出当前行特征串与抓得的数据包的编辑距离，再计算出二者相似度，如果在预先设置的阈值范围内，则调用对应的output\_alert函数输出结果。

### 3.2.5 output\_alert函数

output\_alert函数用于当matchpattern函数匹配到用户设定精度下的特征串时输出结果。输出一个QTreeWidgetItem对象，显示了三列信息，分别是攻击内容、源地址和目标地址。区别在于output\_alert\_1输出具体攻击内容时输出的是AC自动机返回的find\_attackcontent(typeofattack, category)，而void output\_alert\_2\_3则输出的是pOnepattern->attackdes。

## 3.3 模块3——图形化模块

### 3.3.1 Qt的安装与配置

本实验所用的Qt版本是5.12.7，其它部分版本同样适配，安装与配置Qt的过程参考下面链接内容：

[\(32条消息\) Linux安装qt完整版教程linux qt「QT\(C++\)开发工程师」的博客-CSDN博客](#)

### 3.3.2 从命令行到图形化

最开始我们是在命令行进行编译输出的，包括实现设备选择、等级设置等功能，在命令行窗口进行交互。为了使项目产品化、全面化，提高项目的可用性，我们采用了Qt这一图形化软件。Qt可以很好地将代码部分与图形部分区分，更易于上手操作，我们将命令行交互优化为图形界面的交互。在.ui文件中我们可以设置每一个窗口的布局、样式，包括按键、背景等要素。Qt实现交互功能的主要方式是信号与槽，通过发出信号如鼠标点击等以及接受信号后执行槽函数进行代码与图形化的连接。

在本项目中，主要用到的部件有4个：Qpushbutton、QTreeWidgetItem、Qlineedit、QMessageBox。Qpushbutton是按钮功能，每个按钮都对应一个槽函数，点击按钮后会执行槽函数内的程序；QTreeWidgetItem是一个树形控件，用于显示包含层级结构的数据。如设备的输出、Level的选择、攻击检测的输出，都是在这个控件里实现的。其内部类似于表格，可以进行增添、修改、删除，点击选中表格中每一行的内容可以通过currentItem()函数获取，所以用于实现以上功能非常合适；Qlineedit用于读取用户在输入行输入的特征文件名称，根据输入的特征文件名称检测当前文件夹下是否存在该文件，存在即进行特征文件读取和一系列初始化操作，不存在弹出一个对话框，指出无法打开该文件；QMessageBox的功能是弹出对话框给予用户一些提示信息，是图形化交互必不可少的一个部件。

以上几个部件会与代码的逻辑挂钩，其它的图形风格样式设计较为简单，在Qt提供的ui文件里直接进行操作即可。

注意，用Qt打开该文件时，在Qt界面进行编译运行无法提权，所以在show\_detection这一界面当点击show detection按钮时会显示没有用sudo提权，这里必须要在命令行窗口用sudo运行可执行文件。

### 3.3.3 图形化的发布

将完整的attackDetector项目在Linux用Qt Creator编译运行Release版本，将生成的Release文件夹下attackDetector文件放到一个英文文件夹final-release里面；在上一步的文件夹中新建文件pack.sh。内容如下：

```
#!/bin/sh
exe="attackDetector" #你需要发布的程序名称
des="/home/jangle/Desktop/final-release" #创建文件夹的位置
deplist=$(ldd $exe awk 'if (match($3,"/"))[ printf("s "),$3 ] ]')
cp $deplist $des
```

在此目录下再新建一个attackDetector.sh文件， 文件内容如下：

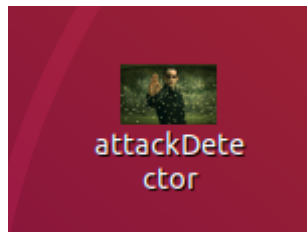
```
#!/bin/sh
appname=`basename $0 | sed s,\.sh$,`
dirname=`dirname $0`
tmp="${dirname#?}"
if [ "${dirname%$tmp}" != "/" ]; then
dirname=$PWD/$dirname
fi
LD_LIBRARY_PATH=$dirname
export LD_LIBRARY_PATH
$dirname/$appname "$@"
```

打开终端，执行命令如下

```
./pack.sh
```

会自动attackDetector.sh 讲所有依赖全部放入这个文件夹里面， 将此目录打包发布即可。

如下是我们发布的一个桌面应用程序：



## 第四章 系统实现

### 4.1 实现环境及开发工具;

本项目是在Linux系统下进行的特征串检测攻击系统，用C++语言进行代码撰写，使用QT进行代码图形化实现。

### 4.2 源文件分析及调度关系图

从程序执行的流程对各个源文件进行分析。

首先代码运行时在main.cpp中，打开Widget窗口，这个窗口在widget.h中定义，大部分项目所需的库都包含在这个头文件中，同时该头文件定义了以下几个数据结构：

```
class Attackpattern、class Node、class ACNode、class widget
```

class Attackpattern 中记录特征文件中每一行的攻击类别、攻击类别长度、特征串内容、指向下一行的 Attackpattern 类指针；class Node 中记录每一行的编号、特征串内容、指向下一节点的 Node 类指针；class ACNode 中记录AC树子节点、结尾标记变量、存储失败分配指针变量；class widget 即为窗口类，包括该窗口含有的按钮函数和指向自己的指针。widget.cpp中包含这些类的构造函数，主要函数是两个按钮函数，点击按钮即执行相应按钮函数里的内容：void on\_exitbutton\_clicked() 关闭窗口退出程序，void on\_actionbutton\_clicked() 打开下一个窗口，进入到set\_parameters.h中。

set\_parameters.h类似于widget.h，含有两个按钮函数：void on\_device\_detection\_clicked()、void on\_parameters\_confirm\_clicked()。在 set\_parameters.cpp中定义了一个全局变量level，方便记录用户选择传入下面的程序。执行 void on\_device\_detection\_clicked() 会检测本计算机可用的设备输出到图形界面中，此时用户可用选中需要检测的设备和设置检测等级，选中后点击confirm按钮执行 void on\_parameters\_confirm\_clicked() 函数，记录下用户选择的设备和Level并进入下一个图形界面。

此时要为用户提供选择特征文件的功能，用户输入选择的特征文件，并且能够对该文件进行读取和初始化链表、节点的构建等。这个功能是在choose\_patternfile.h、choose\_patternfile.cpp中实现的。用户在输入选择的特征文件名称后，根据前面选择的Level值决定选取 int build\_AC() 或者 int readpattern\_2\_3()，当Level=1时，选取前面的函数用AC树算法构造读取文件，当Level=2或3时选取后面的函数进行文件读取。包括：

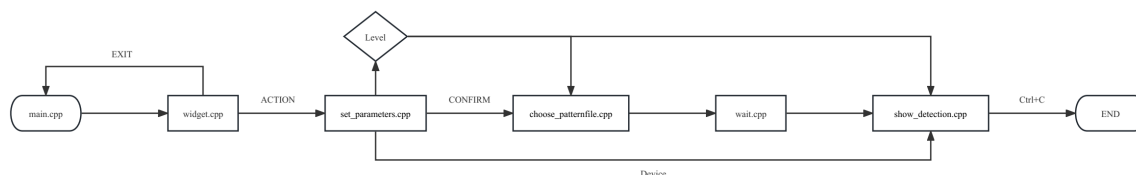
```
char* find_attackcontent()、int find_category()、void insert(c)、void  
construct_fail()、void add_node()、Node* create_list()
```

这几个函数，都是用于AC树的构建和初始化用的，在具体的模块介绍中已经提到，不再赘述。点击confirm按钮后开始进行特征文件的读取和构建。在这里有一个小小的设计，就是当特征文件较大时需要一定的时间进行读取，所以设计了wait.h和wait.cpp，当正在读取时会出现一个对话框显示特征文件正在读取，不过通常读取过程较快，该对话框不易显现。

最后一个界面是攻击检测界面，在show\_detection.h和show\_detection.cpp中实现。void output\_alert\_1()、void output\_alert\_2\_3() 分别是Level1和Level2、3的输出函数，能够将检测到的攻击输出到图形界面中；int stringDistance() 用于计算一个字符串通过增添、替换、删除变成另一个字符串所需最少步数，int matchpattern\_1() 是Level1使用AC树算法的匹配函数，int matchpattern\_2()、int matchpattern\_3() 是用于Level2、Level3的匹配函数，通过攻击串和特征文件串的相似度判定是否受到攻击，最后分别是3个Level的回调函数 void tcp\_callback\_1()、void

`tcp_callback_2()`、`void tcp_callback_3()`。值得注意的是这3个回调函数不能在其内部用一个case语句或if语句来决定选取哪个匹配函数和输出函数，这是回调函数特有的性质。点击Start Detection按钮开始进行检测，这里必须要选择和网络有关的设备并且用sudo提权才能够成功初始化libnids库的内部数据结构，在最后开始检测的 `nids_run()` 函数中，我们为它新建一个线程从而避免阻塞情况的发生，这是通过 `void workThread::run()` 函数实现的。

以下是各个源文件的调度关系图：



## 4.3 目标程序组成和运行方式

在该文件夹下的命令行输入：

```
$ sudo ./Project-code
```

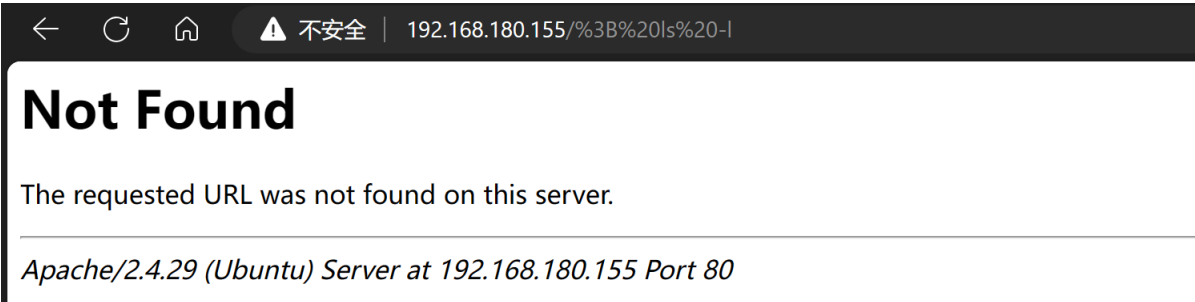
即可运行图形化界面程序。

本项目产生的目标程序位于build-Project-code-Desktop\_Qt\_5\_12\_7\_GCC\_64bit-Debug文件夹下的 `Project-code` 文件，由 `main.o`、`widget.o`、`moc_widget.o`、`set_parameters.o`、`moc_set_parameters.o`、`choose_patternfile.o`、`moc_choose_patternfile.o`、`wait.o`、`moc_wait.o`、`show_detection.o`、`moc_show_detection.o`链接而成。

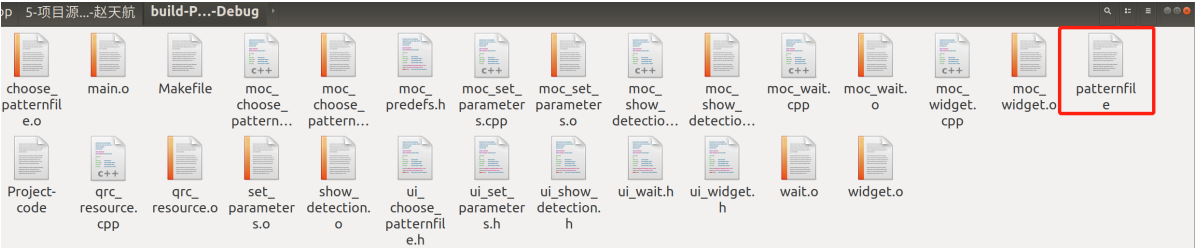
# 第五章 系统测试

## 5.1 测试方法

在攻击机网页上进行如下输入：ip+攻击串，对靶机进行攻击。这是最基本的测试。此外，还将进行有关抗逃避检测能力，稳健性的测试。



在build-Project-code-Desktop\_Qt\_5\_12\_7\_GCC\_64bit-Debug文件夹下提供您的特征文件，如下图所示：



在靶机上运行特征串攻击检测系统对攻击进行检测。需要用户设定检测设备、检测等级和提供特征文件，分别如下图所示：

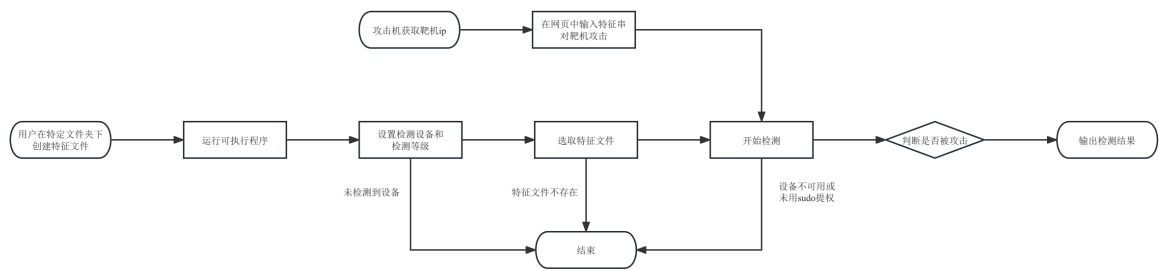
Device ID	Device name	Set security level
0	ens33	Level1
1	any	Level2
2	lo	Level3
3	bluetooth0	
4	nflog	
5	nfqueue	
6	usbmon1	
7	usbmon2	

Please enter the name of the feature string file you have selected

CONFIRM CANCEL

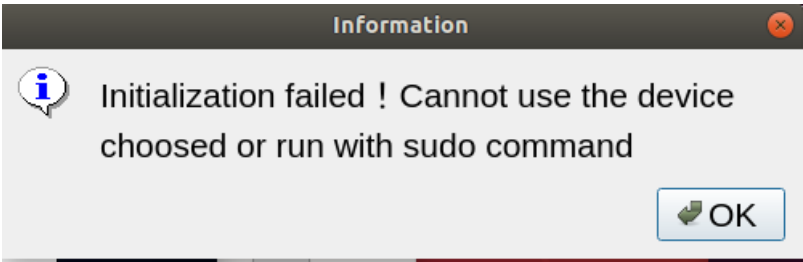


## 5.2 测试流程图

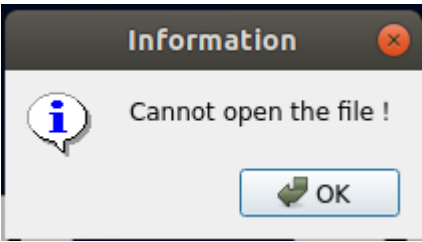


## 5.3 测试效果、现象、结论

当设备不可用或未用sudo模式运行时，能够正确提醒用户：



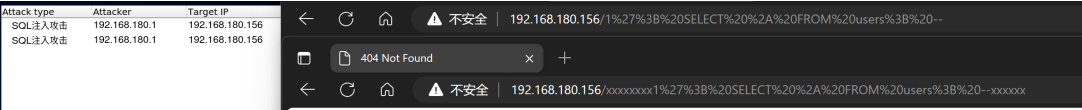
当特征文件不存在时，能够正确提醒用户：



### 1. 基本网页攻击检测

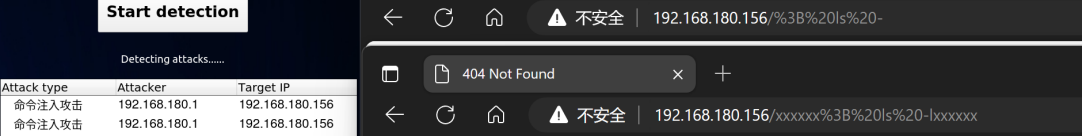
由于该项目需要在网络层面上操作，所以检测到并不是所有的device均可用，一般我们选择第一个device即ens33进行操作。

1)、选用level1测试结果：



由结果可以看出，无论是直接输入特征文件里已有的字符串，还是将该字符串前后用其它字符修饰，level1都可以检测出攻击。

2)、选用level2测试结果：

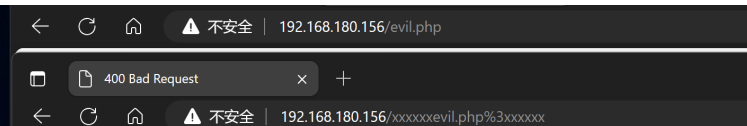


“%3B%20ls%20-l”是特征文件中的攻击串，在网页中输入“%3B%20ls%20-”，结果显示受到攻击，因为后者与前者字符串相似度大于level2所设置的阈值85%，成功检测！修饰该字符串进行攻击，结果依然可以检测成功。



### 3)、选用level3测试结果：

Attack type	Attacker	Target IP
文件上传漏洞攻击	192.168.180.1	192.168.180.156
文件上传漏洞攻击	192.168.180.1	192.168.180.156
文件上传漏洞攻击	192.168.180.1	192.168.180.156
文件上传漏洞攻击	192.168.180.1	192.168.180.156
文件上传漏洞攻击	192.168.180.1	192.168.180.156
文件上传漏洞攻击	192.168.180.1	192.168.180.156



“evil.php%3B”是特征文件中的攻击串，在网页中输入“evil.php”，结果显示受到攻击，因为后者与前者字符串相似度大于level2所设置的阈值70%，成功检测！修饰该字符串进行攻击，结果依然可以检测成功。

该测试方法可以参考目录下的视频演示。


## 2. 抗逃避检测//截图

通过运行以下python代码进行测试：

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('靶机IP', 80))
s.send(b'某攻击串前半段')
print ("已发送攻击串前半段")
s.send(b'某攻击串后半段')
print ("已发送攻击串后半段")
s.close()
```

根据测试情况，对于level1，level2，level3，均可实现抗逃避检测。

以下是用level2进行操作的测试图：



Detecting attacks.....		
Attack type	Attacker	Target IP
whois_raw.cgi		

## 3. 建立多个TCP连接时的稳定性

通过运行以下python代码进行测试：

```
import time
import socket
socks=[]
sent=0
for i in range(0,20000):
    s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('靶机IP', 80))
    s.send(b'某攻击串')
    sent = sent + 1
    print ("已发送 %s 个数据包到 %s 端口 %d"%(sent, '靶机IP', 80))
    socks.append(s)
    time.sleep(暂停时间)
for s in socks:
    s.close()
    time.sleep(暂停时间)
```

测试至建立了737个TCP连接，程序依然稳定。但无法建立737以上个TCP连接，可能与虚拟机本身的配置有关。

```

已发送 736 个数据包到 靶机IP 端口 80
已发送 737 个数据包到 靶机IP 端口 80
Traceback (most recent call last):
  File "C:\Users\user\Desktop\test.py", line 7, in <module>
    s.connect(('靶机IP', 80))
TimeoutError: [WinError 10060] 由于连接方在一段时间后没有正确答复或连接的主机没有反应，连接尝试失败。

```

#### 4. 一次TCP连接中存在大量数据时的稳定性

通过运行以下python代码进行测试：

```

import time
import socket
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sent = 0
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('靶机IP', 80))
while True:
    s.send(b'某攻击串')
    sent = sent + 1
    print ("已发送 %s 个数据包到 %s 端口 %d"%(sent, '靶机IP', 80))
    time.sleep(暂停时间)

```

测试至同一TCP连接中传输了十万量级的数据包时，程序依然稳定。

Attacker	Target IP
已发送 198869 个数据包到	端口 80
已发送 198870 个数据包到	端口 80
已发送 198871 个数据包到	端口 80
已发送 198872 个数据包到	端口 80
已发送 198873 个数据包到	端口 80
已发送 198874 个数据包到	端口 80
已发送 198875 个数据包到	端口 80
已发送 198876 个数据包到	端口 80
已发送 198877 个数据包到	端口 80
已发送 198878 个数据包到	端口 80
已发送 198879 个数据包到	端口 80
已发送 198880 个数据包到	端口 80
已发送 198881 个数据包到	端口 80
已发送 198882 个数据包到	端口 80
已发送 198883 个数据包到	端口 80
已发送 198884 个数据包到	端口 80
已发送 198885 个数据包到	端口 80
已发送 198886 个数据包到	端口 80
已发送 198887 个数据包到	端口 80
已发送 198888 个数据包到	端口 80
已发送 198889 个数据包到	端口 80
已发送 198890 个数据包到	端口 80
已发送 198891 个数据包到	端口 80
已发送 198892 个数据包到	端口 80
已发送 198893 个数据包到	端口 80
已发送 198894 个数据包到	端口 80
已发送 198895 个数据包到	端口 80
已发送 198896 个数据包到	端口 80
已发送 198897 个数据包到	端口 80

## 第六章 项目小节

### 6.1 项目实施情况

整体项目完成是按预期的，较为顺利的。在第一周我们确定了项目的目标和所需完成的任务。第二周完成了项目初始代码的实现并编译成功。此时项目功能比较单一，仅仅可以实现暴力匹配进行特征串攻击检测，效率较低。第三周到第四周结课期间我们采取功能拓展和图形化并进的举措。将特征串库保存在文件中方便管理，同时用户可以增添或删除并根据自己的需要对特征串文件进行选择。在检测前为用户设置了检测的针对性和等级选项，用户可以选择自己所需要检测的设备以及设置特征串攻击检测的Level，Level从1-3，等级越高检测误报率越高，检测越为严格。在报告中会具体讲述这部分模块的工作情况和设计理念。由于各成员的积极努力和团结合作，这部分内容实现的过程整体较为顺利。最耗时的是其中图形化输出容易乱码以及无法输出的bug、segmentation fault的bug、图形化进程和程序进程的阻塞bug，工作量最大的为Level1用AC树的方式进行文件读入、特征串匹配和Level2、Level3计算特征串相似度的方法。在第四周周三结课前，我们所有的代码和图形化实现已经完成。第四周周四阶段性确定了结题报告撰写的各部分内容和任务分配情况，周五到周日进行结题报告初稿的撰写，最后三天优化结题报告的内容以及整个项目在老师前的呈现形式。

### 6.2 不足、展望

由于尚未系统学习过计算机网络的相关知识，本项目直接使用了封装好的libnids库进行数据报文获取与IP重组、TCP重组，故程序的执行依赖于libnids库及库本身所使用的libpcap库，libnet库的正确性。

本项目仅考虑建立在TCP/IP协议上进行的攻击，而未涉及UDP、ICMP等其他协议，有待进一步提升。

本项目中的特征串比对仅限于逐字比对，而未涉及语义层面的分析，在检测能力上仍有不足，有待进一步提升。

本项目仅仅使用预先给定的特征串，而无法根据捕获的数据包进行学习，从而得到新的特征串，尽管level2与level3中相似度的概念在某种意义上试图解决这个问题；且使用特征串文件时需要关闭程序重新启动，而无法直接在原有基础上添加，有待进一步提升。

本项目检测出攻击后仅能以一种方法进行提示，而未进行多种提示方式的扩展，有待进一步提升。

本项目未能在误用检测的基础上总结正常操作应该具有的特征，对用户行为进行统计学分析，进行异常检测，有待进一步提升。

项目发布后点击桌面图标运行项目时并不能默认以管理员身份运行，所以还需要用户到命令行提权打开该可执行文件。

### 6.3 体会、感受

完成基于特征串匹配的攻击检测系统项目的过程中，我们经历了令人难忘的学习过程和挑战。这个项目为我们提供了一个难得的磨炼机会，让我们得以实操平日学习的知识，为自己的未来做准备。

首先，我们通过检查网络、蓝牙和USB端口并读取端口数据，深入了解了不同类型的攻击可能出现的位置和形式。这使我们能够更全面地理解攻击的本质和原理，同时也增加了我们对系统安全性的认识。并且，我们通过已有的攻击串文件进行匹配，学会了如何使用特征串匹配算法来识别潜在的攻击行为。这要求我们熟悉不同类型的攻击特征和模式，并找到高效的匹配算法。

同时，我们的图形界面设计为用户提供了友好和直观的操作环境。通过增加选择攻击串文件、安全级别和设备等功能，我们提高了系统的灵活性和可配置性。

在整个项目中，我们作为一个小组紧密合作，相互支持和协作。我们充分发挥每个人的优势，共同克服了技术和设计上的难题。通过团队合作，我们不仅学到了技术知识，还培养了沟通、协调和解决问题的能力。通过完成了该项目，我们对所取得的成果感到自豪，同时也意识到信息安全是一个不断发展和演变的领域，我们需要不断学习和保持警惕。这个项目为我们的学习和职业道路奠定了坚实的基础，我们期待未来在信息安全领域继续探索和贡献。

### 6.4 建议、意见

本门课程重视学生的动手实践，希望学生在时间的过程中遇到困难通过学习与实践相结合的方式解决问题，此方式确实让学生在边学边做的过程中对知识有很好的理解效果和运用能力，但是在遇到较难解决的困难时容易一筹莫展，无处下手，手足无措。建议该课程添加介绍项目开发应有的流程以及需要学习的基础知识（甚至包括提供学习资料），使学生学习的方向更加明确，不会迷失在钻研的过程中。

成员姓名	是否项目组长	具体承担任务	组成评分
赵天航	是	GUI，Level2、3算法，代码优化，协调各模块运行，统筹分工、进度安排	25%
史轩宇	否	网络模块，命令行主体代码实现，代码优化，代码风格统一	25%
陈梓轩	否	特征文件构建	10%
吴铮	否	AC树读入特征文件，Level1、2、3算法，代码优化	22%
郑祥	否	GUI，特征文件构建，项目发布	18%