# 第一章 引言

1.1 系统背景

随着互联网技术的飞适发展，现在的软件开发、WEB开发、游戏开发……都离不开互联网，但是在开发与调试与网络相关的功能时，需要自已写一些测试代码，如果出了问题之后还需要确定问题究竟是出在服务端程序还是客户端程序上，这些都需要耗费一定的时间和精力，大大降低开发的效率；网络功能测试、故障排察，针对这两个方面，需要一个高效的工具来帮助开发人员来完成这两项繁琐而又重复的工作；而现在网上有关的工具都太过于专业且单一，对于开发人员来说不太友好，所以将在本次毕业设计中做这样一个网络数据构造的工具，它能够帮助开发人员完成一些简单而又繁琐的工作；

1.2 意义

帮助开发人员完成一些简单而又繁琐的工作；

# 第二章 软件需求分析

## 2.1 软件应用场景

涉及到网络相关的开发：比如web开发、游戏开发、APP开发、普通的软件……一系列需要发送或接收TCP数据的应用；

## 2.2 需求分析

既然是开发中会遇到的工作，就先要分析我们在实际的开发中到底会进行哪些操作，首先是web开发，涉及到的基本上是HTTP协议，发送数据有两种方式：POST和GET，请求的内容是URL+参数，响应的数据比较多样化，有html、xml、json、以及各种格式的图片、视频、音乐文件，所以对于web开发者而言，做的事情相对较少，只需要完成POST和GET两种传输数据的方法，以及把收到的数据用适当的形式显示出来即可；

其次是对于普通软件开发和游戏开发而言，现在大多数的开发而言使用的都是面向连接的可靠协议TCP，特别是游戏应用中，对网络的连接要求最高，玩家的数据需要即时而又准确地传输到服务器，所以在这些应用中，为了保证数据传输的可靠性，都是采用的TCP协议，而TCP协议可以传输的数据类型更为多，比如TCP可以传输原始的二进制数据，这一点是HTTP不具备的，但是原始的二进制数据游戏需要考虑的方面实在是太多（图片，基本数据类型、纯文本、音频数据……），在数据太大需要分片时，更加难以处理，所以这里无法对所有的数据类型进行处理；好消息是在使用TCP时，几本上会传输太过复杂的数据，因为复杂的数据都有更上层来处理，比如文件传输使用FTP，网页使用HTTP、邮件传输使用SMTP……作了这些分析后发现在一般的软件和游戏中传输的数据类型都是纯二进制或json、xml数据，而json和xml都属于纯文本的范畴，二进制都是些自定义的数据类型，也就是c/c++、java中的struct/class，这些都是由char、int、long、float、double等基本数据类型构成；

但是对于在程序运行的时候，需要查看一下程序到底发送/接收到了什么数据，这个时候就需要数据抓取的功能，功能也就是网上所说的抓包软件wireshark之类的，但是wireshark对于抓取的数据包是把数据链路层、ip层、tcp层的协议数据也显示出来了，而这些对于应用开发人员来说是完全没有必要 的，所以在我所设计的这个软件中，不再显示这些底层协议，而是直接把我们发送的具体内容显示出来，而这些接收到的数据也是原始的TCP数据，处理方式和发送TCP数据时的处理方式一样，这样可以将它的数据解析部分与发送TCP数据部分的数据解析功能独立出来，设为公共部分。

## 2.3 功能总结

综合上面需求分析里面提到的内容，这个软件的功能具体如下：

1. 发送HTTP请求：请求方式可以选择POST或GET；
2. 发送TCP数据：发送的数据可以是纯文本、二进制数据流；
3. 抓取与指定主机+端口的通信数据，并且可以查看单条数据的详细信息；
4. 整个软件都带图形界面，方便用户操作；
5. 对于TCP数据，支持加密解密，压缩解压功能，应对多变的环境；

## 2.4 开发环境的选择；

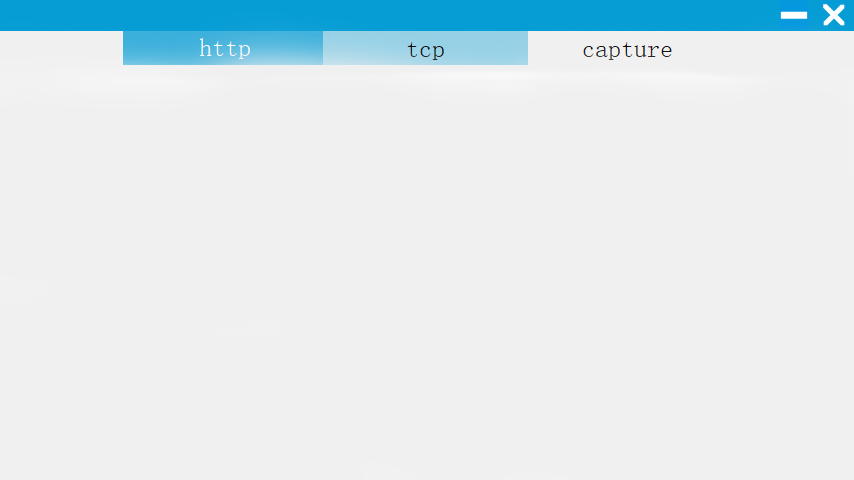
现在主流的操作系统是windows，linux和mac用户只是极少数，所以不作考虑，软件的图形界面使用强大的开源图形框架Qt，与网络交互的部分也使用Qt中封装的网络部分；数据抓取使用开源软件wireshark 的底层库winpcap，数据压缩暂时使用现阶段应用较为广泛的quicklz算法，数据加密解密使用对称加密算法RC4；

# 第三章 软件框架设计

## 3.1 子系统划分以及系统模块设计

a. 主界面：

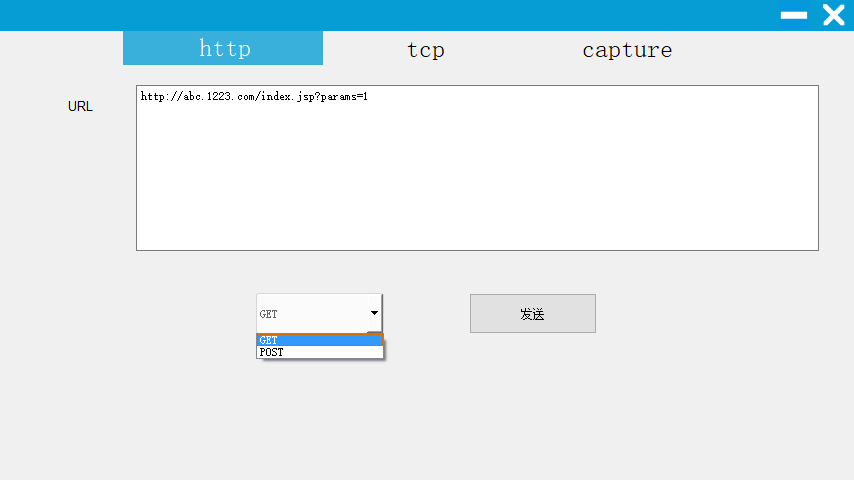
主界面直接将所有的功能放在主界面下，通过Qt提供的TableWidget来回切换



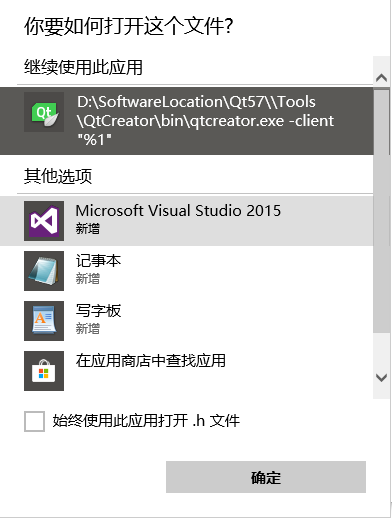
软件中的三个主要功能：发送HTTP数据、发送TCP数据、数据抓取，通过这个TabelWidget来回切换，可以三个功能同时使用，且相互之间不会干扰,主界面的窗口使用QT重写，这样相比于不重写的界面在相同大小的平面上，可以放更多的内容，对于屏幕较小的用户来说无疑是很大的福音

b. HTTP数据包发送模块：

如图，为HTTP数据发送模块的界面，



用户可以在名为URL的文本输入框内输入要访问的链接，格式为标准的GET请求格式，如果需要以POST的方式请求，则直接在下方的下拉列表框中，选择POST请求方式即可，格式相同，在输入URL和选择请求方式后，就可以点击发送按钮发送HTTP请求；收到请求后，会根据根据HTTP响应头中的Content-Type来确定是哪种类型的文件，然后根据文件的类型，将收到的数据保存到本地，文件名的后缀与Content-Type类型相对，然后通过系统的文件资源管理器来选择对应的默认应用程序来打开此文件，若没有设置打开此类型文件的默认应用程序，系统则会弹出对应的提示框，提示用户选择打开此类型文件的应用



选择应用后就可以打开该应用了，不过一般的都会使用浏览器打开应用；

当然，请求的内容只能是文本的格式，这样的功能看起来虽然是单一了点，却是完全够用的，想想我们平时浏览网页的时候，最多也只是点几下鼠标、输入英文字符以及汉字，要说发送其它的数据，可能也就是在特殊的应用场景了，已经脱离绝大多数软件开发者的实际了，而且做起来难度也要大很多，由于时间有限，所以这里不做实现；

c. TCP数据发送模块：

TCP数据发送这个模块，相比HTTP数据发送的模块要复杂得多，因为使用TCP数据传输的时候，多数时间是出于性能、安全性等特殊的原因考虑的，而且还有一个因素是绝大多数使用TCP协议来传输数据的软件，都对数据进行了压缩、加密的操作，且数据完全是二进制的格式，需要将它转化为char、int、double、string等数据类型、所以要完全实现与开发者的应用相接轨，是有相当的难度的，基于以上的原因，需要将这个模块拆分为以下几个模块：（1）数据包的构造；（2）数据包的加密与解密、压缩与解压缩；（3）对于收到的网络数据包的解析；（4）网络数据包的发送与接收（5）几个界面的独立设计；

要将这些功能全部拆分开来看的话，整个软件系统设计得过分完美，同时也是设计过度的，所以需要对这几个功能进行一下合并，将其它设计为适当当的大小；但对于各个模块又完全地解耦合，对于数据包的构造和网络数据的传输，可以放在一个模块，也就是主界面中去，加密解密和压缩解压又作为一个独立的模块，数据的解析也作为一个独立的模块，与后面的数据监听模块共用；

d. 数据监听模块：

数据监听这个模块主要是监听指定网卡，指定IP端口上的数据，也可以对网上上的数据进行简单的筛选，当然也只是对数据包的长度，目的IP和目的端口来进行筛选，对于收到的网络中的数据，显示摘要数据，详细的数据用文件保存起来，需要的时候再对文件进行读取，因为如果将网络中的数据全部通过内存的形式缓存起来，可能会因为数据太多，而出现内存不够用的情况，这种情况对于内存较小的老机器尤其严重；另外，对于数据监听这一块，由于它需要一直循环，且不会退出监听函数，如果要把它与界面放在一个线程里面的话，界面中的函数不会执行，应付出现假死的情况，所以需要把数据监听的功能单独放到一个线程里面去，边监听边保存，并将收到的数据的简要信息发送到主界面的线程中，显示出来，如果需要查看数据包的详细信息，再读取本地文件中的数据，加载到主界面中，但文件读取也是一个耗时的操作，当文件太大时，读取文件的时间也相对变长了，也会使主界面出现假死的情况，所以为了避免此类情况的出现，需要单独开辟一个线程来完成文件读取的操作，这样主界面也就只是进行的数据的显示和控制的作用，真正的数据在另外两个子线程中，这也是现阶段MVC出现最多的形态，控制层和显示层混合在一起；当然数据抓取只针对会用到的TCP协议，与TCP数据发送模块相同，也需要数据加密解密，压缩解压的功能；



如图所示，为数据抓取模块的设计，从上方的网络接口的QComboBox下拉菜单中选取一个本机的网络接口，再输入要抓取数据的IP地址、端口、最小数据长度，点击开始即可进行网络中的数据抓取，如果需要加密解密、压缩解压的功能，那么点击高级选项的按钮，就可以弹出一个子对话框，在子对话框中，可以选取压缩数据的算法，加密解密的算法，从而对将要发送的数据进行加密，对接收到的数据进行解密，操作相当简单直观；

对于接收到的网络数据要相查看详细信息，双击单条记录，就可以显示信息的详细信息，此部分与TCP数据发送部分的数据接收模块公用，将在下一部分介绍；

1. 数据解析模块：

上面介绍了TCP数据发送模块和网络数据监听的模块，一直在提数据解析这个模块，但是一直没有给出实现，下面将在这一部分作详细的分析，数据解析模块的功能看起来比较直观，它的功能就只有一个，那就是显示数据：但是我们不仅仅需要显示ascii字符，还要让它可以显示编程中的基本数据类型，比如short、int、long long int、float、double基本数据类型，这些数据类型都是有着统一的标准的，与语言的实现无关，所以就算是用其它编程语言写的程序中使用到这些数据，都是可以正常显示的；有时候我们只需要看数据指定位置上的数据的值，这时可以能过程序中读取string的长度来设置数据偏移，然后再读取指定位置上的数据；

1. 数据压缩解压、加密解密模块：

数据的压缩和加密，这两个功能涵盖的面实在是太广了，由于时间的和个人能力的限制，这里不可能将所有的加密算法和压缩算法全部加进去，所以只选取两个应用比较广泛且具有代表性的算法来实现；

网络数据传输中的压缩与解压，主要的作用是减小发送数据的大小，减小数据中的冗余、空白部分，使数据在传输过程当中更加快速可靠，但是由于处理的时间很短，所以对算法的效率要求非常之高，但对于能压缩多少的空间这方面并不是特别重要，要选取网络数据压缩算法时，最重要的因素就是要求压缩的时间短，至于压缩比例，只要不是特殊的应用场景，要求都不高；quicklz号称是最快的数据压缩算法，在数据压缩率上相对其它的算法也毫不逊色，所以这次就只选取了quicklz来做为测试算法，完成数据压缩和解压这一功能；

软件中的数据采用透明传输？数据被窃取了怎么办？为了保证用户的合法信息，现在的软件在网络传输中，对于关键的数据都是进行了加密处理的，也就是说发送的数据是已经加密过的数据，服务端接收到数据后，会按照加密的钥匙进行解密，如果数据没有经过加密，解密出来的数据到底是什么估计没有人会知道，还很有可以导致服务端崩溃，所有说数据的发送在特殊的场景是需要进行加密处理的；服务端向我们发来加密的数据之后，我们不能解密，看到的只是一串不明不白的数据，也就是所说的无效信息，这当然不是我们所看到的，所以对于接收到的数据还需要进行解密操作；

上面探讨了数据加密解密这个功能的必要性，那么接下来就需要分析到底采用什么样的数据加密算法来做（同数据压缩算法一样，数据加密算法各类众多，且还有一些新的算法在出现，所以不可能通通实现，这里也只是选取一种具有代表性的加密算法来使用）；现阶段的数据加密算法主要分为两大类，对称加密算法和非对称加密算法，

## 3.2 系统流程图

先放着，有时间再画

# 第四章 软件详细设计

4.1 HTTP数据发送：

4.2 TCP数据发送：

4.3 TCP数据解析：

4.4 数据压缩解压、加密解密：

4.5 数据监听：

4.6 整合：

# 第五章 系统测试

5.1 测试环境简介

5.2 各功能模块的测试

# 第六章 总结

内容

参考文献