# 计算机网络课程设计报告

## 班级：

## 2016211306

## 组员：

## 系统的功能设计

课程设计题目：DNS中继服务器的实现

 设计一个DNS服务器程序，读入“域名-IP地址”对照表，当客户端查询域名对应的IP地址时，用域名检索该对照表，实现下列三种情况：

 1. 检索结果为IP地址0.0.0.0，则向客户端返回“域名不存在”的报错消息（即不良网站拦截功能）

 2. 检索结果为普通IP地址，则向客户返回这个地址（即DNS服务器功能）

 3. 表中未检到该域名，则向实际的本地DNS服务器发出查询，并将结果返给客户端（即DNS中继功能）

此外，我们还实现了如下的附加功能：

1. TTL字段每秒减1，并在减到0时自动删除记录
2. 对 ’A’ 类型的请求，实现了通过 ‘CNAME’ 的递归查询
3. 对 MX类型的请求。。。
4. 由日志记录每个请求的发送方、请求的域名、类型，以及备注（数据源-本地/远端DNS服务器/超时/不良网站拒绝访问）
5. 可以选择由图形化的界面实时查看中继服务器的数据库状态。

## 模块划分

1. **主模块**

**DNSRelay.py**

类 DNSRelay 即为DNS中继服务器类。每当收到新的请求，主模块开启一个新的线程，并把这个请求交给事务处理模块在新线程中处理。主模块另有一个常驻线程负责数据库中TTL的更新。

1. **事务处理模块**

**Handler.py**

类 Handler 对收到的一个请求进行处理，包括：分析数据包、查询数据库、对查询结果的分析、向远端DNS服务器请求并接收（可选），以及向请求方做出应答。

一项事务处理完毕后，事务处理模块会向日志中写入本次处理的记录（时间、请求的发送方、请求的域名、类型、返回的信息来源）

1. **数据存储模块**

**DBFacade.py**

类DBFacade 是数据库的外观控制器。采用了单例模式。数据库给其他模块提供了插入记录、查找一条记录、查看数据库全部值的接口，还提供了更新一次TTL的方法（全部记录TTL减一，减到0自动删除）。

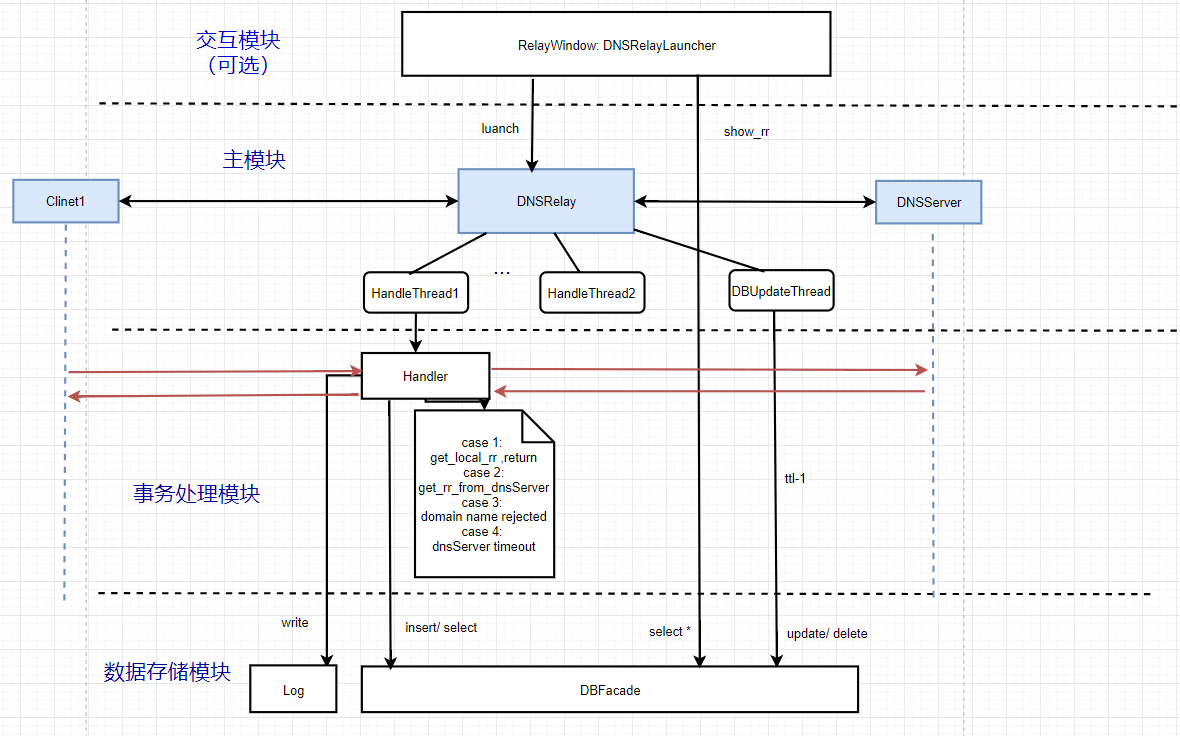
MyLog.py：日志相关

1. **交互模块**

RelayUi.py：由qtdesigner设计的ui界面自动转成的py文件

RelayWindow.py ：图形窗口，可以启动DNSRelay，并查看数据库中的值

## 软件流程图



## 四、测试用例以及运行结果

## 五、调试中遇到并解决的问题

1. WinError 10045：

Windows下socket的Bug，忽略了这个异常。在linux下无此问题。

1. 数据库插入效率过低导致程序卡顿：

优化了sql语句，进行批量插入，全部语句执行结束后，最后再进行提交（commit）操作。

1. 数据库异常：不允许指针的递归调用

Sqlite 不支持并发调用指针。在数据库模块中维护锁来保护sqlite表的指针。

1. 数据库异常：不允许不同的线程访问同一个表

由于前面已经由锁来保护了数据库，多线程访问实际上是安全的。在数据库connect语句中加check\_same\_thread = False ，设置为允许多线程访问。

1. 未成功拦截需要拦截的地址

地址设为了‘0.0.0.0’但是依然向远端DNS服务器查询。发现请求类型不是’A’，而是’AAAA’。于是新增资源记录，将要拦截地址的’AAAA’ 的值设为’0:0:0:0:0:0:0:0’

1. 超时异常捕获失败：Except socket.timeout 语句报错

已有import socket。将文件头的 from socket import \* 删除，并相应更改其他有关语句后解决了此问题。

## 六、小组成员分工及承担任务比例

## 七、心得体会