课程设计报告

1. 系统的功能设计
   1. 接受DNS报文

接受一个DNS的报文，如果是非DNS协议规定的包会不予处理，接受报文时采用非阻塞式IO，以减少CPU在等待数据包和复制数据包时所花的时间。

* 1. 处理DNS报文

处理DNS报文时，采用多线程，为每一个DNS报文建立一个线程，在

* 1. 转发DNS报文

1.4 输出调试信息

二、模块划分

2.1 网络I/O的设计与选择

采用python原生的socket,使用定期轮询的方式处理请求，请求到来时为每个请求开设一个线程处理。

2.1.1 非阻塞式IO设计

2.1.2 多线程设计

2.2 数据库的设计与选择

数据库采用sqlite3,工作在串行模式,采用连接池的方式,初始化指定数目的连接供线程处理调用。

2.2.1 数据库连接池

2.2.2 数据库表的设计

2.3 DNS报文的拆封包

采用struct来对报文进行pack和unpck，其中format参数为big-endian(!),来保证字节序一致。

2.3.1类的设计

Header类

Query类

RR类

Message类

2.3.2

三、软件流程图

四、测试用例以及运行结果

4.1 各模块测试

4.1.1 并发服务器测试

此部分主要测试服务器能否并发而且非阻塞式的响应请求，线程开启后在处理完任务时是否可以正常关闭结束。

4.1.2 数据库连接池测试

此部分主要测试线程之间能否正常共享使用数据库的连接池。

4.1.3 DNS报文拆包测试

此部分主要测试当收到一个dns报文时，能否将报文内容进行还原

① 域名测试

*#域名拆包*

n = Name()

strio = BytesIO(strio.getvalue())

n.decode(strio)

print(n.name)

测试结果

[Running] python -u "c:\Users\咸鱼\Desktop\DnsServer\test.py"

b'\x03www\x05baidu\x03com\x00\xc0\x0c'

b'www.baidu.com'

4.1.4 DNS报文封包测试

此部分主要测试如何根据请求发送一个dns报文

1. 域名测试

*# 域名封包测试*

def testEncodeName():

addr = b'www.baidu.com'

n = Name(addr)

strio = BytesIO()

nameDict = {}

n.encode(strio, *nameDict*=nameDict)

print(nameDict[addr])

n.encode(strio, *nameDict*=nameDict)

print(strio.getvalue())

测试结果

[Running] python -u "c:\Users\咸鱼\Desktop\DnsServer\test.py"

12

b'\x03www\x05baidu\x03com\x00\xc0\x0c'

②

4.1.5 DNS报文转发测试

从部分主要测试在转发报文中来回id是否匹配，能否正确的实现转发

4.2 综合测试

4.2.1 四种资源类型测试

4.2.2

五、调试中遇到并解决的问题

1.win10051

<https://stackoverflow.com/questions/30462729/error-10051-while-trying-to-connect-socket>

2.调用连接池的时候提示连接不能在创建的线程之外使用

check\_same\_thread = false

sqlite3

3.sqlite3不支持smallint等数据类型

自己本着简单的原则，选择了sqlite3这个轻型数据库，但是发现这个数据库在存储时可以识别其他类型数据库的数据格式，比如smallint但是会做一个类型转换

六、小组分工及承担任务比例

七、心得体会

自己本着尝试的心态，想独立做一个东西出来。也算是十分用心，在写代码的途中也遇到了许多问题。但更多的遇到的是对各种技术的探讨与比较以及最后的选择，比如在网络IO这一段,之所以选择非阻塞式IO是因为其比较简单。