

网络安全实验报告

题 目 基于 libnet 的程序设计

专 业 网络安全

学 号 1170300728

学 生 汤添凝

指 导 教 师 王彦

# 一、实验目的

掌握网络爬虫的实现思想以及应用场景。

**二、实验内容**

1. 掌握网络爬虫的原理
2. 编程实现基于 python的网络爬虫。能够有实际用途的爬虫应用。

# 三、实验过程

**基于python的爬虫实验基本信息：**

实验环境：Windows 编程语言：Python 语言

## 1. 需求分析

爬虫并非一定在http上实现，本次实验试图找到一些其他类型的网络流量进行爬虫工作，找到一些“以获取网页为基础”以外的爬虫思路。

这里选择了DHT网络作为爬虫目标。

本文所说的DHT网络是BT网络中的一部分。BitTorrent使用一种叫做分布式哈希表(distributedsloppy hashtable)的技术，来实现在无tracker的torrent文件中peer的联系信息存储。这个时候，每个peer都是一个tracker。这个协议是基于Kademila协议的，并且在UDP协议基础上实现。可以说，DHT网络真正意义上实现了P2P的文件分发传输。

那么在DHT网络中的爬虫要怎么做呢？其实思想是相似的，从一个结点出发爬取更多的结点，最终对收集得到的结点做操作。

仔细阅读DHT协议（链接见参考文献），对DHT网络请求报文有以下描述：

## find\_node

Find node is used to find the contact information for a node given its ID. "q" == "find\_node" A find\_node query has two arguments, "id" containing the node ID of the querying node, and "target" containing the ID of the node sought by the queryer. When a node receives a find\_node query, it should respond with a key "nodes" and value of a string containing the compact node info for the target node or the K (8) closest good nodes in its own routing table.

这里的翻译大致是说：向某个结点发送find\_node请求报文，请求某个结点ID，目的是希望对方结点返回该结点的网络地址，但是对方结点可能没有记录这个结点id，于是对方结点会返回距离该结点最近的k（8）个结点的信息，让我们去距离更近的地方寻找资源。

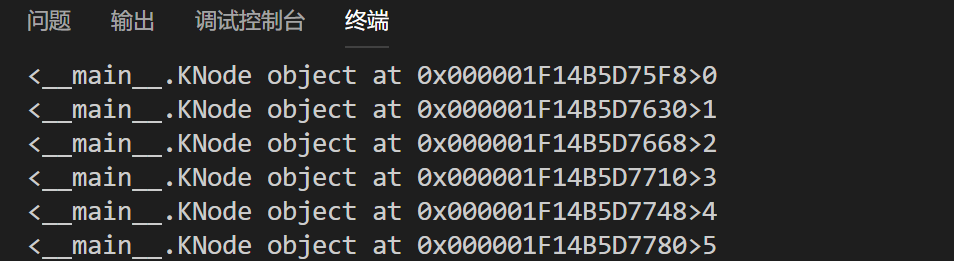
我们利用这一特性就可以每次从一个结点找到8个其他结点，如此循环，即可达到爬虫的目的。

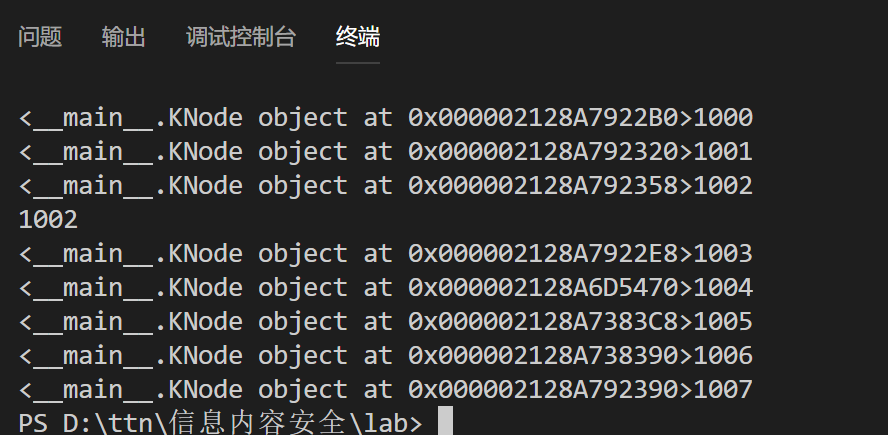
## 2. 程序结构

程序分为两个部分，一个部分是DHT的客户端，另一个部分是DHT的服务器端。分别负责构造findnode请求报文，和接收回复报文

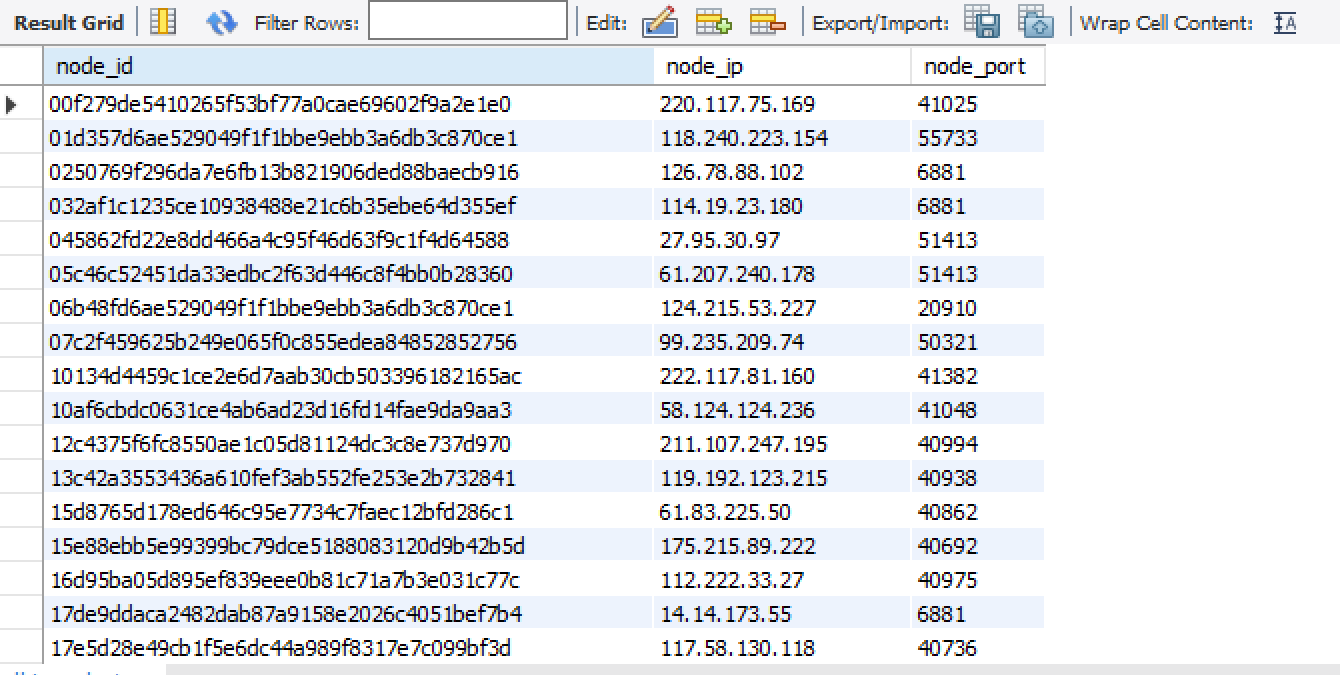
# 实验结果

运行lab4.py文件：





保存到数据库中，每个nodeid对应的结点信息均有体现：



# 心得体会 （出现问题分析）

本次实验内容为爬虫实际体验，开放性强，需要学生更多的独立思考判断并实现，某种程度上而言难度很大，但在不断努力尝试后成功了。大致了解了DHT网络的结构，掌握了爬虫的思想，会在将来的学习生活中活学活用。

# 参考文献

<http://www.bittorrent.org/beps/bep_0005.html> DHT协议原文档