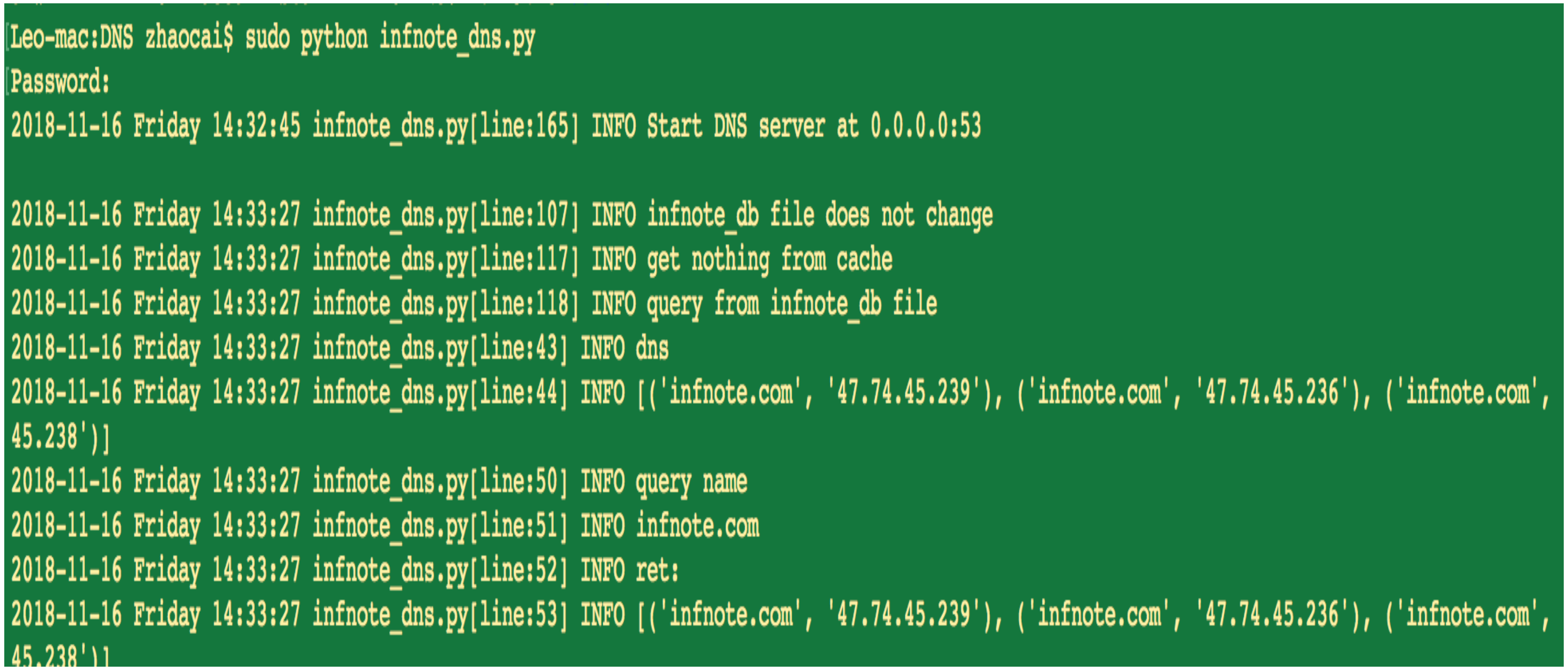
DNS Seeding

Zhao Cai

赵 才

Results(1)

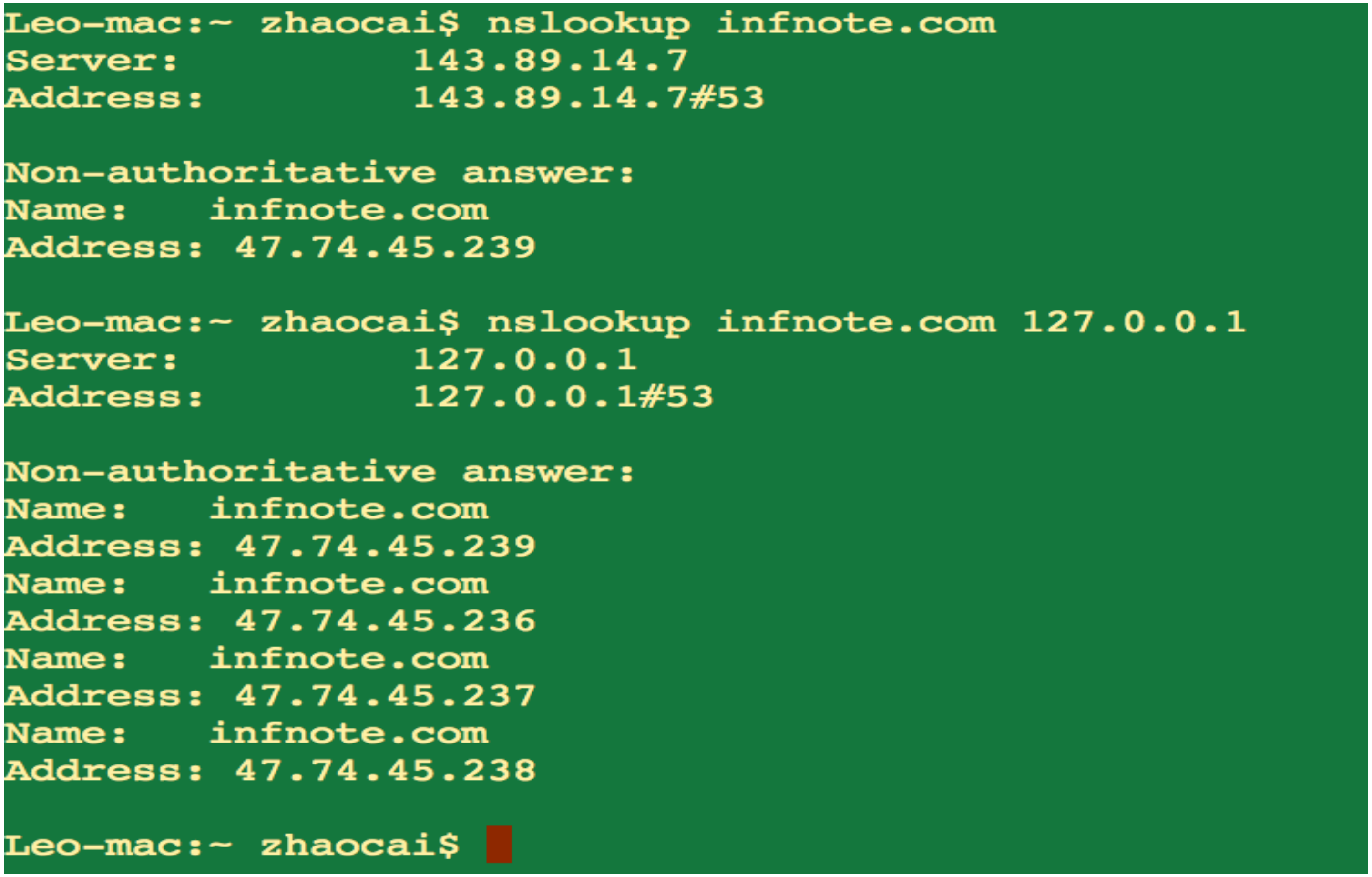
- **1. start infnote_dns sever**



```
[Leo-mac:DNS zhaocai$ sudo python infnote_dns.py
[Password:
2018-11-16 Friday 14:32:45 infnote_dns.py[line:165] INFO Start DNS server at 0.0.0.0:53

2018-11-16 Friday 14:33:27 infnote_dns.py[line:107] INFO infnote_db file does not change
2018-11-16 Friday 14:33:27 infnote_dns.py[line:117] INFO get nothing from cache
2018-11-16 Friday 14:33:27 infnote_dns.py[line:118] INFO query from infnote_db file
2018-11-16 Friday 14:33:27 infnote_dns.py[line:43] INFO dns
2018-11-16 Friday 14:33:27 infnote_dns.py[line:44] INFO [('infnote.com', '47.74.45.239'), ('infnote.com', '47.74.45.236'), ('infnote.com',
45.238')]
2018-11-16 Friday 14:33:27 infnote_dns.py[line:50] INFO query name
2018-11-16 Friday 14:33:27 infnote_dns.py[line:51] INFO infnote.com
2018-11-16 Friday 14:33:27 infnote_dns.py[line:52] INFO ret:
2018-11-16 Friday 14:33:27 infnote_dns.py[line:53] INFO [('infnote.com', '47.74.45.239'), ('infnote.com', '47.74.45.236'), ('infnote.com',
45.238')]
```

Results(2)

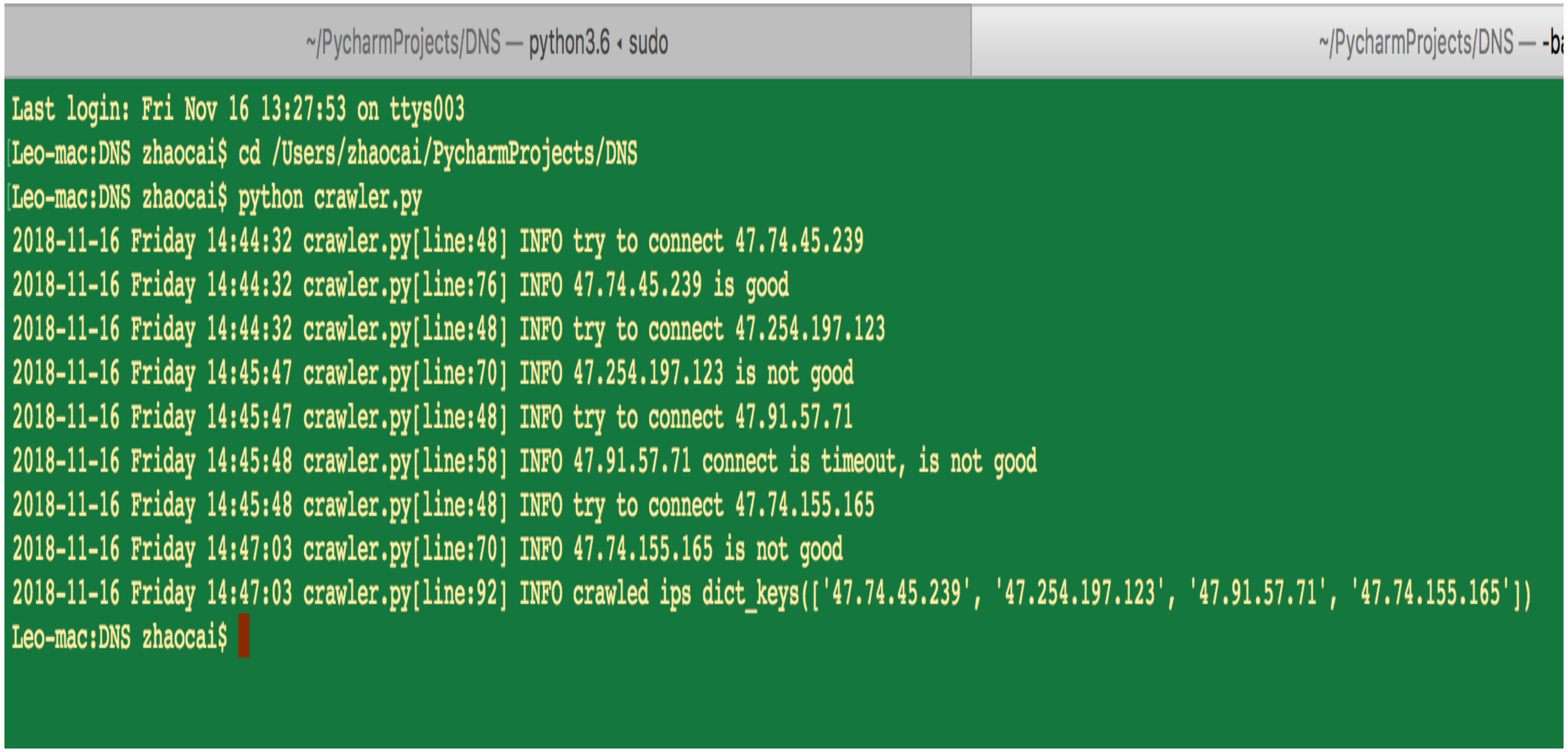
- **2. nslookup infnote.com**



```
Leo-mac:~ zhaocai$ nslookup infnote.com
Server:         143.89.14.7
Address:        143.89.14.7#53

Non-authoritative answer:
Name:   infnote.com
Address: 47.74.45.239

Leo-mac:~ zhaocai$ nslookup infnote.com 127.0.0.1
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   infnote.com
Address: 47.74.45.239
Name:   infnote.com
Address: 47.74.45.236
Name:   infnote.com
Address: 47.74.45.237
Name:   infnote.com
Address: 47.74.45.238

Leo-mac:~ zhaocai$ 
```

Results(3)

- **3. run crawler**



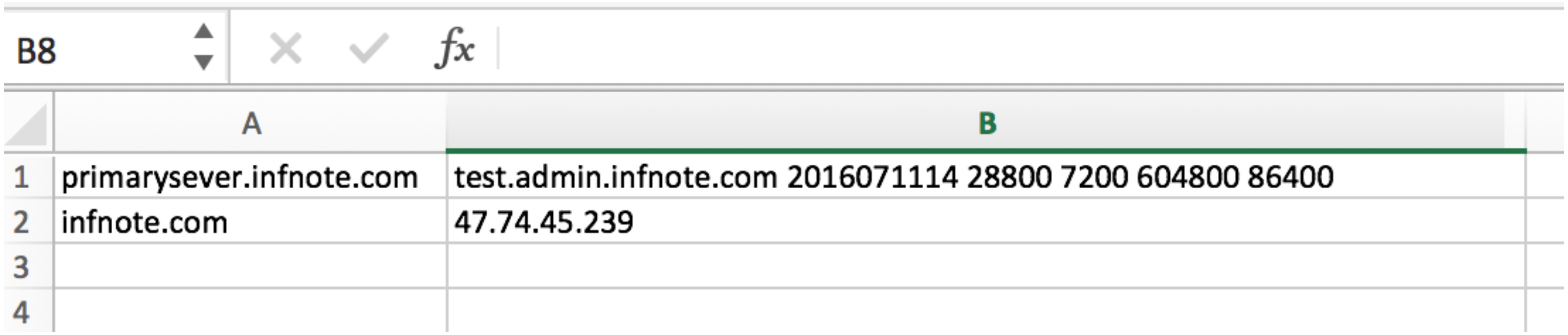
~/PycharmProjects/DNS — python3.6 ◂ sudo

~/PycharmProjects/DNS — -ba

```
Last login: Fri Nov 16 13:27:53 on ttys003
Leo-mac:DNS zhaocai$ cd /Users/zhaocai/PycharmProjects/DNS
Leo-mac:DNS zhaocai$ python crawler.py
2018-11-16 Friday 14:44:32 crawler.py[line:48] INFO try to connect 47.74.45.239
2018-11-16 Friday 14:44:32 crawler.py[line:76] INFO 47.74.45.239 is good
2018-11-16 Friday 14:44:32 crawler.py[line:48] INFO try to connect 47.254.197.123
2018-11-16 Friday 14:45:47 crawler.py[line:70] INFO 47.254.197.123 is not good
2018-11-16 Friday 14:45:47 crawler.py[line:48] INFO try to connect 47.91.57.71
2018-11-16 Friday 14:45:48 crawler.py[line:58] INFO 47.91.57.71 connect is timeout, is not good
2018-11-16 Friday 14:45:48 crawler.py[line:48] INFO try to connect 47.74.155.165
2018-11-16 Friday 14:47:03 crawler.py[line:70] INFO 47.74.155.165 is not good
2018-11-16 Friday 14:47:03 crawler.py[line:92] INFO crawled ips dict_keys(['47.74.45.239', '47.254.197.123', '47.91.57.71', '47.74.155.165'])
Leo-mac:DNS zhaocai$
```

Results(4)

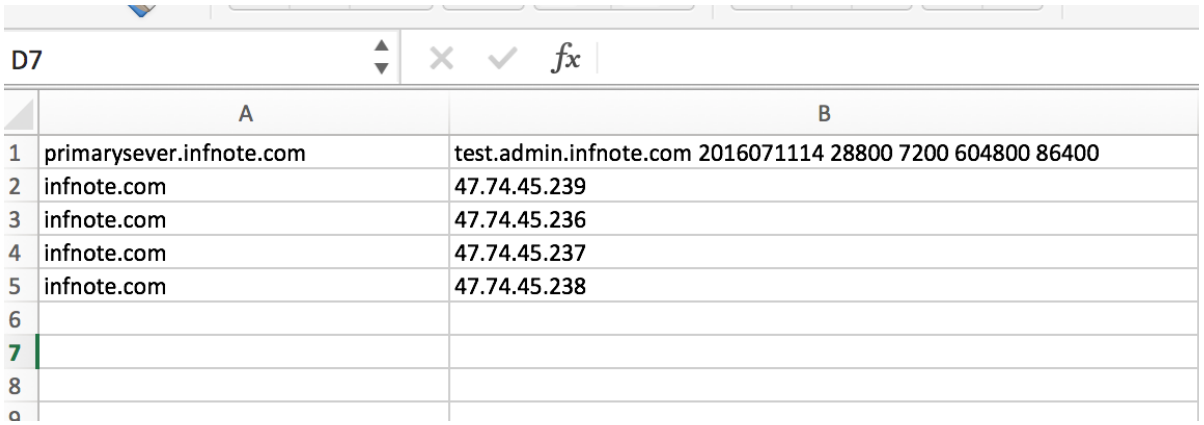
- **4. update infnote_db.csv (the first line is a SOA record)**



B8

	A	B
1	primarysever.infnote.com	test.admin.infnote.com 2016071114 28800 7200 604800 86400
2	infnote.com	47.74.45.239
3		
4		

- **5.backup of infnote_db.csv: infnote_db_old.csv**

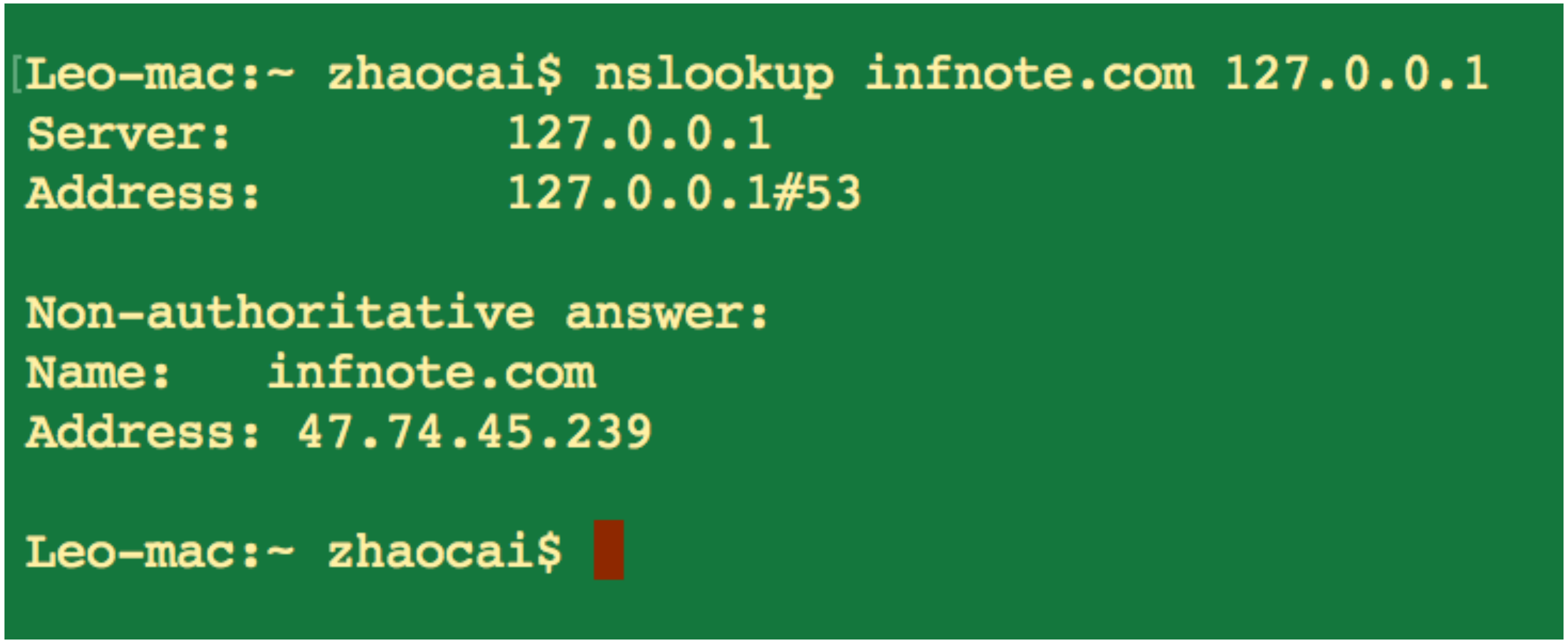


D7

	A	B
1	primarysever.infnote.com	test.admin.infnote.com 2016071114 28800 7200 604800 86400
2	infnote.com	47.74.45.239
3	infnote.com	47.74.45.236
4	infnote.com	47.74.45.237
5	infnote.com	47.74.45.238
6		
7		
8		

Results(5)

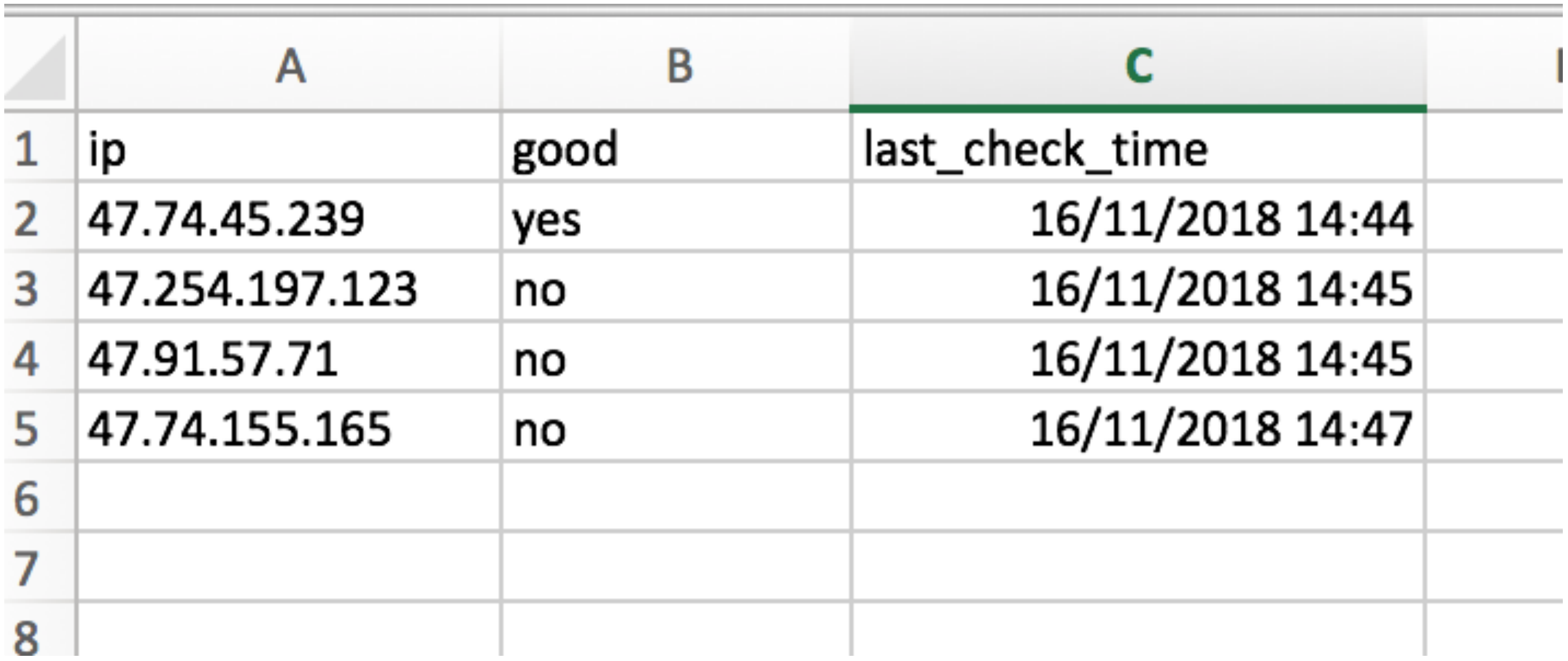
- **5. nslookup infnote.com after crawler**



```
Leo-mac:~ zhaocai$ nslookup infnote.com 127.0.0.1
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   infnote.com
Address: 47.74.45.239

Leo-mac:~ zhaocai$ 
```

Results(6)

- **2. generate nodes check report :nodes.csv**



	A	B	C
1	ip	good	last_check_time
2	47.74.45.239	yes	16/11/2018 14:44
3	47.254.197.123	no	16/11/2018 14:45
4	47.91.57.71	no	16/11/2018 14:45
5	47.74.155.165	no	16/11/2018 14:47
6			
7			
8			

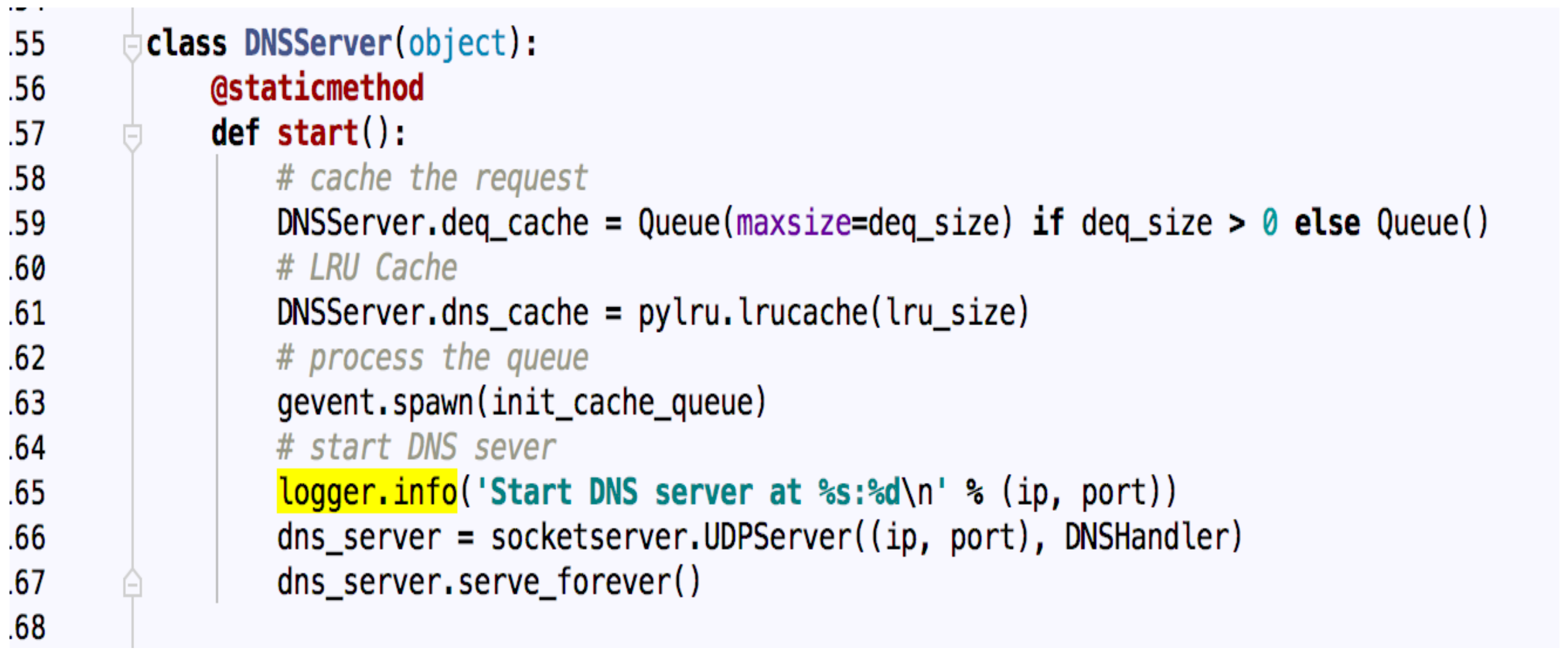
Key Python Pakages

- 1. gevent is used to start coroutines
- 2.gevent.queue is used as request message queue
- 3.dnslib is used to encode/decode DNS packets
- 4.pylru is used as LRU cache

Steps of infnote_dns

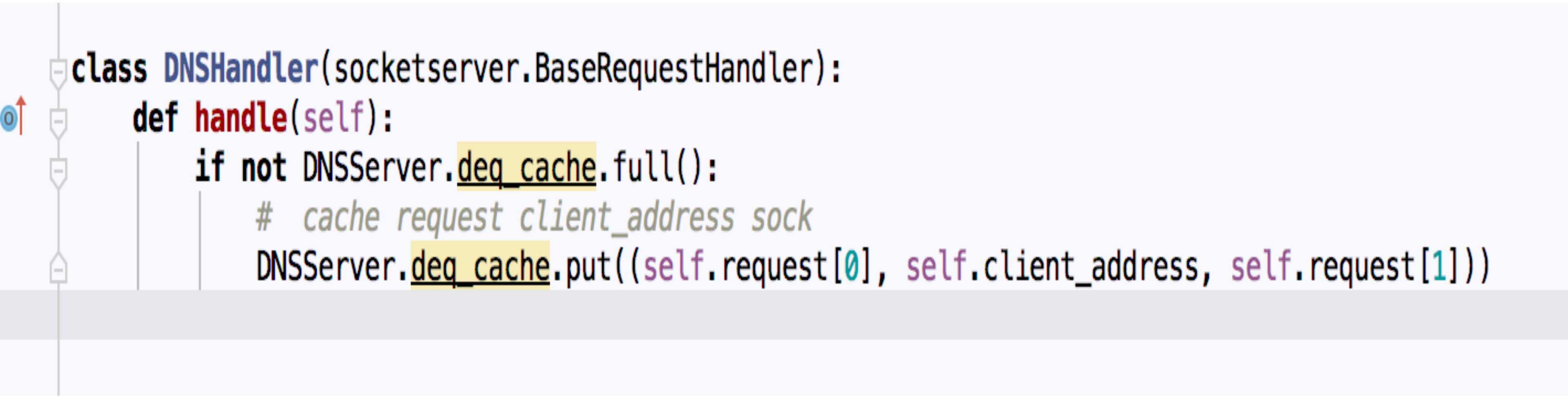
- 1. start udp sever
- 2.accept requests, and store in deq_cache
- 3. get data from deq_cache
- 4. decode DNS packets from data
- 5. if dns db does not change and the result of quary in dns_cache, just return the result from the dns_cache, else quary from dns db and return the result, all the records are packed as DNS packet

1. start udp sever



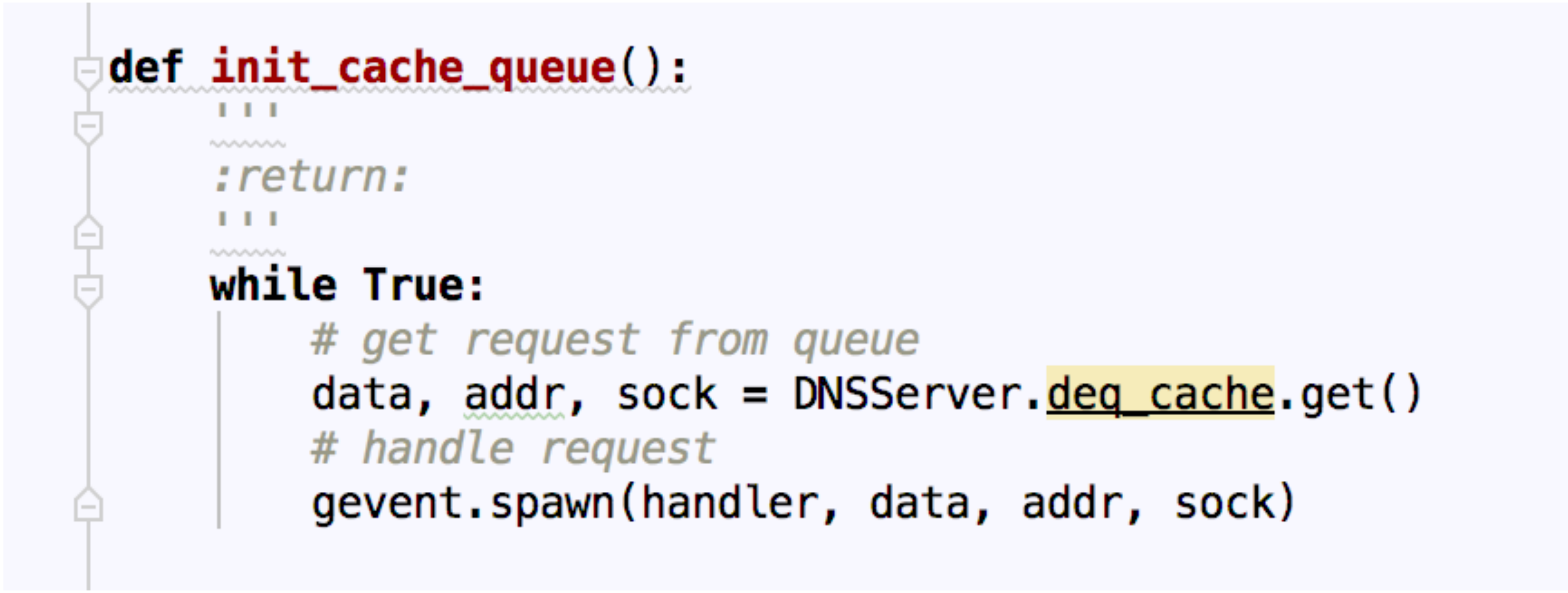
```
class DNSServer(object):
    @staticmethod
    def start():
        # cache the request
        DNSServer.deq_cache = Queue(maxsize=deq_size) if deq_size > 0 else Queue()
        # LRU Cache
        DNSServer.dns_cache = pylru.lrucache(lru_size)
        # process the queue
        gevent.spawn(init_cache_queue)
        # start DNS sever
        logger.info('Start DNS server at %s:%d\n' % (ip, port))
        dns_server = socketserver.UDPServer((ip, port), DNSHandler)
        dns_server.serve_forever()
```

2.accept requests, and store in deq_cache



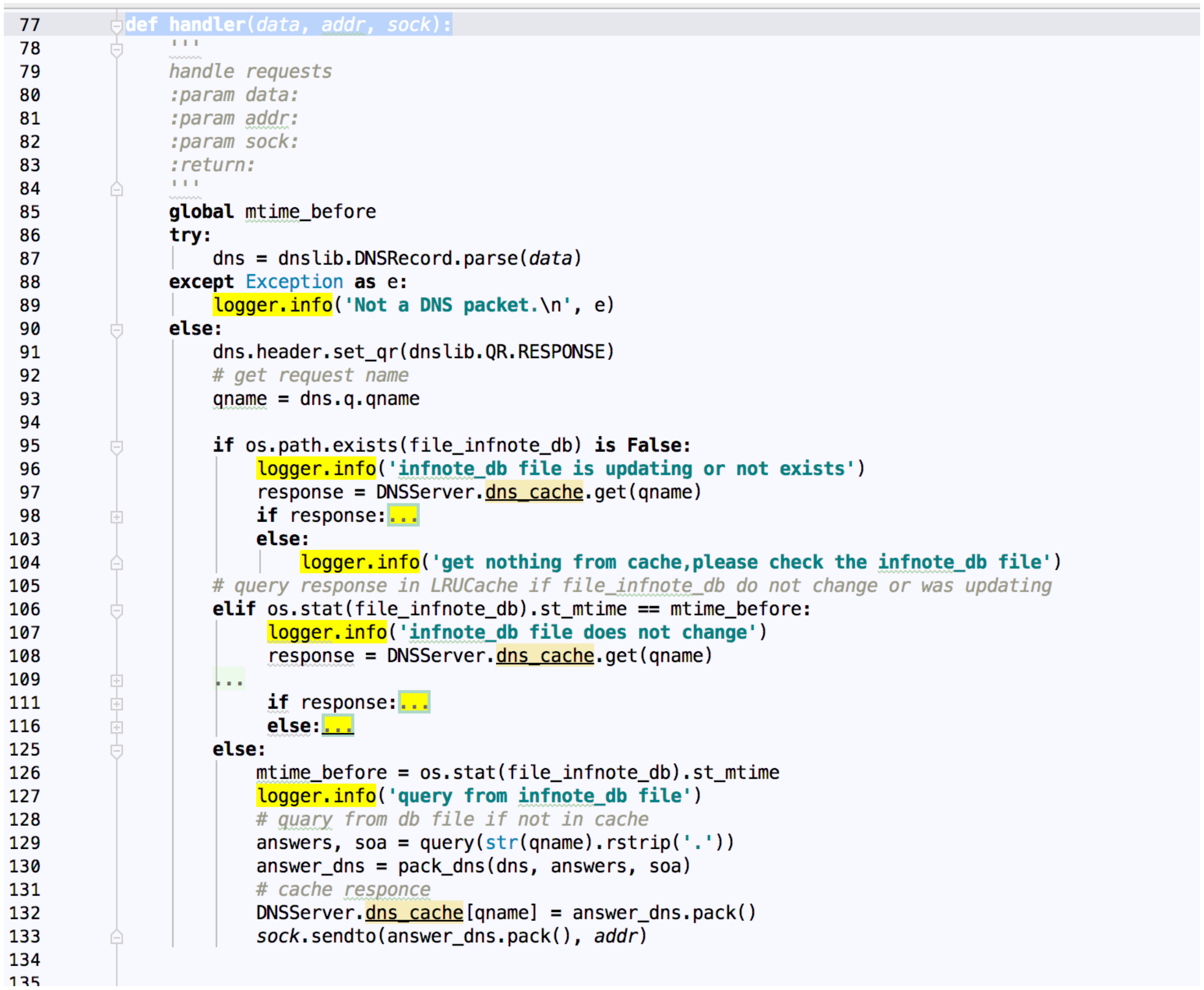
```
class DNSHandler(socketserver.BaseRequestHandler):
    def handle(self):
        if not DNSServer.deq_cache.full():
            #  cache request client_address sock
            DNSServer.deq_cache.put((self.request[0], self.client_address, self.request[1]))
```

3.get data from deq_cache



```
def init_cache_queue():
    '''
    :return:
    '''
    while True:
        # get request from queue
        data, addr, sock = DNSServer.deq_cache.get()
        # handle request
        gevent.spawn(handler, data, addr, sock)
```

Step 4&5



```
def handler(data, addr, sock):
    '''
    handle requests
    :param data:
    :param addr:
    :param sock:
    :return:
    '''
    global mtime_before
    try:
        dns = dnslib.DNSRecord.parse(data)
    except Exception as e:
        logger.info('Not a DNS packet.\n', e)
    else:
        dns.header.set_qr(dnslib.QR.RESPONSE)
        # get request name
        qname = dns.q.qname

        if os.path.exists(file_infnote_db) is False:
            logger.info('infnote_db file is updating or not exists')
            response = DNSServer.dns_cache.get(qname)
            if response:...
            else:
                logger.info('get nothing from cache,please check the infnote_db file')
        # query response in LRUCache if file_infnote_db do not change or was updating
        elif os.stat(file_infnote_db).st_mtime == mtime_before:
             logger.info('infnote_db file does not change')
             response = DNSServer.dns_cache.get(qname)
        ...
             if response:...
             else:...
        else:
            mtime_before = os.stat(file_infnote_db).st_mtime
            logger.info('query from infnote_db file')
            # quary from db file if not in cache
            answers, soa = query(str(qname).rstrip('.'))
            answer_dns = pack_dns(dns, answers, soa)
            # cache responce
            DNSServer.dns_cache[qname] = answer_dns.pack()
            sock.sendto(answer_dns.pack(), addr)
```

Steps of crawler

- 1. define a global ips = {‘47.74.45.239’:False},the key is the first live fullnode’ ip, the value is False indicates the fullnode is not visited
- 2. want_peers from the not visited fullnodes in ips , get peer’s ip from the response meassge and put them in ips if ip is not in ips. If the fullnode is visited set the value as True
- 3.Repeat 2 until, all the fullnode is visited
- 4.Generate infnode_db.csv and nodes.csv, a report of fullnodes’ availability