IP_over_DNS

Pre-install packet

- python-pytun

```
$ python3 -m pip install python-pytun
```

or download the zip packet and decompress it

```
$ wget https://github.com/montag451/pytun/zipball/v2.3.0
$ python3 setup.py
```

Client-Usage

```
$ sudo bash start_client.bash dns_ip_address
```

Server-Usage

```
$ sudo bash start_server.bash dns_ip_address
```

Set up proxy in Client

- First set up system proxy

```
$ sudo ssh -v -i cs305project_cn_nw.pem -D 192.168.0.1:8080 ubuntu@192.168.0.2
```

- Set up the proxy in browser set the proxy server with address: 192.168.0.1:8080 And the sock version is SOCK5
- Then you can browse web page on your browser

Main points of the solution

- Config routes The tun can be set up by python

```
self._tun = pytun.TunTapDevice(name='tun0', flags=pytun.IFF_TUN)
      self._tun.addr = taddr
      self._tun.dstaddr = tdstaddr
      self._tun.netmask = tmask
      self._tun.mtu = tmtu
      self._tun.up()
```

And we set up SOCK5 proxy on both system and browser as we showed above So when we browse the webpage all the packet will go through the tunnel

- Encode the data The outgoing data from client will be stored in the qname of query record. And the server side will store the data into two TXT type answer field
- Protocol between client and server We use roll polling to continually send simple query messages. Then the server can send back the data with this simple query messages. So the client can timely get the data from server side with the response of these simple query messages.
- Extra features We split data into several pieces if it is too large to store in one DNS response. The server side will cut the data into several pieces and send them in several times. And the client will put the together according to the index.