CS168 Programming Assignment 4: DNS spoofing using raw sockets

Assignment Out:	November 22, 2013
Assignment Due:	December 6, 2013 Demo time

1 Introduction

In this project, you will implement a simple DNS spoofing service/daemon that will intercept any outgoing dns requests and redirect to a user supplied domain.
This project is designed to make you familiar with raw sockets and how they can used to capture/fake local packet traffic.

2 The Steps

In this assignment you will write a program to implement a DNS service/daemon.
The daemon would constantly check for any DNS requests to a particular/all domain(s) and spoof a response redirecting to a predefined domain.
No libraries such as libpcap may be used.

2.1 Spoofing behavior

A stepwise description of the process involves:
- Open a raw socket that intercepts all network traffic and finds any DNS requests
- Prepare a spoofed response packet with the same source, destination information as the original request packet
-Remember to switch source and destination addresses in the IP and Ethernet headers
- All header information including checksum must be correct for the reply to be processed successfully
- The spoofing daemon responds to DNS requests faster than the original DNS response and hence is used to complete the request
- The actual DNS information received from the server would be discarded/dropped since the original would have already been completed successfully

3 Implementation

We suggest you use C for this project. However if you choose to work in any other language, please notify us beforehand. No libraries can be used to support packet sniffing or packet manipulation.
A major aspect of this project is to prepare a fake packet with all headers correctly set so that the requesting application accept it. You can check for an actual request-response packet flow in Wireshark.
The IP headers and checksum is something you should be focusing on. You can continue to fake the headers for as many layers as you want but not all layers (physical layer headers) need to be recontructed. We strongly encourage you to try to fake headers for ethernet but it is not essential.
An example run of the service should be:

```
dnspoof redirectDomain originalRequestedDomain
```

The first parameter `redirectDomain` is mandatory. If the second paramter is '*', redirect all requests to the redirect domain.

3.1 Note

If you face problems of incomplete types in headers, we encourage you to setup a virutal machine with a Linux distro such as Ubuntu. We coded our implementation in C on Ubuntu (run Ubuntu as a VM using VirtualBox).

3.2 Demo

For a demo you could have your laptop setup to run the daemon/service in background and have a web browser fire the requests. Any request URL should be redirected to a predefined address.
We will check for two cases:
- Redirect all requests to a particular address
- Redirect a particular request to another one

A few notes:

- You should use the UDP packet format, exactly as-is. You can use the header found in *netinet/udp.h*
- You can use the IP packet format, exactly as-is. You can use the header found in *netinet/ip.h*
- You can use the Ethernet packet format, exactly as-is. You can use the header found in *netinet/ethernet.h*

4 Extra Credit - 10%

Extra credit avaialble for demonstrating following setup:
- Setup two machines - L1 and L2 on the same network with the same DNS server
- L2 should have a raw socket opened and sniff DNS requests by L1
- L2 should send spoofed respone to L1
- L1 is redirected per the response from L2

5 Getting Started

5.1 Spoofing reference implementation

Available in `/course/cs168/pub/spoof` as `node`

5.2 Hand In

```
/course/cs168/bin/cs168_handin spoof
```