Spoofing an IP address involves sending packets that appear to originate from a different IP address than the sender’s. This technique is often used in networking simulations and testing, but it’s worth noting that IP spoofing can also be used in harmful ways, and most network protocols make efforts to detect and prevent it. Additionally, using spoofed IP addresses on a real network can violate terms of service or even laws in many regions.

Below is a simplified example of how an IP spoofing client could be set up using raw sockets in C. This code is for educational purposes only and should be tested in a controlled, local environment.

**Requirements:**

• Root or administrative privileges, as raw sockets require these.

• A local network or controlled environment to avoid network disruptions.

**Sample Code (Educational Purposes)**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <arpa/inet.h>

#include <sys/socket.h>

#include <netinet/ip.h>

#include <netinet/udp.h>

#define DEST\_PORT 8080

#define DEST\_IP "192.168.1.10" // Change to the server's IP

#define SPOOFED\_IP "192.168.1.100" // Change to the desired spoofed IP

// IP header length

#define IP\_HDRLEN 20

#define UDP\_HDRLEN 8

#define PSEUDO\_HDRLEN 12

// Calculate checksum (IP/UDP checksum algorithm)

unsigned short checksum(void \*b, int len) {

unsigned short \*buf = b;

unsigned int sum = 0;

unsigned short result;

for (sum = 0; len > 1; len -= 2) {

sum += \*buf++;

}

if (len == 1) {

sum += \*(unsigned char \*)buf;

}

sum = (sum >> 16) + (sum & 0xFFFF);

sum += (sum >> 16);

result = ~sum;

return result;

}

int main() {

int sockfd;

struct sockaddr\_in dest\_addr;

char \*message = "Hello from spoofed client!";

// Message payload length

int payload\_len = strlen(message);

// Raw socket

if ((sockfd = socket(AF\_INET, SOCK\_RAW, IPPROTO\_UDP)) < 0) {

perror("Socket creation error");

exit(1);

}

// Define destination address

dest\_addr.sin\_family = AF\_INET;

dest\_addr.sin\_port = htons(DEST\_PORT);

inet\_pton(AF\_INET, DEST\_IP, &dest\_addr.sin\_addr);

// Packet length

int packet\_len = IP\_HDRLEN + UDP\_HDRLEN + payload\_len;

char \*packet = malloc(packet\_len);

if (!packet) {

perror("Failed to allocate memory for packet");

exit(1);

}

memset(packet, 0, packet\_len);

// IP Header

struct ip \*ip\_hdr = (struct ip \*)packet;

ip\_hdr->ip\_hl = IP\_HDRLEN / sizeof(unsigned int);

ip\_hdr->ip\_v = 4;

ip\_hdr->ip\_tos = 0;

ip\_hdr->ip\_len = htons(packet\_len);

ip\_hdr->ip\_id = htons(0);

ip\_hdr->ip\_off = 0;

ip\_hdr->ip\_ttl = 64;

ip\_hdr->ip\_p = IPPROTO\_UDP;

inet\_pton(AF\_INET, SPOOFED\_IP, &(ip\_hdr->ip\_src));

inet\_pton(AF\_INET, DEST\_IP, &(ip\_hdr->ip\_dst));

ip\_hdr->ip\_sum = checksum(ip\_hdr, IP\_HDRLEN);

// UDP Header

struct udphdr \*udp\_hdr = (struct udphdr \*)(packet + IP\_HDRLEN);

udp\_hdr->uh\_sport = htons(12345); // Spoofed source port

udp\_hdr->uh\_dport = htons(DEST\_PORT);

udp\_hdr->uh\_ulen = htons(UDP\_HDRLEN + payload\_len);

udp\_hdr->uh\_sum = 0;

// Message payload

memcpy(packet + IP\_HDRLEN + UDP\_HDRLEN, message, payload\_len);

// Send packet

if (sendto(sockfd, packet, packet\_len, 0, (struct sockaddr \*)&dest\_addr, sizeof(dest\_addr)) < 0) {

perror("Sendto error");

exit(1);

}

printf("Packet sent with spoofed IP %s\n", SPOOFED\_IP);

close(sockfd);

free(packet);

return 0;

}

**Explanation:**

1. **IP Header**: We construct a custom IP header and manually set the source IP to the spoofed IP address.

2. **UDP Header**: We set up a UDP header to define the source and destination ports, as well as the packet length.

3. **Checksum**: Calculating a valid checksum is essential to ensure the packet is correctly received.

4. **Sending**: Using sendto, we send the packet to the destination address.

**Important Notes**

• **Root Privileges**: This code requires root privileges to create and send raw packets.

• **Testing Environment**: Use a private network or virtual network setup.

• **Legal Implications**: Spoofing IP addresses on a public network can be illegal; this code is strictly for educational purposes.