<https://blog.csdn.net/nodeathphoenix/article/details/31774885>

一、客户端代码

#include <sys/types.h>

#include <sys/socket.h>  
#include <stdio.h>  
#include <sys/un.h>  
#include <unistd.h>  
#include <stdlib.h>  
#include <iostream>  
  
using namespace std;  
  
int main (int argc, char \*argv[])  
{  
        struct sockaddr\_un address;  
        int sockfd;  
        int len;  
        int result;  
        char buffer[5000] = {0};  
        int rc = 0;  
  
  
        if ((sockfd = socket(AF\_UNIX, SOCK\_STREAM, 0)) == -1) {  
                perror ("socket");  
                exit (EXIT\_FAILURE);  
        }  
  
        address.sun\_family = AF\_UNIX;  
        strcpy (address.sun\_path, "server\_socket");  
        len = sizeof (address);  
  
  
        result = connect (sockfd, (struct sockaddr \*)&address, len);  
        if (result == -1) {  
                printf ("ensure the server is up\n");  
                perror ("connect");  
                exit (EXIT\_FAILURE);  
        }  
  
        unsigned int sum = 0;  
        while(1) {  
                rc = recv(sockfd, buffer, 4096, 0);  
                if (0 >= rc) {  
                    cout << "rc is less than or equal to 0" << endl;  
                    cout << "sum is : " << sum << endl;  
                    return 0;  
                }  
  
                sum += rc;  
                if (rc < 4096) {  
                        cout << "rc is : " << rc << endl;  
                }  
        }  
  
        return 0;  
}

二、服务端代码

#include <sys/types.h>  
#include <sys/socket.h>  
#include <stdio.h>  
#include <sys/un.h>  
#include <unistd.h>  
#include <stdlib.h>  
#include <iostream>  
  
using namespace std;  
  
#define COUNT 5000000  
  
int main (int argc, char \*argv[])  
{  
        int server\_sockfd, client\_sockfd;  
        int server\_len, client\_len;  
        struct sockaddr\_un server\_address;  
        struct sockaddr\_un client\_address;  
        int i, bytes;  
        char ch\_send, ch\_recv;  
        unlink ("server\_socket");  
  
        server\_sockfd = socket (AF\_UNIX, SOCK\_STREAM, 0);  
  
        server\_address.sun\_family = AF\_UNIX;  
  
        strcpy (server\_address.sun\_path, "server\_socket");  
  
        server\_len = sizeof (server\_address);  
  
        bind (server\_sockfd, (struct sockaddr \*)&server\_address, server\_len);  
  
        listen (server\_sockfd, 5);  
        printf ("Server is waiting for client connect...\n");  
        client\_len = sizeof (client\_address);  
  
  
        client\_sockfd = accept (server\_sockfd, (struct sockaddr \*)&server\_address, (socklen\_t \*)&client\_len);  
        if (client\_sockfd == -1) {  
                perror ("accept");  
                exit (EXIT\_FAILURE);  
        }  
  
        char buffer[COUNT] = {0};  
        memset(buffer, 'a', COUNT);  
  
        if ((bytes = write (client\_sockfd, buffer, COUNT)) == -1) {  
                printf ("Failed to write data...\n");  
  
        }  
        cout << "written bytes is : " << bytes << endl;  
  
        close (client\_sockfd);  
        unlink ("server socket");  
}  
  
三、执行测试

先执行server程序，然后再启动客户端程序，结果如下：  
#./server  
Server is waiting for client connect...  
written bytes is : 5000000  
  
# ./client   
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 3776  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 4032  
rc is : 3456  
rc is less than or equal to 0  
sum is : 5000000

可见，server端发送一次数据后，client在recv时指定期望长度4096，则循环接收数据的时候，绝大多数recv返回值是4096，但是可能间或收到的小于4096，即出现返回值为4096和小于4096的值交替的情况。故而验证了，recv确实是返回当前任意长度的可读的数据。  
  
The receive calls normally return any  data  available,up to the requested amount, rather than waiting for receipt of the full amount requested.