**服务器端的步骤如下：**

**1. socket**：      建立一个socket

**2. bind**：          将这个socket绑定在某个文件上（AF\_UNIX）或某个端口上（AF\_INET），我们会分别介绍这两种。

**3. listen**：        开始监听

**4. accept**：      如果监听到客户端连接，则调用accept接收这个连接并同时新建一个socket来和客户进行通信

**5. read/write**：读取或发送数据到客户端

**6. close**：        通信完成后关闭socket

**客户端的步骤如下：**

**1. socket**：      建立一个socket

**2. connect**：   主动连接服务器端的某个文件（AF\_UNIX）或某个端口（AF\_INET）

**3. read/write**：如果服务器同意连接（accept），则读取或发送数据到服务器端

**4. close**：        通信完成后关闭socket

上面是整个流程，我们先给出一个例子，具体分析会在之后给出。**例子实现的功能是客户端发送一个字符到服务器，服务器将这个字符+1后送回客户端，客户端再把它打印出来**：

**Makefile：**

1. all: tcp\_client.c tcp\_server.c
2. gcc -g -Wall -o tcp\_client tcp\_client.c
3. gcc -g -Wall -o tcp\_server tcp\_server.c
5. clean:
6. rm -rf \*.o tcp\_client tcp\_server

**tcp\_server.c：**

1. #include <sys/types.h>
2. #include <sys/socket.h>
3. #include <sys/un.h>
4. #include <unistd.h>
5. #include <stdlib.h>
6. #include <stdio.h>
8. **int** main()
9. {
10. /\* delete the socket file \*/
11. unlink("server\_socket");
13. /\* create a socket \*/
14. **int** server\_sockfd = socket(AF\_UNIX, SOCK\_STREAM, 0);
16. **struct** sockaddr\_un server\_addr;
17. server\_addr.sun\_family = AF\_UNIX;
18. strcpy(server\_addr.sun\_path, "server\_socket");
20. /\* bind with the local file \*/
21. bind(server\_sockfd, (**struct** sockaddr \*)&server\_addr, **sizeof**(server\_addr));
23. /\* listen \*/
24. listen(server\_sockfd, 5);
26. **char** ch;
27. **int** client\_sockfd;
28. **struct** sockaddr\_un client\_addr;
29. socklen\_t len = **sizeof**(client\_addr);
30. **while**(1)
31. {
32. printf("server waiting:\n");
34. /\* accept a connection \*/
35. client\_sockfd = accept(server\_sockfd, (**struct** sockaddr \*)&client\_addr, &len);
37. /\* exchange data \*/
38. read(client\_sockfd, &ch, 1);
39. printf("get char from client: %c\n", ch);
40. ++ch;
41. write(client\_sockfd, &ch, 1);
43. /\* close the socket \*/
44. close(client\_sockfd);
45. }
47. **return** 0;
48. }

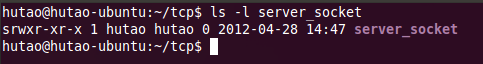
**tcp\_client.c：**

1. #include <sys/types.h>
2. #include <sys/socket.h>
3. #include <sys/un.h>
4. #include <unistd.h>
5. #include <stdlib.h>
6. #include <stdio.h>
8. **int** main()
9. {
10. /\* create a socket \*/
11. **int** sockfd = socket(AF\_UNIX, SOCK\_STREAM, 0);
13. **struct** sockaddr\_un address;
14. address.sun\_family = AF\_UNIX;
15. strcpy(address.sun\_path, "server\_socket");
17. /\* connect to the server \*/
18. **int** result = connect(sockfd, (**struct** sockaddr \*)&address, **sizeof**(address));
19. **if**(result == -1)
20. {
21. perror("connect failed: ");
22. exit(1);
23. }
25. /\* exchange data \*/
26. **char** ch = 'A';
27. write(sockfd, &ch, 1);
28. read(sockfd, &ch, 1);
29. printf("get char from server: %c\n", ch);
31. /\* close the socket \*/
32. close(sockfd);
34. **return** 0;
35. }

如果我们首先运行tcp\_client，会提示没有这个文件：

http://www.linuxidc.com/upload/2012_06/120602163842741.png

因为我们是**以AF\_UNIX方式进行通信的，这种方式是通过文件来将服务器和客户端连接起来的，因此我们应该先运行tcp\_server，创建这个文件，默认情况下，这个文件会创建在当前目录下，并且第一个s表示它是一个socket文件**：



程序运行的结果如下图：