# 域套接字

Unix域套接字只能用于在同一个计算机的进程间进行通信。虽然网络套接字也可以用于单机进程间的通信，但是使用Unix域套接字效率会更高，因为Unix域套接字仅仅进行数据复制，不会执行在网络协议栈中需要处理的添加、删除报文头、计算校验和、计算报文顺序等复杂操作，因而在单机的进程间通信中，更加推荐使用Unix域套接字。

也叫unix套接字.

所以网络套接字用的是ip,那么域套接字用的是文件路径

server代码

const SockAddr = "/Users/yangwenshuo/echo.sock"  
  
func echoServer(c net.Conn) {  
 log.Printf("Client connected [%s]", c.RemoteAddr().Network())  
 io.Copy(c, c)  
 c.Close()  
}  
  
func main() {  
 if err := os.RemoveAll(SockAddr); err != nil {  
 log.Fatal(err)  
 }  
  
 l, err := net.Listen("unix", SockAddr)  
 if err != nil {  
 log.Fatal("listen error:", err)  
 }  
 defer l.Close()  
  
 for {  
 *// 接收新连接, 并分配到echoServer函数处理* conn, err := l.Accept()  
 if err != nil {  
 log.Fatal("accept error:", err)  
 }  
  
 go echoServer(conn)  
 }  
}

client代码

package main  
  
import (  
 "io"  
 "log"  
 "net"  
 "time"  
)  
  
func reader(r io.Reader) {  
 buf := make([]byte, 1024)  
 n, err := r.Read(buf[:])  
 if err != nil {  
 return  
 }  
 println("Client got:", string(buf[0:n]))  
}  
  
func main() {  
 c, err := net.Dial("unix", "/Users/yangwenshuo/echo.sock")  
 if err != nil {  
 log.Fatal(err)  
 }  
 defer c.Close()  
  
 go reader(c)  
 \_, err = c.Write([]byte("hi"))  
 if err != nil {  
 log.Fatal("write error:", err)  
 }  
 reader(c)  
 time.Sleep(100 \* time.Millisecond)  
}