# socat Bidirectional byte relay

**START HERE**

**`socat - TCP:<host>:<port>`**
Talk to a TCP port

**`socat TCP-LISTEN:<port> -`**
Accept one connection

**`socat TCP-LISTEN:<port>,fork,reuseaddr -`**
Serve many clients

**`socat TCP-LISTEN:<lport>,fork TCP:<host>:<port>`**
Forward a port

**`socat UNIX-LISTEN:<path>,fork -`**
Listen on a Unix socket

**`socat - UNIX-CONNECT:<path>`**
Connect to a Unix socket

**`socat -u TCP-LISTEN:<port> CREATE:<file>`**
Receive a file

**`socat -d -d <addr1> <addr2>`**
See what it does

**Address 1** TYPE:args,options → *both ways* → **socat** relay loop → *both ways* → **Address 2** TYPE:args,options

## CONCEPTS

socat <addr1> <addr2> opens two endpoints and copies bytes both ways between them. Everything else is choosing the two addresses.

An address is TYPE:arg:arg,option,option=value. Options follow the commas; the type name is case-insensitive.

The first address that fails or hits EOF ends the relay. Add fork on a listener to keep serving, and shut-down or -t if the peer needs EOF.

## BASICS

**`socat <addr1> <addr2>`**
Relay between two addresses

**`socat - <addr>`**
Connect stdin/stdout to an address

**`socat -u <addr1> <addr2>`**
One way only: first read, second write

**`socat -d -d <addr1> <addr2>`**
Log more: -d notices, -d -d info

## BASICS (cont.)

**`socat -v <addr1> <addr2>`**
Dump transferred text to stderr

**`socat -x <addr1> <addr2>`**
Dump transferred data as hex to stderr

**`socat -T <secs> <addr1> <addr2>`**
Quit after this much inactivity

**`socat -V`**
Print version and built-in features

## TCP AND UDP

**`socat - TCP:<host>:<port>`**
Connect to a TCP port

**`socat TCP-LISTEN:<port> -`**
Listen for one TCP connection

**`socat TCP-LISTEN:<port>,fork,reuseaddr -`**
Serve each client in a child process

**`socat TCP-LISTEN:<port>,bind=<ip> -`**
Listen on one local address only

**`socat TCP-LISTEN:<port>,range=<cidr> -`**
Accept only peers in a network

**`socat - TCP4:<host>:<port>`**
Force IPv4 (TCP6 forces IPv6)

## TCP AND UDP (cont.)

**`socat TCP:<host>:<port>,connect-timeout=<s> -`**
Give up connecting after a while

**`socat - UDP:<host>:<port>`**
Send and receive UDP datagrams

**`socat UDP-LISTEN:<port>,fork -`**
Receive UDP datagrams, one peer per child

## UNIX SOCKETS AND FILES

**`socat UNIX-LISTEN:<path>,fork -`**
Listen on a Unix socket

**`socat - UNIX-CONNECT:<path>`**
Connect to a Unix socket

**`socat UNIX-LISTEN:<path>,mode=<mode> -`**
Set permissions of the socket file

**`socat UNIX-LISTEN:<path>,unlink-early -`**
Remove a stale socket file first

## UNIX SOCKETS AND FILES (cont.)

**`socat - GOPEN:<file>`**
Open a file or socket, whichever it is

**`socat - OPEN:<file>`**
Read an existing file

**`socat -u - CREATE:<file>`**
Write stdin to a new file

**`socat -u - OPEN:<file>,creat,append`**
Append stdin to a file

## FORWARD AND PROXY

**`socat TCP-LISTEN:<lport>,fork,reuseaddr TCP:<host>:<port>`**
Forward a local port to a remote one

**`socat TCP-LISTEN:<lport>,fork,bind=<ip> TCP:<host>:<port>`**
Forward, listening on one address only

**`socat UNIX-LISTEN:<path>,fork TCP:<host>:<port>`**
Expose a TCP service as a Unix socket

## FORWARD AND PROXY (cont.)

**`socat TCP-LISTEN:<port>,fork UNIX-CONNECT:<path>`**
Expose a Unix socket over TCP

**`socat -v TCP-LISTEN:<lport>,fork TCP:<host>:<port>`**
Forward and print the traffic

## TLS

**`socat - OPENSSL:<host>:<port>`**
Connect over TLS (verifies the server)

**`socat - OPENSSL:<host>:<port>,verify=0`**
Connect over TLS, skip verification

**`socat - OPENSSL:<host>:<p>,cafile=<c>`**
Verify against a given CA file

**`socat OPENSSL-LISTEN:<p>,cert=<f>,verify=0 -`**
TLS server, no client cert needed

*Without verify=0 the server demands a client certificate. <f> must hold key and cert, or add key=<key>.*

## PROGRAMS AND TERMINALS

**`socat TCP-LISTEN:<port>,fork EXEC:<prog>`**
Run a program per connection

**`socat TCP-LISTEN:<port>,fork SYSTEM:'<shell cmd>'`**
Run a shell command per connection

**`socat - EXEC:<prog>,pty,stderr`**
Run a program on a pty with stderr merged

**`socat -,raw,echo=0 EXEC:'bash -i',pty,stderr,setsid`**
Full interactive shell over a pty

## RECIPES

**Send a file between hosts**

```
# receiver
socat -u TCP-LISTEN:5000
CREATE:in.tar
# sender
socat -u OPEN:in.tar
TCP:<host>:5000
```

**Watch traffic through a forward**

```
socat -v TCP-LISTEN:8080,fork \
  TCP:example.com:80
# other shell
curl localhost:8080/
```