Session NM058

TCP Programming Example

Joel Snyder
Opus1

Course Roadmap

- NM055 (11:00-12:00) Important Terms and Concepts
  - TCP/IP and Client/Server Model
  - Sockets and TLI
  - Client/Server in TCP/IP
- NM056 (1:00-2:00) Socket Routines
- NM057 (2:00-3:00) Library Routines
- NM058 (3:00-4:00) Sample Client/Server
- NM059 (4:00-5:00) VMS specifics (QIOs)
- NM067 (6:00-7:00) Clinic - Q&A

TCP/IP Programming

Slides and Source Code available via anonymous FTP:

Host:: ftp.process.com
Directory: [pub.decus]
Slides: DECUS_F96_PROG.PS
Examples: DECUS_F96_PROG_EXAMPLES.TXT

Host: ftp.opus1.com
Slides: DECUS_F96_PROG.PS
Examples: DECUS_F96_PROG_EXAMPLES.TXT

TCP Server Example

- See handout

TCP Client Example

- See handout

UDP Programming Example

UDP Programming Example

- We'll create a datagram socket
- We'll fill in the important structures
- We'll explicitly bind()

UDP Example - socket()

- (1) Create an Internet Datagram socket (UDP)

```
/*
 * Create an IP family socket on which to make the connection
 */

s=socket(hp->h_addrtype, SOCK_DGRAM, 0);
if(s<0){
   socket_perror("udpechoclient: socket");
   exit(0x10000000);
}
```

UDP Example - sockaddr

- (2) Fill in socket address structure

```
sp=getservbyname(“echo”,”udp”);
if(sp==NULL){
    fprintf(stderr,”udpechoclient: echo/udp: unknown service\n”);
    exit(0x10000000);
}
/*
 *  Creat a “sockaddr_in” structure which describes the remote
 *  IP address we want to sent to (from gethostbyname()) and
 *  the remote UDP port number (from getservbyname()).
 */
sin.sin_family = hp->h_addrtype;
memcpy(&sin.sin_addr, hp->h_addr, hp->h_length);
sin.sin_port=sp->s_port;
```

UDP Example - sin.sin_addr

- (3) Get address, parse, insert
  - Protocol independent, same as TCP

UDP Example - bind()

- (4) Explicit bind, implicit connect

```
/*
 * Do a psuedo-connect to that address.  This tells the kernel
 * that anything on this socket gets sent to this destination.
 * It also binds us to a local port number (random, but that
 * is ok).
 */
n=connect(s,&sin,sizeof(sin));
if(n<0){
   socket_perron("udpechoclient:connect");
   exit(0x10000004);
}
```

UDP Example - socket_write()

- (5) Write data out
  - Protocol independent, just like TCP

UDP Example - socket_read()

- (6) Read data in
  - Protocol independent, just like TCP

Practical Troubleshooting

Troubleshooting Tips

- Network Byte Order
- Watch out for the Terminator
- Get return status from library routines

Tips - Network Byte Order

- "Big-endian" vs. "Little-endian" hosts
  - VMS systems are little-endian
  - TCP/IP uses big-endian order
- htonl(), htons() convert host to network byte order
- ntohl() and ntohs() are network to host conversions

Tips - Terminators

- TCP is a *stream-oriented* protocol
- No such thing as a “record terminator”
- You have to “roll your own”
  - <CR><LF> is common for ASCII text protocols
  - 4 byte message length followed by message
  - Fixed length message (rarely used)

Use return status from library routines

- Well, it's there for a reason
- It might tell you something
- Failure usually means the opposite of success
  - Typically it's a data structure error
  - make sure using perror() that it's not a network problem

Troubleshooting Tools

- TCPDUMP displays packets
- ECHO and DISCARD ports aid in debugging
- `$ UCX SHOW parameter`
- `$ NETCU SHOW parameter (TCPware)`
- `$ MULTINET SHOW/` *`your_qualifier_here`*

Using TCPDUMP

- Limit your port
  - `$ multinet tcpdump/port=nn`
- Limit your hosts
  - `$ multinet tcpdump src hst my.host.here dst hst remote.host`
- Look at the inside of the packet
  - `$ multinet tcpdump/snapshot=1500/hexadecimal`

TCPware's NETCU DEBUG

- NETCU DEBUG/ (UDP,TCP,IP)
- Limit your port
  - `$NETCU DEBUG/TCP/LPN=nn/FPN=nn`
- Limit your hosts
  - `$NETCU DEBUG/IP/SIA=host/DIA=x.x.x.x`
- Look at the packet headers
  - `$NETCU DEBUG/HEADER`
- Look at the inside of the packet
  - `$NETCU DEBUG/UDP/DATA=N`

Using the ECHO/DISCARD services

- The ``echo'' service returns what you send it
- The ``discard'' service eats what you send it
- Start a TCPDUMP session
- Have your clients talk to these ports
- Have a TELNET client talk to your server
- You can debug UDP by writing TCP code and modifying it to use datagrams

TCPware $ NETCU SHOW/EVERYTHING

- Network connection table
- Route table
- Services
- Etc.

TCPware Network Connections

- Use `$ NETCU show connection`
- ESTABLISHED shows for successful TCP connections
- TIME_WAIT indicates remote side closed connection
- CLOSE_WAIT indicates local side closed connection
- See RFC-793 TCP state machine for more details

TCPware Network Route Table

- Use `$ NETCU show route(/FULL)`
- If host routes with MTUs different from the interface form, MTU discovery may be in effect
- Use NETCU/FULL to see Path MTU and network mask values
- Variable Length Subnets supported

$ MULTINET SHOW/EVERYTHING

- Network connection table
- Route table
- MTU Discovery

Network Connection Table

- Use `$multinet show/connection`
- ESTABLISHED shows for successful TCP connections
- TIME_WAIT indicates remote side closed connection
- CLOSE_WAIT indicates local side closed connection
- See RFC-793 TCP state machine for more details

Network Route Table

- Use `$ multinet show/route`
- If host routes with MTUs different from the interface form, MTU discovery may be in effect
  - stop MTU discovery
    - `$ multinet set/kernel tcp_do_mtu_discovery 0` (3.3B and up)
  - delete the route
    - `$ mu set/rout/delet=(dest:a.b.c.d,gate:a.b.c.d)`
- Create a host route and watch the usage grow!

$ UCX SHOW/EVERYTHING

- Network connection table
- Route table
- Device Sockets

TCP/IP Programming
Key Concepts

- Use examples from similar applications that work
  - “Steal” from the best
- Suggested resources
  - Curry, Donald A., *Using C on the UNIX System,* O’Reilly and Associates
  - Comer, Douglas *Internetworking with TCP/IP, Volume III,* Prentice-Hall
  - DEC C Run-Time Library Reference Manual (May, 1995 or later, Appendix A)
  - MultiNet Programmer’s Guide