
CS3516 Project 3

Help Session

(A15)

Dongqing Xiao
Sep 30th, 2015

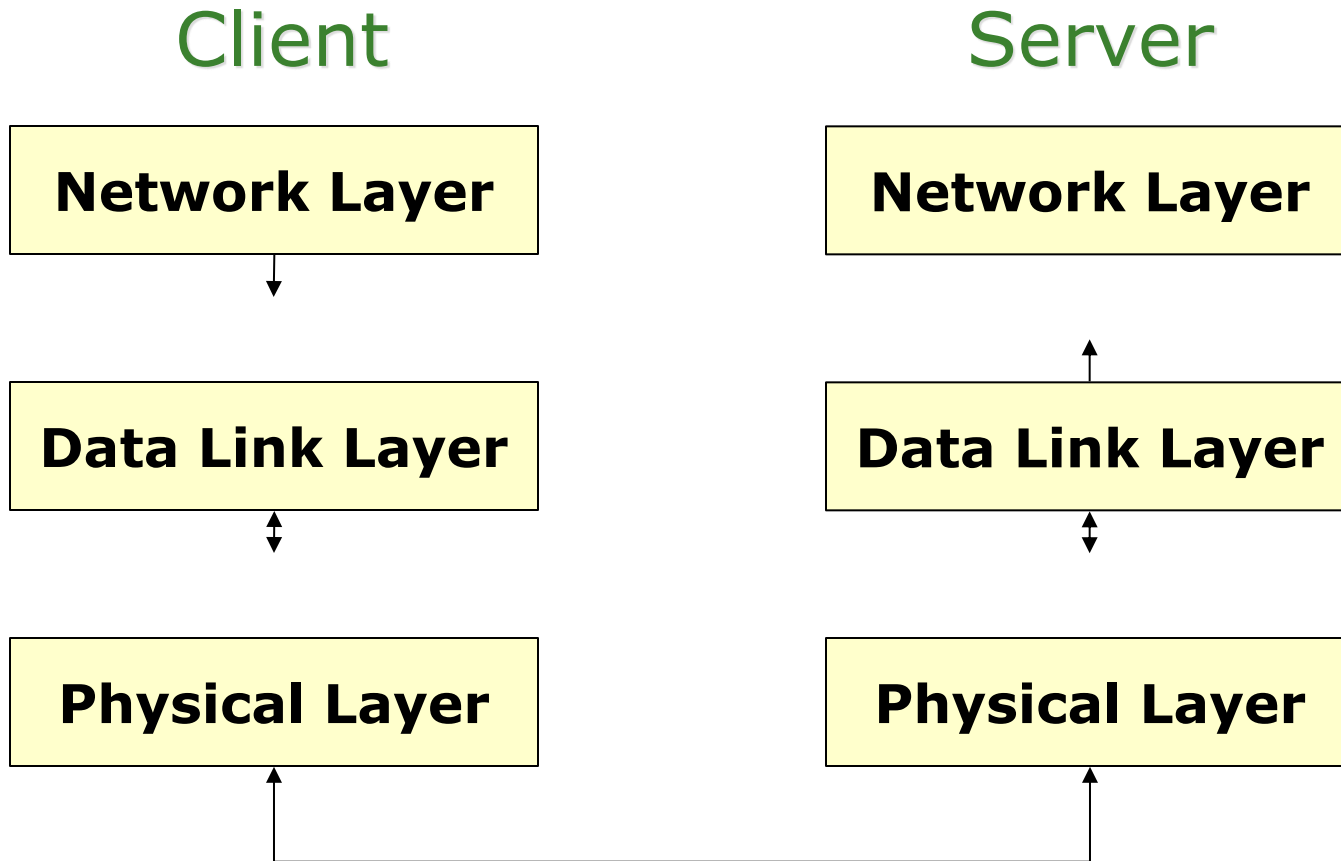
Outline

- Framework
- Data Structure
 - Packet
 - Frame
- Implementation
 - Timer
 - File IO /image related
 - Fork
 - XOR: Error Detection

Description

- The goal is to implement a Positive Acknowledgement with Retransmission (PAR) protocol on top of an emulated physical layer.
 - The receiver acknowledges only the correctly received segments
 - The sender uses timeout to detect and send the lost segment.
 - Physical layer is emulated by a TCP connection plus an error module.
 - Your programs should compile and work on ["ccc.wpi.edu"](http://ccc.wpi.edu)

Framework



Do NOT attempt to put everything in one big main()

Network Layer

Client

Photo[i].jpg

read (block)

Network Layer

nwl_rcv(ack)

Network layer
ACK

pkt

Server

Photonew[i].jpg

write (block)

Network Layer

nwl_rcv(pkt)

Network layer
ACK

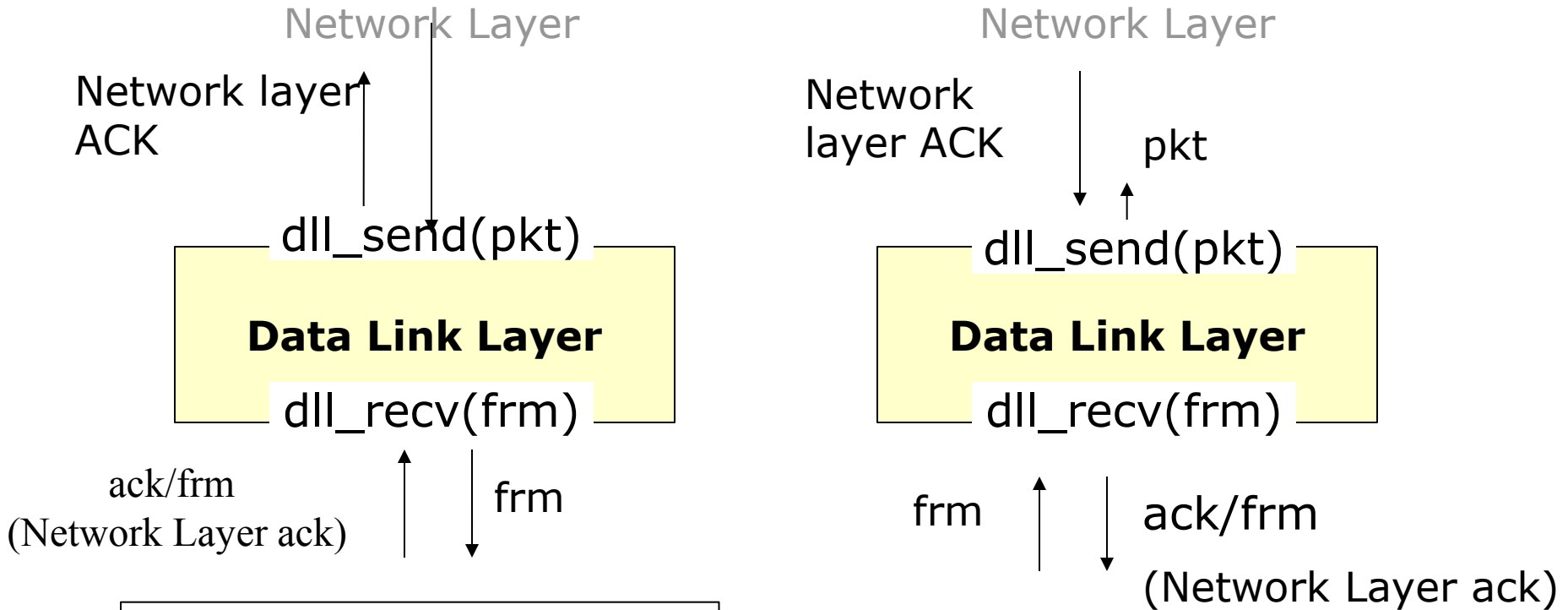
pkt

Packet payload: 256 bytes

Data Link Layer

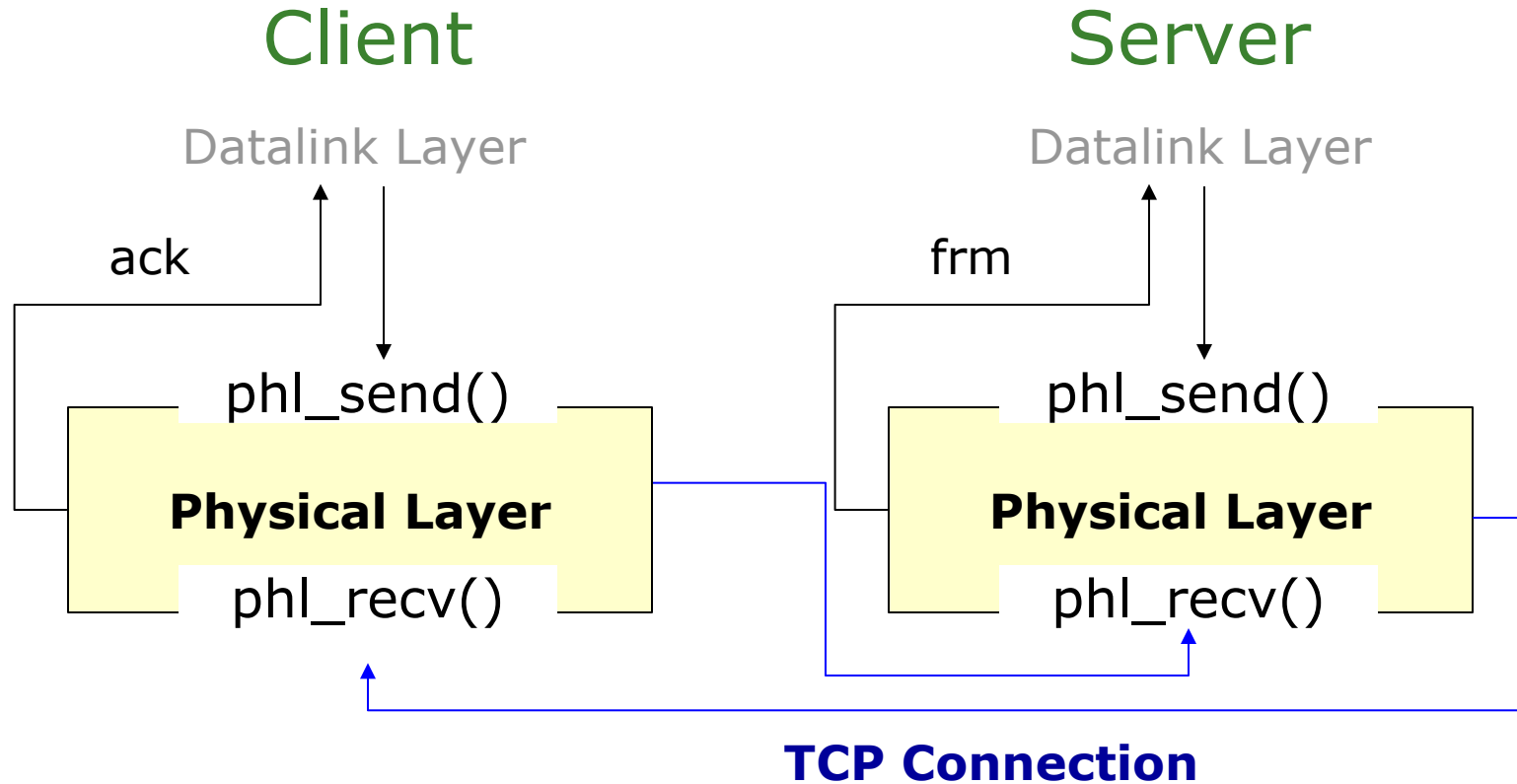
Client

Server



Client link layer does not need to ACK "Network Layer ACK" frame!

Physical Layer



Client: dll_send(pkt, ...)

1. Read a block and split into payloads

phl_send(frm, ...):

2. For each payload

2.1 Create Frame (frm)

2.2 Start a Timer

2.2.1 Timeout Handler:
phl_send(frm, ...)

2.3 phl_send(frm, ...)

2.4 phl_rcv(ack/frm,...)

ack/frm?

2.4.1 nwl_ack received

ack

ack ok?

no

yes

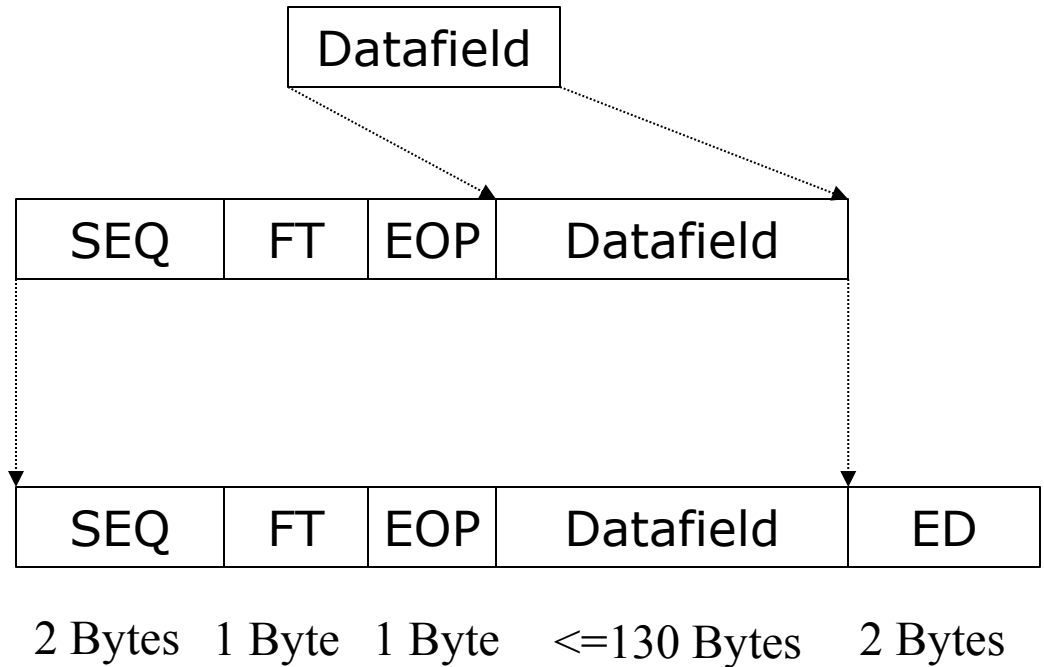
client.log

3. Waiting for Network Layer ack:
if (!nwl_ack) phl_rcv(frm,...)

Create Frame

1. Compute SEQ Number, Frame Type and End-Of-Packet (EOP) bytes

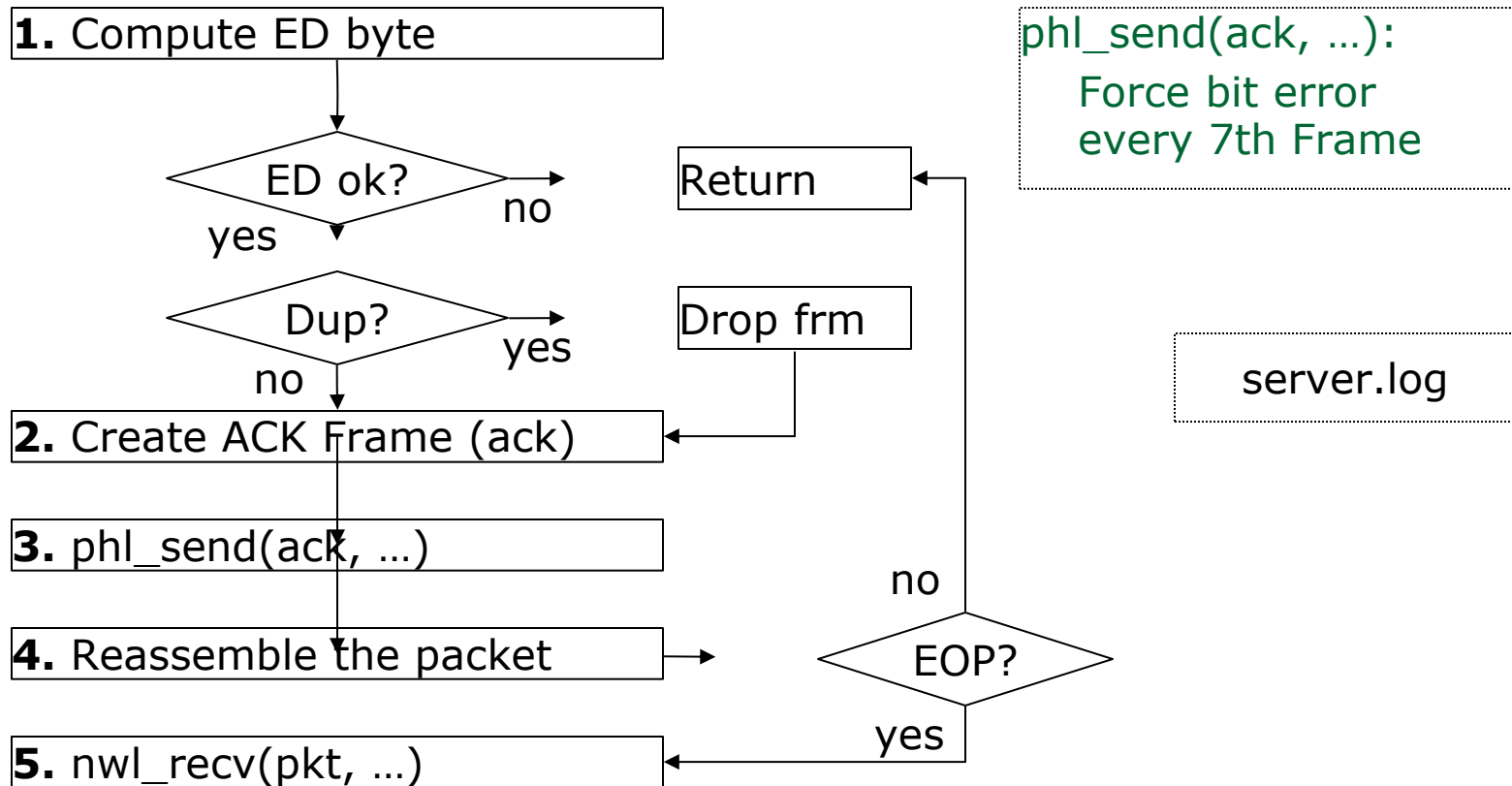
2. Error-Detection (ED) bytes
(XOR on SEQ + FT + EOP + Data)



EOP: End of Packet
FT: Frame Type

ED: Error Detection
SEQ: Sequence Num

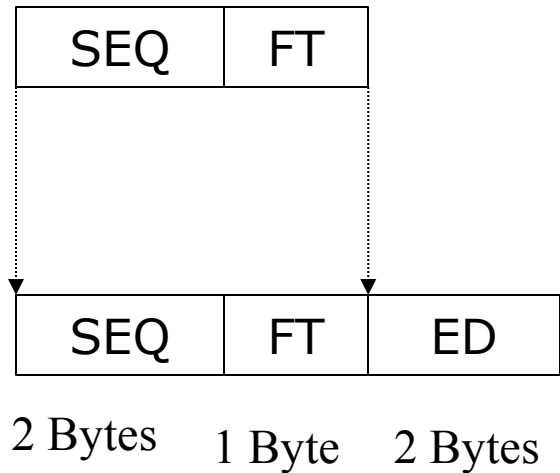
Server: dll_rcv(frm, ...)



Create ACK Frame

1. Compute SEQ Number and Frame Type

2. Error-Detection (ED) bytes (ED = SEQ)



EOP: End of Packet
FT: Frame Type

ED: Error Detection
SEQ: Sequence Num

Timers

- The client uses a timer to detect a frame loss.
 - The client sets a timer when it transmits a frame.
 - When the timer expires, the client retransmits the frame.

- Two kinds of timer
 - Select : easier to use
 - Signal and Timer : nicer implementation

Select: Monitor Given FDs (SDs)

```
# include <sys/select.h>
# include <sys/time.h>
```

```
int select(int nfd, fd_set *readfds, fd_set *writefds, fd_set
           *exceptfds, struct timeval *timeout);
```

```
struct timeval {
    long tv_sec;           /* seconds */
    long tv_usec;        /* microseconds */
}
```

Example: Select

```
fd_set bvfdRead;
int readyNo;
struct timeval timeout;
int sockfd;

while (1) {
    timeout.tv_sec = 0;
    timeout.tv_usec = 500;
    FD_ZERO(&bvfdRead);
    FD_SET(sockfd, &bvfdRead);
```

```
    readyNo = select(sockfd+1,
                    &bvfdRead, 0, 0, &timeout);

    if(readyNo < 0)
        error_handler();
    else if(readyNo == 0)
        timeout_handler();
    else {
        FD_ZERO(&bvfdRead);
        receive_handler();
    }
}
```

Signal and Timer: Soft Interrupt

- Head files

```
#include <signal.h>
```

```
#include <sys/time.h>
```

- Register a function to TIMEOUT signal

```
int sigaction(int signum, const struct sigaction *act,  
              struct sigaction *oldact);
```

- Create a timer and begin to run

```
int setitimer(int which, const struct itimerval  
*new_value, struct itimerval *old_value);
```

Example: Signal and Timer

```
#include <signal.h>
#include <stdio.h>
#include <string.h>
#include <sys/time.h>
```

```
void timer_handler (int signum) {
    printf ("timer expired for %d",
           signum);
    exit(0);
}
```

```
int main ()
{
    struct sigaction sa;
    struct itimerval timer;
```

```
    /* Install timer_handler as the signal handler for
       SIGVTALRM. */
    memset (&sa, 0, sizeof (sa));
    sa.sa_handler = &timer_handler;
    sigaction (SIGVTALRM, &sa, NULL);
```

```
    /* Configure the timer to expire after 250 ms*/
    timer.it_value.tv_sec = 0;
    timer.it_value.tv_usec = 250000;
    timer.it_interval.tv_sec = 0;
    timer.it_interval.tv_usec = 0;
```

```
    setitimer (ITIMER_VIRTUAL, &timer, NULL);
```

```
    /* Do busy work. */
    while (1);
    return 1;
```

Toy Example

Run: /Users/dongqingxiao/cEX/Timer

Code: <https://github.com/dqxiao/cs3516ToyExample>

Open a File

- Open a file for read:

```
int rfile;
if ((rfile = open("filename1", O_RDONLY)) < 0)
{
    perror("Input File Open Error");
    exit(1);
}
```

- Open a file for write (create if not exist):

```
int ofile;
if ((ofile = open("filename2", O_WRONLY|O_CREAT|O_TRUNC,
S_IRUSR|S_IWUSR|S_IRGRP|S_IWGRP)) < 0)
{
    perror("Output File Open Error");
    exit(1);
}
```

File Read

- Read from file

```
while ((rd_size = read(rfile, buf, 256)) > 0)
    {
        do something with "buf" here
    }
if (rd_size < 0)
    {
        perror("File Read Error");
        exit(1);
    }
else
    {
        printf ("Reach the end of the file\n");
    }
```

File Write/Close

- Write to File

```
if ((wr_size = write(ofile, buf, rd_size)) < 0)
{
    perror("Write Error:");
    exit(1);
}
```

- Close files

```
close(rfile);
close(ofile);
```

Display Image in Linux

- Make sure you have “X forwarding” with your ssh client
- And you need have an Xserver (X-Win32 or etc.) running on you windows computer.
- The image display is not required for the Project.
- These code tested on ccc[1-10].wpi.edu

```
if (fork() == 0)
{
    execl("/usr/local/bin/xv", "xv", "image.jpg", NULL);
}
else
{
    wait(NULL);
    printf("Done display! \n");
}
```

Multiple Clients—fork

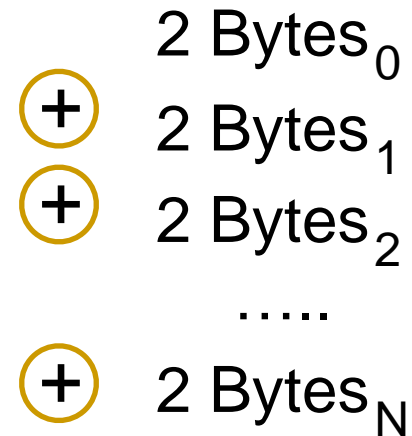
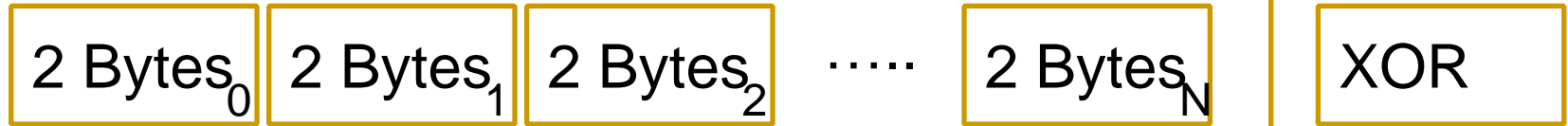
```
for (;;)
{
    clilen = sizeof(cliaddr);
    connfd = accept(listenfd, (SA *) &cliaddr, &clilen);

    if ( (childpid = fork()) == 0) { /* child process */
        close(listenfd); /* close listening socket */
        str_echo_dq(connfd); /* process the request */
        exit(0);
    }

    close(connfd); /* parent closes connected socket */
    exit(0);
}
```

http://www.gnu.org/software/libc/manual/html_node/Creating-a-Process.html

XOR: Error Detection



XOR

https://en.wikipedia.org/wiki/Error_detection_and_correction

Thanks!
and
Questions?