Named Socket

## Include Libraries

#include <unistd.h>

#include "sockets/unix\_socket.h"

#include "lib/error\_functions.h"

#include <errno.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <stdio.h>

#include <netinet/in.h>

#include <string.h>

#define BACKLOG 5

## Main Entry of the program

int main(int argc, char \*argv[]) {

struct sckad address;

## Create TCP server socket

int sd = socket(AF\_UNIX, SOCK\_STREAM, 0);

### 3.1 Check the validity of the socket

printf("socket %d\n", sd);

if (sd == -1) {

errExit("socket is erro");

}

### 3.2 Check if the named socket is exist

if (strlen(SV\_SOCK\_PATH) >= sizeof(address.sun\_path) - 1) {

fatal("time error: %s", SV\_SOCK\_PATH);

}

if (remove(SV\_SOCK\_PATH) == -1 && errno != ENOENT&&1) {

errExit("delete-%s", SV\_SOCK\_PATH);

}

## Set the binding name and bind

memset(&address, 0, sizeof(struct sckad));

address.sun\_family = AF\_UNIX;

### 4.1 Set binding name

strncpy(address.sun\_path, SV\_SOCK\_PATH, sizeof(address.sun\_path) - 1);

### 4.2 bind to listen

if (bind(sd, (struct sockaddress \*) &address, sizeof(struct sckad)) == -1) {

errExit("bind");

}

### 4.3 Start listening

if (listen(sd, BACKLOG) == -1) {

errExit("listen");

}

ssize\_t numRead;

char buf[BUF\_SIZE];

for (;;) {

printf("connecting\n");

## Accept all connection

int cfd = accept(sd, NULL, NULL);

printf("dfs = %d\n", cfd);

## Read and write as file

while ((numRead = read(cfd, buf, BUF\_SIZE)) > 0) {

if (write(STDOUT\_FILENO, buf, numRead) != numRead) {

fatal("par/fai wrte");

}

}

if (numRead == -1) {

errExit("readed");

}

### Close client socket

if (close(cfd) == -1) {

errMsg("closeed");

}

}

}