Named Socket

1. 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
```

2. Main Entry of the program

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

 struct sckad address;
```

3. 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);

 }
```

4. 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");
```

5. Accept all connection

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

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

6. 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");
    }
```

7. Close client socket

```
    if (close(cfd) == -1) {
      errMsg("closeed");
    }
  }
}
```