ISL

Network Programming #4

ISL (IoT Standard Lab)

Index

gethostbyname()

```c
int main(int argc, char** argv) {
    struct hostent *host, *host2;
    struct sockaddr_in addr;

    if(argc < 2) {
        printf("Usage: %s <DomainName> <RemoteAddress>\n", argv[0]);
        return 1;
    }

    host = gethostbyname(argv[1]);
    if(!host) {
        perror("gethostbyname() error");
        return 1;
    }

    printf("gethostbyname()\n");
    printf("Official name: %s \n", host->h_name);
    for(int i=0; host->h_aliases[i]; i++) {
        printf("Aliases %d: %s \n", i, host->h_aliases[i]);
    }

    printf("Address type: %s \n", (host->h_addrtype==AF_INET)? "AF_INET" : "AF_INET6");
    for(int i=0; host->h_addr_list[i]; i++) {
        printf("IP addr %d: %s \n", i, inet_ntoa(*(struct in_addr*)host->h_addr_list[i]));
    }
```

1. argv[1]로 DNS 요청을 하고 결과를 hostent 구조체에 저장

```c
#include <netdb.h>

struct hostent * gethostbyname(const char * hostname);
```

➜ 성공 시 hostent 구조체 변수의 주소 값, 실패 시 NULL 포인터 반환

01 DNS (Domain Name System)

gethostbyname()

```c
int main(int argc, char** argv) {
    struct hostent *host, *host2;
    struct sockaddr_in addr;

    if(argc < 2) {
        printf("Usage: %s <DomainName> <RemoteAddress>\n", argv[0]);
        return 1;
    }

    host = gethostbyname(argv[1]);
    if(!host) {
        perror("gethostbyname() error");
        return 1;
    }

    printf("gethostbyname()\n");
    printf("Official name: %s \n", host->h_name);
    for(int i=0; host->h_aliases[i]; i++) {
        printf("Aliases %d: %s \n", i, host->h_aliases[i]);
    }

    printf("Address type: %s \n", (host->h_addrtype==AF_INET)? "AF_INET" : "AF_INET6");
    for(int i=0; host->h_addr_list[i]; i++) {
        printf("IP addr %d: %s \n", i, inet_ntoa(*(struct in_addr*)host->h_addr_list[i]));
    }
```

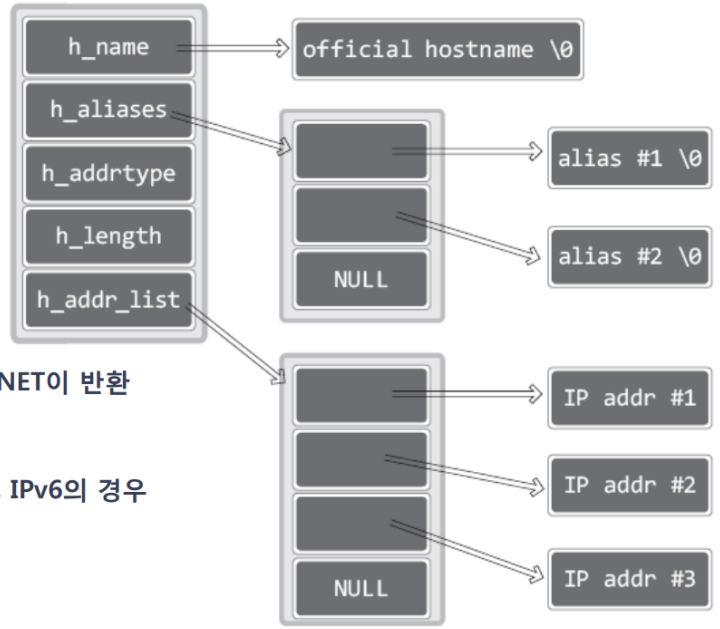
h_name :
공식 도메인 이름

h_aliases :
별칭의 도메인 이름

h_addrtype :
반환된 IP의 정보가 IPv4인 경우, AF_INET이 반환

h_length :
반환된 IP 정보의 크기, IPv4의 경우 4, IPv6의 경우 16이 저장

h_addr_list
IP의 주소정보, 둘 이상의 경우 모두 반환



gethostbyaddr()

```c
memset(&addr, 0, sizeof(addr));
addr.sin_addr.s_addr = inet_addr(argv[2]);
host = gethostbyaddr((char*)&addr.sin_addr, 4, AF_INET);
if(!host) {
    perror("gethostbyaddr() error");
    return 1;
}

printf("\ngethostbyaddr()\n");
printf("Official name: %s \n", host->h_name);
for(int i=0; host->h_aliases[i]; i++) {
    printf("Aliases %d: %s \n", i, host->h_aliases[i]);
}

printf("Address type: %s \n", (host->h_addrtype==AF_INET)? "AF_INET" : "AF_INET6");
for(int i=0; host->h_addr_list[i]; i++) {
    printf("IP addr %d: %s \n", i, inet_ntoa(*(struct in_addr*)host->h_addr_list[i]));
}


memset(&addr, 0, sizeof(addr));
addr.sin_addr.s_addr = inet_addr(argv[2]);
host = gethostbyaddr((char*)&addr.sin_addr, 4, AF_INET);
if(!host) {
    perror("gethostbyaddr() error");
    return 1;
}
```

1. inet_addr(argv[2])로 DNS 요청을 하고 결과를 hostent 구조체에 저장

```c
#include <netdb.h>

struct hostent * gethostbyaddr(const char * addr, socklen_t len, int family);
```

➜ 성공 시 hostent 구조체 변수의 주소 값, 실패 시 NULL 포인터 반환

- addr — IP주소를 지니는 in_addr 구조체 변수의 포인터 전달, IPv4 이외의 다양한 정보를 전달받을 수 있도록 일반화하기 위해서 매개변수를 char형 포인터로 선언.
- len — 첫 번째 인자로 전달된 주소정보의 길이, IPv4의 경우 4, IPv6의 경우 16 전달.
- family — 주소체계 정보 전달. IPv4의 경우 AF_INET, IPv6의 경우 AF_INET6 전달.

01 DNS (Domain Name System)

실행 결과

```
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/dns$ ./hostaddr google.com 8.8.8.8
gethostbyname()
Official name: google.com
Address type: AF_INET
IP addr 0: 172.217.175.238

gethostbyaddr()
Official name: dns.google
Address type: AF_INET
IP addr 0: 8.8.8.8
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/dns$
```

getsockopt()

옵션정보의 참조에 사용되는 함수


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

int getsockopt(int sock, int level, int optname, void *optval, socklen_t *optlen);
```

➜ 성공 시 0, 실패 시 -1 반환

- sock — 옵션확인을 위한 소켓의 파일 디스크립터 전달.
- level — 확인할 옵션의 프로토콜 레벨 전달.
- optname — 확인할 옵션의 이름 전달.
- optval — 확인결과의 저장을 위한 버퍼의 주소 값 전달.
- optlen — 네 번째 매개변수 optval로 전달된 주소 값의 버퍼크기를 담고 있는 변수의 주소 값 전달, 함수호출이 완료되면 이 변수에는 네 번째 인자를 통해 반환된 옵션정보의 크기가 바이트 단위로 계산되어 저장된다.

앞서 표에서 제시한 Protocol Level과 Option Name이 두 번째, 세 번째 인자로 전달되어, 해당 옵션의 등록 정보를 얻어온다.

02 Socket option

getsockopt() : SO_TYPE

```c
int main(int argc, char** argv) {
    int tcp_sock, udp_sock;
    int sock_type;
    socklen_t optlen;
    int state;

    tcp_sock = socket(AF_INET, SOCK_STREAM, 0);
    udp_sock = socket(AF_INET, SOCK_DGRAM, 0);

    printf("SOCK_STREAM: %d \n", SOCK_STREAM);
    printf("SOCK_DGRAM: %d \n", SOCK_DGRAM);

    optlen = sizeof(sock_type);
    state = getsockopt(tcp_sock, SOL_SOCKET, SO_TYPE, (void*)&sock_type, &optlen);
    if(state) {
        perror("getsockopt() error");
        return 1;
    }

    printf("Socket type one: %d \n", sock_type);

    state = getsockopt(udp_sock, SOL_SOCKET, SO_TYPE, (void*)&sock_type, &optlen);
    if(state) {
        perror("getsockopt() error");
        return 1;
    }

    printf("Socket type two: %d \n", sock_type);

    close(tcp_sock);
    close(udp_sock);
    return 0;
}
```

```
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/socket_option$ ./socktype
SOCK_STREAM: 1
SOCK_DGRAM: 2
Socket type one: 1
Socket type two: 2
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/socket_option$
```

Protocol Level	Option Name	Get	Set
SOL_SOCKET	SO_SNDBUF	O	O
	SO_RCVBUF	O	O
	SO_REUSEADDR	O	O
	SO_KEEPALIVE	O	O
	SO_BROADCAST	O	O
	SO_DONTROUTE	O	O
	SO_OOBINLINE	O	O
	SO_ERROR	O	X
	SO_TYPE	O	X
IPPROTO_IP	IP_TOS	O	O
	IP_TTL	O	O
	IP_MULTICAST_TTL	O	O
	IP_MULTICAST_LOOP	O	O
	IP_MULTICAST_IF	O	O
IPPROTO_TCP	TCP_KEEPALIVE	O	O
	TCP_NODELAY	O	O
	TCP_MAXSEG	O	O

02 Socket option

getsockopt() : SO_SNDBUF, SO_RCVBUF

```c
int main(int argc, char** argv) {
    int sock;
    int snd_buf, rcv_buf;
    socklen_t len;
    int state;

    sock = socket(AF_INET, SOCK_STREAM, 0);

    len = sizeof(snd_buf);
    state = getsockopt(sock, SOL_SOCKET, SO_SNDBUF, (void*)&snd_buf, &len);
    if(state) {
        perror("getsockopt() error");
        return 1;
    }

    len = sizeof(rcv_buf);
    state = getsockopt(sock, SOL_SOCKET, SO_RCVBUF, (void*)&rcv_buf, &len);
    if(state) {
        perror("getsockopt() error");
        return 1;
    }


    printf("Input buffer size: %d \n", rcv_buf);
    printf("Output buffer size: %d \n", snd_buf);

    close(sock);
    return 0;
}
```

```
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/socket_option$ ./getbuf
Input buffer size: 131072
Output buffer size: 16384
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/socket_option$
```

Protocol Level	Option Name	Get	Set
SOL_SOCKET	SO_SNDBUF	O	O
	SO_RCVBUF	O	O
	SO_REUSEADDR	O	O
	SO_KEEPALIVE	O	O
	SO_BROADCAST	O	O
	SO_DONTROUTE	O	O
	SO_OOBINLINE	O	O
	SO_ERROR	O	X
	SO_TYPE	O	X
IPPROTO_IP	IP_TOS	O	O
	IP_TTL	O	O
	IP_MULTICAST_TTL	O	O
	IP_MULTICAST_LOOP	O	O
	IP_MULTICAST_IF	O	O
IPPROTO_TCP	TCP_KEEPALIVE	O	O
	TCP_NODELAY	O	O
	TCP_MAXSEG	O	O

02 Socket option

setsockopt() : SO_SNDBUF, SO_RCVBUF

```c
int sock;
int snd_buf, rcv_buf;
socklen_t len;
int state;

snd_buf = rcv_buf = 1024*3;

sock = socket(AF_INET, SOCK_STREAM, 0);
state = setsockopt(sock, SOL_SOCKET, SO_RCVBUF, (void*)&rcv_buf, sizeof(rcv_buf));
if(state) {
    perror("setsockopt() error");
    return 1;
}
state = setsockopt(sock, SOL_SOCKET, SO_SNDBUF, (void*)&snd_buf, sizeof(snd_buf));
if(state) {
    perror("setsockopt() error");
    return 1;
}

len = sizeof(snd_buf);
state = getsockopt(sock, SOL_SOCKET, SO_SNDBUF, (void*)&snd_buf, &len);
if(state) {
    perror("getsockopt() error");
    return 1;
}

len = sizeof(rcv_buf);
state = getsockopt(sock, SOL_SOCKET, SO_RCVBUF, (void*)&rcv_buf, &len);
if(state) {
    perror("getsockopt() error");
    return 1;
}


printf("Input buffer size: %d \n", rcv_buf);
printf("Output buffer size: %d \n", snd_buf);

close(sock);
return 0;
```

1. 설정한 값 그대로 버퍼 크기가 결정되진 않고 커널 정책에 따라 결정됨

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/socket_option$ ./getbuf
Input buffer size: 131072
Output buffer size: 16384
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/socket_option$ ./setbuf
Input buffer size: 6144
Output buffer size: 6144
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/socket_option$
```

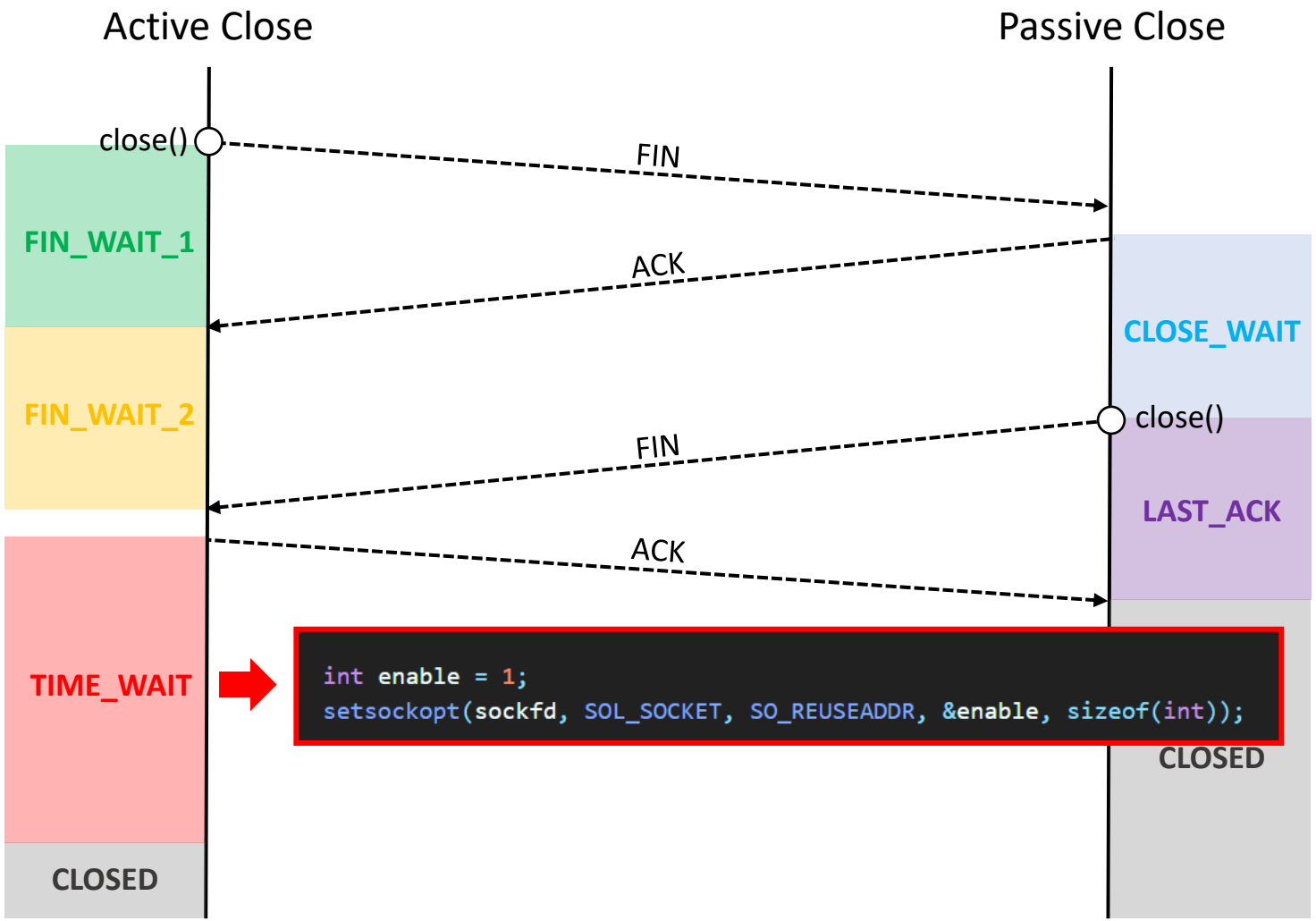
```c
#include <sys/socket.h>

int setsockopt(int sock, int level, int optname, const void *optval, socklen_t optlen);
```

➜ 성공 시 0, 실패 시 -1 반환

- sock — 옵션변경을 위한 소켓의 파일 디스크립터 전달.
- level — 변경할 옵션의 프로토콜 레벨 전달.
- optname — 변경할 옵션의 이름 전달.
- optval — 변경할 옵션정보를 저장한 버퍼의 주소 값 전달.
- optlen — 네 번째 매개변수 optval로 전달된 옵션정보의 바이트 단위 크기 전달.

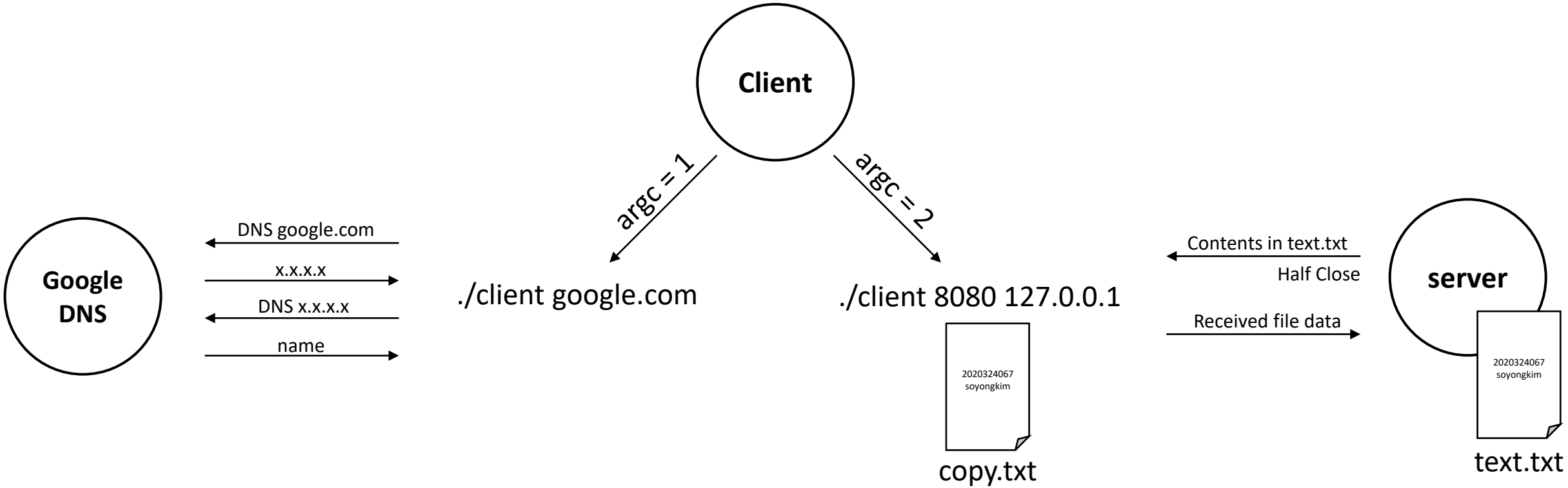
앞서 표에서 제시한 Protocol Level과 Option Name 이 두 번째, 세 번째 인자로 전달하고, 해당 옵션의 등록정보를 변경한다.

setsockopt() : SO_REUSEADDR



03 Network assignment #4

DNS & Half Close Practice



03 Network assignment #4

DNS & Half Close Practice

- **client.c**

1. 클라이언트 실행 시 main 함수의 매개변수의 개수로 실행 분기 정함
    1. 매개 변수로 도메인이름 하나만 입력받는 경우

        Ex) ./client google.com

    2. 매개 변수로 포트 번호와 IP 주소를 입력받는 경우

        Ex) ./client 8080 127.0.0.1 (순서 확인)
2. 도메인 이름만 받는 경우 gethostbyname()을 통해 해당 도메인에 대한 정보를 받아 표준 출력하고, 여기서 얻은 첫 번째 IP 정보를 이용하여 gethostbyaddr()을 통해 다시 이에 대한 정보를 얻고 다시 표준 출력 (google.com으로 정상 동작 확인)
    1. 출력 양식은 다음과 같음

        <gethostbyname() or gethostbyaddr()>
        Official name: <공식 도메인 이름>
        Aliases <index>: <별칭의 도메인 이름>
        …

        Address type: <AF_INET or AF_INET6>
        IP addr <index>: <IP 주소정보>
        …

```
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/assignment$ ./client google.com
gethostbyname()
Official name: google.com
Address type: AF_INET
IP addr 0: 172.217.31.142

gethostbyaddr()
Official name: nrt20s08-in-f14.1e100.net
Address type: AF_INET
IP addr 0: 172.217.31.142
```

03 Network assignment #4

DNS & Half Close Practice

- **client.c**

1. 매개 변수로 포트번호와 IP 주소를 받는 경우
    1. TCP소켓을 생성하고 getsockopt()를 통해 소켓의 타입 정보를 받고 STREAM 타입과 같이 아래처럼 표준 출력

        Ex) This socket type is : getsockopt()에서 얻은 정보 출력(SOCK_STREAM 출력)

        Ex) This sock type is : 1(1)
    2. 서버로부터 받은 파일 데이터를 자신의 파일로 저장하기 위해 fopen()으로 파일을 오픈
        1. 파일명은 copy.txt임(파일이 없을 경우 생성되도록 구성)
        2. 읽고 쓸 수 있는 권한을 줌
    3. 매개 변수로 받은 정보로 서버에게 연결 요청
    4. 서버에게 파일 데이터를 모두 받고 Received file data 문자열을 표준 출력
    5. 서버가 Half Close 상태로 돌입하고 데이터를 받을 준비가 되면 자신이 파일에 저장했던 정보를 다시 읽어 서버에게 재전송
    6. 모든 파일정보를 주고나면 파일과 소켓을 닫고 프로그램 종료

03 Network assignment #4

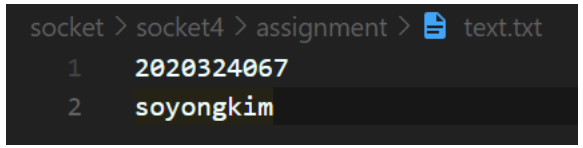
DNS & Half Close Practice

- **server.c**

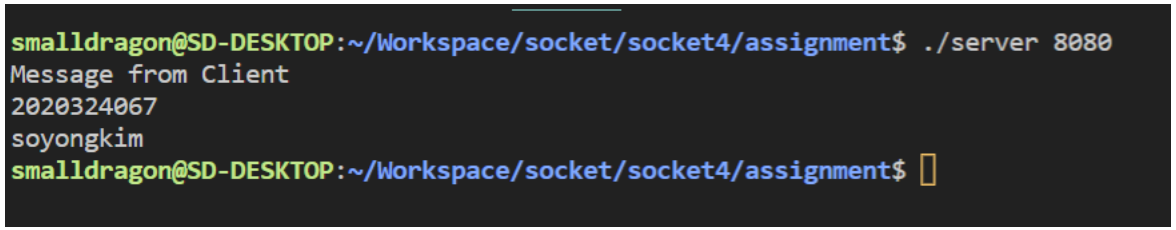
1. 서버 실행 시 main 함수의 매개변수로 포트번호를 받아서 실행

   Ex) ./server 8080

2. TCP 소켓을 생성하고, INADDR_ANY와 매개변수로 받은 포트번호로 소켓 바인드 진행
3. setsockopt()를 통해 SO_REUSEADDR 옵션 설정
4. 서버 프로그램과 같은 디렉토리에 위치한 "text.txt" 파일을 읽기 권한으로 염
   1. text.txt 파일의 내용은 자신의 학번과 영어이름으로 아래와 같이 작성



5. 클라이언트의 연결 요청을 받아 연결을 형성하면 text.txt 파일의 내용을 클라이언트에게 전송
6. 모두 전송하고 나면 Half close를 수행하여 Write Buffer를 닫고 클라이언트의 종료 전에 보내는 파일 데이터를 모두 받고 이를 Message from Client와 함께 표준 출력



7. 클라이언트의 메시지를 받은 후 파일과 소켓을 닫고 프로그램 종료

03 Network assignment #4

DNS & Half Close Practice

```
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/assignment$ ./server 8080
Message from Client
2020324067
soyongkim
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/assignment$
```

```
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/assignment$ ./client google.com
gethostbyname()
Official name: google.com
Address type: AF_INET
IP addr 0: 142.250.207.14

gethostbyaddr()
Official name: nrt13s54-in-f14.1e100.net
Address type: AF_INET
IP addr 0: 142.250.207.14
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/assignment$ ./client 8080 127.0.0.1
This socket type is : 1(1)
Received file data
smalldragon@SD-DESKTOP:~/Workspace/socket/socket4/assignment$
```

03 Network assignment #4

DNS & Half Close Practice

- 참고사항

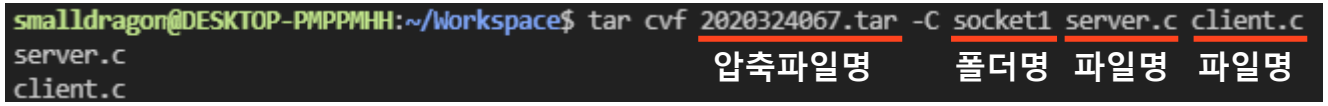
1. 서버가 의도한 것 이외의 값을 받는 케이스를 예외처리할 필요 없음
2. 클라이언트가 매개변수로 도메인 이름을 안 받는 경우도 예외처리할 필요 없음
3. 서버가 Half Close를 하지 않고 클라이언트의 파일 데이터를 받을 경우 점수 없음
4. 과제에서 의도한 대로 데이터를 주고받고 이를 출력하는 방식이 아닌, 겉으로 출력 결과만 똑같이 보인다면 점수 없음
5. 과제 관련 문의 : thdyd324@gmail.com

- 제출관련

1. 서버 프로그램은 server.c, 클라이언트 프로그램은 client.c로 명명하여 과제 진행
2. 빌드 시(gcc) Warning이 발생해서는 안됨
3. 제출 시 세 파일을 “자신의 학번.tar” 파일로 제출

   Ex) 2020324067.tar

   ~/Workspace/socket1/(server.c, text.txt, client.c)



text.txt

4. 과제는 10점 만점
5. 제출 기한: 2023.04.14(금) PM 11:59
6. 지각 제출 허용: 2023.04.18(화) PM 11:59 / 하루 늦을 때 마다 2점 씩 감점

   지각제출 시 보낼 이메일: minji001011@naver.com
7. 기한 안에 아예 제출을 하지 않았을 시 점수 없음