소켓 연결 종료의 문제점 파악 및 해결 방법 & DNS의 이해

인하공업전문대학 컴퓨터정보과
최효현 교수

주요사항

- 소켓 종료
- Half Close의 이해
- 파일 전송 프로그램의 구현 실습

- 도메인 네임과 DNS 이해
- 도메인 이름을 IP 주소로 변환하는 과정 실습
- hostent 구조체 이해

7.1 소켓 연결 종료의 문제점

- 소켓 종료에 대한 이해

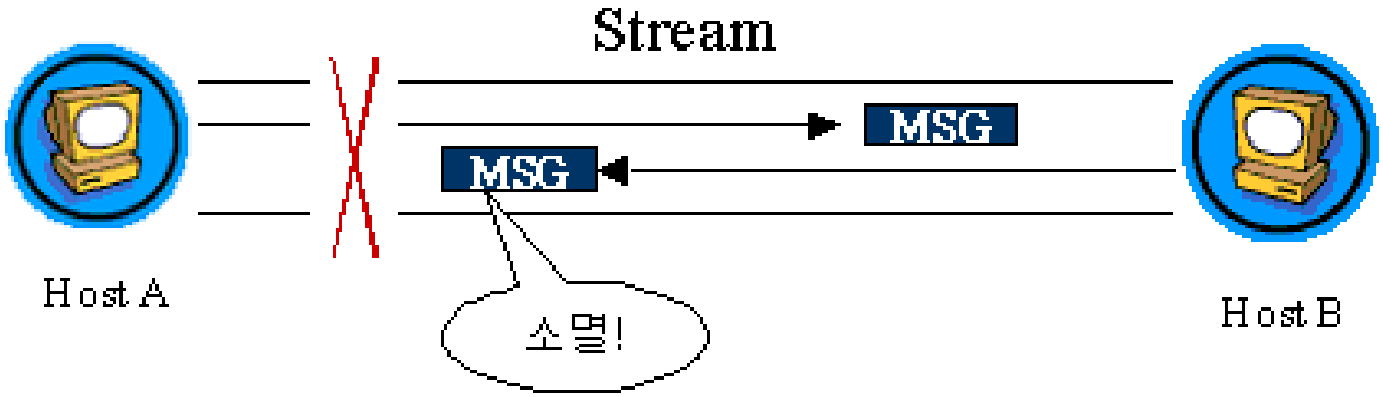
  - 파일 기술자를 이용한 close()함수의 호출

    => 완전한 연결종료 의미를 가짐

    => close() 함수를 이용한 연결종료는 매끄럽지 않음

  - 두 호스트가 전송 중에 한 호스트가 일방적으로 종료

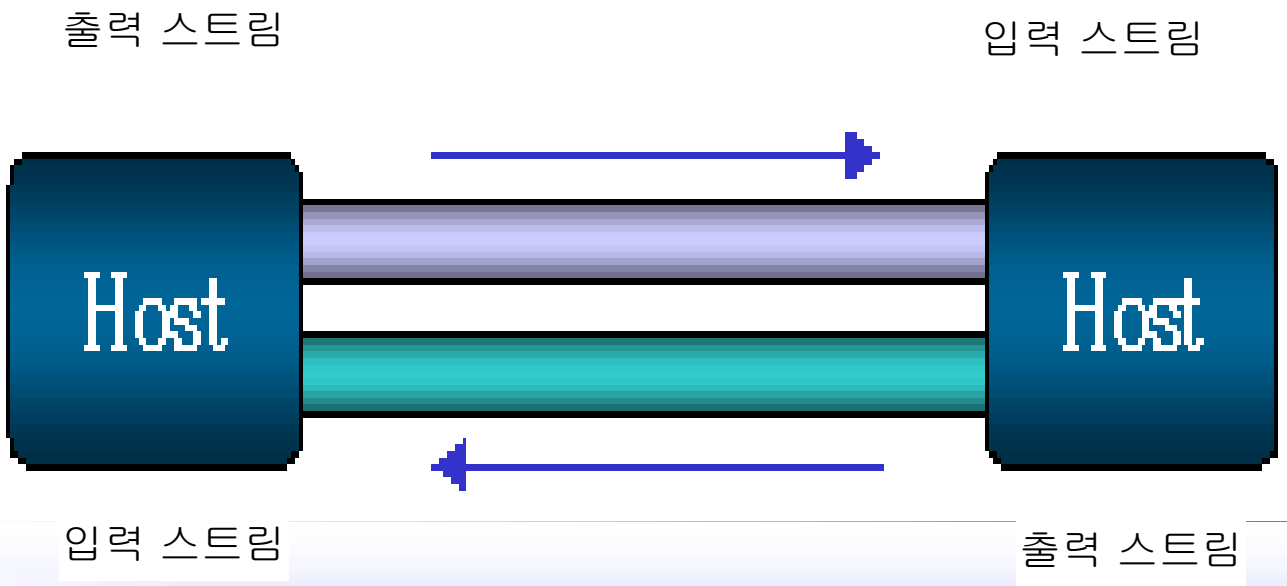
    => 전송 중에 있던 메시지 손실우려 존재



7.1 소켓 연결 종료의 문제점

- 입력 및 출력 스트림

1. 입력 스트림 : 데이터 수신을 위한 스트림
2. 출력 스트림 : 데이터 전송을 위한 스트림



7.2 우아한 소켓의 연결종료

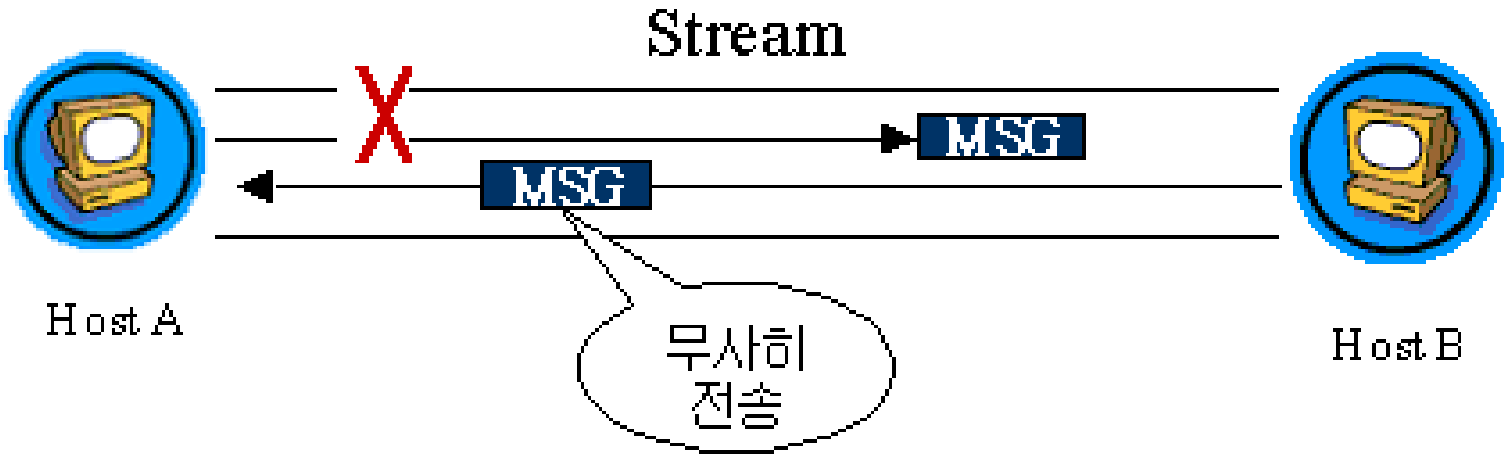
- **Half-Close**

  - 소켓 스트림의 일부만을 종료(half close)

    => 전송은 가능, 수신은 불가한 상황, 또는 수신은 가능, 전송은 불가능

    => Half close : 입력 및 출력 스트림 중 하나의 스트림만 종료하는 행위

    => 스트림의 반만 닫음



7.2 우아한 소켓의 연결종료

- **Half-Close 기능의 함수**

```
#include <sys/socket.h>
int shutdown(int s, int how);
```

s : 종료하고자 하는 소켓 기술자
how : 종료 모드

상수값	모드	정의
0	SHUT_RD	입력 스트림 종료
1	SHUT_WR	출력 스트림 종료
2	SHUT_RDWR	입 출력 스트림 종료

<표 7-1>

7.2 우아한 소켓의 연결종료

- **Half-Close 기능이 필요한 경우의 예제**
  - 출력 스트림의 종료의 필요성

1. 출력 스트림을 종료하게 되면, 연결되어 있던 호스트로 **EOF** 메시지 전달.
2. **EOF**의 전송으로 데이터 전송의 끝을 알려 줄 수 있다.
3. **EOF** 전송 시, 상대 호스트의 데이터 수신 함수**(read, recv)**는 **0**을 리턴.

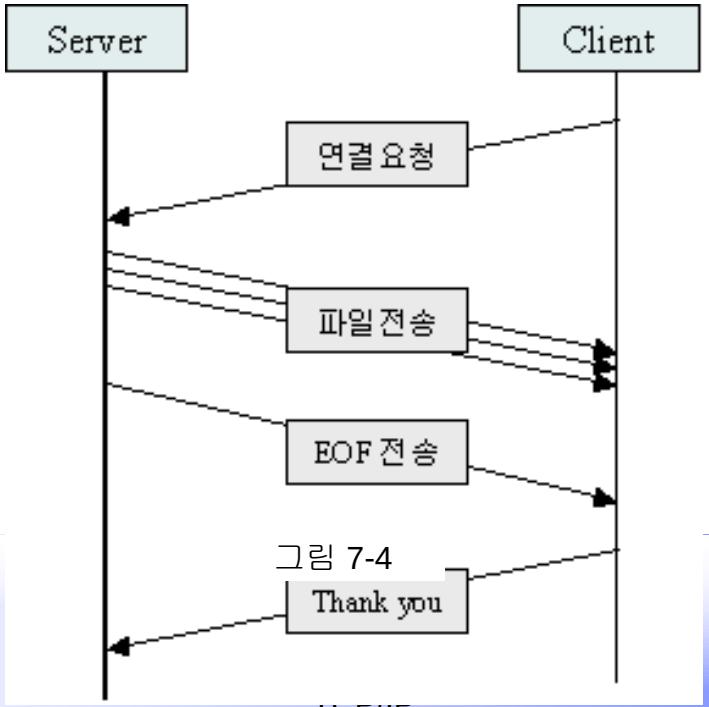


그림 7-4

7.2 우아한 소켓의 연결종료

- 파일 전송 서버 / 클라이언트
  - 서버
    - File Open
    - Socket Open
    - File의 내용을 BUFSIZE (30 byte) 만큼씩 Read 하여 Socket에 Write 함 → File의 모든 내용을 전송할 때 까지
    - **Socket Half-Close (shutdown)**
    - 클라이언트로 부터 메시지 수신 **("Thank You")**
    - File Close
    - Socket Close

7.2 우아한 소켓의 연결종료

- 파일 전송 서버 / 클라이언트
  - 클라이언트
    - File Open
    - Socket Open
    - Socket에서 BUFSIZE (30 byte) 만큼씩 Read하여

      File에 Write 함 → Server의 Socket이 Half-Close될 때까지

      (== Server에서 EOF가 전달될 때)

      (== read()의 읽은 byte 수가 0)
    - 서버에게 **"Thank You"** 메시지 전송
    - File Close
    - Socket Close

7.2 우아한 소켓의 연결종료

- 파일 전송 서버 / 클라이언트

1. 프로그램 예제

- file_server.c, file_client.c

8.1 DNS(Domain Name System)

- 도메인 네임**(Domain name)**

1. **영문으로 표현되는 계층적 주소 체계 방식.**
   : 인터넷에서 호스트를 구별하는 주소 : 4바이트(32비트)
   : 해당 IP 주소에 도메인 이름을 붙여 편리하게 사용

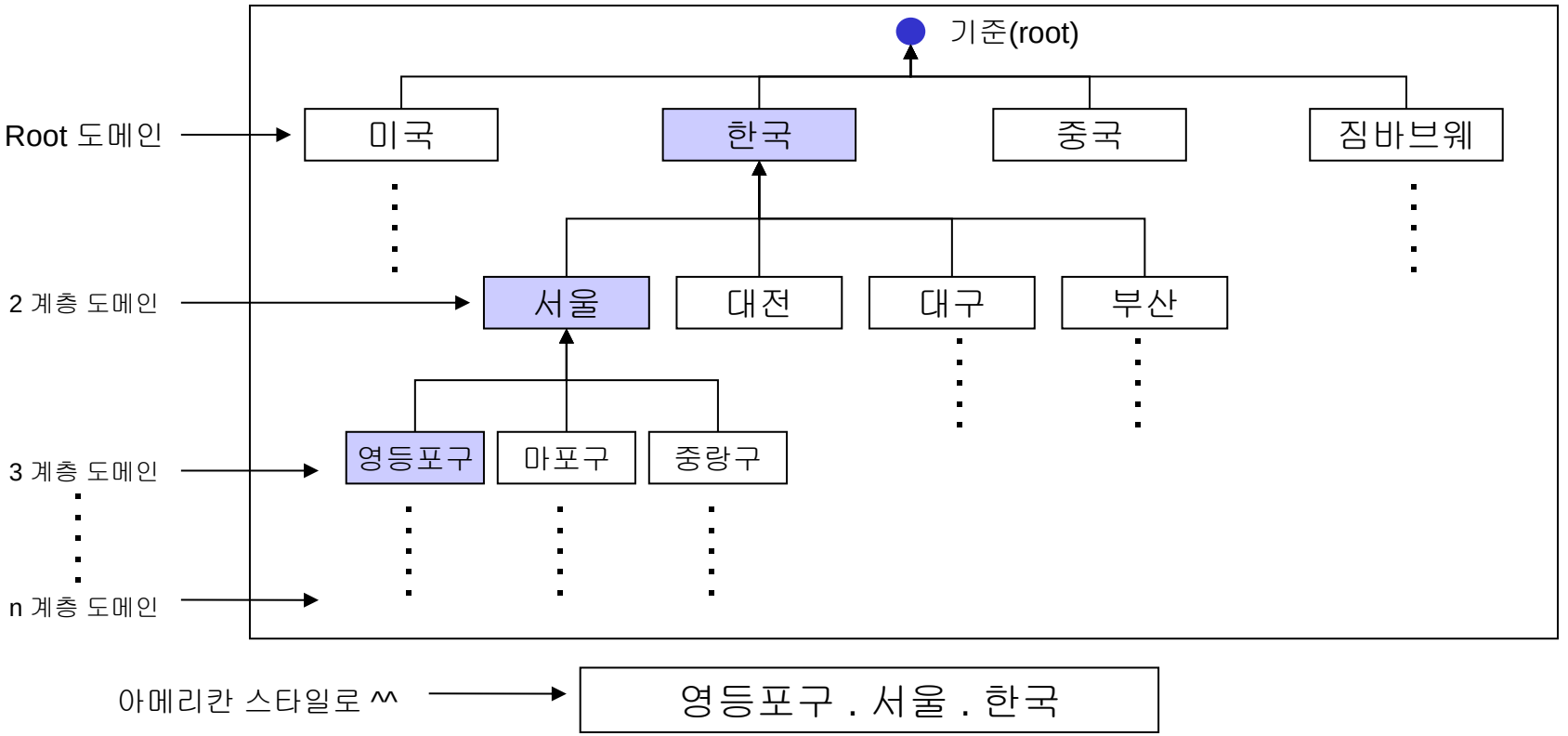
2. **각 나라마다 존재하는 Network Information Center에서 관리.**
   : 한국은 (KRNIC : Korea Network Information Center : www.nic.or.kr)에서 담당

3. **도메인 이름 = 호스트 이름 + 도메인 이름**
   : cs.inhatc.ac.kr = cs + inhatc.ac.kr

4. **TCP/IP**는 도메인 이름을 인식 못함.

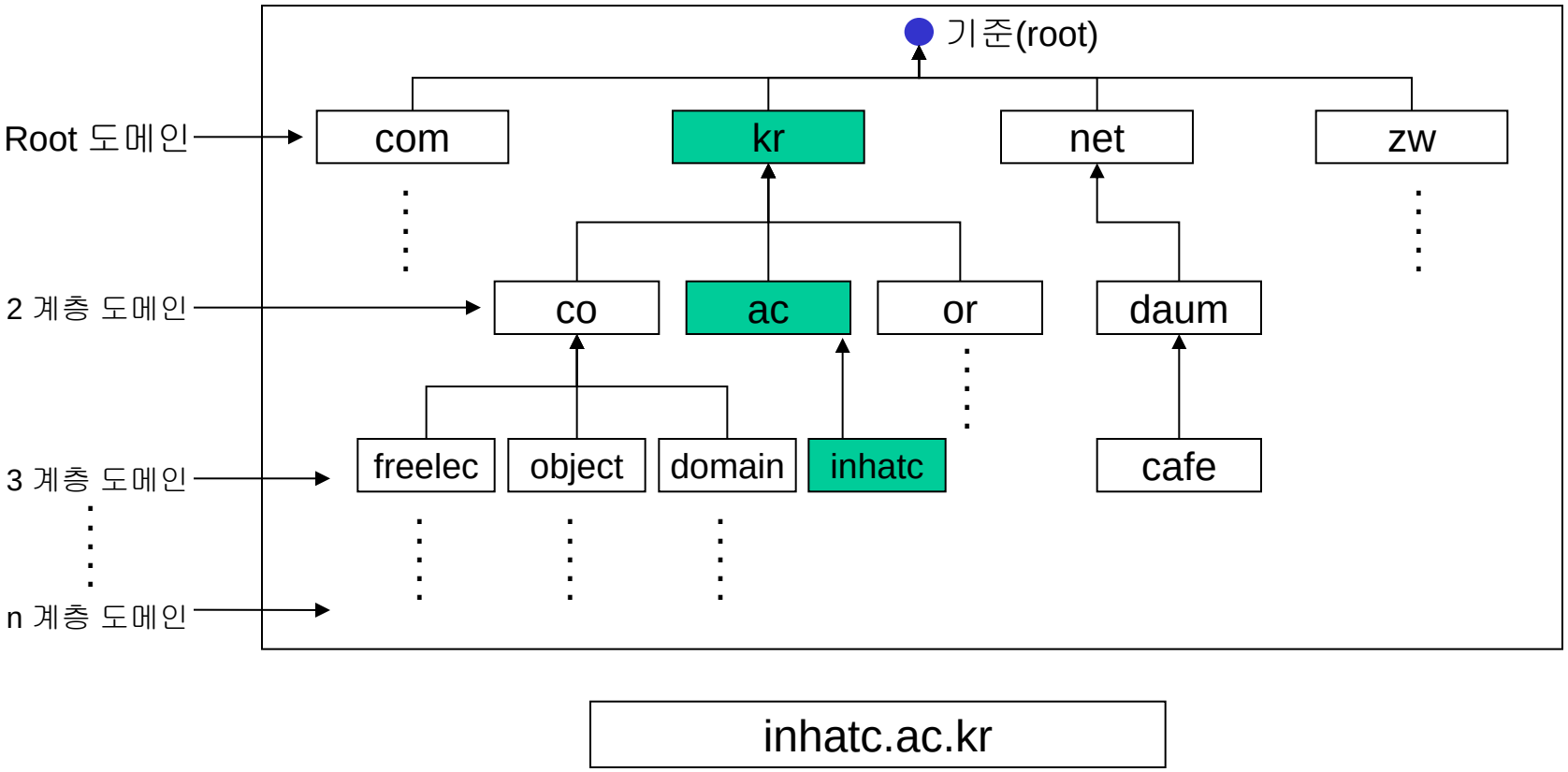
8.1 DNS(Domain Name System)

- 현실세계와 주소체계



8.1 DNS(Domain Name System)

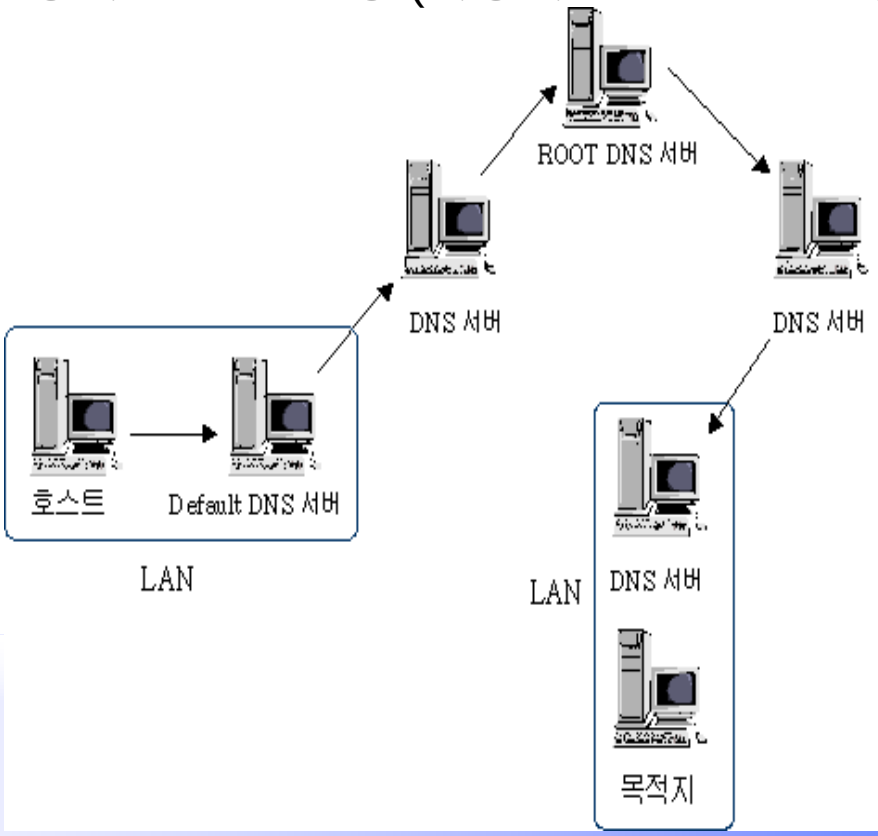
- 인터넷 상에서의 주소체계



• DNS(Domain Name System)

: Domain 이름을 IP 주소로 변환해 주는 작업

- 도메인 이름을 IP 주소로 변환해주는 서버 : 도메인 네임 서버
- 디폴트 DNS 서버가 설정되어 있음, 디폴트 DNS 서버가 이 정보를 가지고 있지 않으면 상위 DNS로 요청 (최상위는 root DNS 서버)



8.2 IP주소와 도메인 이름사이의 변환

- **변환(도메인 이름→IP주소) 함수1**

```
#include <netdb.h>
struct hostent* gethostbyname(const char* name);
```

[성공시 : **hostent 구조체의 포인터**, 실패시 : null 포인터 리턴]

name : 변환하고자 하는 도메인 네임

8.2 IP주소와 도메인 이름사이의 변환

- **struct hostent**

```
struct hostent {
    char* h_name;
    char **h_aliases;
    int h_addrtype;
    int h_length;
    char **h_addr_list;
}
```

Official domain name - 공식 홈페이지 주소

Alias list - 다른 홈페이지주소를 사용할 시

Host address type -IP 주소체계(IPv4 또는 IPv6)

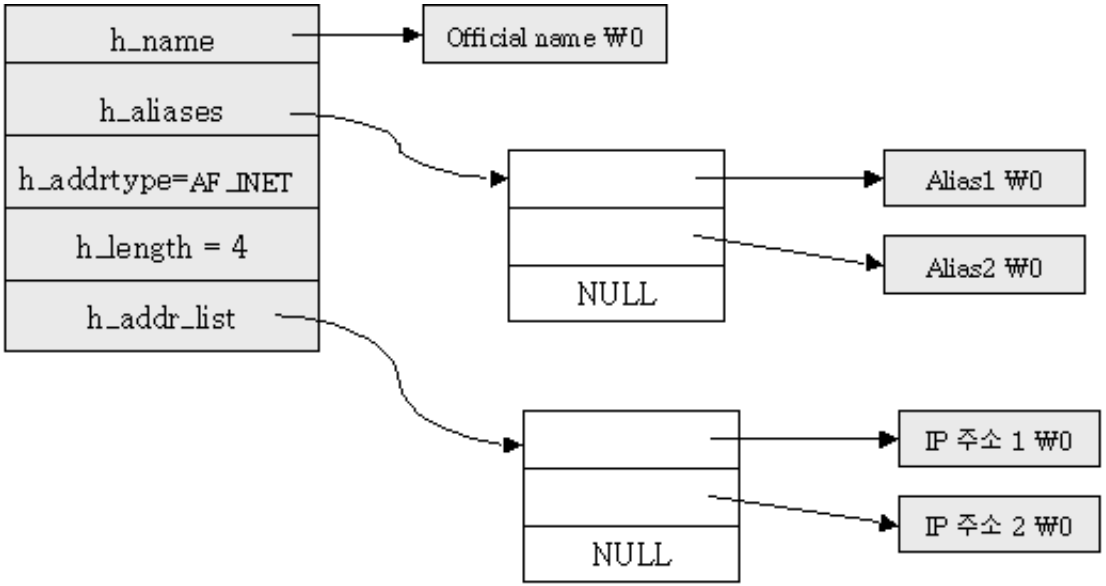
Length of address -주소 길이(4바이트, 16바이트(IPv6))

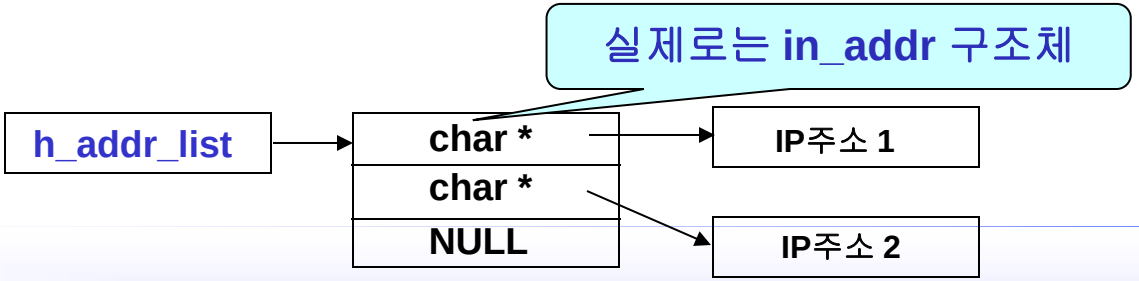
List of address - 가장중요, 도메인 이름에 해당하는 **IP** 주소

한 회사라도 여러 개의 서버를 운영하는 경우, 하나의 도메인에 대응하는 모든 **IP**주소를 리스트 업

8.2 IP주소와 도메인 이름사이의 변환

- **struct hostent**

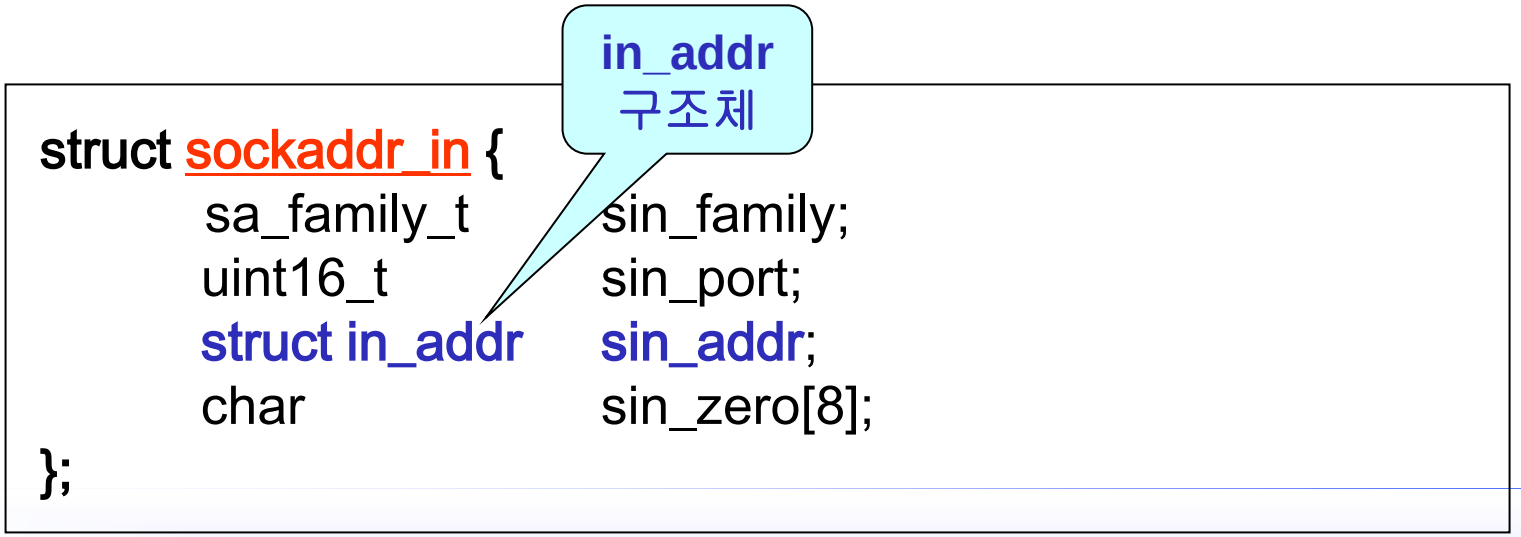




8.2 IP주소와 도메인 이름사이의 변환

- 참고 - in_addr 구조체

```
struct in_addr {
        uint32_t        s_addr;     /* 32비트 IP 주소정보 */
};
```



8.2 IP주소와 도메인 이름사이의 변환

- **h_addr_list[i]는 struct in_addr 로 형 변환을 해서 사용하여야 함**

  예: ***(struct in_addr *)host->h_addr_list[i]**

- **32 bit의 숫자에 해당하는 IP 주소를 "127.0.0.1" 같은 형태로 바꾸어 주어야 함**

  예: **inet_ntoa(*(struct in_addr *)host->h_addr_list[i])**

8.2 IP주소와 도메인 이름사이의 변환

- 예제 확인

1. 프로그램 예제

- gethostbyname.c

[그림 8-4]

8.2 IP주소와 도메인 이름사이의 변환

- 변환(IP주소→도메인 이름) 함수2

in_addr 구조체

```
#include <netdb.h>
struct hostent* gethostbyaddr(const char* addr, int len, int type);
```

[성공시 : **hostent 구조체의 포인터**, 실패시 : null 포인터 리턴]

addr : 실제로는 in_addr 구조체임, IPv4와 IPv6를 모두 수용하기위한 일반화 선언
len : 입력되는 주소 길이(IPv4인 경우는 4)
type : 주소체계(AF_INET, AF_INET6)

Q&A