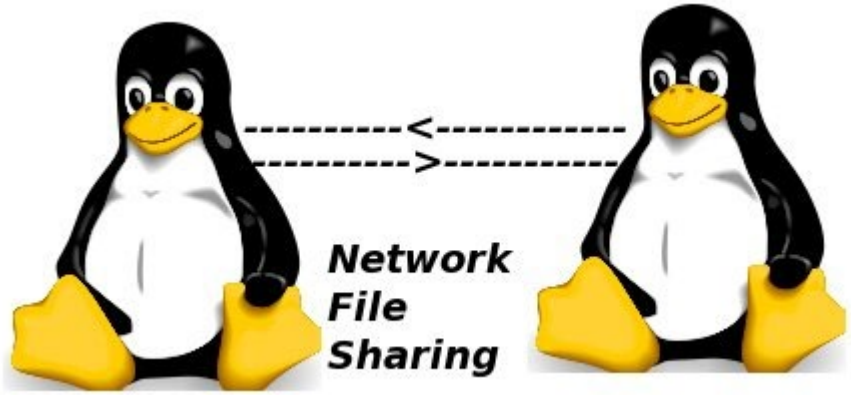
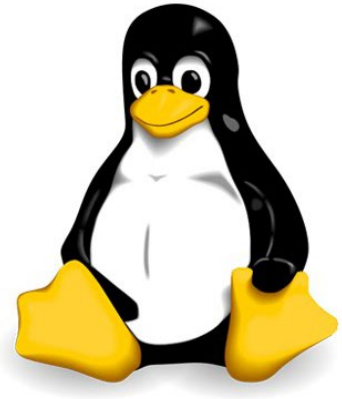
NFS, FTP

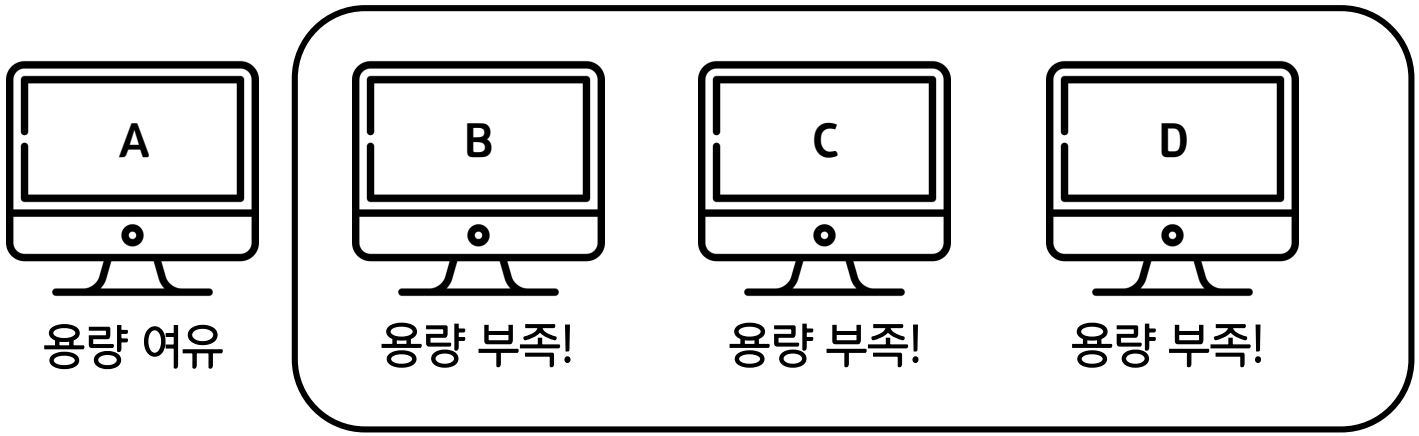


FTP
SERVER



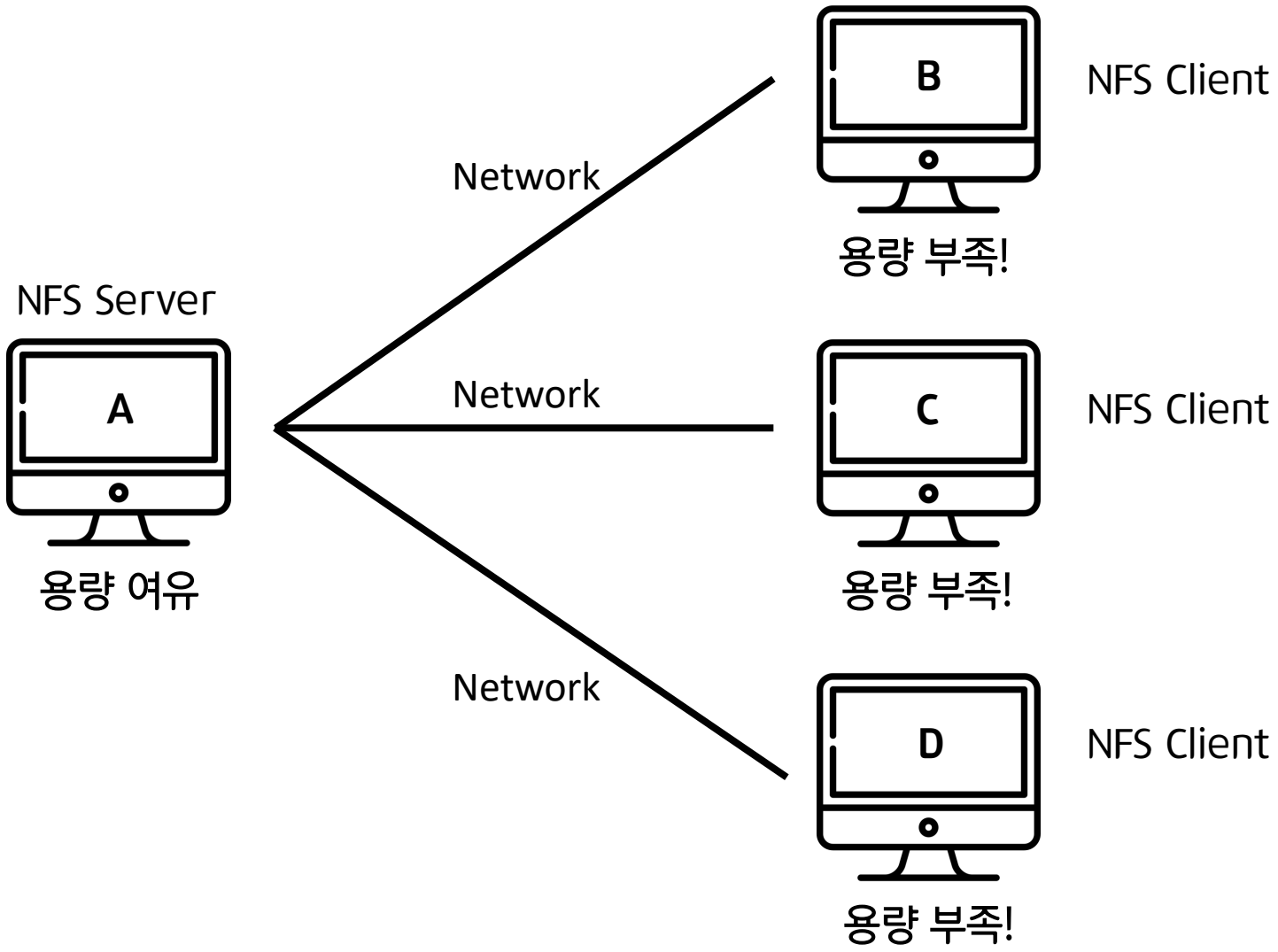
ball

What is NFS ?



A의 저장공간을 사용하고 싶다…

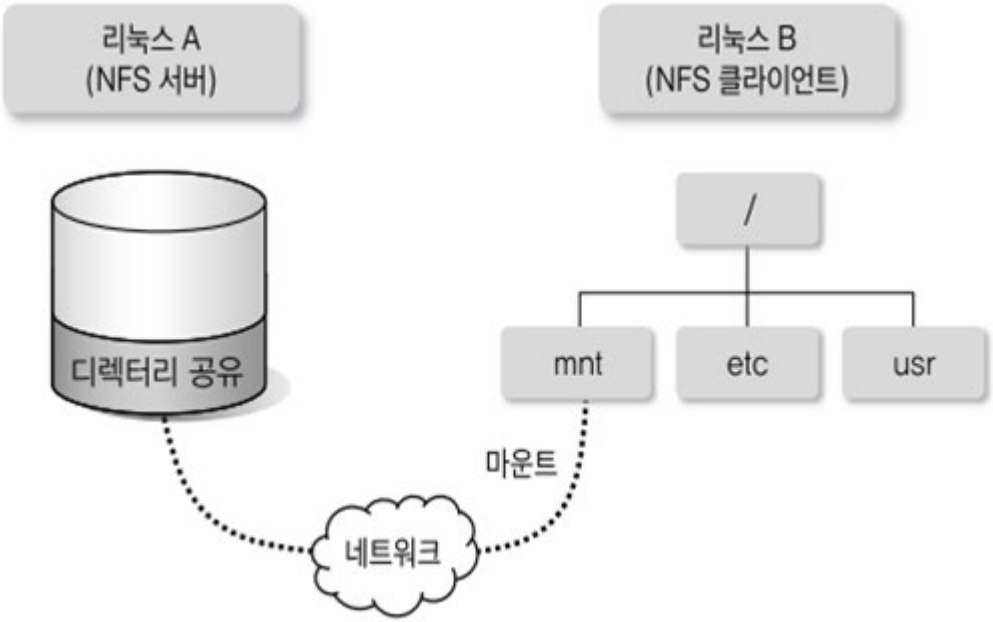
NFS is Network File System



NFS is Network File System

NFS이란?
- 서버/클라이언트 모델로 동작하는 스토리지 프로토콜
- 서버에서 공유한 디렉토리를 로컬에서 Mount하여 자신의 것처럼 이용
- RPC 기반으로 작동한다.

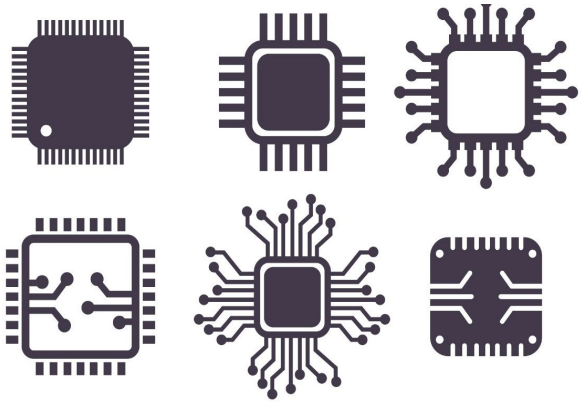
**Mount: 파일 시스템 구조에 있는 파일들을 사용자가 이용할 수 있도록 만드는 것



Where do we use NFS?

CLOUD

클라우드를 통해 클라이언트의
저장공간 부족 문제를 해결

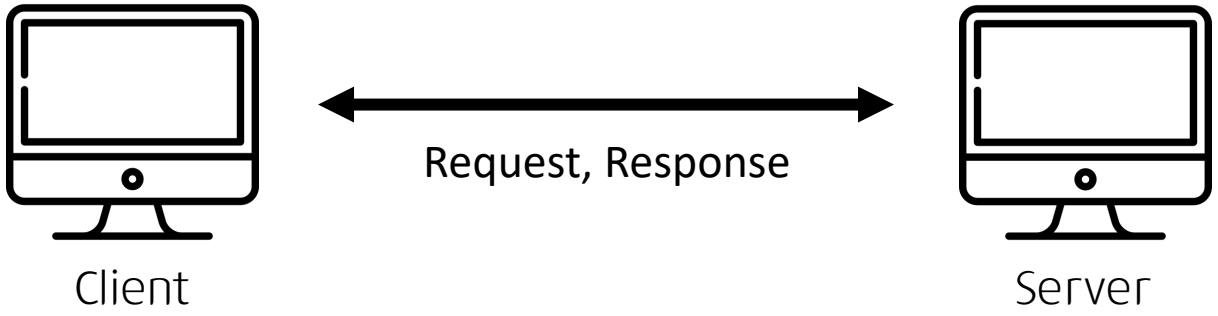


EMBEDDED SYSTEM

Embedded System은 저장공간이 매우 작다.
따라서 NFS를 통해 저장공간 부족문제를 해결할 수 있다.

Embedded System이 무엇인지 궁금하다면 -> https://swev.net/43

Before RPC

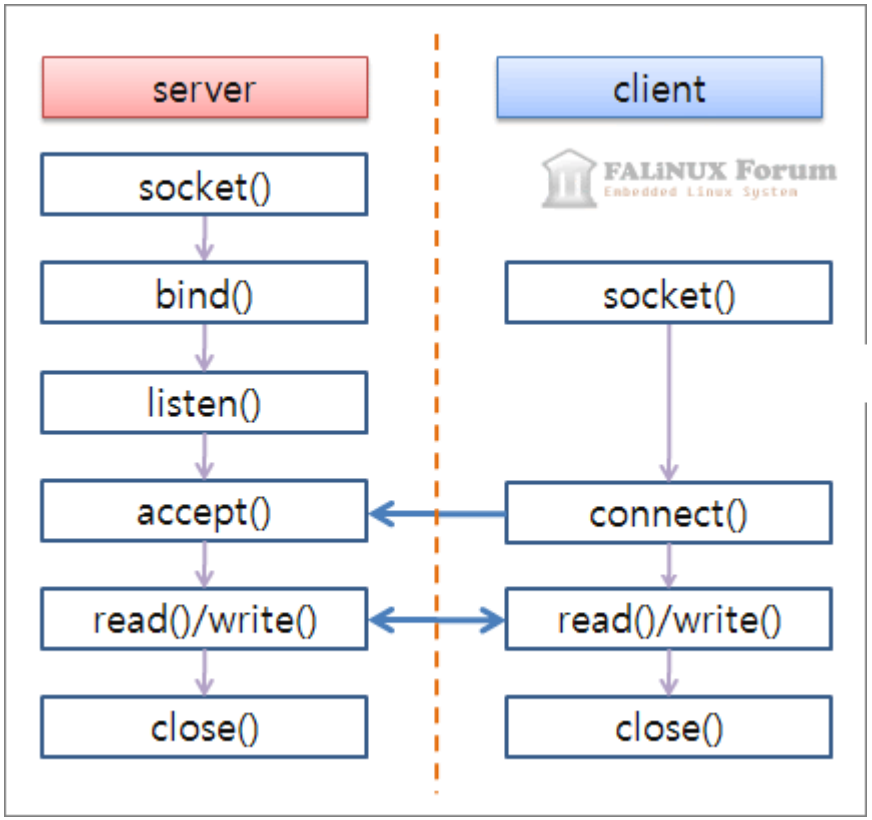


RPC가 등장하기 이전에는, Client와 Server가 socket을 활용하여 연결되었다.

1. Client가 Server에 Request를 보낸다.(Network의 상세 주소(IP, port number)를 입력)
2. Server가 Request를 받아, 요청을 처리한다.
3. Server가 Client에게 Response를 보낸다.
4. “기다리던” Client가 Response를 받고, 작업을 계속 진행한다.

Before RPC

Server	Client
socket() socket을 만든다. 만들어진 소켓은 소켓 지정번호를 가지며 아직 아무런 ip주소, port번호와 도 연결되지 않은 상태이다.	
bind() 만들어진 소켓을 ip주소, port번호와 연결 (bind)한다. 연결된 소켓으로 이제 통신이 가 능하다.	connect() 클라이언트가 서버에 서버의 ip주소와 port번 호로 연결을 시도한다.
listen() 서버가 클라이언트의 연결을 기다린다.	
accept() 서버가 클라이언트의 연결을 받아들인다.	
read()/write() 서버와 클라이언트가 서로에게 데이터를 주고 받는다.	
close() 다 사용한 소켓을 종료한다.	

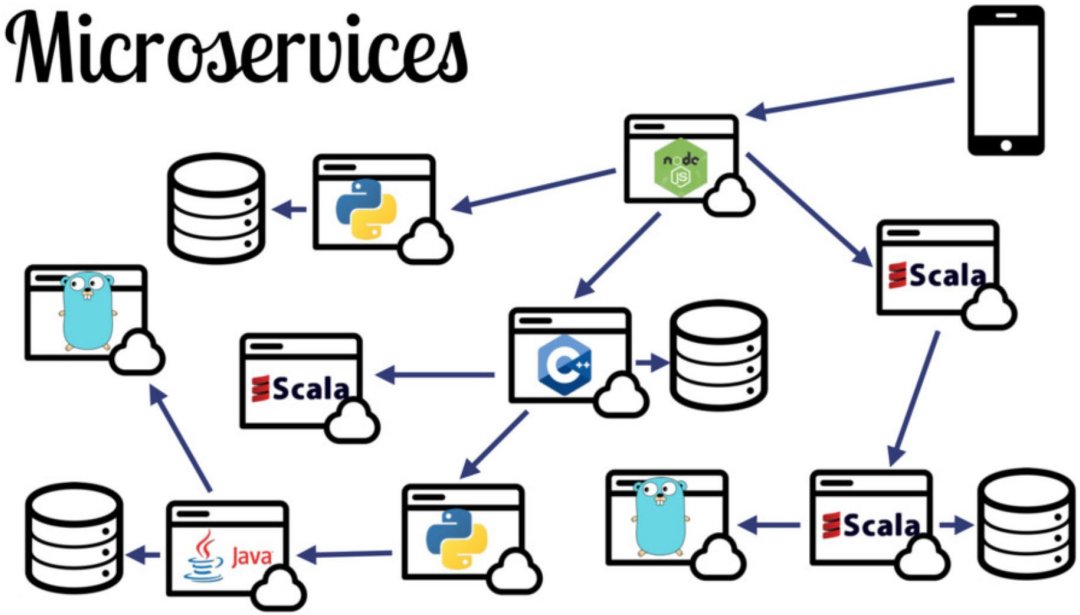


이러한 방법을 Socket Programming이라 한다.
Socket Programming으로 통신을 하기 위해서는 네트워크의 주소를 정확히 알고 있어야 한다.
(즉, 서버의 ip주소와 port 번호를 정확히 알고 있어야 한다.)

Before RPC- 분산시스템의 등장

예를 들어, 어떤 웹사이트의 인기가 많아져서 요청이 늘고, 서버의 처리량이 급증하여 서버 한계 처리량을 넘었다고 하자.
해결책1. 더 좋은 서버로 바꾼다.
해결책2. 여러 서버로 요청 처리를 분산시킨다.(다시말해, 분산시스템을 구축한다.)

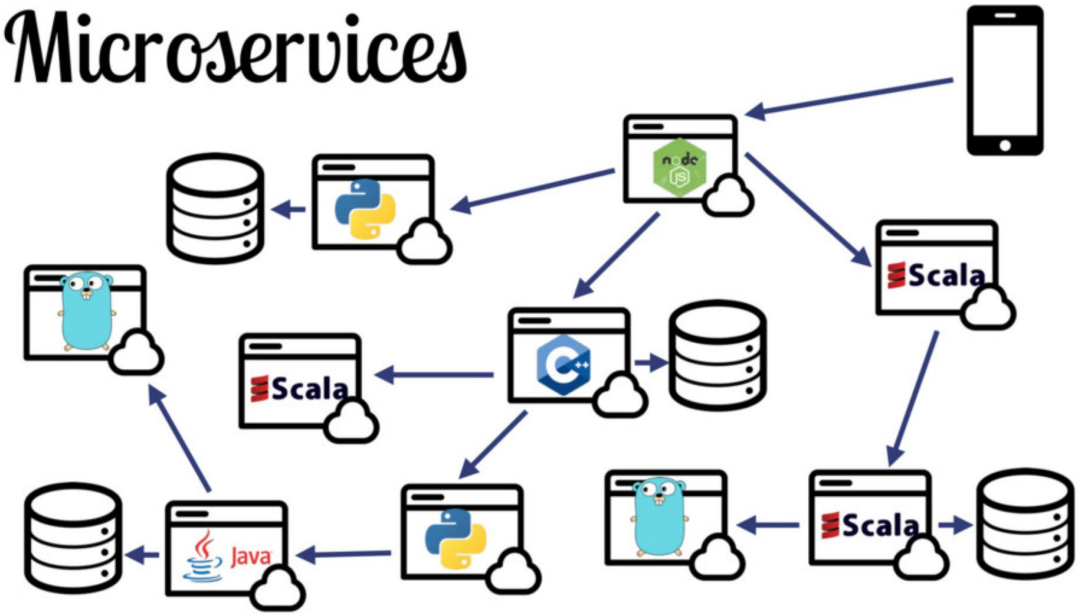
두 해결책 모두 돈이 든다. 하지만 해결책2를 사용할 경우, 퍼포먼스가 지수함수적으로 증가한다.



Before RPC

분산시스템에서 Socket Programming을 사용한다고 하자.
-> 네트워크의 복잡도가 어마무시하다.
-> 일일이 모든 네트워크에 대한 주소를 직접 입력하면서 요청을 보내야 한다.
-> 개발자가 때려 치울 수도 있다.

-> Socket Programming을 대체할 RPC가 등장!!



RPC(Remote Procedure Call)

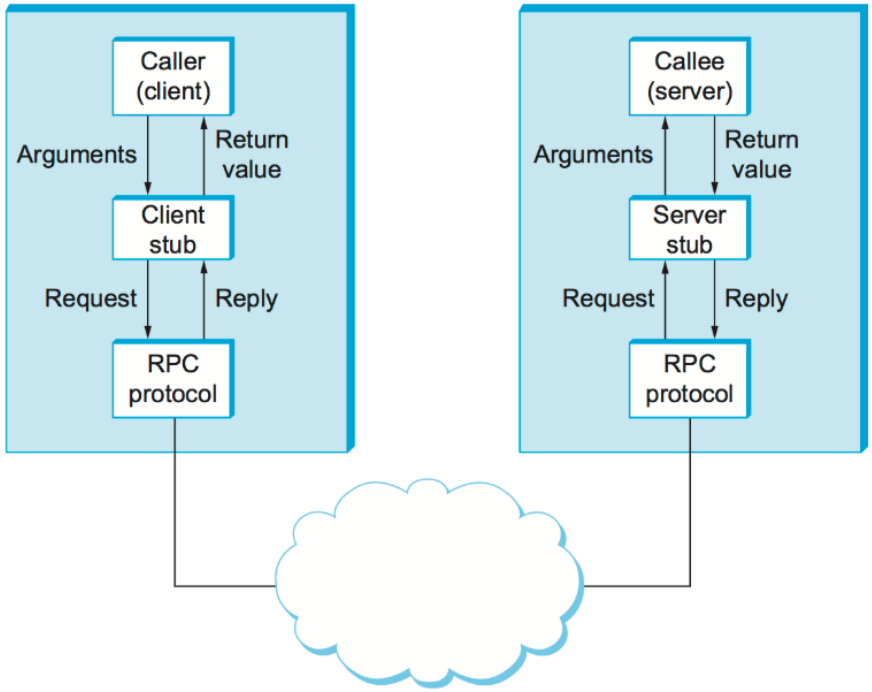


Figure 2. Complete RPC mechanism.

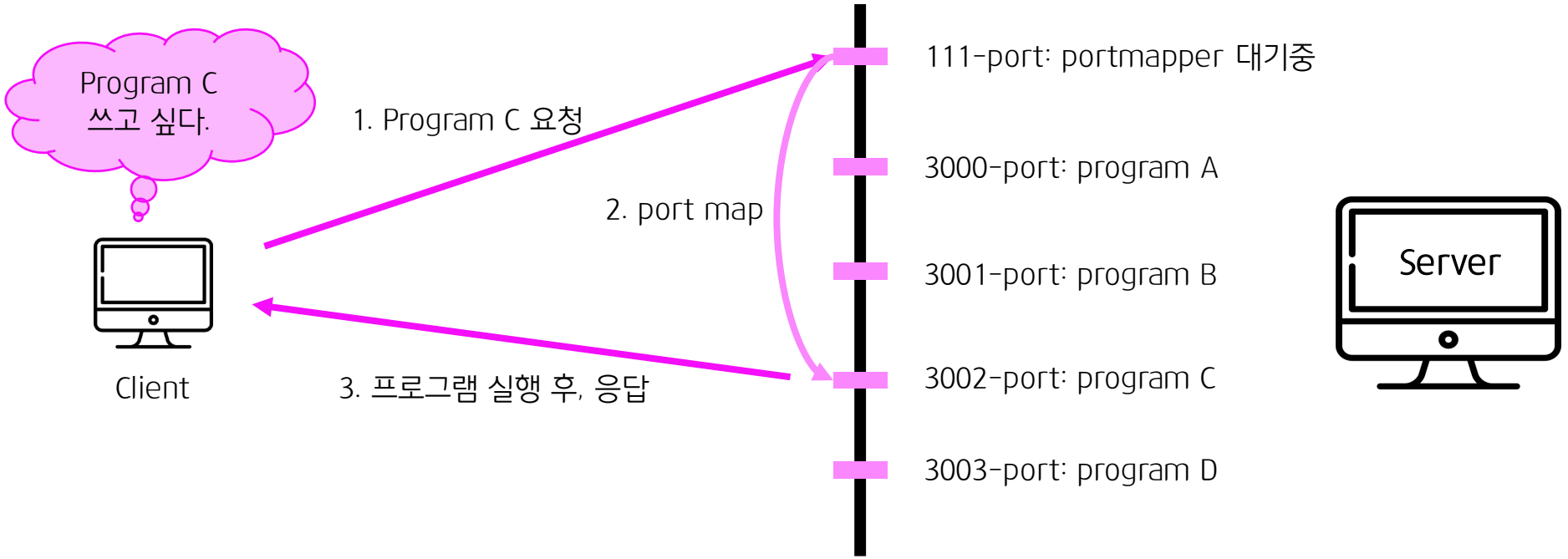
RPC란?

- 원격 절차 호출
- Client에서 Server에 있는 프로그램을 실행할 때 사용하는 프로토콜이다.
- 서비스를 요청하는 프로그램은 **네트워크에 대한 상세 내용을 알 필요가 없다**.
  (RPC 에서 알아서 다 처리해주기 때문이다.) => 분산시스템에 사용된다!
- 정리하자면, Server에 있는 프로그램을 로컬에 있는 것처럼 사용할 수 있게 한다.

How does RPC work?

Portmap(portmapper)

- RPC Network 연결에 관여하는 Daemon이다.
- Server와 Client사이를 연결해준다.(네트워크와 관련된 것들은 알아서 처리해준다.)
- Server의 111번 포트에서 대기중(?)인 프로그램이다.
- 만약, Client에서 Server의 111번 포트로 서비스를 요구하면, 요청한 프로그램이 있는 다른 포트로 접속시켜준다.(포트 할당)
- 요청하는 프로그램에 따라 다른 포트를 할당한다.
- Client는 그냥 Server의 111번 포트로 요청을 보내면 된다.-> portmapper가 알아서 요청에 맞게 port를 mapping해준다.



NFS 실습- 실습환경 설정

NFS 실습을 하기 위해서는 Server, Client 2개의 인스턴스가 필요하다.
Server Instance: aws-EC2에서 ubuntu18.04 Image로 instance 하나를 만든다.
이때, 보안그룹에서는 SSH(내 IP)와 HTTP(위치 무관)만 열어준다.
(그리고 이 보안그룹을 nfs server 보안그룹이라 이름을 짓는다)
인스턴스를 생성한 뒤, 탄력적 IP에 연결시킨다.

그리고 Client Instance: aws-EC2에서 ubuntu18.04 Image로 instance 하나를 만든다.
이때, 보안그룹에서는 SSH(내 IP)와 HTTP(위치무관)만 열어준다.
(그리고 이 보안그룹을 nfs client 보안그룹이라 이름을 짓는다)
인스턴스를 생성한 뒤, 탄력적 IP에 연결시킨다.

Name	인스턴스 ID	인스턴스 유형	가용 영역	인스턴스 상태	상태 검사	경보 상태
nfs client	i-00825748610c06230	t2.micro	ap-northeast-2c	running	2/2개 검사 ...	*없음*
nfs server	i-09cea8410b6c0bef9	t2.micro	ap-northeast-2c	running	2/2개 검사 ...	*없음*

NFS 실습- 실습환경 설정

우리는 RPC를 이용할 것이기 때문에, 서버 쪽에서는 여러 port를 열어줘야 한다.
이때, 사용될 port는 random하게 배치되기 때문에, 그냥 Server의 모든 포트를 열어주자.
(사실 어떤 포트가 사용되도록 지정할 수는 있지만, 실습을 간단히 하기 위해
모든 포트를 열어주겠다. 자세한 내용은 https://minidora.tistory.com/208)
하지만, 보안이 너무 허술해질 것을 대비해, client IP에 대해서만 개방해 주자.
즉, client instanc의 IP에 대하여 Server instanc의 모든 TCP & UDTPort를 열어주자.

nfs server 보안그룹을 다음과 같이 설정해주자.

인바운드 규칙

유형	프로토콜	포트 범위	소스
HTTP	TCP	80	0.0.0.0/0
HTTP	TCP	80	::/0
모든 TCP	TCP	0 - 65535	nfs client의 탄력적 IP
SSH	TCP	22	로컬 컴퓨터의 IP
모든 UDP	UDP	0 - 65535	nfs client의 탄력적 IP

NFS 실습- 실습환경 설정

Server쪽에서 포트를 열고, 또 client쪽에서도 포트를 열어줘야 한다.
이 또한 Server IP에 대해 모든 포트를 열어주도록 하자.

nfs client 보안그룹을 다음과 같이 설정해주자.

인바운드 규칙

유형	프로토콜	포트 범위	소스
HTTP	TCP	80	0.0.0.0/0
HTTP	TCP	80	::/0
모든 TCP	TCP	0 - 65535	nfs server의 탄력적 IP
SSH	TCP	22	로컬 컴퓨터의 IP
모든 UDP	UDP	0 - 65535	nfs server의 탄력적 IP

NFS 실습- NFS Server 설정

$ sudo apt update
$ sudo apt install nfs-common nfs-kernel-server portmap
//client에서 마운트 할 수 있는 디렉토리를 만들고, 읽고 쓰고 실행할 권한을 부여한다.
$ mkdir nfs_test
$ chmod 777 nfs_test
#NFS 환경설정을 하자.
$ vim /etc/exports # sudo 권한이 필요할 수도 있다.
# [공유할 디렉토리] [공유할 client IP](option)
# [공유할 client IP]와 (option)사이에 띄어쓰기가 없다.
# 만약, 모든 client IP에 대해 마운트 할 수 있게 하려면 *를 쓰면 된다.
>> (ex) /home/ubuntu/nfs_test *(rw,sync,no_subtree_check)

List of Options

rw : 읽기, 쓰기 가능
ro : 읽기만 가능
secure : 클라이언트 마운트 요청시 포트를 1024 이하로 합니다.
noaccess : 액세스 거부
root_squach : 클라이언트의 root가 서버의 root권한을 획득하는 것을 막습니다.
no_root_squash : 클라이언트의 root와 서버의 root를 동일하게 합니다.
sync : 파일 시스템이 변경되면 즉시 동기화합니다.
all_squach : root를 제외하고 서버와 클라이언트의 사용자를 동일한 권한으로 설정합니다.
no_all_squach : root를 제외하고 서버와 클라이언트의 사용자들을 하나의 권한을 가지도록 설정합니다.

NFS 실습- NFS Server 설정

# NFS 서비스 시작하기.
$ sudo service nfs-kernel-server restart
$ sudo service portmap restart
# NFS Service 가 잘 실행되는지 확인하기 위해서는
$ sudo rpcinfo -p

```
ubuntu@ip-172-31-44-89:~$ sudo rpcinfo -p
   program vers proto   port  service
    100000    4   tcp    111  portmapper
    100000    3   tcp    111  portmapper
    100000    2   tcp    111  portmapper
    100000    4   udp    111  portmapper
    100000    3   udp    111  portmapper
    100000    2   udp    111  portmapper
    100005    1   udp  46268  mountd
    100005    1   tcp  48121  mountd
    100005    2   udp  35566  mountd
    100005    2   tcp  59985  mountd
    100005    3   udp  36233  mountd
    100005    3   tcp  40257  mountd
    100003    3   tcp   2049  nfs
    100003    4   tcp   2049  nfs
    100227    3   tcp   2049
    100003    3   udp   2049  nfs
    100227    3   udp   2049
    100021    1   udp  41615  nlockmgr
    100021    3   udp  41615  nlockmgr
    100021    4   udp  41615  nlockmgr
    100021    1   tcp  39253  nlockmgr
    100021    3   tcp  39253  nlockmgr
    100021    4   tcp  39253  nlockmgr
```

Port가 Random하게 사용되는 것을 볼 수 있다.
Client IP에 대해 모든 포트를 연 이유이다.

NFS 실습- NFS Client설정 및 Mount

#NFS Client의 설정을 다음과 같이 하자.
$ sudo apt update
# nfs client package download
$ sudo apt install nfs-common
#Server 와 마운트할 폴더 생성
$ mkdir mount_folder
#Server와 마운트하기
$ sudo mount –t nfs [server IP]:/home/ubuntu/nfs_test /home/ubuntu/mount_folder

```
ubuntu@ip-172-31-44-89:~$ ls -l nfs_test/
total 0
-rw-rw-r-- 1 ubuntu ubuntu 0 Jul 17 10:04 a.txt
-rw-rw-r-- 1 ubuntu ubuntu 0 Jul 17 10:04 b.txt
-rw-rw-r-- 1 ubuntu ubuntu 0 Jul 17 10:04 c.log
ubuntu@ip-172-31-44-89:~$ #server
ubuntu@ip-172-31-44-89:~$
```

```
ubuntu@ip-172-31-46-206:~$ ls -l mount_folder/
total 0
-rw-rw-r-- 1 ubuntu ubuntu 0 Jul 17 10:04 a.txt
-rw-rw-r-- 1 ubuntu ubuntu 0 Jul 17 10:04 b.txt
-rw-rw-r-- 1 ubuntu ubuntu 0 Jul 17 10:04 c.log
ubuntu@ip-172-31-46-206:~$ #client
ubuntu@ip-172-31-46-206:~$
```

NFS 실습- Mount 해제하기

#Client에서 mount 해제하기
# sudo umount [mount point]
$ sudo umount nfs

```
ubuntu@ip-172-31-46-206:~$ sudo umount ./mount_folder
ubuntu@ip-172-31-46-206:~$ ls
mount_folder
ubuntu@ip-172-31-46-206:~$ ls -l mount_folder/
total 0
```

#Server에서 export하고 있는 디렉토리 목록 확인하기
$ sudo showmount –e [server IP]

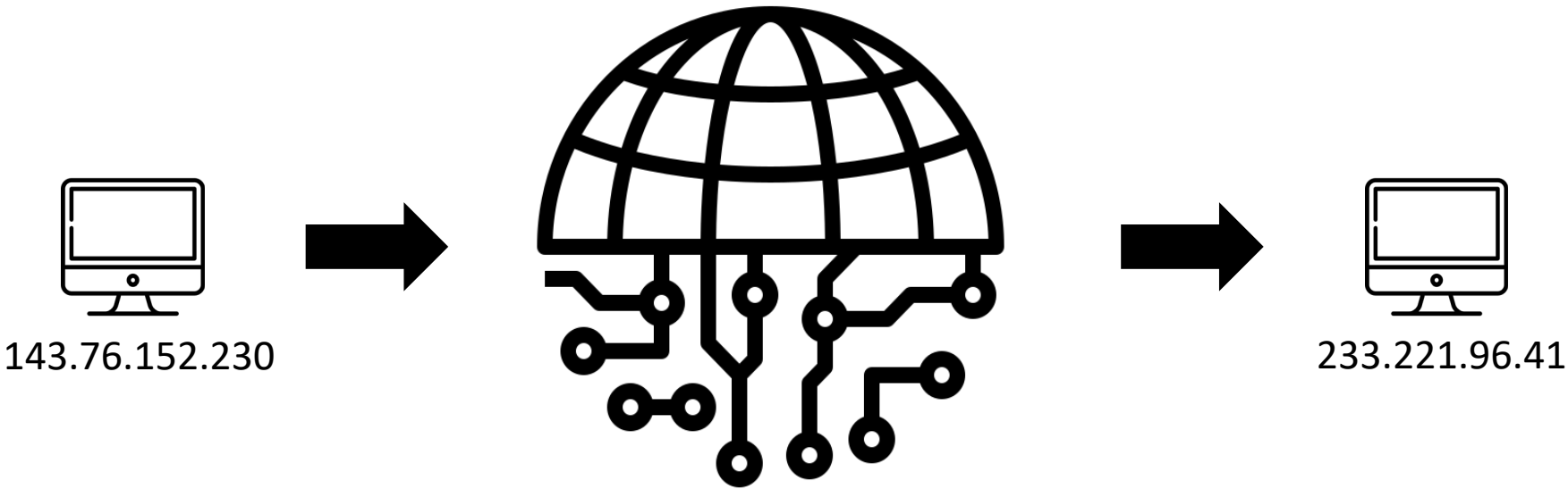
```
ubuntu@ip-172-31-46-206:~$ #client
ubuntu@ip-172-31-46-206:~$ sudo showmount -e 52.78.181.206
Export list for 52.78.181.206:
/home/ubuntu/nfs_test *
ubuntu@ip-172-31-46-206:~$
```

NFS Server에 있는 폴더를 NFS Client에서 마운트 했다는 것이 보이도록 Shell을 캡처해주세요!

TCP/IP

내가 브라우저에 URL을 입력하면, URL에 해당하는 웹페이지가 보여진다.
이는 브라우저가 입력한 URL 의 서버에 웹페이지를 요청하고, HTML 등을 응답으로 받아, 브라우저에 표시된다.

어떻게 요청이 네트워크를 통해 서버로 전달되고, 응답이 네트워크를 타고 브라우저로 전달되는 것일까?
-> TCP/IP를 통해 이러한 것이 가능하다.



What is TCP/IP?

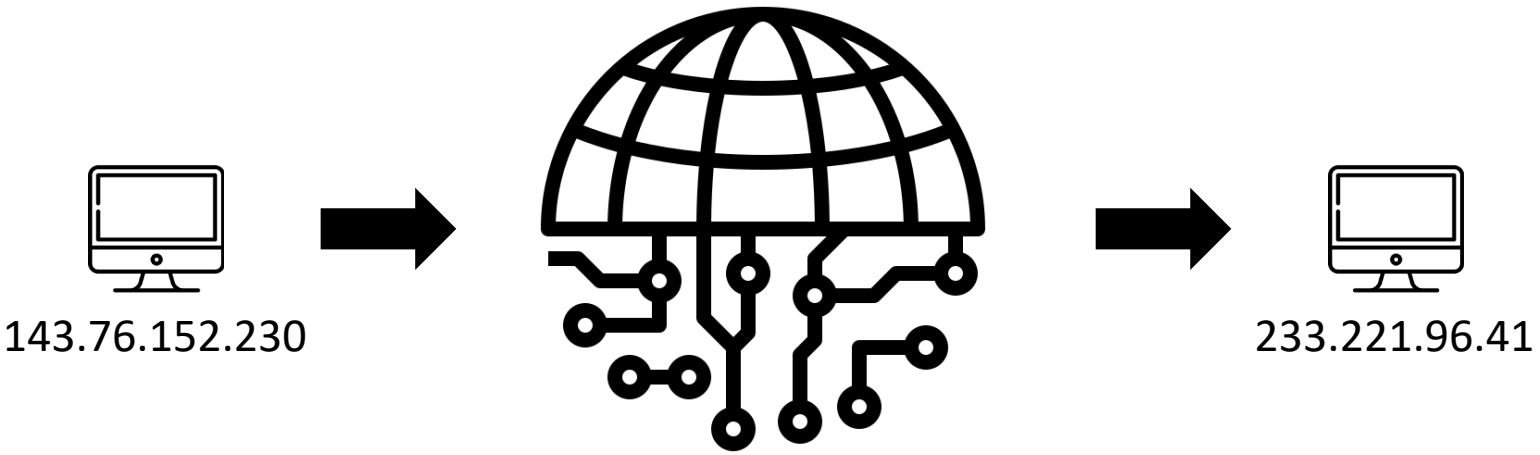
TCP/IP는 패킷 통신과 관련된 규약이다.

IP: Internet Protocol (패킷 통신 방식의 인터넷 프로토콜)

- IP address로 데이터를 전송하기 위해 지켜야 할 규약
- 패킷의 전달여부는 보증하지 않는다
- 패킷을 보낸 순서와 받는 순서가 다를 수 있다

TCP: Transmission Control Protocol (전송 조절 프로토콜)

- Network를 통해 패킷을 전달할 때, 지켜야 할 규약
- 패킷을 조립하고, 손실된 패킷을 확인한 뒤, 재전송을 요청하는 프로토콜
- IP 위에서 작동하는 프로토콜



What is TCP/IP?

Application Layer
- 사용자와 컴퓨터의 인터페이스
- HTTP, DNS, FTP, SSH, LDAP, …

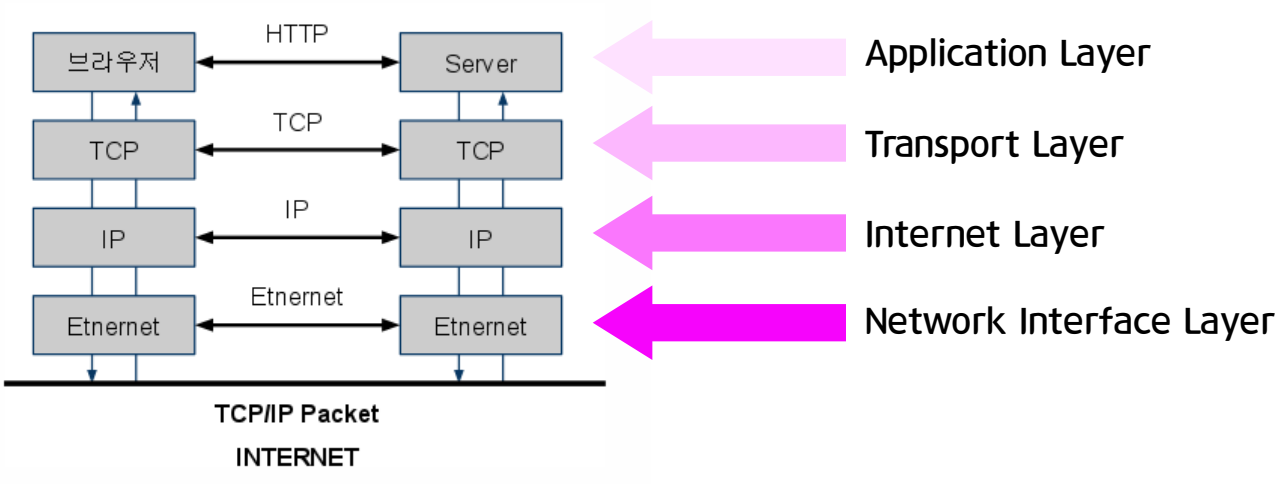
Transport Layer
- 데이터를 쪼개 TCP 패킷으로 만든다.
- TCP 프로토콜을 사용하여 패킷을 전송한다.
- TCP, UDP, SCTP, …

Internet Layer
- IP 프로토콜을 이용해, Client가 네트워크상 어디에 위치하는지를 찾는다
- 패킷전송의 최적 경로를 찾는다
- IPv4, IPv6, ARP, …

Network Interface Layer
- 네트워크와 관련된 하드웨어
- 이더넷, 토큰 링, FDDI, …

What is TCP/IP?

1. 크롬 브라우저에 www.sparcs.org와 같은 URL을 입력하여 웹페이지를 요청한다.
2. 요청은 인터넷 상으로 전달되기 위해 TCP 패킷의 형태로 만들어진다.
3. 이 TCP 패킷은 인터넷 상에서 Server로 전해지기 위해, IP 패킷으로 만들어진다.
   IP패킷에는 자신의 IP address, Server의 IP address 정보가 포함되어있다.
4. TCP/IP 패킷은 이더넷 카드로 보내져 Internet 상으로 나가게 된다.
5. Internet 상에서 패킷은 최적경로를 통해 Server로 보내지게 된다.
6. www.sparcs.org 서버의 이더넷카드에 도착한 패킷은 IP패킷을 분석하여, 어디에서 보냈으며, 도착지가 맞는지를 분석한다.
7. 도착지가 자신이 맞다면, TCP 프로토콜을 이용해 메시지가 누락된게 있는지, 순서를 재조합하여 Data를 완성한다.
8. Data를 Application Layer로 보내고, 이를 검사하여 이에 맞는 웹페이지 Data를 Client로 보내기 위해, …(1번부터의 과정을 반복한다.)



FTP(File Transfer Protocol)

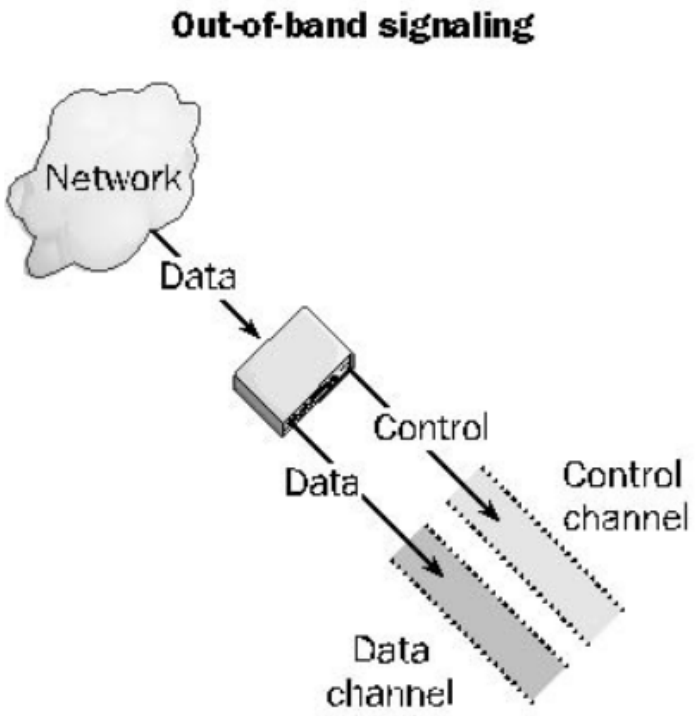
FTP: File Transfer Protocol: 파일을 전송하기 위한 프로토콜
- FTP는 TCP를 이용하는 application layer의 프로토콜

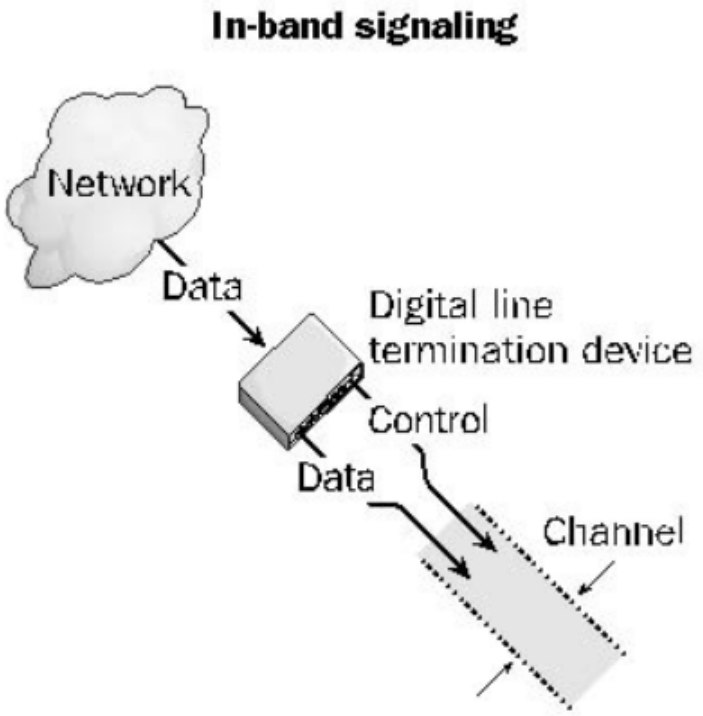
FTP와 HTTP(HyperText Transfer Protocol)의 차이점
- FTP는 HTTP보다 먼저 만들어졌다.
- FTP는 HTTP에 비해 비용(overload)가 적다.(서버의 부하가 적다)
  파일 전송만을 목적으로 하기 때문에, 컨트롤에 필요한 데이터가 적다.
  Out-of-band
  데이터를 전송할 때, 아주 짧은 헤더 혹은 노 헤더로 데이터를 전송한다.
- FTP의 이어받기 기능: 전송 도중, 인터넷 연결이 끊겨도, 전송되던 중간부터 파일전송을 이어서 시작한다.
- FTP Client와 FTP Server에서 양방향으로 파일을 전송할 수 있다. 반면, HTTP는 Server에서 Client로만 웹페이지(파일)을 전송할 수 있다.

What is out-of-band?

Out of band

- 파일을 전송하는 data channel과 전송 제어를 위한 control channel이 다른 것
- 데이터를 전송하기 위해 사용되는 대역/채널/포트/연결과 제어를 위해 사용되는 대역/채널/포트/연결이 다르다.
- In-band에서는 data channel과 control channel이 동일하다.





What is out-of-band?

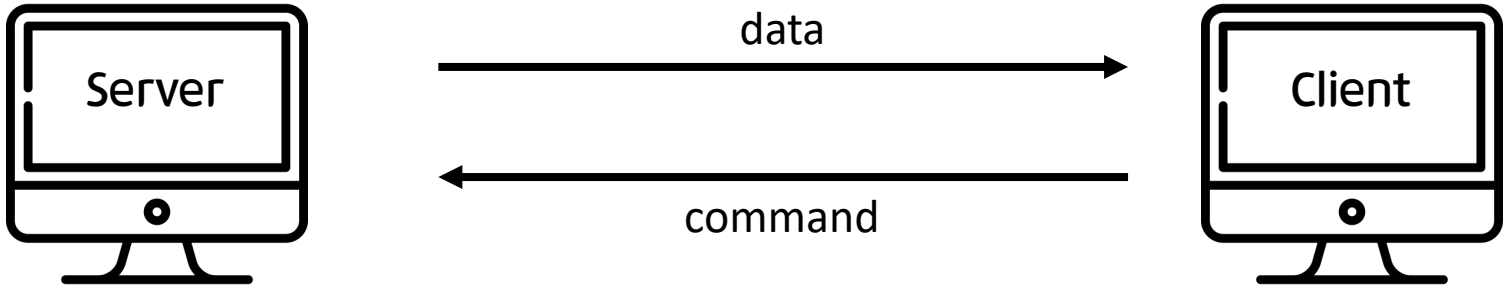
Control Channel
- Client, Server 사이에 서비스 요청 및 결과 통보를 알리기 위한 명령들이 오가는 channel
- 사용자 계정, 암호의 정보, 파일전송 명령, 파일전송의 결과 등등
- FTP 연결의 시작부터 파일 전송이 끝날 때까지 연결되어 있다.
- Server의 21번 포트가 사용된다.

Data Channel
- 실제 파일의 데이터를 전송할 때 사용하는 channel
- 데이터를 전송할 때마다 새로 연결된다. (한 파일의 전송이 끝나면, Data channel이 닫힌다. 그리고 다른 파일의 전송이 시작되면, 다시 연결된다.)
- Server의 20번 포트가 사용된다.
- Data Channel의 연결 방식으로는 Active mode와 Passive mode가 있다.

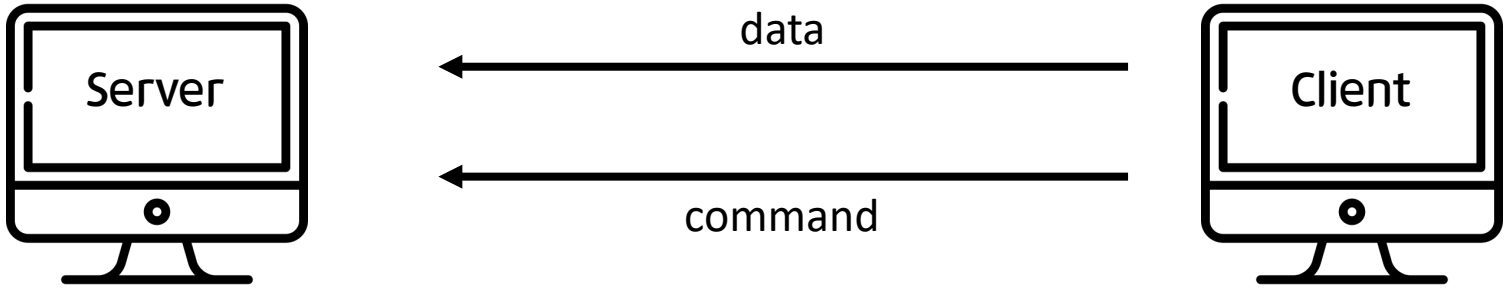
What is out-of-band?

FTP connection: Active mode와 Passive mode의 차이점
-> data channel connection 요청 방향의 차이
-> 다시말해, “data channel을 연결합시다”라는 요청을 누가 보내는가 의 차이이다.

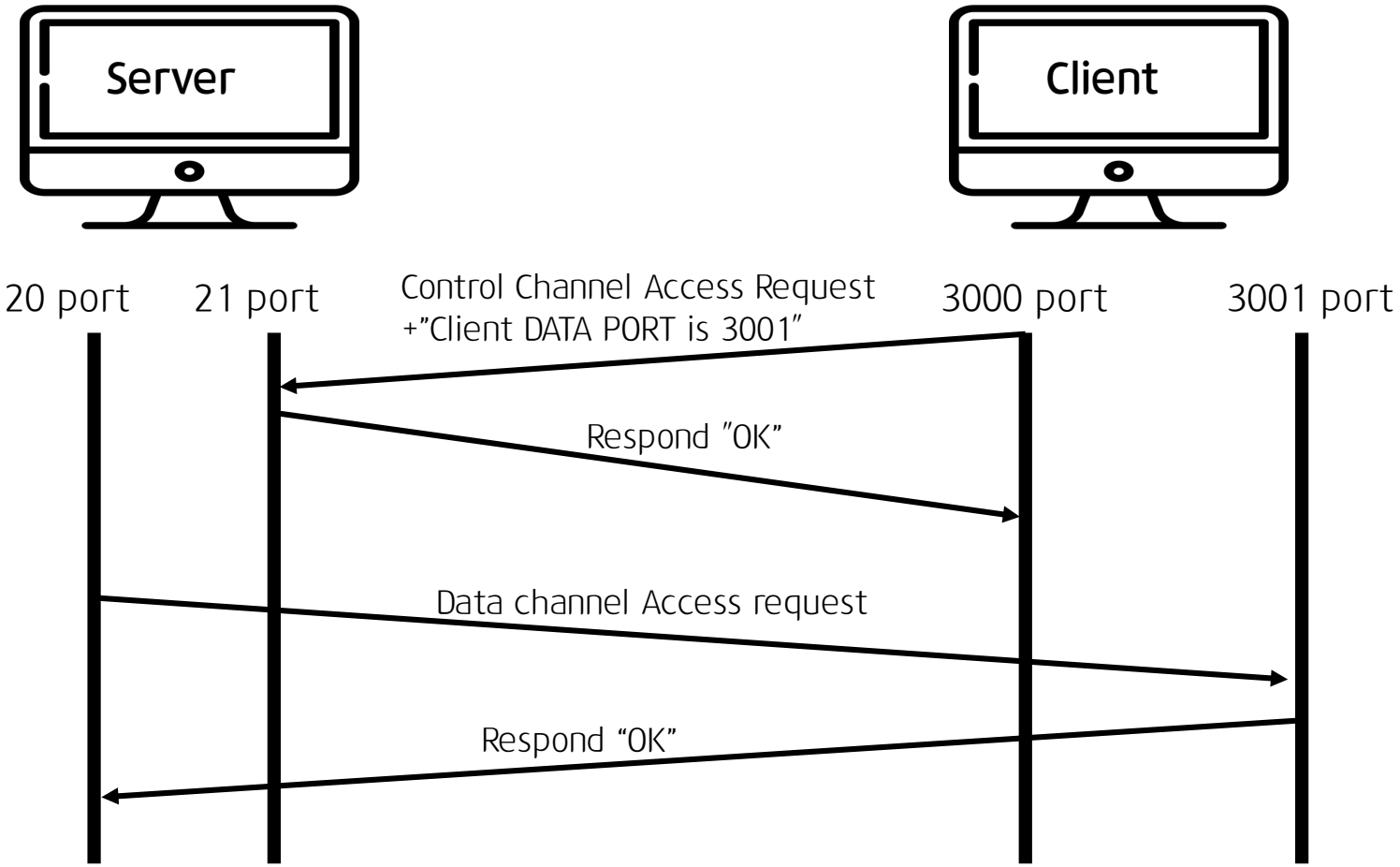
1. Active Mode



2. Passive Mode



Active mode

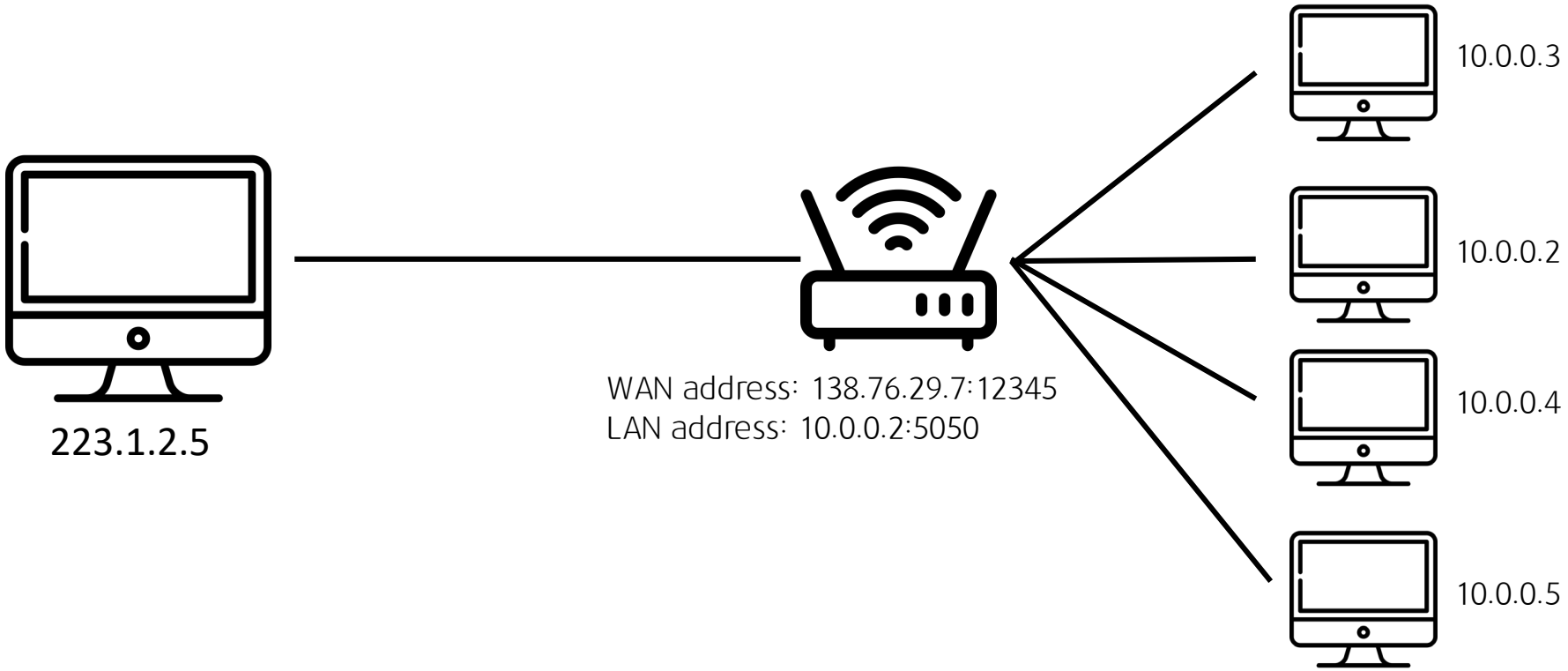


1. FTP Client의 3000번 포트에서 Server의 21번 포트로 Control channel 연결 요청
2. Server에서 Client로 OK 응답하고, Control channel 연결 완료
3. Server의 20번 포트에서 Client의 3001번 포트로 Data channel 연결 요청
4. Client가 Server에 "OK" 응답을 보내고, Data channel 연결 완료

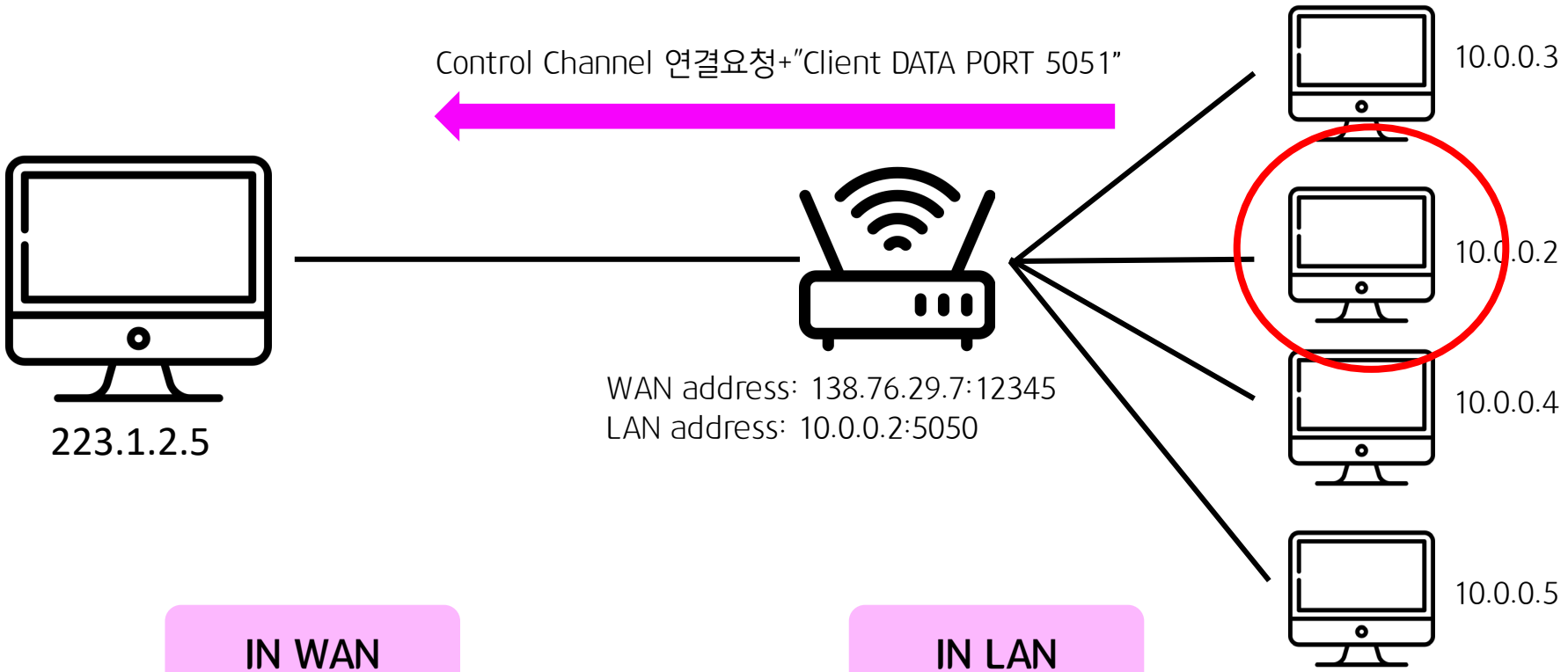
Active mode

1. Data channel 연결 요청: Server -> Client
2. Client의 방화벽에 20번 포트가 차단되어 있다면, 데이터 채널의 연결이 불가능하다.
3. Client의 방화벽에서 20번 포트가 INPUT으로 설정 되어 있고, Server의 방화벽에서 20번 포트가 OUTPUT으로 설정되어 있어야 한다.

하지만, NAT(공유기)에서 방화벽 문제로 작동되지 않을 수도 있다.



Problem of Active mode with NAT



IN WAN

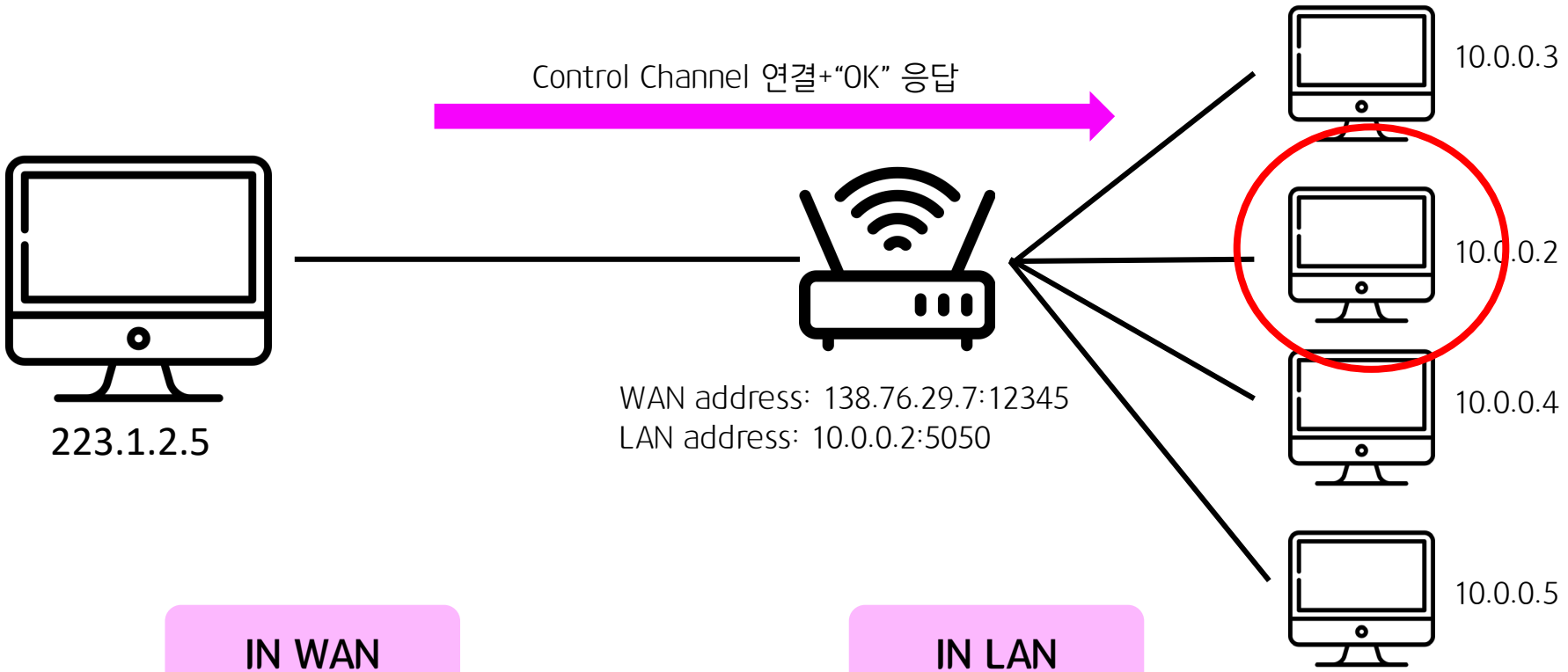
Src IP: 138.76.29.7
Dst IP: 223.1.2.5
Src PORT: 12345
Dst PORT: 21
DATA: "data channel PORT is 5051"

IN LAN

Src IP: 10.0.0.2
Dst IP: 223.1.2.5
Src PORT: 5050
Dst PORT: 21
DATA: "data channel PORT is 5051"

1. Client에서 Server로 Control channel 연결 요청을 보낸다.

Problem of Active mode with NAT



IN WAN

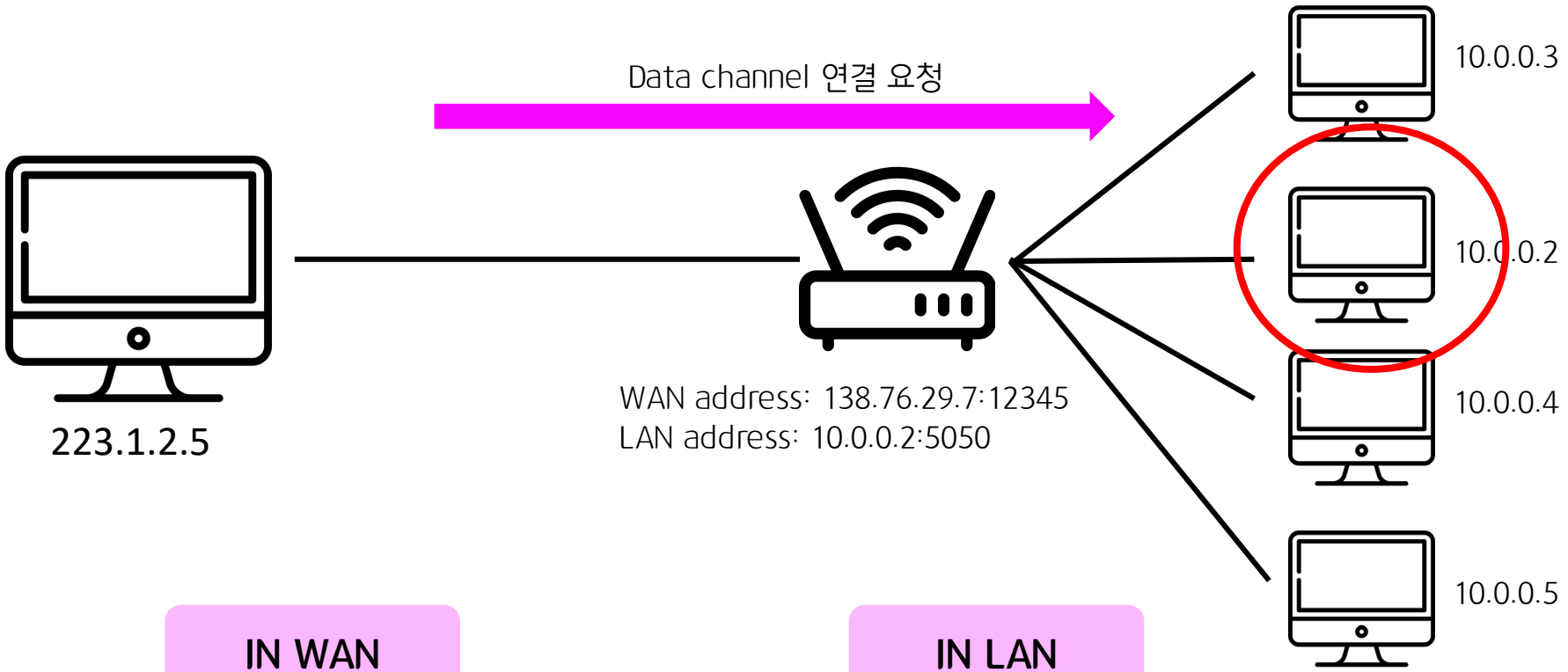
Src IP: 223.1.2.5
Dst IP: 138.76.29.7
Src PORT: 21
Dst PORT: 12345
DATA: "OK"

IN LAN

Src IP: 223.1.2.5
Dst IP: 10.0.0.2
Src PORT: 21
Dst PORT: 5050
DATA: "OK"

2. Control channel 연결. Server에서 Client에게 "OK"응답을 보낸다.

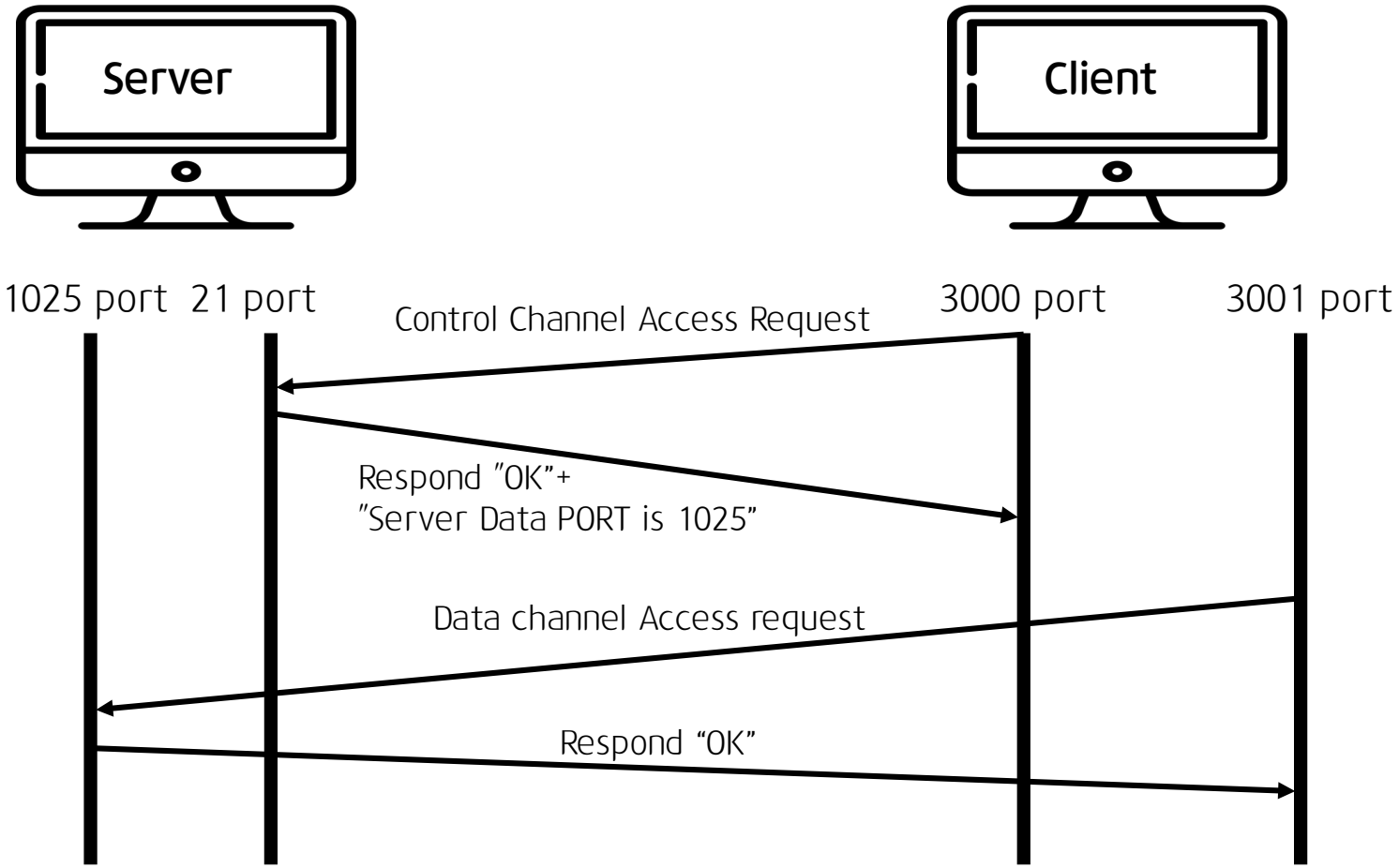
Problem of Active mode with NAT



IN WAN	IN LAN
Src IP: 223.1.2.5 Dst IP: 138.76.29.7 Src PORT: 21 Dst PORT: 5051 DATA: “OK”	??

3. Server에서 Client로 Data channel 연결 요청을 보낸다. -> 문제발생…

Passive mode



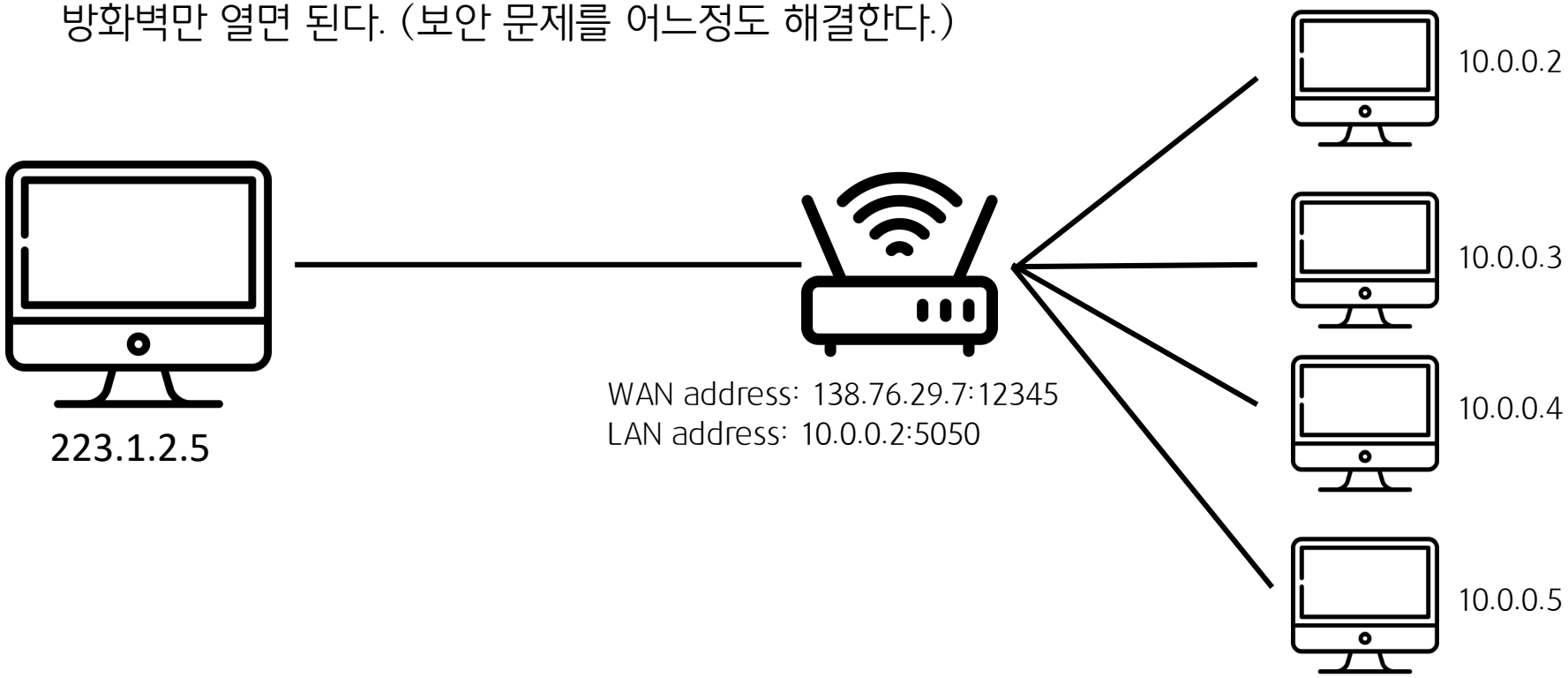
1. FTP Client의 3000번 포트에서 Server의 21번 포트로 Control channel 연결 요청
2. Server에서 Client로 OK 응답하고, Control channel 연결 완료
3. Client의 3001번 포트에서 Server의 1025번 포트로 Data channel 연결 요청
4. Server가 Client에게 "OK"응답을 보내고, Data channel 연결 완료

Passive mode

1. Data channel 연결 요청: Client -> Server
2. Active mode와는 다르게, NAT에 의한 문제를 막을 수 있다.
3. Server의 방화벽에 Client의 Data channel PORT가 차단되어 있으면, 데이터 채널을 연결할 수 없다.

-> 서버의 모든 포트 방화벽을 열어서 해결.(보안 문제…)

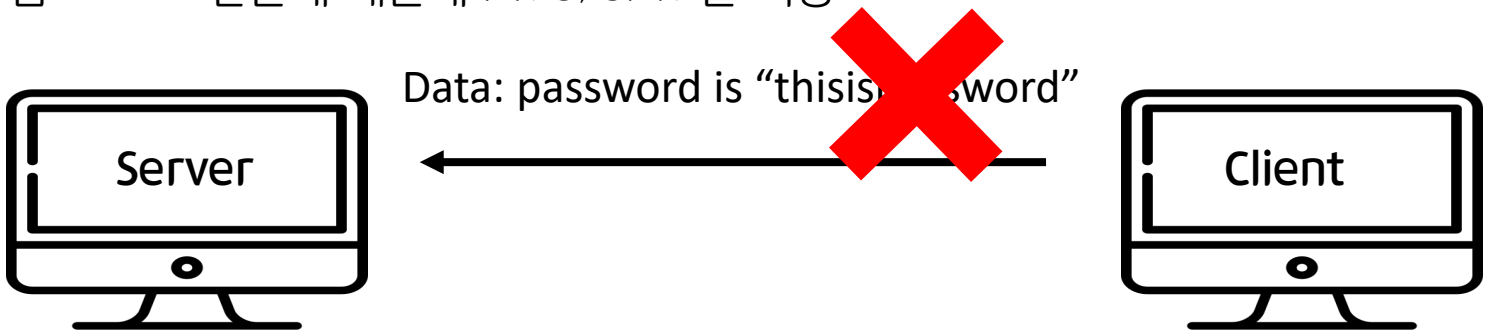
-> FTP Daemon: Client의 Data channel PORT를 제한하면 Server에서는 지정한 포트의 방화벽만 열면 된다. (보안 문제를 어느정도 해결한다.)



FTP

FTP는 HTTP와 다르게, 인증을 해야 한다.(Control channel을 연결하는 과정에서 인증한다.)

인증을 위해서는 Client에서 Server로 Username과 함께 Password를 보내야 한다.
-> 그럼 요청을 보낼 때, Data에 Password를 그냥 Plain Text로 보내면 되나??
-> 그건 좀… -> 보안문제 때문에 FTPS, SFTP를 이용



Anonymous FTP
- Username이 필요 없이 Access할 수 있는 FTP
- 공식적으로 Username, Password 가 필요 없는 FTP Server
- ‘Anonymous’라는 공용계정으로 모두가 Access

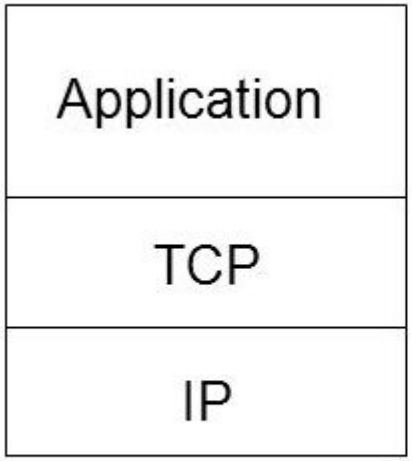
FTPS

FTPs: FTP Secure 또는 FTP on SSL

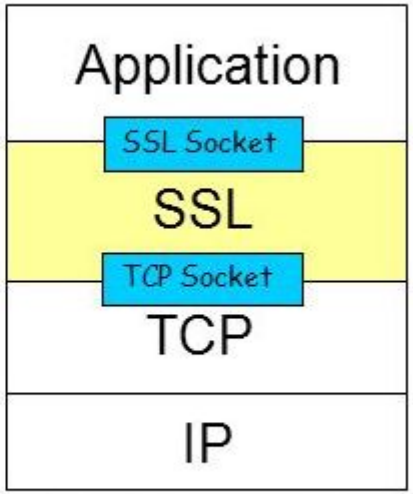
- SSL위에 돌아가는 FTP를 FTPS라고 한다.
- Authentication: 권한이 있는 사람만 접근 가능하다
- Encryption : 암호화 되어있다.
- Integrity: 데이터가 전송과정에서 변형되지 않는다. 권한이 없는 사람에 의해 데이터가 변형되지 않는다.
- 하지만 FTPS는 방화벽에서 막힌다. FTP의 control connection이 암호화된다면, 방화벽의 입장에서는 data channel로 어떤 포트를 열어야 할지 결정하기 어렵다. 이로 인해, FTPS는 방화벽 문제로 실패하는 경우가 대다수이다.

What is SSL?

SSL: Secure Socket Layer
HTTPS, FTPS 등의 통신 Layer에는 SSL이 포함되어 있다.
자세한 내용은 Security 세미나에서 배울 것입니다.



normal application

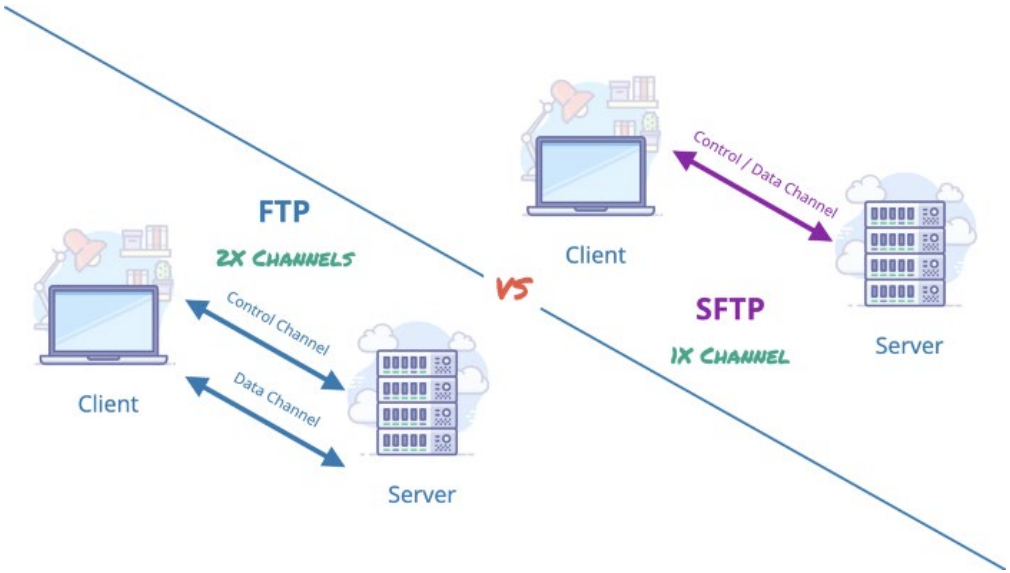


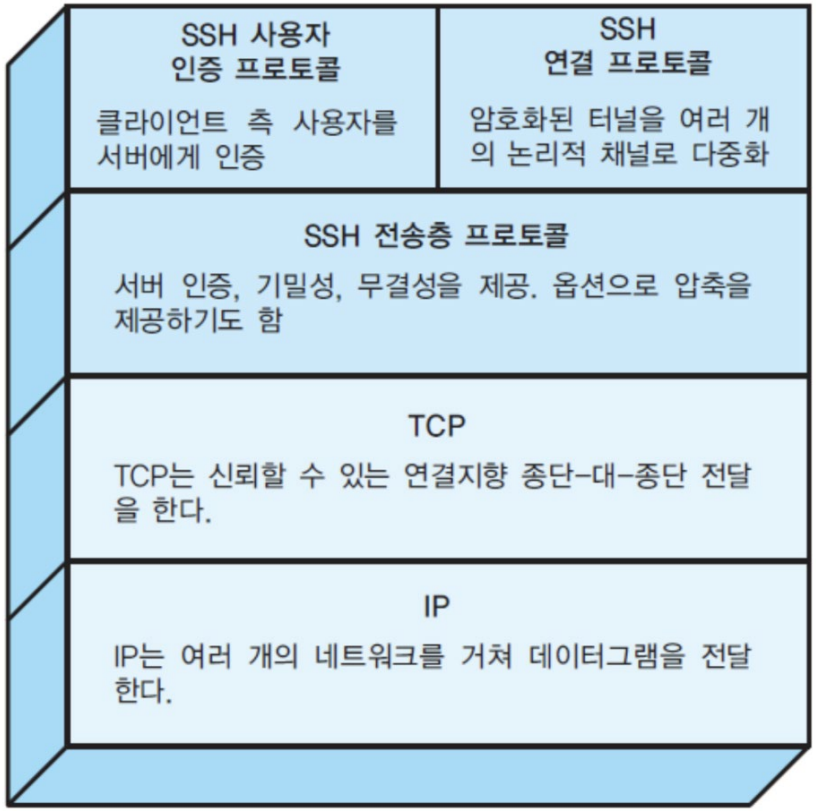
application with SSL

SFTP

SFTP: SSH FTP, Secure FTP

- SSH(Secure Shell)로 FTP의 기능을 구현하였다.
- 이름에는 FTP가 들어가나, 실상은 SSH이다.(SFTP는 In-band signaling을 사용)
- SSH를 통해 구현하였기 때문에, data channel의 port는 22번이다.
- 오른쪽 그림이 SFTP를 이용했을 때의 통신 Layer이다.





FTP 실습 준비

FTP Server Daemon
- Wu-ftpd: 보안문제 취약
- ProFTPD: 널리 사용되고 있는 FTP 서버 프로그램.
- vsftpd: Very Secure FTP Daemon. IPv6, FTPS 지원.

-> 실습에서는 vsftpd를 사용할 것이다.

우선, 앞서 NFS 실습을 할 때 만들었던 aws 인스턴스 nfs-serve를 재활용하자.
ssh로 nfs-server 인스턴스에 접속한 뒤,
$ sudo apt update
$ sudo apt install vsftpd

#만약, sftp 의 옵션을 변경하고 싶다면,
$ sudo vim /etc/vsftpd.config

FTP 실습 준비 – vsftpd Options

1. listen
   - YES: standalone
   - NO: xinetd
2. anonymous_enable
   - YES: allow anonymous FTP
   - NO: Full service ftp: Access시, Username, Password 필요
3. local_enable
   - YES: local user의 접속 허용
   - NO: local user의 접속 허용X
4. write_enable
   - YES: User에게 write permission 부여
   - NO: User에게 write permission 부여 X
5. pasv_enable
   - YES: Passive mode 사용

FTP 실습 준비 – Standalone vs Xinetd

Standalone

항상 실행되고 있는 데몬

/etc/init.d

메모리 부하

응답속도 빠름

Sendmail, apache, mysql,
웹서버, ...

Xinetd

요청이 들어오면 준비하는 데몬

/etc/exinetd.d

메모리 효율

응답속도 느림

TELNET(원격 접속)

FTP 실습 – FTP Server

FTP Server – vsftpd commands

$ sudo service vsftpd start

$ sudo service vsftpd stop

$ sudo service vsftpd restart

$ sudo service vsftpd status

```
ubuntu@ip-172-31-44-89:~$ sudo service vsftpd status
● vsftpd.service - vsftpd FTP server
   Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2020-07-19 15:20:08 UTC; 3min 51s ago
  Process: 13513 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty (code=exited, status=0/SUCCESS)
 Main PID: 13524 (vsftpd)
    Tasks: 1 (limit: 1121)
   CGroup: /system.slice/vsftpd.service
           └─13524 /usr/sbin/vsftpd /etc/vsftpd.conf

Jul 19 15:20:08 ip-172-31-44-89 systemd[1]: Starting vsftpd FTP server...
Jul 19 15:20:08 ip-172-31-44-89 systemd[1]: Started vsftpd FTP server.
```

FTP 실습 – FTP Client

FTP Client
# ftp server 콘솔창을 여는 명령어, 이때 key가 필요하다면 –i 옵션 사용
$ sftp [ftp_server_IP]

```
jinhochoi@DESKTOP-2BUMP27:~$ sftp --help
unknown option -- -
usage: sftp [-46aCfpqrv] [-B buffer_size] [-b batchfile] [-c cipher]
          [-D sftp_server_path] [-F ssh_config] [-i identity_file]
          [-J destination] [-l limit] [-o ssh_option] [-P port]
          [-R num_requests] [-S program] [-s subsystem | sftp_server]
          destination
jinhochoi@DESKTOP-2BUMP27:~$ sftp -i "wheel_aws_key.pem" ubuntu@52.78.181.206
Connected to 52.78.181.206.
sftp> ls
nfs_test
sftp>
```

FTP 실습 – FTP Client

FTP Client에서 ftp server 콘솔창을 열었다면,
#기본 명령어(ls, cd, pwd, mkdir, chmod)는 리눅스와 동일하다.

$ get [file] # 파일 가져오기
$ delete [file] # 파일 지우기
$ put [file] # 파일 올리기
$ quit # 콘솔 탈출

```
jinhochoi@DESKTOP-2BUMP27:~$ sftp -i "wheel_aws_key.pem" ubuntu@52.78.181.206
Connected to 52.78.181.206.
sftp> ls
nfs_test
sftp> get nfs_test
Fetching /home/ubuntu/nfs_test/ to nfs_test
Cannot download non-regular file: /home/ubuntu/nfs_test/
sftp> cd nfs_test
sftp> ls
a.txt  b.txt  c.log
sftp> get a.txt
Fetching /home/ubuntu/nfs_test/a.txt to a.txt
sftp>
```

```
jinhochoi@DESKTOP-2BUMP27:~$ ls -l a.txt
-rw-rw-r-- 1 jinhochoi jinhochoi 0 Jul 20 01:05 a.txt
```

FTP 실습과제

FTP Server(Container)에서 /home/sparcs/testftp/test.txt 파일을 FTP를 통해 다운로드하자.

Container의 SSH PORT: -P 45305 sparcs@ssal.sparcs.org
** sftp --help를 자세히 보면, SSH port를 옵션으로 명시해줄 때, 대문자 P를 사용한다.
**** Sparcs Whale Container에 접근하기 위해서는, ubuntu_sparcs_privkey가 필요합니다!**

다운로드 한 뒤, 로컬에서(혹은 aws 인스턴스에서) 다운로드 한 파일을 cat 명령어를 통해 출력한 뒤, 결과를 캡처한다.

conatiner의 자세한 내용은 https://whale.sparcs.org/#/containers/ 에서
Container Name: ball_wheel_seminar_ftp 를 확인해주세요!!

FTP 실습과제

```
Connected to ssal.sparcs.org.
sftp> ls
testftp     workspace
sftp> cd testft[p
Couldn't stat remote file: No such file or directory
sftp> cd testftp
sftp> ls
test.txt
sftp> pwd
Remote working directory: /home/sparcs/testftp
sftp> get test.txt
Fetching /home/sparcs/testftp/test.txt to test.txt
/home/sparcs/testftp/test.txt                                 100%   65     2.8KB/s   00:00
quit
jinhochoi@DESKTOP-2BUMP27:~$ cat test.txt
```

Reference

[1] 2015~2019 Wheel Seminar- NFS&FTP
[2] About SSL: https://oaksong.github.io/2018/05/04/ats-https/
[3] About SSL: https://opentutorials.org/course/228/4894
[4] About TCP/IP: https://www.joinc.co.kr/w/Site/Network_Programing/Documents/IntroTCPIP
[5] About Active/Passive mode:
https://m.blog.naver.com/PostView.nhn?blogId=leekh8412&logNo=100152842034&proxyReferer=https:%2F%2Fwww.google.com%2F
[6] About FTP: https://en.wikipedia.org/wiki/File_Transfer_Protocol#Communication_and_data_transfer

Q&A