**Ⅰ. Principle of network application**

- Network Application Development : 다른 종단 시스템에서 동작하고 네트워클 통해 서로 통신하는 프로그램을 작성하는 것.

Creating a network application

- Network를 이용하여 서로 의사소통(Communicate)하며, 각기 다른 종단 시스템(End System, host)에서 실행되고 있다.

- No need to write software for network-core device( ex) router ) : 네트워크 코어에서 사용되는 소프트웨어의 작성은 필요하지 않다.

- Network 코어에서는 Application이 실행되지 않는다

- 종단 시스템(End System, host)에서의 Application은 development,

propagation[[1]](#footnote-1)시켰다.

**1. Network Application Architecture**

- 네트워크 개발자에 의해 설계되고, 다양한 종단 시스템에서 어떻게 조직되어야 하는지를 지시한다.

* 1. **Client Server Architecture**

- Server : Always-on host / permanent[[2]](#footnote-2) IP Address / Server farms for scaling[[3]](#footnote-3)

- Client : may be intermittently[[4]](#footnote-4) connected(때때로 연결한다)/ may have dynamic Ip address / do not communicate directly with each other

* 1. **Peer to Peer Architecture(P2P)**

- no always –on server / arbitrary[[5]](#footnote-5) end system directly communicate / Highly scalable but difficult to manage(높은 확장성을 가진다(self-scalabilty)/ 관리어려움)

- 단: ISP우호적( Up/Down stream 비대칭 대역폭 부하야기)/ 보안 / 보상(incentive)

* 1. **Hybrid of client-server and P2P**

- Skype : voice-over-IP / centralized[[6]](#footnote-6) server(IP를 찾고 연결해줌) / client-client (P2P)

- Instant Message : chatting between two user / centeralized server(IP주소 추적) / client-client(P2P)

**2. Process Communicating**

- 실제로 실행되는 것은 Program이 아닌 Process이고 이것으로 통신한다.( inter – process communication [ called by IPC ])

- 두 개의 서로 다른 종단 시스템(end System, host)에 메시지(Message) 교환으로 통신

**2.1 Client Server Process**

- Client Process : 통신을 초기화 하는 프로세스 ( initiate )

- Server Process : 세션을 시작하기 위해 접속을 기다리는 프로세스 (contacted)

**2,2 Interface between Computer Network and Process - Socket**

- Socket을 이용하여 서로 메시지를 주고 받는다

- Process (House) – Socket(Door)

- Socket Called API[[7]](#footnote-7) ( Network Application이 Internet에 만든 Programming Interface로 Application 과 네트워크 사이라는 의미) : Choice of transport protocol(TCP/UDP를 선택) / 최대 Buffer/ 최대 세그먼트 크기 매개변수의 설정

**2.3 Addressing Process(주소화 프로세스)**

- to receive message , process must have **identifier**

- 호스트 주소(IP Address 32bit)와 그 목적지 호스트 내의 수신 프로세스 식별자(Port) 존재

**3. What transport service does an app need? (애플리케이션이 이용 가능한 트랜스포트 서비스)**

- Transport Protocol은 Network를 통해 소켓으로 받은 메시지를 수신 소켓으로 이동시키는 책임을 가진다.

- Application 개발 하기 전에 생각해야 할 것들

**3.1 Reliable Data Transport**

- 프로세스간 신뢰적 데이터 전송을 제공한다(메일 / 파일전송) ↔ Unreliable 하다면 손실 허용 애플리케이션 (loss-tolerant application) 이라 부른다. (실시간 비디오/오디오)

**3.2 Throughput[[8]](#footnote-8)**

- Bandwidth-sensitive application(대역폭 민감 애플리케이션) ↔ Elastic application

**3.3 Timing**

- 실시간 어플리케이션과 같은 것들.

**3.4 Security**

- Encryption[[9]](#footnote-9)

**4. Transport service requirements of common app (인터넷 전송 프로토콜이 제공하는 서비스)**

- UDP(User Datagram Protocol), TCP(Transmission Control Protocol) 두 개중 하나를 선택해야 한다.

**4.1 TCP**

- Connection-oriented(연결지향형) – Handshaking 작업을 한다. , Reliable transport(신뢰적인 데이터 전송 서비스) , Flow Control, Congestion[[10]](#footnote-10) Control, does not provide : timing/ minimum throughput guarantees, security

**4.2 UDP**

- 불안전한(Unreliable) 데이터 전송 - Handshaking 작업을 하지 않는다, does not provide : connection setup / reliability/ flow control / congestion control, timing, throughput guarantee / security

**4. Application Layer Protocol defines**

- Type of message exchanged ( 교환 메시지 타입 : 요청 / 응답 메시지)

- Message syntax ( 여러 메시지 타입의 문법 : 내부의 필드와 필드간의 구별 방법)

- Message semantics (필드의 의미 정보의 의미)

- Rules for when and how processes send & respond to message ( 언제 ,어떻게 프로세스가 메시지를 전송하고 메시지에 응답하는지 결정하는 규칙

- Public domain protocol

Defined in RFC / allow for interoperability / HTTP, SMTP

- Proprietary protocol

독점적인 애플리케이션 계층 프로토콜 / 공중도메인에 없음 / 계층 프로토콜 이용: Skype

**Ⅱ. Web and HTTP**

- 웹 On-demand 방식 : 그들이 원할 때, 원하는 것을 수신한다.

- 웹은 두가지 Server/Client로 구현된다.

- 서로 HTTP Message를 교환한다.

- Web page는 Object(객체)로 구성된다.

- 객체는 HTML / JPEG / Java applet 이 될 수 있다.

- 각 객체는 각자의 URL Address를 가진다.

francis.epril.net / someidt / abc.gif

----host name--- -----path name---

**1. HTTP[[11]](#footnote-11) 개요(HTTP Overview)**

Client – web browser ( Web page) 객체를 보여주는 역할

Server – URL로 각각을 지정할 수 있는 웹 객체를 가진다.

- TCP 전송 프로토콜 사용

- 비상태 프로토콜 이라고 한다(Stateless Protocol)- 서버는 클라이언트에 대한 정보를 기억하지 않는다.

- 연결 방법

1. TCP Connection 을 연결한다. → 2, 서버가 TCP Connection을 Client로부터 받는다.

→ 3. HTTP Message를 browser와 Web Server 사이에 교환 한다. → 4. TCP. 종료 한다.

**2. 비지속 연결(Nonpersistent) 과 지속연결(Persistent)**

**2.1 Nonpersistent HTTP**

- 분리된 TCP 연결을 통해 보내져야 하는가

EX) 하나의 URL에 1개 HTML과 10개의 JPG가 있다면 francis.epril.net / index.html

1. francis.epril.net Server에 TCP 연결 시도 → 2. Client 연결된 소캣으로 HTTP 요청메시지 전송 index.html (Path Name) 포함 → 3. SeverHTTP요청 메시지 받음 index.html 객체 추출 – 객체를 캡슐화 클라이언트 전송 → 4. Server TCP 끊으라고 요청 (TCP Clietn가 응답 을 받을 때 까지 끊지는 않음 ) → 5. 응답메시지 받으면 연결 끊음 응답 메시지에서 HTML 파일 조사 10 개 찾음 → 6. 다시 1~5반복 10개에 대해서

- 총 11한 수행함 ( 동시 수행 여부는 클라이언트(브라우저)에서 조절가능)

- RTT(round trip time) : 작은 패킷이 클라이언트로부터서버까지 가고 다시 클라이언트로 되돌아 오는데 걸리는 시간. ( 패킷 큐잉지연, 처리지연을 포함)

- 세방향 핸드세이크를 포함한다(방금 처리과정) : 처음 두 부분 하나의 RTT가 걸린다 따라서 두개의 RTT가 생성되어 2RTT + (HTML ,JPEG)파일전송걸리는 시간 이 걸린다.응답시간은

단점: 각 요청 객체에 대한 새로운 연결이 설정되고 유지되어야 한다.

TCP 버퍼가 할당되고, TCP 변수들이 클라이언트와 서버 양쪽에 유지되어야 한다. 과부하

2RTT가 필요하다.

**2.2 Persistent**

- 모든 요구와 해당하는 응답들이 같은 TCP 연결상으로 보내져야 하는가.

- 진행 중인 요구에 대한 응답을 기다리지 않고 연속해서 만들어 질수 있다(Pipe Lining )

- 일정시간 사용하지 않으면 서버의 연결을 종료시킨다.

- RTT가 적게든다.

**3. HTTP Message Format**

**2.1 Request Message**

- ASCII 텍스트로 쓰여 있어 사람들이 읽을 수 있다.

- 각 줄은 CR(Carriage return) LF(line feed)로 구별된다.

/somedir/page.htm (객체 여청 path)

( GET, POST, PUT, DELETE, HEAD) –Called HTTP Version Field

GET /somedir/page.html HTTP/1.1 ( Called Request Line - )

HOST : [www.someschool.edu](http://www.someschool.edu)

User-agent : Mozilla/4.0

Connection : close ----- (Called Header line)

Accep-language : fr

(CR/LF)

[ ----------------Here-----------------] POST 방식 사용 entity body개체 몸체)

// HEAD 인 (get과 유사 경우 요청 객체를 받지 않음) 디버깅용

// PUT( file 수정) , DELETE(file 삭제)(

**2.2 Response Message**

‘Version Field’ , ‘ status Code‘ ‘status message‘

HTTP/1.1 200 OK // (called status line)

Connection close

Date: Thu, 06 Aug 1998 12:00:15 GMT

Server: Apache/1.3.0 (Unix)

Last-Modified: Mon, 22 Jun 1998 …... --- (called Header Line)

Content-Length: 6821

Content-Type: text/html

data data data data data.. . – (called entity body)

200 – OK(성공)

301 - moved Permantely(요청 객체가 이동되었다)

400 – Bad Request ( 요청할 수 없다)(

404 – Not Found ( 그런거 존재하지 않는다)

505 – HTTP version Not Supported( 요청 HTTP 프로토콜 버전을 서버가 지원하지않는다)

**4, Cookie( add more information 실행과정)**

- 서버는 사용자 정보를 저장하지 않는다 → 때때로 접속 제한/ 사용자에 따라 컨텐츠 제공 필요 → 쿠키를 이용하여 사용자를 추적

- 4가지 요소

1. HTTP 응답 메시지 쿠키 헤더라인

2. HTTP 요청 메시지 쿠키 헤더라인

3. 사용자의 브라우저에 사용자 종단 시스템과 관리를 지속시키는 쿠키 파일

4. 웹사이트의 백엔드데이터 베이스

쿠키 사생활 침해의 요점이된다 Cookies and privay

**5. Web caches( add more information 계산 실행 과정)**

- 웹 캐시 Proxy Server(프록시서벌)라고도 불리는것으로 기접 웹 서버(Origin Web Server)를 대신하여 HTTP요구를 충족시키는 네트워크 개체이다.

- 자체의 저장 디스크를 가지며, 최근 호출된 객체의 사본을 저장 및 보존한다.

- 웹 캐시는 서버와 클라이언트 역할을 모두 한다.

- 일반적으로 ISP가 구입하고 설치한다.

장 ) 1. 클라이언트의 요구에 대한 응답시간을 줄일 수 있다.

2. 인터넷으로의 기관 접속 회선상의 웹트래픽을 실질적으로 줄일수 있다.

3. 웹 캐시를 가진 고밀도 인터넷은 비록 콘텐츠 제공자가 저속도의 접속회선을 가지 느린서버에서 자신의 사이트를 운영하더라도 빠른 콘탠츠 분배를 위한 기반구조를 제공한다.

CDN(Content Distrubution Network) 공유 CDN와 지정 CDN?

**6. 조건부 Get(Conditional GET) ( add more information 실행과정)**

- 객체가 모두 최신임을 확인 하기 위해서 이를 사용한다.

(1) Get 방식을 사용하고

(2) If-Modified-Since 헤더라인을

포함한다면 조건부 GET이다..

**Ⅲ. FTP**

- Transfer file to/from remote host

- client : side that initiates transfrer

- server : remote host

port 21:

두개의 TCP 연결한다 (제어 연결Control connection ) Data connection 데이터 연결

Out-of band라고 부름 ↔ In band( HTTP, SMTP)

21번 port를 이용하여 TCP connection연결 후 ( 비밀번호 계정 입력 )

세션을 통해서 FTP 서버는 사용자에 대한 상태를 유지해야한다 따라서 동시 유지 전체 세션의 수를 제한한다.

20 으로 다운로드한다.

User username – 계정

PASS password – 비밀번호

LIST – 파일 리스트 보기

RETR filename – 파일 추출

STOR filename – 저장

331 – Username Ok, Password required

125 – Data Connection already open ; transfer starting

425 – Can’t open data connection

452 – Error writing file

**Ⅳ. Electronic Mail (SMTP, POP3, IMAP)**

- 비 동기적인 통신 매체 ( 사람들은 상대방 스케줄과 상관없이 그들이 편할 때 메시지를 보거나 읽는다. )

**1. 구성요소**

- 사용자 에이전트(MS Outlook, Apple mail) : 메시지를 읽고 응답하고, 전달하고 저장하고 구성하게 해주는 것

- 메일 서버 : 전자메일 기반구조의 중심을 형성 (수신자는 메일박스를 가지고 있음)

- SMTP[[12]](#footnote-12)

**2. SMTP?**

- TCP를 사용하며 Server 와 Client사이에 Email Message를 전송한다.

- 25 Port를 사용한다.

- 전송 3단계 handshaking → 메시지 전송 → 종료

- 명령어는 ASCII 코드로 되어있으며 . 코드와 문장으로 상태를 응답한다.

- Message는 7-bit ASCII로 되어있다.

**3. HTTP VS SMTP**

- HTTP 의 경우 Pull Protocol 이고 SMTP 는 Push Protocol 이다.

- 둘다 ASCII를 이ㅛ용하지만 SMTP는 7bit로 제한된다.

- HTTP는 응답메시지를 각 객체를 캡슐화 하지만 STMP는 모든 메시지의 객체를 한 메시지로 만든다.

**4. Format**

- 헤더라인고 몸체는 빈줄(CRLF)로 구분된다.)

- 모든 헤더는 Frrom / To를 가져야 하고/ Subject 같은 것을 더 가질 수있다.

키워드 : 값

- 몸체는 오직 ASCII 만 가능

**5. Protocol**

1.SMTP의 한계점

- 항상 양쪽의 서버가 켜져 있어야 한다. Local PC에 메일서버가 있는 경우? 항상 Local PC가 켜져있어야한다.(Push 방식임으로)

- 송신자의 사용자 에이전트로부터 송신자의 메일서버로 메일을 전송하는데 사용

- 송신자의 메일 서버로부터 수신자의 메일 서버로 메일을 전송하는데 사용

2. SMTP의 한계점의 대안

- 자신의 로컬 PC로 가져오는 것은 무조건 Pull 방식읻.

- 수신자의 메일 서버로부터 수신자의 사용자 에이전트에게 메일을 전송하는데 이용

2.1 POP3

- port 110 이용 TCP 연결

- 인증( 사용자 정보, 비밀번호 ) 트랜잭션 (사용자 에이전트 메시지를 받아옴 삭제를 위해 메시지에 표시하거나 그 삭제 표시르르 지우거나 통계) 갱신(클라이언트가 POP3 세션을끝내는 quit명령이 내려진 후에 일어남 메시지 삭제 표시된 것 삭제)

- +OK / -ERR user <사용자명> // pass <password>

- list, retr, dele, quti 만 사용가능로그인 이후

2.2 IMAP

POP3 보다 복잡함

만든이유- 사용자에게 원격 폴더를 생성하거나, 폴더에 메시지를 할당하는 수단을 제공해주기 위해 POP3 불가능

IMAP은 사용자 정보를 유지 시킨다 (POP3는 아님)

1. Propagation – 전파 증식 [↑](#footnote-ref-1)
2. Permanent – 영구적인 [↑](#footnote-ref-2)
3. 하나의 서버에서 모든 것을 처리할 수 없기 때문에 많은 수의 호스트를 가지는 Data center로 강력한 가상의 서버를 생성한다. [↑](#footnote-ref-3)
4. Intermittently – 간헐적으로 [↑](#footnote-ref-4)
5. arbitrary – 제멋대로의 [↑](#footnote-ref-5)
6. centralized – 집중형 [↑](#footnote-ref-6)
7. API – Application Programming Interface [↑](#footnote-ref-7)
8. Throughput – 처리량 [↑](#footnote-ref-8)
9. Encryption –암호화 [↑](#footnote-ref-9)
10. Congestion – 혼잡 [↑](#footnote-ref-10)
11. HyperText Transfer Protocol [↑](#footnote-ref-11)
12. Simple Mail Transfer Protocol [↑](#footnote-ref-12)