**프로토콜의 의미**

**프로토콜이란?**

컴퓨터 네트워크에서의 프로토콜은 서로 다른 기종의 컴퓨터끼리 통신하기 위해서 미리 정해놓을 통신 규약 및 통신 약속입니다.

**프로토콜의 기본 요소**

* 구문(Syntax)  
  전송하고자 하는 데이터의 형식(Format), 부호화(Coding), 신호 레벨(Signal Level) 등을 규정
* 의미(Semantics)  
  두 기기 간의 효율적이고 정확한 정보 전송을 위한 협조 사항과 오류 관리를 위한 제어 정보를 규정
* 시간(Timing)  
  두 기기 간의 통신 속도, 메시지의 순서 제어 등을 규정

프로토콜의 종류

|  |  |
| --- | --- |
| **계층** | **프로토콜** |
| 응용 | HTTP, SMTP, FTP, Telnet |
| 표현 | ASCII, MPEG, JPEG, MIDI |
| 세션 | NetBIOS, SAP, SDP, NWLinnk |
| 전송 | TCP, UDP, SPX |
| 네트워크 | IP, IPX |
| 데이터링크 | Ethernet, Token Ring, FDDI, Apple Talk |
| 물리 |  |

**프로토콜의 기능**

**단편화(Fragmentation)와 재 합성(Assembly)**

단편화: 송신 측에서는 긴 데이터 블록을 손쉽게 전송할 수 있도록 크기가 똑같은 작은 블록으로 나누어 전송  
재 합성 : 수신 측에서 쪼개진 작은 데이터 블록을 재 합성하여 원래의 메시지로 복원하는 기능

**캡슐화(Encapsulation)**

각 프로토콜에 적합한 데이터 블록을 만들려고 데이터에 정보를 추가하는 것  
플래그, 주소, 제어 정보, 오류 검출 부호 등을 부착하는 기능

**연결 제어(Connection Control)**

비 연결 데이터 전송(데이터 그램)과 연결 위주 데이터 전송(가상회선)을 위한 통신로를 개설· 유지· 종결하는 기능

**흐름 제어(Flow Control)**

데이터양이나 통신속도 등이 수신 측의 처리 능력을 초과하지 않도록 조정하는 기능

**오류 제어(Error Control)**

데이터 전송 중 발생할 수 있는 오류나 착오 등을 검출하고 정정하는 기능

**순서 결정(Sequencing)**

연결 위주의 데이터를 전송할 때 송신 측이 보내는 데이터 단위 순서대로 수신 측에 전달하는

기능

**주소 설정(Addressing)**

발생지, 목적지 등의 주소를 명기하여 데이터를 정확하게 전달하는 기능

**동기화(Synchronization)**

두 통신 객체의 상태(시작, 종류, 검사 등)를 일치시키는 기능

**다중화(Multiplexing)**

하나의 통신로를 여러 개로 나누거나 회선 여러 개를 하나의 통신로로 변환시켜 다수의 가입자가 동시에 사할 수 있도록 하는 기능

**전송 서비스(Transmission Service)**

통신 객체를 사용하기 쉽도록 별도로 추가 서비스(패리티 검사, 보안도, 서비스 등급, 우선순위 등)를 제공하는 기능

**프로토콜의 예시**

**HTTP (Hyper Text Transfer Protocol)**

포트번호: 80  
packet 사용 (header와 body로 구성)

HTTP는 클라이언트와 서버 사이에 이루어지는 요청/응답(request/response) 프로토콜이다.  
주로 W3 상에서 정보를 주고받을 수 있는 프로토콜이다.  
주로 HTML 문서를 주고받는 데에 쓰인다.  
예를 들면, 클라이언트인 웹 브라우저가 HTTP를 통하여 서버로부터 웹페이지(HTML)나 그림 정보를 요청하면, 서버는 이 요청에 응답하여 필요한 정보를 해당 사용자에게 전달하게 된다.  
이 정보가 모니터와 같은 출력 장치를 통해 사용자에게 나타나는 것이다.

**HTTPS (Hyper Text Transfer Protocol Secure)**

포트번호: 443

월드 와이드 웹 통신 프로토콜인 HTTP의 보안이 강화된 버전이다.  
HTTPS는 통신의 인증과 암호화를 위해 넷서 케이프 커뮤니케이션 오퍼레이션이 개발한 넷서 케이프 웹 프로토콜이며, 전자 상거래에서 널리 쓰인다.  
HTTPS는 소켓 통신에서 일반 텍스트를 이용하는 대신에, SSL이나 TLS 프로토콜을 통해 세션 데이터를 암호화한다.  
따라서 데이터의 적절한 보호를 보장한다.

**FTP (File Transfer Protocol)**

포트번호: 21

TCP/IP 프로토콜을 가지고 서버와 클라이언트 사이의 파일 전송을 하기 위한 프로토콜이다.  
파일 전송 프로토콜은 TCP/IP 프로토콜 테이블의 응용 계층에 속하며, 역사는 오래 되었지만 지금도 인터넷에서 자주 사용된다.  
최초의 FTP 클라이언트 애플리케이션들은 운영 체제가 그래픽 사용자 인터페이스를 갖추기 이전에 개발된 명령 줄 프로그램이었으며, 대부분의 윈도우, 유닉스, 리눅스 운영 체제에 현재도 기본 포함되어 있다.  
그 뒤로 수많은 FTP 클라이언트 및 오토메이션 유틸리티들이 데스크톱, 서버, 모바일 장치, 하드웨어용으로 개발되고 있으며, FTP는 웹 페이지 편집기와 같은 생산성 응용 프로그램들에 통합되고 있다.

**SFTP (Secure File Transfer Protocol)**

포트번호: 22

신뢰할 수 있는 데이터 스트림을 통해 파일 접근, 파일 전송, 파일 관리를 제공하는 네트워크 프로토콜이다.  
국제 인터넷 표준화 기구(IETF)가 보안 파일 전송 기능을 제공할 목적으로 시큐어 셸 프로토콜 (SSH) 버전 2.0의 확장으로 설계하였다.  
IETF 인터넷 초안에 따르면 이 프로토콜이 SSH-2 프로토콜의 문맥 안에 기술되어 있지만 전송 계층 보안(TLS)를 통하는 보안 파일 전송 프로그램이나 VPN 응용 프로그램의 관리 정보 전송과 같은 다른 수많은 응용 프로그램들 에도 사용할 수 있다고 언급되어 있다.  
이 프로토콜은 SSH와 같은 보안 채널을 통해 수행되는데, 이 말은 서버가 이미 클라이언트와 인증이 되어 있으면서, 클라이언트 사용자 증명을 프로토콜에 이용할 수 있는 상태여야 한다는 것을 뜻한다.

**MYSQL**

포트번호: 3306

MySQL에서는 접속할 때가든지 SQL문실행을 할 때의 갖은 처리에 있어서 클라이언트 ⇔ 서버 사이에 MySQL독자의 규칙에 따른 패킷 송수신을 한다.

**SSH (Secure Socket Layer)**

포트번호: 22

네트워크 상의 다른 컴퓨터에 로그인하거나 원격 시스템에서 명령을 실행하고 다른 시스템으로 파일을 복사할 수 있도록 해 주는 응용 프로그램 또는 그 프로토콜을 가리킨다.  
기존의 rsh, rlogin, 텔넷 등을 대체하기 위해 설계되었으며, 강력한 인증 방법 및 안전하지 못한 네트워크에서 안전하게 통신을 할 수 있는 기능을 제공한다.  
SSH는 암호화 기법을 사용하기 때문에, 통신이 노출된다고 하더라도 이해할 수 없는 암호화된 문자로 보인다.

**프로토콜 스택**

**프로토콜 스택이란?**

프로토콜 스택이란 데이터 통신에 활용되는 프로토콜의 구조에 관한 개념으로, 계층화 된 구조(스택 구조)로 모여 있는 프로토콜의 집합을 의미합니다.

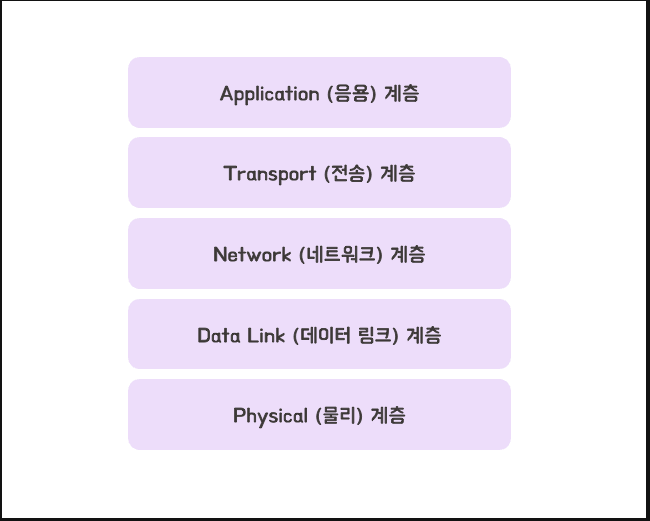
계층을 나누는 목적은 매우 복잡한 네트워크에서 프로토콜들의 역할을 분담하기 때문.

한 계층에 속하는 하나의 프로토콜이 인접한 계층의 다른 프로토콜과 통신

이러한 방식으로 꼭대기 계층에서부터 가장 아래에 있는 계층까지 또는 그 반대 방향으로 프로토콜 스택 내에서 통신이 됨.

컴퓨터 간의 물리적인 데이터 송수신은 가장 아래 계층에서 진행함.

**프로토콜 스택의 구조**



**Application(응용) 계층**

프로토콜: HTTP, FTP, SMTP, POP, Telnet

송수신측 사이에 어떤 종류의 (사용자가 사용하는) 데이터를 주거나 받기 위한 서비스를 정의하는 계층입니다.

**Transport(전송) 계층**

프로토콜: TCP, UDP

송수신측 사이 데이터 전송의 신뢰성을 보장하기 위한 계층입니다.

* 수행하는 일들  
  1. 송신측에서 수신 측으로 패킷이 정상적으로 전달되었는지 확인합니다.  
  2. 패킷이 정상적으로 전달되지 못했다면, 재전송합니다.  
  3. 망에 과부하가 걸리면, 전송 속도를 조절합니다.

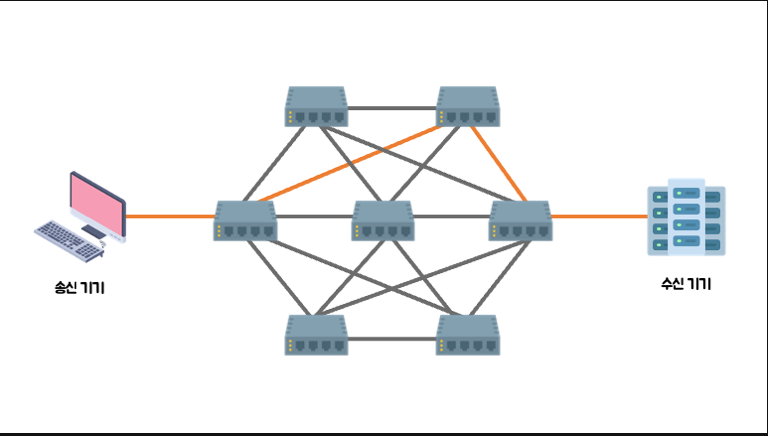
**Network(네트워크) 계층**

프로토콜: IP

송수신 기기 사이에는 라우터(router)라는 네트워크 장비들이 있으며, 패킷들은 이들을 거쳐 수신측으로 전달됩니다.

네트워크 계층은 패킷들이 어떤 라우터들을 거쳐 전달되어야 하는지를 결정합니다.

아래 그림은 송신 및 수신 기기 사이에 전달 경로가 결정된 것을 보여 줍니다. 주황색 선이 경로입니다.



**Link(링크) 계층**

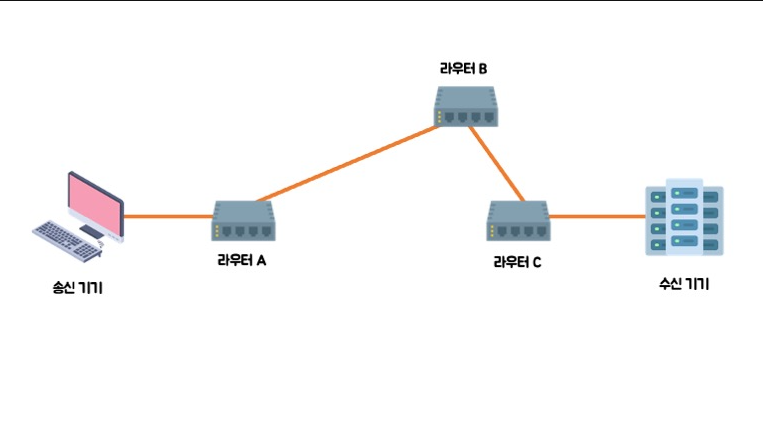
프로토콜: Ethernet, 토큰 링, FDDI, PPP

송수신측 사이에 경로가 결정되었으면, 링크 계층은 한 노드에서 인접한 노드로 패킷을 보내기 위한 역할을 합니다.

**노드**는 송신 기기, 수신 기기, 그 사이에 있는 네트워크 장비 등을 모두 일컫는 용어입니다.

이 계층이 만드는 패킷을 **프레임**(frame)이라고 하며, 이것이 실제로 망에 전송할 패킷입니다.

아래 그림과 같이 경로가 있다고 가정해 봅시다.  
송신 기기의 링크 계층은 송신 기기에서 라우터 A로 프레임을 전송하기 위한 작업을 수행합니다.  
마찬가지로 라우터 A는 라우터 B로 프레임을 보내기 위해 링크 계층을 실행합니다.



**Physical(물리적) 계층**

프로토콜 : IEEE 802.3u

물리적 계층은 한 노드의 링크 계층에서 생성한 프레임을 인접한 노드로 보내는 역할을 수행합니다.

이 때 각 비트는 전기적인 신호로 변조(modulate)되고, 이것이 유선이나 무선을 통해 다른 노드로 전달됩니다.

수신 노드는 이를 원래의 데이터로 복원을 합니다. 이것을 복조(demodulate)라고 합니다.

**계층들 사이에 주고 받는 데이터**

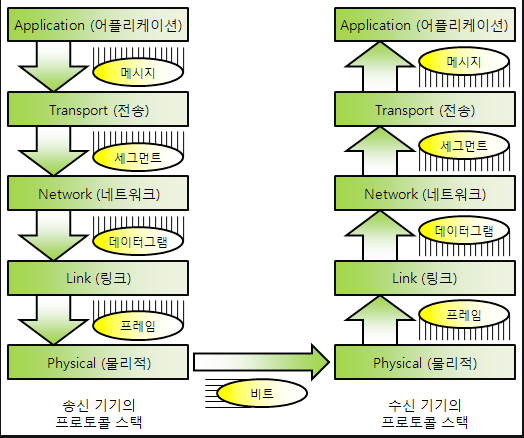
송신 기기가 통신을 하기 위해 수신 기기로 **패킷**을 보낸다고 가정해 봅시다.

이 때 **송신측 프로토콜 스택**에서는 다음과 같은 작업들이 실행됩니다.

1. 어플리케이션 계층은 트랜스포트 계층으로 메시지를 보냅니다.
2. 트랜스포트 계층은 네트워크 계층으로 세그먼트를 보냅니다.
3. 네트워크 계층은 링크 계층으로 데이터그램을 보냅니다.
4. 링크 계층은 물리적 계층으로 프레임을 보냅니다.
5. 물리적 계층은 수신측 물리적 계층으로 프레임의 비트들을 보냅니다.

**수신측 프로토콜 스택**에서는 다음과 같은 작업들이 실행됩니다.

1. 물리적 계층은 송신측 물리적 계층으로부터 프레임의 비트들을 받습니다.
2. 링크 계층은 물리적 계층으로부터 프레임들을 받습니다.
3. 네트워크 계층은 링크 계층으로부터 데이터그램을 받습니다.
4. 트랜스포트 계층은 네트워크 계층으로부터 세그먼트를 받습니다.
5. 어플리케이션 계층은 트랜스포트 계층으로부터 메시지를 받습니다.



**송수신 기기들 사이 동일 계층들이 주고 받는 데이터**

위 그림에서 프로토콜 스택들의 계층들 사이에 전달되는 데이터의 흐름을 보여드렸습니다.

이를 달리 해석하면, 동일한 계층 간의 통신이 이루어진다고도 볼 수 있습니다.

예를 들어, 어플리케이션 계층들과 통신을 할 수 있으며, 송신측 어플리케이션 계층이 수신측 어플리케이션 계층에 메시지를 보냅니다.

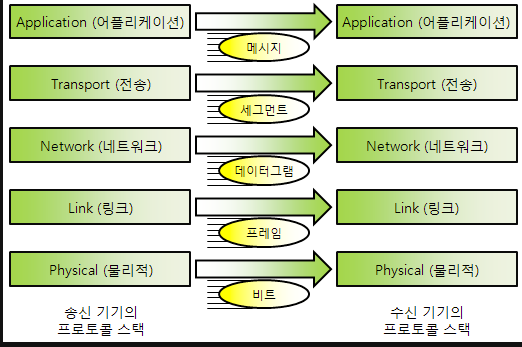
이는 송수신측의 하위 계층들이 잘 동작함을 전제로 합니다.

마찬가지로 송신측 트랜스포트 계층은 수신측 트랜스포트 계층과 통신을 합니다.

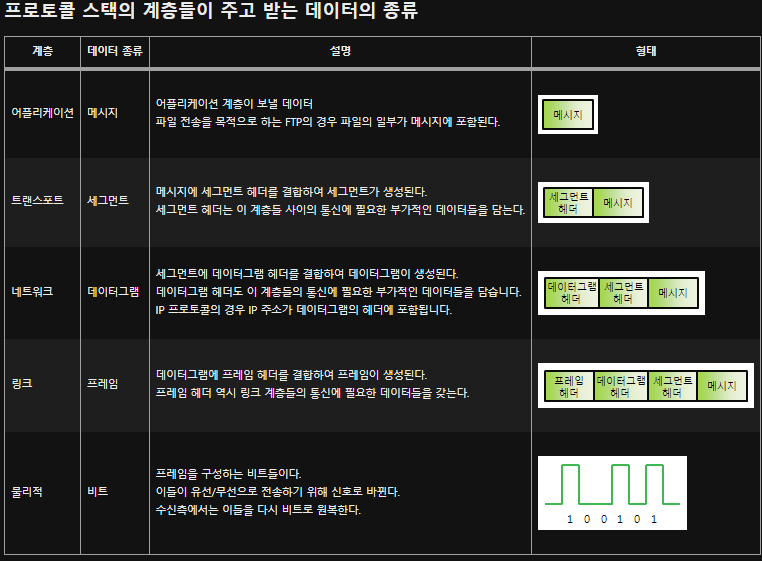
전자가 세그먼트를 보내면, 후자가 그것을 받습니다.

이하 계층들 역시 이와 같은 방식으로 작동합니다.

이를 그림으로 표현하면, 아래와 같습니다.



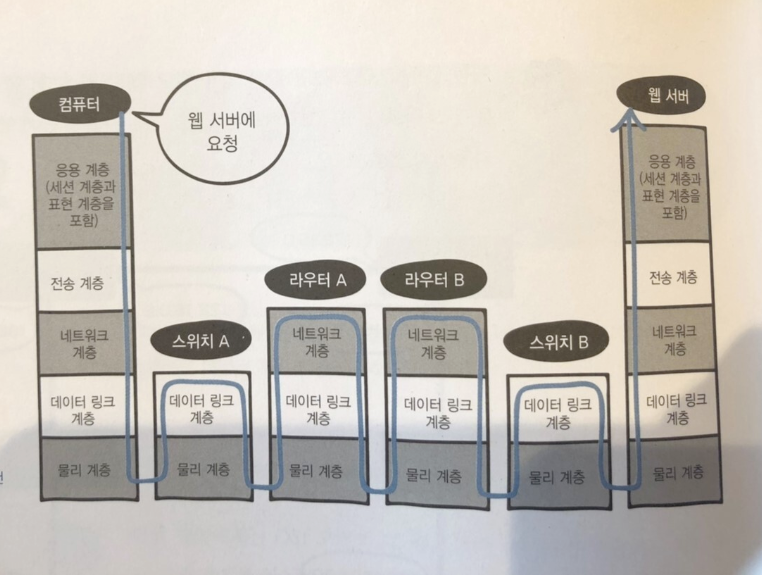
**프로토콜 스택의 계층들이 주고받는 데이터의 종류**



송신 기기가 패킷을 보낼 때 위 계층으로부터 아래 계층으로 내려가면서 헤더들이 차례로 추가

반대로 수신 기기가 패킷을 받을 때에는 아래에서 위 계층으로 올라가면서 헤더들이 반대 순서로 제거 그 결과, 송신 기기가 보냈던 메시지가 수신 기기에서 그대로 복원

**실제 데이터 전송 과정**



**네트워크 레이어의 장점**

OSI 또는 TCP/IP 모델과 같은 프로토콜 스택에서 네트워크 계층의 장점에는 모듈화, 추상화, 오버헤드 감소 및 복잡성 관리가 포함되며, 이는 보다 효율적이고 확장 가능하며 관리 가능한 네트워크 통신에 집합적으로 기여합니다.

**모듈화**

네트워크 계층은 네트워크 통신을 고유한 기능과 책임이 있는 별개의 계층으로 **분리**할 수 있도록 합니다.  
이 모듈화를 통해 다른 레이어에 영향을 주지 않고 한 레이어를 변경하거나 업데이트할 수 있으므로 네트워킹 프로토콜의 개발, 유지 관리 및 문제 해결이 더 쉬워집니다.  
또한 동일한 네트워크 계층 프로토콜을 준수하는 한 서로 다른 네트워킹 기술 및 장치 간의 상호 운용성을 향상시킬 수 있습니다.

**추상화**

네트워크 계층은 상위 계층이 상호 작용할 수 있도록 표준화된 인터페이스와 프로토콜 집합을 제공하여 네트워크 인프라의 근본적인 복잡성을 추상화합니다.  
이 추상화를 통해 전송 및 애플리케이션 계층과 같은 상위 계층이 기본 네트워크 기술 및 토폴로지와 독립적일 수 있습니다.  
이러한 관심사의 분리는 네트워크 인프라의 복잡성에 대해 걱정할 필요가 없기 때문에 응용 프로그램의 개발 및 관리를 단순화합니다.

**오버헤드 감소**

네트워크 계층은 네트워크 통신의 오버헤드를 최소화하는 데 도움이 됩니다.  
주소 지정, 라우팅 및 혼잡 제어와 같은 기능을 제공함으로써 네트워크 계층은 네트워크에서 데이터 패킷의 라우팅을 최적화하여 데이터 전송과 관련된 오버헤드를 줄일 수 있습니다.  
이를 통해 네트워크 리소스를 보다 효율적으로 사용하고 성능을 향상시킬 수 있습니다.

**복잡성 관리**

네트워크 계층은 네트워크 통신을 더 작고 관리 가능한 구성 요소로 나누어 네트워크 통신의 복잡성을 관리하는 데 도움이 됩니다.  
프로토콜 스택의 각 계층에는 표준화된 프로토콜에 의해 정의된 고유한 기능 및 책임 집합이 있습니다.  
이러한 작업 분할을 통해 전체 시스템에 영향을 주지 않고 적절한 계층에서 변경 또는 업데이트를 수행할 수 있으므로 네트워크 통신의 문제 해결, 유지 관리 및 확장성이 더 쉬워집니다.