1.5 프로토콜 계층과 서비스 모델

지금까지의 논의를 통해 인터넷이 매우 복잡한 시스템이라는 것이 분명해졌습니다. 인터넷에는 수많은 애플리케이션과 프로토콜, 다양한 종류의 단말 시스템, 패킷 스위치, 그리고 다양한 링크 레벨 미디어가 포함됩니다. 이러한 엄청난 복잡성 속에서 네트워크 아키텍처를 체계적으로 정리하거나, 적어도 네트워크 아키텍처에 대한 논의를 조직화할 수 있는 희망이 있을까요? 다행히도 두 질문에 대한 답은 "예"입니다.

1.5.1 계층화된 아키텍처

인터넷 아키텍처에 대한 생각을 정리하기 전에, 인간의 일상에서 사용하는 비유를 통해 살펴보겠습니다. 우리는 일상생활에서 복잡한 시스템을 자주 접합니다. 예를 들어, 누군가가 당신에게 항공 시스템을 설명해 달라고 요청했다고 상상해 보세요. 발권 직원, 수하물 검사원, 탑승구 직원, 조종사, 비행기, 항공 교통 관제, 그리고 전 세계적으로 비행기를 운항하는 시스템 등 이 복잡한 시스템을 어떻게 구조적으로 설명할 수 있을까요?

한 가지 방법은 당신이 비행기를 탈 때 취하는 일련의 행동(또는 다른 사람들이 당신을 위해 수행하는 행동)을 설명하는 것입니다. 당신은 항공권을 구매하고, 수하물을 위탁하며, 탑승구로 이동한 뒤 결국 비행기에 탑승합니다. 비행기는 이륙하여 목적지로 운항되고, 착륙 후 당신은 비행기에서 내려 수하물을 찾습니다. 만약 여행이 좋지 않았다면, 당신은 발권 직원에게 항공편에 대해 불평할 수 있습니다(아마 아무 보상도 받지 못할 가능성이 크지만). 이 시나리오는 그림 1.21에 나타나 있습니다.

이미 여기서 컴퓨터 네트워킹과의 유사성을 발견할 수 있습니다. 당신은 항공사에 의해 출발지에서 목적지로 "운송"됩니다. 마찬가지로, 인터넷에서는 패킷이 출발지 호스트에서 목적지 호스트로 "운송"됩니다. 하지만 우리가 찾고자 하는 비유는 정확히 이것이 아닙니다. 우리는 그림 1.21에서 어떤 구조를 찾고자 합니다.

그림 1.21을 살펴보면, 양쪽 끝에 발권 기능이 있고, 이미 발권된 승객을 위한 수하물 기능, 그리고 발권과 수하물 검사를 마친 승객을 위한 탑승구 기능이 있다는 것을 알 수 있습니다. 탑승구를 통과한 승객(즉, 발권과 수하물 검사를 완료한 승객)에 대해서는 이착륙 기능이 있으며, 비행 중에는 비행기 경로 설정 기능이 있습니다. 이는 우리가 그림 1.21의 기능을 수평적으로 살펴볼 수 있음을 시사하며, 이는 그림 1.22에 나타나 있습니다.

그림 1.22는 항공 기능을 계층으로 나누어 항공 여행을 논의할 수 있는 프레임워크를 제공합니다. 각 계층은 아래 계층과 함께 특정 기능, 즉 서비스를 구현합니다. 발권 계층 및 그 아래에서는 사람의 공항 간 이동 서비스가 이루어집니다. 수하물 계층 및 그 아래에서는 사람과 수하물의 위탁-수령 서비스가 이루어집니다. 수하물 계층은 이미 발권된 사람에게만 이 서비스를 제공합니다. 탑승구 계층에서는 출발 탑승구에서 도착 탑승구로의 사람과 수하물 이동 서비스가 이루어집니다. 이착륙 계층에서는 활주로 간 사람과 수하물의 이동 서비스가 이루어집니다.

각 계층은 (1) 해당 계층 내에서 특정 작업을 수행하고 (예: 탑승구 계층에서 비행기에 사람을 태우고 내리는 작업), (2) 바로 아래 계층의 서비스를 활용하여 서비스를 제공합니다 (예: 탑승구 계층에서 이착륙 계층의 활주로 간 승객 이동 서비스를 사용).

계층화된 아키텍처는 크고 복잡한 시스템의 특정 부분을 명확히 정의하여 논의할 수 있게 해줍니다. 이러한 단순화는 모듈성을 제공하여 계층이 제공하는 서비스의 구현을 훨씬 쉽게

변경할 수 있도록 합니다. 계층이 상위 계층에 동일한 서비스를 제공하고 하위 계층의 동일한 서비스를 사용하는 한, 계층의 구현이 변경되더라도 시스템의 나머지 부분은 영향을 받지 않습니다. (서비스의 구현을 변경하는 것은 서비스 자체를 변경하는 것과 매우 다릅니다!) 예를 들어, 탑승구 기능이 변경된다면(예: 사람들을 키 순서대로 비행기에 태우고 내리도록 변경), 탑승구 계층이 여전히 동일한 기능(사람을 태우고 내리는 것)을 제공하므로 항공 시스템의 나머지 부분은 영향을 받지 않습니다. 크고 복잡하며 지속적으로 업데이트되는 시스템에서, 시스템의 다른 구성 요소에 영향을 주지 않고 서비스 구현을 변경할 수 있는 능력은 계층화의 또 다른 중요한 장점입니다.

프로토콜 계층화

이제 항공 이야기는 충분히 다루었으니, 네트워크 프로토콜로 관심을 돌려보겠습니다. 네트워크 프로토콜의 설계를 구조화하기 위해 네트워크 설계자들은 프로토콜과 이를 구현하는 네트워크 하드웨어 및 소프트웨어를 계층으로 조직화합니다. 각 프로토콜은 하나의 계층에 속하며, 이는 그림 1.22의 항공 아키텍처에서 각 기능이 하나의 계층에 속했던 것과 유사합니다. 우리는 계층이 상위 계층에 제공하는 서비스, 즉 계층의 **서비스 모델**에 관심이 있습니다. 항공 예시와 마찬가지로, 각 계층은 (1) 해당 계층 내에서 특정 작업을 수행하고, (2) 바로 아래 계층의 서비스를 활용하여 서비스를 제공합니다. 예를 들어, 계층 n이 제공하는 서비스는 네트워크의 한쪽 끝에서 다른 쪽 끝으로 메시지를 안정적으로 전달하는 것일 수 있습니다. 이는 계층 n-1의 비신뢰적인 메시지 전달 서비스를 사용하고, 계층 n에서 손실된 메시지를 감지하고 재전송하는 기능을 추가하여 구현될 수 있습니다.

프로토콜 계층은 소프트웨어, 하드웨어, 또는 둘의 조합으로 구현될 수 있습니다. 애플리케이션 계층 프로토콜(예: HTTP, SMTP)은 거의 항상 단말 시스템의 소프트웨어로 구현됩니다. 전송 계층 프로토콜도 마찬가지입니다. 물리 계층과 데이터 링크 계층은 특정 링크를 통해 통신을 처리하므로, 일반적으로 해당 링크와 관련된 네트워크 인터페이스 카드(예: 이더넷 또는 WiFi 인터페이스 카드)에 구현됩니다. 네트워크 계층은 종종 하드웨어와 소프트웨어의 혼합 구현입니다. 또한, 항공 아키텍처에서 기능이 시스템을 구성하는 다양한 공항과 비행 관제 센터에 분산되어 있었던 것처럼, 계층 n 프로토콜도 네트워크를 구성하는 단말 시스템, 패킷 스위치, 기타 구성 요소에 분산되어 있습니다. 즉, 네트워크의 각 구성 요소에는 계층 n 프로토콜의 일부가 포함됩니다.

프로토콜 계층화는 개념적, 구조적 이점을 제공합니다 [RFC 3439]. 계층화는 시스템 구성 요소를 논의하는 구조화된 방법을 제공하며, 모듈성은 시스템 구성 요소의 업데이트를 더 쉽게 만듭니다. 그러나 일부 연구자와 네트워크 엔지니어들은 계층화에 강하게 반대합니다 [Wakeman 1992]. 계층화의 잠재적 단점 중 하나는 한 계층이 하위 계층의 기능을 중복할 수 있다는 점입니다. 예를 들어, 많은 프로토콜 스택은 링크별로 그리고 단대단(end-to-end)으로 오류 복구를 제공합니다. 또 다른 잠재적 단점은 한 계층의 기능이 다른 계층에만 존재하는 정보(예: 타임스탬프 값)를 필요로 할 수 있다는 점입니다. 이는 계층 분리의 목표를 위반합니다.

다양한 계층의 프로토콜을 모두 합친 것을 **프로토콜 스택**이라고 합니다. 인터넷 프로토콜 스택은 그림 1.23에 나타난 바와 같이 물리, 링크, 네트워크, 전송, 애플리케이션의 다섯 계층으로 구성됩니다. 이 책의 목차를 살펴보면, 우리는 인터넷 프로토콜 스택의 계층을 따라 대략적으로 구성되어 있음을 알 수 있습니다. 우리는 위에서 아래로, 즉 애플리케이션 계층부터 시작하여 아래로 진행합니다.

애플리케이션 계층

애플리케이션 계층은 네트워크 애플리케이션과 그 애플리케이션 계층 프로토콜이 존재하는 곳입니다. 인터넷의 애플리케이션 계층에는 HTTP(웹 문서 요청 및 전송 제공), SMTP(이메일 메시지 전송 제공), FTP(두 단말 시스템 간 파일 전송 제공) 등 많은 프로토콜이 포함됩니다. 또한, www.ietf.org와 같은 인터넷 단말 시스템의 사용자 친화적 이름을 32비트 네트워크 주소로 변환하는 작업도 특정 애플리케이션 계층 프로토콜, 즉 도메인 네임 시스템(DNS)을 통해 이루어집니다. 2장에서 우리는 새로운 애플리케이션 계층 프로토콜을 만들고 배포하는 것이 매우 쉽다는 것을 볼 것입니다.

애플리케이션 계층 프로토콜은 여러 단말 시스템에 분산되어 있으며, 한 단말 시스템의 애플리케이션이 프로토콜을 사용하여 다른 단말 시스템의 애플리케이션과 정보 패킷을 교환합니다. 우리는 애플리케이션 계층에서의 이 정보 패킷을 **메시지**라고 부를 것입니다.

전송 계층

인터넷의 전송 계층은 애플리케이션 엔드포인트 간에 애플리케이션 계층 메시지를 전달합니다. 인터넷에는 두 가지 전송 프로토콜, 즉 TCP와 UDP가 있으며, 둘 다 애플리케이션 계층 메시지를 전달할 수 있습니다. TCP는 애플리케이션에 **연결 지향 서비스**를 제공합니다. 이 서비스는 애플리케이션 계층 메시지의 목적지로의 보장된 전달과 흐름 제어(즉, 송신자와 수신자의 속도 매칭)를 포함합니다. TCP는 긴 메시지를 더 짧은 세그먼트로 나누고, 네트워크가 혼잡할 때 소스가 전송 속도를 조절하는 혼잡 제어 메커니즘도 제공합니다. 반면, UDP는 애플리케이션에 **비연결 서비스**를 제공합니다. 이 서비스는 신뢰성, 흐름 제어, 혼잡 제어를 제공하지 않는 간단한 서비스입니다. 이 책에서는 전송 계층 패킷을 **세그먼트**라고 부릅니다.

네트워크 계층

인터넷의 네트워크 계층은 **데이터그램**으로 알려진 네트워크 계층 패킷을 한 호스트에서 다른 호스트로 이동시키는 역할을 합니다. 출발지 호스트의 전송 계층 프로토콜(TCP 또는 UDP)은 전송 계층 세그먼트와 목적지 주소를 네트워크 계층에 전달합니다. 이는 마치 당신이 우편 서비스에 편지와 목적지 주소를 주는 것과 같습니다. 네트워크 계층은 세그먼트를 목적지 호스트의 전송 계층으로 전달하는 서비스를 제공합니다.

인터넷의 네트워크 계층에는 데이터그램의 필드와 단말 시스템 및 라우터가 이 필드를 어떻게 처리하는지를 정의하는 유명한 **IP 프로토콜**이 포함됩니다. IP 프로토콜은 하나뿐이며, 네트워크 계층을 가진 모든 인터넷 구성 요소는 IP 프로토콜을 실행해야 합니다. 네트워크 계층에는 또한 데이터그램이 출발지에서 목적지로 가는 경로를 결정하는 **라우팅 프로토콜**도 포함됩니다. 인터넷에는 많은 라우팅 프로토콜이 있습니다. 1.3절에서 보았듯이, 인터넷은 네트워크의 네트워크이며, 하나의 네트워크 내에서 네트워크 관리자는 원하는 어떤 라우팅 프로토콜을 실행할 수 있습니다. 네트워크 계층에는 IP 프로토콜과 수많은 라우팅 프로토콜이 포함되어 있지만, IP가 인터넷을 하나로 묶는 접착제 역할을 하므로 종종 단순히 **IP 계층**이라고 불립니다.

링크 계층

인터넷의 네트워크 계층은 데이터그램을 출발지에서 목적지로 가는 일련의 라우터를 통해 라우팅합니다. 패킷을 한 노드(호스트 또는 라우터)에서 경로상의 다음 노드로 이동시키기

위해 네트워크 계층은 링크 계층의 서비스에 의존합니다. 특히, 각 노드에서 네트워크 계층은 데이터그램을 링크 계층으로 전달하고, 링크 계층은 데이터그램을 경로상의 다음 노드로 전달합니다. 다음 노드에서는 링크 계층이 데이터그램을 네트워크 계층으로 전달합니다.

링크 계층이 제공하는 서비스는 링크에서 사용되는 특정 링크 계층 프로토콜에 따라 다릅니다. 예를 들어, 일부 링크 계층 프로토콜은 송신 노드에서 수신 노드로 한 링크를 통해 신뢰성 있는 전달을 제공합니다. 이 신뢰성 있는 전달 서비스는 TCP의 신뢰성 있는 전달 서비스(한 단말 시스템에서 다른 단말 시스템으로의 신뢰성 있는 전달)와 다릅니다. 링크 계층 프로토콜의 예로는 이더넷, WiFi, 케이블 액세스 네트워크의 DOCSIS 프로토콜 등이 있습니다. 데이터그램은 출발지에서 목적지로 이동하기 위해 여러 링크를 거쳐야 하므로, 경로상의 각기 다른 링크에서 서로 다른 링크 계층 프로토콜에 의해 처리될 수 있습니다. 예를 들어, 데이터그램은 한 링크에서는 이더넷에 의해, 다음 링크에서는 PPP에 의해 처리될 수 있습니다. 네트워크 계층은 각기 다른 링크 계층 프로토콜로부터 서로 다른 서비스를 받게 됩니다. 이 책에서는 링크 계층 패킷을 **프레임**이라고 부릅니다.

물리 계층

링크 계층의 역할이 네트워크 요소 간에 전체 프레임을 이동시키는 것이라면, 물리 계층의 역할은 프레임 내의 개별 비트를 한 노드에서 다음 노드로 이동시키는 것입니다. 이 계층의 프로토콜은 다시 링크에 따라 달라지며, 링크의 실제 전송 매체(예: 꼬임쌍선, 단일 모드 광섬유)에 따라 달라집니다. 예를 들어, 이더넷은 꼬임쌍선, 동축 케이블, 광섬유 등에 대해 각각 다른 물리 계층 프로토콜을 가지고 있습니다. 각 경우에 비트는 링크를 통해 서로 다른 방식으로 이동됩니다.

1.5.2 캡슐화

그림 1.24는 데이터가 송신 단말 시스템의 프로토콜 스택을 내려가고, 중간의 링크 계층 스위치와 라우터의 프로토콜 스택을 오르내리며, 수신 단말 시스템의 프로토콜 스택을 올라가는 물리적 경로를 보여줍니다. 이 책에서 나중에 논의하겠지만, 라우터와 링크 계층 스위치는 모두 패킷 스위치입니다. 단말 시스템과 마찬가지로 라우터와 링크 계층 스위치는 네트워킹 하드웨어와 소프트웨어를 계층으로 조직화합니다. 하지만 라우터와 링크 계층 스위치는 프로토콜 스택의 모든 계층을 구현하지 않습니다. 일반적으로 하위 계층만 구현합니다. 그림 1.24에 나타난 바와 같이, 링크 계층 스위치는 1~2계층을 구현하고, 라우터는 1~3계층을 구현합니다. 이는 예를 들어, 인터넷 라우터는 IP 프로토콜(3계층 프로토콜)을 구현할 수 있지만, 링크 계층 스위치는 그렇지 않다는 것을 의미합니다. 나중에 보겠지만, 링크 계층 스위치는 IP 주소를 인식하지 못하지만, 이더넷 주소와 같은 2계층 주소를 인식할 수 있습니다. 호스트는 다섯 계층 모두를 구현하며, 이는 인터넷 아키텍처가 네트워크의 가장자리에 많은 복잡성을 두고 있다는 관점과 일치합니다.

그림 1.24는 또한 **캡슐화**라는 중요한 개념을 보여줍니다. 송신 호스트에서 애플리케이션 계층 메시지(M, 그림 1.24)는 전송 계층으로 전달됩니다. 가장 간단한 경우, 전송 계층은 메시지를 받아 수신측 전송 계층에서 사용할 추가 정보(소위 전송 계층 헤더 정보, Ht, 그림 1.24)를 추가합니다. 애플리케이션 계층 메시지와 전송 계층 헤더 정보는 함께 **전송 계층 세그먼트**를 구성합니다. 따라서 전송 계층 세그먼트는 애플리케이션 계층 메시지를 캡슐화합니다. 추가된 정보에는 수신측 전송 계층이 메시지를 적절한 애플리케이션으로 전달할 수 있게 하는 정보와, 경로에서 비트가 변경되었는지 확인할 수 있는 오류 감지 비트가 포함될 수 있습니다. 전송 계층은 세그먼트를 네트워크 계층으로

전달하고, 네트워크 계층은 출발지 및 목적지 단말 시스템 주소와 같은 네트워크 계층 헤더 정보(Hn, 그림 1.24)를 추가하여 **네트워크 계층 데이터그램**을 생성합니다. 데이터그램은 링크 계층으로 전달되고, 링크 계층은 자체 링크 계층 헤더 정보를 추가하여 **링크 계층 프레임**을 생성합니다. 따라서 각 계층에서 패킷은 헤더 필드와 페이로드 필드라는 두 가지 유형의 필드를 가집니다. 페이로드는 일반적으로 상위 계층의 패킷입니다.

여기서 유용한 비유는 한 기업 지사에서 다른 지사로 공공 우편 서비스를 통해 사내 메모를 보내는 상황입니다. 한 지사에 있는 앨리스가 다른 지사에 있는 밥에게 메모를 보내고 싶다고 가정해 봅시다. 메모는 애플리케이션 계층 메시지에 해당합니다. 앨리스는 메모를 사내 봉투에 넣고, 봉투 앞면에 밥의 이름과 부서명을 적습니다. 사내 봉투는 전송 계층 세그먼트에 해당하며, 헤더 정보(밥의 이름과 부서 번호)를 포함하고 애플리케이션 계층 메시지(메모)를 캡슐화합니다. 송신 지사의 우편실은 사내 봉투를 받아 공공 우편 서비스를 통해 보낼 수 있는 또 다른 봉투에 넣습니다. 우편실은 또한 우편 봉투에 송신 및 수신 지사의 우편 주소를 적습니다. 여기서 우편 봉투는 데이터그램에 해당하며, 전송 계층 세그먼트(사내 봉투)를 캡슐화하고, 이는 원래 메시지(메모)를 캡슐화합니다. 우편 서비스는 우편 봉투를 수신 지사의 우편실로 배달합니다. 그곳에서 캡슐화 해제 과정이 시작됩니다. 우편실은 사내 메모를 추출하여 밥에게 전달합니다. 마지막으로, 밥은 봉투를 열어 메모를 꺼냅니다.

캡슐화 과정은 위에서 설명한 것보다 더 복잡할 수 있습니다. 예를 들어, 큰 메시지는 여러 전송 계층 세그먼트로 나뉠 수 있으며, 각 세그먼트는 다시 여러 네트워크 계층 데이터그램으로 나뉠 수 있습니다. 수신측에서는 이러한 세그먼트를 구성 데이터그램들로부터 재구성해야 합니다.