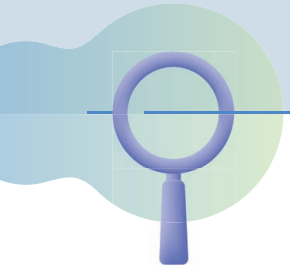


인터넷에서 사용되는 여러 기술

DHCP 이야기 1

Dynamic
Host
Configuration
Protocol

글
조인준
KBS 미디어기술연구부 수석연구원

오늘날 네트워크 환경에서 DHCP(Dynamic Host Configuration Protocol)는 필수적인 프로토콜 중 하나입니다. DHCP는 네트워크에 연결된 장치가 자동으로 IP 주소, 서브넷 마스크, 게이트웨이, DNS 서버 등의 설정을 받을 수 있도록 해주며, 이를 통해 효율적으로 네트워크를 운영할 수 있습니다. 오늘날 집이나 회사에서 DHCP가 없는 환경을 상상하기 어려울 정도로 IP 네트워크의 핵심 역할을 담당하고 있습니다.

그러나 DHCP가 등장하기 이전에는 네트워크의 호스트 장치에 IP 주소를 할당하는 것이 간단하지는 않았습니다. 인터넷이 태동할 당시에는 인터넷에 연결되는 호스트 장치 수가 매우 적었고, 이런 장치들 대부분이 전문가들 사이에서만 쓰였기 때문에 관리자가 수동으로 각 호스트 장치의 IP 주소를 설정하는 것이 문제가 되지 않았습니다. 하지만 인터넷에 연결되는 호스트 장치 수가 많아지고 네트워크가 커지면서 수동으로 IP 주소를 설정하는 방식으로는 변화하는 환경에 대응하기 어려워졌고 자동으로 IP 주소를 설정하는 방식을 개발하는 것이 필요했습니다.

이러한 변화에 대응하여 네트워크에 연결된 호스트 장치가 부팅 시 자동으로 IP 주소를 할당받을 수 있도록 하는 프로토콜로 BOOTP(Bootstrap Protocol)가 개발되었고, 이는 DHCP로 이어지게 됩니다. BOOTP는 IP 주소의 설정 방식과 기능의 한계로 인해 확장성이 부족했으며, 새롭게 요구되는 동적 IP 주소 할당이 어려웠습니다. 이에 따라 더욱 유연하고 자동화된 IP 주소 할당이 가능한 DHCP가 개발되었으며, 오늘날 널리 사용되는 표준이 되었습니다.
DHCP에 대해 본격적으로 다루기 전에 이번 편에서는 BOOTP의 작동 원리를 살펴보고, DHCP가 왜 필요하게 되었는지를 알아봄으로써 네트워크 IP 주소 자동 할당 방식의 변화 과정에 대한 이해를 해보겠습니다.

TCP/IP 네트워크에서 정상적 통신이 이루어지기 위해서는 각 호스트 장치들이 자신의 IP 주소를 알아야 합니다. 오늘날 대부분의 네트워크 호스트 장치는 디스크 등의 내부 저장 장치에서 이 정보를 읽어올 수 있지만, 80년대에는 대부분의 호스트 장치에 저장 장치가 없었기 때문에 네트워크상의 다른 장치의 도움을 받아 IP 주소 및 기타 필요한 정보나 소프트웨어를 제공받아야만 정상적인 호스트 장치로 동작할 수 있었습니다. 내부 저장 장치가 없는 네트워크의 호스트 장치를 정상적으로 동작하도록 설정하는 이 과정을 부트스트래핑(Bootstrapping)이라고 부르며, 이러한 기능을 호스트 장치에 제공하기 위해 BOOTP(Bootstrap Protocol)가 개발되었습니다.

BOOTP(Bootstrap Protocol)

IP 네트워크 초기 호스트 장치가 정상적으로 작동하기 위해 필요한 여러 파라미터를 자동으로 설정하는 문제는 꼭 해결해야 하는 과제였습니다. 1980년대 초반, IP 네트워크는 비교적 규모가 작고 단순했습니다. 따라서 수동으로 네트워크를 설정하는 것이 어렵긴 했지만, 자동화된 호스트 장치 구성 기능이 꼭 필요한 것으로 여겨지지는 않았습니다. 그러나 Diskless Workstation과 같이 당시에는 고가였던 내부 저장 장치가 없어 상대적으로 저렴하고 대량 보급이 가능했던 장치들이 늘어났고, 이를 효율적으로 운영하기 위해서는 자동화된 네트워크 구성 방식이 필요했습니다.

내부 저장 장치가 없는 호스트 장치는 전원이 켜질 때마다 자신의 IP 주소를 알아내고 이를 이용해서 OS 등을 다운로드 받아서 자신에게 로딩해야만 동작할 수 있었습니다. 하지만 이러한 장치는 IP 주소가 없으면 통신이 불가능한 상황에서 IP 네트워크를 통해 IP 주소를 알아내야 하는 모순적 상황에 처하게 됩니다. 이를 해결하는 과정이 부트스트래핑(Bootstrapping) 또는 부팅(Booting)이며, 이는 사람이 자기 신발 뒤꿈치의 손잡이 끈(Bootstrap)을 잡아당겨 스스로를 일으키는 모습에서 유래한 개념입니다. 부트스트래핑은 초기에는 '자기 신발의 손잡이 끈(Bootstrap)을 잡아당겨 자신을 울타리 너머로 들어 올리는 행동'을 의미하여 불가능하거나 터무니없는 일을 뜻하는 표현으로 쓰였습니다만, 이후 '자력으로 어떤 작업을 점진적으로 수행하는 나가는 것'을 의미하는 방향으로 변화했고, 이 의미가 현재 우리가 아는 부팅의 의미입니다.

BOOTP 이전에도 RARP(Reverse Address Resolution Protocol)라는 것이 있었습니다. 1984년에 개발된 RARP는 IP 주소와 링크 계층 하드웨어 주소를 매핑하는 ARP(Address Resolution Protocol)를 기반으로 한 프로토콜이었으며 클라이언트와 서버 간의 간단한 요청/응답 방식으로 동작하여 디스크가 없는 호스트 장치에 IP 주소를 제공할 수 있었습니다. 그러나 RARP에는 여러 가지 심각한 제약이 있어서 TCP/IP 환경에서의 호스트 장치 구성 요구 사항을 충족하지 못했습니다. 따라서 디스크 없는 호스트 장치뿐만 아니라 자동화된 호스트 장치 구성 기능이 필요한 다른 상황에서도 활용할 수 있도록 1985년 BOOTP(Bootstrap Protocol)가 개발되었습니다. BOOTP는 RARP의 부족한 점을 보완하기 위해 다음과 같은 기능을 갖추고 있습니다.

하드웨어 독립적 설계
클라이언트/서버 모델을 사용하지만 RARP와 달리 BOOTP는 UDP 기반의 소프트웨어 프로토콜로 구현. 따라서 특정 네트워크 하드웨어에 종속되지 않음

추가적인 구성 정보 제공
BOOTP는 클라이언트에 IP 주소뿐만 아니라 서브넷 마스크, 게이트웨이, DNS 서버 등의 추가 정보를 보낼 수 있음. 대부분의 경우 한 번의 메시지 전송만으로 필요한 모든 정보를 제공할 수 있어 효율적임

다른 네트워크에서도 동작 가능
BOOTP는 클라이언트와 서버가 서로 다른 네트워크에 존재하더라도 사용할 수 있음. 이를 통해 IP 주소를 할당하는 서버를 중앙에서 관리할 수 있으며, 이는 비용 절감과 관리의 편의성을 높이는 장점이 됨

BOOTP라는 이름만 보면, BOOTP가 저장 장치가 없는 네트워크상의 호스트 장치가 완전히 부팅되는 과정 전체를 담당하는 것처럼 보일 수 있습니다. 그러나 실제로 BOOTP는 부트스트래핑 과정의 첫 번째 단계만 처리합니다.

1단계

클라이언트(호스트 장치)가 IP 주소 및 네트워크 구성 정보를 할당받는 단계이며, 여기까지가 BOOTP의 역할

2단계

클라이언트가 운영 체제 및 드라이버 같은 필수 소프트웨어를 다운로드하여 네트워크에서 정상적으로 작동할 수 있도록 설정됨. 이 단계에서는 보통 TFTP(Trivial File Transfer Protocol) 같은 단순 파일 전송 프로토콜이 사용됨

BOOTP는 다양한 호스트 장치에서 활용될 수 있지만, 지금의 시각으로 보면 저장 장치가 없는 매우 단순한 호스트 장치를 자동으로 구성하는 것이 주된 목표였습니다. 이러한 호스트 장치는 기능이 제한적이기 때문에 복잡한 부팅 프로토콜을 지원하기 어렵습니다. 따라서 BOOTP는 불필요한 개념을 배제한 단순한 프로토콜로 설계되어 쉽고 빠르게 호스트 장치 구성을 수행할 수 있도록 만들어졌습니다.

BOOTP는 TCP/IP의 여러 프로토콜과 마찬가지로 클라이언트/서버 구조를 따릅니다. 프로토콜의 동작 방식은 BOOTP 클라이언트와 서버 간의 메시지 교환으로 이루어집니다. BOOTP 클라이언트와 서버를 간단히 정의하면 다음과 같습니다.

BOOTP 클라이언트
IP 주소 및 네트워크 구성이 필요한 어떠한 유형의 호스트 장치도 BOOTP 클라이언트가 될 수 있음

BOOTP 서버
BOOTP 클라이언트의 요청에 응답하도록 특별히 설정된 네트워크 장치이며 클라이언트에게 제공할 수 있는 IP 주소 및 기타 네트워크 정보가 미리 프로그래밍 되어 있음

BOOTP 서버는 각 클라이언트에 대한 정보를 유지 관리합니다. 이 중 핵심 요소는 각 클라이언트의 하드웨어 주소(MAC 주소 같은)와 할당된 IP 주소 간의 매핑 테이블입니다. 클라이언트가 서버에 요청을 보낼 때 자신의 하드웨어 주소를 포함하여 보내면 BOOTP 서버는 해당 주소를 기반으로 사전에 지정된 IP 주소를 조회하고 이 IP 주소를 클라이언트에 응답합니다. 물론 다른 주소 할당 방식도 존재하지만, 예로 든 매핑 테이블 방식이 가장 일반적으로 사용된다고 합니다. BOOTP 메시지는 UDP(User Datagram Protocol) 프로토콜을 사용합니다.
이는 두 가지 주요 이유 때문입니다. 우선 첫째, UDP는 TCP보다 훨씬 간단한 구조를 가지고 있어 BOOTP와 같은 단순한 통신을 위한 프로토콜에 적합합니다. 그리고 둘째, BOOTP 클라이언트는 부팅 시점에 BOOTP 서버의 IP 주소를 모릅니다. 따라서 로컬 네트워크 전체에 브로드캐스트(Broadcast) 방식으로 IP 주소 할당 요청을 전송해야 하는데, UDP는 브로드캐스트를 지원하지만 TCP는 브로드캐스트를 지원하지 않습니다. UDP 메시지 전송에 BOOTP 서버는 67번 포트를 통해 클라이언트가 브로드캐스

트한 BOOTP 요청을 수신하여 처리한 뒤, 그 결과를 BOOTP 클라이언트의 68번 포트로 전송합니다.

BOOTP 서버가 클라이언트에게 응답하는 방식은 클라이언트가 자신의 IP 주소를 알고 있는지 여부에 따라 달라집니다.

BOOTP 클라이언트가 자신의 IP 주소를 알고 있는 경우
특수한 상황에서는 BOOTP 클라이언트가 이미 자신의 IP 주소를 알고 있는 경우가 있음.
이 경우, BOOTP 서버는 클라이언트의 IP 주소를 사용하여 직접 응답을 보낼 수 있음

BOOTP 클라이언트가 자신의 IP 주소를 모르는 경우
클라이언트의 하드웨어 주소를 이용하여 이 하드웨어 주소에 할당된 IP 주소를 찾은 후 유니캐스트(Unicast)로 응답을 전송하고 자신의 ARP 테이블을 업데이트하여 이후에 계속 이 IP 주소를 이용.
만약 운영 체제가 유니캐스트를 지원하지 않는 경우에는 로컬 네트워크 전체로 응답을 브로드캐스트

BOOTP 동작 방식

다음은 일반적인 BOOTP의 부트스트래핑 절차에서 클라이언트와 서버가 수행하는 기본 단계들입니다.

1 클라이언트가 요청을 생성

클라이언트는 BOOTP 요청 메시지를 생성하며, 메시지의 각 필드에 다음과 같은 정보를 설정합니다.

- **메시지 타입** : 값을 1로 설정하여 BOOTREQUEST 메시지임을 나타냄
- **CIAddr(Client IP Address)** : 클라이언트가 이미 자신의 IP 주소를 알고 있고, 계속 사용할 예정이라면 이를 CIAddr 필드에 지정. 만약 IP 주소를 모른다면 이 필드는 모두 0으로 채움
- **CHAddr(Client Hardware Address)** : 클라이언트의 하드웨어 주소(MAC 주소 같은)를 입력. 이 정보는 서버가 올바른 클라이언트를 식별하고, 적절한 설정을 제공하는 데 사용
- **XID(Transaction ID)** : 클라이언트는 랜덤한 트랜잭션 식별자(XID)를 생성하여 이 필드에 입력. 이를 통해 요청과 응답을 매칭할 수 있음
- **SName(Server Name)** : 특정한 BOOTP 서버로부터 응답을 받고 싶다면, 해당 서버의 이름을 지정할 수 있음
- **File(파일명)** : 클라이언트가 특정한 부트 파일(예: 운영 체제 이미지)을 요청할 경우, 해당 파일명을 입력할 수 있음

2 클라이언트가 요청을 전송

- 클라이언트가 BOOTP 요청(BOOTREQUEST) 메시지를 브로드캐스트 주소(255.255.255.255)로 전송하여 네트워크상의 모든 장치가 받을 수 있도록 함. 만약 클라이언트가 이미 BOOTP 서버의 IP 주소를 알고 있다면, 브로드캐스트 대신 유니캐스트로 요청을 보낼 수도 있음

3 서버의 요청 수신 및 처리

- BOOTP 서버는 UDP 포트 67에서 수신 대기
- 클라이언트가 브로드캐스트한 BOOTREQUEST 메시지를 포트 67로 수신하고 처리

- 클라이언트가 특정한 서버의 이름을 SName 필드에 지정하고 이 필드의 이름이 현재 서버의 이름과 다르다면 서버는 요청을 무시할 수도 있음
- 특정 서버가 지정되지 않았거나 클라이언트가 요청한 서버가 현재 서버와 일치한다면, 서버는 응답을 생성하여 클라이언트에게 회신

4 서버의 응답 생성

- BOOTP 서버는 클라이언트의 요청 메시지를 기반으로 BOOTREPLY 메시지를 생성
- **메시지 타입** : 값을 2로 설정하여 요청에 대한 응답인 BOOTREPLY 메시지임을 나타냄
- **클라이언트의 CHAddr(Client Hardware Address) 조회** : 클라이언트가 요청 메시지에서 제공한 CHAddr를 사용하여 서버의 IP 주소 할당 테이블에서 해당 클라이언트에 맞는 IP 주소를 조회
- **클라이언트의 YIAddr(Your IP Address) 설정** : 조회된 클라이언트의 IP 주소를 YIAddr 필드에 설정하여 응답 메시지에 보냄
- **File 필드 처리** : 클라이언트가 특정한 부트 파일을 요청했다면 서버는 해당 파일의 TFTP 다운로드 경로를 응답 메시지에 포함. 만약 File 필드가 비어 있다면, 서버는 기본 부트 파일의 TFTP 다운로드 경로를 제공
- **서버 정보 설정** : 서버는 자신의 IP 주소를 SIAddr 필드에 설정 및 자신의 이름을 SName 필드에 넣어 클라이언트에게 알림

5 서버의 응답 전송

- 서버는 생성한 BOOTREPLY 메시지를 클라이언트에게 전송
- 전송 방식은 클라이언트의 요청 메시지 내용에 따라 달라짐
- **브로드캐스트 전송** : 만약 BOOTREQUEST에 브로드캐스트 플래그가 설정되어 있으면, 클라이언트는 유니캐스트로 응답을 받을 수 없는 상태이고 서버는 브로드캐스트(255.255.255.255)로 응답을 보냄
- **유니캐스트 전송** : 만약 클라이언트가 자신의 IP 주소(CIAddr 필드)를 알고 있다면, 서버는 CIAddr에 명시된 IP 주소로 직접 유니캐스트 전송
- **ARP 또는 브로드캐스트 선택 전송** : 만약 브로드캐스트 플래그가 꺼져 있고, 클라이언트의 IP를 직접 알 수 없는 경우, 서버는 ARP 프로토콜로 IP를 알아낸 후 전송하거나 브로드캐스트를 사용할 수 있음

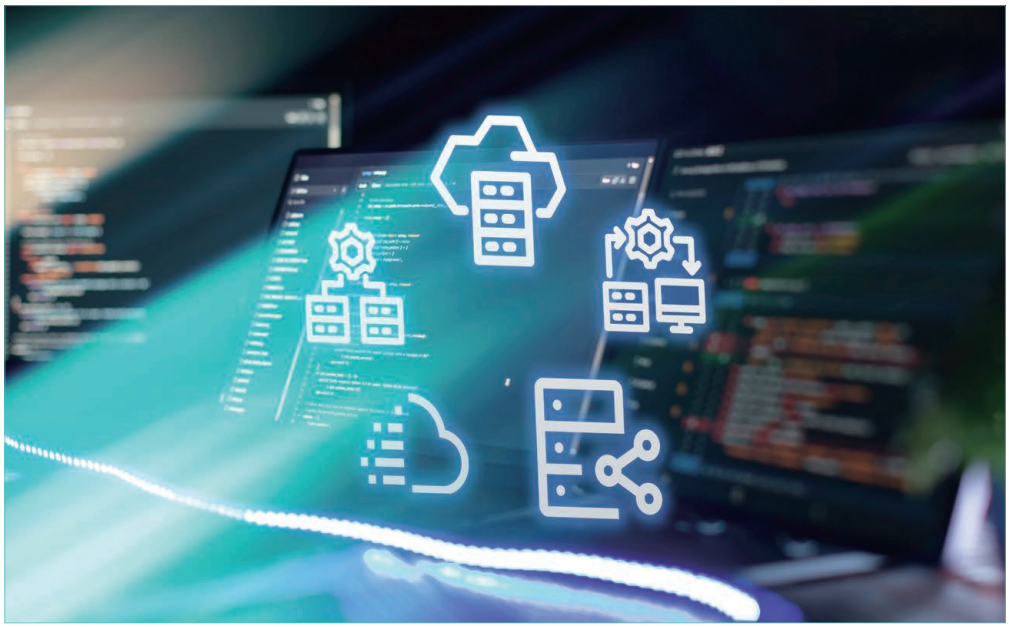
6 클라이언트의 응답 처리

- 클라이언트는 서버로부터 수신한 BOOTREPLY 메시지를 처리
- 메시지에 포함된 IP 주소, 서버 정보, 부트 파일명, 네트워크 설정값 등을 저장
- 정보가 정상적으로 수신되었는지 확인한 후, 부팅 과정의 다음 단계로 진행

7 클라이언트의 부팅 완료

- 클라이언트는 BOOTP에서 제공받은 부트 파일명을 이용하여 TFTP(Trivial File Transfer Protocol) 등의 네트워크 파일 전송 프로토콜을 사용해 OS를 다운로드
- 다운로드한 OS를 실행하여 부팅을 완료

Bootstrap Protocol



BOOTP의 발전과 DHCP의 개발

BOOTP는 1980년대 중반부터 1990년대 중후반까지 TCP/IP 환경에서 가장 널리 사용된 호스트 구성 프로토콜이었습니다. 이렇듯 매우 성공적이었음에도 불구하고 몇 가지 근본적인 한계가 있었습니다. 그 중에서도 가장 중요한 문제는 동적 주소 할당(Dynamic Address Assignment)을 지원하지 않는다는 점이었습니다. 1990년대 인터넷의 폭발적인 성장과 함께 동적으로 IP 주소를 할당할 수 있는 기능이 절실히 필요해졌고, 이는 DHCP(Dynamic Host Configuration Protocol, 동적 호스트 구성 프로토콜)의 개발로 이어졌습니다. 다음 편부터 오늘날 우리가 매일 사용하고 있는 DHCP에 대해 하나씩 알아보도록 하겠습니다.

P.S.

C군이 여러분께 전하는 내용 중 전문적 성격이 짙은 것은 엄밀한 언어를 사용하여 설명하기에는 한계가 있습니다. 본 내용은 설명하는 대상에 대한 전체적 맥락의 이해에만 이용하시고, 그 이상은 권위 있는 전문자료를 참고하시기 바랍니다.