웹서버와 WAS

Tags	
S/NS	Done

웹

웹을 지탱하고 구성하는 3가지 기술: HTTP(통신 규칙), URL이나 URI(주소), HTML(내용)

서버

네트워크를 통해 클라이언트에게 정보나 서비스를 제공하는 컴퓨터 시스템

웹서버

인터넷을 기반으로 클라이언트에게 웹 서비스를 제공하는 컴퓨터

> 클라이언트의 입장: 웹 서버에게 주소(url)을 가지고 통신 규칙(http)에 맞게 요청하면, 알맞은 내용(html)을 응답 받음
>
> **서버** 입장: 클라이언트의 요청을 기다리고, **웹 요청(http)에 대한 데이터**를 만들어서 응답, 이때 데이터는 웹에서 처리할 수 있는 html, css, 이미지 등 **정적인 데이터로 한정**

WAS

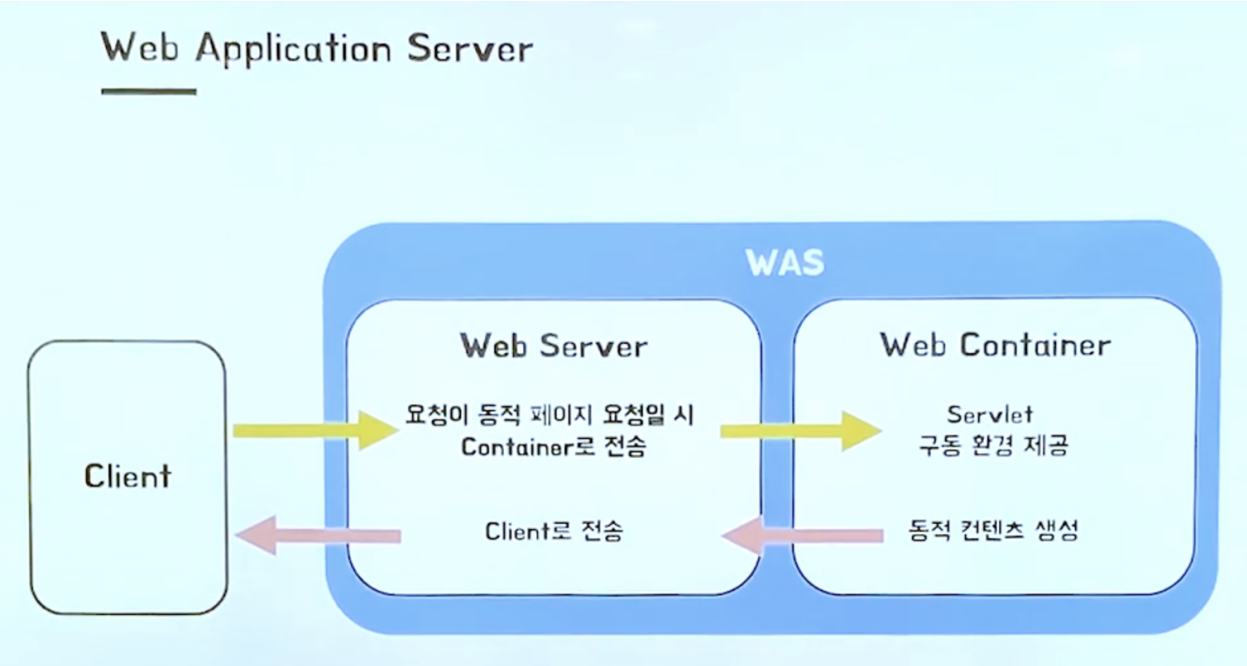
웹에서 실행되는 응용 프로그램

애플리케이션을 통해서 html의 한계를 극복할 수 있음

- WAS는 웹 애플리케이션을 실행시켜 필요한 기능을 수행하고 그 결과를 웹 서버에게 전달하는 일종의 미들웨어
- php, jsp와 같은 언어들을 사용해 동적인 페이지를 만들어낼 수 있는 서버

- 웹 서버+웹 컨테이너를 합친 형태

WAS 동작 방식



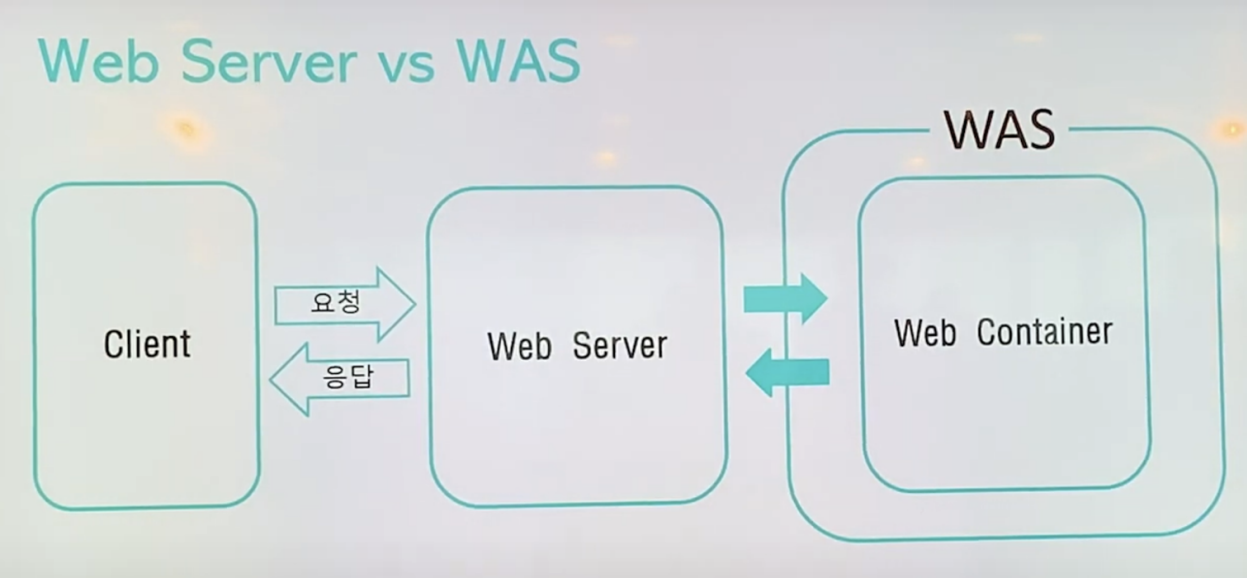
웹서버와 WAS종류

웹서버 종류: 아파치, Nginx

WAS 종류: 톰캣

웹서버와 WAS를 따로 사용하는 이유

- 서로의 기능을 분리하여 서버 부하를 방지할 수 있음
- 물리적으로 분리하여 보안을 강화할 수 있음
- 여러대의 was를 연결할 수 있음



로드 밸랜싱

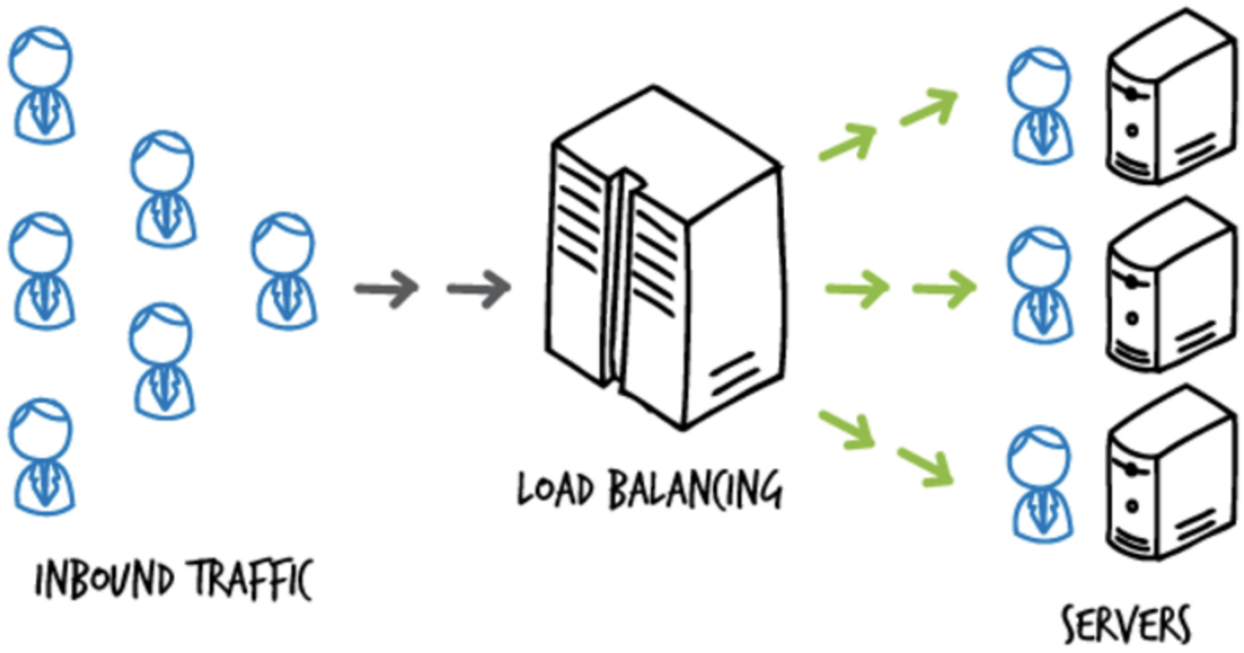
컴퓨터 네트워크 기술의 일종으로 작업, 부하를 나누는 것

→ 가용성 및 응답시간을 최적화할 수 있음

분산 처리는 부하 분산 Network Switch 혹은 소프트웨어가 담당한다

즉, 외부로부터의 요청을 서버가 직접 받는 것이 아닌 부하 분산 Network Switch 혹은 소프트웨어가 받은 후 이를 서버에 적절히 나누어 주는 것이다

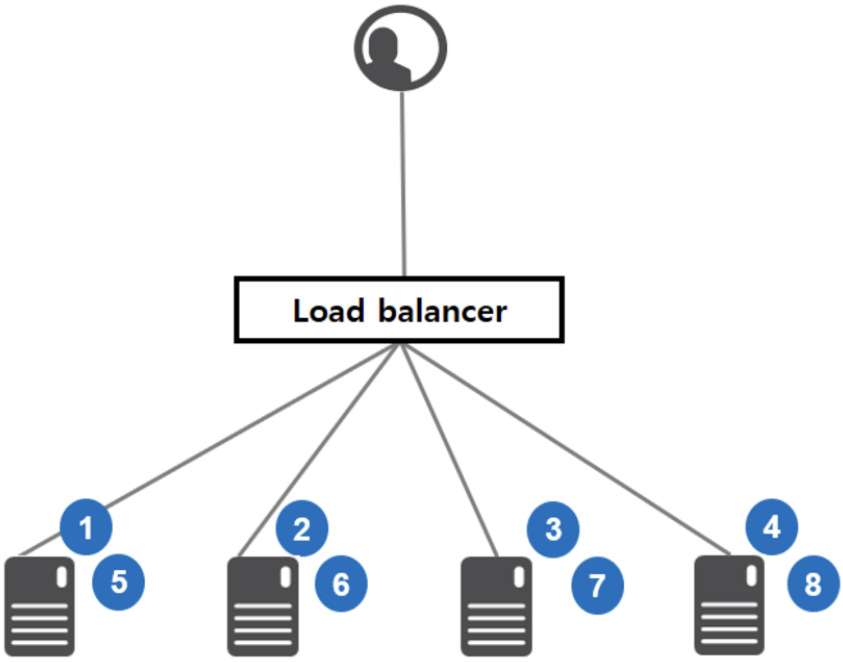
Network Switch를 L4/L7 Switch(layer 4)라고 부르며, 클라우드에서는 로드 밸런서(LB)라고 부릅니다



방법

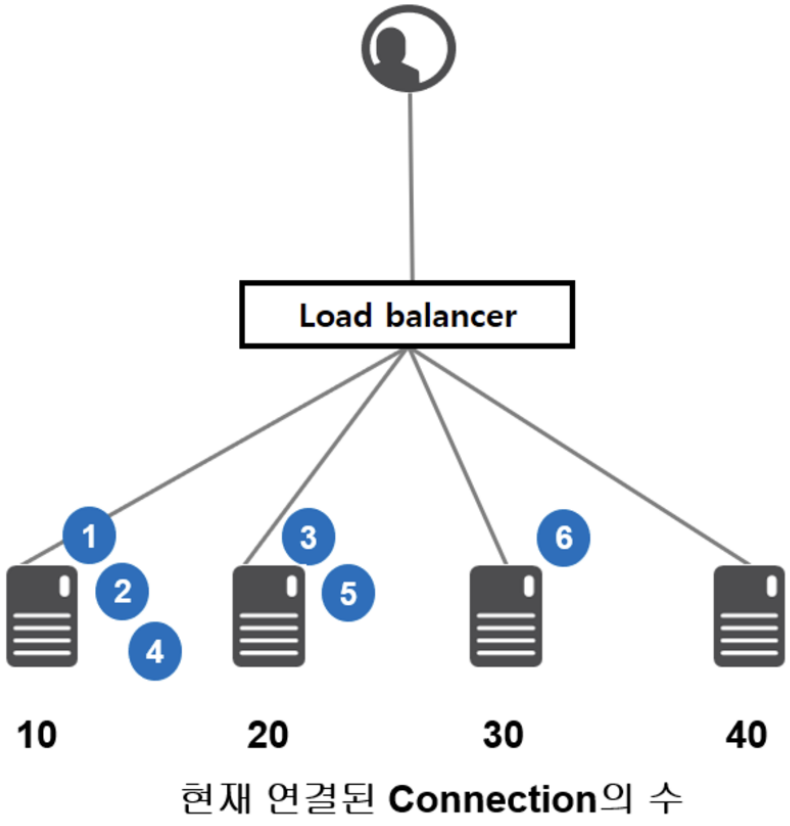
1. Round Robin

   로드 밸런서가 다수의 서버에게 순서대로 요청을 할당하는 방법



2. least Connection

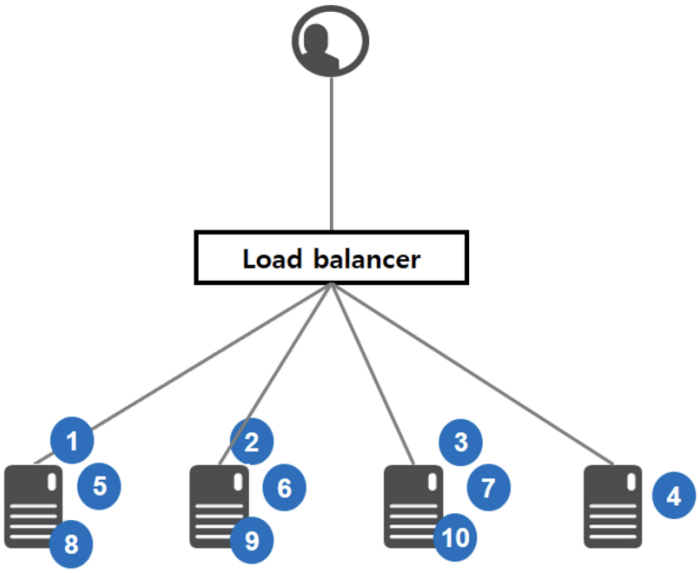
   로드밸런서가 서버에게 요청을 전달한 뒤, 사용자와 서버가 정상적인 연결을 맺으면 사용자와 서버는 Connection을 함, Connection이 적은 서버, 즉 부하가 가장 덜한 서버에게 요청을 함



3. Ratio(가중치)

   서버의 처리 능력을 고려하여 할당될 수 있는 각 서버가 가질 수 있는 Connection 비율을 정해둠

   성능이 가장 떨어지는 서버에게 10%, 나머지 서버 3대에 각각 30%를 할당



4. Fastest(Response Time)

   응답속도가 가장 빠른 서버에게 우선적으로 할당하는 방식

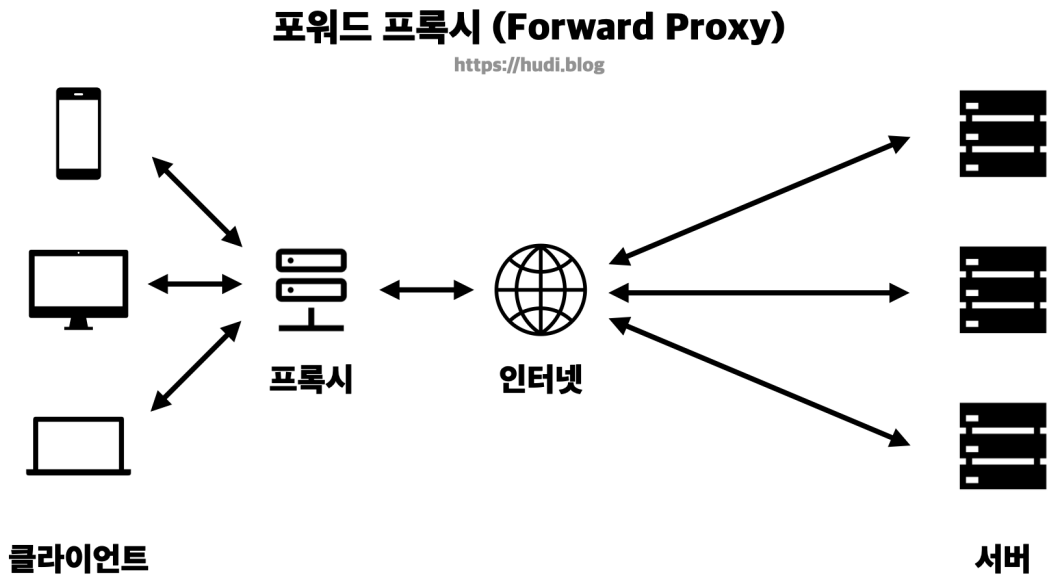
주요 기능

- 부하 분산
- 오토 스케일링: 조건에 맞춰 필요시 서버에 컴퓨터 수를 늘리거나 줄여서 부하 관리가 가능함(scale in, scale out)
- health check: 해당 포트에 트래픽을 보내애플리케이션이 올바르게 작동하는지 여부 판별
- 보안 서비스 WAF, NAT
  - WAF(Web Application Firewall): 웹의 비정상 트래픽을 탐지하고 차단하기 위한 방화벽
  - Cloud NAT(NetworkAddress Translation): 외부 연결에 노출되는 IP를 관리하여 위험 최소화

❓ nginx를 통해 reverse proxy를 적용한 경험이 있는데, reverse proxy가 무엇인지!!

프록시 서버

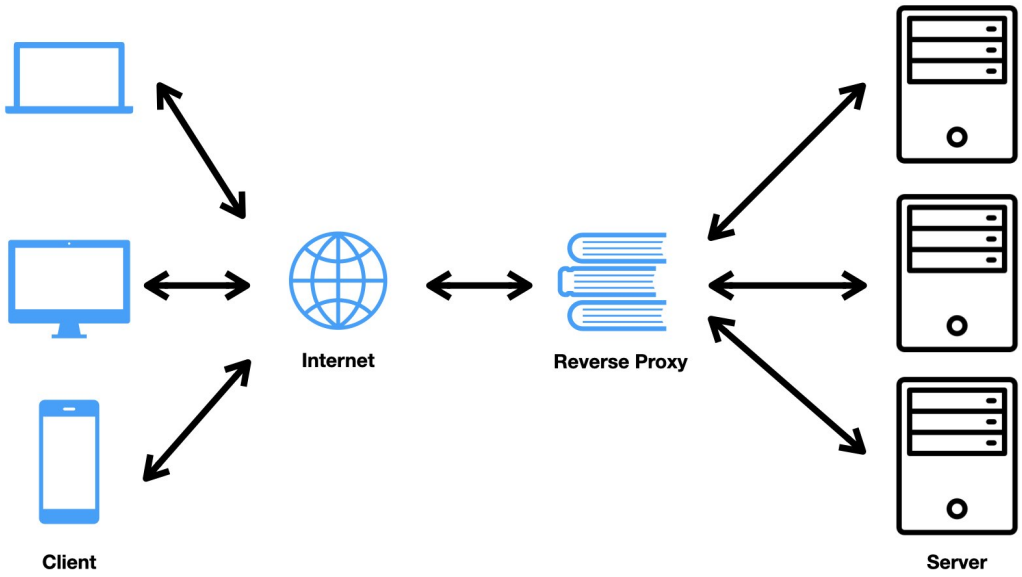
1. 포워드 프록시 서버



1. 캐싱: 클라이언트가 요청한 캐싱
2. 익명성: 클라이언트가 보낸 내용을 숨김

2. 리버스 프록시 서버

Hamad



1. 캐싱
2. 보안: 서버 정보를 클라이언트로부터 숨길 수 있음, 실제 서버의 ip가 노출되지 않음
3. load balancing: 부하분산

❓ 로드밸런싱을 하는데 서버a는 사용자1의 정보를 가지고 있는데, 사용자1의 정보를 가지고 있지 않는 서버b로 로드밸런싱이 되면 어떻게 해?

해결 방안

1. 세션 유지 (Session Persistence)

- **IP 해시 방식**: 클라이언트의 IP 주소를 해싱하여 동일한 서버로 요청을 지속적으로 보내는 방법입니다. 이를 통해 사용자1의 요청이 항상 서버 A로 전달되도록 할 수 있습니다[3][6].
- **쿠키 기반 세션 유지**: 클라이언트에게 특정 서버에 대한 정보를 쿠키로 저장하여 동일한 서버로 요청을 보낼 수 있도록 합니다[4].

2. 데이터 동기화

- **세션 클러스터링**: 모든 서버가 사용자 데이터를 공유할 수 있도록 세션 데이터를 중앙 저장소(예: 데이터베이스, 캐시)에 저장하여 각 서버가 동일한 데이터를 접근할 수 있게 합니다[2].
- **데이터 복제**: 사용자 데이터를 모든 서버에 복제하여 어느 서버로 요청이 가더라도 일관된 데이터를 제공할 수 있도록 합니다[5].

3. 로드 밸런싱 알고리즘 선택

- **L7 로드 밸런서 사용**: HTTP 헤더나 쿠키 정보를 기반으로 트래픽을 분산하여 세션 유지를 보다 세밀하게 관리할 수 있습니다[6].
- **동적 로드 밸런싱**: 최소 연결 수나 최소 응답 시간과 같은 동적 알고리즘을 사용하여 실시간으로 가장 적합한 서버에 트래픽을 분산합니다[5][6].

L2: 맥주소

L3: IP주소

L4: IP와 Port level에서 로드밸런싱을 함

L7: application level에서 로드밸런싱(url에 따라서 어떤 서버로 로드밸런싱 할지)

Google cloud Load Balancing GCP 실습용!!!!!!

GCP가 제공하는 로드밸런서에는 3가지 종류가 있다

1. 애플리케이션 레이어 7 부하 분산기
2. TLS 오프로딩을 지원하는 레이어 4 부하 분산기
3. IP 프로토콜을 지원하는 네트워크 부하 분산기