Load Balancer
Network

2025.10.23

CS Study

SSAFY

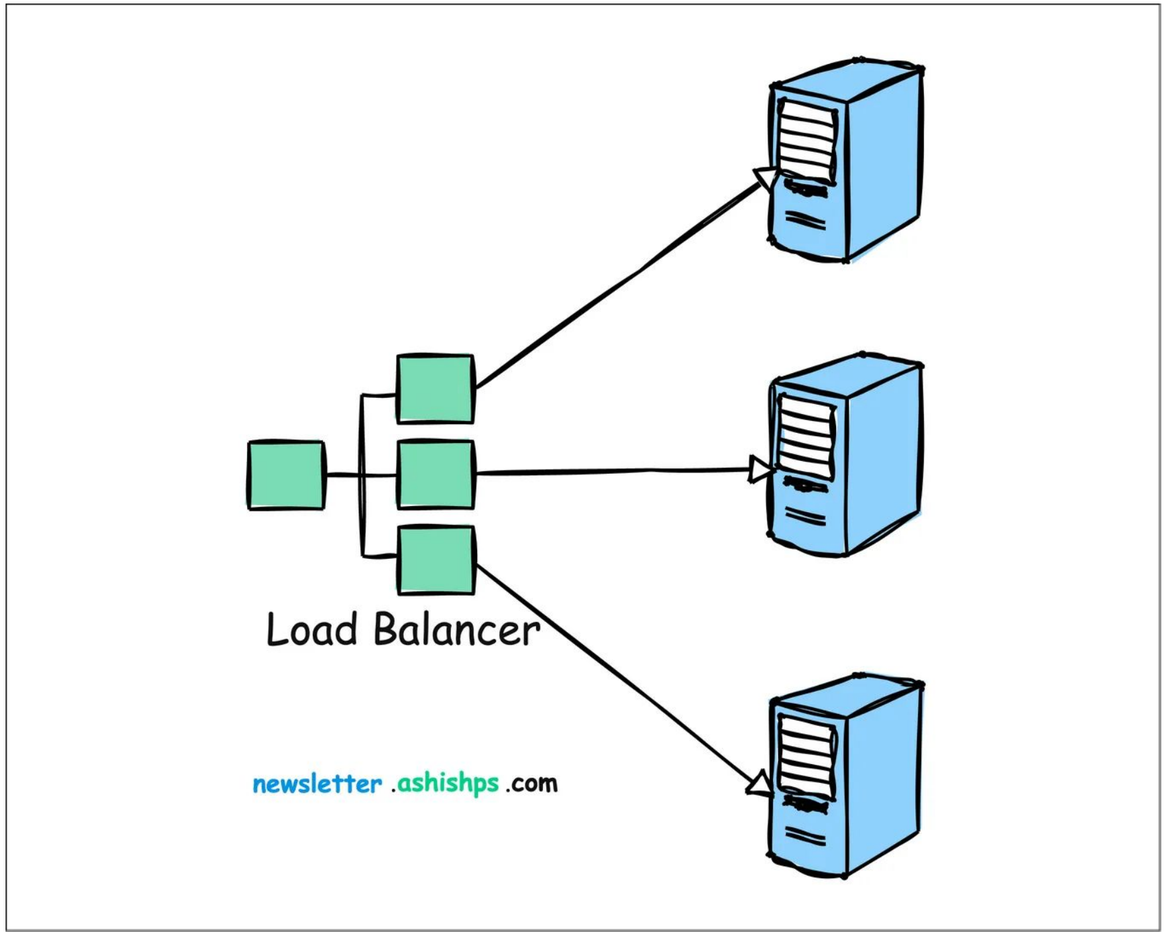
송현우

Index

Load Balancing

Load Balancing

Load Balancing



출처 : https://maily.so/devpill/posts/wjzdevd0o3p

Scalability

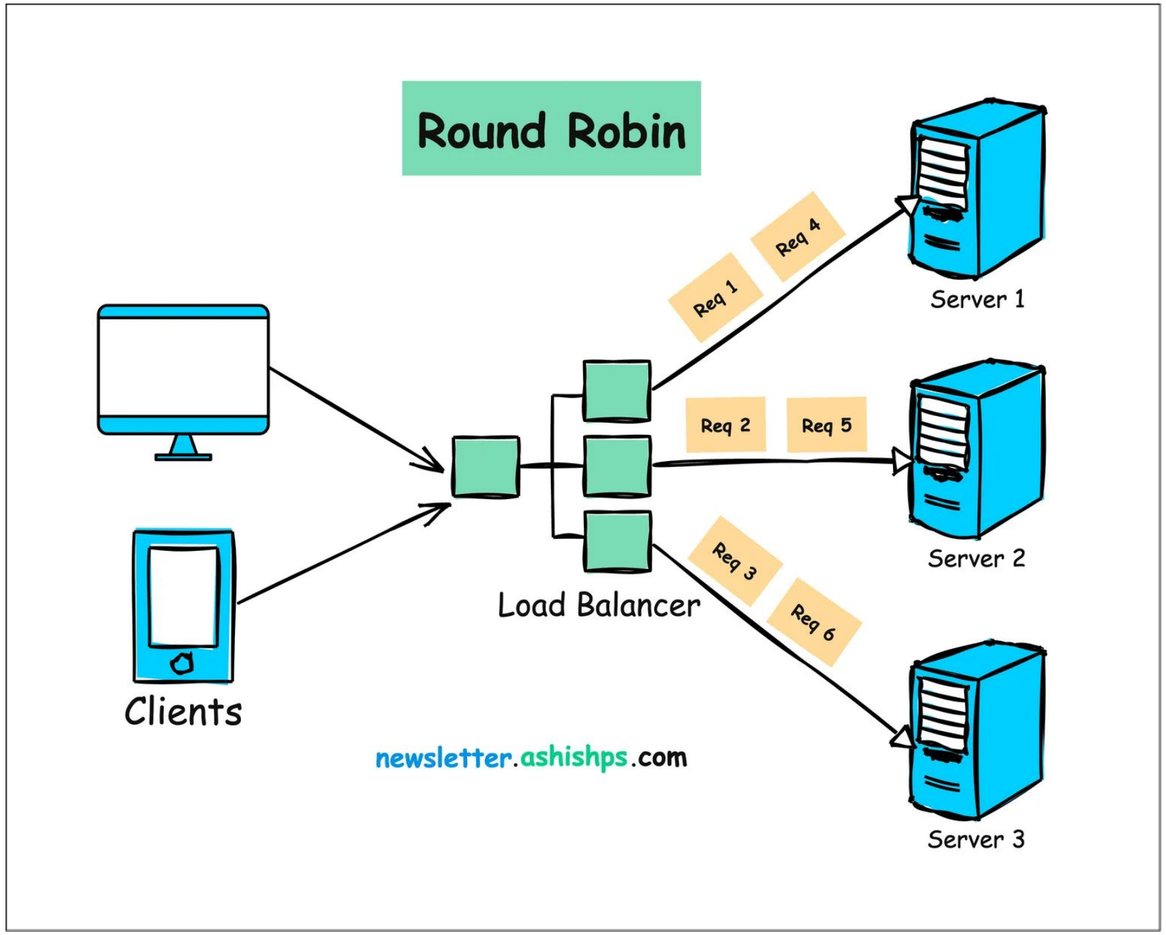
- 시스템이 사용자의 증가나 비즈니스 성장에 따라 늘어나는 작업 부하(Load)에 얼마나 잘 대처할 수 있는지
- 수평 확장(Scale-out)을 통해 확장성을 보장

High Availability

- 시스템이 장애가 발생하는 상황에서도 중단 없이 지속적으로 정상적인 서비스를 제공할 수 있는지
- Health Check, Failover

LB Algorithm - RR

Load Balancing



출처 : https://maily.so/devpill/posts/wjzdevd0o3p

사용 사례

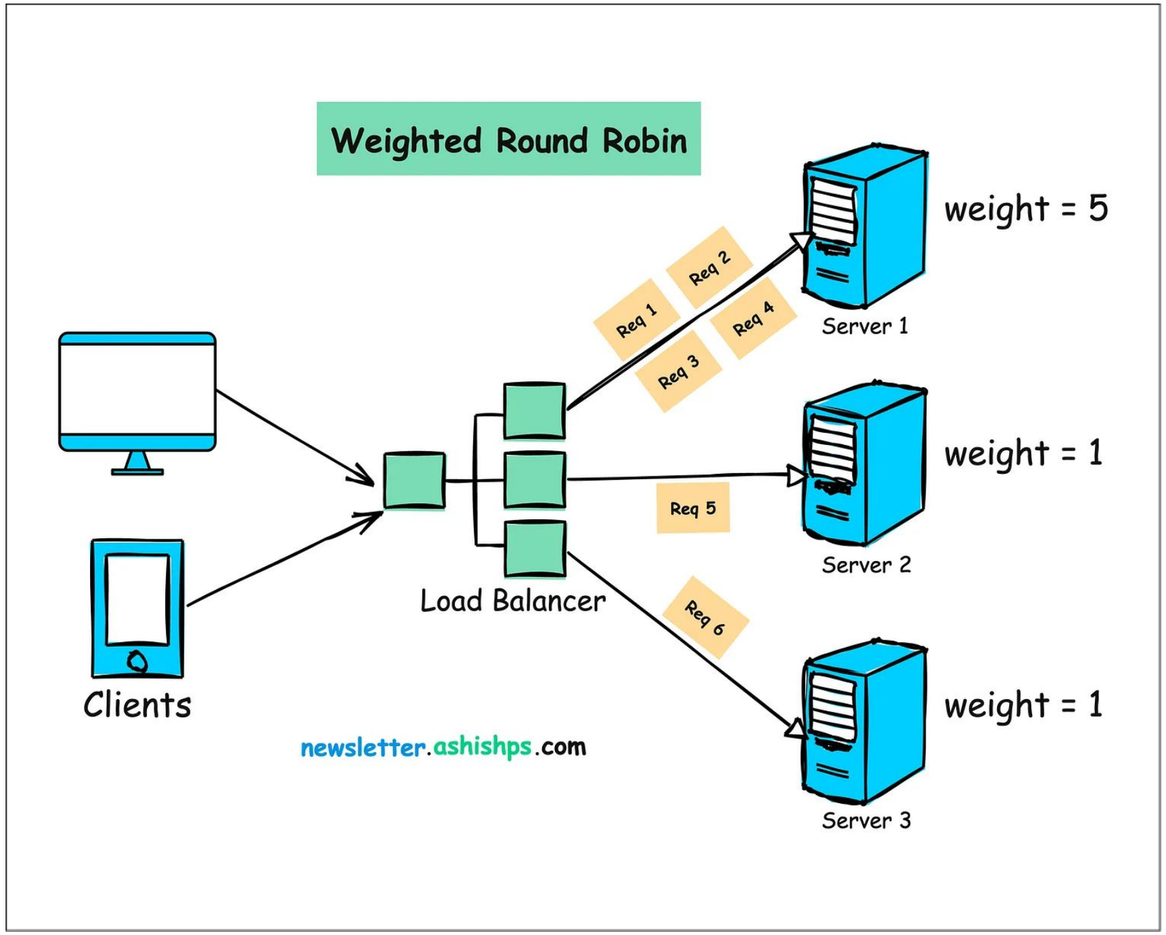
- 모든 서버의 사양과 성능이 유사할 때
- 단순성과 부하의 공평한 분산이 중요할 때
- ALB

한계

- 서버의 현재 상태(부하, 응답 속도)를 고려하지 않는 맹목적인(Blind) 분산
- 서버 간 처리 능력이 불균형할 경우 비효율적

LB Algorithm - Weighted RR

Load Balancing



출처 : https://maily.so/devpill/posts/wjzdevd0o3p

사용 사례

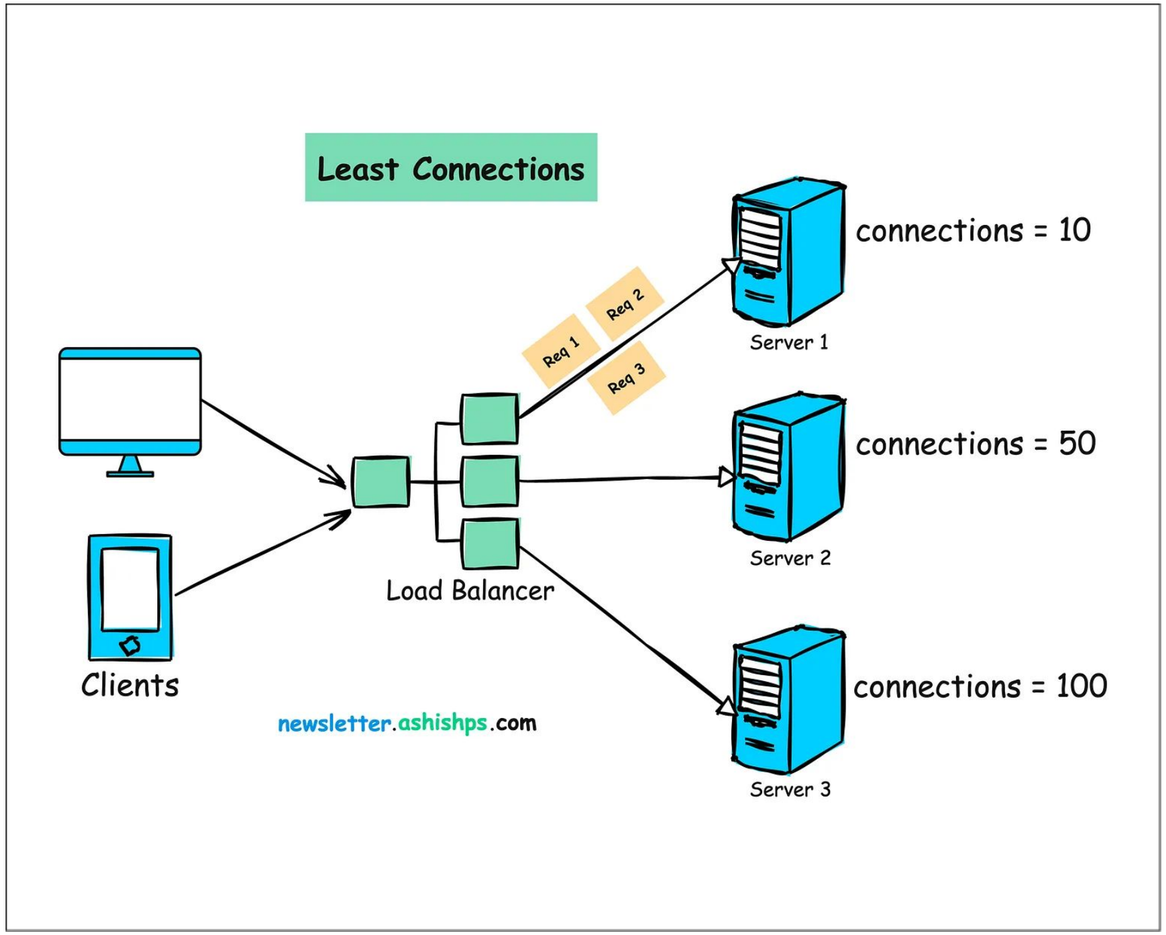
- 서버의 사양이 서로 다른 환경
- 특정 서버에 더 많은 트래픽을 의도적으로 보내고 싶을 때
- ALB(타겟 그룹을 통한 가중치 조절)
- 블루/그린 배포, 카나리 배포 등

한계

- RR과 마찬가지로 서버의 현재 상태(부하, 응답 속도)를 고려하지 않는 맹목적인(Blind) 분산

LB Algorithm - Least Connections

Load Balancing



출처 : https://maily.so/devpill/posts/wjzdevd0o3p

사용 사례

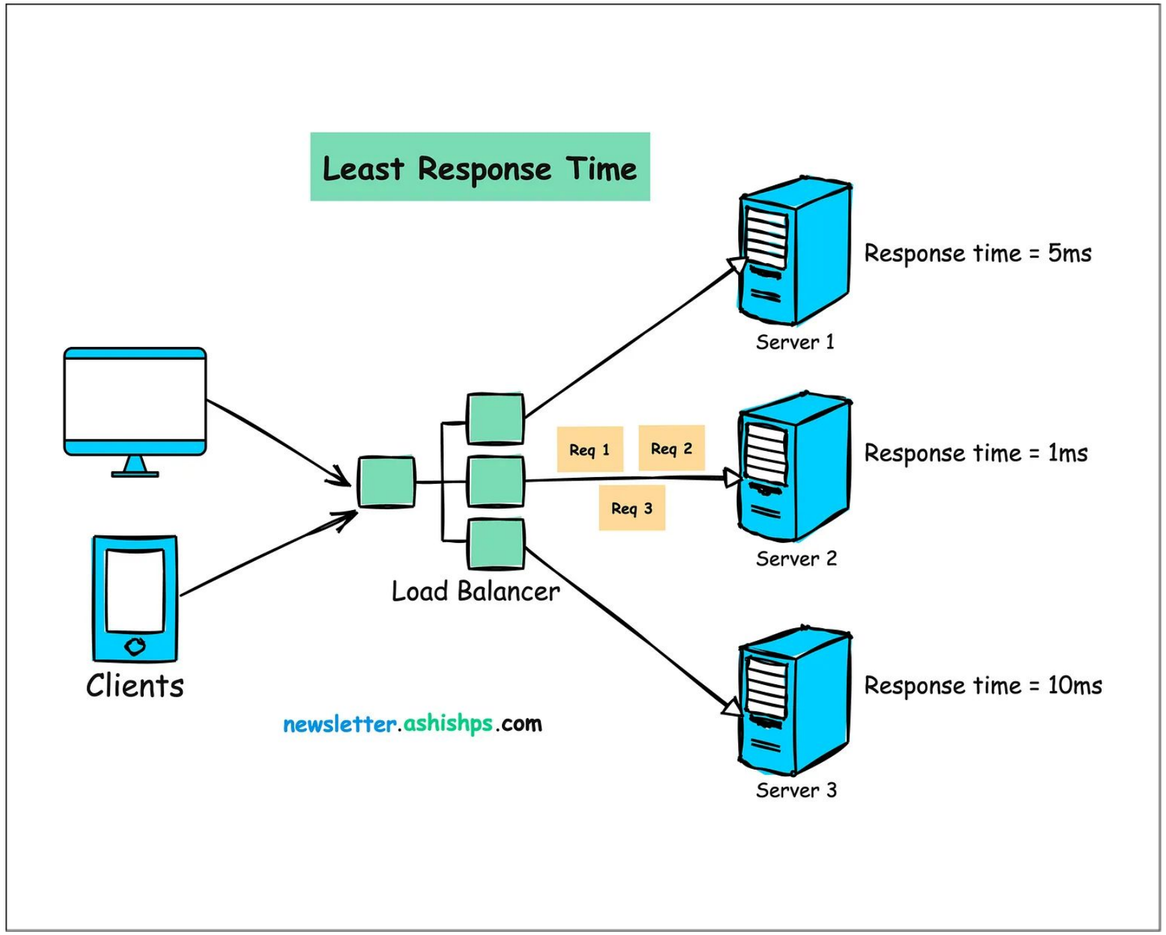
- 서버의 처리 능력은 유사하지만 연결 수준이 다를 수 있을 때(서버의 현재 상태를 고려)
- 요청 별 처리 시간이 서로 다르며 예측하기 어려울 때
- ALB의 LOR(Least Outstanding Requests)과 유사

한계

- 각 서버의 활성 연결 상태를 추적해야함(복잡성 증가)
- 새로운 서버가 추가되면 트래픽이 순간적으로 몰릴 수 있음(Thundering Herd Problem)
  - Slow start duration 지정

LB Algorithm - Least Response Time

Load Balancing



출처 : https://maily.so/devpill/posts/wjzdevd0o3p

사용 사례

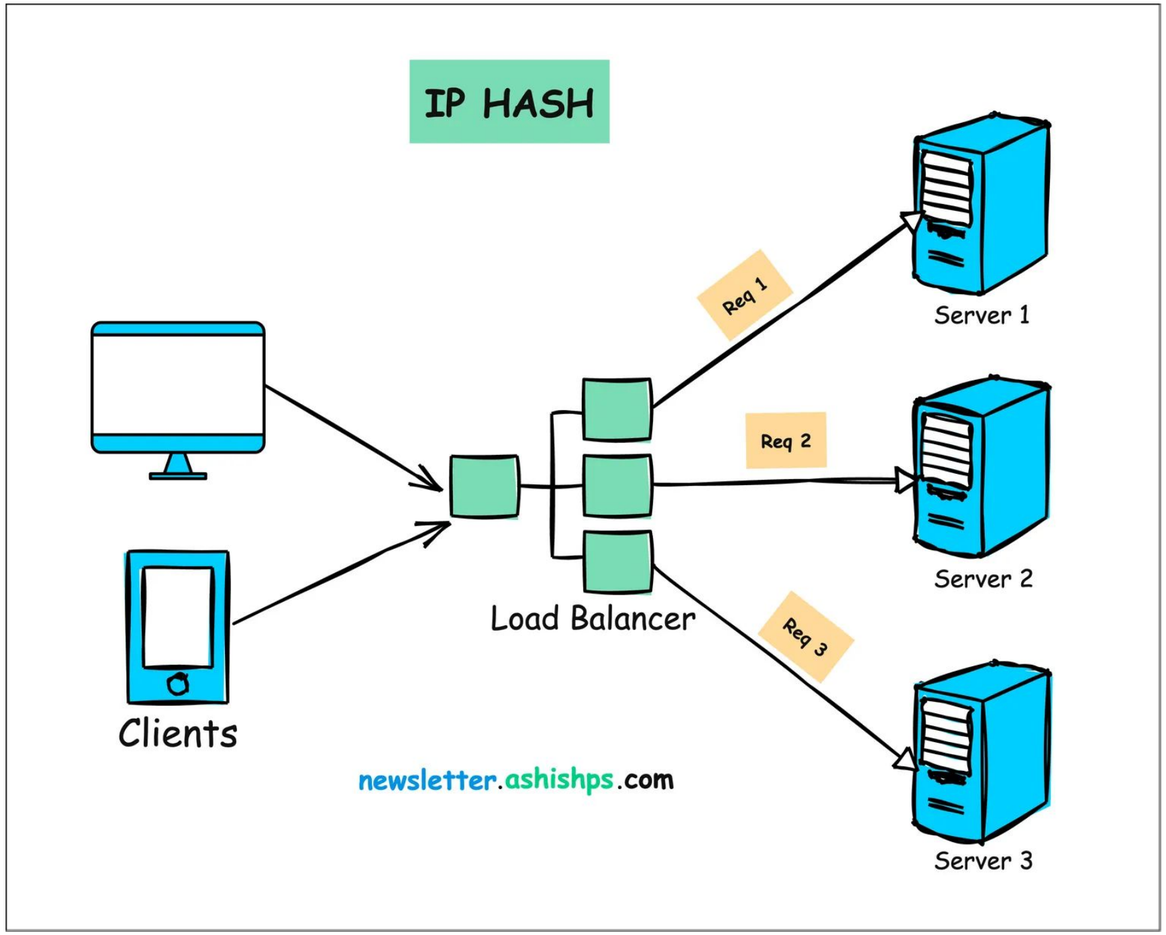
- 사용자에게 가장 빠른 응답 속도를 제공하는 것이 최우선 목표일 때(서버의 현재 상태를 고려)
- 서버의 성능이나 네트워크 상태가 동적으로 변하는 환경

한계

- 각 서버의 현재 응답 시간을 정확하게 추적하기 어려움
  - 응답 시간이 빠르더라도, 실제 API 별로 처리 시간이 다를 수도 있음
- 가장 빠른 서버 한 곳으로 트래픽이 몰릴 수 있음 (Thundering Herd Problem)

LB Algorithm - IP Hash

Load Balancing



출처 : https://maily.so/devpill/posts/wjzdevd0o3p

사용 사례

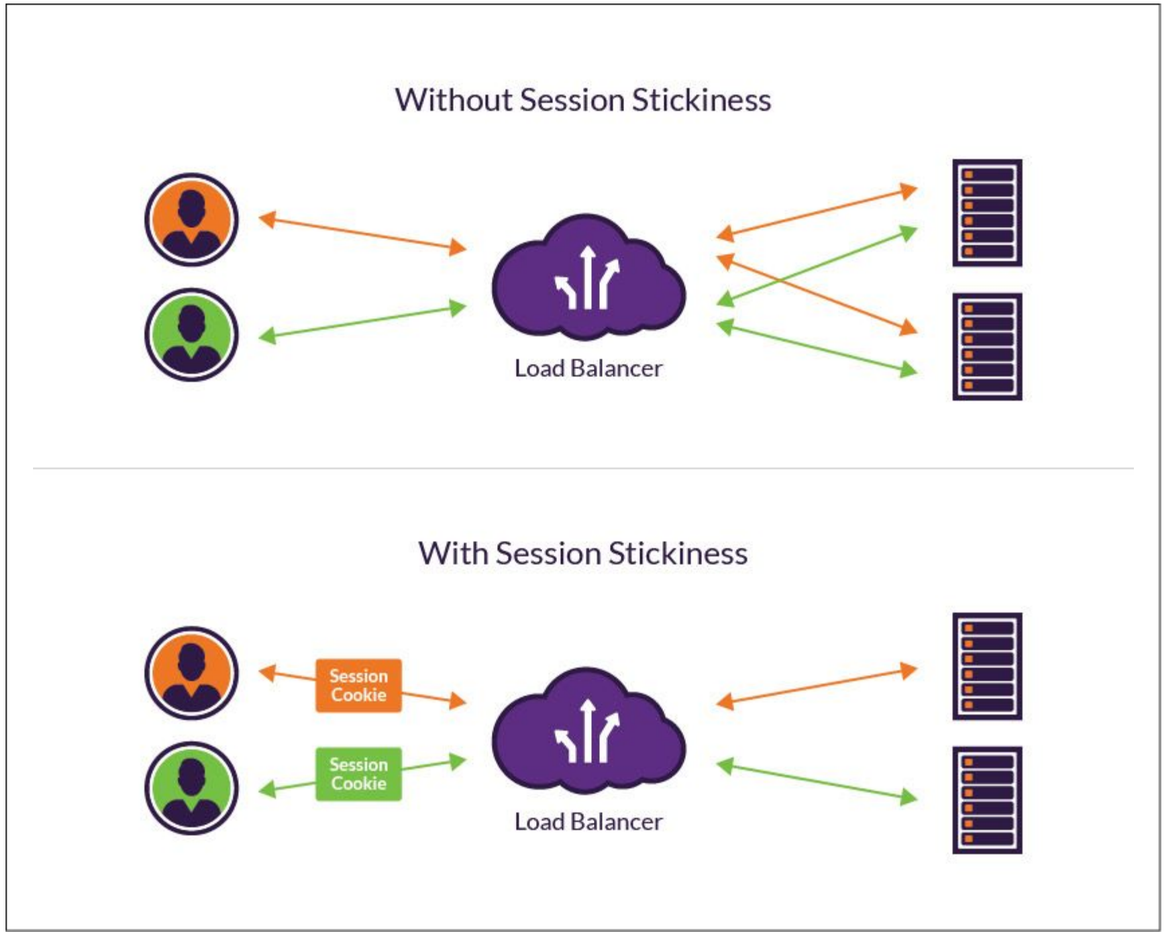
- 세션 지속성이 필요할 때
- ALB(타겟 그룹 바인딩 - Cookie 기준)
- NLB(Flow Hash)
- Software LB(Nginx 등)

한계

- 트래픽 분산이 불균등해질 수 있다.
- 서버 장애 시 유연성이 떨어진다.
    - 해시 테이블 재구성

Sticky Session

Load Balancing



출처 : https://www.imperva.com/learn/availability/sticky-session-persistence-and-cookies/

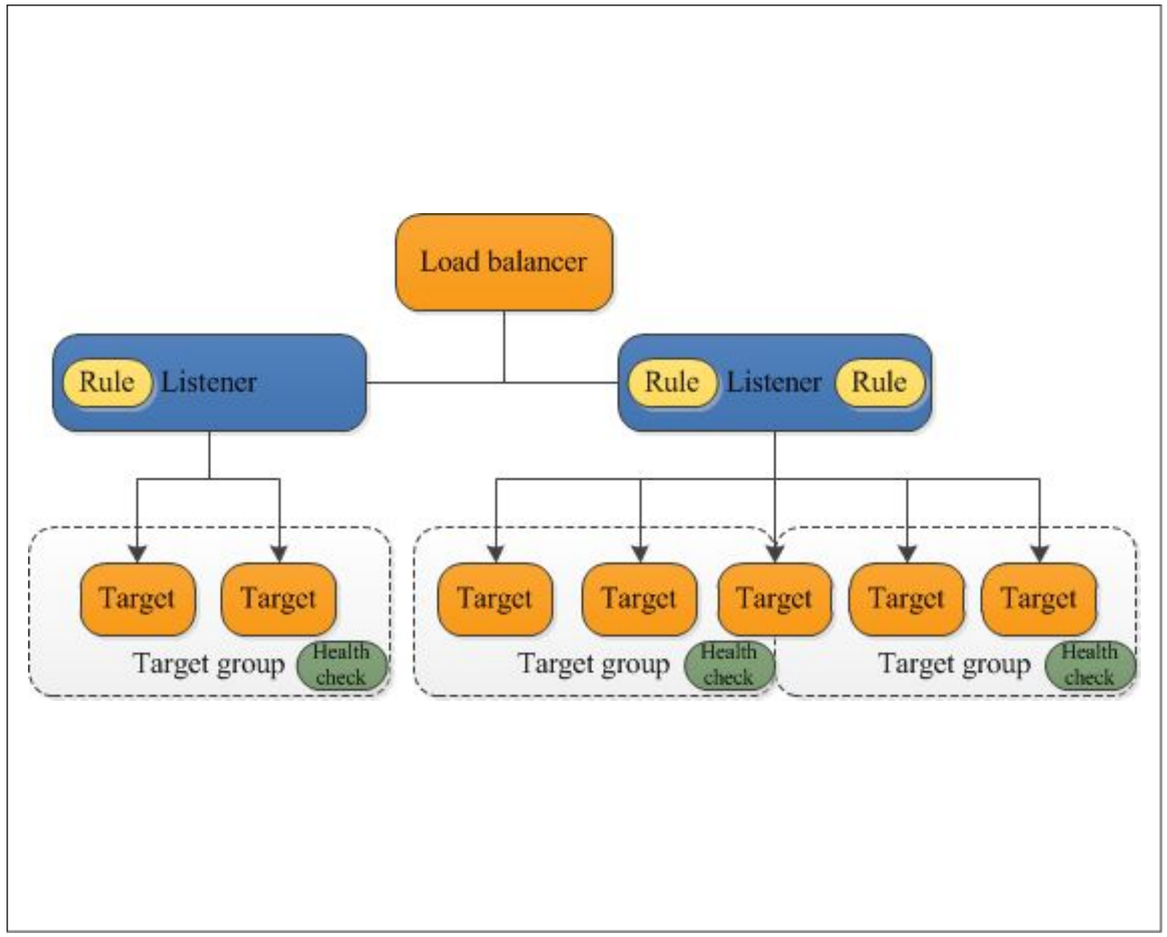
세션 지속성

- 세션 기간(즉, 특정 IP가 웹사이트에 머무르는 시간) 동안 클라이언트와 특정 네트워크 서버 간의 연결 관계를 생성
  - 로드밸런서를 통한 발급 or 애플리케이션을 통한 발급
  - 일반적으로 로드밸런서에서 발급(단순성)
- 네트워크 내의 서버는 세션 데이터를 교환할 필요가 없음
- 서버 사이드에서는 각 관리하는 session의 수를 줄이면서 ram으로부터 세션 데이터를 효율적으로 재사용
- 서버에 세션이 너무 많이 누적되거나 특정 스티키 세션에 많은 리소스가 필요한 경우, 특정 서버가 과부하 상태가 될 수 있음
- 서버 장애 시, 해당 서버의 세션 소실

ALB

Application Load Balancer

ALB



출처 : https://aws.amazon.com/ko/compare/the-difference-between-the-difference-between-application-network-and-gateway-load-balancing/

7-layer LB

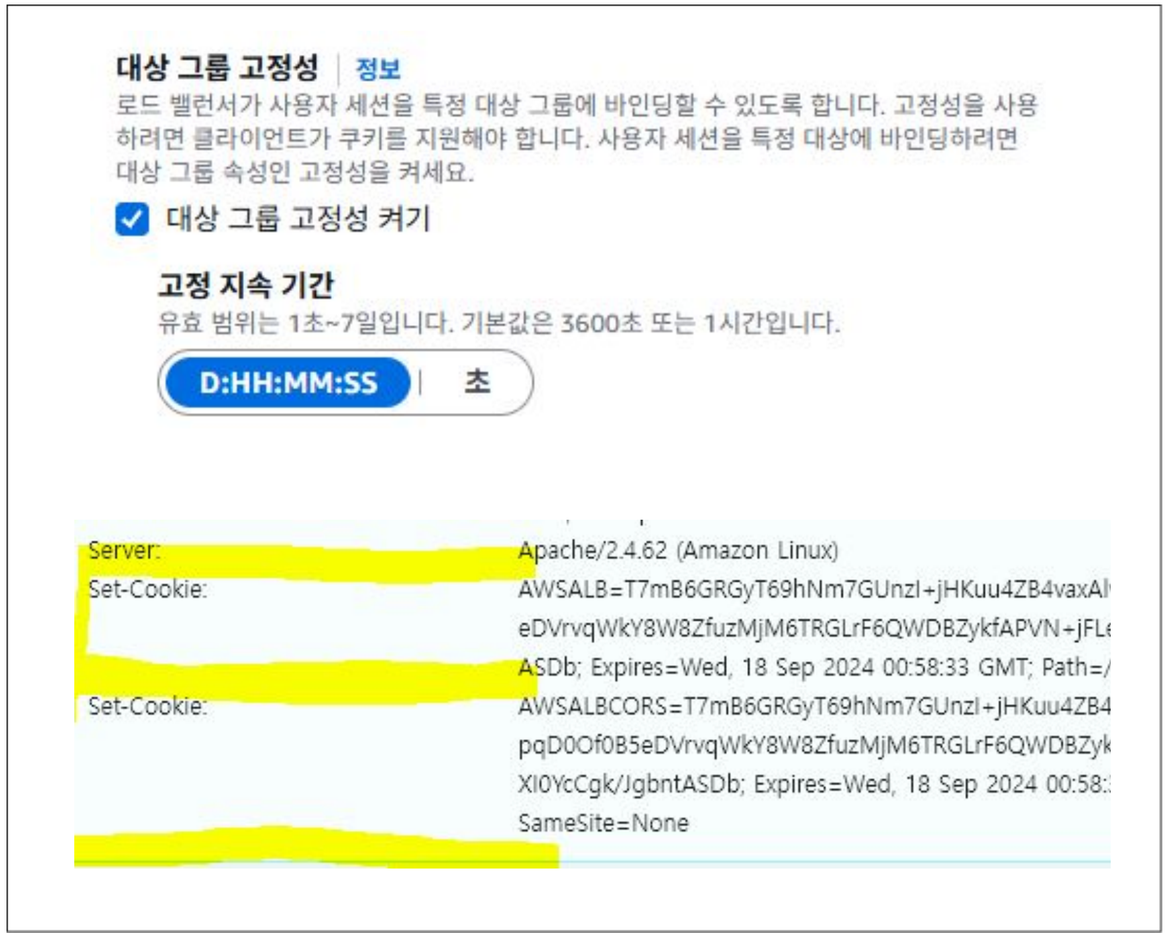
- OSI 7계층에서 동작
- 콘텐츠 기반 라우팅
- TLB 오프로딩, WAF 연동 지원
- RR || Cookie Hash || LOR 알고리즘
- Internet-facing 또는 Internal로 연결

Target Group

- 트래픽을 전달할 대상들의 묶음
- Health Check를 통해 가용성 확보

Sticky Session

ALB

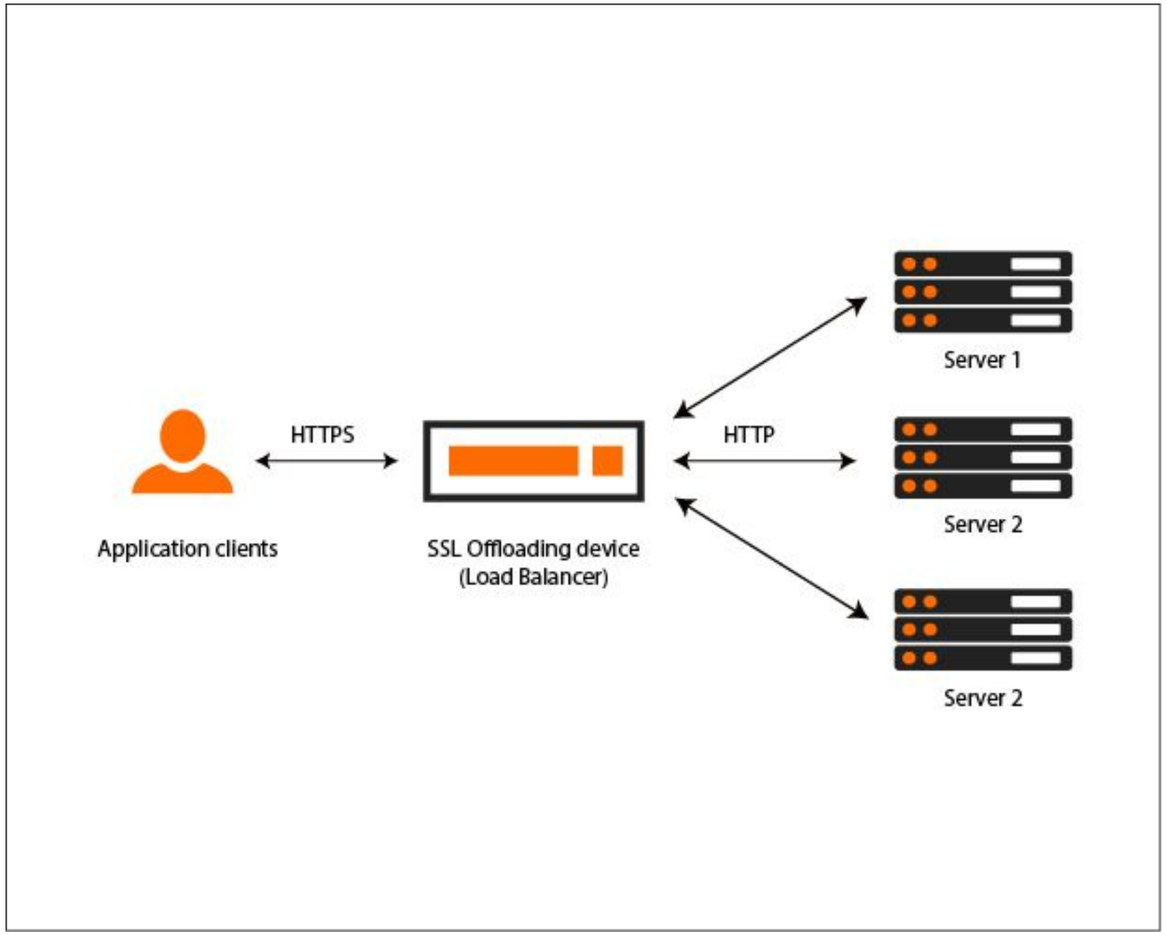


ALB 고정 세션

- Application Load Balancer는 세션 유지를 위해 쿠키(cookie)를 사용하기에 클라이언트에서 쿠키를 지원해야 함
- ALB에서 설정하거나, 애플리케이션에서 관리할 수 있음
- ALB에서 설정한 쿠키는 response 헤더에 AWSALB 쿠키가 자동으로 삽입

TLS/SSL OffLoading

ALB



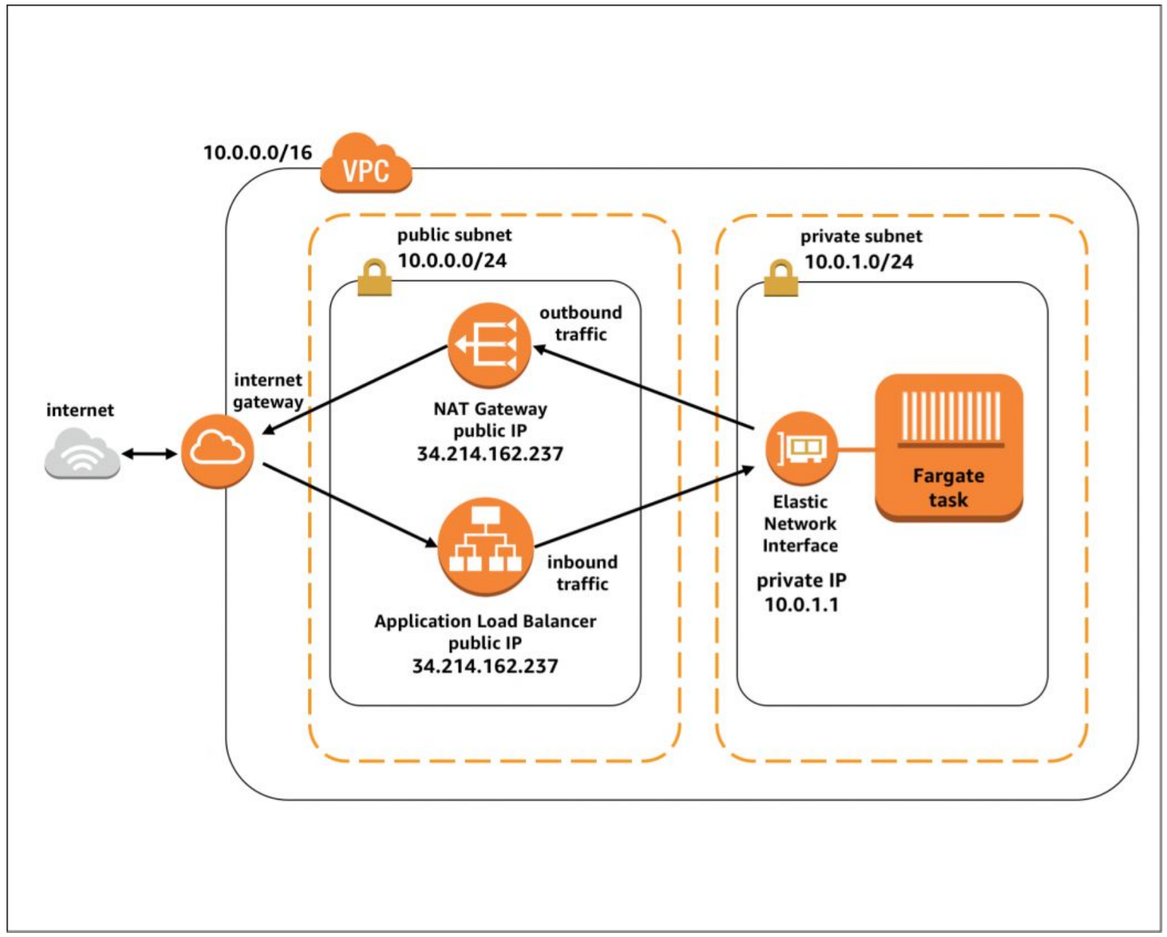
출처 : https://www.appviewx.com/blogs/the-benefits-of-offloading-ssl-certs-on-f5-devices-and-how-to-automate-it/amp/

TLS OffLoad

- HTTPS 암호화 및 복호화 작업을 웹 서버(WAS) 대신 앞단의 로드밸런서나 전용 장비가 대신 처리
  - 관심사 분리
  - 내부에선 HTTP로 통신 가능!!
- COS 같이 내부적으로 certbot 등이 불가능하거나 서버가 분산되어 있다면 효과적
  - Nginx 웹 서버 추가 구성 등 불필요
- 인증서는 ACM(AWS Certificate Manager)사용 또는 클라이언트 인증서 사용 가능

Source IP NAT

ALB



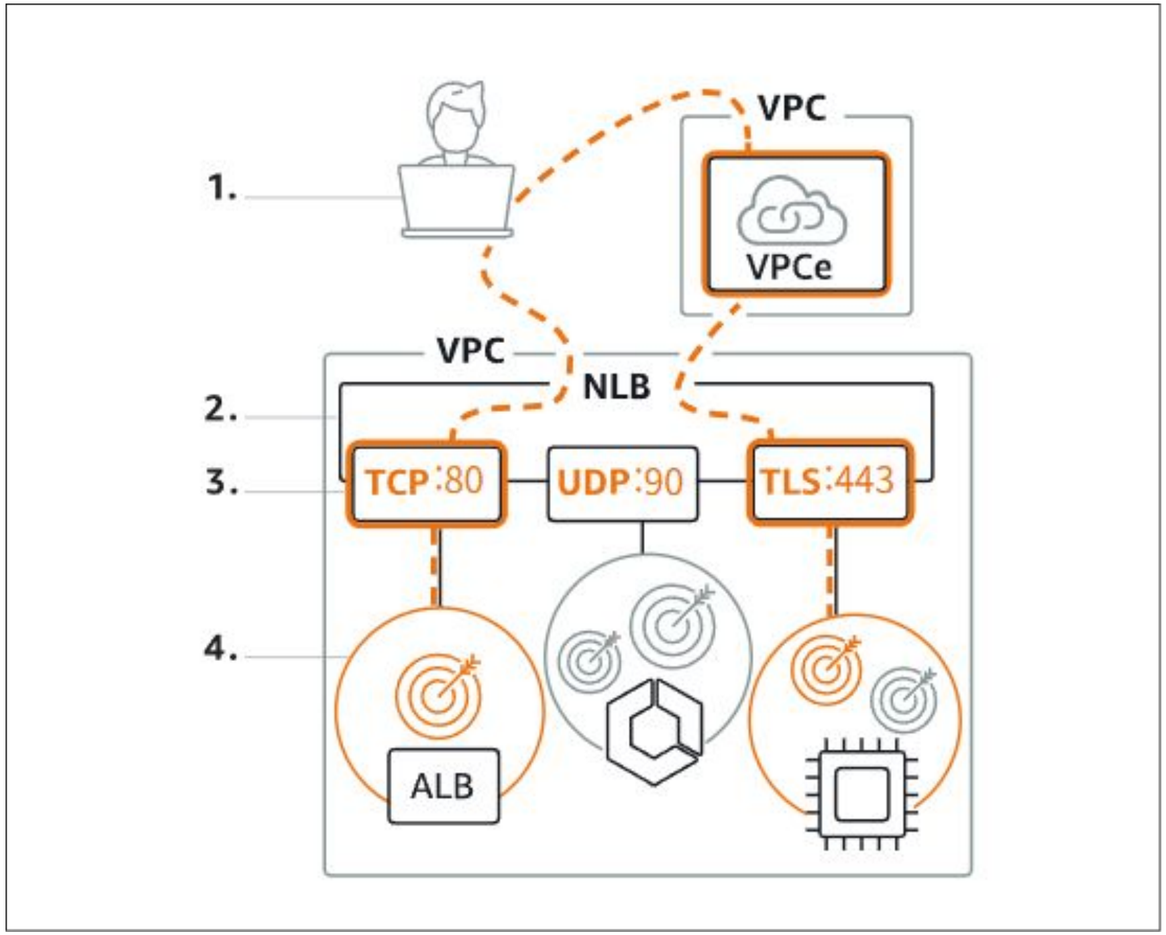
Source IP NAT

- ALB는 일단 사용자와의 연결을 끊고, ALB 자신이 직접 내부 EC2 서버에 새로운 연결을 만든다.
  - Proxy로 동작
- 이때, WAS에서는 이 요청이 사용자가 아닌 ALB의 Private IP 주소에서 온 것처럼 보인다. 대신, ALB는 정보를 HTTP 헤더에 X-Forwarded-For 로 담아 전달해준다.
- 때문에, IP 기반 블랙리스트를 관리할 때는 WAS에서 보이는 Source IP를 차단해서는 안된다!!

NLB

Network Load Balancer

NLB

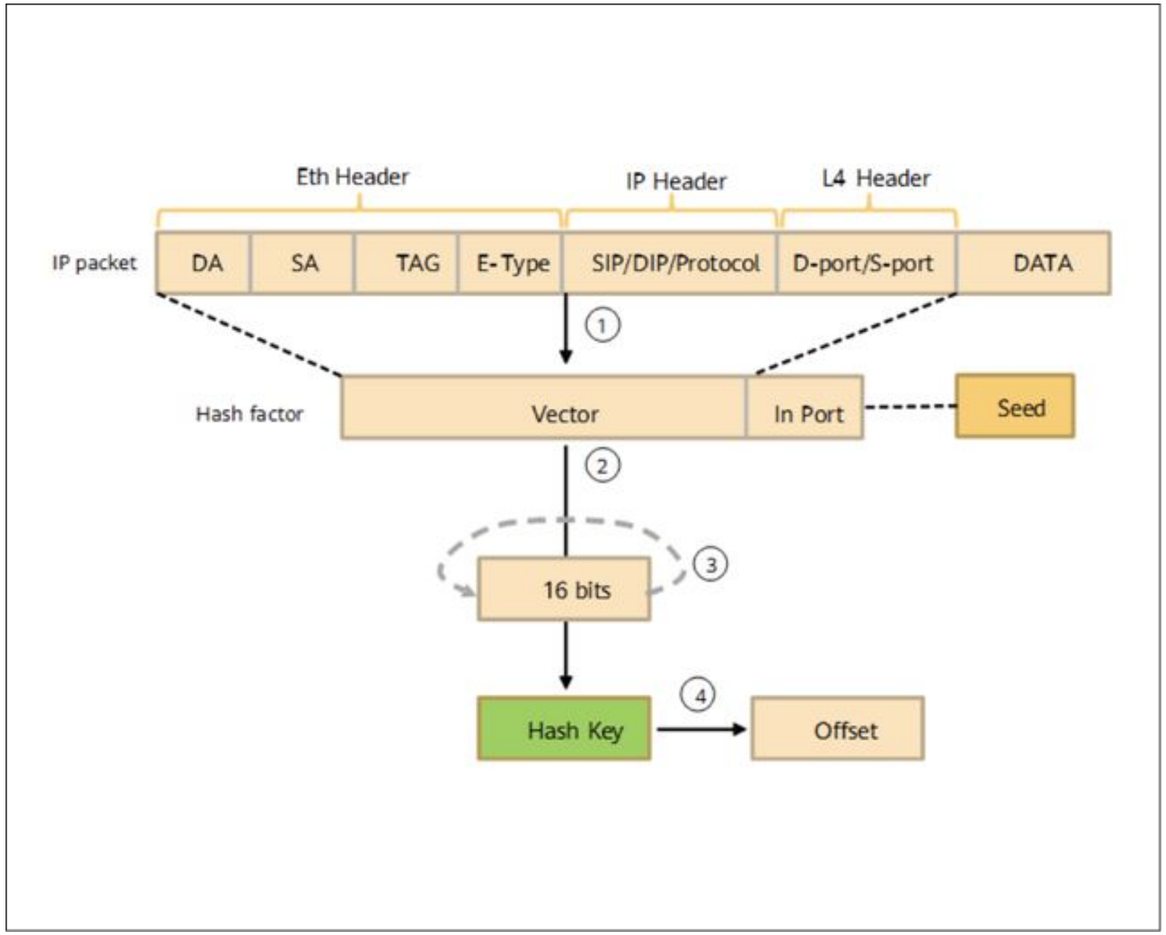


4-layer LB

- OSI 4계층에서 동작
  - 요청의 내용을 보지 않고, IP주소와 Port로 분산
- Internet-facing 또는 Internal로 연결
- 고정 IP 주소 할당 가능
- 대규모 TLS 오프로딩
- 초당 수백만 개의 요청과 갑작스럽고 변동성이 높은 트래픽 패턴을 처리, 대기 시간이 매우 짧음
- Flow Hash 알고리즘
- ALB로의 라우팅 가능!!
- Traffic Flow(DSR 환경)

Flow Hash Algorithm

NLB



NLB Algorithm

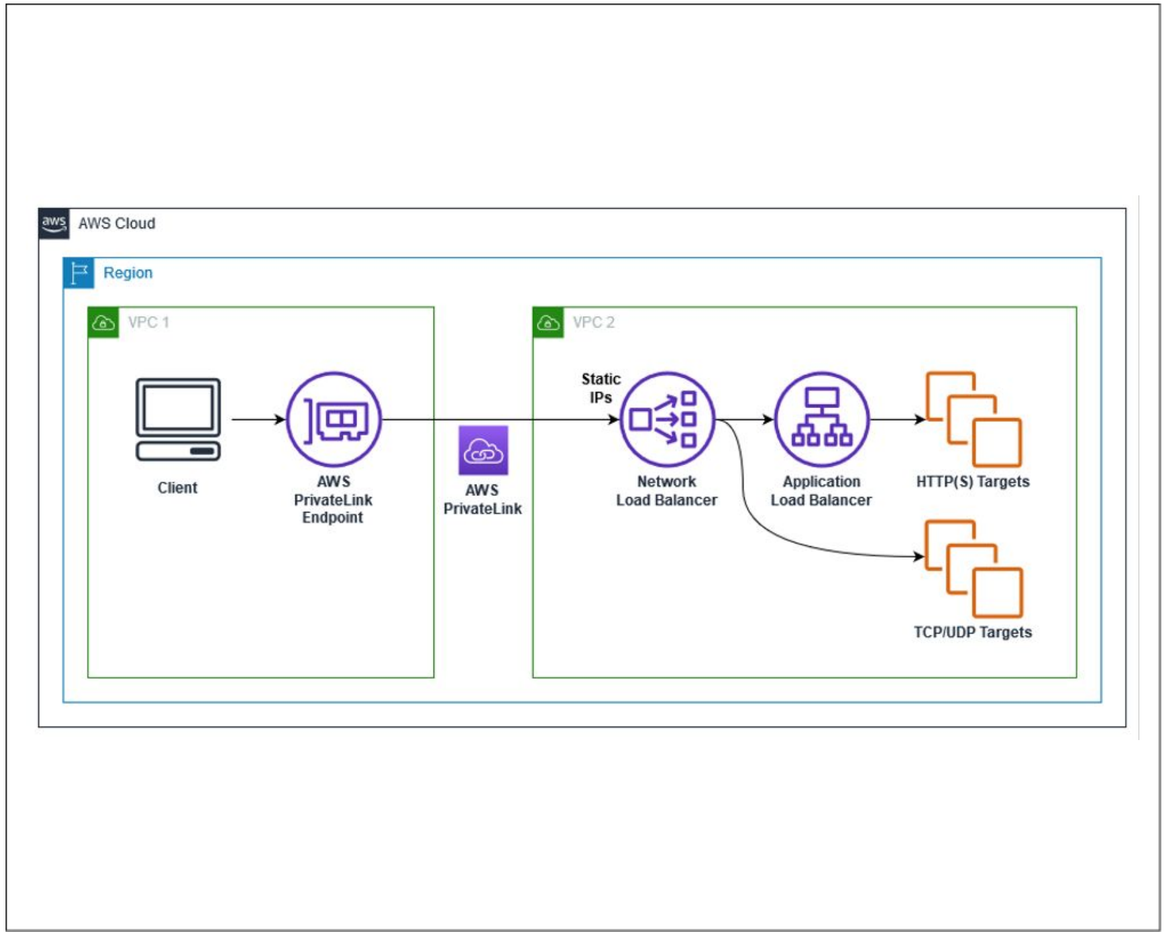
- Source IP&Port, Destination IP&Port, Protocol, TCP sequence number
- 5-Tuple + tcp 시퀀스 넘버를 기준으로 연결을 구분
- Sticky한 방식이지만 기준을 Connection으로 설정
- 단순하고 빠른 라우팅

한계

- 세밀한 제어가 불가능
  - 다양한 트래픽 패턴에 적용 불가

ALB for NLB

NLB



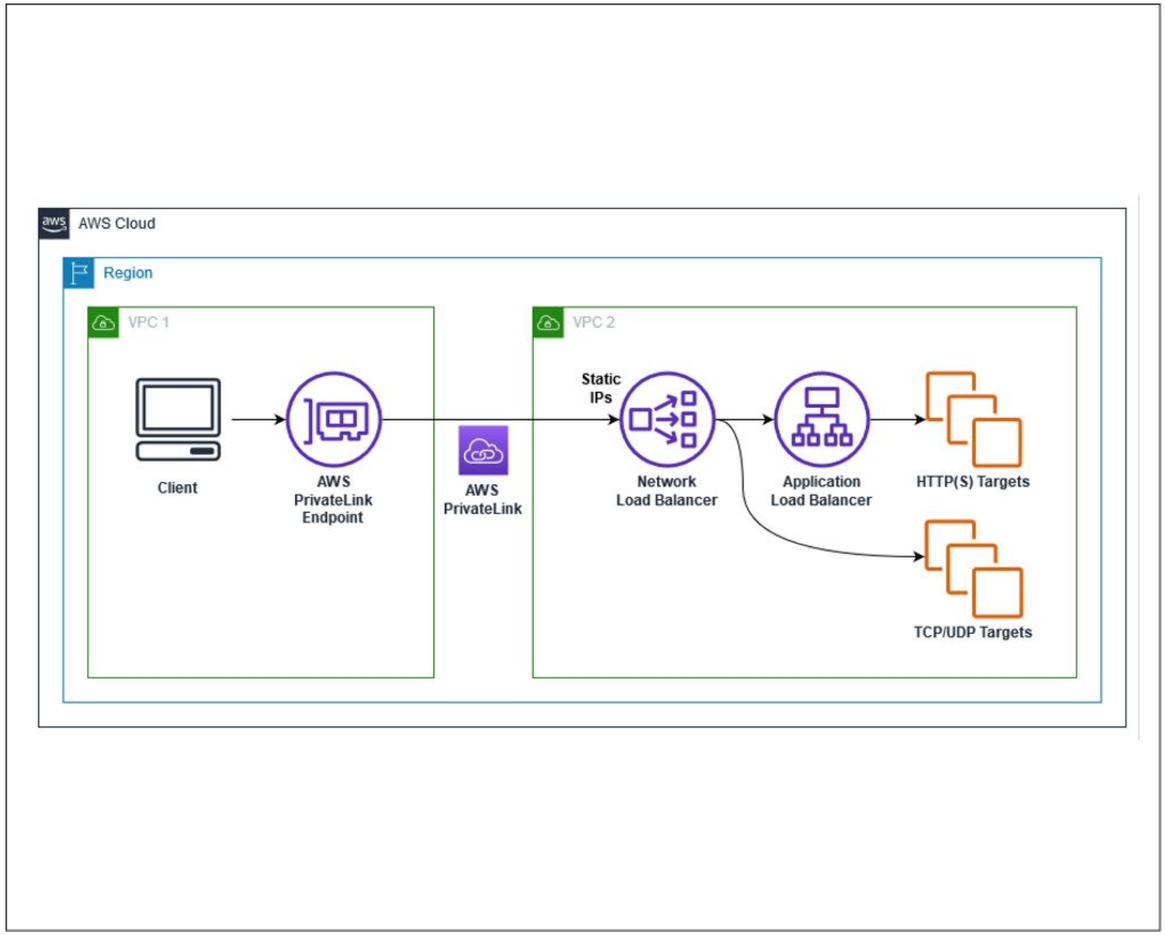
출처 : https://aws.amazon.com/ko/blogs/networking-and-content-delivery/application-load-balancer-type-target-group-for-network-load-balancer/

IP Whitelisting

- 고정 IP 주소가 반드시 필요하지만, 동시에 URL 경로, 호스트 이름 등에 따라 트래픽을 유연하게 제어하고 싶을 때
- 외부측 방화벽에 우리 서비스의 IP를 등록해야만 통신이 가능한 경우
- 기존에는 ALB의 IP 주소가 변경될 때마다 업데이트하는 람다(Lambda) 함수를 직접 만들어서 운영해야했지만, 현재는 타겟 그룹 유형에 ALB가 추가됨

ALB for NLB

NLB



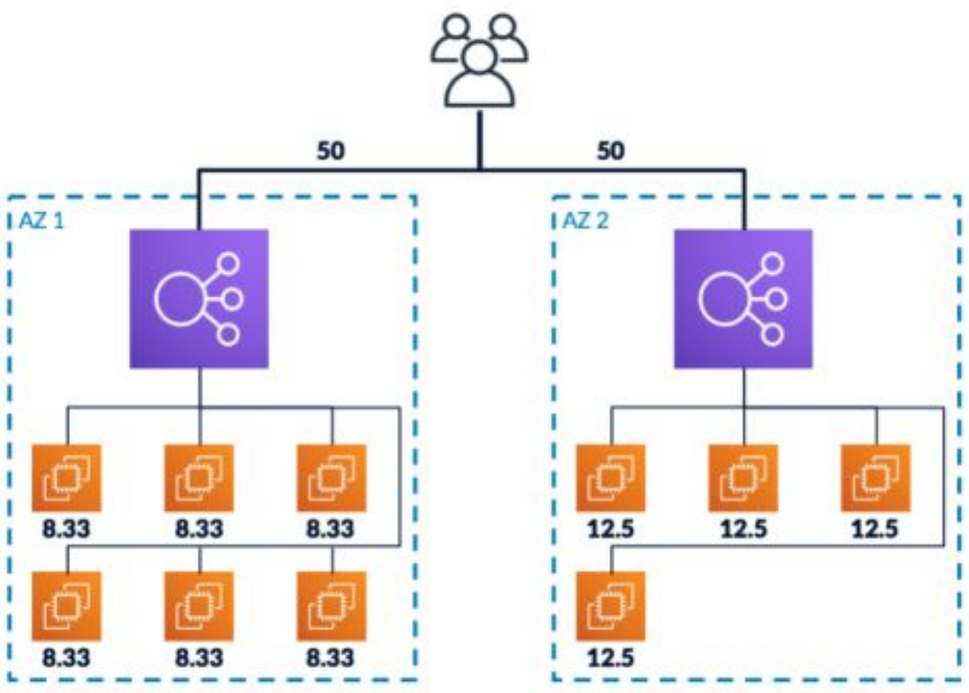
출처 : https://aws.amazon.com/ko/blogs/networking-and-content-delivery/application-load-balancer-type-target-group-for-network-load-balancer/

Things to know

- NLB에서 HTTPS 통신을 처리하는 TLS 리스너는 이 기능을 사용할 수 없다
  - TLS 오프로딩은 ALB에서 처리하는 걸 권장
- 단일 ALB만 등록 가능
- NLB와 ALB는 같은 VPC와 같은 AWS 계정 안에 있도록 하는 것을 권장
  - 그렇지 않으면 불필요한 Latency 발생
- NLB의 교차 영역(Cross-zone) 로드 밸런싱을 끄는 것을 권장
  - ALB가 이미 자체적으로 처리하기 때문

Cross-Zone Load Balancing

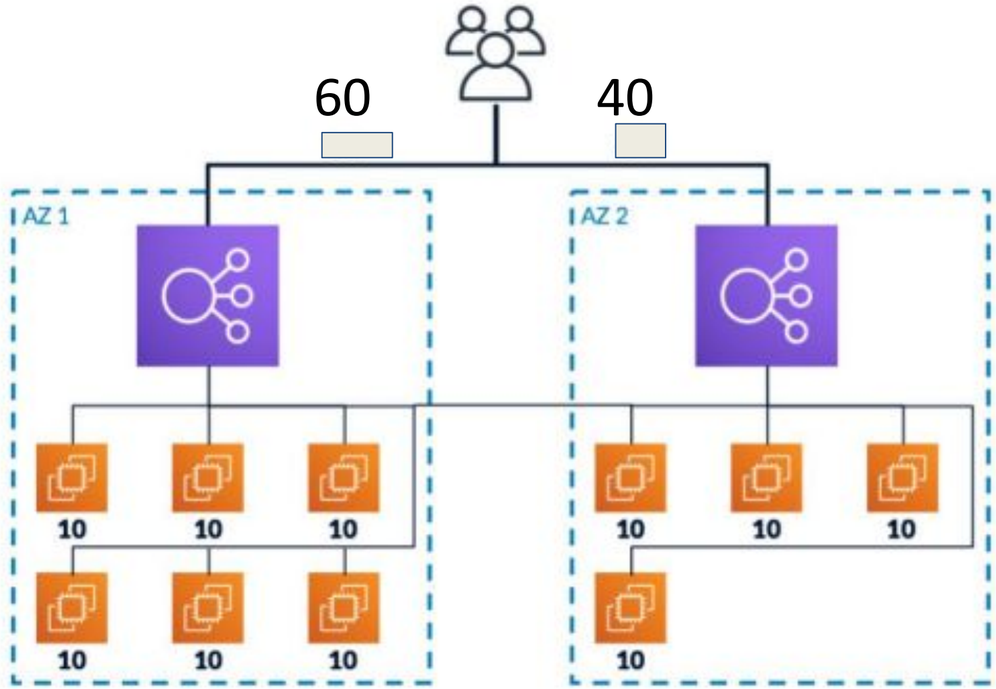
NLB



비활성화

가용영역 내에 있는 타겟에만 트래픽을 분배

NLB default



활성화

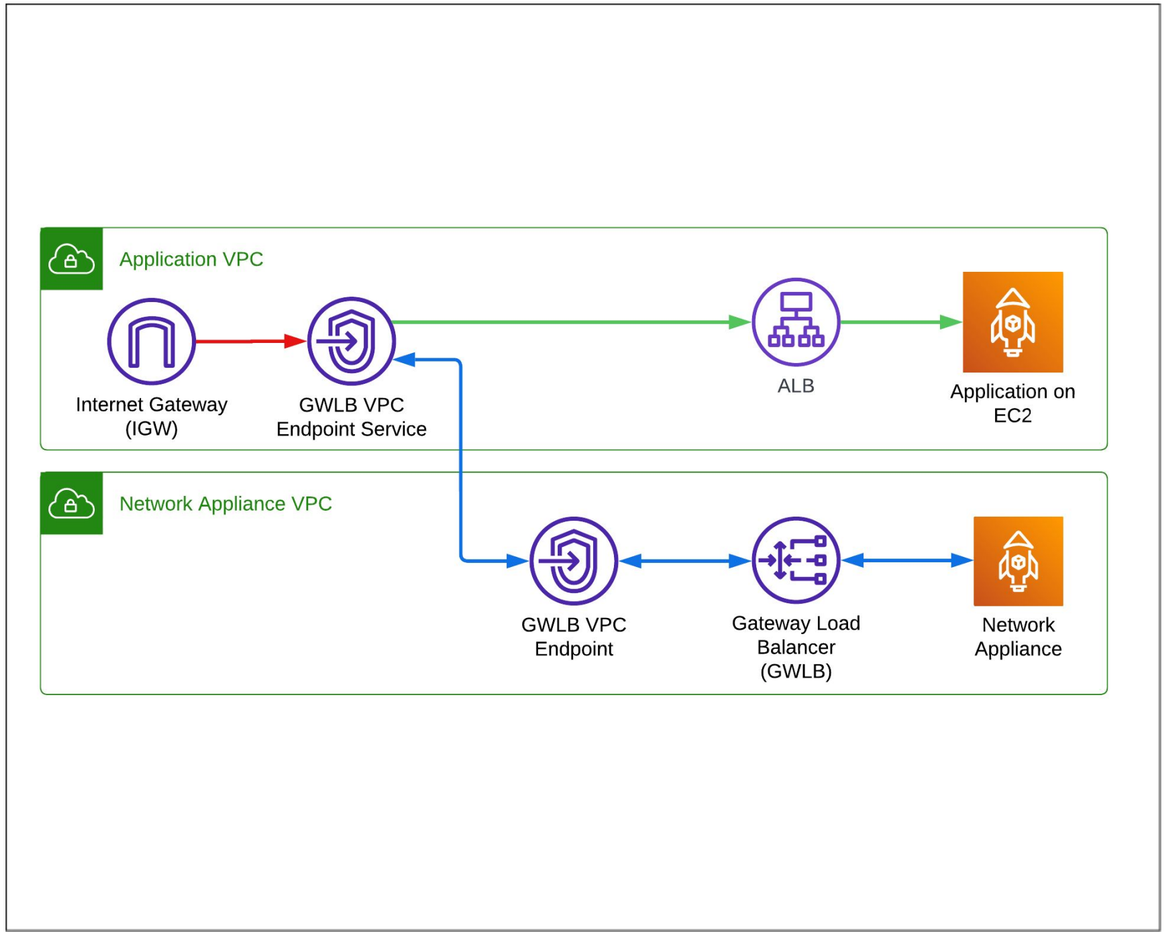
모든 가용영역의 등록된 모든 타겟 인스턴스에
동일하게 트래픽을 분배

ALB default

GWLB

GWLB

GWLB



왜 필요한가?

- 전통적인 전문 보안 장비(방화벽, IDS/IPS 등)를 클라우드 환경으로 이관하기 위해
- SPOF, 확장성의 한계를 극복
- 라우팅 테이블 조회

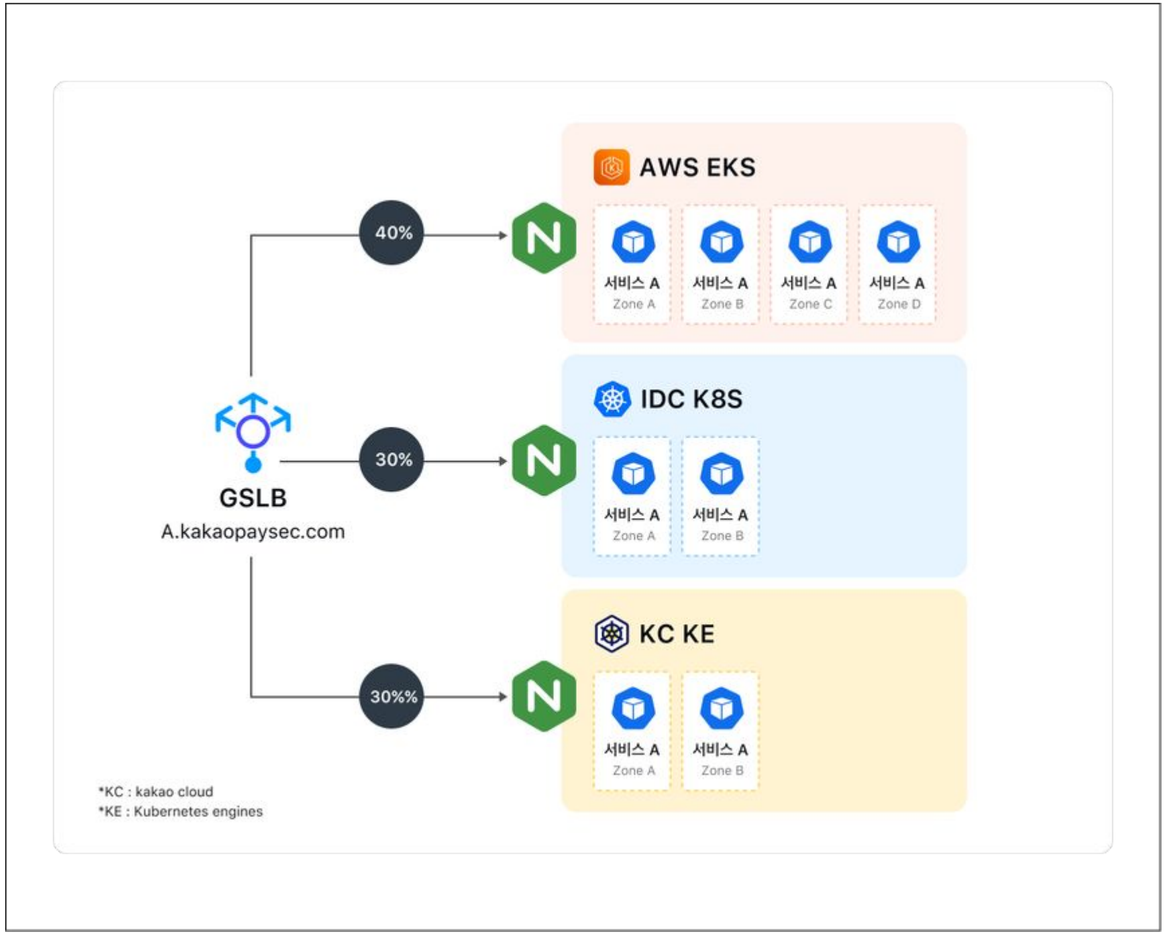
GENEVE 프로토콜

- GENEVE 프로토콜을 통해 캡슐화된 페이로드가 네트워크 어플라이언스로 전송
- UDP 6081 포트로 송수신

DNS LB

DNS LB

DNS LB



출처 : https://tech.kakaopay.com/post/multi-cluster/

DNS Load Balancing

- DNS를 사용해서, 웹사이트에 접속하려는 사용자들을 여러 대의 서버에 골고루 나눠 보내주는 기술
- 권한 있는 네임서버가 수행
- 주로 라운드 로빈 방식을 채택
  - 지리적 위치 등 다양한 특성을 기준으로 한 동적 부하 분산 알고리즘을 채택할 수 있음!!
- health check 기능이 없음
  - GLSB(Golbal Server LB)를 사용할 수 있음
  - 등록된 호스트에게 주기적인 health check
  - GSLB는 일반적으로 IP 기반으로만 엔드포인트를 등록할 수 있으며, CNAME의 경우 등록 자체가 안되고 등록해도 트래픽 제어나 헬스 체크가 제대로 작동하지 않기 때문에 대부분의 실무에서는 사용하지 않는다

QnA