AWS BASIC · 01

AWS 네트워크 기초

클라우드를 왜 쓰는지, 그 안의 네트워크는 어떻게 생겼는지

왜 클라우드를 쓰나
온프레미스와 무엇이 다른지

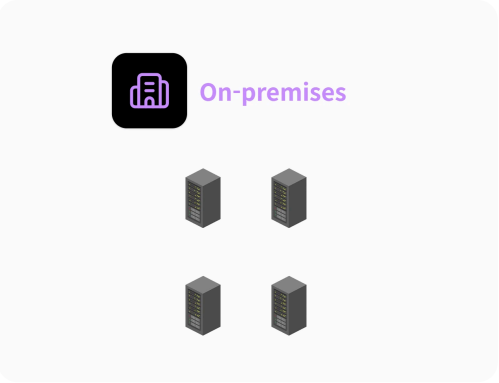
AWS 인프라
데이터 센터, 가용영역과 리전

어떻게 그리나
VPC와 서브넷을 CIDR로 잘라 만든다.

01 On-premises

직접 서버를 사는 방식

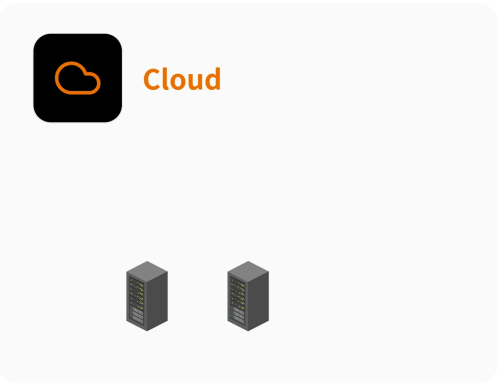
- 쓰든 안 쓰든, 사 둔 만큼의 값을 먼저 치른다.
- 모자라면 주문하고 배송을 기다린다.



02 Cloud

서버를 빌려 쓰는 방식

- 쓴 만큼만 낸다.
- 한 대로 시작해서 필요할 때 늘렸다 줄인다.



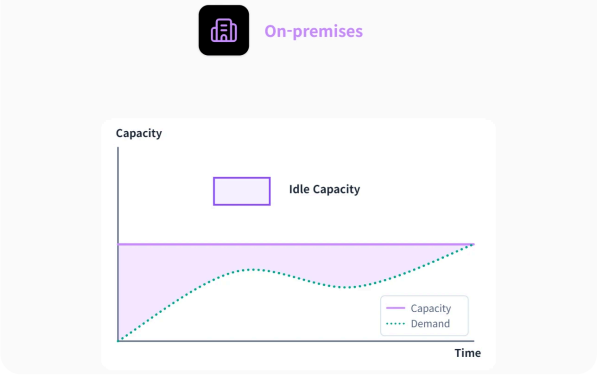
무엇이 다른가

—	온프레미스	클라우드
비용	먼저 크게 치른다 (자본 지출)	쓴 만큼 낸다 (운영 지출)
용량	살 때 정해지고 못 바꾼다	필요할 때 바꾼다
준비 시간	주문 · 배송 · 설치	몇 분
관리	전원 · 냉각 · 교체까지 직접	건물과 하드웨어는 AWS 몫
해외 진출	그 나라에 장비를 들여야 한다	리전을 고르면 된다

03 놀고 있는 서버

Idle Capacity

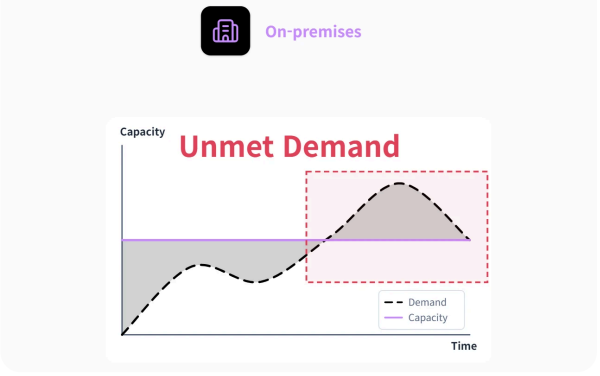
- 수요가 적은 날에도 사 둔 서버의 처리량은 그대로다.
- 사 둔 처리량은 줄일 수 없다.
- 남는 만큼이 그대로 비용이다.



04 수요가 몰리는 날

Unmet Demand

- 사 둔 처리량을 넘어서는 순간
- 넘어선 만큼의 요청은 받지 못한다.
- 서버를 더 주문해도 배송과 설치가 끝날 때까지 기다린다.



먼저 치르는 것

서버 한 대를 켜기 전에

장비값	서버 · 스토리지 · 네트워크 장비를 미리 산다	**공간과 설비**	랙을 둘 자리, 전원, 냉각, 회선이 따로 든다
사람	설치와 교체, 새벽 장애 대응까지 내 일이 된다	**시간**	용량이 모자란 순간부터 새 장비가 도착할 때까지 걸린다

그래도 온프레미스인 경우

규제가 서버의 위치를 정할 때
데이터를 특정 건물 안에 둬야 한다는 조건이 붙는 산업이 있다.

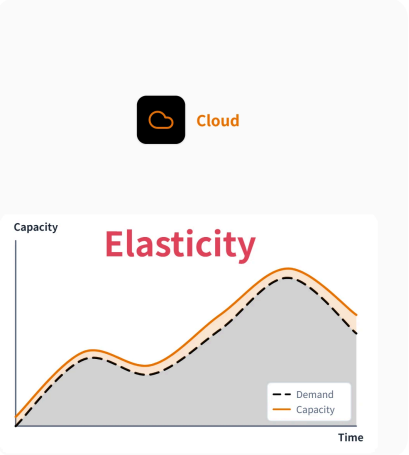
특수한 하드웨어
클라우드가 빌려주지 않는 장비가 필요한 일도 있다.

이미 사둔 장비
감가상각이 끝나지 않은 서버는 그냥 두는 편이 쌀 수 있다.

05 Elasticity

탄력성 — 수요를 따라 늘렸다가 줄이는 것

- 늘리는 것만이 아니라 **줄이는 것**까지가 탄력성이다.
- 용량을 미리 맞힐 필요가 없어진다.



클라우드의 여섯 가지 이점

시험 포인트

- [ ] 고정 비용을 변동 비용으로 대체
- [ ] 용량 추정 불필요
- [ ] 데이터 센터 운영·유지 비용 절감
- [ ] 규모의 경제에 따른 이점
- [ ] 속도와 민첩성 개선
- [ ] 몇 분 만에 전 세계로 배포

전부 옮기는 것만이 클라우드는 아니다. 배포 방식은 셋으로 나뉜다 — 전부 클라우드 위에서 운영은 **클라우드**, 기존 장비와 함께 쓰는 **하이브리드**, 내 장비로만 운영하는 **온프레미스**(프라이빗 클라우드).

클라우드라고 무조건 비용이 줄지는 않는다

켜 두면 계속 나간다
쓴 만큼 낸다는 말은 안 쓰면서 켜 두면 그만큼 낸다는 말이기도 하다.

데이터 전송 비용
들여오는 데이터는 대체로 무료지만, 밖으로 내보내는 데이터에는 요금이 붙는다.

옮기기만 하면 그대로다
온프레미스를 그대로 옮기면 놀던 서버가 그대로 놀 뿐이다. 적합한 아키텍처와 운영이 필요하다.

어디까지 맡기나

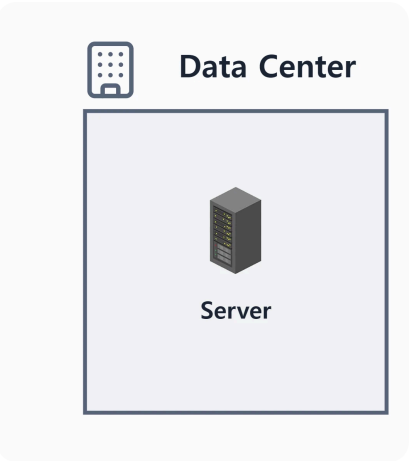
시험 포인트

—	AWS가 하는 것	내가 해야 것
이름	클라우드 자체 보안	클라우드 **안**의 보안
지키는 대상	건물 · 하드웨어 · 리전과 가용영역 · 가상화 계층	게스트 OS와 패치 · 애플리케이션 · 데이터
네트워크	물리 회선과 그 아래 전부	보안그룹 · 라우팅 · 무엇을 열어 둘지

06 Data Center

데이터 센터 — 서버 수백 대가 들어찬 건물 하나

- 클라우드도 결국 어딘가에 꽂힌 물리 서버다.
- 전원도 냉각도 회선도 건물 단위로 따로 들어온다.



07 Availability Zone

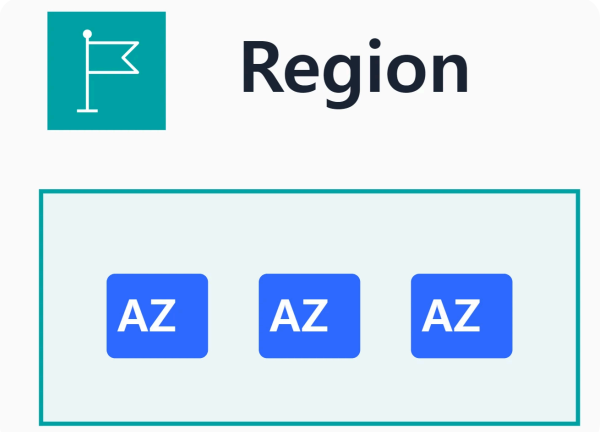
가용영역 — 가까운 데이터 센터를 묶은 것

- 재난 상황을 고려한 설계
- 한 채가 정전이어도 옆 건물은 멀쩡할 만큼 떨어뜨려 둔다 (서로 **100km 안쪽**).
- 대신 지연이 느껴지지 않을 만큼은 가깝게 묶는다.

08 Region

리전 — 가용영역을 묶은 것

- 한 리전은 **가용영역 세 곳 이상**으로 이루어진다.
- 전 세계 39개 리전 · 124개 가용영역 (2026년).
- 계속해서 늘어나고 있다



쌓이는 순서

서버 물리 한 대	**데이터 센터** 건물 한 채	**가용영역** AZ · 100km 안	**리전** AZ 3곳 이상

여기까지가 AWS가 지어 둔 것이다. 내가 만드는 것은 리전을 고른 다음부터 시작한다.

리전 코드 읽는 법

대륙 · 방향 · 몇 번째

ap-northeast-1	도쿄
ap-northeast-2	서울
ap-northeast-3	오사카
us-east-1	버지니아 북부

앞은 대륙(ap · us · eu), 가운데는 방향, 끝은 그 지역에서 몇 번째로 열렸는지다. 서울이 2번인 것은 도쿄가 먼저 열렸기 때문이다.

리전을 고르는 네 가지

시험 포인트

사용자와 가까운 곳
거리가 곧 지연이다. 서비스 대상이 한국이면 서울.

요금
같은 인스턴스도 리전마다 값이 다르다.

쓰려는 서비스가 있는지
새 서비스는 리전마다 열리는 시점이 다르다.

규제와 데이터 위치
데이터를 국내에 둬야 하는 경우가 있다.

다양한 형태의 AWS 인프라

리전 · 가용영역 말고도

로컬 존
리전을 사용자 가까이로 늘린 것. 46곳

Wavelength 존
통신사 5G 망 안에 둔 것. 33곳

Outposts
AWS 장비를 내 건물에 들여놓는 것

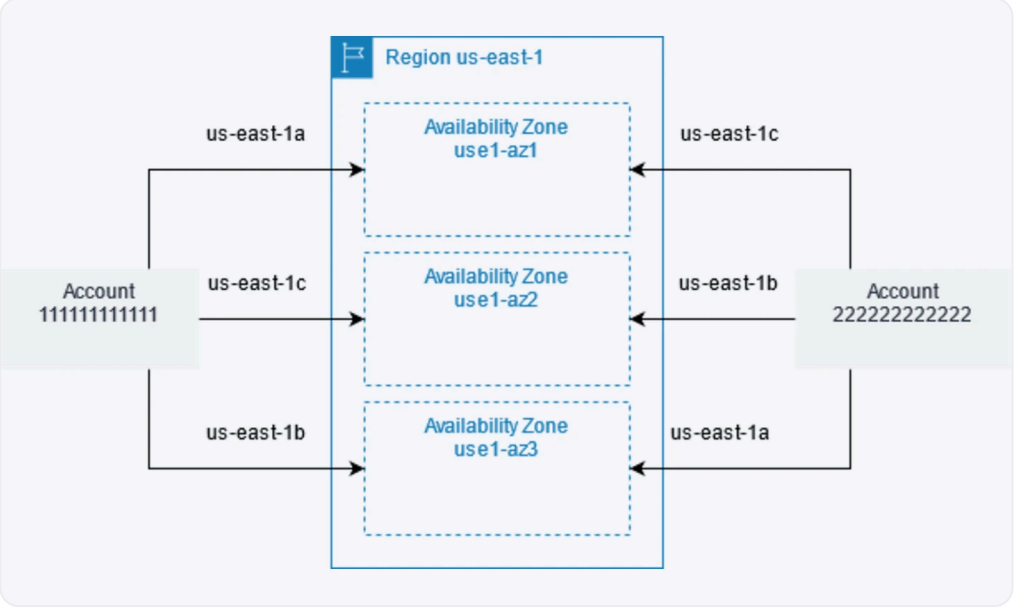
엣지 로케이션
CloudFront가 캐시하는 곳. 750곳 이상

리전과 가용영역의 함정

리전끼리는 격리돼 있다
생성한 리소스의 리전을 확인하자.

a · b · c는 계정마다 다르다
`us-east-1a`가 가리키는 물리 영역은 계정마다 무작위로 섞인다. 두 계정을 같은 곳에 맞추려면 이름이 아니라 **AZ ID**(`use1-az1`)를 본다.

가용영역을 넘는 통신은 돈이 든다
같은 리전 안이어도 오가는 양쪽에 GB당 $0.01이 붙는다 (같은 AZ 안은 무료).



계정이 다르면 가리키는 곳이 다르다. 계정을 가로질러 같은 곳을 가리키는 것은 **AZ ID** 쪽이다.

AWS 「가용 영역 ID」 문서 ↗

가용영역을 쓰는 법

시험 포인트

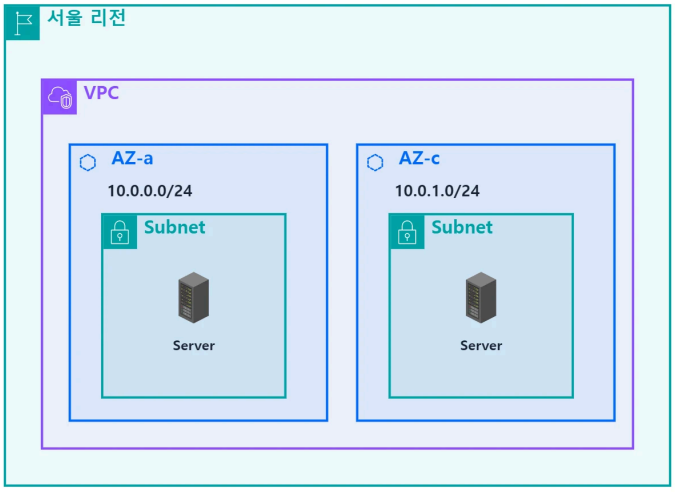
리소스는 두 곳 이상에 나눠 둔다
한 곳이 통째로 멈춰도 서비스는 살아 있다 — 가용영역을 써야 하는 이유.

서브넷을 AZ마다 하나씩
이중화는 서브넷을 나누는 데서 시작한다.

계정이 다른 경우
콘솔은 a·b·c를 보여준다. 다른 계정과 같은 곳을 맞출 때만 AZ ID를 본다.

09 내려오는 순서

리전을 고르고, VPC를 만들고, 서브넷으로 쪼갠다



- VPC는 **리전 하나**를 골라 만든다. 그 리전을 벗어나지 못한다.
- 대신 리전 안의 가용영역 **전체**에 걸쳐 있다.
- 서브넷은 **가용영역 하나**에 속한다.
- 크기는 CIDR을 이용해 할당한다.

무엇이 무엇에 속하나

리전	가용영역 3곳 이상	**가용영역**	데이터 센터 한 곳 이상
VPC	리전 하나 · 그 안의 모든 가용영역에 걸친다	**서브넷**	가용영역 하나 · VPC 하나

서비스는 어디에 속하나

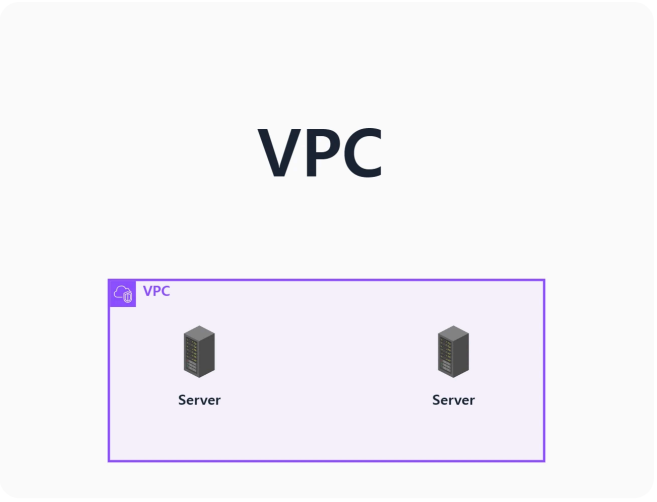
시험 포인트

범위	무엇이 있나	무슨 뜻인가
글로벌	IAM · Route 53 · CloudFront	리전을 고르지 않는다. 어느 리전에서 봐도 같다
리전	S3 · VPC · DynamoDB · Lambda	리전 안의 가용영역 전체에 걸친다
가용영역	EC2 인스턴스 · EBS 볼륨 · 서브넷	한 곳에만 있다. 그 곳이 멈추면 같이 멈춘다

10 VPC

Virtual Private Cloud

- 내 계정 안에 논리적으로 격리된 가상 네트워크.
- 문을 열기 전까지 밖의 트래픽은 안으로 들어오지 못한다.



VPC의 규칙

리전 하나
만들 때 고른 리전을 벗어나지 못한다 (리전 안의 AZ 전체에는 걸친다).

주소는 사설 대역에서 고른다
RFC 1918 — `10.0.0.0/8` · `172.16.0.0/12` · `192.168.0.0/16`. AWS가 권하는 범위다.

기본 VPC가 이미 있다
계정에는 리전마다 기본 VPC가 하나씩 있어 바로 인스턴스를 띄울 수 있다.

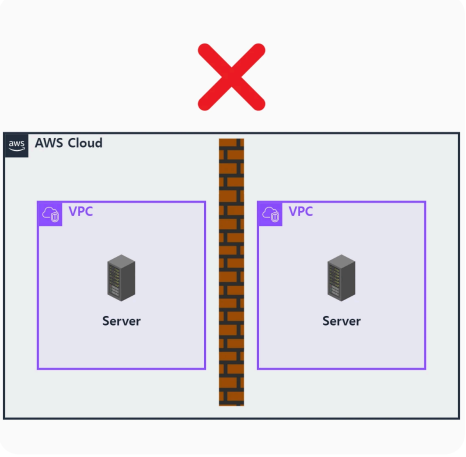
리전당 5개
기본 할당량. 늘려 달라고 요청하면 수백 개까지 늘릴 수 있다.

VPC 자체는 무료
비용이 붙는 것은 NAT 게이트웨이 같은 서비스와 공인 IPv4 주소다.

11 VPC 간 통신

기본은 안 된다

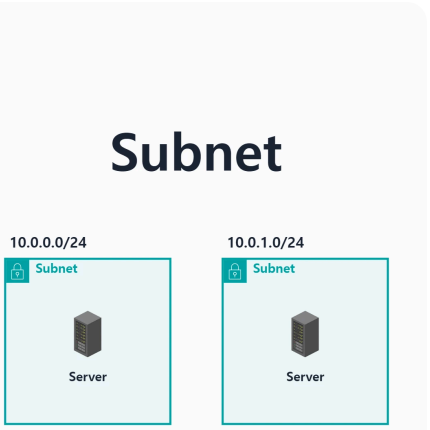
- 서로 완전히 격리돼 있어서 옆 VPC의 서버에는 닿지 못한다.
- 연결 하려면 피어링이나 Transit Gateway로 따로 연결한다.



12 Subnet

VPC의 IP 대역을 잘라 놓은 구역

- `10.0.0.0/16` VPC 안을 `10.0.0.0/24`처럼 잘라 둔다.
- 서버는 반드시 어느 서브넷 안에 자리를 잡는다.



서브넷의 규칙

시험 포인트

가용영역 하나
서브넷 하나는 AZ 하나에만 속한다. 두 곳에 걸칠 수 없다.

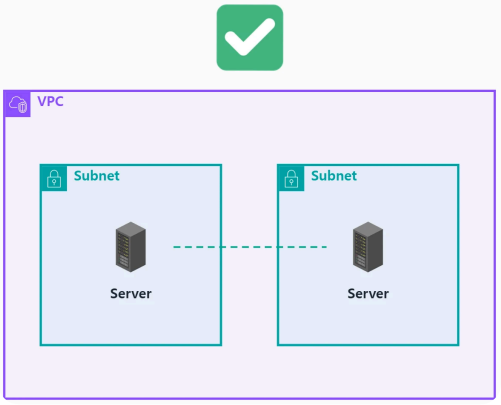
이중화는 여기서 시작한다
AZ마다 서브넷을 하나씩 두는 것이 첫 걸음이다.

VPC 하나에 200개
기본 할당량. 요청하면 늘릴 수 있다.

13 서브넷 간 통신

같은 VPC 안이라면 된다

- VPC를 만들 때 대역 전체를 향한 `local` 경로가 라우팅 테이블에 깔린다.
- 막고 싶다면 보안그룹이나 네트워크 ACL로 따로 막는다.



퍼블릭 서브넷과 프라이빗 서브넷

시험 포인트

퍼블릭 서브넷	라우팅 테이블에 **인터넷 게이트웨이**로 가는 길이 있다
프라이빗 서브넷	인터넷 게이트로 가는 길이 없다. 밖으로 나가려면 NAT를 거친다
무엇이 가르나	만들 때 고르는 옵션이 아니라, 붙여 둔 라우팅 테이블에 인터넷 게이트웨이가 있는지 없는지!

14 CIDR

Classless Inter-Domain Routing

- 클래스 없이, 필요한 만큼만 잘라 쓰는 표기법 (1993년).
- 클래스 표기법은 필요 이상으로 큰 덩어리를 사용하는 불편함이 존재.

CIDR

10.0.0.0 / 16

192.168.0.0 / 24

15 Octet

옥텟 — 8비트, 그러니까 256개

- 8비트로 표현할 수 있는 숫자는 2^8 = 256개.
- IPv4 주소는 그 옥텟 **네 개**다
- 점을 기준으로 네 개의 숫자 덩어리, 네 개의 옥텟으로 구성
- `192.168.0.0`

Octet

$= 2^8$

$= 256$

16 슬래시 뒤 숫자

프리픽스 — 앞에서 몇 비트를 고정할지

- 고정된 앞이 **네트워크**, 남은 뒤가 그 안에서 쓰는 **호스트**.
- `/24`면 앞 24비트가 고정이고, 남은 8비트로 `192.168.0.0` ~ `192.168.0.255`를 쓴다.

192.168.0.0 / 24

11000000.10101000.00000000.00000000

11000000.10101000.00000000.11111111

사설 IP 대역

시험 포인트

대역	범위	예시
10.0.0.0/8	10.0.0.0 ~ 10.255.255.255	`10.0.0.0/16`
172.16.0.0/12	172.16.0.0 ~ 172.31.255.255	`172.31.0.0/16`
192.168.0.0/16	192.168.0.0 ~ 192.168.255.255	`192.168.0.0/20`

VPC도 서브넷도 /16 ~ /28

시험 포인트

프리픽스	주소 수	쓸 수 있는 IP
/16	65,536	65,531
/20	4,096	4,091
/24	256	251
/28	16	11

서브넷에서 쓸수 없는 주소

앞의 넷과 맨 뒤 하나

```
10.0.0.0      네트워크 주소
10.0.0.1      VPC 라우터
10.0.0.2      DNS
10.0.0.3      예약 (미래용)
10.0.0.255    브로드캐스트
```

CIDR 정하기 전에

VPC 범위 안에 있어야 한다
`10.0.0.0/16` VPC 안에 `192.168.0.0/24` 서브넷은 만들 수 없다.

서브넷끼리 겹치면 안 된다
한 주소는 한 서브넷의 것. 겹치는 CIDR은 거절당한다.

처음부터 넉넉하게 잡는다
주소를 남겨 둔다고 요금이 붙지는 않는다. 서브넷은 /24가 무난하다.

CIDR의 함정

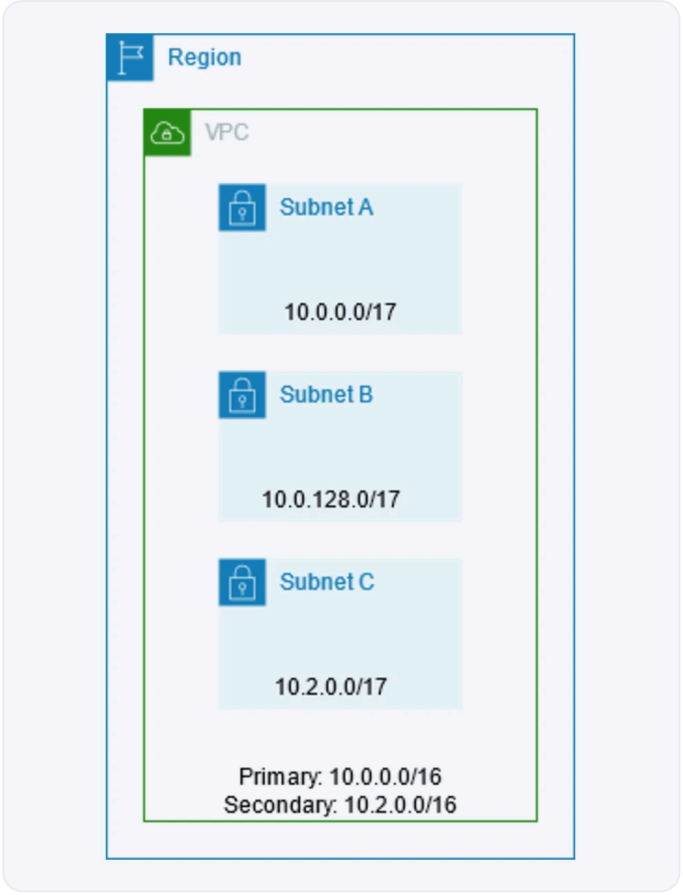
한 번 정한 CIDR은 크기를 못 바꾼다
넓히려면 두 번째 CIDR을 덧붙이고 (VPC당 5개), 줄이려면 VPC를 새로 만드는 수밖에 없다.

/28보다 잘게는 못 쪼갠다
16개 중 11개만 쓸 수 있어, 실무에서 쓰기엔 이미 빠듯하다.

고르면 안 되는 대역

아예 거절당하는 것
`0.0.0.0/8` · `127.0.0.0/8` · `169.254.0.0/16` · `224.0.0.0/4`는 VPC에 쓸 수 없다.

172.17.0.0/16은 피한다
Cloud9 · SageMaker 같은 서비스가 이 대역을 쓴다. 겹치면 그 서비스에 못 붙는다.



기본 CIDR 하나에 **두 번째 CIDR**을 덧붙인 VPC. 서브넷 C만 새 대역에서 잘라냈다. 크기를 바꾼 것이 아니라 더한 것이다.

AWS 「VPC CIDR 블록」 문서 ↗

IPv6는 무엇이 다른가

주소를 고를 수 없다

VPC	Amazon이 주는 /56 하나. 어느 대역일지는 못 고른다	**서브넷**	/64 — IPv4처럼 잘게 나누지 않는다
예약 주소	IPv4와 똑같이 다섯 개다 — 앞의 넷과 맨 뒤 하나	**사설 대역**	IPv6에는 없다. 나가고 말고는 라우팅이 정한다

풀어 보자!

실전 문제

1 데이터를 국내에 둬야 한다는 규정이 있다. AWS에서 이 조건을 지키는 첫 단계는?

- A 가용영역
- B 리전
- C 서브넷
- D VPC

2 서울 리전에 `10.0.0.0/16` VPC를 만들고, 가용영역 두 곳에 서브넷을 하나씩 두려고 한다. 만들 수 없는 서브넷은?

- A `10.0.0.0/24` — ap-northeast-2a
- B `10.0.1.0/24` — ap-northeast-2c
- C `192.168.0.0/24` — ap-northeast-2a
- D `10.0.2.0/28` — ap-northeast-2c

3 리전과 가용영역에 대해 맞는 것 2개는? 정답 2개

- A 서브넷 하나는 여러 가용영역에 걸칠 수 있다
- B VPC는 리전 하나 안에서만 존재한다
- C `us-east-1a`는 모든 계정에서 같은 물리 영역을 가리킨다
- D 리전끼리는 격리돼 있어 리소스가 자동으로 복제되지 않는다
- E 가용영역은 데이터 센터 한 채와 같은 말이다

4 `10.0.0.0/24` 서브넷 하나에 둘 수 있는 EC2 인스턴스는 최대 몇 대인가?

- A 256대
- B 255대
- C 254대
- D 251대

정답과 해설

1 **B** 데이터가 어느 나라에 있느냐를 정하는 것은 리전이다. 가용영역은 고른 리전 *안*의 위치라 나라를 바꾸지 못하고, VPC와 서브넷은 리전을 고른 뒤에 그린다.

2 **C** 서브넷 CIDR은 VPC CIDR 안에 들어야 한다. `192.168.0.0/24`는 `10.0.0.0/16` 밖이라 거절당한다. 나머지 셋은 범위 안이고 서로 겹치지 않으며, `/28`은 만들 수 있는 가장 작은 크기다.

3 **B · D** 서브넷은 가용영역 하나에만 속하고, a·b·c라는 이름은 계정마다 다른 물리 영역에 무작위로 붙는다 (같은 곳을 가리키는 것은 AZ ID). 가용영역은 데이터 센터 한 곳 **이상**을 묶은 단위다.

4 **D** `/24`는 주소 256개지만, 서브넷마다 앞의 네 개(네트워크 · VPC 라우터 · DNS · 예약)와 맨 뒤 브로드캐스트까지 다섯 개를 AWS가 가져간다. 흔히 떠올리는 254는 앞의 넷 중 셋을 빼먹은 값이다.

외워 둘 숫자

시험 포인트

3
리전당 최소 AZ
보통은 그보다 많다

100km
AZ 사이의 거리
이보다 멀지 않게 묶는다

/16
가장 큰 CIDR
주소 65,536개

/28
가장 작은 CIDR
주소 16개

5
서브넷의 예약 주소
앞의 넷과 맨 뒤 하나

251
/24로 쓸 수 있는 IP
256 − 5

5
리전당 VPC 수
기본 할당량, 늘릴 수 있다

200
VPC당 서브넷 수
기본 할당량, 늘릴 수 있다

용어 정리

이 시트에 나온 말, 한 줄씩

리전 (Region) 가용영역 세 곳 이상을 묶은 지리적 단위

가용영역 (AZ) 데이터 센터 한 곳 이상. 서로 100km 안

AZ ID 계정이 달라도 같은 물리 영역을 가리키는 이름 (`use1-az1`)

VPC 리전 안에 만드는, 밖과 격리된 가상 네트워크

서브넷 (Subnet) VPC를 쪼갠 구역. 가용영역 하나에만 속한다

퍼블릭 서브넷 라우팅 테이블에 인터넷 게이트웨이로 가는 길이 있는 서브넷

CIDR 슬래시 뒤 숫자로 크기를 적는 주소 표기법

프리픽스 슬래시 뒤 숫자. 앞에서 고정한 비트 수를 말한다

옥텟 (Octet) 8비트. IPv4 주소는 옥텟 네 개

사설 IP RFC 1918의 세 대역. 인터넷에 그대로 나가지 않는다

탄력성 (Elasticity) 수요를 따라 늘렸다가 다시 줄이는 것

온프레미스 서버를 사서 내 자리에 두고 쓰는 방식

다음에 볼 것

VPC 안에 길을 내는 이야기

라우팅 테이블
어디로 보낼지 정하는 표. 퍼블릭과 프라이빗을 가르는 것도 이것

인터넷 게이트웨이
VPC가 인터넷과 만나는 문

NAT 게이트웨이
프라이빗 서브넷이 나가기만 할 때 거치는 곳

보안그룹 · NACL
인스턴스 앞의 방화벽과 서브넷 앞의 방화벽

한 줄 정리

데이터 센터 → 가용영역 → 리전까지가 AWS가 지어 둔 물리적인 단위이고, 그 위에 VPC → 서브넷을 CIDR로 잘라 내 네트워크를 그린다.

Data Center · AZ · Region · VPC · Subnet · CIDR

공식 문서

AWS 설명서 · 표준 문서 · 영상 · 글

AWS 글로벌 인프라 — 리전과 가용영역 4 · 6쪽
리전 39 · 가용영역 124, AZ 세 곳 이상, 100km 기준이 나온 곳
aws.amazon.com/about-aws/global-infrastructure/regions_az

클라우드 컴퓨팅의 여섯 가지 이점 3쪽
AWS가 붙인 여섯 이름 그대로
docs.aws.amazon.com/whitepapers/latest/aws-overview/six-advantages-of-cloud-computing.html

공동 책임 모델 3쪽
클라우드의 보안과 클라우드 안의 보안을 가르는 선
aws.amazon.com/compliance/shared-responsibility-model

Regions and Zones (EC2 사용 설명서) 6 · 7쪽
리전 격리와 AZ 코드, 로컬 존과 Outposts까지
docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html

가용 영역 ID 6쪽
a · b · c가 계정마다 섞이는 이유와 use1-az1
docs.aws.amazon.com/ram/latest/userguide/working-with-az-ids.html

What is Amazon VPC? 8 · 9쪽
VPC · 서브넷 · 라우팅의 정의, 그리고 VPC 자체는 무료라는 문장
docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.html

기본 VPC 8쪽
계정에 이미 있는 VPC가 무엇을 갖추고 있나
docs.aws.amazon.com/vpc/latest/userguide/default-vpc.html

VPC CIDR 블록 11 · 12쪽
/16 ~ /28, RFC 1918 권장, 크기는 못 바꾸되 두 번째 CIDR은 붙일 수 있다
docs.aws.amazon.com/vpc/latest/userguide/vpc-cidr-blocks.html

서브넷 CIDR 블록 11쪽
서브넷마다 예약되는 다섯 주소를 하나씩
docs.aws.amazon.com/vpc/latest/userguide/subnet-sizing.html

Amazon VPC 할당량 8 · 9쪽
리전당 VPC 5개, VPC당 서브넷 200개, CIDR 5개
docs.aws.amazon.com/vpc/latest/userguide/amazon-vpc-limits.html

RFC 1918 — 사설 IP 대역 12쪽
10 · 172.16 · 192.168이 어디서 왔나. AWS 문서가 아니라 표준 문서다
datatracker.ietf.org/doc/html/rfc1918

Overview of Data Transfer Costs for Common Architectures 6쪽
가용영역과 리전을 넘을 때 무엇에 값이 붙는지 그림으로
aws.amazon.com/blogs/architecture/overview-of-data-transfer-costs-for-common-architectures

AWS를 사용한다면 반드시 알아야 할 네트워크 기초 지식 4 · 8쪽
이 시트와 같은 범위를 한국어로. AWS Korea 공식 채널
www.youtube.com/watch?v=vCNexbgYmQ8

AWS Networking Fundamentals (re:Invent 2025) 14쪽
이 시트 다음 단계를 한 시간에 훑는 공식 세션 (영어)
www.youtube.com/watch?v=nXBqxnp5ybY