

## Lab. Using Amazon EBS on Amazon Linux 2

### 목적

이번 실습에서는 Volume을 생성하고 생성한 Volume을 기존에 생성돼 있는 EC 인스턴스에 연결하고 사용 가능하게 하는 방법을 다룬다.

### 사전 준비물

AWS Free-Tier 계정

# 실습을 위한 네트워크 자원 설정 및 EC2 인스턴스 생성

1. 실습을 위해 다음 그림과 같은 네트워크 환경을 구축했다. 주의할 점은 한 개의 가용영역만 생성했다.

- A. [VPC] : lab-vpc
- B. [가용 영역] : ap-northeast-2a
- C. [퍼블릭 서브넷 수] : 1, 10.0.10.0/24
- D. [프라이빗 서브넷 수] : 1, 10.0.20.0/24

### VPC 설정

**생성할 리소스 정보**  
VPC 리소스 또는 VPC 및 기타 네트워킹 리소스만 생성합니다.

☐ VPC만

☒ VPC 등

**이름 태그 자동 생성 정보**  
이름 태그의 값을 입력합니다. 이 값은 VPC의 모든 리소스에 대한 이름 태그를 자동으로 생성하는 데 사용됩니다.

☒ 자동 생성

lab

**IPv4 CIDR 블록 정보**  
CIDR 표기법을 사용하여 VPC의 시작 IP와 크기를 결정합니다.

10.0.0.0/16

65,536 IPs

CIDR 블록 크기는 /16에서 /28 사이여야 합니다.

**가용 영역(AZ) 수 정보**  
서브넷을 프로비저닝할 AZ 수를 선택합니다.고가용성을 위해서는 최소 2개 이상의 AZ를 사용하는 것이 좋습니다.

123

▼ AZ 사용자 지정

첫 번째 가용 영역

ap-northeast-2a

**퍼블릭 서브넷 수 정보**  
VPC에 추가할 퍼블릭 서브넷 수입니다. 인터넷을 통해 공개적으로 액세스할 수 있어야 하는 웹 애플리케이션에는 퍼블릭 서브넷을 사용합니다.

01

**프라이빗 서브넷 수 정보**  
VPC에 추가할 프라이빗 서브넷 수입니다. 프라이빗 서브넷을 사용하여 퍼블릭 액세스가 필요 없는 백엔드 리소스를 보호합니다.

012

▼ 서브넷 CIDR 블록 사용자 지정

ap-northeast-2a 퍼블릭 서브넷 CIDR 블록

10.0.10.0/24

256 IPs

ap-northeast-2a 프라이빗 서브넷 CIDR 블록

10.0.20.0/24

256 IPs

<>^v

**NAT 게이트웨이(\$\$) 정보**  
NAT 게이트웨이를 생성할 가용 영역(AZ) 수를 선택합니다. 각 NAT 게이트웨이마다 요금이 부과됩니다.

없음1개의 AZ에서AZ당 1개

**VPC 엔드포인트 정보**  
엔드포인트는 VPC에서 S3에 직접 액세스하여 NAT 게이트웨이 요금을 줄이고 보안을 강화할 수 있습니다. 기본적으로 모든 액세스 정책이 사용됩니다. 언제든지 이 정책을 사용자 지정할 수 있습니다.

없음S3 게이트웨이

**DNS 옵션 정보**

☒ DNS 호스트 이름 활성화

☒ DNS 확인 활성화

2. 이제 EC2 인스턴스를 생성한다. 설정정보는 다음과 같다.

A. [이름] : lab-amazon-ec2

B. [AMI 선택] : Amazon Linux 2 AMI(HVM)- Kernel 5.10, SSD Volume Type, 64비트(x86)

C. [인스턴스 유형] : t2.micro

D. [키 페어] > [새 키 페어 생성] : lab-amazon-ec2-key.pem

E. [네트워크] : lab-vpc, lab-subnet-public1-ap-northeast-2a, 퍼블릭 IP 자동 할당 활성화

F. [보안 그룹] > [새 보안 그룹 생성] : lab-sg, SSH, 22, 위치 무관, 0.0.0.0/0

G. [스토리지] : 마그네틱(standard), 8GiB

▼ 요약

인스턴스 개수 [정보](#)

[소프트웨어 이미지\(AMI\)](#)

Amazon Linux 2 Kernel 5.10 AMI...[더 보기](#)  
ami-03a633fd8200146fb

[가상 서버 유형\(인스턴스 유형\)](#)

t2.micro

[방화벽\(보안 그룹\)](#)

새 보안 그룹

[스토리지\(볼륨\)](#)

1개의 볼륨 – 8GiB

취소

인스턴스 시작

[명령 검토](#)

## EBS Volume 생성하기

- 좌측 메뉴 중 [Elastic Block Store] > [볼륨] 페이지로 이동하여 페이지 우측 상단의 [볼륨 생성]을 클릭한다.
- 첫 번째 볼륨 유형(Volume type)에서 생성할 볼륨 유형을 선택한다. [크기(Size)]에 볼륨 크기를 GiB 단위로 입력한다. 만일 볼륨 유형을 gp3로 선택했다면, 처리량(Throughput)에 볼륨에서 제공해야 하는 처리량(MiB/s)이 입력가능하다. 그리고 가용 영역에서 볼륨을 생성할 가용 영역을 선택한다. 볼륨은 동일한 가용 영역의 인스턴스에만 연결할 수 있다.
  - [볼륨 유형] : 범용 SSD(gp2)
  - [크기(GiB)] : 50
  - [가용 영역] : ap-northeast-2a
  - [스냅샷 ID] : 스택샷에서 볼륨을 생성하지 않음.
  - [태그] > [키] : Name, [값] : lab-volume

**볼륨 설정**

볼륨 유형 | 정보

범용 SSD(gp2) ▼

크기(GiB) | 정보

50

최소: 1 GiB, 최대: 16384 GiB. 같은 정수여야 합니다.

IOPS | 정보

150 / 3000

GiB당 3 IOPS 기준, 최소 100 IOPS, 3000 IOPS로 버스트 가능

처리량(MiB/s) | 정보

해당 사항 없음

가용 영역 | 정보

ap-northeast-2a ▼

스냅샷 ID - 선택 사항 | 정보

스냅샷에서 볼륨을 생성하지 않음 ▼

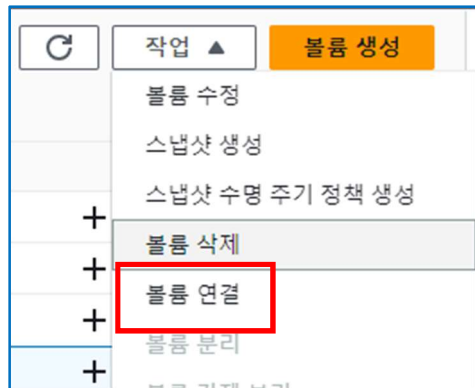
↻

- 50GiB의 볼륨이 잘 생성되었다. 방금 생성한 볼륨의 [볼륨 상태]는 사용 가능이어야 한다.

볼륨 (4) 정보												
Q 검색												
<input type="checkbox"/>	Name ▼	볼륨 ID ▼	유형 ▼	크기 ▼	IOPS ▼	스...	생성 ... ▼	가용 영역 ▼	볼륨 상태 ▼			
<input type="checkbox"/>	-	<a href="#">vol-0a562e2dab57b5d73</a>	gp2	2 GiB	100	-	2023/10/...	ap-northeast-2a	🟢 사용 중			
<input type="checkbox"/>	-	<a href="#">vol-05b350b7e8f5a536a</a>	standard	8 GiB	-	-	2023/10/...	ap-northeast-2a	🟢 사용 중			
<input type="checkbox"/>	-	<a href="#">vol-0650c744c24fa5da7</a>	standard	8 GiB	-	-	2023/10/...	ap-northeast-2a	🟢 사용 중			
<input type="checkbox"/>	lab-volume	<a href="#">vol-075dce7ec58752cf2</a>	gp2	50 GiB	150	-	2023/10/...	ap-northeast-2a	🟢 사용 가능			

## 인스턴스에 Amazon EBS 볼륨 연결

1. 방금 생성한 볼륨을 선택하고 [작업] > [볼륨 연결]을 클릭한다.



2. [볼륨 연결] 페이지에서 [인스턴스]는 인스턴스 ID를 입력하거나 목록에서 인스턴스를 선택한다. 여기서는 lab-amazon-ec2를 선택한다. [디바이스 이름]은 기본값을 사용한다. 중요한 것은 이 값을 기억해야 한다. 그리고 [볼륨 연결] 버튼을 클릭한다. 볼륨이 연결할 인스턴스는 볼륨과 동일한 가용 영역에 위치해야 하며 해당 볼륨은 사용 가능한 상태여야 한다.

### 볼륨 연결 정보

볼륨을 인스턴스에 연결하여 일반 물리적 하드 디스크 드라이브처럼 사용합니다.

#### 기본 세부 정보

볼륨 ID  
vol-075dce7ec58752cf2 (lab-volume)

가용 영역  
ap-northeast-2a

인스턴스 정보  
i-004a4022d7693750c

선택한 볼륨과 동일한 가용 영역에 있는 인스턴스만 표시됩니다.

디바이스 이름 정보  
/dev/sdf

Linux용 권장 디바이스 이름: 루트 볼륨의 경우 /dev/sda1, 데이터 볼륨의 경우 /dev/sd[f-p].

여기에 입력된 디바이스 이름(세부 정보에 표시됨)이 /dev/sdf~/dev/sdp여도 디바이스의 이름이 내부적으로 /dev/xvdf~/dev/xvdp로 바뀔 수 있습니다.




취소

볼륨 연결

3. 해당 볼륨의 상세 페이지에서 **[연결된 인스턴스]**를 확인할 수 있다. 연결상태가 **attaching**이 아니라 **attached**로 변경되어 있어야 한다.

볼륨 ID: vol-075dce7ec58752cf2 (lab-volume)			
세부 정보	상태 검사	모니터링	태그
볼륨 ID  vol-075dce7ec58752cf2 (lab-volume)  AWS Compute Optimizer 검색 결과 ⓘ 권장 사항을 받으려면 AWS Compute Optimizer에 옵트인하십시오.   자세히 알아보기 <a href="#">[링크]</a>  암호화 암호화되지 않음  빠른 스냅샷 복원 아니요  다중 연결 활성화됨 아니요	크기  50 GiB  볼륨 상태  사용 중  KMS 키 ID -  스냅샷 -	유형 gp2  IOPS 150  KMS 키 별칭 -  가용 영역 ap-northeast-2a  Outposts ARN -	볼륨 상태  정상  처리량 -  KMS 키 ARN -  생성 완료  Sun Oct
연결된 인스턴스 i-004a4022d7693750c (lab-amazon-ec2): /dev/sdf (attached)			

4. 볼륨과 연결된 인스턴스 요약 페이지로 이동해 보자. **[스토리지]** 탭의 **[블록 디바이스]** 섹션으로 이동하여 확인해 보면, 처음 인스턴스 생성시 attach했던 **8GiB**외에 방금 추가한 **50GiB**가 연결됨을 확인할 수 있다.

인스턴스: i-004a4022d7693750c(lab-amazon-ec2)							
세부 정보	보안	네트워킹	스토리지	상태 검사	모니터링	태그	
▼ 루트 디바이스 세부 정보 루트 디바이스 이름  /dev/xvda  루트 디바이스 유형 EBS  EBS 최적화 비활성							
▼ 블록 디바이스 <input type="text" value="블록 디바이스 필터링"/>							
볼륨 ID	디바이스 이름	볼륨 크기(GiB)	연결 상태	연결 시간	암호화됨	KMS 키 ID	종료 시 삭제
vol-0650c744c24fa5da7	/dev/xvda	8	 연결됨	2023/10/29 11:17 GMT+9	아니요	-	예
vol-075dce7ec58752cf2	/dev/sdf	50	 연결됨	2023/10/29 11:28 GMT+9	아니요	-	아니요

5. SSH를 사용하여 해당 인스턴스에 연결한다. 디바이스 이름은 블록 디바이스 매핑에 지정한 것과는 다른 디바이스 이름으로 인스턴스에 연결할 수 있다. 인스턴스 요약 페이지에서 **[연결]**을 클릭한다. **[SSH 클라이언트]** 탭에서 연결할 퍼블릭 DNS의 값 또는 퍼블릭 IP를 이용하여 인스턴스에 연결한다.
6. **lsblk** 명령을 사용하면 사용 가능한 디스크 디바이스 및 마운트 포인트(해당하는 경우)가 표시되어 사용 가능한 올바른 디바이스 이름을 결정하는 데 도움을 받을 수 있다. **lsblk** 명령의 출력에서는 전체 디바이스 경로 중 맨 앞에 **/dev/**가 생략된다.

\$ lsblk

```
[ec2-user@ip-10-0-10-54 ~]$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda        202:0    0   8G  0 disk
└─xvda1     202:1    0   8G  0 part /
xvdf        202:80   0  50G  0 disk
[ec2-user@ip-10-0-10-54 ~]$
```

7. 위의 그림에서 확인해 보면, 루트 디바이스는 **xvda1**이라는 파티션이 하나 있는 **/dev/xvda**이다. 연결된 볼륨은 파티션이 없고 아직 탑재되지 않은 **/dev/xvdf**임을 확인할 수 있다. 볼륨에 파일 시스템이 있는지 확인한다. 새 볼륨은 원시 블록 디바이스이므로 볼륨을 탑재하고 사용하기 전에 해당 볼륨에서 파일 시스템을 생성해야 한다. 스냅샷에서 생성된 볼륨에는 이미 파일 시스템이 있을 수 있다. 기존 파일 시스템 위에 새 파일 시스템을 생성하면 해당 작업으로 데이터가 덮어쓰게 된다. 볼륨에 파일 시스템이 있는지 여부를 확인하려면 다음 방법 중 하나 또는 모두를 사용한다.
8. 첫 번째 방법은 **file -s** 명령을 사용해서 파일 시스템 유형 등의 특정 디바이스 정보를 확인할 수 있다. 다음 예시 출력에서와 같이 출력에 data만 표시된다면, 디바이스에는 파일 시스템이 없는 것이다.

```
$ sudo file -s /dev/xvdf
```

```
[ec2-user@ip-10-0-10-54 ~]$ sudo file -s /dev/xvdf
/dev/xvdf: data
[ec2-user@ip-10-0-10-54 ~]$
```

9. 두 번째 방법은 **lsblk -f** 명령을 사용하여 인스턴스에 연결된 모든 디바이스 관련 정보를 가져온다.

```
$ sudo lsblk -f
```

```
[ec2-user@ip-10-0-10-54 ~]$ sudo lsblk -f
NAME      FSTYPE LABEL UUID                                MOUNTPOINT
xvda
└─xvda1  xfs     /      92462e9b-de38-4177-8f96-ab97410b4979 /
xvdf
[ec2-user@ip-10-0-10-54 ~]$
```

10. 위의 그림에서 확인해보면, 인스턴스에 연결된 2개의 디바이스를 보여주는데, 첫 번째 열에는 디바이스와 해당 파티션이 나열된다. **FSTYPE** 열에는 각 디바이스의 파일 시스템 유형이 표시된다. 특정 디바이스에 대한 열이 비어 있으면 디바이스에 파일 시스템이 없음을 의미한다. 위의 그림에서, 디바이스 **xvda**는 **XFS** 파일 시스템을 사용하여 포맷되어 있지만, 디바이스 **xvdf**에는 파일 시스템이 없는 것을 확인할 수 있다. 이러한 명령의 출력에 디바이스에 파일 시스템이 없다고 표시된 경우 생성해야 한다. 빈 볼륨이 있다면 **mkfs -t** 명령을 이용해 볼륨에서 파일 시스템을 생성한다.

```
$ sudo mkfs.xfs /dev/xvdf
```

```
[ec2-user@ip-10-0-10-54 ~]$ sudo mkfs.xfs /dev/xvdf
meta-data=/dev/xvdf            isize=512    agcount=4, agsize=3276800 blks
                =               sectsz=512    attr=2, projid32bit=1
                =               crc=1        finobt=1, sparse=1, rmapbt=0
                =               reflink=1    bigtime=0 inobtcount=0
data        =                  bsize=4096    blocks=13107200, imaxpct=25
                =               sunit=0      swidth=0 blks
naming      =version 2          bsize=4096    ascii-ci=0, ftype=1
log         =internal log      bsize=4096    blocks=6400, version=2
                =               sectsz=512    sunit=0 blks, lazy-count=1
realtime    =none              extsz=4096    blocks=0, rtextents=0
[ec2-user@ip-10-0-10-54 ~]$
```



11. 디바이스에 파일 시스템 생성이 성공적으로 끝나면 다음 명령을 통해 파일 시스템 유형에 관한 정보를 표시하게 된다.

```
$ sudo file -s /dev/xvdf
```

```
[ec2-user@ip-10-0-10-54 ~]$ sudo file -s /dev/xvdf
/dev/xvdf: SGI XFS filesystem data (blksz 4096, inosz 512, v2 dirs)
[ec2-user@ip-10-0-10-54 ~]$
```

12. **lsblk -f** 명령을 사용하여 인스턴스에 연결된 모든 디바이스 관련 정보를 가져와서 확인해 보자.

```
$ sudo lsblk -f
```

```
[ec2-user@ip-10-0-10-54 ~]$ sudo lsblk -f
NAME      FSTYPE LABEL UUID                                MOUNTPOINT
xvda
└─xvda1   xfs     /      92462e9b-de38-4177-8f96-ab97410b4979 /
xvdf      xfs     36ff19ea-8b12-4c0a-ae0c-d204d37f9d11
```

13. 추가된 볼륨의 마운트를 위해 **mkdir** 명령을 사용하여 볼륨에서 사용할 탑재 지점 디렉터리를 생성한다. 마운트 포인트는 파일 시스템 트리에 볼륨이 위치하고 볼륨을 마운트한 후 파일을 읽고 쓰는 위치이다. 이제 **/data**라는 이름의 디렉터리를 생성하자.

```
$ sudo mkdir /data
```

다음 명령을 사용하여 이전 단계에서 생성한 디렉터리에 볼륨을 탑재한다.

```
$ sudo mount /dev/xvdf /data
```

14. 마운트 후 **lsblk**를 실행하면 성공적으로 마운트됐음을 확인할 수 있다

```
[ec2-user@ip-10-0-10-54 ~]$ sudo mkdir /data
[ec2-user@ip-10-0-10-54 ~]$ sudo mount /dev/xvdf /data
[ec2-user@ip-10-0-10-54 ~]$ lsblk
NAME      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda      202:0    0  8G  0 disk
└─xvda1   202:1    0  8G  0 part /
xvdf      202:80   0 50G  0 disk /data
```



15. 새 볼륨 마운트의 파일 권한을 검토하여 사용자 및 애플리케이션이 볼륨에 기록할 수 있는지 확인해보자. **data** 디렉토리로 이동하여 **Hello.txt** 파일을 생성한다

```
[ec2-user@ip-10-0-10-54 ~]$ cd /data
[ec2-user@ip-10-0-10-54 data]$ sudo touch Hello.txt
[ec2-user@ip-10-0-10-54 data]$
```

16. 또한 에디터를 사용하여 **Hello.txt**에 다음과 같이 **Hello, Amazon EBS!!!**라고 텍스트를 입력한다. 새로 추가된 볼륨에 데이터를 기록할 수 있음을 알 수 있다.

```
[ec2-user@ip-10-0-10-54 data]$ sudo nano Hello.txt
[ec2-user@ip-10-0-10-54 data]$ cat Hello.txt
Hello, Amazon EBS!!!
[ec2-user@ip-10-0-10-54 data]$
```

17. Mount한 탑재 지점은 인스턴스를 재부팅하면 자동으로 보존되지 않는다. 재부팅 후에도 이 **EBS 볼륨**을 자동으로 탑재하고 싶다면 추가 작업을 해야 한다. 시스템을 재부팅할 때마다 연결된 **EBS 볼륨**을 탑재하려면, 디바이스에 대한 항목을 **/etc/fstab** 파일에 추가한다. **/dev/xvdf**에 있는 **/etc/fstab** 같은 디바이스 이름을 사용할 수 있지만 디바이스의 128비트 **UUID(Universally Unique Identifier)**를 사용할 것을 권장한다. 디바이스 이름은 바꿀 수 있지만, **UUID**는 파티션 수명이 다할 때까지 유지되기 때문이다. **UUID**를 사용하면 하드웨어 재구성 후 시스템을 부팅할 수 없게 되는 경우가 줄어들게 된다. 먼저 수정 도중 실수로 이 파일이 손상되거나 삭제되는 경우에 대비하여 **/etc/fstab** 파일의 백업을 생성한다.

**\$ sudo cp /etc/fstab /etc/fstab.bak**

```
[ec2-user@ip-10-0-10-54 data]$ sudo cp /etc/fstab /etc/fstab.bak
[ec2-user@ip-10-0-10-54 data]$
```

18. **blkid** 명령을 사용하여 디바이스의 **UUID**를 찾는다. 재부팅 후 탑재할 장치의 **UUID**를 기록해 둔다. 다음 단계에서 필요하게 된다.

**\$ sudo blkid**

```
[ec2-user@ip-10-0-10-54 data]$ sudo blkid
/dev/xvda1: LABEL="/" UUID="92462e9b-de38-4177-8f96-ab97410b4979" TYPE="xfs" PARTLABEL="Linux" PARTUUID="31b1f562-5e23-4716-9676-2a025e74b6ea"
/dev/xvdf: UUID="36ff19ea-8b12-4c0a-ae0c-d204d37f9d11" TYPE="xfs"
[ec2-user@ip-10-0-10-54 data]$
```

19. **/etc/fstab**를 텍스트 편집기를 사용하여 파일을 오픈하여 다음의 값을 추가하고 파일을 저장한다. 여기서 **UUID**의 값은 추가된 볼륨의 **UUID** 값이다.

```
$ sudo nano /etc/fstab
```

```
UUID=e8b813f1-13e5-4a4c-9e69-016667a9b805 /data xfs defaults,nofail 0 2
```

```
GNU nano 2.9.8 /etc/fstab

#
UUID=92462e9b-de38-4177-8f96-ab97410b4979 / xfs defaults,noatime 1 1
UUID=36ff19ea-8b12-4c0a-ae0c-d204d37f9d11 /data xfs defaults,nofail 0 2
```

20. 항목이 제대로 작동하는지 확인하기 위해, 다음 명령을 실행해 디바이스 탑재를 해제하고 **/etc/fstab**에서 모든 파일 시스템을 탑재한다. 오류가 없다면 **/etc/fstab** 파일에 문제가 없다는 뜻이며, 파일 시스템은 재부팅 후 자동으로 탑재될 것이다.

```
$ cd ~
```

```
$ sudo umount /data
```

```
$ sudo mount -a
```

```
[ec2-user@ip-10-0-10-54 ~]$ sudo umount /data
[ec2-user@ip-10-0-10-54 ~]$ sudo mount -a
[ec2-user@ip-10-0-10-54 ~]$
```

21. **lab-amazon-ec2** 인스턴스를 재 부팅 후 다음의 명령으로 재 마운트된 것을 확인할 수 있다.

```
$ lsblk
```

```
[ec2-user@ip-10-0-10-54 ~]$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda        202:0    0   8G  0 disk
└─xvda1     202:1    0   8G  0 part /
xvdf        202:80   0  50G  0 disk /data
[ec2-user@ip-10-0-10-54 ~]$
```