EXT4, XFS ,ZFS, NTFS

파일 시스템

용어

파일 시스템

장치에 데이터를 저장할 때 데이터의 위치나 파일명, 형식 등을 결정하는 방법

운영체제가 파일을 시스템의 디스크상에 구성하는 방식

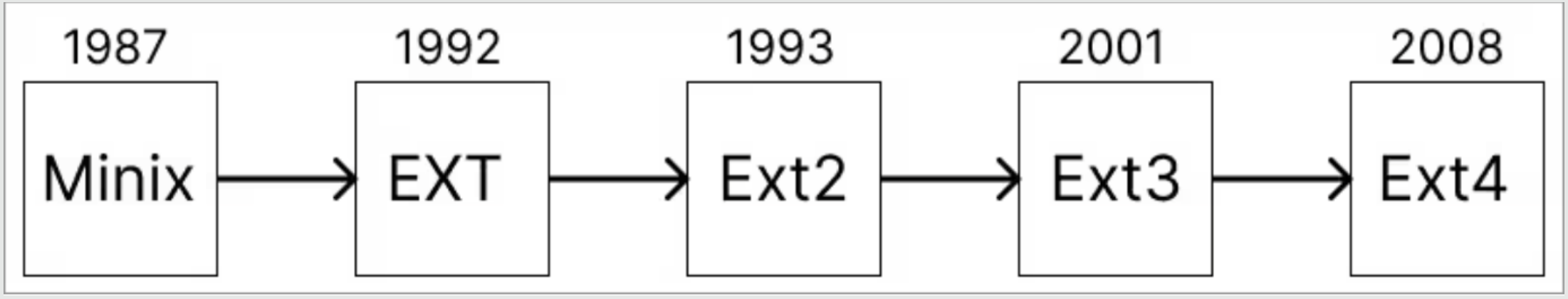
저널링(Journaling)

데이터베이스에서 일관성 체크를 위해 사용되는 방법이다.
파일시스템이 깨지는 것과 같은 시스템 장애를 방지하기 위해 저널(기록)을 남겨두는 기능

매일 갱신되는 데이터의 변경이력을 남기는 것

데이터 복구, 백업 등에 이용

EXT4



EXT4 (EXTended file system)

우분투 및 데비안과 같이 널리 사용되는 리눅스 배포판의 기본 저널링 파일 시스템으로 사용

Ext4에는 성능, 확장성 및 신뢰성을 향상시킨 수많은 새 기능이 도입

가장 눈에 띄는 특징은 ext4가 1EB(exabyte)의 파일 시스템을 지원한다는 것이다.

EXT4 특징 및 기능

대용량 파일 시스템

파일 시스템 볼륨, 파일 크기 및 서브디렉토리 제한에 대한 지원이 향상

최대 1EB(1000PB)의 파일 시스템을 지원.
ext4에서 허용되는 최대 파일 크기는 16TB(4KB 블록 가정)

서브디렉토리 제한도 32KB 디렉토리 깊이에서 거의 무한대로 확장

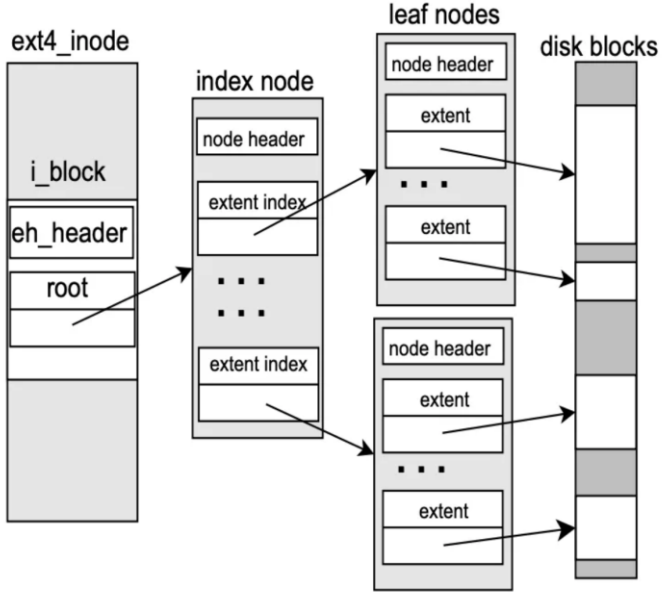
	EXT3	EXT4
Max filesystem size	16 TiB	1 EiB
Max file size	2 TiB	16 TiB

익스텐트(Extend)

대용량 파일의 메타데이터를 줄이고 성능을 향상시키기 위한 기법

연속 블록으로 구성된 긴 목록의 저장 위치에 대한 정보가
유지되기 때문에 저장되는 전체 메타데이터의 용량이 줄어든다.

계층화된 접근 방법을 통해 작은 파일을 효율적으로 나타내며
익스텐트 트리를 사용하여 대용량 파일을 효율적으로 나타낸다.



EXT4 특징 및 기능

파일 시스템 저널에 대한 체크섬 검사

좀 더 안정적으로 변경 사항을 구현할 수 있으며
작업 중에 시스템 오류 또는 전원 문제가 발생하더라도 일관성을 유지

저널에 대한 체크섬 기능을 구현하여 올바른 변경 사항만 기본 파일 시스템에 적용되도록 보장

사전할당(Pre-allocation), 지연할당(Delayed-allocation)

사전할당은 파일 생성 시 미리 일정 개수의 블록을 보장하는 기법

fallocate() 시스템 콜을 이용하여 사전 할당 기능을 제공

지연할당은 free block count만 갱신하고 실제 블록 할당은 뒤로 미루는 방식

block allocator가 블록 할당을 최적화할 수 있기 때문에 성능이 향상되고 fragmentation을 개선

XFS

XFS

강력하고 확장성 높은 단일 호스트 64비트 저널링 파일 시스템

익스텐스 기반으로 매우 큰 파일 시스템을 지원

빠른 복구를 위한 메타데이터 저널링 지원

마운트되어 활성화된 상태로 조각 모음 및 확장 가능

특정 유틸리티의 백업 및 복원을 지원

b-tree 구조를 사용하여 우수한 I/O 확장성을 제공하고 모든 사용자 데이터 및 메타 데이터를 인덱스한다.

XFS 특징 및 기능

저널링(Journaling)

파일 수에 관계없이 예상치 못한 상황으로부터 신속하게 복구하여 재시작이 가능

익스텐스 기반으로 매우 큰 파일 시스템을 지원

XFS는 체크 프로그램을 사용하지 않아도 된다. 이를 통해 데이터 손실 가능성을 줄일 수 있다.

익스텐트 기반 할당(Extent-based Allocation)

메타데이터가 소비되는 공간과 조각화를 최소화하며, 대용량 파일의 성능을 향상

지연 할당(Lazy Allocation)과 사전 할당(Pre-allocation) 또한 지원하여, 성능 저하를 최소화

XFS 특징 및 기능

트랜잭션

데이터 읽기/쓰기 트랜잭션으로 인한 성능 저하를 최소화

XFS의 저널링 구조와 알고리즘은 트랜잭션에 대한 로그 기록을 신속하게 할 수 있도록 최적화

높은 확장성

완전한 64비트 파일 시스템이기 때문에 100만 TB 크기의 파일도 다룰 수 있다.

사용 가능한 공간에 의해서만 파일 수가 제한되며, 마운트된 상태에서 확장이 가능

ZFS

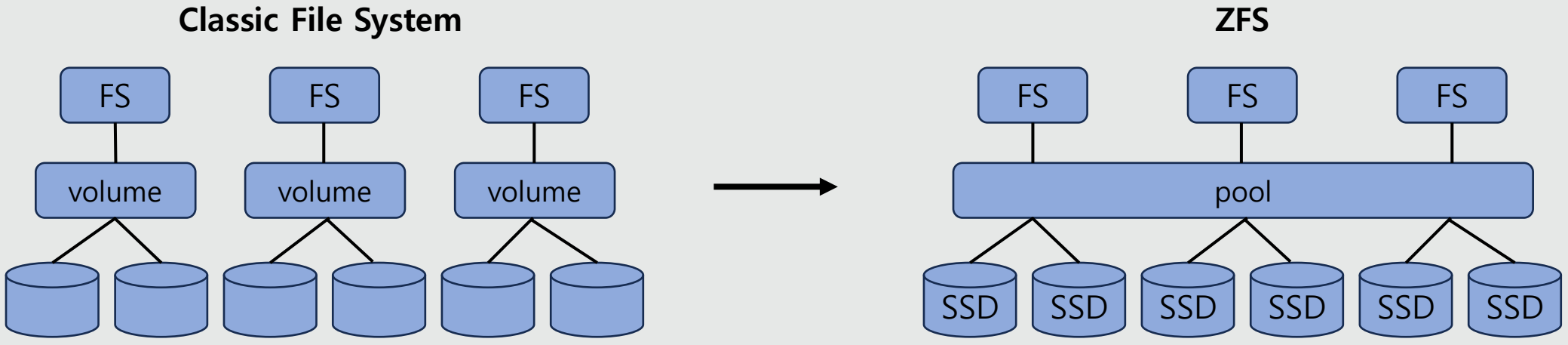
ZFS

안정성과 성능, 그리고 데이터 보호 기능들이 강화된 파일 스토리지를 목적으로 개발

ZFS의 핵심 컨셉은 파일 시스템과 볼륨 매니저를 같이 제공

물리적인 스토리지와 이로 구성된 볼륨 구조 뿐만 아니라,
여기에 저장된 파일에 대한 정보도 가지고 있다.

ZFS 특징 및 기능



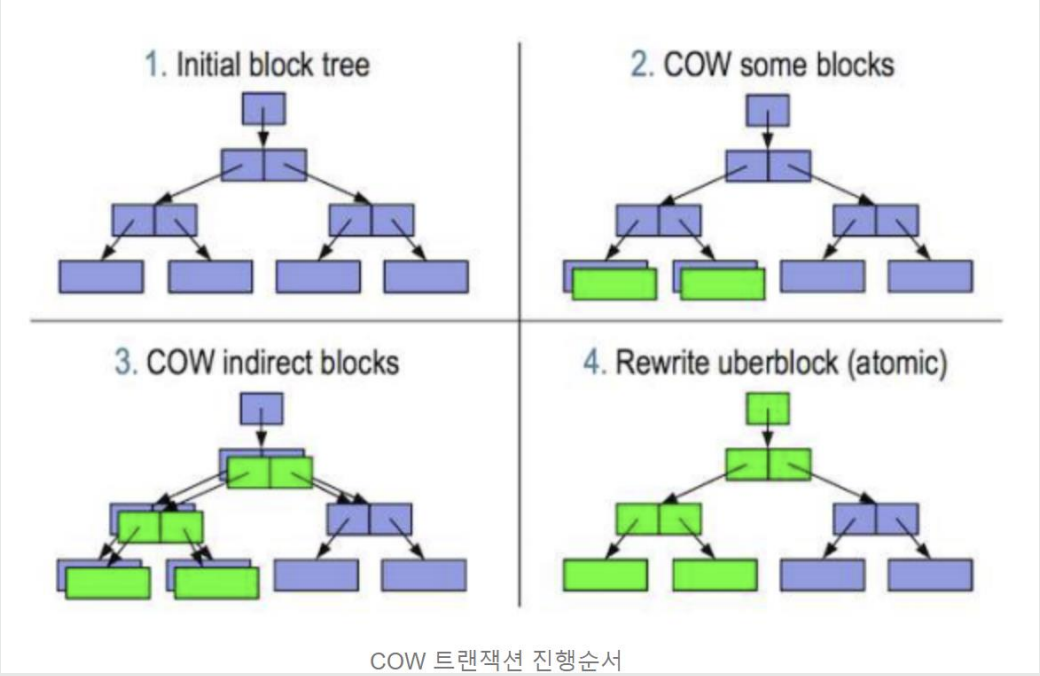
스토리지 풀(Storage Pool)

볼륨 확장 혹은 디스크 추가 시 스토리지 풀에 결합하며,
결합 직후 직시 용량이 반영되어 사용이 가능

임의의 데이터 저장소 역할을 수행하며, 모든 파일 시스템이
스토리지 풀로 구성된 스토리지에 데이터 저장이 가능

실제로 사용한 공간 만큼만 차지하기 때문에 효율적으로 디스크 사용이 가능

ZFS 특징 및 기능



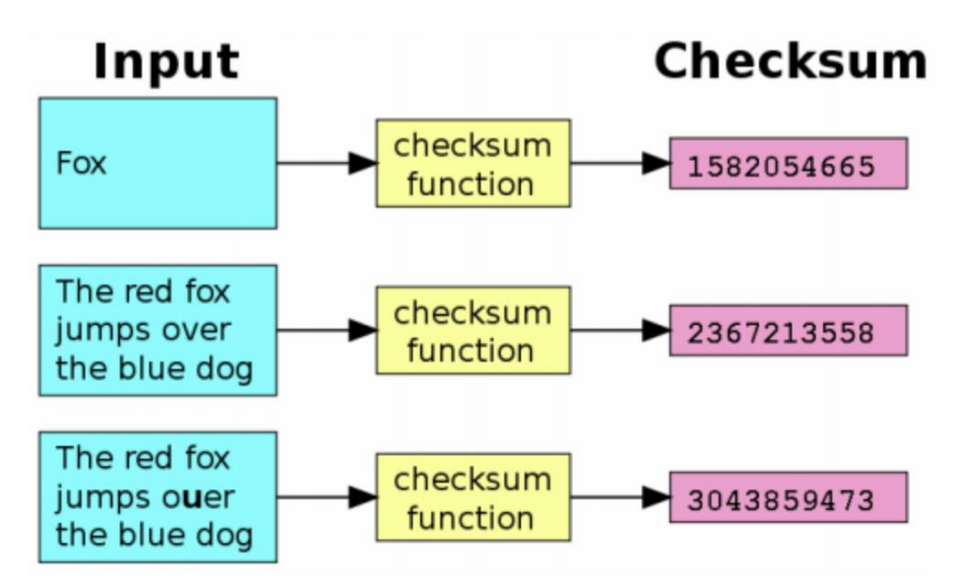
COW 트랜잭션 진행순서

COW(Copy-On-Write) 트랜잭션

스토리지에서 데이터를 관리하기 위해 기록 시 복사(copy-on-write) 모델을 사용

이전 데이터는 새로운 쓰기 작업이 완료된 후에 유지되고 데이터를 절대 겹쳐 쓰지 않기 때문에
변경 작업이 완료되면 포인터를 새로 쓰여진 블록으로 변경하여 저장

ZFS 특징 및 기능



체크섬 예시

체크섬(Checksum)

체크섬이란 중복검사의 한 형태로 오류 정정을 통해 공간이나 송신된 자료의 무결성을 검증하는 방법

메모리 기반의 엔드 투 엔드(end-to-end) 체크섬을 제공

메모리 기반이기 때문에 유효성 검사 시 서버 메모리 하에서 작업이 이루어져 외부 작업이 가능하며,
체크섬 데이터는 블록을 가리키는 포인터의 옆에 저장

파일 시스템 확인하기

```
dpwls@BOOK-PMTSHF3Q4P:~$ findmnt
TARGET                                SOURCE        FSTYPE     OPTIONS
/                                     /dev/sdc      ext4       rw,relatime,discard,errors=remount-ro,data=ordere
├─/mnt/wsl                            none          tmpfs      rw,relatime
├─/usr/lib/wsl/drivers                none          9p         ro,nosuid,nodev,noatime,dirsync,aname=drivers;fma
├─/usr/lib/modules                    none          tmpfs      rw,relatime
│ └─/usr/lib/modules/5.15.146.1-microsoft-standard-WSL2
│                                     none          overlay    rw,nosuid,nodev,noatime,lowerdir=/modules,upperdi
├─/mnt/wslg                           none          tmpfs      rw,relatime
│ ├─/mnt/wslg/distro                  /dev/sdc      ext4       ro,relatime,discard,errors=remount-ro,data=ordere
│ ├─/mnt/wslg/versions.txt            none[/etc/versions.txt]
│ │                                                overlay    rw,relatime,lowerdir=/systemvhd,upperdir=/system/
│ └─/mnt/wslg/doc                     none[/usr/share/doc]
│                                                  overlay    rw,relatime,lowerdir=/systemvhd,upperdir=/system/
├─/usr/lib/wsl/lib                    none          overlay    rw,nosuid,nodev,noatime,lowerdir=/gpu_lib_package
├─/init                               rootfs[/init] rootfs     ro,size=8079192k,nr_inodes=2019798
├─/dev                                none          devtmpfs   rw,nosuid,relatime,size=8079192k,nr_inodes=201979
│ ├─/dev/pts                          devpts        devpts     rw,nosuid,noexec,noatime,gid=5,mode=620,ptmxmode=
│ ├─/dev/shm                          none          tmpfs      rw,nosuid,nodev,noatime
│ ├─/dev/hugepages                    hugetlbfs     hugetlbfs  rw,relatime,pagesize=2M
```

findmnt

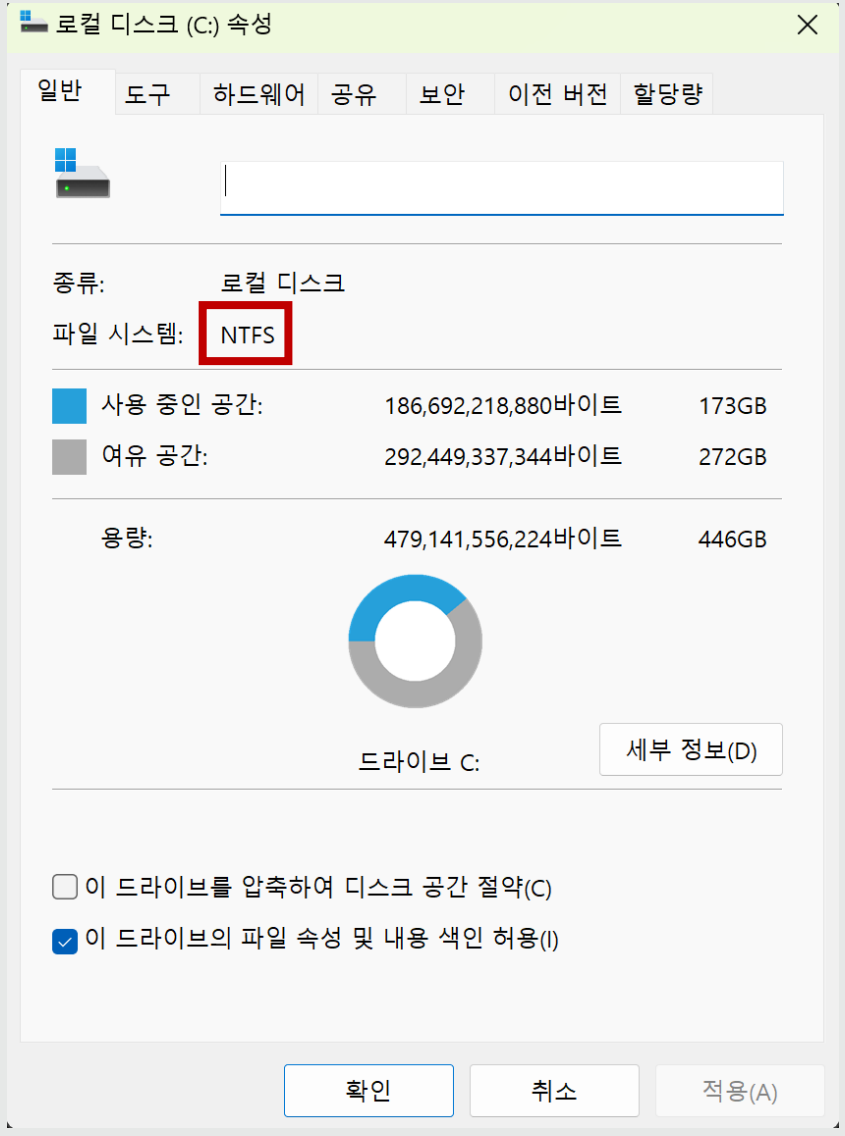
리눅스 시스템에서 파일 시스템의 마운트 정보를 확인하는 데 사용되는 효과적인 도구이다.

마운트 정보를 찾아 출력하여 보여준다.

전체 파일 시스템의 상세한 정보를 한눈에 파악할 수 있기 때문에
디스크 관리와 문제 해결에 매우 유용하다.

하지만, 특정 디렉토리나 파일에 대한 디스크 사용량은 확인할 수 없다.

NTFS



NTFS (New Technology File System)

윈도우 운영체제의 기본으로 탑재되는 파일시스템

FAT이후 등장한 NTFS

NTFS는 운영체제 기능의 변화에 따라 지속적으로 발전

- 파일시스템 안정성 강화 및 자체적인 보안 기능 추가
- 다른 파일 시스템과의 호환성을 개선

NTFS 안정성 강화 기능

저널링(Journaling)

- 파일의 변경 내용을 기록하는 로깅(Logging) 기능 -> 시스템 오류 발생 시, 변경 작업을 복원(Rollback)
  - $LogFile : NTFS Metadata 변경 사항(파일 생성, 삭제, 변경 등)을 기록
  - $UsnJrnl : 파일 및 폴더 변경 사항(보안 설정 등)을 기록

VSC(Volume Shadow Copy)

- Windows 2003부터 사용되는 파일시스템 백업 기능
- 파일 및 폴더의 백업본을 유지하여, 저널링 기능과 함께 좀 더 안전한 복구를 지원

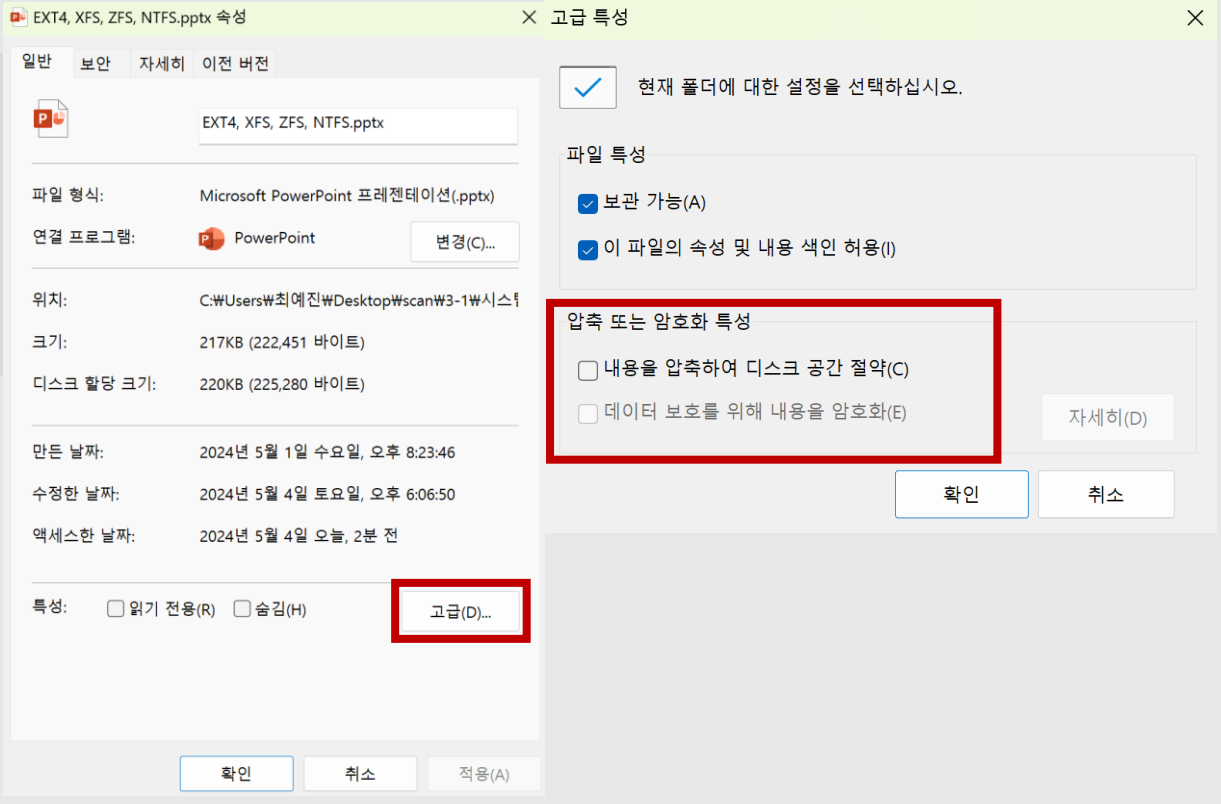
NTFS 보호 기능

EFS(Encrypting File System)

- 인증 받지 않은 사용자나 프로그램으로부터 사용자의 데이터를 암호화하여 보호하는 기능
- 암호화된 파일 또는 폴더를 이동 및 복사해도 암호화는 유지된다.

파일 압축(File Compression)

- 파일 압축 기능을 제공하여 공간 절약 가능
- 압축으로 인한 파일 접근 성능이 저하될 수 있다.



타 파일 시스템과의 호환성을 위한 기능

유니코드(Unicode) 지원

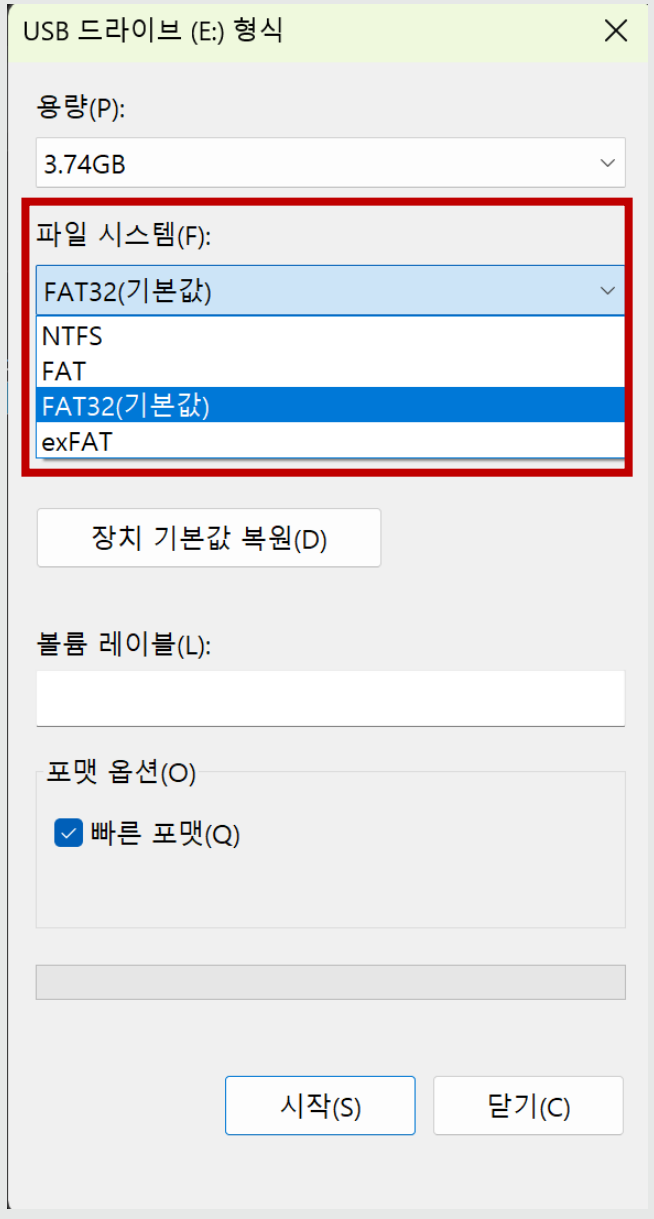
- 다국어 지원을 위해 유니코드 지원
- 파일, 폴더, 볼륨 등의 이름 지정 시 모두 유니코드 사용

ADS(Alternate Data Streams)

- MAC 운영체제에서 사용하는 파일시스템과 호환성을 위한 ADS 영역 추가
  - MAC 운영체제 파일 구조 

  - MAC 운영체제는 Resource 영역에 파일의 아이콘 등의 정보를 저장
- ADS는 파일 요약 정보를 저장하거나 영역 식별자로 사용이 가능
  - ADS 특징을 이용하여 정보 은닉 가능

이동장치(USB) NTFS



USB 파일시스템

FAT32

- 윈도우에서 포맷 기준 최대 드라이브 크기는 32GB 지원
- 파일 하나의 크기는 최대 4GB까지 지원
- 구조가 간단해서 읽고 쓰는 속도가 빠르다.
- 호환성과 안전성이 좋아 다른 OS에도 사용 가능

NTFS

- 최대 드라이브 크기는 256TB, 최대 파일 크기는 16TB
- 보안기능 및 복구 기능 강화
- 타 운영체제에서 인식이 안되거나 읽기만 지원 (호환성↓)
- 거의 윈도우 전용

exFAT

- 최대 드라이브 크기와 최대 파일 크기는 512TB
- 타 운영체제에서 읽기와 쓰기 가능
- 호환성이 좋음
- 안정성은 떨어짐

Reference

저널링
https://velog.io/@mysprtlty/저널링-Journaling
https://m.blog.naver.com/have29/221325099350

EXT4
https://moaimoai.tistory.com/111
https://ddongwon.tistory.com/66
https://medium.com/naver-cloud-platform/posix-알아보기-1-linux-리눅스-파일-시스템의-종류와-특징-96a2e93e33b3
https://recoverit.wondershare.kr/file-system/ext4-file-system.html

XFS
https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=hymne&logNo=220976678541
https://easyitwanner.tistory.com/87
https://thinking-developer.tistory.com/52

ZFS
https://tech.gluesys.com/blog/2023/12/20/ZFSintro.html
https://majjangjjang.tistory.com/140 - 기능
https://dataonair.or.kr/db-tech-reference/d-lounge/technical-data/?mod=document&uid=236552
https://admion.net/what-is-zfs/

NTFS
https://blog.forensicresearch.kr/15
https://www.youtube.com/watch?v=t8GPuO31iJQ – 개념과 기능
https://www.youtube.com/watch?v=pLmzIwMJ-EA

파일시스템 확인하기
https://www.infracody.com/2023/09/understanding-linux-file-systems-guide-to-checking-file-systems.html#elcreative_toc_df_____
https://www.lesstif.com/system-admin/findmnt-93127532.html
https://ko.linux-console.net/?p=8380

감사

합니다