챕터 2 : 가상 장치 (Virtual Devices)

이 장에서는 소시지가 어떻게 만들어지는지 살펴보겠습니다. 이것은 ... 돼지, 즉 디스크입니다. 디스크는 스토리지의 물리적 표현입니다. 디스크는 사악합니다. 디스크는 그 특성과 레이아웃에 대해 거짓말을 하고, 오류를 숨기며, 예상치 못한 방식으로 실패합니다. ZFS를 사용하면 더 이상 디스크가 몰래 음모를 꾸미는 것은 아닌지 걱정할 필요가 없습니다. 예, 디스크가 음모를 꾸미고 있기는 하지만, ZFS는 디스크의 배신을 폭로하고 이를 중단시킵니다.

ZFS에서 사용 가능한 디스크를 가장 효과적으로 사용하려면 운영 체제가 ZFS에 디스크를 제공하는 방식과 ZFS가 해당 디스크의 데이터를 정렬하는 방식에 대한 기본적인 이해가 필요합니다.

디스크 및 기타 저장 매체 (Disks and Other Storage Media)

ZFS는 디스크가 아닌 다른 저장 매체에서도 실행할 수 있습니다. FreeBSD GEOM 저장소 제공자라면 무엇이든 ZFS 저장 매체가 될 수 있습니다. ZFS는 파일을 백업 스토리지로 사용하는 기능도 지원하는데, 이는 테스트용으로는 정말 좋지만 프로덕션용으로는 적합하지 않습니다. ZFS는 모든 블록 장치를 물리적 저장소로 사용할 수 있지만 각 유형마다 장단점이 있습니다.

원시 디스크 스토리 (Raw Disk Storage)

전체 물리적 디스크를 사용하면 복잡성이 줄어듭니다. 또한, 파티셔닝에 대해 걱정할 필요가 없으며, ZFS와 물리적 디스크 사이에 소프트웨어나 구성이 필요하지 않습니다. 그러나 일반적으로 이러한 장점보다 단점이 더 큽니다.

디스크에서 부팅하려면 디스크에 부트 로더가 있어야 합니다. 부트 로더는 파티션된 디스크에서만 사용할 수 있습니다. 원시 디스크는 부팅할 수 없습니다. FreeBSD는 디스크에 유용한 레이블을 지정할 수 있도록 지원하지만, 이러한 레이블은 파티션 정보 안에 있습니다.

더 나쁜 점은, 대체 디스크는 원본 디스크와 정확히 같은 크기이거나 더 커야 한다는 것입니다. 6TB 디스크의 크기가 모두 같은 것은 아니며, 공급업체마다 디스크 크기가 몇 메가바이트씩 다릅니다. 시스템을 설정할 때는 이러한 차이가 중요하지 않지만 디스크를 교체할 때는 매우 중요합니다. 대부분의 카탈로그에는 각 디스크의 섹터 수가 나열되어 있지 않고 크기만 나와 있기 때문에 사용 가능한 대체품을 찾으려면 여러 번 시도해야 할 수 있습니다. 기존 512바이트 섹터를 사용하는 드라이브를 4096바이트 (4K, 고급 포맷이라고도 함) 섹터를 사용하는 드라이브로 교체하면 상황이 더욱 복잡해집니다. 원래 드라이브에는 8로 균등하게 나눌 수 없는 수의 섹터가 있었을 가능성이 높습니다. 디스크 드라이브에서 사용하는 특수 수학 덕분에 새 드라이브는 이전 드라이브보다 몇 바이트 더 크더라도 몇 바이트 더 작아 보일 수 있습니다.

파티션 스토리지

□□ □□ □□□□□ □□□□□ □□ □□□□□ □□□□□ □□ □ □□□□ □ □□□□ ZFS□□□ □□□□ □ □□□□□ . □ □□□□ □□ □ □□□□ □□□□ □□ □□□□ □□ □□ □□ □□ □□□□□ □□□□ ZFS □□□□ □□□□ □□□□□□ □□□□ □ □□□□ □□□□□ . □□ □□□□□ □□□□□ □□ □□ □□□□□□ □□ □□ □□□□□ □□ □□ □□□□ □□□□ □□□□ □□□□ □□□□ □ □□□□ , □□ □□□□□ □□ □□ □□□□□ □□□□ □□□□ □□ □□□□ □□ □□□□ □□□□ □ □□□□□ . □□ □□□□□□ □□ □□□□□□ '□□ ' □□□□ □□□□ □□□□□ □□ □□□□□ .

□□ □□ Solaris ZFS □□ □□□□□□□ □□□□□ □□□□ □□□□ (□□ Solaris □□□□□ □□□□□)□ □□□□□ □ □□ □□□□□□ . Solaris□□ □□ □□□□□ □□□□□ □□□□□ □□ □□□□ □□□□□□□□ . FreeBSD□□ □□ □□□□ □□□□□□□ □□ □□□□ □□□□□□□ □□ □□□□□□ □□□□ □□□□□□ . FreeBSD□ □□□□□□ ZFS□ □□□□ □ □□□□ □□□□ □□□□□□ .

FreeBSD□ □□ □□ □□□□□ □□□□ □□□□□□ , GPT□ □□□□ □□□□□□ . □□ □□□□□ □□□□□ MBR □ □□ □□□□□ □□ 4□□ □□□□ □□ , GPT□ □□ 128□□ □□□□□ □□□□□□ . MBR□ □□ 2TB □□□□ □□□□□ □□□□ □ □□ □□ , GPT□ 512□□□ □□ □□□□□ □□ 8ZB, 4K-□□ □□□□□ □□ 64ZB□□ □□□□ □ □□□□□ . FreeBSD Mastery: Storage Essentials□□□ □ □□ □□□□□ □□□□ □□ FreeBSD□ □□□□ □□□□□ .

> □□□□ □□□□□ □□□□□ □□□□□ □□ , bsdlabe(8) □□□□□ □□□□ □ □□□□□ .

□□□□ □□□□ □□□□ ZFS□ □□□□□ □□ □□□□□ □□ □ □□□□ □□□□□ . □ □□□□□□ □□ □□□□□□ □□□□□ □□□□□ □□ , □□ □□□□□ □□□□ □□□□□ □□□□ □ □□□□ □□□□ .

GEOM □□ □□□□□ (GEOM Device Storage)

ZFS□ □□□□ FreeBSD GEOM □□□□□ □□ □□□□□□ □□□□ □□ □□□□□ . □□□□ □□□□□ □□ □□□□□ □□□□ □□ □□□□ □□□□□ □□□□ □□□□ □□□□□□ . GEOM □□□□□ □□ □□□□ □□□□ (GELI, GBDE), □□□□□ , □□□□ , □□ □□ □ □□□□□ □□□□□□ □□ □□□□ □□□□□□ . GEOM □□□□□ □□ □□□□ □□□□□ □□□□□ □□□□ , □□ □□ □□ □□ □□□□□ □□□□□ □□ □□ GEOM □□□□ □□ □□ □ □□□□□ .

□□□□□ ZFS □□ □□□□□ □□ □□ □□□□ GELI(FreeBSD Disk Encryption Subsystem)□□□ . GELI□ ZFS□ □□□□ □□□□ □□□□ □□ □ □□□□ □□□□□□ □□□□□□ ZFS□□ □□ □□□□ □ □□□□ □□□□□ . GELI□ □□□□ □□□□ □□□□□□ □□□□□□ , □□□□□ AES-XTS□ □□ AES-NI(AES New Instructions)□ □□□□□ □□ CPU□□ □□□□ □□□□ □□□□□□ . □ □□□□□ □□□□□ □□□□ □□□□□ GELI□ □□ 1GB □□□□ □□□□ □□□□□ □□□□□□ □ □□□□ □□□□ □ □□□□□ , SSD□□□ □□□□□ □□□□□ □□□□ □□□□□ □□□□□ . □□ GELI□ □ □□□□ □□ □□□□ □□ □□ (Heshed Message Authentification Code, HMAC)□ □□□□□ □□□□ □□ (□□□□ □□) □□□□ □□□□□□ □□□□ □ □□□□□ . □ HMAC□ □□□□□ □□□□□ □□□□ (□□□□□ □□□□□ □□□□□)□ □□□□ (□ □□□□□ □□□□□ □□□□ □□□□)□ □□□□□□ . □□□□ □□ □□ □ HMAC□ □□□□□ □□□□□ □□□□ □□□□ □□□□□□ . HMAC □□□□ □□□□□□ □□□□□□ □□ □□□□ , ZFS□ □ □□□□ □□□□ □□ □□ □□□□□ □□□□□ □□□□ ZFS□ □□ □□ □□□□ □ □□□□□ .

□□□□□ □□□□□ □□ (High Availability Storage Technology, HAST)□ FreeBSD□ □□ □□□□□ □□□□□□□ . □□□□□□ □□ □□□□ □□ □□ □□□□ □□□□□ □ □□□□□ . HAST□ ZFS □□ □□

스토리지로 사용하면 각 백업 디스크를 두 번째 머신에 미러링할 수 있습니다. HAST의 장점은 실시간이라는 점입니다. 블록은 HAST 클러스터의 모든 호스트에 쓰여질 때까지 쓰여진 것으로 간주되지 않습니다. 반면에 ZFS 복제는 주기적인 스냅샷 동기화를 기반으로 합니다. 그러나 HAST를 사용하면 두 번째 머신은 첫 번째 머신과 동시에 풀을 가져오거나 마운트할 수 없습니다. 복제된 풀을 동시에 활성화(읽기 전용)할 수 있는 ZFS 복제와 비교했을 때, HAST는 몇 가지 경우에만 의미가 있습니다.

GEOM 레이블은 각 디스크나 파티션에 의미 있는 메모를 첨부할 수 있는 편리한 방법을 제공합니다. disk ident, gptid, GPT 레이블과 같은 표준과 GEOM-specific glabel을 포함한 다양한 레이블 유형이 있습니다. 드라이브 라벨링 모범 사례는 0장에 나와 있습니다.

GEOM은 고가용성을 위한 다중 경로도 지원합니다. 디스크뿐만 아니라 컨트롤러 카드, 백플레인 또는 케이블도 고장이 나는 경우가 있습니다. 다중 경로를 사용하면 '듀얼 포트'인 엔터프라이즈 드라이브를 둘 이상의 HBA(RAID 기능이 없는 디스크 컨트롤러 카드)에 연결할 수 있습니다. 각 드라이브에 두 개의 서로 다른 스토리지 컨트롤러에 대한 경로가 있는 경우, 해당 컨트롤러 중 하나가 손실되더라도 드라이브는 살아남을 수 있습니다. 그러나 각 디스크가 서로 다른 두 개의 컨트롤러에 연결되어 있으면 운영 체제는 각 디스크를 각 컨트롤러를 통해 한 번씩 두 번 인식합니다. GEOM 다중 경로 클래스를 사용하면 각 디스크에 레이블을 작성하여 동일한 디스크에 대한 연속적인 경로가 그렇게 감지되도록 할 수 있습니다. 이렇게 하면 각 디스크에 대한 하나의 표현을 얻을 수 있으며, 다른 컨트롤러를 통해 해당 디스크에 대한 여러 경로로 뒷받침됩니다. 다중 경로에 대해서는 FreeBSD Mastery: Advanced ZFS에서 다중 경로에 대해 설명합니다.

관리자는 GEOM 스케줄러 모듈을 사용하여 더 나은 성능을 달성하기 위해 다양한 I/O 스케줄링 알고리즘을 지정할 수 있습니다. 이 글을 쓰는 현재, 현재 사용 가능한 스케줄러는 하나의 큐만 사용하는 간단한 형태의 예상 스케줄링인 "as"와 각 클라이언트 큐에 걸쳐 라운드 로빈 서비스를 사용하는 예상 스케줄링인 "rr"입니다. 자세한 내용은 **gsched(8)**을 참조하세요. GEOM 시스템을 사용하면 특정 워크로드에 대한 추가 스케줄링 모듈을 비교적 쉽게 작성할 수 있습니다.

파일 백업 스토리지 (File-Backed Storage)

파일 백업 가상 디스크를 ZFS 저장 장치로 사용할 수 있습니다. 프로덕션 환경에서는 권장하지 않지만, 파일 백업 디스크는 테스트 및 실험용으로 유용할 수 있습니다.

프로바이더와 디스크 (Providers vs. Disks)

"프로바이더"는 FreeBSD의 기술 용어입니다. GEOM 스토리지 제공자는 데이터 스토리지를 제공하는 것을 말합니다. 디스크일 수도 있습니다. 어떤 식으로든 저장소를 변환하는 GEOM 클래스일 수도 있습니다. 기술적으로 말하자면, 이 책에서는 거의 모든 곳에서 디스크 대신 프로바이더라는 단어를 사용해야 합니다. 모든 GEOM 공급자를 ZFS의 백엔드로 사용할 수 있습니다. 문제는 하나의 물리적 디스크가 여러 개의 다른 공급자를 제공할 수 있다는 것입니다. 풀에 여러 개의 서로 다른 공급자가 있을 수 있지만, 이 모든 공급자가 하나의 디스크에 있다면 중복성이 머릿속을 가득 채울 것입니다.

> FreeBSD의 유연한 스토리지 시스템은 멍청한 짓을 할 수 있는 힘을 제공합니다. 하지 마세요.

이 책에서 "디스크"에 대해 설명하는 것은 "디스크 위에 있는 일종의 공급자"를 의미합니다. 디스크에 스왑 파티션과 ZFS 파티션이 있어도 문제없지만, 이 디스크가 전적으로 ZFS 전용일 필요는 없습니다. 하지만 하나의 물리적 디스크에 두 개의 ZFS 파티션을 만들어 미러링하고 물리적 중복성을 확보할 수는 없습니다.

VDEV: 가상 장치 (VDEVs: Virtual Devices)

가상 장치 또는 VDEV는 ZFS의 논리적 저장 장치입니다. 각 VDEV는 하나 이상의 GEOM 공급자로 구성됩니다. ZFS는 여러 가지 유형의 VDEV를 지원하며, 이는 VDEV가 제공하는 중복성 유형에 따라 구분됩니다. 각 디스크에 다른 디스크의 복사본이 들어 있는 일반적인 미러 디스크는 VDEV의 한 유형입니다. 중복성이 없는 플레인 디스크는 또 다른 유형의 VDEV입니다. 그리고 ZFS에는 RAID-Z라고 하는 세 가지 종류의 정교한 RAID가 있습니다.

이러한 VDEV는 3장에서 설명한 스토리지 풀로 배열됩니다. 실제 데이터는 4장에서 다루는 것처럼 풀 위에 저장됩니다. 하지만 가상 디바이스의 배열에 따라 풀의 성능과 물리적 손상에 대한 저항력이 결정됩니다. ZFS의 거의 모든 중복성은 가상 장치에서 비롯됩니다.

스토리지 풀은 하나 이상의 VDEV로 구성되며, 풀 데이터는 중복성 없이 해당 VDEV에 분산됩니다. (4장에서 설명한 대로 복사 속성을 사용하여 중복성을 추가할 수 있지만 전체 디스크 장애에 대한 보호는 제공하지 않습니다.) ZFS 풀은 VDEV를 스토리지 공간을 제공하는 단일 단위로 취급합니다. 스토리지 풀은 VDEV가 손실되더라도 살아남을 수 없으므로, 중복성이 있는 VDEV를 사용하거나 이 풀의 데이터를 손실해도 괜찮다고 미리 결정하는 것이 중요합니다.

한 풀에 여러 개의 VDEV를 사용하면 고급 RAID 어레이와 유사한 시스템이 만들어집니다. RAID-Z2 어레이는 RAID-6과 유사하지만, 두 개의 RAID-Z2 VDEV가 있는 ZFS 풀은 RAID-60과 유사합니다. 미러링된 VDEV는 RAID-1처럼 보이지만, 그룹은 RAID-10과 유사합니다. 이 두 경우 모두 ZFS는 중복성 없이 각 VDEV에 걸쳐 데이터를 스트라이핑합니다. 개별 VDEV가 중복성을 제공합니다.

VDEV 중복성 (VDEVs Redundancy)

둘 이상의 디스크가 포함된 VDEV는 여러 가지 중복성 체계를 사용하여 내결함성을 제공할 수 있습니다. 단일 디스크를 단독으로 이중화할 수 있는 방법은 없습니다. ZFS는 미러 디스크와 여러 패리티 기반 어레이 사용을 지원합니다.

ZFS는 중복성을 사용하여 자가 복구합니다. 중복성이 없는 VDEV는 자가 복구를 지원하지 않습니다. 데이터 세트 레이어에서 이 문제를 해결할 수 있지만 (**copies** 속성을 사용하여), 이중화 VDEV는 자동으로 자가 복구를 지원합니다.

Stripe (1개 공급자)

단일 디스크로 구성된 VDEV를 스트라이프라고 하며, 중복성이 없습니다. 예상할 수 있듯이 단일 공급자가 손실되면 디스크의 모든 데이터가 사라집니다. 스트라이프 풀에는 단일 디스크 VDEV만 포함됩니다.

ZFS 풀은 풀의 모든 VDEV에 걸쳐 데이터를 스트라이핑하고 중복성을 제공하기 위해 VDEV에 의존합니다. 하나의 스트라이프 디바이스에 장애가 발생하면 전체 풀이 장애가 발생합니다. 풀에 저장된 모든 데이터가 사라집니다. 스크래치 파티션의 경우 이 방식은 괜찮지만 데이터가 중요하다면 내결함성을 제공하는 VDEV 유형을 사용하세요.

Mirrors (2개 이상의 공급자)

미러 VDEV는 모든 디스크에 모든 데이터의 전체 사본을 저장합니다. 공급자의 드라이브 중 하나를 제외한 모든 드라이브를 잃어버려도 데이터에 액세스할 수 있습니다. 미러에는 원하는 수의 디스크를 사용할 수

있습니다.

미러는 모든 디스크에서 데이터를 한 번에 읽을 수 있기 때문에 랜덤 및 순차 읽기 속도가 매우 우수합니다. 모든 데이터를 모든 디스크에 써야 하고 가장 느린 디스크가 끝날 때까지 작업이 완료되지 않기 때문에 쓰기 성능이 저하됩니다.

RAID-Z1(3개 이상의 공급자)

ZFS에는 RAID-Z라고 하는 3개의 최신 RAID 스타일 중복 VDEV가 포함되어 있습니다. RAID-Z는 RAID-5와 비슷하지만 파일 무결성을 보장하기 위해 체크섬을 포함합니다. 체크섬과 ZFS의 쓰기 시 복사 기능 (7장) 사이에서 RAID-Z는 불완전한 쓰기로 인해 파일 시스템이 일관되지 않게 되지 않도록 보장합니다.

RAID-Z는 데이터 및 패리티 정보를 모든 디스크에 분산시킵니다. RAID-Z의 공급자가 죽거나 손상된 데이터를 제공하기 시작하면 RAID-Z는 패리티 정보를 사용하여 누락된 데이터를 다시 계산합니다. RAID-Z가 하나의 공급자를 사용하여 패리티 정보를 저장한다고 생각할 수 있지만, 단일 패리티 공급자가 아니라 여러 공급자가 패리티 역할을 순환하며 데이터를 분산합니다.

RAID-Z1 VDEV는 단일 스토리지 공급업체의 장애를 견딜 수 있습니다. 첫 번째 장애가 발생한 드라이브가 교체되기 전에 두 번째 공급자가 장애를 일으키면 모든 데이터가 손실됩니다. 패리티 데이터로 디스크 어레이를 재구축하는 데는 시간이 오래 걸릴 수 있습니다. 2TB 이상의 대용량 디스크를 사용하는 경우, 첫 번째 드라이브를 복구하는 동안 두 번째 드라이브에 장애가 발생할 가능성이 적지 않습니다. 더 큰 디스크의 경우 RAID-Z2를 고려해야 합니다.

RAID-Z2(4개 이상의 공급자

RAID-Z2는 RAID-Z1과 유사하지만, VDEV당 2개의 패리티 디스크가 있습니다. RAID-6과 마찬가지로 RAID-Z2는 두 개의 공급자가 실패하더라도 계속 작동할 수 있습니다. RAID-Z1보다 약간 느리지만 드라이브를 교체하는 데 다소 느긋할 수 있습니다.

RAID-Z3(5개 이상의 공급자)

RAID-Z의 가장 편집증적인 형태인 RAID-Z3는 VDEV당 3개의 패리티 디스크를 사용합니다. 즉, 5개의 디스크 어레이에 3개의 장애 디스크를 사용할 수 있습니다. RAID-Z2보다 약간 느립니다. 네 번째 디스크에 장애가 발생하면 전체 데이터가 손실됩니다.

RAID-Z 디스크 구성 (RAID-Z Disk Configurations)

모든 버전의 RAID-Z를 사용할 때 기억해야 할 중요한 사항 중 하나는 RAID-Z의 공급자 수가 완전히 고정되어 있다는 것입니다. RAID-Z VDEV에 드라이브를 추가하여 확장할 수 없습니다. VDEV를 추가하여 풀을 확장할 수는 있지만 디스크를 추가하여 VDEV를 확장할 수는 없습니다. 이 기능을 추가할 계획이 없습니다.

20개의 하드 드라이브를 수용할 수 있는 호스트가 있다고 가정해 보겠습니다. 나중에 필요에 따라 드라이브를 풀에 추가할 수 있다고 생각하고 12개의 드라이브를 설치하여 단일 RAID-Z2로 사용합니다. 이러한 새 드라이브는 별도의 RAID-Z2 VDEV로 연결해야 합니다. 게다가 VDEV의 밸런스가 맞지 않게 됩니다. 풀에는 단일 12-드라이브 VDEV와 두 번째 8-드라이브 VDEV가 있습니다. 하나는 다른 하나보다 느릴 것입니다. ZFS를 사용하면 이러한 장치를 강제로 함께 풀링할 수 있지만, 그렇게 하는 것은 정말 나쁜 생각입니다.

미리 계획하세요. 물리적 장비, 시작해야 하는 드라이브 수, 스토리지를 확장하는 방법을 고려하세요. 예제 서버는 하나의 RAID-Z2 VDEV를 포함하는 온풀과 나머지 8개의 디스크를 포함하는 완전히 분리된 풀을 원하는 배열로 구성해도 괜찮습니다. 시작도 하기 전에 스스로 목을 자르지 마세요!

RAID-Z 2의 규칙 (The RAID-Z Rule of 2s)

일반적으로 논의되는 구성 중 하나는 2의 배수만큼의 데이터 디스크 수와 주어진 RAID-Z 레벨에 필요한 패리티 디스크를 더하는 것입니다. 즉, 이 규칙에 따르면 RAID-Z1은 2n+1 디스크 또는 3, 5, 7, 9 등의 디스크를 사용해야 합니다. RAID-Z2는 2n+2 디스크 (4개, 6개, 8개 등)를 사용해야 하고, RAID-Z3는 2n+3(5개, 7개, 9개 등)을 사용해야 합니다. 이 규칙은 데이터가 2의 거듭제곱인 크기의 작은 블록으로 구성된 경우에만 유효합니다. 하지만 다른 요인들이 훨씬 더 큰 차이를 만듭니다. 일반적으로 압축이 훨씬 더 효과적인 것으로 간주됩니다. 데이터를 압축하면 블록의 크기가 줄어들어 이러한 이점이 사라집니다.

VDEV 복구 (Repairing VDEVs)

중복 VDEV에 속한 공급자가 장애가 발생하면, 해당 공급자가 속한 VDEV는 "degraded(성능 저하)" 상태가 됩니다. 성능이 저하된 VDEV는 여전히 모든 데이터를 보유하지만 성능이 저하될 수 있습니다. 5장에서는 장애가 발생한 공급자를 교체하는 방법을 다룹니다.

공급자가 교체된 후에는 시스템이 새 공급자에 데이터를 저장해야 합니다. 미러를 사용하면 남은 디스크에서 데이터를 읽고 교체된 디스크에 쓰면 됩니다. RAID-Z의 경우, 데이터를 패리티에서 다시 계산해야 합니다.

ZFS가 RAID와 파일 시스템을 결합하는 방식은 ZFS가 어떤 블록에 데이터가 들어 있고 어떤 블록이 비어 있는지를 알고 있다는 것을 의미합니다. 새 드라이브에 모든 바이트를 기록할 필요 없이 ZFS는 실제로 사용 중인 블록만 기록하면 됩니다. 기존 RAID 컨트롤러는 파일 시스템 계층을 이해하거나 인식하지 못하기 때문에 무엇이 사용 중이고 무엇이 여유 공간인지 알 수 없습니다. RAID 컨트롤러는 디스크를 교체할 때 새 디스크의 모든 바이트를 복사해야 합니다. 즉, 손상된 ZFS RAID-Z가 훨씬 더 빨리 복구되어 데이터 손실을 유발할 수 있는 동시 장애의 가능성이 줄어듭니다. 5장에서 ZFS 복구에 대해 설명합니다.

RAID-Z와 기존 RAID 비교 (RAID-Z versus Traditional RAID)

RAID-Z는 기존 RAID에 비해 여러 가지 장점이 있지만, 가장 큰 장점은 ZFS가 디스크 중복성 계층 외에 볼륨 관리자이자 파일 시스템이라는 사실에서 비롯됩니다.

과거에는 파일시스템이 하나의 디스크에서만 작동할 수 있었습니다. 디스크가 두 개라면 두 개의 파일시스템이 필요했습니다. 기존 RAID를 사용하면 여러 디스크를 하나의 가상 디스크로 결합하여 100MB 이상의 대용량 디스크를 만들 수 있었습니다! 그런 다음 운영 체제는 물리적 디스크에 블록이 어떻게 배치될지 전혀 이해하지 못한 채 그 위에 자체 파일 시스템을 얹었습니다. 동시에 RAID는 내결함성을 제공할 수 있습니다. 당시 하드웨어와 소프트웨어의 한계를 고려할 때 RAID는 꽤 괜찮은 선택으로 보였습니다.

파일 시스템과 볼륨 관리자를 결합함으로써 ZFS는 모든 데이터가 어디에 있는지, 스토리지 계층과 데이터가 어떻게 상호 작용하는지 정확히 파악할 수 있습니다. 이를 통해 ZFS는 디토 블록(3장)과 같은 중요한 데이터의 추가 복사본을 별도의 디스크에 저장하는 것과 같은 여러 가지 중요한 결정을 내릴 수 있습니다. 하나의 기본 스토리지 공급업체에 중요한 데이터의 사본을 두세 개씩 저장해 두면 한 번의 하드웨어 장애로 인해 데이터가 모두 지워질 수 있기 때문입니다. ZFS는 인접한 디스크에 동일한 블록을 배치하는데, 이는 두 개의 디스크가 동시에 장애가 발생할 가능성이 통계적으로 낮기 때문입니다.

기존 RAID는 2단계 작업이 중간에 중단되는 "쓰기 구멍 (write hole)"이라는 단점이 있습니다. RAID 5 및 6 장치는 모든 데이터 디스크에 기록할 데이터를 청크 단위로 묶습니다. 이 작업이 완료되면 패리티 블록이 계산되어 패리티 디스크에 저장됩니다. 데이터가 기록된 후 패리티가 기록되기 전에 시스템이 충돌하거나 전원이 차단되면 디스크는 불확실한 상태가 됩니다. 시스템이 다시 시작되면 데이터가 패리티와 일치하지 않습니다. 미러링된 드라이브에서도 한 드라이브는 업데이트를 완료하고 다른 드라이브는 완료하지 못하면 같은 일이 발생할 수 있습니다.

쓰기 구멍 문제는 고장난 디스크를 교체하기 전까지는 발견되지 않습니다. 패리티가 잘못되었거나 미러가 잘못되면 RAID 장치가 파일시스템에 가비지 데이터를 반환하게 됩니다. 기존 파일시스템은 이 가비지 데이터를 파일의 콘텐츠로 반환합니다.

ZFS는 쓰기 시 복사 및 체크섬으로 이러한 문제를 해결합니다. 쓰기 시 덮어쓰기 (Copy-on-write, 7장)는 데이터를 제자리에서 덮어쓰지 않는다는 의미입니다. 각 업데이트는 트랜잭션 방식으로 진행되며, 완전히 완료되거나 수행되지 않아 시스템을 업데이트 전 상태로 되돌립니다. 또한 ZFS에는 체크섬이 있어 드라이브가 잘못된 데이터를 반환할 때 이를 감지할 수 있습니다. ZFS는 유효하지 않은 데이터를 감지하면 해당 데이터를 데이터의 추가 복사본, 미러링된 드라이브 또는 RAID-Z 패리티와 같은 다른 소스의 올바른 데이터로 대체합니다. 이러한 기능을 결합하여 ZFS의 자가 복구 속성을 생성합니다.

특수 VDEV (Special VDEVs)

풀은 풀의 성능을 향상시키기 위해 특수 목적의 VDEV를 사용할 수 있습니다. 이러한 특수 VDEV 유형은 데이터를 영구적으로 저장하는 데 사용되지 않고, 대신 더 빠른 장치에 데이터의 추가 사본을 임시로 보관합니다.

별도의 인텐트 로그 (Seperate Intent Log; SLOG, ZIL)

ZFS는 풀의 일부로 ZFS 인텐트 로그 (ZIL)를 유지합니다. 다른 파일시스템의 저널과 마찬가지로 진행 중인 작업을 기록하는 곳으로, 시스템 충돌이나 정전 시 작업을 완료하거나 롤백할 수 있습니다. ZIL은 디스크의 정상적인 작동 조건에 따라 달라집니다. 풀의 사용량이 갑자기 급증하거나 부하와 관련된 지연 시간이 발생하여 성능이 저하될 수 있습니다.

성능을 향상시키는 한 가지 방법은 ZIL을 일반 풀 작업에서 분리하는 것입니다. 풀의 일반 부분을 사용하는 대신 전용 장치를 별도의 인텐트 로그 (SLOG)로 사용할 수 있습니다. 전용 장치는 일반적으로 매우 높은 내구성의 SSD와 같이 작지만 매우 빠른 장치입니다.

ZFS는 데이터를 수신한 SLOG를 순서대로 풀의 메인 스토리지에 복사하는 대신 데이터를 합리적인 그룹으로 일괄 처리하여 보다 효율적으로 쓸 수 있습니다.

특정 소프트웨어는 디스크에 쓰는 데이터가 실제로 디스크에 있는지 확인을 받은 후에 작업을 진행합니다. 데이터베이스는 시스템 충돌이나 정전 시 손상을 방지하기 위해 이 작업을 수행하는 경우가 많습니다. 특정 NFS 작업도 마찬가지입니다. 이러한 요청을 더 빠른 로그 장치에 쓰고 "all done"를 보고함으로써 ZFS는 이러한 작업을 가속화합니다. 데이터베이스는 트랜잭션을 완료하고 계속 진행합니다. 저렴한 디스크를 저장 매체로 사용하면서 거의 SSD 수준의 쓰기 성능을 얻을 수 있습니다.

ZIL을 미러링하여 데이터 손실을 방지할 수 있습니다.

캐시 (Cache: L2ARC)

디스크에서 파일을 읽을 때 시스템은 다른 용도로 메모리가 필요할 때까지 파일을 메모리에 보관합니다. 이것은 초기 BSD 시절에도 사용된 오래된 기술입니다. UFS 기반 BSD 시스템의 상단 (1)을 보면 Buf라고 표시된 메모리 덩어리를 볼 수 있습니다. 이것이 바로 버퍼 캐시입니다.

하지만 전통적인 버퍼 캐시는 수십 년 전에 설계되었습니다. ZFS에는 최신 하드웨어용으로 설계된 적응형 대체 캐시 (ARC)가 있어 속도가 향상됩니다. ARC는 가장 최근에 자주 액세스한 파일을 유지합니다.

하지만 최신 시스템 중 원하는 만큼 캐시를 저장할 수 있는 충분한 RAM을 갖춘 시스템은 거의 없습니다. ZFS가 SLOG를 사용해 쓰기를 가속화하는 것처럼, 매우 빠른 디스크를 사용해 읽기를 가속할 수 있습니다. 이를 레벨 2 ARC 또는 L2ARC라고 합니다.

개체가 캐싱의 이점을 누릴 만큼 자주 사용되지만 RAM에 저장할 만큼 자주 사용되지는 않는 경우 ZFS는 캐시 장치에 개체를 저장할 수 있습니다. L2ARC는 일반적으로 매우 빠르고 내구성이 뛰어난 SSD 또는 NVMe 장치입니다. 이제 해당 데이터 블록이 요청되면 나머지 풀을 구성하는 느린 디스크가 아닌 더 빠른 SSD에서 데이터를 읽을 수 있습니다. ZFS는 백엔드 디스크에서 어떤 데이터가 변경되었는지 알기 때문에 읽기 캐시가 스토리지 풀의 데이터와 동기화되도록 할 수 있습니다.

VDEV가 성능에 미치는 영향 (How VDEVs Affect Performance)

각기 다른 유형의 VDEV는 서로 다른 성능을 발휘합니다. 디스크 성능을 벤치마킹하고 분석하는 것은 매우 복잡한 주제이므로, 누군가 귀찮아서 읽지 않는다면 커다란 교과서 한 권이 필요할 것입니다. 여기서 구체적인 조언을 하면 금방 지루해지므로 몇 가지 일반적인 용어에 대해서만 논의하겠습니다.

일반적인 측정 방법 중 하나는 초당 입출력 횟수, 즉 드라이브가 초당 수행할 수 있는 고유한 작업의 수를 나타내는 초당 입출력 횟수 (IOPS)입니다. 회전식 드라이브의 IOPS는 일반적으로 읽기/쓰기 헤드가 플래터 위에서 얼마나 빨리 이동할 수 있는지에 따라 물리적으로 제한됩니다. 솔리드 스테이트 디스크는 물리적으로 아무것도 움직일 필요가 없기 때문에 성능이 매우 뛰어납니다.

비패리티 스핀들의 수는 손상되지 않은 풀의 스트리밍 읽기 및 쓰기 성능을 제한합니다. "스트리밍" 성능은 드라이브가 읽거나 쓸 수 있는 초당 메가바이트 (MB/s)의 수로 요약됩니다. 드라이브가 데이터를 순차적으로 읽거나 쓸 때, 헤드가 서로 다른 위치를 왔다 갔다 할 필요가 없습니다. 이러한 조건에서 드라이브는 최상의 스트리밍 성능을 달성하여 최고의 처리량을 제공합니다. 스핀들 수는 랜덤 및 스트리밍 성능 모두에 영향을 미칩니다. 스핀들 및 헤드 수가 많을수록 IOPS와 스트리밍 성능이 모두 향상되므로 일반적으로 12개의 1 테라바이트 (12 x 1TB) 드라이브 어레이가 6개의 2테라바이트 (6 x 2TB) 드라이브 어레이보다 성능이 더 뛰어납니다. 헤드 수가 많다는 것은 ZFS가 한 번에 디스크의 더 많은 다른 위치에서 읽거나 쓸 수 있다는 것을 의미하며, 그 결과 IOPS 성능이 향상됩니다. 스핀들이 많다는 것은 더 많은 디스크가 최대한 빠르게 작동하여 데이터를 읽고 쓸 수 있다는 것을 의미합니다. 그러나 드라이브 수가 많을수록 더 큰 셸프 또는 섀시, 더 많은 전력, 더 많은 컨트롤러가 필요합니다.

다른 일반적인 측정 항목으로는 읽기 대역폭, 쓰기 대역폭, 공간 효율성, 스트리밍 성능 등이 있습니다.

일반적으로 미러는 더 나은 IOPS와 읽기 대역폭을 제공할 수 있지만, RAID-Z는 더 나은 쓰기 대역폭과 훨씬 더 나은 공간 효율성을 제공할 수 있습니다.

여러 개의 VDEV가 있는 풀은 모든 VDEV에 걸쳐 데이터를 스트라이핑합니다. 이렇게 하면 성능이 향상되지만, 각 VDEV마다 중복 디스크가 있기 때문에 공간을 많이 차지할 수 있습니다. 여러 개의 VDEV가 있는 풀은 안정성과 내결함성이 향상될 수 있습니다. ZFS의 중복성은 모두 VDEV 수준이지만, 중복 VDEV가 여러 개 있는 풀은 더 많은 디스크 장애를 견딜 수 있습니다. 풀에 VDEV가 많을수록 풀의 성능은 더 좋아집니다.

몇 가지 일반적인 VDEV 구성을 살펴보고 가능한 다양한 배열이 성능과 용량에 어떤 영향을 미치는지 알아보겠습니다. 일반적인 1TB 스피닝 디스크 세트를 사용한다고 가정해 보겠습니다. 각 디스크는 250 IOPS와 100MB/s의 스트리밍 읽기/쓰기가 가능합니다.

디스크 하나 (One Disk)

디스크가 하나뿐인 경우 가능한 구성은 단일 ZFS 스트라이프 VDEV 하나뿐입니다. 이 구성은 가장 기본적인 구성이며 내결함성을 제공하지 않습니다. 디스크 하나가 죽으면 모든 데이터가 사라집니다.

Table 1. Single Disk Virtual Device Configurations

Disks	Config	Read IOPS	Write IOPS	Read MB/s	Write MB/s	Usable Space	Fault Tolerance
1	Stripe	250	250	100	100	1 TB (100%)	none

단일 디스크 스트라이프 장치의 성능 특성은 기본 디스크의 특성과 의심스러울 정도로 비슷합니다. 이상하지 않나요?

두 개의 디스크 (Two Disks)

시스템에 두 개의 디스크가 있는 경우, 두 개의 스트라이프 VDEV 또는 하나의 미러 VDEV로 풀을 구성할 수 있습니다.

스트라이프 VDEV는 사용 가능한 스토리지와 대역폭을 두 배로 늘리지만 장애 위험도 두 배로 증가합니다. ZFS는 각 파일의 블록을 두 개의 디스크에 분산시킵니다. 디스크 중 하나에 장애가 발생하면 모든 데이터를 사용할 수 없게 됩니다.

미러링된 단일 VDEV를 사용하면 각 데이터 블록이 두 디스크에 모두 저장됩니다. 이렇게 하면 두 디스크에서 한 번에 블록을 읽을 수 있으므로 읽기 성능이 향상됩니다. 하지만 가장 작은 디스크의 용량만 사용할 수 있습니다. 쓰기 성능은 가장 느린 디스크의 속도로 제한됩니다. 그러나 하나의 디스크에 장애가 발생해도 풀은 계속 사용할 수 있습니다.

Table 2: Two-Disk Virtual Device Configurations

Disks	Config	Read IOPS	Write IOPS	Read MB/s	Write MB/s	Usable Space	Fault Tolerance
2	2 x Stripe	500	500	200	200	2 TB (100%)	none
2	1 x 2 disk Mirror	500	250	200	100	1 TB (50%)	1

표에서 볼 수 있듯이 미러 풀은 스트라이프 풀의 쓰기 성능은 절반이고 공간은 절반입니다.

디스크 3개 (Three Disks)

디스크 3개는 더 깊은 미러와 RAID-Z를 포함하여 더 많은 옵션을 의미합니다. 스트라이프 디스크 3개로 구성된 풀을 사용할 수도 있지만 실패 확률이 훨씬 높습니다.

더 깊은 미러에는 더 많은 디스크가 있으므로 더 많은 내결함성과 향상된 읽기 성능을 제공합니다. 스핀들과 헤드가 많다는 것은 VDEV가 세 개의 드라이브 중 가장 사용량이 적은 드라이브에서 데이터를 읽을 수 있어 랜덤 읽기를 더 빠르게 제공할 수 있다는 의미입니다. 미러의 쓰기 성능은 여전히 미러에 있는 드라이브 중 가장 느린 드라이브로 제한됩니다.

RAID-Z1은 내결함성을 위해 VDEV의 디스크 중 하나만 필요하므로 공간 효율성이 더 뛰어납니다. 데이터는 모든 드라이브에 분산되어 있으므로 읽기 및 쓰기를 수행하려면 함께 작동해야 합니다. 모든 드라이브에 데이터를 분산하면 스트리밍 쓰기 성능이 향상됩니다. 미러와 달리 RAID-Z에서는 각 드라이브가 동일한 데이터를 쓰는 대신 모든 드라이브가 동시에 자신의 몫의 데이터를 쓸 수 있습니다.

Table 3: Three-Disk Virtual Device Configurations

Disks	Config	Read IOPS	Write IOPS	Read MB/s	Write MB/s	Usable Space	Fault Tolerance
3	1 x 3 disk Mirror	750	250	300	100	1 TB (33%)	2
3	1 x 3 disk RAID-Z1	250	250	200	200	2 TB (66%)	1

여기서 주의할 점은 IOPS가 반드시 실제 읽기/쓰기 성능으로 확장되는 것은 아니라는 점입니다. 미러 VDEV는 각 드라이브의 헤드가 독립적으로 작동할 수 있는 반면, RAID-Z에서는 헤드가 함께 작동해야 하기 때문에 읽기 IOPS가 RAID-Z1의 3배에 달합니다. 초당 메가바이트 단위로 미러는 모든 디스크의 처리량을 데이터에 사용할 수 있다는 장점이 있는 반면, RAID-Z1은 패리티 데이터로 인해 디스크 한 개 분량의 처리량이 손실됩니다. 또한 3 디스크 미러는 모든 디스크에 동일한 데이터를 쓰기 때문에 쓰기 속도가 절반인 MB/s인 반면, RAID-Z1은 모든 디스크에 쓰기를 분산할 수 있지만 패리티로 인해 처리량이 일부 손실됩니다.

디스크 4개 또는 5개 디스크 (Four or Five Disks)

4개 또는 5개의 디스크를 사용하면 더 많은 옵션을 사용할 수 있습니다.

다중 미러 VDEV(기존 RAID 10과 유사)는 데이터베이스와 같은 랜덤 I/O 워크로드에 최상의 성능을 제공합니다. 4개의 디스크를 각각 2개의 디스크가 있는 2개의 미러 VDEV로 나누면 ZFS는 두 미러에 걸쳐 쓰기를 스트라이핑합니다. 한 미러는 데이터의 절반을, 다른 미러는 나머지 절반을 저장합니다. 이렇게 하면 미러의 쓰기 병목 현상을 완화하는 동시에 인상적인 읽기 성능을 제공할 수 있습니다.

디스크가 4개이면 RAID-Z2가 옵션이 됩니다. RAID-Z2의 패리티 디스크 2개는 디스크 2개가 손실되더라도 VDEV가 계속 작동할 수 있음을 의미합니다. 디스크 수가 같은 미러와 비교하면 성능은 떨어지지만, 디스크 두 개가 동시에 장애가 발생해도 더 이상 문제가 되지 않습니다.

5개의 디스크로 RAID-Z3를 구축할 수 있습니다. RAID-Z3 VDEV는 디스크 3개가 모두 손실되더라도 살아남을 수 있습니다. RAID-Z3는 성능과 내결함성을 교환합니다.

물론 RAID-Z1은 여전히 옵션으로 남아 있습니다.

Table 4: Four- or Five-Disk Virtual Device Configurations

Disks	Config	Read IOPS	Write IOPS	Read MB/s	Write MB/s	Usable Space	Fault Tolerance
4	2 x 2 disk Mirror	1000	500	400	200	2 TB (50%)	2 (1/VDEV)
4	1 x 4 disk RAIDZ-Z1	250	250	300	300	3 TB (75%)	1
4	1 x 4 disk RAIDZ-Z2	250	250	200	200	2 TB (50%)	2
5	1 x 5 disk RAIDZ-Z1	250	250	400	400	4 TB (80%)	1
5	1 x 5 disk RAIDZ-Z2	250	250	300	300	3 TB (60%)	2
5	1 x 5 disk RAIDZ-Z3	250	250	200	200	2 TB (40%)	3

RAID-Z1의 스트리밍 (MB/s) 읽기 및 쓰기 성능이 RAID-Z2와 어떻게 비교되는지, RAID-Z3의 성능은 둘 다와 어떻게 비교되는지 참고하세요. 패리티 디스크를 추가하면 해당 디스크의 처리량이 저하될 수 있습니다.

다중 미러 VDEV의 내결함성은 약간 까다롭습니다. 중복성은 풀 단위가 아니라 VDEV 단위라는 점을 기억하세요. 각 미러 VDEV는 여전히 n - 1의 내결함성을 제공합니다. 각 미러 VDEV에서 하나의 드라이브가 계속 작동하는 한, 모든 데이터에 액세스할 수 있습니다. 풀에 두 개의 디스크 미러 VDEV가 있는 경우, 각 VDEV에서 하나의 디스크가 손실되더라도 계속 실행할 수 있습니다. 하지만 동일한 VDEV에서 디스크 두 개가 손실되면 풀이 죽고 모든 데이터가 손실됩니다.

6~12개의 디스크 (Six to Twelve Disks)

디스크 수가 많으면 내결함성, 공간 효율성, 성능의 균형을 맞추는 것이 중요합니다.

6개의 디스크를 3개의 2 디스크 미러 VDEV로 구성하면 상당한 공간과 우수한 쓰기 성능을 확보할 수 있습니다. 한 쌍의 3-디스크 미러 VDEV를 선택하면 공간은 더 적지만 데이터 손실 위험 없이 각 디스크 세트 3개 중 2개에 장애가 발생할 수 있습니다. 또는 RAID-Z VDEV가 될 수도 있습니다.

디스크를 6개 이상 확보하면 한 풀에 여러 개의 RAID-Z VDEV를 사용할 수 있습니다. 12개의 디스크를 단일 VDEV로 함께 운영하여 가용 공간을 최대한 확보하거나, 두 개의 개별 VDEV로 분할하여 가용 공간은 줄어들지만 성능과 내결함성은 향상시킬 수 있습니다.

Table 5: Six- to Twelve-Disk Virtual Device Configurations

Disks	Config	Read IOPS	Write IOPS	Read MB/s	Write MB/s	Usable Space	Fault Tolerance
6	3 x 2 disk Mirror	1500	750	600	300	3 TB (50%)	3 (1/VDEV)

6	2 x 3 disk Mirror	1500	500	600	200	2 TB (33%)	4 (2/VDEV)
6	1 x 6 disk RAIDZ-Z1	250	250	500	500	5 TB (83%)	1
6	1 x 6 disk RAIDZ-Z2	250	250	400	400	4 TB (66%)	2
6	1 x 6 disk RAIDZ-Z3	250	250	300	300	3 TB (50%)	3
12	6 x 2 disk Mirror	3000	1500	1200	600	6 TB (50%)	6 (1/VDEV)
12	4 x 3 disk Mirror	3000	1000	1200	400	4 TB (33%)	8 (2/VDEV)
12	1 x 12 disk RAIDZ-Z1	250	250	1100	1100	11 TB (92%)	1
12	2 x 6 disk RAIDZ-Z1	500	500	1000	1000	10 TB (83%)	2 (1/VDEV)
12	3 x 4 disk RAIDZ-Z1	750	750	900	900	9 TB (75%)	3 (1/VDEV)
12	1 x 12 disk RAIDZ-Z2	250	250	1000	1000	10 TB (83%)	2
12	2 x 6 disk RAIDZ-Z2	500	500	800	800	8 TB (66%)	4 (2/VDEV)
12	1 x 12 disk RAIDZ-Z3	250	250	900	900	9 TB (75%)	3
12	2 x 6 disk RAIDZ-Z3	500	500	600	600	6 TB (50%)	6 (3/VDEV)

풀에서 여러 RAID-Z 장치를 사용하는 것은 풀에서 여러 미러 장치를 사용하는 것과 매우 유사합니다 . 디스크 장애에 대한 허용 오차는 풀이 아닌 VDEV별로 적용됩니다 . 6개의 디스크 RAID-Z2 VDEV 2개로 구성된 12 개의 디스크 어레이는 VDEV당 2개의 디스크만 손실되는 경우 4개의 디스크 손실을 처리할 수 있습니다 .

많은 디스크 (Many Disks)

일반적으로 VDEV당 9~12개 이하의 디스크를 사용하는 것이 좋습니다 . 더 많이 사용할 수도 있지만 ZFS는 그렇게 설계되지 않았습니다 . 36개의 디스크를 배열하여 몇 가지 가능한 배열과 성능에 미치는 영향을 살펴보겠습니다 .

Table 5: Six- to Twelve-Disk Virtual Device Configurations

Disks	Config	Read IOPS	Write IOPS	Read MB/s	Write MB/s	Usable Space	Fault Tolerance
36	18 x 2 disk Mirror	9000	4500	3600	1800	18 TB (50%)	18 (1/VDEV)

36	12 x 3 disk Mirror	9000	3000	3600	1200	12 TB (33%)	24 (2/VDEV)
36	1 x 36 disk RAID-Z2	250	250	3400	3400	34 TB (94%)	2
36	2 x 18 disk RAID-Z2	500	500	3200	3200	32 TB (89%)	4 (2/VDEV)
36	4 x 9 disk RAID-Z2	1000	1000	2800	2800	28 TB (78%)	8 (2/VDEV)
36	6 x 6 disk RAID-Z2	1500	1500	2400	2400	24 TB (66%)	12 (2/VDEV)

더 많은 VDEV를 사용하면 매우 빠른 풀을 만들 수 있습니다. 18개의 디스크 미러 VDEV로 구성된 풀은 그 어떤 것보다 더 빠르게 데이터를 읽을 수 있으며, 장애가 발생하기 전에 18개의 드라이브를 잃을 수도 있습니다! 물론 18개의 드라이브가 정확히 일치해야 하지만, 전원 공급 장치가 다른 두 개의 디스크 셸이 있다면 충분히 가능합니다. 반면에 해당 풀에 있는 디스크 두 개가 잘못되면 전체 풀이 죽습니다.

각 VDEV에 패리티 또는 미러를 추가하면 안정성이 향상됩니다. VDEV의 수가 많을수록 성능이 향상됩니다. 여러분의 역할은 이 두 가지 특성을 적절히 조합하여 환경을 지원하는 것입니다.

각 VDEV는 가장 느린 디스크의 랜덤 읽기/쓰기 성능으로 제한되므로 하나의 VDEV에 디스크가 너무 많으면 공간 효율성을 조금만 높이기 위해 성능을 포기하는 것입니다. 성능을 개선하기 위해 L2ARC 및 SLOG 장치를 추가할 수는 있지만, 이러한 문제를 완전히 피하는 것이 가장 좋습니다.

그렇다면 VDEV가 많으면 많을수록 좋은데, 왜 6 x 6 디스크 RAID-Z2 풀이 1 x 36 디스크 RAID-Z2 풀에 비해 읽기 및 쓰기 속도가 훨씬 느린 것일까요? 답은 내결함성 열에 있습니다. RAID-Z2 VDEV가 많을수록 중복성이 높아져 더 많은 장애를 견딜 수 있습니다. 디스크가 내결함성을 제공하는 경우, 디스크는 데이터의 여분의 복사본을 저장하므로 디스크에 장애가 발생했을 때 손실된 복사본을 대체할 수 있습니다. 시스템은 데이터가 변경될 때마다 패리티 데이터를 다시 계산하여 저장합니다. 패리티 데이터는 원본 사본이 없는 경우를 제외하고는 파일을 읽을 때 사용되지 않습니다. 패리티에 사용된 디스크는 더 이상 스트리밍 성능에 기여하지 않습니다. 디스크를 더 추가하여 해당 성능을 복원할 수 있습니다. 6 x 8 디스크 RAID-Z2 풀은 36개의 데이터 디스크와 12개의 패리티 디스크로 구성되며, 1 x 36 디스크 RAID-Z2 풀보다 뛰어난 성능을 발휘할 수 있습니다.

VDEV에 대해 알고 있는 내용을 바탕으로 실제 풀을 만들어 보겠습니다.

Revision #1
Created 21 January 2024 04:45:25 by 고기만두 (MeatDumpling)
Updated 21 January 2024 04:45:36 by 고기만두 (MeatDumpling)