PURESTORAGE®

splunk> turn data into doing™

기술 백서

엔터프라이즈 규모에서의 Splunk 가속화

퓨어스토리지의 올플래시 스토리지 시스템을 통해 Splunk를 가속화하기 위한 프레임워크

목차

소개

올플래시 어레이 업계를 선도하는 퓨어스토리지는 수많은 조직들이 스토리지 관리의 복잡성을 감소시켜 IT 분야에서 민첩성과 효율성을 향상하는 데 일조했습니다. Splunk® Enterprise는 시스템 데이터를 위한 업계 선도적인 플랫폼으로 기업이 복잡한 데이터 관리와 네트워크 보안 문제를 해결할 수 있도록 지원합니다.

IBM에 따르면, 데이터 침해를 탐지하는 데 평균 280일이 소요되며, 평균 840만 달러의 손해가 발생합니다. 일반 개인정보 보호법(GDPR) 제33조에 보안 침해 사실을 알게 된 후 72시간 이내에 감독기관에 통지해야 한다고 규정되어 있는 사실을 감안하면, 보안 애널리스트들이 왜 직무를 수행하는 데 어려움을 겪는지 이해할 수 있습니다. 또한 데이터가 기하급수적으로 증가함에 따라 보안 팀은 더 적은 리소스로 더 많은 데이터를 빠르고 쉽게 분석할 수 있는 방법이 필요해졌습니다. 직접 연결 스토리지(DAS)를 사용하는 기존의 Splunk Enterprise 인덱서 클러스터 구현 모델은 높은 데이터 가용성과 충실도를 제공하지만 데이터 볼륨이 증가하면 심각한 문제에 부딪히게 됩니다.

Splunk 고객은 일반적으로 다음과 같은 문제로 어려움을 겪고 있습니다.

- 과거 데이터를 검색할 때 쿼리 성능이 일관되지 않아 사용자 생산성에 영향을 미칩니다.
- 스토리지 요구 사항이 증가함에 따라 많은 수의 인덱서를 관리하는 것이 더욱 복잡해졌습니다.
- 애플리케이션을 최신 상태로 유지하는 데 따른 관리 오버헤드가 높아 총소유비용(TCO)이 높고 비즈니스 가치는 낮습니다.

클라우드 옵션은 초기에는 인프라 아웃소싱과 무한 확장이라는 매력적인 가치 제안을 제공하지만, 시간이 지나면서 제어 역량이 부족해지고 비용도 통제되지 않는 경우가 많습니다.

목표

이 백서에서는 플래시어레이(FlashArray™)와 플래시블레이드(FlashBlade®)를 사용하는 퓨어스토리지의 데이터 중심 아키텍처에 Splunk Enterprise를 구현할 때 얻는 이점과 모범 사례를 설명합니다. 퓨어스토리지의 올플래시 어레이에서 Splunk Enterprise를 테스트한 결과를 살펴봅니다. 또한 퓨어스토리지 제품에 Splunk가 구현된 모범 사례를 살펴봅니다.

대상 독자

이 문서는 퓨어스토리지의 올플래시 스토리지 플랫폼에 Splunk Enterprise를 설계 및 구현하고자 하는 시스템 관리자, 스토리지 관리자, IT 관리자, 시스템 설계자, 영업 엔지니어, 현장 컨설턴트, 전문 서비스 및 파트너를 대상으로 합니다. Splunk, Linux, 서버, 스토리지 및 네트워크에 대한 실무 지식이 있다고 가정하지만 이 문서를 읽는 데 반드시 필요하지는 않습니다.

주요 혜택 요약

이 백서에서는 퓨어스토리지의 올플래시 스토리지 시스템(플래시어레이 및 플래시블레이드)에서 Splunk Enterprise가 제공하는 혜택에 대해 중점적으로 설명합니다. 이 공동 솔루션에서 플래시어레이는 Splunk 핫/웜 티어용 통합 블록 스토리지로 사용되고, 플래시블레이드는 Splunk 콜드 티어용의 초고속 파일 및 오브젝트 스토리지로 사용됩니다. 이 백서에는 통합 솔루션의 이점을 보여주는 실질적인 사례와 역량, 구현 모범 사례를 소개합니다. 이 백서는 온-프레미스 퓨어스토리지에서 Splunk를 실행하는 것이 어떻게 매력적인 장기적 가치를 제공하는지 다음의 다양한 이점과 함께 보여줍니다.

Splunk 가속화: 보다 신속한 인사이트 확보

Splunk 사용자가 비즈니스 가치를 창출하고, 보안 침해를 방지하며, IT 시스템의 확장성을 유지하려면 일관된 검색과 인덱서 성능이 필수적입니다.

- 사용자는 기존 아키텍처나 SmartStore 아키텍처에 있는 웜 원격 스토리지의 콜드 티어에서 중요한 데이터를 검색하다 답답해서 화가 날 수도 있습니다. 이는 Splunk SmartStore 시나리오에서 콜드 스토리지를 직접 쿼리하는 동안 IO 성능이 저하되거나, 과도한 캐시 제거와 재구축이 원인일 수 있습니다.
- 플래시어레이 및 플래시블레이드 솔루션에 구현된 Splunk Enterprise는 핫, 웜 및 콜드 데이터용의 올플래시 스토리지 시스템을 통해 티어 간에 일관된 검색 경험을 제공하는 동시에, 통합 스토리지 솔루션의 비용상 이점을 제공합니다.
- 핫 티어, 웜 티어 및 콜드 티어에서 120회의 동시 희소 및 희귀 검색을 테스트했습니다. 모든 데이터 티어를 검색하는 경우, 핫/웜 데이터와 콜드 티어의 검색 간에 평균 몇 초 이내의 시간 차이가 발생했습니다. 콜드 데이터를 검색하는 속도와 핫 데이터와 웜 데이터를 검색하는 속도는 거의 비슷하다 하겠습니다.

최적화: 인프라 통합 및 효율적인 확장

직접 연결 스토리지(DAS)를 사용하는 기존 Splunk 구현은 용량이 더 필요할 때마다 인덱서를 추가해야 하므로 복잡성이 급격히 증가합니다. 더 많은 인덱서를 사용할수록 패치 적용, 리밸런싱, 업그레이드 및 오류가 더 많아진다는 것을 의미하기 때문입니다.

- 핫/웜 데이터용 DAS를 대체하는 퓨어스토리지의 올플래시 블록 스토리지로 이제 인덱서 전반에서 스토리지를 통합할 수 있습니다. 이 솔루션을 사용하면 컴퓨팅과는 별도로 스토리지를 페타바이트 용량까지 확장할 수 있기 때문에 필요한 인덱서 수와 복잡성이 줄어듭니다. 콜드 데이터를 위해 퓨어스토리지의 다차원적인 스케일 아웃 스토리지인 플래시블레이드를 결합하면 중단이나 복잡성 없이 티어 전반에 걸쳐 용량을 확장할 수 있습니다.
- Splunk는 Splunk Enterprise에 인덱서당 일일 300GB의 인제스트 속도를 권장합니다. 테스트 결과, CPU 사용량 25%를 초과하지 않고, 인덱서 당 일일 최대 2.45TB(서버당 40개 코어)를 인제스트할 수 있었습니다. (참고: 결과는 서버, 데이터 유형 및 검색 볼륨에 따라 달라질 수 있습니다.)
- Kafka, Spark, Vertica 같은 분석 애플리케이션을 비롯해 전체 데이터 파이프라인 전반에서 스토리지를 통합합니다. 씬 프로비저닝 스토리지는 서버와 플랫폼 간에 스토리지를 공유하여 스토리지 요구 사항을 줄이고 통합할 수 있도록 합니다.
- Splunk 압축 외에도 추가적인 데이터 절감 효과를 얻을 수 있습니다. 퓨어스토리지 솔루션은 최소한 1:35:1의 추가적인 데이터 절감 효과를 제공합니다. (참고: 결과는 데이터의 카디널리티(cardinality)에 따라 달라질 수 있습니다.)

간소화: Splunk 관리자의 오버헤드 감소

Splunk 관리자는 새로운 데이터 소스 도입, 클리닝, 포맷팅 및 강화, Splunk User 커뮤니티를 위한 새로운 기능 구축 등 가치를 창출 하는 활동에 집중할 필요가 있습니다. 운영 작업에 시간을 소비하다 보면 가치 창출이 어려워집니다. 퓨어스토리지 솔루션과 함께 하면 관리자는 다음과 같은 혜택을 누릴 수 있습니다:

- 인덱서를 추가, 업그레이드 또는 재시작하는 동안이나 그 이후에도 데이터 리밸런싱에 걸리는 시간을 대폭 단축할 수 있습니다. 테스트에서 퓨어스토리지는 스토리지를 추가하는 데 걸리는 시간을 11시간 45분에서 단 2초로 단축했으며 용량 업그레이드 동안 검색 또는 인덱서 성능에 영향을 미치지 않았습니다.
- 내장된 RAID, 이레이저 인코딩 및 상시 가동 암호화를 통해, 내장된 데이터 보호 기능을 확보하고 관리 오버헤드를 줄일 수 있습니다.
- 서비스형 퓨어(Pure as-a-Service™)를 사용하면 스토리지 규모를 미리 산정할 필요가 없으며 사용한 만큼만 비용을 지불하면 됩니다. DAS 아키텍처에 비해 전력 소비가 적어 Splunk를 통해 지구환경 개선에 기여할 수 있습니다.
- 퓨어스토리지 에버그린 스토리지(Evergreen Storage™) 구독 프로그램을 활용하면 스토리지 업그레이드에 대해 걱정할 필요가 없습니다.

기술 개요

플래시어레이//X (FlashArray//X)

세계 최초의 올플래시 엔드-투-엔드 NVMe 및 NVMe-oF 어레이인 플래시어레이//X가 이제 가장 까다로운 엔터프라이즈 애플리케이션의 성능 요구 사항을 해결할 수 있도록 스토리지급 메모리 부스트를 옵션으로 제공합니다. 플래시어레이//X는 혁신적인 성능, 간소화 및 통합 기능을 제공합니다. 부서별 스토리지에서 대규모 엔터프라이즈 공유 스토리지는 물론 고성능 미션 크리티컬 애플리케이션까지, 모든 것을 지원하도록 설계되었습니다. 온-프레미스와 퍼블릭 클라우드에 쉽게 연결되어, 엔터프라이즈 및 클라우드 네이티브 웹 스케일 애플리케이션의 성능 결과와 유연성을 극대화해줍니다. 퓨어스토리지의 에버그린 모델은 성능, 용량 및 기능들이 중단 없이 지속적으로 향상된다는 것을 의미합니다.

- **미션 크리티컬 애플리케이션의 가속화:** 레이턴시가 150µs인 플래시어레이//X의 NVMe 아키텍처(플러그 앤 플레이 스토리지급 메모리 포함)는 미션 크리티컬 비즈니스 애플리케이션과 데이터베이스에 새로운 수준의 성능과 매우 낮은 레이턴시를 제공합니다. 더 빠른 트랜잭션과 의사결정이 가능해지고 더 몰입감 있는 고객 경험을 제공할 수 있게 됩니다.
- **클라우드의 하이퍼 통합:** NVMe는 프라이빗 클라우드에서 티어 1, 혼합 워크로드 통합에 필요한 탁월한 성능 집적도를 지원합니다. 플래시어레이//X는 현재 집적도가 매우 높은 18.3TB 다이렉트플래시(DirectFlash®) 모듈을 제공하며, 퓨리티의 상시 가동 QoS를 통해 대역폭이나 I/O 경합에 대한 걱정 없이 매우 다양한 애플리케이션들을 통합할 수 있습니다.
- **현재 및 미래 애플리케이션 통합:** 조직들은 전통적인 비즈니스 애플리케이션을 새롭고 현대적인 웹 스케일 애플리케이션과 함께 실행하는 방향으로 진화해 왔습니다. 이전에는 이러한 혼합 환경을 위해 완전히 다른 아키텍처가 필요했습니다. 플래시어레이//X, 엔드-투-엔드 NVMe와 NVMe-oF를 사용하면, 모든 것을 단일 공유 아키텍처에서 실행하고 SAN(Storage Area Networks)과 DAS를 통합할 수 있습니다. 이를 통해 DAS의 성능을 제공하는 동시에 현대적인 공유 스토리지의 효율성, 안정성 및 간소함을 실현할 수 있습니다.
- **간소화된 클라우드 기반 관리:** 퓨어1(Pure1)은 간소화된 클라우드 기반 관리와 전체 스택 분석, 퓨어1 메타(Meta®)의 AI 기반 역량을 통해 간편한 예측을 지원합니다. 퓨어1은 플래시어레이, 플래시블레이드, NFS 또는 Amazon S3 같은 퍼블릭 클라우드 등 대상에 관계없이 모든 백업의 스냅샷 카탈로그를 한 곳에 제공합니다.
- **에버그린 스토리지:** 플래시어레이는 SaaS 및 클라우드처럼 작동합니다. 일단 한 번 구현하면 구독 프로그램을 통해 성능, 용량, 집적도 및/또는 기능을 확장 및 개선하여 10년 이상 지속적인 혁신의 혜택을 누릴 수 있습니다. 이 모든 것이 다운타임, 성능 영향 또는 데이터 마이그레이션

없이 가능합니다. 퓨어스토리지는 플래시어레이의 모듈식 스테이트리스 아키텍처를 통해 향후 기술에 대한 호환성을 제품에 직접 설계해 넣었습니다. 적정 규모 보장(Right Size Guarantee™) 프로그램은 필요한 유효 용량을 확보할 수 있도록 합니다. 용량 통합(Capacity Consolidation) 프로그램은 확장과 더불어 스토리지를 집적도가 높은 최신 상태로 유지해줍니다. 에버그린 스토리지가 있으면 이미 보유하고 있는 테라바이트를 다시 구입할 필요가 없습니다. 스토리지를 언제나 최신 상태로 높은 집적도를 유지하며 항상 비즈니스 요구 사항을 충족할 수 있습니다. 퓨어스토리지는 서비스형 퓨어(Pure-as-a-service) 포트폴리오를 통해 제품(CAPEX) 또는 서비스(OPEX)로 모든 핵심 솔루션을 제공합니다.

플래시블레이드(FlashBlade)

퓨어스토리지는 데이터 기반 비즈니스의 스토리지 요구 사항을 충족하기 위해 플래시블레이드 아키텍처를 개발했습니다. 플래시블레이드는 비정형 데이터의 저장 및 처리 위주로 최적화된 올플래시 시스템입니다. 플래시블레이드 시스템은 수천 개의 클라이언트를 위해 다중 파일 시스템과 멀티 테넌트 오브젝트 저장소를 동시에 호스팅할 수 있습니다.

플래시블레이드 시스템의 성능 및 용량 확장 기능은 다음과 같은 5가지 주요 혁신 기술을 기반으로 합니다.

- **고성능 스토리지 장치:** 플래시블레이드는 데이터를 스토리지 장치에 저장하고, 기존의 회전식 디스크나 솔리드 스테이트 드라이브 같이 레이턴시가 높은 스토리지 미디어를 제거함으로써 올플래시 아키텍처의 이점을 극대화합니다. 확장 가능한 NVRAM이 각 스토리지 유닛에 통합되어, 새 블레이드를 시스템에 추가하면 비례적으로 성능과 용량이 확장됩니다.
- **통합 네트워크:** 플래시블레이드 시스템은 클라이언트와 내부 관리 호스트 간의 높은 통신 트래픽을 이더넷 링크를 통해 최대 100GB/s의 IPv4 및 IPv6 클라이언트 액세스를 모두 지원하는 신뢰할 수 있는 단일 고성능 네트워크로 통합합니다.
- **퓨리티//FB(Purity//FB) 스토리지 운영 체제:** 플래시블레이드의 패브릭 모듈에서 대칭적으로 운영 체제를 실행하는 퓨리티//FB는 플래시블레이드의 블레이드 간에 모든 클라이언트 작업 요청을 균등하게 분산시켜 워크로드 리밸런싱 문제를 최소화합니다.
- **파일 및 오브젝트를 위한 공통된 미디어 아키텍처 설계:** 플래시블레이드의 단일 기저 미디어 아키텍처는 전체 플래시블레이드 구성에서 NFSv3, NFS over HTTP, SMB(Samba 수준 기능 사용) 등의 다양한 프로토콜과 Amazon S3를 통해 각각 파일과 오브젝트에 대한 동시 액세스를 지원합니다.
- **간소화된 사용:** 플래시블레이드의 퓨리티//FB는 일상적인 관리 작업을 자율 수행함으로써 스토리지 운영을 간소화하여 시스템 관리 부담을 줄여줍니다. 견고한 운영 체제를 갖춘 플래시블레이드는 구성 요소에 장애가 발생할 경우 스스로 튜닝을 수행하고 시스템 알림을 제공합니다.

완전한 플래시블레이드 시스템 구성은 2개의 외부 패브릭 모듈(XFM)로의 고속 링크를 통해 상호 연결된 최대 5개의 독립형 랙 마운트 섀시로 구성됩니다. 각 섀시의 후면에는 고속 이더넷을 통한 TCP/IP를 사용해 블레이드, 다른 섀시 및 클라이언트를 상호 연결하기 위한 2개의 온보드 패브릭 모듈이 있습니다. 두 패브릭 모듈은 서로 연결되어 있으며, 각각 컨트롤 프로세서와 이더넷 스위치 ASIC를 포함합니다. 안정성을 위해 각 섀시에는 예비 전원 공급 장치와 냉각 팬이 장착되어 있습니다.

각 섀시의 전면에는 데이터 연산과 저장을 위해 최대 15개의 블레이드가 장착됩니다. 각 블레이드 어셈블리는 프로세서, 통신 인터페이스 및 영구 데이터 저장을 위한 17TB 또는 52TB의 플래시 메모리를 갖춘 독립형 컴퓨팅 모듈입니다.

현재 플래시블레이드 시스템은 초당 150만여 개의 NFSv3 getattrs를 지원할 수 있습니다. 또는 15개의 블레이드가 포함된 단일 4U 섀시에서 3:1로 압축 가능한 데이터 세트에 대한 512KiB 읽기를 최소 17GiB/초에, 512KiB 덮어쓰기를 최소 8GiB/초에 수행할 수 있으며, 75개의 블레이드가 있는 최대 5 x 4U 섀시까지 컴퓨팅과 성능을 확장할 수 있습니다.

Splunk Enterprise

Splunk Enterprise는 기술 인프라, 보안 시스템 및 비즈니스 애플리케이션에서 생성되는 빅데이터를 간편하게 수집 및 분석하고 가치를 실현할 수 있게 해주며, 운영 성과와 비즈니스 결과를 향상할 수 있는 인사이트를 제공합니다.

Splunk Enterprise는 모든 소스에서 시스템 데이터를 모니터링 및 분석해 운영 인텔리전스를 제공함으로써 IT, 보안 및 비즈니스 성과를 최적화합니다. 직관적인 분석 기능, 머신러닝(ML), 패키지 애플리케이션 및 개방형 API를 갖춘 Splunk Enterprise는 집중화된 사용 사례에서 전사적 분석 백본까지 확장 가능한 유연한 플랫폼입니다. Splunk Enterprise는 시스템 데이터에 대한 포괄적인 실시간 솔루션으로, 다음과 같은 핵심 기능을 제공합니다.

- 거의 모든 소스에서 시스템 데이터를 포괄적으로 수집 및 인덱싱
- 실시간 데이터와 과거 데이터를 검색 및 분석하는 강력한 검색 처리 언어(SPL)
- 앱으로 보안, IT 운영, 비즈니스 분석 등을 위한 솔루션 제공
- 패턴 및 임계값에 대한 실시간 모니터링; 특정 조건 발생 시 실시간 알림 제공
- 강력한 보고 및 분석
- 다양한 역할별로 사용자 정의 대시보드 및 뷰 지원
- 복원성 및 수평적 확장성
- 세분화된 역할 기반 보안 및 액세스 제어
- 온-프레미스 또는 클라우드에서 멀티 테넌시 및 유연한 분산 구현 지원
- 빅데이터 애플리케이션을 위한 강력하고 유연한 플랫폼

CentOS Linux

Red Hat Enterprise Linux (RHEL) 소스에서 파생된 Linux 배포판인 CentOS 버전 7.5는 Linux 컨테이너 지원을 비롯한 엔터프라이즈급 컴퓨팅 플랫폼 기능을 무료로 제공합니다. Splunk는 Linux, Windows 및 MacOS에서 Splunk Enterprise를 지원합니다. Splunk Enterprise용 Linux 3.x 및 4.x 커널 버전을 지원하며 호스트에서 실행하는 Linux 버전에 따라 RPM 또는 DEB 패키지 또는 tar 파일을 제공합니다. 대부분의 Splunk 구현은 Linux 운영 체제를 기반으로 하기 때문에, 테스트에서는 CentOS Linux를 사용하기로 했습니다.

기업이 직면하는 도전과제

연결된 스토리지의 확장으로 인한 복잡성 증가

Splunk Enterprise 인덱서 클러스터 구현 모델(분산 스케일 아웃 모델)은 데이터 가용성과 충실도가 높지만 상당한 비용이 듭니다. 이 구현에서는 서로의 데이터를 복제하도록 Splunk Enterprise 인덱서를 구성하여 데이터 손실을 방지하고 노드 장애 시 검색을 용이하게 했습니다. 이러한 모델은 서버와 스토리지 간의 근접성에 의존해 고성능을 실현하는 Hadoop 같은 이전의 빅데이터 기술에 적합합니다.

분산 스케일 아웃 모델에서는 모든 Splunk 인덱서가 주로 DAS를 통해 구성되며 핫/웜 및 콜드 티어용 스토리지의 사이즈가 비슷합니다. 과거에는 이 모델이 필요한 데이터 볼륨을 효과적으로 처리했습니다. 그러나 데이터가 증가할 때, 스토리지와 컴퓨팅 요구 사항은 선형적으로 확장되지 않습니다. 스토리지 요구 사항을 해결하기 위해 컴퓨팅과 스토리지에 같은 유형의 노드를 추가하는 것은 차선책에 불과하며 비용도 많이 듭니다.

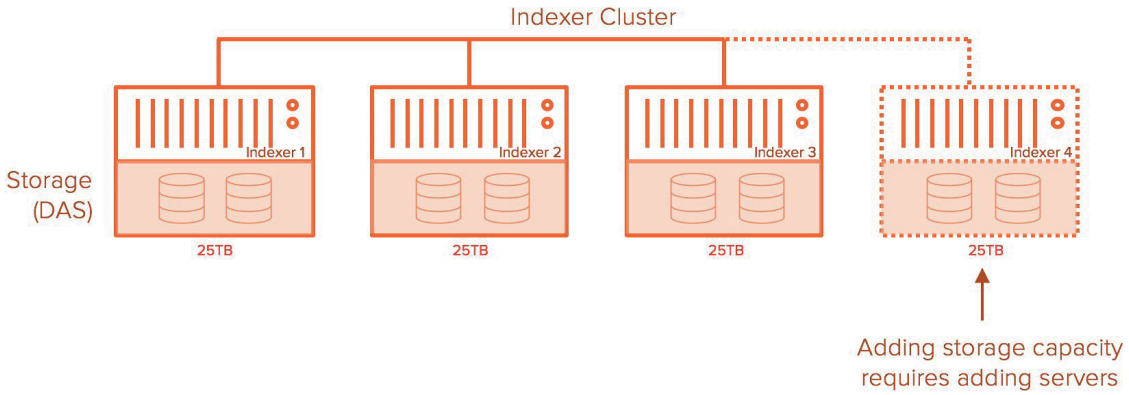


그림 1: Splunk 인덱서 구성.

기대에 못 미치거나 일관성 없는 검색 성능

분산 스케일 아웃 모델의 핵심은 복제 계수를 기반으로 추가 노드에 있는 데이터의 복사본 또는 복제본입니다. 복제 계수(RF)가 2이고 검색 계수(SF)가 2인 인덱서 클러스터는 원시 데이터와 인덱스 데이터를 인제스트된 인덱서 노드 위에 있는 추가 인덱서 노드에 복제해야 합니다. 따라서 RF 및 SF를 기반으로 하는 인덱서 클러스터 환경에서는 스토리지 요구 사항이 크게 급증합니다. Splunk는 TSIDX 감소 기능을 제공하여 저장 공간을 절약해주면서도 검색 성능을 저하시키지 않습니다.

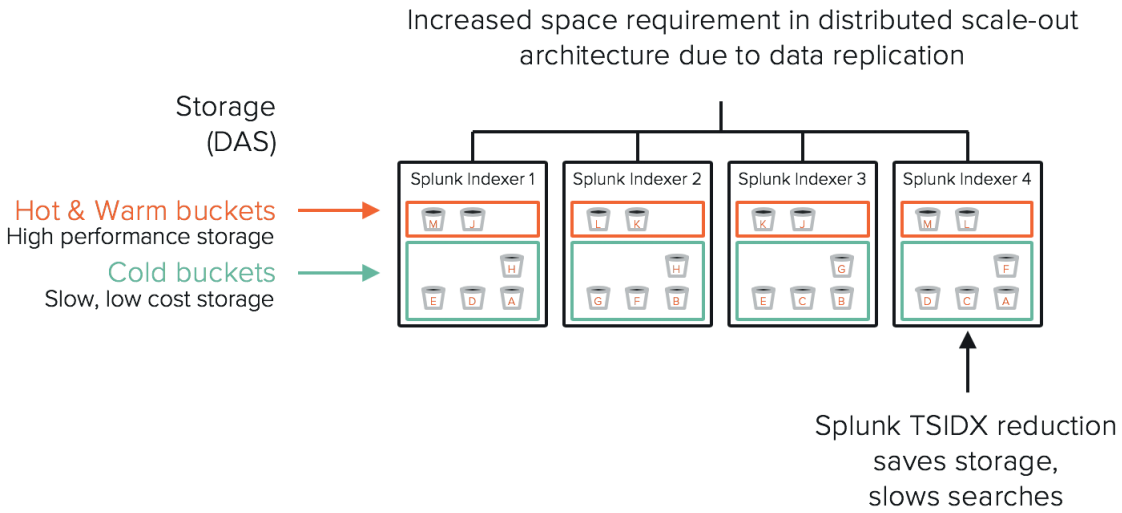


그림 2: 분산 스케일 아웃 아키텍처의 인덱서 공간 요구 사항

데이터의 기하급수적인 증가와 더불어 대규모 데이터를 장기간 보관하기 위한 운영 및/또는 규정 준수 요건은 스토리지와 성능의 측면에서 상당한 문제를 야기합니다. 데이터 급증에 따른 스토리지 요구 사항을 충족하기 위해, 최근에 인제스트된 데이터나 사용 빈도가 높은 핫 데이터는 SSD 드라이브나 PCIe 플래시 카드 같은 보다 빠른 매체에서 호스팅하고, 오래된 데이터나 사용 빈도가 낮은 콜드 데이터는 로우 엔드 HDD 드라이브에 저장하는 계층화된 방식을 사용했습니다. 이러한 방식은 오래된 데이터나 콜드 데이터를 더 저렴한 디스크 스토리지에 저장하여 비용을 절감할 수는 있었지만 검색 속도가 느리다는 단점이 있었습니다.

데이터 분석의 등장으로, 기업들은 최근 데이터에만 의존하는 것이 아니라, 모든 데이터에서 의미 있는 인사이트를 얻기 위해 노력을 기울이고 있습니다. Splunk에서 이것은 검색이 이제 핫/웜 티어에 존재하는 최근 데이터에만 국한되지 않고 콜드 티어까지 확장된다는 것을 의미합니다. 빅데이터는 분석 속도만큼만 유용하며, 분석 속도가 빨라지면 데이터에 더 빠르게 액세스할 수 있어야 합니다. 따라서 Splunk에서 콜드 티어 데이터에 더 빠르게 액세스하려면, 비용이 낮고 깊은 기존 디스크 스토리지는 적합하지 않으며, 올플래시 스토리지 같이 더 빠르고 성능이 더 좋은 스토리지가 필요합니다.

솔루션

이 문서에서 말하는 솔루션은 핫/웜 티어에 플래시어레이를, 콜드 티어에 플래시블레이드를 사용하는 Splunk Enterprise 솔루션입니다.

- 플래시어레이//X는 세계 최초의 엔터프라이즈급 올 NVMe 및 NVMe 플래시 스토리지 어레이입니다. 가트너가 공유 가속 스토리지(shared accelerated storage)라 새롭게 명명한 새로운 카테고리의 스토리지로, 획기적인 성능, 간소함 및 통합을 제공합니다.
- 플래시블레이드는 퓨어스토리지가 선보인 혁신적인 스케일 아웃 올플래시 파일 및 오브젝트 스토리지 시스템으로, Splunk 같은 애플리케이션을 사용해 전체 데이터 사일로를 통합하는 동시에 시스템 데이터에서 실시간 인사이트를 빠르게 확보할 수 있도록 설계되었습니다.

엔터프라이즈 스토리지

퓨어스토리지는 직접 연결 스토리지(DAS)로 야기되는 다양한 비효율성을 극복하기 위해 애플리케이션과 관계없이 서버에서 스토리지를 분리할 것을 권장합니다. 이러한 분리는 컴퓨팅과 스토리지를 효율적으로 활용하는 동시에, 비즈니스 요구에 맞게 컴퓨팅과 스토리지를 독립적으로 확장할 수 있도록 합니다. 즉, 스토리지 쉘프를 플래시어레이에 추가하거나, 플래시블레이드나 플래시블레이드 섀시에 블레이드를 추가하면 필요에 따라 스토리지 공간을 독립적으로 추가할 수 있습니다 기존 DAS 모델에서처럼 인덱서 노드를 더 추가할 필요가 없습니다. 이를 통해 인덱서 노드가 감소하여 많은 수의 노드를 관리하는 데 따른 총소유비용(TCO)과 복잡성을 줄일 수 있습니다.

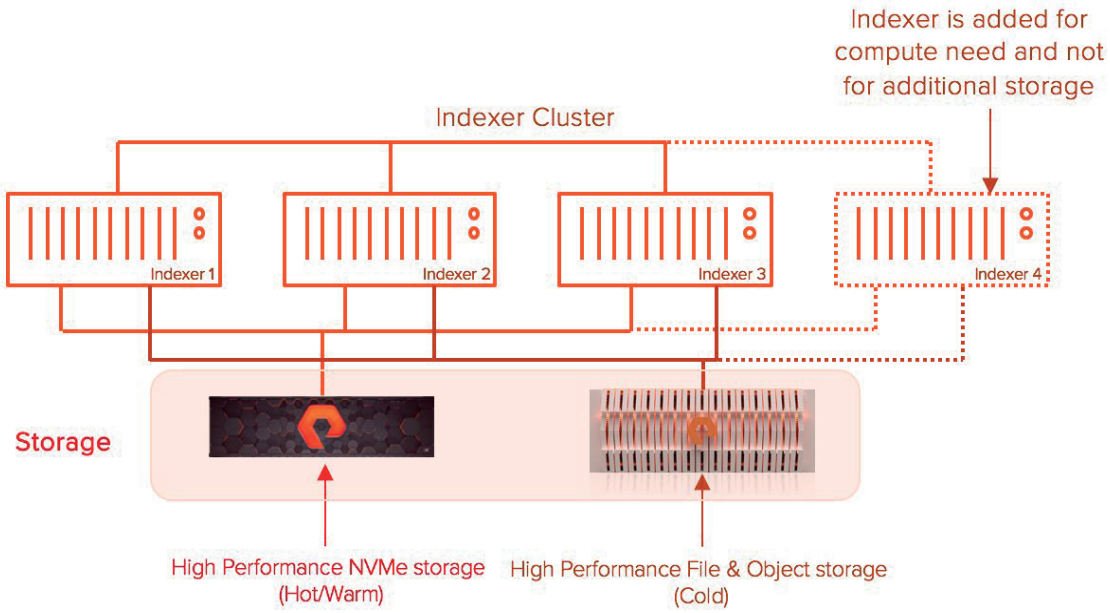


그림 3: 스토리지 분리.

확장 가능한 올플래시 스토리지 솔루션

플래시어레이와 플래시블레이드는 모두 압축 등의 데이터 서비스를 제공하는 씬 프로비저닝 스토리지 시스템으로, Splunk Enterprise의 인덱서 클러스터 구현을 통해 스토리지 요구 사항을 줄여줍니다. 시계열 인덱스 파일(tsidx)은 기본적으로 ASCII 형식이므로 스토리지 레벨에서 압축된다는 이점이 있습니다.

플래시어레이는 압축된 원시 데이터의 중복을 제거해 스토리지에 단일 복사본으로 저장하는 중복 제거 서비스를 통해 스토리지 요구 사항을 더욱 줄일 수 있습니다. 단일 복사본으로 데이터의 중복을 제거하는 것이 애플리케이션의 단일 장애 지점이 된다는 통상적인 생각과 달리, 플래시어레이는 RAID-HA, 이중 패리티 보호 기반의 데이터 보호 메커니즘, 그리고 단일 장애 지점을 제거하는 이중 컨트롤러 아키텍처를 사용합니다.

기존의 직접 연결 스토리지는 데이터 서비스를 제공하지 않습니다. 이는 구현하는 데 오버헤드일 뿐 아니라 호스트의 추가 CPU 주기가 필요하기 때문입니다. 이는 인덱서 노드에서 인제스트 및 검색 등의 Splunk 작업을 지원하는 데 사용할 수 있는 CPU 리소스가 더 적다는 의미입니다.

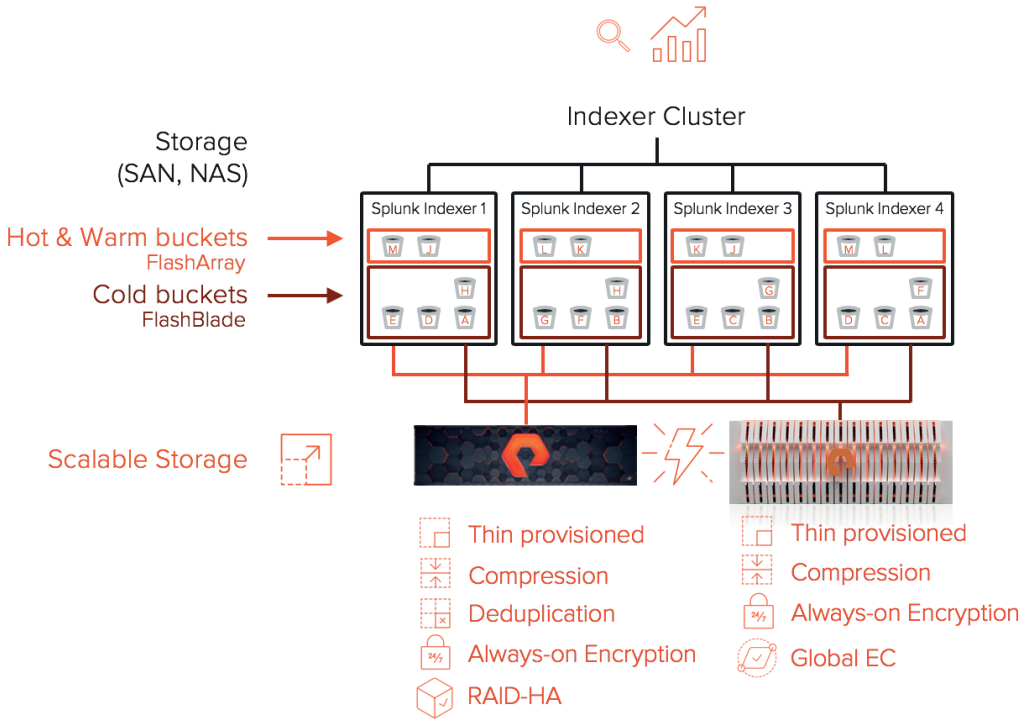


그림 4: 핫, 콜드 및 웜 버킷을 포함하는 Splunk 인덱스 클러스터

플래시어레이와 플래시블레이드 같은 엔터프라이즈 스토리지의 또 다른 이점은 소프트웨어 RAID 및 이레이저 인코딩을 통한 데이터 보호 기능이 내장되어 있어, 시스템 관리자가 RAID 유형을 수동으로 결정하고 추가된 스토리지에 대해 인덱서를 추가할 때마다 이를 생성해야 하는 운영 오버헤드를 줄일 수 있다는 점입니다. 데이터 보호 기능 이외에도 플래시어레이와 플래시블레이드는 모두 데이터를 암호화된 상태로 유지하는 상시 암호화 서비스를 제공합니다.

플래시어레이와 플래시블레이드의 올플래시 기술은 Splunk 기능, 데이터 인제스트 및 검색 성능이 DAS의 기술과 비교해 더 우수합니다. 플래시블레이드의 기술 덕분에, 콜드 티어의 성능 특성은 이제 핫/웜 티어와 일치합니다. 고객은 핫/웜 및 콜드 티어에서 사용 가능한 모든 데이터를 검색하고 유사한 성능을 얻을 수 있으므로, 계층화된 성능을 제거하고 '분석에서는 콜드 데이터란 없다'는 점을 실현할 수 있습니다.

퓨어스토리지 시스템 기반 Splunk Enterprise 솔루션은 다음과 같은 혜택을 제공합니다.

- 확장성이 뛰어난 스토리지 솔루션 제공
- 모든 티어에서 올플래시 성능 제공
- 컴퓨팅 및 스토리지를 독립적으로 추가하여 동적인 클러스터 확장 지원
- 압축을 통해 공간 사용량 추가 감소
- 암호화를 통해 저장된 Splunk 데이터 보호
- 인덱서 서버를 줄이는 동시에 동일하거나 더 높은 용량 요구 사항 지원

참조 아키텍처 설계

설계 토폴로지

이 섹션에서는 랩에서 테스트한 퓨어스토리지 시스템의 Splunk Enterprise 설계 토폴로지에 대해 설명합니다.

이 솔루션에는 Splunk Enterprise 구성을 호스팅하기 위해 20대의 인텔 CPU 기반 Cisco UCS B-시리즈 M4 및 M5 블레이드 서버가 포함된 3대의 섀시가 포함되었습니다. Splunk는 인덱서 8개, 검색 헤드 3개, 클러스터 마스터 1개 및 범용 포워더 8개로 구성됩니다. 인덱서에는 Cisco UCS B 시리즈 M5 블레이드가 사용되었고, 다른 모든 구성 요소에는 M4 블레이드가 사용되었습니다. 이 솔루션에는 핫 티어과 웜 티어를 호스팅하는 플래시어레이//X70과 콜드 티어를 호스팅하는 플래시블레이드가 포함되었습니다.

플래시어레이 구성

플래시어레이는 Splunk 인덱스의 핫 데이터와 웜 데이터를 호스팅합니다. 플래시어레이에서 모든 인덱서 노드에 대한 별도의 볼륨을 분할해 iSCSI를 통해 각 인덱서 노드에 블록 디바이스로 연결합니다. 플래시어레이의 핫 데이터와 웜 데이터는 중복이 제거되고, 압축 및 암호화됩니다.

참고: 플래시어레이 볼륨은 iSCSI 또는 FC를 통해 인덱서 노드로 연결될 수 있습니다.

구성 요소	설명
플래시어레이	//X70R2
용량	38.34TB 원시 용량 26.84TB 가용 용량(데이터 절감 안함)
연결성	4 x 40GB/초 이중 iSCSI 2 x 1GB/초 이중 이더넷(관리 포트)
외부 사양	3U
소프트웨어	퓨리티//FA 5.1.2

표 1: 플래시어레이 구성 요소

플래시어레이 구성

플래시블레이드는 스토리지 레벨에서 압축 및 암호화된 Splunk의 콜드 데이터를 호스팅합니다. 모든 인덱서에 대해 별도의 NFS 파일 시스템을 플래시블레이드에서 생성해 이더넷을 통해 연결하고 인덱서에 하드 마운트합니다.

구성 요소	설명
플래시어레이	15 x 17TB 블레이드
용량	240TB 원시 용량 162.46 TB 가용 용량(데이터 절감 안함)
연결성	4 x 40GB/초 이더넷(데이터) 2 x 1GB/초 이중 이더넷(관리 포트)
외부 사양	4U
소프트웨어	퓨리티//FB 2.4.0

표 2: 플래시블레이드 구성 요소.

운영 체제 및 소프트웨어 구성

구성 요소	설명
Linux	CentOS 7.5(64비트)
Splunk	Splunk 8.0.5
Cisco UCS 매니저	3.2 (1d)

표 3: 운영 체제 및 소프트웨어 구성.

물리적 토폴로지

퓨어스토리지에 기반한 Splunk Enterprise 솔루션은 하드웨어(컴퓨팅, 스토리지, 네트워크)와 소프트웨어(Splunk Enterprise, CentOS Linux, Cisco UCS 매니저)가 결합된 스택으로 구성됩니다.

구성 요소	설명
인덱서	각각 다음이 포함된 8개의 Cisco UCS B200-M5 서버 블레이드: 2x Intel Xeon Gold 6138 @ 2GHz(20코어) 256GB 메모리
검색 헤드	각각 다음이 포함된 3개의 Cisco UCS B200-M4 서버 블레이드: Intel Xeon 프로세서 E5-2670 v3 CPU 2개(코어 12개) 256GB 메모리
클러스터 마스터	각각 다음이 포함된 1개의 Cisco UCS B200-M4 서버 블레이드: Intel Xeon 프로세서 E5-2609 v4 CPU 2개(코어 8개) 64GB 메모리
포워더	각각 다음이 포함된 8개의 Cisco UCS B200-M4 서버 블레이드: Intel Xeon 프로세서 E5-2680 v4 CPU 2개(코어 14개) 128 GB 메모리
네트워킹	Cisco UCS 6332 UP 16포트 패브릭 상호 연결 2개 Cisco MDS 9148S 패브릭 16G FC 스위치 2개 Cisco Nexus 9372PX 이더넷 스위치 2개
가상 인터페이스 카드	Cisco UCS VIC 1340의 40 Gbps 통합 I/O 포트
섀시	Cisco UCS 섀시 5108 3개(각각 6RU)

표 4: 퓨어스토리지 기반 Splunk Enterprise의 토폴로지.

논리적 토폴로지

매일 테라바이트 규모의 데이터를 인제스트하며 수많은 동시 검색을 지원하는 Splunk Enterprise의 엔터프라이즈급 구현의 경우, 인덱서 클러스터링의 분산 구현을 권장합니다. 이 랩에서 테스트한 구성은 다음과 같습니다.

- 인덱서 클러스터 설정의 인덱서 8개
- 검색 헤드 클러스터링에 검색 헤드 3개
- 클러스터 마스터 1개
- 범용 포워더 8개

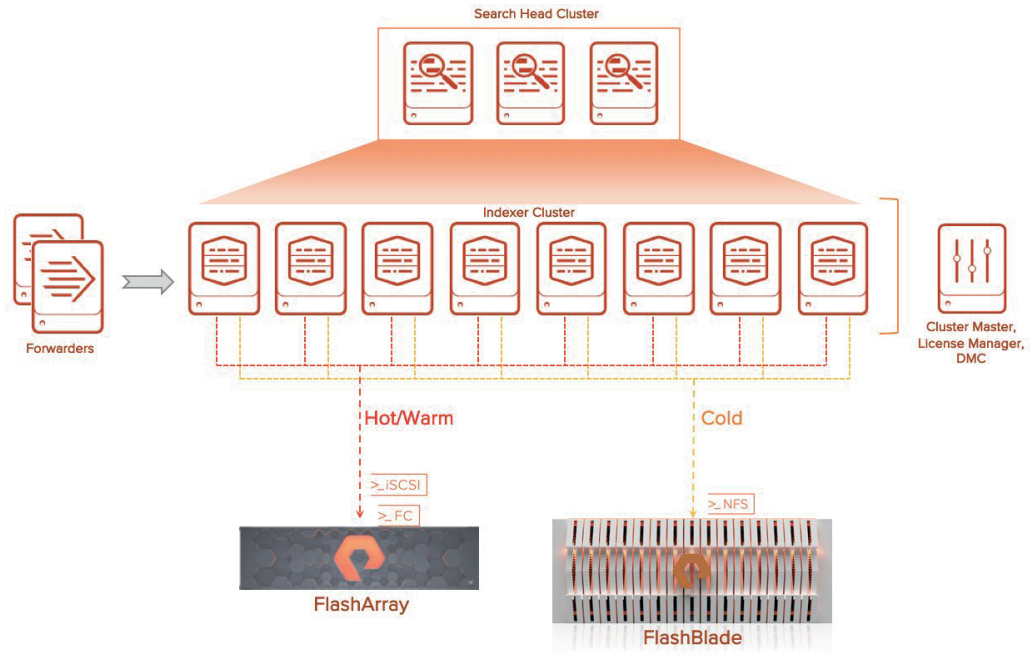


그림 5: 테스트한 분산 구현의 구성.

설계 고려 사항

Splunk Enterprise 구현은 일반적으로 다음과 같은 요소를 기반으로 합니다.

- **일일 인제스트 속도 또는 인덱싱 볼륨:** 높은 인덱스 속도를 확보하려면 더 많은 인덱서를 사용해야 합니다.
- **검색 수 및 유형:** 수많은 동시 검색 또는 리소스 집약적인 검색(예: 고집적도 검색)은 검색 헤드와 인덱서에 부담을 줄 수 있습니다.
- **동시 사용자 수:** 대시보드를 보거나 검색을 하는 사용자가 많은 경우, 더 많은 검색 헤드가 필요하며, 검색 헤드 클러스터가 이상적입니다.
- **데이터 충실도:** 데이터 손실을 방지하기 위해 인덱서 클러스터가 필요합니다.
- **데이터 가용성:** 인덱서 클러스터와 검색 헤드 클러스터를 모두 구현하여 전체 데이터 세트에 액세스합니다.
- **재해 복구 요구 사항:** 두 데이터센터 사이에 분산된 멀티사이트 인덱서 클러스터는 동일한 데이터 세트를 유지하여 신속히 복구할 수 있게 합니다.

Splunk Enterprise는 인덱스 데이터를 복제하여 소프트웨어 티어에서 데이터 충실도, 가용성 및 재해 복구 요구 사항을 관리합니다.

용량

Splunk는 인덱서 클러스터를 제공하여 데이터 손실을 방지하는 동시에, Splunk가 들어오는 데이터의 여러 복사본을 보관하는 인덱스 복제를 통해 검색을 위한 데이터 가용성을 촉진합니다. 인덱스 복제는 데이터 가용성이라는 이점을 제공하지만, 복제된 데이터를 보관하기 위한 추가 스토리지가 필요하고, 인덱서에서 처리되는 로드가 더 적어지며 인덱서 간 데이터를 복제하기 위한 추가 네트워크 트래픽이 생깁니다.

내결함성(fault-tolerance) 또는 검색을 위한 데이터 가용성 요구 사항을 결정하는 두 가지 주요 요인(복제 계수 및 검색 계수)이 있습니다.

- 복제 계수(RF)

  - 인덱서 클러스터의 내결함성을 결정합니다.

  - 인덱서 클러스터가 유지 관리하는 데이터의 복사본 수를 결정합니다.

  - 클러스터는 (RF-1) 피어 노드의 장애를 허용할 수 있습니다.

  - 원시 데이터를 압축된 형식으로 포함합니다.

  - 피어 노드의 동일한 복사본입니다.

- 검색 계수(SF)

  - 인덱서 클러스터가 유지 관리하는 검색 가능한 데이터 복사본 수를 결정합니다.

  - 압축된 형식의 원시 데이터와 인덱스 파일을 모두 포함합니다.

  - 인덱스 데이터가 복제된 원시 데이터를 기반으로 피어 노드에서 로컬로 생성될 때 피어 노드의 논리적 복사본입니다.

  - 검색 요소가 증가한다고 해서 검색 성능이 향상되는 것이 아니라 검색 가용성이 향상됩니다.

  - 권장되는 기본값은 2입니다.

스토리지 적정 규모 산정 지침

인덱서 클러스터 구현 모델에서 Splunk를 위한 적절한 스토리지 규모를 산정하려면 다음 항목이 필요합니다. 스토리지 규모를 산정하는 데 정해진 하나의 공식이 존재하는 것은 아니라는 사실을 유념하세요.

- 일일 인제스트 속도 – Splunk 일일 라이선스를 사용하기 위해 권장됨
- 보존 기간(핫/웜 및 콜드 사용량 모두 포함)
- RF 및 SF

개괄적으로 Splunk의 스토리지 요구 사항은 다음과 같이 계산됩니다.

$$Storage\ Required = retention\ period\ *\ daily\ ingest\ data\ *(0.15 * RF + 0.35 * SF)$$

> **참고:** Splunk 압축률은 일반적으로 50% (원시 데이터로 인제스트된 데이터의 15%, 인덱스 메타데이터로 인제스트된 데이터의 35%)이지만, 이는 전적으로 데이터 유형에 달려 있습니다. 카디널리티가 높은 데이터의 경우, 이 비율이 낮아져 전체적으로 압축된 데이터 비율이 낮아지거나 스토리지 규모 산정 관련 요구 사항이 증가할 수 있습니다.

이 솔루션을 지원하기 위한 플래시어레이 및 플래시블레이드의 스토리지 요구 사항을 산정하려면 다음 두 항목이 추가로 필요합니다.

- 핫/웜 티어 보존 기간(HWR)
- 콜드 티어 보존 기간(CR)

플래시어레이에서의 핫/웜 티어에 대한 스토리지 요구 사항은 다음과 같이 계산됩니다.

$$FA\ Storage\ required = \ daily\ ingest\ data\ * HWR * \ (0.15 + 0.35 * SF)$$

플래시블레이드에서의 콜드 티어에 대한 스토리지 요구 사항은 다음과 같이 계산됩니다.

$$FB\ Storage\ required = \ daily\ ingest\ data\ * CR * \ (0.15 * RF + 0.35 * SF)$$

> **참고:** 이러한 계산은 예상치이며, 플래시어레이 또는 플래시블레이드에서 제공하는 데이터 절감률 및 RAID 오버헤드는 포함되지 않습니다. 퓨어스토리지 어카운트 팀과 협력하여 Splunk 환경에 적합한 규모 요구조건을 확인하시기 바랍니다.

위의 계산에서 알 수 있듯이, 복제 계수는 플래시어레이에서의 핫/웜 티어 계산에 큰 영향을 주지 않습니다. 원시 데이터의 여러 복사본이 항상 스토리지 수준에서 중복 제거되어 추가적인 스토리지 공간이 렌더링 되기 때문입니다. 원시 데이터의 중복 제거된 단일 복사본이 단일 장애 지점이 될 것이 우려되는 경우, 플래시어레이는 데이터 블록 그룹에서 하나 또는 두 개의 동시 읽기 장애로부터 복구 성능을 제공하는 RAID-HA 데이터 보호를 사용합니다.

예를 들어, 핫/웜 데이터의 보존 기간이 90일, 콜드 데이터의 보존 기간이 270일이고 RF=2, SF=2이며, 일일 인제스트 속도가 4TB인 경우, 총 스토리지 요구 사항은 다음과 같습니다.

- 필요한 스토리지 = 360 * 4TB * (0.15 * 2 + 0.35 * 2) = 1,440TB
- 핫/웜 요구 사항 = 90 * 4TB(0.15 * 2 + 0.35 * 2) = 360TB
- 콜드 요구 사항 = 270 * 4TB(0.15 * 2 + 0.35 * 2) = 1,080TB

핫/웜 지원을 위한 가용 플래시어레이 스토리지 요구 사항은 원시 데이터의 중복 제거 외에 인덱스 데이터의 압축 가능성 여부에 따라 140~180TB 사이가 됩니다.

콜드 티어를 지원하기 위한 가용 플래시블레이드 스토리지 요구 사항은 인덱스 데이터의 압축률에 따라 600~750TB 사이입니다. 두 경우 모두, 실제 원시 스토리지는 RAID 또는 EC 오버헤드에 따라 달라질 수 있습니다.

성능

분산 인덱서 클러스터 환경에서 검색 및 인덱스 성능을 활성화 하기 위해 시스템 리소스와 대역폭을 계획하려면 인덱싱되는 총 데이터 볼륨과 활성 동시 검색(예약 또는 기타) 수를 항상 고려해야 합니다.

Splunk의 성능 권장 사항에 따르면, Splunk의 참조 하드웨어 사양을 충족하는 인덱서는 Splunk Enterprise 구현의 경우는 검색 로드를 지원하면서 일일 300GB를 인제스트하고, Enterprise Security 구현의 경우는 일일 100GB를 지원할 수 있습니다. Splunk는 분산 Splunk Enterprise 구현에서 보다 나은 인덱싱 성능과 검색 동시성을 제공하기 위해 두 가지 새로운 하드웨어 사양을 도입했습니다. 중급 사양에서는 2GHz 이상의 CPU 코어 24개와 64GB RAM을 권장하며, 고성능 사양에서는 2GHz 이상의 CPU 코어 48개와 128GB RAM을 권장합니다.

Splunk 검색 헤드와 인덱서는 CPU를 많이 사용하므로, 분산 검색이 활성화된 상태에서는 검색과 인덱싱 기능을 분리하는 것이 좋습니다. 이를 통해 검색 헤드가 병렬 검색을 분산할 수 있습니다.

보다 자세한 성능 권장 사항은 Splunk Enterprise 용량 계획 매뉴얼을 참조하십시오.

솔루션 검증 및 테스트

Splunk Enterprise 아키텍처의 목표는 검색 유형 또는 스토리지 티어에 관계없이 빠른 인덱싱을 지원하고 검색 응답을 개선하는 동시에 직접 연결 스토리지와 관련된 문제를 제거하는 것입니다.

퓨어스토리지 시스템에 구축된 Splunk 솔루션은 데이터 인제스트와 검색, 그리고 Splunk 공간 관리의 운영 효율성이라는 Splunk Enterprise의 주요 기능들의 테스트를 통해 검증되었습니다.

- 운영 효율성
- 데이터 인제스트
- 검색 동작

운영 효율성

시스템에서 생성되는 데이터 양이 지속적으로 늘어나는 환경에서 Splunk를 사용해 공간을 효과적으로 관리하려면, 항상 성장을 예측하는 것이 중요했습니다. 이 때문에 Splunk 관리자에게 스토리지 공간 관리는 힘든 작업이었습니다. 스토리지 관리를 위한 운영 효율성 테스트는 인덱서 노드에 스토리지 공간을 추가할 때 플래시어레이와 플래시블레이드 같은 엔터프라이즈 스토리지가 어떤 이점을 갖는지 보여 줍니다.

퓨어스토리지 기반 Splunk Enterprise 솔루션은 컴퓨팅에서 스토리지가 분리되기 때문에, 인덱서 클러스터에 스토리지 공간을 추가하는 운영 작업은 기존 방식보다 더 빠르고 더 간소해 집니다.

스토리지 관리 테스트 개요

다음 섹션에서는 기존 접근 방식 대비 퓨어스토리지에서 스토리지 공간이 어떻게 추가되는지 비교해보고 각 테스트가 어떻게 수행됐는지 설명합니다.

기존 접근 방식

기존의 직접 연결 스토리지 모델에서는 서버의 사용 가능한 슬롯에 있는 HDD 또는 SSD 같은 스토리지 장치로 인덱서 노드가 미리 채워집니다. 이러한 스토리지 장치를 설정하는 가장 좋은 방법은 RAID-1 같은 RAID 형식을 사용해 단일 장애 지점을 방지하는 것이지만, 이로 인해 스토리지 공간이 두 배로 늘어납니다. 인덱서 노드 공간이 부족한 경우, 클러스터에 새 피어 노드를 추가하고 클러스터를 리밸런싱하여 모든 노드에 데이터를 균등하게 분산시키는 것이 표준 관행입니다.

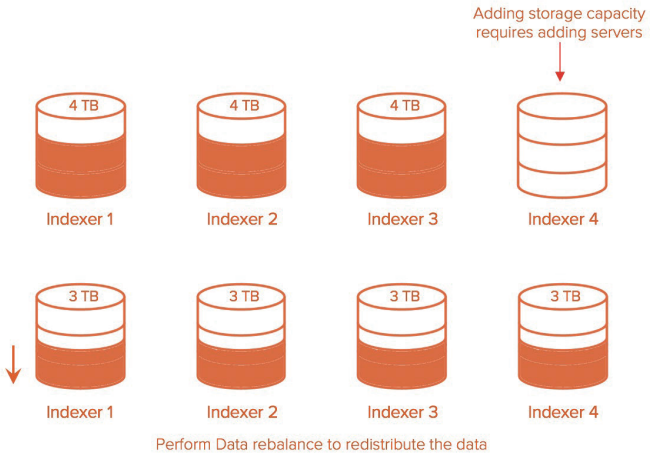


그림 6: 데이터 재분산을 위한 리밸런싱.

추가 스토리지 공간을 지원하기 위해 새 인덱서를 추가하려면 비용이 많이 들 뿐만 아니라, 클러스터 내의 모든 노드에 버킷 데이터를 분산하기 위해 데이터 리밸런싱을 수행해야 하기 때문에 시간도 많이 듭니다. 8개 노드로 구성된 인덱서 클러스터의 총 공간은 67,000개의 버킷에 걸쳐 60TB였습니다. 이 테스트에서는 인덱서를 추가하여 공간을 추가하고, 다음 명령을 사용하여 데이터 리밸런싱을 수행한 후, 총 시간을 측정했습니다.

```
$SPLUNK_HOME/bin/splunk rebalance cluster-data -action start -auth admin:splunk123
```

퓨어스토리지의 접근 방식

분리된 모델에서는 전체 인덱서 클러스터를 위한 엔터프라이즈 스토리지가 서버에서 직접 연결된 개별 스토리지 미디어를 대체합니다. 따라서 공간을 추가하는 기능이 엔터프라이즈 스토리지 솔루션으로 오프로드됩니다.

플래시어레이의 경우, 각 인덱서 피어에 대해 새 볼륨을 추가하고 피어 노드에 연결한 다음 논리적 볼륨 관리자에 추가하여 추가 스토리지를 사용하는 것이 더 쉽고 간단합니다. 추가 공간을 확보하기 위해 새로운 서버를 추가할 필요가 없습니다.

플래시블레이드의 경우, GUI에서, 또는 CLI를 통해 NFS 파일 시스템을 편집하고 크기를 늘리면 인덱서가 자동으로 새 크기를 반영합니다. 스토리지를 추가하는 것이 이보다 더 간단할 수는 없습니다.

콜드 티어의 현재 파일 시스템은 10TB로 설정되어 있습니다. 이 테스트에서는 다음의 CLI 명령을 통해 플래시블레이드 파일 시스템을 업데이트하여, 클러스터에 있는 8개의 인덱서 전반에서 콜드 티어의 공간을 늘렸습니다.

```
purefs setattr --size 20T splunk-cold-ix1
purefs setattr --size 20T splunk-cold-ix2
purefs setattr --size 20T splunk-cold-ix3
purefs setattr --size 20T splunk-cold-ix4
purefs setattr --size 20T splunk-cold-ix5
purefs setattr --size 20T splunk-cold-ix6
purefs setattr --size 20T splunk-cold-ix7
purefs setattr --size 20T splunk-cold-ix8
```

운영 효율성 테스트 결과

공간을 추가하기 위해 새 인덱서 노드를 추가하기 전에, Settings => Distributed Environment에서 'Indexer Clustering'을 선택하여 전체 인덱서의 버킷 사용량을 캡처했습니다.

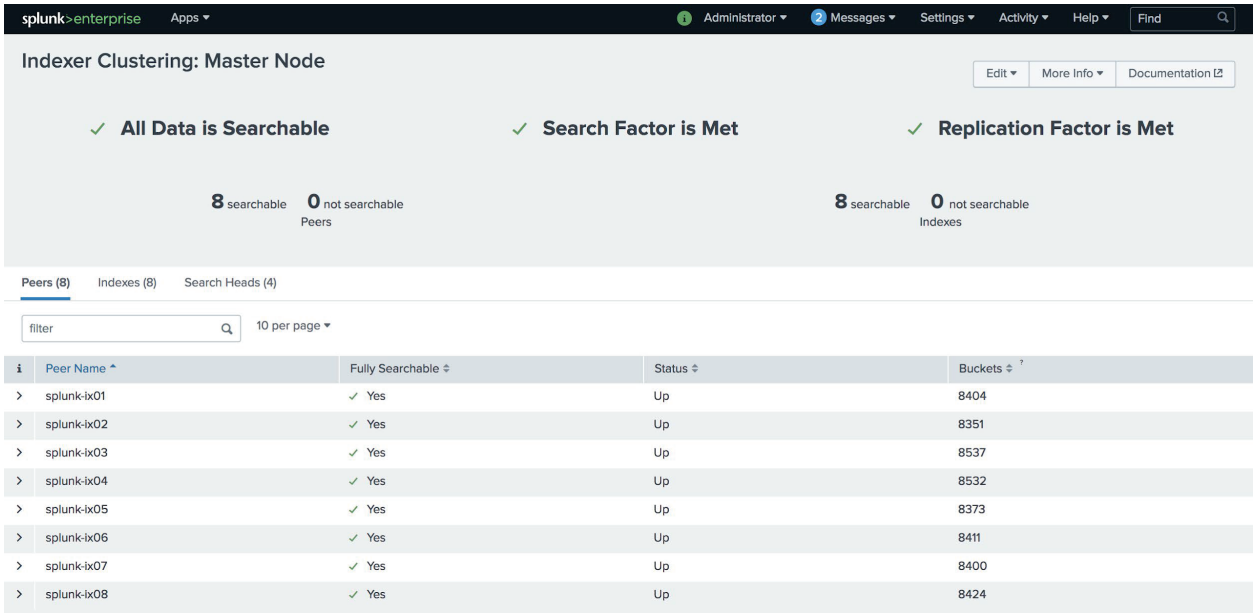


그림 7: 인덱서 전반에서 버킷 사용.

인덱서 노드가 추가되면, 기본적으로 데이터 리밸런싱이 수행되지 않으며 새 인덱서는 최소 버킷을 표시합니다.

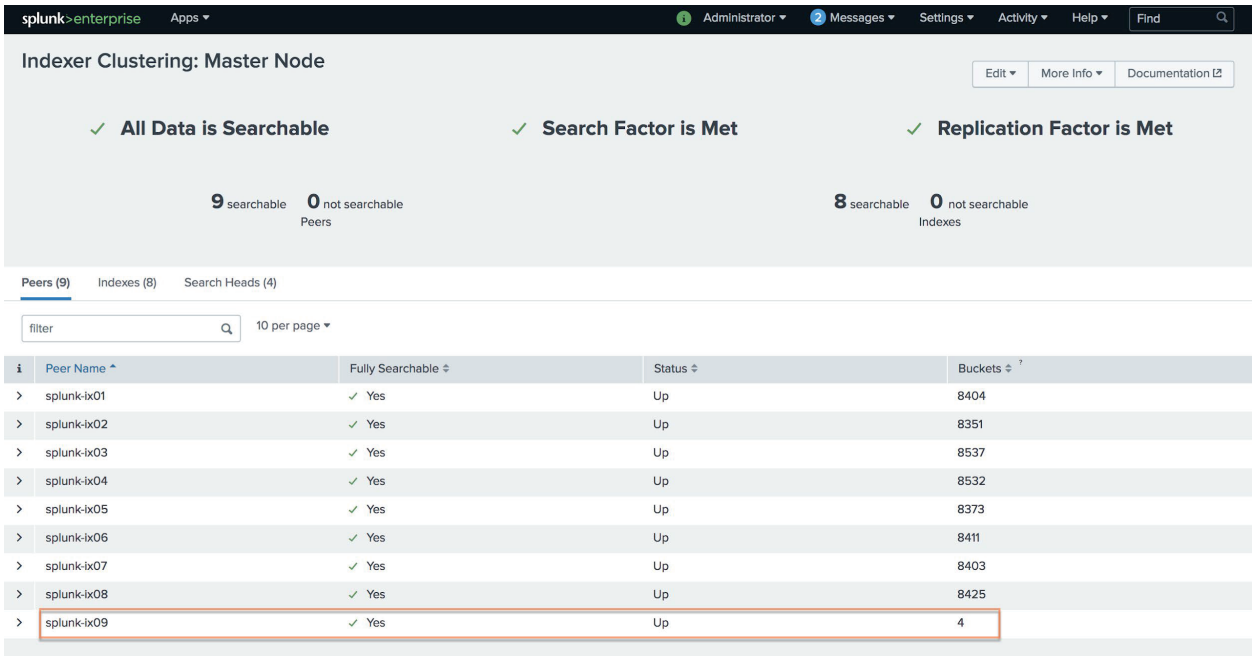


그림 8: 최소 버킷을 포함한 새로운 인덱서.

CLI를 통해 데이터 리밸런싱 명령이 호출되었으며, 완료하는 데 11시간 45분이 걸렸습니다. 데이터 리밸런싱이 끝난 후 버킷은 9개의 인덱서 전체에 균등하게 분산되었습니다.

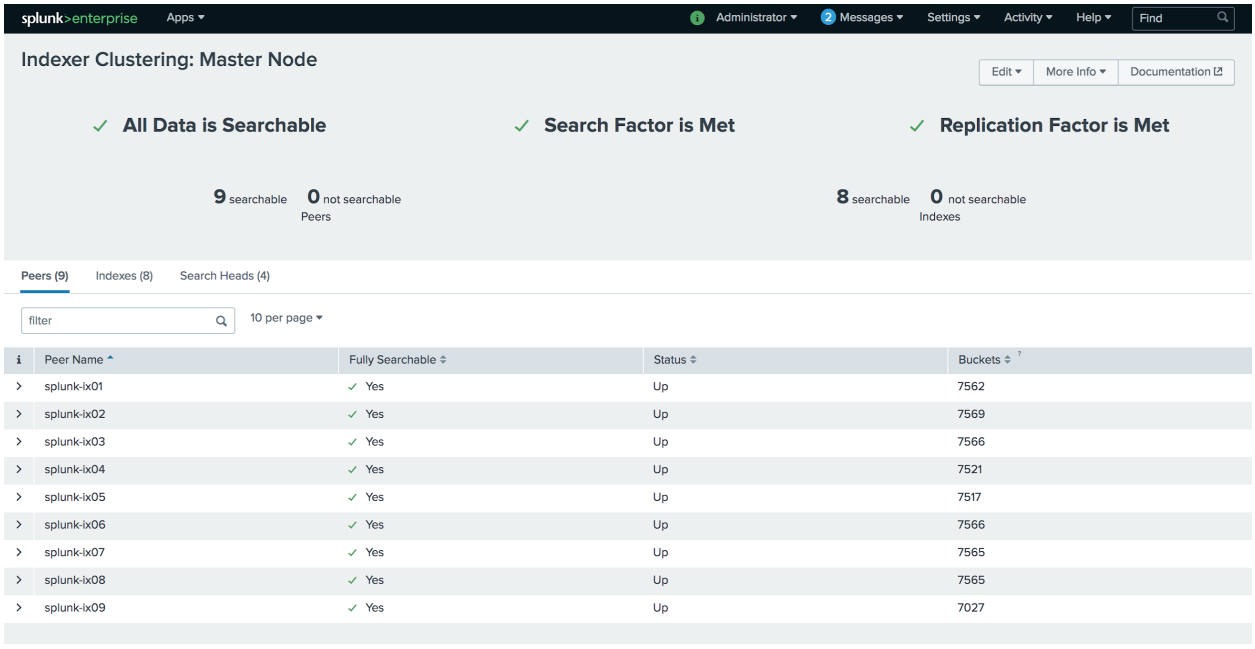


그림 9: 데이터 리밸런싱 후 균일한 버킷 분산.

데이터 리밸런싱 중 버킷 수정 세부 정보에 해당 기간 동안의 'to_fix_rebalance' 작업이 표시됩니다. 이 프로세스 중에 Splunk는 재분산의 일부로 노드 전반에 데이터를 복제하고, 버킷을 검색할 수 있도록 하며, 이동된 버킷의 크기를 잘라냅니다.

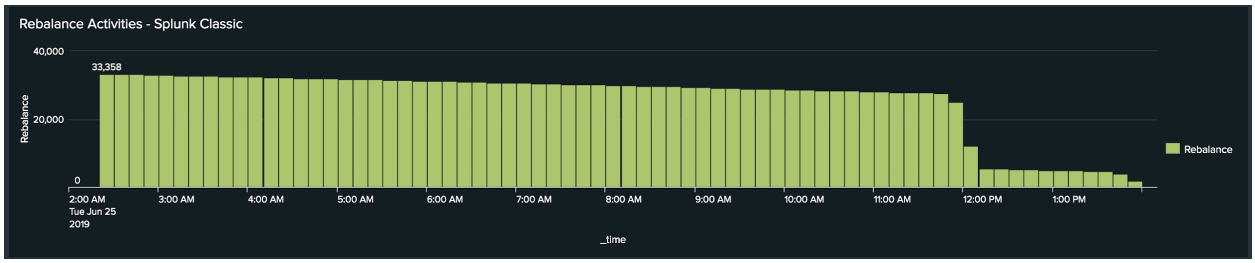


그림 10: 시간 경과에 따른 리밸런싱 작업.

플래시블레이드에서 콜드 티어 공간 증가

공간이 증가하기 전에 콜드 티어 볼륨의 공간 사용량이 캡처되었습니다. 첫 번째 인덱서의 출력이 아래에 간결하게 나와 있습니다.

```
[root@splunk-ix01 ~]# df -h /cold
Filesystem                      Size  Used Avail Use% Mounted on
10.21.214.203:/splunk-cold-ix01   10T  6.6T   10T  66% /cold
```

그림 11: 첫 번째 인덱서의 출력.

플래시블레이드에서 콜드 티어 파일 시스템 공간을 늘리기 위한 CLI가 실행되었고 8개 파일 시스템 모두가 2초 이내에 완료되었습니다.

```
pureuser@sn1-fb-d077-am2> purefs setattr --size 20T splunk-cold-ix01
Name              Size  Used     Hard Limit  Created                  Protocols
splunk-cold-ix01  20T    6.6T    False       2019-05-16 18:17:28 PDT  nfs
pureuser@sn1-fb-d077-am2>
```

그림 12: 완료된 CLI.

이제 첫 번째 인덱서 노드에 마운트된 동일한 NFS 파일 시스템에 스토리지 크기가 증가한 것을 볼 수 있습니다.

```
[root@splunk-ix01 ~]# df -h /cold
Filesystem                      Size  Used Avail Use% Mounted on
10.21.214.203:/splunk-cold-ix01   20T  6.6T   10T  33% /cold
```

그림 13: 증가된 스토리지 크기.

표 5는 저장 공간 추가 테스트 결과를 요약해 보여줍니다.

운영 활동	기존 접근 방식	퓨어스토리지의 접근 방식
스토리지 공간 추가	11시간 45분	2초

표 5: 기존 스토리지 공간 vs 퓨어스토리지 공간 테스트 결과.

이와 같이, 분리된 스토리지로 Splunk Enterprise에 공간을 추가하는 것이 훨씬 간단하고 빠르며, 무엇보다 비용이 저렴합니다. 데이터 리밸런싱이 왜 필요한지에 대해서는 분분한 이유가 있을 수 있지만, 단순히 공간 사용 문제를 해결하기 위해 수행되는 경우 퓨어스토리지 제품과 함께 분리된 스토리지 옵션을 사용하면 이러한 문제를 완전히 방지할 수 있습니다.

데이터 인제스트

개요

데이터 인제스트 중에 Splunk 인덱서는 원시 데이터를 타임스탬프를 기준으로 이벤트로 변환하여 유입되는 데이터를 인덱싱하고 이를 인덱스 파일에 저장합니다. Splunk는 인덱서 노드의 버킷에 모든 데이터를 저장합니다. 버킷은 인덱서 노드에서 호스팅되는 인덱스 디렉터리입니다. 버킷은 핫(hot), 웜(warm), 콜드(cold), 프로즌(frozen), 쏘드(thawed)의 다양한 단계를 거치게 됩니다.

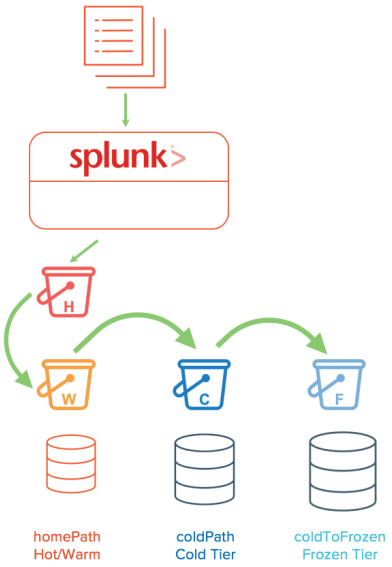


그림 14: Splunk 버킷의 단계.

데이터가 인덱싱되면 핫 버킷에 저장되는데, 이 버킷은 검색이 가능하고 쓰기도 가능합니다. 핫 버킷은 쓰기 될 수 있는 유일한 버킷입니다. 미리 지정된 크기에 도달하거나 최대 핫 버킷 수에 도달하면, 핫 버킷은 웜 버킷으로 롤링됩니다. 웜 버킷은 검색이 가능하지만 쓰기가 안됩니다. 최대 웜 버킷 수에 도달하는 등의 일정 조건이 충족되면, 인덱서는 웜 버킷을 연령에 따라 콜드 버킷으로 롤링합니다. 가장 오래된 웜 버킷이 먼저 콜드 버킷으로 롤링됩니다.

이 솔루션에서 핫 버킷과 웜 버킷은 모두 플래시어레이에서 호스팅되고 콜드 버킷은 플래시블레이드에서 호스팅됩니다. 핫 버킷에서 데이터가 인제스트되긴 하지만, 엔터프라이즈 규모에서는 온종일 버킷이 핫 티어에서 웜 티어로, 웜 티어에서 콜드 티어로 이동을 하기 때문에 콜드 티어의 쓰기 성능은 핫 티어에서 만큼 중요합니다.

인제스트 테스트 개요

복제 요소 및 검색 요소 모두 2로 설정된 모든 인덱스를 사용하여 8개의 피어 노드로 구성된 인덱서 클러스터에서 64TB의 데이터를 인제스트하는 테스트를 수행했습니다. 64TB의 데이터를 로드하는 데 걸리는 시간을 측정하고, 시스템 활용도와 기타 관련 메트릭을 수집했습니다.

인제스트 테스트 개요

Splunk Enterprise 인제스트 테스트를 확장하기 위해, 4개의 16TB 배치, 즉 총 64TB를 인제스트 했습니다. 생성하기로 한 테스트 데이터는 희소 및 희귀 검색에 사용될 수 있는 모든 이벤트에 추가 키워드를 더한 sourcetype access_combined 아파치 유형 로그였습니다. 데이터세트는 Go 프로그래밍 언어를 사용하여 사용자 정의 개발 스크립트를 통해 생성되었습니다. 데이터 생성 스크립트는 8개의 포워더 상에서 실행되었으며, 서버당 2TB로 각 배치당 총 16TB를 생성했습니다.

포워더를 통해 데이터 속도를 높이기 위해, $SPLUNK_HOME/etc/system/local/limits.conf의 처리량 입력 maxKBps를 0으로 업데이트했습니다. 이는 포워더에서 인덱서로 데이터 처리량 조절이 없음을 의미합니다. 범용 포워더의 기본 처리량은 256Kbps입니다. 테스트를 위해 더 큰 처리량(예: 10240 또는 0)을 사용해도 되지만, 표준 운영 절차의 경우라면 이 값을 네트워크 인프라에 부담을 주지 않도록 환경을 반영하는 값으로 설정할 수 있습니다.

16TB의 데이터는 원샷 방법을 사용해 8개의 범용 포워더로부터 4개 인덱스 apache-pure, apache-pure2, apache-pure3, apache-pure4에 순서대로 인제스트되었습니다.

```
$SPLUNK_HOME/bin/splunk add oneshot <source log file>  -index apache-pure2 -sourcetype access_combined -auth
admin:splunk123
```

인제스트 테스트 결과

8개의 인덱서를 사용하여 8개의 범용 포워더로부터 인덱서 클러스터로 64TB의 데이터를 인제스트했으며, 이는 평균 238MBps(19.62TB/일)의 속도로 78시간 17분이 소요되었습니다.

인제스트된 데이터	경과 시간	인덱싱 속도	인덱서 당 인덱싱 속도	일일 인제스트 속도
64TB	78시간 45분	238.12MB/초	29.76MB/초	19.62TB/일

표 6: 인제스트 테스트 결과.

그림 15는 인덱스 중 하나에 대한 16TB 배치 로드의 인덱싱 속도를 보여 줍니다. 표에서 보듯이 인덱서는 최대 인덱싱 속도인 272MB/초에 도달했으며 평균 인덱싱 속도는 238MB/초입니다.

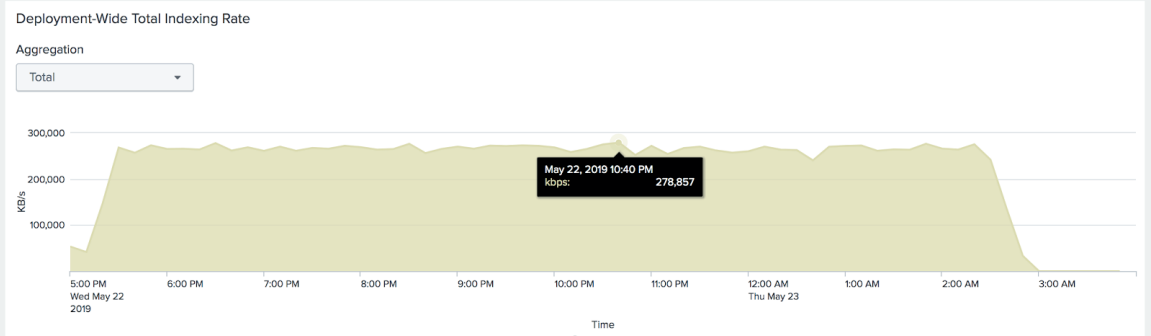


그림 15: 인덱스 중 하나에 대한 16TB 배치 로드의 인덱싱 속도.

인제스트 도중, 8개 인덱서 노드의 전체 CPU 사용률은 25% 미만이었습니다. 다음 스크린샷은 인제스트가 진행되는 동안 플래시어레이와 플래시블레이드의 성능 메트릭을 보여 줍니다.

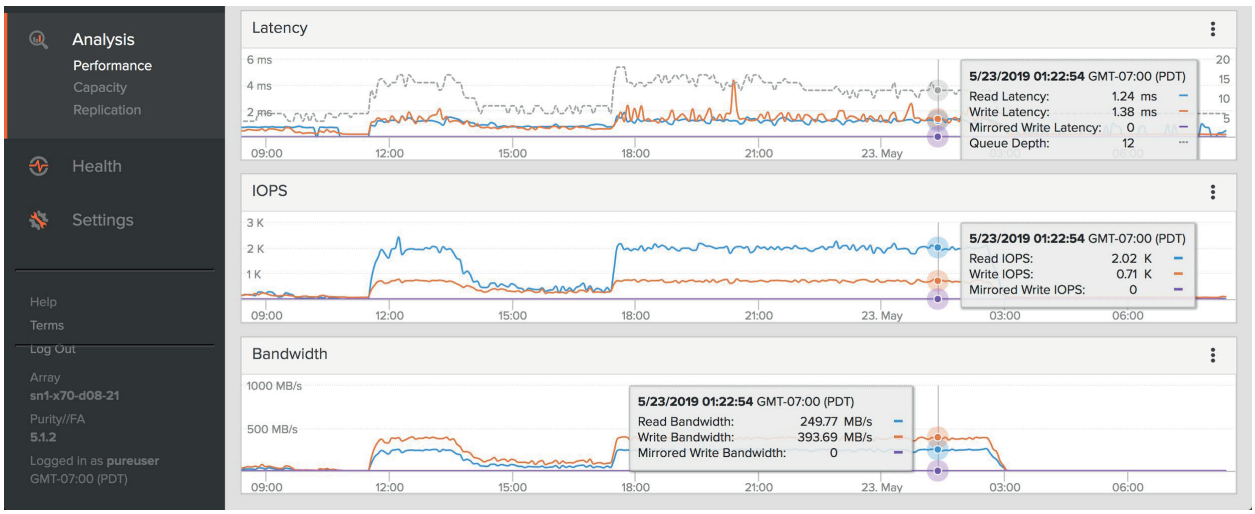


그림 16: 인제스트 프로세스 중 플래시어레이의 성능 메트릭.

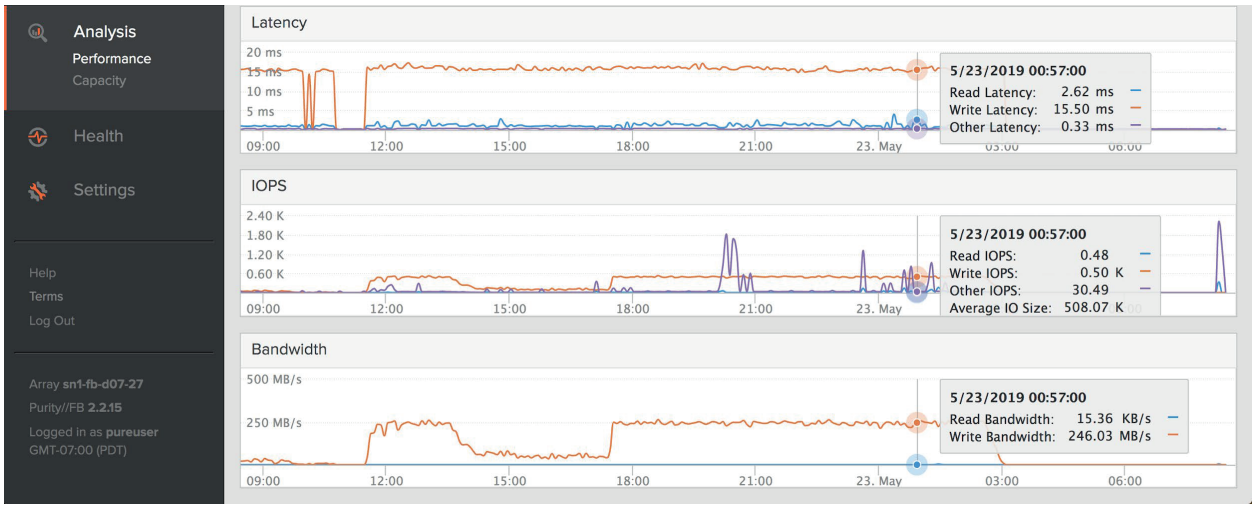


그림 17: 인제스트 프로세스 중 플래시블레이드의 성능 메트릭.

대규모 인제스트 중에 버킷이 핫에서 웜, 콜드로 지속적으로 롤링될 때, 플래시어레이와 플래시블레이드 모두 쓰기 작업이 동일하게 진행되었습니다.

테스트 결과를 보면, 플래시어레이와 플래시블레이드의 성능은 64TB의 인제스트 프로세스 중에 전혀 영향을 받지 않았으며, 두 시스템 모두 추가 워크로드를 처리할 수 있는 처리 성능이 많이 남아 있었습니다.

공간 사용량

Splunk는 유입되는 원시 데이터를 압축하고 인덱스 데이터 생성과 함께 인제스트된 데이터의 약 50%의 데이터 절감률을 제공합니다. 인덱서 클러스터링의 경우, 이러한 데이터 절감은 원시 데이터와 인덱스 데이터의 RF 및 SF 복사본에 의해 상쇄되어 데이터의 고가용성을 제공합니다. 데이터 절감은 데이터 카디널리티의 영향을 추가로 받을 수 있습니다.

따라서 RF=2, SF=2인 Splunk Enterprise에서 인제스트된 64TB의 데이터는 Splunk에서 59.84TB의 공간을 소비합니다. 이는 Splunk 레벨에서 데이터 절감률이 1.06:1라는 것을 의미합니다.

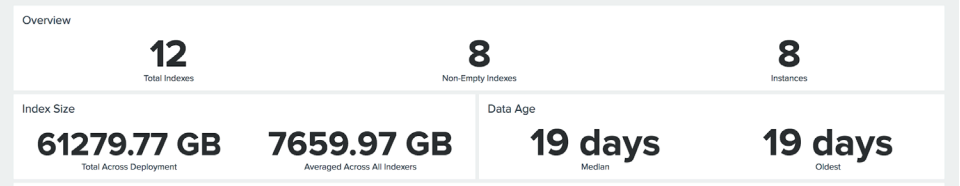


그림 18: 인덱스 개요.

Splunk를 통한 데이터 절감 외에도 플래시어레이와 플래시블레이드 모두 Splunk 데이터를 압축하여 데이터 유형과 카디널리티를 기준으로 데이터를 추가로 절감합니다. 이 테스트에 사용된 인제스트 데이터는 카디널리티가 더 높은 데이터로, RF = 2, SF = 2에서 데이터 절감률은 1.35:1 이었습니다.

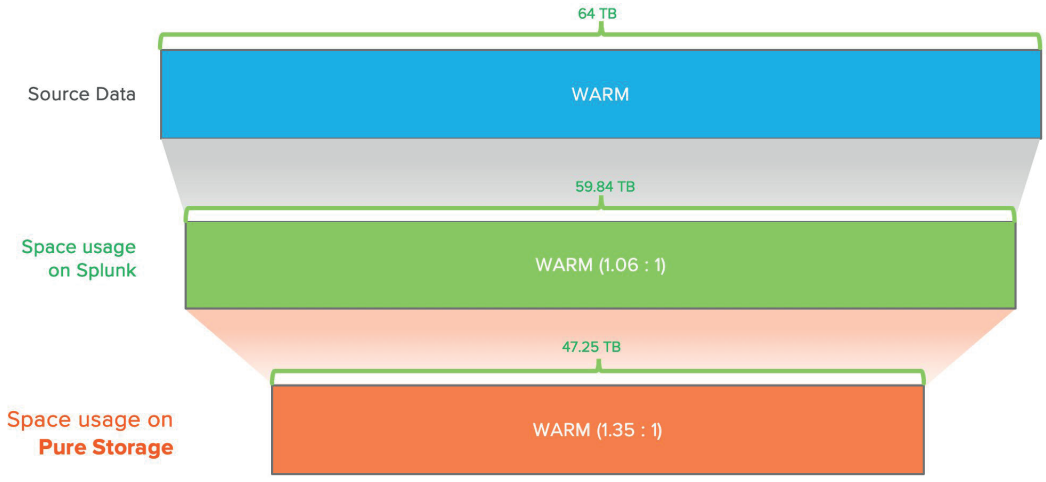


그림 19: 퓨어스토리지를 사용한 데이터 절감.

Splunk Enterprise의 경우 2GHz 12개 코어가 포함된 참조 하드웨어를 기준으로 한 인덱서당 Splunk의 일일 권장 수집 속도는 300GB이며 일부 검색 로드를 지원합니다. 이는 8개의 인덱서 노드 구성에 대해 **일일 2.4TB**를 인제스트하는 것과 동일합니다. Splunk는 엔터프라이즈 규모에 48개 코어가 포함된 고성능 참조 하드웨어를 권장합니다. 이 경우 검색 로드를 지원하면서 300GB/일의 인제스트 속도보다 높은 성능을 얻을 수 있습니다.

동시 검색 로드를 사용한 이 테스트에서, 8개의 인덱서 노드가 25% 미만의 CPU 활용도에서 정상적인 인덱싱 대기열을 사용한 경우 이론 상으로는 **일일 19.61TB**의 인제스트가 가능했습니다. 총 2x20개의 CPU 코어(40개 코어)로 구성된 인덱서 노드를 위해 Cisco UCS M5 블레이드를 선택했기 때문입니다. 이는 Splunk의 고성능 참조 하드웨어의 성능 역량과 일치하는 것으로, 탁월한 인프라에서 성능을 향상시킬 수 있다는 가능성을 보여줍니다.

검색 작업

대부분의 Splunk 고객은 일반적으로 핫/웜 티어에 있는 단기 데이터에 대한 검색을 수행합니다. 따라서 Splunk 모범 사례는 검색 성능을 향상시키기 위해 핫/웜 데이터를 더 빠른 스토리지에 두는 것입니다.

단기적인 데이터만이 아니라 보유하고 있는 모든 데이터에서 인사이트를 얻는 데 관심을 갖는 엔터프라이즈 고객이 늘어남에 따라, 콜드 티어에서 데이터를 검색해야 할 필요성도 커졌습니다. 동시에 고객들은 콜드 티어에서 검색이 완료될 때까지 오래 기다리는 것을 원하지 않습니다. 장기 데이터 검색 시간이 단기 데이터 검색만큼이나 빠르기를 원합니다.

검색 테스트 개요

다음과 같은 테스트를 수행하여 경과 시간, 제거된 버킷 정보 같은 중요한 메트릭을 측정했습니다.

- 120개의 동시 검색(희소 및 희귀) – 100% 핫/웜 티어
- 120개의 동시 검색(희소 및 희귀) – 90% 핫/웜 티어, 10% 콜드 티어

첫 번째 검색은 핫/웜 티어에서 120개의 동시 검색을 모두 수행했으며, 두 번째 테스트는 핫/웜 티어에서 108개의 검색을 수행하고 나머지 12개의 검색을 콜드 티어에서 수행하여 10%의 콜드 티어에 대한 검색을 시뮬레이션했습니다. 두 경우 모두에서 희소와 희귀 두 가지 검색 유형을 수행했습니다. 이 검색 테스트의 목적은 플래시어레이에서 호스팅되는 핫/웜 티어의 전체적인 검색 성능과 플래시블레이드에서 호스팅되는 콜드 티어의 전체적인 검색 성능을 보여 주기 위해서 입니다. 기존 설정에서 콜드 티어는 보통 가격이 더 낮은 디스크 드라이브에 저장되기 때문에 검색 속도가 느립니다.

검색 테스트 설정

4개 인덱스에서 64TB의 데이터 세트(인덱스당 16TB)가 16일 동안 매일 1TB씩 분산되었습니다. 평균적으로 각 인덱스는 하루에 최대 500개의 버킷을 사용하며, 모든 인덱스의 버킷 크기(maxDataSize)는 **auto**, 즉 750MB로 설정됩니다.

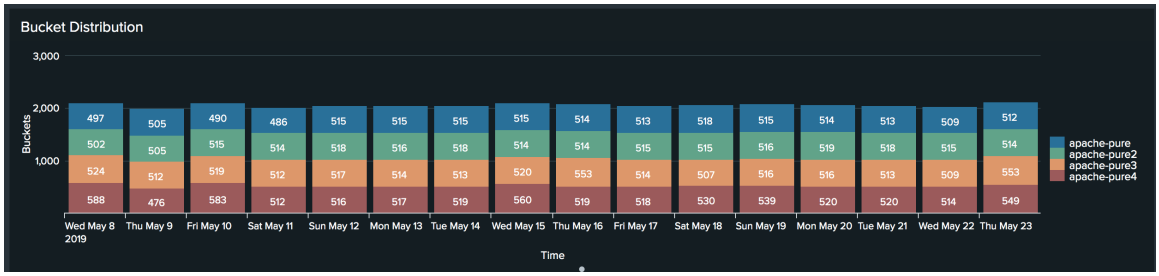


그림 20: 시간 경과에 따른 버킷 분산.

총 데이터의 10%가 핫/웜 티어에 있고 나머지 데이터는 콜드 티어에 있는 실제 환경 시나리오를 시뮬레이션하려면 핫/웜 티어가 6.4TB가 되어야 합니다.

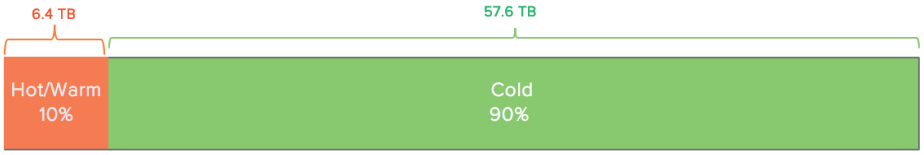


그림 21: 핫/웜 및 콜드 티어에서 데이터 분산.

다음 계산을 사용하여 올바른 homePath 최대 데이터 크기를 계산했습니다.

아파치 인덱스당 핫/웜 크기 = 10% * 인덱싱된 데이터 / 인덱서 수 / 아파치 인덱스 수

⇒ 10/100 * 64TB / 8 / 4 = 인덱스당 200GB

따라서 아래와 같이 각 아파치 인덱스의 homePath.maxDataSizeMB를 indexits.conf 파일에서 200GB로 설정합니다.

```
[apache-pure]
homePath = volume:hot/$_index_name/db
coldPath = volume:cold/$_index_name/colddb
thawedPath = $SPLUNK_DB/$_index_name/thaweddb
homePath.maxDataSizeMB = 204800
# 200 GB 의 핫/웜
```

동시 검색

다음 두 가지 차원에 대해 동시 검색 테스트를 수행했습니다.

- 검색 유형(희소 및 희귀)
- 핫/웜 티어(100% 및 90%)

이 테스트에서는 각 검색 헤드에 대해 40개의 검색을 사용하여 120개의 동시 검색을 실행했습니다. 120개의 동시 검색을 통해 피어 노드당 15개의 검색을 테스트할 수 있었습니다. (모든 피어 노드에 균등하게 분산될 것이라는 보장은 없습니다.) 표 7은 검색 유형(희소 및 희귀), 검색 데이터 세트당 크기, 검색당 이벤트 및 예상 이벤트를 보여 줍니다.

검색 분산	검색 유형	검색당 데이터 세트	검색당 이벤트	핫/웜 검색	총 핫/웜 데이터 세트	콜드 티어 검색	총 콜드 데이터 세트
100%	희소	15GB	9820	120	1.8TB	0	-
90%	희소	15GB	9020	108	1.6TB	12	180GB
100%	희귀	10GB	2	120	1.2TB	0	-
90%	희귀	10GB	2	108	1.08TB	12	120GB

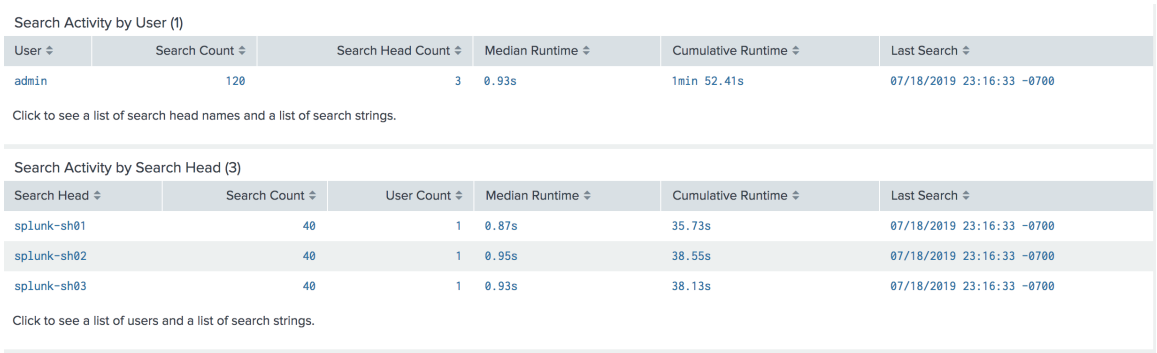
표 7: 검색 유형(희소 및 희귀), 검색 데이터 세트당, 크기, 검색당 이벤트 및 예상 이벤트.

100% 동시 검색의 경우, 120개 검색 모두 핫/웜 티어의 데이터를 검색합니다. 각 희소 검색은 약 15GB의 Splunk 데이터를 검색하므로 120개 검색을 모두 실행하면 1.8TB의 Splunk 데이터 세트가 검색될 것으로 예상됩니다. 희귀 검색은 핫/웜 티어에서 1.2TB Splunk 데이터를 검색합니다. 90% 핫/웜 티어에서 동시 검색을 수행한 경우, 90%(108개)가 핫/웜 티어의 데이터를 검색하고 10%(12개)가 콜드 티어의 데이터를 검색합니다.

검색은 REST 엔드포인트를 사용하여 curl을 통해 제출되었습니다. 동일한 메커니즘을 통해 검색 결과를 추출했습니다. 또한 의미 있는 I/O 응답 시간을 얻기 위해 모든 검색 전에 운영 체제의 캐시를 삭제했습니다.

검색 테스트 결과

검색 사용 통계: DMC => Search => Activity 아래 구현은 검색 활동을 보여주며, 다음 스크린샷은 100% 캐시 적중률 아래에 120개 동시 희소 검색의 세부 정보를 보여줍니다.


Search Activity by User (1)

User	Search Count	Search Head Count	Median Runtime	Cumulative Runtime	Last Search
admin	120	3	0.93s	1min 52.41s	07/18/2019 23:16:33 -0700

Click to see a list of search head names and a list of search strings.

Search Activity by Search Head (3)

Search Head	Search Count	User Count	Median Runtime	Cumulative Runtime	Last Search
splunk-sh01	40	1	0.87s	35.73s	07/18/2019 23:16:33 -0700
splunk-sh02	40	1	0.95s	38.55s	07/18/2019 23:16:33 -0700
splunk-sh03	40	1	0.93s	38.13s	07/18/2019 23:16:33 -0700

Click to see a list of users and a list of search strings.

그림 22: 100% 캐시 적중률 아래 보여지는 120개 동시 희소 검색의 세부 정보

그림 23은 플래시어레이에서 호스팅되는 핫/웜 티어에 대해 120개의 동시 검색을 실행한 결과를 보여 줍니다. 각 희소 검색은 9,820만 개 이벤트 중 9,820개 이벤트를 반환하는 데 0.93초가 걸렸고, 희귀 검색은 3,270만 개 이벤트 중 2개를 반환하는 데 0.28초가 걸렸습니다.

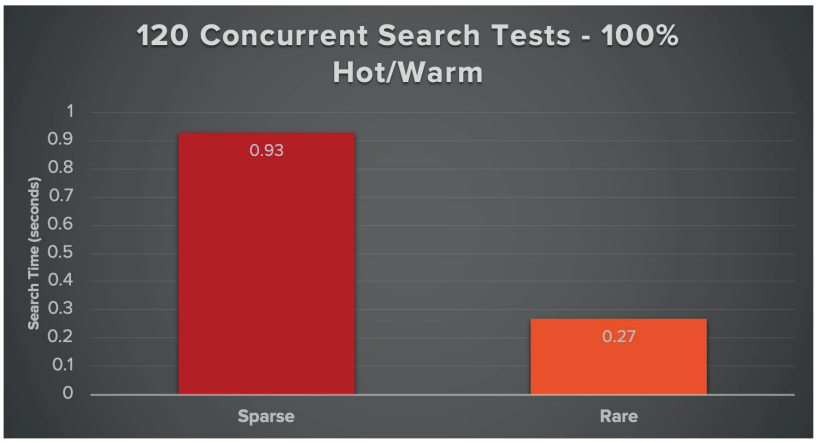


그림 23: 플래시어레이에서 호스팅되는 핫/웜 티어에 대해 120개의 동시 검색 실행 결과.

그림 24는 핫/웜 티어 100%와 핫/웜 티어 90% 및 콜드 티어 10%에 대한 희소 및 희귀 검색의 실행 시간을 보여 줍니다. 이는 120개 검색 중 90%인 108개가 핫/웜 티어 데이터에 대해 실행되었고, 나머지 10%(12개 검색)는 콜드 티어 데이터에 대해 실행되었다는 것을 의미합니다.

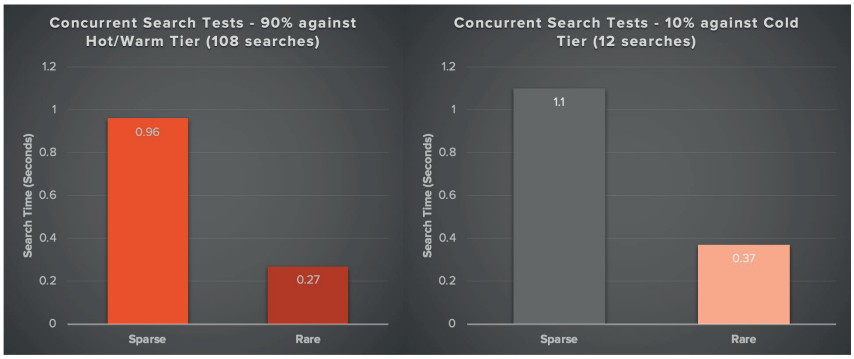


그림 24: 핫/웜 티어 100%, 핫/웜 티어 90%와 콜드 티어 10%에 대한 희소 및 희귀 검색 실행 시간

위의 그래프에서 보듯이 핫/웜 티어 100%와 90%에 대한 희소 및 희귀 검색은 큰 차이가 없었습니다. 플래시어레이에서 모두 읽기 I/O가 수행되기 때문입니다. 플래시블레이드에서 수행된 콜드 티어 10%에 대한 검색은 완료하는 데 1.1초가 걸려 핫/웜 티어의 응답 시간인 0.96초보다 약간 높았지만, 각 희소 검색이 데이터세트 180GB에 대해 수행된다는 점을 감안하면 이는 거의 동등한 수치라 할 만합니다.

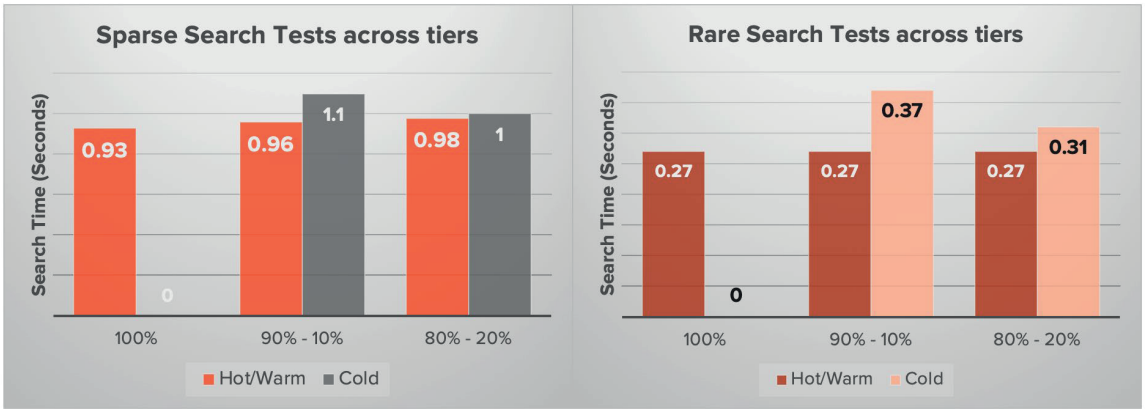


그림 25: 티어 간 희귀 검색 vs. 희소 검색 테스트.

위의 그래프에 표시된 검색 테스트 결과를 보면, 검색 유형에 관계없이 콜드 티어에 대한 검색 성능이 핫/웜 티어에 대한 검색 성능과 거의 비슷하다는 것을 알 수 있습니다. 이는 콜드 티어가 퓨어스토리지의 올플래시 스토리지 시스템에 배치될 때 더 이상 콜드가 아니라는 점을 다시 한번 입증해줍니다.

이러한 모든 동시 검색 중 인덱서 클러스터의 CPU 사용률은 7%를 넘지 않았습니다. 다음 그래프는 6개 테스트 모두에서의 CPU 사용률을 보여줍니다.

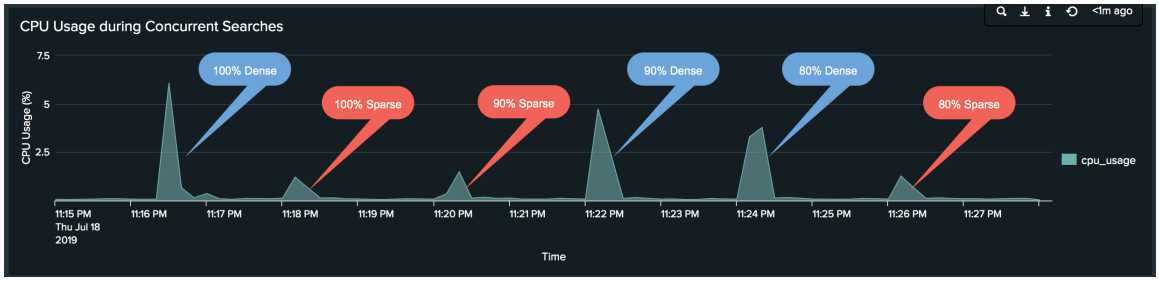


그림 26: 6개 테스트에서 CPU 사용률..

핫/웜 티어에 플래시어레이를 사용하고 콜드 티어에 플래시블레이드를 사용할 경우, Splunk Enterprise는 Splunk 내의 모든 티어에서 검색 성능을 가속화합니다.

모범 사례

퓨어스토리지 플래시어레이

플래시어레이 볼륨

- 항상 핫/웜 티어를 호스팅하는 모든 인덱서에 대해 별도의 플래시어레이 볼륨을 생성합니다.
- 플래시어레이 볼륨은 항상 씬 프로비저닝되므로, Splunk 관리자는 대규모 볼륨을 프로비저닝하여 추후 공간을 늘리기 위해 볼륨을 추가할 필요가 없습니다.
- 클러스터의 모든 인덱서에 대한 모든 플래시어레이 볼륨을 동일한 크기로 유지합니다.

논리적 볼륨 관리자

인덱서 수준에서 논리적 볼륨 관리자(Logical Volume Manager, LVM)를 사용하여 플래시어레이 볼륨을 볼륨 그룹에 연결하고 거기에 핫/웜 티어의 논리적 볼륨을 구성하는 것이 좋습니다. 이렇게 하면 인덱서에 핫/웜 티어에 대한 저장 공간이 더 필요할 때 동적으로 스토리지를 추가할 수 있습니다.

플래시어레이 볼륨에서의 Linux 마운트 옵션

Splunk 인덱서에 EXT4 및 XFS 파일 시스템을 모두 사용하는 경우를 검증했습니다. 버킷이 노후되고 디렉터리가 제거되면, DISCARD 마운트 옵션으로 퓨어스토리지 플래시어레이에 대한 TRIM 명령을 실행하여 해당 디렉터리가 차지하는 공간을 해제합니다. 권장되는 마운트 옵션은 다음과 같습니다.

```
discard,noatime
```

표준 운영 절차에서 discard 옵션을 선호하지 않는 경우 **fstrim** 명령을 하루에 한 번 또는 일주일에 한 번처럼 주기적으로 실행해 플래시어레이 수준에서 공간을 확보해야 합니다.

Linux

퓨어스토리지 플래시어레이를 위한 Linux 권장 설정은 퓨어스토리지 지원 사이트의 솔루션 페이지에 설명되어 있습니다. Linux를 사용하는 인덱서 노드에 플래시어레이 볼륨을 연결할 때도 동일한 절차를 따릅니다.

퓨어스토리지 플래시블레이드

플래시블레이드 파일 시스템

- 콜드 티어를 호스팅하는 모든 인덱서에 대해 항상 별도의 NFS 파일 시스템을 생성합니다.
- 플래시블레이드 파일 시스템은 항상 씬 프로비저닝되므로, Splunk 관리자는 대규모 파일 시스템을 프로비저닝함으로써 데이터 증가를 충족하기 위해 사이즈를 업데이트하지 않아도 됩니다.
- 파일 시스템 크기를 제한하는 매개 변수를 설정하지 않는 것이 좋습니다. 그러면 필요할 때 공간을 더 추가할 수 있는 유연성이 제한되기 때문입니다.
- 클러스터의 모든 인덱서에 대한 모든 NFS 파일 시스템을 동일한 크기로 유지합니다.

Linux 마운트 옵션

다음 마운트 옵션을 사용하여 콜드 티어의 인덱서 노드에 NFS 파일 시스템을 마운트합니다.

```
rw,bg,nointr,hard,tcp,vers=3,rsize=16384
```

호스트가 플래시블레이드에서 제공하는 기본 크기(512K)를 가져올 수 있으므로 wsize 옵션을 지정하지 마십시오.

Splunk 구성

버킷 크기

Splunk에는 버킷 크기가 미리 정해져 있으며, 이 크기는 indexes.conf maxDataSize = <positive integer>|auto|auto_high_volume의 maxDataSize 매개 변수 아래에서 구성할 수 있습니다

```
maxDataSize = <positive integer>|auto|auto_high_volume
```

기본값은 750MB에서 'auto'이며, auto_high_volume은 64비트 시스템에서 10GB, 32비트 시스템에서 1GB입니다. Splunk에서 대용량 환경에 대해 권장하는 일반적인 방법은 버킷 크기를 auto_high_volume으로 설정하는 것입니다.

권장 설정: maxDataSize=auto_high_volume(엔터프라이즈 규모)

TSIDX 감소

퓨어스토리지 시스템은 추가적인 데이터 절감을 제공해 용량 걱정을 할 필요가 없기 때문에 TSIDX를 최대한 활용하는 것이 좋습니다. 따라서 매개변수 enableTsidxReduction을 'true'로 설정하지 마십시오.

권장 설정: enableTsidxReduction=false

Bloom 필터

Splunk의 Bloom 필터는 검색 동작에서 중요한 역할을 하기 때문에 활성화하는 것이 좋습니다.

권장 설정: createBloomfilter=true.

결론

Splunk는 보안 정보 및 이벤트 관리(SIEM) 부문에서 시장을 선도하고 있으며 IT 운영 관리(ITOM) 부문 2위 기업입니다. Splunk 인프라에 부담을 주는 데이터 장기 보존을 원하고 더 많은 데이터 소스를 추가하려는 다양한 조직들 사이에서 Splunk 사용이 늘고 있습니다. Splunk 퓨어스토리지에 기반한 Splunk Enterprise 솔루션은 조직의 이러한 요구사항을 달성할 수 있도록 확장 가능한 아키텍처를 제공할 뿐만 아니라 다음과 같은 이점을 제공하도록 설계되었습니다.

- 서버와 스토리지를 독립적으로 확장하여 자산 활용도 향상 및 총소유비용(TCO) 절감
- 업계 최고의 인프라 구성 요소를 통해 인제스트 성능 향상
- 분산 스토리지를 사용해 스토리지 공간을 추가하는 등 운영 효율성 향상
- 플래시어레이 및 플래시블레이드에서 압축을 통해 더 큰 데이터 절감 효과 제공
- 플래시어레이 및 플래시블레이드에서 저장 데이터 암호화를 통해 추가 보안 제공
- 대규모 확장을 위해 설계된 퓨어스토리지 기반 Splunk Enterprise

Splunk 문서

- 용량 계획 매뉴얼: 성능 권장 사항 요약
- 용량 계획 매뉴얼: 참조 하드웨어
- 용량 계획 매뉴얼: Splunk Enterprise가 디스크 스토리지를 계산하는 방법

부록

부록 A: Splunk Enterprise 구성 요소

인덱스(Index)는 하위 디렉터리인 데이터베이스의 집합입니다. Splunk는 모든 인덱스 데이터를 플랫 파일 형식으로 관리하며 정교한 데이터베이스 관리 시스템을 사용하지 않습니다.

인덱서 클러스터(Indexer Cluster)는 서로의 데이터를 복제하도록 구성된 인덱서 그룹입니다. 이를 통해 시스템은 모든 데이터의 복사본을 여러 개 보관하여 데이터 손실을 방지하는 동시에, 인덱서 노드 장애 시 검색을 위한 데이터 가용성을 높일 수 있습니다. 인덱서 클러스터에는 자동 페일오버 기능이 있어 인덱서 장애 시, Splunk는 인덱서를 다음 인덱서로 자동 페일 오버하여 유입되는 데이터를 인덱싱하고 검색에 사용할 수 있도록 합니다.

검색 헤드(Search Head)는 유입되는 검색 요청을 처리합니다. 분산 검색 환경에서, 검색 헤드에서 인덱서 그룹 즉, '검색 피어'로 요청을 보내고, 인덱서 그룹은 해당 인덱스에 대한 실제 검색을 수행해 결과를 반환합니다. 검색 헤드는 결과를 병합하고 사용자에게 표시합니다. 전용 검색 헤드에는 통합 및 디스플레이와 같은 검색 관리 기능을 수행할 때 사용되는 전용 인덱스가 없습니다.

검색 헤드 클러스터(Search Head Cluster)는 구성, 작업 일정 및 검색 아티팩트를 공유하는 교환 가능한 고가용성 검색 헤드 그룹 즉, 클러스터 멤버(Cluster Members)입니다. 검색 헤드 클러스터는 동시 사용자 용량을 늘리고 단일 장애 지점을 제거하여 가용성과 확장성이 뛰어난 검색 서비스를 지원합니다.

포워더(Forwarder)는 Splunk 인스턴스의 소규모 풋프린트 버전으로, 데이터 처리 및 저장을 위해 데이터를 원격 인덱서로 전달합니다.

클러스터 마스터(Cluster Master) 또는 **마스터 노드(Master Node)**는 인덱서 클러스터의 기능을 조절하는 또 다른 인스턴스입니다.

구현자(Deployer)는 앱 및 기타 구성을 검색 헤드 클러스터 멤버들에 분산하는 Splunk Enterprise 인스턴스입니다. 구현자는 클러스터 멤버와 동일한 인스턴스에서 실행될 수 없습니다.

모니터링 콘솔(Monitoring Console)은 Splunk Enterprise 모니터링 툴로, 사전 정의된 대시보드를 통해 Splunk Enterprise 구축에 대한 자세한 성능 정보를 볼 수 있습니다. 사용 가능한 대시보드에는 인덱싱 성능, 인덱스 및 볼륨 사용, 라이선스 사용, 검색 성능, 검색 헤드 및 인덱서 클러스터링 등이 있습니다.

라이선스 마스터(License Master)는 하나 이상의 라이선스 슬레이브를 제어하며, 일반적으로 둘 이상의 인덱서가 있고 중앙 위치에서 구입한 라이선스 용량에 대한 인덱서 액세스를 관리하는 경우 사용합니다.

구현 서버(Deployment Server)는 중앙화된 구성 관리자 역할을 하는 Splunk Enterprise 인스턴스입니다. Splunk Enterprise 인스턴스 그룹들에 구성, 앱 및 콘텐츠 업데이트를 분산하는 툴입니다. 일반적으로 포워더, 클러스터링되지 않은 인덱서 및 검색 헤드를 업데이트하는 데 사용됩니다. 구현 서버는 포워더 및 기타 인스턴스를 관리하는 데 필요한 구성 요소는 아닙니다. 선호도 및/또는 표준 운영 절차에 따라 Chef, Puppet, Salt 등과 같은 툴을 사용할 수 있습니다.

부록 B: Cluster Shell 유틸리티 설치 방법

Cluster Shell **(clush)** 유틸리티를 사용하면 시스템 및 Splunk 관리자가 클러스터에서 명령을 병렬로 실행하여 Linux 클러스터를 보다 효율적으로 관리할 수 있습니다. 대화형으로 또는 쉘 스크립트 내에서 명령을 실행할 수 있습니다. **clush** 유틸리티를 사용하려면 ssh가 필요합니다.

clush를 설치하려면:

1. 다른 서버를 관리하는 데 사용되는 중앙 서버에 EPEL(Extra Packages for Enterprise Linux) 저장소를 설치합니다.
2. Cluster Shell 유틸리티를 설치합니다.

   참고: 이 문서의 설정에서 EPEL은 **splunk-cm01** 서버에서 설정되었으며 다른 모든 Splunk 노드를 관리하는 데 사용되었습니다.

```
[root@splunk-cm01 ~]# wget https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
[root@splunk-cm01 ~]# yum install -y epel-release-latest-7.noarch.rpm
```

3. 중앙 서버에서 clush를 통해 관리되는 모든 노드에 SSH 액세스(암호 없음)를 설정합니다.
4. (선택 사항) Splunk 사용자로서 모든 노드를 관리하려면 Splunk 사용자로 다음을 수행합니다. 이 예시에서는 루트 사용자를 사용했습니다.
   a. ssh-keygen 명령을 입력하여 암호 없이 액세스할 수 있도록 SSH 공개 키를 구성합니다.
   b. ssh-copy -id 명령을 실행하여 id_rsa.pub 파일을 모든 노드에 복사합니다.

```
[root@splunk-cm01 ~]$
Generating public/private rsa key pair.
Enter file in  which to save the key (/home/splunk/.ssh/id_rsa): Enter passphrase (empty
for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/splunk/.ssh/id_rsa. Your public key has been
saved in /home/splunk/.ssh/id_rsa.pub. The key fingerprint is:
66:5b:97:cc:5b:3e:f2:8a:d4:4a:91:11:88:e6:89:34 splunk@splunk-admin1 The key's randomart
image is:
+--[ RSA 2048]----+
|       . ..    |
|       E o . .|
|       . = . .|
|       . o  = .       |
|       S + = .|
|       o + +   |
|       . o + o|
|       o o .   |
|       o ...   |
+-----------------+
[root@splunk-cm01 ~]$
```

```
[root@splunk-cm01 ~]$ for host in cm01 ix01 ix02 ix03 ix04 ix05 ix06 ix07 ix08 sh01 sh02
sh03;
do
echo -n "$host -> ";
ssh-copy-id -I ~/.ssh/id_rsa.pub splunk-$host;
done
```

5. 노드 세부 정보와 함께 다음 파일을 수정하여 Cluster Shell에 대한 그룹 세부 정보를 구성합니다.

```
vi /etc/clustershell/groups
all: splunk-cm01,splunk-ix[01-08],splunk-sh[01-03]
ixs: splunk-ix[01-08]
shs: splunk-sh[01-03]
```

6. 명령을 실행하여 Cluster Shell 유틸리티가 작동하는지 확인합니다. 인덱서 노드에서만 명령을 실행하려면 지정된 그룹으로만 명령을 제한하는 '-g indexers' 인수를 사용합니다.

```
[root@splunk-cm01 local]# clush -a hostname
splunk-ix07: splunk-ix07
splunk-ix03: splunk-ix03
splunk-ix01: splunk-ix01
splunk-ix04: splunk-ix04
splunk-ix02: splunk-ix02
splunk-ix08: splunk-ix08
splunk-ix06: splunk-ix06
splunk-ix05: splunk-ix05
splunk-sh01: splunk-sh01
splunk-cm01: splunk-cm01
splunk-sh03: splunk-sh03
splunk-sh02: splunk-sh02
[root@splunk-cm01 local]# clush -g shs hostname
splunk-sh01: splunk-sh01
splunk-sh03: splunk-sh03
splunk-sh02: splunk-sh02
```

ClusterShell에 대한 보다 자세한 내용은 http://clustershell.readthedocs.io/en/latest/를 참조하세요.

부록 C: 유용한 Splunk 검색

핫 티어 공간 사용량

```
| dbinspect index=<index-name> cached=false | search state=hot | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024 |fields diskinGB
```

웜 티어 공간 사용량

```
| dbinspect index=<index-name> cached=false | search state=warm | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024
```

핫/웜 티어 공간 사용량

```
| dbinspect index=* cached=false | stats sum(sizeOnDiskMB) AS diskTotalinMB by state
```

인덱스의 버킷 세부 정보

```
|dbinspect index=<index-name>|eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S") |fields start_time, end_time, state, bucketId, eventCount
```

날짜 범위별 검색에 참여하는 인덱스별 요약된 버킷 세부 정보

다음 검색은 날짜 범위 05/20/19 08:00에서 05/20/19 20:00:00 사이의 데이터가 있는 버킷 세부 정보를 검색합니다. 인덱스 이름과 날짜 범위를 필요에 맞게 수정합니다.

```
|dbinspect index=apache-pure* |dedup bucketId| eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S")
| where ("05/20/19 08:00:00" >= start_time AND "05/20/19 08:00:00" <= end_time) OR ("05/20/19
```

부록 C: 유용한 Splunk 검색

핫 티어 공간 사용량

```
| dbinspect index=<index-name> cached=false | search state=hot | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024 |fields diskinGB
```

웜 티어 공간 사용량

```
| dbinspect index=<index-name> cached=false | search state=warm | stats sum(sizeOnDiskMB) AS diskTotalinMB
|eval diskinGB = diskTotalinMB/1024
```

핫/웜 티어 공간 사용량

```
| dbinspect index=* cached=false | stats sum(sizeOnDiskMB) AS diskTotalinMB by state
```

인덱스의 버킷 세부 정보

```
|dbinspect index=<index-name>|eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S") |fields start_time, end_time, state, bucketId, eventCount
```

날짜 범위별 검색에 참여하는 인덱스별 요약된 버킷 세부 정보

다음 검색은 날짜 범위 05/20/19 08:00에서 05/20/19 20:00:00 사이의 데이터가 있는 버킷 세부 정보를 검색합니다. 인덱스 이름과 날짜 범위를 필요에 맞게 수정합니다.

```
|dbinspect index=apache-pure* |dedup bucketId| eval start_time=strftime(startEpoch,"%m/%d/%y %H:%M:%S"),
end_time=strftime(endEpoch,"%m/%d/%y %H:%M:%S")
| where ("05/20/19 08:00:00" >= start_time AND "05/20/19 08:00:00" <= end_time) OR ("05/20/19 20:00:00" >=
start_time AND "05/20/19 20:00:00" <= end_time) OR ( start_time >= "05/20/19 08:00:00" AND end_time <=
"05/20/19 20:00:00")|stats sum(eventCount) as ec sum(sizeOnDiskMB) as mb count as Buckets by index |eval
diskinGB=round(mb/1024,2), EventCount=tostring(ec, "commas") |fields index, EventCount, diskinGB, Buckets .
   )
```


Pure Storage, Inc.
650 Castro Street, #400
Mountain View, CA 94041