Ubuntu 또는 Debian에서 설치를 계획하고 준비합니다

StorageGRID

NetApp
March 12, 2025

목차

Ubuntu 또는 Debian에서 설치를 계획하고 준비합니다

필요한 정보 및 자료

StorageGRID를 설치하기 전에 필요한 정보와 자료를 수집하고 준비합니다.

필수 정보입니다

네트워크 계획

각 StorageGRID 노드에 연결할 네트워크 StorageGRID는 트래픽 분리, 보안 및 관리의 편의를 위해 여러 네트워크를 지원합니다.

StorageGRID 를 "네트워킹 지침"참조하십시오.

네트워크 정보

각 그리드 노드에 할당할 IP 주소와 DNS 및 NTP 서버의 IP 주소입니다.

그리드 노드용 서버

구축할 StorageGRID 노드의 수와 유형을 지원하기에 충분한 리소스를 제공하는 물리적 서버 세트, 가상 서버 또는 둘 다 식별합니다.

> StorageGRID 설치에서 StorageGRID 어플라이언스(하드웨어) 스토리지 노드를 사용하지 않는 경우 BBWC(배터리 지원 쓰기 캐시)와 함께 하드웨어 RAID 스토리지를 사용해야 합니다. StorageGRID는 VSAN(Virtual Storage Area Network), 소프트웨어 RAID 또는 RAID 보호 사용을 지원하지 않습니다.

노드 마이그레이션(필요한 경우)

"노드 마이그레이션에 대한 요구사항"서비스 중단 없이 물리적 호스트에 대해 예약된 유지 관리를 수행하려는 경우을 이해합니다.

관련 정보

"NetApp 상호 운용성 매트릭스 툴"

필수 자료

NetApp StorageGRID 라이센스

디지털 서명된 유효한 NetApp 라이센스가 있어야 합니다.

> 테스트 및 개념 증명 그리드에 사용할 수 있는 비운영 라이센스가 StorageGRID 설치 아카이브에 포함되어 있습니다.

StorageGRID 설치 아카이브

"StorageGRID 설치 아카이브를 다운로드하고 파일 압축을 풉니다"..

서비스 노트북

StorageGRID 시스템은 서비스 랩톱을 통해 설치됩니다.

서비스 랩톱의 구성 요소:

- 네트워크 포트
- SSH 클라이언트(예: PuTTY)
- "지원되는 웹 브라우저"

StorageGRID 설명서

- "릴리스 정보"
- "StorageGRID 관리 지침"

StorageGRID 설치 파일을 다운로드하고 압축을 풉니다

StorageGRID 설치 아카이브를 다운로드하고 필요한 파일을 추출해야 합니다. 선택적으로 설치 패키지의 파일을 수동으로 확인할 수 있습니다.

단계

1. 로 이동합니다 "StorageGRID용 NetApp 다운로드 페이지".
2. 최신 릴리스를 다운로드하려면 버튼을 선택하거나 드롭다운 메뉴에서 다른 버전을 선택하고 * GO * 를 선택합니다.
3. NetApp 계정의 사용자 이름과 암호를 사용하여 로그인합니다.
4. Caution/MustRead 문이 나타나면 해당 문을 읽고 확인란을 선택합니다.

   StorageGRID 릴리스를 설치한 후 필요한 핫픽스를 적용해야 합니다. 자세한 내용은 를 참조하십시오 "복구 및 유지 관리 지침의 핫픽스 절차"

5. 최종 사용자 사용권 계약을 읽고 확인란을 선택한 다음 * 동의 및 계속 * 을 선택합니다.
6. StorageGRID 설치 * 열에서 Ubuntu 또는 Debian용 .tgz 또는 .zip 설치 아카이브를 선택합니다.

   `.zip`서비스 랩톱에서 Windows를 실행하는 경우 파일을 선택합니다.

7. 설치 아카이브를 저장합니다.
8. 설치 아카이브를 확인해야 하는 경우:
   a. StorageGRID 코드 서명 확인 패키지를 다운로드합니다. 이 패키지의 파일 이름은 StorageGRID 소프트웨어 버전의 형식을 `StorageGRID_<version-number>_Code_Signature_Verification_Package.tar.gz` 사용합니다. `<version-number>`
   b. 의 단계를 "설치 파일을 수동으로 확인합니다"따릅니다.
9. 설치 아카이브에서 파일 압축을 풉니다.
10. 필요한 파일을 선택합니다.

    필요한 파일은 계획된 그리드 토폴로지와 StorageGRID 시스템 배포 방법에 따라 다릅니다.

    표에 나열된 경로는 추출된 설치 아카이브에서 설치한 최상위 디렉토리에 상대적입니다.

경로 및 파일 이름입니다	설명
	StorageGRID 다운로드 파일에 포함된 모든 파일을 설명하는 텍스트 파일입니다.
/debs/NLF000000.txt 를 참조하십시오	테스트 및 개념 증명 배포에 사용할 수 있는 비프로덕션 NetApp 라이센스 파일.
/debs/storagegrid-webscale-images-version-SHA.deb 를 참조하십시오	StorageGRID 노드 이미지를 Ubuntu 또는 Debian 호스트에 설치하기 위한 DEB 패키지.
/debs/storagegrid-webscale-images-version-SHA.deb.md5 를 참조하십시오	파일의 MD5 체크섬 `/debs/storagegrid-webscale-images-version-SHA.deb`.
/debs/storagegrid-webscale-service-version-SHA.deb 를 참조하십시오	Ubuntu 또는 Debian 호스트에 StorageGRID 호스트 서비스를 설치하기 위한 DEB 패키지.
배포 스크립팅 도구	설명
/debs/configure-storagegrid.py 를 참조하십시오	StorageGRID 시스템 구성을 자동화하는 데 사용되는 Python 스크립트입니다.
/debs/configure-sga.py 를 참조하십시오	StorageGRID 어플라이언스 구성을 자동화하는 데 사용되는 Python 스크립트입니다.
/debs/storagegrid-ssoauth.py 를 참조하십시오	SSO(Single Sign-On)가 활성화된 경우 Grid Management API에 로그인하는 데 사용할 수 있는 Python 스크립트 예제 이 스크립트를 Ping 연합 통합에 사용할 수도 있습니다.
/debs/configure -StorageGrid.sample.json 을 참조하십시오	스크립트와 함께 사용할 예제 구성 파일 `configure-storagegrid.py`
/debs/configure -StorageGrid.blank.json 을 참조하십시오	스크립트와 함께 사용할 빈 구성 `configure-storagegrid.py` 파일입니다.
	StorageGRID 컨테이너 배포를 위한 Ubuntu 또는 Debian 호스트 구성을 위한 Ansible 역할 및 플레이북 예 필요에 따라 역할 또는 플레이북을 사용자 지정할 수 있습니다.
	Active Directory 또는 Ping 연방을 사용하여 SSO(Single Sign-On)를 사용하도록 설정한 경우 Grid Management API에 로그인하는 데 사용할 수 있는 Python 스크립트 예제

경로 및 파일 이름입니다	설명
/debs/StorageGrid-ssoauth-Azure.js를 입력합니다	Azure와의 SSO 상호 작용을 수행하기 위해 Python 스크립트에 의해 호출되는 도우미 스크립트입니다. `storagegrid-ssoauth-azure.py`
/debs/Extras/API-schemas	StorageGRID에 대한 API 스키마입니다. • 참고 *: 업그레이드를 수행하기 전에 이러한 스키마를 사용하여 StorageGRID 관리 API를 사용하도록 작성한 코드가 업그레이드 호환성 테스트를 위한 비프로덕션 StorageGRID 환경이 없는 경우 새 StorageGRID 릴리스와 호환되는지 확인할 수 있습니다.

설치 파일 수동 확인(선택 사항)

필요한 경우 StorageGRID 설치 아카이브의 파일을 수동으로 확인할 수 있습니다.

시작하기 전에

에서 "StorageGRID용 NetApp 다운로드 페이지" 가져온 "검증 패키지를 다운로드했습니다"것입니다.

단계

1. 검증 패키지에서 아티팩트를 추출합니다.

```
tar -xf StorageGRID_11.9.0_Code_Signature_Verification_Package.tar.gz
```

2. 이러한 아티팩트가 추출되었는지 확인합니다.
   - Leaf 인증서: `Leaf-Cert.pem`
   - 인증서 체인: `CA-Int-Cert.pem`
   - 타임 스탬프 응답 체인: `TS-Cert.pem`
   - 체크섬 파일: `sha256sum`
   - 체크섬 서명: `sha256sum.sig`
   - 타임 스탬프 응답 파일: `sha256sum.sig.tsr`
3. 체인을 사용하여 리프 인증서가 유효한지 확인합니다.
   - 예 *: `openssl verify -CAfile CA-Int-Cert.pem Leaf-Cert.pem`
   - 예상 출력 *: `Leaf-Cert.pem: OK`
4. leaf 인증서가 만료되어 step_2_에 실패한 경우 파일을 사용하여 `tsr` 확인합니다.
   - 예 *: `openssl ts -CAfile CA-Int-Cert.pem -untrusted TS-Cert.pem -verify -data sha256sum.sig -in sha256sum.sig.tsr`
   - 예상 출력 포함 *: `Verification: OK`
5. 리프 인증서에서 공용 키 파일을 만듭니다.

- 예 *: `openssl x509 -pubkey -noout -in Leaf-Cert.pem > Leaf-Cert.pub`
- 예상 출력 *:_none_

6. 공개 키를 사용하여 `sha256sum` 에 대해 파일을 `sha256sum.sig` 확인합니다.
   - 예 *: `openssl dgst -sha256 -verify Leaf-Cert.pub -signature sha256sum.sig sha256sum`
   - 예상 출력 *: `Verified OK`
7. `sha256sum`새로 생성된 체크섬을 기준으로 파일 내용을 확인합니다.
   - 예 *: `sha256sum -c sha256sum`

   *예상 출력 * *<filename>*: OK:+는 *<filename>* 다운로드한 아카이브 파일의 이름입니다.

8. "나머지 단계를 완료합니다" 를 눌러 적절한 설치 파일을 추출하고 선택합니다.

Ubuntu 및 Debian용 소프트웨어 요구 사항

가상 머신을 사용하여 모든 유형의 StorageGRID 노드를 호스팅할 수 있습니다. 각 그리드 노드에 대해 하나의 가상 머신이 필요합니다.

Ubuntu 또는 Debian에 StorageGRID를 설치하려면 타사 소프트웨어 패키지를 설치해야 합니다. 지원되는 일부 Linux 배포판에는 기본적으로 이러한 패키지가 포함되어 있지 않습니다. StorageGRID 설치를 테스트하는 소프트웨어 패키지 버전에는 이 페이지에 나열된 버전이 포함됩니다.

이러한 패키지를 필요로 하는 Linux 배포 및 컨테이너 런타임 설치 옵션을 선택했는데 Linux 배포판에 의해 자동으로 설치되지 않은 경우, 해당 공급자 또는 Linux 배포판의 지원 공급업체에서 제공하는 경우 여기에 나열된 버전 중 하나를 설치하십시오. 그렇지 않으면 공급업체에서 제공하는 기본 패키지 버전을 사용하십시오.

모든 설치 옵션에는 Podman 또는 Docker가 필요합니다. 두 패키지를 모두 설치하지 마십시오. 설치 옵션에 필요한 패키지만 설치합니다.

> ⓘ 소프트웨어 전용 배포를 위한 컨테이너 엔진으로 Docker에 대한 지원은 더 이상 사용되지 않습니다. Docker는 향후 릴리즈에서 다른 컨테이너 엔진으로 대체될 예정입니다.

Python 버전을 테스트했습니다

- 3.5.2-2
- 3.6.8-2
- 3.6.8-38
- 3.6.9-1
- 3.7.3-1
- 3.8.10-0
- 3.9.2-1
- 3.9.10-2
- 3.9.16-1

- 3.10.6-1
- 3.11.2-6

Podman 버전을 테스트했습니다

- 3.2.3-0
- 3.4.4 + DS1
- 4.1.1-7
- 4.2.0-11
- 4.3.1+DS1-8+B1
- 4.4.1-8
- 4.4.1-12

Docker 버전을 테스트했습니다

Docker 지원은 더 이상 사용되지 않으며 향후 릴리즈에서 제거될 예정입니다.

- Docker-CE 20.10.7
- Docker-CE 20.10.20-3 을 참조하십시오
- Docker-CE 23.0.6-1 을 참조하십시오
- Docker-CE 24.0.2-1 을 참조하십시오
- Docker-CE 24.0.4-1 을 참조하십시오
- Docker-CE 24.0.5-1 을 참조하십시오
- Docker-CE 24.0.7-1 을 참조하십시오
- 1.5-2 을 참조하십시오

CPU 및 RAM 요구 사항

StorageGRID 소프트웨어를 설치하기 전에 StorageGRID 시스템을 지원할 준비가 되도록 하드웨어를 확인 및 구성하십시오.

각 StorageGRID 노드에는 다음과 같은 최소 리소스가 필요합니다.

- CPU 코어: 노드당 8개
- RAM: 사용 가능한 총 RAM과 시스템에서 실행되는 비 StorageGRID 소프트웨어의 양에 따라 다릅니다
  - 일반적으로 노드당 최소 24GB, 총 시스템 RAM보다 2 ~ 16GB 작습니다
  - 각 테넌트당 최소 64GB의 버킷이 약 5,000개 있습니다

소프트웨어 기반 메타데이터 전용 노드 리소스는 기존 스토리지 노드 리소스와 일치해야 합니다. 예를 들면 다음과 같습니다.

- 기존 StorageGRID 사이트에서 SG6000 또는 SG6100 어플라이언스를 사용 중인 경우 소프트웨어 기반

메타데이터 전용 노드가 다음과 같은 최소 요구사항을 충족해야 합니다.

  - 128GB RAM
  - 8코어 CPU
  - Cassandra 데이터베이스용 8TB SSD 또는 동급 스토리지(rangedb/0)

- 기존 StorageGRID 사이트에서 24GB RAM, 8코어 CPU 및 3TB 또는 4TB의 메타데이터 스토리지를 사용하는 가상 스토리지 노드를 사용하는 경우 소프트웨어 기반 메타데이터 전용 노드에서 유사한 리소스(24GB RAM, 8코어 CPU 및 4TB 메타데이터 스토리지(rangedb/0)를 사용해야 합니다.

새 StorageGRID 사이트를 추가할 때 새 사이트의 총 메타데이터 용량은 최소한 기존 StorageGRID 사이트와 일치해야 하며 새 사이트 리소스는 기존 StorageGRID 사이트의 스토리지 노드와 일치해야 합니다.

각 물리적 또는 가상 호스트에서 실행하려는 StorageGRID 노드 수가 사용 가능한 CPU 코어 수 또는 물리적 RAM을 초과하지 않는지 확인합니다. 호스트가 StorageGRID 실행 전용이 아닌 경우(권장되지 않음) 다른 애플리케이션의 리소스 요구 사항을 고려해야 합니다.

> CPU 및 메모리 사용량을 정기적으로 모니터링하여 이러한 리소스가 작업 부하를 지속적으로 수용할 수 있도록 합니다. 예를 들어, 가상 스토리지 노드에 대한 RAM 및 CPU 할당을 두 배로 하면 StorageGRID 어플라이언스 노드에 제공되는 것과 유사한 리소스를 제공할 수 있습니다. 또한 노드당 메타데이터 양이 500GB를 초과하는 경우 노드당 RAM을 48GB 이상으로 늘리는 것이 좋습니다. 개체 메타데이터 스토리지 관리, 메타데이터 예약 공간 설정 증가, CPU 및 메모리 사용량 모니터링에 대한 자세한 내용은"관리" "모니터링", 및 "업그레이드 중"StorageGRID에 대한 지침을 참조하십시오.

하이퍼스레딩이 기본 물리적 호스트에서 활성화된 경우 노드당 8개의 가상 코어(4개의 물리적 코어)를 제공할 수 있습니다. 하이퍼스레딩이 기본 물리적 호스트에서 사용되지 않는 경우 노드당 8개의 물리적 코어를 제공해야 합니다.

가상 시스템을 호스트로 사용하고 VM의 크기와 수를 제어하는 경우 각 StorageGRID 노드에 대해 단일 VM을 사용하고 그에 따라 VM 크기를 조정해야 합니다.

운영 구축 환경에서는 동일한 물리적 스토리지 하드웨어 또는 가상 호스트에서 여러 스토리지 노드를 실행하지 않아야 합니다. 단일 StorageGRID 구축 환경의 각 스토리지 노드는 자체 격리된 장애 도메인에 있어야 합니다. 단일 하드웨어 장애가 단일 스토리지 노드에만 영향을 줄 수 있도록 하는 경우 오브젝트 데이터의 내구성과 가용성을 최대화할 수 있습니다.

도 "요구사항을 충족해야 합니다"참조하십시오.

요구사항을 충족해야 합니다

초기 구성과 향후 스토리지 확장을 지원할 충분한 공간을 제공할 수 있도록 StorageGRID 노드의 스토리지 요구사항을 이해해야 합니다.

StorageGRID 노드에는 다음과 같은 세 가지 논리적 스토리지 범주가 필요합니다.

- StorageGRID 노드를 지원할 호스트에 Docker를 설치 및 구성할 때 Docker 스토리지 드라이버에 할당되는 노드 컨테이너용 * 컨테이너 풀 * — 성능 계층(10K SAS 또는 SSD) 스토리지입니다.
- * 시스템 데이터 * — StorageGRID 호스트 서비스가 사용하고 개별 노드에 매핑하는 시스템 데이터 및 트랜잭션 로그의 노드당 영구 스토리지를 위한 성능 계층(10K SAS 또는 SSD) 스토리지입니다.
- * 오브젝트 데이터 * — 객체 데이터 및 객체 메타데이터의 영구 스토리지를 위한 Performance-Tier(10K SAS 또는 SSD) 스토리지 및 Capacity-Tier(NL-SAS/SATA) 대용량 스토리지

모든 스토리지 범주에 RAID 지원 블록 장치를 사용해야 합니다. 비중복 디스크, SSD 또는 JBOD는 지원되지 않습니다. 모든 스토리지 범주에서 공유 또는 로컬 RAID 스토리지를 사용할 수 있지만 StorageGRID의 노드 마이그레이션 기능을 사용하려면 시스템 데이터와 오브젝트 데이터를 모두 공유 스토리지에 저장해야 합니다. 자세한 내용은 을 "노드 컨테이너 마이그레이션 요구사항"참조하십시오.

성능 요구사항

컨테이너 풀, 시스템 데이터 및 오브젝트 메타데이터에 사용되는 볼륨의 성능은 시스템의 전반적인 성능에 큰 영향을 미칩니다. 이러한 볼륨에 성능 계층(10K SAS 또는 SSD) 스토리지를 사용하면 지연 시간, IOPS(초당 입출력 작업) 및 처리량 측면에서 디스크 성능이 적절하게 보장됩니다. 객체 데이터의 영구 스토리지를 위해 용량 계층(NL-SAS/SATA) 스토리지를 사용할 수 있습니다.

컨테이너 풀, 시스템 데이터 및 오브젝트 데이터에 사용되는 볼륨에는 다시 쓰기 캐시가 설정되어 있어야 합니다. 캐시는 보호되거나 영구 미디어에 있어야 합니다.

NetApp ONTAP 스토리지를 사용하는 호스트의 요구 사항입니다

StorageGRID 노드가 NetApp ONTAP 시스템에서 할당된 스토리지를 사용하는 경우 볼륨에 FabricPool 계층화 정책이 활성화되어 있지 않은지 확인합니다. StorageGRID 노드와 함께 사용되는 볼륨에 대해 FabricPool 계층화를 사용하지 않도록 설정하면 문제 해결과 스토리지 작업이 간소화됩니다.

> FabricPool를 사용하여 StorageGRID 관련 데이터를 StorageGRID 자체로 계층화하지 마십시오. StorageGRID 데이터를 StorageGRID로 다시 계층화하면 문제 해결과 운영 복잡성이 늘어납니다.

필요한 호스트 수입니다

각 StorageGRID 사이트에는 최소 3개의 스토리지 노드가 필요합니다.

> 운영 구축 시 단일 물리적 호스트 또는 가상 호스트에서 스토리지 노드를 두 개 이상 실행하지 마십시오. 각 스토리지 노드에 대해 전용 호스트를 사용하면 격리된 장애 도메인이 제공됩니다.

관리 노드 또는 게이트웨이 노드와 같은 다른 유형의 노드는 동일한 호스트에 구축하거나 필요에 따라 전용 호스트에 구축할 수 있습니다.

각 호스트의 스토리지 볼륨 수입니다

다음 표에는 각 호스트에 필요한 스토리지 볼륨(LUN) 수와 해당 호스트에 구축할 노드를 기준으로 각 LUN에 필요한 최소 크기가 나와 있습니다.

테스트된 최대 LUN 크기는 39TB입니다.

> 이러한 숫자는 전체 그리드가 아닌 각 호스트에 대한 것입니다.

LUN 사용 목적	스토리지 범주입니다	**LUN** 수입니다	최소 크기/**LUN**
컨테이너 엔진 스토리지 풀입니다	컨테이너 풀입니다	1	총 노드 수 × 100GB

LUN 사용 목적	스토리지 범주입니다	**LUN** 수입니다	최소 크기/**LUN**
`/var/local` 볼륨	시스템 데이터	이 호스트의 각 노드에 대해 1개	90GB
스토리지 노드	오브젝트 데이터	이 호스트의 각 스토리지 노드에 대해 3개 • 참고: * 소프트웨어 기반 스토리지 노드는 1-16개의 스토리지 볼륨을 가질 수 있습니다. 최소 3개의 스토리지 볼륨을 사용하는 것이 좋습니다.	12TB(4TB/LUN) 자세한 내용은 을 참조하십시오스토리지 노드의 스토리지 요구 사항.
스토리지 노드 (메타데이터만)	오브젝트 메타데이터	1	4TB 자세한 내용은 을 스토리지 노드의 스토리지 요구 사항참조하십시오. • 참고 *: 메타데이터 전용 스토리지 노드에는 하나의 rangedb만 필요합니다.
관리자 노드 감사 로그	시스템 데이터	이 호스트의 각 관리 노드에 대해 1개	200GB
관리자 노드 테이블	시스템 데이터	이 호스트의 각 관리 노드에 대해 1개	200GB

구성된 감사 레벨에 따라 S3 오브젝트 키 이름 등의 사용자 입력 크기, 그리고 보존해야 하는 감사 로그 데이터의 양을 위해 각 관리 노드에서 감사 로그 LUN의 크기를 늘려야 할 수도 있습니다. 일반적으로 그리드는 S3 작업당 약 1KB의 감사 데이터를 생성합니다. 즉, 200GB LUN이 2일에서 3일 동안 매일 7천만 개의 작업 또는 초당 800개의 작업을 지원하게 됩니다.

호스트의 최소 스토리지 공간입니다

다음 표에는 각 노드 유형에 필요한 최소 스토리지 공간이 나와 있습니다. 이 표를 사용하여 각 스토리지 범주에서 호스트에 구축해야 하는 최소 스토리지 양을 해당 호스트에 구축될 노드를 기반으로 결정할 수 있습니다.

디스크 스냅샷을 사용하여 그리드 노드를 복원할 수 없습니다. 대신 "그리드 노드 복구"각 노드 유형에 대한 절차를 참조하십시오.

노드 유형입니다	컨테이너 풀입니다	시스템 데이터	오브젝트 데이터
스토리지 노드	100GB	90GB	4,000GB
관리자 노드	100GB	490GB(LUN 3개)	_해당 사항 없음 _

노드 유형입니다	컨테이너 풀입니다	시스템 데이터	오브젝트 데이터
게이트웨이 노드	100GB	90GB	_해당 사항 없음 _

예: 호스트에 대한 스토리지 요구 사항 계산

동일한 호스트에 스토리지 노드 1개, 관리 노드 1개, 게이트웨이 노드 1개 등 3개의 노드를 구축하려고 한다고 가정해 보겠습니다. 호스트에 최소 9개의 스토리지 볼륨을 제공해야 합니다. 노드 컨테이너용 300GB 이상의 성능 계층 스토리지, 시스템 데이터 및 트랜잭션 로그용 670GB 성능 계층 스토리지, 오브젝트 데이터를 위한 12TB의 용량 계층 스토리지가 필요합니다.

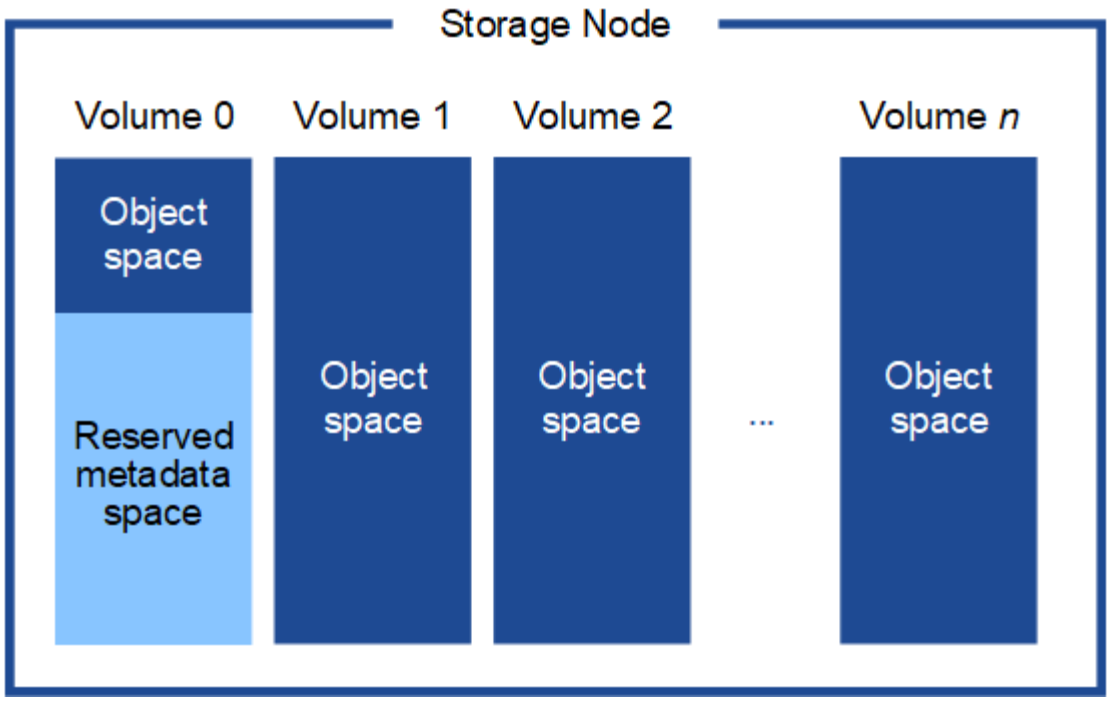
노드 유형입니다	**LUN** 사용 목적	**LUN** 수입니다	**LUN** 크기입니다
스토리지 노드	Docker 스토리지 풀	1	300GB(100GB/노드)
스토리지 노드	`/var/local` 볼륨	1	90GB
스토리지 노드	오브젝트 데이터	3	12TB(4TB/LUN)
관리자 노드	`/var/local` 볼륨	1	90GB
관리자 노드	관리자 노드 감사 로그	1	200GB
관리자 노드	관리자 노드 테이블	1	200GB
게이트웨이 노드	`/var/local` 볼륨	1	90GB
• 합계 *		• 9 *	• 컨테이너 풀: * 300GB • 시스템 데이터: * 670GB • 오브젝트 데이터: * 12,000GB

스토리지 노드의 스토리지 요구 사항

소프트웨어 기반 스토리지 노드는 1-16개의 스토리지 볼륨을 가질 수 있습니다. -3개 이상의 스토리지 볼륨을 사용하는 것이 좋습니다. 각 스토리지 볼륨은 4TB 이상이어야 합니다.

어플라이언스 스토리지 노드는 최대 48개의 스토리지 볼륨을 가질 수 있습니다.

그림에 나와 있는 것처럼 StorageGRID는 각 스토리지 노드의 스토리지 볼륨 0에 객체 메타데이터를 위한 공간을 예약합니다. 스토리지 볼륨 0 및 스토리지 노드의 다른 스토리지 볼륨의 나머지 공간은 오브젝트 데이터에만 사용됩니다.



이중화를 제공하고 개체 메타데이터를 손실로부터 보호하기 위해 StorageGRID는 각 사이트의 시스템 모든 개체에 대한 메타데이터 복사본을 3개 저장합니다. 오브젝트 메타데이터의 복사본 3개는 각 사이트의 모든 스토리지 노드에 균등하게 분산됩니다.

메타데이터 전용 스토리지 노드가 있는 그리드를 설치할 경우 그리드에는 오브젝트 스토리지용 최소 노드 수도 있어야 합니다. 메타데이터 전용 스토리지 노드에 대한 자세한 내용은 을 "스토리지 노드 유형"참조하십시오.

- 단일 사이트 그리드의 경우 객체 및 메타데이터에 대해 2개 이상의 스토리지 노드가 구성됩니다.
- 다중 사이트 그리드의 경우 사이트당 하나 이상의 스토리지 노드가 객체 및 메타데이터에 대해 구성됩니다.

새 스토리지 노드의 볼륨 0에 공간을 할당하는 경우 모든 오브젝트 메타데이터의 해당 노드에 적절한 공간이 있는지 확인해야 합니다.

- 적어도 볼륨 0에 4TB 이상을 할당해야 합니다.

  스토리지 노드에 대해 하나의 스토리지 볼륨만 사용하고 볼륨에 4TB 이하의 용량을 할당하면 스토리지 노드가 시작 시 스토리지 읽기 전용 상태로 전환되고 객체 메타데이터만 저장할 수 있습니다.

  볼륨 0에 500GB 미만의 용량을 할당할 경우(비운영 전용) 스토리지 볼륨 용량의 10%가 메타데이터용으로 예약됩니다.

- 소프트웨어 기반 메타데이터 전용 노드 리소스는 기존 스토리지 노드 리소스와 일치해야 합니다. 예를 들면 다음과 같습니다.
  - 기존 StorageGRID 사이트에서 SG6000 또는 SG6100 어플라이언스를 사용 중인 경우 소프트웨어 기반 메타데이터 전용 노드가 다음과 같은 최소 요구사항을 충족해야 합니다.
    - 128GB RAM
    - 8코어 CPU

        - Cassandra 데이터베이스용 8TB SSD 또는 동급 스토리지(rangedb/0)
    - 기존 StorageGRID 사이트에서 24GB RAM, 8코어 CPU 및 3TB 또는 4TB의 메타데이터 스토리지를 사용하는 가상 스토리지 노드를 사용하는 경우 소프트웨어 기반 메타데이터 전용 노드에서 유사한 리소스(24GB RAM, 8코어 CPU 및 4TB 메타데이터 스토리지(rangedb/0)를 사용해야 합니다.

      새 StorageGRID 사이트를 추가할 때 새 사이트의 총 메타데이터 용량은 최소한 기존 StorageGRID 사이트와 일치해야 하며 새 사이트 리소스는 기존 StorageGRID 사이트의 스토리지 노드와 일치해야 합니다.

- 새 시스템(StorageGRID 11.6 이상)을 설치하고 각 스토리지 노드에 128MB 이상의 RAM이 있는 경우 볼륨 0에 8TB 이상을 할당합니다. 볼륨 0에 더 큰 값을 사용하면 각 스토리지 노드에서 메타데이터에 허용되는 공간이 증가할 수 있습니다.
- 사이트에 대해 서로 다른 스토리지 노드를 구성할 때 가능하면 볼륨 0에 대해 동일한 설정을 사용합니다. 사이트에 크기가 다른 스토리지 노드가 있는 경우 볼륨이 0인 스토리지 노드가 해당 사이트의 메타데이터 용량을 결정합니다.

자세한 내용은 을 "오브젝트 메타데이터 스토리지 관리"참조하십시오.

노드 컨테이너 마이그레이션 요구사항

노드 마이그레이션 기능을 사용하면 노드를 한 호스트에서 다른 호스트로 수동으로 이동할 수 있습니다. 일반적으로 두 호스트는 동일한 물리적 데이터 센터에 있습니다.

노드 마이그레이션을 통해 그리드 작업을 중단하지 않고 물리적 호스트 유지 관리를 수행할 수 있습니다. 물리적 호스트를 오프라인으로 전환하기 전에 한 번에 하나씩 모든 StorageGRID 노드를 다른 호스트로 이동합니다. 노드를 마이그레이션하려면 각 노드의 다운타임만 짧고 그리드 서비스의 운영 또는 가용성에 영향을 미치지 않아야 합니다.

StorageGRID 노드 마이그레이션 기능을 사용하려면 배포가 추가 요구 사항을 충족해야 합니다.

- 단일 물리적 데이터 센터의 호스트 전반에서 일관된 네트워크 인터페이스 이름
- 단일 물리적 데이터 센터의 모든 호스트에서 액세스할 수 있는 StorageGRID 메타데이터 및 오브젝트 저장소 볼륨을 위한 공유 스토리지입니다. 예를 들어, NetApp E-Series 스토리지 어레이를 사용할 수 있습니다.

가상 호스트를 사용 중이고 기본 하이퍼바이저 계층에서 VM 마이그레이션을 지원하는 경우 StorageGRID의 노드 마이그레이션 기능 대신 이 기능을 사용할 수 있습니다. 이 경우 이러한 추가 요구 사항을 무시할 수 있습니다.

마이그레이션 또는 하이퍼바이저 유지 보수를 수행하기 전에 노드를 정상적으로 종료합니다. 의 지침을 "그리드 노드 종료"참조하십시오.

VMware Live Migration은 지원되지 않습니다

VMware VM에서 베어 메탈 설치를 수행할 때 OpenStack Live Migration 및 VMware Live vMotion을 사용하면 가상 머신 클록 시간이 증가하며 어떠한 유형의 그리드 노드에서도 지원되지 않습니다. 드물지만 잘못된 클럭 시간으로 인해 데이터 또는 구성 업데이트가 손실될 수 있습니다.

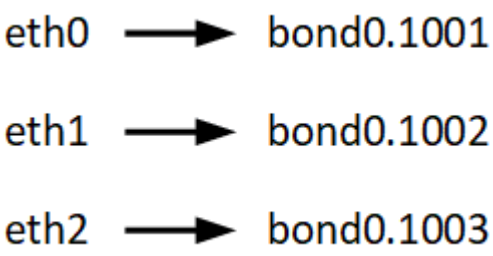
콜드 마이그레이션이 지원됩니다. 콜드 마이그레이션에서는 StorageGRID 노드를 호스트 간에 마이그레이션하기 전에 종료해야 합니다. 의 지침을 "그리드 노드 종료"참조하십시오.

일관된 네트워크 인터페이스 이름

한 호스트에서 다른 호스트로 노드를 이동하려면 StorageGRID 호스트 서비스가 노드가 현재 위치에 있는 외부

네트워크 연결을 새 위치에서 복제할 수 있다는 확신을 가져야 합니다. 호스트에서 일관된 네트워크 인터페이스 이름을 사용하면 이러한 자신감을 얻을 수 있습니다.

예를 들어 호스트 1에서 실행되는 StorageGRID NodeA가 다음과 같은 인터페이스 매핑으로 구성되었다고 가정합니다.



화살표의 왼쪽 면은 StorageGRID 컨테이너 내에서 보는 기존 인터페이스(즉, 그리드, 관리자 및 클라이언트 네트워크 인터페이스)에 해당합니다. 화살표의 오른쪽은 동일한 물리적 인터페이스 결합에 종속된 세 개의 VLAN 인터페이스인 이러한 네트워크를 제공하는 실제 호스트 인터페이스에 해당합니다.

이제 NodeA를 Host2로 마이그레이션한다고 가정해 보겠습니다. Host2에 bond0.1001, bond0.1002 및 bond0.1003이라는 인터페이스도 있는 경우 시스템은 Host1에서와 같이 같은 이름의 인터페이스가 Host2에서 동일한 연결을 제공한다고 가정하여 이동을 허용합니다. 호스트 2에 동일한 이름의 인터페이스가 없으면 이동이 허용되지 않습니다.

여러 호스트 간에 일관된 네트워크 인터페이스 이름을 지정하는 방법은 여러 가지가 있습니다. 몇 가지 예는 를 참조하십시오."호스트 네트워크를 구성합니다"

공유 스토리지

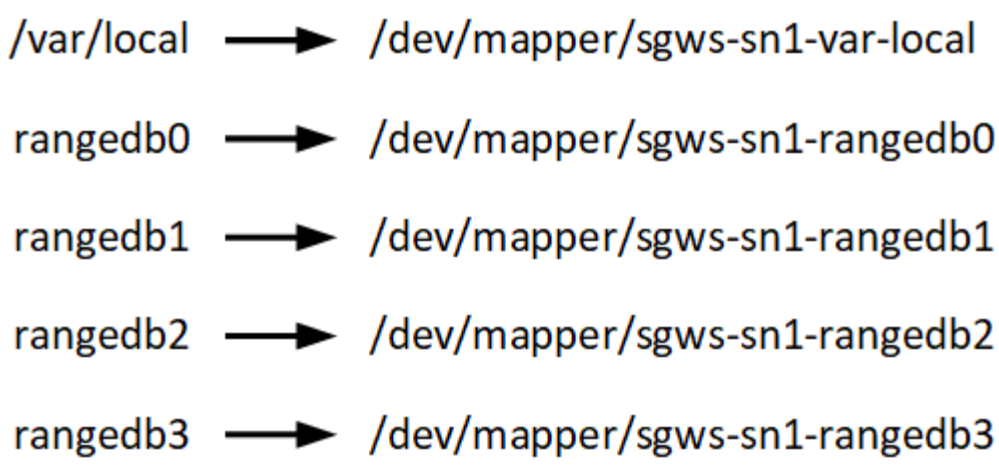
오버헤드가 낮은 노드를 신속하게 마이그레이션하기 위해 StorageGRID 노드 마이그레이션 기능은 노드 데이터를 물리적으로 이동하지 않습니다. 대신 노드 마이그레이션은 다음과 같이 한 쌍의 엑스포트 및 임포트 작업으로 수행됩니다.

단계

1. "노드 내보내기" 작업 중에 HostA에서 실행 중인 노드 컨테이너에서 소량의 영구 상태 데이터가 추출되고 해당 노드의 시스템 데이터 볼륨에 캐시됩니다. 그런 다음 HostA의 노드 컨테이너가 인스턴스화됩니다.
2. "노드 가져오기" 작업 중에 HostA에 적용된 동일한 네트워크 인터페이스와 블록 스토리지 매핑을 사용하는 HostB의 노드 컨테이너가 인스턴스화됩니다. 그런 다음 캐시된 영구 상태 데이터가 새 인스턴스에 삽입됩니다.

이 작업 모드가 주어지면 마이그레이션을 허용하고 작동하기 위해서는 노드의 모든 시스템 데이터와 객체 스토리지 볼륨을 HostA와 HostB에서 액세스할 수 있어야 합니다. 또한 HostA 및 HostB에서 동일한 LUN을 참조하도록 보장된 이름을 사용하여 노드에 매핑되어야 합니다.

다음 예에서는 StorageGRID 스토리지 노드에 대한 블록 디바이스 매핑 솔루션 중 하나를 보여 줍니다. 이 경우 DM 다중 경로가 호스트에서 사용되고 별칭 필드는 모든 호스트에서 사용할 수 있는 일관되고 알기 쉬운 블록 디바이스 이름을 제공하기 위해 에 `/etc/multipath.conf` 사용되었습니다.



호스트 준비(Ubuntu 또는 Debian)

설치 중에 호스트 전체의 설정이 변경되는 방식

베어 메탈 시스템에서 StorageGRID는 호스트 전체 설정을 일부 변경합니다 `sysctl`.

다음과 같은 변경 사항이 적용됩니다.

```
# Recommended Cassandra setting: CASSANDRA-3563, CASSANDRA-13008, DataStax
documentation
vm.max_map_count = 1048575

# core file customization
# Note: for cores generated by binaries running inside containers, this
# path is interpreted relative to the container filesystem namespace.
# External cores will go nowhere, unless /var/local/core also exists on
# the host.
kernel.core_pattern = /var/local/core/%e.core.%p

# Set the kernel minimum free memory to the greater of the current value
or
# 512MiB if the host has 48GiB or less of RAM or 1.83GiB if the host has
more than 48GiB of RTAM
vm.min_free_kbytes = 524288

# Enforce current default swappiness value to ensure the VM system has
some
# flexibility to garbage collect behind anonymous mappings. Bump
watermark_scale_factor
# to help avoid OOM conditions in the kernel during memory allocation
bursts. Bump
# dirty_ratio to 90 because we explicitly fsync data that needs to be
```

```
persistent, and
# so do not require the dirty_ratio safety net. A low dirty_ratio combined
with a large
# working set (nr_active_pages) can cause us to enter synchronous I/O mode
unnecessarily,
# with deleterious effects on performance.
vm.swappiness = 60
vm.watermark_scale_factor = 200
vm.dirty_ratio = 90

# Turn off slow start after idle
net.ipv4.tcp_slow_start_after_idle = 0

# Tune TCP window settings to improve throughput
net.core.rmem_max = 8388608
net.core.wmem_max = 8388608
net.ipv4.tcp_rmem = 4096 524288 8388608
net.ipv4.tcp_wmem = 4096 262144 8388608
net.core.netdev_max_backlog = 2500

# Turn on MTU probing
net.ipv4.tcp_mtu_probing = 1

# Be more liberal with firewall connection tracking
net.ipv4.netfilter.ip_conntrack_tcp_be_liberal = 1

# Reduce TCP keepalive time to reasonable levels to terminate dead
connections
net.ipv4.tcp_keepalive_time = 270
net.ipv4.tcp_keepalive_probes = 3
net.ipv4.tcp_keepalive_intvl = 30

# Increase the ARP cache size to tolerate being in a /16 subnet
net.ipv4.neigh.default.gc_thresh1 = 8192
net.ipv4.neigh.default.gc_thresh2 = 32768
net.ipv4.neigh.default.gc_thresh3 = 65536
net.ipv6.neigh.default.gc_thresh1 = 8192
net.ipv6.neigh.default.gc_thresh2 = 32768
net.ipv6.neigh.default.gc_thresh3 = 65536

# Disable IP forwarding, we are not a router
net.ipv4.ip_forward = 0

# Follow security best practices for ignoring broadcast ping requests
net.ipv4.icmp_echo_ignore_broadcasts = 1
```

```
# Increase the pending connection and accept backlog to handle larger
connection bursts.
net.core.somaxconn=4096
net.ipv4.tcp_max_syn_backlog=4096
```

Linux를 설치합니다

모든 Ubuntu 또는 Debian GRID 호스트에 StorageGRID를 설치해야 합니다. 지원되는 버전 목록은 NetApp 상호 운용성 매트릭스 툴을 참조하십시오.

시작하기 전에

운영 체제가 아래 나열된 StorageGRID의 최소 커널 버전 요구 사항을 충족하는지 확인합니다. 명령을 사용하여 `uname -r` 운영 체제의 커널 버전을 가져오거나 OS 공급업체에 문의하십시오.

- 참고: * Ubuntu 버전 18.04 및 20.04에 대한 지원은 더 이상 사용되지 않으며 향후 릴리스에서 제거될 예정입니다.

Ubuntu 버전	최소 커널 버전	커널 패키지 이름입니다
18.04.6(사용되지 않음)	5.4.0-150 - 일반	linux-image-5.4.0-150-generic/bionic-updates, bionic-security, 현재 5.4.0-150.167-18.04.1
20.04.5(사용되지 않음)	5.4.0-131 - 일반	linux-image-5.4.0-131-generic/focal-updates, 현재 5.4.0-131.147
22.04.1	5.15.0-47 - 일반	linux-image-5.15.0-47-generic/jammy-updates, jammy-security, 현재 5.15.0-47.51
24.04	6.8.0-31 - 일반	linux-image-6.8.0-31-generic/noble, 현재 6.8.0-31.31

참고**:** 데비안 버전 11에 대한 지원은 더 이상 사용되지 않으며 향후 릴리스에서 제거될 예정입니다.

데비안 버전	최소 커널 버전	커널 패키지 이름입니다
11(폐기됨)	5.10.0-18-AMD64	Linux-image-5.10.0-18-AMD64/stable, 현재 5.10.150-1
12	6.1.0-9-AMD64	linux-image-6.1.0-9-amd64/stable, 현재 6.1.27-1

단계

1. 배포자의 지침 또는 표준 절차에 따라 모든 물리적 또는 가상 그리드 호스트에 Linux를 설치합니다.

> 그래픽 데스크톱 환경을 설치하지 마십시오. Ubuntu를 설치할 때 * 표준 시스템 유틸리티 * 를 선택해야 합니다. Ubuntu 호스트에 대한 ssh 액세스를 활성화하려면 * OpenSSH 서버 * 를 선택하는 것이 좋습니다. 다른 모든 옵션은 선택 취소 상태를 유지할 수 있습니다.

2. 모든 호스트가 Ubuntu 또는 Debian 패키지 리포지토리에 액세스할 수 있는지 확인합니다.
3. 스왑이 활성화된 경우:
   a. 다음 명령을 실행합니다. `$ sudo swapoff --all`
   b. 설정을 유지하려면 에서 모든 스왑 항목을 `/etc/fstab` 제거합니다.

> 스왑을 완전히 사용하지 않도록 설정하면 성능이 크게 저하될 수 있습니다.

AppArmor 프로필 설치를 이해합니다

사용자가 자체 배포된 Ubuntu 환경에서 AppArmor 필수 액세스 제어 시스템을 사용하는 경우 기본 시스템에 설치하는 패키지와 관련된 AppArmor 프로필이 StorageGRID와 함께 설치된 해당 패키지에 의해 차단될 수 있습니다.

기본적으로 AppArmor 프로필은 기본 운영 체제에 설치하는 패키지에 설치됩니다. StorageGRID 시스템 컨테이너에서 이러한 패키지를 실행하면 AppArmor 프로필이 차단됩니다. DHCP, MySQL, NTP 및 tcdump 기본 패키지가 AppArmor와 충돌하고 다른 기본 패키지도 충돌할 수 있습니다.

AppArmor 프로필을 처리할 수 있는 두 가지 옵션이 있습니다.

- StorageGRID 시스템 컨테이너의 패키지와 겹치는 기본 시스템에 설치된 패키지의 개별 프로필을 비활성화합니다. 개별 프로필을 비활성화하면 AppArmor가 활성화되었음을 나타내는 항목이 StorageGRID 로그 파일에 나타납니다.

  다음 명령을 사용합니다.

  ```
  sudo ln -s /etc/apparmor.d/<profile.name> /etc/apparmor.d/disable/
  sudo apparmor_parser -R /etc/apparmor.d/<profile.name>
  ```

- 예: *

  ```
  sudo ln -s /etc/apparmor.d/bin.ping /etc/apparmor.d/disable/
  sudo apparmor_parser -R /etc/apparmor.d/bin.ping
  ```

- AppArmor를 모두 비활성화합니다. Ubuntu 9.10 이상의 경우 Ubuntu 온라인 커뮤니티의 지침을 따릅니다 "AppArmor를 비활성화합니다". 최신 Ubuntu 버전에서는 AppArmor를 완전히 비활성화할 수 없습니다.

  AppArmor를 비활성화하면 StorageGRID 로그 파일에 AppArmor가 활성화되었음을 나타내는 항목이 나타나지 않습니다.

호스트 네트워크 구성(Ubuntu 또는 Debian)

호스트에서 Linux 설치를 완료한 후 나중에 배포할 StorageGRID 노드에 매핑하는 데 적합한 네트워크 인터페이스 세트를 준비하기 위해 몇 가지 추가 구성을 수행해야 할 수 있습니다.

시작하기 전에

- 를 검토했습니다."StorageGRID 네트워킹 지침"
- 에 대한 정보를 검토했습니다."노드 컨테이너 마이그레이션 요구사항"
- 가상 호스트를 사용하는 경우 호스트 네트워크를 구성하기 전에 를 읽은 MAC 주소 복제에 대한 고려 사항 및 권장 사항것입니다.

> VM을 호스트로 사용하는 경우 가상 네트워크 어댑터로 VMXNET 3을 선택해야 합니다. VMware E1000 네트워크 어댑터로 인해 특정 Linux 배포판에 배포된 StorageGRID 컨테이너의 연결 문제가 발생했습니다.

이 작업에 대해

그리드 노드는 그리드 네트워크와 선택적으로 관리자 및 클라이언트 네트워크에 액세스할 수 있어야 합니다. 호스트의 물리적 인터페이스를 각 그리드 노드의 가상 인터페이스에 연결하는 매핑을 생성하여 이 액세스를 제공합니다. 호스트 인터페이스를 생성할 때 이름을 friendly 로 사용하여 모든 호스트에 쉽게 구축하고 마이그레이션을 설정할 수 있습니다.

호스트와 하나 이상의 노드 간에 동일한 인터페이스를 공유할 수 있습니다. 예를 들어, 호스트 액세스 및 노드 관리 네트워크 액세스에 동일한 인터페이스를 사용하여 호스트 및 노드 유지 관리를 용이하게 할 수 있습니다. 호스트와 개별 노드 간에 동일한 인터페이스를 공유할 수 있지만 모두 IP 주소가 서로 달라야 합니다. IP 주소는 노드 간 또는 호스트와 노드 간에 공유할 수 없습니다.

동일한 호스트 네트워크 인터페이스를 사용하여 호스트의 모든 StorageGRID 노드에 그리드 네트워크 인터페이스를 제공하거나, 각 노드에 대해 다른 호스트 네트워크 인터페이스를 사용하거나, 둘 사이에 작업을 수행할 수 있습니다. 그러나 일반적으로 단일 노드에 대한 Grid 및 Admin Network 인터페이스와 동일한 호스트 네트워크 인터페이스를 제공하거나 한 노드에 대한 Grid Network 인터페이스와 다른 노드에 대한 Client Network 인터페이스를 제공하지 않습니다.

이 작업은 여러 가지 방법으로 완료할 수 있습니다. 예를 들어, 호스트가 가상 머신이고 각 호스트에 대해 하나 또는 두 개의 StorageGRID 노드를 구축하는 경우 하이퍼바이저에서 올바른 수의 네트워크 인터페이스를 생성하고 일대일 매핑을 사용할 수 있습니다. 운영 용도로 베어 메탈 호스트에 여러 노드를 구축하는 경우 Linux 네트워킹 스택이 VLAN 및 LACP 지원을 활용하여 내결함성 및 대역폭 공유를 제공할 수 있습니다. 다음 섹션에서는 이러한 두 가지 예에 대해 자세히 설명합니다. 이러한 예제 중 하나를 사용할 필요가 없습니다. 필요에 맞는 방법을 사용할 수 있습니다.

> Bond 또는 Bridge 장치를 컨테이너 네트워크 인터페이스로 직접 사용하지 마십시오. 이렇게 하면 컨테이너 네임스페이스의 연결 및 브리지 장치와 함께 MACVLAN을 사용하는 커널 문제로 인해 노드 시작이 방지될 수 있습니다. 대신 VLAN 또는 가상 이더넷(veth) 쌍과 같은 비연결 장치를 사용하십시오. 이 디바이스를 노드 구성 파일의 네트워크 인터페이스로 지정합니다.

MAC 주소 복제에 대한 고려 사항 및 권장 사항

MAC 주소 클로닝은 컨테이너가 호스트의 MAC 주소를 사용하고 호스트는 사용자가 지정한 주소나 임의로 생성된 주소의 MAC 주소를 사용하게 합니다. 무차별 모드 네트워크 구성을 사용하지 않으려면 MAC 주소 복제를 사용해야 합니다.

MAC 클론 생성 활성화

특정 환경에서는 관리 네트워크, 그리드 네트워크 및 클라이언트 네트워크에 전용 가상 NIC를 사용할 수 있으므로 MAC 주소 클로닝을 통해 보안을 강화할 수 있습니다. 컨테이너가 호스트에 있는 전용 NIC의 MAC 주소를 사용하도록 하면 무차별 모드 네트워크 구성을 사용하지 않도록 할 수 있습니다.

MAC 주소 복제는 가상 서버 설치에 사용하기 위한 것이며 모든 물리적 어플라이언스 구성에서 제대로 작동하지 않을 수 있습니다.

MAC 클론 대상 인터페이스가 사용 중이어서 노드가 시작되지 않는 경우 노드를 시작하기 전에 링크를 "다운"으로 설정해야 할 수 있습니다. 또한 링크가 작동 중일 때 가상 환경에서 네트워크 인터페이스에서 MAC 클로닝을 방지할 수 있습니다. 노드가 MAC 주소를 설정하지 못하고 사용 중인 인터페이스로 인해 시작되는 경우 노드를 시작하기 전에 링크를 "다운"으로 설정하면 문제가 해결될 수 있습니다.

MAC 주소 복제는 기본적으로 해제되어 있으며 노드 구성 키로 설정해야 합니다. StorageGRID를 설치할 때 활성화해야 합니다.

각 네트워크마다 하나의 키가 있습니다.

- `ADMIN_NETWORK_TARGET_TYPE_INTERFACE_CLONE_MAC`
- `GRID_NETWORK_TARGET_TYPE_INTERFACE_CLONE_MAC`
- `CLIENT_NETWORK_TARGET_TYPE_INTERFACE_CLONE_MAC`

키를 "true"로 설정하면 컨테이너가 호스트 NIC의 MAC 주소를 사용하게 됩니다. 또한 호스트는 지정된 컨테이너 네트워크의 MAC 주소를 사용합니다. 기본적으로 컨테이너 주소는 무작위로 생성된 주소이지만 노드 구성 키를 사용하여 주소를 설정한 경우 `_NETWORK_MAC` 해당 주소가 대신 사용됩니다. 호스트와 컨테이너의 MAC 주소는 항상 다릅니다.

하이퍼바이저에서 무차별 모드를 설정하지 않고 가상 호스트에서 MAC 클로닝을 활성화하면 호스트의 인터페이스를 사용하는 Linux 호스트 네트워킹이 작동하지 않을 수 있습니다.

Mac 클론 복제 활용 사례

MAC 클로닝에는 다음 두 가지 사용 사례를 고려해야 합니다.

- MAC 클론 생성이 활성화되지 않음: `_CLONE_MAC` 노드 구성 파일의 키가 설정되지 않았거나 "false"로 설정되어 있지 않으면 호스트는 호스트 NIC MAC을 사용하고 컨테이너에 StorageGRID 생성 MAC을 갖게 됩니다. `_NETWORK_MAC` 키에 주소가 설정되어 있으면 `_NETWORK_MAC` 컨테이너에 키에 지정된 주소가 `_NETWORK_MAC` 지정됩니다. 이러한 키 구성을 위해서는 무차별 모드를 사용해야 합니다.
- MAC 클론 생성 활성화: 노드 구성 파일의 키가 "true"로 설정된 경우 `_CLONE_MAC` 컨테이너에서 호스트 NIC MAC을 사용하고, 키에 MAC이 지정되지 않은 경우 호스트는 StorageGRID에서 생성된 MAC을 사용합니다. `_NETWORK_MAC` 키에 주소가 설정된 경우 `_NETWORK_MAC` 호스트는 생성된 주소가 아닌 지정된 주소를 사용합니다. 이 키 구성에서 무차별 모드를 사용해서는 안 됩니다.

MAC 주소 클로닝을 사용하지 않고 하이퍼바이저에 의해 할당된 것이 아닌 MAC 주소에 대한 데이터를 모든 인터페이스에서 수신 및 전송하도록 허용하려면 가상 스위치 및 포트 그룹 수준의 보안 속성이 Promiscuous Mode, MAC Address 변경 및 Forged 전송에 대해 * Accept * 로 설정되어 있는지 확인합니다. 가상 스위치에 설정된 값은 포트 그룹 수준의 값으로 재정의할 수 있으므로 두 위치에서 설정이 동일한지 확인합니다.

MAC 복제를 활성화하려면 을 "노드 구성 파일 생성 지침"참조하십시오.

Mac 클론 복제의 예

인터페이스 ens256의 경우 MAC 주소가 11:22:33:44:55:66이고 노드 구성 파일의 경우 다음 키가 있는 호스트에서

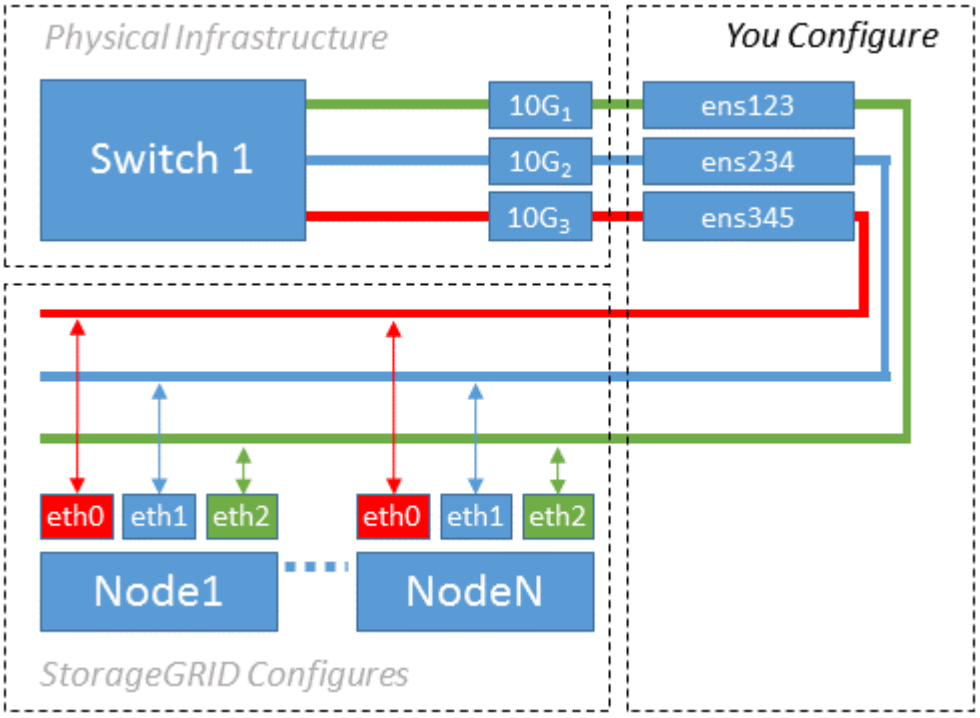
활성화된 MAC 클론 복제의 예:

- `ADMIN_NETWORK_TARGET = ens256`
- `ADMIN_NETWORK_MAC = b2:9c:02:c2:27:10`
- `ADMIN_NETWORK_TARGET_TYPE_INTERFACE_CLONE_MAC = true`

결과: en256의 호스트 MAC은 B2:9c:02:C2:27:10이고 관리 네트워크 MAC은 11:22:33:44:55:66입니다

예 **1:** 물리적 **NIC** 또는 가상 **NIC**에 **1:1** 대 **1** 매핑

예제 1에서는 호스트측 구성이 거의 또는 전혀 필요하지 않은 간단한 물리적 인터페이스 매핑에 대해 설명합니다.



Linux 운영 체제는 설치 또는 부팅 중에 또는 인터페이스가 핫 애드 상태일 때 자동으로 ensXYZ 인터페이스를 생성합니다. 부팅 후 인터페이스가 자동으로 실행되도록 설정하는 것 외에는 구성이 필요하지 않습니다. 나중에 구성 프로세스에서 올바른 매핑을 제공할 수 있도록 StorageGRID 네트워크(그리드, 관리자 또는 클라이언트)에 해당하는 ensXYZ를 결정해야 합니다.

이 그림에서는 여러 StorageGRID 노드를 보여 줍니다. 그러나 일반적으로 단일 노드 VM에 이 구성을 사용합니다.

스위치 1이 물리적 스위치인 경우 액세스 모드에 대해 인터페이스 $10G_1$$10G_3$에 연결된 포트를 구성하고 해당 VLAN에 배치해야 합니다.

예 **2: VLAN**을 전달하는 **LACP** 결합

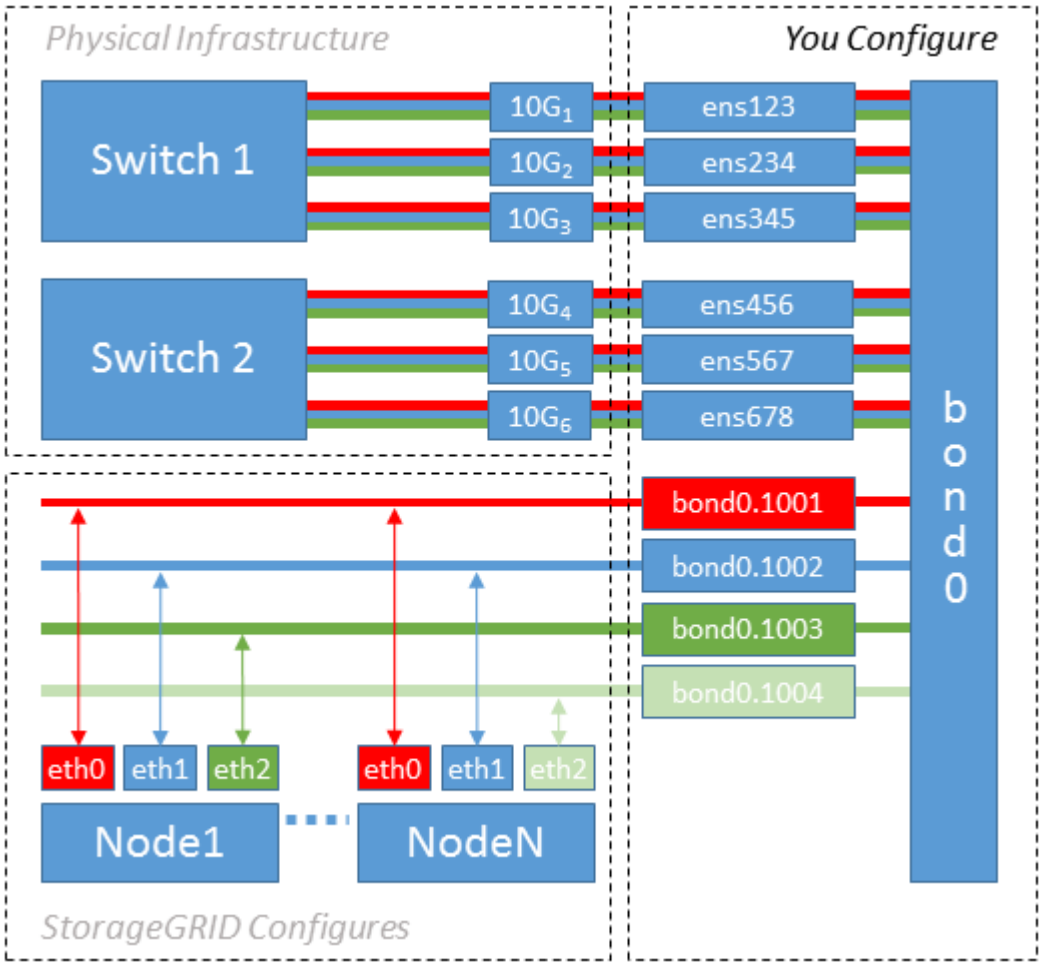
예제 2에서는 네트워크 인터페이스를 결합하거나 사용 중인 Linux 배포판에서 VLAN 인터페이스를 만드는 방법에 대해 잘 알고 있다고 가정합니다.

이 작업에 대해

예제 2에서는 단일 호스트의 모든 노드에서 사용 가능한 모든 네트워크 대역폭을 쉽게 공유할 수 있도록 지원하는 일반, 유연한 VLAN 기반 체계를 설명합니다. 이 예는 특히 베어 메탈 호스트에 적용할 수 있습니다.

이 예제를 이해하려면 각 데이터 센터에 그리드, 관리자 및 클라이언트 네트워크에 대한 세 개의 개별 서브넷이 있다고 가정합니다. 서브넷은 별도의 VLAN(1001, 1002 및 1003)에 있으며 LACP 결합 트렁크 포트(bond0)의 호스트에 제공됩니다. Bond.0.1001, bond0.1002 및 bond0.1003의 세 가지 VLAN 인터페이스를 구성합니다.

동일한 호스트에서 노드 네트워크에 대해 별도의 VLAN과 서브넷이 필요한 경우, 결합에 VLAN 인터페이스를 추가하고 이를 호스트에 매핑할 수 있습니다(그림에서 bond0.1004로 표시됨).



단계

1. StorageGRID 네트워크 연결에 사용할 모든 물리적 네트워크 인터페이스를 단일 LACP 결합으로 통합합니다.

   예를 들어, bond0과 같이 모든 호스트의 본드 결합에 동일한 이름을 사용합니다.

2. 표준 VLAN 인터페이스 명명 규칙을 사용하여 이 연결을 관련 "물리적 장치"로 사용하는 VLAN 인터페이스를 `physdev-name.VLAN ID` 생성합니다.

   1단계와 2단계는 네트워크 링크의 다른 끝을 종료하는 에지 스위치에 적절한 구성이 필요합니다. 에지 스위치 포트도 LACP 포트 채널로 집계되고 트렁크로 구성되어 필요한 모든 VLAN을 통과할 수 있도록 허용해야 합니다.

   이 호스트별 네트워킹 구성 체계에 대한 인터페이스 구성 파일 예가 제공됩니다.

관련 정보

"예 /etc/network/interfaces"

호스트 스토리지를 구성합니다

각 호스트에 블록 스토리지 볼륨을 할당해야 합니다.

시작하기 전에

이 과제를 수행하는 데 필요한 정보를 제공하는 다음 주제를 검토했습니다.

- "요구사항을 충족해야 합니다"
- "노드 컨테이너 마이그레이션 요구사항"

이 작업에 대해

블록 스토리지 볼륨(LUN)을 호스트에 할당할 때 "스토리지 요구 사항"의 표를 사용하여 다음을 확인합니다.

- 각 호스트에 필요한 볼륨 수(해당 호스트에 구축할 노드 수 및 유형 기준)
- 각 볼륨의 스토리지 범주(즉, 시스템 데이터 또는 오브젝트 데이터)
- 각 볼륨의 크기입니다

호스트에 StorageGRID 노드를 배포할 때 이 정보와 Linux가 각 물리적 볼륨에 할당한 영구 이름을 사용합니다.

이러한 볼륨을 파티션, 포맷 또는 마운트할 필요가 없습니다. 호스트가 볼 수 있도록 해야 합니다.

메타데이터 전용 스토리지 노드에는 하나의 오브젝트 데이터 LUN만 필요합니다.

(/dev/sdb`볼륨 이름 목록을 작성할 때 "raw" 특수 장치 파일을 사용하지 마십시오. 이러한 파일은 호스트의 재부팅 시 변경될 수 있으며, 이는 시스템의 올바른 작동에 영향을 줍니다. iSCSI LUN 및 Device Mapper Multipathing을 사용하는 경우, 특히 SAN 토폴로지에 공유 스토리지에 대한 중복 네트워크 경로가 포함되어 있는 경우 디렉토리에서 다중 경로 별칭을 사용하는 `/dev/mapper 것이 좋습니다. 또는 영구 장치 이름에 대해 에서 시스템에서 만든 소프트링크를 사용할 수 /dev/disk/by-path/ 있습니다.

예를 들면 다음과 같습니다.

```
ls -l
$ ls -l /dev/disk/by-path/
total 0
lrwxrwxrwx 1 root root  9 Sep 19 18:53 pci-0000:00:07.1-ata-2 -> ../../sr0
lrwxrwxrwx 1 root root  9 Sep 19 18:53 pci-0000:03:00.0-scsi-0:0:0:0 ->
../../sda
lrwxrwxrwx 1 root root 10 Sep 19 18:53 pci-0000:03:00.0-scsi-0:0:0:0-part1
-> ../../sda1
lrwxrwxrwx 1 root root 10 Sep 19 18:53 pci-0000:03:00.0-scsi-0:0:0:0-part2
-> ../../sda2
lrwxrwxrwx 1 root root  9 Sep 19 18:53 pci-0000:03:00.0-scsi-0:0:1:0 ->
../../sdb
lrwxrwxrwx 1 root root  9 Sep 19 18:53 pci-0000:03:00.0-scsi-0:0:2:0 ->
../../sdc
lrwxrwxrwx 1 root root  9 Sep 19 18:53 pci-0000:03:00.0-scsi-0:0:3:0 ->
../../sdd
```

각 설치 환경에 따라 결과가 달라집니다.

각 블록 스토리지 볼륨에 알기 쉬운 이름을 할당하여 초기 StorageGRID 설치 및 향후 유지 관리 절차를 간소화하십시오. 공유 스토리지 볼륨에 대한 중복 액세스를 위해 디바이스 매퍼 다중 경로 드라이버를 사용하는 경우 파일의 필드를 `/etc/multipath.conf` 사용할 수 `alias` 있습니다.

예를 들면 다음과 같습니다.

```
multipaths {
      multipath {
            wwid 3600a09800059d6df00005df2573c2c30
            alias docker-storage-volume-hostA
      }
      multipath {
            wwid 3600a09800059d6df00005df3573c2c30
            alias sgws-adm1-var-local
      }
      multipath {
            wwid 3600a09800059d6df00005df4573c2c30
            alias sgws-adm1-audit-logs
      }
      multipath {
            wwid 3600a09800059d6df00005df5573c2c30
            alias sgws-adm1-tables
      }
      multipath {
            wwid 3600a09800059d6df00005df6573c2c30
            alias sgws-gw1-var-local
      }
      multipath {
            wwid 3600a09800059d6df00005df7573c2c30
            alias sgws-sn1-var-local
      }
      multipath {
            wwid 3600a09800059d6df00005df7573c2c30
            alias sgws-sn1-rangedb-0
      }
      …
```

이러한 방식으로 별칭 필드를 사용하면 별칭이 호스트의 디렉토리에 블록 디바이스로 나타나므로 구성 또는 유지 관리 작업에서 블록 `/dev/mapper` 스토리지 볼륨을 지정해야 할 때마다 쉽게 검증된 친숙한 이름을 지정할 수 있습니다.

StorageGRID 노드 마이그레이션을 지원하고 장치 매퍼 다중 경로를 사용하도록 공유 스토리지를 설정하는 경우 모든 공동 위치 호스트에 공통 을 생성하고 설치할 수 `/etc/multipath.conf` 있습니다. 각 호스트에서 다른 Docker 스토리지 볼륨을 사용하기만 하면 됩니다. 각 Docker 스토리지 볼륨 LUN의 별칭에 타겟 호스트 이름을 포함하여 별칭을 사용하면 기억하기 쉽고 이 방법이 권장됩니다.

> ⓘ 소프트웨어 전용 배포를 위한 컨테이너 엔진으로 Docker에 대한 지원은 더 이상 사용되지 않습니다. Docker는 향후 릴리즈에서 다른 컨테이너 엔진으로 대체될 예정입니다.

관련 정보

- "요구사항을 충족해야 합니다"
- "노드 컨테이너 마이그레이션 요구사항"

컨테이너 엔진 저장소 볼륨을 구성합니다

컨테이너 엔진(Docker 또는 Podman)을 설치하기 전에 스토리지 볼륨을 포맷하고 마운트해야 할 수 있습니다.

> 소프트웨어 전용 배포를 위한 컨테이너 엔진으로 Docker에 대한 지원은 더 이상 사용되지 않습니다. Docker는 향후 릴리즈에서 다른 컨테이너 엔진으로 대체될 예정입니다.

이 작업에 대해

Docker 저장소 볼륨에 로컬 스토리지를 사용할 계획이고 이 포함된 호스트 파티션에 사용 가능한 공간이 충분하다면 이 단계를 건너뛸 수 있습니다 `/var/lib`.

단계

1. Docker 스토리지 볼륨에 파일 시스템을 생성합니다.

```
sudo mkfs.ext4 docker-storage-volume-device
```

2. Docker 스토리지 볼륨을 마운트합니다.

```
sudo mkdir -p /var/lib/docker
sudo mount docker-storage-volume-device /var/lib/docker
```

3. /etc/fstab에 Docker-storage-volume-device 항목을 추가합니다.

   이 단계를 수행하면 호스트가 재부팅된 후 스토리지 볼륨이 자동으로 다시 마운트됩니다.

Docker를 설치합니다

StorageGRID 시스템은 Linux에서 Docker 컨테이너 모음으로 실행됩니다. StorageGRID를 설치하기 전에 Docker를 설치해야 합니다.

> 소프트웨어 전용 배포를 위한 컨테이너 엔진으로 Docker에 대한 지원은 더 이상 사용되지 않습니다. Docker는 향후 릴리즈에서 다른 컨테이너 엔진으로 대체될 예정입니다.

단계

1. Linux 배포에 대한 지침에 따라 Docker를 설치합니다.

   > Docker가 Linux 배포판에 포함되어 있지 않은 경우 Docker 웹 사이트에서 다운로드할 수 있습니다.

2. 다음 두 명령을 실행하여 Docker를 활성화하고 시작했는지 확인합니다.

```
sudo systemctl enable docker
```

```
sudo systemctl start docker
```

3. 다음을 입력하여 예상 버전의 Docker를 설치했는지 확인합니다.

```
sudo docker version
```

클라이언트 및 서버 버전은 1.11.0 이상이어야 합니다.

관련 정보

"호스트 스토리지를 구성합니다"

StorageGRID 호스트 서비스를 설치합니다

StorageGRID DEB 패키지를 사용하여 StorageGRID 호스트 서비스를 설치합니다.

이 작업에 대해

다음 지침은 DEB 패키지에서 호스트 서비스를 설치하는 방법을 설명합니다. 또는 설치 아카이브에 포함된 APT 리포지토리 메타데이터를 사용하여 DEB 패키지를 원격으로 설치할 수 있습니다. Linux 운영 체제에 대한 APT 리포지토리 지침을 참조하십시오.

단계

1. 각 호스트에 StorageGRID DEB 패키지를 복사하거나 공유 스토리지에서 사용할 수 있도록 합니다.

   예를 들어, `/tmp` 다음 단계에서 예제 명령을 사용할 수 있도록 디렉토리에 배치합니다.

2. 각 호스트에 루트로 로그인하거나 sudo 권한이 있는 계정을 사용하여 다음 명령을 실행합니다.

   먼저 패키지를 `service` 설치하고 패키지를 두 번째로 설치해야 `images` 합니다. 패키지를 이외의 디렉토리에 배치한 경우 `/tmp` 사용한 경로를 반영하도록 명령을 수정합니다.

```
sudo dpkg --install /tmp/storagegrid-webscale-images-version-SHA.deb
```

```
sudo dpkg --install /tmp/storagegrid-webscale-service-version-SHA.deb
```

> StorageGRID 패키지를 설치하기 전에 Python 2.7이 이미 설치되어 있어야 합니다. 이 `sudo dpkg --install /tmp/storagegrid-webscale-images-version-SHA.deb` 명령을 실행할 때까지 명령이 실패합니다.

저작권 정보

상표 정보