BeeGFS 파일 시스템을 구축합니다

BeeGFS on NetApp with E-Series Storage

NetApp
June 18, 2024

목차

BeeGFS 파일 시스템을 구축합니다

Ansible 플레이북 개요

Ansible을 사용하여 BeeGFS HA 클러스터 구축 및 관리

개요

이전 섹션에서는 BeeGFS HA 클러스터를 나타내는 Ansible 재고를 구축하는 데 필요한 단계를 안내했습니다. 이 섹션에서는 클러스터를 구축 및 관리하기 위해 NetApp에서 개발한 Ansible 자동화를 소개합니다.

Ansible: 주요 개념

진행하기 전에 Ansible의 몇 가지 주요 개념을 숙지하는 것이 좋습니다.

- Ansible 인벤토리에 대해 실행할 작업은 * 플레이북 * 으로 알려진 내용에 정의됩니다.
  - Ansible에서 수행하는 대부분의 작업은 * idemtiple * 으로 설계되어 여러 번 실행할 수 있으므로, 원하는 구성/상태가 중단 없이 적용되었는지, 불필요한 업데이트를 하지 않고 그대로 적용되는지를 확인할 수 있습니다.
- Ansible에서 실행할 수 있는 가장 작은 단위는 * 모듈 * 입니다.
  - 일반적인 플레이북에서는 여러 모듈을 사용합니다.
    - 예: 패키지 다운로드, 구성 파일 업데이트, 서비스 시작/활성화
  - NetApp은 모듈을 분산하여 NetApp E-Series 시스템을 자동화합니다.
- 복잡한 자동화는 하나의 역할로 더 잘 패키징됩니다.
  - 기본적으로 재사용 가능한 플레이북을 배포하기 위한 표준 형식입니다.
  - NetApp은 Linux 호스트 및 BeeGFS 파일 시스템에 대한 역할을 분산합니다.

Ansible용 BeeGFS HA 역할: 주요 개념

NetApp에서 각 버전의 BeeGFS를 구축 및 관리하는 데 필요한 모든 자동화 기능이 Ansible 역할로 패키지되어 의 일부로 배포됩니다 "BeeGFS용 NetApp E-Series Ansible 컬렉션":

- 이 역할은 * 설치 프로그램 * 과 BeeGFS용 최신 * 구축/관리 * 엔진 사이의 어딘가에 있다고 생각할 수 있습니다.
  - 최신 인프라를 코드 사례 및 철학으로 적용하여 모든 규모의 스토리지 인프라 관리를 단순화합니다.
  - 와 유사합니다 "구베기도" Project에서는 전체 Kubernetes 배포를 구축/유지 관리하여 컴퓨팅 인프라를 확장할 수 있습니다.
- NetApp에서 NetApp 솔루션에서 BeeGFS를 패키징, 배포 및 유지 관리하는 데 사용하는 * 소프트웨어 정의 * 형식입니다.
  - 전체 Linux 배포판이나 큰 이미지를 배포할 필요 없이 "어플라이언스와 유사한" 환경을 조성하기 위해 노력합니다.
  - 지능형 Pacemaker/BeeGFS 통합을 제공하는 맞춤형 BeeGFS 타겟, IP 주소, 모니터링을 위해 NetApp에서 작성한 OCF(Open Cluster Framework) 규격 클러스터 리소스 에이전트가 포함되어 있습니다.
- 이 역할은 단순히 "자동화"를 구축하는 것이 아니라 다음을 포함한 전체 파일 시스템 수명주기를 관리하는 데

사용됩니다.

- 서비스별 또는 클러스터 전체 구성 변경 및 업데이트 적용
- 하드웨어 문제가 해결된 후 클러스터 복구 및 복구 자동화
- BeeGFS 및 NetApp 볼륨을 사용한 광범위한 테스트 결과를 기준으로 설정된 기본값으로 성능 조정을 단순화합니다.
- 구성 드리프트 확인 및 수정

NetApp은 또한 Ansible 역할을 제공합니다 "BeeGFS 클라이언트"필요에 따라 BeeGFS를 설치하고 파일 시스템을 컴퓨팅/GPU/로그인 노드에 마운트하는 데 사용할 수 있습니다.

BeeGFS HA 클러스터를 구축합니다

Playbook을 사용하여 BeeGFS HA 클러스터를 구축하기 위해 실행해야 할 작업을 지정합니다.

개요

이 섹션에서는 NetApp에서 BeeGFS를 구축/관리하는 데 사용되는 표준 플레이북을 취합하는 방법에 대해 설명합니다.

단계

Ansible 플레이북을 작성합니다

파일을 만듭니다 `playbook.yml` 다음과 같이 채웁니다.

1. 먼저 작업 집합(일반적으로 라고 함)을 정의합니다 "재생") NetApp E-Series 블록 노드에서만 실행되어야 합니다. 일시 중지 작업을 사용하여 설치를 실행하기 전에 메시지를 표시한 다음(우발적인 플레이북 실행을 피하기 위해) 을(를) 가져옵니다 `nar_santricity_management` 역할. 이 역할은 에 정의된 모든 일반 시스템 구성을 적용하는 작업을 처리합니다 `group_vars/eseries_storage_systems.yml` 있습니다 `host_vars/<BLOCK NODE>.yml` 파일.

```
- hosts: eseries_storage_systems
  gather_facts: false
  collections:
    - netapp_eseries.santricity
  tasks:
    - name: Verify before proceeding.
      pause:
        prompt: "Are you ready to proceed with running the BeeGFS HA
role? Depending on the size of the deployment and network performance
between the Ansible control node and BeeGFS file and block nodes this
can take awhile (10+ minutes) to complete."
    - name: Configure NetApp E-Series block nodes.
      import_role:
        name: nar_santricity_management
```

2. 모든 파일 및 블록 노드에 대해 실행할 플레이를 정의합니다.

```
- hosts: all
  any_errors_fatal: true
  gather_facts: false
  collections:
    - netapp_eseries.beegfs
```

3. 이 플레이에서는 HA 클러스터를 구축하기 전에 실행해야 하는 "사전 작업" 세트를 선택적으로 정의할 수 있습니다. Python과 같은 필수 구성 요소를 확인/설치하는 데 유용할 수 있습니다. 제공된 Ansible 태그가 지원되는지 확인하는 등 비행 전 점검을 삽입할 수도 있습니다.

```
  pre_tasks:
    - name: Ensure a supported version of Python is available on all
file nodes.
      block:
        - name: Check if python is installed.
          failed_when: false
          changed_when: false
          raw: python --version
          register: python_version

        - name: Check if python3 is installed.
          raw: python3 --version
          failed_when: false
          changed_when: false
          register: python3_version
          when: 'python_version["rc"] != 0 or (python_version["stdout"]
| regex_replace("Python ", "")) is not version("3.0", ">=")'

        - name: Install python3 if needed.
          raw: |
            id=$(grep "^ID=" /etc/*release* | cut -d= -f 2 | tr -d '"')
            case $id in
              ubuntu) sudo apt install python3 ;;
              rhel|centos) sudo yum -y install python3 ;;
              sles) sudo zypper install python3 ;;
            esac
          args:
            executable: /bin/bash
          register: python3_install
          when: python_version['rc'] != 0 and python3_version['rc'] != 0
          become: true

        - name: Create a symbolic link to python from python3.
```

```
          raw: ln -s /usr/bin/python3 /usr/bin/python
          become: true
          when: python_version['rc'] != 0
      when: inventory_hostname not in
groups[beegfs_ha_ansible_storage_group]

    - name: Verify any provided tags are supported.
      fail:
        msg: "{{ item }} tag is not a supported BeeGFS HA tag. Rerun
your playbook command with --list-tags to see all valid playbook tags."
      when: 'item not in ["all", "storage", "beegfs_ha",
"beegfs_ha_package", "beegfs_ha_configure",
"beegfs_ha_configure_resource", "beegfs_ha_performance_tuning",
"beegfs_ha_backup", "beegfs_ha_client"]'
      loop: "{{ ansible_run_tags }}"
```

4. 마지막으로, 이 플레이는 구축할 BeeGFS 버전에 대한 BeeGFS HA 역할을 가져옵니다.

```
  tasks:
    - name: Verify the BeeGFS HA cluster is properly deployed.
      import_role:
        name: beegfs_ha_7_3 # Alternatively specify: beegfs_ha_7_2.
```

> 지원되는 각 주요 버전 BeeGFS에 대해 BeeGFS HA 역할이 유지됩니다. 따라서 사용자는 주/부 버전을 업그레이드할 시기를 선택할 수 있습니다. 현재 BeeGFS 7.3.x입니다 (`beegfs_7_3`) 또는 BeeGFS 7.2.x (`beegfs_7_2`)가 지원됩니다. 기본적으로 두 역할 모두 릴리즈 시점에 최신 BeeGFS 패치 버전을 구축합니다. 하지만 사용자가 원할 경우 이를 무시하고 최신 패치를 배포할 수 있습니다. 최신 을 참조하십시오 "업그레이드 가이드" 를 참조하십시오.

5. 선택 사항: 추가 작업을 정의하려면 작업이 에 지정되어야 하는지 여부를 염두에 두십시오 `all` 호스트(E-Series 스토리지 시스템 포함) 또는 파일 노드만 포함됩니다. 필요한 경우 를 사용하여 파일 노드를 대상으로 하는 새로운 플레이를 정의합니다 `- hosts: ha_cluster`.

을 클릭합니다 "여기" 전체 플레이북 파일의 예

NetApp Ansible Collections를 설치합니다

Ansible용 BeeGFS 컬렉션 및 모든 종속 항목이 에 유지됩니다 "Ansible 갤럭시". Ansible 제어 노드에서 다음 명령을 실행하여 최신 버전을 설치합니다.

```
ansible-galaxy collection install netapp_eseries.beegfs
```

일반적으로 권장하지는 않지만 컬렉션의 특정 버전을 설치할 수도 있습니다.

```
ansible-galaxy collection install netapp_eseries.beegfs:
==<MAJOR>.<MINOR>.<PATCH>
```

Playbook을 실행합니다

이 포함된 Ansible 제어 노드의 디렉토리에서 `inventory.yml` 및 `playbook.yml` 파일을 실행하고 다음과 같이 플레이북을 실행합니다.

```
ansible-playbook -i inventory.yml playbook.yml
```

클러스터의 크기에 따라 초기 구축에는 20분 이상 걸릴 수 있습니다. 어떠한 이유로든 구축에 실패하는 경우, 잘못된 케이블 연결, 노드 시작 등 문제를 해결하고 Ansible 플레이북을 다시 시작하십시오.

지정할 때 "공통 파일 노드 구성"기본 옵션을 선택하여 Ansible에서 연결 기반 인증을 자동으로 관리하도록 할 경우, a `connAuthFile` 공유 비밀로 사용되는 은 에서 확인할 수 있습니다 `<playbook_dir>/files/beegfs/<sysMgmtdHost>_connAuthFile` (기본값). 파일 시스템에 액세스해야 하는 모든 클라이언트는 이 공유 암호를 사용해야 합니다. 이 작업은 클라이언트가 를 사용하여 구성된 경우 자동으로 처리됩니다 "BeeGFS 클라이언트 역할입니다".

BeeGFS 클라이언트를 구축합니다

선택적으로 Ansible을 사용하여 BeeGFS 클라이언트를 구성하고 파일 시스템을 마운트할 수 있습니다.

개요

BeeGFS 파일 시스템을 액세스하려면 파일 시스템을 마운트해야 하는 각 노드에서 BeeGFS 클라이언트를 설치 및 구성해야 합니다. 이 섹션에서는 사용 가능한 를 사용하여 이러한 작업을 수행하는 방법을 설명합니다 "Ansible 역할".

단계

클라이언트 인벤토리 파일을 생성합니다

1. 필요한 경우, Ansible 제어 노드에서 BeeGFS 클라이언트로 구성하려는 각 호스트에 대해 암호 없는 SSH를 설정합니다.

   ```
   ssh-copy-id <user>@<HOSTNAME_OR_IP>
   ```

2. 아래에서 `host_vars/`에서 각 BeeGFS 클라이언트에 대한 파일을 생성합니다 `<HOSTNAME>.yml` 다음 콘텐츠를 사용하여 환경에 맞는 올바른 정보로 자리 표시자 텍스트를 입력합니다.

   ```
   # BeeGFS Client
   ansible_host: <MANAGEMENT_IP>
   ```

3. 선택적으로 NetApp E-Series 호스트 컬렉션의 역할을 사용하여 클라이언트가 BeeGFS 파일 노드에 연결할 수 있도록 InfiniBand 또는 이더넷 인터페이스를 구성하려면 다음 중 하나를 포함합니다.

   a. 네트워크 유형이 인 경우 "InfiniBand(IPoIB 사용)":

```
eseries_ipoib_interfaces:
- name: <INTERFACE>  # Example: ib0 or i1b
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

   b. 네트워크 유형이 인 경우 "RoCE(RDMA over Converged Ethernet)":

```
eseries_roce_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

   c. 네트워크 유형이 인 경우 "이더넷(TCP 전용, RDMA 없음)":

```
eseries_ip_interfaces:
- name: <INTERFACE>  # Example: eth0.
  address: <IP/SUBNET> # Example: 100.127.100.1/16
- name: <INTERFACE>  # Additional interfaces as needed.
  address: <IP/SUBNET>
```

4. 새 파일을 만듭니다 `client_inventory.yml` 그리고 Ansible이 각 클라이언트에 연결하는 데 사용해야 하는 사용자 지정과 Ansible이 권한 에스컬레이션을 위해 사용해야 하는 암호(이 경우 필요)를 지정합니다 `ansible_ssh_user` 루트 또는 sudo 권한 보유):

```
# BeeGFS client inventory.
all:
  vars:
    ansible_ssh_user: <USER>
    ansible_become_password: <PASSWORD>
```

   암호를 일반 텍스트로 저장하지 마십시오. 대신 Ansible Vault를 사용하십시오(참조 "Ansible 설명서" Ansible Vault로 콘텐츠 암호화)를 사용하거나 을 사용합니다 `--ask-become-pass` 옵션을 클릭합니다.

5. 에 있습니다 `client_inventory.yml` File(파일): 에 BeeGFS 클라이언트로 구성해야 하는 모든 호스트를 나열합니다 `beegfs_clients` 그룹화한 다음 인라인 주석을 참조하여 시스템에서 BeeGFS 클라이언트 커널

모듈을 구축하는 데 필요한 추가 구성을 제거합니다.

```
children:
    # Ansible group representing all BeeGFS clients:
    beegfs_clients:
      hosts:
        <CLIENT HOSTNAME>:
        # Additional clients as needed.

      vars:
        # OPTION 1: If you're using the Mellanox OFED drivers and they
are already installed:
        #eseries_ib_skip: True # Skip installing inbox drivers when
using the IPoIB role.
        #beegfs_client_ofed_enable: True
        #beegfs_client_ofed_include_path:
"/usr/src/ofa_kernel/default/include"

        # OPTION 2: If you're using inbox IB/RDMA drivers and they are
already installed:
        #eseries_ib_skip: True # Skip installing inbox drivers when
using the IPoIB role.

        # OPTION 3: If you want to use inbox IB/RDMA drivers and need
them installed/configured.
        #eseries_ib_skip: False # Default value.
        #beegfs_client_ofed_enable: False # Default value.
```

Mellanox OFED 드라이버를 사용할 때 Beegfs_client_OFED_Include_path가 Linux 설치를 위한 올바른 "헤더 포함 경로"를 가리키는지 확인하십시오. 자세한 내용은 의 BeeGFS 설명서를 참조하십시오 "RDMA 지원".

6. 에 있습니다 `client_inventory.yml` 파일, 이전에 정의한 모든 파일 아래에 마운트할 BeeGFS 파일 시스템을 나열합니다 `vars`:

```
        beegfs_client_mounts:
          - sysMgmtdHost: <IP ADDRESS>  # Primary IP of the BeeGFS
management service.
            mount_point: /mnt/beegfs    # Path to mount BeeGFS on the
client.
            connInterfaces:
              - <INTERFACE> # Example: ibs4f1
              - <INTERFACE>
            beegfs_client_config:
              # Maximum number of simultaneous connections to the same
node.
              connMaxInternodeNum: 128 # BeeGFS Client Default: 12
              # Allocates the number of buffers for transferring IO.
              connRDMABufNum: 36 # BeeGFS Client Default: 70
              # Size of each allocated RDMA buffer
              connRDMABufSize: 65536 # BeeGFS Client Default: 8192
              # Required when using the BeeGFS client with the shared-
disk HA solution.
              # This does require BeeGFS targets be mounted in the
default "sync" mode.
              # See the documentation included with the BeeGFS client
role for full details.
              sysSessionChecksEnabled: false
        # Specify additional file system mounts for this or other file
systems.
```

7. BeeGFS 7.2.7 및 7.3.1 "연결 인증" 구성 또는 명시적으로 비활성화해야 합니다. 를 지정할 때 연결 기반 인증을 구성하는 방법에 따라 다릅니다 "공통 파일 노드 구성"클라이언트 구성을 조정해야 할 수 있습니다.

   a. 기본적으로 HA 클러스터 배포는 연결 인증을 자동으로 구성하고 를 생성합니다 `connauthfile` 이 정보는 에서 Ansible 제어 노드에 배치/유지됩니다 `<INVENTORY>/files/beegfs/<sysMgmtdHost>_connAuthFile`. 기본적으로 BeeGFS 클라이언트 역할은 에 정의된 클라이언트에 이 파일을 읽고 배포하도록 설정되어 있습니다 `client_inventory.yml`추가 조치가 필요하지 않습니다.

      i. 고급 옵션은 에 포함된 기본값 전체 목록을 참조하십시오 "BeeGFS 클라이언트 역할입니다".

   b. 을 사용하여 사용자 지정 암호를 지정하도록 선택한 경우 `beegfs_ha_conn_auth_secret` 에서 지정합니다 `client_inventory.yml` 파일 또한:

```
beegfs_ha_conn_auth_secret: <SECRET>
```

   c. 을 사용하여 연결 기반 인증을 완전히 사용하지 않도록 선택하는 경우 `beegfs_ha_conn_auth_enabled`에서 를 지정합니다 `client_inventory.yml` 파일 또한:

```
beegfs_ha_conn_auth_enabled: false
```

지원되는 매개 변수의 전체 목록과 추가 세부 정보는 를 참조하십시오 "전체 BeeGFS 클라이언트 문서". 클라이언트 인벤토리의 전체 예제를 보려면 을 클릭합니다 "여기".

BeeGFS Client Playbook File을 생성합니다

1. 새 파일을 만듭니다 `client_playbook.yml`

```
# BeeGFS client playbook.
- hosts: beegfs_clients
  any_errors_fatal: true
  gather_facts: true
  collections:
    - netapp_eseries.beegfs
    - netapp_eseries.host
  tasks:
```

2. 선택 사항: NetApp E-Series Host Collection의 역할을 사용하여 클라이언트가 BeeGFS 파일 시스템에 연결할 수 있도록 인터페이스를 구성하려면 구성 중인 인터페이스 유형에 해당하는 역할을 가져옵니다.
   a. InfiniBand(IPoIB)를 사용하는 경우:

```
    - name: Ensure IPoIB is configured
      import_role:
        name: ipoib
```

   b. RoCE(RDMA over Converged Ethernet)를 사용 중인 경우:

```
    - name: Ensure IPoIB is configured
      import_role:
        name: roce
```

   c. 를 사용 중인 경우 이더넷(TCP 전용, RDMA 없음)을 사용합니다.

```
    - name: Ensure IPoIB is configured
      import_role:
        name: ip
```

3. 마지막으로 BeeGFS 클라이언트 역할을 가져와 클라이언트 소프트웨어를 설치하고 파일 시스템 마운트를 설정합니다.

```
    # REQUIRED: Install the BeeGFS client and mount the BeeGFS file
system.
    - name: Verify the BeeGFS clients are configured.
      import_role:
        name: beegfs_client
```

클라이언트 플레이북의 전체 예제를 보려면 을 클릭합니다 "여기".

BeeGFS Client Playbook을 실행합니다

클라이언트를 설치/구축하고 BeeGFS를 마운트하려면 다음 명령을 실행합니다.

```
ansible-playbook -i client_inventory.yml client_playbook.yml
```

BeeGFS 구축을 확인합니다

시스템을 운영 환경에 배치하기 전에 파일 시스템 구축을 확인하십시오.

개요

BeeGFS 파일 시스템을 운영 환경에 배치하기 전에 몇 가지 검증 검사를 수행하십시오.

단계

1. 모든 클라이언트에 로그인하고 다음을 실행하여 모든 예상 노드가 존재하고 연결 가능한지, 불일치 또는 보고된 다른 문제가 없는지 확인합니다.

```
beegfs-fsck --checkfs
```

2. 전체 클러스터를 종료한 다음 재시작합니다. 모든 파일 노드에서 다음을 실행합니다.

```
pcs cluster stop --all # Stop the cluster on all file nodes.
pcs cluster start --all  # Start the cluster on all file nodes.
pcs status # Verify all nodes and services are started and no failures
are reported (the command may need to be reran a few times to allow time
for all services to start).
```

3. 각 노드를 스탠바이에 배치하고 BeeGFS 서비스가 보조 노드로 페일오버할 수 있는지 확인합니다. 이 작업을 수행하려면 파일 노드에 로그인하고 다음을 실행합니다.

```
pcs status # Verify the cluster is healthy at the start.
pcs node standby <FILE NODE HOSTNAME> # Place the node under test in
standby.
pcs status # Verify services are started on a secondary node and no
failures are reported.
pcs node unstandby <FILE NODE HOSTNAME> # Take the node under test out
of standby.
pcs status # Verify the file node is back online and no failures are
reported.
pcs resource relocate run # Move all services back to their preferred
nodes.
pcs status # Verify services have moved back to the preferred node.
```

4. IOR 및 MDTest와 같은 성능 벤치마킹 툴을 사용하여 파일 시스템 성능이 기대에 부합하는지 확인합니다. BeeGFS에 사용되는 일반적인 테스트 및 매개 변수의 예는 에서 찾을 수 있습니다 "설계 Verification" 섹션: NetApp 검증 아키텍처의 BeeGFS

특정 현장/설치에 대해 정의된 수용 기준에 따라 추가 테스트를 수행해야 합니다.