使用Astra Trident

Astra Trident

NetApp
November 20, 2023

目錄

使用Astra Trident

設定後端

後端定義了Astra Trident與儲存系統之間的關係。它告訴Astra Trident如何與該儲存系統通訊、以及Astra Trident如何從該儲存系統配置磁碟區。Astra Trident會自動從後端提供符合儲存類別所定義需求的儲存資源池。深入瞭解如何根據您擁有的儲存系統類型來設定後端。

- "設定Azure NetApp Files 一個靜態後端"
- "設定Cloud Volumes Service AWS後端的功能"
- "設定Cloud Volumes Service 適用於Google Cloud Platform後端的功能"
- "設定NetApp HCI 一個不只是功能的SolidFire 後端"
- "使用ONTAP NetApp NAS驅動程式設定後端"
- "使用ONTAP SAN驅動程式設定後端"
- "使用Astra Trident搭配Amazon FSX for NetApp ONTAP 解決方案"

設定Azure NetApp Files 一個靜態後端

瞭Azure NetApp Files 解如何使用提供的範例組態、將靜態（anf）設定為Astra Trident安裝的後端。

該支援服務不支援低於100 GB的磁碟區。Azure NetApp Files如果要求較小的磁碟區、Astra Trident會自動建立100-GB磁碟區。

您需要的產品

若要設定及使用 "Azure NetApp Files" 後端、您需要下列項目：

- `subscriptionID` 透過啟用Azure NetApp Files 了支援功能的Azure訂閱。
- `tenantID`、 `clientID`和 `clientSecret` 從 "應用程式註冊" 在Azure Active Directory中、具備Azure NetApp Files 充分的權限執行此功能。應用程式註冊應使用 `Owner` 或 `Contributor` Azure預先定義的角色。

  若要深入瞭解Azure內建角色、請參閱 "Azure文件"。

- Azure `location` 至少包含一個 "委派的子網路"。
- 如果Azure NetApp Files 您是第一次使用或是在新的位置使用、則需要進行一些初始組態。請參閱 "快速入門指南"。

關於這項工作

Trident會根據後端組態（子網路、虛擬網路、服務層級和位置）、在所要求位置可用的容量集區上建立ANF磁碟區、並符合所要求的服務層級和子網路。

Astra Trident 21.04.0及更早版本不支援手動QoS容量資源池。

後端組態選項

如需後端組態選項、請參閱下表：

參數	說明	預設
`version`		永遠為1
`storageDriverName`	儲存驅動程式名稱	「Azure - NetApp-Files」
`backendName`	自訂名稱或儲存後端	驅動程式名稱+「_」+隨機字元
`subscriptionID`	Azure訂閱的訂閱ID	
`tenantID`	應用程式註冊的租戶ID	
`clientID`	應用程式註冊的用戶端ID	
`clientSecret`	應用程式註冊的用戶端機密	
`serviceLevel`	其中之一 `Standard`、 `Premium`或 `Ultra`	""（隨機）
`location`	要建立新磁碟區的Azure位置名稱	""（隨機）
`virtualNetwork`	具有委派子網路的虛擬網路名稱	""（隨機）
`subnet`	委派給的子網路名稱 `Microsoft.Netapp/volumes`	""（隨機）
`nfsMountOptions`	精細控制NFS掛載選項。	"nfsves=3"
`limitVolumeSize`	如果要求的磁碟區大小高於此值、則資源配置失敗	""（預設不強制執行）
`debugTraceFlags`	疑難排解時要使用的偵錯旗標。範例： `\{"api":false, "method":true}`。除非您正在進行疑難排解並需要詳細的記錄傾印、否則請勿使用此功能。	null

修改 `capacityPools` 在現有後端中的欄位中、如此可減少用於資源配置的容量資源池數量、進而產生孤立的磁碟區、這些磁碟區是在不屬於的容量資源池/資源池上進行資源配置 `capacityPools` 再列出一項。在這些孤立磁碟區上進行複製作業將會失敗。

如果嘗試建立永久虛擬基礎架構時發生「找不到容量資源池」錯誤、您的應用程式登錄可能沒有相關的必要權限和資源（子網路、虛擬網路、容量資源池）。Astra Trident會在啟用偵錯功能時、記錄在建立後端時所探索到的Azure資源。請務必檢查是否使用適當的角色。

如果您想要使用NFS 4.1版來掛載磁碟區、您可以加入 `nfsvers=4` 在以逗號分隔的掛載選項清單中、選擇NFS v4.1。儲存類別中設定的任何掛載選項、都會覆寫在後端組態檔中設定的掛載選項。

您可以在組態檔的特殊區段中指定下列選項、以控制預設的每個Volume佈建方式。請參閱下列組態範例。

參數	說明	預設
`exportRule`	新磁碟區的匯出規則	「0.00.0.0/0」
`size`	新磁碟區的預設大小	100公克

。 `exportRule` 值必須是以逗號分隔的清單、以CIDR表示法列出所有的IPv4位址或IPv4子網路組合。

對於在ANF後端上建立的所有磁碟區、Astra Trident會將儲存資源池上的所有標籤複製到資源配置時的儲存磁碟區。儲存管理員可以定義每個儲存資源池的標籤、並將儲存資源池中建立的所有磁碟區分組。這是根據後端組態中提供的一組可自訂標籤、方便區分磁碟區的方法。

範例**1**：最低組態

這是絕對最低的後端組態。有了這項組態、Astra Trident就能在全球各地探索所有NetApp帳戶、容量集區和委派給ANF的子網路、並隨機將新磁碟區放在其中一個上。

當您剛開始使用ANF並嘗試各種功能時、這種組態是理想的選擇、但實際上您想要為您所配置的磁碟區提供額外的範圍。

```
{
    "version": 1,
    "storageDriverName": "azure-netapp-files",
    "subscriptionID": "9f87c765-4774-fake-ae98-a721add45451",
    "tenantID": "68e4f836-edc1-fake-bff9-b2d865ee56cf",
    "clientID": "dd043f63-bf8e-fake-8076-8de91e5713aa",
    "clientSecret": "SECRET"
}
```

範例**2**：單一位置和特定服務層級組態

此後端組態可將Volume置於Azure中 `eastus` 位置在A `Premium` 容量資源池：Astra Trident會自動探索該位置委派給ANF的所有子網路、並隨機在其中一個磁碟區上放置新磁碟區。

```
    {
        "version": 1,
        "storageDriverName": "azure-netapp-files",
        "subscriptionID": "9f87c765-4774-fake-ae98-a721add45451",
        "tenantID": "68e4f836-edc1-fake-bff9-b2d865ee56cf",
        "clientID": "dd043f63-bf8e-fake-8076-8de91e5713aa",
        "clientSecret": "SECRET",
        "location": "eastus",
        "serviceLevel": "Premium"
    }
```

範例**3**：進階組態

此後端組態可進一步將磁碟區放置範圍縮小至單一子網路、並修改部分Volume資源配置預設值。

```
{
    "version": 1,
    "storageDriverName": "azure-netapp-files",
    "subscriptionID": "9f87c765-4774-fake-ae98-a721add45451",
    "tenantID": "68e4f836-edc1-fake-bff9-b2d865ee56cf",
    "clientID": "dd043f63-bf8e-fake-8076-8de91e5713aa",
    "clientSecret": "SECRET",
    "location": "eastus",
    "serviceLevel": "Premium",
    "virtualNetwork": "my-virtual-network",
    "subnet": "my-subnet",
    "nfsMountOptions": "vers=3,proto=tcp,timeo=600",
    "limitVolumeSize": "500Gi",
    "defaults": {
        "exportRule": "10.0.0.0/24,10.0.1.0/24,10.0.2.100",
        "size": "200Gi"
    }
}
```

範例**4**：虛擬儲存池組態

此後端組態可在單一檔案中定義多個儲存集區。當您有多個容量集區支援不同的服務層級、而且想要在Kubernetes中建立代表這些層級的儲存類別時、這很有用。

```
{
    "version": 1,
    "storageDriverName": "azure-netapp-files",
    "subscriptionID": "9f87c765-4774-fake-ae98-a721add45451",
    "tenantID": "68e4f836-edc1-fake-bff9-b2d865ee56cf",
    "clientID": "dd043f63-bf8e-fake-8076-8de91e5713aa",
    "clientSecret": "SECRET",
    "nfsMountOptions": "vers=3,proto=tcp,timeo=600",
    "labels": {
        "cloud": "azure"
    },
    "location": "eastus",

    "storage": [
        {
            "labels": {
                "performance": "gold"
            },
            "serviceLevel": "Ultra"
        },
        {
            "labels": {
                "performance": "silver"
            },
            "serviceLevel": "Premium"
        },
        {
            "labels": {
                "performance": "bronze"
            },
            "serviceLevel": "Standard",
        }
    ]
}
```

以下內容 `StorageClass` 定義請參閱上述儲存資源池。使用 `parameters.selector` 欄位中、您可以為每個欄位指定 `StorageClass` 用於裝載磁碟區的虛擬資源池。該磁碟區會在所選的資源池中定義各個層面。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze"
allowVolumeExpansion: true
```

接下來呢？

建立後端組態檔之後、請執行下列命令：

```
tridentctl create backend -f <backend-file>
```

如果後端建立失敗、表示後端組態有問題。您可以執行下列命令來檢視記錄、以判斷原因：

```
tridentctl logs
```

識別並修正組態檔的問題之後、您可以再次執行create命令。

設定AWS後端的CVS

瞭Cloud Volumes Service 解如何使用所提供的範例組態、將AWS的NetApp功能（CVS）設定為Astra Trident安裝的後端。

AWS適用的支援磁碟區容量不低於100 GB。Cloud Volumes Service如果要求較小的磁碟區、Trident會自動建立100-GB磁碟區。

您需要的產品

以設定及使用 "AWS 適用的 Cloud Volumes Service" 後端、您需要下列項目：

- 使用NetApp CVS設定的AWS帳戶
- CVS帳戶的API區域、URL和金鑰

後端組態選項

如需後端組態選項、請參閱下表：

參數	說明	預設
`version`		永遠為1
`storageDriverName`	儲存驅動程式名稱	「AWs-CVS」
`backendName`	自訂名稱或儲存後端	驅動程式名稱+「_」+ API金鑰的一部分
`apiRegion`	CVS帳戶區域。您可以在CVS入口網站的「帳戶設定/API存取」中找到該值。	
`apiURL`	CVS帳戶API URL。您可以在CVS入口網站的「帳戶設定/API存取」中找到該值。	
`apiKey`	CVS帳戶API金鑰。您可以在CVS入口網站的「帳戶設定/API存取」中找到該值。	
`secretKey`	CVS帳戶秘密金鑰。您可以在CVS入口網站的「帳戶設定/API存取」中找到該值。	
`proxyURL`	Proxy URL（如果需要代理伺服器才能連線至CVS帳戶）。Proxy伺服器可以是HTTP Proxy或HTTPS Proxy。對於HTTPS Proxy、會跳過憑證驗證、以允許在Proxy伺服器中使用自我簽署的憑證。不支援已啟用驗證的Proxy伺服器。	
`nfsMountOptions`	精細控制NFS掛載選項。	"nfsves=3"
`limitVolumeSize`	如果要求的磁碟區大小高於此值、則資源配置失敗	""（預設不強制執行）
`serviceLevel`	新磁碟區的CVS服務層級。這些值包括「標準」、「高級」和「極端」。	"標準"

參數	說明	預設
debugTraceFlags	疑難排解時要使用的偵錯旗標。範例： \{"api":false, "method":true} 。除非您正在進行疑難排解並需要詳細的記錄傾印、否則請勿使用此功能。	null

> apiURL 每個都是唯一的 apiRegion 。例如us-west-2 apiRegion 擁有 https://cv.us-west-2.netapp.com:8080/v1/ apiURL 。同樣地、us-east-1 apiRegion 擁有 https://cds-aws-bundles.netapp.com:8080/v1/ apiURL 。請務必檢查CVS儀表板是否正確 apiRegion 和 apiURL 後端組態的參數。

每個後端都會在單一AWS區域中配置磁碟區。若要在其他區域建立磁碟區、您可以定義其他後端。

您可以在組態檔的特殊區段中指定下列選項、以控制預設的每個Volume佈建方式。請參閱下列組態範例。

參數	說明	預設
exportRule	新磁碟區的匯出規則	「0.00.0.0/0」
snapshotDir	控制的可見度 .snapshot 目錄	"假"
snapshotReserve	保留給快照的磁碟區百分比	""（接受CVS預設值為0）
size	新磁碟區的大小	100公克

。 exportRule 值必須是以逗號分隔的清單、以CIDR表示法列出所有的IPv4位址或IPv4子網路組合。

> 對於在CVS AWS後端上建立的所有磁碟區、Astra Trident會將儲存資源池上的所有標籤複製到資源配置時的儲存磁碟區。儲存管理員可以定義每個儲存資源池的標籤、並將儲存資源池中建立的所有磁碟區分組。這是根據後端組態中提供的一組可自訂標籤、方便區分磁碟區的方法。

範例**1**：最低組態

這是絕對最低的後端組態。

當您剛開始使用CVS AWS並試用資料時、這種組態是理想的選擇、但實際上您想要為您所配置的磁碟區提供額外的範圍。

```
{
    "version": 1,
    "storageDriverName": "aws-cvs",
    "apiRegion": "us-east-1",
    "apiURL": "https://cds-aws-bundles.netapp.com:8080/v1",
    "apiKey": "znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE",
    "secretKey": "rR0rUmWXfNioN1KhtHisiSAnoTherboGuskey6pU"
}
```

範例2：單一服務層級組態

此範例顯示後端檔案、可將相同層面套用至AWS use-east-1區域中所有由Astra Trident建立的儲存設備。此範例也會顯示的用途 `proxyURL` 在後端檔案中。

```
{
    "version": 1,
    "storageDriverName": "aws-cvs",
    "backendName": "cvs-aws-us-east",
    "apiRegion": "us-east-1",
    "apiURL": "https://cds-aws-bundles.netapp.com:8080/v1",
    "apiKey": "znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE",
    "secretKey": "rR0rUmWXfNioN1KhtHisiSAnoTherboGuskey6pU",
    "proxyURL": "http://proxy-server-hostname/",
    "nfsMountOptions": "vers=3,proto=tcp,timeo=600",
    "limitVolumeSize": "50Gi",
    "serviceLevel": "premium",
    "defaults": {
        "snapshotDir": "true",
        "snapshotReserve": "5",
        "exportRule": "10.0.0.0/24,10.0.1.0/24,10.0.2.100",
        "size": "200Gi"
    }
}
```

範例3：虛擬儲存池組態

此範例顯示使用虛擬儲存資源池設定的後端定義檔案、以及參照回溯的StorageClass。

在下圖所示的範例後端定義檔中、會針對所有設定的儲存資源池設定特定的預設值 `snapshotReserve` 5%和 `exportRule` 至0.00.0/0。虛擬儲存集區是在中定義 `storage` 區段。在此範例中、每個個別的儲存資源池都會自行設定 `serviceLevel`和某些資源池會覆寫預設值。

```
{
    "version": 1,
    "storageDriverName": "aws-cvs",
    "apiRegion": "us-east-1",
    "apiURL": "https://cds-aws-bundles.netapp.com:8080/v1",
    "apiKey": "EnterYourAPIKeyHere**********************",
    "secretKey": "EnterYourSecretKeyHere******************",
    "nfsMountOptions": "vers=3,proto=tcp,timeo=600",

    "defaults": {
        "snapshotReserve": "5",
        "exportRule": "0.0.0.0/0"
    },
```

```
    "labels": {
        "cloud": "aws"
    },
    "region": "us-east-1",

    "storage": [
        {
            "labels": {
                "performance": "extreme",
                "protection": "extra"
            },
            "serviceLevel": "extreme",
            "defaults": {
                "snapshotDir": "true",
                "snapshotReserve": "10",
                "exportRule": "10.0.0.0/24"
            }
        },
        {
            "labels": {
                "performance": "extreme",
                "protection": "standard"
            },
            "serviceLevel": "extreme"
        },
        {
            "labels": {
                "performance": "premium",
                "protection": "extra"
            },
            "serviceLevel": "premium",
            "defaults": {
                "snapshotDir": "true",
                "snapshotReserve": "10"
            }
        },

        {
            "labels": {
                "performance": "premium",
                "protection": "standard"
            },
            "serviceLevel": "premium"
        },
```

```
        {
            "labels": {
                "performance": "standard"
            },
            "serviceLevel": "standard"
        }
    ]
}
```

下列StorageClass定義係指上述儲存資源池。使用 `parameters.selector` 欄位中、您可以為每個StorageClass指定用於裝載Volume的虛擬集區。該磁碟區會在所選的資源池中定義各個層面。

第一個StorageClass (`cvs-extreme-extra-protection`）對應至第一個虛擬儲存資源池。這是唯一提供極致效能、快照保留率為10%的資源池。最後一個StorageClass (`cvs-extra-protection`）撥出提供快照保留10%的任何儲存資源池。Astra Trident決定選取哪個虛擬儲存池、並確保符合快照保留需求。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=extreme; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=extreme; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
```

```
  name: cvs-premium
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: netapp.io/trident
parameters:
  selector: "performance=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "protection=extra"
allowVolumeExpansion: true
```

接下來呢？

建立後端組態檔之後、請執行下列命令：

```
tridentctl create backend -f <backend-file>
```

如果後端建立失敗、表示後端組態有問題。您可以執行下列命令來檢視記錄、以判斷原因：

```
tridentctl logs
```

識別並修正組態檔的問題之後、您可以再次執行create命令。

針對GCP後端設定CVS

瞭Cloud Volumes Service 解如何使用所提供的範例組態、將NetApp for Google Cloud Platform（GCP）設定為Astra Trident安裝的後端。

適用於Google Cloud的NetApp Cloud Volumes Service 支援的CVS效能磁碟區大小不得低於100 GiB、或CVS磁碟區大小不得低於300 GiB。如果所要求的磁碟區小於最小大小、Astra Trident會自動建立最小大小的磁碟區。

您需要的產品

以設定及使用 "適用於 Google Cloud Cloud Volumes Service" 後端、您需要下列項目：

- 使用NetApp CVS設定的Google Cloud帳戶
- Google Cloud帳戶的專案編號
- Google Cloud服務帳戶 `netappcloudvolumes.admin` 角色
- CVS服務帳戶的API金鑰檔

Astra Trident現在支援預設的較小磁碟區 "服務類型"： [GCP^] 上的 CVS 服務類型。用於建立的後端 `storageClass=software`、現在、磁碟區的資源配置大小最小可達300 GiB。CVS目前在「管制可用度」下提供此功能、並不提供技術支援。使用者必須註冊才能存取低於1TiB的磁碟區 "請按這裡"。NetApp建議客戶使用低於1TiB的磁碟區來處理*非正式作業*的工作負載。

使用預設的CVS服務類型部署後端 (`storageClass=software`) 、使用者必須取得GCP上有關專案編號和專案ID的子1TiB Volume功能存取權。這是Astra Trident配置子1TiB磁碟區所需的功能。如果沒有、則低於600 GiB的PVCS將無法建立Volume。使用取得對低於1TiB磁碟區的存取權 "這份表格"。

由Astra Trident針對預設CVS服務層級所建立的磁碟區、將會配置如下：

- 小於300 GiB的PVCS會導致Astra Trident建立300 GiB CVS Volume。
- 介於300 GiB到600 GiB之間的PVCS會導致Astra Trident建立一個所需大小的CVS Volume。
- 介於600 GiB和1 TiB之間的PVCS會導致Astra Trident建立1TiB CVS Volume。
- 大於1 TiB的PVCS會導致Astra Trident建立所需大小的CVS Volume。

後端組態選項

如需後端組態選項、請參閱下表：

參數	說明	預設
`version`		永遠為1
`storageDriverName`	儲存驅動程式名稱	「GCP-CVS」
`backendName`	自訂名稱或儲存後端	驅動程式名稱+「_」+ API金鑰的一部分
`storageClass`	儲存類型：任您選擇 `hardware` （效能最佳化）或 `software` （CVS服務類型）	
`projectNumber`	Google Cloud帳戶專案編號。此值可在Google Cloud入口網站的首頁找到。	
`apiRegion`	CVS帳戶區域。這是後端配置磁碟區的區域。	

參數	說明	預設
apiKey	的Google Cloud服務帳戶API金鑰 netappcloudvolumes.admin 角色：其中包含Google Cloud服務帳戶私密金鑰檔案（逐字複製到後端組態檔）的JSON-格式內容。	
proxyURL	Proxy URL（如果需要代理伺服器才能連線至CVS帳戶）。Proxy伺服器可以是HTTP Proxy或HTTPS Proxy。對於HTTPS Proxy、會跳過憑證驗證、以允許在Proxy伺服器中使用自我簽署的憑證。不支援已啟用驗證的Proxy伺服器。	
nfsMountOptions	精細控制NFS掛載選項。	"nfsves=3"
limitVolumeSize	如果要求的磁碟區大小高於此值、則資源配置失敗	""（預設不強制執行）
serviceLevel	新磁碟區的CVS服務層級。這些值包括「標準」、「高級」和「極端」。	"標準"
network	用於CVS磁碟區的GCP網路	「預設」
debugTraceFlags	疑難排解時要使用的偵錯旗標。範例： \{"api":false, "method":true}。除非您正在進行疑難排解並需要詳細的記錄傾印、否則請勿使用此功能。	null

如果使用共享的VPC網路、兩者都是 projectNumber 和 hostProjectNumber 必須指定。在這種情況下、projectNumber 是服務專案、以及 hostProjectNumber 是主機專案。

。 apiRegion 代表Astra Trident建立CVS磁碟區的GCP區域。Astra Trident可在屬於同一個GCP區域的Kubernetes節點上掛載及附加磁碟區。建立後端時、務必確保 apiRegion 符合Kubernetes節點部署的區域。建立跨區域Kubernetes叢集時、會在指定的中建立CVS磁碟區 apiRegion 只能用於排程在相同GCP區域的節點上的工作負載。

storageClass 是選用參數、可用來選取所需的參數 "CVS服務類型"。您可以從基礎CVS服務類型中進行選擇 (storageClass=software）或CVS效能服務類型 (storageClass=hardware）、這是Trident預設使用的功能。請務必指定 apiRegion 提供各自的CVS storageClass 在後端定義中。

Astra Trident與Google Cloud上的基礎CVS服務類型整合、是一項*測試版功能、不適用於正式作業工作負載。Trident 完全支援 CVS效能服務類型、並依預設使用。

每個後端都會在單一Google Cloud區域中配置磁碟區。若要在其他區域建立磁碟區、您可以定義其他後端。

您可以在組態檔的特殊區段中指定下列選項、以控制預設的每個Volume佈建方式。請參閱下列組態範例。

參數	說明	預設
exportRule	新磁碟區的匯出規則	「0.00.0.0/0」
snapshotDir	存取 .snapshot 目錄	"假"
snapshotReserve	保留給快照的磁碟區百分比	""（接受CVS預設值為0）
size	新磁碟區的大小	「100Gi」

。 exportRule 值必須是以逗號分隔的清單、以CIDR表示法列出所有的IPv4位址或IPv4子網路組合。

針對在CVS Google Cloud後端上建立的所有磁碟區、Trident會在儲存資源池上的所有標籤配置時複製到儲存磁碟區。儲存管理員可以定義每個儲存資源池的標籤、並將儲存資源池中建立的所有磁碟區分組。這是根據後端組態中提供的一組可自訂標籤、方便區分磁碟區的方法。

範例1：最低組態

這是絕對最低的後端組態。

```
{
    "version": 1,
    "storageDriverName": "gcp-cvs",
    "projectNumber": "012345678901",
    "apiRegion": "us-west2",
    "apiKey": {
        "type": "service_account",
        "project_id": "my-gcp-project",
        "private_key_id": "1234567890123456789012345678901234567890",
        "private_key": "-----BEGIN PRIVATE KEY-----
\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZ
srrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisI
sAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSa
PIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZN
chRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlz
ZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl
/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kw
s8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY
9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHc
zZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHi
sIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOgu
SaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyA
ZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz
lzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3
bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4
Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5o
jY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nzn
HczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrt
```

```
HisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbO
guSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKe
yAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRA
GzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4j
K3bl/qp8B4Kws8zX5ojY9m\nXsYg6gyxy4zq7OlwWgLwGa==\n-----END PRIVATE
KEY-----\n",
        "client_email": "cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com",
        "client_id": "123456789012345678901",
        "auth_uri": "https://accounts.google.com/o/oauth2/auth",
        "token_uri": "https://oauth2.googleapis.com/token",
        "auth_provider_x509_cert_url":
"https://www.googleapis.com/oauth2/v1/certs",
        "client_x509_cert_url":
"https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com"
    }
}
```

範例2：基礎CVS服務類型組態

此範例顯示使用基本CVS服務類型的後端定義、此服務類型適用於一般用途的工作負載、提供輕度/中度效能、以及高分區可用度。

```
{
    "version": 1,
    "storageDriverName": "gcp-cvs",
    "projectNumber": "012345678901",
    "storageClass": "software",
    "apiRegion": "us-east4",
    "apiKey": {
        "type": "service_account",
        "project_id": "my-gcp-project",
        "private_key_id": "1234567890123456789012345678901234567890",
        "private_key": "-----BEGIN PRIVATE KEY-----
\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZ
srrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisI
sAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSa
PIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZN
chRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlz
ZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl
/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kw
s8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY
9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHc
zZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHi
sIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOgu
```

```
SaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyA
ZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz
lzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3
bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4
Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5o
jY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nzn
HczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrt
HisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbO
guSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKe
yAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRA
GzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4j
K3bl/qp8B4Kws8zX5ojY9m\nXsYg6gyxy4zq7OlwWgLwGa==\n-----END PRIVATE
KEY-----\n",
        "client_email": "cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com",
        "client_id": "123456789012345678901",
        "auth_uri": "https://accounts.google.com/o/oauth2/auth",
        "token_uri": "https://oauth2.googleapis.com/token",
        "auth_provider_x509_cert_url":
"https://www.googleapis.com/oauth2/v1/certs",
        "client_x509_cert_url":
"https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com"
    }
}
```

範例3：單一服務層級組態

此範例顯示後端檔案、可將相同層面套用至Google Cloud us-west2區域中所有由Astra Trident建立的儲存設備。此範例也會顯示的用途 `proxyURL` 在後端組態檔中。

```
{
    "version": 1,
    "storageDriverName": "gcp-cvs",
    "projectNumber": "012345678901",
    "apiRegion": "us-west2",
    "apiKey": {
        "type": "service_account",
        "project_id": "my-gcp-project",
        "private_key_id": "1234567890123456789012345678901234567890",
        "private_key": "-----BEGIN PRIVATE KEY-----
\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZ
srrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisI
sAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSa
PIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZN
```

```
chRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlz
ZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl
/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kw
s8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY
9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHc
zZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHi
sIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOgu
SaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyA
ZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz
lzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3
bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4
Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5o
jY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nzn
HczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrt
HisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbO
guSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKe
yAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRA
GzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4j
K3bl/qp8B4Kws8zX5ojY9m\nXsYg6gyxy4zq7OlwWgLwGa==\n-----END PRIVATE
KEY-----\n",
        "client_email": "cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com",
        "client_id": "123456789012345678901",
        "auth_uri": "https://accounts.google.com/o/oauth2/auth",
        "token_uri": "https://oauth2.googleapis.com/token",
        "auth_provider_x509_cert_url":
"https://www.googleapis.com/oauth2/v1/certs",
        "client_x509_cert_url":
"https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com"
    },
    "proxyURL": "http://proxy-server-hostname/",
    "nfsMountOptions": "vers=3,proto=tcp,timeo=600",
    "limitVolumeSize": "10Ti",
    "serviceLevel": "premium",
    "defaults": {
        "snapshotDir": "true",
        "snapshotReserve": "5",
        "exportRule": "10.0.0.0/24,10.0.1.0/24,10.0.2.100",
        "size": "5Ti"
    }
}
```

範例4：虛擬儲存池組態

此範例顯示使用虛擬儲存資源池設定的後端定義檔案 `StorageClasses` 請回頭參考。

在下圖所示的範例後端定義檔中、會針對所有設定的儲存資源池設定特定的預設值 `snapshotReserve` 5%和 `exportRule` 至0.00.0/0。虛擬儲存集區是在中定義 `storage` 區段。在此範例中、每個個別的儲存資源池都會自行設定 `serviceLevel`和某些資源池會覆寫預設值。

```
{
    "version": 1,
    "storageDriverName": "gcp-cvs",
    "projectNumber": "012345678901",
    "apiRegion": "us-west2",
    "apiKey": {
        "type": "service_account",
        "project_id": "my-gcp-project",
        "private_key_id": "1234567890123456789012345678901234567890",
        "private_key": "-----BEGIN PRIVATE KEY-----
\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZ
srrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisI
sAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSa
PIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZN
chRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlz
ZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl
/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kw
s8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY
9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHc
zZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHi
sIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOgu
SaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyA
ZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz
lzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3
bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4
Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5o
jY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nzn
HczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrt
HisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbO
guSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKe
yAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRA
GzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\nznHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4j
K3bl/qp8B4Kws8zX5ojY9m\nXsYg6gyxy4zq7OlwWgLwGa==\n-----END PRIVATE
KEY-----\n",
        "client_email": "cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com",
        "client_id": "123456789012345678901",
        "auth_uri": "https://accounts.google.com/o/oauth2/auth",
        "token_uri": "https://oauth2.googleapis.com/token",
        "auth_provider_x509_cert_url":
"https://www.googleapis.com/oauth2/v1/certs",
        "client_x509_cert_url":
```

```
"https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com"
    },
    "nfsMountOptions": "vers=3,proto=tcp,timeo=600",

    "defaults": {
        "snapshotReserve": "5",
        "exportRule": "0.0.0.0/0"
    },

    "labels": {
        "cloud": "gcp"
    },
    "region": "us-west2",

    "storage": [
        {
            "labels": {
                "performance": "extreme",
                "protection": "extra"
            },
            "serviceLevel": "extreme",
            "defaults": {
                "snapshotDir": "true",
                "snapshotReserve": "10",
                "exportRule": "10.0.0.0/24"
            }
        },
        {
            "labels": {
                "performance": "extreme",
                "protection": "standard"
            },
            "serviceLevel": "extreme"
        },
        {
            "labels": {
                "performance": "premium",
                "protection": "extra"
            },
            "serviceLevel": "premium",
            "defaults": {
                "snapshotDir": "true",
                "snapshotReserve": "10"
            }
        },
```

```
        {
            "labels": {
                "performance": "premium",
                "protection": "standard"
            },
            "serviceLevel": "premium"
        },

        {
            "labels": {
                "performance": "standard"
            },
            "serviceLevel": "standard"
        }
    ]
}
```

下列StorageClass定義係指上述儲存資源池。使用 `parameters.selector` 欄位中、您可以為每個StorageClass指定用於裝載Volume的虛擬集區。該磁碟區會在所選的資源池中定義各個層面。

第一個StorageClass (`cvs-extreme-extra-protection`）對應至第一個虛擬儲存資源池。這是唯一提供極致效能、快照保留率為10%的資源池。最後一個StorageClass (`cvs-extra-protection`）撥出提供快照保留10%的任何儲存資源池。Astra Trident決定選取哪個虛擬儲存池、並確保符合快照保留需求。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=extreme; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
```

```
  name: cvs-premium-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: netapp.io/trident
parameters:
  selector: "performance=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "protection=extra"
allowVolumeExpansion: true
```

接下來呢？

建立後端組態檔之後、請執行下列命令：

```
tridentctl create backend -f <backend-file>
```

如果後端建立失敗、表示後端組態有問題。您可以執行下列命令來檢視記錄、以判斷原因：

```
tridentctl logs
```

識別並修正組態檔的問題之後、您可以再次執行create命令。

設定NetApp HCI 一個不只是功能的SolidFire 後端

瞭解如何在安裝Astra Trident時建立及使用元素後端。

您需要的產品

- 支援的儲存系統、可執行Element軟體。
- 提供給NetApp HCI / SolidFire叢集管理員或租戶使用者的認證、以管理磁碟區。
- 您所有的Kubernetes工作節點都應該安裝適當的iSCSI工具。請參閱 "工作節點準備資訊"。

您需要知道的資訊

。 `solidfire-san` 儲存驅動程式支援兩種Volume模式：檔案和區塊。適用於 `Filesystem` 磁碟區代碼、Astra Trident會建立磁碟區並建立檔案系統。檔案系統類型由StorageClass指定。

驅動程式	傳輸協定	**Volume**模式	支援的存取模式	支援的檔案系統
`solidfire-san`	iSCSI	區塊	rwo、ROX、rwx	無檔案系統。原始區塊裝置。
`solidfire-san`	iSCSI	區塊	rwo、ROX、rwx	無檔案系統。原始區塊裝置。
`solidfire-san`	iSCSI	檔案系統	Rwo、ROX	`xfs、 ext3、 ext4`
`solidfire-san`	iSCSI	檔案系統	Rwo、ROX	`xfs、 ext3、 ext4`

Astra Trident在做為增強型的csi資源配置程式時使用CHAP。如果您使用的是CHAP（這是「csi」的預設值）、則不需要進一步準備。建議明確設定 `UseCHAP` 可選擇搭配非csi Trident使用CHAP。否則請參閱 "請按這裡"。

Volume存取群組僅受Astra Trident的傳統非csi架構支援。當Astra Trident設定為以「csi」模式運作時、會使用CHAP。

否則 `AccessGroups` 或 `UseCHAP` 已設定、適用下列其中一項規則：

- 如果是預設值 `trident` 偵測到存取群組、使用存取群組。
- 如果未偵測到存取群組、且Kubernetes版本為1.7或更新版本、則會使用CHAP。

後端組態選項

如需後端組態選項、請參閱下表：

參數	說明	預設
`version`		永遠為1
`storageDriverName`	儲存驅動程式名稱	永遠是「solidfire-san」

參數	說明	預設
backendName	自訂名稱或儲存後端	「S指_」+儲存設備（iSCSI）IP位址SolidFire
Endpoint	MVIP、適用於SolidFire 採用租戶認證的不含用戶身分證明的叢集	
SVIP	儲存設備（iSCSI）IP位址和連接埠	
labels	套用到磁碟區的任意JSON-格式化標籤集。	「」
TenantName	要使用的租戶名稱（如果找不到、請建立）	
InitiatorIFace	將iSCSI流量限制在特定的主機介面	「預設」
UseCHAP	使用CHAP驗證iSCSI	是的
AccessGroups	要使用的存取群組ID清單	尋找名為「Trident」的存取群組ID
Types	QoS規格	
limitVolumeSize	如果要求的磁碟區大小高於此值、則資源配置失敗	「」（預設不強制執行）
debugTraceFlags	疑難排解時要使用的偵錯旗標。範例： {"API"：假、「方法」：true }	null

請勿使用 debugTraceFlags 除非您正在疑難排解並需要詳細的記錄傾印。

對於所有建立的磁碟區、Astra Trident會在儲存資源池上的所有標籤配置時、複製到備用儲存LUN。儲存管理員可以定義每個儲存資源池的標籤、並將儲存資源池中建立的所有磁碟區分組。這是根據後端組態中提供的一組可自訂標籤、方便區分磁碟區的方法。

範例**1**：的後端組態 solidfire-san 三種磁碟區類型的驅動程式

此範例顯示使用CHAP驗證的後端檔案、並建立具有特定QoS保證的三種Volume類型模型。您很可能會定義儲存類別、以便使用來使用這些類別 IOPS 儲存類別參數。

```
{
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://<user>:<password>@<mvip>/json-rpc/8.0",
    "SVIP": "<svip>:3260",
    "TenantName": "<tenant>",
    "labels": {"k8scluster": "dev1", "backend": "dev1-element-cluster"},
    "UseCHAP": true,
    "Types": [{"Type": "Bronze", "Qos": {"minIOPS": 1000, "maxIOPS": 2000,
"burstIOPS": 4000}},
             {"Type": "Silver", "Qos": {"minIOPS": 4000, "maxIOPS": 6000,
"burstIOPS": 8000}},
             {"Type": "Gold", "Qos": {"minIOPS": 6000, "maxIOPS": 8000,
"burstIOPS": 10000}}]
}
```

範例2：的後端與儲存類別組態 `solidfire-san` 驅動程式搭配虛擬儲存資源池

此範例顯示使用虛擬儲存資源池設定的後端定義檔案、以及參照回溯的StorageClass。

在下圖所示的範例後端定義檔中、會針對所有設定的儲存資源池設定特定的預設值 `type` 銀級。虛擬儲存集區是在中定義 `storage` 區段。在此範例中、有些儲存資源池會設定自己的類型、有些資源池則會覆寫上述設定的預設值。

```
{
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://<user>:<password>@<mvip>/json-rpc/8.0",
    "SVIP": "<svip>:3260",
    "TenantName": "<tenant>",
    "UseCHAP": true,
    "Types": [{"Type": "Bronze", "Qos": {"minIOPS": 1000, "maxIOPS": 2000,
"burstIOPS": 4000}},
             {"Type": "Silver", "Qos": {"minIOPS": 4000, "maxIOPS": 6000,
"burstIOPS": 8000}},
             {"Type": "Gold", "Qos": {"minIOPS": 6000, "maxIOPS": 8000,
"burstIOPS": 10000}}],

    "type": "Silver",
    "labels":{"store":"solidfire", "k8scluster": "dev-1-cluster"},
    "region": "us-east-1",

    "storage": [
        {
            "labels":{"performance":"gold", "cost":"4"},
            "zone":"us-east-1a",
            "type":"Gold"
        },
        {
            "labels":{"performance":"silver", "cost":"3"},
            "zone":"us-east-1b",
            "type":"Silver"
        },
        {
            "labels":{"performance":"bronze", "cost":"2"},
            "zone":"us-east-1c",
            "type":"Bronze"
        },
        {
            "labels":{"performance":"silver", "cost":"1"},
            "zone":"us-east-1d"
        }
    ]
}
```

下列StorageClass定義係指上述虛擬儲存資源池。使用 `parameters.selector` 欄位中、每個StorageClass會呼叫哪些虛擬資源池可用於裝載Volume。磁碟區將會在所選的虛擬資源池中定義各個層面。

第一個StorageClass (`solidfire-gold-four`）將對應至第一個虛擬儲存資源池。這是唯一提供黃金級效能的

資源池 `Volume Type QoS` 金級。最後一個StorageClass (`solidfire-silver`）撥出任何提供銀級效能的儲存資源池。Astra Trident將決定選取哪個虛擬儲存資源池、並確保符合儲存需求。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold; cost=4"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=3"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze; cost=2"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=1"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
  fsType: "ext4"
```

如需詳細資訊、請參閱

- "Volume存取群組"

使用ONTAP SAN驅動程式設定後端

深入瞭解如何使用ONTAP 支援支援功能的SAN驅動程式來設定支援功能的後端。ONTAP

- "準備"
- "組態與範例"

使用者權限

Astra Trident希望以ONTAP 支援的形式執行、通常是以支援的方式執行 `admin` 叢集使用者或 `vsadmin` SVM使用者、或具有相同角色之不同名稱的使用者。對於Amazon FSX for NetApp ONTAP 支援的NetApp功能、Astra Trident預期會以ONTAP 使用叢集的形式執行、以執行支援或SVM管理員的身分 `fsxadmin` 使用者或 `vsadmin` SVM使用者、或具有相同角色之不同名稱的使用者。。 `fsxadmin` 使用者是叢集管理使用者的有限替代。

> 如果您使用 `limitAggregateUsage` 參數：需要叢集管理權限。當使用Amazon FSX for NetApp ONTAP 時、搭配Astra Trident `limitAggregateUsage` 參數無法搭配使用 `vsadmin` 和 `fsxadmin` 使用者帳戶：如果您指定此參數、組態作業將會失敗。

準備

瞭解如何準備使用ONTAP 支援不支援的SAN驅動程式來設定支援功能的後端。ONTAP對於所有ONTAP 的不支援端點、Astra Trident至少需要指派一個集合體給SVM。

請記住、您也可以執行多個驅動程式、並建立指向一個或多個驅動程式的儲存類別。例如、您可以設定 `san-dev` 使用的類別 `ontap-san` 驅動程式與 `san-default` 使用的類別 `ontap-san-economy` 一、

您所有的Kubernetes工作節點都必須安裝適當的iSCSI工具。請參閱 "請按這裡" 以取得更多詳細資料。

驗證

Astra Trident提供兩種驗ONTAP 證功能來驗證支援的後端。

- 認證型：ONTAP 對具備所需權限的使用者名稱和密碼。建議使用預先定義的安全登入角色、例如 `admin` 或 `vsadmin` 以確保與ONTAP 更新版本的最大相容性。
- 憑證型：Astra Trident也能ONTAP 使用安裝在後端的憑證與某個叢集進行通訊。在此處、後端定義必須包含用戶端憑證、金鑰及信任的CA憑證（建議使用）的Base64編碼值。

使用者也可以選擇更新現有的後端、選擇從認證移至憑證型、反之亦然。如果*同時提供認證資料和憑證*、Astra Trident將預設使用憑證、同時發出警告、從後端定義中移除認證資料。

啟用認證型驗證

Astra Trident需要SVM範圍/叢集範圍管理員的認證資料、才能與ONTAP 該後端進行通訊。建議使用預先定義的標準角色、例如 `admin` 或 `vsadmin`。這可確保與未來ONTAP 的支援版本保持前瞻相容、因為未來的Astra Trident版本可能會使用功能API。您可以建立自訂的安全登入角色、並與Astra Trident搭配使用、但不建議使用。

後端定義範例如下所示：

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "secret",
}
```

請記住、後端定義是唯一以純文字儲存認證的位置。建立後端之後、使用者名稱/密碼會以Base64編碼、並儲存為Kubernetes機密。建立/更新後端是唯一需要知道認證資料的步驟。因此、這是一項純管理員操作、由Kubernetes /儲存管理員執行。

啟用憑證型驗證

新的和現有的後端可以使用憑證、並與ONTAP 該後端通訊。後端定義需要三個參數。

- 用戶端憑證：用戶端憑證的Base64編碼值。
- 用戶端私密金鑰：關聯私密金鑰的Base64編碼值。
- 信任的CACertifate：受信任CA憑證的Base64編碼值。如果使用信任的CA、則必須提供此參數。如果未使用信任的CA、則可忽略此問題。

典型的工作流程包括下列步驟。

步驟

1. 產生用戶端憑證和金鑰。產生時、請將Common Name（CN）（一般名稱（CN））設定為ONTAP 驗證身分。

   ```
   openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
   -out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=admin"
   ```

2. 將信任的CA憑證新增ONTAP 至整個叢集。這可能已由儲存管理員處理。如果未使用信任的CA、請忽略。

   ```
   security certificate install -type server -cert-name <trusted-ca-cert-
   name> -vserver <vserver-name>
   ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
   true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
   <cert-authority>
   ```

3. 在ONTAP 支援叢集上安裝用戶端憑證和金鑰（步驟1）。

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. 確認ONTAP 支援的不安全登入角色 `cert` 驗證方法。

```
security login create -user-or-group-name admin -application ontapi
-authentication-method cert
security login create -user-or-group-name admin -application http
-authentication-method cert
```

5. 使用產生的憑證測試驗證。以ONTAP Management LIF IP和SVM名稱取代<SfManagement LIF>和<vserver name>。

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. 使用Base64編碼憑證、金鑰和信任的CA憑證。

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. 使用從上一步取得的值建立後端。

```
$ cat cert-backend.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"trustedCACertificate": "QNFinfO...SiqOyN",
"storagePrefix": "myPrefix_"
}

$ tridentctl create backend -f cert-backend.json -n trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       0 |
+------------+----------------+--------------------------------------
+--------+---------+
```

更新驗證方法或旋轉認證資料

您可以更新現有的後端、以使用不同的驗證方法、或是旋轉其認證資料。這兩種方法都可行：使用使用者名稱/密碼的後端可更新以使用憑證；使用憑證的後端可更新為使用者名稱/密碼。若要這麼做、請使用更新的 `backend.json` 包含所需執行參數的檔案 `tridentctl backend update`。

```
$ cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "secret",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
$ tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

當您旋轉密碼時、儲存管理員必須先更新ONTAP 使用者的密碼（位於BIOS）。接著是後端更新。在循環憑證時、可將多個憑證新增至使用者。然後更新後端以使用新的憑證、之後可從ONTAP 該叢集刪除舊的憑證。

更新後端不會中斷對已建立之磁碟區的存取、也不會影響之後建立的磁碟區連線。成功的後端更新顯示Astra Trident可以與ONTAP 該後端通訊、並處理未來的Volume作業。

指定**igroup**

Astra Trident使用igroup來控制其所配置的磁碟區（LUN）存取。系統管理員在指定後端的igroup時有兩種選擇：

- Astra Trident可自動建立及管理每個後端的igroup。如果 `igroupName` 未包含在後端定義中、Astra Trident會建立名為的igroup `trident-<backend-UUID>` 在SVM上。如此可確保每個後端都有專屬的igroup、並處理Kubernetes節點IQN的自動新增/刪除作業。
- 或者、也可以在後端定義中提供預先建立的igroup。您可以使用來完成此作業 `igroupName` 組態參數。Astra Trident會將Kubernetes節點IQN新增/刪除至預先存在的igroup。

適用於具有的後端 `igroupName` 定義 `igroupName` 可以使用刪除 `tridentctl backend update` 使用Astra Trident自動處理igroup。這不會中斷對已附加至工作負載之磁碟區的存取。未來的連線將使用建立的igroup

Astra Trident來處理。

> 針對每個獨特的Astra Trident執行個體指定igroup是最適合Kubernetes管理員和儲存管理員的最佳實務做法。「csi Trident」可自動新增及移除igroup的叢集節點IQN、大幅簡化其管理。在Kubernetes環境中使用相同的SVM（以及Astra Trident安裝）時、使用專屬的igroup可確保對Kubernetes叢集所做的變更不會影響與其他叢集相關的igroup。此外、也必須確保Kubernetes叢集中的每個節點都有唯一的IQN。如上所述、Astra Trident會自動處理IQN的新增與移除。重複使用主機間的IQN可能會導致主機彼此誤用、並拒絕存取LUN的不良情況。

如果將Astra Trident設定為使用「csi資源配置程式」、則Kubernetes節點IQN會自動新增至igroup或從其中移除。當節點新增至Kubernetes叢集時、 `trident-csi` 示範集部署Pod (`trident-csi-xxxxx`）在新增的節點上、登錄可附加磁碟區的新節點。節點IQN也會新增至後端的igroup。當節點封鎖、排放及從Kubernetes刪除時、類似的一組步驟可處理刪除IQN。

如果Astra Trident並未以csi資源配置程式的形式執行、則必須手動更新igroup、以包含Kubernetes叢集中每個工作節點的iSCSI IQN。加入Kubernetes叢集的節點IQN必須新增至igroup。同樣地、從Kubernetes叢集移除的節點IQN也必須從igroup移除。

使用雙向**CHAP**驗證連線

Astra Trident可以使用雙向CHAP驗證iSCSI工作階段 `ontap-san` 和 `ontap-san-economy` 驅動程式：這需要啟用 `useCHAP` 選項。設定為時 `true`Astra Trident將SVM的預設啟動器安全性設定為雙向CHAP、並從後端檔案設定使用者名稱和機密。NetApp建議使用雙向CHAP來驗證連線。請參閱下列組態範例：

```
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "ontap_san_chap",
    "managementLIF": "192.168.0.135",
    "svm": "ontap_iscsi_svm",
    "useCHAP": true,
    "username": "vsadmin",
    "password": "FaKePaSsWoRd",
    "igroupName": "trident",
    "chapInitiatorSecret": "cl9qxIm36DKyawxy",
    "chapTargetInitiatorSecret": "rqxigXgkesIpwxyz",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
}
```

> 。 `useCHAP` 參數是布林選項、只能設定一次。預設值設為假。將其設為true之後、您就無法將其設為假。

此外 `useCHAP=true`、 `chapInitiatorSecret`、 `chapTargetInitiatorSecret`、 `chapTargetUsername`和 `chapUsername` 欄位必須包含在後端定義中。執行建立後端後端之後、即可變更機密資訊 `tridentctl update`。

運作方式

透過設定 `useCHAP` 為真、儲存管理員指示Astra Trident在儲存後端上設定CHAP。這包括下列項目：

- 在SVM上設定CHAP：
  - 如果SVM的預設啟動器安全性類型為無（預設設定）和、則磁碟區中沒有已存在的預先存在LUN、Astra Trident會將預設安全性類型設為 `CHAP` 並繼續設定CHAP啟動器和目標使用者名稱和機密。
  - 如果SVM包含LUN、Astra Trident將不會在SVM上啟用CHAP。如此可確保不限制存取SVM上已存在的LUN。
- 設定CHAP啟動器和目標使用者名稱和機密；這些選項必須在後端組態中指定（如上所示）。
- 管理在中新增的init1 `igroupName` 在後端中提供。如果未指定、則預設為 `trident`。

建立後端之後、Astra Trident會建立對應的 `tridentbackend` 將CHAP機密與使用者名稱儲存為Kubernetes機密。由Astra Trident在此後端上建立的所有PV、都會掛載並附加於CHAP上。

旋轉認證資料並更新後端

您可以更新中的CHAP參數來更新CHAP認證 `backend.json` 檔案：這需要更新CHAP機密並使用 `tridentctl update` 命令以反映這些變更。

> 更新後端的CHAP機密時、您必須使用 `tridentctl` 以更新後端。請勿透過CLI/ONTAP UI更新儲存叢集上的認證資料、因為Astra Trident無法接受這些變更。

```
$ cat backend-san.json
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "ontap_san_chap",
    "managementLIF": "192.168.0.135",
    "svm": "ontap_iscsi_svm",
    "useCHAP": true,
    "username": "vsadmin",
    "password": "FaKePaSsWoRd",
    "igroupName": "trident",
    "chapInitiatorSecret": "cl9qxUpDaTeD",
    "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
}

$ ./tridentctl update backend ontap_san_chap -f backend-san.json -n
trident
+----------------+----------------+--------------------------------------
+--------+---------+
|   NAME         | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+----------------+----------------+--------------------------------------
+--------+---------+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffbeb5c |
online |       7 |
+----------------+----------------+--------------------------------------
+--------+---------+
```

現有的連線不會受到影響；如果SVM上的Astra Trident更新認證、它們將繼續保持作用中狀態。新連線將使用更新的認證資料、而現有連線仍保持作用中狀態。中斷舊PV的連線並重新連線、將會使用更新的認證資料。

組態選項與範例

瞭解如何透過ONTAP Astra Trident安裝來建立及使用支援NetApp的SAN驅動程式。本節提供後端組態範例、以及如何將後端對應至StorageClass的詳細資料。

後端組態選項

如需後端組態選項、請參閱下表：

參數	說明	預設
`version`		永遠為1

參數	說明	預設
`storageDriverName`	儲存驅動程式名稱	「ONTAP-NAS」、「ONTAP-NAS-節約 型」、「ONTAP-NAS-flexgroup」、「ONTAP-SAN」、「ONTAP-san經濟型」
`backendName`	自訂名稱或儲存後端	驅動程式名稱+「_」+ dataLIF
`managementLIF`	叢集或SVM管理LIF的IP位址	「10.0.0.1」、「[2001:1234:abcd:::fefo]」
`dataLIF`	傳輸協定LIF的IP位址。IPv6使用方括弧。設定後無法更新	除非另有說明、否則由SVM衍生
`useCHAP`	使用CHAP驗證iSCSI以供ONTAP支援不支援的SAN驅動程式使用[布林值]	錯
`chapInitiatorSecret`	CHAP啟動器密碼。必要條件 `useCHAP=true`	「」
`labels`	套用到磁碟區的任意JSON-格式化標籤集	「」
`chapTargetInitiatorSecret`	CHAP目標啟動器機密。必要條件 `useCHAP=true`	「」
`chapUsername`	傳入使用者名稱。必要條件 `useCHAP=true`	「」
`chapTargetUsername`	目標使用者名稱。必要條件 `useCHAP=true`	「」
`clientCertificate`	用戶端憑證的Base64編碼值。用於憑證型驗證	「」
`clientPrivateKey`	用戶端私密金鑰的Base64編碼值。用於憑證型驗證	「」
`trustedCACertificate`	受信任CA憑證的Base64編碼值。選用。用於憑證型驗證	「」
`username`	連線至叢集/ SVM的使用者名稱。用於認證型驗證	「」
`password`	連線至叢集/ SVM的密碼。用於認證型驗證	「」
`svm`	要使用的儲存虛擬機器	如果是SVM則衍生 `managementLIF` 已指定
`igroupName`	要使用之SAN磁碟區的igroup名稱	「Trident -<後端-UUID>」
`storagePrefix`	在SVM中配置新磁碟區時所使用的前置碼。設定後無法更新	「Trident」
`limitAggregateUsage`	如果使用率高於此百分比、則無法進行資源配置。*不適用於Amazon FSX for ONTAP Sfor Sfor *	「」（預設不強制執行）

參數	說明	預設
limitVolumeSize	如果要求的磁碟區大小高於此值、則資源配置失敗。	「」（預設不強制執行）
lunsPerFlexvol	每FlexVol 個LUN的最大LUN數量、範圍必須在[50、200]	「100」
debugTraceFlags	疑難排解時要使用的偵錯旗標。範例： {"API"：假、「方法」：true }	null

若要與ONTAP 此叢集通訊、您應該提供驗證參數。這可能是安全登入或安裝憑證的使用者名稱/密碼。

如果您使用Amazon FSX for NetApp ONTAP Sendbackend、請勿指定 limitAggregateUsage 參數。。 fsxadmin 和 vsadmin Amazon FSX for NetApp ONTAP 的角色不包含擷取Aggregate使用量及透過Astra Trident加以限制所需的存取權限。

請勿使用 debugTraceFlags 除非您正在疑難排解並需要詳細的記錄傾印。

適用於 ontap-san 驅動程式、預設為使用SVM的所有資料LIF IP、並使用iSCSI多重路徑。為的指定dataLIF 的IP位址 ontap-san 驅動程式會強制他們停用多重路徑、並只使用指定的位址。

建立後端時、請記住這一點 dataLIF 和 storagePrefix 無法在建立後修改。若要更新這些參數、您需要建立新的後端。

igroupName 可設定為ONTAP 已在叢集上建立的igroup。如果未指定、Astra Trident會自動建立名為Trident 的igroup -<後端UUID >。如果提供預先定義的igroupName、則如果要在環境之間共用SVM、NetApp建議使用每個Kubernetes叢集的igroup。這是Astra Trident自動維護IQN新增/刪除的必要條件。

後端也可以在建立後更新igroup：

- 可以更新igroupName、以指向在Astra Trident以外的SVM上建立及管理的新igroup。
- 可以省略igroupName。在此案例中、Astra Trident會自動建立及管理Trident -<backend-UUUUID> igroup。

在這兩種情況下、仍可繼續存取Volume附件。未來的Volume附件將使用更新的igroup。此更新不會中斷對後端磁碟區的存取。

您可以為指定完整網域名稱（FQDN） managementLIF 選項。

`managementLIF` 對於所有ONTAP 的版本、也可以將所有的版本設定為IPv6位址。請務必使用安裝Trident `--use-ipv6` 旗標。必須謹慎定義 `managementLIF` 方括弧內的IPv6位址。

使用IPv6位址時、請務必確認 managementLIF 和 dataLIF （如果包含在後端定義中）是在方括弧內定義、例如[28e8：d9fb：a825：b7bf：69a8：d02f：9e7b：3555]。如果 dataLIF 未提供、Astra Trident會從SVM擷取IPv6資料lifs。

若要讓ONTAP-SAN驅動程式使用CHAP、請設定 useCHAP 參數至 true 在後端定義中。然後Astra Trident會設

定並使用雙向CHAP做為後端所指定SVM的預設驗證。請參閱 "請按這裡" 以瞭解其運作方式。

適用於 `ontap-san-economy` 驅動程式 `limitVolumeSize` 選項也會限制其管理的qtree和LUN磁碟區大小上限。

> Astra Trident會在使用建立的所有磁碟區的「Comments」（註解）欄位中設定資源配置標籤 `ontap-san` 驅動程式：針對所建立的每個Volume、FlexVol 將會在顯示於其儲存資源池中的「Comments」（註解）欄位中填入所有標籤。儲存管理員可以定義每個儲存資源池的標籤、並將儲存資源池中建立的所有磁碟區分組。這是根據後端組態中提供的一組可自訂標籤、方便區分磁碟區的方法。

用於資源配置磁碟區的後端組態選項

您可以在組態的特定區段中、使用這些選項來控制預設配置每個Volume的方式。如需範例、請參閱下列組態範例。

參數	說明	預設
`spaceAllocation`	LUN的空間分配	「真的」
`spaceReserve`	空間保留模式；「無」（精簡）或「Volume」（完整）	「無」
`snapshotPolicy`	要使用的Snapshot原則	「無」
`qosPolicy`	要指派給所建立磁碟區的QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy	「」
`adaptiveQosPolicy`	要指派給所建立磁碟區的調適性QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy	「」
`snapshotReserve`	保留給快照「0」的磁碟區百分比	如果 `snapshotPolicy` 為「無」、否則為「」
`splitOnClone`	建立複本時、從其父複本分割複本	「假」
`splitOnClone`	建立複本時、從其父複本分割複本	「假」
`encryption`	啟用NetApp Volume加密	「假」
`securityStyle`	新磁碟區的安全樣式	「UNIX」
`tieringPolicy`	分層原則以使用「無」	ONTAP 9.5之前的SVM-DR組態為「純快照」

> 搭配Astra Trident使用QoS原則群組需要ONTAP 使用更新版本的版本。建議使用非共用的QoS原則群組、並確保原則群組會個別套用至每個組成群組。共享的QoS原則群組將強制所有工作負載的總處理量上限。

以下是已定義預設值的範例：

```
{
 "version": 1,
 "storageDriverName": "ontap-san",
 "managementLIF": "10.0.0.1",
 "dataLIF": "10.0.0.2",
 "svm": "trident_svm",
 "username": "admin",
 "password": "password",
 "labels": {"k8scluster": "dev2", "backend": "dev2-sanbackend"},
 "storagePrefix": "alternate-trident",
 "igroupName": "custom",
 "debugTraceFlags": {"api":false, "method":true},
 "defaults": {
     "spaceReserve": "volume",
     "qosPolicy": "standard",
     "spaceAllocation": "false",
     "snapshotPolicy": "default",
     "snapshotReserve": "10"
 }
}
```

針對使用建立的所有Volume `ontap-san` 驅動程式Astra Trident在FlexVol 支援LUN中繼資料的過程中、額外增加10%的容量。LUN的配置大小與使用者在PVc中要求的大小完全相同。Astra Trident在FlexVol 整個過程中增加10%的速度（顯示ONTAP 在畫面上可用的尺寸）。使用者現在可以取得所要求的可用容量。此變更也可防止LUN成為唯讀、除非可用空間已充分利用。這不適用於ONTAP-san經濟型。

用於定義的後端 `snapshotReserve`、Astra Trident會依照下列方式計算Volume大小：

```
Total volume size = [(PVC requested size) / (1 - (snapshotReserve
percentage) / 100)] * 1.1
```

1.1是額外10%的Astra Trident加入FlexVol 到the支援LUN中繼資料的功能。適用於 `snapshotReserve` = 5%、而PVC要求= 5GiB、磁碟區總大小為5.79GiB、可用大小為5.5GiB。。 `volume show` 命令應顯示類似以下範例的結果：

```
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
          _pvc_89f1c156_3801_4de4_9f9d_034d54c395f4
                       online     RW         10GB     5.00GB    0%
          _pvc_e42ec6fe_3baa_4af6_996d_134adbbb8e6d
                       online     RW       5.79GB     5.50GB    0%
          _pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba
                       online     RW          1GB    511.8MB    0%
3 entries were displayed.
```

目前、只有調整大小、才能將新計算用於現有的Volume。

下列範例顯示基本組態、讓大部分參數保留預設值。這是定義後端最簡單的方法。

> 如果您在NetApp ONTAP 支援Astra Trident的NetApp上使用Amazon FSX、建議您指定lifs的DNS名稱、而非IP位址。

`ontap-san` 具有憑證型驗證的驅動程式

這是最小的後端組態範例。 `clientCertificate`、 `clientPrivateKey`和 `trustedCACertificate`（選用、如果使用信任的CA）會填入 `backend.json` 並分別取得用戶端憑證、私密金鑰及信任CA憑證的基礎64編碼值。

```
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "DefaultSANBackend",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.3",
    "svm": "svm_iscsi",
    "useCHAP": true,
    "chapInitiatorSecret": "cl9qxIm36DKyawxy",
    "chapTargetInitiatorSecret": "rqxigXgkesIpwxyz",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
    "igroupName": "trident",
    "clientCertificate": "ZXR0ZXJwYXB...ICMgJ3BhcGVyc2",
    "clientPrivateKey": "vciwKIyAgZG...0cnksIGRlc2NyaX",
    "trustedCACertificate": "zcyBbaG...b3Igb3duIGNsYXNz"
}
```

`ontap-san` 使用雙向**CHAP**的驅動程式

這是最小的後端組態範例。此基本組態會建立 `ontap-san` 後端 `useCHAP` 設定為 `true`。

```
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.3",
    "svm": "svm_iscsi",
    "labels": {"k8scluster": "test-cluster-1", "backend": "testcluster1-
sanbackend"},
    "useCHAP": true,
    "chapInitiatorSecret": "cl9qxIm36DKyawxy",
    "chapTargetInitiatorSecret": "rqxigXgkesIpwxyz",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
    "igroupName": "trident",
    "username": "vsadmin",
    "password": "secret"
}
```

`ontap-san-economy` 驅動程式

```
{
    "version": 1,
    "storageDriverName": "ontap-san-economy",
    "managementLIF": "10.0.0.1",
    "svm": "svm_iscsi_eco",
    "useCHAP": true,
    "chapInitiatorSecret": "cl9qxIm36DKyawxy",
    "chapTargetInitiatorSecret": "rqxigXgkesIpwxyz",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
    "igroupName": "trident",
    "username": "vsadmin",
    "password": "secret"
}
```

虛擬儲存資源池的後端範例

在下圖所示的範例後端定義檔案中、會針對所有儲存資源池設定特定的預設值、例如 `spaceReserve` 無、`spaceAllocation` 假、和 `encryption` 錯。虛擬儲存資源池是在儲存區段中定義。

在此範例中、有些儲存資源池會自行設定 `spaceReserve`、 `spaceAllocation`和 `encryption` 值、部分集區會覆寫上述設定的預設值。

```
{
```

```
    "version": 1,
    "storageDriverName": "ontap-san",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.3",
    "svm": "svm_iscsi",
    "useCHAP": true,
    "chapInitiatorSecret": "cl9qxIm36DKyawxy",
    "chapTargetInitiatorSecret": "rqxigXgkesIpwxyz",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
    "igroupName": "trident",
    "username": "vsadmin",
    "password": "secret",

    "defaults": {
          "spaceAllocation": "false",
          "encryption": "false",
          "qosPolicy": "standard"
    },
    "labels":{"store": "san_store", "kubernetes-cluster": "prod-cluster-
1"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"protection":"gold", "creditpoints":"40000"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceAllocation": "true",
                "encryption": "true",
                "adaptiveQosPolicy": "adaptive-extreme"
            }
        },
        {
            "labels":{"protection":"silver", "creditpoints":"20000"},
            "zone":"us_east_1b",
            "defaults": {
                "spaceAllocation": "false",
                "encryption": "true",
                "qosPolicy": "premium"
            }
        },
        {
            "labels":{"protection":"bronze", "creditpoints":"5000"},
            "zone":"us_east_1c",
            "defaults": {
                "spaceAllocation": "true",
```

```
                "encryption": "false"
            }
        }
    ]
}
```

以下是的iSCSI範例 `ontap-san-economy` 驅動程式：

```
{
    "version": 1,
    "storageDriverName": "ontap-san-economy",
    "managementLIF": "10.0.0.1",
    "svm": "svm_iscsi_eco",
    "useCHAP": true,
    "chapInitiatorSecret": "cl9qxIm36DKyawxy",
    "chapTargetInitiatorSecret": "rqxigXgkesIpwxyz",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
    "igroupName": "trident",
    "username": "vsadmin",
    "password": "secret",

    "defaults": {
          "spaceAllocation": "false",
          "encryption": "false"
    },
    "labels":{"store":"san_economy_store"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"app":"oracledb", "cost":"30"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceAllocation": "true",
                "encryption": "true"
            }
        },
        {
            "labels":{"app":"postgresdb", "cost":"20"},
            "zone":"us_east_1b",
            "defaults": {
                "spaceAllocation": "false",
                "encryption": "true"
            }
        },
```

```
        {
            "labels":{"app":"mysqldb", "cost":"10"},
            "zone":"us_east_1c",
            "defaults": {
                "spaceAllocation": "true",
                "encryption": "false"
            }
        }
    ]
}
```

將後端對應至**StorageClass**

下列StorageClass定義係指上述虛擬儲存資源池。使用 `parameters.selector` 欄位中、每個StorageClass會呼叫哪些虛擬資源池可用於裝載Volume。磁碟區將會在所選的虛擬資源池中定義各個層面。

- 第一個StorageClass (`protection-gold`）將對應至中的第一個、第二個虛擬儲存資源池 `ontap-nas-flexgroup` 後端與中的第一個虛擬儲存資源池 `ontap-san` 後端：這是唯一提供金級保護的資源池。
- 第二個StorageClass (`protection-not-gold`）將對應至中的第三、第四個虛擬儲存資源池 `ontap-nas-flexgroup` 中的後端和第二個、第三個虛擬儲存資源池 `ontap-san` 後端：這是唯一提供金級以外保護層級的資源池。
- 第三個StorageClass (`app-mysqldb`）將對應至中的第四個虛擬儲存資源池 `ontap-nas` 中的後端和第三個虛擬儲存資源池 `ontap-san-economy` 後端：這些是唯一提供mysqldb類型應用程式儲存池組態的集區。
- 第四個StorageClass (`protection-silver-creditpoints-20k`）將對應至中的第三個虛擬儲存資源池 `ontap-nas-flexgroup` 中的後端和第二個虛擬儲存資源池 `ontap-san` 後端：這些資源池是唯一能以20000個信用點數提供金級保護的資源池。
- 第五個StorageClass (`creditpoints-5k`）將對應至中的第二個虛擬儲存資源池 `ontap-nas-economy` 中的後端和第三個虛擬儲存資源池 `ontap-san` 後端：這些是唯一提供5000個信用點數的資源池產品。

Astra Trident將決定選取哪個虛擬儲存資源池、並確保符合儲存需求。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: netapp.io/trident
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: netapp.io/trident
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: netapp.io/trident
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

使用ONTAP NetApp NAS驅動程式設定後端

瞭解如何使用ONTAP NetApp NAS驅動程式設定功能的功能。ONTAP

- "準備"
- "組態與範例"

使用者權限

Astra Trident希望以ONTAP 支援的形式執行、通常是以支援的方式執行 `admin` 叢集使用者或 `vsadmin` SVM使用者、或具有相同角色之不同名稱的使用者。對於Amazon FSX for NetApp ONTAP 支援的NetApp功能、Astra Trident預期會以ONTAP 使用叢集的形式執行、以執行支援或SVM管理員的身分 `fsxadmin` 使用者或 `vsadmin` SVM使用者、或具有相同角色之不同名稱的使用者。。 `fsxadmin` 使用者是叢集管理使用者的有限替代。

> 如果您使用 `limitAggregateUsage` 參數：需要叢集管理權限。當使用Amazon FSX for NetApp ONTAP 時、搭配Astra Trident `limitAggregateUsage` 參數無法搭配使用 `vsadmin` 和 `fsxadmin` 使用者帳戶：如果您指定此參數、組態作業將會失敗。

準備

瞭解如何準備使用ONTAP 不含NetApp功能的NAS驅動程式來設定功能完善的後端。ONTAP對於所有ONTAP 的不支援端點、Astra Trident至少需要指派一個集合體給SVM。

對於所有ONTAP 的不支援端點、Astra Trident至少需要指派一個集合體給SVM。

請記住、您也可以執行多個驅動程式、並建立指向一個或多個驅動程式的儲存類別。例如、您可以設定使用的Gold類別 `ontap-nas` 驅動程式和銅級、使用 `ontap-nas-economy` 一、

您所有的Kubernetes工作節點都必須安裝適當的NFS工具。請參閱 "請按這裡" 以取得更多詳細資料。

驗證

Astra Trident提供兩種驗ONTAP 證功能來驗證支援的後端。

- 認證型：ONTAP 對具備所需權限的使用者名稱和密碼。建議使用預先定義的安全登入角色、例如 `admin` 或 `vsadmin` 以確保與ONTAP 更新版本的最大相容性。
- 憑證型：Astra Trident也能ONTAP 使用安裝在後端的憑證與某個叢集進行通訊。在此處、後端定義必須包含用戶端憑證、金鑰及信任的CA憑證（建議使用）的Base64編碼值。

使用者也可以選擇更新現有的後端、選擇從認證移至憑證型、反之亦然。如果*同時提供認證資料和憑證*、Astra Trident將預設使用憑證、同時發出警告、從後端定義中移除認證資料。

啟用認證型驗證

Astra Trident需要SVM範圍/叢集範圍管理員的認證資料、才能與ONTAP 該後端進行通訊。建議使用預先定義的標準角色、例如 `admin` 或 `vsadmin`。這可確保與未來ONTAP 的支援版本保持前瞻相容、因為未來的Astra Trident版本可能會使用功能API。您可以建立自訂的安全登入角色、並與Astra Trident搭配使用、但不建議使用。

後端定義範例如下所示：

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "secret"
}
```

請記住、後端定義是唯一以純文字儲存認證的位置。建立後端之後、使用者名稱/密碼會以Base64編碼、並儲存為Kubernetes機密。建立/更新後端是唯一需要知道認證資料的步驟。因此、這是一項純管理員操作、由Kubernetes /儲存管理員執行。

啟用憑證型驗證

新的和現有的後端可以使用憑證、並與ONTAP 該後端通訊。後端定義需要三個參數。

- 用戶端憑證：用戶端憑證的Base64編碼值。
- 用戶端私密金鑰：關聯私密金鑰的Base64編碼值。
- 信任的CACertifate：受信任CA憑證的Base64編碼值。如果使用信任的CA、則必須提供此參數。如果未使用信任的CA、則可忽略此問題。

典型的工作流程包括下列步驟。

步驟

1. 產生用戶端憑證和金鑰。產生時、請將Common Name（CN）（一般名稱（CN））設定為ONTAP 驗證身分。

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. 將信任的CA憑證新增ONTAP 至整個叢集。這可能已由儲存管理員處理。如果未使用信任的CA、請忽略。

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. 在ONTAP 支援叢集上安裝用戶端憑證和金鑰（步驟1）。

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. 確認ONTAP 支援的不安全登入角色 `cert` 驗證方法。

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

5. 使用產生的憑證測試驗證。以ONTAP Management LIF IP和SVM名稱取代<SfManagement LIF>和<vserver name>。您必須確保LIF的服務原則設定為 `default-data-management`。

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. 使用Base64編碼憑證、金鑰和信任的CA憑證。

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. 使用從上一步取得的值建立後端。

```
$ cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
$ tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

更新驗證方法或旋轉認證資料

您可以更新現有的後端、以使用不同的驗證方法、或是旋轉其認證資料。這兩種方法都可行：使用使用者名稱/密碼的後端可更新以使用憑證；使用憑證的後端可更新為使用者名稱/密碼。若要這麼做、請使用更新的 `backend.json` 包含所需執行參數的檔案 `tridentctl backend update`。

```
$ cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "secret",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
$ tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

當您旋轉密碼時、儲存管理員必須先更新ONTAP 使用者的密碼（位於BIOS）。接著是後端更新。在循環憑證時、可將多個憑證新增至使用者。然後更新後端以使用新的憑證、之後可從ONTAP 該叢集刪除舊的憑證。

更新後端不會中斷對已建立之磁碟區的存取、也不會影響之後建立的磁碟區連線。成功的後端更新顯示Astra Trident可以與ONTAP 該後端通訊、並處理未來的Volume作業。

管理**NFS**匯出原則

Astra Trident使用NFS匯出原則來控制其所配置之磁碟區的存取。

使用匯出原則時、Astra Trident提供兩種選項：

- Astra Trident可動態管理匯出原則本身；在此作業模式中、儲存管理員會指定代表可接受IP位址的CIDR區塊清單。Astra Trident會自動將這些範圍內的節點IP新增至匯出原則。或者、如果未指定CIDR、則會將節點上找到的任何全域範圍單點傳送IP新增至匯出原則。
- 儲存管理員可以建立匯出原則、並手動新增規則。除非在組態中指定不同的匯出原則名稱、否則Astra Trident會使用預設的匯出原則。

動態管理匯出原則

「csi Trident」的20.04版提供動態管理輸出原則的能力ONTAP 、以利實現幕後。這可讓儲存管理員為工作節點IP指定允許的位址空間、而非手動定義明確的規則。它可大幅簡化匯出原則管理；修改匯出原則不再需要在儲存叢集上進行手動介入。此外、這有助於將儲存叢集的存取限制在指定範圍內有IP的工作者節點、以支援精簡且自動化的管理。

只有「csi Trident」才能動態管理匯出原則。請務必確保工作節點未被NATed。

範例

必須使用兩種組態選項。以下是後端定義範例：

```
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "ontap_nas_auto_export,
    "managementLIF": "192.168.0.135",
    "svm": "svm1",
    "username": "vsadmin",
    "password": "FaKePaSsWoRd",
    "autoExportCIDRs": ["192.168.0.0/24"],
    "autoExportPolicy": true
}
```

使用此功能時、您必須確保SVM中的根連接點具有預先設定的匯出原則、並具有允許節點CIDR區塊（例如預設匯出原則）的匯出規則。請務必遵循NetApp建議的最佳實務做法、為Astra Trident指定SVM。

以下是使用上述範例說明此功能的運作方式：

- `autoExportPolicy` 設為 `true`。這表示Astra Trident將為建立匯出原則 `svm1` 並使用來處理新增和刪除規則的作業 `autoExportCIDRs` 位址區塊。例如、UUID為403b5326-8482-40dB/96d0-d83fb3f4daec和的後端 `autoExportPolicy` 設定為 `true` 建立名為的匯出原則 `trident-403b5326-8482-40db-96d0-d83fb3f4daec` 在SVM上。
- `autoExportCIDRs` 包含位址區塊清單。此欄位為選用欄位、預設為「0.00.0.0/0」、「：/0」。如果未定義、Astra Trident會新增在工作者節點上找到的所有全域範圍單點傳送位址。

在此範例中 `192.168.0.0/24` 提供位址空間。這表示、屬於此位址範圍的Kubernetes節點IP將新增至Astra Trident所建立的匯出原則。當Astra Trident登錄其執行的節點時、會擷取節點的IP位址、並對照中提供的位址區塊來檢查這些位址 `autoExportCIDRs`。篩選IP之後、Astra Trident會針對所探索的用戶端IP建立匯出原則規則、並針對所識別的每個節點建立一個規則。

您可以更新 `autoExportPolicy` 和 `autoExportCIDRs` 建立後端後端。您可以為自動管理或刪除現有CIDR的後端附加新的CIDR。刪除CIDR時請務必謹慎、以確保不會中斷現有的連線。您也可以選擇停用 `autoExportPolicy` 用於後端、然後回到手動建立的匯出原則。這需要設定 `exportPolicy` 參數。

在Astra Trident建立或更新後端之後、您可以使用檢查後端 `tridentctl` 或對應的 `tridentbackend` 客戶需

求日：

```
$ ./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileSystemType: ext4
```

當節點新增至Kubernetes叢集並向Astra Trident控制器登錄時、會更新現有後端的匯出原則（前提是它們位於中指定的位址範圍內） `autoExportCIDRs` （後端）。

移除節點時、Astra Trident會檢查所有線上的後端、以移除節點的存取規則。Astra Trident將此節點IP從託管後端的匯出原則中移除、可防止惡意掛載、除非叢集中的新節點重複使用此IP。

對於先前現有的後端、請使用更新後端 `tridentctl update backend` 將確保Astra Trident自動管理匯出原則。這會建立以後端UUID命名的新匯出原則、而後端上的磁碟區會在重新掛載時使用新建立的匯出原則。

> ⓘ 刪除具有自動管理匯出原則的後端、將會刪除動態建立的匯出原則。如果重新建立後端、則會將其視為新的後端、並導致建立新的匯出原則。

如果即時節點的IP位址已更新、您必須重新啟動節點上的Astra Trident Pod。Astra Trident接著會更新其管理的後端匯出原則、以反映此IP變更。

組態選項與範例

瞭解如何透過ONTAP Astra Trident安裝來建立及使用NetApp NAS驅動程式。本節提供後端組態範例、以及如何將後端對應至StorageClass的詳細資料。

後端組態選項

如需後端組態選項、請參閱下表：

參數	說明	預設
`version`		永遠為1
`storageDriverName`	儲存驅動程式名稱	「ONTAP-NAS」、「ONTAP-NAS-節約 型」、「ONTAP-NAS-flexgroup」、「ONTAP-SAN」、「ONTAP-san經濟型」
`backendName`	自訂名稱或儲存後端	驅動程式名稱+「_」+ dataLIF
`managementLIF`	叢集或SVM管理LIF的IP位址	「10.0.0.1」、「[2001:1234:abcd:::fefo]」
`dataLIF`	傳輸協定LIF的IP位址。IPv6使用方括弧。設定後無法更新	除非另有說明、否則由SVM衍生
`autoExportPolicy`	啟用自動匯出原則建立與更新[布 林值]	錯
`autoExportCIDRs`	根據時間篩選Kubernetes節點IP的CIDR清單 `autoExportPolicy` 已啟用	[「0.00.0/0」、「：/0」]
`labels`	套用到磁碟區的任意JSON-格式化標籤集	「」
`clientCertificate`	用戶端憑證的Base64編碼值。用於憑證型驗證	「」
`clientPrivateKey`	用戶端私密金鑰的Base64編碼值。用於憑證型驗證	「」
`trustedCACertificate`	受信任CA憑證的Base64編碼值。選用。用於憑證型驗證	「」
`username`	連線至叢集/ SVM的使用者名稱。用於認證型驗證	
`password`	連線至叢集/ SVM的密碼。用於認證型驗證	
`svm`	要使用的儲存虛擬機器	如果是SVM則衍生 `managementLIF` 已指定
`igroupName`	要使用之SAN磁碟區的igroup名稱	「Trident -<後端-UUID>」
`storagePrefix`	在SVM中配置新磁碟區時所使用的前置碼。設定後無法更新	「Trident」
`limitAggregateUsage`	如果使用率高於此百分比、則無法進行資源配置。*不適用於Amazon FSX for ONTAP Sfor Sfor *	「」（預設不強制執行）
`limitVolumeSize`	如果要求的磁碟區大小高於此值、則資源配置失敗。	「」（預設不強制執行）
`lunsPerFlexvol`	每FlexVol 個LUN的最大LUN數量、範圍必須在[50、200]	「100」

參數	說明	預設
debugTraceFlags	疑難排解時要使用的偵錯旗標。範例： {"API"：假、「方法」：true }	null
nfsMountOptions	以逗號分隔的NFS掛載選項清單	「」
qtreesPerFlexvol	每FlexVol 個邊的最大qtree數、必須在範圍內[50、300]	「200」
useREST	使用ONTAP Isrest API的布林參數。技術預覽	錯

useREST 以*技術預覽*的形式提供、建議用於測試環境、而非用於正式作業工作負載。設定為時 true、Astra Trident將使用ONTAP 靜止API與後端進行通訊。此功能需要ONTAP 使用更新版本的版本9.8。此外ONTAP 、所使用的登入角色必須能夠存取 ontap 應用程式：這是預先定義的 vsadmin 和 cluster-admin 角色：

若要與ONTAP 此叢集通訊、您應該提供驗證參數。這可能是安全登入或安裝憑證的使用者名稱/密碼。

如果您使用Amazon FSX for NetApp ONTAP Sendbackend、請勿指定 limitAggregateUsage 參數。。 fsxadmin 和 vsadmin Amazon FSX for NetApp ONTAP 的角色不包含擷取Aggregate使用量及透過Astra Trident加以限制所需的存取權限。

請勿使用 debugTraceFlags 除非您正在疑難排解並需要詳細的記錄傾印。

建立後端時、請記住 dataLIF 和 storagePrefix 無法在建立後修改。若要更新這些參數、您需要建立新的後端。

您可以為指定完整網域名稱（FQDN） managementLIF 選項。也可以為指定FQDN dataLIF 選項、此時FQDN將用於NFS掛載作業。如此一來、您就能建立循環DNS、在多個資料生命期之間實現負載平衡。

`managementLIF` 對於所有ONTAP 的版本、也可以將所有的版本設定為IPv6位址。請務必使用安裝Astra Trident `--use-ipv6` 旗標。必須謹慎定義 `managementLIF` 方括弧內的IPv6位址。

使用IPv6位址時、請務必確認 managementLIF 和 dataLIF （如果包含在後端定義中）是在方括弧內定義、例如[28e8：d9fb：a825：b7bf：69a8：d02f：9e7b：3555]。如果 dataLIF 未提供、Astra Trident會從SVM擷取IPv6資料lifs。

使用 autoExportPolicy 和 autoExportCIDRs 選項：「csi Trident」可自動管理匯出原則。所有的ONTAP-NAS-*驅動程式均支援此功能。

適用於 ontap-nas-economy 驅動程式 limitVolumeSize 選項也會限制其管理的qtree和LUN及的最大磁碟區大小 qtreesPerFlexvol 選項可自訂每FlexVol 個支援區的配額樹數上限。

。 nfsMountOptions 參數可用於指定掛載選項。Kubernetes持續磁碟區的掛載選項通常會在儲存類別中指定、但如果儲存類別中未指定掛載選項、則Astra Trident會改回使用儲存後端組態檔中指定的掛載選項。如果儲

存類別或組態檔中未指定掛載選項、則Astra Trident不會在相關的持續磁碟區上設定任何掛載選項。

> Astra Trident會在使用建立的所有磁碟區的「Comments」（註解）欄位中設定資源配置標籤(ontap-nas 和(ontap-nas-flexgroup。根據所使用的驅動程式、意見會設定在FlexVol 這個功能上 (ontap-nas）FlexGroup 或 (ontap-nas-flexgroup）。Astra Trident會在儲存資源池配置時、將儲存資源池上的所有標籤複製到儲存磁碟區。儲存管理員可以定義每個儲存資源池的標籤、並將儲存資源池中建立的所有磁碟區分組。這是根據後端組態中提供的一組可自訂標籤、方便區分磁碟區的方法。

用於資源配置磁碟區的後端組態選項

您可以在組態的特定區段中、使用這些選項來控制預設配置每個Volume的方式。如需範例、請參閱下列組態範例。

參數	說明	預設
spaceAllocation	LUN的空間分配	「真的」
spaceReserve	空間保留模式；「無」（精簡）或「Volume」（完整）	「無」
snapshotPolicy	要使用的Snapshot原則	「無」
qosPolicy	要指派給所建立磁碟區的QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy	「」
adaptiveQosPolicy	要指派給所建立磁碟區的調適性QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy。不受ONTAP-NAS-經濟支援。	「」
snapshotReserve	保留給快照「0」的磁碟區百分比	如果 snapshotPolicy 為「無」、否則為「」
splitOnClone	建立複本時、從其父複本分割複本	「假」
encryption	啟用NetApp Volume加密	「假」
securityStyle	新磁碟區的安全樣式	「UNIX」
tieringPolicy	分層原則以使用「無」	ONTAP 9.5之前的SVM-DR組態為「純快照」
unix權限	新磁碟區的模式	「777」
snapshotDir	控制的可見度 .snapshot 目錄	「假」
匯出原則	要使用的匯出原則	「預設」
安全性樣式	新磁碟區的安全樣式	「UNIX」

> 搭配Astra Trident使用QoS原則群組需要ONTAP 使用更新版本的版本。建議使用非共用的QoS原則群組、並確保原則群組會個別套用至每個組成群組。共享的QoS原則群組將強制所有工作負載的總處理量上限。

以下是已定義預設值的範例：

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "customBackendName",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "labels": {"k8scluster": "dev1", "backend": "dev1-nasbackend"},
  "svm": "trident_svm",
  "username": "cluster-admin",
  "password": "password",
  "limitAggregateUsage": "80%",
  "limitVolumeSize": "50Gi",
  "nfsMountOptions": "nfsvers=4",
  "debugTraceFlags": {"api":false, "method":true},
  "defaults": {
    "spaceReserve": "volume",
    "qosPolicy": "premium",
    "exportPolicy": "myk8scluster",
    "snapshotPolicy": "default",
    "snapshotReserve": "10"
  }
}
```

適用於 `ontap-nas` 和 `ontap-nas-flexgroups`Astra Trident現在使用新的計算方法、確保FlexVol利用snapshotReserve百分比和PVc正確調整尺寸。當使用者要求使用PVCs時、Astra Trident會FlexVol 使用新的計算方式、建立原始的包含更多空間的候選區。此計算可確保使用者在永久虛擬磁碟中獲得所要求的可寫入空間、且空間不得小於所要求的空間。在v21.07之前、當使用者要求使用PVC（例如5GiB）、快照保留區達到50%時、他們只能獲得2.5GiB的可寫入空間。這是因為使用者要求的是整個Volume和 `snapshotReserve 佔此比例。使用Trident 21.07時、使用者要求的是可寫入空間、而Astra Trident定義了 `snapshotReserve 數字表示整個Volume的百分比。這不適用於 `ontap-nas-economy`。請參閱下列範例以瞭解此功能的運作方式：

計算方式如下：

```
Total volume size = (PVC requested size) / (1 - (snapshotReserve
percentage) / 100)
```

對於snapshotReserve = 50%、而PVc要求= 5GiB、磁碟區總大小為2/0.5 = 10GiB、可用大小為5GiB、這是使用者在PVC要求中要求的大小。。 `volume show` 命令應顯示類似以下範例的結果：

在升級Astra Trident時、先前安裝的現有後端會按照上述說明來配置磁碟區。對於在升級之前建立的磁碟區、您應該調整其磁碟區大小、以便觀察變更。例如、採用的2GiB PVC `snapshotReserve=50` 先前產生的磁碟區提供1GiB的可寫入空間。例如、將磁碟區大小調整為3GiB、可讓應用程式在6 GiB磁碟區上擁有3GiB的可寫入空間。

最低組態範例

下列範例顯示基本組態、讓大部分參數保留預設值。這是定義後端最簡單的方法。

> 如果您在NetApp ONTAP 支援Trident的NetApp支援上使用Amazon FSX、建議您指定lifs的DNS名稱、而非IP位址。

`ontap-nas` 具有憑證型驗證的驅動程式

這是最小的後端組態範例。 `clientCertificate`、 `clientPrivateKey`和 `trustedCACertificate`（選用、如果使用信任的CA）會填入 `backend.json` 並分別取得用戶端憑證、私密金鑰及信任CA憑證的基礎64編碼值。

```
{
  "version": 1,
  "backendName": "DefaultNASBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.15",
  "svm": "nfs_svm",
  "clientCertificate": "ZXR0ZXJwYXB...ICMgJ3BhcGVyc2",
  "clientPrivateKey": "vciwKIyAgZG...0cnksIGRlc2NyaX",
  "trustedCACertificate": "zcyBbaG...b3Igb3duIGNsYXNz",
  "storagePrefix": "myPrefix_"
}
```

`ontap-nas` 具有自動匯出原則的驅動程式

本範例說明如何指示Astra Trident使用動態匯出原則來自動建立及管理匯出原則。這對的運作方式相同 `ontap-nas-economy` 和 `ontap-nas-flexgroup` 驅動程式：

```
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "labels": {"k8scluster": "test-cluster-east-1a", "backend": "test1-
nasbackend"},
    "autoExportPolicy": true,
    "autoExportCIDRs": ["10.0.0.0/24"],
    "username": "admin",
    "password": "secret",
    "nfsMountOptions": "nfsvers=4",
}
```

`ontap-nas-flexgroup` 驅動程式

{"version"：1、"storageDriverName"："ontap-nas-flexgroup"、"managementLIF"："10.0.0.1"、"dataLIF"：："10.0.0.2"、"標籤"：｛"k8scluster」："test-cluster東1b"、「後端」：「test1-ontap叢集」｝、「SVM」：「SVM NFS」、「使用者名稱」：「vsadmin」、「password」：「secret」、｝

`ontap-nas` 使用**IPv6**的驅動程式

```
{
 "version": 1,
 "storageDriverName": "ontap-nas",
 "backendName": "nas_ipv6_backend",
 "managementLIF": "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]",
 "labels": {"k8scluster": "test-cluster-east-1a", "backend": "test1-ontap-
ipv6"},
 "svm": "nas_ipv6_svm",
 "username": "vsadmin",
 "password": "netapp123"
}
```

`ontap-nas-economy` 驅動程式

```
{
    "version": 1,
    "storageDriverName": "ontap-nas-economy",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "secret"
}
```

虛擬儲存資源池的後端範例

在下圖所示的範例後端定義檔案中、會針對所有儲存資源池設定特定的預設值、例如 `spaceReserve` 無、`spaceAllocation` 假、和 `encryption` 錯。虛擬儲存資源池是在儲存區段中定義。

在此範例中、有些儲存資源池會自行設定 `spaceReserve`、 `spaceAllocation`和 `encryption` 值、部分集區會覆寫上述設定的預設值。

`ontap-nas` 驅動程式

```
{
    {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "admin",
    "password": "secret",
    "nfsMountOptions": "nfsvers=4",

    "defaults": {
          "spaceReserve": "none",
          "encryption": "false",
          "qosPolicy": "standard"
    },
    "labels":{"store":"nas_store", "k8scluster": "prod-cluster-1"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"app":"msoffice", "cost":"100"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "true",
```

```
                "unixPermissions": "0755",
                "adaptiveQosPolicy": "adaptive-premium"
            }
        },
        {
            "labels":{"app":"slack", "cost":"75"},
            "zone":"us_east_1b",
            "defaults": {
                "spaceReserve": "none",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"app":"wordpress", "cost":"50"},
            "zone":"us_east_1c",
            "defaults": {
                "spaceReserve": "none",
                "encryption": "true",
                "unixPermissions": "0775"
            }
        },
        {
            "labels":{"app":"mysqldb", "cost":"25"},
            "zone":"us_east_1d",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "false",
                "unixPermissions": "0775"
            }
        }
    ]
}
```

`ontap-nas-flexgroup` 驅動程式

```
{
    "version": 1,
    "storageDriverName": "ontap-nas-flexgroup",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "secret",
```

```
    "defaults": {
          "spaceReserve": "none",
          "encryption": "false"
    },
    "labels":{"store":"flexgroup_store", "k8scluster": "prod-cluster-1"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"protection":"gold", "creditpoints":"50000"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"protection":"gold", "creditpoints":"30000"},
            "zone":"us_east_1b",
            "defaults": {
                "spaceReserve": "none",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"protection":"silver", "creditpoints":"20000"},
            "zone":"us_east_1c",
            "defaults": {
                "spaceReserve": "none",
                "encryption": "true",
                "unixPermissions": "0775"
            }
        },
        {
            "labels":{"protection":"bronze", "creditpoints":"10000"},
            "zone":"us_east_1d",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "false",
                "unixPermissions": "0775"
            }
        }
    ]
}
```

ontap-nas-economy 驅動程式

```
{
    "version": 1,
    "storageDriverName": "ontap-nas-economy",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "secret",

    "defaults": {
          "spaceReserve": "none",
          "encryption": "false"
    },
    "labels":{"store":"nas_economy_store"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"department":"finance", "creditpoints":"6000"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"department":"legal", "creditpoints":"5000"},
            "zone":"us_east_1b",
            "defaults": {
                "spaceReserve": "none",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"department":"engineering", "creditpoints":"3000"},
            "zone":"us_east_1c",
            "defaults": {
                "spaceReserve": "none",
                "encryption": "true",
                "unixPermissions": "0775"
            }
        },
        {
```

```
            "labels":{"department":"humanresource",
"creditpoints":"2000"},
            "zone":"us_east_1d",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "false",
                "unixPermissions": "0775"
            }
        }
    ]
}
```

將後端對應至**StorageClass**

下列StorageClass定義係指上述虛擬儲存資源池。使用 `parameters.selector` 欄位中、每個StorageClass會呼叫哪些虛擬資源池可用於裝載Volume。磁碟區將會在所選的虛擬資源池中定義各個層面。

- 第一個StorageClass (`protection-gold`）將對應至中的第一個、第二個虛擬儲存資源池 `ontap-nas-flexgroup` 後端與中的第一個虛擬儲存資源池 `ontap-san` 後端：這是唯一提供金級保護的資源池。
- 第二個StorageClass (`protection-not-gold`）將對應至中的第三、第四個虛擬儲存資源池 `ontap-nas-flexgroup` 中的後端和第二個、第三個虛擬儲存資源池 `ontap-san` 後端：這是唯一提供金級以外保護層級的資源池。
- 第三個StorageClass (`app-mysqldb`）將對應至中的第四個虛擬儲存資源池 `ontap-nas` 中的後端和第三個虛擬儲存資源池 `ontap-san-economy` 後端：這些是唯一提供mysqldb類型應用程式儲存池組態的集區。
- 第四個StorageClass (`protection-silver-creditpoints-20k`）將對應至中的第三個虛擬儲存資源池 `ontap-nas-flexgroup` 中的後端和第二個虛擬儲存資源池 `ontap-san` 後端：這些資源池是唯一能以20000個信用點數提供金級保護的資源池。
- 第五個StorageClass (`creditpoints-5k`）將對應至中的第二個虛擬儲存資源池 `ontap-nas-economy` 中的後端和第三個虛擬儲存資源池 `ontap-san` 後端：這些是唯一提供5000個信用點數的資源池產品。

Astra Trident將決定選取哪個虛擬儲存資源池、並確保符合儲存需求。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: netapp.io/trident
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: netapp.io/trident
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: netapp.io/trident
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

使用Astra Trident搭配Amazon FSX for NetApp ONTAP 解決方案

"Amazon FSX for NetApp ONTAP 產品"是一項完全託管的AWS服務、可讓客戶啟動及執行採用NetApp ONTAP的一套儲存作業系統的檔案系統。Amazon FSX for NetApp ONTAP 功能可讓您運用熟悉的NetApp功能、效能和管理功能、同時充分發揮儲存AWS資料的簡易性、敏捷度、安全性和擴充性。FSX支援ONTAP的許多檔案系統功能和管理API。

檔案系統是Amazon FSX的主要資源、類似ONTAP 於內部部署的一個叢集。在每個SVM中、您可以建立一個或多個磁碟區、這些磁碟區是儲存檔案系統中檔案和資料夾的資料容器。有了Amazon FSX for NetApp ONTAP 的功能、Data ONTAP 即可在雲端以託管檔案系統的形式提供支援。新的檔案系統類型稱為* NetApp ONTAP Sing*。

使用Astra Trident搭配Amazon FSX for NetApp ONTAP 時、您可以確保在Amazon Elastic Kubernetes Service（EKS）中執行的Kubernetes叢集、能夠配置區塊並歸檔以ONTAP Sure為後盾的持續磁碟區。

瞭解Astra Trident

如果您是Astra Trident的新手、請使用下列連結來熟悉您的操作：

- "常見問題集"
- "使用Astra Trident的要求"
- "部署Astra Trident"
- "最佳實務做法、可設定ONTAP 適用於Cloud Volumes ONTAP NetApp ONTAP 的功能、包括功能、功能及Amazon FSX"
- "整合Astra Trident"
- "支援SAN後端組態ONTAP"
- "ASNAS後端組態ONTAP"

深入瞭解驅動程式功能 "請按這裡"。

適用於NetApp ONTAP 的Amazon FSX "FabricPool" 管理儲存層。它可讓您根據資料是否經常存取、將資料儲存在一個層級中。

Astra Trident希望能以ONTAP 使用叢集的方式執行成一個SVM管理員或一個SVM管理員 `fsxadmin` 使用者或 `vsadmin` SVM使用者、或具有相同角色之不同名稱的使用者。。 `fsxadmin` 使用者是的有限替換品 `admin` 叢集使用者：Astra Trident通常使用 `admin` 非Amazon FSX的叢集使用者ONTAP 、可進行不實部署。

驅動程式

您ONTAP 可以使用下列驅動程式、將Astra Trident與Amazon FSX for NetApp整合：

- `ontap-san`：配置的每個PV都是自己Amazon FSX for NetApp ONTAP 的LUN。
- `ontap-san-economy`：配置的每個PV都是LUN、每個Amazon FSX for NetApp ONTAP 的LUN數量可設定。
- `ontap-nas`：配置的每個PV都是完整的Amazon FSX for NetApp ONTAP Sf2 Volume。
- `ontap-nas-economy`：每個配置的PV都是qtree、每個Amazon FSX for NetApp ONTAP 供應的qtree有可設定的配額樹數。

- `ontap-nas-flexgroup`：配置的每個PV都是完整的Amazon FSX for NetApp ONTAP FlexGroup Sf2 Volume。

驗證

Astra Trident提供兩種驗證模式：

- 認證型：您可以使用 `fsxadmin` 檔案系統或的使用者 `vsadmin` 為SVM設定的使用者。我們建議使用 `vsadmin` 使用者來設定後端。Astra Trident將使用此使用者名稱和密碼與FSX檔案系統通訊。
- 憑證型：Astra Trident會使用SVM上安裝的憑證、與FSX檔案系統上的SVM進行通訊。

若要深入瞭解驗證、請參閱下列連結：

- "NAS ONTAP"
- "SAN ONTAP"

在EKS上部署及設定Astra Trident搭配Amazon FSX for NetApp ONTAP

您需要的產品

- 現有的Amazon EKS叢集或自我管理的Kubernetes叢集 `kubectl` 已安裝。
- 現有的Amazon FSX-適用於NetApp ONTAP 的支援資料系統和儲存虛擬機器（SVM）、可從叢集的工作節點存取。
- 已準備好的工作節點 "NFS和/或iSCSI"。

> 請務必遵循Amazon Linux和Ubuntu所需的節點準備步驟 "Amazon機器映像"（AMis）、視您的EKS AMI類型而定。

如需其他Astra Trident需求、請參閱 "請按這裡"。

步驟

1. 使用以下其中一種方式部署Astra Trident：./ Trident入門/ Kubernetes-Deploy .html[部署方法]。
2. 設定Astra Trident如下：
   a. 收集SVM的管理LIF DNS名稱。例如、使用AWS CLI尋找 `DNSName` 輸入 `Endpoints → Management` 執行下列命令之後：

```
aws fsx describe-storage-virtual-machines --region <file system region>
```

3. 建立及安裝驗證憑證。如果您使用的是 `ontap-san` 後端、請參閱 "請按這裡"。如果您使用的是 `ontap-nas` 後端、請參閱 "請按這裡"。

> 您可以使用SSH從任何位置登入檔案系統（例如安裝憑證）、而該SSH可連至檔案系統。使用 `fsxadmin` 使用者、您在建立檔案系統時設定的密碼、以及管理DNS名稱 `aws fsx describe-file-systems`。

4. 使用您的憑證和管理LIF的DNS名稱建立後端檔案、如下例所示：

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "customBackendName",
  "managementLIF": "svm-XXXXXXXXXXXXXXXXX.fs-XXXXXXXXXXXXXXXXX.fsx.us-
east-2.aws.internal",
  "svm": "svm01",
  "clientCertificate": "ZXR0ZXJwYXB...ICMgJ3BhcGVyc2",
  "clientPrivateKey": "vciwKIyAgZG...0cnksIGRlc2NyaX",
  "trustedCACertificate": "zcyBbaG...b3Igb3duIGNsYXNz",
 }
```

如需建立後端的相關資訊、請參閱下列連結：

- "使用ONTAP NetApp NAS驅動程式設定後端"
- "使用ONTAP SAN驅動程式設定後端"

請勿指定 `dataLIF` 適用於 `ontap-san` 和 `ontap-san-economy` 允許Astra Trident使用多重路徑的驅動程式。

當使用Amazon FSX for NetApp ONTAP 時、搭配Astra Trident `limitAggregateUsage` 參數無法搭配使用 `vsadmin` 和 `fsxadmin` 使用者帳戶：如果您指定此參數、組態作業將會失敗。

部署之後、請執行建立的步驟 "儲存類別、配置磁碟區、然後將磁碟區掛載到Pod中"。

如需詳細資訊、請參閱

- "Amazon FSX for NetApp ONTAP 的支援文件"
- "Amazon FSX for NetApp ONTAP 的部落格文章"

使用kubecl建立後端

後端定義了Astra Trident與儲存系統之間的關係。它告訴Astra Trident如何與該儲存系統通訊、以及Astra Trident如何從該儲存系統配置磁碟區。安裝Astra Trident之後、下一步是建立後端。。 `TridentBackendConfig` 自訂資源定義（CRD）可讓您直接透過Kubernetes介面建立及管理Trident後端。您可以使用執行此作業 `kubectl` 或相當於Kubernetes發佈版本的CLI工具。

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`、 `tbconfig`、 `tbackendconfig`）是前端、命名式CRD、可讓您使用管理Astra Trident後端 `kubectl`。Kubernetes與儲存管理員現在可以直接透過Kubernetes CLI建立及管理後端、而無需使用專屬的命令列公用程式 (`tridentctl`）。

建立時 `TridentBackendConfig` 物件：

- Astra Trident會根據您提供的組態自動建立後端。這會在內部顯示為 `TridentBackend` (`tbe`、`tridentbackend`）CR.
- 。 `TridentBackendConfig` 唯一綁定到 `TridentBackend` 這是由Astra Trident所建立。

每個 `TridentBackendConfig` 使用維護一對一對應 `TridentBackend`。前者是提供給使用者設計及設定後端的介面、後者是Trident代表實際後端物件的方式。

> `TridentBackend` CRS由Astra Trident自動建立。您*不應該*修改這些項目。如果您想要對後端進行更新、請修改以執行此動作 `TridentBackendConfig` 物件：

請參閱下列範例以瞭解的格式 `TridentBackendConfig` CR：

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

您也可以查看中的範例 "Trident安裝程式" 所需儲存平台/服務的範例組態目錄。

。 `spec` 採用後端特定的組態參數。在此範例中、後端使用 `ontap-san` 儲存驅動程式、並使用此處列出的組態參數。如需所需儲存驅動程式的組態選項清單、請參閱 "儲存驅動程式的後端組態資訊"。

。 `spec` 部分也包括 `credentials` 和 `deletionPolicy` 欄位中新增的 `TridentBackendConfig` CR：

- `credentials`：此參數為必填欄位、包含用於驗證儲存系統/服務的認證資料。此設定為使用者建立的Kubernetes Secret。認證資料無法以純文字格式傳遞、因此會產生錯誤。
- `deletionPolicy`：此欄位可定義在下列情況下應發生的情況 `TridentBackendConfig` 已刪除。可能需要兩種可能的值之一：
  - `delete`：這會導致兩者都被刪除 `TridentBackendConfig` 和相關後端。這是預設值。
  - `retain`：當 `TridentBackendConfig` 刪除CR後、後端定義仍會顯示、並可透過進行管理 `tridentctl`。將刪除原則設定為 `retain` 可讓使用者降級至較早版本（21.04之前）、並保留建立的後端。此欄位的值可在之後更新 `TridentBackendConfig` 已建立。

後端名稱是使用設定 `spec.backendName`。如果未指定、則會將後端名稱設為的名稱 `TridentBackendConfig` 物件（metadata.name）。建議使用明確設定後端名稱 `spec.backendName`。

以建立的後端 `tridentctl` 沒有關聯的 `TridentBackendConfig` 物件：您可以選擇使用來管理此類後端 `kubectl` 建立 `TridentBackendConfig` CR.必須謹慎指定相同的組態參數（例如 `spec.backendName`、 `spec.storagePrefix`、 `spec.storageDriverName`等）。`Astra Trident`會自動連結新建立的 `TridentBackendConfig` 使用預先存在的後端。

步驟總覽

若要使用建立新的後端 `kubectl`、您應該執行下列步驟：

1. 建立 "Kubernetes機密"。此機密包含Astra Trident與儲存叢集/服務通訊所需的認證資料。
2. 建立 `TridentBackendConfig` 物件：其中包含有關儲存叢集/服務的詳細資訊、並參考上一步建立的機密。

建立後端之後、您可以使用觀察其狀態 `kubectl get tbc <tbc-name> -n <trident-namespace>` 並收集其他詳細資料。

步驟1：建立Kubernetes機密

建立包含後端存取認證的秘密。這是每個儲存服務/平台所獨有的功能。以下是範例：

```
$ kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: t@Ax@7q(>
```

下表摘要說明每個儲存平台的機密必須包含的欄位：

儲存平台機密欄位說明	秘密	欄位說明
Azure NetApp Files	ClientID	應用程式註冊的用戶端ID
AWS 適用的 Cloud Volumes Service	每個金鑰	CVS帳戶API金鑰
AWS 適用的 Cloud Volumes Service	秘密金鑰	CVS帳戶秘密金鑰

儲存平台機密欄位說明	秘密	欄位說明
適用於 GCP Cloud Volumes Service	Private金鑰ID	私密金鑰的ID。GCP服務帳戶API金鑰的一部分、具有CVS管理員角色
適用於 GCP Cloud Volumes Service	Private金鑰	私密金鑰：GCP服務帳戶API金鑰的一部分、具有CVS管理員角色
元素（NetApp HCI / SolidFire）	端點	MVIP、適用於SolidFire 採用租戶認證的不含用戶身分證明的叢集
ONTAP	使用者名稱	連線至叢集/ SVM的使用者名稱。用於認證型驗證
ONTAP	密碼	連線至叢集/ SVM的密碼。用於認證型驗證
ONTAP	用戶端權限金鑰	用戶端私密金鑰的Base64編碼值。用於憑證型驗證
ONTAP	chap使用 者名稱	傳入使用者名稱。如果useCHAP=true則需要。適用於 `ontap-san` 和 `ontap-san-economy`
ONTAP	chapInitiator機密	CHAP啟動器密碼。如果useCHAP=true則需要。適用於 `ontap-san` 和 `ontap-san-economy`
ONTAP	chapTargetUsername	目標使用者名稱。如果useCHAP=true則需要。適用於 `ontap-san` 和 `ontap-san-economy`
ONTAP	chapTargetInitiator機密	CHAP目標啟動器機密。如果useCHAP=true則需要。適用於 `ontap-san` 和 `ontap-san-economy`

在此步驟中建立的機密將會在中參考 `spec.credentials` 的欄位 `TridentBackendConfig` 下一步建立的物件。

步驟2：建立 `TridentBackendConfig` CR

您現在已準備好建立 `TridentBackendConfig` CR.在此範例中、使用的後端 `ontap-san` 驅動程式是使用建立的 `TridentBackendConfig` 物件如下所示：

```
$ kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

步驟3：確認的狀態 `TridentBackendConfig` CR

現在您已經建立了 `TridentBackendConfig` 您可以驗證狀態。請參閱下列範例：

```
$ kubectl -n trident get tbc backend-tbc-ontap-san
NAME                      BACKEND NAME          BACKEND UUID
PHASE   STATUS
backend-tbc-ontap-san   ontap-san-backend     8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success
```

已成功建立後端並連結至 `TridentBackendConfig` CR.

階段可以採用下列其中一個值：

- `Bound`： `TridentBackendConfig` CR與後端相關聯、且後端包含 `configRef` 設定為 `TridentBackendConfig` CR的uid。
- `Unbound`：表示使用 ""。。 `TridentBackendConfig` 物件未繫結至後端。所有新建立的 `TridentBackendConfig` CRS預設處於此階段。階段變更之後、就無法再恢復為Unbound（未綁定）。
- `Deleting`： `TridentBackendConfig` 請參閱 `deletionPolicy` 已設定為刪除。當 `TridentBackendConfig` 系統會刪除CR、並轉換為「刪除」狀態。
  - 如果後端上不存在持續磁碟區宣告（PVCS）、請刪除 `TridentBackendConfig` 將導致Astra Trident刪除後端及 `TridentBackendConfig` CR.
  - 如果後端上有一個或多個PVCS、則會進入刪除狀態。。 `TridentBackendConfig` 接著、CR也會進入刪除階段。後端和 `TridentBackendConfig` 僅在刪除所有PVCS之後才會刪除。
- `Lost`：與關聯的後端 `TridentBackendConfig` 意外或蓄意刪除及 `TridentBackendConfig` CR仍有已

刪除後端的參考資料。。 `TridentBackendConfig` 無論使用何種方法、仍可刪除CR `deletionPolicy` 價值。

- `Unknown`：Astra Trident無法判斷與相關聯的後端狀態或存在 `TridentBackendConfig` CR.例如、如果API伺服器沒有回應或是 `tridentbackends.trident.netapp.io` CRD遺失。這可能需要使用者介入。

在此階段、成功建立後端！還有多種作業可以額外處理、例如 "後端更新和後端刪除"。

（選用）步驟4：取得更多詳細資料

您可以執行下列命令來取得有關後端的詳細資訊：

```
kubectl -n trident get tbc backend-tbc-ontap-san -o wide
```

```
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success   ontap-san        delete
```

此外、您也可以取得的YAML/Json傾印 `TridentBackendConfig`。

```
$ kubectl -n trident get tbc backend-tbc-ontap-san -o yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: "2021-04-21T20:45:11Z"
  finalizers:
  - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound
```

`backendInfo` 包含 `backendName` 和 `backendUUID` 為回應所建立的後端 `TridentBackendConfig` CR.。`lastOperationStatus` 欄位代表的上次作業狀態 `TridentBackendConfig` 可由使用者觸發的CR（例如、使用者在中變更了內容 `spec`）或由Astra Trident觸發（例如、在Astra Trident重新啟動期間）。可能是「成功」或「失敗」。 `phase` 表示之間關係的狀態 `TridentBackendConfig` 和後端。在上述範例中、 `phase` 具有綁定的值、這表示 `TridentBackendConfig` CR與後端相關聯。

您可以執行 `kubectl -n trident describe tbc <tbc-cr-name>` 命令以取得事件記錄的詳細資料。

您無法更新或刪除包含相關聯的後端 `TridentBackendConfig` 物件使用 `tridentctl`。瞭解切換的步驟 `tridentctl` 和 `TridentBackendConfig`、 "請參閱此處"。

以KECBECVL執行後端管理

瞭解如何使用執行後端管理作業 `kubectl`。

刪除後端

刪除 `TridentBackendConfig`、您可以指示Astra Trident刪除/保留後端（根據 `deletionPolicy`）。若要刪除後端、請確定 `deletionPolicy` 設定為刪除。僅刪除 `TridentBackendConfig`、請務必確認 `deletionPolicy` 設定為保留。如此可確保後端仍存在、並可使用進行管理 `tridentctl`。

執行下列命令：

```
$ kubectl delete tbc <tbc-name> -n trident
```

Astra Trident並不會刪除使用中的Kubernetes Secrets `TridentBackendConfig`。Kubernetes使用者負責清除機密。刪除機密時必須小心。只有在後端未使用機密時、才應刪除這些機密。

檢視現有的後端

執行下列命令：

```
$ kubectl get tbc -n trident
```

您也可以執行 `tridentctl get backend -n trident` 或 `tridentctl get backend -o yaml -n trident` 以取得所有後端的清單。此清單也會包含使用建立的後端 `tridentctl`。

更新後端

更新後端可能有多種原因：

- 儲存系統的認證資料已變更。若要更新認證資料、請使用中的Kubernetes Secret `TridentBackendConfig` 物件必須更新。Astra Trident會自動以提供的最新認證資料更新後端。執行下列命令以更新Kubernetes Secret：

```
$ kubectl apply -f <updated-secret-file.yaml> -n trident
```

- 需要ONTAP 更新參數（例如使用的SVM名稱）。在此案例中、 `TridentBackendConfig` 物件可直接透過Kubernetes更新。

```
$ kubectl apply -f <updated-backend-file.yaml>
```

或者、也可以變更現有的 `TridentBackendConfig` 請執行下列命令以執行CR..

```
$ kubectl edit tbc <tbc-name> -n trident
```

如果後端更新失敗、後端仍會繼續維持其最後已知的組態。您可以檢視記錄、藉由執行來判斷原因 `kubectl get tbc <tbc-name> -o yaml -n trident` 或 `kubectl describe tbc <tbc-name> -n`

trident。

識別並修正組態檔的問題之後、即可重新執行update命令。

使用tridentctl執行後端管理

瞭解如何使用執行後端管理作業 tridentctl。

建立後端

建立之後 "後端組態檔"，執行下列命令：

```
$ tridentctl create backend -f <backend-file> -n trident
```

如果後端建立失敗、表示後端組態有問題。您可以執行下列命令來檢視記錄、以判斷原因：

```
$ tridentctl logs -n trident
```

識別並修正組態檔案的問題之後、您只需執行即可 create 命令。

刪除後端

若要從Astra Trident刪除後端、請執行下列步驟：

1. 擷取後端名稱：

   ```
   $ tridentctl get backend -n trident
   ```

2. 刪除後端：

   ```
   $ tridentctl delete backend <backend-name> -n trident
   ```

> 如果Astra Trident已從這個後端配置磁碟區和快照、但該後端仍存在、則刪除後端會使新的磁碟區無法由其進行資源配置。後端將繼續處於「刪除」狀態、而Trident將繼續管理這些磁碟區和快照、直到它們被刪除為止。

檢視現有的後端

若要檢視Trident知道的後端、請執行下列步驟：

- 若要取得摘要、請執行下列命令：

```
$ tridentctl get backend -n trident
```

- 若要取得所有詳細資料、請執行下列命令：

```
$ tridentctl get backend -o json -n trident
```

更新後端

建立新的後端組態檔之後、請執行下列命令：

```
$ tridentctl update backend <backend-name> -f <backend-file> -n trident
```

如果後端更新失敗、表示後端組態有問題、或是您嘗試了無效的更新。您可以執行下列命令來檢視記錄、以判斷原因：

```
$ tridentctl logs -n trident
```

識別並修正組態檔案的問題之後、您只需執行即可 `update` 命令。

識別使用後端的儲存類別

這是您可以用Json回答的問題類型範例 `tridentctl` 後端物件的輸出。這會使用 `jq` 公用程式、您需要安裝。

```
$ tridentctl get backend -o json | jq '[.items[] | {backend: .name,
storageClasses: [.storage[].storageClasses]|unique}]'
```

這也適用於使用建立的後端 `TridentBackendConfig`。

在後端管理選項之間切換

瞭解Astra Trident管理後端的不同方法。隨之推出 `TridentBackendConfig`管理員現在有兩種獨特的後端管理方法。這會提出下列問題：

- 可以使用建立後端 `tridentctl` 以進行管理 `TridentBackendConfig`？
- 可以使用建立後端 `TridentBackendConfig` 使用進行管理 `tridentctl`？

管理 `tridentctl` 後端使用 `TridentBackendConfig`

本節說明管理使用建立之後端所需的步驟 `tridentctl` 透過建立、直接透過Kubernetes介面 `TridentBackendConfig` 物件：

這將適用於下列案例：

- 預先存在的後端、但沒有 `TridentBackendConfig` 因為它們是使用建立的 `tridentctl`。
- 使用建立的新後端 `tridentctl`、而其他 `TridentBackendConfig` 物件存在。

在這兩種情況下、後端仍會繼續存在、Astra Trident排程磁碟區會繼續運作。系統管理員有兩種選擇之一：

- 繼續使用 `tridentctl` 管理使用它建立的後端。
- 使用建立連結後端 `tridentctl` 新功能 `TridentBackendConfig` 物件：如此一來、後端就會使用進行管理 `kubectl` 而非 `tridentctl`。

若要使用管理預先存在的後端 `kubectl`、您需要建立 `TridentBackendConfig` 連結至現有後端。以下是如何運作的總覽：

1. 建立Kubernetes機密。此機密包含Astra Trident與儲存叢集/服務通訊所需的認證資料。
2. 建立 `TridentBackendConfig` 物件：其中包含有關儲存叢集/服務的詳細資訊、並參考上一步建立的機密。必須謹慎指定相同的組態參數（例如 `spec.backendName`、 `spec.storagePrefix`、 `spec.storageDriverName`等）。 `spec.backendName` 必須設定為現有後端的名稱。

步驟**0**：識別後端

以建立 `TridentBackendConfig` 如果綁定到現有的後端、您將需要取得後端的組態。在此範例中、假設使用下列Json定義建立後端：

```
$ tridentctl get backend ontap-nas-backend -n trident
+---------------------+----------------
+--------------------------------------+---------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE   | VOLUMES |
+---------------------+----------------
+--------------------------------------+---------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+---------+---------+

$ cat ontap-nas-backend.json

{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.10.10.1",
    "dataLIF": "10.10.10.2",
    "backendName": "ontap-nas-backend",
    "svm": "trident_svm",
    "username": "cluster-admin",
    "password": "admin-password",
```

```
    "defaults": {
        "spaceReserve": "none",
        "encryption": "false"
    },
    "labels":{"store":"nas_store"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"app":"msoffice", "cost":"100"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"app":"mysqldb", "cost":"25"},
            "zone":"us_east_1d",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "false",
                "unixPermissions": "0775"
            }
        }
    ]
}
```

步驟1：建立Kubernetes機密

建立包含後端認證的秘密、如以下範例所示：

```
$ cat tbc-ontap-nas-backend-secret.yaml

apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  passWord: admin-password

$ kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created
```

步驟2：建立 `TridentBackendConfig` CR

下一步是建立 `TridentBackendConfig` 會自動連結至預先存在的CR `ontap-nas-backend` （如本範例所示）。確保符合下列要求：

- 中定義了相同的後端名稱 `spec.backendName`。
- 組態參數與原始後端相同。
- 虛擬儲存資源池（若有）必須維持與原始後端相同的順序。
- 認證資料是透過Kubernetes Secret提供、而非以純文字提供。

在此案例中 `TridentBackendConfig` 如下所示：

```
$ cat backend-tbc-ontap-nas.yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: 'true'
      unixPermissions: '0755'
  - labels:
      app: mysqldb
      cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'

$ kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created
```

步驟3：確認的狀態 `TridentBackendConfig` CR

之後 `TridentBackendConfig` 已經建立、其階段必須是 `Bound`。它也應反映與現有後端相同的後端名稱和UUID。

```
$ kubectl -n trident get tbc tbc-ontap-nas-backend -n trident
NAME                   BACKEND NAME          BACKEND UUID
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

#confirm that no new backends were created (i.e., TridentBackendConfig did
not end up creating a new backend)
$ tridentctl get backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

後端現在將使用完全管理 `tbc-ontap-nas-backend` `TridentBackendConfig` 物件：

管理 `TridentBackendConfig` 後端使用 `tridentctl`

`tridentctl` 可用來列出使用建立的後端 `TridentBackendConfig`。此外、系統管理員也可以選擇透過完全管理此類後端 `tridentctl` 刪除 `TridentBackendConfig` 並確保 `spec.deletionPolicy` 設為 `retain`。

步驟0：識別後端

例如、假設下列後端是使用建立的 `TridentBackendConfig`：

```
$ kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                     BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

$ tridentctl get backend ontap-san-backend -n trident
+-------------------+----------------
+--------------------------------------+--------+---------+
|       NAME        | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+-------------------+----------------
+--------------------------------------+--------+---------+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

從輸出中可以看出這一點 `TridentBackendConfig` 已成功建立、並繫結至後端（觀察後端的UUID）。

步驟1：確認 `deletionPolicy` 設為 `retain`

讓我們來看看的價值 `deletionPolicy`。這需要設定為 `retain`。這可確保在發生時 `TridentBackendConfig` 刪除CR後、後端定義仍會顯示、並可透過進行管理 `tridentctl`。

```
$ kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                     BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

# Patch value of deletionPolicy to retain
$ kubectl patch tbc backend-tbc-ontap-san --type=merge -p
'{"spec":{"deletionPolicy":"retain"}}' -n trident
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-san patched

#Confirm the value of deletionPolicy
$ kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                     BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        retain
```

除非如此、否則請勿繼續下一步 `deletionPolicy` 設為 `retain`。

步驟2：刪除 `TridentBackendConfig` CR

最後一個步驟是刪除 `TridentBackendConfig` CR.確認之後 `deletionPolicy` 設為 `retain`、您可以繼續刪除：

```
$ kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

$ tridentctl get backend ontap-san-backend -n trident
+-------------------+----------------
+--------------------------------------+--------+---------+
|       NAME        | STORAGE DRIVER |                 UUID
| STATE   | VOLUMES |
+-------------------+----------------
+--------------------------------------+--------+---------+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

刪除時 `TridentBackendConfig` 物件：Astra Trident只會移除它、而不會實際刪除後端本身。

管理儲存類別

尋找建立儲存類別、刪除儲存類別及檢視現有儲存類別的相關資訊。

設計儲存類別

請參閱 "請按這裡" 以取得有關儲存類別及其設定方式的詳細資訊。

建立儲存類別

取得儲存類別檔案之後、請執行下列命令：

```
kubectl create -f <storage-class-file>
```

`<storage-class-file>` 應以您的儲存類別檔案名稱取代。

刪除儲存類別

若要從Kubernetes刪除儲存類別、請執行下列命令：

```
kubectl delete storageclass <storage-class>
```

<storage-class> 應更換為您的儲存類別。

透過此儲存類別所建立的任何持續磁碟區都將維持不變、Astra Trident將繼續管理這些磁碟區。

> Astra Trident強制執行空白 fsType 針對所建立的磁碟區。對於iSCSI後端、建議強制執行 parameters.fsType 在StorageClass中。您應該刪除「儲存類別」、然後重新建立這些「儲存類別」 parameters.fsType 已指定。

檢視現有的儲存類別

- 若要檢視現有的Kubernetes儲存類別、請執行下列命令：

```
kubectl get storageclass
```

- 若要檢視Kubernetes儲存類別詳細資料、請執行下列命令：

```
kubectl get storageclass <storage-class> -o json
```

- 若要檢視Astra Trident的同步儲存類別、請執行下列命令：

```
tridentctl get storageclass
```

- 若要檢視Astra Trident的同步儲存類別詳細資料、請執行下列命令：

```
tridentctl get storageclass <storage-class> -o json
```

設定預設儲存類別

Kubernetes 1.6新增了設定預設儲存類別的功能。如果使用者未在「持續磁碟區宣告」（PVc）中指定一個、則此儲存類別將用於配置「持續磁碟區」。

- 設定註釋以定義預設儲存類別 storageclass.kubernetes.io/is-default-class 儲存類別定義中的「真」。根據規格、任何其他值或不存在附註都會解譯為假。
- 您可以使用下列命令、將現有的儲存類別設定為預設的儲存類別：

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

- 同樣地、您也可以使用下列命令移除預設儲存類別註釋：

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"}}}'
```

Trident安裝程式套件中也有包含此附註的範例。

在任何指定時間、您的叢集中都只應有一個預設儲存類別。Kubernetes在技術上並不妨礙您擁有多個儲存類別、但它的行為方式就如同完全沒有預設的儲存類別一樣。

識別儲存類別的後端

這是您可以用Json回答的問題類型範例 `tridentctl` Astra Trident後端物件的輸出。這會使用 `jq` 公用程式、您可能需要先安裝。

```
tridentctl get storageclass -o json | jq  '[.items[] | {storageClass:
.Config.name, backends: [.storage]|unique}]'
```

執行Volume作業

深入瞭解Astra Trident提供的功能、協助您管理磁碟區。

- "使用「csi拓撲」"
- "使用快照"
- "展開Volume"
- "匯入磁碟區"

使用「csi拓撲」

Astra Trident可以利用、選擇性地建立磁碟區、並將磁碟區附加至Kubernetes叢集中的節點 "「csi拓撲」功能"。使用「csi拓撲」功能、可根據區域和可用性區域、限制對磁碟區的存取、只能存取一部分節點。如今、雲端供應商可讓Kubernetes管理員建立以區域為基礎的節點。節點可以位於可用度區域內的不同區域、或是不同的可用度區域。為了協助在多區域架構中配置工作負載的磁碟區、Astra Trident使用了csi拓撲。

深入瞭解「csi拓撲」功能 "請按這裡"。

Kubernetes提供兩種獨特的Volume繫結模式：

- 與 `VolumeBindingMode` 設定為 `Immediate`Astra Trident在沒有任何拓撲感知的情況下建立磁碟區。建立永久虛擬磁碟時、即會處理磁碟區繫結和動態資源配置。這是預設值 `VolumeBindingMode` 適用於未強制拓撲限制的叢集。建立永續性磁碟區時、不會對要求的Pod排程需求有任何相依性。
- 與 `VolumeBindingMode` 設定為 `WaitForFirstConsumer`、永久磁碟區的建立與繫結會延遲、直到排程並建立使用該永久磁碟的Pod為止。如此一來、就能建立磁碟區、以符合拓撲需求所強制執行的排程限制。

。 `WaitForFirstConsumer` 繫結模式不需要拓撲標籤。這可獨立於「csi拓撲」功能使用。

您需要的產品

若要使用「csi拓撲」、您需要下列項目：

- 執行1.17或更新版本的Kubernetes叢集。

```
$ kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- 叢集中的節點應該有標籤來介紹拓撲認知 (`topology.kubernetes.io/region` 和 `topology.kubernetes.io/zone`）。在安裝Astra Trident以識別拓撲之前、這些標籤*應該會出現在叢集*的節點上。

```
$ kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

步驟**1**：建立可感知拓撲的後端

Astra Trident儲存後端可根據可用性區域、選擇性地配置磁碟區。每個後端都可隨附選用功能 `supportedTopologies` 代表必須支援之區域和區域清單的區塊。對於使用此類後端的StorageClass、只有在受支援地區/區域中排程的應用程式要求時、才會建立Volume。

以下是後端定義的範例：

```
{
 "version": 1,
 "storageDriverName": "ontap-san",
 "backendName": "san-backend-us-east1",
 "managementLIF": "192.168.27.5",
 "svm": "iscsi_svm",
 "username": "admin",
 "password": "xxxxxxxxxxxx",
 "supportedTopologies": [
{"topology.kubernetes.io/region": "us-east1",
"topology.kubernetes.io/zone": "us-east1-a"},
{"topology.kubernetes.io/region": "us-east1",
"topology.kubernetes.io/zone": "us-east1-b"}
]
}
```

`supportedTopologies` 用於提供每個後端的區域和區域清單。這些區域和區域代表StorageClass中可提供的允許值清單。對於包含後端所提供之區域和區域子集的StorageClass、Astra Trident會在後端建立磁碟區。

您可以定義 `supportedTopologies` 也可依儲存資源池。請參閱下列範例：

```
{"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "nas-backend-us-central1",
"managementLIF": "172.16.238.5",
"svm": "nfs_svm",
"username": "admin",
"password": "Netapp123",
"supportedTopologies": [
      {"topology.kubernetes.io/region": "us-central1",
"topology.kubernetes.io/zone": "us-central1-a"},
      {"topology.kubernetes.io/region": "us-central1",
"topology.kubernetes.io/zone": "us-central1-b"}
    ]
"storage": [
   {
       "labels": {"workload":"production"},
        "region": "Iowa-DC",
        "zone": "Iowa-DC-A",
        "supportedTopologies": [
            {"topology.kubernetes.io/region": "us-central1",
"topology.kubernetes.io/zone": "us-central1-a"}
        ]
    },
    {
        "labels": {"workload":"dev"},
         "region": "Iowa-DC",
         "zone": "Iowa-DC-B",
         "supportedTopologies": [
             {"topology.kubernetes.io/region": "us-central1",
"topology.kubernetes.io/zone": "us-central1-b"}
         ]
     }
]
}
```

在此範例中 `region` 和 `zone` 標籤代表儲存資源池的位置。 `topology.kubernetes.io/region` 和 `topology.kubernetes.io/zone` 指定儲存資源池的使用來源。

步驟**2**：定義可感知拓撲的**StorageClass**

根據提供給叢集中節點的拓撲標籤、可以定義StorageClass以包含拓撲資訊。這將決定做為所提出之永久虛擬磁碟要求候選的儲存資源池、以及可以使用Trident所提供之磁碟區的節點子集。

請參閱下列範例：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
- matchLabelExpressions:
- key: topology.kubernetes.io/zone
  values:
  - us-east1-a
  - us-east1-b
- key: topology.kubernetes.io/region
  values:
  - us-east1
parameters:
  fsType: "ext4"
```

在上述StorageClass定義中、 `volumeBindingMode` 設為 `WaitForFirstConsumer`。在Pod中引用此StorageClass所要求的PVCS之前、系統不會對其採取行動。而且、 `allowedTopologies` 提供要使用的區域和區域。。 `netapp-san-us-east1` StorageClass會在上建立PVCS `san-backend-us-east1` 上述定義的後端。

步驟**3**：建立並使用**PVC**

建立StorageClass並對應至後端後端後端之後、您現在就可以建立PVCS。

請參閱範例 `spec` 以下：

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1
```

使用此資訊清單建立永久虛擬環境可能會產生下列結果：

```
$ kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
$ kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
$ kubectl describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type    Reason                Age   From                         Message
  ----    ------                ----  ----                         -------
  Normal  WaitForFirstConsumer  6s    persistentvolume-controller  waiting
for first consumer to be created before binding
```

若要Trident建立磁碟區並將其連結至PVc、請在Pod中使用PVc。請參閱下列範例：

```
apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.kubernetes.io/region
            operator: In
            values:
            - us-east1
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - us-east1-a
            - us-east1-b
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  volumes:
  - name: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

此podSpec會指示Kubernetes在中的節點上排程pod `us-east1` 區域、並從中的任何節點中進行選擇 `us-east1-a` 或 `us-east1-b` 區域。

請參閱下列輸出：

```
$ kubectl get pods -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP               NODE
NOMINATED NODE   READINESS GATES
app-pod-1   1/1     Running   0          19s   192.168.25.131   node2
<none>           <none>
$ kubectl get pvc -o wide
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS          AGE   VOLUMEMODE
pvc-san   Bound    pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b   300Mi
RWO            netapp-san-us-east1   48s   Filesystem
```

更新後端以納入 `supportedTopologies`

您可以更新現有的後端、以納入清單 `supportedTopologies` 使用 `tridentctl backend update`。這不會影響已配置的磁碟區、而且只會用於後續的PVCS。

如需詳細資訊、請參閱

- "管理容器的資源"
- "節點選取器"
- "關聯性與反關聯性"
- "污染與容許"

使用快照

從2001版Astra Trident開始、您可以在Kubernetes層建立PV快照。您可以使用這些快照來維護由Astra Trident所建立之磁碟區的時間點複本、並排程建立其他磁碟區（複本）。支援Volume Snapshot `ontap-nas`、`ontap-san`、`ontap-san-economy`、`solidfire-san`、`aws-cvs`、`gcp-cvs`和 `azure-netapp-files` 驅動程式：

> 此功能可從Kubernetes 1.17（試用版）取得、GA版本為1.20。若要瞭解從試用版移轉至GA所涉及的變更、請參閱 "版本部落格"。隨著GA畢業 `v1` API版本已推出、並向下相容 `v1beta1` 快照：

您需要的產品

- 建立Volume快照需要建立外部快照控制器、以及一些自訂資源定義（CRD）。這是正在使用的Kubernetes Orchestrator的責任（例如：Kubeadm,GKe, OpenShift）。

您可以建立外部快照控制器和快照客戶需求日、如下所示：

1. 建立Volume Snapshot客戶需求日：

```
$ cat snapshot-setup.sh
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
3.0/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
3.0/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yam
l
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
3.0/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. 在所需的命名空間中建立Snapshot控制器。編輯下方的Yaml清單以修改命名空間。

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-3.0/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-3.0/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

「csi Snapshotter」提供 "正在驗證Webhook" 協助使用者驗證現有的v1Beta1快照、並確認它們是有效的資源物件。驗證Webhook會自動標示無效的快照物件、並防止未來建立無效物件。驗證Webhook是由Kubernetes Orchestrator部署。請參閱指示以手動部署驗證Webhook "請按這裡"。尋找無效快照資訊清單的範例 "請按這裡"。

以下範例說明使用快照所需的架構、並說明如何建立及使用快照。

步驟**1**：設定 `VolumeSnapshotClass`

在建立Volume Snapshot之前、請先設定連結：../ Trident參考/objects.html[`VolumeSnapshotClass`^]。

```
$ cat snap-sc.yaml
#Use apiVersion v1 for Kubernetes 1.20 and above. For Kubernetes 1.17 -
1.19, use apiVersion v1beta1.
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

。 `driver` 指向Astra Trident的SCSI驅動程式。 `deletionPolicy` 可以 `Delete` 或 `Retain`。設定為時 `Retain`、儲存叢集上的基礎實體快照、即使在 `VolumeSnapshot` 物件已刪除。

步驟**2**：建立現有**PVc**的快照

```
$ cat snap.yaml
#Use apiVersion v1 for Kubernetes 1.20 and above. For Kubernetes 1.17 -
1.19, use apiVersion v1beta1.
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: pvc1-snap
spec:
  volumeSnapshotClassName: csi-snapclass
  source:
    persistentVolumeClaimName: pvc1
```

正在為名為的PVc建立快照 `pvc1`，並將快照的名稱設為 `pvc1-snap`。

```
$ kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

$ kubectl get volumesnapshots
NAME                   AGE
pvc1-snap              50s
```

這會建立一個 `VolumeSnapshot` 物件：Volume Snapshot類似於PVC、並與相關聯 `VolumeSnapshotContent` 代表實際快照的物件。

您可以識別 `VolumeSnapshotContent` 的物件 `pvc1-snap` 描述Volume Snapshot。

```
$ kubectl describe volumesnapshots pvc1-snap
Name:         pvc1-snap
Namespace:    default
.
.
.
Spec:
  Snapshot Class Name:    pvc1-snap
  Snapshot Content Name:  snapcontent-e8d8a0ca-9826-11e9-9807-525400f3f660
  Source:
    API Group:
    Kind:       PersistentVolumeClaim
    Name:       pvc1
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:   true
  Restore Size:   3Gi
.
.
```

。 `Snapshot Content Name` 識別提供此快照的Volume SnapshotContent物件。。 `Ready To Use` 參數表示Snapshot可用於建立新的PVc。

步驟3：從Volume Snapshot建立PVCS

請參閱下列範例、瞭解如何使用快照建立永久虛擬資料：

```
$ cat pvc-from-snap.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

`dataSource` 顯示必須使用名為的Volume Snapshot建立PVc `pvc1-snap` 做為資料來源。這會指示Astra Trident從快照建立一個永久虛擬資料。建立好永久虛擬基礎架構之後、就能將它附加到Pod上、就像使用任何其他永久虛擬基礎架構一樣使用。

> 刪除具有相關快照的持續Volume時、對應的Trident Volume會更新為「刪除狀態」。若要刪除Astra Trident磁碟區、則應移除該磁碟區的快照。

如需詳細資訊、請參閱

- "Volume快照"
- 連結：../ Trident參考/ objects.html[`VolumeSnapshotClass`^]

展開Volume

Astra Trident可讓Kubernetes使用者在建立磁碟區之後擴充磁碟區。尋找擴充iSCSI和NFS磁碟區所需組態的相關資訊。

展開iSCSI Volume

您可以使用「SCSI資源配置程式」來擴充iSCSI持續磁碟區（PV）。

> 支援iSCSI Volume擴充 `ontap-san`、 `ontap-san-economy`、 `solidfire-san` 並需要Kubernetes 1.16及更新版本。

總覽

擴充iSCSI PV包括下列步驟：

- 編輯StorageClass定義以設定 `allowVolumeExpansion` 欄位至 `true`。
- 編輯PVc定義並更新 `spec.resources.requests.storage` 以反映新的所需大小、此大小必須大於原始大小。
- 必須將PV附加至Pod、才能調整其大小。調整iSCSI PV的大小有兩種情況：
  - 如果PV附加至Pod、Astra Trident會在儲存後端擴充磁碟區、重新掃描裝置、並重新調整檔案系統的大小。
  - 嘗試調整未附加PV的大小時、Astra Trident會在儲存後端上擴充磁碟區。在將永久虛擬磁碟綁定至Pod之後、Trident會重新掃描裝置並重新調整檔案系統的大小。然後、Kubernetes會在擴充作業成功完成後、更新PVc大小。

以下範例顯示擴充iSCSI PV的運作方式。

步驟1：設定StorageClass以支援Volume擴充

```
$ cat storageclass-ontapsan.yaml
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

對於已存在的StorageClass、請編輯此類以納入 `allowVolumeExpansion` 參數。

步驟2：使用您建立的**StorageClass**建立一個永久虛擬儲存設備

```
$ cat pvc-ontapsan.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: san-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-san
```

Astra Trident會建立持續磁碟區（PV）、並將其與此持續磁碟區宣告（PVc）建立關聯。

```
$ kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

$ kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               10s
```

步驟3：定義一個連接至PVc的Pod

在此範例中、會建立使用的Pod `san-pvc`。

```
$  kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
centos-pod   1/1     Running   0          65s

$  kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    centos-pod
```

步驟4：展開PV

若要調整從1Gi建立至2Gi的PV大小、請編輯PVc定義並更新 `spec.resources.requests.storage` 至2Gi。

```
$ kubectl edit pvc san-pvc
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 ...
```

步驟5：驗證擴充

您可以檢查PVc、PV和Astra Trident Volume的大小、以正確驗證擴充作業：

```
$ kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
$ kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               12m
$ tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

展開NFS Volume

Astra Trident支援在上配置NFS PV的Volume擴充 `ontap-nas`、 `ontap-nas-economy`、 `ontap-nas-flexgroup`、 `aws-cvs`、 `gcp-cvs`和 `azure-netapp-files` 後端：

步驟1：設定StorageClass以支援Volume擴充

若要調整NFS PV的大小、管理員必須先設定儲存類別、以允許透過設定來擴充磁碟區 `allowVolumeExpansion` 欄位至 `true`：

```
$ cat storageclass-ontapnas.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true
```

如果您已建立不含此選項的儲存類別、則只要使用編輯現有的儲存類別即可 `kubectl edit storageclass` 以允許磁碟區擴充。

步驟2：使用您建立的**StorageClass**建立一個永久虛擬儲存設備

```
$ cat pvc-ontapnas.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

Astra Trident應為此PVC建立20MiB NFS PV：

```
$ kubectl get pvc
NAME           STATUS   VOLUME
CAPACITY     ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi
RWO            ontapnas        9s

$ kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi       RWO
Delete           Bound    default/ontapnas20mb   ontapnas
2m42s
```

步驟3：展開**PV**

若要將新建立的20MiB PV調整至1GiB、請編輯該PVC並設定組合 `spec.resources.requests.storage` 至1GB：

```
$ kubectl edit pvc ontapnas20mb
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
...
```

步驟4：驗證擴充

您可以檢查PVc、PV和Astra Trident Volume的大小、以正確驗證調整大小：

```
$ kubectl get pvc ontapnas20mb
NAME           STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi
RWO            ontapnas        4m44s

$ kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                   STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi        RWO
Delete           Bound    default/ontapnas20mb   ontapnas
5m35s

$ tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n
trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 | 1.0 GiB | ontapnas      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

匯入磁碟區

您可以使用將現有的儲存磁碟區匯入為Kubernetes PV `tridentctl import`。

支援Volume匯入的驅動程式

下表說明支援匯入磁碟區的驅動程式、以及這些磁碟區所引進的版本。

驅動程式	版本
`ontap-nas`	19.04
`ontap-nas-flexgroup`	19.04
`solidfire-san`	19.04
`aws-cvs`	19.04

驅動程式	版本
azure-netapp-files	19.04
gcp-cvs	19.04
ontap-san	19.04

為什麼要匯入磁碟區？

將Volume匯入Trident的使用案例有多種：

- 容器化應用程式、並重新使用現有的資料集
- 將資料集的複本用於暫時性應用程式
- 重建故障的Kubernetes叢集
- 在災難恢復期間移轉應用程式資料

匯入如何運作？

Volume匯入程序會使用持續磁碟區宣告（PVc）檔案來建立PVc。至少、PVc檔案應包含名稱、命名空間、存取模式及storageClassName欄位、如下例所示。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my_claim
  namespace: my_namespace
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: my_storage_class
```

。 `tridentctl` 用戶端用於匯入現有的儲存磁碟區。Trident會持續儲存Volume中繼資料並建立PVc和PV、以匯入Volume。

```
$ tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-
file>
```

若要匯入儲存磁碟區、請指定包含該磁碟區的Astra Trident後端名稱、以及唯一識別儲存區上該磁碟區的名稱（例如ONTAP FlexVol ：Wsel, Element Volume、CVS Volume path）。儲存磁碟區必須允許讀取/寫入存取、且可由指定的Astra Trident後端存取。。 `-f` 字串引數為必填、並指定Yaml或Json PVc檔案的路徑。

當Astra Trident收到匯入磁碟區要求時、現有的磁碟區大小會在PVc中決定及設定。儲存驅動程式匯入磁碟區之後、PV會以PVc的ClaimRef建立。回收原則一開始設定為 `retain` 在PV中。Kubernetes成功繫結了PVc和PV之後、系統會更新回收原則以符合儲存類別的回收原則。如果儲存類別的回收原則為 `delete`、儲存磁碟區會

在PV刪除時刪除。

使用匯入Volume時 `--no-manage` 引數：Trident不會在物件生命週期的PVc或PV上執行任何其他作業。因為Trident會忽略的PV和PVc事件 `--no-manage` 物件、儲存磁碟區不會在PV刪除時刪除。此外、也會忽略其他作業、例如Volume Clone和Volume resize。如果您想要將Kubernetes用於容器化工作負載、但想要管理Kubernetes以外儲存磁碟區的生命週期、則此選項非常實用。

將註釋新增至PVc和PV、這有兩種用途、表示已匯入磁碟區、以及是否管理了PVc和PV。不應修改或移除此附註。

Trident 19.07及更新版本可處理PV的附加元件、並在匯入磁碟區時掛載磁碟區。對於使用舊版Astra Trident的匯入、資料路徑不會有任何作業、而且磁碟區匯入不會驗證是否可以掛載磁碟區。如果在匯入磁碟區時發生錯誤（例如、StorageClass不正確）、您可以將PV上的回收原則變更為來恢復 `retain`、刪除PVC和PV、然後重新嘗試Volume匯入命令。

`ontap-nas` 和 `ontap-nas-flexgroup` 匯入

使用建立的每個Volume `ontap-nas` 驅動程式FlexVol 是ONTAP 指在整個叢集上執行的功能。使用匯入FlexVols `ontap-nas` 驅動程式的運作方式相同。可將已存在於某個叢集上的一個功能、匯入為FlexVol ONTAP `ontap-nas` PVC.同樣地FlexGroup 、也可以將此資訊匯入為 `ontap-nas-flexgroup` PVCs：

若要由Trident匯入某個類型的Rw。ONTAP如果磁碟區是DP類型、則它是SnapMirror目的地磁碟區；在將磁碟區匯入Trident之前、您應該先中斷鏡射關係。

。 `ontap-nas` 驅動程式無法匯入及管理qtree。。 `ontap-nas` 和 `ontap-nas-flexgroup` 驅動程式不允許重複的磁碟區名稱。

例如、匯入名為的磁碟區 `managed_volume` 在名為的後端上 `ontap_nas`，請使用下列命令：

```
$ tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>

+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |              BACKEND UUID              | STATE  | MANAGED |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
| pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
```

匯入名為的磁碟區 `unmanaged_volume` （在上 `ontap_nas backend`）（Trident無法管理）、請使用下列命令：

```
$ tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-file>
--no-manage

+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+---------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE   | MANAGED |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+---------+---------+
| pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | false   |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+---------+---------+
```

使用時 `--no-manage` 引數：Trident不會重新命名磁碟區、也不會驗證磁碟區是否已掛載。如果未手動掛載磁碟區、則磁碟區匯入作業會失敗。

已修正先前使用自訂Unix權限 匯入磁碟區的錯誤。您可以在您的PVc定義或後端組態中指定unixPermissions、並指示Astra Trident依此匯入磁碟區。

`ontap-san` 匯入

Astra Trident也能匯入ONTAP 包含單一LUN的SAN FlexVols。這與一致 `ontap-san` 驅動程式、為FlexVol 每個實體磁碟和FlexVol 一個LUN建立一個實體。您可以使用 `tridentctl import` 命令的方式與其他情況相同：

- 包括的名稱 `ontap-san` 後端：
- 請提供FlexVol 需要匯入的名稱。請記住FlexVol 、這個功能只包含一個必須匯入的LUN。
- 提供必須搭配使用的PVc定義路徑 `-f` 旗標。
- 您可以選擇管理或不受管理的永久虛擬網路。根據預設、Trident會管理PVc、並在FlexVol 後端重新命名該LUN。若要匯入為未受管理的Volume、請傳遞 `--no-manage` 旗標。

匯入未受管理的時 `ontap-san` Volume中的LUN FlexVol 名稱 `lun0` 並對應至具有所需啟動器的igroup。Astra Trident會自動處理這項作業、以便進行託管匯入。

然後Astra Trident會匯入FlexVol 該等物件、並將其與PVc定義建立關聯。Astra Trident也將FlexVol 該等功能重新命名為 `pvc-<uuid>` 格式化及FlexVol LUN在功能區內 `lun0`。

建議匯入沒有現有作用中連線的磁碟區。如果您要匯入使用中的Volume、請先複製該Volume、然後再執行匯入。

範例

以匯入 `ontap-san-managed` 上的顯示FlexVol `ontap_san_default` 後端、執行 `tridentctl import` 命令形式：

```
$ tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d

+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic         |
block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true    |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
```

若要由Astra Trident匯入某個類型的RW磁碟區。ONTAP如果磁碟區為DP類型、則為SnapMirror目的地磁碟區；您應該先中斷鏡射關係、再將磁碟區匯入Astra Trident。

`element` 匯入

您可以使用NetApp Element Trident將支援功能的軟體/NetApp HCI磁碟區匯入Kubernetes叢集。您需要Astra Trident後端的名稱、以及磁碟區的唯一名稱和PVc檔案做為的引數 `tridentctl import` 命令。

```
$ tridentctl import volume element_default element-managed -f pvc-basic-
import.yaml -n trident -d

+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe | 10 GiB | basic-element |
block    | d3ba047a-ea0b-43f9-9c42-e38e58301c49 | online | true    |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
```

Element驅動程式支援重複的Volume名稱。如果有重複的Volume名稱、Trident的Volume匯入程序會傳回錯誤。因應措施是複製磁碟區、並提供唯一的磁碟區名稱。然後匯入複製的Volume。

`aws-cvs` 匯入

若要匯入以AWS中的NetApp Cloud Volumes Service 支援為後盾的磁碟區、請使用磁碟區路徑來識別該磁碟區、而非其名稱。

若要匯入 `aws-cvs` 後端上的Volume稱為 `awscvs_YEppr` 的磁碟區路徑 `adroit-jolly-swift`，請使用下列命令：

```
$ tridentctl import volume awscvs_YEppr adroit-jolly-swift -f <path-to-
pvc-file> -n trident

+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-a46ccab7-44aa-4433-94b1-e47fc8c0fa55 | 93 GiB | aws-storage   | file
| e1a6e65b-299e-4568-ad05-4f0a105c888f | online | true    |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
```

Volume路徑是Volume匯出路徑的一部分、位於：/之後。例如、如果匯出路徑為 `10.0.0.1:/adroit-jolly-swift`、磁碟區路徑為 `adroit-jolly-swift`。

`gcp-cvs` 匯入

匯入 `gcp-cvs` Volume的運作方式與匯入相同 `aws-cvs` Volume：

`azure-netapp-files` 匯入

若要匯入 `azure-netapp-files` 後端上的Volume稱為 `azurenetappfiles_40517` 磁碟區路徑 `importvol1`，執行下列命令：

```
$ tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage   |
file     | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true    |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

anf磁碟區的磁碟區路徑會出現在裝載路徑中的：/之後。例如、如果掛載路徑為 `10.0.0.2:/importvol1`、磁碟區路徑為 `importvol1`。

準備工作節點

Kubernetes叢集中的所有工作節點都必須能夠掛載您已為Pod配置的磁碟區。如果您使用的是 `ontap-nas`、`ontap-nas-economy`或 `ontap-nas-flexgroup` 您的其中一個後端的驅動程式、您的工作節點需要NFS工具。否則他們需要iSCSI工具。

最新版本的RedHat CoreOS預設會同時安裝NFS和iSCSI。

安裝NFS或iSCSI工具之後、您應該一律重新啟動工作節點、否則將磁碟區附加至容器可能會失敗。

NFS磁碟區

傳輸協定	作業系統	命令
NFS	RHEL/CentOS	`sudo yum install -y nfs-utils`
NFS	Ubuntu / DEBIAN	`sudo apt-get install -y nfs-common`

您應確保NFS服務在開機期間啟動。

iSCSI磁碟區

使用iSCSI磁碟區時、請考量下列事項：

- Kubernetes叢集中的每個節點都必須具有唯一的IQN。這是必要的先決條件。
- 若搭配使用RMCOS 4.5或更新版本、或RHEL或CentOS 8.2或更新版本 `solidfire-san` 驅動程式、請確定CHAP驗證演算法已在中設定為MD5 `/etc/iscsi/iscsid.conf`。

```
sudo sed -i 's/^\(node.session.auth.chap_algs\).*/\1 = MD5/'
/etc/iscsi/iscsid.conf
```

- 使用執行RHEL/RedHat CoreOS搭配iSCSI PV的工作節點時、請務必指定 `discard` StorageClass中的掛載選項、以執行即時空間回收。請參閱 "RedHat的文件"。

傳輸協定	作業系統	命令

iSCSI	RHEL/CentOS	1. 安裝下列系統套件：
		
		`sudo yum install -y lsscsi iscsi-initiator-utils sg3_utils device-mapper-multipath`
		
		2. 檢查iscsite-initier-utils版本是否為6.6.0.874-2.el7或更新版本：
		
		`rpm -q iscsi-initiator-utils`
		
		3. 將掃描設為手動：
		
		`sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/' /etc/iscsi/iscsid.conf`
		
		4. 啟用多重路徑：
		
		`sudo mpathconf --enable --with_multipathd y`
		
		5. 請確保如此 `iscsid` 和 `multipathd` 執行中：
		
		`sudo systemctl enable --now iscsid multipathd`
		
		6. 啟用並啟動 `iscsi`：
		
		`sudo systemctl enable --now iscsi`

傳輸協定	作業系統	命令

iSCSI	Ubuntu / DEBIAN	1. 安裝下列系統套件：
		
		`sudo apt-get install -y open-iscsi lsscsi sg3-utils multipath-tools scsitools`
		
		2. 檢查開放式iSCSI版本是否為2.0.874-5ubuntu2・10或更新版本（適用於雙聲網路）或2.0.874-7.1ubuntu6.1或更新版本（適用於焦點）：
		
		`dpkg -l open-iscsi`
		
		3. 將掃描設為手動：
		
		`sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/' /etc/iscsi/iscsid.conf`
		
		4. 啟用多重路徑：
		
		`sudo tee /etc/multipath.conf <←'EOF'`
		`defaults {`
		`user_friendly_names yes`
		`find_multipaths yes`
		`}`
		`EOF`
		`sudo systemctl enable --now multipath-tools.service`
		`sudo service multipath-tools restart`
		
		5. 請確保如此 `open-iscsi` 和 `multipath-tools` 已啟用並執行：
		
		`sudo systemctl status multipath-tools`
		`sudo systemctl enable --now open-iscsi.service`
		`sudo systemctl status open-iscsi`

若為Ubuntu 18.04、您必須使用探索目標連接埠 `iscsiadm` 開始之前 `open-iscsi` 以啟動iSCSI精靈。您也可以修改 `iscsi` 服務開始 `iscsid` 自動：

如果您想要深入瞭解自動工作節點準備（這是試用版功能）、請參閱 "請按這裡"。

自動準備工作節點

Astra Trident可以自動安裝所需的 `NFS` 和 `iSCSI` Kubernetes叢集中節點上的工具。這是*試用版功能*、不適用於*正式作業叢集。目前、此功能適用於執行 CentOS、RHEL及Ubuntu *的節點。

針對此功能、Astra Trident包含新的安裝旗標： `--enable-node-prep` 適用於與一起部署的安裝 `tridentctl`。對於使用Trident運算子的部署、請使用布林選項 `enableNodePrep`。

。 `--enable-node-prep` 安裝選項可讓Astra Trident在工作節點上掛載磁碟區時、安裝並確保NFS和iSCSI套件及/或服務正在執行。這是*試用版功能*、適用於*不符合*正式作業使用資格的開發/測試環境。

當 `--enable-node-prep` 部署Astra Trident安裝時會包含此旗標 `tridentctl`、現在的情況如下：

1. 在安裝過程中、Astra Trident會登錄其執行的節點。
2. 當提出持續磁碟區宣告（PVc）要求時、Astra Trident會從其管理的其中一個後端建立PV。
3. 在Pod中使用永久虛擬磁碟時、需要使用Astra Trident將磁碟區掛載到執行Pod的節點上。Astra Trident會嘗試安裝所需的NFS/iSCSI用戶端公用程式、並確保所需的服務處於作用中狀態。這是在掛載磁碟區之前完成的。

在第一次嘗試掛載磁碟區時、只需準備一次工作節點。只要Astra Trident以外的任何變更都不會接觸到、所有後續的Volume掛載就會成功 `NFS` 和 `iSCSI` 公用程式：

如此一來、Astra Trident就能確保Kubernetes叢集中的所有節點都具備必要的公用程式、以掛載及附加磁碟區。對於NFS磁碟區、匯出原則也應該允許掛載磁碟區。Trident可以自動管理每個後端的匯出原則、也可以管理頻外的匯出原則。

監控Astra Trident

Astra Trident提供一組Prometheus指標端點、可用來監控Astra Trident的效能。

Astra Trident提供的指標可讓您執行下列作業：

- 隨時掌握Astra Trident的健全狀況與組態。您可以檢查作業的成功程度、以及是否能如預期般與後端進行通訊。
- 檢查後端使用資訊、並瞭解後端上配置的磁碟區數量、以及所耗用的空間量等。
- 維護可用後端配置的磁碟區數量對應。
- 追蹤效能。您可以查看Astra Trident與後端及執行作業所需的時間。

根據預設、Trident的度量會顯示在目標連接埠上 `8001` 在 `/metrics` 端點：安裝Trident時*預設會啟用這些度量。

您需要的產品

- 安裝Astra Trident的Kubernetes叢集。
- Prometheus執行個體。這可以是 "容器化Prometheus部署" 或者、您也可以選擇以執行Prometheus "原生應用程式"。

步驟1：定義Prometheus目標

您應該定義Prometheus目標、以收集指標並取得有關後端Astra Trident管理的資訊、以及其建立的磁碟區等資訊。這 "部落格" 說明如何使用Prometheus和Grafana搭配Astra Trident來擷取指標。部落格說明如何在Kubernetes叢集中以營運者的形式執行Prometheus、以及建立ServiceMonitor來取得Astra Trident的指標。

步驟2：建立Prometheus ServiceMonitor

若要使用Trident指標、您應該建立監控的Prometheus ServiceMonitor `trident-csi` 服務並傾聽 `metrics` 連接埠。ServiceMonitor範例如下所示：

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
      release: prom-operator
  spec:
    jobLabel: trident
    selector:
      matchLabels:
        app: controller.csi.trident.netapp.io
    namespaceSelector:
      matchNames:
      - trident
    endpoints:
    - port: metrics
      interval: 15s
```

此ServiceMonitor定義會擷取由傳回的度量 `trident-csi` 服務、並特別尋找 `metrics` 服務的端點。因此、Prometheus現在已設定為瞭解Astra Trident的指標。

除了直接從Astra Trident取得的指標之外、Kibelet也公開了許多指標 `kubelet_volume_*` 透過IT本身的指標端點來建立指標。Kubelet可提供有關所附加磁碟區、Pod及其處理的其他內部作業的資訊。請參閱 "請按這裡"。

步驟3：使用PromQL查詢Trident度量

PromQL適用於建立傳回時間序列或表格資料的運算式。

以下是一些您可以使用的PromQL查詢：

取得**Trident**健全狀況資訊

- 來自Astra Trident的HTTP 2XX回應百分比*

```
(sum (trident_rest_ops_seconds_total_count{status_code=~"2.."} OR on()
vector(0)) / sum (trident_rest_ops_seconds_total_count)) * 100
```

- 透過狀態代碼*來自**Astra Trident**的休息回應百分比

```
(sum (trident_rest_ops_seconds_total_count) by (status_code)  / scalar
(sum (trident_rest_ops_seconds_total_count))) * 100
```

- 由**Astra Trident**執行的平均營運持續時間

```
sum by (operation)
(trident_operation_duration_milliseconds_sum{success="true"}) / sum by
(operation)
(trident_operation_duration_milliseconds_count{success="true"})
```

取得**Astra Trident**使用資訊

- 平均Volume大小*

```
trident_volume_allocated_bytes/trident_volume_count
```

- 每個後端配置的Volume空間總計*

```
sum (trident_volume_allocated_bytes) by (backend_uuid)
```

取得個別**Volume**使用量

只有同時收集kubelet度量時、才會啟用此功能。

- 每個Volume的已用空間百分比*

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

深入瞭解Astra Trident AutoSupport 遙測技術

依預設、Astra Trident會每日傳送Prometheus指標和基本後端資訊給NetApp。

- 若要停止Astra Trident將Prometheus指標和基本後端資訊傳送給NetApp、請通過 `--silence -autosupport` Astra Trident安裝期間的旗標。
- Astra Trident也可透過傳送容器記錄至NetApp Support隨選服務 `tridentctl send autosupport`。您需要觸發Astra Trident來上傳記錄。在您提交記錄之前、您應該接受NetApp 的https://www.netapp.com/company/legal/privacy-policy/["隱私權政策"^]。
- 除非另有說明、Astra Trident會從過去24小時擷取記錄。
- 您可以使用指定記錄保留時間範圍 `--since` 旗標。例如： `tridentctl send autosupport --since=1h`。此資訊會透過收集和傳送 `trident-autosupport` 安裝於Astra Trident旁的容器。您可以從取得Container映像 "Trident AutoSupport 的"。
- Trident AutoSupport 無法收集或傳輸個人識別資訊（PII）或個人資訊。隨附a "EULA" 這不適用於Trident Container映像本身。您可以深入瞭解NetApp對資料安全性與信任的承諾 "請按這裡"。

Astra Trident傳送的有效負載範例如下：

```
{
  "items": [
    {
      "backendUUID": "ff3852e1-18a5-4df4-b2d3-f59f829627ed",
      "protocol": "file",
      "config": {
        "version": 1,
        "storageDriverName": "ontap-nas",
        "debug": false,
        "debugTraceFlags": null,
        "disableDelete": false,
        "serialNumbers": [
          "nwkvzfanek_SN"
        ],
        "limitVolumeSize": ""
      },
      "state": "online",
      "online": true
    }
  ]
}
```

- 此資訊將傳送至NetApp的「不只是」端點。AutoSupport AutoSupport如果您使用私有登錄來儲存容器映像、可以使用 `--image-registry` 旗標。
- 您也可以產生安裝Yaml檔案來設定Proxy URL。您可以使用來完成這項作業 `tridentctl install --generate-custom-yaml` 以建立Yaml檔案並新增 `--proxy-url` 的引數 `trident-autosupport` 中的Container `trident-deployment.yaml`。

停用Astra Trident度量

若要在報告中停用*指標、您應該產生自訂YAM（使用 `--generate-custom-yaml` 標記）並加以編輯以移除 `--metrics` 無法為呼叫旗標 `trident-main`容器。

版權資訊

商標資訊