Trident オペレータとともに導入

Astra Trident

NetApp
April 16, 2024

目次

Trident オペレータとともに導入

Trident のオペレータが、 Astra Trident を導入できます。Trident オペレータは、手動または Helm を使用して導入できます。

をまだ理解していない場合は、を参照してください "基本概念"今こそ、そのための絶好の機会です。

必要なもの

Astra Trident を導入するには、次の前提条件を満たしている必要があります。

- Kubernetes 1.17 以降を実行するサポート対象の Kubernetes クラスタに対するすべての権限が必要です。
- サポートされているネットアップストレージシステムを利用できるようにしておきます。
- すべての Kubernetes ワーカーノードからボリュームをマウントできます。
- 使用する Kubernetes クラスタを管理するために 'kubectl(OpenShift を使用している場合は 'OC) をインストールして構成した Linux ホストがあります
- 「 KUBECONFIG 」環境変数を、 Kubernetes クラスタ構成を指すように設定しておきます。
- を有効にしておきます "Astra Trident に必要な機能ゲート"。
- Kubernetes と Docker Enterprise を併用する場合は、 "CLI へのアクセスを有効にする手順は、ユーザが行ってください"。

それはすべてですか？最高！それでは始めましょう。

Helm を使用して Trident オペレータを導入します

Helm を使用して Trident オペレータを導入するには、以下の手順を実行します。

必要なもの

上記の前提条件に加え、 Helm を使用して Trident Operator を導入するには、次のものが必要です。

- Kubernetes 1.17 以降
- Helm バージョン 3

手順

1. からインストーラバンドルをダウンロードします "Trident GitHub" ページインストーラバンドルには 'Helm チャートが用意されています
2. 「 helm install 」コマンドを使用して、展開の名前を指定します。次の例を参照してください。

```
helm install <name> trident-operator-21.07.1.tgz --namespace <namespace
you want to use for Trident>
```

Trident の名前空間をまだ作成していない場合は '--create-namespace パラメータを helm install コマンドに追加できますHelm によってネームスペースが自動的に作成されます。

インストール中に設定データを渡すには、次の 2 つの方法があります。

- `--values]` （ または `'-f`）：オーバーライドを含む YAML ファイルを指定しますこれは複数回指定でき、右端のファイルが優先されます。
- `--set`：コマンドラインでオーバーライドを指定します

たとえば ' デフォルト値の `debug` を変更するには ' 次のように --set` コマンドを実行します

```
$ helm install <name> trident-operator-21.07.1.tgz --set tridentDebug=true
```

Helm チャートの一部である `values] の .yaml ファイルには、キーのリストとデフォルト値が表示されます。

「 helm list 」は、名前、名前空間、グラフ、ステータスなど、インストールに関する詳細を表示します。 アプリケーションのバージョン、リビジョン番号など。

Trident オペレータを手動で導入

Trident のオペレータを手動で導入するには、以下の手順を実行します。

ステップ 1 ： Kubernetes クラスタを確認する

まず、 Linux ホストにログインして、 _working _ 、 "サポートされる Kubernetes クラスタ" に必要な権限があることを確認します。

> OpenShift では、以降のすべての例で「 kubectl 」ではなく「 OC 」を使用し、「 OC login-u SYSTEM ： admin 」または「 OC login-u kube-admin 」を実行して最初に「 *system:admin 」としてログインします。

Kubernetes のバージョンが 1.17 よりも新しいかどうかを確認するには、次のコマンドを実行します。

```
kubectl version
```

Kubernetes クラスタ管理者の権限があるかどうかを確認するには、次のコマンドを実行します。

```
kubectl auth can-i '*' '*' --all-namespaces
```

Docker Hub のイメージを使用してポッドを起動し、ポッドネットワーク経由でストレージシステムにアクセスできるかどうかを確認するには、次のコマンドを実行します。

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

手順 2 ：オペレータをダウンロードして設定します

21.01 以降、 Trident Operator はクラスタを対象とします。Trident オペレータを使用して Trident をインストールするには、「 TridentOrchestrator 」カスタムリソース定義（ CRD ）を作成し、その他のリソースを定義する必要があります。Astra Trident をインストールする前に、次の手順を実行してオペレータをセットアップする必要があります。

1. の最新バージョンをダウンロードします "Trident インストーラバンドル" _Downloads_section から抽出します

```
wget https://github.com/NetApp/trident/releases/download/v21.04/trident-
installer-21.04.tar.gz
tar -xf trident-installer-21.04.tar.gz
cd trident-installer
```

2. 適切な CRD マニフェストを使用して、「 TridentOrchestrator 」 CRD を作成します。次に、後で「 TridentOrchestrator 」カスタムリソースを作成して、オペレータによってインストールをインスタンス化します。

   次のコマンドを実行します。

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. 「 TridentOrchestrator 」 CRD が作成されたら、オペレータの展開に必要な次のリソースを作成します。

   - オペレータのサービスアカウント
   - ClusterRole および ClusterRoleBinding をサービスアカウントにバインドする
   - 専用の PodSecurityPolicy
   - 演算子自体

     Trident インストーラには、これらのリソースを定義するマニフェストが含まれています。デフォルトでは ' 演算子は trident' 名前空間に配置されます'trident' 名前空間が存在しない場合は ' 次のマニフェストを使用して名前空間を作成します

```
$ kubectl apply -f deploy/namespace.yaml
```

4. デフォルトの 'trident` 名前空間以外の名前空間に演算子を配備するには `erviceaccount.yaml 'clusterrolebinding.yaml ' および `operator.yML' マニフェストを更新し 'bundle.yaml を生成する必要があります

   次のコマンドを実行して YAML マニフェストを更新し、「 customizization.yaml 」を使用して「 bundle.yaml 」を生成します。

```
kubectl kustomize deploy/ > deploy/bundle.yaml
```

次のコマンドを実行してリソースを作成し、オペレータを配置します。

```
kubectl create -f deploy/bundle.yaml
```

5. 展開後にオペレータのステータスを確認するには、次の手順を実行します。

```
$ kubectl get deployment -n <operator-namespace>
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
trident-operator    1/1     1            1           3m

$ kubectl get pods -n <operator-namespace>
NAME                                READY   STATUS              RESTARTS
AGE
trident-operator-54cb664d-lnjxh     1/1     Running             0
3m
```

オペレータによる導入で、クラスタ内のいずれかのワーカーノードで実行されるポッドが正常に作成されます。

Kubernetes クラスタには、オペレータのインスタンスが * 1 つしか存在しないようにしてください。Trident のオペレータが複数の環境を構築することは避けてください。

手順3：作成 `TridentOrchestrator` Tridentをインストール

これで、オペレータを使って Astra Trident をインストールする準備ができました。これには 'TridentOrchestrator を作成する必要がありますTrident インストーラには 'TridentOrchestrator を作成するための定義例が付属していますこれは 'trident' 名前空間にインストールされます

```
$ kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

$ kubectl describe torc trident
Name:         trident
Namespace:
Labels:       <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:         TridentOrchestrator
...
Spec:
  Debug:     true
  Namespace: trident
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:21.04
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Enable Node Prep:          false
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Silence Autosupport:  false
    Trident Image:        netapp/trident:21.04.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v21.04.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Trident オペレータは 'TridentOrchestrator 仕様の属性を使用して 'Astra Trident のインストール方法をカスタマイズできますを参照してください "Trident の導入をカスタマイズ"。

「 TridentOrchestrator 」のステータスは、インストールが成功したかどうかを示し、インストールされている Trident のバージョンを表示します。

ステータス	説明
インストール中です	オペレータは、この「 TridentOrchestrator 」 CR を使用して Astra Trident をインストールしています。
インストール済み	Astra Trident のインストールが完了しました。
アンインストール中です	オペレータは 'stra Trident をアンインストールしていますこれは 'pec.uninstall=true だからです
アンインストール済み	Astra Trident がアンインストールされました。
失敗しました	オペレータは Astra Trident をインストール、パッチ適用、更新、またはアンインストールできませんでした。オペレータはこの状態からのリカバリを自動的に試みます。この状態が解消されない場合は、トラブルシューティングが必要です。
更新中です	オペレータが既存のインストールを更新しています。
エラー	「 TridentOrchestrator 」は使用されません。別のファイルがすでに存在します。

インストール中に 'TridentOrchestrator のステータスが Installing から Installed に変わります「失敗」ステータスが表示され、オペレータが自身で回復できない場合は、オペレータのログを確認する必要があります。を参照してください "トラブルシューティング" セクション。

Astra Trident のインストールが完了しているかどうかを確認するには、作成したポッドを確認します。

```
$ kubectl get pod -n trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-7d466bf5c7-v4cpw        5/5     Running   0          1m
trident-csi-mr6zc                   2/2     Running   0          1m
trident-csi-xrp7w                   2/2     Running   0          1m
trident-csi-zh2jt                   2/2     Running   0          1m
trident-operator-766f7b8658-ldzsv   1/1     Running   0          3m
```

また 'tridentctl を使用して 'Astra Trident のバージョンを確認することもできます

```
$ ./tridentctl -n trident version
+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 21.04.0        | 21.04.0        |
+----------------+----------------+
```

これで、バックエンドを作成できます。を参照してください "導入後のタスク"。

導入時の問題のトラブルシューティングについては、を参照してください "トラブルシューティング" セクション。

Trident オペレータの環境をカスタマイズ

Trident オペレータは 'TridentOrchestrator 仕様の属性を使用して 'Astra Trident のインストール方法をカスタマイズできます

属性のリストについては、次の表を参照してください。

パラメータ	説明	デフォルト
「 namespace 」を参照してください	Astra Trident をインストールするネームスペース	デフォルト
「バグ」	Astra Trident のデバッグを有効にします	いいえ
「 IPv6 」	IPv6 経由の Astra Trident をインストール	いいえ
k8sTimeout	Kubernetes 処理のタイムアウト	30 秒
'ilenceAutosupport	AutoSupport バンドルをネットアップに自動的に送信しない	いいえ
enableNodePrep	ワーカーノードの依存関係を自動的に管理（ * beta * ）	いいえ
「 autosupportImage 」を参照してください	AutoSupport テレメトリのコンテナイメージ	「 NetApp/trident-autosupport ： 21.04.0 」
「 autosupportProxy 」と入力します	AutoSupport テレメトリを送信するプロキシのアドレス / ポート	"http://proxy.example.com:8888"
uninstall	Astra Trident のアンインストールに使用するフラグ	いいえ
「 logFormat 」	Astra Trident のログ形式が使用 [text 、 JSON]	テキスト（ Text ）
「 tridentImage 」のように入力します	インストールする Astra Trident イメージ	「 NetApp / Trident ： 21.04 」
「 imageRegistry 」と入力します	内部レジストリへのパス。形式は「 <registry fqdn>[:port][/ssubpath]] 」です	"k83.gcr.io/sig-storage (k8s 1.17+) または Qua.io/k8scsi"
「 kubeletDir 」を参照してください	ホスト上の kubelet ディレクトリへのパス	「 /var/lib/kubelet 」
「 wippeout 」	Astra Trident を完全に削除するために削除するリソースのリスト	
「 imagePullSecrets 」	内部レジストリからイメージをプルするシークレット	

「 TridentOrchestrator 」では、どのネームスペース Astra Trident がにインストールされているかを示すために pec.namespace` が指定されています。このパラメータ * は、 Astra Trident のインストール後に更新できません * 。これを実行すると、「 TridentOrchestrator 」のステータスが「 Failed 」に変わります。Astra Trident は、ネームスペース間での移行を意図したものではありません。

自動ワーカーノードの前処理は、非本番環境でのみ使用することを目的とした * ベータ機能です。

「 TridentOrchestrator 」を定義するときに上記の属性を使用して、インストールをカスタマイズできます。次に例を示します。

```
$ cat deploy/crds/tridentorchestrator_cr_imagepullsecrets.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  tridentImage: netapp/trident:21.04.0
  imagePullSecrets:
  - thisisasecret
```

「 TridentOrchestrator 」引数で許可される範囲を超えてインストールをカスタマイズする場合は、「 tridentctl 」を使用して必要に応じて変更できるカスタム YAML マニフェストを生成することを検討してください。

著作権に関する情報

商標に関する情報