安装Trident Protect

Trident

NetApp
March 03, 2025

目录

安装Trident Protect

Trident Protect要求

首先验证您的操作环境、应用程序集群、应用程序和许可证是否已准备就绪。确保您的环境满足这些要求、才能部署和运行Trident Protect。

Trident Protect Kubbernetes集群兼容性

Trident Protect与各种完全托管和自行管理的Kubernetes产品兼容、其中包括：

- Amazon Elelic Kubelnetes Service (EKS)
- Google Kubernetes Engine （ GKEE ）
- Microsoft Azure Kubernetes Service （ AKS ）
- Red Hat OpenShift
- SUSE试用者
- VMware Tanzu产品组合
- 上游Kubernetes

确保为安装Trident Protect的集群配置了正在运行的快照控制器以及相关的CRD。要安装快照控制器，请参见 "这些说明"。

Trident保护存储后端兼容性

Trident Protect支持以下存储后端：

- 适用于 NetApp ONTAP 的 Amazon FSX
- Cloud Volumes ONTAP
- ONTAP存储阵列
- Google Cloud NetApp卷
- Azure NetApp Files

确保存储后端满足以下要求：

- 确保连接到集群的NetApp存储使用的是Astra Trident 24.02或更高版本(建议使用Trident 24.10)。
  - 如果Astra Trident的版本低于24.06.1、而您计划使用NetApp SnapMirror灾难恢复功能、则需要手动启用Astra Control配置程序。
- 确保已安装最新的Astra控件配置程序(从Astra Trident 24.06.1开始、默认情况下已安装并启用)。
- 确保您有一个NetApp ONTAP存储后端。
- 确保已配置用于存储备份的对象存储分段。
- 创建您计划用于应用程序或应用程序数据管理操作的任何应用程序卷。Trident Protect不会为您创建这些命名空间；如果在自定义资源中指定不存在的命名空间、则操作将失败。

NAS经济型卷的要求

Trident Protect支持对NAS经济型卷执行备份和还原操作。目前不支持将快照、克隆和SnapMirror复制到NAS经济型卷。您需要为计划与Trident Protect结合使用的每个NAS经济型卷启用一个Snapshot目录。

某些应用程序与使用Snapshot目录的卷不兼容。对于这些应用程序、您需要通过在ONTAP存储系统上运行以下命令来隐藏Snapshot目录：

```
nfs modify -vserver <svm> -v3-hide-snapshot enabled
```

您可以通过对每个NAS经济型卷运行以下命令来启用Snapshot目录、并将其替换 `<volume-UUID>`为要更改的卷的UUID：

```
tridentctl update volume <volume-UUID> --snapshot-dir=true --pool-level
=true -n trident
```

通过将Trident后端配置选项设置为，您可以默认为 `true`新卷启用快照目录 `snapshotDir`。现有卷不受影响。

使用KubeVirt VM保护数据

保护在KubeVirt VM上运行的应用程序时、Trident Protect 24.10 24.10.1及更高版本的行为会有所不同。对于这两个版本、您都可以在数据保护操作期间启用或禁用文件系统冻结和解除冻结。

Trident智能驭领保障24.10

Trident Protect 24.10不会在数据保护操作期间自动确保KubeVirt VM文件系统的状态一致。如果要使用Trident Protect 24.10保护KubeVirt VM数据、则需要在执行数据保护操作之前手动为文件系统启用冻结/取消冻结功能。这样可确保文件系统处于一致状态。

您可以先配置Trident Protect 24.10、以便在执行数据保护操作期间管理VM文件系统的冻结和解除冻结"正在配置虚拟化"、然后使用以下命令：

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=true -n trident-protect
```

Trident®24.10.1及更高版本

从Trident Protect 24.10.1开始、Trident Protect会在数据保护操作期间自动冻结和解除冻结KubeVirt文件系统。或者、您可以使用以下命令禁用此自动行为：

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=false -n trident-protect
```

SnapMirror复制的要求

NetApp SnapMirror复制可用于以下ONTAP解决方案的Trident Protect：

- 内部NetApp FAS、AFF和ASA集群
- NetApp ONTAP Select
- NetApp Cloud Volumes ONTAP
- 适用于 NetApp ONTAP 的 Amazon FSX

SnapMirror复制的ONTAP集群要求

如果您计划使用SnapMirror复制、请确保ONTAP集群满足以下要求：

- **Astra**控件配置程序或：Astra控件配置程序或Trident必须同时位于使用ONTAP作为后端的源和目标Trident集群上。Trident Protect支持使用以下驱动程序支持的存储类通过NetApp SnapMirror技术进行复制：
  - `ontap-NAS`
  - `ontap-san`
- 许可证：必须在源和目标ONTAP集群上启用使用数据保护包的ONTAP SnapMirror异步许可证。有关详细信息、请参见 "ONTAP 中的SnapMirror许可概述" 。

SnapMirror复制的对等注意事项

如果您计划使用存储后端对等、请确保您的环境满足以下要求：

- 集群和**SVM**：ONTAP存储后端必须建立对等状态。有关详细信息、请参见 "集群和 SVM 对等概述" 。

  确保两个ONTAP集群之间的复制关系中使用的SVM名称是唯一的。

- **Astra**控件配置程序或**Trident**和**SVM**：对等远程SVM必须可供目标集群上的Astra控件配置程序或Trident使用。
- 托管后端：您需要在Trident Protect中添加和管理ONTAP存储后端、才能创建复制关系。
- 基于**TCP**的**NVMe**：对于使用基于TCP协议的Trident的存储后端、NVMe保护不支持NetApp SnapMirror复制。

用于SnapMirror复制的Trident / ONTAP配置

Trident Protect要求您至少配置一个存储后端、以便为源集群和目标集群同时支持复制。如果源集群和目标集群相同、则目标应用程序应使用与源应用程序不同的存储后端、以获得最佳故障恢复能力。

安装和配置Trident Protect

如果您的环境满足Trident Protect的要求、您可以按照以下步骤在集群上安装Trident Protect。您可以从NetApp获取Trident Protect、也可以从您自己的私人注册表安装它。如果集群无法访问Internet、则从专用注册表进行安装非常有用。

安装Trident Protect

安装**Trident Protect from NetApp**

步骤

1. 添加Trident Helm存储库：

```
helm repo add netapp-trident-protect
https://netapp.github.io/trident-protect-helm-chart
```

2. 安装Trident Protect CRD：

```
helm install trident-protect-crds netapp-trident-protect/trident-
protect-crds --version 100.2502.0 --create-namespace --namespace
trident-protect
```

3. 使用Helm安装Trident Protect。替换 `<name_of_cluster>`为集群名称、此名称将分配给集群并用于标识集群的备份和快照：

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name_of_cluster> --version 100.2502.0 --create
-namespace --namespace trident-protect
```

从专用注册表安装**Trident Protect**

如果您的Trident集群无法访问Internet、则可以从专用映像注册表安装Kubbernetes Protect。在以下示例中、将括号中的值替换为您环境中的信息：

步骤

1. 将以下映像提取到本地计算机、更新标记、然后将其推送到您的私人注册表：

```
netapp/controller:25.02.0
netapp/restic:25.02.0
netapp/kopia:25.02.0
netapp/trident-autosupport:25.02.0
netapp/exechook:25.02.0
netapp/resourcebackup:25.02.0
netapp/resourcerestore:25.02.0
netapp/resourcedelete:25.02.0
bitnami/kubectl:1.30.2
kubebuilder/kube-rbac-proxy:v0.16.0
```

例如：

```
docker pull netapp/controller:25.02.0
```

```
docker tag netapp/controller:25.02.0 <private-registry-
url>/controller:25.02.0
```

```
docker push <private-registry-url>/controller:25.02.0
```

2. 创建Trident Protect系统命名空间：

```
kubectl create ns trident-protect
```

3. 登录到注册表：

```
helm registry login <private-registry-url> -u <account-id> -p <api-
token>
```

4. 创建用于私人注册表身份验证的拉机密：

```
kubectl create secret docker-registry regcred --docker
-username=<registry-username> --docker-password=<api-token> -n
trident-protect --docker-server=<private-registry-url>
```

5. 添加Trident Helm存储库：

```
helm repo add netapp-trident-protect
https://netapp.github.io/trident-protect-helm-chart
```

6. 创建一个名为的文件 `protectValues.yaml`。确保它包含以下Trident保护设置：

```
---
image:
  registry: <private-registry-url>
imagePullSecrets:
  - name: regcred
controller:
  image:
    registry: <private-registry-url>
rbacProxy:
  image:
    registry: <private-registry-url>
crCleanup:
  imagePullSecrets:
    - name: regcred
webhooksCleanup:
  imagePullSecrets:
    - name: regcred
```

7. 安装Trident Protect CRD：

```
helm install trident-protect-crds netapp-trident-protect/trident-protect-crds --version 100.2502.0 --create-namespace --namespace trident-protect
```

8. 使用Helm安装Trident Protect。替换 `<name_of_cluster>`为集群名称、此名称将分配给集群并用于标识集群的备份和快照：

```
helm install trident-protect netapp-trident-protect/trident-protect --set clusterName=<name_of_cluster> --version 100.2502.0 --create-namespace --namespace trident-protect -f protectValues.yaml
```

安装Trident Protect命令行界面插件

您可以使用Trident Protect命令行插件(Trident实用程序的扩展)创建Trident Protect `tridentctl`自定义资源(CRS)并与之交互。

安装Trident Protect命令行界面插件

在使用命令行实用程序之前、您需要将其安装在用于访问集群的计算机上。根据您的计算机使用的是x64 CPU还是ARM CPU、执行以下步骤。

下载适用于**Linux amd64 CPU**的插件

步骤

1. 下载Trident Protect命令行界面插件：

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.02.0/tridentctl-protect-linux-amd64
```

下载适用于**Linux ARM64 CPU**的插件

步骤

1. 下载Trident Protect命令行界面插件：

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.02.0/tridentctl-protect-linux-arm64
```

下载适用于**Mac amd64 CPU**的插件

步骤

1. 下载Trident Protect命令行界面插件：

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.02.0/tridentctl-protect-macos-amd64
```

下载适用于**Mac ARM64 CPU**的插件

步骤

1. 下载Trident Protect命令行界面插件：

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.02.0/tridentctl-protect-macos-arm64
```

1. 为插件二进制文件启用执行权限：

```
chmod +x tridentctl-protect
```

2. 将插件二进制文件复制到路径变量中定义的位置。例如、 `/usr/bin`或 `/usr/local/bin`(您可能需要提升Privileges)：

```
cp ./tridentctl-protect /usr/local/bin/
```

3. 您也可以将插件二进制文件复制到主目录中的某个位置。在这种情况下、建议确保此位置属于您的路径变量：

```
cp ./tridentctl-protect ~/bin/
```

通过将插件复制到路径变量中的某个位置、您可以通过在任意位置键入或 `tridentctl protect`来使用此插件 `tridentctl-protect。

查看Trident命令行界面插件帮助

您可以使用内置插件的帮助功能获取有关插件功能的详细帮助：

步骤

1. 使用帮助功能查看使用指南：

```
tridentctl-protect help
```

启用命令自动完成

安装Trident Protect命令行界面插件后、您可以对某些命令启用自动完成。

为**bash shell**启用自动完成

步骤

1. 下载完成脚本：

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.02.0/tridentctl-completion.bash
```

2. 在主目录中创建一个新目录以包含该脚本：

```
mkdir -p ~/.bash/completions
```

3. 将下载的脚本移动到 `~/.bash/completions`目录：

```
mv tridentctl-completion.bash ~/.bash/completions/
```

4. 将以下行添加到 `~/.bashrc`主目录中的文件：

```
source ~/.bash/completions/tridentctl-completion.bash
```

为**Z shell**启用自动完成

步骤

1. 下载完成脚本：

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.02.0/tridentctl-completion.zsh
```

2. 在主目录中创建一个新目录以包含该脚本：

```
mkdir -p ~/.zsh/completions
```

3. 将下载的脚本移动到 `~/.zsh/completions`目录：

```
mv tridentctl-completion.zsh ~/.zsh/completions/
```

4. 将以下行添加到 `~/.zprofile`主目录中的文件：

```
source ~/.zsh/completions/tridentctl-completion.zsh
```

结果

下次shell登录时、您可以使用带有trdentcd-protect插件的命令自动完成。

自定义Trident Protect安装

您可以自定义Trident Protect的默认配置、以满足您环境的特定要求。

指定Trident Protect容器资源限制

安装Trident Protect后、您可以使用配置文件为Trident Protect容器指定资源限制。通过设置资源限制、您可以控制Trident Protect操作占用的集群资源量。

步骤

1. 创建一个名为的文件 `resourceLimits.yaml`。
2. 根据您的环境需求、使用Trident Protect容器的资源限制选项填充文件。

   以下示例配置文件显示了可用设置、并包含每个资源限制的默认值：

```
---
jobResources:
  defaults:
    limits:
      cpu: 8000m
      memory: 10000Mi
      ephemeralStorage: ""
    requests:
      cpu: 100m
      memory: 100Mi
      ephemeralStorage: ""
  resticVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  resticVolumeRestore:
    limits:
      cpu: ""
```

```
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeRestore:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
```

3. 应用文件中的值 `resourceLimits.yaml`：

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f resourceLimits.yaml --reuse-values
```

自定义安全上下文约束

安装Trident Protect后、您可以使用配置文件修改Trident Protect容器的OpenShift安全上下文约束(SCC)。这些限制为Red Hat OpenShift集群中的Pod定义了安全限制。

步骤

1. 创建一个名为的文件 `sccconfig.yaml`。
2. 将scc选项添加到文件中、然后根据环境的需要修改参数。

   以下示例显示了SCC选项参数的默认值：

```
scc:
  create: true
  name: trident-protect-job
  priority: 1
```

下表介绍了SCC选项的参数：

参数	Description	Default
创建	确定是否可以创建SCC资源。只有在将设置为 `true`且Helm安装过程标识OpenShift环境时、才会创建SCC资源 `scc.create`。如果不在OpenShift上运行，或者如果 `scc.create`设置为 `false`，则不会创建任何SCC资源。	true
name	指定SCC的名称。	Trident保护作业
优先级	定义SCC的优先级。优先级值较高的SCC会在优先级值较低的SCC之前进行评估。	1

3. 应用文件中的值 `sccconfig.yaml`：

```
helm upgrade trident-protect netapp-trident-protect/trident-protect -f
sccconfig.yaml --reuse-values
```

此操作会将默认值替换为文件中指定的值 `sccconfig.yaml`。

为Trident Protect配置NetApp AutoSupport连接

您可以通过为连接配置代理来更改Trident Protect连接到NetApp支持以上传支持包的方式。您可以根据需要将代理配置为使用安全或不安全连接。

配置安全代理连接

步骤

1. 为Trident Protect支持包上传配置安全代理连接：

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect --set autoSupport.proxy=http://my.proxy.url
--reuse-values
```

配置不安全的代理连接

步骤

1. 为跳过Trident验证的TLS Protect支持包上传配置不安全的代理连接：

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect --set autoSupport.proxy=http://my.proxy.url
--set autoSupport.insecure=true --reuse-values
```

将Trident保护Pod限制为特定节点

您可以使用Kubenetes nodeSelecte节点选择约束根据节点标签控制哪些节点有资格运行Trident Protect Pod。默认情况下、Trident Protect仅限于运行Linux的节点。您可以根据需要进一步自定义这些限制。

步骤

1. 创建一个名为的文件 `nodeSelectorConfig.yaml`。
2. 将nodeSelector选项添加到文件中、然后修改文件以添加或更改节点标签、从而根据环境需求进行限制。例如、以下文件包含默认操作系统限制、但也针对特定区域和应用程序名称：

```
nodeSelector:
  kubernetes.io/os: linux
  region: us-west
  app.kubernetes.io/name: mysql
```

3. 应用文件中的值 `nodeSelectorConfig.yaml`：

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f nodeSelectorConfig.yaml --reuse-values
```

此操作会将默认限制替换为您在文件中指定的限制 `nodeSelectorConfig.yaml`。

禁用每日Trident Protect AutoSupport捆绑包上传

您也可以禁用计划的每日Trident Protect AutoSupport支持包上传。

默认情况下、Trident Protect会收集有助于处理您可能创建的任何NetApp支持案例的支持信息、包括有关集群和托管应用程序的日志、指标和拓扑信息。Trident Protect会按每日计划将这些支持包发送给NetApp。您可以随时手动"生成支持包"执行此操作。

步骤

1. 创建一个名为的文件 `autosupportconfig.yaml`。
2. 将AutoSupport选项添加到文件中、然后根据您的环境需求修改参数。

   以下示例显示了AutoSupport选项参数的默认值：

   ```
   autoSupport:
     enabled: true
   ```

   如果 `autoSupport.enabled`将设置为 `false`，则会禁用AutoSupport支持包的每日上传。

3. 应用文件中的值 `autosupportconfig.yaml`：

   ```
   helm upgrade trident-protect netapp-trident-protect/trident-protect -f
   autosupportconfig.yaml --reuse-values
   ```