还原应用程序
Trident

NetApp
February 02, 2026

目录

还原应用程序

使用Trident Protect 恢复应用程序

您可以使用Trident Protect 从快照或备份中恢复您的应用程序。将应用程序恢复到同一集群时，从现有快照恢复速度会更快。

- 还原应用程序时、为该应用程序配置的所有执行挂钩都会随该应用程序还原。如果存在还原后执行挂钩、则它会在还原操作中自动运行。
- 对于 qtree 卷，支持从备份还原到其他命名空间或原始命名空间。但是，对于 qtree 卷，不支持从快照还原到其他命名空间或原始命名空间。
- 您可以使用高级设置来自定义恢复操作。欲了解更多信息，请参阅 "使用高级Trident Protect 恢复设置"。

从备份还原到其他命名空间

当您使用 BackupRestore CR 将备份还原到不同的命名空间时， Trident Protect 会在新的命名空间中还原应用程序，并为还原的应用程序创建一个应用程序 CR。为了保护已恢复的应用程序，可以创建按需备份或快照，或者制定保护计划。

- 将备份还原到具有现有资源的其他命名空间不会更改与备份中的资源共享名称的任何资源。要还原备份中的所有资源、请删除并重新创建目标命名空间、或者将备份还原到新命名空间。
- 使用 CR 恢复到新命名空间时，必须先手动创建目标命名空间，然后再应用 CR。Trident Protect 仅在使用 CLI 时才会自动创建命名空间。

开始之前

确保AWS会话令牌到期时间足以执行任何长时间运行的S3还原操作。如果令牌在还原操作期间过期、则操作可能会失败。

- 有关检查当前会话令牌到期时间的详细信息、请参见 "AWS API文档"。
- 有关AWS资源凭据的详细信息、请参见 "AWS IAM文档"。

当您使用 Kopia 作为数据移动器恢复备份时，您可以选择在 CR 中指定注释或使用 CLI 来控制 Kopia 使用的临时存储的行为。请参阅 "文档"有关您可以配置的选项的更多信息。使用 `tridentctl-protect create --help`有关使用Trident Protect CLI 指定注释的更多信息，请参阅命令。

使用CR

步骤

1. 创建自定义资源(CR)文件并将其命名为 `trident-protect-backup-restore-cr.yaml`。
2. 在创建的文件中、配置以下属性：
   - 。**metadata.name**:(*required*_)此自定义资源的名称；请为您的环境选择一个唯一且合理的名称。
   - **spec.appArchivePath**：AppVault中存储备份内容的路径。您可以使用以下命令查找此路径：

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**：(*required*)存储备份内容的AppVault的名称。
   - 。**spec.namespaceMapping**:还原操作的源命名空间到目标命名空间的映射。将和 `my-destination-namespace`替换 `my-source-namespace`为您环境中的信息。

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: BackupRestore
     metadata:
       name: my-cr-name
       namespace: my-destination-namespace
     spec:
       appArchivePath: my-backup-path
       appVaultRef: appvault-name
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (可 选)如果只需要选择要还原的应用程序的某些资源、请添加包含或排除带有特定标签的资源的筛选：

   Trident Protect 会自动选择一些资源，因为它们与您选择的资源存在关联。例如，如果您选择持久卷声明资源并且它有一个关联的 pod， Trident Protect 还会恢复关联的 pod。

   - **resourceFilter.resourceSourcedionCritera**：(筛选时需要)使用 `Include`或包含或 `Exclude`排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：
     - **resourceFilter.resourceMatcher**：resourceMatcher对象数组。如果在此数组中定义多个元素，它们将作为OR操作进行匹配，每个元素(组、种类、版本)中的字段将作为AND操作进行匹配。
       - **resourceMatcher[].group**：(可 选)要筛选的资源的组。
       - **resourceMatcher[].KIND**：(可 选)要筛选的资源种类。
       - **resourceMatcher[].version**：(可 选)要筛选的资源版本。
       - **resourceMatcher[].names**：(可 选)要筛选的资源的Kubernetes metadata.name字段中的

名称。

- **resourceMatcher[].namespies**：(可 选)要筛选的资源的Kubernetes metadata.name字段中的命名空间。
- *resourceMatcher[].labelSelectors *：(可 选)资源的Kubernetes metadata.name字段中的标签选择器字符串，如中所定义 "Kubernetes 文档"。例如：`"trident.netapp.io/os=linux"`。

  例如：

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. 使用正确的值填充文件后 `trident-protect-backup-restore-cr.yaml` 、应用CR：

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

使用命令行界面

步骤

1. 将备份还原到其他命名空间、将括号中的值替换为环境中的信息。此 `namespace-mapping`参数使用冒号分隔的卷来将源卷的源卷映射到格式为的正确目标卷的 `source1:dest1,source2:dest2`卷。例如：

```
tridentctl-protect create backuprestore <my_restore_name> \
--backup <backup_namespace>/<backup_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
-n <application_namespace>
```

从备份还原到原始命名空间

您可以随时将备份还原到原始命名空间。

开始之前

确保AWS会话令牌到期时间足以执行任何长时间运行的S3还原操作。如果令牌在还原操作期间过期、则操作可能会失败。

- 有关检查当前会话令牌到期时间的详细信息、请参见 "AWS API文档"。
- 有关AWS资源凭据的详细信息、请参见 "AWS IAM文档"。

> 当您使用 Kopia 作为数据移动器恢复备份时，您可以选择在 CR 中指定注释或使用 CLI 来控制 Kopia 使用的临时存储的行为。请参阅 "文档"有关您可以配置的选项的更多信息。使用 `tridentctl-protect create --help`有关使用Trident Protect CLI 指定注释的更多信息，请参阅命令。

使用CR

步骤

1. 创建自定义资源(CR)文件并将其命名为 `trident-protect-backup-ipr-cr.yaml`。
2. 在创建的文件中、配置以下属性：
   - 。**metadata.name**:(*required_*)此自定义资源的名称；请为您的环境选择一个唯一且合理的名称。
   - **spec.appArchivePath**：AppVault中存储备份内容的路径。您可以使用以下命令查找此路径：

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**：(*required*)存储备份内容的AppVault的名称。

     例如：

```
---
apiVersion: protect.trident.netapp.io/v1
kind: BackupInplaceRestore
metadata:
  name: my-cr-name
  namespace: my-app-namespace
spec:
  appArchivePath: my-backup-path
  appVaultRef: appvault-name
```

3. (可 选)如果只需要选择要还原的应用程序的某些资源、请添加包含或排除带有特定标签的资源的筛选：

   Trident Protect 会自动选择一些资源，因为它们与您选择的资源存在关联。例如，如果您选择持久卷声明资源并且它有一个关联的 pod， Trident Protect 还会恢复关联的 pod。

   - **resourceFilter.resourceSourcedionCritera**：(筛选时需要)使用 `Include`或包含或 `Exclude`排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：
     - **resourceFilter.resourceMatcher**：resourceMatcher对象数组。如果在此数组中定义多个元素，它们将作为OR操作进行匹配，每个元素(组、种类、版本)中的字段将作为AND操作进行匹配。
       - **resourceMatcher[].group**：(可 选)要筛选的资源的组。
       - **resourceMatcher[].KIND**：(可 选)要筛选的资源种类。
       - **resourceMatcher[].version**：(可 选)要筛选的资源版本。
       - **resourceMatcher[].names**：(可 选)要筛选的资源的Kubernetes metadata.name字段中的名称。
       - **resourceMatcher[].namespies**：(可 选)要筛选的资源的Kubernetes metadata.name字段

中的命名空间。

- *resourceMatcher[].labelSelectors *：(可 选)资源的Kubernetes metadata.name字段中的标签选择器字符串，如中所定义 "Kubernetes 文档"。例如：`"trident.netapp.io/os=linux"`。

  例如：

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. 使用正确的值填充文件后 `trident-protect-backup-ipr-cr.yaml` 、应用CR：

```
kubectl apply -f trident-protect-backup-ipr-cr.yaml
```

使用命令行界面

步骤

1. 将备份还原到原始命名空间、将括号中的值替换为环境中的信息。 `backup`参数使用格式为的命名空间和备份名称 `<namespace>/<name>`。例如：

```
tridentctl-protect create backupinplacerestore <my_restore_name> \
--backup <namespace/backup_to_restore> \
-n <application_namespace>
```

从备份还原到其他集群

如果原始集群出现问题、您可以将备份还原到其他集群。

- 当您使用 Kopia 作为数据移动器恢复备份时，您可以选择在 CR 中指定注释或使用 CLI 来控制 Kopia 使用的临时存储的行为。请参阅 "文档"有关您可以配置的选项的更多信息。使用 `tridentctl-protect create --help`有关使用Trident Protect CLI 指定注释的更多信息，请参阅命令。
- 使用 CR 恢复到新命名空间时，必须先手动创建目标命名空间，然后再应用 CR。Trident Protect 仅在使用 CLI 时才会自动创建命名空间。

开始之前

确保满足以下前提条件：

- 目标集群已安装Trident Protect。
- 目标集群可以访问与存储备份的源集群相同的AppVault的分段路径。
- 确保在运行 AppVault CR 时，本地环境可以连接到 AppVault CR 中定义的对象存储桶。 `tridentctl-protect get appvaultcontent`命令。如果网络限制阻止访问，请改为从目标集群上的 pod 内运行Trident Protect CLI。
- 确保AWS会话令牌到期时间足以执行任何长时间运行的还原操作。如果令牌在还原操作期间过期、则操作可能会失败。
  - 有关检查当前会话令牌到期时间的详细信息、请参见 "AWS API文档"。
  - 有关AWS资源凭据的详细信息、请参见 "AWS 文档"。

步骤

1. 使用Trident Protect CLI 插件检查目标集群上 AppVault CR 的可用性：

```
tridentctl-protect get appvault --context <destination_cluster_name>
```

确保目标集群上存在用于应用程序还原的命名空间。

2. 从目标集群查看可用AppVault的备份内容：

```
tridentctl-protect get appvaultcontent <appvault_name> \
--show-resources backup \
--show-paths \
--context <destination_cluster_name>
```

运行此命令可显示AppVault中的可用备份、包括其原始集群、相应的应用程序名称、时间戳和归档路径。

示例输出：

```
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
|   CLUSTER   |    APP    |  TYPE  |      NAME       |         TIMESTAMP
|    PATH     |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
| production1 | wordpress | backup | wordpress-bkup-1| 2024-10-30
08:37:40 (UTC)| backuppath1 |
| production1 | wordpress | backup | wordpress-bkup-2| 2024-10-30
08:37:40 (UTC)| backuppath2 |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
```

3. 使用AppVault名称和归档路径将应用程序还原到目标集群：

使用**CR**

1. 创建自定义资源(CR)文件并将其命名为 `trident-protect-backup-restore-cr.yaml`。
2. 在创建的文件中、配置以下属性：
   - 。**metadata.name**:(*required_*)此自定义资源的名称；请为您的环境选择一个唯一且合理的名称。
   - **spec.appVaultRef**：(*required*)存储备份内容的AppVault的名称。
   - **spec.appArchivePath**：AppVault中存储备份内容的路径。您可以使用以下命令查找此路径：

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
jsonpath='{.status.appArchivePath}'
```

   如果BackupRestore CR不可用、您可以使用步骤2中提到的命令查看备份内容。

   - 。**spec.namespaceMapping**:还原操作的源命名空间到目标命名空间的映射。将和 `my-destination-namespace`替换 `my-source-namespace`为您环境中的信息。

     例如：

```
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: my-cr-name
  namespace: my-destination-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-backup-path
  namespaceMapping: [{"source": "my-source-namespace", "
destination": "my-destination-namespace"}]
```

3. 使用正确的值填充文件后 `trident-protect-backup-restore-cr.yaml` 、应用CR：

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

使用命令行界面

1. 使用以下命令还原应用程序、将括号中的值替换为环境中的信息。命名空间映射参数使用冒号分隔的命名空间将源命名空间映射到正确的目标命名空间、格式为SOURCE1：dest1、Source2：dest2。例如：

```
tridentctl-protect create backuprestore <restore_name> \
--namespace-mapping <source_to_destination_namespace_mapping> \
--appvault <appvault_name> \
--path <backup_path> \
--context <destination_cluster_name> \
-n <application_namespace>
```

从快照还原到其他命名空间

您可以使用自定义资源 (CR) 文件从快照恢复数据，恢复到不同的命名空间或原始源命名空间。当您使用 SnapshotRestore CR 将快照还原到不同的命名空间时， Trident Protect 会在新的命名空间中还原应用程序，并为还原的应用程序创建一个应用程序 CR。为了保护已恢复的应用程序，可以创建按需备份或快照，或者制定保护计划。

- SnapshotRestore 支持 `spec.storageClassMapping`属性，但仅当源和目标存储类使用相同的存储后端时。如果您尝试恢复到 `StorageClass`如果使用不同的存储后端，则恢复操作将失败。
- 使用 CR 恢复到新命名空间时，必须先手动创建目标命名空间，然后再应用 CR。Trident Protect 仅在使用 CLI 时才会自动创建命名空间。

开始之前

确保AWS会话令牌到期时间足以执行任何长时间运行的S3还原操作。如果令牌在还原操作期间过期、则操作可能会失败。

- 有关检查当前会话令牌到期时间的详细信息、请参见 "AWS API文档"。
- 有关AWS资源凭据的详细信息、请参见 "AWS IAM文档"。

使用CR

步骤

1. 创建自定义资源(CR)文件并将其命名为 `trident-protect-snapshot-restore-cr.yaml`。
2. 在创建的文件中、配置以下属性：
   - 。**metadata.name**:(*required_*)此自定义资源的名称；请为您的环境选择一个唯一且合理的名称。
   - **spec.appVaultRef**：(*required*)存储快照内容的AppVault的名称。
   - **spec.appArchivePath**：AppVault中存储快照内容的路径。您可以使用以下命令查找此路径：

     ```
     kubectl get snapshots <SNAPHOT_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - 。**spec.namespaceMapping**:还原操作的源命名空间到目标命名空间的映射。将和 `my-destination-namespace`替换 `my-source-namespace`为您环境中的信息。

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: my-cr-name
       namespace: my-app-namespace
     spec:
       appVaultRef: appvault-name
       appArchivePath: my-snapshot-path
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (可 选)如果只需要选择要还原的应用程序的某些资源、请添加包含或排除带有特定标签的资源的筛选：

   Trident Protect 会自动选择一些资源，因为它们与您选择的资源存在关联。例如，如果您选择持久卷声明资源并且它有一个关联的 pod， Trident Protect 还会恢复关联的 pod。

   - **resourceFilter.resourceSourcedionCritera**：(筛选时需要)使用 `Include`或包含或 `Exclude`排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：
     - **resourceFilter.resourceMatcher**：resourceMatcher对象数组。如果在此数组中定义多个元素，它们将作为OR操作进行匹配，每个元素(组、种类、版本)中的字段将作为AND操作进行匹配。
       - **resourceMatcher[].group**：(可 选)要筛选的资源的组。
       - **resourceMatcher[].KIND**：(可 选)要筛选的资源种类。
       - **resourceMatcher[].version**：(可 选)要筛选的资源版本。
       - **resourceMatcher[].names**：(可 选)要筛选的资源的Kubernetes metadata.name字段中的

名称。

- **resourceMatcher[].namespies**：(可 选)要筛选的资源的Kubernetes metadata.name字段中的命名空间。
- *resourceMatcher[].labelSelectors *：(可 选)资源的Kubernetes metadata.name字段中的标签选择器字符串，如中所定义 "Kubernetes 文档"。例如：`"trident.netapp.io/os=linux"`。

  例如：

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. 使用正确的值填充文件后 `trident-protect-snapshot-restore-cr.yaml` 、应用CR：

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

使用命令行界面

步骤

1. 将快照还原到其他命名空间、将括号中的值替换为环境中的信息。

   - `snapshot`参数使用格式为的命名空间和快照名称 `<namespace>/<name>`。
   - 此 `namespace-mapping`参数使用冒号分隔的卷来将源卷的源卷映射到格式为的正确目标卷的`source1:dest1,source2:dest2`卷。

     例如：

```
tridentctl-protect create snapshotrestore <my_restore_name> \
--snapshot <namespace/snapshot_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
-n <application_namespace>
```

从快照还原到原始命名空间

您可以随时将快照还原到原始命名空间。

开始之前

确保AWS会话令牌到期时间足以执行任何长时间运行的S3还原操作。如果令牌在还原操作期间过期、则操作可能会失败。

- 有关检查当前会话令牌到期时间的详细信息、请参见 "AWS API文档"。
- 有关AWS资源凭据的详细信息、请参见 "AWS IAM文档"。

使用CR

步骤

1. 创建自定义资源(CR)文件并将其命名为 `trident-protect-snapshot-ipr-cr.yaml`。
2. 在创建的文件中、配置以下属性：
   - 。**metadata.name**:(*required*_)此自定义资源的名称；请为您的环境选择一个唯一且合理的名称。
   - **spec.appVaultRef**：(*required*)存储快照内容的AppVault的名称。
   - **spec.appArchivePath**：AppVault中存储快照内容的路径。您可以使用以下命令查找此路径：

```
kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o
jsonpath='{.status.appArchivePath}'
```

```
---
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotInplaceRestore
metadata:
  name: my-cr-name
  namespace: my-app-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-snapshot-path
```

3. (可 选)如果只需要选择要还原的应用程序的某些资源、请添加包含或排除带有特定标签的资源的筛选：

> Trident Protect 会自动选择一些资源，因为它们与您选择的资源存在关联。例如，如果您选择持久卷声明资源并且它有一个关联的 pod， Trident Protect 还会恢复关联的 pod。

   - **resourceFilter.resourceSourcedionCritera**：(筛选时需要)使用 `Include`或包含或 `Exclude`排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：
     - **resourceFilter.resourceMatcher**：resourceMatcher对象数组。如果在此数组中定义多个元素，它们将作为OR操作进行匹配，每个元素(组、种类、版本)中的字段将作为AND操作进行匹配。
       - **resourceMatcher[].group**：(可 选)要筛选的资源的组。
       - **resourceMatcher[].KIND**：(可 选)要筛选的资源种类。
       - **resourceMatcher[].version**：(可 选)要筛选的资源版本。
       - **resourceMatcher[].names**：(可 选)要筛选的资源的Kubernetes metadata.name字段中的名称。
       - **resourceMatcher[].namespies**：(可 选)要筛选的资源的Kubernetes metadata.name字段中的命名空间。
       - *resourceMatcher[].labelSelectors *：(可 选)资源的Kubernetes metadata.name字段中的标

签选择器字符串，如中所定义 "Kubernetes 文档"。例如：`"trident.netapp.io/os=linux"`。

例如：

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. 使用正确的值填充文件后 `trident-protect-snapshot-ipr-cr.yaml` 、应用CR：

```
kubectl apply -f trident-protect-snapshot-ipr-cr.yaml
```

使用命令行界面

步骤

1. 将快照还原到原始命名空间、将括号中的值替换为环境中的信息。例如：

```
tridentctl-protect create snapshotinplacerestore <my_restore_name> \
--snapshot <namespace/snapshot_to_restore> \
-n <application_namespace>
```

检查还原操作的状态

您可以使用命令行检查正在进行、已完成或失败的还原操作的状态。

步骤

1. 使用以下命令检索还原操作的状态、将括号中的值替换为环境中的信息：

```
kubectl get backuprestore -n <namespace_name> <my_restore_cr_name> -o
jsonpath='{.status}'
```

使用高级Trident Protect 恢复设置

您可以使用高级设置（例如注释、命名空间设置和存储选项）自定义恢复操作，以满足您的特定要求。

还原和故障转移操作期间的命名空间标注和标签

在还原和故障转移操作期间、目标命名空间中的标签和标注会与源命名空间中的标签和标注相匹配。此时将添加源命名空间中目标命名空间中不存在的标签或标注、并覆盖已存在的任何标签或标注、以便与源命名空间中的值匹配。仅存在于目标命名空间上的标签或标注保持不变。

如果您使用 Red Hat OpenShift，请务必注意命名空间注释在 OpenShift 环境中的重要作用。命名空间注释确保恢复的 pod 遵守 OpenShift 安全上下文约束 (SCC) 定义的适当权限和安全配置，并且可以访问卷而不会出现权限问题。欲了解更多信息，请参阅"OpenShift安全上下文约束文档"。

在执行还原或故障转移操作之前、您可以通过设置Kubornetes环境变量来防止目标命名空间中的特定标注被覆盖 `RESTORE_SKIP_NAMESPACE_ANNOTATIONS`。例如：

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect \
  --set-string
restoreSkipNamespaceAnnotations="{<annotation_key_to_skip_1>,<annotation_k
ey_to_skip_2>}" \
  --reuse-values
```

执行恢复或故障转移操作时，任何命名空间注释和标签都将生效。`restoreSkipNamespaceAnnotations` 和 `restoreSkipNamespaceLabels` 不参与恢复或故障转移操作。确保在初始 Helm 安装期间配置这些设置。欲了解更多信息，请参阅 "配置其他Trident Protect 舵图设置"。

如果您使用 Helm 安装了源应用程序， `--create-namespace`国旗，给予特殊待遇 `name`标签键。在恢复或故障转移过程中， Trident Protect 会将此标签复制到目标命名空间，但如果源命名空间的值与源命名空间的值匹配，则会将值更新为目标命名空间的值。如果此值与源命名空间不匹配，则会将其复制到目标命名空间，而不做任何更改。

示例

以下示例显示了一个源和目标命名空间、每个命名空间都具有不同的标注和标签。您可以查看目标命名空间在操作前后的状态、以及标注和标签在目标命名空间中的组合或覆盖方式。

在执行还原或故障转移操作之前

下表说明了执行还原或故障转移操作之前示例源和目标名称卷的状态：

命名空间	标注	标签
命名空间ns-1 (源)	• 标注.One/键："updatedvalue" • 标注。双/键："TRUE"	• 环境=生产 • 合规性=HIPAA • name=ns-1
命名空间ns-2 (目标)	• 标注.One/键："TRUE" • 标注三个/项："false"	• Role=database

还原操作之后

下表显示了还原或故障转移操作后示例目标命名空间的状态。已添加某些密钥、某些密钥已被覆盖、并且 `name`标签已更新以与目标命名空间匹配：

命名空间	标注	标签
命名空间ns-2 (目标)	• 标注.One/键："updatedvalue" • 标注。双/键："TRUE" • 标注三个/项："false"	• name=nS-2 • 合规性=HIPAA • 环境=生产 • Role=database

支持的字段

本节介绍可用于恢复操作的其他字段。

存储类别映射

这 `spec.storageClassMapping`属性定义从源应用程序中的现有存储类到目标集群上的新存储类的映射。您可以在具有不同存储类别的集群之间迁移应用程序时或更改 BackupRestore 操作的存储后端时使用此功能。

- 示例： *

```
storageClassMapping:
  - destination: "destinationStorageClass1"
    source: "sourceStorageClass1"
  - destination: "destinationStorageClass2"
    source: "sourceStorageClass2"
```

支持的注释

本节列出了系统中支持配置各种行为的注解。如果用户未明确设置注解，系统将使用默认值。

标注	Type	Description	默认值
protected.trident.netapp.io/data-mover-timeout-sec	string	允许数据移动设备操作停止的最长时间（以秒为单位）。	“300”
protected.trident.netapp.io/kopia-content-cache-size-limit-mb	string	Kopia 内容缓存的最大大小限制（以兆字节为单位）。	“1000”
protect.trident.netapp.io/pvc-bind-timeout-sec	string	等待新创建的持久卷声明 (PVC) 到达的最大时间（以秒为单位）`Bound`操作失败前的阶段。适用于所有恢复 CR 类型（备份恢复、备份就地恢复、快照恢复、快照就地恢复）。如果您的存储后端或集群经常需要更多时间，请使用更高的值。	1200（20分钟）