Gestione Trident Protect

Trident

NetApp
September 26, 2025

Tabla de contenidos

Gestione Trident Protect

Gestione la autorización y el control de acceso de Trident Protect

Trident Protect utiliza el modelo de Kubernetes de control de acceso basado en roles (RBAC). De forma predeterminada, Trident Protect proporciona un único espacio de nombres del sistema y su cuenta de servicio predeterminada asociada. Si cuenta con una organización con muchos usuarios o con necesidades de seguridad específicas, puede utilizar las funciones de control de acceso basado en roles de Trident Protect para obtener un control más granular sobre el acceso a los recursos y los espacios de nombres.

El administrador de clúster siempre tiene acceso a los recursos del espacio de nombres predeterminado `trident-protect` y también puede acceder a los recursos en el resto de espacios de nombres. Para controlar el acceso a recursos y aplicaciones, es necesario crear espacios de nombres adicionales y agregar recursos y aplicaciones a esos espacios de nombres.

Tenga en cuenta que ningún usuario puede crear CRS de gestión de datos de aplicaciones en el espacio de nombres predeterminado `trident-protect`. Debe crear CRS de gestión de datos de aplicaciones en un espacio de nombres de aplicaciones (como práctica recomendada, crear CRS de gestión de datos de aplicaciones en el mismo espacio de nombres que la aplicación asociada).

> Sólo los administradores deben tener acceso a los objetos de recursos personalizados Privileged Trident Protect, que incluyen:
>
> - **AppVault**: Requiere datos de credenciales de bucket
> - **Paquete de Protección**: Recopila métricas, registros y otros datos sensibles de Trident
> - **BundleSchedule**: Gestiona los horarios de recopilación de registros
>
> Como práctica recomendada, use RBAC para restringir el acceso a los objetos con privilegios a los administradores.

Para obtener más información sobre cómo el RBAC regula el acceso a los recursos y espacios de nombres, consulte la "Documentación de RBAC de Kubernetes".

Para obtener información sobre las cuentas de servicio, consulte la "Documentación de la cuenta de servicio de Kubernetes".

Ejemplo: Administrar el acceso para dos grupos de usuarios

Por ejemplo, una organización tiene un administrador de clústeres, un grupo de usuarios de ingeniería y un grupo de usuarios de marketing. El administrador del clúster debe realizar las siguientes tareas para crear un entorno en el que el grupo de ingeniería y el grupo de marketing tengan acceso solo a los recursos asignados a sus respectivos espacios de nombres.

Paso 1: Crear un espacio de nombres para contener recursos para cada grupo

La creación de un espacio de nombres permite separar los recursos de forma lógica y controlar mejor quién

tiene acceso a dichos recursos.

Pasos

1. Cree un espacio de nombres para el grupo de ingeniería:

```
kubectl create ns engineering-ns
```

2. Cree un espacio de nombres para el grupo de marketing:

```
kubectl create ns marketing-ns
```

Paso 2: Crear nuevas cuentas de servicio para interactuar con los recursos de cada espacio de nombres

Cada nuevo espacio de nombres que cree viene con una cuenta de servicio predeterminada, pero debe crear una cuenta de servicio para cada grupo de usuarios para que pueda dividir aún más Privileges entre grupos en el futuro si es necesario.

Pasos

1. Cree una cuenta de servicio para el grupo de ingeniería:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eng-user
  namespace: engineering-ns
```

2. Cree una cuenta de servicio para el grupo de marketing:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mkt-user
  namespace: marketing-ns
```

Paso 3: Crear un secreto para cada nueva cuenta de servicio

Un secreto de cuenta de servicio se utiliza para autenticarse con la cuenta de servicio, y se puede eliminar y volver a crear fácilmente si está comprometido.

Pasos

1. Cree un secreto para la cuenta de servicio de ingeniería:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: eng-user
  name: eng-user-secret
  namespace: engineering-ns
type: kubernetes.io/service-account-token
```

2. Cree un secreto para la cuenta de servicio de marketing:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: mkt-user
  name: mkt-user-secret
  namespace: marketing-ns
type: kubernetes.io/service-account-token
```

Paso 4: Cree un objeto RoleBinding para enlazar el objeto ClusterRole a cada nueva cuenta de servicio

Al instalar Trident Protect, se crea un objeto ClusterRole predeterminado. Puede enlazar este ClusterRole a la cuenta de servicio creando y aplicando un objeto RoleBinding.

Pasos

1. Enlazar ClusterRole a la cuenta de servicio de ingeniería:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: engineering-ns-tenant-rolebinding
  namespace: engineering-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

2. Enlazar ClusterRole a la cuenta de servicio de marketing:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: marketing-ns-tenant-rolebinding
  namespace: marketing-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: mkt-user
  namespace: marketing-ns
```

Paso 5: Probar permisos

Compruebe que los permisos son correctos.

Pasos

1. Confirme que los usuarios de ingeniería pueden acceder a los recursos de ingeniería:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n engineering-ns
```

2. Confirme que los usuarios de ingeniería no pueden acceder a los recursos de marketing:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n marketing-ns
```

Paso 6: Otorgar acceso a los objetos de AppVault

Para realizar tareas de gestión de datos, como backups e instantáneas, el administrador de clúster debe conceder acceso a los objetos de AppVault a usuarios individuales.

Pasos

1. Cree y aplique un archivo YAML de combinación secreta y AppVault que otorgue acceso a un usuario a un AppVault. Por ejemplo, el siguiente CR otorga acceso a un AppVault al usuario `eng-user`:

```
apiVersion: v1
data:
  accessKeyID: <ID_value>
  secretAccessKey: <key_value>
kind: Secret
metadata:
  name: appvault-for-eng-user-only-secret
  namespace: trident-protect
type: Opaque
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: appvault-for-eng-user-only
  namespace: trident-protect # Trident protect system namespace
spec:
  providerConfig:
    azure:
      accountName: ""
      bucketName: ""
      endpoint: ""
    gcp:
      bucketName: ""
      projectID: ""
    s3:
      bucketName: testbucket
      endpoint: 192.168.0.1:30000
      secure: "false"
      skipCertValidation: "true"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: appvault-for-eng-user-only-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: appvault-for-eng-user-only-secret
  providerType: GenericS3
```

2. Cree y aplique un CR de rol para permitir que los administradores del cluster concedan acceso a recursos específicos en un espacio de nombres. Por ejemplo:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: eng-user-appvault-reader
  namespace: trident-protect
rules:
- apiGroups:
  - protect.trident.netapp.io
  resourceNames:
  - appvault-for-enguser-only
  resources:
  - appvaults
  verbs:
  - get
```

3. Cree y aplique un CR de RoleBinding para enlazar los permisos al usuario eng-user. Por ejemplo:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eng-user-read-appvault-binding
  namespace: trident-protect
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: eng-user-appvault-reader
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

4. Compruebe que los permisos son correctos.

   a. Se ha intentado recuperar la información del objeto AppVault para todos los espacios de nombres:

   ```
   kubectl get appvaults -n trident-protect
   --as=system:serviceaccount:engineering-ns:eng-user
   ```

   Debería ver una salida similar a la siguiente:

```
Error from server (Forbidden): appvaults.protect.trident.netapp.io is
forbidden: User "system:serviceaccount:engineering-ns:eng-user"
cannot list resource "appvaults" in API group
"protect.trident.netapp.io" in the namespace "trident-protect"
```

b. Prueba para ver si el usuario puede obtener la información de AppVault a la que ahora tiene permiso para acceder:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get appvaults.protect.trident.netapp.io/appvault-for-eng-user-only -n
trident-protect
```

Debería ver una salida similar a la siguiente:

```
yes
```

Resultado

Los usuarios a los que ha otorgado permisos de AppVault deben poder usar objetos de AppVault autorizados para operaciones de gestión de datos de aplicaciones y no deben poder acceder a ningún recurso fuera de los espacios de nombres asignados ni crear nuevos recursos a los que no tengan acceso.

Generar un bundle de soporte de Trident Protect

Trident Protect permite a los administradores generar paquetes que incluyen información útil para el soporte de NetApp, incluidos registros, métricas e información de topología sobre los clústeres y las aplicaciones que se están gestionando. Si está conectado a Internet, puede cargar paquetes de soporte en el sitio de soporte de NetApp (NSS) mediante un archivo de recursos personalizados (CR).

Cree un paquete de soporte mediante un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-support-bundle.yaml`).
2. Configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.triggerType**: (*required*) Determina si el paquete de soporte se genera inmediatamente o se programa. La generación de paquetes programada se tiene lugar a LAS 12am UTC. Los posibles valores son los siguientes:
     - Programado
     - Manual
   - **SPEC.uploadEnabled**: (*Opcional*) Controla si el paquete de soporte debe cargarse en el sitio de soporte de NetApp después de que se genere. Si no se especifica, el valor por defecto es `false`. Los posibles valores son los siguientes:
     - verdadero
     - false (predeterminado)
   - **Spec.dataWindowStart**: (*Optional*) Una cadena de fecha en formato RFC 3339 que especifica la fecha y la hora en que debe comenzar la ventana de datos incluidos en el paquete de soporte. Si no se especifica, el valor predeterminado es hace 24 horas. La fecha de ventana más antigua que puede especificar es hace 7 días.

     Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AutoSupportBundle
metadata:
  name: trident-protect-support-bundle
spec:
  triggerType: Manual
  uploadEnabled: true
  dataWindowStart: 2024-05-05T12:30:00Z
```

3. Después de rellenar `astra-support-bundle.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-support-bundle.yaml
```

Cree un bundle de soporte mediante la CLI

Pasos

1. Cree el paquete de soporte, reemplazando valores entre paréntesis con la información del entorno.

`trigger-type`Determina si el grupo se crea inmediatamente o si la hora de creación está determinada por la programación, y puede ser `Manual` o. `Scheduled` El valor predeterminado es `Manual`.

Por ejemplo:

```
tridentctl-protect create autosupportbundle <my_bundle_name>
--trigger-type <trigger_type>
```

Actualice Trident Protect

Puede actualizar Trident Protect a la última versión para aprovechar las nuevas funciones o correcciones de errores.

Para actualizar Trident Protect, realice los siguientes pasos.

Pasos

1. Actualice el repositorio de Trident Helm:

```
helm repo update
```

2. Actualice los CRD de Trident Protect:

```
helm upgrade trident-protect-crds netapp-trident-protect/trident-
protect-crds --version 100.2410.1  --namespace trident-protect
```

3. Actualizar Trident Protect:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect
--version 100.2410.1 --namespace trident-protect
```