etcd

概览 **etcd** 实践 **etcd** 加密

概览

etcd 存储集群的 Kubernetes 控制平面状态。平台管理员使用 etcd 指南来了解加密、运行状况、容量考量，以及如何交接到备份和恢复操作步骤。

需求	继续阅读
启用并了解 etcd 加密密钥轮换。	etcd encryption
查看运行实践，并转到运行状况、容量和备份主题。	etcd practices
备份或恢复 etcd 数据。	etcd backup and restore

使用这些 etcd 页面来规划控制平面状态的加密、运行状况、容量和维护。当您需要保护或恢复 etcd 数据时，请继续阅读备份和恢复操作步骤。

etcd 实践

使用 etcd 实践，在执行会影响集群恢复或可用性的更改之前，规划控制平面状态操作。

实践领域	指导
高风险维护前备份	在进行控制平面维护、节点操作系统更改或其他可能影响集群状态的操作之前，先备份 etcd。请遵循 etcd 备份和恢复。
加密	当敏感的 Kubernetes 资源需要通过文档中的加密管理器在静态存储时进行加密时，请使用 etcd 加密。
容量和健康状况	通过你环境中可用的可观测性和集群操作工具，监控 etcd 健康状况、磁盘延迟和可用容量。
恢复范围	将 etcd 备份和恢复用于控制平面状态恢复。Registry 数据、监控数据、日志数据和应用数据需要各自独立的备份和恢复路径。

不要将此页面用作生产加固检查清单。请遵循特定主题的支持指南，以获取经过验证的恢复流程、规模限制和维护窗口。

etcd 加密

在 ACP 中安装并运行 etcd Encryption Manager，以自动完成集群内 etcd 数据加密密钥轮换。

它可确保存储在 etcd 中的敏感数据（例如 secrets 和 configmaps）使用安全算法加密，从而提升集群安全性。

目录

密钥策略

该插件支持两种密钥策略。请选择与集群角色匹配的策略。

策略	适用场景	密钥生成方式
`random`	默认适用于独立运行（无 etcd 同步）的 workload 和 DCS 集群。	每次轮换都会在本地生成一个新的随机密钥。
`deterministic`	必须用于 `global` 集群，尤其是在通过 etcd Synchronizer 进行 DR 配对时。	每个密钥都从共享的 root Secret 和共享的 SeedBundle 确定性派生，因此 Active 和 Standby 集群在相同修订版本下始终得到相同的密钥。

在 `deterministic` 模式下，会自动检测运行时 Active / Standby 角色；这不是手动安装参数。

安装

该插件根据目标集群支持两种安装路径：

- **workload / DCS** 集群 — 使用默认的 `random` 密钥策略。
- **global** 集群 — 必须使用 `deterministic` 密钥策略（请参见下方的 Global 集群（Deterministic 模式））。

有关通用插件安装流程，请参见 集群插件。

workload / DCS 集群（random 策略）

- 支持的集群类型：本地部署、DCS。
- 无需额外安装参数——通过标准集群插件工作流安装即可。

Global 集群（Deterministic 模式）

DR 配对要求

对于全局 DR 对，必须在 **Active** 和 **Standby** 两个集群 上以 `deterministic` 模式安装该插件，并在两侧配置 完全相同 的 `replication_group_id` 和 `root_secret_name`（以及匹配的根密钥材料）。如果配置不对称或不匹配，Standby 将派生出与 Active 不同的按修订版本密钥，并导致故障切换失败。

前提条件

- 集群必须是 `global` 集群。当 `global.cluster.name` 为 `global` 时，chart 会强制设置 `etcdEncryption.keyStrategy=deterministic`；对于标记为 `is-global=true` 的集群，插件 UI 也会将该字段默认设为 `deterministic`。

etcd Synchronizer 版本

在以 `deterministic` 模式安装 etcd Encryption Manager 之前，必须先安装版本为 **v4.3.7** 或更高的 **etcd Synchronizer** 插件。更早版本不会复制共享的 SeedBundle，因此 Standby 集群无法派生匹配的密钥。

插件参数

插件表单会为 deterministic 安装暴露以下参数。默认值反映了 chart 当前随附的 `plugin-config.yaml`。

插件参数	是否必需	默认值	用途
`key_strategy`	是	`random`；对于 `global` 集群会动态默认为 `deterministic`。	在每次轮换生成随机密钥与确定性派生之间进行选择。
`replication_group_id`	是	插件 UI 会预填当前集群名称（不建议这样做——见下文）。	标识 Active / Standby DR 组，并参与 HKDF 域分离。

插件参数	是否必需	默认值	用途
`root_secret_name`	是	`etcd-derivation-root`	kube-system 中保存确定性根密钥材料的 Secret 名称。

`rootSecretRef.namespace` 固定为 `kube-system`，不会在表单中暴露。

`key_strategy`

值	含义
`random`	单集群随机轮换。
`deterministic`	基于共享 root Secret 和 SeedBundle 派生密钥。

对于 `global` 集群，这里必须设置为 `deterministic`。

`replication_group_id`

属性	值
用途	标识一个 Active / Standby DR 复制组，并参与 HKDF 域分离。
一致性要求	同一 DR 组中的 Active 和 Standby 集群 必须 使用完全相同的值。
UI 中的默认值	当前集群名称。
是否建议保留默认值	否。

建议：

- 对于使用 etcd Synchronizer 构建的 DR 集群，同一 DR 组的两侧必须共享相同的 `replication_group_id`。
- 为避免泄露真实业务、环境或集群标识，即使 UI 已预填集群名称，也不要重复使用该名称。请在安装前规划一个单独的不透明标识符。
- 类似 UUID 的标识符可能很合适，例如 `6f1b9e2c-7c3a-4a9c-8b72-2fd0f8f3c1ab`。

要从命令行生成类似 UUID 的值：

```
openssl rand -hex 16 | sed -E 's/^(.{8})(.{4})(.{4})(.{4})(.{12})$/\1-\2-\3-\4-\5/'
```

> **在两个集群上使用相同的值**
>
> 只生成一次该值，然后在 DR 组的 Active 和 Standby 集群上配置相同的 `replication_group_id`。在两侧生成不同的值会产生不同的加密密钥。

`root_secret_name`

属性	值
用途	指向确定性根密钥 Secret。
默认值	`etcd-derivation-root`
固定命名空间	`kube-system`
跨集群要求	同一复制组中的所有集群必须使用相同的根密钥材料。

插件表单仅暴露 Secret 名称；在渲染时命名空间始终为 `kube-system`。

准备根 Secret

在安装插件之前，确定性根 Secret 必须已存在于 `kube-system` 中。请按两个步骤准备：

1. 将 32 字节的密码学强随机材料生成到 `./root-key.bin`。这些字节必须来自高熵源——通常是操作系统的 CSPRNG。`openssl rand` 是安全的默认选择；基于等效操作系统 CSPRNG 的命令，例如 `head -c 32 /dev/urandom > ./root-key.bin`，也可以使用。不要使用 shell 的 `$RANDOM`、语言级别的 `rand()`，或任何基于时间种子的生成器。

   ```
   openssl rand -out ./root-key.bin 32
   ```

2. 使用该文件在 `kube-system` 中创建 Secret：

```
kubectl -n kube-system create secret generic etcd-derivation-root \
  --from-file=root-key=./root-key.bin
```

如果您使用不同的 Secret 名称，请将上面的 `etcd-derivation-root` 替换为您将作为 `root_secret_name` 输入的值。

> 在两个集群上使用相同的密钥材料
>
> 只生成一次 `./root-key.bin`，然后将 同一个文件 传输到复制组中的每个集群，并在每个集群上执行 `kubectl create secret` 步骤。不要为每个集群重新生成密钥材料——不同的根密钥字节会产生不同的加密密钥。

高级（**chart** 值，未在插件表单中暴露）

▶ 显示高级 chart 值

工作原理

安装后，会在 `kube-system` 命名空间部署 `etcd-encryption-manager` 控制器，它会：

- 定期轮换 etcd 数据加密密钥。
- 保留最近 8 个密钥以保证回滚兼容性。
- 更新所有控制节点上的加密配置。
- 触发 kube-apiserver 热加载新密钥。
- 自动迁移资源，使用新密钥重新加密数据。

在这些操作过程中，集群稳定性始终得以保持。

当插件在 `global` DR 对上以 `deterministic` 模式安装时，控制器还会：

- 在运行时自动检测自身是 **Active** 还是 **Standby**——这不是手动配置。
- 在 Active 端生成 SeedBundle，并让 etcd Synchronizer 将其复制到 Standby。

- 在 Standby 端，基于共享的 SeedBundle 和 root Secret 派生相同的按修订版本密钥，因此故障切换无需手动复制密钥即可获得一致的加密密钥。

默认配置

参数	值
已加密资源	secrets, configmaps
加密算法	256 位 AES-GCM
轮换间隔	168 小时（7 天）

运维指南

配置文件

路径	内容
`/etc/kubernetes/encryption-provider.conf`	当前加密配置
`/etc/kubernetes/encryption-provider-history.bak`	历史密钥记录（用于恢复）
`/etc/kubernetes/encryption-provider-bak/`	已过期的加密配置版本

检查状态

运行以下命令检查当前轮换状态：

```
kubectl get EtcdEncryptionConfig default -o yaml
```

示例输出：

```yaml
apiVersion: cluster.alauda.io/v1alpha1
kind: EtcdEncryptionConfig
metadata:
  name: default
spec:
  resources:
    - secrets
    - configmaps
  rotationInterval: 168h0m0s
  type: aesgcm
status:
  deployStatus:
    192.168.100.1:
      revision: 3
      state: Success
    192.168.100.2:
      revision: 3
      state: Success
    192.168.100.3:
      revision: 3
      state: Success
  migration:
    completeTimestamp: "2025-05-27T05:47:01Z"
    resources:
      - secrets
      - configmaps
    revision: 3
    state: Success
  revision: 3
```