Options to manage secret

We want to:

- Store credentials (DB password, API keys, tokens, etc.) securely.
- Maintain **different values** for `dev`, `stage`, and `prod`.
- Control **access and rotation** cleanly.
- Avoid storing plain-text secrets in Git or in Kubernetes manifests.

Options for Implementing Confidentiality in Kubernetes Secrets

#	Approach	Description	Pros	Cons
1	**Native Kubernetes Secrets**	Store secrets directly in Kubernetes using `Secret` objects (`kubectl create secret`).	Simple, native, integrated with RBAC.	Base64 encoded, *not encrypted* by default (only obfuscated). Secrets stored in `etcd` in plain text unless encryption is enabled.
2	**Kubernetes Secrets with Encryption at Rest (EKS/GKE/AKS)**	Enable secret encryption in the API server configuration using a KMS provider (like AWS KMS, Azure Key Vault, or GCP KMS).	Secrets in `etcd` are encrypted with a KMS key.	Still requires secure KMS key management; doesn't solve secret rotation or versioning.
3	**External Secrets Operator (ESO)**	Open-source operator that syncs secrets from external stores (e.g. AWS Secrets Manager, HashiCorp Vault, GCP Secret Manager) into Kubernetes Secrets automatically.	Supports multiple secret backends, can use GitOps-friendly configs, environment-specific secrets possible.	Adds complexity, external dependency on secret stores, requires IAM configuration.
4	**Sealed Secrets (Bitnami)**	Secrets are encrypted into YAML files (`SealedSecret`) that can safely be stored in Git. Controller decrypts them inside the cluster.	GitOps-friendly, simple integration.	Key rotation is tricky; if sealing key is lost, secrets cannot be decrypted.
	SOPS	Encrypt secret files in Git using SOPS and decrypt	Full control, integrates with	Requires CI/CD integration; not

5	**(Mozilla)**with KMS or PGP	them during deployment via CI/CD.	GitOps (Flux/Argo).	ideal for dynamic rotation.
6	**AWS Secrets Manager / Parameter Store Integration**	Applications fetch secrets from AWS Secrets Manager or SSM Parameter Store via SDK or sidecar/Init container.	Centralized secrets management, versioning, rotation supported.	Application must handle AWS API calls or sidecar configuration; not platform-agnostic.
7	**HashiCorp Vault Integration with Kubernetes (Recommended)**	Vault securely stores all environment secrets. Applications or pods authenticate via Kubernetes Service Accounts and pull secrets dynamically.	Enterprise-grade security, automatic secret rotation, policy-based access, audit logging, multi-environment segregation.	Setup complexity (Vault server, auth method, agent sidecar), initial learning curve.

Recommended Option: HashiCorp Vault with Kubernetes Integration

Vault is the **industry-standard** for secret management when you care about **confidentiality, access control, rotation, and auditability**.

Architecture

1. Vault runs as a service (self-hosted or managed).
2. Kubernetes **Service Accounts** authenticate to Vault using the **Kubernetes Auth Method**.
3. Pods request secrets from Vault dynamically (via:
    - Vault Agent Injector (sidecar),
    - Vault CSI Driver,
    - or API calls in app).
4. Vault enforces policies to allow different secrets for each namespace/environment:
    - `/secret/dev/db-creds`
    - `/secret/stage/db-creds`
    - `/secret/prod/db-creds`

Advantages

Feature	Explanation
Dynamic secrets	Vault can generate credentials on-demand (e.g., dynamic DB users with TTL).
Fine-grained access	Access controlled via Vault Policies; Dev team can’t read Prod secrets.
Secret rotation	Automatic rotation supported for DBs, AWS, etc.
Audit logs	Every secret read/write is logged for compliance.
Encryption as a service	You can encrypt/decrypt data without storing it.
Multi-environment isolation	Each environment (dev/stage/prod) can have separate paths and policies.

Disadvantages

Limitation	Description
Setup complexity	Requires managing Vault cluster and integration.
Performance overhead	Vault must be highly available (use integrated storage or HA mode).
Learning curve	Teams need to understand Vault policies, auth methods, and agent sidecar setup.
