# HashiCorp Vault (KMIP Secrets Engine) Setup Guide for CBOM KMIP Sensor

---

## 1. Purpose

This guide explains how to configure **HashiCorp Vault Enterprise's KMIP Secrets Engine** (part of the Advanced Data Protection add on) as a read only data source for the CBOM `Discover_KMIP` sensor. The sensor connects to Vault's KMIP listener over the KMIP 1.4 protocol (TLS, default port `5696`) and discovers cryptographic key metadata managed within a dedicated Vault KMIP **scope**.

> **Note:** This guide covers the KMIP Secrets Engine only. Certificate issuance through Vault's **PKI engine**, and general secret storage through the **Transit** and **KV** engines, are covered by the separate `Discover_PKI` sensor and its own setup guide. Do not confuse the two, they are distinct Vault Enterprise features discovered by different CBOM sensors.

The following cryptographic material is discovered:

- Symmetric and asymmetric key identifiers (KMIP Unique Identifiers)
- Key algorithms (e.g., AES, RSA, EC) and associated key lengths / curve parameters
- Key state (Pre Active, Active, Deactivated, Destroyed, Compromised)
- Key names and Vault managed custom attributes
- The Vault **scope** and **role** the key belongs to

The sensor operates in a strictly read only mode. It invokes only the KMIP `Locate` and `Get Attributes` operations, it never invokes `Get`, which would return raw key material.

---

## 2. Prerequisites

Before beginning this setup, ensure the following conditions are met:

1. You are running **Vault Enterprise** with the **Advanced Data Protection (ADP) KMIP** module licensed and enabled. The KMIP secrets engine is not available in Vault Community Edition.
2. You have `vault` CLI access with a token authorized to enable secrets engines and manage the `kmip/` mount (typically an `admin` or equivalent policy).
3. Vault's KMIP listener is reachable on TCP port `5696` (or your configured port) from the host running the CBOM sensor.
4. OpenSSL is available on the CBOM sensor host to verify certificate files after generation.
5. The CBOM sensor binary (`cbom-sensor`) version 1.4.0 or later is installed on the sensor host.
6. Outbound TLS connections from the sensor host to the Vault KMIP listener are permitted by any intermediate firewalls or security groups.
7. You have noted the Vault server's hostname or IP address that will be used in the sensor configuration.

## 3. Enable the KMIP Secrets Engine and Create a Scope

Vault's KMIP engine organizes keys into **scopes** (isolated KMIP tenants) and **roles** (permission sets within a scope). A dedicated scope keeps the CBOM sensor's visibility separate from any scopes used by production encryption workflows.

### 3.1 Enable the KMIP Secrets Engine

1. Enable the engine at its default mount path:

```
vault secrets enable kmip
```

1. Configure the KMIP listener, including the address and port CBOM will connect to:

```
vault write kmip/config \
  listen_addrs="0.0.0.0:5696" \
  tls_ca_key_type="ec" \
  tls_ca_key_bits=256 \
  default_tls_client_key_type="rsa" \
  default_tls_client_key_bits=2048
```

1. Confirm the engine is active:

```
vault secrets list | grep kmip
```

### 3.2 Create a Scope for CBOM Discovery

1. Create an isolated scope so the CBOM sensor's role cannot see keys managed under other scopes:

```
vault write -f kmip/scope/cbom-discovery
```

1. Confirm the scope was created:

```
vault list kmip/scope
```

> **Note:** Each Vault KMIP scope has its own internal CA. Do not reuse a CA certificate from a different scope when configuring the sensor in Section 6, connections will fail TLS verification if the wrong scope's CA is used.

## 4. Create a Read Only KMIP Role

Vault KMIP roles deny all operations by default; only operations explicitly listed in `operation_filters` are permitted. This makes it straightforward to build a role that can enumerate key metadata but can never retrieve, create, or destroy key material.

### 4.1 Define the Role

```
vault write kmip/scope/cbom-discovery/role/cbom-readonly \
  operation_filters="locate,get_attributes,get_attribute_list" \
  tls_client_key_type="rsa" \
  tls_client_key_bits=2048 \
  tls_client_ttl="8760h"
```

> **Note:** Do not include `get`, `create`, `register`, `activate`, `revoke`, or `destroy` in `operation_filters`. The `get` operation returns raw key material, and the write/lifecycle operations would let the sensor's credential mutate the key inventory if compromised.

### 4.2 Verify the Role's Effective Permissions

```
vault read kmip/scope/cbom-discovery/role/cbom-readonly
```

Confirm the `operation_filters` field lists only `locate`, `get_attributes`, and `get_attribute_list`.

---

## 5. Generate a Read Only Client Certificate

The CBOM sensor authenticates to Vault's KMIP listener using mutual TLS. Vault issues both the client certificate and the scope's CA directly through the CLI.

### 5.1 Generate the Client Certificate and Private Key

```
vault write -format=json \
  kmip/scope/cbom-discovery/role/cbom-readonly/credential/generate \
  format=pem > /tmp/cbom-kmip-cred.json
```

Extract the certificate and private key into separate files:

```
jq -r '.data.certificate' /tmp/cbom-kmip-cred.json > /etc/cbom/certs/vault-kmip/cbom-sensor-cert.pem
jq -r '.data.private_key' /tmp/cbom-kmip-cred.json > /etc/cbom/certs/vault-kmip/cbom-sensor-key.pem
shred -u /tmp/cbom-kmip-cred.json
```

### 5.2 Retrieve the Scope's CA Certificate

```
vault read -field=ca_chain kmip/scope/cbom-discovery/ca \
  > /etc/cbom/certs/vault-kmip/vault-kmip-ca.pem
```

### 5.3 Set Permissions and Verify the Certificate Files

```
chmod 600 /etc/cbom/certs/vault-kmip/cbom-sensor-key.pem
chmod 644 /etc/cbom/certs/vault-kmip/cbom-sensor-cert.pem
chmod 644 /etc/cbom/certs/vault-kmip/vault-kmip-ca.pem
chown cbom-agent:cbom-agent /etc/cbom/certs/vault-kmip/
```

Confirm the client certificate was signed by the scope's CA:

```
openssl verify -CAfile /etc/cbom/certs/vault-kmip/vault-kmip-ca.pem \
  /etc/cbom/certs/vault-kmip/cbom-sensor-cert.pem
```

Expected output:

```
/etc/cbom/certs/vault-kmip/cbom-sensor-cert.pem: OK
```

Confirm the private key matches the certificate:

```
openssl x509 -noout -modulus -in /etc/cbom/certs/vault-kmip/cbom-sensor-cert.pem |
md5sum
openssl rsa  -noout -modulus -in /etc/cbom/certs/vault-kmip/cbom-sensor-key.pem  |
md5sum
```

Both MD5 values must be identical.

---

## 6. Configure CBOM Sensor

Add the Vault KMIP sensor configuration to the CBOM sensor configuration file.

```
sensors:
  - name: Discover_KMIP
    type: kmip
    enabled: true
    schedule: "0 2 * * *"        # Run daily at 02:00 sensor-host local time

    connection:
      host: vault.example.internal      # Vault server hostname or IP
      port: 5696
      protocol: kmip
      kmip_version: "1.4"
      vendor_scope: cbom-discovery       # Vault KMIP scope this sensor is bound to
      tls:
        enabled: true
        ca_cert: /etc/cbom/certs/vault-kmip/vault-kmip-ca.pem
```

```
        client_cert: /etc/cbom/certs/vault-kmip/cbom-sensor-cert.pem
        client_key: /etc/cbom/certs/vault-kmip/cbom-sensor-key.pem
        verify_server_cert: true
        min_tls_version: "1.2"

    discovery:
      operations:
        - Locate
        - GetAttributes
      attribute_list:
        - Unique Identifier
        - Name
        - Cryptographic Algorithm
        - Cryptographic Length
        - Cryptographic Usage Mask
        - State
        - Initial Date
        - Activation Date
        - Deactivation Date
        - Object Type
      max_objects_per_request: 500
      include_destroyed_objects: false

    output:
      tags:
        source: hashicorp-vault-kmip
        environment: production
        sensor: Discover_KMIP
      forward_to: cbom-collector
```

Save this configuration to the sensor configuration file, typically located at:

```
/etc/cbom/config/sensors.yaml
```

Reload the sensor service to apply the new configuration:

```
systemctl reload cbom-sensor
```

---

## 7. Validation

After configuring the sensor, confirm that it connects successfully and discovers key metadata.

1. Trigger a manual sensor run to test the connection immediately without waiting for the scheduled interval:

```
cbom-sensor run --sensor Discover_KMIP --once
```

1. Check the sensor log for a successful KMIP handshake and discovery output:

```
journalctl -u cbom-sensor -n 100 --no-pager | grep -i "Discover_KMIP"
```

A successful run produces log lines similar to:

```
INFO  Discover_KMIP: TLS handshake completed with vault.example.internal:5696
INFO  Discover_KMIP: Bound to Vault KMIP scope 'cbom-discovery'
INFO  Discover_KMIP: KMIP Locate returned 96 object identifiers
INFO  Discover_KMIP: GetAttributes completed for 96 objects
INFO  Discover_KMIP: Discovery cycle finished. Forwarded 96 records to cbom-collector
```

1. Verify that key metadata records appear in the CBOM inventory by querying the collector API or UI and filtering by `source: hashicorp-vault-kmip`.
2. Confirm that no key material (raw key bytes) appears in the discovered records, only metadata fields listed in the `attribute_list` should be present.
3. Enable Vault audit logging if not already active, and confirm entries for the `cbom-readonly` role's client certificate show only `locate` and `get_attributes` KMIP operations:

```
vault audit list
vault read sys/audit
```

---

## 8. Common Errors and Resolution

### Error 1: TLS Handshake Failure, Unknown CA

```
ERROR Discover_KMIP: TLS handshake failed: x509: certificate signed by unknown
authority
```

**Cause:** The `ca_cert` configured on the sensor does not belong to the `cbom-discovery` scope. Each Vault KMIP scope has its own independent CA, a CA certificate exported from a different scope will never validate the client or server certificate for this scope.

**Resolution:** 1. Re run `vault read -field=ca_chain kmip/scope/cbom-discovery/ca` and confirm the output matches the file referenced by `ca_cert`. 2. If multiple scopes exist, double check the scope name used in both Section 3.2 and the `vendor_scope` field in Section 6. 3. Redeploy the correct CA file to the sensor host and re run validation.

---

### Error 2: KMIP Permission Denied on Locate

```
ERROR Discover_KMIP: KMIP operation Locate returned result status: Operation Failed
(ResultReason: Permission Denied)
```

**Cause:** The `cbom-readonly` role's `operation_filters` does not include `locate` and/or `get_attributes`, or the client certificate presented was generated against a different role.

**Resolution:** 1. Run `vault read kmip/scope/cbom-discovery/role/cbom-readonly` and confirm `operation_filters` includes `locate`, `get_attributes`, and `get_attribute_list`. 2. If the filters are missing, update the role as shown in Section 4.1, then regenerate the client certificate, Vault does

not retroactively apply role changes to certificates already issued. 3. Confirm the certificate deployed to the sensor host was generated from `kmip/scope/cbom-discovery/role/cbom-readonly/credential/generate` and not from an unrelated role.

---

### Error 3: Connection Refused on Port 5696

```
ERROR Discover_KMIP: dial tcp vault.example.internal:5696: connect: connection
refused
```

**Cause:** Vault's KMIP listener is not configured or the KMIP engine's ADP license is not active, so the listener never starts. This can also occur if a firewall blocks TCP `5696` between the sensor host and the Vault server.

**Resolution:** 1. Confirm the Advanced Data Protection KMIP module is present in the active Vault Enterprise license:

```
vault read sys/license/status
```

1. Confirm the listener configuration is present and the engine is mounted:

```
vault read kmip/config
```

1. From the sensor host, test basic TCP connectivity:

```
nc -zv vault.example.internal 5696
```

1. If the connection times out rather than refusing, inspect firewall rules on both the sensor host and the network path to the Vault server, and confirm any load balancer in front of Vault forwards TCP `5696` (KMIP is not an HTTP based protocol and cannot be proxied like Vault's HTTP API).

---

## 9. Security Recommendations

- **Keep the CBOM discovery scope isolated:** Do not reuse `cbom-discovery` for any application's actual encryption workflow. A dedicated scope guarantees the sensor's role can never enumerate keys belonging to production KMIP clients.
- **Never widen `operation_filters` beyond read operations:** `locate`, `get_attributes`, and `get_attribute_list` are sufficient for `Discover_KMIP`. Adding `get`, `create`, `activate`, `revoke`, or `destroy` to the `cbom-readonly` role would let a compromised sensor credential export or tamper with key material.
- **Rotate the client certificate on a schedule:** The `tls_client_ttl` set in Section 4.1 bounds the certificate's lifetime. Re run `credential/generate` before expiry and redeploy the new certificate and key to the sensor host; Vault does not auto renew KMIP client certificates.
- **Store the private key securely:** Restrict `cbom-sensor-key.pem` to the `cbom-agent` service account with `chmod 600`, and never commit it to source control or a shared filesystem.

- **Enable and monitor Vault's audit device:** Route Vault's audit log to your SIEM and alert on any KMIP operation from the `cbom-readonly` role other than `locate`, `get_attributes`, or `get_attribute_list`, such activity would indicate a misconfigured role or a compromised credential.

---

## 10. Conclusion

This guide walked through enabling Vault's KMIP secrets engine, creating an isolated scope and a read only role scoped to `locate` and `get_attributes` only, and issuing a mutual TLS client certificate for the CBOM `Discover_KMIP` sensor. Once configured, the sensor maintains a continuously updated inventory of key metadata, algorithm, length, state, and lifecycle dates, from Vault's KMIP managed keys, without ever retrieving raw key material. For questions about multi scope deployments, Vault namespaces, or integrating discovery output with the CBOM aggregator pipeline, refer to the CBOM platform documentation or contact your CBOM administrator.