Java KeyStore (JKS) Setup

CBOM Secure · Keystore Sensor (Discover_Keystore)

Overview

The CBOM Secure Discover_Keystore sensor scans Java KeyStore (JKS) files and extracts cryptographic asset metadata. JKS is Java's native format, used in Tomcat, JBoss, WebSphere, and middleware to store PrivateKeyEntry (keys + chains) and TrustedCertificateEntry (trusted certs). The sensor is read-only.

Note JKS is deprecated since JDK 9; Oracle recommends PKCS12. The sensor supports both — use JKS discovery to plan migration.

Prerequisites

- JDK 8+ on the host with keytool available.
- CBOM Agent 2.4.0+ on the host (or one with filesystem access).
- Keystore paths (.jks / .keystore) and store passwords.
- Read-only filesystem access for the agent user, and a CBOM API token.

Step-by-Step Guide

Step 1: Locate and Inspect JKS Files

```
find /opt /etc /home /var/lib /usr/local \( -name "*.jks" -o -name "*.keystore" \) -type f
2>/dev/null
keytool -list -v -keystore /opt/app/conf/keystore.jks -storepass changeit
```

Step 2: Configure a Least-Privilege Service Account

```
sudo useradd --system --no-create-home --shell /sbin/nologin cbom-sensor
sudo setfacl -m u:cbom-sensor:r /opt/app/conf/keystore.jks
sudo -u cbom-sensor keytool -list -keystore /opt/app/conf/keystore.jks -storepass changeit
```

Step 3: Store Keystore Passwords Securely

```
cbom-agent secret set jks_keystore_password "changeit"
# or a systemd EnvironmentFile at /etc/cbom/jks-secrets.env (chmod 600)
```

Step 4: Configure the CBOM Secure Sensor

```
sensor:
  name: Discover_Keystore
  enabled: true
  format: jks
cbom_platform:
  endpoint: "https://cbom.example.com"
  api_token: "${CBOM_API_TOKEN}"
discovery:
```

```
  keystores:
    - path: "/opt/app/conf/keystore.jks"
      store_password: "${secret:jks_keystore_password}"
      entry_types: [PrivateKeyEntry, TrustedCertificateEntry]
  filesystem_scan:
    enabled: true
    scan_roots: ["/opt", "/etc", "/var/lib"]
    patterns: ["*.jks", "*.keystore"]
    max_depth: 10
reporting:
  include_certificate_metadata: true
  include_key_algorithm: true
  include_validity_dates: true
agent:
  run_as_user: "cbom-sensor"
sudo cbom-agent config validate /etc/cbom/sensors/discover_keystore.yaml
sudo systemctl reload cbom-agent
```

Step 5: Validate

```
sudo cbom-agent sensor run Discover_Keystore --config
/etc/cbom/sensors/discover_keystore.yaml
tail -f /var/log/cbom/discover_keystore.log
```

Confirm entries are found per keystore and reported, then verify under Assets → Keystores with correct aliases, algorithms, sizes, and validity dates.

Common Errors

Keystore was tampered with, or password was incorrect

Cause: The store password does not match, or the file is corrupted.

Resolution: Verify with keytool -list; update the CBOM secret if changed; check file integrity.

Permission denied

Cause: The cbom-sensor user lacks read permission.

Resolution: Grant read via setfacl and confirm with sudo -u cbom-sensor keytool -list.

JKS not supported in FIPS mode

Cause: The JVM runs in FIPS mode, which disables the legacy JKS type.

Resolution: Convert the keystore to PKCS12 (keytool -importkeystore) and set format: pkcs12.

Security Recommendations

- Never store keystore passwords in plaintext — use the CBOM secret store or a secrets manager.
- Run the agent under a dedicated least-privilege account with file-level read only.
- Migrate JKS to PKCS12 (deprecated since JDK 9) for interoperability and stronger protection.
- Restrict keystore file permissions (chmod 640, owning app user + shared group).
- Rotate keystore passwords and use CBOM expiry alerts to plan renewals.

Conclusion

The Discover_Keystore sensor provides read-only discovery of private-key and trusted-certificate entries in JKS files — and, since JKS is deprecated, the discovery data doubles as a migration plan toward PKCS12 across Java infrastructure.