IBM Security Directory Server (SDS) Setup

CBOM Secure · LDAP Sensor (Discover_LDAP)

Overview

This guide configures IBM Security Directory Server (SDS, formerly Tivoli Directory Server) so the CBOM Secure Discover_LDAP sensor can perform read-only discovery of cryptographic assets in the directory:

```
userCertificate — X.509 certificates on user/service entries
cACertificate — X.509 certificates on Certificate Authority entries
```

Prerequisites

- IBM SDS 6.3+ installed and running.
- Admin access to the Web Administration Tool (http://<server>:12100/IDSWebApp).
- LDAP (389) or LDAPS (636) enabled and reachable from the sensor host.
- The base DN of the directory tree (e.g. dc=example,dc=com).
- Shell access to restart the instance (ibmdiradm); the server cert/CA for LDAPS.

Step-by-Step Guide

Step 1: Create the CBOM Bind Account

In the Web Administration Tool → Directory Management → Add an Entry, create an inetOrgPerson with RDN cn=cbom-reader under your service-account container, producing cn=cbom-reader,ou=serviceaccounts,dc=example,dc=com. Set sn and a strong userPassword.

Step 2: Configure Attribute-Level ACLs

Under Directory Management → Access Control Lists → Add ACL, create two entries scoped to the base DN subtree, with subject DN cn=cbom-reader,ou=serviceaccounts,dc=example,dc=com, granting Read on userCertificate and cACertificate respectively.

Step 3: Restart the SDS Instance

```
ibmdiradm -n                    # list instances
ibmdiradm -I dsrdbm01 restart
ibmdiradm -I dsrdbm01 status
```

Step 4: (LDAPS) Export the Server Certificate

```
gsk8capicmd_64 -cert -extract \
  -db /opt/IBM/ldap/V6.4/etc/dsrdbm01/ldap.kdb \
  -pw <keystore-password> \
  -label "Server Certificate" \
  -target /tmp/sds-server.crt -format ascii
```

Copy sds-server.crt to the sensor host for the ca_cert_path.

Step 5: Configure the CBOM Secure Sensor

```
sensors:
  - name: Discover_LDAP
    enabled: true
    label: "IBM SDS - Production Directory"
    connection:
      host: "sds.example.com"
      port: 636
      use_ssl: true
      verify_ssl: true
      ca_cert_path: "/etc/cbom/certs/sds-server.crt"
    bind:
      dn: "cn=cbom-reader,ou=serviceaccounts,dc=example,dc=com"
      password: "${CBOM_LDAP_BIND_PASSWORD}"
    discovery:
      base_dn: "dc=example,dc=com"
      scope: "subtree"
      filter: "(|(objectClass=inetOrgPerson)(objectClass=certificationAuthority))"
      attributes: ["userCertificate", "cACertificate", "dn", "cn", "mail"]
```

Step 6: Validate

```
cbom sensor run --name Discover_LDAP
ldapsearch -H ldaps://sds.example.com:636 \
  -D "cn=cbom-reader,ou=serviceaccounts,dc=example,dc=com" -w "<pw>" \
  -b "dc=example,dc=com" -s sub "(userCertificate=*)" userCertificate cACertificate
```

Confirm a successful last run and discovered assets under the ibm-sds tag; spot-check a certificate's subject, expiry, and issuer.

Common Errors

LDAP error 49 — Invalid Credentials

Cause: Wrong bind DN/password, or the account is in a different subtree than configured.

Resolution: Verify the full bind DN in the Web Admin Tool; reset the password if needed.

LDAP error 32 — No Such Object

Cause: The configured base_dn does not exist or the scope is too narrow.

Resolution: Confirm the base DN by browsing the tree; adjust base_dn/scope and test with ldapsearch -s base.

SSL handshake / certificate verify failed

Cause: The host value does not match the cert CN/SAN, or the cert is untrusted.

Resolution: Match host to the cert CN/SAN; re-export the full CA chain to ca_cert_path.

Security Recommendations

- Use LDAPS exclusively; disable plain LDAP (389) in production.

- Limit ACLs to userCertificate and cACertificate — never userPassword or operational attributes.
- Rotate the bind password on schedule via a secrets manager; inject via env var.
- Restrict the bind account by IP/time-of-day where supported.
- Enable SDS audit logging for the cbom-reader DN.

Conclusion

With a scoped read-only bind account and attribute-level ACLs, the Discover_LDAP sensor continuously inventories X.509 certificates in IBM SDS — tracking expiry, weak algorithms, and ownership without any write access.