CADI Capabilities

1. Authentication
   1. Basic Auth
   2. X.509-based authentication via TLS
   3. OAuth2.0
   4. Company specific (requires configuration)
2. Identification
   1. Normalizes authenticated identity to identity recognized by AAF
   2. X.509 Subject identity is normalized based on the X.500 Issuer (X.500 Issuer certificate, and normalization rules must be configured in the CADI client)
   3. Conversion of Identity from OAuth2.0 access token
3. Retrieval of Permissions from AAF
   1. Calls the AAF RESTful APIs with Identity to get permissions for that Identity
4. Manages TLS negotiation
   1. Performs TLS handshaking
5. Enrich outgoing API request to credentials
   1. Determines credentials expected by APIs called by the application – must be configured
   2. Negotiates TLS protocol including X.509-based authentication (one-way and mutual authentication)
   3. OAuth2.0 access token negotiation
   4. Basic Auth
   5. Company specific (requires configuration)
6. Certificate management – CertMan agent
   1. CertMan agent requests certificate (initial certificate or renewal) from the CertMan server using CertMan API
   2. CertMan server generates key pair, generates CSR, submits CSR
   3. CertMan API responds with certificate and private key
   4. CertMan agent converts certificate and private key into required format (e.g., PKCS12, PEM) and installs in the proper location (e.g., key store, Docker volume)

Note that the CertMan server in AAF acts as a Registration Authority, authenticating all certificate requests.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Capability** | **CADI** | **Notes** | **CADI + Sidecar** | **Notes** | **ISTIO** | **Notes** |
| Authentication |  |  | As per CADI – embedded in reverse-proxy sidecar | to be extended to allow configuration of which credentials/tokens to be used for AuthN/Z | * X.509v3 certificates * JWT (JSON Web Token) * OAuth2 * OpenID\_Connect |  |
| Identification |  |  | As per CADI – embedded in reverse-proxy sidecar |  | Subject name in case of certificates. Support SPIFFE identity.  Claims in case of JWT. |  |
| Permissions Mapping |  |  | AuthZ user admin as per CADI/AAF or alternate provider.  Configuration driven AuthZ enforcement point in reverse proxy – compares CADI retrieved auths with those set for URI pattern.  Primary service agnostic to AuthZ check | Future extension to allow query of request auths by primary service via REST – language agnostic. | ISTIO RBAC (ISTIO has its own RBAC) | Supports creation of roles, rules for each role and role binding with identities (Subject name in case of Certificates, Claims in case of JWT.  Rules can contain source services, URI etc.. |
| TLS negotiation |  |  | As Per CADI |  | Yes | Mutual TLS |
| Outgoing API Request Enrichment |  |  | Isolated forward proxy in POD enriches outgoing requests from primary microservice with desired credentials.  Agnostic to primary service language or libs.  Credentials not exposed to primary service. |  | * X.509 certificate * JWT in each request |  |
| Certificate Management |  |  | As per CADI |  | * Traditional (CSR) * gRPC based |  |