Agent Trust Broker (ATB)

Documentation Guide

Generated: January 13, 2026

Version: 0.1.0

Table of Contents

1. Overview

ATB (Agent Trust Broker) is a security enforcement layer for enterprise AI agent deployments, implementing the AI Safe Enterprise Autonomy Architecture.

ATB provides a single enforcement boundary between AI agent platforms and enterprise systems. Every agent action is:

- **Authenticated** via SPIFFE/SPIRE workload identity
- **Authorized** via signed Proof-of-Authorization (PoA) mandates
- **Constrained** by OPA policy with risk-tiered controls
- **Audited** with immutable, tamper-evident logs

2. Architecture

ATB is an enterprise security enforcement layer that validates AI agent actions before they execute on backend systems. It implements a Proof-of-Authorization (PoA) framework with risk-tiered governance.

Request Flow

1. Agent → Broker: mTLS with SPIFFE cert + PoA token
2. Broker extracts SPIFFE ID and validates PoA JWT signature
3. Broker → OPA: Policy decision request
4. If allowed: Broker → Upstream: Proxy request + Audit log
5. If denied: Broker → Agent: 403 + denial reasons + Audit log

3. Key Features

Feature	Description
SPIFFE/SPIRE Identity	X509-SVID for mTLS, JWT-SVID for external APIs
PoA Mandates	Short-lived, signed authorization tokens with act/con/leg claims
Risk-Tiered Policy	145+ enterprise actions across low/medium/high risk tiers
Dual Control	High-risk actions require two distinct approvers

Semantic Guardrails	Prompt injection detection with external service support
Immutable Audit	Azure Blob/S3 Object Lock with hash-chain tamper evidence
Platform Binding	OIDC platform tokens bound to SPIFFE identities

4. Components

Component	Description
atb-broker	Main enforcement gateway (Go)
atb-agentauth	PoA issuance service with dual-control support
opa	Policy decision engine (sidecar)
spire-agent	SPIFFE workload identity

ATB Broker

The broker is the core gateway that:

- Terminates mTLS connections from AI agents
- Extracts SPIFFE IDs from client certificates
- Validates PoA tokens (RS256 JWT mandates)
- Queries OPA for policy decisions
- Proxies authorized requests to upstream backends
- Emits audit events for compliance

AgentAuth Service

Issues PoA tokens to authorized agents:

- Validates agent identity via mTLS/SPIFFE
- Mints short-lived PoA JWTs with action scope
- Enforces platform-specific constraints
- Supports risk-tier approval requirements

5. Proof-of-Authorization (PoA)

PoA tokens are short-lived, signed JWTs that authorize specific actions. They are the core authorization mechanism in ATB.

PoA Token Structure

A PoA token is a signed JWT mandate that authorizes a specific action:

```
{
"sub": "spiffe://example.org/agent/demo",
"act": "crm.contact.update",
"con": {
"max_records": 10,
"allowed_fields": ["name", "email"]
},
"leg": {
"basis": "contract",
"jurisdiction": "US",
"accountable_party": {
"type": "human",
"id": "user@example.com"
}
},
"iat": 1736679600,
"exp": 1736679900,
"jti": "poa_abc123xyz"
}
```

PoA Claims

Claim	Required	Description
sub	Yes	Subject (agent's SPIFFE ID)
act	Yes	Action being authorized
con	No	Constraints (limits, filters)
leg	Yes	Legal basis for the action
iat	Yes	Issued at (Unix timestamp)
exp	Yes	Expiration (Unix timestamp)
jti	Yes	Unique token ID (replay protection)

Legal Basis (leg)

Every PoA must include a legal basis explaining why the action is permitted:

- **basis**: contract, consent, legitimate_interest, legal_obligation
- **ref**: Reference to legal document (e.g., MSA-2026-001)
- **jurisdiction**: Legal jurisdiction (e.g., US, DE, UK)
- **accountable_party**: Who is accountable (human or organization)

Constraints (con)

Constraints limit what the action can do:

- max_amount, currency - Financial limits
- allowed_vendors - Vendor allowlists
- max_records - Record count limits
- exclude_fields - PII field exclusions

6. Risk Tiers

ATB enforces three risk tiers based on the action being performed:

Tier	Actions	Approval	Examples
HIGH	60+	Dual control (2 approvers)	SAP payments, PII export, IAM escalation
MEDIUM	40+	Single approver	CRM updates, order management
LOW	45+	PoA only	Read operations, status checks

Dual Control Rules (High Risk)

- Two approvers must be **distinct** (different approver_id)
- The **requester** cannot be an approver
- Both approvals must happen before the challenge expires
- Approval order doesn't matter

7. Authentication Flow

ATB uses a zero-trust model where every action must be explicitly authorized:

Step 1: Agent requests a challenge from AgentAuth (POST /v1/challenge)

Step 2: Approvals collected based on risk tier

Step 3: AgentAuth issues PoA token after approvals

Step 4: Agent calls Broker with X-Poa-Token header

Step 5: Broker validates token and proxies to upstream

Medium-Risk Flow (Single Approval)

```
# 1. Create challenge
POST /v1/challenge
{ "action": "crm.contact.update", ... }

# 2. Submit approval
POST /v1/challenge/{id}/approve
{ "approver_id": "manager@example.com" }

# 3. Get PoA token in response
{ "poa": "eyJhbGciOiJSUzI1NiI..." }
```

High-Risk Flow (Dual Control)

```
# 1. Create challenge (requires 2 approvers)
POST /v1/challenge
{ "action": "sap.payment.execute", ... }

# 2. First approval
POST /v1/challenge/{id}/approve
{ "approver_id": "finance-manager@example.com" }

# 3. Second approval (different person!)
POST /v1/challenge/{id}/approve
{ "approver_id": "cfo@example.com" }

# 4. PoA token issued after both approvals
```

8. SPIFFE/SPIRE Identity

Every workload in ATB has a cryptographic identity via SPIFFE (Secure Production Identity Framework for Everyone).

SPIFFE ID Format

```
spiffe://<trust-domain>/<workload-path>
```

Examples:

- spiffe://prod.company.com/ns/agents/sa/claude-assistant
- spiffe://prod.company.com/ns/connectors/sa/sap-connector

How Identity Works

- **SPIRE Agent** runs on each node
- **Workloads** request SVIDs (SPIFFE Verifiable Identity Documents)
- **X.509-SVID** used for mTLS connections
- **JWT-SVID** can be used for API authentication

9. API Reference

Endpoint	Method	Service	Purpose
/health	GET	Both	Health check
/authorize	POST	AgentAuth	Request PoA token (low-risk)
/challenge	POST	AgentAuth	Create approval challenge
/challenge/{id}/approve	POST	AgentAuth	Submit approval
/challenge/{id}/complete	POST	AgentAuth	Get PoA after approval
/*	ANY	Broker	Proxy to upstream with PoA validation

Headers

Header	Required	Description
X-Poa-Token	Yes (Broker)	Signed PoA JWT token
X-Request-Id	No	Correlation ID for tracing
Authorization	Alt	Bearer token (alternative to X-Poa-Token)

10. Configuration

Environment Variables (Broker)

Variable	Default	Description
SPIFFE_ENDPOINT_SOCKET	/run/spire/sockets/agent.sock	SPIRE Workload API socket
OPA_DECISION_URL	http://localhost:8181/...	OPA policy endpoint
POA_SINGLE_USE	true	Enable PoA replay protection
ALLOW_UNMANDATED_LOW_RISK	false	Allow low-risk without PoA
GUARDRAILS_URL	-	External guardrails service
AUDIT_SINK_URL	-	Audit event sink endpoint

Quick Start

Deploy with Helm:

```
helm install atb charts/atb \
-n atb \
-f charts/atb/values-staging.yaml \
-f charts/atb/values-observability.yaml
```

Docker Compose (Development):

```
make docker-up

# Services:
# OPA: http://localhost:8181
# Upstream: http://localhost:9000
# Broker: https://localhost:8443 (mTLS)
# AgentAuth: http://localhost:8444
```