Enterprise Readiness Report

Pilot Protocol

Workload Identity, Policy, Revocation, Audit,
and Dark-by-Default Reachability

March 2026

Calin Teodor

Vulture Labs

https://vulturelabs.com

An assessment of enterprise infrastructure alignment and the roadmap to close remaining gaps.

Abstract

"Solves connectivity" is only part of the enterprise problem. The harder bar is whether the connectivity layer also aligns with workload identity, policy, revocation, audit, and dark-by-default reachability—instead of just making peer connectivity easier. This report audits Pilot Protocol's current capabilities against these six enterprise dimensions, identifies the gaps, maps to emerging industry standards (SPIFFE/SPIRE, ONUG AOMC, CSA ATF/AICM), and presents a five-phase implementation roadmap. The goal: Pilot Protocol should interoperate with existing identity models rather than becoming a parallel trust island.

Contents

1. Executive Summary

Pilot Protocol gives AI agents first-class network citizenship: virtual addresses, encrypted UDP tunnels, NAT traversal, port-based services, and a bilateral trust model. The reference implementation has been validated across five geographic regions with 683 tests.

But connectivity alone does not meet the enterprise bar. An enterprise connectivity layer must also provide:

1. **Workload identity** that integrates with existing infrastructure (OIDC, SPIFFE, x509)—not a parallel identity system.
2. **Policy enforcement** that controls who can reach whom, on which ports, based on identity attributes.
3. **Revocation** that cascades instantly when a node is compromised—not just bilateral untrust.
4. **Audit** that is persistent, machine-parseable, and queryable—not fire-and-forget webhooks.
5. **Dark-by-default reachability** that is enforced at every layer, including the data plane—not just the control plane.
6. **Interoperability** with enterprise identity providers—OIDC issuers, SPIFFE trust domains, x509 CAs.

This report finds that Pilot Protocol has strong foundations in several dimensions (Ed25519 identity, private-by-default discovery, bilateral trust, webhook events) but has critical gaps in others (SYN-level trust enforcement, external identity integration, cascading revocation, policy engine).

The five-phase roadmap addresses each gap incrementally. Each phase is independently shippable and delivers measurable enterprise value.

2. Current Capabilities Audit

This section inventories what Pilot Protocol provides today across all six enterprise dimensions.

2.1 Identity

Ed25519 keypair per node. Every node receives an Ed25519 keypair from the registry at registration. The private key serves as the node's identity token. Identities can be persisted to disk (`-identity`) for address and key continuity across restarts.

Signed mutations. All registry write operations routed through the daemon (set-hostname, set-visibility, deregister) are signed with the node's Ed25519 private key. The registry verifies signatures before applying mutations, preventing spoofed requests.

Key rotation. The `rotate_key` command supports two authentication paths: signature-based (proving possession of the current key) and owner-based (matching an email identifier). After rotation, the registry issues a fresh keypair while preserving the node's ID and network memberships.

TLS cert pinning. The registry client supports TLS with certificate pinning (`DialTLSPinned`), preventing man-in-the-middle attacks on the control plane.

Owner binding. Nodes can be bound to an owner identifier (typically an email) via the `-owner` flag. This enables key recovery and re-registration after deregistration.

Authenticated key exchange. When a daemon has a persisted Ed25519 identity, the tunnel key exchange is upgraded from anonymous (`PILK`) to authenticated (`PILA`). The signature proves the X25519 ephemeral key belongs to the claimed identity, preventing man-in-the-middle substitution.

2.2 Trust

Bilateral handshake protocol. Port 444 provides an application-level trust handshake with three message types: request (with justification), accept, and reject (with reason). Trust requires mutual consent.

Three auto-approval paths. (1) Mutual: if both nodes independently request a handshake, trust is auto-approved. (2) Network: if two nodes share a non-backbone network, trust is auto-approved. (3) Manual: queued for operator approval.

Trust persistence. Approved peer relationships are persisted to disk and survive daemon restarts without re-negotiation.

Trust revocation. The `untrust` command revokes a trust pair bilaterally. The revocation is propagated to the peer and the registry. The associated tunnel is torn down.

2.3 Privacy and Dark-by-Default

Private by default. All nodes are private at registration. A node's physical IP:port is never exposed unless the node has explicitly opted into public visibility.

Resolve gating. To discover a private node's endpoint via `resolve`, the requester must satisfy one of three conditions: the target is public, the requester and target share a mutual trust pair, or both belong to a common non-backbone network.

Backbone enumeration blocked. Listing nodes on network 0 (the global backbone) is rejected by the registry. With 4 billion possible node IDs, individual nodes are addressed by ID, not discovered by enumeration.

Handshake relay. Private nodes are reachable for trust negotiation through the registry's inbox relay. No IP address is exposed until both parties have consented.

2.4 Audit and Observability

Webhook system. The daemon supports 20+ webhook event types: connect, disconnect, handshake request/accept/reject, trust established/revoked, datagram sent/received, connection state changes, and more. Events are delivered via HTTP POST with JSON payloads.

Structured logging. All components use Go's `log/slog` for structured, leveled logging. JSON format is available for production log aggregation.

Per-connection statistics. Each connection tracks bytes/segments sent/received, retransmissions, fast retransmits, SACK blocks, duplicate ACKs, congestion window, in-flight data, SRTT, RTTVAR, and recovery status.

2.5 Rate Limiting and Connection Control

Two-tier SYN rate limiting. A global SYN rate limiter and per-source SYN rate limiter prevent SYN flood attacks.

Connection limits. Per-port connection limits and global connection limits prevent resource exhaustion.

Handshake replay protection. SYN deduplication ensures retransmitted SYN packets reuse existing connections rather than creating duplicates.

Registry rate limiting. Per-connection sliding window rate limits with automatic cleanup of stale entries.

2.6 Encryption

Tunnel-layer encryption. Enabled by default. X25519 ECDH key exchange with AES-256-GCM authenticated encryption. All packets between peers are encrypted regardless of virtual port.

Application-layer encryption. Port 443 provides end-to-end X25519 + AES-256-GCM encrypted channels for applications requiring an additional encryption layer.

Zero external dependencies. All cryptography uses Go's standard library (`crypto/ecdh`, `crypto/aes`, `crypto/cipher`, `crypto/ed25519`).

3. Enterprise Gap Analysis

This section identifies what is missing in each dimension and assesses severity.

3.1 Identity Gaps

Gap	Severity	Impact
No external identity (OIDC, SPIFFE, x509)	High	Creates parallel trust island
No identity expiry or forced rotation	Medium	Compromised keys live forever
No workload attestation	Medium	Cannot verify *what* a process is
No trust domain concept	Medium	No federation boundary
Owner field is unverified	Low	Email is informational only

Table 1: Identity gaps.

Verdict. The self-contained Ed25519 identity system is cryptographically solid. But enterprises will not deploy a second identity system. The absence of OIDC/SPIFFE integration means every Pilot deployment is a parallel trust island disconnected from the organization's IAM infrastructure.

3.2 Policy Gaps

Gap	Severity	Impact
No tag-based access rules	High	Tags exist but are purely informational
No per-port ACLs tied to identity	High	No "who can reach whom on which ports"
No policy engine or rule evaluation	High	No declarative access control
No data-plane policy enforcement	Critical	Resolve gating bypassed if endpoint known

Table 2: Policy gaps.

Verdict. The resolve layer is effective dark-by-default gating. But once a node knows an endpoint (by any means), there is no policy enforcement at the data plane. Tags exist (up to

10 per node) but are not used for access decisions. The gap between "visibility control" and "access control" is significant.

3.3 Revocation Gaps

Gap	Severity	Impact
No network-wide ban/revoke	High	Admin cannot revoke a compromised node
No cascading revocation	High	Revocation does not propagate to all peers
No block list	Medium	Cannot permanently deny a specific node
Network departure does not cascade	Medium	Leaving a network does not revoke trust
No CRL/OCSP-like mechanism	Medium	No certificate revocation infrastructure

Table 3: Revocation gaps.

Verdict. Revocation is purely bilateral. When a node is compromised, the only option is for each individual peer to manually untrust it. An administrator cannot revoke a node from all relationships at once. This is inadequate for any deployment with more than a handful of nodes.

3.4 Audit Gaps

Gap	Severity	Impact
Audit is fire-and-forget HTTP POST	Medium	Events lost on delivery failure
No persistent audit log	Medium	No historical query capability
Queue overflow drops events silently	Medium	Gaps in audit trail undetectable
No registry-side audit events	Medium	Admin operations not tracked

Table 4: Audit gaps.

Verdict. The webhook foundation is comprehensive (20+ event types with timestamps, node IDs, and ports). The gaps are in reliability and persistence. Silent event drops, no delivery retry, and no registry-side audit trail make the current system unsuitable for compliance environments.

3.5 Dark-by-Default Gaps

Gap	Severity	Impact
SYN handler has no trust check	**Critical**	Open door if endpoint is known
`resolve_hostname` leaks existence	High	Confirms node exists without auth
No ongoing authorization	Medium	Trust checked once, never re-validated

Table 5: Dark-by-default gaps. The SYN handler gap is the most critical security finding.

Verdict. The resolve layer provides strong dark-by-default gating at the control plane. But the daemon's SYN handler accepts connections from any source that knows the endpoint. If an attacker obtains an endpoint address by any means (network sniffing, leaked config, prior trust), they can connect without trust verification. This is the most critical security gap in the current implementation.

3.6 Interoperability Gaps

Verdict. Currently a full parallel trust island. Zero integration with enterprise identity infrastructure. Enterprises must choose between Pilot's identity system and their existing IAM—they cannot use both.

Gap	Severity	Impact
No OIDC token validation	High	Cannot use enterprise SSO
No SPIFFE SVID acceptance	High	Cannot use SPIRE infrastructure
No trust domain federation	Medium	No cross-org trust boundaries
No external identity mapping	Medium	Audit trail disconnected from IAM

Table 6: Interoperability gaps.

4. Standards Landscape

This section maps Pilot Protocol's position to emerging standards and frameworks for agent security and identity.

4.1 SPIFFE/SPIRE

The Secure Production Identity Framework for Everyone (SPIFFE) defines a standard for workload identity in dynamic environments. SPIRE (the SPIFFE Runtime Environment) is the reference implementation.

Core concepts:

- **SPIFFE ID.** A URI identifying a workload: `spiffe://trust-domain/path`. Platform-agnostic and infrastructure-independent.
- **SVID.** A SPIFFE Verifiable Identity Document—either an x509 certificate or a JWT carrying the SPIFFE ID as the subject. Short-lived and automatically rotated.
- **Trust domain.** An administrative boundary. Workloads within the same trust domain share a root CA. Federation links trust domains across organizations.
- **Workload API.** A local Unix socket that workloads call to fetch their own SVID. No secrets management required—the SPIRE agent handles attestation and certificate delivery.

Relevance to Pilot. SPIFFE solves the "what is this process?" problem that Pilot's Ed25519 identity does not address. A Pilot node proves it holds a private key, but cannot prove it is authorized to act as a specific workload. SPIFFE SVIDs provide this attestation. Accepting JWT-SVIDs as proof for network join operations would make Pilot a SPIFFE-aware workload rather than a parallel identity system.

4.2 ONUG AOMC

The Open Networking User Group (ONUG) published the Autonomous Operations and Management Controls (AOMC) framework defining six mandatory controls for securing autonomous AI agent operations in enterprise environments.

The six AOMC controls:

1. **Authentication and Authorization.** Verify agent identity before granting access.
2. **Access Management.** Enforce least-privilege access to resources and other agents.
3. **Traffic Inspection and Control.** Monitor and filter agent-to-agent communication.
4. **Communication Security.** Encrypt data in transit between agents.
5. **Audit and Compliance.** Log all agent actions for forensic analysis.

6. **Lifecycle Management.** Control agent provisioning, rotation, and decommissioning.

Relevance to Pilot. Pilot currently satisfies controls 1 (partial—Ed25519 auth, no external identity), 4 (full—AES-256-GCM tunnel encryption), and 6 (partial—registration/deregistration, key rotation). Controls 2, 3, and 5 have significant gaps.

4.3 CSA ATF/AICM

The Cloud Security Alliance (CSA) Agent Trust Framework (ATF) and Agentic Identity and Context Model (AICM) define trust requirements for autonomous AI agents.

Key ATF elements:

- **Progressive autonomy.** Agents should earn expanded permissions through demonstrated trustworthy behavior.
- **Identity binding.** Agent identity should be cryptographically bound to the deploying organization.
- **Context propagation.** Trust decisions should consider the full context: who deployed the agent, what task it is performing, and what permissions it needs.
- **Revocation and quarantine.** Compromised agents should be immediately isolated from the network.

Relevance to Pilot. Pilot's polo score (reputation system) aligns with progressive autonomy. The bilateral trust handshake (with justification) provides context for trust decisions. But identity binding to the deploying organization is missing (no OIDC/SPIFFE), and revocation does not cascade.

4.4 Zero-Trust Patterns

Modern overlay networks (OpenZiti, Tailscale, WireGuard) implement zero-trust patterns that provide useful comparison points.

OpenZiti. Dark-by-default: all services are invisible until a policy explicitly allows access. Identity is x509-based with automatic rotation. Policies are declarative ("service A is accessible by identity B on port C"). The closest model to what Pilot should become.

Tailscale. Identity is OIDC-based (Google, Microsoft, GitHub, Okta). ACLs are declarative JSON defining which users/groups can reach which services. MagicDNS provides name resolution. The reference for enterprise-friendly identity integration.

WireGuard. Lightweight encrypted tunnels with pre-shared public keys. No identity layer, no policy engine, no discovery. Demonstrates that minimal protocol + external tooling can achieve enterprise deployment.

Comparison. Pilot Protocol's architecture is most similar to OpenZiti (overlay with virtual addresses, dark-by-default, identity-based access). The key difference is that OpenZiti has a mature policy engine and CA-based identity, while Pilot has stronger NAT traversal and agent-native semantics (ports, trust handshakes, reputation).

5. Implementation Roadmap

The roadmap is structured in five phases. Each phase is independently shippable and delivers measurable enterprise value.

`Phase 1` → `Phase 2` → `Phase 3` → `Phase 4` → `Phase 5`

Phase	Name	Delivers
1	Activate & Harden	Network create/join/leave via CLI. SYN-level trust enforcement. Hostname privacy fix.
2	Auto-Join & Audit	Daemon auto-joins networks at startup. Registry audit events. Webhook reliability.
3	Org + Policy + Revoke	Organization control plane. Tag-based access policies. Cascading admin revocation. Block list.
4	CA Enrollment	CA-signed certificates as join proof. Per-network revocation lists. Identity expiry.
5	External Identity	OIDC token validation. SPIFFE SVID acceptance. Identity mapping. Not a trust island.

Table 7: Five-phase implementation roadmap.

5.1 Phase 1: Activate Network Primitives + Harden Dark-by-Default

Objective. Unblock existing WIP network code. Fix the critical SYN trust gap. Fix the hostname information leak.

Deliverables:

- **Registry: enable network operations.** Remove WIP guards from `create_network`, `join_network`, and `leave_network` handlers.
- **Rendezvous: admin token flag.** Add `-admin-token` flag to the rendezvous binary.
- **Daemon: enable broadcast.** Remove WIP guard from the broadcast datagram path.
- **Daemon: SYN trust gate (critical).** After rate limiting and before connection limits, add a trust check to the SYN handler. Incoming connections from untrusted sources (no trust pair, no shared network) are rejected with RST and a webhook event.
- **Registry: hostname privacy.** Add requester signature verification to `resolve_hostname`. Private nodes require trust or shared network before confirming existence.
- **IPC + Driver + CLI: network commands.** Full stack for create, join, leave, list networks, and list nodes.
- **Tests.** Unskip 20 WIP-gated test sites. Add `TestSYNTrustGate`.

Enterprise value. Token-gated and invite-gated networks. Same-network auto-trust. SYN-level dark-by-default enforcement. Closes the most critical security gap.

5.2 Phase 2: Daemon Auto-Join + Audit Improvements

Objective. Agents auto-join networks at startup via configuration. Audit events gain structure and reliability.

Deliverables:

- **Daemon config: network enrollment.** Add `networks` array and `admin_token` to daemon config. On startup, the daemon auto-joins configured networks.

- **Registry audit events.** Structured log entries for administrative operations: network created, network joined, node registered/deregistered, trust reported/revoked, visibility/hostname changed. Machine-parseable JSON for SIEM integration.
- **Webhook reliability.** Log dropped events on queue overflow. Add monotonic `event_id` for gap detection. Retry failed POSTs once with 1s backoff.
- **CLI flags.** `--admin-token` and `--networks` for daemon and pilotctl.

Enterprise value. Deploy agent swarms with identical configs. Machine-parseable audit trail for SIEM. Reliable webhook delivery with gap detection.

5.3 Phase 3: Organization Control Plane + Policy + Cascading Revocation

Objective. Enterprise fleet management. Tag-based policy. Admin-initiated cascading revocation.

Deliverables:

- **Organization data model.** Orgs with admin tokens, member lists, and associated networks. Single `org_enroll` joins all org networks.
- **Network access policies.** Tag-based access rules: "nodes with tag `frontend` can reach nodes with tag `api` on ports 80, 443." Policies are attached to networks and evaluated at resolve time and SYN time. Default-deny when policies exist.
- **Cascading revocation.** `org_revoke` removes a node from the org, all org networks, revokes all trust pairs, and pushes revocation to all online peers. Affected peers tear down tunnels immediately.
- **Block list.** Persistent node block list in the handshake manager. Blocked nodes are rejected at SYN time before trust checks.
- **Network departure cascade.** Leaving a network revokes trust pairs that were established via that network.
- **Full stack wiring.** Registry, client, IPC, driver, and CLI for all org and policy operations.

Enterprise value. Organization-level fleet management. Declarative access policies. Instant cascading revocation on compromise. Permanent block capability.

5.4 Phase 4: CA-Based Enrollment + Hardened Trust

Objective. Cryptographic enrollment without shared secrets. Revocation lists. Identity expiry.

Deliverables:

- **New join rule: "ca".** Networks can require a CA-signed certificate as proof of authorization. The registry verifies the signature against the network's CA public key.
- **CA tooling.** Generate CA keypairs, sign node public keys, verify certificates. All Ed25519-based, using Go's standard library.
- **Per-network revocation lists.** Certificate fingerprints can be revoked. The registry checks the revocation list during join validation.
- **Identity expiry.** Optional expiry timestamps on node identities. The registry rejects re-registration after expiry, forcing key rotation.

- **CLI.** `pilotctl ca generate`, `pilotctl ca sign`, `pilotctl network create --join-rule ca`.

Enterprise value. Cryptographic proof of authorization with no shared secrets. Offline certificate signing. Per-network revocation lists. Forced key rotation via expiry.

5.5 Phase 5: External Identity Integration (OIDC / SPIFFE)

Objective. Accept enterprise identity as proof for network membership. Stop being a parallel trust island.

Deliverables:

- **New join rule: "oidc".** Networks can require a valid OIDC JWT from a configured issuer (Google, Azure, Okta, GitHub). The registry validates the JWT signature via JWKS, checks issuer/audience/expiry, and optionally matches claim values.
- **New join rule: "spiffe".** Networks can require a valid JWT-SVID from a configured trust domain. The registry validates against the trust bundle and matches SPIFFE ID patterns.
- **Daemon SPIFFE Workload API integration.** If a SPIFFE Workload API socket is available (`SPIFFE_ENDPOINT_SOCKET`), the daemon can automatically fetch JWT-SVIDs and use them for network join operations.
- **Identity mapping.** `NodeInfo` gains `external_id` and `external_issuer` fields, linking Pilot's 48-bit address to the enterprise identity. Audit logs reference both.
- **CLI and config.** `pilotctl network create --join-rule oidc --oidc-issuer ...` and daemon config with `oidc_token_path` or `spiffe_auto`.

Enterprise value. Agents prove identity via existing enterprise infrastructure. No parallel trust island. No shared secrets to distribute. Audit trail links Pilot addresses to enterprise identities. Enterprises adopt Pilot without deploying a second identity system.

6. Standards Alignment Matrix

The following matrix maps ONUG AOMC controls and CSA ATF elements to Pilot Protocol capabilities—both current and planned.

Summary. After all five phases:

- All six ONUG AOMC controls: Full
- All four CSA ATF elements: Full
- SPIFFE/SPIRE integration: Full
- Zero-trust alignment: Full

7. Strategic Positioning

After all five phases, Pilot Protocol can be positioned as:

> *A connectivity substrate beneath MCP and A2A that understands your existing identity infrastructure. Agents with valid OIDC tokens or SPIFFE SVIDs auto-join the right networks. Tag-based policies control who can reach whom. Admin revocation*

cascades instantly to all peers. The whole thing works across NAT and cloud boundaries without VPN infrastructure.

A2A defines what agents say to each other. MCP defines what tools agents can use. Pilot defines how agents reach each other—and with enterprise identity integration, it does so within the organization's existing trust boundaries, not in a parallel island.

The competitive landscape:

- **vs. Tailscale/WireGuard.** Pilot is agent-native (ports, trust handshakes, reputation, service discovery), not device-native. Pilot has bilateral trust, not admin-controlled ACLs. Both should support OIDC.
- **vs. OpenZiti.** Similar architecture (overlay, dark-by-default, identity-based access). OpenZiti has a mature policy engine; Pilot has stronger NAT traversal and agent-native semantics. Phase 3 closes the policy gap.
- **vs. libp2p.** Pilot provides structured overlay semantics (addresses, ports, networks) rather than a general-purpose P2P toolkit. Pilot is simpler to deploy (single binary, zero dependencies).
- **vs. NATS/Kafka.** Message brokers, not connectivity substrates. Pilot provides the transport layer that message brokers can run on top of.

The key insight: enterprises do not need another connectivity tool. They need a connectivity layer that respects their existing identity, policy, and compliance infrastructure. The five-phase roadmap transforms Pilot from a capable connectivity tool into an enterprise-grade infrastructure component.

Pilot Protocol is developed by Calin Teodor at Vulture Labs. The reference implementation is available at the project repository.

Standard / Control	Current State	Planned (Phase)	Status
ONUG AOMC Controls			
1. Authentication & Authorization	Ed25519 identity, signed mutations, TLS pinning	OIDC/SPIFFE auth (P5), CA enrollment (P4)	Partial
2. Access Management	Visibility: public/private, resolve gating	Tag-based policies (P3), per-port ACLs (P3)	Gap
3. Traffic Inspection	Rate limiting, connection limits	Policy enforcement at SYN (P3), audit events (P2)	Gap
4. Communication Security	X25519+AES-256-GCM tunnel, E2E on port 443	—	Full
5. Audit & Compliance	20+ webhook events, structured logging	Persistent audit (P2), registry events (P2), external ID (P5)	Partial
6. Lifecycle Management	Registration, deregistration, key rotation	Identity expiry (P4), cascading revocation (P3), org management (P3)	Partial
CSA ATF Elements			
Progressive Autonomy	Polo score (reputation system)	—	Aligned
Identity Binding	Ed25519 to node, owner field	OIDC/SPIFFE binding (P5), CA binding (P4)	Partial
Context Propagation	Handshake justification, tags	Tag-based policies (P3), external identity context (P5)	Partial
Revocation & Quarantine	Bilateral untrust with tunnel teardown	Cascading revocation (P3), block list (P3), cert revocation (P4)	Gap
SPIFFE/SPIRE Integration			
SPIFFE ID as identity	Not supported	JWT-SVID join rule (P5)	Gap
Trust domain federation	Not supported	Trust domain in network config (P5)	Gap
Workload API integration	Not supported	Auto-fetch SVIDs from SPIRE agent (P5)	Gap
Zero-Trust Patterns			
Dark-by-default	Private-by-default, resolve gating, backbone blocked	SYN trust gate (P1), hostname privacy (P1)	Mostly
Least-privilege access	Network membership as trust boundary	Tag-based per-port policies (P3)	Partial
Continuous verification	Trust checked at handshake time	Policy re-evaluation (P3)	Gap

Table 8: Standards alignment matrix: current capabilities vs. planned phases.