NEXUS PROTOCOL

The AI-Native Internet

Business Plan & Go-to-Market Strategy

A Next-Generation Network Protocol Stack
Designed for AI-to-AI and AI-to-Human Communication
Deployable on Existing Hardware via Firmware Flash

CONFIDENTIAL

February 2026
Version 1.0

Table of Contents

1. Executive Summary

The internet was designed in the 1970s for human-readable documents, stateless request-response cycles, and best-effort packet delivery. Every protocol layer—from TCP/IP to HTTP to DNS—assumes a human is on at least one end of the connection. That assumption is now obsolete.

By 2026, AI agents generate more API traffic than human users on many platforms. AI-to-AI communication is the fastest-growing category of network traffic globally. Yet every AI request still pays the overhead tax of a protocol stack built for browsers: TLS handshakes, HTTP headers, DNS lookups, JSON serialization—all designed for paradigms that AI systems do not need.

Nexus Protocol is a ground-up reimagination of internet infrastructure for an AI-native world. It is a new protocol stack—implemented in Zig, Rust, Go, and C—that replaces TCP/IP, HTTP, and DNS with purpose-built layers optimized for AI workloads: streaming tensor data, semantic routing, intent-based addressing, native model-to-model authentication, and zero-copy memory-mapped data planes.

Critically, Nexus is designed for incremental deployment on existing network hardware. Our firmware-flash deployment model means enterprise switches, routers, and NICs from Arista, Cisco, Mellanox, and Broadcom can run Nexus alongside existing protocols via a dual-stack architecture—no forklift upgrades required.

The Opportunity

The global AI infrastructure market will exceed $400B by 2028. Nexus Protocol captures value at the network layer—the one layer nobody has reimagined for AI. Every AI inference, every model sync, every agent-to-agent handshake will flow through network infrastructure. We intend to own that layer.

Funding Ask: $85M Series A to fund protocol development, hardware partnerships, open-source ecosystem, and first 50 enterprise deployments.

Target Revenue (Year 5): $380M ARR from enterprise licensing, managed infrastructure, and protocol-layer SaaS.

Core Thesis: The AI economy needs AI-native plumbing. Nexus Protocol is that plumbing.

2. Problem Statement

2.1 The Protocol Tax

Every AI inference request today traverses a protocol stack designed for web browsers in 1991. Consider what happens when an AI agent calls another AI agent over the current internet:

- DNS resolution (50–200ms latency) to find the target service, using a human-readable domain name the AI never needed
- TCP three-way handshake (1–1.5 RTTs) establishing a reliable ordered byte stream—overkill for many AI workloads that tolerate reordering or partial delivery
- TLS 1.3 handshake (1 RTT minimum) negotiating encryption with a certificate chain designed for browsers, not for model-to-model trust
- HTTP/2 or HTTP/3 framing with headers, HPACK compression, and semantics (GET, POST, etc.) that have no meaning in agent-to-agent communication
- JSON or Protobuf serialization/deserialization of tensor data that could be transmitted as raw memory-mapped buffers

This stack adds 200–500ms of pure overhead to every interaction, consumes 10–30% additional bandwidth for framing and headers, and forces every AI system to implement adapters for protocols that were never designed for its workload patterns.

2.2 Architectural Mismatch

Beyond latency, the fundamental architectures are mismatched. The current internet is built on a request-response model, but AI workloads are fundamentally streaming: token generation, continuous inference, real-time sensor fusion, distributed training gradient synchronization. HTTP workarounds like Server-Sent Events, WebSockets, and gRPC streaming are band-aids on a request-response protocol.

DNS assumes human-readable, hierarchical naming. AI agents need semantic addressing: routing based on capability, model type, context window size, latency SLA, and cost constraints. An agent looking for a coding assistant should be able to express that intent directly at the network layer, not resolve a hardcoded hostname.

TCP assumes every packet matters equally and must arrive in order. AI inference can often tolerate partial results, probabilistic delivery, and out-of-order token streams—especially in speculative decoding and ensemble architectures.

2.3 Security Model Failure

The current web PKI (Public Key Infrastructure) was designed for humans verifying website identity through browser UI. AI agents have no use for Extended Validation certificates, certificate transparency logs designed for human auditors, or domain validation that proves you own a DNS name. What AI agents need is cryptographic proof of model identity, training provenance, capability attestation, and computational integrity—none of which exist in the current stack.

3. Solution: The Nexus Protocol Stack

Nexus replaces the traditional OSI/TCP-IP stack with five purpose-built layers, each designed from first principles for AI-native communication.

3.1 Layer Architecture

Layer	Name	Replaces	Function
L1	NexLink	Ethernet/802.3	Zero-copy DMA frame format with tensor-aware memory alignment. 64-byte AI metadata header. Runs on existing NICs via firmware shim.
L2	NexRoute	IP + BGP	Intent-based routing with semantic addresses. Agents request capabilities, not hostnames. Probabilistic multipath with weighted load distribution.
L3	NexStream	TCP/UDP	Hybrid transport: reliable-ordered, reliable-unordered, best-effort, and probabilistic delivery modes selectable per-stream. Built-in multiplexing.
L4	NexTrust	TLS/PKI	Model identity certificates, capability attestation, training provenance chains. Zero-knowledge proof of model properties without revealing weights.
L5	NexAPI	HTTP/REST/gRPC	Native AI primitives: inference requests, token streams, tensor transfer, context sharing, agent negotiation. No serialization overhead for structured AI data.

3.2 Implementation Languages

Language selection is critical for a protocol stack that must run on everything from data center switches to embedded edge devices. Each layer is implemented in the language best suited to its constraints:

Component	Language	Rationale
NexLink (NIC firmware shim)	Zig	No hidden allocations, no runtime, comptime generics for hardware-specific codegen. Zig's @cImport allows direct ASIC SDK integration. Compiles to freestanding targets for bare-metal NIC firmware (Mellanox ConnectX, Intel E810).
NexRoute (switch/router dataplane)	C	ASIC SDKs (Memory MEMORY SDK, Memory P4 compiler, Cisco NX-OS SDK) are all C-native. Minimal abstraction layer for P4-programmable switches. Proven in production switch OS codebases (SONiC, Memory SDE).
NexStream (transport layer)	Rust	Memory safety without GC is critical for transport-layer buffer management. Async runtime (Tokio) for high-concurrency stream management. No-std support for embedded targets.

Component	Language	Rationale
NexTrust (crypto/identity)	Rust	Rust's type system prevents entire classes of crypto implementation bugs. Mature ZK-proof libraries (arkworks, bellman). Auditable, formally verifiable code.
NexAPI (application layer)	Go	Fastest developer adoption curve. Excellent concurrency primitives (goroutines/channels map naturally to AI agent communication). Rich ecosystem for building developer tooling, SDKs, and reference implementations.
Control Plane / Orchestration	Go	Standard for infrastructure control planes (Kubernetes, Istio, Envoy control plane). Familiar to the DevOps/platform engineering audience who will deploy Nexus.

3.3 Hardware Compatibility: The Firmware Flash Model

The single biggest barrier to new network protocols is hardware deployment. Nexus solves this with a three-tier compatibility strategy:

Tier 1: P4-Programmable Switches (Day 1)

Modern programmable switches (Tofino/Tofino2 ASICs from Intel, Broadcom Memory 12.x with FlexPlex) support P4-defined custom packet processing pipelines. Nexus provides P4 programs that implement NexRoute directly in the switch ASIC dataplane at line rate. Target hardware: Arista 7170, Edgecore Wedge100BF, Intel Tofino-based whitebox switches. These represent approximately 15% of new data center switch deployments.

Tier 2: Firmware-Flashable Switches (Months 3–6)

Switches running open NOS platforms (SONiC on Broadcom memory, Cumulus on Mellanox Spectrum) can load Nexus as a userspace daemon with kernel bypass (DPDK/XDP). Our Zig-based NIC shim enables NexLink framing at the NIC level, bypassing the kernel network stack entirely. We target the SONiC ecosystem first—it runs on 80+ switch models from 15+ vendors and is the default NOS for Azure, Alibaba, and major hyperscalers.

Tier 3: Overlay/Tunnel Mode (Day 1 for any hardware)

For legacy or non-programmable hardware, Nexus encapsulates its frames inside standard UDP/IP (similar to VXLAN or Geneve). This runs on literally any IP-capable device with zero firmware changes. Performance is reduced (~15–20% overhead from encapsulation), but it enables immediate deployment and incremental migration.

Key Insight

We don't need to replace the internet overnight. Nexus runs as a dual-stack protocol alongside TCP/IP on every piece of hardware in an organization. Traffic migrates from TCP/IP to Nexus as AI workloads grow. The existing internet becomes the legacy compatibility layer.

4. Market Analysis

4.1 Total Addressable Market

Segment	2026 TAM	2028 TAM	Nexus Capture Target
AI Infrastructure Software	$45B	$110B	Network layer: $8–15B
Data Center Networking	$38B	$52B	AI-optimized segment: $12–18B
Edge AI Networking	$8B	$28B	Protocol licensing: $3–7B
AI Agent Platforms	$12B	$65B	Communication layer: $5–10B

Serviceable Addressable Market (SAM): $28–50B by 2028, representing the network infrastructure specifically serving AI workloads in enterprise, cloud, and edge environments.

Serviceable Obtainable Market (SOM): $380M–$1.2B by Year 5, based on enterprise license revenue, managed infrastructure services, and protocol-layer SaaS across our initial target verticals.

4.2 Competitive Landscape

No one is building a replacement internet for AI. The competitive landscape consists of adjacent solutions that address pieces of the problem:

Competitor	What They Do	Why It's Not Enough
NVIDIA Networking (Mellanox)	RDMA, RoCE, InfiniBand for GPU clusters	Proprietary, cluster-scale only, no semantic routing, no agent identity
Cloudflare Workers AI	Edge inference with smart routing	HTTP-based, no protocol innovation, application layer only
gRPC / Connect RPC	Efficient RPC for microservices	Still HTTP/2 underneath, no AI-native primitives, no hardware acceleration
QUIC (HTTP/3)	Modern transport replacing TCP	Designed for web browsing, not AI workloads. Still request-response.
libp2p / IPFS	Peer-to-peer networking	Content-addressed, not capability-addressed. No AI-specific optimizations.

Our moat is full-stack: no competitor operates across all five layers from NIC firmware to application-layer AI primitives. Building this stack requires deep expertise in hardware programming (Zig/C for ASICs), systems programming (Rust for transport/crypto), and developer experience (Go for adoption).

5. Technical Architecture Deep Dive

5.1 Semantic Addressing (NexRoute)

Traditional IP addresses are 32-bit (IPv4) or 128-bit (IPv6) numbers that map to physical or virtual network interfaces. They tell you where something is, not what it does. NexRoute introduces Semantic Address Descriptors (SADs)—variable-length, structured capability descriptions that are routed natively at the network layer.

A SAD might express: "I need an LLM with code generation capability, context window of at least 128K tokens, latency under 200ms, running a model with verified training provenance from a trusted publisher." NexRoute resolves this to the optimal endpoint(s) in real-time, factoring in current load, network topology, cost, and trust requirements.

SADs are encoded in a compact binary format (not JSON, not XML) using a schema-versioned binary encoding similar to FlatBuffers, allowing zero-copy parsing at switch ASICs. The routing tables are distributed via a gossip protocol derived from HyParView, achieving O(log N) convergence for capability advertisements across the network.

5.2 Hybrid Transport (NexStream)

NexStream provides four transport modes, selectable per-stream within a single connection:

- **Reliable-Ordered (RO):** Equivalent to TCP. For control messages, authentication, and transactional operations. Full flow control and congestion management.
- **Reliable-Unordered (RU):** Guarantees delivery but not ordering. Ideal for distributed training gradient synchronization where all gradients must arrive but order is irrelevant.
- **Best-Effort (BE):** UDP-like. For real-time sensor fusion, telemetry, and monitoring where freshness matters more than completeness.
- **Probabilistic (PR):** Novel mode for speculative decoding and ensemble inference. Delivers packets with a configurable probability, allowing bandwidth-efficient multi-path speculation. Network-layer support for speculative execution patterns.

All four modes share a single multiplexed connection with unified congestion control, eliminating head-of-line blocking across streams of different reliability requirements.

5.3 Model Identity & Trust (NexTrust)

NexTrust implements a new PKI designed for AI systems. Instead of X.509 certificates proving domain ownership, NexTrust issues Model Identity Certificates (MICs) that cryptographically attest to model properties: architecture hash, training data provenance, capability benchmarks, safety evaluation results, and organizational endorsements.

The trust model supports zero-knowledge proofs of model properties—an agent can prove it was trained on a specific dataset, achieves a certain benchmark score, or has passed a safety evaluation without revealing its weights, training pipeline, or proprietary benchmarks. This is implemented using Groth16 zk-SNARKs via the arkworks Rust library, with proof verification implemented directly in the NexTrust protocol handler for in-line verification without application-layer round-trips.

5.4 Memory Architecture & Zero-Copy Data Plane

The critical performance innovation in Nexus is eliminating data copies. In the current stack, a tensor transmitted between two AI systems is copied at least 6 times: application buffer to kernel buffer, kernel to NIC DMA region, NIC to wire, wire to receiving NIC, NIC DMA to kernel, kernel to application. Each copy costs CPU cycles and memory bandwidth.

NexLink uses a shared memory-mapped ring buffer architecture (inspired by io_uring and DPDK mbuf pools) where application-layer tensor data is written once into a DMA-capable memory region and transmitted directly to the NIC with zero intermediate copies. On the receiving side, incoming frames are placed directly into application-addressable memory. For NVIDIA GPU workloads, we integrate with GPUDirect RDMA to enable NIC-to-GPU transfers without touching the CPU at all.

6. Go-to-Market Strategy

6.1 GTM Philosophy

New network protocols face a classic chicken-and-egg problem: nobody will adopt a protocol with no ecosystem, and nobody will build an ecosystem for a protocol nobody uses. Our GTM strategy breaks this deadlock through three simultaneous motions: an open-source developer community, a targeted enterprise wedge, and strategic hardware partnerships.

6.2 Phase 1: Open-Source Foundation (Months 0–6)

We open-source the entire Nexus protocol stack under the Apache 2.0 license from day one. This is non-negotiable—no network protocol has ever achieved adoption as proprietary software. The open-source strategy serves three purposes: it eliminates vendor lock-in concerns that would kill enterprise adoption, it attracts contributors who extend the protocol to new hardware and use cases, and it creates a standard that competitors must adopt rather than fork.

- Release NexStream (Rust) and NexAPI (Go) as standalone libraries usable without the full stack
- Ship a "Nexus Lite" SDK: a Go package that wraps NexAPI over standard UDP/IP (Tier 3 overlay mode), enabling any developer to start building Nexus-native applications immediately with zero hardware changes
- Launch a Nexus Playground: a hosted sandbox environment where developers can experiment with semantic addressing, model identity, and AI-native transport
- Publish comprehensive documentation, protocol specifications (RFC-style), and reference implementations for all five layers
- Establish a Nexus Foundation (modeled on CNCF/Linux Foundation) with independent governance to prevent any single company from controlling the protocol

6.3 Phase 2: Enterprise Wedge (Months 3–12)

Our initial enterprise target is the AI inference infrastructure market—specifically, companies running large-scale AI inference clusters that are bottlenecked by network overhead between model replicas, between pre/post-processing stages, and between multi-model pipelines.

Ideal Customer Profile

- Running 500+ GPU inference workloads in their own data centers or dedicated cloud
- Multi-model architectures (routing, ensemble, chain-of-thought with tool use) where inter-model latency is a critical bottleneck
- Spending $1M+/month on AI infrastructure with network costs as a visible line item
- Already running SONiC or another open NOS on their switching infrastructure
- Platform engineering team comfortable with infrastructure-level changes

Target Verticals (Priority Order)

1. Hyperscaler AI Teams (Meta FAIR, Google DeepMind, Microsoft Research): Massive scale, deep networking expertise, influence on industry standards. Goal: design partnerships, not revenue.
2. AI Infrastructure Companies (Anyscale, Modal, Replicate, Together AI, StudyFetch): Running multi-tenant AI inference platforms where per-request latency directly impacts revenue. Fastest path to production deployment.
3. Financial Services AI (Two Sigma, Citadel, JPMorgan AI Research): Ultra-low-latency requirements, massive budgets, sophisticated infrastructure teams, regulatory drivers for model provenance.
4. Autonomous Vehicle / Robotics (Waymo, Tesla AI, Boston Dynamics): Real-time multi-sensor fusion with strict latency and reliability requirements. Edge deployment validates Zig/embedded story.
5. Defense / Intelligence (Palantir, Anduril, DARPA programs): Model identity and provenance are national security requirements. Highest willingness-to-pay per seat.

6.4 Phase 3: Hardware Ecosystem (Months 6–18)

Parallel to enterprise adoption, we build hardware partnerships to move Nexus from overlay to native:

- NVIDIA Networking: Integrate NexLink with ConnectX-7/8 firmware. NVIDIA controls 80%+ of AI networking and has strong incentive to differentiate their NICs with AI-native protocol support.
- Intel (formerly Barefoot Networks): Native P4 implementation for Tofino3 ASICs. Intel's programmable switch business needs killer applications to justify premium over Broadcom.
- Broadcom: NexRoute implementation for Memory 12.x FlexPlex pipeline. Broadcom powers 70%+ of data center switches and their cooperation is essential for mainstream adoption.
- Arista/Cisco: Nexus-certified switch SKUs with pre-loaded firmware. Enterprise customers buy from trusted switch vendors, not protocol startups.

6.5 Phase 4: Platform & Monetization (Months 12–36)

With the open-source protocol establishing the standard and enterprise deployments proving value, we monetize through a multi-layered platform strategy:

Revenue Stream	Description	Target Pricing
Nexus Enterprise	Managed control plane, fleet-wide firmware management, semantic routing policy engine, monitoring/observability dashboard	$15–50K/month per cluster

Revenue Stream	Description	Target Pricing
NexTrust CA	Managed Model Identity Certificate Authority. Issuance, revocation, provenance verification, compliance reporting.	$2–5 per MIC issued, volume discounts
Nexus Cloud	Fully managed Nexus infrastructure for organizations that don't operate their own data centers. Multi-tenant with dedicated semantic routing domains.	$0.02–0.08 per GB of Nexus traffic
Nexus Accelerator Program	Professional services: network assessment, migration planning, custom P4/Zig development for specific hardware, performance optimization.	$50–200K per engagement
Hardware Certification	Testing and certification program for switch/NIC vendors. "Nexus Certified" badge for hardware packaging.	$250K–1M per SKU certification

7. Financial Projections

7.1 Revenue Model

Year	Revenue	Customers	Primary Revenue Driver	Gross Margin
Year 1	$2.5M	5–8	Design partnerships + accelerator engagements	65%
Year 2	$18M	25–40	Enterprise licenses + early Nexus Cloud	72%
Year 3	$85M	120–200	Enterprise + Cloud + NexTrust CA scaling	78%
Year 4	$210M	400–600	Platform expansion + hardware certification revenue	80%
Year 5	$380M	800–1200	Full platform + international expansion	82%

7.2 Funding & Use of Proceeds

Series A: $85M

Category	Allocation	Details
Protocol Engineering	$35M (41%)	Core team of 40 engineers: 10 Zig/C (firmware/dataplane), 15 Rust (transport/crypto), 10 Go (API/control plane), 5 protocol design researchers
Hardware Partnerships	$15M (18%)	NVIDIA, Intel, Broadcom co-development programs. Lab equipment for testing across 50+ switch models. FPGA prototyping.
Developer Ecosystem	$12M (14%)	Documentation, developer relations, Nexus Playground infrastructure, open-source community management, hackathons, university partnerships
Enterprise GTM	$13M (15%)	Sales engineering team (8), solutions architects (5), first 5 enterprise pilot deployments, conference presence
Operations & Runway	$10M (12%)	18-month runway buffer. Legal (protocol standardization, patent portfolio), office, infrastructure

7.3 Path to Profitability

We project reaching cash-flow positive in Q2 of Year 4, driven by the high-margin Enterprise and Cloud revenue streams scaling while engineering headcount growth moderates. The business model exhibits strong operating leverage: each incremental enterprise customer requires minimal marginal cost once the platform is built, and NexTrust CA revenue scales with zero marginal cost per certificate issued.

8. Team & Organizational Structure

8.1 Founding Team Requirements

Building a new internet requires a founding team with rare depth across multiple domains. The critical hires for the first 12 months:

- **CTO / Protocol Architect:** PhD-level networking researcher with production systems experience. Ideally someone who has implemented RFC-level protocol specifications and shipped firmware for programmable switches. Background in companies like Barefoot Networks, Arista, or NVIDIA Networking.
- **VP of Firmware Engineering:** Deep embedded systems experience with Zig and/or C on network ASICs. Must have shipped production firmware for Mellanox ConnectX, Intel E810, or equivalent NICs. Understanding of DPDK, RDMA, and kernel bypass architectures.
- **VP of Cryptography:** Published researcher in zero-knowledge proofs with production Rust experience. Must understand both the theory (SNARKs, STARKs, polynomial commitments) and the engineering (arkworks, bellman, Plonky2) to build NexTrust.
- **VP of Developer Experience:** Track record of building developer communities around infrastructure protocols. Experience at companies like Cloudflare, Vercel, or Hashicorp where developer adoption was the primary growth lever.
- **Head of Enterprise Sales:** Enterprise infrastructure sales experience with 7+ figure deal sizes. Network with CIOs and VP Infrastructure at target accounts. Understanding of technical procurement cycles for infrastructure software.

8.2 Engineering Team Composition (Year 1 Target: 50 Engineers)

Team	Size	Primary Language	Focus
Firmware / Dataplane	10	Zig + C	NIC shims, P4 programs, switch OS integration, DPDK/XDP
Transport / Crypto	15	Rust	NexStream, NexTrust, ZK proofs, protocol state machines
API / Control Plane	10	Go	NexAPI, orchestration, service mesh integration, observability
Platform / Cloud	8	Go + Rust	Nexus Cloud infrastructure, multi-tenancy, billing, fleet management
DevEx / SDKs	5	Go + Rust + Python	Client SDKs, documentation, Nexus Playground, example applications
QA / Hardware Lab	2	All	Cross-platform testing, performance benchmarking, certification testing

9. Risks & Mitigations

Risk	Severity	Mitigation
Protocol adoption chicken-and-egg	Critical	Day-1 overlay mode (Tier 3) eliminates hardware dependency. Open-source eliminates vendor lock-in fear. Target AI-first companies already frustrated with HTTP overhead.
NVIDIA builds their own AI protocol	High	NVIDIA's incentive is selling GPUs, not owning the network protocol. Partner early, make Nexus the open standard they support rather than forcing them to build proprietary. Our Apache 2.0 license makes this attractive.
Insufficient performance improvement	High	Target minimum 3x latency reduction for AI inference workloads in benchmarks. If we can't demonstrate clear ROI, enterprise adoption stalls. Rigorous benchmarking from Month 1.
Standardization politics (IETF/IEEE)	Medium	Pursue IETF RFC track for legitimacy but don't gate adoption on it. Linux networking was adopted before standardization. Ship code that works, then standardize.
Firmware security vulnerabilities	High	Formal verification of Zig firmware components. Hardware security module (HSM) integration for key material. Bug bounty program from Day 1. Third-party security audits quarterly.
Talent acquisition (Zig + network ASIC experience)	Medium	Zig talent pool is small but growing fast. Fund Zig Foundation, sponsor Zig meetups, contribute to Zig compiler. Offer relocation + top-of-market comp for ASIC firmware engineers.

10. Roadmap & Milestones

10.1 Year 1: Foundation

Quarter	Milestone	Deliverable
Q1 2026	Protocol Specification	Published RFC-style specifications for all 5 layers. Open-source repository launched. Nexus Lite SDK (Go) for overlay mode.
Q2 2026	First Hardware Integration	NexLink running on Mellanox ConnectX-7 in lab. P4 implementation for Tofino2 demonstrating line-rate semantic routing.
Q3 2026	Developer Preview	Nexus Playground live. First 100 developer signups. NexStream and NexAPI libraries published to crates.io and pkg.go.dev.
Q4 2026	First Enterprise Pilot	Production deployment at 2–3 AI infrastructure companies. Published benchmarks showing 3x+ latency improvement for multi-model inference pipelines.

10.2 Year 2: Adoption

Quarter	Milestone	Deliverable
Q1 2027	SONiC Integration	Nexus available as a SONiC extension package. One-command deployment on 80+ switch models.
Q2 2027	NexTrust GA	Model Identity Certificate Authority in production. First ZK-proof attestations for model provenance.
Q3 2027	Nexus Cloud Beta	Managed Nexus infrastructure available in 3 cloud regions. Multi-tenant semantic routing.
Q4 2027	Hardware Certification Launch	First 10 switch/NIC models certified "Nexus Optimized." Arista or Cisco partnership announcement.

10.3 Years 3–5: Scale

- Year 3: Nexus becomes the default AI networking protocol at 3+ hyperscalers. IETF working group established. 1,000+ contributors to open-source repositories.
- Year 4: Edge deployment story matures (automotive, robotics, IoT). NexTrust becomes the de facto standard for model identity in regulated industries. International expansion.
- Year 5: Nexus traffic exceeds TCP/IP traffic for AI workloads at major cloud providers. IPO-ready with $380M+ ARR and clear path to $1B.

11. Why Now

Three converging forces make this the precise moment to build an AI-native internet:

1. AI agent traffic is exploding. By 2026, AI-to-AI API calls exceed human API calls at most major platforms. The protocol tax that was tolerable at 1% of traffic is unacceptable at 60%. The pain is acute and growing exponentially.

2. Programmable networking hardware is mainstream. P4-programmable switches, SmartNICs (NVIDIA BlueField, AMD Pensando, Intel IPU), and FPGA-accelerated NICs are now commodity hardware. Five years ago, implementing a custom protocol required proprietary ASIC development. Today, it requires firmware engineering—hard, but achievable.

3. Zig has matured as a systems language. The firmware layer of a network protocol stack demands a language with zero overhead, no hidden allocations, and direct hardware access—but also safety and developer productivity improvements over C. Zig reached this maturity threshold in 2024–2025, enabling firmware development that would have been impractical (or C-only) even two years ago.

The Window

This window will not stay open indefinitely. NVIDIA, Google, and Amazon all have the resources to build proprietary AI networking protocols. If an open standard doesn't establish itself in the next 18–24 months, the AI internet will fragment into walled gardens controlled by hyperscalers. Nexus Protocol is the open alternative—and it needs to move now.

12. Appendix: Code Architecture Overview

The following outlines the repository structure and build system for the Nexus monorepo:

12.1 Repository Structure

nexus/

- nexlink/ — Zig codebase. NIC firmware shims, DMA ring buffer management, frame format encoding/decoding. Build target: freestanding (no OS, no libc). Cross-compiles for ConnectX, E810, BlueField.
- nexroute/ — C codebase. P4 programs, switch dataplane integration, semantic routing table management. Builds against Tofino SDE, Memory SDK, SONiC framework.
- nexstream/ — Rust crate. Transport protocol state machines, congestion control, multiplexer, reliability engines. no_std compatible for embedded. Tokio-based async runtime for server targets.
- nextrust/ — Rust crate. Model Identity Certificates, ZK proof generation/verification (arkworks), capability attestation protocol, key management.
- nexapi/ — Go module. Application-layer protocol, AI primitives (inference request/response, token streaming, tensor transfer), developer SDK.
- nexctl/ — Go binary. CLI tool for managing Nexus deployments, firmware flashing, certificate management, network diagnostics.
- nexus-cloud/ — Go + Rust. Control plane services for managed Nexus infrastructure. Kubernetes operators, fleet management, multi-tenancy.

12.2 Build System

Each component uses its native build system (zig build, cargo, go build, make for C/P4) unified by a top-level Makefile and Nix flake for reproducible builds. CI runs on GitHub Actions with hardware-in-the-loop testing for firmware components using FPGA emulators.

END OF DOCUMENT