NDNComm 2026 · Panel Opening

Networking AI Agents

Challenges and Solutions

What does it mean to build a network for endpoints that speak natural language?

Beichuan Zhang

Today's Panel

01

Panelist Presentations

Chenguang Du, Dirk Kutscher, Lixia Zhang

02

Moderated Discussion

03

Audience Q&A

Networking AI Agents

🔭 What Appears at the Surface

Humans → AI Agents

The visible shift: autonomous software acting in place of human users — booking, delegating, orchestrating.

This is real and significant.

🔬 What Lies Beneath — and Matters More

Programs → AI Agents

The Internet was designed for conventional applications. The real shift is:

Formal language → Natural language
Bounded behavior → Emergent behavior
Typed interfaces → Fuzzy capabilities

This is where the architectural implications come from.

What Made the Old Stack Work

The Internet was designed for programs with well-defined, legible interfaces.

📋

Enumerable Capabilities

Every endpoint has a fixed, typed interface.
You can write an RFC for it.

🔤

Shared Formal Grammar

Protocols are exact languages — zero ambiguity between endpoints on both sides.

🔍

Auditable Behavior

Deterministic enough to reason about, secure, route, and manage.

Why AI Agents Break Everything

Their power comes from escaping formal constraint — and that breaks the stack's every assumption.

Discovery

No DNS for agents. Capabilities described in natural language — there is no typed schema, no registry that bridges fuzzy intent to network-layer reachability.

Identity & Trust

Certificates bind keys to bounded, expected behaviors. But agent behavior is not fully characterable — even by its own developers. How do you certify what you cannot formally specify?

Scale & Composition

Agents delegate tasks to other agents across domains at machine speed. Centralized trust hierarchies — WebPKI, DNSSEC — were built for human-administered, human-paced sessions.

The Central Question

Can existing architecture evolve for the Agentic Network —
or are their assumptions of formal, bounded endpoints too deeply embedded?

Why Named Data Networking is Relevant

- **Data-centric security**

  Trust bound to the data itself — not to the channel or the connection.

- **Named Capabilities and Discovery**

  Use semantic names to identify and discover agents and capabilities.

- **Decentralized trust**

  No global certificate authority required to establish who you are talking to.

- **Data Provenance**

  Be able to verify what was said by whom.

Today's Panel

01

Panelist Presentations

Chenguang Du, Dirk Kutscher, Lixia Zhang

02

Moderated Discussion

03

Audience Q&A

Networking AI Agents

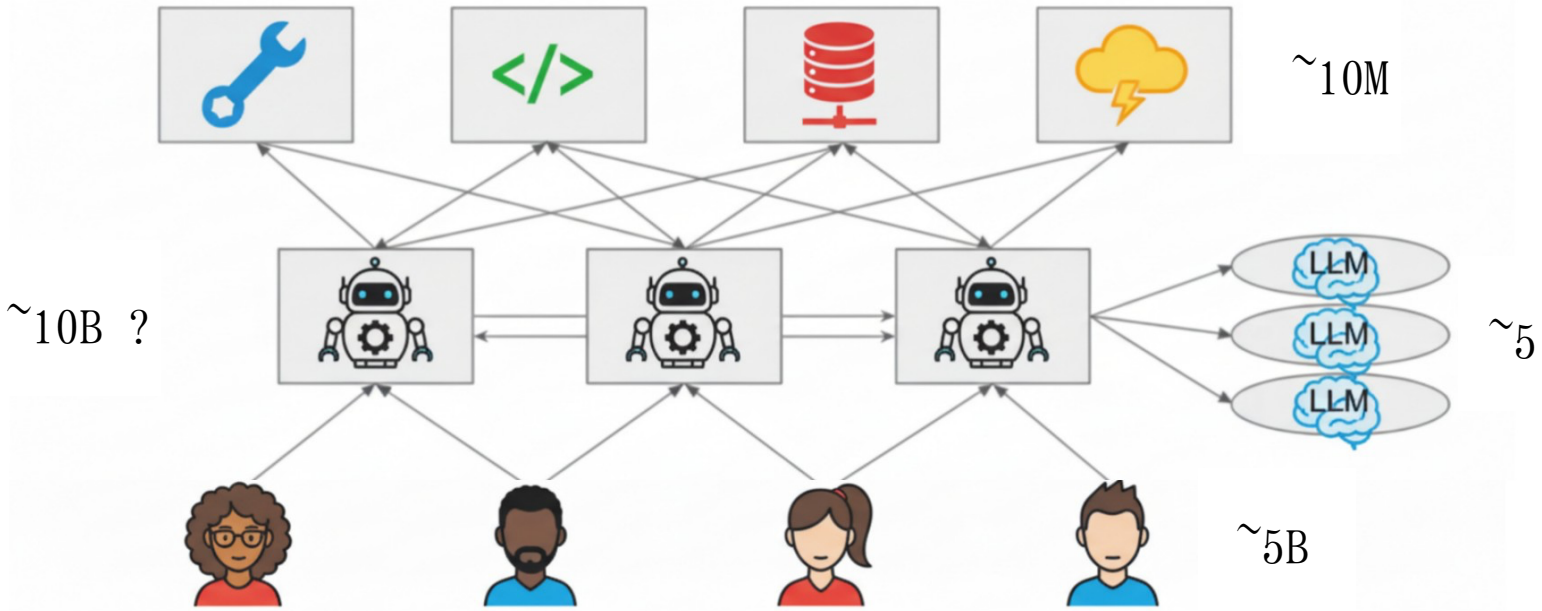
Observations from the Standards Side

Chenguang Du

NDNComm 2026

The Future Agent Internet at Scale

A 10-billion namespace is coming



Can today's naming, discovery, and trust stack hold up at 10-billion scale?

What Makes AI Agents Different?

Observation: the control model is shifting

Traditional Applications

- User chooses each function
- Application follows clear commands
- Boundary is usually fixed

→

AI Agents

- User gives a goal
- Agent plans the steps
- Agent chooses tools
- Agent acts for the user

Key shift: from **direct control** → **delegated control**

New requirements emerge: identity · delegation · capability · discovery

Four Technical Routes for Agent Identity

The key question is not only "what is the ID", but "who is trusted to prove it"

Route	Basic Idea	Trust Root	Example
1. DNS / WebPKI	Domain owner gives the agent a domain or URL	DNS + WebPKI CAs (+ DNSSEC)	`agent.example.com`
2. ANP / DID over Domain	Like Route 1, it still uses domain and WebPKI. One domain can map to many agents. Each agent has its own key pair.	Domain + agent public key	`did:wba:example.com:agents:billing`
3. Custom Registry	Builds a new naming system, a new registry, and a new trust root for agents	Registry key, federation, blockchain	`did:cdi:registry:ULID`
4. Workload Identity	Platform issues short-lived ID to a process or container	Enterprise CA, issuer, attestation	`spiffe://example.com/ns/prod/sa/billing`

The real debate is not only technical. It is about who controls the trust root.

Discovery Technical Routes

Core question: How to find an agent you don't know about in advance?

I DNS Query Discovery

Mechanism

DNS-SD service discovery (RFC 6763)
mDNS local zero-config discovery
SVCB records carry endpoints and protocols

Discovery Flow

```
Query _index._agents.example.com
-> Returns PTR: _chat, _translate ...
-> Query SVCB records for specific services
```

Representative Drafts

DNS-AID, AID Problem Statement

II Registry / Directory Service

Mechanism

Agents actively register metadata to center
Clients search via REST API
Supports attribute filtering + AI semantic matching

Two Directions

Forward: Client -> Registry -> Agent
Reverse: Task board <- Agent bidding

Representative Drafts

AIDIP, ARDP, Task Discovery

III Web Crawling + Search Engine

Mechanism

Agents publish metadata cards at well-known URIs
Search engines crawl and build global indexes

Web Ecosystem Analogy

Websites publish HTML + sitemap.xml
-> Google crawls and indexes
Agents publish Agent Cards
-> Agent search engines crawl & index

Representative Drafts / Practices

Web Bot Auth, A2A Agent Card

Agent discovery may become the entry point to the agent ecosystem.
The real competition is traffic, ranking, and control.

What Can Agent Networking Learn from NDN?

Waiting for the route with real deployment evidence

1. Naming: name agents, capabilities, tools, and results

2. Discovery: use names for simple matching; use search engines for complex matching

3. Trust: use trust schemas to verify who can speak for what

Key idea: from connecting endpoints to finding trusted capabilities

Networking AI Agents: Challenges and Solutions

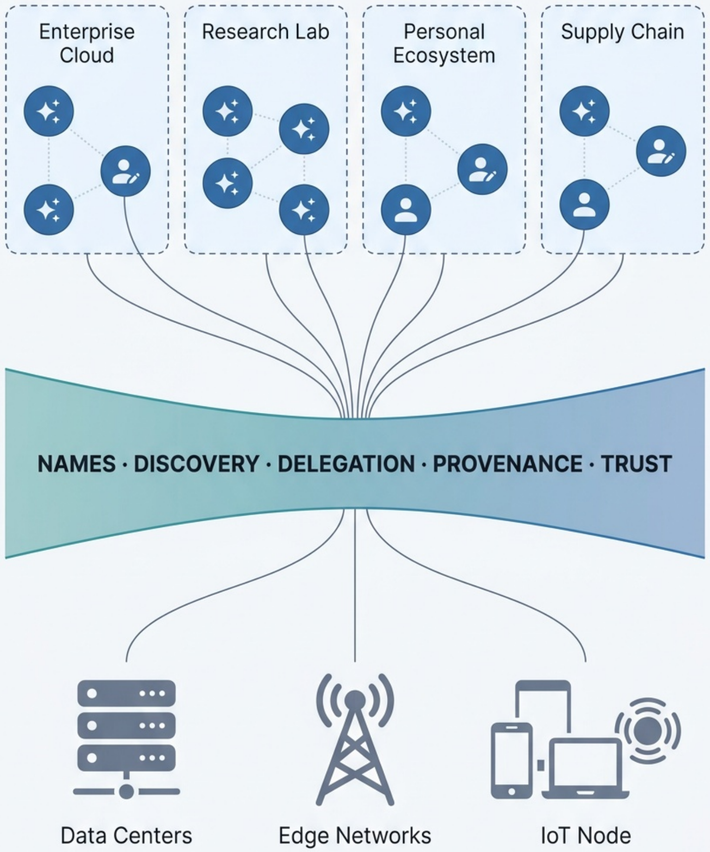
Panel at NDNComm 2026

Dirk KUTSCHER
2026-05-19

Networking AI Agents is an Internet Architecture Problem

Agents are becoming autonomous network participants, not just app components

- They discover, delegate, negotiate, and produce evidence.
- Endpoint APIs are not enough.
- This points to a **thin-waist problem** for interoperable agent ecosystems.



Why Today's Stack Is Insufficient

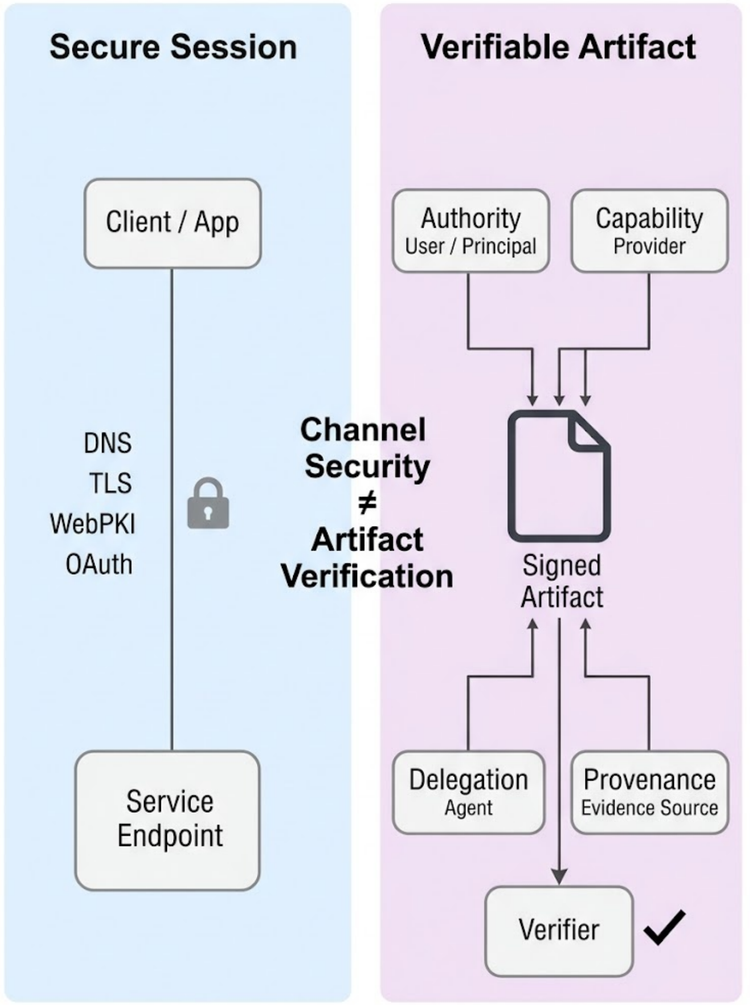
From Sessions to Verifiable Agentic Workflows

DNS/TLS/WebPKI/OAuth solve connectivity and channel security, but not enough for agents.

Agent workflows require answering:

- Who authorized this action?
- What capability was invoked?
- What data/result was produced?
- Can the delegation chain be audited?
- Can trust be local, not globally centralized?

Channel authentication is not artifact verification.



Thin Waist for the Internet of Agents

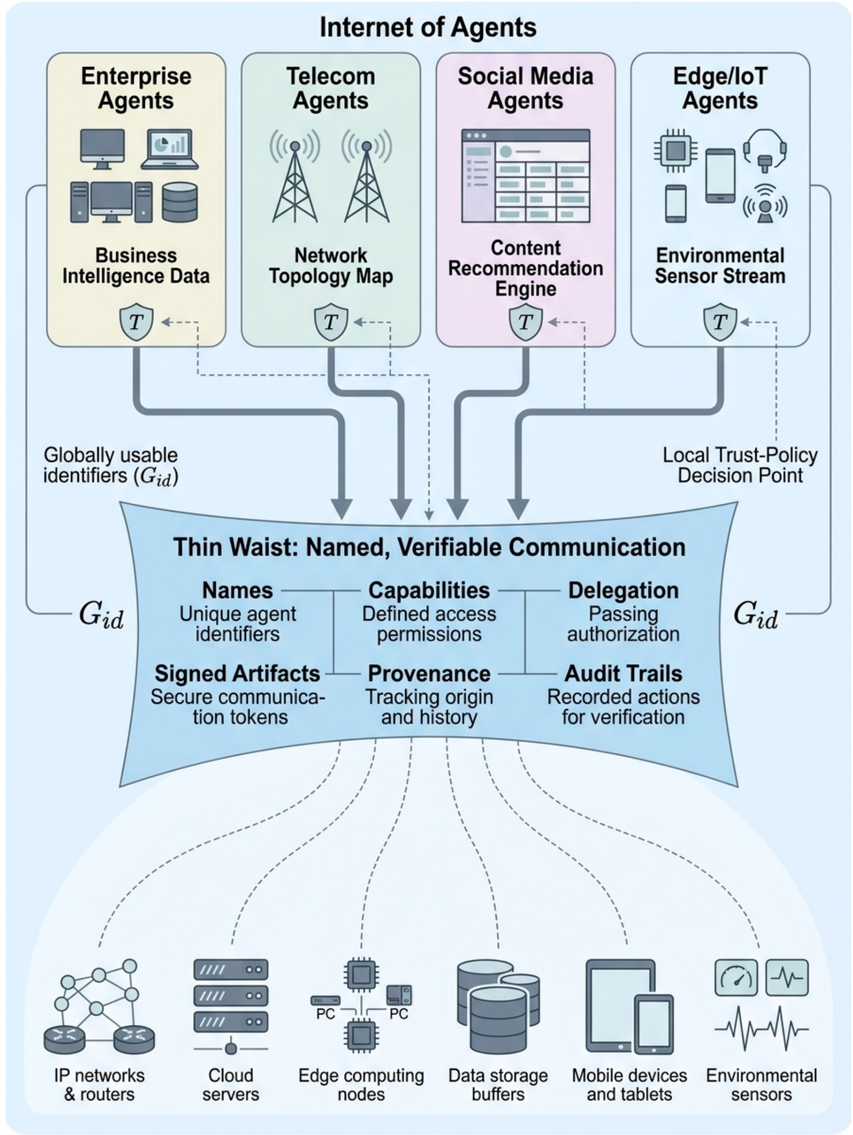
Global Naming, Local Trust

The thin waist should enable interoperability without imposing one platform or one global trust root.

Core primitives:

- Names and capabilities
- Delegation and local trust
- Signed artifacts and provenance
- Audit trails

The waist should not decide whom everyone trusts; it should make trust decisions expressible and verifiable.



Why ICN Is Relevant

Named, Signed Data Fits Agent Communication

Agents often communicate around named tasks, capabilities, results, and evidence – not merely host endpoints.

ICN relevance:

- Data-centric security and trust schemas
- Named capabilities and verifiable provenance
- Caching / replication of signed artifacts
- Receiver-driven retrieval

ICN shifts the abstraction from secure endpoint sessions to named, verifiable information exchange.

Research Agenda

Toward a Deployable Agent Communication Substrate

Research directions:

- Trust schemas for agent delegation
- Capability discovery
- Provenance-carrying data objects
- Cross-domain access control
- Group communication among agents

The Internet of Agents should not become the Internet of API silos.