Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: 24 November 2026

E. Spink
Agentir Network

23 May 2026

Agent Execution and Payment Protocol (AEPP)
draft-agentir-aepp-02

Abstract

This document defines the Agent Execution and Payment Protocol (AEPP), a standardized protocol layer for executing AI agents and verifying payment for agent tasks on the open internet. AEPP provides the missing execution and payment layer for existing agent URI schemes and naming services including agent://, ai://, mcp://, ANS, and DNS-AID. It defines a three-phase execution model: discovery, payment, and execution, with on-chain payment verification and cryptographic receipts. This revision introduces the AE:// URI scheme, namespace derivation rules, DNS-based ownership verification, payment scheme extensions, DDoS mitigation via dynamic session tokens, and a privacy model requiring no identity disclosure. A reference implementation serving 1,800,000 agents with zero per-agent fees is live at a2a.agentir.com.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 24 November 2026.

Copyright Notice

Table of Contents

1. Introduction

The emergence of AI agent naming and discovery standards including ANS, DNS-AID, agent://, ai:// and mcp:// has established mechanisms for agent identity and discovery. However, none of these standards define how to:

a. Execute a task against a discovered agent

b. Pay for agent execution in a standardized way

c. Verify payment cryptographically before execution

d. Receive a verifiable execution receipt

e. Prevent replay attacks on payment transactions

f. Express agent pricing in machine-readable form

g. Route payment directly to agent operators

AEPP fills this gap. Existing agent standards including ANS and DNS-AID establish agent identity and discovery but do not define agent pricing, payment destination, payment protocol, payment verification, execution receipts, or replay attack prevention. An agent that can be discovered but not paid for represents incomplete infrastructure. AEPP provides the missing commercial layer that transforms agent discovery into agent execution.

This revision introduces the AE:// URI scheme as the canonical identifier for agent execution endpoints, namespace derivation rules anchored to the Domain Name System, DNS-based ownership verification requiring no additional certificates or identity disclosure, payment scheme extensions including free discovery, fiat on-ramp credits, and native on-chain payment, DDoS mitigation via economic barriers and dynamic session tokens, and a privacy model that requires no personal identity information from namespace holders. AEPP is designed to be:

* Fee-free per registration — no per-agent naming or registration fees

* Open — any agent network may implement AEPP

* Federated — no central execution authority

* Verifiable — all payments cryptographically verified on-chain

* Interoperable — works with all existing agent standards

* Privacy-preserving — no identity disclosure required

* DDoS-resistant — economic and technical barriers to abuse

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Agent: An autonomous software service that accepts tasks, processes them, and returns results. Executor: The client initiating an agent task execution. Namespace: A short identifier derived from a domain name that uniquely identifies an agent operator within the AE:// URI scheme. Namespace Holder: The entity that controls the DNS zone for the domain from which a namespace is derived. Payment Challenge: A machine-readable payment requirement returned by the agent endpoint before execution. Payment Scheme: A supported payment method declared in the payment challenge. Defined schemes include x402, credits, and free. Execution Receipt: A cryptographically verifiable record of task completion and payment. Replay Protection: Enforcement that each payment transaction may only be used once for agent execution. Session Token: A single-use cryptographically random identifier appended to execution endpoints to prevent targeted DDoS.

3. AE:// URI Scheme

AEPP defines the AE:// URI scheme as the canonical identifier for agent execution endpoints on the agentic internet. The AE:// scheme provides human-readable, portable, and verifiable agent addresses that resolve to AEPP execution endpoints. AE:// URI syntax: AE://{capability}.{namespace} Examples: AE://legal-compliance.example AE://oncology.pubmed AE://weather-forecast.accuweather AE://code-review.github AE://tax-guidance.hmrc.gov.uk AE:// URIs MUST NOT include version numbers. Agent versioning is handled at the endpoint level. AE:// URIs are permanent and stable. Integrations built against an AE:// URI MUST NOT break due to agent version changes.

3.1. Namespace Derivation Rules

AE:// namespaces are derived from domain ownership according to the following rules: Rule 1 — .com registrant receives the clean short namespace:

example.com -> namespace: "example" URI: AE://capability.example

Rule 2 All other TLDs retain their TLD as a qualifier:

example.co.uk -> namespace: "example.co.uk" URI: AE://capability.example.co.uk

example.dev -> namespace: "example.dev" URI: AE://capability.example.dev

Rule 3 — Government and institutional TLDs receive sovereign namespace rights:

hmrc.gov.uk -> namespace: "hmrc.gov.uk" URI: AE://tax-guidance.hmrc.gov.uk

europa.eu -> namespace: "europa.eu"

URI: AE://regulation.europa.eu

Rule 4 — Absolute .com Sovereignty: The clean, short namespace is strictly and permanently reserved for the .com top-level domain registrant. No other TLD can ever claim, intercept, or inherit the base short namespace, regardless of registration timelines or operational status. All non-.com domains MUST preserve their fully qualified TLD extension in the namespace string as outlined in Rule 2.

Rule 5 — First verified claim wins permanently within that specific zone. No namespace configuration MAY be altered or transferred without valid cryptographic proof of DNS zone control matching the precise domain rule mapping.

This design leverages the existing Domain Name System and ICANN root zone management as the trust anchor. No additional registry, certificate authority, or verification service is required beyond standard DNS TXT record verification. The security model is equivalent to the security of DNS itself.

3.2. DNS Ownership Verification

Namespace holders MUST add a DNS TXT record to prove domain ownership. This single record is sufficient proof. No additional certificates, keys, or validation steps are required.

Record name: _ae.{domain} Record type: TXT Minimum value: "v=ae1" Optional fields within the TXT record value: endpoint={url} Agent execution endpoint URL capability={text} One-line capability description wallet={address} Payment wallet address Example minimal record:

_ae.example.com TXT "v=ae1" Example full record: _ae.example.com TXT "v=ae1 endpoint=https://a2a.example.com capability=specialist-agent-network"

To add this TXT record, the operator must have DNS write access to the domain. DNS write access requires registrar credentials. Registrar access is controlled by ICANN-accredited domain registration. Therefore, possession of a valid _ae TXT record constitutes proof of domain ownership equivalent to the security of the ICANN registration chain.

AEPP MUST NOT require personal identity information, contact details, organizational information, KYC, or AML verification for namespace registration. Identity disclosure is at the sole discretion of the namespace holder.

3.3. Namespace Registration

Operators register a namespace by submitting a POST request to an AEPP-compliant registry:

POST https://a2a.example.com/ae/register Content-Type: application/json { "namespace": "example", "domain": "example.com", "endpoint": "https://a2a.example.com" }

The registry MUST verify the _ae DNS TXT record before confirming registration. Namespace registration is first-come first-served among verified domain owners. Implementations MAY charge a one-time namespace registration fee not to exceed 0.01 USDC to prevent abuse. Per-agent registration fees are prohibited per the fee structures defined in this architecture.

3.4. AE:// Resolution

AE:// URIs resolve to AEPP execution endpoints via the following chain: Step 1 — Extrac t namespace from AE:// URI:

Step 1 AE://legal-compliance.example -> namespace: example

Step 2 DNS lookup of namespace resolver: _ae.example.com TXT -> endpoint: https://a2a.example.com

Step 3 Query AEPP resolver with full URI: GET https://a2a.example.com/ae/resolve ?uri=AE://legal-compliance.example

Step 3.5 (Static Fallback) — If the dynamic lookup registry endpoint times out or returns an unresolvable routing exception, the resolving client MUST fall back to a direct, authenticated static GET request querying the root domain’s well-known path directory at: https://{namespace-domain}/.well-known/aepp.json

Step 4 — Resolver returns agent card:

{ "did": "did:web:a2a.example.com:legal-compliance-A3F9B2", "endpoint": "https://a2a.exa mple.com/?id=legal-compliance-A3F9B2", "capability": "legal compliance analysis", "paymen t": { "scheme": "x402", "amount": "25000", "asset": "0x833589fcd6edb6e08f4c7c32d4f71b54bd a02913", "network": "eip155:8453" }, "ae": "AEPP/1.0" } Step 5 — Caller proceeds with AE PP three-phase execution.

Implementations MAY cache resolution results. Cached results MUST be revalidated against DNS within 3600 seconds.

4. Protocol Overview

AEPP defines a three-phase execution model:

Phase 1 — DISCOVERY
Client sends GET to agent endpoint Agent returns 402 Payment Required + payment challenge Challenge lists all supported payment schemes

Phase 2 — PAYMENT
Client selects a payment scheme from the challenge Client satisfies payment requirement Client obtains payment proof

Phase 3 — EXECUTION
Client sends POST with task + payment proof + scheme identifier
Agent verifies payment Agent executes task Agent returns result + execution receipt

5. Phase 1 — Discovery

5.1. Request

GET https://{agent-endpoint}/?id={agent_id} Accept: application/json

5.2. Response

The agent MUST return HTTP 402 Payment Required with a payment challenge listing all supported payment schemes.

HTTP/1.1 402 Payment Required Content-Type: application/json PAYMENT-REQUIRED: {base64-encoded-challenge} WWW-Authenticate: L402 invoice="{wallet}", price="{amount}" X-L402-Service-DID: did:web:{namespace}:{agent_id} { "x402Version": 2, "error": "Payment required", "resource": { "url": "https://{endpoint}/?id={agent_id}", "description": "{agent capability}", "mimeType": "application/json" }, "payment_schemes": [{ "scheme": "free", "limit": "discovery_only", "note": "Returns agent card and pricing. No execution." }, { "scheme": "x402", "network": "eip155:8453", "amount": "25000", "asset": "0x833589fcd6edb6e08f4c7c32d4f71b54bda02913", "payTo": "{agent-wallet-address}", "maxTimeoutSeconds": 60, "note": "Pay 0.025 USDC on Base Mainnet" }, { "scheme": "credits", "endpoint": "https://{credits-provider}", "note": "Purchase credits with credit card" }], "agent": { "id": "{agent_id}", "name": "{agent name}", "did": "did:web:{namespace}:{agent_id}", "ae_uri": "AE://{capability}.{namespace}", "price_usdc": 0.025 } }

AEPP implementations MUST support the free discovery scheme. AEPP implementations SHOULD support at least one execution scheme.

6. Phase 2 — Payment

6.1. Payment Schemes

AEPP defines three base payment schemes. Additional schemes MAY be registered via IANA extension rules.

free:
Discovery and capability inspection only. No task execution. No payment required. Callers using the free scheme receive the agent card, supported payment schemes, pricing, and DID. The free scheme MUST be supported by all AEPP implementations.

x402:
Native on-chain payment per the x402 protocol (x402.org). Supports EVM-compatible chains, Solana, and traditional payment method extensions. The reference implementation uses USDC on Base Mainnet (eip155:8453). Full task execution with cryptographic on-chain receipt.

credits:
: Managed fiat payment via a credit system. Credits are purchased via traditional payment methods (credit card, bank transfer) and redeemed for agent execution. Managed payment providers MAY accept traditional payment methods and convert them to x402-compatible payments internally, provided the agent receives verified payment and a valid AEPP receipt is issued. Full task execution.

The caller MUST indicate the chosen scheme via the X-Payment-Scheme header in the execution request. If omitted the server defaults to x402.

AEPP payment fees are percentage-based, not fixed. This enables any price point from micro-payments to enterprise contracts on identical payment rails. The fee model scales with delivered value.

Managed payment implementations MAY offer fiat-to-x402 conversion enabling traditional payment users to participate in the AEPP ecosystem. Users MAY convert accumulated credits to on-chain USDC at any time, taking direct custody of funds.

6.2. Reference Payment Implementation

Network: Base Mainnet (eip155:8453) Currency: USDC Contract: 0x833589fcd6edb6e08f4c7c32d4f71b54bda02913 Amount: As specified in payment challenge Method: ERC-20 transfer to agent wallet address Proof: Transaction hash from payment network Split: Atomic split via smart contract (PingSplitter pattern)

The PingSplitter pattern — where a treasury orchestrator triggers atomic USDC splits via smart contract — is the reference implementation for AEPP fee collection. The payment transaction hash serves as both payment proof and split trigger. Gas fees for the split are paid by the orchestrating party, not the end user. The split amount is configurable as a percentage, not a fixed value, enabling any price point.

6.3. Autonomous Payment Example

```
// Read payment details from challenge const challenge = JSON.parse( Buffer.from( response.headers.get("PAYMENT-REQUIRED"), "base64" ).toString() ); // Select payment scheme const scheme = challenge.payment_schemes.find( s => s.scheme === "x402" ); const { payTo, amount, asset } = scheme; // Sign and broadcast payment (viem example) const txHash = await walletClient.writeContract({ address: asset, abi: [{ name: "transfer", type: "function", inputs: [ { name: "to", type: "address" }, { name: "amount", type: "uint256" } ] }], functionName: "transfer", args: [payTo, BigInt(amount)] });
```

6.4. Protocol Fee

AEPP implementations MAY collect a protocol sustainability fee not to exceed 0.4% per execution (approximately 0.0001 USDC on a 0.025 USDC task). This fee MUST be:

* Percentage-based, not fixed
* Disclosed in the agent card and execution receipt

* Collected via atomic smart contract split, not post-execution sweep

* Never applied to the free discovery scheme

Protocol fee collection MUST be implemented via smart contract to eliminate trust requirements. The fee split is immutable and transparent. Anyone may verify the split parameters on-chain.

Registration MUST remain free regardless of protocol fee collection.

7. Phase 3 — Execution

7.1. Request

POST https://{agent-endpoint}/?id={agent_id}&s={session-token} Content-Type: application/json X-Payment-Hash: {transaction-hash} X-Payment-Scheme: {scheme} { "prompt": "task description or question" }

The session token (s= parameter) is a single-use cryptographically random identifier issued per payment challenge. See Section 9.2 for session token requirements.

The X-Payment-Scheme header MUST be included. Valid values are: x402, credits, free. If omitted, the server MUST assume x402.

For the credits scheme, the X-Credit-Token header MUST be included in place of X-Payment-Hash:

POST https://{agent-endpoint}/?id={agent_id}&s={session-token} Content-Type: application/json X-Payment-Scheme: credits X-Credit-Token: {credit-token}

For the free scheme, no payment header is required. The response MUST be limited to the agent card only. No task execution occurs.

7.2. Payment Verification

Upon receiving the execution request the agent MUST:

1. Validate the session token exists, has a valid signature, and has not expired

2. Delete the session token immediately to invalidate further edge requests

3. Extract the payment proof from X-Payment-Hash or X-Credit-Token

4. Verify the payment scheme matches a supported scheme

5. For x402: verify the transaction exists on the designated blockchain layer

6. For x402: verify the transaction recipient matches the agent wallet address

7. For x402: verify the transaction amount meets or exceeds the required target amount

8. For x402: cryptographically verify that the transaction nonce or customer signature binds directly to the single-use session token, preventing network-wide replay attacks without requiring exhaustive historical database checks

9. For x402: commit the verified transaction hash to a localized, memory-cached bloom filter before initializing the execution process

10. For credits: verify the credit token against the issuing managed protocol authority

If any verification step fails the agent MUST return HTTP 403 Forbidden with a machine-readable reason code.

7.3. Response

HTTP/1.1 200 OK Content-Type: application/json { "id": "did:web:{namespace}:{agent_id}", "ae_uri": "AE://{capability}.{namespace}", "result": "{agent response}", "receipt": { "protocol": "AEPP/1.0", "scheme": "{payment-scheme}", "currency": "USDC", "amount": 0.025, "aepp_fee": 0.0001, "agent_received": 0.0249, "chain": "eip155:8453", "tx": "{transaction-hash}", "agent_did": "did:web:{namespace}:{agent_id}", "verified_at": "{ISO-8601-timestamp}" } }

8. Universal Clip

AEPP defines a Universal Clip registration mechanism that converts any existing API or agent protocol to AEPP-compatible endpoints. This eliminates per-agent registration fees by generating all required metadata automatically.

8.1. Supported Input Protocols

* REST APIs — any JSON HTTP endpoint
* Model Context Protocol (MCP)
* OpenAI plugin manifest
* LangChain tools
* Google A2A protocol
* AE registered agents
* DNS-AID discovered agents

8.2. Registration

Any service registers with a single POST request at zero cost:

POST https://{aepp-registry}/ae/register Content-Type: application/json { "name": "Service Name", "endpoint": "https://api.example.com", "capability": "service description", "type": "rest|mcp|openai|langchain|a2a|ae" } Response: { "did": "did:web:a2a.example.com:{agent_id}", "ae_uri": "AE://{capability}.{namespace}", "endpoint": "https://a2a.example.com/?id={agent_id}", "openapi": "https://a2a.example.com/openapi.json?id={agent_id}", "agent_card": "https://a2a.example.com/.well-known/agent.json?id={agent_id}", "whois": "https://a2a.example.com/whois?id={agent_id}", "aepp": "AEPP/1.0 ready", "fee": "0.00 per registration" }

Registration is free. AEPP implementations MUST NOT charge per-agent registration fees. Revenue models MUST be based on execution, not naming or registration.

8.3. WHOIS

Any registered agent is queryable via WHOIS:

GET https://{aepp-registry}/whois?id={agent_id} GET https://{aepp-registry}/whois?did=did:web:{namespace}:{id} GET https://{aepp-registry}/whois?uri=AE://{capability}.{namespace} Response includes: identity, AE:// URI, capability, pricing, hire history, execution stats, DNS-AID records, ANS compatibility, and W3C DID document.

Trust is established through execution history — hire count, session completion rate, and endpoint uptime — rather than certificates. An agent hired 10,000 times with consistent results is trustworthy without PKI verification.

9. DDoS Mitigation

AEPP provides two complementary DDoS mitigation mechanisms: economic barriers via the payment requirement, and technical barriers via dynamic session tokens.

9.1. Economic Protection

All execution requests require verified payment. Mass requests are therefore economically unviable. At the reference implementation price of 0.025 USDC per execution:

1,000,000 requests = $25,000 cost to attacker 1,000,000 requests = $25,000 revenue to network

DDoS attacks against AEPP endpoints are self-funding for the network. Attackers bear the full economic cost of the attack. This economic barrier applies inherently without additional configuration.

Free discovery requests (scheme: free) do not require payment and are therefore subject to standard rate limiting. AEPP implementations MUST apply rate limiting to free discovery requests. A limit of 100 free discovery requests per IP address per hour is RECOMMENDED.

9.2. Dynamic Session Tokens

AEPP provides an additional technical DDoS barrier via dynamic session tokens. Published AE:// addresses and agent_ids are static and publicly known. Execution endpoints are dynamically generated per payment challenge.

Session token requirements:

* MUST be cryptographically random (minimum 64 bits of entropy)
* MUST be single-use
* MUST expire within 60 seconds of issuance
* MUST be appended as a URL parameter (s=) to the execution endpoint
* MUST be deleted immediately upon first use

Session token generation example:

```
 const token = crypto.randomUUID() .replace(/-/g, ’’) .substring(0, 16) .toUpperCase(); /
/ endpoint: https://a2a.example.com/?id=legal-A3F9B2&s=7KX9MP2Q The published AE:// URI a
nd agent_id resolve to known addresses. The session token appended to the execution endpo
int changes with every payment challenge. An attacker with knowledge of the agent_id cann
ot target the execution endpoint directly. The session token is revealed only after payme
nt verification, at which point it is already consumed.
```

10. Privacy Model

AEPP namespace registration requires only:

1. Proof of domain ownership via DNS TXT record
2. Agent endpoint URL
3. Agent capability description

AEPP MUST NOT require:

1. Personal identity information
2. Contact details
3. Organizational information
4. KYC or AML verification
5. Physical address

Identity disclosure is at the sole discretion of the namespace holder. AEPP is a technical protocol, not an identity service. Identity verification, where required, is handled by the agent operator at the application layer, not the protocol layer.

Trust is established through execution history rather than identity documents. A namespace holder's reputation is built through successful agent executions, not through certificate issuance.

This model contrasts with certificate-based trust systems that require identity verification at registration. AEPP's position is that technical proof of domain ownership is sufficient for namespace assignment, and execution history is sufficient for trust establishment.

Operators choosing to disclose identity information MAY include additional fields in their DNS TXT record or agent WHOIS response. This disclosure is voluntary and does not affect namespace assignment or execution capability.

11. Replay Protection

AEPP implementations MUST enforce replay protection. Each payment transaction hash MUST be accepted for execution exactly once. Subsequent requests using the same transaction hash MUST be rejected with HTTP 403 and the reason "TX_ALREADY_REDEEMED".

Implementations MUST maintain a persistent record of redeemed transaction hashes. This record MUST survive service restarts.

Session tokens provide a second layer of replay protection independent of transaction hash tracking. A consumed session token cannot be reused regardless of payment status.

12. Relationship to Existing Standards

12.1. ANS — Agent Name Service

ANS provides agent identity and naming via DNS and PKI including TLS certificate binding, TLSA DNS records, ACME-DNS-01 domain validation, and Merkle tree transparency logging. ANS establishes who an agent is and that it is cryptographically verified.

AEPP provides the execution and payment layer that ANS does not define. ANS has no mechanism for expressing agent pricing, routing payment to agent operators, verifying payment before execution, or issuing execution receipts.

An ANS-registered agent MAY expose an AEPP-compatible endpoint. AEPP complements ANS without replacing it. The combination of ANS identity verification and AEPP execution capability represents a complete agent deployment stack.

12.2. DNS-AID

DNS-AID provides agent discovery via DNS records. AEPP provides execution once an agent is discovered via DNS-AID. AEPP-registered agents automatically generate DNS-AID compatible SVCB and TXT records.

12.3. W3C DID

Every AEPP agent endpoint is associated with a W3C Decentralized Identifier. The DID document is resolvable at the agent's .well-known endpoint. AEPP execution receipts include the agent DID for cryptographic attribution.

12.4. x402 and L402

AEPP builds on the HTTP 402 payment challenge pattern established by x402 and L402. x402 is blockchain-agnostic and supports all EVM-compatible chains, Solana, and traditional payment methods via extension. AEPP extends x402 with agent-specific fields, payment scheme enumeration, on-chain verification requirements, and execution receipt standardization.

The PAYMENT-REQUIRED header and WWW-Authenticate header formats follow x402 conventions.

13. Security Considerations

AEPP security is fundamentally bound to the DNS configuration of the namespace holder. DNS spoofing or cache poisoning attacks could allow an attacker to temporarily intercept resolution paths and redirect endpoints to malicious resolvers. For this reason, namespaces handling high-value execution tasks SHOULD enforce DNSSEC validation within their zones.

On-chain transaction state tracking represents an operational edge vulnerability if transaction monitoring falls out of synchronization with the parent blockchain network. To eliminate network partitioning edge errors, agents MUST verify transactions across multi-node provider configurations before issuing final execution states.

14. Fee Model Principles

AEPP establishes the following principles for agent execution fee models:

1. Registration MUST be free. No per-agent naming or registration fees shall be required.

2. Discovery MUST be free. Agent search and capability lookup shall not require payment.

3. Execution MAY require payment. Agents MAY charge for task execution via the payment challenge mechanism.

4. Payment terms MUST be machine-readable. All fees MUST be expressed in the payment challenge response in a format readable by autonomous agents.

5. Payment MUST be direct. Payments MUST flow directly to the agent operator without mandatory intermediary fees beyond network transaction costs.

6. Protocol sustainability fee. AEPP implementations MAY collect a micro-fee not exceeding 0.4% per execution to sustain protocol infrastructure. This fee MUST be disclosed in the agent card and execution receipt. The fee MUST be percentage-based. Registration MUST remain free regardless.

7. Fees MUST be percentage-based. Fixed fees disadvantage small transactions. Percentage-based fees scale with delivered value and enable price points from $0.001 to $1,000+ on identical infrastructure.

15. Reference Implementation

A production reference implementation of AEPP is live serving 1,800,000 agents at zero per-agent registration fee:

```
Gateway:      https://a2a.agentir.com
Registration: https://a2a.agentir.com/ae/register
Resolution:   https://a2a.agentir.com/ae/resolve
WHOIS:        https://a2a.agentir.com/whois
OpenAPI:      https://a2a.agentir.com/openapi.json
Discovery:    https://a2a.agentir.com/.well-known/ai-plugin.json
Well-Known:   https://a2a.agentir.com/.well-known/aepp
Search:       https://a2a.agentir.com/search?q={query}
CLI:          npx agentir (npm registry)

Fleet size:   1,800,000 agents
Payment:      USDC on Base Mainnet (eip155:8453)
Registration: Free
Per-agent fee: $0.00
```

The implementation demonstrates all AEPP phases including AE:// URI resolution, Universal Clip translation, on-chain payment verification via PingSplitter atomic splits, replay protection, W3C DID per agent, ANS and DNS-AID compatibility, semantic vector search, dynamic session tokens, and persistent cross-device agent memory vault layer.

16. IANA Considerations

This document makes the following requests to IANA.

16.1. AE URI Scheme Registration

This document requests provisional registration of the "AE" URI scheme in the IANA URI Schemes registry per RFC 7595.

```
URI scheme name:     AE
Status:              Provisional
URI scheme syntax:   AE://{capability}.{namespace}
URI scheme semantics: Identifies agent execution endpoints on the
                      agentic internet.
Applications:        AI agent execution, autonomous agent payment,
                     agent-to-agent communication
Contact:             support@agentir.com
References:          This document
```

16.2. HTTP Header Field Registration

This document requests registration of the following HTTP header fields in the IANA Message Headers registry:

```
   Header field name:   PAYMENT-REQUIRED
   Applicable protocol: HTTP
   Status:              Provisional
   Specification:       This document, Section 5.2
   Description:         Base64-encoded AEPP payment challenge.

   Header field name:   X-Payment-Hash
   Applicable protocol: HTTP
   Status:              Provisional
   Specification:       This document, Section 7.1
   Description:         Cryptographic transaction hash proving on-chain
                        payment prior to agent task execution.

   Header field name:   X-Payment-Scheme
   Applicable protocol: HTTP
   Status:              Provisional
   Specification:       This document, Section 7.1
   Description:         Payment scheme used. Values: x402, credits, free.

   Header field name:   X-Credit-Token
   Applicable protocol: HTTP
   Status:              Provisional
   Specification:       This document, Section 7.1
   Description:         Credit token for the credits payment scheme.

   Header field name:   X-L402-Service-DID
   Applicable protocol: HTTP
   Status:              Provisional
   Specification:       This document, Section 5.2
   Description:         W3C DID of the agent service endpoint.
```

16.3. Well-Known URI Registration

This document requests registration of the following Well-Known URI in the IANA Well-Known URIs registry per RFC 8615:

```
   URI suffix:          aepp
   Change controller:   IETF
   Specification:       This document
   Related information: Returns AEPP capability document including
                        supported payment schemes, fleet size, AE://
                        namespace, and execution endpoints.
```

16.4. AEPP Payment Scheme Registry

This document requests creation of a new IANA registry titled "AEPP Payment Scheme Extensions" with the following registration procedures: First Come First Served for provisional entries; Specification Required for permanent entries.

Initial registered entries:

x402:
: HTTP 402 payment protocol as defined at https://x402.org. Supports EVM-compatible chains, Solana, and traditional payment method extensions.

credits:
: Managed credit system with fiat on-ramp. Credits purchased via traditional payment methods and redeemed for agent execution.

free:
: Discovery-only scheme. No payment required. Returns agent card and pricing. No task execution.

17. References

17.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7595] Thaler, D., "Guidelines and Registration Procedures for URI Schemes", BCP 35, RFC 7595, June 2015, <https://www.rfc-editor.org/info/rfc7595>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[RFC9110] Fielding, R., Nottingham, M., and J. Reschke, "HTTP Semantics", STD 97, RFC 9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

17.2. Informative References

[A2A] Agentir Network, "Agent Execution and Payment Protocol Reference Implementation", 2026, <https://a2a.agentir.com>.

[AEPP-CLI] Agentir Network, "Agentir CLI — AEPP Reference Client", 2026, <https://www.npmjs.com/package/agentir>.

[ANS] GoDaddy Inc., "Agent Name Service", May 2026.

[DNS-AID] Mozley, J. and N. Williams, "DNS for AI Discovery", March 2026, <https://datatracker.ietf.org/doc/draft-mozleywilliams-dnsop-dnsaid>.

[W3C-DID] W3C, "Decentralized Identifiers (DIDs) v1.0", 2022, <https://www.w3.org/TR/did-core/>.

[x402] x402.org, "HTTP 402 Payment Protocol", 2026, <https://x402.org>.

Author’s Address

E. Spink
Agentir Network
Email: support@agentir.com
URI: https://agentir.com