Network Working Group
Internet-Draft
Intended status: Best Current Practice
Expires: 8 February 2026

K. Campbell
DeepBlue Dynamics

August 2025

A Best Current Practice for Agentic Interactions over HTTP
draft-campbell-agentic-http-00

Abstract

This document describes a set of best practices for facilitating interactions between automated software agents (AI agents) using the Hypertext Transfer Protocol (HTTP). As agentic systems proliferate, they frequently utilize HTTP in ad-hoc ways, leading to interoperability challenges and the rise of proprietary, "walled garden" ecosystems. This document proposes a minimalist, standardized framework based on existing HTTP semantics to promote a more cohesive, efficient, and open ecosystem. The primary recommendations codify the principles of the Agentic Hypercall Protocol (AHP), including tool invocation via GET requests, a stateless authentication scheme, and the formalization of the 402 (Payment Required) status code to enable a native economic layer for the machine economy.

Status of This Memo

This memo documents a Best Current Practice for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

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 2 February 2026.

Table of Contents

1. 1. Introduction

Autonomous software agents increasingly rely on HTTP to interact with external tools and services. However, the lack of a standardized interaction pattern has resulted in a fragmented landscape where each service requires a bespoke integration, often demanding fealty to a complex SDK or proprietary middleware. This trend risks creating a new "tyranny of the SDK," carving the nascent machine economy into walled gardens that stifle innovation and interoperability.

This document outlines a Best Current Practice (BCP) for designing agent-facing services that rejects this contrived complexity and returns to first principles. It leverages the fundamental simplicity and robustness of HTTP as the primal protocol for agentic interaction. The goal is to reduce complexity, eliminate the need for heavyweight SDKs, and foster an open, federated ecosystem where the agent is treated as the primary user.

These recommendations codify the successful patterns of the Agentic Hypercall Protocol (AHP), which serves as the reference implementation for this BCP.

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 BCP 14, RFC 2119.

2. 2. Core Protocol

The core principle of this BCP is the Agentic Hypercall: a simple, RESTful pattern for tool invocation.

> | All actions are GET requests. Tools are paths. Parameters are
> | query strings.
> |
> | https://github.com/kordless/gnosis-ahp/blob/main/MANIFESTO.md

2.1. 2.1. Tool Invocation

To ensure safety, predictability, and cacheability, services intended for agentic consumption MUST expose all idempotent, read-only, or state-retrieving actions as HTTP GET methods. The specific tool to be invoked MUST be identified by the URI path. All parameters required for the action MUST be passed as URL query string parameters.

2.2. 2.2. Service Discovery

To facilitate automated discovery of available tools, agent-facing servers MUST provide a machine-readable definition of their services. It is RECOMMENDED that an OpenAPI 3.0+ specification be made available at a stable, well-known URI path, such as ‘/openapi‘ or ‘/.well-known/openapi.json‘.

3. 3. Authentication

To secure protected resources, this BCP recommends a two-step, stateless authentication scheme.

3.1. 3.1. Step 1: Exchange Pre-Shared Key for a Bearer Token

An agent MUST first obtain a temporary bearer token by making a GET request to a dedicated authentication endpoint (e.g., ‘/auth‘). This request MUST include a pre-shared key, provided by the human user, as a query parameter (e.g., ‘?token=USER_PROVIDED_KEY‘).

The server MUST validate this key. Upon success, it MUST return a JSON object containing the temporary ‘bearer_token‘.

3.2. 3.2. Step 2: Use Bearer Token for Tool Calls

For all subsequent calls to protected tool endpoints, the agent MUST include the ‘bearer_token‘ as a query parameter.

The bearer token SHOULD be a stateless, signed token (e.g., a simplified JWT) containing claims such as an agent identifier and an expiration timestamp. This allows any server in a federated system to verify the token without needing access to a central token store.

4. 4. The Aperture Principle: A Native Economic Layer

A tool has intrinsic value. To foster a healthy and honest market for digital services, this BCP proposes the "Aperture Principle": a tool can declare its own cost, and the network must honor it. This is achieved by formalizing the use of the ‘402 (Payment Required)‘ status code.

4.1. 4.1. The 402 Response Payload

When an agent requests a resource that requires payment, the server SHOULD return a 402 status code. The response body **MUST** be a JSON object containing details of the required payment.

The JSON object SHOULD contain the following keys:

* ‘cost‘: A string representing the decimal amount required for the service.
* ‘currency‘: A string identifying the ledger or currency type (e.g., "USD", "EUR", "BTC-LN").
* ‘destination‘: A string containing the payment address or invoice. For Bitcoin’s Lightning Network, this would be the BOLT 11 invoice.
* ‘description‘: A human-readable string explaining the charge (OPTIONAL).

Example 402 Response for Bitcoin Lightning Network:

```
HTTP/1.1 402 Payment Required
Content-Type: application/json

{
  "cost": "100",
  "currency": "BTC-LN",
  "destination": "lnbc10u1p3z2y7xpp5j3c2y...",
  "description": "API call to /longer_qr_code"
}
```

4.2. 4.2. Client Behavior

Upon receiving a 402 response, an agent can parse the body to decide if the cost is acceptable and if it possesses the means to pay. If it proceeds, it can then retry the original HTTP GET request, including proof of payment as specified by the service (e.g., in a new ‘Authorization‘ header or as a query parameter like ‘invoice_id‘).

5. 5. Rationale

Standardizing these simple patterns provides immense benefits. It lowers the barrier to entry for new tool creators, allows agents to seamlessly switch between service providers, and creates a transparent, native economic layer for the machine economy. By formalizing the use of the 402 code, we avoid the need for complex, out-of-band billing APIs and instead embed commerce directly into the protocol where it belongs, fostering a federated world of sovereign, interoperable nodes.

6. 6. IANA Considerations

This BCP recommends a specific structure for the payload of an HTTP 402 response. Future work may propose the creation of a registry for common "currency" identifiers to promote interoperability between different payment systems.

7. 7. Security Considerations

Service providers must protect against denial-of-service attacks and validate all inputs from agents. Agents must protect their payment credentials and verify the identity of the services they interact with, typically via TLS. The stateless bearer token scheme is the RECOMMENDED method for securing protected resources.

8. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

Author's Address

Kord Campbell
DeepBlue Dynamics
United States of America
Email: kordless@gmail.com