Workgroup:	Network Working Group		
Internet-Draft:	draft-payment-discovery-00		
Published:	3 March 2026		
Intended Status:	Informational		
Expires:	4 September 2026		
Authors:	J. Moxey	B. Ryan	T. Meagher
	Tempo Labs	*Tempo Labs*	*Tempo Labs*

Payment Method Discovery Mechanisms for HTTP Payment Authentication

Abstract

This document defines discovery mechanisms for the "Payment" HTTP authentication scheme [I-D.httpauth-payment]. It specifies how clients can discover a server's payment capabilities before initiating requests, including supported payment methods, accepted currencies, and intents.

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 4 September 2026.

Copyright Notice

Table of Contents

1. Introduction

The "Payment" HTTP authentication scheme [I-D.httpauth-payment] enables servers to require payment for resource access. While the 402 response with `WWW-Authenticate: Payment` header provides all information needed to complete a paid exchange, clients may benefit from discovering payment capabilities before making requests.

This specification defines an optional discovery mechanism using a well-known HTTP endpoint that returns structured payment capability information.

Discovery is **OPTIONAL**. Servers **MAY** implement this mechanism to improve client experience. Clients **MUST NOT** require discovery to function; the 402 challenge provides all information needed to complete payment.

2. Requirements Language

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

Currency An identifier for an accepted unit of value, using the same formats defined in the "charge" intent specification's Currency Formats section. This includes ISO 4217 codes (e.g., `"usd"`) and method-defined identifiers (e.g., token contract addresses).

Discovery The process by which a client learns a server's payment capabilities before initiating a request that may require paid access.

Payment Capabilities The set of payment methods, intents, and accepted currencies that a server accepts as payment.

4. Well-Known Endpoint

4.1. Endpoint Location Section

Servers **MAY** expose payment capabilities at the following location:

```
GET /.well-known/payment
```

4.2. Request

The client issues a GET request to `/.well-known/payment`. The request **SHOULD** include an `Accept` header with `application/json`:

```
GET /.well-known/payment HTTP/1.1
Host: api.example.com
Accept: application/json
```

4.3. Response

The server responds with a JSON object describing its payment capabilities. The response **MUST** use `Content-Type: application/json`.

Response Schema:

Field	Type	Required	Description
`version`	integer	**REQUIRED**	Schema version. Currently 1.
`realm`	string	**OPTIONAL**	Default realm for payment challenges.
`description`	string	**OPTIONAL**	Human-readable description of the service.
`methods`	object	**REQUIRED**	Map of supported payment methods.

Table 1

Method Object Schema:

Each key in `methods` is a registered payment method identifier. The value is an object with:

Field	Type	Required	Description
`intents`	array	**REQUIRED**	Supported intent types.
`currencies`	array	**REQUIRED**	Accepted currency identifiers, using the formats defined by the payment method specification (e.g., ISO 4217 codes, token addresses).

Table 2

Example Response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: max-age=300

{
  "version": 1,
  "realm": "api.example.com",
  "description": "AI inference API",
  "methods": {
    "tempo": {
      "intents": ["charge", "session"],
      "currencies": ["0x20c0000000000000000000000000000000000000"]
    },
    "lightning": {
      "intents": ["charge"],
      "currencies": ["sat"]
    }
  }
}
```

4.4. Caching

Servers **SHOULD** include `Cache-Control` headers with short durations to allow clients to detect capability changes. A maximum age of 5 minutes is **RECOMMENDED**:

```
Cache-Control: max-age=300
```

Longer durations (e.g., `max-age=3600`) **MAY** be used for capabilities that change infrequently. Clients **SHOULD** respect cache headers and refetch when capabilities may have changed (e.g., after receiving an unexpected 402 challenge for a method not in the cached discovery response).

4.5. Version Handling

Clients **MUST** check the `version` field before processing the response. If the `version` value is higher than the version the client supports, the client **SHOULD** treat the response as unsupported and fall back to the 402 challenge flow. Clients **MUST NOT** assume forward compatibility with unknown schema versions.

4.6. Error Handling

If the server does not support discovery, it **SHOULD** return 404 Not Found. Clients **MUST NOT** treat a 404 response as an error; it simply indicates discovery is unavailable.

5. Security Considerations

5.1. Discovery Spoofing

Discovery information is advisory and not cryptographically authenticated. Clients **MUST NOT** rely on discovery for security decisions. The actual payment challenge in the 402 response is authoritative.

5.2. Well-Known Endpoint Security

The well-known endpoint **MUST** be served over HTTPS. Clients **MUST NOT** accept discovery information over unencrypted HTTP.

5.3. Information Disclosure

Discovery endpoints reveal payment capabilities to unauthenticated clients. Servers should consider whether this information disclosure is acceptable.

5.4. Cross-Origin Requests

Browser-based clients (e.g., wallets, payment agents) may need to access the discovery endpoint cross-origin. Servers that intend to support browser-based clients **SHOULD** include appropriate CORS headers (e.g., `Access-Control-Allow-Origin`) on responses to `/.well-known/payment`. This aligns with the cross-origin considerations in Section 11.11 of [I-D.httpauth-payment].

6. IANA Considerations

6.1. Well-Known URI Registration

This document registers the following well-known URI in the "Well-Known URIs" registry established by [RFC8615]:

- **URI Suffix**: payment
- **Change Controller**: IETF
- **Reference**: This document, Section 4
- **Status**: permanent
- **Related Information**: None

7. Normative 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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

[I-D.httpauth-payment] Moxey, J., "The 'Payment' HTTP Authentication Scheme", January 2026, <https://datatracker.ietf.org/doc/draft-httpauth-payment/>.

Authors' Addresses

Jake Moxey
Tempo Labs
Email: jake@tempo.xyz

Brendan Ryan
Tempo Labs
Email: brendan@tempo.xyz

Tom Meagher
Tempo Labs
Email: thomas@tempo.xyz