Web Bot Auth
Internet-Draft
Intended status: Informational
Expires: 27 November 2026

M. Guerreiro
Cloudflare
U. Kirazci
Amazon
T. Meunier
Cloudflare

26 May 2026

Registry and Signature Agent card for Web bot auth
draft-meunier-webbotauth-registry-02

Abstract

This document describes a JSON based format for clients using [DIRECTORY] to advertise information about themselves.

This document describes a JSON-based "Signature Agent Card" format for signature agent using [DIRECTORY] to advertise metadata about themselve. This includes identity, purpose, rate expectations, and cryptographic keys. It also establishes an IANA registry for Signature Agent Card parameters, enabling extensible and interoperable discovery of agent information.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://thibmeu.github.io/http-message-signatures-directory/draft-meunier-webbotauth-registry.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-meunier-webbotauth-registry/.

Discussion of this document takes place on the Web Bot Auth Working Group mailing list (mailto:web-bot-auth@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/web-bot-auth/. Subscribe at https://www.ietf.org/mailman/listinfo/web-bot-auth/.

Source for this draft and an issue tracker can be found at https://github.com/thibmeu/http-message-signatures-directory.

Status of This Memo

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

This Internet-Draft will expire on 27 November 2026.

Copyright Notice

Table of Contents

1. Introduction

Signature Agents are entities that originate or forward signed HTTP requests on behalf of users or services. They include bots developers, platforms providers, and other intermediaries using [DIRECTORY]. These agents often need to identify themselves, and establish trust with origin servers.

Today, the mechanisms for doing so are inconsistent: some rely on User-Agent strings (e.g. MyCompanyBot/1.0), others on IP address lists hosted on file servers (e.g. https://curated-bots.com/ips.json), and still others on out-of-band definitions (e.g. documentation on docs.example.com/mybot). This diversity makes it difficult for operators and origin servers to reliably discover and share a Signature Agent' s purpose, contact information, or rate expectations. Existing discovery mechanisms, such as [OPENID-CONNECT-DISCOVERY], do not have the necessary granularity, and pursue different goals.

This document introduces a JSON-based "Signature Agent Card" format for Signature Agents, to be published in registries and discovered by servers. It also creates a new IANA registry of "Signature Agent Card Parameters" to ensure extensibility and consistency of future attributes.

2. Conventions and Definitions

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. Signature Agent Card

Signature-Agent header is defined in Section 4.1 of [DIRECTORY]. This section describes Signature Agent Card, a JSON object containing parameters describing the Signature Agent.


```
{
  "client_name": "Example Bot",
  "client_uri": "https://example.com/bot/about.html",
  "logo_uri": "https://example.com/",
  "contacts": ["mailto:bot-support@example.com"],
  "expected-user-agent": "Mozilla/5.0 ExampleBot",
  "rfc9309-product-token": "ExampleBot",
  "rfc9309-compliance": ["User-Agent", "Allow", "Disallow", "Content-Usage"],
  "trigger": "fetcher",
  "purpose": "tdm",
  "targeted-content": "Cat pictures",
  "rate-control": "429",
  "rate-expectation": "avg=10rps;max=100rps",
  "known-urls": ["/", "/robots.txt", "*.png"],
  "jwks_uri": "https://example.com/.well-known/http-message-signatures-directory",
  "ips_uri": "https://example.com/ips.json",
  "keys": [{
    "kty": "OKP",
    "crv": "Ed25519",
    "kid": "NFcWBst6DXG-N35nHdzMrioWntdzNZghQSkjHNMMSjw",
    "x": "JrQLj5P_89iXES9-vFgrIy29clF9CC_oPPsw3c5D0bs",
    "use": "sig",
    "nbf": 1712793600,
    "exp": 1715385600
  }]
}
```

Unless otherwise specified, all parameters in this document are OPTIONAL. There MUST be at least one parameter set.

Parameters for which the value is unknown MUST be ignored. All string values are UTF-8.

3.1. Client Name

The client_name parameter provides a friendly identifier for the Signature Agent.

Example

* ExampleBot

* My remote browser company

3.2. Client URI

The client_uri parameter provides inline content or a web page describing the bot: e.g. what does it do, how it handles data it fetches.

Only http, https or data:text/plain are allowed.

Example

* https://example.com/bot/about.html

* data:text/plain,The Example bot is about providing an example.

3.3. Contacts

The contacts parameter provides reliable communication channels in URI forms. Typically, this is an email address.

Example

* ["mailto:bot-support@example.com"]

* ["https://example.com/contact"]

3.4. Logo URI

The logo_uri parameter provides an image reference for visual identification.

TODO: Recommendation for size and format, if there is a clear consensus or reference we can point to.

Example

*

* https://example.com/logo.png

3.5. Expected user agent

The expected-user-agent parameter specifies one or more User-Agent strings as defined in Section 10.1.5 of [HTTP] or prefix matches. Prefixes MAY use * as a wildcard.

Example

* Mozilla/5.0 ExampleBot

3.6. robots.txt product token

The rfc9309-product-token parameter specifies the product token used for robots.txt directives per Section 2.2.1 of [ROBOTSTXT].

Example

* ExampleBot

3.7. robots.txt compliance

The rfc9309-compliance parameter lists directives from robots.txt that the agent implements.

Example

* ["User-Agent", "Disallow"]

* ["User-Agent", "Disallow", "CrawlDelay"]

3.8. Trigger

The trigger parameter indicates the operational mode of the agent.

Valid values:

1. fetcher - request initiated by the user

2. crawler - autonomous scanning

3.9. Purpose

The purpose parameter describes the intended use of collected data. Values SHOULD be drawn from a controlled vocabulary, such as [AIPREF-VOCAB].

Example

* search
* tdm

3.10. Targeted content

The targeted-content parameter specifies the type of data the agent seeks. Its format is arbitrary UTF-8 encoded string.

Example

* SEO analysis
* Vulnerability scanning
* Ads verification

3.11. Rate control

The rate-control parameter indicates how origins can influence the agent's request rate.

TODO: specify a format

Example

* CrawlDelay in robots.txt (non-standard)
* Custom tool
* 429 + [RATELIMIT-HEADER]

3.12. Rate expectation

The rate-expectation parameter specifies anticipated request volume or burstiness.

TODO: consider a format such as avg=10rps;max=100rps

Example

* 500 rps

* Spikes during reindexing

3.13. Known URLs

The known-urls parameter lists predictable endpoints accessed by the agent.

These URLs may be absolute URLs like https://example.com/index.html. They could be relative path like /ads.txt. Or they can use * as wildcard such as *.png.

Example

* ["/"]

* ["/ads.txt"]

* ["/favicon.ico"]

* ["/index.html"]

3.14. JWKS URI

The jwks_uri parameter provides the URL of the signature agent's JWKS keys as defined in Section 5 of [JWK]. This covers HTTP Message Signatures Directory as defined in [DIRECTORY].

When present, this parameter separates key material discovery from metadata discovery. Clients that need key material SHOULD fetch the directory at the given URL rather than relying on the keys parameter. This separation allows registry operators to host metadata and key material on different endpoints, supporting deployment scenarios where the registry endpoint itself contains signature agent card metadata but the key directory is hosted elsewhere.

If both jwks_uri and keys are present, the jwks_uri takes precedence for key discovery.

The URI scheme MUST be https.

Example

* https://example.com/.well-known/http-message-signatures-directory

3.15. IP Address List URI

The ips_uri parameter provides the URL of the signature agent’s IP address list as defined in [JAFAR].

The URI scheme MUST be https.

Example

* https://example.com/ips.json

3.16. Keys

The keys parameter contains a JWKS as defined in Section 5 of [JWK].

If keys is present, it is RECOMMENDED that the card is signed using [HTTP-MESSAGE-SIGNATURES]. Content-Digest header MUST be included in the covered components.

TODO: describe signature, CWS keys.

Example

* https://example.com/.well-known/http-message-signatures-directory

* JWKS-directory

4. Discovery

A registry is a list of URLs, each refering to a signature agent card.

The URI scheme MUST be one of:

* https (RECOMMENDED): Points to an HTTPS resource serving a signature agent card

* http: Points to an HTTP resource serving a signature agent card

* data: Contains an inline signature agent card

Example

```
   # An example list of bots
   https://bot1.example.com/.well-known/http-message-signatures-directory
   https://crawler2.example.com/.well-known/http-message-signatures-directory

   # Now the list of platforms
   https://zerotrust-gateway.example.com/v1/signature-agent-card

   # Below is an inlined card with the data URL scheme
   data:application/json,{"client_name":"Inline Bot","jwks_uri":"https://inline.example.c
om/.well-known/http-message-signatures-directory"}
```

4.1. Formal Syntax

Below is an Augmented Backus-Naur Form (ABNF) description, as described in [ABNF].

The below definition imports http-URI and https-URI from [HTTP], and dataurl from [DATAURL].

```
   registry = *(cardendpointline / emptyline)
   cardendpointline = (
       http-URI /       ; As defined in Section 4.2.1 of RFC 9110
       https-URI /      ; As defined in Section 4.2.2 of RFC 9110
       dataurl          ; As defined in Section 3 of RFC 2397
   ) EOL


   comment = "#" *(UTF8-char-noctl / WS / "#")
   emptyline = EOL
   EOL = *WS [comment] NL ; end-of-line may have
                          ; optional trailing comment
   NL = %x0D / %x0A / %x0D.0A
   WS = %x20 / %x09

   ; UTF8 derived from RFC 3629, but excluding control characters

   UTF8-char-noctl = UTF8-1-noctl / UTF8-2 / UTF8-3 / UTF8-4
   UTF8-1-noctl = %x21 / %x22 / %x24-7F ; excluding control, space, "#"
   UTF8-2 = %xC2-DF UTF8-tail
   UTF8-3 = %xE0 %xA0-BF UTF8-tail / %xE1-EC 2UTF8-tail /
            %xED %x80-9F UTF8-tail / %xEE-EF 2UTF8-tail
   UTF8-4 = %xF0 %x90-BF 2UTF8-tail / %xF1-F3 3UTF8-tail /
            %xF4 %x80-8F 2UTF8-tail

   UTF8-tail = %x80-BF
```

4.2. Out-of-band communication between client and origin

A signature agent MAY submit their signature agent card to an origin, or the origin MAY manually add them to their local registry.

4.3. Registry Endpoint

A registry MAY be provided via a GitHub repository, a public file server, or a dedicated endpoint.

The registry SHOULD be served over HTTPS.

A client application SHOULD validate the registry format and reject malformed entries.

4.3.1. Authentication

A registry endpoint MAY require authentication to restrict access to authorized clients. This allows a registry operator to expose registries without revealing their contents publicly.

No specific authentication mechanism is mandated. Implementations MAY use pre-shared keys as mentioned in [PSK-TLS], bearer tokens as defined in [OAUTH-BEARER], or HTTP Message Signatures as defined in [HTTP-MESSAGE-SIGNATURES]. HTTP Message Signatures are RECOMMENDED when key rotation without out-of-band coordination is desired.

4.3.2. Efficient Polling with Conditional Requests

Registry servers SHOULD include ETag and Last-Modified response header fields as defined in Section 8.8 of [HTTP].

Clients SHOULD send If-None-Match or If-Modified-Since precondition header fields as defined in Section 13.1 of [HTTP] on subsequent requests. A server SHOULD respond with 304 (Not Modified) when the registry has not changed, avoiding redundant data transfer.

4.3.3. Caching

Registry servers SHOULD include a Cache-Control response header field as defined in [HTTP-CACHE] to communicate the intended freshness lifetime of the registry content. Clients SHOULD respect these cache directives and SHOULD NOT poll more frequently than indicated.

4.4. Signature-Agent header

Signature Agent Card format defined in Section 3 extends the format of Signature-Agent header as defined in Section 4.1 of [DIRECTORY].

When used for HTTP Message Signatures, and hosted on a well-known URL, Signature Agent Card MAY be discovered via a Signature-Agent header.

4.5. Change Notification

Pull-based consumption with conditional requests is sufficient for most deployments. When lower notification latency is required (e.g., to promptly act on entry removal), a registry operator MAY implement a push-based change notification mechanism.

4.5.1. Advertising the Notification Endpoint

A registry operator that supports change notifications SHOULD advertise its notification endpoint in the registry HTTP response using a Link header field as defined in [WEB-LINKING]:

```
Link: <https://registry.example/v1/registry-changes>; rel="registry-changes"
```

The registry-changes link relation identifies an endpoint to which clients may register callbacks out of band.

TODO: Register the registry-changes link relation with IANA, or replace it with an extension relation URI.

4.5.2. Callback Registration

Registration of a client callback URL with the registry operator is performed out of band. No specific registration protocol is defined by this document. An unsubscribe mechanism SHOULD be considered, and MAY also be out of band.

4.5.3. Notification Requests

When an entry is added to or removed from the registry, the registry operator MUST send an HTTP request to each registered callback URL.

The format in [CDDL] is as follows:

```
Action = "put" / "delete"

Notification = {
  action: Action,
  signature-agent: tstr
}
```

TODO: should we use application/json, a new media-type, permit binary encoding? TODO: should signature-agent be an array?

action denotes the operation performed on the registry: 1. put when a new entry has been added, 2. delete when an entry has been removed.

The request MUST use POST. It MUST be signed using [HTTP-MESSAGE-SIGNATURES] with a key present in the registry operator's existing signature agent card. This is the signature agent card associated with the registry operator during out-of-band callback registration.

The Content-Type SHOULD be application/json.

Example notification for an added entry:

```
POST /registry-callback HTTP/1.1
Host: origin.example
Content-Type: application/json
Signature-Input: sig1=("@method" "@authority" "@path" "content-digest"); \
  created=1741046400; keyid="NFcWBst6DXG-N35nHdzMrioWntdzNZghQSkjHNMMSjw"
Signature: sig1=:base64signature:
Content-Digest: sha-256=:base64hash:

{
  "action": "put",
  "signature-agent": "https://abc123.registry.example/.well-known/signature-agent-card"
}
```

4.5.4. Notification Processing

Upon receiving a notification, the client MUST verify the HTTP Message Signature against the registry operator's key, discovered via the registry operator's signature-agent card. Notifications with missing or invalid signatures MUST be rejected.

A notification is advisory only. The registry endpoint remains the authoritative source of truth. After verifying a put notification, clients SHOULD re-fetch the affected signature agent card to obtain current metadata.

For delete notifications, clients SHOULD confirm the removal by re-fetching the full registry. This confirmation does not need to happen synchronously with notification processing.

Registry operators SHOULD retry delivery of failed notifications with exponential backoff. Clients that miss notifications will recover on their next conditional pull from the registry endpoint.

5. Security Considerations

Malicious actors may put properties which are not theirs in the registry. If signatures are present, clients MUST verify them. Clients SHOULD reject cards with invalid signatures.

5.1. Registry Integrity

When a registry is served over HTTPS, TLS provides channel integrity between the server and the client. To additionally bind the registry contents to the registry operator's cryptographic identity, registry servers SHOULD sign their HTTP responses using [HTTP-MESSAGE-SIGNATURES] with a key present in the registry operator's own signature agent card. Clients SHOULD verify such signatures when present.

5.2. Binding Delegated Keys to the Directory Authority

When a signature agent card delegates key discovery using jwks_uri, clients SHOULD validate the referenced directory response to ensure the authenticity and integrity of the delegated key material.

When a directory server provides key material over HTTP or HTTPS, it is RECOMMENDED that it include one HTTP Message Signature per key in the response, with each key used to provide one signature. The signature SHOULD cover @authority with the req flag set, and SHOULD include the created, expires, keyid, and tag signature parameters. The tag value MUST be http-message-signatures-directory.

Clients SHOULD validate these signatures using the keys provided by the directory. Clients SHOULD ignore keys from a directory response that do not have a corresponding valid signature binding the key material to the directory authority.

5.3. Private Registries

Registry endpoints that require authentication as described in Section 4.3.1 limit exposure of registry entries to authorized clients only. Registry operators SHOULD use authenticated endpoints when the enumeration of their registry entries is sensitive.

6. Privacy Considerations

TODO

6.1. Access Patterns

Registry servers SHOULD avoid logging personally identifiable information from client requests. Clients fetching a registry reveal their interest in its entries; registry servers SHOULD treat access logs as sensitive.

7. IANA Considerations

7.1. Registration template

New registrations need to list the following attributes:

Parameter Name: The name requested (e.g. "useragent"). This name is case sensitive. Names may not match other registered names in a case-insensitive manner unless the Designated Experts state that there is a compelling reason to allow an exception
Parameter Description: Brief description of the Header Parameter
Change Controller: For Standards Track RFCs, list the "IESG". For others, give the name of the responsible party. Other details (e.g., postal address, email address, home page URI) may also be included.
Reference: Where this parameter is defined
Notes: Any notes associated with the entry

New entries in this registry are subject to the Specification Required registration policy ([RFC8126], Section 4.6). Designated experts need to ensure that the token type is defined to be used for both token issuance and redemption. Additionally, the experts can reject registrations on the basis that they do not meet the security and privacy requirements defined in TODO.

7.1.1. Initial Registry content

This section registers the Signature Agent Card Parameter names defined in Section 3 in this registry.

7.1.1.1. Client Name Parameter

Parameter Name: client_name

Parameter Description: A friendly name for your signature agent.

Change Controller: IETF

Reference: Section 3.1

Notes: N/A

7.1.1.2. Client URI Parameter

- *Parameter Name:* client_uri
- *Parameter Description:* Describes what the bot does inline with data URI or with an HTTP link to an external resource
- *Change Controller:* IETF
- *Reference:* Section 3.2
- *Notes:* N/A

7.1.1.3. Logo URI Parameter

- *Parameter Name:* logo_uri
- *Parameter Description:* Image for a quick visual identification
- *Change Controller:* IETF
- *Reference:* Section 3.4
- *Notes:* N/A

7.1.1.4. Contacts Parameter

- *Parameter Name:* contacts
- *Parameter Description:* An array of URI with a reliable communication channel; typically email addresses
- *Change Controller:* IETF
- *Reference:* Section 3.3
- *Notes:* N/A

7.1.1.5. Expected User Agent Parameter

- *Parameter Name:* expected-user-agent
- *Parameter Description:* String or fragment patterns
- *Change Controller:* IETF
- *Reference:* Section 3.5

Notes: N/A

7.1.1.6. RFC9309 Product Token Parameter

Parameter Name: rfc9309-product-token

Parameter Description: Robots.txt product token your signature-agent satisfies.

Change Controller: IETF

Reference: Section 3.6

Notes: N/A

7.1.1.7. RFC9309 Compliance Parameter

Parameter Name: rfc9309-compliance

Parameter Description: Does your signature-agent respect robots.txt.

Change Controller: IETF

Reference: Section 3.7

Notes: N/A

7.1.1.8. Trigger Parameter

Parameter Name: trigger

Parameter Description: Fetcher/Crawler

Change Controller: IETF

Reference: Section 3.8

Notes: N/A

7.1.1.9. Purpose Parameter

Parameter Name: purpose

Parameter Description: Intended use for the collected data

Change Controller: IETF

Reference: Section 3.9

Notes: N/A

7.1.1.10. Targeted Content Parameter

Parameter Name: targeted-content

Parameter Description: Type of data your agent seeks

Change Controller: IETF

Reference: Section 3.10

Notes: N/A

7.1.1.11. Rate control Parameter

Parameter Name: rate-control

Parameter Description: How can an origin control your crawl rate

Change Controller: IETF

Reference: Section 3.11

Notes: N/A

7.1.1.12. Rate expectation Parameter

Parameter Name: rate-expectation

Parameter Description: Expected traffic and intensity

Change Controller: IETF

Reference: Section 3.12

Notes: N/A

7.1.1.13. Known URLs Parameter

Parameter Name: known-urls

Parameter Description: Predictable endpoint accessed

Change Controller: IETF

Reference: Section 3.13

Notes: N/A

7.1.1.14. JWKS URI Parameter

Parameter Name: jwks_uri

Parameter Description: URL of the signature agent’s key set. This can be an HTTP Message Signatures Directory for key discovery

Change Controller: IETF

Reference: Section 3.14

Notes: N/A

7.1.1.15. IP Address List URI Parameter

Parameter Name: ips_uri

Parameter Description: URL of the signature agent’s IP address list in [JAFAR] format.

Change Controller: IETF

Reference: Section 3.15

Notes: N/A

7.1.1.16. Keys Parameter

Parameter Name: keys

Parameter Description: JWKS Endpoint

Change Controller: IETF

Reference: Section 3.16

Notes: N/A

8. References

8.1. Normative References

[ABNF] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/rfc/rfc5234>.

[AIPREF-VOCAB] Keller, P. and M. Thomson, "A Vocabulary For Expressing AI Usage Preferences", Work in Progress, Internet-Draft, draft-ietf-aipref-vocab-06, 27 April 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-aipref-vocab-06>.

[CDDL] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/rfc/rfc8610>.

[DIRECTORY] Meunier, T. and S. Major, "HTTP Message Signatures Directory", Work in Progress, Internet-Draft, draft-meunier-http-message-signatures-directory-05, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-meunier-http-message-signatures-directory-05>.

[HTTP] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/rfc/rfc9110>.

[HTTP-CACHE] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Caching", STD 98, RFC 9111, DOI 10.17487/RFC9111, June 2022, <https://www.rfc-editor.org/rfc/rfc9111>.

[HTTP-MESSAGE-SIGNATURES] Backman, A., Ed., Richer, J., Ed., and M. Sporny, "HTTP Message Signatures", RFC 9421, DOI 10.17487/RFC9421, February 2024, <https://www.rfc-editor.org/rfc/rfc9421>.

[HTTP-MORE-STATUS-CODE] Nottingham, M. and R. Fielding, "Additional HTTP Status Codes", RFC 6585, DOI 10.17487/RFC6585, April 2012, <https://www.rfc-editor.org/rfc/rfc6585>.

[JAFAR] Illyes, G., "A JSON-Based Format for Publishing IP Ranges of Automated HTTP Clients", Work in Progress, Internet-Draft, draft-illyes-webbotauth-jafar-00, 21 April 2026, <https://datatracker.ietf.org/doc/html/draft-illyes-webbotauth-jafar-00>.

[JWK] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/rfc/rfc7517>.

[JWK-OKP] Liusvaara, I., "CFRG Elliptic Curve Diffie-Hellman (ECDH) and Signatures in JSON Object Signing and Encryption (JOSE)", RFC 8037, DOI 10.17487/RFC8037, January 2017, <https://www.rfc-editor.org/rfc/rfc8037>.

[JWK-THUMBPRINT] Jones, M. and N. Sakimura, "JSON Web Key (JWK) Thumbprint", RFC 7638, DOI 10.17487/RFC7638, September 2015, <https://www.rfc-editor.org/rfc/rfc7638>.

[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/rfc/rfc2119>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/rfc/rfc8126>.

[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/rfc/rfc8174>.

[WEB-LINKING] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/rfc/rfc8288>.

8.2. Informative References

[CBCP] Illyes, G., Khlewind, M., and K. Aj, "Crawler best practices", Work in Progress, Internet-Draft, draft-illyes-webbotauth-cbcp-00, 21 April 2026, <https://datatracker.ietf.org/doc/html/draft-illyes-webbotauth-cbcp-00>.

[DATAURL] Masinter, L., "The "data" URL scheme", RFC 2397, DOI 10.17487/RFC2397, August 1998, <https://www.rfc-editor.org/rfc/rfc2397>.

[OAUTH-BEARER] Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, DOI 10.17487/RFC6750, October 2012, <https://www.rfc-editor.org/rfc/rfc6750>.

[OPENID-CONNECT-DISCOVERY] "OpenID Connect Discovery 1.0", n.d., <https://openid.net/specs/openid-connect-discovery-1_0.html>.

[PSK-TLS] Housley, R., Hoyland, J., Sethi, M., and C. A. Wood, "Guidance for External Pre-Shared Key (PSK) Usage in TLS", RFC 9257, DOI 10.17487/RFC9257, July 2022, <https://www.rfc-editor.org/rfc/rfc9257>.

[RATELIMIT-HEADER] Polli, R., Ruiz, A. M., and D. Miller, "RateLimit header fields for HTTP", Work in Progress, Internet-Draft, draft-ietf-httpapi-ratelimit-headers-11, 23 May 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-httpapi-ratelimit-headers-11>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

[ROBOTSTXT] Koster, M., Illyes, G., Zeller, H., and L. Sassman, "Robots Exclusion Protocol", RFC 9309, DOI 10.17487/RFC9309, September 2022, <https://www.rfc-editor.org/rfc/rfc9309>.

[UTF8] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/rfc/rfc3629>.

Appendix A. Test Vectors

TODO

Appendix B. Implementations

TODO

Acknowledgments

TODO

The editor would also like to thank the following individuals (listed in alphabetical order) for feedback, insight, and implementation of this document -

Changelog

v02

* Add ips_uri parameter so client can expose IP addresses
* Add optional jwks_uri parameter to separate key material from metadata
* Fix inline data URL example in registry to use valid signature agent card
* Rename "Public list" to "Registry Endpoint" for clarity
* Add authentication guidance for private registry endpoints
* Add conditional GET (ETag/If-Modified-Since) guidance for efficient polling
* Add caching guidance referencing HTTP-CACHE
* Add change notification section: PUT/DELETE webhook signed with HTTP Message Signatures, OOB callback registration, pull as source of truth
* Expand Security Considerations: registry integrity via HTTP Message Signatures, private registry guidance
* Expand Privacy Considerations: customer list exposure, access patterns
* Add normative reference to RFC 8288 (Web Linking)

v01

* Add contributors

* Aligning registry draft with oauth dynamic client registration iana registry

* Add an about-url field

* Add ABNF for discovery of signature-agent card (registry)

* Add precisions about known URLs

* Add placeholder for image size

v00

* Initial draft

Authors' Addresses

Maxime Guerreiro
Cloudflare
Email: maxime.guerreiro@gmail.com

Ulas Kirazci
Amazon
Email: ulaskira@amazon.com

Thibault Meunier
Cloudflare
Email: ot-ietf@thibault.uk