Internet Engineering Task Force (IETF)
Request for Comments: 9209
Category: Standards Track
ISSN: 2070-1721

M. Nottingham
Fastly
P. Sikora
Google

June 2022

The Proxy-Status HTTP Response Header Field

draft-ietf-httpbis-proxy-status-08

Abstract

This document defines the Proxy-Status HTTP response field to convey the details of an intermediary's response handling, including generated errors.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841[1].

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9209[2].

Copyright Notice

ERROR: User-supplied boilerplate differs from auto-generated boilerplate (inserting auto-generated); Strings differ at position 367, 1st string ends in: [[[ection 2 of RFC 7841.Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9209.Copyright NoticeCopyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.]]], 2nd string ends in: [[[ection 2 of RFC 78

[1] https://www.rfc-editor.org/rfc/rfc7841.html#section-2
[2] https://www.rfc-editor.org/info/rfc9209
[3] https://trustee.ietf.org/license-info

4111 https://www.rfc-editor.org/rfc/rfc7841.html#section-2.Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc920922 https://www.rfc-editor.org/info/rfc9209.Copyright NoticeCopyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info33 https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.]]] (at line 6)

Table of Contents

1. Introduction

HTTP intermediaries (see Section 3.7 of [HTTP]) -- including both forward proxies and gateways (also known as "reverse proxies") -- have become an increasingly significant part of HTTP deployments. In particular, reverse proxies and content delivery networks (CDNs) form part of the critical infrastructure of many websites.

Typically, HTTP intermediaries forward requests towards the origin server (inbound) and then forward their responses back to clients (outbound). However, if an error occurs before a response is obtained from an inbound server, the response is often generated by the intermediary itself.

HTTP accommodates these types of errors with a few status codes -- for example, 502 (Bad Gateway) and 504 (Gateway Timeout). However, experience has shown that more information is necessary to aid debugging and communicate what's happened to the client. Additionally, intermediaries sometimes want to convey additional information about their handling of a response, even if they did not generate it.

To enable these uses, Section 2 defines a new HTTP response field to allow intermediaries to convey details of their handling of a response. Section 2.1 enumerates the information that can be added to the field by intermediaries, which can be extended per Section 2.2. Section 2.3 defines a set of error types for use when a proxy encounters an issue when obtaining a response for the request; these can likewise be extended per Section 2.4.

1.1. Notational Conventions

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 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses the following terminology from Section 3 of [STRUCTURED-FIELDS] to specify syntax and parsing: List, String, Token, Integer, and Byte Sequence.

Note that in this specification, "proxy" is used to indicate both forward and reverse proxies, otherwise known as gateways. "Next hop" indicates the connection in the direction leading to the origin server for the request.

2. The Proxy-Status HTTP Field

The Proxy-Status HTTP response field allows an intermediary to convey additional information about its handling of a response and its associated request.

Its value is a List (see Section 3.1 of [STRUCTURED-FIELDS]). Each member of the List represents an intermediary that has handled the response. The first member represents the intermediary closest to the origin server, and the last member represents the intermediary closest to the user agent.

For example:

```
Proxy-Status: revproxy1.example.net, ExampleCDN
```

indicates that this response was handled first by revproxy1.example.net (a reverse proxy adjacent to the origin server) and then ExampleCDN.

Intermediaries determine when it is appropriate to add the Proxy-Status field to a response. Some might decide to append it to all responses, whereas others might only do so when specifically configured to or when the request contains a header field that activates a debugging mode.

Each member of the List identifies the intermediary that inserted the value and MUST have a type of either String or Token. Depending on the deployment, this might be a service name (but not a software or hardware product name; e.g., "ExampleCDN" is appropriate, but "ExampleProxy" is not because it doesn't identify the deployment), a hostname ("proxy-3.example.com"), an IP address, or a generated string.

Parameters of each member (per Section 3.1.2 of [STRUCTURED-FIELDS]) convey additional information about that intermediary's handling of the response and its associated request; see Section 2.1. While all of these parameters are OPTIONAL, intermediaries are encouraged to provide as much information as possible (but see Section 4 for security considerations in doing so).

When adding a value to the Proxy-Status field, intermediaries SHOULD preserve the existing members of the field to allow debugging of the entire chain of intermediaries handling the request unless explicitly configured to remove them (e.g., to prevent internal network details from leaking; see Section 4).

Origin servers MUST NOT generate the Proxy-Status field.

Proxy-Status MAY be sent as an HTTP trailer field. For example, if an intermediary is streaming a response and the inbound connection suddenly terminates, Proxy-Status can only be appended to the trailer section of the outbound message since the header section has already been sent. However, because it might be silently discarded along the path to the user agent (as is the case for all trailer fields; see Section 6.5 of [HTTP]), Proxy-Status SHOULD NOT be sent as a trailer field unless it is not possible to send it in the header section.

To allow recipients to reconstruct the relative ordering of Proxy-Status members conveyed in trailer fields with those conveyed in header fields, an intermediary MUST NOT send Proxy-Status as a trailer field unless it has also generated a Proxy-Status header field with the same member (although potentially different parameters) in that message.

For example, a proxy identified as 'ThisProxy' that receives a response bearing a header field:

```
Proxy-Status: SomeOtherProxy
```

would add its own entry to the header field:

```
Proxy-Status: SomeOtherProxy, ThisProxy
```

thus allowing it to append a trailer field:

```
Proxy-Status: ThisProxy; error=read_timeout
```

which would thereby allow a downstream recipient to understand that processing by 'SomeOtherProxy' occurred before 'ThisProxy'.

A client MAY promote the Proxy-Status trailer field into a header field by following these steps:

1. For each member trailer_member of the Proxy-Status trailer field value:
2. Remove the Proxy-Status trailer field if empty.

2.1. Proxy-Status Parameters

This section lists parameters that can be used on the members of the Proxy-Status field. Unrecognised parameters MUST be ignored.

2.1.1. error

The `error` parameter's value is a Token that is a proxy error type. When present, it indicates that the intermediary encountered an issue when obtaining this response.

The presence of some proxy error types indicates that the response was generated by the intermediary itself, rather than being forwarded from the origin server. This is the case when, for example, the origin server can't be contacted, so the proxy has to create its own response.

Other proxy error types can be added to (potentially partial) responses that were generated by the origin server or some other inbound server. For example, if the forward connection abruptly closes, an intermediary might add Proxy-Status with an appropriate error as a trailer field.

Proxy error types that are registered with a 'Response only generated by intermediaries' value of 'true' indicate that they can only occur in responses generated by the intermediary. If the value is 'false', the response might be generated by the intermediary or an inbound server.

Section 2.3 lists the proxy error types defined in this document; new ones can be defined using the procedure outlined in Section 2.4.

For example:

```
HTTP/1.1 504 Gateway Timeout
Proxy-Status: ExampleCDN; error=connection_timeout
```

indicates that this 504 response was generated by ExampleCDN due to a connection timeout when going forward.

Or:

```
HTTP/1.1 429 Too Many Requests
Proxy-Status: r34.example.net; error=http_request_error, ExampleCDN
```

indicates that this 429 (Too Many Requests) response was generated by r34.example.net, not the CDN or the origin.

When sending the error parameter, the most specific proxy error type SHOULD be sent, provided that it accurately represents the error condition. If an appropriate proxy error type is not defined, there are a number of generic error types (e.g., proxy_internal_error, http_protocol_error) that can be used. If they are not suitable, consider registering a new proxy error type (see Section 2.4).

Each proxy error type has a recommended HTTP status code. When generating an HTTP response containing the `error`, its HTTP status code SHOULD be set to the recommended HTTP status code. However, there may be circumstances (e.g., for backwards compatibility with previous behaviours, a status code has already been sent) when another status code might be used.

Proxy error types can also define any number of extra parameters for use with that type. Their use, like all parameters, is optional. As a result, if an extra parameter is used with a proxy error type for which it is not defined, it will be ignored.

2.1.2. next-hop

The `next-hop` parameter's value is a String or Token that identifies the intermediary or origin server selected (and used, if contacted) to obtain this response. It might be a hostname, IP address, or alias.

For example:

```
Proxy-Status: cdn.example.org; next-hop=backend.example.org:8001
```

indicates that cdn.example.org used backend.example.org:8001 as the next hop for this request.

2.1.3. next-protocol

The `next-protocol` parameter's value indicates the Application-Layer Protocol Negotiation (ALPN) protocol identifier [RFC7301] of the protocol used by the intermediary to connect to the next hop when obtaining this response.

The value MUST be either a Token or Byte Sequence representing a TLS ALPN Protocol ID (see <https://www.iana.org/assignments/tls-extensiontype-values#alpn-protocol-ids>). If the protocol identifier is able to be expressed as a Token using ASCII encoding, that form MUST be used.

For example:

```
Proxy-Status: "proxy.example.org"; next-protocol=h2
```

Note that the ALPN identifier is being used here to identify the protocol in use; it may or may not have been actually used in the protocol negotiation.

2.1.4. received-status

The `received-status` parameter's value indicates the HTTP status code that the intermediary received from the next-hop server when obtaining this response.

The value MUST be an Integer.

For example:

```
Proxy-Status: ExampleCDN; received-status=200
```

2.1.5. details

The `details` parameter's value is a String containing additional information not captured anywhere else. This can include implementation-specific or deployment-specific information.

For example:

```
Proxy-Status: proxy.example.net; error="http_protocol_error";
              details="Malformed response header: space before colon"
```

2.2. Defining New Proxy-Status Parameters

New Proxy-Status parameters can be defined by registering them in the "HTTP Proxy-Status Parameters" registry.

Registration requests are reviewed and approved by Expert Review, per [RFC8126], Section 4.5. A specification document is appreciated but not required.

The expert(s) should consider the following factors when evaluating requests:

- Community feedback
- If the value is sufficiently well defined
- Generic parameters are preferred over vendor-specific, application-specific, or deployment-specific values. If a generic value cannot be agreed upon in the community, the parameter's name should be correspondingly specific (e.g., with a prefix that identifies the vendor, application, or deployment).
- Parameter names should not conflict with registered extra parameters in the "HTTP Proxy Error Types" registry.

Registration requests should use the following template:

Name: [a name for the Proxy-Status parameter that matches key]

Description: [a description of the parameter semantics and value]

Reference: [to a specification defining this parameter; optional]

See the registry at <https://www.iana.org/assignments/http-proxy-status> for details on where to send registration requests.

2.3. Proxy Error Types

This section lists the proxy error types defined by this document. See Section 2.4 for information about defining new proxy error types.

Note that implementations might not produce all proxy error types. The set of types below is designed to map to existing states in implementations and therefore may not be applicable to some.

2.3.1. DNS Timeout

Name: dns_timeout

Description: The intermediary encountered a timeout when trying to find an IP address for the next-hop hostname.

Extra Parameters: None

Recommended HTTP Status Code: 504

Response Only Generated by Intermediaries: true

Reference: RFC 9209

2.3.2. DNS Error

Name: dns_error

Description: The intermediary encountered a DNS error when trying to find an IP address for the next-hop hostname.

Extra Parameters:

rcode: A String conveying the DNS RCODE that indicates the error type. See [RFC8499], Section 3.

info-code: An Integer conveying the Extended DNS Error Code INFO-CODE. See [RFC8914].

Recommended HTTP Status Code: 502

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.3. Destination Not Found

Name: destination_not_found

Description: The intermediary cannot determine the appropriate next hop to use for this request; for example, it may not be configured. Note that this error is specific to gateways, which typically require specific configuration to identify the "backend" server; forward proxies use in-band information to identify the origin server.

Extra Parameters: None

Recommended HTTP Status Code: 500

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.4. Destination Unavailable

Name: destination_unavailable

Description: The intermediary considers the next hop to be unavailable; e.g., recent attempts to communicate with it may have failed, or a health check may indicate that it is down.

Extra Parameters: None

Recommended HTTP Status Code: 503

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.5. Destination IP Prohibited

Name: destination_ip_prohibited

Description: The intermediary is configured to prohibit connections to the next-hop IP address.

Extra Parameters: None

Recommended HTTP Status Code: 502

Response Only Generated by Intermediaries: true

Reference: RFC 9209

2.3.6. Destination IP Unroutable

Name: destination_ip_unroutable

Description: The intermediary cannot find a route to the next-hop IP address.

Extra Parameters: None

Recommended HTTP Status Code: 502

Response Only Generated by Intermediaries: true

Reference: RFC 9209

2.3.7. Connection Refused

Name: connection_refused

Description: The intermediary's connection to the next hop was refused.

Extra Parameters: None

Recommended HTTP Status Code: 502

Response Only Generated by Intermediaries: true

Reference: RFC 9209

2.3.8. Connection Terminated

Name: connection_terminated

Description: The intermediary's connection to the next hop was closed before a complete response was received.

Extra Parameters: None

Recommended HTTP Status Code: 502

Response Only Generated by Intermediaries: false

Reference: RFC 9209

2.3.9. Connection Timeout

Name: connection_timeout

Description: The intermediary's attempt to open a connection to the next hop timed out.

Extra Parameters: None

Recommended HTTP Status Code: 504

Response Only Generated by Intermediaries: true

Reference: RFC 9209

2.3.10. Connection Read Timeout

Name: connection_read_timeout

Description: The intermediary was expecting data on a connection (e.g., part of a response) but did not receive any new data in a configured time limit.

Extra Parameters: None

Recommended HTTP Status Code: 504

Response Only false

Generated by Intermediaries: no

Reference: RFC 9209

2.3.11. Connection Write Timeout

Name: connection_write_timeout

Description: The intermediary was attempting to write data to a connection but was not able to (e.g., because its buffers were full).

Extra Parameters: None

Recommended HTTP Status Code: 504

Response Only: false

Generated by Intermediaries: no

Reference: RFC 9209

2.3.12. Connection Limit Reached

Name: connection_limit_reached

Description: The intermediary is configured to limit the number of connections it has to the next hop, and that limit has been exceeded.

Extra Parameters: None

Recommended HTTP Status Code: 503

Response Only: no

Generated by Intermediaries: no

Reference: RFC 9209

2.3.13. TLS Protocol Error

Name: tls_protocol_error

Description: The intermediary encountered a TLS error when communicating with the next hop, either during the handshake or afterwards.

Extra Parameters: None

Recommended HTTP Status Code: 502

Response Only: false

Generated by Intermediaries: true

Reference: RFC 9209

Notes: Not appropriate when a TLS alert is received; see tls_alert_received.

2.3.14. TLS Certificate Error

Name: tls_certificate_error

Description: The intermediary encountered an error when verifying the certificate presented by the next hop.

Extra Parameters: None

Recommended HTTP Status Code: 502

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.15. TLS Alert Received

Name: tls_alert_received

Description: The intermediary received a TLS alert from the next hop.

Extra Parameters:

alert_id: An Integer containing the applicable value from the "TLS Alerts" registry. See [TLS].

alert_message: A Token or String containing the applicable description string from the "TLS Alerts" registry. See [TLS].

Recommended HTTP Status Code: 502

Response Only: false

Generated by Intermediaries: true

Reference: RFC 9209

2.3.16. HTTP Request Error

Name: http_request_error

Description: The intermediary is generating a client (4xx) response on the origin's behalf. Applicable status codes include (but are not limited to) 400, 403, 405, 406, 408, 411, 413, 414, 415, 416, 417, and 429.

Extra Parameters: status-code: An Integer containing the generated status code.

status-phrase: A String containing the generated status phrase.

Recommended HTTP Status Code: The applicable 4xx status code

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

Notes: This type helps distinguish between responses generated by intermediaries from those generated by the origin.

2.3.17. HTTP Request Denied

Name: http_request_denied

Description: The intermediary rejected the HTTP request based on its configuration and/or policy settings. The request wasn't forwarded to the next hop.

Extra Parameters: None

Recommended HTTP Status Code: 403

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.18. HTTP Incomplete Response

Name: http_response_incomplete

Description: The intermediary received an incomplete response to the request from the next hop.

Extra Parameters: None

Recommended HTTP: 502

Status Code:

Response Only:

false

Generated by Intermediaries:

Reference:

RFC 9209

2.3.19. HTTP Response Header Section Too Large

Name:

http_response_header_section_size

Description:

The intermediary received a response to the request whose header section was considered too large.

Extra Parameters:

header-section-size:

An Integer indicating how large the received headers were. Note that they might not be complete; i.e., the intermediary may have discarded or refused additional data.

Recommended HTTP Status Code:

502

Response Only:

false

Generated by Intermediaries:

Reference:

RFC 9209

2.3.20. HTTP Response Header Field Line Too Large

Name:

http_response_header_size

Description:

The intermediary received a response to the request containing an individual header field line that was considered too large.

Extra Parameters:

header-name:

A String indicating the name of the header field that triggered the error.

header-size:

An Integer indicating the size of the header field that triggered the error.

Recommended HTTP Status Code:

502

Response Only:

false

Generated by Intermediaries:

Reference:

RFC 9209

2.3.21. HTTP Response Body Too Large

Name: http_response_body_size

Description: The intermediary received a response to the request whose body was considered too large.

Extra Parameters:
- body_size: An Integer indicating how large the received body was. Note that it may not have been complete; i.e., the intermediary may have discarded or refused additional data.

Recommended HTTP Status Code: 502

Response Only: false

Generated by Intermediaries: true

Reference: RFC 9209

2.3.22. HTTP Response Trailer Section Too Large

Name: http_response_trailer_section_size

Description: The intermediary received a response to the request whose trailer section was considered too large.

Extra Parameters:
- trailer_section_size: An Integer indicating how large the received trailers were. Note that they might not be complete; i.e., the intermediary may have discarded or refused additional data.

Recommended HTTP Status Code: 502

Response Only: false

Generated by Intermediaries: true

Reference: RFC 9209

2.3.23. HTTP Response Trailer Field Line Too Large

Name: http_response_trailer_size

Description: The intermediary received a response to the request containing an individual trailer field line that was considered too large.

Extra Parameters:
- trailer_name: A String indicating the name of the trailer field that triggered the error.
- trailer_size: An Integer indicating the size of the trailer field that triggered the error.

Recommended HTTP Status Code: 502

Response Only: false

Generated by Intermediaries:

Reference: RFC 9209

2.3.24. HTTP Response Transfer-Coding Error

Name: http_response_transfer_coding

Description: The intermediary encountered an error decoding the transfer coding of the response.

Extra Parameters: coding: A Token containing the specific coding (from the "HTTP Transfer Coding Registry") that caused the error.

Recommended HTTP Status Code: 502

Response Only: false

Generated by Intermediaries:

Reference: RFC 9209

2.3.25. HTTP Response Content-Coding Error

Name: http_response_content_coding

Description: The intermediary encountered an error decoding the content coding of the response.

Extra Parameters: coding: A Token containing the specific coding (from the "HTTP Content Coding Registry") that caused the error.

Recommended HTTP Status Code: 502

Response Only: false

Generated by Intermediaries:

Reference: RFC 9209

2.3.26. HTTP Response Timeout

Name: http_response_timeout

Description: The intermediary reached a configured time limit waiting for the complete response.

Extra Parameters: None

Recommended HTTP

Status Code:

Response Only:

false

Generated by Intermediaries:

Reference:

RFC 9209

2.3.27. HTTP Upgrade Failed

Name:

http_upgrade_failed

Description:

The process of negotiating an upgrade of the HTTP version between the intermediary and the next hop failed.

Extra Parameters:

None

Recommended HTTP Status Code:

502

Response Only:

true

Generated by Intermediaries:

Reference:

RFC 9209

2.3.28. HTTP Protocol Error

Name:

http_protocol_error

Description:

The intermediary encountered an HTTP protocol error when communicating with the next hop. This error should only be used when a more specific one is not defined.

Extra Parameters:

None

Recommended HTTP Status Code:

502

Response Only:

false

Generated by Intermediaries:

Reference:

RFC 9209

2.3.29. Proxy Internal Response

Name:

proxy_internal_response

Description:

The intermediary generated the response itself without attempting to connect to the next hop.

Extension Parameters: None

Recommended HTTP Status Code: The most appropriate status code for the response

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.30. Proxy Internal Error

Name: proxy_internal_error

Description: The intermediary encountered an internal error unrelated to the origin.

Extension Parameters: None

Recommended HTTP Status Code: 500

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.31. Proxy Configuration Error

Name: proxy_configuration_error

Description: The intermediary encountered an error regarding its configuration.

Extension Parameters: None

Recommended HTTP Status Code: 500

Response Only: true

Generated by Intermediaries: true

Reference: RFC 9209

2.3.32. Proxy Loop Detected

Name: proxy_loop_detected

Description: The intermediary tried to forward the request to itself, or a loop has been detected using different means (e.g., [RFC8586]).

Extra Parameters: None

Recommended HTTP Status Code: 502

Response Only Generated by Intermediaries: true

Reference: RFC 9209

2.4. Defining New Proxy Error Types

New proxy error types can be defined by registering them in the "HTTP Proxy Error Types" registry.

Registration requests are reviewed and approved by Expert Review, per [RFC8126], Section 4.5. A specification document is appreciated but not required.

The expert(s) should consider the following factors when evaluating requests:

- Community feedback
- If the value is sufficiently well-defined
- Generic types are preferred over vendor-specific, application-specific, or deployment-specific values. If a generic value cannot be agreed upon in the community, the type's name should be correspondingly specific (e.g., with a prefix that identifies the vendor, application, or deployment).
- Extra parameters should not conflict with registered Proxy-Status parameters.

Registration requests should use the following template:

Name: [a name for the proxy error type that is of type Token]

Description: [a description of the conditions that generate the proxy error type]

Extra Parameters: [zero or more optional parameters, along with their allowable Structured Type(s)]

Recommended HTTP Status Code: [the appropriate HTTP status code for this entry]

Response Only Generated by Intermediaries: [true or 'false']

Reference: [to a specification defining this error type; optional]

Notes: [optional]

If the proxy error type might occur in responses that are not generated by the intermediary -- for example, when an error is detected as the response is streamed from a forward connection, causing a Proxy-Status trailer field to be appended -- the 'Response only generated by intermediaries' should be 'false'. If the proxy error type only occurs in responses that are generated by the intermediary, it should be 'true'.

See the registry at <https://www.iana.org/assignments/http-proxy-status> for details on where to send registration requests.

3. IANA Considerations

IANA has created the "HTTP Proxy-Status Parameters" registry and the "HTTP Proxy Error Types" registry at <https://www.iana.org/assignments/http-proxy-status> and has populated them with the types defined in Sections 2.1 and 2.3 respectively; see Sections 2.2 and 2.4 for their associated procedures.

Additionally, the following entry has been added to the "Hypertext Transfer Protocol (HTTP) Field Name Registry":

Field name:	Proxy-Status
Status:	permanent
Specification document(s):	RFC 9209
Comments:	

4. Security Considerations

One of the primary security concerns when using Proxy-Status is leaking information that might aid an attacker. For example, information about the intermediary's configuration and backend topology can be exposed, allowing attackers to directly target backend services that are not prepared for high traffic volume or malformed inputs. Some information might only be suitable to reveal to authorized parties.

As a result, care needs to be taken when deciding to generate a Proxy-Status field and what information to include in it. Note that intermediaries are not required to generate a Proxy-Status field in any response and can conditionally generate them based upon request attributes (e.g., authentication tokens, IP address).

Likewise, generation of all parameters is optional, as is the generation of the field itself. Also, the field's content is not verified; an intermediary can claim certain actions (e.g., sending a request over an encrypted channel) but fail to actually do that.

5. References

5.1. Normative References

[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/info/rfc9110>.

[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>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/info/rfc7301>.

[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/info/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/info/rfc8174>.

[RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <https://www.rfc-editor.org/info/rfc8499>.

[RFC8914] Kumari, W., Hunt, E., Arends, R., Hardaker, W., and D. Lawrence, "Extended DNS Errors", RFC 8914, DOI 10.17487/RFC8914, October 2020, <https://www.rfc-editor.org/info/rfc8914>.

[STRUCTURED-FIELDS] Nottingham, M. and P-H. Kamp, "Structured Field Values for HTTP", RFC 8941, DOI 10.17487/RFC8941, March 2021, <https://www.rfc-editor.org/info/rfc8941>.

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

5.2. Informative References

[RFC8586] Ludin, S., Nottingham, M., and N. Sullivan, "Loop Detection in Content Delivery Networks (CDNs)", RFC 8586, DOI 10.17487/RFC8586, April 2019, <https://www.rfc-editor.org/info/rfc8586>.

Authors' Addresses

Mark Nottingham
Fastly
Prahran
Australia
EMail: mnot@mnot.net
URI: https://www.mnot.net/

Piotr Sikora
Google
EMail: piotrsikora@google.com