Workgroup: ohai
Internet-Draft:
draft-rdb-ohai-feedback-to-proxy-01
Published: 20 March 2022
Intended Status: Standards Track
Expires: 21 September 2022
Authors: T. Reddy D. Wing M. Boucadair
Akamai Citrix Orange

Oblivious Proxy Feedback

Abstract

To provide equitable service to clients, servers often rate-limit incoming requests, often based upon the source IP address. However, oblivious HTTP removes the ability for the server to distinguish amongst clients so the server can only rate-limit traffic from the oblivious proxy. This harms all clients behind that oblivious proxy.

This specification provides feedback from a server to an oblivious proxy, enabling the oblivious proxy to rate-limit incoming requests from clients. Cooperating oblivious proxies can thus provide more equitable service to their distinguishable clients without triggering rate-limiting on the request resource or the target resource that would impact all clients behind that Oblivious proxy.

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 21 September 2022.

Table of Contents

1. Introduction

Oblivious HTTP [I-D.ietf-ohai-ohttp] describes a method of encapsulation for binary HTTP messages [BINARY] using Hybrid Public Key Encryption (HPKE; [HPKE]). This protects the content of both requests and responses and enables a deployment architecture that can separate the identity of a requester from the request. This scheme requires that servers and proxies explicitly support it. The server is susceptible to attacks described below, but the server cannot take any mitigation action per client to protect itself from various attacks -- the server can only take mitigation actions per oblivious proxy. Rate-limiting traffic from an oblivious proxy impacts all clients behind that proxy -- both misbehaving clients and well-behaved clients.

Attacks against the Request and Target Resources can be classified into three primary categories:

1. A client sends a malformed encapsulated request causing decryption failure or decryption overload failure on the oblivious request resource. This causes the oblivious request

resource to send an error status code back to the oblivious proxy.

2. A client sends an HTTP transaction that causes an HTTP error on the oblivious target resource. This might be a malformed HTTP request, or request for a missing resource.,

3. HTTP flood: A botnet performing an HTTP flood attack against a victim's server. Because each bot in a botnet makes seemingly legitimate network requests the traffic is not spoofed and may appear "normal" in origin. This might be too many requests from a single client, too many requests from the clients behind the same oblivious proxy or too many requests from all clients on the Internet.

This document defines how an overload indication is communicated to an oblivious proxy so that this proxy can rate limit transactions by overzealous or misbehaving clients, allowing the oblivious proxy to continue servicing well-behaved clients to that same oblivious target resource.

"RateLimit Fields for HTTP" specification [I-D.ietf-httpapi-ratelimit-headers] allows servers to publish current service limits to clients, whereas this draft allows servers to publish current service limits to oblivious proxies. The former specification allows clients to shape their request policy and avoid being throttled out, whereas this specification allows oblivious proxies to shape their request policy and avoid being throttled out.

2. Terminology

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.

This document makes use of the terms defined in [I-D.ietf-ohai-ohttp].

3. Ohai-Proxy-Feedback Header

The "Ohai-Proxy-Feedback" header field is defined in this specification. The Ohai-Proxy-Feedback header provides feedback information from the request resource or target resource to the proxy in the HTTP response. The proxy MUST remove the Ohai-Proxy-Feedback header before sending the HTTP response containing the encapsulated response to the client. If the feedback information is generated by the request resource before removing the protection (including being unable to remove encapsulation for any reason)(see

Section 6.2 of [I-D.ietf-ohai-ohttp]), it will result in the Ohai-Proxy-Feedback Header added in the status code being sent without protection in response to the POST request from the client.

Figure 1 describes the syntax (Augmented Backus-Naur Form) of the header field, using the grammar defined in [RFC5234] and the rules defined in Section 3.2 of [RFC7230]. The field values of the header field conform to the same rules.

```
Ohai-Proxy-Feedback = feedback-parameter *( OWS ";" OWS feedback-param
feedback-parameter   =
        feedback-parameter-name [ "=" feedback-parameter-value ]
feedback-parameter-name = registered-token
registered-token = token
feedback-parameter-value = 1*DIGIT
```

Figure 1: Ohai-Proxy-Feedback Header Syntax

[[NOTE: CHECK IF WE CAN REUSE THE STRUCTURED FIELDS IN RFC 8941]]

Optional white space (OWS) is used as defined in Section 3.2.3 of [RFC7230] and token is used as defined in Section 3.2.6 of [RFC7230].

The overall processing of the parameters is discussed below:

1. The order of appearance of the parameters is not significant.

2. A given parameter MUST NOT appear more than once in the Ohai-Proxy-Feedback header.

3. Parameters are either optional or required, as explicited in their definitions.

4. Parameter names are case insensitive.

5. Proxies MUST ignore any parameters or values, that do not conform to the syntax defined in this specification. In particular, proxies must not attempt to fix malformed parameters or parameter values.

6. If the parameter is not recognized by the proxy, it MUST be ignored by the proxy.

4. Ohai-Proxy-Feedback Header Parameters

The feedback information includes the following parameters:

RateLimit-p-Limit:

It indicates the maximum number of requests that the server is willing to accept from the proxy. This is an optional attribute.

RateLimit-p-Reset: It indicates the number of seconds until the maximum number of requests quota resets for the proxy. This is an optional attribute.

RateLimit-p-outstanding-Limit: It indicates the maximum number of outstanding requests that the server is willing to accept from the proxy. This is an optional attribute.

RateLimit-p-outstanding-Reset: It indicates the number of seconds until the maximum number of outstanding requests quota resets for the proxy. This is an optional attribute.

RateLimit-Limit: It indicates the maximum number of requests that the server is willing to accept from the offending client. It is defined in Section 5.1 of [I-D.ietf-httpapi-ratelimit-headers]. This is an optional attribute.

RateLimit-Reset: It indicates the number of seconds until the maximum number of requests quota resets for the offending client. It is defined in Section 5.3 of [I-D.ietf-httpapi-ratelimit-headers]. This is an optional attribute.

RateLimit-Outstanding-Limit: It indicates the maximum number of outstanding requests that the server is willing to accept from the offending client. This is an optional attribute.

RateLimit-Outstanding-Reset: It indicates the number of seconds until the maximum number of outstanding requests quota resets for the offending client. This is an optional attribute.

TBD: Use of any other parameters like min-encap-request-size and max-encap-request-size to defend from garbled encapsulated requests.

TBD: RateLimit-Outstanding-Limit parameter is not specific to OHAI and it can be added to [I-D.ietf-httpapi-ratelimit-headers].

Note that we plan to use short parameter names in future versions of the draft as recommended by [I-D.ietf-httpbis-bcp56bis].

The above parameters are in the form of a "name=value" pair.

The feedback information header MUST include at least one of the parameters RateLimit-p-Limit, RateLimit-p-outstanding-Limit, RateLimit-Limit, or RateLimit-Outstanding-Limit.

The RateLimit-Limit, RateLimit-Reset, RateLimit-Outstanding-Limit, and RateLimit-Outstanding-Reset parameters are set if the client is

attacking the server (e.g., the client using an abnormal header that matches an attack rule).

Example: A target resource receives a malformed message and generates an HTTP response with a 400 status code. It adds the "Ohai-Proxy-Feedback" header with the appropriate rate limit values to the 400 response and then sends the 400 response to the request resource. The request resource copies the "Ohai-Proxy-Feedback" header from the 400 response, removes the "Ohai-Proxy-Feedback" header from the 400 response, and encapsulates the 400 response. The request resource sends a single 200 response along with the copied "Ohai-Proxy-Feedback" header in the 200 response and encapsulated 400 response as the response content.

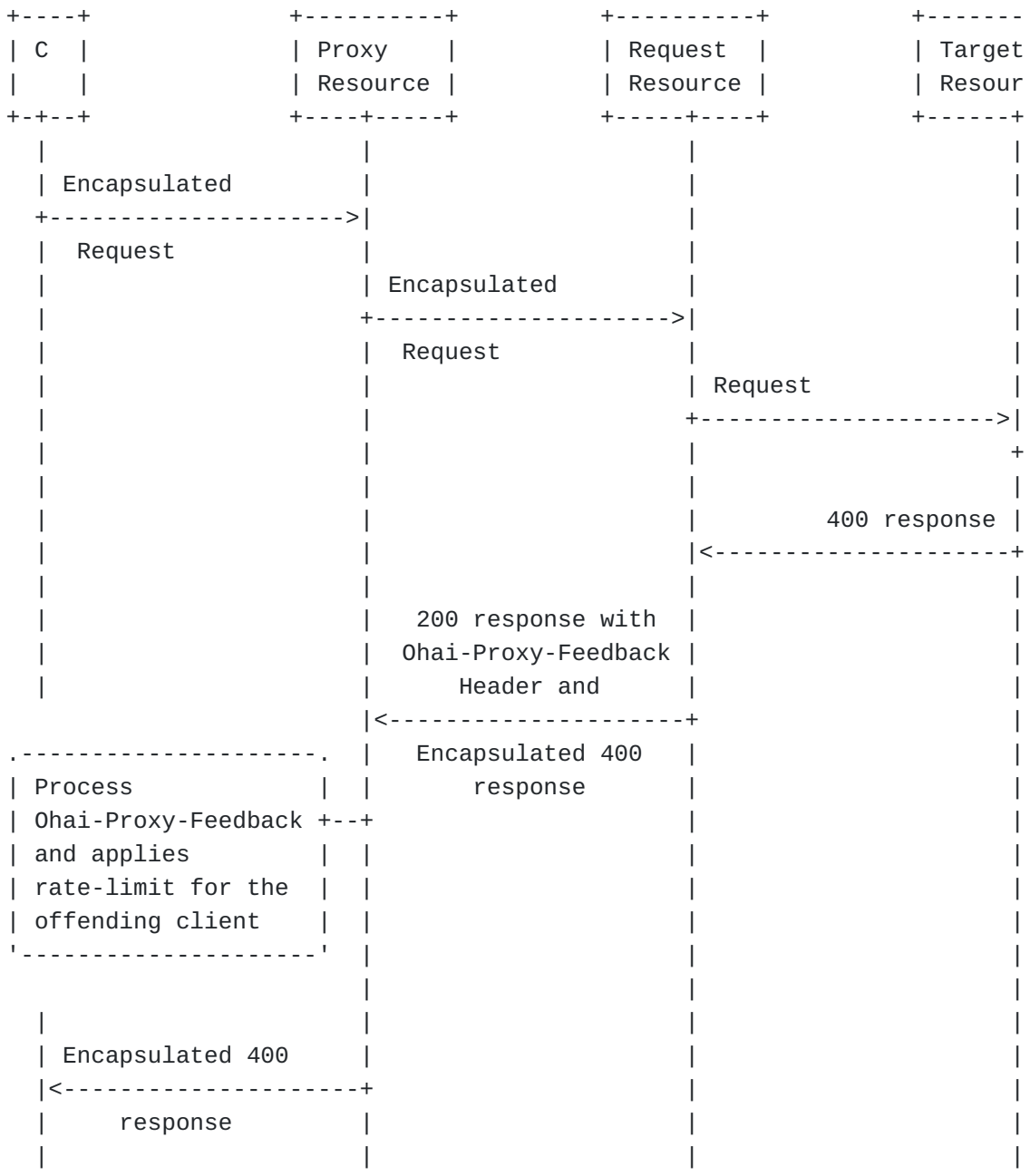

Figure 2: An Example of Feedback to Proxy

The response constructed by the oblivious request resource is depicted below:

```
HTTP/1.1 200 OK
Date: Wed, 27 March 2022 04:45:07 GMT
Cache-Control: private, no-store
Ohai-Proxy-Feedback: RateLimit-p-Limit=10000; RateLimit-p-Reset=600
Content-Type: message/ohttp-res
Content-Length: 38 <content is the encapsulated 400 response>
```

5. Request or Target Resource Generating Ohai-Proxy-Feedback Header

When an overload is experienced by the request or target resource it adds the Ohai-Proxy-Feedback header and parameters to request load adjustment. For example, when a HTTP server itself identifies high frequency or high volume anomalies in the traffic directed to the server it would include the Ohai-Proxy-Feedback header. Ideally the Ohai-Proxy-Feedback header provides enough detail to the oblivious proxy to avoid the server rate limiting the oblivious proxy's IP address.

6. Proxy Processing of Ohai-Proxy-Feedback Header

When presented with a response that contains the Ohai-Proxy-Feedback Header, the proxy can process the parameters in the header and take appropriate actions. There is no mechanism for the proxy to indicate to the server that feedback information was processed or was ignored. The proxy can honor the rate indicated by the request resource/resource target. To that aim, the proxy may take appropriate additional actions such as (1) rate-limiting the requests from a client not to exceed requests per second (RateLimit-Limit) value (2) rate-limit the outstanding HTTP requests from a client not to exceed outstanding requests (RateLimit-Outstanding-Limit) value.

If the proxy ignores the feedback information, there is a risk that the overload may still be encountered by the request and target resources. More severe actions may be, then, taken at the server, e.g., block all the requests from this proxy for a given time duration.

7. Security Considerations

The security considerations for the Oblivious HTTP protocol are discussed in Section 8 of [I-D.ietf-ohai-ohttp]. The client needs to trust the proxy that it does not leak the client identity to the server. The target and request resources SHOULD convey the Ohai-Proxy-Feedback header to trusted oblivious proxy. However, if this oblivious proxy is not trusted, security risks discussed below may arise:

* If oblivious proxy and clients attacking the server are managed by an attacker, the attacker can use the Feedback information to identify the server has detected the attack and possibly change the attack strategy.

* The oblivious proxy can colloude with the attacking clients and leak the Feedback information to the clients.

8. IANA Considerations

8.1. Registration of new HTTP Header Field

8.1.1. Ohai-Proxy-Feedback Header

This section describes a header field for registration in the Permanent Message Header Field Registry [RFC3864].

Header field name
Feedback

Applicable protocol
http

Status
standard

Author/Change controller
IETF

Specification document(s)
RFC XXXX

Related information
This header field is only used for Oblivious HTTP.

8.1.2. Ohai-Proxy-Feedback Parameter Name Registry

This specification requests the creation of a new IANA registry for Feedback Parameter Names to be sent in the Feedback Header in accordance with the principles set out in [RFC5226].

As part of this registry IANA will maintain the following information:

Parameter Name
The name of the parameter.

9. Acknowledgements

Thanks to Lucas Pardue, Rich Salz and Brandon Williams for the discussion and comments.

10. References

10.1. Normative References

[BINARY] Thomson, M. and C. A. Wood, "Binary Representation of HTTP Messages", Work in Progress, Internet-Draft, draft-

ietf-httpbis-binary-message-01, 3 February 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-binary-message-01>.

[HPKE] Barnes, R. L., Bhargavan, K., Lipp, B., and C. A. Wood, "Hybrid Public Key Encryption", Work in Progress, Internet-Draft, draft-irtf-cfrg-hpke-12, 2 September 2021, <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-hpke-12>.

[I-D.ietf-httpapi-ratelimit-headers] Polli, R. and A. M. Ruiz, "RateLimit Fields for HTTP", Work in Progress, Internet-Draft, draft-ietf-httpapi-ratelimit-headers-03, 7 March 2022, <https://www.ietf.org/archive/id/draft-ietf-httpapi-ratelimit-headers-03.txt>.

[I-D.ietf-ohai-ohttp] Thomson, M. and C. A. Wood, "Oblivious HTTP", Work in Progress, Internet-Draft, draft-ietf-ohai-ohttp-01, 15 February 2022, <https://www.ietf.org/archive/id/draft-ietf-ohai-ohttp-01.txt>.

[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>.

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, DOI 10.17487/RFC3864, September 2004, <https://www.rfc-editor.org/info/rfc3864>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <https://www.rfc-editor.org/info/rfc5226>.

[RFC5234] 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/info/rfc5234>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[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>.

10.2. Informative References

[I-D.ietf-httpbis-bcp56bis]
Nottingham, M., "Building Protocols with HTTP", Work in Progress, Internet-Draft, draft-ietf-httpbis-bcp56bis-15, 27 August 2021, <https://www.ietf.org/archive/id/draft-ietf-httpbis-bcp56bis-15.txt>.

Authors' Addresses

Tirumaleswar Reddy
Akamai
Embassy Golf Link Business Park
Bangalore 560071
Karnataka
India

Email: kondtir@gmail.com

Dan Wing
Citrix Systems, Inc.
4988 Great America Pkwy
Santa Clara, CA 95054
United States of America

Email: danwing@gmail.com

Mohamed Boucadair
Orange
35000 Rennes
France

Email: mohamed.boucadair@orange.com