Workgroup: ohai
Internet-Draft:
draft-rdb-ohai-feedback-to-proxy-02
Published: 23 May 2022
Intended Status: Standards Track
Expires: 24 November 2022
Authors: T. Reddy D. Wing M. Boucadair
Akamai Citrix Orange
R. Polli
Team Digitale, Italian Government

Oblivious Proxy Feedback

Abstract

To provide equitable service to clients, servers often rate-limit incoming requests, for example, 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 enables a server to convey rate-limit information to an oblivious proxy, which can use it to apply rate-limit policies on oblivious clients. Cooperating oblivious proxies can thus provide more equitable service to their distinguishable clients without impacting on all clients behind that oblivious proxy.

Table of Contents

1. Introduction

Oblivious HTTP [OHTTP] requires three parties to exchange HTTP messages: the client, the proxy, and the target (formally, the Oblivious Request Resource and Oblivious Target Resource). Oblivious HTTP enables a client to send requests to a target in such a way that the target cannot tell whether two requests came from the same client, and the proxy cannot see the contents of the requests.

Since oblivious clients are located behind a proxy, a target cannot distinguish between well-behaving and malicious clients: an unexpected behavior from one or more clients can then impact on all the intermediated clients, as described in Section 8.2.1 of [OHTTP]. This can be problematic when the target implements rate limiting policies based on an information masked by the intermediary, such as the source IP address.

This document defines a mechanism that allows Oblivious request and target resource to provide rate-limit information to an Oblivious proxy via the RateLimit fields defined in [RATELIMIT]. This is useful when such servers identify traffic anomalies or unexpected request volumes. The Oblivious proxy can then use this information to apply rate-limit policies on oblivious clients.

While [RATELIMIT] provides enough information to generic clients to shape their request policy and avoid being throttled out, this specification allows an Oblivious request and target resource to indicate their RateLimit information is intended for the Oblivious proxy (rather than to the client).

How an Oblivious proxy can use this information to avoid being throttled out or shape its request policy is outside the scope of this specification.

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.

The terms "content", "receiver", "request", and "response" are to be interpreted as described in [HTTP].

The terms "Encapsulated request", "Encapsulated response", "Oblivious proxy resource", "Oblivious request resource", "Oblivious target resource", and "Client" are to be interpreted as described in [OHTTP].

The collective term "Oblivious resource" indicates either an "Oblivious request resource" or an "Oblivious target resource".

The terms "quota policy", "service limit", "expiring limit", and "RateLimit fields" are to be interpreted as described in [RATELIMIT].

This document uses the Integer type from [STRUCTURED-FIELDS].

3. Providing RateLimit Information to an Oblivious Proxy

An Oblivious resource that uses RateLimit fields [RATELIMIT] to return service limit information MAY add the "ohttp-target" quota policy parameter defined in Section 4 to signal to the receiver that the associated quota policy is intended for an Oblivious proxy. For example, when an Oblivious target identifies a high frequency or

high volume anomalies in the HTTP requests it would include the "ohttp-target" parameter.

The term "Oblivious Proxy Feedback" denotes both the mechanism described in this specification and the complete set of RateLimit fields together with the "ohttp-target" parameter.

To know whether the RateLimit fields provides Oblivious Proxy Feedback (see Section 3.1), an Oblivious proxy MUST:

1. Identify the quota policy associated to the expiring limit.

2. Check whether the "ohttp-target" parameter is present and its syntax is correct.

In the example shown in Figure 1, the expiring limit value is "100", so the associated quota policy is the second one. This quota policy includes the "ohttp-target" parameter: this indicates that the RateLimit fields are intended for an Oblivious proxy.

```
RateLimit-Limit: 100, 10;w=1, 100;w=60;ohttp-target=1
RateLimit-Remaining: 8
RateLimit-Reset: 15
```

Figure 1: An Example of Oblivious Proxy Feedback.

4. The ohttp-target Quota Policy Parameter

4.1. ohttp-target Parameter

The following quota policy parameter is defined for the RateLimit-Limit field [RATELIMIT]:

ohttp-target: Indicates that the associated quota policy provides Oblivious Proxy Feedback. This parameter is OPTIONAL.

The "ohttp-target" parameter has the following syntax:

```
ohttp-target = sf-integer
```

Its value MUST be an Integer (Section 3.3.1 of [STRUCTURED-FIELDS]) and indicates whether the quota policy is applicable to all the clients that are serviced by the Oblivious proxy or applicable only to a specific client. The "ohttp-target" parameter MUST have one of the following values:

1: Indicates that RateLimit fields are applicable to all the clients that are serviced by the same Oblivious proxy.

2:

Indicates that RateLimit fields are applicable only to the offending client. For example, this value is used if the client is attacking the server (e.g., the client is using an abnormal header that matches an attack pattern). The Oblivious proxy can shadowban requests from the offending client for a certain duration instead of rate-limiting the requests when the client has a high ratio of malicious requests to legitimate requests.

Other values MUST cause the parameter to be ignored.

The "ohttp-target" parameter MUST NOT appear more than once in a quota policy. If the parameter is malformed or its value is invalid, it MUST be ignored, and the receiving Oblivious proxy MUST NOT attempt to fix neither the parameter nor its value. That is, the RateLimit fields must not be considered as providing Oblivious Proxy Feedback.

4.2. Processing the ohttp-target Parameter

An Oblivious proxy receiving RateLimit fields providing Oblivious Proxy Feedback will do the following:

1. It MUST remove the RateLimit fields from the response, since they are not intended to be forwarded to clients.

2. It MAY add a new set of RateLimit fields that are intended to be forwarded to a client.

An Oblivious request resource receiving RateLimit fields providing Oblivious Proxy Feedback will do the following:

1. It MUST remove the RateLimit fields from the HTTP response, since they are not intended to be forwarded to the client. It, then, encapsulates the HTTP response.

2. It MUST add the above RateLimit fields to the response containing the encapsulated response sent to the Oblivious proxy, so that the Oblivious proxy can access them.

If the RateLimit fields along with the "ohttp-target" parameter are generated by the oblivious request resource before removing the protection (including being unable to remove the encapsulation for any reason)(Section 6.2 of [OHTTP]), it will result in the RateLimit fields added in the response being sent without protection in response to a POST request from a client.

While this specification does not mandate specific traffic shaping actions for Oblivious proxies in addition to the ones indicated in [RATELIMIT], an Oblivious proxy failing to reshape traffic from a

specific client or from all the clients according to the received Oblivious Proxy Feedback can experience different levels of service denial by the Oblivious request and target resources. There is no explicit mechanism for an Oblivious proxy to indicate to the server that the rate-limit information was processed or was ignored.

5. The attack-severity Quota Policy Parameter

The following quota policy parameter is defined for the RateLimit-Limit field defined in [RATELIMIT]:

attack-severity: Is used by the Oblivious resource to convey the likeliness that an Oblivious request is malicious. This parameter is OPTIONAL.

```
attack-severity = sf-string
```

Note that sf-string is defined in Section 3.3.3 of [STRUCTURED-FIELDS].

The value of the "attack-severity" parameter is a String (Section 3.3.3 of [RFC8941]) that takes one of the values defined in [SEVERITY]. This parameter MUST NOT appear more than once in a quota policy. If the parameter is malformed or its value is invalid, the parameter MUST be ignored, and the proxies MUST NOT attempt to fix neither the parameter nor the value.

6. Use of The ohttp-target Quota Policy Parameters: An Example

The example depicted in Figure 2 illustrates the use of the "ohttp-target" parameter. An oblivious target resource receives a malformed message and uses the source IP address to identify that it was an oblivious HTTP request decapsulated by an oblivious request resource. The Oblivious target resource generates a 400 response and adds the RateLimit fields along with the "ohttp-target" quota policy parameter. The oblivious request resource proceeds as follows:

1. Copy the RateLimit fields from the original response.

2. Remove them from the original response before encapsulating it.

3. Generate a single 200 response containing the encapsulated response for the oblivious proxy resource along with the copied RateLimit fields.

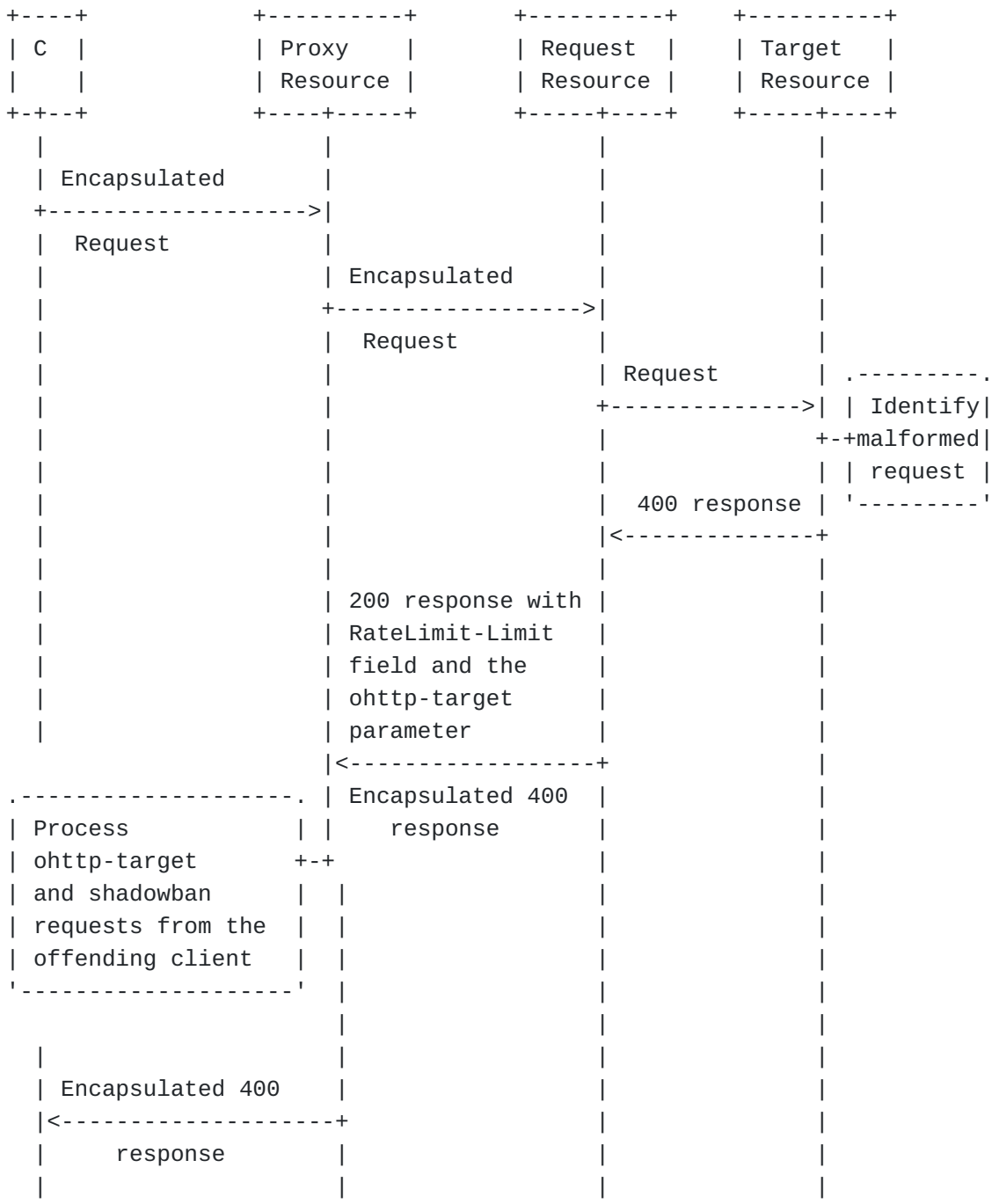

Figure 2: An Example of Ratelimit Feedback to Proxy

The response that is generated by the Oblivious request resource is depicted in Figure 3. This response includes an unregistered, informative "comment" quota policy parameter providing the rationale for the "attack- severity".

```
=============== NOTE: '\' line wrapping per RFC 8792 ================

  HTTP/1.1 200 OK
  Date: Wed, 27 March 2022 04:45:07 GMT
  Cache-Control: private, no-store
  RateLimit-Limit: 10,10;ohttp-target=2;attack-severity="high";\
comment="abnormal header matching a WAF rule"
  Content-Type: message/ohttp-res
  Content-Length: 38 <content is the encapsulated 400 response>
  ...encrypted content...
```

Figure 3: Example of a Response

7. Security Considerations

The security considerations for the Oblivious HTTP protocol (Section 8 of [OHTTP]) as well as the ones for RateLimit-Limit fields (Section 6 of [RATELIMIT]) apply. The following sub-sections discuss security considerations specific to this specification.

7.1. Client and Oblivous Proxy Collusion

While Oblivious HTTP relies upon an Oblivious proxy to prevent leaking the client identity to the Oblivious resources, it might be the case that the Oblivious proxy colludes with clients in attacking Oblivious resources. RateLimit fields might disclose operational capacity information useful to design denial of service attacks or to circumvent defensive measures put in place by the Oblivious resources (Section 6.2 of [RATELIMIT]). The Oblivious target and request resources SHOULD convey Oblivious Proxy Feedback only to trusted Oblivious proxies.

7.2. Attack Categories

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

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

3. A botnet performing an application layer denial of service attack (e.g. HTTP flood) against an Oblivious resource. Because each bot in a botnet makes seemingly legitimate network

requests the traffic may appear "normal" in origin, nonetheless as a whole it not only can saturate the Oblivious resources, but also makes appear the Oblivious proxy as an attacker. 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.

8. IANA Considerations

8.1. RateLimit Parameter Value Registrations

This specification requests IANA to add the following parameters to the "Hypertext Transfer Protocol (HTTP) RateLimit Parameters" registry defined in [RATELIMIT].

Field Name	Parameter Name	Description	Specification
RateLimit-Limit	ohttp-target	ohttp ratelimit	Section 3 of this document
RateLimit-Limit	attack-severity	ohttp ratelimit	Section 5 of this document

9. Acknowledgements

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

10. References

10.1. Normative References

[HTTP] Fielding, R. T., Nottingham, M., and J. Reschke, "HTTP Semantics", Work in Progress, Internet-Draft, draft-ietf-httpbis-semantics-19, 12 September 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-semantics-19>.

[OHTTP] Thomson, M. and C. A. Wood, "Oblivious HTTP", Work in Progress, Internet-Draft, draft-ietf-ohai-ohttp-01, 15 February 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-ohai-ohttp-01>.

[RATELIMIT] 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://datatracker.ietf.org/doc/html/draft-ietf-httpapi-ratelimit-headers-03>.

[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>.

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

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

10.2. Informative References

[SEVERITY] IANA, "Incident Object Description Exchange Format v2 (IODEF)", <https://www.iana.org/assignments/iodef2/iodef2.xhtml#businessimpact-severity>.

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

Roberto Polli
Team Digitale, Italian Government

Email: robipolli@gmail.com