Workgroup:
Oblivious HTTP Application Intermediation
Internet-Draft:
draft-wood-ohai-unreliable-ohttp-00
Published: 30 August 2022
Intended Status: Informational
Expires: 3 March 2023
Authors: R. Giles
Brave
C. A. Wood
Cloudflare

An Unreliable Oblivious HTTP Extension

Abstract

This document describes an extension to Oblivious HTTP (OHTTP) that supports unreliable application data transfer from Client to Target. Beyond enabling application uses that do not require explicit responses from the Target, such as privacy-preserving data collection, this extension allows the Oblivious Relay Resource to buffer, batch, and shuffle requests to Oblivious Gateway Resources as a way of amplifying end-to-end client privacy protections.

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://chris-wood.github.io/draft-unreliable-ohttp/draft-wood-ohai-unreliable-ohttp.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-wood-ohai-unreliable-ohttp/.

Discussion of this document takes place on the Oblivious HTTP Application Intermediation Working Group mailing list (mailto:ohai@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/ohai/. Subscribe at https://www.ietf.org/mailman/listinfo/ohai/.

Source for this draft and an issue tracker can be found at https://github.com/chris-wood/draft-unreliable-ohttp.

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 3 March 2023.

Table of Contents

1. Introduction

A typical HTTP transaction consists of a request and response between a Client and Target Resource. Oblivious HTTP ([OHTTP]) adds an Oblivious Relay Resource and Oblivious Gateway Resource between Client and Target Resource to help process HTTP transactions. In particular, requests flow from the Client to the Target through the Oblivious Relay Resource and Oblivious Gateway Resource in sequence.

In effect, an OHTTP transaction decouples the identity of the Client, i.e. its IP address, from the request to the Target Resource. Only the Client knows both its identity and the contents of the Target Resource request.

In a typical OHTTP transaction, Clients receive an Encapsulated Response from the Oblivious Gateway Resource containing the response from the Target Resource. This is useful for applications that require a response from the Target Resource. However, there are many settings in which Clients do not require a response from the Target Resource, including privacy-preserving data collection [STAR], publish-subscribe applications, and more generally applications which submit data under a "best effort" policy. Beyond these application use cases, unreliable requests also allow the relay to play a more active role towards improving client privacy, e.g. by batching, buffering, and shuffling requests to mitigate traffic analysis by network eavesdroppers or amplify local differential privacy protections used by clients [LOCALDP].

This document describes an extension to OHTTP that supports unreliable requests. An unreliable request is one wherein the client does not have explicit confirmation of receipt from the Target Resource, and therefore has limited application uses.

2. Motivation and Applicability

Unreliable application data transmission is sufficient for a number of applications, as discussed in Section 1. However, the primary motivations for this feature are agnostic to most applications. In particular, unreliable data transmission allows the Oblivious Gateway Resource to be deployed in a more performant and secure manner, and it also allows the Oblivious Relay Resource to be deployed to improve Client request privacy. We describe these motivating properties below.

2.1. Gateway Performance and Security

[STAR] is a proposed system for privacy-preserving data collection aimed at the heavy hitters problem. For privacy reasons, it is important that client reports, containing their individual measurements, are separated from any client identifying information, including their IP address. STAR can use OHTTP to send client reports, but it requires the Oblivious Gateway Resource to produce an encrypted acknowledgement to the clients for every report.

Depending on the Oblivious Gateway Resource implementation and scale of deployment, this can lead to reduced performance. It also requires the Oblivious Gateway Resource to have access to the

private key necessary to process the Encapsulated Request carrying a report and produce a response.

Unreliable data transmission would allow the Oblivious Gateway Resource to return an unencrypted acknowledgement of receipt, buffer Encapsulated Requests for future processing, and even allow the Oblivious Gateway Resource to operate without access to any private key material.

2.2. Relay Privacy Protections

In OHTTP, the Oblivious Relay Resource simply forwards Encapuslated Requests and Encapsulated Responses between Client and Oblivious Gateway Resource. Depending on the implementation, the Oblivious Relay Resource can introduce delays before forwarding each request or response. This can help mitigate traffic analysis by passive eavesdroppers observing traffic between the Oblivious Relay Resource and Oblivious Gateway Resource.

Unreliable data transmission gives the Oblivious Relay Resource more leeway in how these delays are introduced. In particular, the Oblivious Relay Resource can buffer Encapsulated Requests for an arbitrary amount of time, optionally shuffle them, and send them to the Oblivious Gateway Resource in batches. Beyond making traffic analysis by passive eavesdroppers more difficult, this is sometimes a necessary function for differential privacy protections [LOCALDP].

3. 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.

4. Unreliable Oblivious HTTP

Unreliable OHTTP extends the basic OHTTP protocol in the following ways:

1. It introduces a new Media Type for unreliable OHTTP responses that represent a "request acknowledgement message." A 202 Accepted response with this Content-Type signals that the corresponding Encapsulated Request was accepted and will be processed later.

2. It extends Client and Oblivious Relay Resource behavior to accept 202 Accepted responses when Clients opt in to receive unreliable OHTTP responses.

At a high level, an unreliable OHTTP request can be accepted by either the Oblivious Relay Resource or Oblivious Gateway Resource. In other words, the Oblivious Relay Resource and the Oblivious Gateway Resource can both buffer and acknowledge an Encapsulated Request. This end-to-end interaction is shown below.

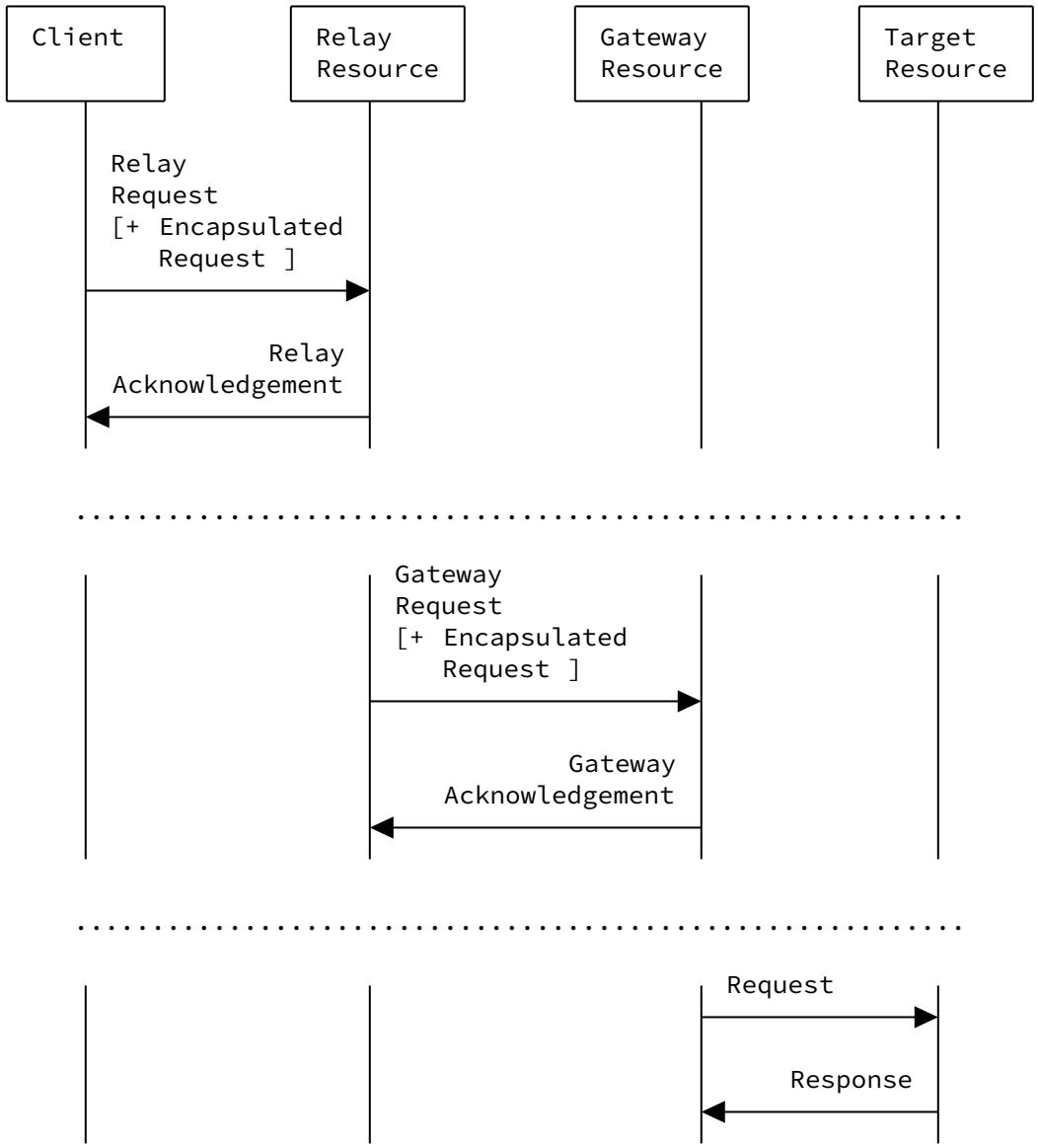

Figure 1: Overview of Unreliable Oblivious HTTP

A Client interacts with the Oblivious Relay Resource by constructing an Encapsulated Request as described in [OHTTP]. This Encapsulated Request is included as the content of a POST request to the Oblivious Relay Resource. Importantly, this request **MUST** include the "message/ohttp-ack" Media Type in the Accept header (see Section

6.1). The Client receives a 202 Accepted response with content type "message/ohttp-ack" and an empty body upon successful transmission of the request. If the Accept header also allows the normal "message/bhttp" content type, the client has left the choice of reliable or unreliable delivery up to the Relay and Gateway Resources.

Upon receipt of an unreliable OHTTP request from the Client, the Oblivious Relay Resource **MUST** reply with a 202 Accepted response with the "message/ohttp-ack" content type to the Client. It can buffer the request to be sent to the Oblivious Gateway Resource at some point in the future, or choose to forward it immediately. Similarly, upon receipt of an unreliable OHTTP request from the Oblivious Relay Resource, the Oblivious Gateway Resource **MUST** reply with a 202 Accepted response and the "message/ohttp-ack" content type to the Oblivious Relay Resource and buffer the request for decapsulation and processing at some point in the future.

4.1. Client Considerations

By the nature of this extension, unreliable OHTTP has some limitations for applications. In particular, Clients do not receive authenticated confirmation that their requests were processed by the Oblivious Gateway Resource. Moreover, Clients cannot implement any sort of retry mechanism in the event that their requests are too old. This means that applications using unreliable OHTTP should tolerate some amount of data loss.

This extension is triggered when the Client specifies message/ohttp-ack in the Accept header in a request sent to the Oblivious Relay Resource. Likewise, this extension can be disabled by specifying message/ohttp-res (or sending no Accept header at all), requiring the return of an Encapsulated Response according to [OHTTP].

A Client **MAY** signal that unreliable delivery is optional by setting a request header of Accept: message/* or Accept: */* which matches both the message/ohttp-res and message/ohttp-ack Media Types.

If a Client does not specify unreliable delivery by setting a compatible Accept header in its request, and receives a 406 Not Acceptable status code in the response from the Relay Resource, this means that the OHTTP configuration for that request only supports unreliable OHTTP. The client **MAY** then retry, this time allowing unreliable delivery.

4.2. Relay Considerations

Unreliable OHTTP allows the Oblivious Relay Resource to buffer Encapsulated Requests for future transmission to an Oblivious Gateway Resource. The relay can choose to buffer Client requests

until a sufficiently large number of requests is reached and then send all requests in a single batch to the gateway. Additionally, the relay can shuffle requests before forwarding them to the gateway to further obscure the request timing.

The choice of minimum buffer size is an implementation detail. Relays should take care to not introduce too much delay between when a request was received and when it is forwarded. Such delay may cause the gateway to drop the request due to a variety of reasons, e.g., because the gateway configuration changed or rotated or the decapsulated request became too old.

A Relay which supports unreliable OHTTP may be configured to enforce unreliable delivery. In such cases, if the Relay receives a normal OHTTP request, i.e., without an Accept header matching "message/ohttp-ack", it **SHOULD** respond with 406 Not Acceptable to signal this requirement to the client. If a Relay has forwarded an Encapsulated Request and receives a 406 Not Acceptable response, it **MUST** return the same status code in its response to the Client.

If a Relay receives a request from the Client which allows either reliable or unreliable OHTTP, because the Accept header is message/* or */*, it can choose to interpret it as one or the other, or it can delegate the decision to the Gateway Resource. If it chooses to interpret the request as an unreliable OHTTP request, and has already returned a 202 Accepted reponse to the Client, it **SHOULD** set the corresponding Accept: message/ohttp-ack header when it forwards to request to the Gateway Resource.

4.3. Gateway Considerations

Similar to the Oblivious Relay Resource, the Oblivious Gateway Resource can buffer requests for future processing. This can be helpful if the key material needed to process each request is isolated to a different process. Similar to the relay, the choice of how to buffer incoming requests is an implementation decision, and any delay in processing a request can increase the likelihood that the request is dropped or discarded due to it being stale.

If an Oblivious Gateway Resource requires unreliable delivery for a request, by implementation constraint or policy, it **SHOULD** respond with 406 Not Acceptable to any requests which require reliable return of an Encapsulated Response. The Relay Resource will forward this to the Client, and the Client can then fail or retry with unreliable delivery according to its own requirements.

5. Security Considerations

Unreliable OHTTP does not change the security profile of OHTTP since an Oblivious Relay Resource and Oblivious Gateway Resource could

always reply with non-2xx and no body to clients. Nevertheless, unreliable OHTTP is only appropriate for applications that do not require explicit confirmation of response or otherwise require privacy amplification by the Oblivious Relay Resource.

6. IANA Considerations

Please update the "Media Types" registry at https://www.iana.org/assignments/media-types for the media type and "message/ohttp-ack" (Section 6.1).

6.1. message/ohttp-ack Media Type

The "message/ohttp-ack" media type identifies a key configuration used by Oblivious HTTP.

Type name: message
Subtype name: ohttp-ack
Required parameters: N/A
Optional parameters: None
Encoding considerations: only "8bit" or "binary" is permitted
Security considerations: see Section 5
Interoperability considerations: N/A
Published specification: this specification
Applications that use this media type: Unreliable Oblivious HTTP and applications that use Unreliable Oblivious HTTP
Fragment identifier considerations: N/A
Additional information:
- **Magic number(s):** N/A
- **Deprecated alias names for this type:** N/A
- **File extension(s):** N/A
- **Macintosh file type code(s):** N/A

Person and email address to contact for further information: see Authors' Addresses section
Intended usage: COMMON
Restrictions on usage: N/A
Author: see Authors' Addresses section
Change controller: IESG

7. References

7.1. Normative References

[OHTTP] Thomson, M. and C. A. Wood, "Oblivious HTTP", Work in Progress, Internet-Draft, draft-ietf-ohai-ohttp-03, 8 August 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-ohai-ohttp-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/rfc/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/rfc/rfc8174>.

7.2. Informative References

[LOCALDP] Erlingsson, Ú., Feldman, V., Mironov, I., Raghunathan, A., Talwar, K., and A. Thakurta, "Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity", arXiv article, DOI 10.48550/ARXIV.1811.12469, 2018, <https://doi.org/10.48550/ARXIV.1811.12469>.

[STAR] Davidson, A., Sahib, S. K., and P. Snyder, "STAR: Distributed Secret Sharing for Private Threshold Aggregation Reporting", Work in Progress, Internet-Draft, draft-dss-star-01, 11 July 2022, <https://datatracker.ietf.org/doc/html/draft-dss-star-01>.

Acknowledgments

This draft was motivated by discussions with Martin Thomson, Tommy Pauly, and Lucas Pardue.

Authors' Addresses

Ralph Giles
Brave

Email: giles@thaumas.net

Christopher A. Wood
Cloudflare

Email: caw@heapingbits.net