Workgroup: HTTPBIS
Internet-Draft:
draft-bt-httpbis-reverse-http-01
Published: 23 October 2023
Intended Status: Standards Track
Expires: 25 April 2024
Authors: B. M. Schwartz
Meta Platforms, Inc.

T. Reddy
Nokia

M. Boucadair
Orange

P. S. Tiesel
SAP SE

Reverse HTTP Transport

Abstract

This document defines a secure transport for HTTP in which the client and server roles are reversed. This arrangement improves the origin server's protection from Layer 3 and Layer 4 denial-of-service attacks when an intermediary is in use. It allows origin server's to be hosted without being publicly (and directly) accessible but allows clients to access these servers via an intermediary.

About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-bt-httpbis-reverse-http/.

Discussion of this document takes place on the HTTPBIS Working Group mailing list (mailto:ietf-http-wg@w3.org), which is archived at http://lists.w3.org/Archives/Public/ietf-http-wg/.

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 25 April 2024.

Copyright Notice

Table of Contents

1. Introduction

The Hypertext Transfer Protocol (HTTP) has long supported the ability of clients to access origins via an intermediary. There are a variety of well-defined intermediary types:

- Client-selected
  - HTTP request proxies (a.k.a. forward proxies)
  - Transport proxies (e.g., CONNECT (Section 9.3.6 of [RFC9110]), CONNECT-UDP [RFC9298])
  - IP relays (e.g., CONNECT-IP [I-D.ietf-masque-connect-ip])
  - Oblivious HTTP Relays [I-D.ietf-ohai-ohttp]
- Origin-selected
  - HTTP gateways (a.k.a. reverse proxies)
  - Transport load balancers (e.g., Performance-Enhancing Proxies (PEPs, Section 2.1.1 of [RFC3135])

Although these intermediaries differ widely in their functionality, many of them act as an HTTP client when connecting to the origin. Client-selected intermediaries reach the origin based on its hostname or IP address specified in the HTTP request, while origin-selected intermediaries first translate this destination address into a "backend address".

One of the main advantages of origin-selected intermediaries is their ability to defend the origin from attacks, especially Denial of Service (DoS) and Distributed Denial-of-Service (DDoS) attacks in which an attacker floods the origin server with requests/packets. To prevent attackers from simply bypassing the intermediary, common practices include keeping the backend address hidden and/or instituting filtering rules (ACL, typically) that only allow packets from the intermediary. These practices are reasonably effective with origin-selected intermediaries, but they cannot be used with client-selected intermediaries, as those intermediaries do not know the hidden backend IP address and/or port number, and the origin does not know their "exit" IP addresses.

This specification defines a protocol for HTTP connections between origins and arbitrary intermediaries that can limit the impact of Layer 3 and Layer 4 DoS/DDoS attacks. When this protocol is in use, origins have the ability to partition the infrastructure that serves each intermediary. This ensures that attacks targeting the origin's public IP address or attacks via one intermediary will not affect

any other intermediaries. By partitioning the infrastructure, the impact of the attacks is contained within the affected intermediary or the origin's public IP address.

This protocol works by reversing the roles of the Transport Layer Security (TLS) or QUIC transport that supports an HTTP connection: the origin acts as the transport client, and the intermediary acts as the server. HTTP operates normally inside the secure transport but with the client and server roles reversed.

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. Reverse HTTP: Protocol Overview

The protocol defined in this document is termed "Reverse HTTP". The "Reverse HTTP/2" version is identified by the ALPN ID "h2-reverse", and "Reverse HTTP/3" by "h3-reverse" (see Section 8.1). These protocols represent HTTP/2 and HTTP/3, operating with the roles reversed but otherwise unmodified, except as follows:

* The intermediary **MUST** send a TLS CertificateRequest message (Section 4.4.2 of [RFC8446]) to indicate that certificate-based client authentication is required.

* The origin **MUST** respond with a valid certificate chain.

* Each party **MUST** send a SETTINGS frame (Section 6.5 of [RFC9113]) as soon as the transport becomes writable to them. This means that the intermediary will typically send its SETTINGS frame first.

* The origin **MUST** immediately send an ORIGIN frame identifying the origins it claims. This frame is used as originally defined, except that wildcard names are permitted (contrary to Section 2.2 of [RFC8336]).

  - Otherwise, use of Reverse HTTP with wildcard certificates would be impossible.

* The intermediary **MUST** check the ORIGIN frame contents against the provided TLS client certificate.

Reverse HTTP/1.1 is not defined, as the lack of multiplexing renders it unsuitable for this use.

Once this process has completed, the origin has proved ownership of the Origin Set and is ready to receive requests. The intermediary **SHOULD** direct subsequent HTTP requests for this origin over this Reverse HTTP channel absent explicit policy (e.g., overload limit or ACL). For example, a policy can be provided to the intermediary to help determining unacceptable load threshold.

3.1. Use with Transport Proxies

Transport proxies do not normally act as HTTP clients, so they cannot use Reverse HTTP directly. Instead, if a transport proxy receives a request whose destination host and port number appears in the Origin Set, the proxy establishes a transport proxy connection to this origin over the Reverse HTTP connection. For example, this corresponds to an HTTP CONNECT request for TCP, and for UDP it corresponds to an extended-CONNECT request with the "connect-udp" protocol, using the registered .well-known URI template.

Note that transport destinations identified by an IP address can only use this mechanism if the origin's certificate includes that IP address explicitly.

For IP relays, the destination does not include a port number. By default, the intermediary **MUST** add a port number of 443 before attempting to forward packets using this procedure.

3.2. Discovery of Client-selected Intermediaries

The HTTP "Via" header can indicate the presence of a client-selected intermediary. If a "Via" header arrives with an unrecognized host, the origin **MAY** attempt a Reverse HTTP connection for use by future requests from this intermediary. If the intermediary does not confirm the protocol via ALPN, the origin **MUST** close the connection.

4. Service Binding Mapping

Intermediaries that support Reverse HTTP **SHOULD** indicate this by publishing a SVCB record [I-D.ietf-dnsop-svcb-https] with port-prefix naming, using the scheme "http-reverse" with a default port number of 443. Applicable SvcParamKeys include "alpn", "ipv4hint"/"ipv6hint", and "port". There is no default ALPN value, so the "alpn" key is **REQUIRED**.

```
proxy.example.net.               IN HTTPS 1 . alpn=h2,h3 ech=ABC..123
_http-reverse.proxy.example.net. IN SVCB  1 proxy.example.net. \
      alpn=h2-reverse,h3-reverse ech=ABC..123
```

Figure 1: Example Records for a Forward Proxy that Supports Reverse HTTP

5. Operational Considerations

5.1. Compatibility

Reverse HTTP applies to a single hop of a multihop HTTP connection, especially the hop between an intermediary and the origin. When used in this way, Reverse HTTP is compatible with any ordinary HTTP user agent, and user agents ordinarily cannot tell that Reverse HTTP is in use. (Reverse HTTP does not alter the contents of the "Via" response header.)

5.2. Efficiency

Reverse HTTP does not use appreciably more bandwidth or CPU time than ordinary HTTP on active connections. However, it is much less efficient for idle connections, which use memory and other connection-related resources on the intermediary even when no requests are being processed.

5.3. Connection Management

To accommodate loss of state in firewalls or translators (Section 2 of [RFC6269]), especially in the absence of application traffic, the origin **SHOULD** use appropriate transport-level keepalives and **MUST** re-establish a new connection when an application-level communication has failed.

Origins backed by multiple servers **MAY** attempt to establish a separate Reverse HTTP connection from each one in order to tolerate node failures and support optimized path selection. However, intermediaries **SHOULD** limit the number of idle Reverse HTTP connections operated on behalf of each registrable domain, in order to avoid resource exhaustion attacks. A local configuration may be provided to an intermediary to control the maximum number of active connections.

For very high-traffic origins and multi-instance intermediaries, a disruption could occur if the intermediary immediately directs all user traffic onto the first Reverse HTTP connection. Very large intermediaries **SHOULD** ensure that transitions to Reverse HTTP are gradual, so that large origins have time to establish multiple connections.

> Note: define more what is meant precisly by "Very large", "multi-instance", etc.

5.4. Non-public Origins

In some cases, an HTTP origin may be intended exclusively for use via one or more client-selected intermediaries that are known to the

origin. In this situation, the publication of DNS records outside a domain for the origin is **OPTIONAL**.

5.5. Other Applications

Reverse HTTP is principally intended for use between intermediaries and origins. It is not applicable to general HTTP clients, as it requires that the origin knows that the client will issue a request before the request occurs. However, in cases where an HTTP client is publicly reachable and produces frequent requests to one origin over a long period of time, Reverse HTTP may be applicable. It is out of scope of this document to identify a comprehensive list of the protocol applicability.

6. An Example

The following shows the example of the reverse HTTP in the context of Oblivious HTTP deployments.

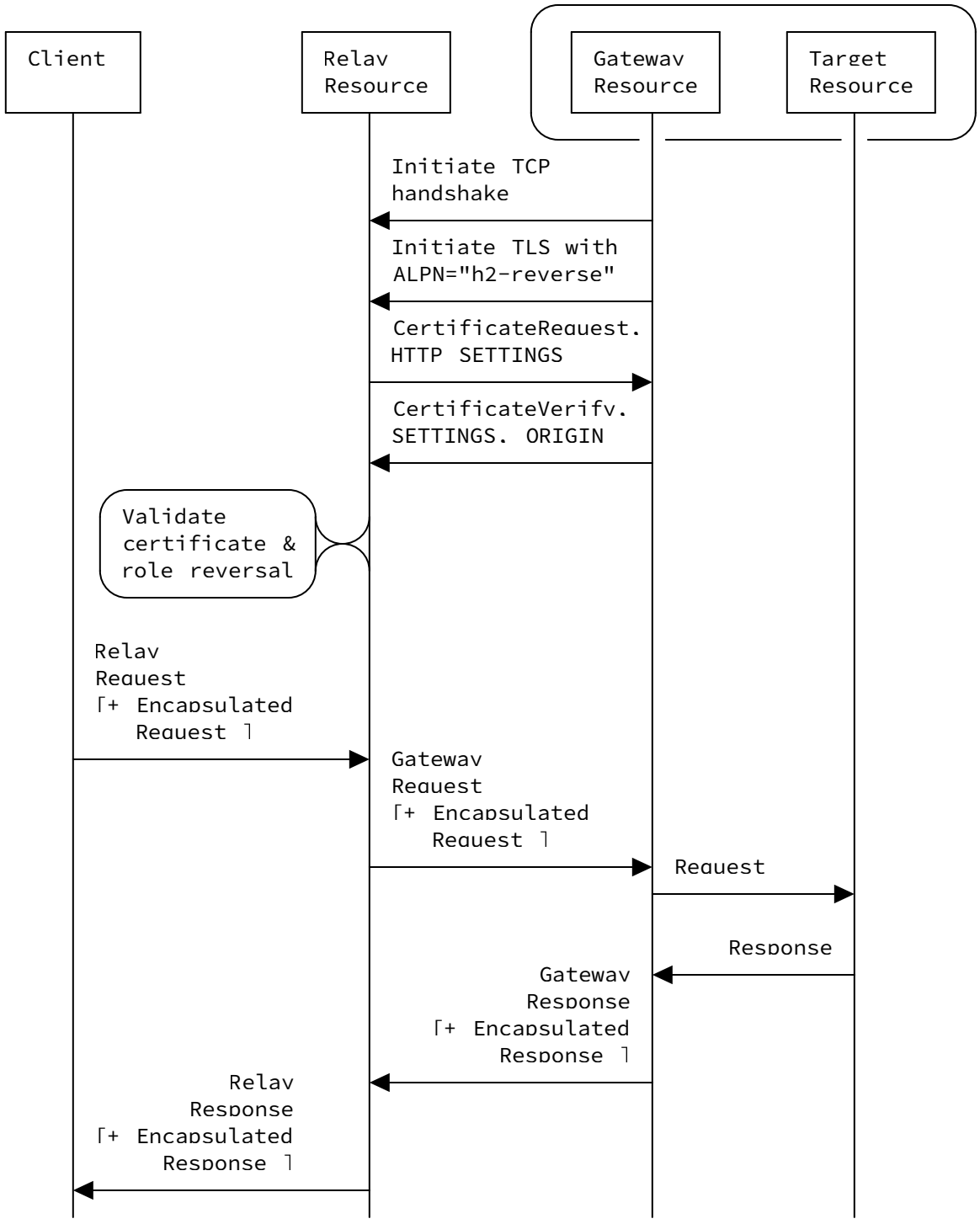

Figure 2: Example: Reverse HTTP/2 for Oblivious HTTP

7. Security Considerations

7.1. Stolen Key Attacks

As noted in Section 4 of [RFC8336], accepting ORIGIN frames without DNS confirmation facilitates the use of stolen keys, and thus increases the incentive to steal these keys. The mitigations listed in that section also apply here, and are **RECOMMENDED**. Intermediaries also **MAY** impose a DNS confirmation requirement, although this weakens the DoS/DDoS defense provided by Reverse HTTP (Section 7.2).

> QUESTION: Should we do more about this? For example, we could define an OID to mark these certificates as Reverse HTTP-only, or we could commit to an IP range by placing a MAC in a DNS record and revealing the message via a SETTINGS value.

7.2. IP Address Leaks

One goal of Reverse HTTP is to prevent DoS/DDoS attackers from learning the IP addresses used by an origin to communicate with this intermediary. These IP addresses can be leaked in various ways, requiring certain mitigations:

* In the ordinary DNS address records for the origin.
  - Origins **SHOULD** use different IP addresses for Reverse HTTP (unless the intermediary imposes a DNS confirmation requirement as described in Section 7.1).
* In the Proxy-Status HTTP header field's "next-hop" attribute.
  - Intermediaries **MUST NOT** populate the "next-hop" attribute when using Reverse HTTP to the origin.
* From the probes sent to intermediaries discovered from the "Via" header field.
  - Origins **SHOULD** use distinct, unrelated IP addresses to contact each intermediary.
* From connection monitoring of the ALPN values in ClientHellos.
  - Intermediaries **MAY** use a single Encrypted ClientHello configuration for HTTP and Reverse HTTP.
* From statistical analysis of traffic patterns.
  - Origins **SHOULD** regularly change the IP address that is used.

Even if the origin's Reverse HTTP IP addresses do leak, Reverse HTTP still provides significant protection by simplifying the filtering rules required to block unsolicited connections.

7.3. Key Consistency with Oblivious HTTP

The security considerations for the Oblivious HTTP (Section 8 of [OHTTP]) as well as the security considerations for Discovery of Oblivious Services via Service Binding Records (Section 6 of [OHAI-SVCB]) apply.

[CONSISTENCY] provides an analysis of the options for ensuring the key configurations are consistent between different clients. Clients **MUST** employ techniques to mitigate key targeting attacks. Note that the option of confirming the key with a shared proxy as described in [CONSISTENCY] will work if the Oblivious HTTP relay and the forward proxy operate from a single origin.

8. IANA Considerations

8.1. ALPN IDs

This document rquests IANA to add two new registrations in the "TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs" registry [RFC7301]:

Protocol	Identification Sequence	Specification
Reverse HTTP/2	0x68 0x32 0x2d 0x72 0x65 0x76 0x65 0x72 0x73 0x65 ("h2-reverse")	(This document)
Reverse HTTP/3	0x68 0x33 0x2d 0x72 0x65 0x76 0x65 0x72 0x73 0x65 ("h3-reverse")	(This document)

Table 1

8.2. New Scheme

Per [Attrleaf], IANA is requested to add the following entry to the DNS Underscore Global Scoped Entry registry:

RR TYPE	_NODE NAME	Reference
SVCB	_http-reverse	(This document, Section 4)

Table 2

TODO: Register the URI scheme as well.

9. References

9.1. Normative References

[I-D.ietf-dnsop-svcb-https] Schwartz, B. M., Bishop, M., and E. Nygren, "Service binding and parameter specification via

the DNS (DNS SVCB and HTTPS RRs)", Work in Progress, Internet-Draft, draft-ietf-dnsop-svcb-https-12, 11 March 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-dnsop-svcb-https-12>.

[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>.

[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/rfc/rfc7301>.

[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>.

[RFC8336] Nottingham, M. and E. Nygren, "The ORIGIN HTTP/2 Frame", RFC 8336, DOI 10.17487/RFC8336, March 2018, <https://www.rfc-editor.org/rfc/rfc8336>.

[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>.

[RFC9113] Thomson, M., Ed. and C. Benfield, Ed., "HTTP/2", RFC 9113, DOI 10.17487/RFC9113, June 2022, <https://www.rfc-editor.org/rfc/rfc9113>.

9.2. Informative References

[Attrleaf] Crocker, D., "Scoped Interpretation of DNS Resource Records through "Underscored" Naming of Attribute Leaves", BCP 222, RFC 8552, DOI 10.17487/RFC8552, March 2019, <https://www.rfc-editor.org/rfc/rfc8552>.

[CONSISTENCY] Davidson, A., Finkel, M., Thomson, M., and C. A. Wood, "Key Consistency and Discovery", Work in Progress, Internet-Draft, draft-ietf-privacypass-key-consistency-01, 10 July 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-privacypass-key-consistency-01>.

[I-D.ietf-masque-connect-ip] Pauly, T., Schinazi, D., Chernyakhovsky, A., Kühlewind, M., and M. Westerlund, "Proxying IP in HTTP", Work in Progress, Internet-Draft, draft-ietf-masque-connect-

ip-13, 28 April 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-masque-connect-ip-13>.

[I-D.ietf-ohai-ohttp] Thomson, M. and C. A. Wood, "Oblivious HTTP", Work in Progress, Internet-Draft, draft-ietf-ohai-ohttp-10, 25 August 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-ohai-ohttp-10>.

[OHAI-SVCB] Pauly, T. and T. Reddy.K, "Discovery of Oblivious Services via Service Binding Records", Work in Progress, Internet-Draft, draft-ietf-ohai-svcb-config-07, 6 October 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-ohai-svcb-config-07>.

[OHTTP] Thomson, M. and C. A. Wood, "Oblivious HTTP", Work in Progress, Internet-Draft, draft-ietf-ohai-ohttp-10, 25 August 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-ohai-ohttp-10>.

[RFC3135] Border, J., Kojo, M., Griner, J., Montenegro, G., and Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations", RFC 3135, DOI 10.17487/RFC3135, June 2001, <https://www.rfc-editor.org/rfc/rfc3135>.

[RFC6269] Ford, M., Ed., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", RFC 6269, DOI 10.17487/RFC6269, June 2011, <https://www.rfc-editor.org/rfc/rfc6269>.

[RFC9110] 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>.

[RFC9298] Schinazi, D., "Proxying UDP in HTTP", RFC 9298, DOI 10.17487/RFC9298, August 2022, <https://www.rfc-editor.org/rfc/rfc9298>.

Authors' Addresses

Benjamin M. Schwartz
Meta Platforms, Inc.

Email: ietf@bemasc.net

Tirumaleswar Reddy
Nokia

Email: kondtir@gmail.com

Mohamed Boucadair
Orange

Email: mohamed.boucadair@orange.com

Philipp S. Tiesel
SAP SE

Email: philipp@tiesel.net