Workgroup: HTTP Working Group
Internet-Draft:
draft-bishop-httpbis-distributed-origin-00
Published: 4 October 2021
Intended Status: Informational
Expires: 7 April 2022
Authors: M. Bishop
Akamai Technologies

Distributed HTTP Origins: Solution Space Exploration

Abstract

Certain content libraries are logically a single origin, but too large to be practically served by a single origin server. This document discusses existing solutions and explores possible directions for future protocol development.

Discussion Venues

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

Discussion of this document takes place on the mailing list (httpbis@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/httpbis/.

Source for this draft and an issue tracker can be found at https://github.com/MikeBishop/alt-svc-bis.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

With increasingly large content deployments, certain origins become too large to contain all the data which is logically connected on the same server. A similar issue exists on CDNs, where an origin being served through a reverse-proxy contains too many large resources for a single instance to cache effectively.

Examples of this abound in the real world -- consider the video libraries of Netflix or YouTube, the photo library of Facebook, or the software library of any large software publisher which must make available multiple full and patch versions of multiple editions of multiple software products.

While there are existing ways to address this problem, they are suboptimal in various ways. This document discusses existing approaches (Section 2), previous standards efforts which may provide solutions (Section 3), and possible directions for future development (Section 4).

1.1. 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.

2. Existing Solutions

In the real world, the origin users initially visit in a browser is typically one that a human can remember and type. This user-facing origin serves HTML that references content, which may be on other origins. A similar approach exists in non-browser cases, where a user-locatable front-end indicates the actual location of the desired content.

2.1. Content-Specific Hostnames

One solution, visible in multiple services, uses granular hostnames to identify the server or servers with the particular content in question, such as r2---sn-jpocxaa-j8bl.googlevideo.com. This hostname, with its own HTTP origin, controls a particular slice of the media available on YouTube.com. The YouTube service indicates to a player loading a video which origin contains or caches the requested content.

Note that there are several ways of providing these hostnames to clients, depending on the interaction model between the client and the server. For example:

*The server might generate HTML or JSON content in response to an initial request, providing absolute URIs for each dependent resource which indicate the specific host from which the resource can be retrieved

*The server might return a 3XX (Redirect) response to a client's query for a resource, directing the client to the resource at a different hostname

*An API might enable a client to query for the location of a resource before requesting it

One drawback of this approach is that the content belongs to a different origin than the primary origin of the page. While this is less of an issue in APIs or bulk data transfer, this limits the types of requests that can be made and the access to the data from scripts loaded by the primary origin without first making CORS preflight requests [CORS], which introduce additional latency.

This approach can also complicate certain protocol features which rely on previous contact with the server. The primary server typically cannot provide Alt-Svc entries for the secondary, though the targeting of the specific hostname may avoid the need for Alt-Svc. TLS session resumption and 0-RTT will typically not be usable, adding latency to the request.

2.2. Internal Load-Balancing

A second solution, which is generally not visible to the client, is to have all requests terminated by a front-end which does not cache or serve any content directly. Rather, this front-end is responsible for inspecting the request, identifying the server which can actually respond, and forwarding the request to that server.

This solution has its own challenges. While the data access and storage requirements can be distributed amongst back-end machines, throughput on the front-end load balancer becomes a bottleneck. For certain protocols, direct server return (DSR) avoids this bottleneck by sending response packets back to the client instead of sending them via the load balancer. However, DSR is challenging with reliable and encrypted protocols, and even moreso with multiplexed protocols like HTTP/2 or HTTP/3.

3. Previous Standards Efforts

Several previous drafts in the IETF have offered partial solutions for this problem, but have not been published as RFCs or achieved widespread adoption.

3.1. Out-of-Band Encoding

[OOB] describes an HTTP content coding that can be used to describe the location of a secondary resource that contains the payload. The origin returns an HTTP field set which describes the content, including a Content-Encoding header which indicates the content can be fetched from a different URL, typically hosted on a different origin server.

This approach is similar in spirit to Content-Specific Hostnames as described in Section 2.1, except that the resources continue to belong to a single origin regardless of which origin server actually delivers the bytes. Unlike Content-Specific Hostnames, however, a separate request must be made for each resource -- first to the origin server to receive the headers, then to the secondary server to retrieve the content of the response.

3.1.1. Resource Map

[SCD] references a possible extension to this idea, where the origin server would indicate to a client that a particular set of resources would all be available from a particular secondary server. However, the specifics of this interaction were not identified in that draft.

One drawback to this approach is that an origin might prefer not to distribute the full set of endpoints or resources, either because this information is considered proprietary or because the set itself is large enough to be prohibitive.

3.2. Alternative Services

[AltSvc] describes a way in which an origin server can delegate authority over the origin to another host which might be preferable in some way. However, this mechanism delegates the entire origin and cannot be subdivided.

A 421 response being used to work around this dramatically reduces efficiency, as the client has no insight into which paths the alternative might or might not support.

4. Possible Future Directions

Any new solution should fit within the following constraints:

*No new feature to address this scenario can expect to entirely replace the existing approaches given client upgrade and hardware replacement schedules, so the solution needs to be easily layered on top of current approaches. This likely implies a client-advertised extension.

*Unlike Alt-Svc ([AltSvc]), the solution should permit delegation of portions of the origin's URI space to one or more secondary servers.

*Unlike resource maps (Section 3.1.1), the solution should permit incremental new information about secondary server(s) and delegated ranges of resources.

This section describes one possible solution in this vein, based on HTTP Alternative Services [AltSvc]. The components of this solution might be generally useful and incorporated into various specifications, or might be tightly coupled and belong in a single specification.

Other solutions within these constraints should also be considered.

4.1. Scope-Restricted Alt-Svc Entries

When an alternative service is advertised by an origin, by default the indicated server is authoritative for all resources in the origin. The scope parameter can be used to adjust this scope.

The scope parameter contains the path portion of a URI; see Section 3.3 of [RFC3986]. The indicated alternative is authoritative only for resources where the path begins with the indicated prefix.

```
scope = DQUOTE path DQUOTE ; see [RFC3986], Section 3.3
```

For example:

```
Alt-Svc: h2=":443"; ma=3600; scope="/sn-jpocxaa-j8bl/",
         h2=":443"; ma=3600; scope="/sn-5ualdn7s"
```

A scope-restricted alternative SHOULD NOT be sent requests for resources unless the path portion of the URI is a prefix match with the indicated scope.

[AltSvc] indicates that parameters are optional to understand. Therefore, origin servers SHOULD NOT send an alternative service advertisement to a client which has not indicated support for this extension (Section 4.2). Alternatives MUST be prepared to receive requests for any resource in the origin. However, the alternative MAY respond with a 421 (Misdirected Request) to any request it is unable to serve.

4.2. Indicating Support for Alt-Svc Parameters

Certain origins might prefer to take different actions based on whether the client supports HTTP Alternative Services or not. For example, many clients are unable to implement the persist parameter defined in [AltSvc]. Servers that offer alternatives based on the client's current network connection might choose not to send Alt-Svc entries to such a client.

The client can optionally send an Accept-Alt-Svc request header field indicating which Alt-Svc parameters it is able to understand. The content of this field is an sf-list [RFC8941] of Alt-Svc parameter names. To reduce fingerprinting surface, the contents of the list SHOULD be sorted alphabetically.

For example:

```
Accept-Alt-Svc: host, ma, persist, scope
```

A server MAY publish alternative services containing parameters which are not understood by the client, since unknown parameters are ignored per [AltSvc].

While [AltSvc] enables an alternative to reside on a different host than the origin server, not all clients implement this behavior. This draft registers the "host" parameter for Alt-Svc to enable clients to indicate support for Alt-Svc entries which provide a different hostname from the origin. The "host" parameter MUST NOT be used in Alt-Svc field generation and MUST be ignored if present.

The presence of this header can be assumed to indicate support for Alt-Svc, even if empty.

4.3. Incremental Alt-Svc Advertisements

[AltSvc] says that when an Alt-Svc response header field is received from an origin, its value invalidates and replaces all cached alternative services for that origin.

In certain circumstances, a server might prefer not to publish the full list of alternatives, but instead incrementally add to them. For example, a server might provide scope-restricted alternatives as a client makes requests for resources in various scopes.

This draft defines the Additional-Alt-Svc header field. The parsing and semantics of this field are identical to that of Alt-Svc, with the following modifications:

*The value MUST NOT be "clear"

*The entries presented augment, rather than replace, any cached alternatives already known to the client.

4.4. The 3NN (Use Alternative) Status Code

This document defines a new status code directing that a client attempt to satisfy the request from an alternative.

A server MUST include an Alt-Svc or Additional-Alt-Svc header field in the response indicating which alternative(s) the client can use to satisfy the given request. A server MUST NOT send the 3NN status code in response to a request which did not contain the Accept-Alt-Svc header field.

Upon receipt of this status code, a client SHOULD choose an alternative service and retry the request with that alternative. If all configured alternatives are unsuccessful, or the client chooses not to use an alternative, the client MAY retry the request with the origin server, omitting the Accept-Alt-Svc header field.

5. Security Considerations

TODO Security

6. IANA Considerations

Lots of stuff to register later.

7. References

7.1. Normative References

[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>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/rfc/rfc3986>.

[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>.

[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/rfc/rfc8941>.

7.2. Informative References

[AltSvc] Nottingham, M., McManus, P., and J. Reschke, "HTTP Alternative Services", RFC 7838, DOI 10.17487/RFC7838, April 2016, <https://www.rfc-editor.org/rfc/rfc7838>.

[CORS] "Cross-Origin Resource Sharing (CORS)", n.d., <https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS>.

[OOB] Reschke, J. F. and S. Loreto, "'Out-Of-Band' Content Coding for HTTP", Work in Progress, Internet-Draft, draft-reschke-http-oob-encoding-12, 24 June 2017, <https://datatracker.ietf.org/doc/html/draft-reschke-http-oob-encoding-12>.

[SCD]
Thomson, M., Eriksson, G. A., and C. Holmberg, "An Architecture for Secure Content Delegation using HTTP", Work in Progress, Internet-Draft, draft-thomson-http-scd-02, 30 October 2016, <https://datatracker.ietf.org/doc/html/draft-thomson-http-scd-02>.

Acknowledgments

TODO acknowledge.

Author's Address

Mike Bishop
Akamai Technologies

Email: mbishop@evequefou.be