Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 01, 2012

E. Nygren
S. Ludin
M. Levine
A. Champagne
Akamai

June 2012

Method for suggestion Alternative Servers to Enhance Connection Level Performance
draft-nygren-httpbis-connection-redirect-00

Abstract

This memo presents a method for allowing an HTTP server to provide hints in its HTTP responses to a user-agent that indicate the user could be better off connecting to a different IP than what they are currently connected to. This is highly relevant in a multi-homed server world where DNS may not have all of the relevant state necessary to make a proper routing decision. Additionally it is done in a manner that does not require application level changes as could be the case for an application level redirect such as a 3xx response

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 http://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 December 01, 2012.

Table of Contents

1. Overview

Large deployments of HTTP servers commonly use DNS for Mapping clients to servers, both for performance and load-balancing. The current HTTP protocol provides no mechanism for a server to move clients to a better alternative server without URI changes that can interfere with application semantics. The primary goal of this initiative is to develop a proposed standard set of extensions to the HTTP/1.1 protocol that will enable HTTP servers to influence the behavior of HTTP clients at the connection-level, rather than at the application-level. This must be done in a way that enables an incremental roll-out (both to servers and clients), which doesn't add undue complexity, and which does not introduce or exacerbate security vulnerabilities.

At the core of the initial proposal is a set of additional HTTP response headers which servers can return to clients to influence connection-level client behavior for subsequent HTTP requests. These response headers are intended to provide changes to client behavior by allowing servers to take the actual network location of clients into account and give a per-client answer.

A goal here is to minimize the complexity of client implementations and to provide service providers the leeway to evolve actual heuristics used to assign clients to servers.

The closest existing alternative is for a server to issue a 302 (or other 3xx) redirect to clients. The major down-side of this 302-redirect mechanism is that to provide an alternative server, it requires altering the host component of URIs and, thus, the http redirect approach is not transparent to applications. Due to user agent same-origin and cross-domain security policies surrounding cookies, and some scripting languages (ECMAScript, Silverlight, Java, Flash, and potentially others), this can require significant application changes as part of implementation. Experience has shown that there is a high barrier to adoption for technologies that require significant application-level changes, as many companies have organizational separation between web application development and the teams responsible for application performance and deployment. As such, there is a strong desire for an alternative to 3xx redirects that minimizes (or eliminates) the need for changes to applications or URIs, but which also does not introduce any additional security vulnerabilities.

2. Illustrative Use Case

In the following use-case, a user agent (UA) requests a URI under https://www.example.com/ and is returned a hint that subsequent requests for www.example.com should be sent to an alternative server (sfo-server.example.com), which (based on considerations outside of this proposal) has better performance:

request: UA --> server (www.example.com)

```
GET /application.asp HTTP/1.1
Host: www.example.com
```

```
response: server (www.example.com) --> UA

   HTTP/1.1 200 OK
   Alt-Server: sfo-server.example.com ; Max-Age=900
   Alt-Server-Policy: Path=/
   (...body...)

request: UA --> alternative server (sfo-server.example.com)

   GET /ajax-handler.asp HTTP/1.1
   Host: www.example.com
   Alt-Server-Name: sfo-server.example.com

response: alternative server (sfo-server.example.com) --> UA

   HTTP/1.1 200 OK
   Alt-Server: sfo-server.example.com ; Max-Age=900
   Alt-Server-Policy: Path=/
   Alt-Server-Allow-Host: www.example.com
   (...body...)
```


Note that in this example, URI of the request is not changed by the response and clients/servers not supporting this new header would continue to behave without changes. In this example, subsequent requests under https://www.example.com/ will also continue to be sent to the alternative server unless another is provided, the Max-Age is reached without the Alt-Server being refreshed, or the client experiences a failure in retrieving content from the Alt-Server.

3. Goals and use-cases

The desire is to handle as many of the following use-cases as possible with this proposal. Earlier use-cases are considered to be more important to satisfy if trade-offs need to be made. In all of these cases, the common theme is to provide an application-transparent manner to enable clients to be directed to alternative servers without modifying the absoluteURI (http_URL) or the corresponding user agent behavior.

1. A goal is to enable HTTP servers to propose to clients semantically- equivalent alternative servers to provide improved performance in certain scenarios. This is desirable both for

applications requiring high throughput (such as Video content that can face challenges with the bandwidth*delay product), as well as for latency-sensitive interactive applications.

2. Provide a mechanism to provide better performance and locality of reference for large libraries of content. A problem is that popular content may need to be replicated to spread out load, but replicating all the unpopular content is not feasible. Enable clients to be directed to alternative servers that may provide better performance for particular content.

3. Provide an additional mechanism within HTTP to address potential performance and scalability issues during the roll-outs of IPv6, TLS SNI, and other Internet protocols with compatibility challenges. In particular, provide a mechanism where clients can be initially directed by DNS to a smaller set of servers guaranteed to be reachable and fully compatible, but then provide those a mechanism to direct newer clients to alternative servers that may provide better performance at the expense of full compatibility with older clients.

4. Terminology

The terms user agent, client, server, proxy, tunnel, origin server, connection, request, and http_URL have the same meaning as in the HTTP/1.1 specification [RFC2616]. The term absoluteURI has the same meaning as in the URI Syntax specification [RFC2396].

The term alt-server or "alternative server" refers to a domain name

(and optional port) or numeric IP address (and optional port) to which a client should initiate connections for subsequent HTTP requests.

The terms request-hostport and request-path refer to the hostport and path portions of the absoluteURI (http_URL) of the HTTP request.

5. Proposed HTTP protocol extension

It is proposed to provide appropriate user agent mechanisms that can interpret and respond appropriately to optional Alt-Server and Alt-Sever-Policy HTTP response headers, as well as an Alt-Server-Host

HTTP request header.

The actual details of these headers (and the Alt-Server-Policy header in-particular) are included as an initial starting point for discussions, rather than as a final proposal. One potential grammar for such headers might look like:

```
Alt-Server            = "Alt-Server:" alt-servers
alt-servers           = 1#alt-server-host
alt-server-host       = hostport *(";" alt-server-av)
alt-server-av         = "Max-Age" "=" delta-seconds

Alt-Server-Policy     = "Alt-Server-Policy:" alt-server-policies
alt-server-policies   = 1#alt-server-policy-av
alt-server-policy-av  = "Path" "=" path-absolute ; [RFC3986]
                      | "When" "=" policy-when-value
policy-when-value     = "immediate"
                      | "next"
                      | "sticky"


Alt-Server-Name       = "Alt-Server-Name:" hostport *(";" alt-server-host-av
Alt-Server-Allow-Host = "Alt-Server-Allow-Host:" hostport | "*"
hostport              = host [ ":" port ]        ; [RFC3986]
delta-seconds         = 1*DIGIT                  ; [RFC2616]
```

Both of the Alt-Server and Alt-Server-Policy headers are optional. If the Alt-Server-Policy response header is provided, an Alt-Server header MUST also be provided.

On subsequent requests to an alt-server-host, the hostport of the alternative server MUST be specified in the Alt-Server-Name request header.

On subsequent responses, the Alt-Server-Allow-Host needs to match the Host header or the client MUST drop the response.

6. When requests may be sent to an Alt-Server

Subject to the policy rules described subsequently, clients SHOULD

attempt to issue subsequent matching requests to an alt-server-host while preserving the absoluteURI of the request. Clients MUST NOT send non-matching requests to an alt-server-host.

A subsequent request is matching to a <request-hostport, Path> pair if and only if all of these conditions are met:

1. The request-hostport of the subsequent request is equal to the request-hostport of the request corresponding to the response for which the alt-server-host was set.

2. The Path specified in the Alt-Server-Policy is a prefix of the request-path for the subsequent request (including the case where they string-compare equal).

3. Max-Age seconds have not passed since a corresponding alt-server-host and Path were last returned in a response.

If multiple <request-hostport, Path> pairs match, the one with the longest Path should be applied.

Depending on the policy-when-value the behavior should change:

> "immediate" - the client SHOULD abort this HTTP request and re-issue the request to an alt-server. This MUST NOT be returned in response to non-idempotent requests (such as POST/PUT).
>
> "next" = the client SHOULD continue with any in-flight HTTP requests but should in-parallel establish a connection to the alt-server for use in subsequent requests.

Clients should flush the Alt-Server data when it IP address changes, for example when establishing a VPN connection or switching between wi-fi and cellular connection on a mobile device

7. Mechanisms for clients to balance performance

Independent of the protocol elements for enabling redirects to alternative servers while preserving semantic transparency and maintaining security, the majority of the remaining protocol details surround the area of how clients should behave to give servers flexibility in moving client traffic around to meet various performance objectives. The exact details here are an area for further discussion.

Some considerations include:

> It would be good for clients to minimize the number of times they switch between servers (to minimize the performance impact of setting up new connections).

The desired "stickiness" of clients to servers may vary between systems and applications, and as such the protocol should give servers some degree of control over how and when clients move between servers or go back to the original server specified in the request-hostport.

For sites that consist of many performance-sensitive small requests (such as AJAX-based websites, map tiles, or chunked HTTP streaming), a good behavior is often to err on the side of clients being able to receive the first few objects from a sub-optimal server while in-parallel setting up connections to an alternative server for subsequent requests.

For large objects (such as downloads or progressive media), it may be advantageous for the client to immediately disconnect after receiving response headers and to issue a new request to an alternative server.

Clients should be able to obtain a reaffirmation from an alternative server that it is a good choice without needing to return back to the initial host.

In error cases (such as when an alternative server has failed), client behavior should involve retrying other alternative servers and then eventually falling back to the initial host.

Some approaches to achieve this include allowing alt-servers to be returned with a Max-Age (which is how long a client should remember the answer if it hasn't been refreshed) as well as by having a set of different alt-server-policies which may be specified (for example, for suggesting that the client immediately disconnect, wait until subsequent requests to connect, etc.) As the policy behaviors here are performance hints to clients, it may also be possible to design this in such a way that future policies can be added without requiring full adoption by all clients.

A guiding design tension here will be finding a small set of features to implement within clients that enables as many use-cases as possible, enables future performance improvements from server-side-only changes, but also minimizes client and protocol complexity.

8. Security Considerations

To retain semantic transparency, user agents must continue to follow existing origin and domain-oriented security policies (for cookies, scripting domains, etc.) as per the request-hostport of the request, regardless of the current alt-server. The Origin (as applies to same-origin policies) should continue to be derived from the request-hostport of the request. This has potential security ramifications that will need to be evaluated at in significant detail.

However, we believe that this protocol extension will not introduce significant new vulnerabilities when used within the HTTPS/TLS context. In this case, user agents must be validating that the Common Name (CN) or subjectAltName of the certificate continues to match the request-host (and not the alt-server). In this manner, the server is cryptographically authenticating that it continues to be an authoritative server for the request-host. (Due to constraints with IPv4 availability and the context in which this protocol extension will be used, it may be prudent to also encourage user agents implementing it to also implement the TLS SNI extension.)

It is likely that the Alt-Server-Allow-Host constraints will protect agaist many types of attacks, but further restrictions may need to be imposed when using this protocol extension outside of the HTTPS/TLS context and will need to be explored in more detail. In particular, there may be ways in which this extension could worsen existing XSS, CSRF, and similar vulnerabilities.

Additional security could be provided by signing the response in some way. This should be explored as a method to allow more safe flexibility in the Alt-Server names.

9. IANA Considerations

The new headers:

Alt-Server

Alt-Server-Policy

Alt-Server-Allow-Host

Alt-Server-Name

Require IANA assignment.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informational References

[Cors] "Cross-Origin Resource Sharing", , <http://www.w3.org/TR/cors/>.

[HappyEyes] "Happy Eyeballs", , <http://tools.ietf.org/html/draft-ietf-v6ops-happy-eyeballs-07>.

Appendix A. Other Considerations

Should we clients be encouraged to do races if multiple Alt-Servers are returned? The Happy Eyeballs draft may be relevant (http://tools.ietf.org/html/draft-ietf-v6ops-happy-eyeballs-07). Some of the changes to Chrome/FireFox to support Happy Eyeballs may also be of help in implementing this extension if we want to prototype it.

Authors' Addresses

Erik Nygren
Akamai Technologies

Stephen Ludin
Akamai Technologies

Matt Levine
Akamai Technologies

Andy Champagne
Akamai Technologies