Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2014

M. Nottingham
Akamai
P. McManus
Mozilla

March 21, 2014

HTTP Alternative Services
draft-nottingham-httpbis-alt-svc-04

Abstract

This document specifies "alternative services" for HTTP, which allow an origin's resources to be authoritatively available at a separate network location, possibly accessed with a different protocol configuration.

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 September 22, 2014.

Table of Contents

1. Introduction

HTTP [I-D.ietf-httpbis-p1-messaging] conflates the identification of resources with their location. In other words, http:// (and https://) URLs are used to both name and find things to interact with.

In some cases, it is desirable to separate these aspects; to be able to keep the same identifier for a resource, but interact with it using a different location on the network.

For example:

- An origin server might wish to redirect a client to an alternative when it needs to go down for maintenance, or it has found an alternative in a location that is more local to the client.
- An origin server might wish to offer access to its resources using a new protocol (such as HTTP/2 [I-D.ietf-httpbis-http2]) or one using improved security (such as TLS {{RFC5246}).
- An origin server might wish to segment its clients into groups of capabilities, such as those supporting SNI (see [RFC6066]) and those not supporting it, for operational purposes.

This specification defines a new concept in HTTP, "Alternative Services", that allows a resource to nominate additional means of interacting with it on the network. It defines a general framework for this in Section 2, along with a specific mechanism for discovering them using HTTP headers in Section 3.

It also introduces a new status code in Section 4, so that origin servers (or their nominated alternatives) can indicate that they are not authoritative for a given origin, in cases where the wrong location is used.

1.1. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This document uses the Augmented BNF defined in [RFC5234] along with the "OWS", "DIGIT", "parameter", "port" and "delta-second" rules from [I-D.ietf-httpbis-p1-messaging], and uses the "#rule" extension defined in Section 7 of that document.

2. Alternative Services Concepts

This specification defines a new concept in HTTP, the "alternative service." When an origin (see [RFC6454]) has resources are accessible through a different protocol / host / port combination, it is said to have an alternative service.

An alternative service can be used to interact with the resources on an origin server at a separate location on the network, possibly using a different protocol configuration. Alternative services are considered authoritative for an origin's resources, in the sense of [I-D.ietf-httpbis-p1-messaging] Section 9.1.

For example, an origin:

("http", "www.example.com", "80")

might declare that its resources are also accessible at the alternative service:

("h2", "new.example.com", "81")

By their nature, alternative services are explicitly at the granularity of an origin; i.e., they cannot be selectively applied to resources within an origin.

Alternative services do not replace or change the origin for any given resource; in general, they are not visible to the software "above" the access mechanism. The alternative service is essentially alternative routing information that can also be used to reach the origin in the same way that DNS CNAME or SRV records define routing information at the name resolution level. Each origin maps to a set of these routes - the default route is derived from origin itself and the other routes are introduced based on alternative-protocol information.

Furthermore, it is important to note that the first member of an alternative service tuple is different from the "scheme" component of an origin; it is more specific, identifying not only the major version of the protocol being used, but potentially communication options for that protocol.

This means that clients using an alternative service will change the host, port and protocol that they are using to fetch resources, but these changes MUST NOT be propagated to the application that is using HTTP; from that standpoint, the URI being accessed and all information derived from it (scheme, host, port) are the same as before.

Importantly, this includes its security context; in particular, when TLS [RFC5246] is in use, the alternative server will need to present a certificate for the origin's host name, not that of the alternative. Likewise, the Host header is still derived from the origin, not the alternative service (just as it would if a CNAME were being used).

The changes MAY, however, be made visible in debugging tools, consoles, etc.

Formally, an alternative service is identified by the combination of:

- An ALPN protocol, as per [I-D.ietf-tls-applayerprotoneg]
- A host, as per [RFC3986]
- A port, as per [RFC3986]

Additionally, each alternative service MUST have:

- A freshness lifetime, expressed in seconds; see Section 2.2

There are many ways that a client could discover the alternative service(s) associated with an origin.

2.1. Host Authentication

Clients MUST NOT use alternative services with a host other than the origin's without strong server authentication; this mitigates the attack described in Section 6.2. One way to achieve this is for the alternative to use TLS with a certificate that is valid for that origin.

For example, if the origin's host is "www.example.com" and an alternative is offered on "other.example.com" with the "h2" protocol, and the certificate offered is valid for "www.example.com", the client can use the alternative. However, if "other.example.com" is offered with the "h2c" protocol, the client cannot use it, because there is no mechanism in that protocol to establish strong server authentication.

Furthermore, this means that the HTTP Host header and the SNI information provided in TLS by the client will be that of the origin, not the alternative.

2.2. Alternative Service Caching

Mechanisms for discovering alternative services can associate a freshness lifetime with them; for example, the Alt-Svc header field uses the "ma" parameter.

Clients MAY choose to use an alternative service instead of the origin at any time when it is considered fresh; see Section 2.4 for specific recommendations.

Clients with existing connections to alternative services are not required to fall back to the origin when its freshness lifetime ends; i.e., the caching mechanism is intended for limiting how long an alternative service can be used for establishing new requests, not limiting the use of existing ones.

To mitigate risks associated with caching compromised values (see Section 6.2 for details), user agents SHOULD examine cached alternative services when they detect a change in network configuration, and remove any that could be compromised (for example, those whose association with the trust root is questionable). UAs that do not have a means of detecting network changes SHOULD place an upper bound on their lifetime.

2.3. Requiring Server Name Indication

A client must only use a TLS-based alternative service if the client also supports TLS Server Name Indication (SNI) [RFC6066]. This supports the conservation of IP addresses on the alternative service host.

2.4. Using Alternative Services

By their nature, alternative services are optional; clients are not required to use them. However, it is advantageous for clients to behave in a predictable way when they are used by servers (e.g., for load balancing).

Therefore, if a client becomes aware of an alternative service, the client SHOULD use that alternative service for all requests to the associated origin as soon as it is available, provided that the security properties of the alternative service protocol are desirable, as compared to the existing connection.

The client is not required to block requests; the origin's connection can be used until the alternative connection is established. However, if the security properties of the existing connection are weak (e.g. cleartext HTTP/1.1) then it might make sense to block until the new connection is fully available in order to avoid information leakage.

Furthermore, if the connection to the alternative service fails or is unresponsive, the client MAY fall back to using the origin. Note, however, that this could be the basis of a downgrade attack, thus

losing any enhanced security properties of the alternative service.

3. The Alt-Svc HTTP Header Field

A HTTP(S) origin server can advertise the availability of alternative services (see Section 2) to clients by adding an Alt-Svc header field to responses.

```
Alt-Svc     = 1#( alternative *( OWS ";" OWS parameter ) )
alternative   = <"> protocol-id <"> "=" port
protocol-id = <ALPN protocol identifier>
```

For example:

```
Alt-Svc: "http2"=8000
```

This indicates that the "http2" protocol on the same host using the indicated port (in this case, 8000).

Alt-Svc MAY occur in any HTTP response message, regardless of the status code.

Alt-Svc does not allow advertisement of alternative services on other hosts, to protect against various header-based attacks.

It can, however, have multiple values:

```
Alt-Svc: "h2c"=8000, "h2"=443
```

The value(s) advertised by Alt-Svc can be used by clients to open a new connection to one or more alternative services immediately, or simultaneously with subsequent requests on the same connection.

Intermediaries MUST NOT change or append Alt-Svc values.

Finally, note that while it may be technically possible to put content other than printable ASCII in a HTTP header, some implementations only support ASCII (or a superset of it) in header field values. Therefore, this field SHOULD NOT be used to convey protocol identifiers that are not printable ASCII, or those that contain quote characters.

3.1. Caching Alt-Svc Header Field Values

When an alternative service is advertised using Alt-Svc, it is considered fresh for 24 hours from generation of the message. This can be modified with the 'ma' (max-age) parameter;

```
Alt-Svc: "h2"=443;ma=3600
```

which indicates the number of seconds since the response was generated the alternative service is considered fresh for.

ma = delta-seconds

See [I-D.ietf-httpbis-p6-cache] Section 4.2.3 for details of determining response age. For example, a response:

```
HTTP/1.1 200 OK
Content-Type: text/html
Cache-Control: 600
Age: 30
Alt-Svc: "h2c"=8000; ma=60
```

indicates that an alternative service is available and usable for the next 60 seconds. However, the response has already been cached for 30 seconds (as per the Age header field value), so therefore the alternative service is only fresh for the 30 seconds from when this response was received, minus estimated transit time.

When an Alt-Svc response header is received from an origin, its value invalidates and replaces all cached alternative services for that origin.

See Section 2.2 for general requirements on caching alternative services.

Note that the freshness lifetime for HTTP caching (here, 600 seconds) does not affect caching of Alt-Svc values.

4. The 4NN Not Authoritative HTTP Status Code

The 4NN (Not Authoritative) status code indicates that the current origin server (usually, but not always an alternative service; see Section 2) is not authoritative for the requested resource, in the sense of [I-D.ietf-httpbis-p1-messaging], Section 9.1.

Clients receiving 4NN (Not Authoritative) from an alternative service MUST remove the corresponding entry from its alternative service cache (see Section 2.2) for that origin. Regardless of the idempotency of the request method, they MAY retry the request, either at another alternative server, or at the origin.

4NN (Not Authoritative) MAY carry an Alt-Svc header field.

This status code MUST NOT be generated by proxies.

A 4NN response is cacheable by default; i.e., unless otherwise indicated by the method definition or explicit cache controls (see Section 4.2.2 of [I-D.ietf-httpbis-p6-cache]).

5. IANA Considerations

5.1. The Alt-Svc Message Header Field

This document registers Alt-Svc in the Permanent Message Header Registry [RFC3864].

- Header Field Name: Alt-Svc
- Application Protocol: http
- Status: standard
- Author/Change Controller: IETF
- Specification Document: [this document]
- Related Information:

5.2. The 4NN Not Authoritative HTTP Status Code

This document registers the 4NN (Not Authoritative) HTTP Status code [I-D.ietf-httpbis-p2-semantics].

- Status Code: 4NN
- Short Description: Not Authoritative
- Specification: [this document], Section 4

6. Security Considerations

Identified security considerations should be enumerated in the appropriate documents depending on which proposals are accepted. Those listed below are generic to all uses of alternative services; more specific ones might be necessary.

6.1. Changing Ports

Using an alternative service implies accessing an origin's resources on an alternative port, at a minimum. An attacker that can inject alternative services and listen at the advertised port is therefore able to hijack an origin.

For example, an attacker that can add HTTP response header fields can redirect traffic to a different port on the same host using the Alt-Svc header field; if that port is under the attacker's control, they

can thus masquerade as the HTTP server.

This risk can be mitigated by restricting the ability to advertise alternative services, and restricting who can open a port for listening on that host.

6.2. Changing Hosts

When the host is changed due to the use of an alternative service, it presents an opportunity for attackers to hijack communication to an origin.

For example, if an attacker can convince a user agent to send all traffic for "innocent.example.org" to "evil.example.com" by successfully associating it as an alternative service, they can masquerade as that origin. This can be done locally (see mitigations above) or remotely (e.g., by an intermediary as a man-in-the-middle attack).

This is the reason for the requirement in Section 2.1 that any alternative service with a host different to the origin's be strongly authenticated with the origin's identity; i.e., presenting a certificate for the origin proves that the alternative service is authorized to serve traffic for the origin.

However, this authorization is only as strong as the method used to authenticate the alternative service. In particular, there are well-known exploits to make an attacker's certificate appear as legitimate.

Alternative services could be used to persist such an attack; for example, an intermediary could man-in-the-middle TLS-protected communication to a target, and then direct all traffic to an alternative service with a large freshness lifetime, so that the user agent still directs traffic to the attacker even when not using the intermediary.

As a result, there is a requirement in Section 2.2 to examine cached alternative services when a network change is detected.

6.3. Changing Protocols

When the ALPN protocol is changed due to the use of an alternative service, the security properties of the new connection to the origin can be different from that of the "normal" connection to the origin, because the protocol identifier itself implies this.

For example, if a "https://" URI had a protocol advertised that does

not use some form of end-to-end encryption (most likely, TLS), it violates the expectations for security that the URI scheme implies.

Therefore, clients cannot blindly use alternative services, but instead evaluate the option(s) presented to assure that security requirements and expectations (of specifications, implementations and end users) are met.

7. References

7.1. Normative References

[I-D.ietf-httpbis-p1-messaging]
Fielding, R. and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", draft-ietf-httpbis-p1-messaging-26 (work in progress), February 2014.

[I-D.ietf-httpbis-p6-cache]
Fielding, R., Nottingham, M., and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Caching", draft-ietf-httpbis-p6-cache-26 (work in progress), February 2014.

[I-D.ietf-tls-applayerprotoneg]
Friedl, S., Popov, A., Langley, A., and S. Emile, "Transport Layer Security (TLS) Application Layer Protocol Negotiation Extension", draft-ietf-tls-applayerprotoneg-05 (work in progress), March 2014.

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

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

[RFC6454] Barth, A., "The Web Origin Concept", RFC 6454, December 2011.

7.2. Informative References

[I-D.ietf-httpbis-http2]
Belshe, M., Peon, R., and M. Thomson, "Hypertext Transfer Protocol version 2", draft-ietf-httpbis-http2-10 (work in progress), February 2014.

[I-D.ietf-httpbis-p2-semantics]
Fielding, R. and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", draft-ietf-httpbis-p2-semantics-26 (work in progress), February 2014.

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, September 2004.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

Appendix A. Acknowledgements

Thanks to Eliot Lear, Stephen Farrell, Guy Podjarny, Stephen Ludin, Erik Nygren, Paul Hoffman, Adam Langley, Will Chan and Richard Barnes for their feedback and suggestions.

The Alt-Svc header field was influenced by the design of the Alternative-Protocol header in SPDY.

Authors' Addresses

Mark Nottingham
Akamai

Email: mnot@mnot.net
URI: http://www.mnot.net/

Patrick McManus
Mozilla

Email: mcmanus@ducksong.com
URI: https://mozillians.org/u/pmcmanus/