HTTP Alternative Services
draft-ietf-httpbis-alt-svc-06

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.

Editorial Note (To be removed by RFC Editor)

Discussion of this draft takes place on the HTTPBIS working group mailing list (ietf-http-wg@w3.org), which is archived at <https://lists.w3.org/Archives/Public/ietf-http-wg/>.

Working Group information can be found at <https://tools.ietf.org/wg/httpbis/> and <http://httpwg.github.io/>; source code and issues list for this draft can be found at <https://github.com/httpwg/http-extensions>.

The changes in this draft are summarized in Appendix A.

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 August 9, 2015.
Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the
document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal
Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info) in effect on the date of
publication of this document. Please review these documents
carefully, as they describe your rights and restrictions with respect
to this document. Code Components extracted from this document must
include Simplified BSD License text as described in Section 4.e of
the Trust Legal Provisions and are provided without warranty as
described in the Simplified BSD License.
1. Introduction

HTTP [RFC7230] 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 identification and location in HTTP; keeping the same identifier for a resource, but interacting with it at a different location on the network.

For example:

- An origin server might wish to redirect a client to a different server when it needs to go down for maintenance, or it has found a server 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, see [HTTP2]) or one using improved security (such as Transport Layer Security (TLS), see [RFC5246]).
- An origin server might wish to segment its clients into groups of capabilities, such as those supporting Server Name Indication (SNI, see Section 3 of [RFC6066]) and those not supporting it, for operational purposes.

This specification defines a new concept in HTTP, "Alternative Services", that allows an origin server to nominate additional means of interacting with it on the network. It defines a general framework for this in Section 2, along with specific mechanisms for advertising their existence using HTTP header fields (Section 3) or an HTTP/2 frame type (Section 4).

It also introduces a new status code in Section 6, 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", "delta-seconds", "parameter", "port", "quoted-string", "token", and "uri-host" rules from [RFC7230], 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 that are accessible through a different protocol / host / port combination, it is said to have an alternative service available.

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 [RFC7230], 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 can 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 service will need to present a certificate for the origin’s host name, not that of the alternative. Likewise, the Host header field ([RFC7230], Section 5.4) 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 Application Layer Protocol Negotiation (ALPN) protocol, as per [RFC7301]
- A host, as per [RFC3986], Section 3.2.2
- A port, as per [RFC3986], Section 3.2.3

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. This document describes two such mechanisms: an HTTP header field (Section 3) and an HTTP/2 frame type (Section 4).

The remainder of this section describes requirements that are common to alternative services, regardless of how they are discovered.

2.1. Host Authentication

Clients MUST NOT use alternative services with a host that is different than the origin’s without strong server authentication; this mitigates the attack described in Section 9.2. 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.
2.2. Alternative Service Caching

Mechanisms for discovering alternative services also 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 an alternative service do not need to stop using it 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.

Clients ought to consider that changes in network configurations can result in suboptimal or compromised cached alternative services.

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). This supports the conservation of IP addresses on the alternative service host.

Note that the SNI information provided in TLS by the client will be that of the origin, not the alternative (as will the Host HTTP header field-value).

2.4. Using Alternative Services

By their nature, alternative services are OPTIONAL: clients do not need 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.

If a client becomes aware of multiple alternative services, it MAY choose the most suitable according to its own criteria (again, keeping security properties in mind). For example, an origin might advertise multiple alternative services to notify clients of support for multiple versions of HTTP; or, an alternative service might itself advertise an alternative.
When a client uses an alternative service for a request, it can indicate this to the server using the Alt-Used header field (Section 5).

The client does not need to block requests on any existing connection; it 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 or another alternative service. 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

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

```
Alt-Svc = 1#( alternative *( OWS ';' OWS parameter ) )
alternative = protocol-id "=" alt-authority
protocol-id = token ; percent-encoded ALPN protocol identifier
alt-authority = quoted-string ; containing [ uri-host ] ":" port
```

ALPN protocol names are octet sequences with no additional constraints on format. Octets not allowed in tokens ([RFC7230], Section 3.2.6) MUST be percent-encoded as per Section 2.1 of [RFC3986]. Consequently, the octet representing the percent character "%" (hex 25) MUST be percent-encoded as well.

In order to have precisely one way to represent any ALPN protocol name, the following additional constraints apply:

1. Octets in the ALPN protocol MUST NOT be percent-encoded if they are valid token characters except "%", and
2. When using percent-encoding, uppercase hex digits MUST be used.

With these constraints, recipients can apply simple string comparison to match protocol identifiers.

The "alt-authority" component consists of an OPTIONAL uri-host ("host" in Section 3.2.2 of [RFC3986]), a colon (":"), and a port number.
For example:

Alt-Svc: h2=":8000"

This indicates the "h2" protocol ([HTTP2]) on the same host using the indicated port 8000.

An example involving a change of host:

Alt-Svc: h2="new.example.org:80"

This indicates the "h2" protocol on the host "new.example.org", running on port 80. Note that the "quoted-string" syntax needs to be used because "." is not an allowed character in "token".

Examples for protocol name escaping:

<table>
<thead>
<tr>
<th>ALPN protocol name</th>
<th>protocol-id</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>h2</td>
<td>h2</td>
<td>No escaping needed</td>
</tr>
<tr>
<td>w=x:y#z</td>
<td>w%3Dx%3Ay#z</td>
<td>&quot;=&quot; and &quot;:&quot; escaped</td>
</tr>
<tr>
<td>x%y</td>
<td>x%25y</td>
<td>&quot;%&quot; needs escaping</td>
</tr>
</tbody>
</table>

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

The Alt-Svc field value can 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.

When using HTTP/2 ([HTTP2]), servers SHOULD instead send an ALT SVC frame (Section 4). A single ALTSVC frame can be sent for a connection; a new frame is not needed for every request.

Note that all field elements that allow "quoted-string" syntax MUST be processed as per Section 3.2.6 of [RFC7230].
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 Section 4.2.3 of [RFC7234] 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.

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

When an Alt-Svc response header field 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.

4. The ALTSVC HTTP/2 Frame

The ALTSVC HTTP/2 frame ([HTTP2], Section 4) advertises the availability of an alternative service to an HTTP/2 client.

The ALTSVC frame is a non-critical extension to HTTP/2. Endpoints that do not support this frame can safely ignore it.
An ALTSVC frame from a server to a client on a client-initiated stream indicates that the conveyed alternative service is associated with the origin of that stream.

An ALTSVC frame from a server to a client on stream 0 indicates that the conveyed alternative service is associated with the origin contained in the Origin field of the frame. An association with an origin that the client does not consider authoritative for the current connection MUST be ignored.

The ALTSVC frame type is 0xa (decimal 10).

```
+-------------------------------+-------------------------------+
|         Origin-Len (16)       | Origin? (*)                 ...
+-------------------------------+-------------------------------+
|                   Alt-Svc-Field-Value (*)                   ...
+---------------------------------------------------------------+
```

ALTSVC Frame Payload

The ALTSVC frame contains the following fields:

- **Origin-Len**: An unsigned, 16-bit integer indicating the length, in octets, of the Origin field.

- **Origin**: An OPTIONAL sequence of characters containing the ASCII serialization of an origin ([RFC6454], Section 6.2) that the alternative service is applicable to.

- **Alt-Svc-Field-Value**: A sequence of octets (length determined by subtracting the length of all preceding fields from the frame length) containing a value identical to the Alt-Svc field value defined in Section 3 (ABNF production "Alt-Svc").

The ALTSVC frame does not define any flags.

The ALTSVC frame is intended for receipt by clients; a server that receives an ALTSVC frame MUST treat it as a connection error of type PROTOCOL_ERROR.

An ALTSVC frame on a client-initiated stream containing non-empty "Origin" information is invalid and MUST be ignored. Likewise, an ALTSVC frame on stream 0 with empty (length 0) "Origin" information is invalid and MUST be ignored.

The ALTSVC frame is processed hop-by-hop. An intermediary MUST NOT forward ALTSVC frames, though it can use the information contained in ALTSVC frames in forming new ALTSVC frames to send to its own
5. The Alt-Used HTTP Header Field

The Alt-Used header field is used in requests to indicate the identity of the alternative service in use, just as the Host header field (Section 5.4 of [RFC7230]) identifies the host and port of the origin.

Alt-Used = uri-host [ "": port ]

Alt-Used is intended to allow alternative services to detect loops, differentiate traffic for purposes of load balancing, and generally to ensure that it is possible to identify the intended destination of traffic, since introducing this information after a protocol is in use has proven to be problematic.

When using an alternative service, clients SHOULD include a Alt-Used header field in all requests.

As the Alt-Used header field might be used by the server for tracking the client, a client MAY choose not to include it in its requests for protecting its privacy (see Section 9.4).

For example:

```
GET /thing HTTP/1.1
Host: origin.example.com
Alt-Used: alternate.example.net
```

The extension parameters (ext-param) are reserved for future use; specifications that want to define an extension will need to update this document (and ought to introduce an extension registry).

6. The 421 Misdirected Request HTTP Status Code

The 421 (Misdirected Request) status code is defined in Section 9.1.2 of [HTTP2] to indicate that the current server instance is not authoritative for the requested resource. This can be used to indicate that an alternative service is not authoritative; see Section 2).

Clients receiving 421 (Misdirected Request) 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.
A 421 (Misdirected Request) response MAY carry an Alt-Svc header field.

7. IANA Considerations

7.1. Header Field Registrations

HTTP header fields are registered within the "Message Headers" registry maintained at <https://www.iana.org/assignments/message-headers/>.

This document defines the following HTTP header fields, so their associated registry entries shall be added according to the permanent registrations below (see [BCP90]):

+-------------------+----------+----------+-----------+
| Header Field Name | Protocol | Status   | Reference |
+-------------------+----------+----------+-----------+
| Alt-Svc           | http     | standard | Section 3 |
| Alt-Used          | http     | standard | Section 5 |
+-------------------+----------+----------+-----------+

The change controller is: "IETF (iesg@ietf.org) - Internet Engineering Task Force".

7.2. The ALTSVC HTTP/2 Frame Type

This document registers the ALTSVC frame type in the HTTP/2 Frame Types registry ([HTTP2], Section 11.2).

Frame Type: ALTSVC

Code: 0xa

Specification: Section 4 of this document

8. Internationalization Considerations

An internationalized domain name that appears in either the header field (Section 3) or the HTTP/2 frame (Section 4) MUST be expressed using A-labels ([RFC5890], Section 2.3.2.1).

9. Security Considerations

9.1. 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.

9.2. 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 in Section 9.1) 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.

9.3. 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.

9.4. Tracking Clients Using Alternative Services

Choosing an alternative service implies connecting to a new, server-supplied host name. By using many different (potentially unique) host names, servers could conceivably track client requests.

Clients concerned by the additional fingerprinting can choose to ignore alternative service advertisements.

In a browser, any alternative service information MUST be removed when origin-specific data is cleared (for instance, when cookies are cleared).

10. Acknowledgements


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

11. References

11.1. Normative References


11.2. Informative References


Appendix A. Change Log (to be removed by RFC Editor before publication)

A.1. Since draft-nottingham-httpbis-alt-svc-05

This is the first version after adoption of draft-nottingham-httpbis-alt-svc-05 as Working Group work item. It only contains editorial changes.

A.2. Since draft-ietf-httpbis-alt-svc-00

Selected 421 as proposed status code for "Not Authoritative".

Changed header field syntax to use percent-encoding of ALPN protocol names (<https://github.com/http2/http2-spec/issues/446>).

Nottingham, et al. Expires August 9, 2015
A.3. Since draft-ietf-httpbis-alt-svc-01

Updated HTTP/1.1 references.

Renamed "Service" to "Alt-Svc-Used" and reduced information to a flag to address fingerprinting concerns (<https://github.com/http2/http2-spec/issues/502>).

Note that ALTSVC frame is preferred to Alt-Svc header field (<https://github.com/http2/http2-spec/pull/503>).


Moved definition of status code 421 to HTTP/2.


A.4. Since draft-ietf-httpbis-alt-svc-02

Updated ALPN reference.


A.5. Since draft-ietf-httpbis-alt-svc-03


Remove/tune language with respect to tracking risks (see <https://github.com/httpwg/http-extensions/issues/34>).


"421 (Not Authoritative)" -> "421 (Misdirected Request)".

Allow the frame to carry multiple indicator and use the same payload formats for both (<https://github.com/httpwg/http-extensions/issues/37>).
A.7. Since draft-ietf-httpbis-alt-svc-05

Go back to specifying the origin in Alt-Used, but make it a "SHOULD" (<https://github.com/httpwg/http-extensions/issues/34>).


Authors’ Addresses

Mark Nottingham
Akamai

EMail: mnot@mnot.net
URI: https://www.mnot.net/

Patrick McManus
Mozilla

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

Julian F. Reschke
greenbytes GmbH

EMail: julian.reschke@greenbytes.de
URI: https://greenbytes.de/tech/webdav/
The Hypertext Transfer Protocol (HTTP) Authentication-Info and Proxy-
Authentication-Info Response Header Fields
draft-ietf-httpbis-auth-info-04

Abstract

This specification defines the "Authentication-Info" and "Proxy-
Authentication-Info" response header fields for use in HTTP
authentication schemes which need to return information once the
client's authentication credentials have been accepted.

Editorial Note (To be removed by RFC Editor)

Discussion of this draft takes place on the HTTPBIS working group
mailing list (ietf-http-wg@w3.org), which is archived at
<https://lists.w3.org/Archives/Public/ietf-http-wg/>.

Working Group information can be found at
source code and issues list for this draft can be found at

The changes in this draft are summarized in Appendix A.5.

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 12, 2015.
Table of Contents

1. Introduction ............................................. 3
2. Notational Conventions ................................. 3
3. The Authentication-Info Response Header Field .......... 3
   3.1. Parameter Value Format .......................... 4
4. The Proxy-Authentication-Info Response Header Field ...... 4
5. Security Considerations .................................. 4
6. IANA Considerations ..................................... 5
7. Acknowledgements ....................................... 5
8. References ............................................... 5
   8.1. Normative References .............................. 5
   8.2. Informative References ............................ 6
Appendix A. Change Log (to be removed by RFC Editor before publication) .................................. 6
   A.1. draft-reschke-httpauth-auth-info-00 ................ 6
   A.2. draft-ietf-httpbis-auth-info-00 ................... 6
   A.3. draft-ietf-httpbis-auth-info-01 ................... 6
   A.4. draft-ietf-httpbis-auth-info-02 ................... 6
   A.5. draft-ietf-httpbis-auth-info-03 ................... 7
1. Introduction

This specification defines the "Authentication-Info" and "Proxy-Authentication-Info" response header fields for use in HTTP authentication schemes ([RFC7235]) which need to return information once the client's authentication credentials have been accepted.

Both were previously defined in Section 3 of [RFC2617], defining the HTTP "Digest" authentication scheme. This document generalizes the description for use not only in "Digest" ([DIGEST]), but also in other future schemes that might have the same requirements for carrying additional information during authentication.

2. Notational Conventions

This specification uses the Augmented Backus-Naur Form (ABNF) notation of [RFC5234] with a list extension, defined in Section 7 of [RFC7230], that allows for compact definition of comma-separated lists using a '#' operator (similar to how the '*' operator indicates repetition). The ABNF production for "auth-param" is defined in Section 2.1 of [RFC7235].

3. The Authentication-Info Response Header Field

HTTP authentication schemes can use the Authentication-Info response header field to communicate information after the client’s authentication credentials have been accepted. This information can include a finalization message from the server (e.g., it can contain the server authentication).

The field value is a list of parameters (name/value pairs), using the "auth-param" syntax defined in Section 2.1 of [RFC7235]. This specification only describes the generic format; authentication schemes using "Authentication-Info" will define the individual parameters. The "Digest" Authentication Scheme, for instance, defines multiple parameters in Section 3.5 of [DIGEST].

Authentication-Info = #auth-param

The Authentication-Info header field can be used in any HTTP response, independently of request method and status code. Its semantics are defined by the authentication scheme indicated by the Authorization header field of the corresponding request.

A proxy forwarding a response is not allowed to modify the field value in any way.

Authentication-Info can be used inside trailers ([RFC7230], Section
4.1.2) when the authentication scheme explicitly allows this.

3.1. Parameter Value Format

Parameter values can be expressed either as "token" or as "quoted-string" (Section 3.2.6 of [RFC7230]).

Authentication scheme definitions need to allow both notations, both for senders and recipients. This allows recipients to use generic parsing components, independent of the authentication scheme in use.

For backwards compatibility, authentication scheme definitions can restrict the format for senders to one of the two variants. This can be important when it is known that deployed implementations will fail when encountering one of the two formats.

4. The Proxy-Authentication-Info Response Header Field

The Proxy-Authentication-Info response header field is equivalent to Authentication-Info, except that it applies to proxy authentication ([RFC7235]):

Proxy-Authentication-Info = #auth-param

However, unlike Authentication-Info, the Proxy-Authentication-Info header field applies only to the next outbound client on the response chain. This is because only the client that chose a given proxy is likely to have the credentials necessary for authentication. However, when multiple proxies are used within the same administrative domain, such as office and regional caching proxies within a large corporate network, it is common for credentials to be generated by the user agent and passed through the hierarchy until consumed. Hence, in such a configuration, it will appear as if Proxy-Authentication-Info is being forwarded because each proxy will send the same field value.

5. Security Considerations

Adding information to HTTP responses that are sent over an unencrypted channel can affect security and privacy. The presence of the header fields alone indicates that HTTP authentication is in use. Additional information could be exposed by the contents of the authentication-scheme specific parameters; this will have to be considered in the definitions of these schemes.
6. IANA Considerations

HTTP header fields are registered within the "Message Headers" registry located at <http://www.iana.org/assignments/message-headers>, as defined by [BCP90].

This document updates the definitions of the "Authentication-Info" and "Proxy-Authentication-Info" header fields, so the "Permanent Message Header Field Names" registry shall be updated accordingly:

<table>
<thead>
<tr>
<th>Header Field Name</th>
<th>Protocol</th>
<th>Status</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication-Info</td>
<td>http</td>
<td>standard</td>
<td>Section 3 of this document</td>
</tr>
<tr>
<td>Proxy-Authentication-Info</td>
<td>http</td>
<td>standard</td>
<td>Section 4 of this document</td>
</tr>
</tbody>
</table>

7. Acknowledgements

This document is based on the header field definitions in RFCs 2069 and 2617, whose authors are: John Franks, Phillip M. Hallam-Baker, Jeffery L. Hostetler, Scott D. Lawrence, Paul J. Leach, Ari Luotonen, Eric W. Sink, and Lawrence C. Stewart.

Additional thanks go to the members of the HTTPAuth and HTTPbis Working Groups, namely Amos Jeffries, Benjamin Kaduk, Alexey Melnikov, Mark Nottingham, Yutaka Oiwa, Rifaat Shekh-Yusef, and Martin Thomson.

8. References

8.1. Normative References


8.2. Informative References


Appendix A. Change Log (to be removed by RFC Editor before publication)

A.1. draft-reschke-httpauth-auth-info-00

Changed boilerplate to make this an HTTPbis WG draft. Added Acknowledgements.

In the Security Considerations, remind people that those apply to unencrypted channels.

Make it clearer that these are really just response header fields.

A.2. draft-ietf-httpbis-auth-info-00

Rephrase introduction of header field to be closer to what RFC 2617 said ("successful authentication").

Update DIGEST reference.

A.3. draft-ietf-httpbis-auth-info-01

State that scheme definitions need to define whether the header field can be used in trailers.

Add "updates: 2617" to boilerplate.

A.4. draft-ietf-httpbis-auth-info-02

Updated DIGEST reference.

The do-not-modify rule does not include proxies that consume Authentication-Info (<https://github.com/httpwg/http-extensions/issues/50>).


A.5. draft-ietf-httpbis-auth-info-03

Updated DIGEST reference.

Clarify how the applicable auth scheme is determined (it is present in the request’s (Proxy-)Authorization header field).

Adjust the IPR boilerplate because we are using some text from RFC 2617.

Author’s Address

Julian F. Reschke
greenbytes GmbH
Hafenweg 16
Muenster, NW 48155
Germany

EMail: julian.reschke@greenbytes.de
URI: http://greenbytes.de/tech/webdav/
Abstract

This document describes how "http" URIs can be accessed using Transport Layer Security (TLS) to mitigate pervasive monitoring attacks.

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 June 18, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
1. Introduction

This document describes a use of HTTP Alternative Services [I-D.ietf-httpbis-alt-svc] to decouple the URI scheme from the use and configuration of underlying encryption, allowing a "http" URI to be accessed using TLS [RFC5246] opportunistically.

Currently, "https" URIs require acquiring and configuring a valid certificate, which means that some deployments find supporting TLS difficult. Therefore, this document describes a usage model whereby sites can serve "http" URIs over TLS without being required to support strong server authentication.

Opportunistic Security [I-D.dukhovni-opportunistic-security] does not provide the same guarantees as using TLS with "https" URIs; it is vulnerable to active attacks, and does not change the security context of the connection. Normally, users will not be able to tell that it is in use (i.e., there will be no "lock icon").

By its nature, this technique is vulnerable to active attacks. A mechanism for partially mitigating them is described in Section 5. It does not offer the same level of protection as afforded to "https" URIs, but increases the likelihood that an active attack be detected.
1.1. Goals and Non-Goals

The immediate goal is to make the use of HTTP more robust in the face of pervasive passive monitoring [RFC7258].

A secondary goal is to limit the potential for active attacks. It is not intended to offer the same level of protection as afforded to "https" URIs, but instead to increase the likelihood that an active attack can be detected.

A final (but significant) goal is to provide for ease of implementation, deployment and operation. This mechanism should have a minimal impact upon performance, and should not require extensive administrative effort to configure.

1.2. 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].

2. Using HTTP URIs over TLS

An origin server that supports the resolution of "http" URIs can indicate support for this specification by providing an alternative service advertisement [I-D.ietf-httpbis-alt-svc] for a protocol identifier that uses TLS, such as "h2" [I-D.ietf-httpbis-http2].

A client that receives such an advertisement MAY make future requests intended for the associated origin ([RFC6454]) to the identified service (as specified by [I-D.ietf-httpbis-alt-svc]).

A client that places the importance of protection against passive attacks over performance might choose to withhold requests until an encrypted connection is available. However, if such a connection cannot be successfully established, the client MAY resume its use of the cleartext connection.

A client can also explicitly probe for an alternative service advertisement by sending a request that bears little or no sensitive information, such as one with the OPTIONS method. Likewise, clients with existing alternative services information could make such a request before they expire, in order minimize the delays that might be incurred.
3. Server Authentication

By their nature, "http" URIs do not require cryptographically strong server authentication; that is only implied by "https" URIs. Furthermore, doing so (as per [RFC2818]) creates a number of operational challenges. For these reasons, server authentication is not mandatory for "http" URIs when using the mechanism described in this specification.

When connecting to an alternative service for an "http" URI, clients are not required to perform the server authentication procedure described in Section 3.1 of [RFC2818]. The server certificate, if one is proffered by the alternative service, is not necessarily checked for validity, expiration, issuance by a trusted certificate authority or matched against the name in the URI. Therefore, the alternative service MAY provide any certificate, or even select TLS cipher suites that do not include authentication.

A client MAY perform additional checks on the offered certificate if the server does not select an unauthenticated TLS cipher suite. This document doesn’t define any such checks, though clients could be configured with a policy that defines what is acceptable.

As stipulated by [I-D.ietf-httpbis-alt-svc], clients MUST NOT use alternative services with a host other than the origin’s, unless the alternative service itself is strongly authenticated (as the origin’s host); for example, using TLS with a certificate that validates as per [RFC2818].

4. Interaction with "https" URIs

When using alternative services, both "http" and "https" URIs might use the same connection, because HTTP/2 permits requests for multiple origins on the same connection.

Since "https" URIs rely on server authentication, a connection that is initially created for "http" URIs without authenticating the server cannot be used for "https" URIs until the server certificate is successfully authenticated. Section 3.1 of [RFC2818] describes the basic mechanism, though the authentication considerations in [I-D.ietf-httpbis-alt-svc] also apply.

Connections that are established without any means of server authentication (for instance, the purely anonymous TLS cipher suites), cannot be used for "https" URIs.
5. Requiring Use of TLS

Editors’ Note: this is a very rough take on an approach that would provide a limited form of protection against downgrade attack. It’s unclear at this point whether the additional effort (and modest operational cost) is worthwhile.

The mechanism described in this specification is trivial to mount an active attack against, for two reasons:

- A client that doesn’t perform authentication an easy victim of server impersonation, through man-in-the-middle attacks.
- A client that is willing to use cleartext to resolve the resource will do so if access to any TLS-enabled alternative services is blocked at the network layer.

Given that the primary goal of this specification is to prevent passive attacks, these are not critical failings (especially considering the alternative - HTTP over cleartext). However, a modest form of protection against active attacks can be provided for clients on subsequent connections.

When an alternative service is able to commit to providing service for a particular origin over TLS for a bounded period of time, clients can choose to rely upon its availability, failing when it cannot be contacted. Effectively, this makes the choice to use a secured protocol "sticky" in the client.

5.1. The HTTP-TLS Header Field

A alternative service can make this commitment by sending a "HTTP-TLS" header field:

HTTP-TLS     = 1#parameter

When it appears in a HTTP response from a strongly authenticated alternative service, this header field indicates that the availability of the origin through TLS-protected alternative services is "sticky", and that the client MUST NOT fall back to cleartext protocols while this information is considered fresh.

For example:
GET /index.html HTTP/1.1
Host: example.com

HTTP/1.1 200 OK
Content-Type: text/html
Cache-Control: 600
Age: 30
Date: Thu, 1 May 2014 16:20:09 GMT
HTTP-TLS: ma=3600

This header field creates a commitment from the origin [RFC6454] of the associated resource (in the example, "http://example.com"). For the duration of the commitment, clients SHOULD strongly authenticate the server for all subsequent requests made to that origin, though this creates some risks for clients Section 5.2.

Authentication for HTTP over TLS is described in Section 3.1 of [RFC2818], noting the additional requirements in [I-D.ietf-httpbis-alt-svc]. The header field MUST be ignored if strong authentication fails; otherwise, an attacker could create a persistent denial of service by falsifying a commitment.

The commitment to use authenticated TLS persists for a period determined by the value of the "ma" parameter. See Section 4.2.3 of [RFC7234] for details of determining response age.

ma-parameter = delta-seconds

The commitment made by the "HTTP-TLS" header field applies only to the origin of the resource that generates the "HTTP-TLS" header field. Requests for an origin that has a persisted, unexpired value for "HTTP-TLS" MUST fail if they cannot be made over an authenticated TLS connection.

Note that the commitment is not bound to a particular alternative service. Clients SHOULD use alternative services that they become aware of. However, clients MUST NOT use an unauthenticated alternative service for an origin with this commitment. Where there is an active commitment, clients MAY instead ignore advertisements for unsecured alternatives services.

5.2. Operational Considerations

To avoid situations where a persisted value of "HTTP-TLS" causes a client to be unable to contact a site, clients SHOULD limit the time that a value is persisted for a given origin. A lower limit might be
appropriate for initial observations of "HTTP-TLS"; the certainty that a site has set a correct value - and the corresponding limit on persistence - can increase as the value is seen more over time.

Once a server has indicated that it will support authenticated TLS, a client MAY use key pinning [I-D.ietf-websec-key-pinning] or any other mechanism that would otherwise be restricted to use with "https" URIs, provided that the mechanism can be restricted to a single HTTP origin.

6. Security Considerations

6.1. Security Indicators

User Agents MUST NOT provide any special security indicia when an "http" resource is acquired using TLS. In particular, indicators that might suggest the same level of security as "https" MUST NOT be used (e.g., using a "lock device").

6.2. Downgrade Attacks

A downgrade attack against the negotiation for TLS is possible. With the "HTTP-TLS" header field, this is limited to occasions where clients have no prior information (see Section 6.3), or when persisted commitments have expired.

For example, because the "Alt-Svc" header field [I-D.ietf-httpbis-alt-svc] likely appears in an unauthenticated and unencrypted channel, it is subject to downgrade by network attackers. In its simplest form, an attacker that wants the connection to remain in the clear need only strip the "Alt-Svc" header field from responses.

Downgrade attacks can be partially mitigated using the "HTTP-TLS" header field, because when it is used, a client can avoid using cleartext to contact a supporting server. However, this only works when a previous connection has been established without an active attacker present; a continuously present active attacker can either prevent the client from ever using TLS, or offer its own certificate.

6.3. Privacy Considerations

Cached alternative services can be used to track clients over time; e.g., using a user-specific hostname. Clearing the cache reduces the ability of servers to track clients; therefore clients MUST clear cached alternative service information when clearing other origin-based state (i.e., cookies).
6.4. Confusion Regarding Request Scheme

Many existing HTTP/1.1 implementations use the presence or absence of TLS in the stack to determine whether requests are for "http" or "https" resources. This is necessary in many cases because the most common form of an HTTP/1.1 request does not carry an explicit indication of the URI scheme.

HTTP/1.1 MUST NOT be sent over HTTP/1.1 or earlier versions of the protocol. Opportunistically secured HTTP requests MUST include an explicit scheme identifier.

7. References

7.1. Normative References

[I-D.ietf-httpbis-alt-svc]

[I-D.ietf-httpbis-http2]

[I-D.ietf-websec-key-pinning]
Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", draft-ietf-websec-key-pinning-21 (work in progress), October 2014.


7.2. Informative References

[I-D.dukhovni-opportunistic-security]


Appendix A. Acknowledgements


Authors' Addresses

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

Martin Thomson
Mozilla
Email: martin.thomson@gmail.com
The Tunnel-Protocol HTTP Header Field
draft-ietf-httpbis-tunnel-protocol-02

Abstract

This specification allows HTTP CONNECT requests to indicate what protocol will be used within the tunnel once established, using the Tunnel-Protocol header field.

Editorial Note (To be removed by RFC Editor)

Discussion of this draft takes place on the HTTPBIS working group mailing list (ietf-http-wg@w3.org), which is archived at [1].

Working Group information can be found at [2] and [3]; source code and issues list for this draft can be found at [4].

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 25, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

The HTTP CONNECT method (Section 4.3.6 of [RFC7231]) requests that the recipient establish a tunnel to the identified origin server and thereafter forward packets, in both directions, until the tunnel is closed. Such tunnels are commonly used to create end-to-end virtual connections, through one or more proxies.

The HTTP Tunnel-Protocol header field identifies the protocol that will be spoken within the tunnel, using the Application Layer Protocol Negotiation identifier (ALPN, [RFC7301]).

When the CONNECT method is used to establish a tunnel, the Tunnel-Protocol header field can be used to identify the protocol that the client intends to use with that tunnel. For a tunnel that is then secured using TLS [RFC5246], the header field carries the same application protocol label as will be carried within the TLS handshake. If there are multiple possible application protocols, all of those application protocols are indicated.

The Tunnel-Protocol header field carries an indication of client intent only. In TLS, the final choice of application protocol is made by the server from the set of choices presented by the client. Other protocols could negotiate protocols differently.
Proxies do not implement the tunneled protocol, though they might choose to make policy decisions based on the value of the header field. For example, a proxy could use the application protocol to select appropriate traffic prioritization.

1.1. Requirements Language

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 RFC 2119 [RFC2119].

2. The Tunnel-Protocol HTTP Header Field

Clients include the Tunnel-Protocol header field in an HTTP CONNECT request to indicate the application layer protocol that will be used within the tunnel, or the set of protocols that might be used within the tunnel.

2.1. Header Field Values

Valid values for the protocol field are taken from the "Application-Layer Protocol Negotiation (ALPN) Protocol ID" registry ([5]) established by [RFC7301].

2.2. Syntax

The ABNF (Augmented Backus-Naur Form) syntax for the Tunnel-Protocol header field is given below. It is based on the Generic Grammar defined in Section 2 of [RFC7230].

Tunnel-Protocol = "Tunnel-Protocol": 1#protocol-id
protocol-id = token ; percent-encoded ALPN protocol identifier

ALPN protocol names are octet sequences with no additional constraints on format. Octets not allowed in tokens ([RFC7230], Section 3.2.6) MUST be percent-encoded as per Section 2.1 of [RFC3986]. Consequently, the octet representing the percent character "%" (hex 25) MUST be percent-encoded as well.

In order to have precisely one way to represent any ALPN protocol name, the following additional constraints apply:

- Octets in the ALPN protocol MUST NOT be percent-encoded if they are valid token characters except "%", and
- When using percent-encoding, uppercase hex digits MUST be used.
With these constraints, recipients can apply simple string comparison to match protocol identifiers.

For example:

CONNECT www.example.com HTTP/1.1
Host: www.example.com
Tunnel-Protocol: h2, http%2F1.1

3. IANA Considerations

HTTP header fields are registered within the "Message Headers" registry maintained at [6]. This document defines and registers the Tunnel-Protocol header field, according to [RFC3864] as follows:

Header Field Name: Tunnel-Protocol
Protocol: http
Status: Standard
Reference: Section 2
Change Controller: IETF (iesg@ietf.org) - Internet Engineering Task Force

4. Security Considerations

In case of using HTTP CONNECT to a TURN server ("Traversal Using Relays around NAT", [RFC5766]) the security considerations of Section 4.3.6 of [RFC7231] apply. It states that there "are significant risks in establishing a tunnel to arbitrary servers, particularly when the destination is a well-known or reserved TCP port that is not intended for Web traffic. Proxies that support CONNECT SHOULD restrict its use to a limited set of known ports or a configurable whitelist of safe request targets."

The Tunnel-Protocol header field described in this document is an OPTIONAL header field. Clients and HTTP proxies could choose to not support the header and therefore fail to provide it, or ignore it when present. If the header is not available or ignored, a proxy cannot identify the purpose of the tunnel and use this as input to any authorization decision regarding the tunnel. This is indistinguishable from the case where either client or proxy does not support the Tunnel-Protocol header field.
The value of the Tunnel-Protocol header field could be falsified by a client. If the data being sent through the tunnel is encrypted (for example, with TLS [RFC5246]), then the proxy might not be able to directly inspect the data to verify that the claimed protocol is the one which is actually being used, though a proxy might be able to perform traffic analysis [TRAFFIC]. A proxy therefore cannot rely on the value of the Tunnel-Protocol header field as a policy input in all cases.

5. References

5.1. Normative References


5.2. Informative References

5.3. URIs


Authors’ Addresses

Andrew Hutton
Unify
Technology Drive
Nottingham NG9 1LA
UK
EMail: andrew.hutton@unify.com

Justin Uberti
Google
747 6th Ave S
Kirkland, WA 98033
US
EMail: justin@uberti.name

Martin Thomson
Mozilla
331 E Evelyn Street
Mountain View, CA 94041
US
EMail: martin.thomson@gmail.com