Network Working Group
Internet-Draft
Expires: August 22, 2002

M. Nottingham
February 21, 2002

HTTP Authentication Credential Caching Extension
draft-nottingham-http-auth-cache-00

Status of this Memo

Copyright Notice

Abstract

This note proposes an HTTP cache-control extension mechanism that allows caching of authentication credentials, thereby allowing authenticated resources to be served from cache without incurring the cost of a round-trip to the origin server more than once during the freshness lifetime of the credentials.

1. Introduction

HTTP [2] allows messages which are subject to authentication (such as that defined by RFC2617 [3]) to be cached when certain directives are present. In particular, Section 14.8 of RFC2616 says:

> When a shared cache (see section 13.7) receives a request containing an Authorization field, it MUST NOT return the corresponding response as a reply to any other request, unless one of the following specific exceptions holds:
>
> 1. If the response includes the s-maxage cache-control directive, the cache MAY use that response in replying to a subsequent request. But (if the specified maximum age has passed) a proxy cache MUST first revalidate it with the origin server, using the request-headers from the new request to allow the origin server to authenticate the new request. (This is the defined behavior for s-maxage.) If the response includes s-maxage= 0 , the proxy MUST always revalidate it before re-using it.
>
> 2. If the response includes the must-revalidate cache-control directive, the cache MAY use that response in replying to a subsequent request. But if the response is stale, all caches MUST first revalidate it with the origin server, using the request-headers from the new request to allow the origin server to authenticate the new request.
>
> 3. If the response includes the public cache-control directive, it MAY be returned in reply to any subsequent request.

The most useful approach here is that described in the end of #1, whereby a cache keeps the response, but revalidates new requests before serving it (Note that this can also be effected by use of a combination of the 'public' and 'max-age' cache-control directives).

This is useful for caching large representations (e.g., distributed binary programs, PDF files); the efficiency of the cache hit offsets the cost of going back to the origin server to authenticate the request. It is less useful for caching of smaller representations (such as images or HTML pages), because the efficiency gained from the cache does not overcome the latency introduced by the round trip to the origin server.

This note proposes an HTTP cache-control extension directive that allows caching of authentication credentials, thereby allowing authenticated resources to be served from cache without incurring the

cost of a round-trip to the origin server more than once during the freshness lifetime of the credentials.

Please direct comments to the HTTP-WG mailing list, http-wg@cuckoo.hpl.hp.com.

1.1 Requirements

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 [1].

An implementation is not compliant if it fails to satisfy one or more of the MUST or REQUIRED level requirements. An implementation that satisfies all the MUST or REQUIRED level and all the SHOULD level requirements is said to be "unconditionally compliant"; one that satisfies all the MUST level requirements but not all the SHOULD level requirements is said to be "conditionally compliant".

2. The Auth-Cache Cache-Control Extension Directive

The auth-realm cache-control directive allows caches to serve an authenticated response without validation on the origin server under controlled conditions.

auth-cache = "auth-cache" ["=" delta-seconds]

When a shared cache receives a request containing an Authorization field, it MAY return the corresponding response as a reply to a subsequent request, if all of the following conditions hold;

1. The auth-cache cache-control extension is present in the (cached) response.

2. The cached response credentials' realm matches that presented in the request, and the cached response and the Request-URI have the same canonical root URL (as defined by RFC2617, Section 1.2).

3. The presented credentials match the stored authentication state.

4. The response is fresh, according to its normal (non-authenticated) HTTP freshness lifetime.

5. The cached credentials are fresh, as outlined below.

By default, the freshness lifetime of the stored credentials is equal to that of the cached response. However, if the auth-cache directive includes a value, it is interpreted as the cached credentials'

freshness lifetime.

Implementations MUST generate 401 Authentication Required HTTP responses and WWW-Authenticate headers when requests for such resources do not present appropriate credentials.

3. Example

For example, if a shared cache contains a response for the URI http://www.example.org/resource which includes the following response headers:

```
Cache-Control: max-age=86400, auth-cache
WWW-Authenticate: Basic realm="WallyWorld"
```

This cached response can be served without validation, if:

- the request includes credentials that are valid for http://www.example.org
- the request includes credentials with the realm 'WallyWorld'
- the credentials have been validated on the origin server in the last day
- the response is fresh (i.e., has been validated on or directly fetched from the origin server in the last day)

Note that the cached credentials may have been associated with a different resource (e.g., http://www.example.org/Another/resource).

If the auth-cache directive included a value, for example:

```
Cache-Control: max-age=86400, auth-cache=3600
WWW-Authenticate: Basic realm="WallyWorld"
```

the same constraints would apply, except that the cached credentials would need to be one hour or fresher.

4. Security Considerations

Authentication caching is vulnerable in the same ways as normal HTTP authentication (as explained in RFC2616 and RFC2617), with the added risk inherent in delegating authority for authentication to another device or administrative domain, as applicable.

Additionally, the use of cached credentials introduces the possibility of a replay attack, sometimes in cases where there may

not have been such a risk previously. In particular, cached credentials SHOULD NOT be used in conjunction with Digest authentication, as doing so seriously weakens its security.

It should be noted that if the auth-cache directive is implemented by multiple devices in a chain of caches (e.g., hierarchical caching proxies), the cached credentials in some caches may in fact be older than the specified freshness lifetime. This issue may be addressed in future revisions of this note.

References

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

[2] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P. and T. Berners-Lee, "Hypertext Transfer Protocol - HTTP/1.1", RFC 2616, June 1999.

[3] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A. and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

Author's Address

Mark Nottingham

EMail: mnot@pobox.com
URI: http://www.mnot.net/

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.