An Architecture for Secure Content Delegation using HTTP
draft-thomson-http-scd-02

Abstract

An architecture is described for content distribution using a secondary server that might be operated with reduced privileges. This architecture allows a primary server to delegate the responsibility for delivery of the payload of an HTTP response to a secondary server. The secondary server is unable to modify this content. The content is encrypted, which in some cases will prevent the secondary server from learning about the content.

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 May 3, 2017.

Copyright Notice

Copyright (c) 2016 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
The distribution of content on the web at scale is necessarily highly distributed. Large amounts of content need large numbers of servers. And distributing those servers closer to clients has a significant, positive impact on performance.

A major drawback of existing solutions for content distribution is that a primary server is required to cede control of resources to the secondary server. The secondary server is able to see and modify content that they distribute.

There are few technical mechanisms in place to limit the capabilities of servers that provide content for a given origin. Mechanisms like content security policy [CSP] and sub-resource integrity [SRI] can be used to prevent modification of resources in some contexts, but these mechanisms are limited in what they can protect and they can impose certain operational costs. For the most part, server operators are forced to limit the content that is served on servers that are not
directly under their control or rely on non-technical measures such as contracts and courts to proscribe bad behavior.

1.1. Secure Content Delegation

This document describes how an primary origin server might securely delegate the responsibility for serving content to a secondary server.

The solution comprises three basic components:

- A delegation component allows a primary server to delegate specific resources to another server.
- Integrity attributes ensure that the content cannot be modified by the secondary server.
- Confidentiality protection limits the ability of the secondary server to learn what the content holds.

Note that the guarantees provided by confidentiality protection are not strong, see Section 4 for details.

In addition to these basic components, a fourth mechanism provides a client with the ability to learn resource metadata from the primary server prior to making a request for specific resources. This can dramatically improve performance where a client needs to acquire multiple delegated resources.

No new mechanisms are described in this document; the application of several existing and separately-proposed protocol mechanisms to this problem is described. A primary server can use these mechanisms to take advantage of secondary servers where concerns about security might have otherwise prevented their use. This might be for content that was previously considered too sensitive for third-party distribution, or to access secondary servers that were previously considered insufficiently trustworthy.

1.2. Notational Conventions

The words "MUST", "MUST NOT", "SHOULD", and "MAY" are used in this document. It’s not shouting; when they are capitalized, they have the special meaning defined in [RFC2119].

This document uses the terms client, primary server and secondary server. These terms refer to the three roles played in this architecture. Note that "primary server" as used in this document
encompasses the notion of both an origin server and a gateway as defined in [RFC7230].

2. Out-of-Band Content Encoding

The out-of-band content encoding [I-D.reschke-http-oob-encoding] provides the basis for delegation of content distribution. A request is made to the primary server, but in place of the complete response only response header fields and an out-of-band content encoding is provided. The out-of-band content encoding directs the client to retrieve content from another resource.

![Diagram of Request Flow]

Figure 1: Using Out-of-Band Content Encoding

Out-of-band content encoding behaves much like a redirect. In fact, a redirect was considered as part of the early design, but rejected because without defining a new set of 3xx status codes it would change the effective origin [RFC6454] of the resource. Furthermore, the content encoding specifically preserves header fields sent by the primary server, rejecting any unauthenticated header fields that might be provided by the secondary server.

2.1. Performance Trade-Off

An additional request is necessary to retrieve content. This has a negative impact on latency. However, if the secondary server is positioned close to the client, there are several potential benefits:

Fewer bit-miles: Content hosted in the secondary server that is nearby can be served to those clients without having to traverse a long network path.
Better server resource allocation: Using a dedicated secondary server reduces the load on the primary server, allowing it more capacity for serving other requests.

Better throughput: If a secondary server is closer to a client, more bandwidth might be available for delivery of content when compared with the link between client and primary server.

Lower time to last byte: For some resources, increased bandwidth can counteract the added latency cost of the extra requests, and potentially reduce the time needed to retrieve the entire resource.

The problems of providing integrity protection for content delivered in this fashion is discussed in Section 3; confidentiality protection and its limitations is described in Section 4; and reducing the latency impact of making multiple requests for each resource is described in Section 5.

2.2. Confidentiality of Resource Identity

The URL used to acquire a resource from a secondary server can be unrelated to the URL of the resource that refers to its contents. This allows a primary server to hide the relationship between content in a secondary server and the original resources that use that content.

Any entity SHOULD be unable to determine the URL of the original resource based on the URL of the secondary server resource alone. This can be achieved by having randomized URLs for secondary resources and maintaining a mapping table, or by using a fixed mapping function with a secret input such as HMAC [RFC2104].

Without other information, this would prevent the secondary server from learning which resources are requested from the primary server by observing the requests that it serves for out-of-band content. While in some cases, information about the resource is obtainable by the secondary server cache, see Section 4, an unpredictable mapping ensures that other protection mechanisms can be effective if possible.

3. Content Integrity

Ensuring that content is not modified by the secondary server is critical. Information that is acquired from the secondary server is not integrity protected and therefore MUST NOT be used without being authenticated.
A cryptographic hash over the content sent in the initial response could be compared against a hash of the content delivered by the secondary server. This is an expansion of the the basic design of [SRI].

A progressive integrity mechanism like the one described in [I-D.thomson-http-mice] ensures that there are no significant performance penalties imposed by the integrity protection. Progressive integrity allows for consumption of content as it is delivered without losing integrity protection.

A response from the primary server could include an M-I header field with an integrity proof, allowing the content to be delivered out-of-band without any additional header fields.

4. Content Confidentiality

Confidentiality protection for content is provided by applying an encryption content encoding [I-D.ietf-httpbis-encryption-encoding] to content before that content is provided to a secondary server.

Much of the value provided by a secondary server derives from its ability to deliver the same content to multiple nearby clients. The more clients that can be delivered the same resource, the greater the efficiency gains. As a result, resources that are provided to many or all clients are the ones that benefit most from caching.

This means that unless a resource has access control mechanisms that would prevent the secondary from accessing a resource, the confidentiality protections provided by encrypting content is limited. A secondary server need only independently request resources from the primary server in order to learn everything about the content it is serving, including the mapping of primary URLs to secondary URLs. For instance, employing a web crawler on a web site might reveal the identity of numerous resources and the location of the any out-of-band content for those resources.

Confidentiality protection allows resources that are protected by client authentication to remain confidential. Confidentiality protection also improves protections against cross-origin theft of confidential data (see Section 7.2).

5. Resource Map

Learning about header fields and out-of-band cache locations for resources in advance of needing to make requests to those resources allows a client to avoid making requests to the primary server. This can greatly improve the performance of applications that make
multiple requests of the same server, such as web browsing or video streaming.

Without defining any new additional protocol mechanisms, HTTP/2 server push [RFC7540] can be used to provide requests, responses and the out-of-band content encoding information describing resources. Since no actual content is included, this requires relatively little data to describe a number of resources. Once this information is available, the client no longer needs to contact the origin server to acquire the described resources.

This approach has some significant deployment drawbacks, so explicit data formats for carrying this data might be defined.

Note: We need a separate draft on these alternative methods.

6. Error Handling

Error handling for clients is described in [I-D.reschke-http-oob-encoding].

For idempotent requests, a second request might be made to the primary server. This request would omit any indication of support for out-of-band content coding from the Accept-Encoding header field, plus a link relation indicating the secondary resource and the reason for failure.

A primary server can use this information to make informed choices about whether to use content delegation.

Non-idempotent requests cannot be safely retried. Therefore, clients cannot retry a request and provide information about errors to the primary server. For this reason, primary servers SHOULD NOT delegate content for non-idempotent methods.

7. Security Considerations

This document describes a framework whereby content might be distributed to a secondary server, without losing integrity with respect to the content that is distributed.

This design relies on integrity and confidentiality for the request and response made to the primary server. These requests MUST be made using HTTP over TLS (HTTPS) [RFC2818] only. Though there is a lesser requirement for confidentiality, requests made to the secondary server MUST also be secured using HTTPS.
7.1. Confidentiality Protection Limitations

Content that requires only integrity protection can be safely distributed by a third-party using this design. Entities that make a decision about confidentiality for others have often been shown to be incorrect in the past. An incorrect conclusion have serious consequences. Thus the choice of whether confidentiality protection is needed is quite important.

Some confidentiality protection against the secondary server is provided, but that is limited to content that is not otherwise accessible to that server (see Section 4). Only content that has access controls on the primary server that prevent access by the secondary server can retain confidentiality protection.

Content with different access control policies MUST use different keying material for encryption. This prevents a client with access to one resource from acquiring keys that can be used for resources they are not authorized to access.

Clients that wish to retain control over the confidentiality of responses can omit the out-of-band label from the Accept-Encoding header field on requests, thereby indicating that a direct response is necessary.

7.2. Cross-Origin Access

The content delegation creates the possibility that a primary server could adopt remotely hosted content. On the web, this is normally limited by Cross-Origin Resource Sharing [CORS], which requires that a client first request permission to make a resource accessible to another origin.

This document describes a method whereby content hosted on a remote secondary server can be made accessible to another origin. The content of the out-of-band resource is written into the content of a response from the origin. All an origin needs to make this happen is knowledge of the identity of the out-of-band resource, something that might be difficult based on the guidance in Section 2.2, but not infeasible. A client requests this content using any ambient authority available to it (such as HTTP authentication header fields and cookies).

The simplest option for reducing the ability to steal content in this fashion is to require that the origin demonstrate that it knows the content of the resource. Unfortunately, this demonstration is difficult without imposing significant performance penalties, so we
require a lesser assurance: that the origin knows how to decrypt the content.

This makes content confidentiality (Section 4) mandatory and limits the resources that can be stolen by an origin to those that are already encrypted. Most importantly, only resources for which the origin knows the encryption key can be stolen.

For this protection to be effective, origins MUST use different encryption keys for resources with different sets of authorized recipients. Otherwise, an attacker might learn the encryption key for one resource then use that to decrypt a resource that it is not authorized to read.

Resources that rely on signature-based integrity protection are made only marginally more difficult to steal, since the origin needs to learn the signing public key. However, this is not expected to be difficult, since confidentiality protection for public keys.

Resources that rely on hash-based integrity protection require that the origin learn the hash of the resource.

7.3. Traffic Analysis

Using a secondary server reveals a great deal of information to the secondary server about resources even if confidentiality protection is effective. The size of responses and the pattern of requests for resources can reveal information about their contents. When used carefully, padding as described in [I-D.ietf-httpbis-encryption-encoding] can obscure the length of responses and reduce the information that the secondary server is able to learn.

A random or unpredictable mapping from the primary resource URL on the primary server to the URL of the content is necessary, see Section 2.2.

Length hiding for header fields on responses from the primary server might be more important when an out-of-band encoding is used, since the body of the response becomes less variable.

Making requests for content to multiple different servers can improve the amount of content length information available to network observers. HTTP/2 multiplexing might have otherwise reduced the exposure of length information, but using out-of-band content encoding could expose lengths for those resources that can be distributed by a secondary server. Note that this is not fundamentally worse than HTTP/1.1 in the absence of pipelining.
Padding in HTTP/2 or encrypted content encoding can be used to further obscure lengths.

8. IANA Considerations

This document has no IANA actions.

9. References

9.1. Normative References

[I-D.reschke-http-oob-encoding]

[I-D.thomson-http-mice]


9.2. Informative References


[I-D.ietf-httpbis-encryption-encoding]
[I-D.thomson-http-content-signature]


Appendix A. Acknowledgements

Magnus Westerlund noted the potential for a violation of the cross origin protections offered in browsers.

Authors’ Addresses

Martin Thomson
Mozilla
Email: martin.thomson@gmail.com

Goeran AP Eriksson
Ericsson
Email: goran.ap.eriksson@ericsson.com

Christer Holmberg
Ericsson
Email: christer.holmberg@ericsson.com