

Internet Engineering Task Force
Internet-Draft
Expires: May 16, 2012

D. Balfanz, Ed.
D. Smetters
M. Upadhyay
A. Barth
Google Inc.
November 13, 2011

TLS Origin-Bound Certificates
draft-balfanz-tls-obc-01

Abstract

This document specifies a Transport Layer Security (TLS) extension and associated semantics that allow clients and servers to negotiate the use of origin-bound, self-signed certificates for TLS client authentication.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on May 16, 2012.

Copyright Notice

Copyright (c) 2011 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

Internet-Draft

TLS OBC

November 2011

(<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 BSD License.

Table of Contents

1.	Introduction	3
1.1.	Requirements Language	3
1.2.	Extension Dependencies	3
2.	Origin-Bound Certificates	3
2.1.	Certificate Creation	4
2.1.1.	Origin-Bound Certificate Extension	4
2.2.	Certificate and Key Rotation	5
3.	Changes to The Handshake Message Contents	5
3.1.	Extension Definition	5
3.2.	Client Hello	6
3.3.	Server Hello	6
3.4.	Certificate Request	6
3.5.	Client Certificate	6
4.	Security Considerations	7
4.1.	Third-Party Tracking	7
4.2.	Server Tracking	7
4.3.	Cookie Hardening	7
4.4.	Key Lengths	8
5.	References	8
	Authors' Addresses	8

Internet-Draft

TLS OBC

November 2011

1. Introduction

Transport Layer Security (TLS [[RFC5246](#)]) allows clients to authenticate themselves using Client Certificates. In practice, clients tend to ask for user consent before using Client Certificates. This is to allay privacy concerns about user-identifying information in the Client Certificate, and also to let the user choose among possibly many certificates that can be used by the client for the TLS session.

The user experience of obtaining this consent, along with that of obtaining the certificates in the first place, has traditionally presented a hurdle to user adoption. Additionally, operational constraints on the server side can make it difficult for service providers to switch from a cookie-based authentication scheme to certificate-based TLS client authentication.

The TLS Origin-Bound Certificates extension (TLS-OBC) is a TLS extension [[RFC6066](#)] that allows clients to use certificate-based client authentication without having to obtain user consent before using certificates. A client creates at most one (self-signed) certificate of any given type (e.g., `rsa_sign` or `ecdsa_sign`) per web origin [[WebOrigin](#)], and does not include user-identifying information into such origin-bound certificates, thus making user consent and user-assisted certificate selection unnecessary.

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](#)].

1.2. Extension Dependencies

The client SHOULD use TLS-OBC during a TLS session only if it also uses the TLS Encrypted Client Certificates [[EncryptedClientCerts](#)]

extension during that TLS session.

2. Origin-Bound Certificates

An origin-bound certificate is a self-issued certificate that the client uses for TLS client authentication for a particular web origin [[WebOrigin](#)]. Origin-bound certificates MUST be self-signed, i.e., a private key corresponding to the certified public key MUST be used to sign the certificate.

Clients MUST NOT re-use the same origin-bound certificate for more

than one web origin.

2.1. Certificate Creation

Clients create origin-bound certificates when asked to perform TLS client authentication after negotiating the "ob_cert" extension with a server from that origin (see [Section 3.5](#)), and they do not already have a valid origin-bound certificate for that origin available for use.

Clients SHOULD NOT re-use the same key pair for more than one origin-bound certificate. To do so would violate the privacy properties required by origin-bound certificates.

When creating an origin-bound certificate, it is RECOMMENDED that clients use the PrintableString representation of "anonymous.invalid" as the common name component of the Distinguished Name of both the certificate's Issuer and Subject, with no other name components present.

It is RECOMMENDED that clients pick a lifetime for the new certificate between one month and one year (perhaps depending on an assessment of how well the private key is protected on the host platform). The client SHALL use the notBefore and notAfter fields in the new certificate to record the lifetime of the new certificate.

2.1.1. Origin-Bound Certificate Extension

Origin-bound certificates MUST be X.509 v3 certificates, and MUST

include the origin-bound certificate extension (OID 1.3.6.1.4.1.11129.2.1.6), shown below. This extension MUST be marked critical.

The data of the extension, "OriginBoundCertificate", will be the ASCII Serialization of the certificate's web origin [[WebOrigin](#)] as an IA5String.

```
id-ce-originBoundCertificate OBJECT IDENTIFIER ::=
    { enterprises Google 2 1 6 }

OriginBoundCertificate ::=
    IA5String
```

Figure 1

The server MAY reject the origin-bound certificate if it determines that the web origin included in the "OriginBoundCertificate" extension does not match the server's origin (for example, if it

doesn't match the host name specified in the Server Name Indication TLS extension). It is, however, RECOMMENDED that the server merely report the value of the web origin to the upper layers in the protocol stack, and leave it up to those layers to make use of the web origin value (e.g., to assist developers in debugging the feature).

[2.2.](#) Certificate and Key Rotation

After a client has created an origin-bound certificate for a certain web origin, the client SHOULD re-use the certificate for a period of time, and then discard it.

The client MUST discard existing origin-bound certificate at or before the notAfter time indicated in each certificate.

The client MAY opt to discard an existing origin-bound certificate at any time of its choosing prior to the notAfter time indicated in the certificate. In particular, the client may delete an existing origin-bound certificate in response to a user request to clear privacy-sensitive state from their user-agent, ensuring that a fresh certificate will be generated the next time a user visits that

origin.

The client SHOULD use a new key pair when it generates a new origin-bound certificate.

[3.](#) Changes to The Handshake Message Contents

[3.1.](#) Extension Definition

This document defines a new TLS extension, "ob_cert" (with tentative extension type 13175, or 0x3377), which indicates that client and server agree to allow the client to use an origin-bound certificate to authenticate itself to the server.

To indicate support for origin-bound certificates, the client includes an extension of type "ob_cert" in the Extended Client Hello message. To indicate that a subsequent CertificateRequest message requests an origin-bound certificate, the server includes the same extension in its Extended Server Hello message.

The "ob_cert" TLS extension will be assigned a value from the TLS ExtensionType registry, and will contain zero length "extension_data". For testing, we will use the extension type value of 0x3377. For this type value, the entire encoding of the extension will be 33 77 00 00.

[3.2.](#) Client Hello

A client wishing to indicate support for origin-bound client certificates MUST include the "ob_cert" in the extended Client Hello message to a server from a particular web origin.

[3.3.](#) Server Hello

A server that receives a Client Hello message containing the "ob_cert" extension MAY choose to include the same extension in the extended Server Hello message. If it does, the client and server are considered to have negotiated origin-bound client certificates for the session.

[3.4.](#) Certificate Request

A server MAY choose to send a Certificate Request message to the client. If the session uses origin-bound client certificates, the server SHOULD use an empty "certificate_authorities" list.

A client that receives a Certificate Request during a session for which origin-bound client certificates were negotiated MUST ignore the "certificate_authorities" list.

[3.5.](#) Client Certificate

TLS [[RFC5246](#)] requires that a client that receives a Certificate Request message must send a Client Certificate message in response (which may or may not include client certificates). If client and server negotiated origin-bound client certificates, then the client SHOULD include an origin-bound certificate in the Client Certificate message.

If a client includes an origin-bound certificate in the Client Certificate message during a session for which origin-bound client certificates were negotiated, the included certificate MUST

- o be an origin-bound certificate for the web origin of the server, and
- o match a certificate type requested by the server in the Certificate Request message.

If a client doesn't possess a certificate meeting the above requirements, it SHOULD self-issue a certificate of a type requested by the server in the Certificate Request message and include the newly-generated certificate in the Client Certificate message. Only if the client is not capable of providing a certificate of the

requested type SHOULD it include an empty "certificate_list" in the Client Certificate message.

A server verifying a Client Certificate message in a handshake that negotiated origin-bound client certificates MUST verify that the certificate is self-signed, rather than being signed by any particular CA. The server MUST verify that the public key in the certificate corresponds to the key used to authenticate the client in

the handshake. The server SHOULD ignore the Issuer and Subject in the presented certificate. The server MUST ignore the notBefore and notAfter fields in the certificate.

Finally, the server MUST verify that the certificate contains the origin-bound extension, and MAY verify that the origin identified in that certificate matches the server (see [Section 2.1.1](#)).

[4.](#) Security Considerations

[4.1.](#) Third-Party Tracking

A third party might be able to track a client across multiple sessions by observing the use of the same origin-bound certificate. To alleviate this concern, clients SHOULD also implement the Encrypted Client Certificates Extension [[EncryptedClientCerts](#)], and use it concurrently with TLS-OBC. The Encrypted Client Certificates Extension [[EncryptedClientCerts](#)] ensures that the origin-bound certificate is sent after TLS established secrecy on the channel between client and server.

[4.2.](#) Server Tracking

A client can prevent server tracking by deleting the origin-bound certificate for the server's Web origin. This could happen, for example, when the user elects to remove all cookies for that origin.

[4.3.](#) Cookie Hardening

One way TLS-OBC can be used to strengthen cookie-based authentication is by "binding" cookies to an origin-bound certificate. The server, when issuing a cookie for an HTTP session, would associate the client's origin-bound certificate with the session (either by encoding information about the certificate unforgeably in the cookie, or by associating the certificate with the cookie's session through some other means). That way, if and when a cookie gets stolen from a client, it cannot be used over a TLS connection initiated by a different client - the cookie thief would also have to steal the private key associated with the client's origin-bound certificate, a

task considerably harder especially when we assume the existence of a

Trusted Platform Module or other Secure Element that can store the origin-bound-certificate's private key.

[4.4.](#) Key Lengths

If the client uses RSA key pairs, the key length should be at least 1024 bits. For longer-lived origin-bound certificates, the key length should be 2048 bits.

If the client uses ECDSA key pairs, the key length should be at least 160 bits.

[5.](#) References

[EncryptedClientCerts]

Langley, A., "TLS Encrypted Client Certificates", <<http://tools.ietf.org/html/draft-agl-tls-encryptedclientcerts>>.

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

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

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

[WebOrigin]

Barth, A., "The Web Origin Concept", <<http://tools.ietf.org/html/draft-abarth-origin-09>>.

Authors' Addresses

Dirk Balfanz (editor)
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA
USA

Phone:
Email: balfanz@google.com

Internet-Draft

TLS OBC

November 2011

D K Smetters
Google Inc.

Mayank Upadhyay
Google Inc.

Adam Barth
Google Inc.

