Internet Engineering Task Force	D.B. Balfanz, Ed.
Internet-Draft	D.S. Smetters
Expires: January 27, 2012	M.U. Upadhyay
	A.B. Barth
	Google Inc.
	July 26, 2011

TLS Origin-Bound Certificates
draft-balfanz-tls-obc-00

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

Copyright Notice

Table of Contents

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 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-MessageReordering *[CertMessageReordering]* 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 1.3.6.1.4.111129.2.1.6), shown below. This extension MUST be marked critical.
The data of the extension will be the ASCII Serialization *[WebOrigin]* of the certificate's web origin *[WebOrigin]* as an IA5String.

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

 OriginBoundCertificate ::=
        IA5String
```

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 0xff0f), 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 request client authentication with an origin-bound certificate, the server includes the same extension in 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 0xff0f. For this type value, the entire encoding of the extension will be ff 0f 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 MAY 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 a certificate in the Client Certificate message. If the certificate_authorities list in the Certificate Request is empty, that certificate SHOULD be an origin-bound certificate, and it should be selected by the client without any assistance or approval by the end-user. If the certificate_authorities list in the Certificate Request is not empty, the client MAY select a regular certificate for inclusion in the Client Certificate message, and MAY involve the end-user in the certificate selection process.
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

- be an origin-bound certificate for the web origin of the server, and
- 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 SHOULD verify that the origin identified in that certificate matches the server. In particular, the scheme MUST be https. The port MUST be the server port to which the client connected for the TLS connection, and the host name MUST be a valid host name for the server. If the client uses SNI, then the hostname in the origin-bound certificate must match the hostname specified in the SNI extension.

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 CertificateMessage Reordering Extension *[CertMessageReordering]*, and use it concurrently with TLS-OBC. The CertificateMessage Reordering Extension *[CertMessageReordering]*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 tacking 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 different TLS connection - 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.

5. References

, "

[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.B., "The Web Origin Concept", .
[CertMessageReordering]	TODO: Reference for ClientCertificate message reordering", .

Authors' Addresses

Dirk Balfanz editor Balfanz Google Inc. 1600 Ampitheatre Parkway Mountain View, CA USA EMail: balfanz@google.com

D K Smetters Smetters Google Inc.

Mayank Upadhyay Upadhyay Google Inc.

Adam Barth Barth Google Inc.