TLS extension for Proxies to transfer Server certificate
draft-babu-serv-cert-trans-from-proxy-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of 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/1id-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 January 10, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

Intercepting transparent proxies splice the client-Server connection into two connections: Client-Proxy connection, Proxy-server connection. On Client-Proxy connection, proxy sends it’s certificate to the client. As client is generally (in such a scenario) pre-configured to accept proxy’s certificate, client accepts and proceeds further with the connection. On Proxy-Server connection, server sends its certificate to the proxy. Proxy typically doesn’t possess the information (like MX domain name in case of SMTP) required to validate the certificate. The certificate validation is at times very complex & hence it is better to offload this responsibility to the original client itself.

This document addresses this issue by extending TLS to let proxy send server’s certificate to the client for validation and suggests how client can indicate certificate validation result to the proxy.

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].
Table of Contents

1. Introduction ............................................. 2
2. Mechanism Overview ...................................... 2
3. Need Server certificate extension ....................... 2
4. Handshake message to transfer server certificate .... 3
5. various scenarios ........................................ 4
6. Security Considerations ................................. 6
7. Normative References ..................................... 6
Author’s Address ............................................. 6
Intellectual Property and Copyright Statements ............ 7

1. Introduction

Today, intercepting transparent proxies are very common in applications (say SMTP, HTTP) using [TLS]. In SMTP, these intercepting proxies may provide functionality like anti-virus scanning, anti-spam scanning. HTTP intercepting proxies may provide functionality like anti-virus scanning, URL filtering etc.

Client ---- Transparent Proxy -------- Server

This document defines a way for proxy to send original server’s certificate to the original client and suggests how client can indicate certificate validation result to the proxy. The mechanism makes use of TLS extension framework defined in [RFC4366] and defines a new TLS handshake message type.

The clients supporting this extension receive certificates of intercepting proxy (if interception happens) as well as the original server. So clients should be capable of handling validations on both the certificates.

2. Mechanism Overview

This extension defines
- A new extension type (need_server_certificate) for extended client hellos defined in [RFC4366].
- A new handshake message (Orig_server_certificate)

3. Need Server certificate extension

Who should send this extension?
- A client which is configured to request the original server certificate for validation includes an extension of type "need_server_certificate" in (extended) client hello.
- It is possible that there could be more than one proxy between client and server:

Client --- P1 --- P2 ------ Server

In such a scenario, P1 also includes "need_server_certificate" in (extended) client hello in its connection to P2, unless it has the knowledge that it is the last proxy between client and server. If a proxy is configured that it is the edge proxy in client’s trust domain, then it need not send this extension.

How should a receiver respond to this?
- If a proxy intercepts the connection, it SHOULD respond back to
the client with "need_server_certificate" extension.
- When there are no intercepting proxies, a server receives this extension. A server which understands this extension should ignore this. It is not clear from [RFC4366] what a server does when it receives an extension which it doesn’t understand. This item is TBD.
- If a proxy which doesn’t have the capability to validate server certificate or is configured to offload this responsibility to the original client doesn’t receive "need_server_certificate" extension, it should return a fatal error like handshake failure or insufficient security (TBD).

When a proxy responds with "need_server_certificate" extension to the client, proxy MUST send the its certificate as well as the original server certificate to the client (discussed in section 4).

How should client handle ServerHello?
- If the client receives "need_server_certificate" extension in ServerHello, it MUST expect the nexthop proxy certificate as well as the original server certificate. Client MUST perform validations on both proxy certificate as well as original server certificate. If a client doesn’t receive server certificate, it MUST abort the connection.
- If the client doesn’t receive "need_server_certificate" extension in ServerHello, client MUST assume that there is no proxy in between and MUST perform server certificate validations on the received certificate.

The "extension_data" field of this extension in both clientHello as well as ServerHello SHALL be empty.

Note on backward compatibility: Suppose a client supports this extension, but a intercepting proxy or the actual server doesn’t understand extended hello or "need_server_certificate", client MUST proceed with the connection and MUST perform server certificate validations on the received certificate. By validating this way, client can deny connections from any proxies (because certificate validation fails) which do not support this mechanism, but still accept connections from server which do not support this.

4. Handshake message to transfer server certificate

This document suggests the use of a new handshake message, "orig_server_certificate" to transfer the original server’s certificate to the client. The new handshake message structure therefore becomes:

```c
enum {
    hello_request(0),
    client_hello(1),
    server_hello(2),
    certificate(11),
    server_key_exchange (12),
    certificate_request(13),
    server_hello_done(14),
    certificate_verify(15),
    client_key_exchange(16),
    finished(20),
    certificate_url(21),
    certificate_status(22),
    orig_server_certificate(23),
    (255)
} HandshakeType;

struct {
    HandshakeType msg_type;  /* handshake type */
    uint24 length;           /* bytes in message */
```
select (HandshakeType) {
    case hello_request:       HelloRequest;
    case client_hello:        ClientHello;
    case server_hello:        ServerHello;
    case certificate:         Certificate;
    case server_key_exchange: ServerKeyExchange;
    case certificate_request: CertificateRequest;
    case server_hello_done:   ServerHelloDone;
    case certificate_verify:  CertificateVerify;
    case client_key_exchange: ClientKeyExchange;
    case finished:            Finished;
    case certificate_url:     CertificateURL;
    case certificate_status:  CertificateStatus;
    case orig_server_certificate: Certificate; /*new*/
} body;
} Handshake;

The structure of Certificate is defined in [RFC4346].

If proxy responded to the client with "need_server_certificate" extension, this message MUST be sent immediately after the "Certificate" handshake message in Client-Proxy connection.

The client MUST perform validations on the received proxy certificate as well as the server certificate. If either proxy or server certificate is not valid, client should respond with certificate related error messages defined in [RFC4346]. On reception of such an error, proxy MUST close the Proxy-Server connection.

It should be noted that proxy transparency is lost at TLS layer due to the fact that client is sent both proxy as well as original server certificate for validation. Though transparency is not possible at TLS layer, application protocols can still remain transparent to the proxy operation.

5. various scenarios

Client, Proxy understand this extension.

<table>
<thead>
<tr>
<th></th>
<th>Proxy</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientHelo</td>
<td>ProxyHelo (with &quot;need_orig_server&quot;) --&gt;</td>
<td></td>
</tr>
</tbody>
</table>
|           | ClientHelo (with "need_orig_server")--->
|           | ServerHelo (without "need_orig_server")
|           | Certificate
|           | ServerKeyExchange <-- ServerHelloDone
|           | ServerHelo (with "need_orig_server")
|           | Certificate
|           | orig_server_certificate
|           | ServerKeyExchange <-- ServerHelloDone
|           | ClientKeyExchange
CertificateVerify
[ChangeCipherSpec]  -->
[Finished]  <-- [ChangeCipherSpec]
[Finished]  -->
[ChangeCipherSpec]  <-- [Finished]
[Finished]

Client understands this extension. Proxy doesn’t. In this case, certificate validation fails on the received proxy certificate.

<table>
<thead>
<tr>
<th>Client</th>
<th>Proxy</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientHelo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(with</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;need_orig_server&quot;)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ServerHelo</td>
<td>(without &quot;need_orig_server&quot;)</td>
<td></td>
</tr>
<tr>
<td>Certificate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>orig_server_certificate</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ServerkeyExchange</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;-- ServerHelloDone</td>
<td></td>
</tr>
<tr>
<td>Certificate error</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Client doesn’t understand this extension, but proxy is configured to offload original server certificate responsibility to the original client:

<table>
<thead>
<tr>
<th>Client</th>
<th>Proxy</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientHelo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(with</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;need_orig_server&quot;)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fatal Error</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TBD: Two proxies between client and server. Client as well as two proxies understand this extension.

6. IANA Considerations

This document (if approved) requests IANA to allocate "need-server_certificate" TLS extension and "Orig_server_certificate" handhsake message.

Note to RFC Editor: this section may be removed on publication as an RFC.

7. Security Considerations

Though this extension equips clients with an ability to validate original server certificate as well as its nexthop, it doesn’t provide a mechanism to transmit certificates of any proxies between the first proxy and the original server. It is assumed that client trusts the first proxy to either not allow any other proxies in
between or to allow only a proxy which is in the trusted domain.

8. Normative References


Author’s Address

Babu Neelam
Intoto Software India Private Ltd.
8-3-1111/2, kesava nagar colony,
sriniagar colony main road,
punjagutta,
Hyderabad,
India.

Email: babun@intoto.com

Comments are solicited and should be addressed to the working group’s mailing list at ietf-http-wg@w3.org and/or the author(s).

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this
specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).