| | |
|---|---|
| IETF | P. Wouters |
| Internet-Draft | Xelerance |
| Intended status: Standards Track | J. Gilmore |
| Expires: May 20, 2012 | S. Weiler |
| | SPARTA, Inc. |
| | T. Kivinen |
| | AuthenTec |
| | H. Tschofenig |
| | Nokia Siemens Networks |
| | November 17, 2011 |

# TLS out-of-band public key validation
draft-wouters-tls-oob-pubkey-02

## Abstract

This document specifies a new TLS certificate type for exchanging raw public keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) for use with out-of-band authentication. Currently, TLS authentication can only occur via PKIX or OpenPGP certificates. By specifying a minimum resource for raw public key exchange, implementations can use alternative authentication methods. One such method is using DANE Resource Records secured by DNSSEC, Another use case is to provide authentication functionality when used with devices in a constrained environment that use whitelists and blacklists, as is the case with sensors and other embedded devices that are constrained by memory, computational, and communication limitations where the usage of PKIX is not feasible.

The new certificate type specified can also be used to reduce the latency of a TLS client that is already in possession of a validated public key of the TLS server before it starts a (non-resumed) TLS handshake.

## Status of this Memo

## Copyright Notice

## Table of Contents

## 1. Introduction

### 1.1. Motivation

Traditionally, TLS server public keys are obtained in PKIX containers in-band using the TLS connection and validated using trust anchors based on a [PKIX] certification authority (CA). This method can add a complicated trust relationship that is difficult to validate. Examples of such complexity can be seen in [Defeating-SSL].

Alternative methods are available that allow a TLS client to obtain the TLS server public key:

*The TLS server public key is obtained from a DNSSEC secured RRset using [DANE]

*The TLS server public key is obtained from a [PKIX] certificate chain from an [LDAP] server

*The TLS server public key is provisioned by the operating system and updated via software updates

*A TLS client has connected to the TLS server before and has cached the TLS server certificate chain or TLS server public key for re-use

[RFC5246] does not provide a mechanism for a TLS client to tell the TLS server it is already in possession of the authenticated public key. Therefore, a TLS server must always send a list of trusted CA keys and its EE certificate containing its public key, even when the TLS client does not require or desire that data for authentication.

[RFC6066] allows suppression of the certificate trust anchor chain, but not suppression of the PKIX EE certificate container. These certificate chains are large opaque blocks of data containing much more than the public key of the TLS server. Since the TLS client might only be able to validate the PKIX SubjectPublicKeyInfo via an out-of-band method such as [DANE], it has to ignore any additional information received that was sent by the server that it could not validate. Furthermore, information that comes in via these certificate chains could contain contradicting or additional information that the TLS client cannot validate or trust, such as an expiry date that conflicts with information obtained from DNS or LDAP. This document specifies a method to suppress sending this additional information.

Some small embedded devices use the UDP based [CoAP], a specialized constrained networks and nodes for machine-to-machine applications. These devices interact with a Web server to upload data such as temperature sensor readings at a regular intervals. Constrained Application Protocol (CoAP) [CoAP] can utilize DTLS for its communication security. As part of the provisioning procedure, the embeded device is configured with the address and public key of a dedicated CoAP server to upload sensor data. Receiving PKIX information [PKIX] from a webserver would be an unneccesarry burden on a sensor networking deployment environment that requires pre-configured client-server public keys. These devices often also lack a real-time clock to perform any PKIX epixry checks.

### 1.2. Applicability

The Transport Layer Security (TLS) Protocol Version 1.2 is specified in [RFC5246] and provides a framework for extensions to TLS as well as considerations for designing such extensions. [RFC6066] defines several new TLS extensions. This document extends the specifications of those RFCs with one new TLS Certificate Type to facilitate suppressing unneeded [PKIX] information from being sent during the TLS handshake when this information is not required to authenticate the TLS server.

### 1.3. Terminology

Most security-related terms in this document are to be understood in the sense defined in [SECTERMS]; such terms include, but are not limited to, "attack", "authentication", "authorization", "certification authority", "certification path", "certificate", "credential", "identity", "self-signed certificate", "trust", "trust anchor", "trust chain", "validate", and "verify".
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]*.

## 2. Changes to the Handshake Message Contents

This section describes the changes to the TLS handshake message contents when raw public keys are to be used for authentication. Figure 1 illustrates the exchange of messages as described in the sub-sections below. The new "RawPublicKey" value in the cert_type extension indicates the ability and desire to exchange raw public keys, which are then exchanged as part of the certificate payloads.

```
 client_hello,
 cert_type="RawPublicKey" ->

                           <-  server_hello,
                               cert_type="RawPublicKey",
                               certificate,
                               server_key_exchange,
                               certificate_request,
                               server_hello_done

 certificate,
 client_key_exchange,
 certificate_verify,
 change_cipher_spec,
 finished                  ->

                           <- change_cipher_spec,
                              finished

Application Data        <------->     Application Data
```

### 2.1. Client Hello

In order to indicate the support of out-of-bound raw public keys, clients MUST include an extension of type "cert_type" to the extended client hello message. The "cert_type" TLS extension, which is defined in [RFC6091], is assigned the value of 9 from the TLS ExtensionType registry. This value is used as the extension number for the extensions in both the client hello message and the server hello message. The hello extension mechanism is described in [RFC5246].
The "cert_type" TLS extension carries a list of supported certificate types the client can use, sorted by client preference. This extension MUST be omitted if the client only supports X.509 certificates. The "extension_data" field of this extension contains a CertificateTypeExtension structure. Note that the CertificateTypeExtension structure is being used both by the client and the server, even though the structure is only specified once in this document.

```
enum { client, server } ClientOrServerExtension;

enum { X.509(0), OpenPGP(1),
   RawPublicKey([TBD]),
   (255) } CertificateType;

struct {
   select(ClientOrServerExtension)
       case client:
         CertificateType certificate_types<1..2^8-1>;
       case server:
         CertificateType certificate_type;
   }
} CertificateTypeExtension;
```

The [RFC6091] defined CertificateTypeExtension is extended as follows: No new cipher suites are required to use raw public keys. All existing cipher suites that support a key exchange method compatible with the defined extension can be used.

## 2.2. Server Hello

If the server receives a client hello that contains the "cert_type" extension and chooses a cipher suite then two outcomes are possible. The server MUST either select a certificate type from the certificate_types field in the extended client hello or terminate the session with a fatal alert of type "unsupported_certificate".
The certificate type selected by the server is encoded in a CertificateTypeExtension structure, which is included in the extended server hello message using an extension of type "cert_type". Servers that only support X.509 certificates MAY omit including the "cert_type" extension in the extended server hello.
If the negotiated certificate type is RawPublicKey the TLS server MUST send a CertificateTypeExtension structure with a PKIX [PKIX] certificate containing ONLY the SubjectPublicKeyInfo. The public key MUST match the selected key exchange algorithm.

## 2.3. Certificate Request

The semantics of this message remain the same as in the TLS specification. However, if this message is sent, and the negotiated certificate type is RawPublicKey, the "certificate_authorities" list MUST be empty.

## 2.4. Other Handshake Messages

All the other handshake messages are identical to the TLS specification.

## 3. Security Considerations

The TLS cert_type extension defined here lets a TLS client attempt to supress the sending of server certificate as well as the certification chain for that certificate.
A client using this cert_type needs to be confident in the authenticity of the public key it is using. Since those public keys were obtained out-of-band extension), the authentication must also be out-of-band. Depending on exactly how the public keys were obtained, it may be appropriate to use authentication mechanisms tied to the public key transport. For example, if public keys were obtained using [DANE] it is appropriate to use DNSSEC to authenticate the public keys.

## 4. IANA Considerations

We request that IANA assign a TLS cert_type value for RawPublicKey.

## 5. Contributors

The following individuals made important contributions to this document: Paul Hoffman.

## 6. Acknowledgements

This document is based on material from RFC 6066 for which the author is Donald Eastlake 3rd. Contributions to that document also include Joseph Salowey, Alexey Melnikov, Peter Saint-Andre, and Adrian Farrel. The second version of this document was made after feedback from the TLS working group at IETF#81. The support for hashes of public keys has been removed after the discussions at the IETF#82 meeting.

## 7. References

### 7.1. Normative 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. |
| **[PKIX]** | Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R. and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008. |
| **[SECTERMS]** | Shirey, R., "Internet Security Glossary, Version 2", RFC 4949, August 2007. |

### 7.2. Informative References

| | |
|---|---|
| **[RFC6066]** | |

| | |
|---|---|
| | Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011. |
| **[RFC6091]** | Mavrogiannopoulos, N. and D. Gillmor, "Using OpenPGP Keys for Transport Layer Security (TLS) Authentication", RFC 6091, February 2011. |
| **[LDAP]** | Sermersheim, J., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006. |
| **[DANE]** | Hoffman, P and J Schlyter, "Using Secure DNS to Associate Certificates with Domain Names For TLS", Internet-Draft draft-ietf-dane-protocol-12, September 2011. |
| **[CoAP]** | Shelby, Z, Hartke, K, Bormann, C and B Frank, "Constrained Application Protocol", Internet-Draft draft-ietf-core-coap-07, July 2011. |
| **[Defeating-SSL]** | Marlinspike, M., "New Tricks for Defeating SSL in Practice", February 2009. |

## Authors' Addresses

Paul Wouters Wouters Xelerance 4130 Ramsayville Road Ottawa, On K1G 3N4 Canada Phone: +1-647-722-5653 EMail: paul@xelerance.com URI: https://www.xelerance.com/

John Gilmore Gilmore PO Box 170608 San Francisco, California 94117 USA Phone: +1 415 221 6524 EMail: gnu@toad.com URI: https://www.toad.com/

Samuel Weiler Weiler SPARTA, Inc. 7110 Samuel Morse Drive Columbia, Maryland, 21046 US EMail: weiler@tislabs.com

Tero Kivinen Kivinen AuthenTec Eerikinkatu 28 HELSINKI, FI-00180 FI EMail: kivinen@iki.fi

Hannes Tschofenig Tschofenig Nokia Siemens Networks Linnoitustie 6 Espoo, 02600 Finland Phone: +358 (50) 4871445 EMail: Hannes.Tschofenig@gmx.net URI: http://www.tschofenig.priv.at