TLS P. Wouters
Internet-Draft No Hats Corporation
Intended status: Standards Track J. Gilmore
Expires: September 12, 2012
S. Weiler
SPARTA, Inc.
T. Kivinen
AuthenTec
H. Tschofenig
Nokia Siemens Networks
March 11, 2012

TLS Out-of-Band Public Key Validation
draft-ietf-tls-oob-pubkey-02.txt

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 public key validation. Currently, TLS authentication can only occur via X.509-based Public Key Infrastructure (PKI) or OpenPGP certificates. By specifying a minimum resource for raw public key exchange, implementations can use alternative public key validation methods.

One such alternative public key valiation method is offered by the DNS-Based Authentication of Named Entities (DANE) together with DNS Security. Another alternative is to utilize pre-configured keys, as is the case with sensors and other embedded devices. The usage of raw public keys, instead of X.509-based certificates, leads to a smaller code footprint.

The support for raw public keys is introduced into TLS via a new non-PKIX certificate type.

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 September 12, 2012.

Table of Contents

1. Introduction

Traditionally, TLS server public keys are obtained in PKIX containers in-band using the TLS handshake 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:

- o The TLS server public key is obtained from a DNSSEC secured resource records using DANE [I-D.ietf-dane-protocol].
- o The TLS server public key is obtained from a [PKIX] certificate chain from an Lightweight Directory Access Protocol (LDAP) [LDAP] server.
- o The TLS client and server public key is provisioned into the operating system firmware image, and updated via software updates.

Some smart objects use the UDP-based Constrained Application Protocol (CoAP) [I-D.ietf-core-coap] to interact with a Web server to upload sensor data at a regular intervals, such as temperature readings. CoAP [I-D.ietf-core-coap] can utilize DTLS for securing the client-to-server communication. As part of the manufacturing process, the embeded device may be configured with the address and the public key of a dedicated CoAP server, as well as a public key for the client itself. The usage of X.509-based PKIX certificates [PKIX] may not suit all smart object deployments and would therefore be an unneccesarry burden.

The Transport Layer Security (TLS) Protocol Version 1.2 [RFC5246] provides a framework for extensions to TLS as well as guidelines for designing such extensions. This document uses the TLS Certificate Type extension point to define a new non-X.509 certificate type for carrying raw public keys.

2. Terminology

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].

3. TLS Handshake Extension

This section describes the changes to the TLS handshake message contents when raw public key certificates are to be used. 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. Note that the certificate payloads only contain the SubjectPublicKeyInfo structure instead of the entire certificate.

```
 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
```

Figure 1: Example Message Flow

3.1. Client Hello

In order to indicate the support of out-of-band 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.

The [RFC6091] defined CertificateTypeExtension is extended as follows:

```
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;
```

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.

3.2. 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 CertificateType 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 place the SubjectPublicKeyInfo structure into the Certificate payload. The public key MUST match the selected key exchange algorithm.

3.3. Certificate Request

The semantics of this message remain the same as in the TLS specification.

3.4. Other Handshake Messages

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

4. Security Considerations

The transmission of raw public keys, as described in this document, provides benefits by lowering the over-the-air transmission overhead since raw public keys are quite naturally smaller than an entire certificate. There are also advantages from a codesize point of view for parsing and processing these keys. The crytographic procedures for assocating the public key with the possession of a private key also follows standard procedures.

The main security challenge is, however, how to associate the public key with a specific entity. This information will be needed to make authorization decisions. Without a secure binding, man-in-the-middle attacks may be the consequence. This document assumes that such binding can be made out-of-band and we list a few examples in Section 1. DANE [I-D.ietf-dane-protocol] offers one such approach. If public keys are obtained using DANE, these public keys are authenticated via DNSSEC. Pre-configured keys is another out of band method for authenticating raw public keys. While pre-configured keys are not suitable for a generic Web-based e-commerce environment such keys are a reasonable approach for many smart object deployments where there is a close relationship between the software running on the device and the server-side communication endpoint. Regardless of the chosen mechanism for out-of-band public key validation an assessment of the most suitable approach has to be made prior to the start of a deployment to ensure the security of the system.

5. IANA Considerations

This document requests IANA to assign a TLS cert_type value for RawPublicKey. The cert_type registry is established with [RFC6091].

6. Contributors

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

7. Acknowledgements

The feedback from the TLS working group meeting at IETF#81 has substantially shaped the document and we would like to thank the meeting participants for their input. The support for hashes of public keys has been moved to [I-D.ietf-tls-cached-info] after the discussions at the IETF#82 meeting and the feedback from Eric Rescorla.

We would like to thank Martin Rex, Bill Frantz, Zach Shelby, Carsten Bormann, Cullen Jennings, Rene Struik, Alper Yegin, and Jim Schaad.

8. References

8.1. Normative References

[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.

[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.

8.2. Informative References

[Defeating-SSL]
Marlinspike, M., "New Tricks for Defeating SSL in Practice", February 2009, <http://www.blackhat.com/presentations/bh-dc-09/Marlinspike/BlackHat-DC-09-Marlinspike-Defeating-SSL.pdf>.

[I-D.ietf-core-coap]
Frank, B., Bormann, C., Hartke, K., and Z. Shelby, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-08 (work in progress), October 2011.

[I-D.ietf-dane-protocol]

Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Protocol for Transport Layer Security (TLS)", draft-ietf-dane-protocol-18 (work in progress), March 2012.

[I-D.ietf-tls-cached-info]
Santesson, S. and H. Tschofenig, "Transport Layer Security (TLS) Cached Information Extension", draft-ietf-tls-cached-info-11 (work in progress), December 2011.

[LDAP] Sermersheim, J., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006.

[RFC6091] Mavrogiannopoulos, N. and D. Gillmor, "Using OpenPGP Keys for Transport Layer Security (TLS) Authentication", RFC 6091, February 2011.

Authors' Addresses

Paul Wouters
No Hats Corporation

Email: paul@nohats.ca

John 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
SPARTA, Inc.
7110 Samuel Morse Drive
Columbia, Maryland 21046
US

Email: weiler@tislabs.com

Tero Kivinen
AuthenTec
Eerikinkatu 28
HELSINKI FI-00180
FI

Email: kivinen@iki.fi

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at