

Internet Engineering Task Force
Internet-Draft
Updates: 6698, 7671 (if approved)
Intended status: Standards Track
Expires: 18 August 2022

S. Huque
Salesforce
V. Dukhovni
Two Sigma
A. Wilson
Valimail
14 February 2022

TLS Client Authentication via DANE TLSA records
draft-huque-dane-client-cert-08

Abstract

The DANE TLSA protocol [RFC6698] [RFC7671] describes how to publish Transport Layer Security (TLS) server certificates or public keys in the DNS. This document updates RFC 6698 and RFC 7671. It describes how to additionally use the TLSA record to publish client certificates or public keys, and also the rules and considerations for using them with TLS.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on 18 August 2022.

Copyright Notice

Copyright (c) 2022 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 (<https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

| | |
|--|---|
| 1. Introduction and Motivation | 2 |
| 2. Associating Client Identities in TLSA Records | 3 |
| 2.1. Format 1: Service specific client identity | 3 |
| 2.2. Format 2: DevId: IOT Device Identity | 3 |
| 3. Authentication Model | 3 |
| 4. Client Identifiers in X.509 certificates | 4 |
| 5. Signaling the Client's DANE Identity in TLS | 4 |
| 6. Example TLSA records for clients | 5 |
| 6.1. Format 1: Service Specific Client Identity | 5 |
| 6.2. Format 2: DevID | 5 |
| 7. Changes to Client and Server behavior | 5 |
| 8. Raw Public Keys | 7 |
| 9. Acknowledgements | 7 |
| 10. IANA Considerations | 7 |
| 11. Security Considerations | 7 |
| 12. References | 7 |
| 12.1. Normative References | 7 |
| 12.2. Informative References | 8 |
| Authors' Addresses | 8 |

1. Introduction and Motivation

The Transport Layer Security (TLS) protocol [RFC5246] [RFC8446] optionally supports the authentication of clients using X.509 certificates [RFC5280] or raw public keys [RFC7250]. TLS applications that perform DANE authentication of servers using TLSA records may also desire to authenticate clients using the same mechanism, especially if the client identity is in the form of or can be represented by a DNS domain name. Some design patterns from the Internet of Things (IoT) plan to make use of this form of authentication, where large networks of physical objects identified by DNS names may authenticate themselves using TLS to centralized device management and control platforms.

In this document, the term TLS is used generically to describe both the TLS and DTLS (Datagram Transport Layer Security) [RFC6347] protocols.

2. Associating Client Identities in TLSA Records

Different applications may have quite different conventions for naming clients via domain names. This document thus does not proscribe a single format, but mentions a few that may have wide applicability.

2.1. Format 1: Service specific client identity

In this format, the owner name of the client TLSA record has the following structure:

```
_service.[client-domain-name]
```

The first label identifies the application service name. The remaining labels are composed of the client domain name.

Encoding the application service name into the owner name allows the same client domain name to have different authentication credentials for different application services. There is no need to encode the transport label - the same name form is usable with both TLS and DTLS.

The `_service` label could be a custom string for an application, but more commonly is expected to be a service name registered in the IANA Service Name Registry [SRVREG].

The RDATA or data field portion of the TLSA record is formed exactly as specified in RFC 6698 and RFC 7671, and carries the same meaning.

2.2. Format 2: DevId: IOT Device Identity

The DevID form of the TLSA record has the following structure:

```
[devicename]._device.[org-domain-name]
```

It is loosely based on the proposed PKI Certificate Identifier Format for Devices [CERTDEVID], but is simpler in form. It makes no distinction between manufacturer issued and locally issued certificates, and does away with the "serial" and "type" labels. The `"_device"` label that precedes the organization domain name allows all the device identities to be delegated to a subzone or to another party.

3. Authentication Model

The authentication model assumed in this document is the following:

The client is assigned an identity corresponding to a DNS domain name. This domain name doesn't necessarily have any relation to its network layer addresses. Clients often have dynamic or unpredictable addresses, and may move around the network, so tying their identity to network addresses is not feasible or wise in the general case.

The client generates (or has generated for it) a private and public key pair. Where client certificates are being used, the client also has a certificate binding the name to its public key. The certificate or public key has a corresponding TLSA record published in the DNS, which allows it to be authenticated directly via the DNS (using the DANE-TA or DANE-EE certificate usage modes) or via a PKIX public CA system constraint if the client's certificate was issued by a public CA (using the PKIX-TA or PKIX-EE DANE usage modes).

4. Client Identifiers in X.509 certificates

If the TLS DANE Client Identity extension (see Section 5) is not being used, the client certificate **MUST** have the client's DNS name specified in the Subject Alternative Name extension's `dNSName` type.

If the TLS DANE Client Identity extension is in use, then with DANE-EE(3), the subject name need not be present in the certificate.

5. Signaling the Client's DANE Identity in TLS

The client **SHOULD** explicitly signal that it has a DANE identity. The most important reason is that the server may want an explicit indication from the client that it has a DANE record, so as to avoid unnecessary DNS queries in-band with the TLS handshake.

The DANE Client Identity TLS extension [TLSCLIENTID] is used for this purpose. This extension can also be used to convey the actual DANE client identity (i.e. domain name) that the TLS server should attempt to authenticate. This is required when using TLS raw public key authentication, since there is no client certificate from which to extract the client's DNS identity. It is also helpful when the client certificate contains multiple identities, and only a specific one has a DANE record.

An additional case where such client signaling is helpful, is one where DANE client authentication is optional, and there is a population of buggy client software that does not react gracefully to receipt of a Certificate Request message from the TLS server. This extension allows TLS servers to deal with this situation by selectively sending a Certificate Request message only to clients that have sent this extension.

6. Example TLSA records for clients

The following examples are provided in the textual presentation format of the TLSA record.

6.1. Format 1: Service Specific Client Identity

An example TLSA record for the client "device1.example.com." and the application "smtp-client". This record specifies the SHA-256 hash of the subject public key component of the end-entity certificate corresponding to the client. The certificate usage for this record is 3 (DANE-EE) and thus is validated in accordance with section 5.1 of RFC 7671.

```
_smtp-client.device1.example.com. IN TLSA (  
  3 1 1 d2abde240d7cd3ee6b4b28c54df034b9  
        7983ald16e8a410e4561cb106618e971 )
```

6.2. Format 2: DevID

An example TLSA record for the device named "sensor7" managed by the organization "example.com" This record specifies the SHA-512 hash of the subject public key component of an EE certificate corresponding to the client.

```
sensor7._device.example.com. IN TLSA (  
  3 1 2 0f8b48ff5fd94117f21b6550aaee89c8  
        d8adbc3f433c8e587a85a14e54667b25  
        f4dcd8c4ae6162121ea9166984831b57  
        b408534451fd1b9702f8de0532ecd03c )
```

7. Changes to Client and Server behavior

A TLS Client conforming to this specification MUST have a signed DNS TLSA record published corresponding to its DNS name and X.509 certificate or public key. The client presents this certificate or public key in the TLS handshake with the server. The client should not offer ciphersuites that are incompatible with its certificate or public key. If the client's certificate has a DANE record with a certificate usage other than DANE-EE, then the presented client certificate MUST have the client's DNS name specified in the Subject Alternative Name extension's dNSName type.

Additionally, when using raw public key authentication, the client MUST send the TLS DANE Client Identity extension [TLSCLIENTID] in its Client Hello message. When using X.509 certificate authentication, it SHOULD send this extension.

A TLS Server implementing this specification performs the following steps:

- * Request a client certificate in the TLS handshake (the "Client Certificate Request" message). This could be done unconditionally, or only when it receives the TLS DANE Client Identity extension from the client.
- * If the client has sent a non-empty DANE Client Identity extension, then extract the client's domain name from the extension. Otherwise, extract the client identity from the Subject Alternative Name extension's `dNSName` type.
- * Construct the DNS query name for the corresponding TLSA record. If the TLS DANE client identity extension was present, then this name should be used. Otherwise, identities from the client certificate are used.
- * Look up the TLSA record set in the DNS. The response MUST be cryptographically validated using DNSSEC. The server could perform the DNSSEC validation itself. It could also be configured to trust responses obtained via a validating resolver to which it has a secure connection.
- * Extract the RDATA of the TLSA records and match them to the presented client certificate according to the rules specified in the DANE TLS protocol [RFC6698] [RFC7671]. If successfully matched, the client is authenticated and the TLS session proceeds. If unsuccessful, the server MUST treat the client as unauthenticated (e.g. it could terminate the session, or proceed with the session giving the client access to resources as a generic unauthenticated user).
- * If there are multiple records in the TLSA record set, then the client is authenticated as long as at least one of the TLSA records matches, subject to RFC7671 digest agility, which SHOULD be implemented.

If the DANE Client Identity extension is not present, and the presented client certificate has multiple distinct reference identifier types (e.g. a `dNSName`, and an `rfc822Name`) then TLS servers configured to perform DANE authentication according to this specification should only examine and authenticate the `dNSName`.

If the presented client certificate has multiple `dNSName` identities, then the client MUST use the TLS DANE client identity extension to unambiguously indicate its intended name to the server.

Specific applications may be designed to require additional validation steps. For example, a server might want to verify the client's IP address is associated with the certificate in some manner, e.g. by confirming that a secure reverse DNS lookup of that address ties it back to the same domain name, or by requiring an `iPAddress` component to be included in the certificate. Such details are outside the scope of this document, and should be outlined in other documents specific to the applications that require this behavior.

Servers may have their own whitelisting and authorization rules for which certificates they accept. For example a TLS server may be configured to only allow TLS sessions from clients with certificate identities within a specific domain or set of domains.

8. Raw Public Keys

When using raw public keys in TLS [RFC7250], this specification requires the use of the TLS DANE Client Identity extension. The associated DANE TLSA records employ only certificate usage 3 (DANE-EE) and a selector value of 1 (SPKI), as described in [RFC7671].

9. Acknowledgements

TBD.

10. IANA Considerations

This document includes no request to IANA.

11. Security Considerations

This document updates RFC 6698 by defining the use of the TLSA record for client TLS certificates. There are no security considerations for this document beyond those described in RFC 6698 and RFC 7671 and in the specifications for TLS and DTLS [RFC8446], [RFC5246], [RFC6347].

12. References

12.1. Normative References

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.

- [RFC5280] 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, DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<https://www.rfc-editor.org/info/rfc6347>>.
- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <<https://www.rfc-editor.org/info/rfc6698>>.
- [RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <<https://www.rfc-editor.org/info/rfc7250>>.
- [RFC7671] Dukhovni, V. and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance", RFC 7671, DOI 10.17487/RFC7671, October 2015, <<https://www.rfc-editor.org/info/rfc7671>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.
- [TLSCLIENTID] Huque, S. and V. Dukhovni, "TLS Extension for DANE Client Identity", <<https://tools.ietf.org/html/draft-huque-tls-dane-clientid>>.

12.2. Informative References

- [CERTDEVID] Friel, O. and R. Barnes, "PKI Certificate Identifier Format for Devices", <<https://tools.ietf.org/id/draft-friel-pki-for-devices-00.html>>.
- [SRVREG] IANA, "Service Name and Transport Protocol Port Number Registry", <<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.txt>>.

Authors' Addresses

Shumon Huque
Salesforce

Email: shuque@gmail.com

Viktor Dukhovni
Two Sigma

Email: ietf-dane@dukhovni.org

Ash Wilson
Valimail

Email: ash.wilson@valimail.com

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: 18 August 2022

S. Huque
Salesforce
V. Dukhovni
Two Sigma
A. Wilson
Valimail
14 February 2022

TLS Extension for DANE Client Identity
draft-huque-tls-dane-clientid-06

Abstract

This document specifies a TLS and DTLS extension to convey a DNS-Based Authentication of Named Entities (DANE) Client Identity to a TLS or DTLS server. This is useful for applications that perform TLS client authentication via DANE TLSA records.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on 18 August 2022.

Copyright Notice

Copyright (c) 2022 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 (<https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

| | |
|---|---|
| 1. Introduction | 2 |
| 2. Overview | 2 |
| 3. DANE Client Identity Extension | 3 |
| 4. Security Considerations | 4 |
| 5. IANA Considerations | 4 |
| 6. Normative References | 4 |
| Authors' Addresses | 5 |

1. Introduction

This document specifies a Transport Layer Security (TLS) extension [RFC6066] to convey a DANE [RFC6698] Client Identity to the TLS server. This is useful for applications that perform TLS client authentication via DANE TLSA records, as described in [DANECLIENT]. The extension could be empty to indicate to the server that the client has a DANE record and that the server can perform DANE authentication of the client with the identity extracted from the client certificate. Or the extension can contain the full client identity, in the form of the DNS domain name that is expected to have a DANE TLSA record published for it.

This extension supports both TLS [RFC5246] [RFC8446] and DTLS [RFC6347], and the term TLS in this document is used generically to describe both protocols.

2. Overview

When TLS clients use X.509 client certificates or raw public keys that are authenticated via DANE TLSA records, it is useful for them to convey their intent to be authenticated via DANE, or even to convey their complete DANE identity to the server. The TLS extension defined in this document is used to accomplish this.

In the case of X.509 client certificates, a TLS server can learn the client's identity by examining subject alternative names included in the certificate itself. However, without a mechanism such as the one defined in this extension, the TLS server cannot know apriori that

the client has a published TLSA record, and thus may unnecessarily issue DNS queries for DANE TLSA records in-band with the TLS handshake even in cases where the client has no TLSA record associated with it. When multiple identities are present in the certificate, a client must use this extension to specify exactly which one the server should use. An additional situation in which this extension helps is where some TLS servers may need to selectively prompt for client certificate credentials only for clients that are equipped to provide certificates.

When TLS raw public keys [RFC7250] are being used to authenticate the client, the client uses this extension to explicitly indicate to the server what its domain name identity is (since there is no X.509 certificate from which the identity can be extracted).

Detailed protocol behavior of TLS clients and servers is described in [DANECLIENT].

3. DANE Client Identity Extension

The DANE Client Identity Extension type, "dane_clientid", will have a value assigned and registered in the IANA TLS Extensions registry. Its extension data (if not empty) has the following format:

opaque ClientName<1..2⁸-1>;

The ClientName field contains the single domain name of the client in textual presentation format, as described in RFC 1035 [RFC1035], omitting the trailing dot.

A TLS server implementing this specification MUST send an empty extension of type "dane_clientid" to indicate that it understands the extension and is capable of performing DANE client authentication. In TLS 1.2, the empty extension is sent in the ServerHello message. In TLS 1.3, it is sent in the CertificateRequest message.

A TLS client implementing this specification SHOULD send an extension of type "dane_clientid". If the client only needs to indicate that it has a DANE record and that the client's domain name identity can be obtained from its certificate, then the extension sent can be empty. If the client needs to send its domain name identity, then the "extension_data" field of the extension MUST contain a "ClientName" data structure populated with the domain name.

In TLS 1.2, the client extension is sent in the ClientHello message. In TLS 1.3, it is sent in the Certificate message. Additionally, in TLS 1.3, the client is only permitted to send the extension if it sees the corresponding empty extension in the server's CertificateRequest message.

4. Security Considerations

In TLS 1.3, this extension is sent in the CertificateRequest and Certificate messages, which are encrypted.

In TLS 1.2, this extension cannot be encrypted. When used with TLS 1.2, to prevent unnecessary privacy leakage of the client's name in cleartext, a TLS client implementing this specification should be configured to only send this extension to TLS servers it intends to perform client authentication with.

5. IANA Considerations

This extension requires the registration of a new value in the TLS ExtensionsType registry.

6. Normative References

[DANECLIENT]

Huque, S., Dukhovni, V., and A. Wilson, "TLS Client Authentication via DANE TLSA Records", 2 May 2021, <<https://tools.ietf.org/html/draft-huque-dane-client-cert>>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <<https://www.rfc-editor.org/info/rfc6066>>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<https://www.rfc-editor.org/info/rfc6347>>.

- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <<https://www.rfc-editor.org/info/rfc6698>>.
- [RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <<https://www.rfc-editor.org/info/rfc7250>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

Authors' Addresses

Shumon Huque
Salesforce

Email: shuque@gmail.com

Viktor Dukhovni
Two Sigma

Email: ietf-dane@dukhovni.org

Ash Wilson
Valimail

Email: ash.wilson@valimail.com

Internet Engineering Task Force
Internet-Draft
Updates: 6698, 7671 (if approved)
Intended status: Standards Track
Expires: 17 March 2022

A. Wilson
Valimail
S. Huque
Salesforce
13 September 2021

PKI-Authenticated Certificate Discovery Using DANE TLSA records
draft-wilson-dane-pkix-cd-02

Abstract

The DNS-Based Authentication of Named Entities (DANE) TLSA specification [RFC6698] and The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance [RFC7671] describe how to publish Transport Layer Security (TLS) server certificates or public keys in the DNS. This document updates [RFC6698] and [RFC7671]. It describes how to use the TLSA record to enable entity and CA certificate discovery for object security and trust chain discovery use cases, and how to use PKIX validation for TLSA records queried without the benefit of DNSSEC.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on 17 March 2022.

Copyright Notice

Copyright (c) 2021 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 (<https://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 Simplified BSD License.

Table of Contents

| | |
|---|---|
| 1. Background and Motivation | 2 |
| 1.1. Background | 2 |
| 1.2. Motivation | 3 |
| 2. Terminology | 3 |
| 3. Authentication Model | 3 |
| 3.1. Object Signing And Encryption | 3 |
| 3.2. Trust Anchors | 3 |
| 3.3. Trust Model For DNS-Based Identities | 4 |
| 3.4. DNS Naming Convention Or Labeling Format | 4 |
| 3.5. Trust Anchor Discovery | 5 |
| 4. Update to DANE | 6 |
| 5. PKIX Constraints | 7 |
| 6. IANA Considerations | 7 |
| 7. Security Considerations | 7 |
| 8. References | 8 |
| 8.1. Normative References | 8 |
| Authors' Addresses | 9 |

1. Background and Motivation

1.1. Background

A digital identity consists of at least two essential elements: a name, and a method for proving ownership of the name. Digital identities are often represented using X.509 certificates [RFC5280], which bind a name to a public key under a certification authority's signature. This allows all entities which trust the certification authority to trust that the public key associated with the name in the certificate is authentic and unaltered. The public key may be used for authentication, in the establishment of a secure session between two entities using a protocol like TLS or DTLS. The public key in the certificate may also be used to provide object security mechanisms like cryptographic signature verification or payload encryption. The certificate discovery process for object security usually relies on either in-band transmission of the certificate by the sender (IEEE 802.11p DSRC), or out-of-band via a proprietary API presented by the certification authority.

1.2. Motivation

Internet of Things (IoT) applications increasingly need to authenticate messages sent through decoupled applications. Without some sort of message security mechanism in place, trust in sender identity is assumed based on the trustworthiness of all systems and networks involved in handling the message between the sending entity and the recipient. This document proposes the use of the DANE TLSA record for certificate discovery, and provides an optional method for certificate authentication, in the event DNSSEC is not available.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Authentication Model

3.1. Object Signing And Encryption

An entity which generates a cryptographic signature for a message has an associated name in DNS, where a TLSA record may be found containing the entity's certificate. The entity's certificate contains the public key which can be used to authenticate the signature.

Likewise, an entity which can receive messages encrypted using a public key (either directly, or by way of a key-wrapping technique) has a DNS-based identity where a public key suitable for message encryption can be found.

3.2. Trust Anchors

A PKIX-CD identity has a discoverable certificate and trust anchor. The trust anchor MAY be used by a TLS server to populate a local certificate store for the purpose of enabling traditional PKI-based mutual TLS. This approach comes with constraints and caveats, as described in Security (Section 7), below. The use of PKIX-CD for enabling mutual TLS is a wrapper for PKI-based mutual authentication, and does not change the way that TLS operates. This approach provides a transitional state from PKIX-based DANE to DNSSEC-based DANE for mutual TLS authentication. Trust anchor discovery MAY happen on-the-fly within the TLS handshake only when using DNSSEC-based DANE.

3.3. Trust Model For DNS-Based Identities

DNSSEC SHOULD be used to ensure the integrity of the certificate presented via the TLSA record. For a variety of reasons, DNSSEC is not ubiquitous across the entire DNS namespace. For zones which are not protected by DNSSEC, the CA certificate which can be used to verify the certificates presented by the TLSA records MUST be retrieved from a known location derived from the identity's full DNS name.

3.4. DNS Naming Convention Or Labeling Format

This document defines a specific label or DNS naming format for the TLSA DNS records which carry entity certificates, and this format is compatible with the [TLSCIENTID]. This is necessary to provide a well-known location for securely retrieving a CA certificate which can be used to verify certificates retrieved from DNS without the benefit of DNSSEC.

For a device identity with DNS name `alb2c3._device.subdomain.organization.example`, we can decompose the DNS name into a few important parts:

- * `alb2c3`: device identifier
- * `_device`: identity grouping label
- * `subdomain`: organizational label
- * `organization.example`: organizational domain
- * `_device.subdomain.organization.example`: identity domain

The device identifier MAY consist of multiple labels, but for simplicity's sake we will represent it here as only one label.

The identity grouping label is the rightmost label prefixed by an underscore, to the left of the organizational domain. This does not need to be immediately adjacent to the organizational domain; labels MAY exist between the identity grouping label and the organizational domain. In the above example, this is `_device`. Another example might be `abc123._messagesender.subdomain.example.net`, where the identity grouping label would be `_messagesender`.

The organizational label(s) is any label existing between the identity grouping label and the organizational domain.

The organizational domain is the domain that was registered with a domain name registrar.

The identity domain is all labels from the top-level domain label through the identity grouping label.

3.5. Trust Anchor Discovery

In order to authenticate device certificates presented in TLSA records, without DNSSEC, we must safely obtain a CA certificate which can be used to verify the entity certificate.

Using the components of the device name, together with the authorityKeyIdentifier (AKI) found in the entity certificate (described in [RFC5280]), we can compose the URL where the signing certificate can be found.

The process whereby the signing certificate URL can be constructed for a device named `alb2c3._device.environment.example.net` is as follows:

1. Perform a DNS query for `alb2c3._device.environment.organization.example rrtype==TLSA`.
2. Extract the AKI from the certificate. We will use AA-BB-CC as the placeholder in this example.
3. Identify the organizational domain: `organization.example`
4. Identify any organizational labels: `environment`
5. Identify the identity grouping label, and remove the underscore prefix: `device`
6. Using the following pattern, create the hostname: `${IDENTITY_GROUPING_LABEL}.${ORGANIZATIONAL_LABELS}.${ORGANIZATIONAL_DOMAIN}`:
`device.environment.organization.example`
7. Adding HTTPS and the AKI, build the authority URI:
`https://device.environment.organization.example/.well-known/ca/AA-BB-CC.pem`

The file found at the URL MUST contain exactly one PEM-encoded CA certificate. The HTTPS server presenting the file MUST use TLS with a certificate that can be validated to the TLS client's client root certificate store, if DANE is not used for the server's identity. The certificate in the PEM file does not need to chain to the TLS client's root certificate store.

A PEM-encoded certificate being presented by a server possessing a DANE or Web PKI-validated identity is sufficient to indicate the trustworthiness of the CA certificate for validating certificates presented for associated identities. In this example, the certificate found at `https://device.environment.organization.example/.well-known/ca/AA-BB-CC.pem` MAY be cached and used to validate PKIX-CD certificates with identities matching `*._device.environment.example.net`, if the certificates contain AKI AA-BB-CC.

The chain of trust from signing certificate to root certificate may be discovered using the `authorityInfoAccess` ([RFC5280] section 4.2.2.1) extension within the certificate, or by substituting the `authorityKeyIdentifier` in the signing certificate for the AKI in the authority URI, and querying the updated authority URI. This process may be repeated to retrieve the entire trust chain, the root certificate of which is useful for enabling certificate-based authentication for TLS clients.

Benefits of this approach:

- * This method uses an HTTPS server as a content-addressable storage mechanism for public keys in CA certificates. The HTTPS server MAY host any number of CA certificates, and the HTTPS server does not need to provide any directory listing service. This makes the discovery of CA certificates possible by already knowing an identity's AKI, and the entire CA bundle for the zone is not required to be distributable as a bundle.
- * Device identities exist in a zone apart from other identities, which may have granular management access controls applied.
- * The DNS record used for locating the CA certificates does not exist in the same zone as the records it is used to verify. The zone where the authority record is located may have granular rules applied which create compartmentalized failure zones. If the device identity zone is compromised, altered certificates will not validate unless the authority server can also be impersonated.

4. Update to DANE

The method of certificate discovery via TLSA records, without DNSSEC, SHALL be referred to as PKIX-CD, and shall use the DANE Certificate Usage value of 4. The presence of the certificate usage value 4 indicates that the party requesting the certificate is responsible for validating it using the CA certificate located at the authority URL, which can be composed using the process described above.

5. PKIX Constraints

The certificates presented with PKIX-CD MUST contain the subjectAlternativeName OID, with at least one dNSName which matches the DNS name used to retrieve the certificate. A certificate which is delivered without the benefit of DNSSEC, even if the CA's signature is valid, MUST NOT be trusted without alignment between the DNS name used to retrieve the certificate and one dNSName attribute within the certificate. The SubjectAlternativeName field MAY have other entries in addition to the one which aligns with the DNS name where the certificate can be discovered.

6. IANA Considerations

This draft updates the TLSA Certificate Usages registry maintained at <https://www.iana.org/assignments/dane-parameters/dane-parameters.xhtml#certificate-usages> to add the following value:

| Value | Acronym | Short Description | Reference |
|-------|---------|-------------------------|-----------|
| 4 | PKIXCD | PKI-Auth Cert Discovery | [PKIXCD] |

Table 1

7. Security Considerations

This document updates RFC 6698 by defining the use of the TLSA record for discovering client TLS certificates and associated CA certificates. However, the security model presented here is quite different than the security model presented in RFC 6698. RFC 6698 relies on DNSSEC as the public key infrastructure (PKI) used for validating certificates (or certificate metadata) presented via TLSA resource records in DNS. This draft proposes the use of Web PKI as the root of trust, in the event the zone presenting TLSA records for this use case is not protected by DNSSEC. Web PKI's root of trust is in the browser CA bundle, and the substitution of Web PKI for the DNSSEC PKI trades the security implications of DNSSEC for Web PKI.

Because PKIX-CD can provide an avenue for certificate chain discovery, care must be taken to ensure that identities are authenticated via the correct chain. There may be a desire to locally cache all discovered CA certificates into a general certificate store. This must not happen without certain controls in place. Without the identity consumer (authenticator) ensuring that the signing certificate for a particular zone is only used to verify certificates ONLY from that particular zone, cross-domain

impersonation (is this a cousin of the confused deputy problem?) may be possible. An example of this is an application where organization.example and supplier.example both present identities via PKIX-CD. If the CA certificates from organization.example and supplier.example are loaded into a local certificate store and used for authenticating certificates from both zones, then devices under supplier.example may be deemed authentic even if signed by organization.example.

8. References

8.1. Normative References

- [PKIXCD] Wilson, A. and S. Huque, "DANE PKIX Certificate Discovery",
<<https://tools.ietf.org/html/draft-wilson-dane-pkix-cd>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5280] 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, DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <<https://www.rfc-editor.org/info/rfc6698>>.
- [RFC7671] Dukhovni, V. and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance", RFC 7671, DOI 10.17487/RFC7671, October 2015, <<https://www.rfc-editor.org/info/rfc7671>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [TLSCLIENTID] Huque, S. and V. Dukhovni, "TLS Extension for DANE Client Identity", <<https://tools.ietf.org/html/draft-huque-tls-dane-clientid>>.

Authors' Addresses

Ash Wilson
Valimail

Email: ash.d.wilson@gmail.com

Shumon Huque
Salesforce

Email: shuque@gmail.com