Protocol Goal & History

- **Goal:** Authenticate client side of TLS connection with DANE
- **History**
  - Drafts originally developed in mid 2015
  - Target use cases: IOT device authentication & SMTP Transport security
Protocol Summary

● Client has a DNS domain name identity
  ○ A public/private key pair & a certificate binding the public key to the domain name
  ○ Corresponding DANE TLSA record published in DNS

● TLS server
  ○ Sends Certificate Request message in handshake; extracts client identity from presented certificate, constructs TLSA query, validates DANE TLSA response with DNSSEC
Protocol Summary

- New TLS extension for conveying client’s DANE identity to the server
  - For signaling support for DANE TLS client authentication (empty extension if signal only)
  - For conveying client DNS identity when used with TLS raw public key auth (RFC 7250)
  - In TLS 1.3, this extension is carried in the (encrypted) Client Certificate message.
  - In TLS 1.2 it is carried in the first client Client Hello extension, and thus has no provision for privacy protection.
  - (Optionally, the server can also send an empty extension to signal that it supports this capability. TLS 1.3: Certificate Request message, TLS 1.2: Server Hello extension)
Client DNS Naming Convention

Draft is not proscriptive, but proposes 2 naming formats that may be generally suitable for many types of applications.

Format 1: Service specific client identity

_service.[client-domain-name]

e.g.

_smtp-client.relay1.example.com

1st label identifies the application service name. The remaining labels are composed of the client domain name. Allows the same client to have distinct authentication credentials for distinct application services.
Client DNS Naming Convention

Format 2: (IOT?) Device Identity

[deviceid]_.device.[org-domain-name]

e.g.

a1b2c3._device.subdomain.example.net.

- “a1b2c3”: device identifier (could be multiple left most labels)
- _device: identity grouping label
- subdomain: organizational label(s) (optional)
- example.net: organizational domain
sensor7._device.example.com. IN TLSA ( 
  3 1 2
  0f8b48ff5fd94117f21b6550aaee89c8
d8adbc3f433c8e587a85a14e54667b25
f4dcd8c4ae6162121ea9166984831b57
b408534451fd1b9702f8de0532ecd03c )
TLS Client e.g. IOT Device

TLS Handshake Start

Server Certificate; Client Certificate Request

Client Certificate + DANE Indication

TLS Server e.g. IOT Controller

Verifying client's certificate against DANE TLSA record in the DNS

root

org

com

example
Protocol annotation for TLS 1.3
**TLS CLIENT**

Key | ClientHello  
---|---
Exch | + key_share*  
| + psk_key_exchange_modes*  
v | + pre_shared_key*  

--------->

**TLS SERVER**

ServerHello | Key  
---|---
| + key_share*  
| + pre_shared_key*  

v  

{EncryptedExtensions} | Server  
---|---

{CertificateRequest} | Params  

**+DANE Client ID ext**  

{Certificate*} |  

{CertificateVerify*} | Auth  

{Finished} |  

<--------

[Application Data*]

^ {Certificate}

Auth | {CertificateVerify*}  
---|---

v {Finished}  

--------->

[Application Data]  

Optional capability advertisement via empty extension.
Empty extension: convey intent to be authenticated via DANE. For raw pubkey authentication, convey client’s full domain name.
TLS CLIENT

Key ^ ClientHello
Exch | + key_share*
     | + psk_key_exchange_modes*
v + pre_shared_key*

-------->  

ServerHello ^ Key
            + key_share* | Exch
            + pre_shared_key* v
{EncryptedExtensions} ^ Server
{CertificateRequest v Params
  *+DANE Client ID ext}
{Certificate*} ^
{CertificateVerify*} | Auth
{Finished} v

<-------- [Application Data*]

^ {Certificate
  +DANE Client ID ext]}
Auth | {CertificateVerify*}
v {Finished}  

-------->  

[Verify Client w/ DANE]
[TLS alert on failure ]

[Application Data] <-------- [Application Data]

Extract client’s identity, lookup TLSA RRset and authenticate the client’s cert or pubkey.
Discussion & next steps

- Should these drafts be used as the initial protocol building blocks for DANCE?
- If so, we should adopt them as WG documents.
Extra slides for Reference
DANE record in this example specifies the SHA256 hash of the subject public key of the certificate that should match the End-Entity certificate. Authenticated entirely in the DNS.