DANCE
Internet-Draft
Intended status: Informational
Expires: 24 October 2025

A. Wilson
Valimail
S. Huque
Salesforce
O. Johansson
Edvina.net
M. Richardson
Sandelman Software Works Inc

22 April 2025

An Architecture for DNS-Bound Client and Sender Identities
draft-ietf-dance-architecture-08

Abstract

This architecture document defines terminology, interaction, and authentication patterns, related to the use of DANE DNS records for TLS client and messaging peer identity, within the context of existing object security and TLS-based protocols.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the DANE Authentication for Network Clients Everywhere Working Group mailing list (dance@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/dance/.

Source for this draft and an issue tracker can be found at https://github.com/ashdwilson/draft-dance-architecture.

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 24 October 2025.

Table of Contents

1. Introduction

A digital identity, in an abstract sense, possesses at least two features: an identifier (or name), and a means of proving ownership of the identifier. One of the most resilient mechanisms for tying an identifier to a method for proving ownership of the identifier is the digital certificate, issued by a well-run Certification Authority (CA). The CA acts as a mutually trusted third party, a root of trust.

Certificate-based identities are limited in scope by the issuing CA, or by the namespace of the application responsible for issuing or validating the identity.

An example of this limitation is well-illustrated by organizational Public Key Infrastructure (PKI). Organizational PKI is very often coupled with email and LDAP systems, and can be used for associating a human or machine identity identifier with a public key. Within the organization, authentication systems already agree on the roots of trust for validating entity certificates issued by organizational PKI.

Attempting to use organizational PKI outside the organization can be challenging. In order to authenticate a certificate, the certificate's CA must be trusted. CAs have no way of controlling identifiers in certificates issued by other CAs. Consequently, trusting multiple CAs at the same time can enable entity identifier collisions. Asking an entity to trust your CA implies trust in anything that your CA signs. This is why many organizations operate a private CA, and require users and devices connecting to the organization's networks or applications to possess certificates issued by the organization's CA.

These limitations make the implementation and ongoing maintenance of a PKI costly, and have a chilling effect on the broader adoption of certificate-based IoT device identity and user identity. If certificate-based device and user identity were easier to manage, more broadly trusted, and less operationally expensive, more organizations and applications would be able to use it.

The lack of trust between PKI domains has lead to a lack of simple and globally scalable solutions for secure end-to-end inter-domain communication between entities, such as SIP phones, email and chat accounts and IoT devices belonging to different organizations.

DANCE seeks to make PKI-based user and IoT device identity universally discoverable, more broadly recognized, and less expensive to maintain by using DNS as the constraining namespace and lookup mechanism. DANCE builds on patterns established by the original DANE RFCs to enable client and sending entity certificate, public key, and trust anchor discovery. DANCE allows entities to possess a first-class identity, which, thanks to DNSSEC, may be trusted by any application also trusting the DNS. A first-class identity is an application-independent identity.

2. Conventions and Definitions

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.

How to DANCE with ENTITY: This architecture document delegates many details of how DANCE can be used with some specific protocol to a document with the name "How to DANCE with _entity_".

Identity provisioning: This refers to the set of tasks required to securely provision an asymmetric key pair for the device, sign the certificate (if the public credential is not simply a raw public key), and publish the public key or certificate in DNS. These steps may not be performed by the same party or organization. Examples:

* A device manufacturer may instantiate the key pair, and a systems integrator may be responsible for issuing (and publishing) the device certificate in DNS.

* A device manufacturer publish device identity records in DNS. The system integrator needs to perform network and application access configuration, since the identity already exists in DNS.

* A user may instantiate a key pair, based upon which an organization’s CA may produce a certificate after internally assuring the user identity, and the systems integrator may publish the CA root certificate in DNS.

DANCEr: A DANCEr is the term which is used to describe a protocol that has been taught to use DANE, usually through a _How to DANCE with_ document.

Security Domain: DNS-bound client identity allows the device to establish secure communications with any server with a DNS-bound identity, as long as a network path exists, the entity is configured to trust its communicating peer by its DNS owner name, and agreement on protocols can be achieved. The act of joining a security domain, in the past, may have involved certificate provisioning. Now, it can be as simple as using a manufacturer-provisioned identity to join the device to the network and application.

Client: This architecture document adopts the definition of "Client" from RFC 8446: "The endpoint initiating the TLS connection"

User: A client whose name consists of a user identity and a DNS owner name prefixed with a _user label.

Server: This architecture document adopts the definition of "Server" from RFC 8446: "The endpoint that did not initiate the TLS connection"

Sending agent: Software which encodes and transmits messages. A sending agent may perform tasks related to generating cryptographic signatures and/or encrypting messages before transmission.

Receiving agent: Software which interprets and processes messages. A receiving agent may perform tasks related to the decryption of messages, and verification of message signatures.

Store-and-forward system: A message handling system in-path between the sending agent and the receiving agent.

Hardware supplier role: The entity which manufactures or assembles the physical device. In many situations, multiple hardware suppliers are involved in producing a given device. In some cases, the hardware supplier may provision an asymmetric key pair for the device and establish the device identity in DNS. In some cases, the hardware supplier may ship a device with software pre-installed.

Systems integrator: The party responsible for configuration and deployment of application components. In some cases, the systems integrator also installs the software onto the device, and may provision the device identity in DNS.

Consumer: The entity or organization which pays for the value provided by the application, and defines the success criteria for the output of the application.

3. Communication Patterns

3.1. Client/Server

Client/server communication patterns imply a direct connection between an entity which provides a service (the server), and an entity which initiates a connection to the server, called a client. A secure implementation of this pattern includes a TLS-protected session directly between the client and the server. A secure implementation may also include public key-based mutual authentication.

Extending DANE to include client identity allows the server to authenticate clients independent of the private PKI used to issue the client certificate. This reduces the complexity of managing the CA certificate collection, and mitigates the possibility of client identifier collision. If the client is a user, the certificate holds an additional user identity supplied under the prerogative of a DNS owner name, which reduces the complexity of authenticating both internal and external users, through protocol mechanisms like SASL EXTERNAL [RFC4422].

3.2. Peer2peer

The extension also allows an application to find an application identity and set up a secure communication channel directly. This pattern can be used in mesh networking, IoT and in many communication protocols for multimedia sessions, chat and messaging, where each endpoint may represent a device or a user.

3.3. Decoupled

Decoupled architecture, frequently incorporating store-and-forward systems, provides no direct connection between the producer and consumer of information. The producer (or sending agent) and consumer (or receiving agent) are typically separated by at least one host running messaging-oriented middleware. The Messaging-oriented middleware components may act as a server for the purpose of establishing TLS sessions for the producer and consumer. This allows

the assertion of identity between the middleware and sending agent, and the middleware and receiving agent. The trust relationship between the sending agent and receiving agent is based on the presumed trustworthiness of the middleware, unless an identity can be attached to the message itself, independent of transport and middleware components.

Within many existing store-and-forward protocols, certificates may be transmitted within the signed message itself. An example of this is S/MIME. Within IoT applications, we find that networks may be more constrained. Including certificates in message payloads can present an unnecessary overhead on constrained network links. Decoupled applications benefit from an out-of-band public key discovery mechanism, which may enable the retrieval of certificates only when needed, and sometimes using a less expensive network connection.

4. Client authentication

4.1. Overview - DANCE usage examples

The client sets up a TLS connection to a server, attaches a client certificate with one subjectAltName element dNSName indicating the DNS owner name of the client [RFC5280]. If the client is a user, their user identity is added in one subjectAltName element otherName holding their uid attribute [RFC4519] or email address [RFC9598].

In the TLS connection the DANE-client-id extension is used to tell the server to use the certificate dNSName to find a DANE record including the public key of the certificate to be able to validate. If the server can validate the DNSSEC response, the server validates the certificate and completes the TLS connection setup.

Using DANE to convey certificate information for authenticating TLS clients gives a not-yet-authenticated client the ability to trigger a DNS lookup on the server side of the TLS connection. An opportunity for DDOS may exist when malicious clients can trigger arbitrary DNS lookups. For instance, an authoritative DNS server which has been configured to respond slowly, may cause a high concurrency of in-flight TLS authentication processes as well as open connections to upstream resolvers. This sort of attack (of type slowloris) could have a performance or availability impact on the TLS server.

4.1.1. Example 1: TLS authentication for HTTPS API interaction, DANE pattern assurance

* The client initiates a TLS connection to the server.

* The TLS server compares the dane_clientid (conveyed via the DANE Client Identity extension) to a list of allowed client domains.

* If the dane_clientid is allowed, the TLS server then performs a DNS lookup for the client's TLSA record. If the dane_clientid is not allowed, authentication fails.

* If the client's TLSA record matches the presented certificate or public key, the TLS handshake completes successfully and the authenticated dane_clientid is presented to the web application in a header field.

This pattern has the following advantages:

* This pattern translates well to TLS/TCP load balancers, by using a TCP TLV instead of an HTTP header.

* No traffic reaches the application behind the load balancer unless DANE client authentication is successful.

4.1.2. Example 2: TLS authentication for HTTPS API interaction, DANE matching in web application

* The client initiates a TLS connection to the server.

* The TLS server accepts any certificate for which the client can prove possession of the corresponding private key.

* The TLS server passes the certificate to the web application in a header field.

* The HTTP request body contains the dane_clientid, and is passed to the web application.

* The web application compares the dane_clientid to a list of allowed clients or client domains.

* If the dane_clientid is allowed, the web application makes the DNS query for the TLSA records for dane_clientid

* If the presented certificate (which was authenticated by the TLS server) matches at least one TLSA record for dane_clientid, authentication succeeds.

This pattern has the following advantages:

* In a web application where a TLS-terminating load balancer sits in front of a web application, the authentication logic in the load balancer remains simple.

* The web application ultimately decides whether to make the DNS query to support DANE authentication. This allows the web application to reject clients with identifiers which are not allowed, before making a DNS query for TLSA retrieval and comparison. No need to manage an allow-list in the load balancer.

* This can be implemented with no changes to the TLS handshake.

4.1.3. Example 3: TLS user authentication for an LDAP query

* The LDAP client initiates a TLS connection to the server, conveying the user's domain via the DANE Client Identity extension.

* If the dane_clientid is allowed and begins with a _user label, the TLS server then performs a DNS lookup for TLSA records holding the user's CA, and includes them when requesting a client certificate.

* If the client's certificate is signed by a CA found in the TLSA records and the certificate's dNSName prefixed with a _user label matches the dane_clientid then the client identity is authenticated to consist of the lowercase uid in the certificate, an "@" symbol and the lowercase UTF-8 representation of the certificate's dNSName (which lacks the "_user." prefix).

* The LDAP server responds to SASL EXTERNAL authentication by obtaining the authenticated user identity in userid@domain.name form and, if so requested, attempts to change to an authorization identity.

This pattern has the following advantages:

* SASL authentication under TLS encryption is common to many protocols, including new ones.

* This LDAP example demonstrates the potential of authentication with realm crossover support as a precursor to fine access control to possibly sensitive data.

* User identities cannot be iterated in DNS; TLS 1.3 conceals the client certificate; TLS in general conceals the user's choice of authorization identity during SASL EXTERNAL.

* This can be implemented with no changes to the TLS handshake.

4.1.4. Example 4: IoT: Device to cloud

Direct device-to-cloud communication is common in simple IoT applications. Authentication in these applications is usually accomplished using shared credentials like API keys, or using client certificates. Client certificate authentication frequently requires the consumer to maintain a CA. Before client DANE, the CA trust anchor certificate would be installed into the cloud application, and used in the TLS authentication process.

Using client DANE for device identity can allow parties other than the implementer to operate the CA. A hardware manufacturer can provide a pre-established identity, with the certificate or public key already published in DNS. This makes PKI-based identity more approachable for small organizations which currently lack the resources to operate an organizational CA.

4.1.5. Example 5: LoRaWAN

For the end-device onboarding in LoRaWAN, the "network server" and the "join server" [RFC8376] needs to establish mutual TLS authentication in order to exchange configuration parameters. Certificate Authority based mutual TLS authentication doesn’t work in LoRaWAN due to the non availability of the CA trust store in the LoRaWAN network stack. Self-signed certificate based mutual-TLS authentication method is the alternative solution.

DANE based client identity allows the server to authenticate clients during the TLS handhsake. Thus, independent of the private PKI used to issue the client’s self-signed certificate, the "network server" and the "join server" could be mutually authenticated.

4.1.6. Example 6: Edge Computing

[I-D.hong-t2trg-iot-edge-computing] may require devices to mutually authenticate in the field. A practical example of this pattern is the edge computing in construction use case [I-D.hong-t2trg-iot-edge-computing], Section 6.2.1 Using traditional certificate-based identity, the sensor and the gateway may have certificates issued by the same organizational PKI. By using DANE for client and sender identity, the sensor and the gateway may have identities represented by the equipment supplier, and still be able to mutually authenticate. Important sensor measurements forwarded by the gateway to the cloud may bear the DNS owner name and signature of the originating sensor, and the cloud application may authenticate the measurement independent of the gateway which forwarded the information to the application.

4.1.7. Example 7: Domain Users

The allocation of user identities is the prerogative of a domain, in line with the nesting suggested in URI notation. Domains may even choose to assign domain user identities to services, possibly with easily recognised identities like +mail+archive@domain.name. Domains who publish TLSA records for a CA under a _user name underneath their domain allow the validation of user identities as mentioned in a certificate as TLS client or peer identities. This mechanism is not restricted to domain-internal users, but can be used to validate users under any domain.

Since ENUM maps telephone numbers to DNS owner names, it is possible to employ these same mechanisms for telephone number users. Any DANCEr may however define alternate derivation procedures to obtain the DNS owner name for a phone number from specialised PKIX or LDAP attributes such as telephoneNumber, telexNumber, homePhone, mobile and pager.

There is no reason why other uses, such as store-and-forward with S/MIME, could not benefit from this DNS-based PKI, as long as they remain mindful that anything in the certificate is the prerogative of the domain publishing the TLSA record, and the only reliable identity statements are for resources underneath the domain -- notably, the assignment of uid names.

4.1.8. Example 8: SIP and WebRTC inter-domain privacy

End to end security in SIP is currently based on a classical S/MIME model which has not received much implementation. There are also SIP standards that build upon a trust chain anchored on the HTTP trust chain (SIP identity, STIR). WebRTC has a trust model between the web browser and the servers using TLS, but no inter-domain trust infrastructure. WebRTC lacks a definition of namespace to map to DNS, where SIP is based on an email-style addressing scheme. For WebRTC the application developer needs to define the name space and mapping to DNS.

By using DNS as a shared root of trust, SIP and WebRTC end points can anchor the keys used for DTLS/SRTP media channel setup. In addition, SIP devices can establish security in the SIP messaging by using DNS to find the callee’ s and the callers digital identity.

For an example, read [I-D.johansson-sipcore-dane-sip](SIPDANE).

4.1.9. Example 9: DNS over TLS client authentication

DNS-over-TLS client authentication is applicable to most portions of the transport segments of the DNS infrastructure. Current best practise for authentication between DNS infrastructure tends to be based upon a shared secret in the form of TSIG.

From authoritative to authoritative secondary, it can be applied to XFR-over-TLS ("XoT") as an upgrade to TSIG, removing the need for out-of-band communication of shared secrets, currently a weak point in that portion of the infrastructure.

From authoritative servers to recursive servers, in situations in which both are part of a common trust-group or have access to the same non-public or split-horizon zone data, client authentication allows authoritative servers to give selective access to specific recursive servers. Alternatively, some recursive servers could authenticate in order to gain access to non-content-related special services, such as a higher query rate-limit quota than is publicly available.

Between recursive resolvers and caching/forwarding or stub resolvers, authentication can be used to gain access to special services, such as subscription-based malware blocking, or visibility of corporate split-horizon internal zone, or to distinguish between subscribers to different performance tiers.

In the ideal implementation, client and server would bidirectionally authenticate, using DANE client certificates to bootstrap TLS transport security.

4.1.10. Example 10: SMTP, STARTTLS

SMTP has included the ability to upgrade in-protocol to TLS using the STARTTLS [RFC7817] command. When upgrading the connection, the client checks the server certificate using the DNS-ID mechanisms described in [RFC9525]. Support for this is very common and most email on the Internet is transmitted in this way.

The use of client TLS certificates has not yet become common, in part because it is unclear how or what the server would check the certificate against.

For mail-transfer-agent (MTA) to MTA communications, the use of a TLSA RR as described in [I-D.ietf-dance-client-auth] permits the SMTP server to check the identity of the parties trying to send email. There are many use cases, but a major one is often dealing with authenticated relaying of email.

4.1.11. Example 11: SSH client

SSH servers have for some time been able to put their host keys into DNS using [RFC4255].

In many SSH server implementations the list of users that is authorized to login to an account is given by listing their public keys in a per-user file ("authorized_keys"). The file provides both authorization (who may login), and authentication (how they prove their identity). While this is an implementation detail, doing both in one place has been one of Secure Shell's major reason for success.

However, there are downsides to this: a user can not easily replace their key without visiting every host they are authorized to access and update the key on that host. Separation of authorization and authentication in this case would involve putting the key material in a third place, such as in a DANE record in DNS, and then listing only the DNS owner name in the authorization file:

* A user who wants to update their key need only update DNS in that case.

* A user who has lost access to their key, but can still update DNS (or can have a colleague update it) would more easily be able to recover.

* An administrator who controls the domain would be able to remove a departing user's key from DNS, preventing the user from authenticating in the future.

The DNS record used could be TLSA, but it is possible with some protocol work that it could instead be SSHFP. Since SSH can trust CA certificates from X.509, those may be published for user authentication.

4.1.12. Example 12: Network Access

Network access refers to an authentication process by which a node is admitted securely onto network infrastructure. This is most common for wireless networks (wifi, 802.15.4), but has also routinely been done for wired infrastructure using 802.1X mechanisms with EAPOL.

While there are EAP protocols that do not involve certificates, such as EAPSIM [RFC4186], the use of symmetric key mechanisms as the "network key" is common in many homes. The use of certificate based mechanisms are expected to increase, due to challenges, such as Randomized and Changing MAC addresses (RCM), as described in [I-D.ietf-madinas-use-cases].

4.1.12.1. EAP-TLS with RADIUS

Enterprise EAP methods use a version of TLS to form a secure transport. Client and server-side certificates are used as credentials. EAP-TLS does not run over TCP, but rather over a reliable transport provided by EAP. To keep it simple the EAP "window" is always one, and there are various amounts of overhead that needs to be accounted for, and the EAP segment size is often noticeably smaller than the normal ethernet 1500 bytes. [RFC3748] does guarantee a minimum payload of 1020 bytes.

The client side certificates are often larger than 1500 bytes and can take two or three round trip times to transport from the supplicant to the authenticator. In worst case scenarios, which are common with eduroam [RFC7593], the EAP packets are transported some distance, easily across the entire planet. The authenticating system (the "authentication server" in EAP terms) is a system at the institute that issued the client side certificate, and so already has access to the entire client certificate. Transferring the client certificate is redundant. That is, the authenticator already has access to the entire certificate, but the client does not know this to the case, so it sends the entire certificate anyway.

The use of DANE Client IDs in TLS as described in [I-D.ietf-dance-tls-clientid] reduces the redundant bytes of certificate sent.

4.1.12.1.1. Terminology

Supplicant: The entity which acts as the TLS client in the EAP-TLS authentication protocol. This term is defined in IEEE 802.1x. The suppliant acts as a client in the EAPOL (EAP over LAN) protocol, which is terminated at the authenticator (defined below).

Authentication server: The entity which acts as the TLS server in the EAP-TLS protocol. RADIUS (RFC 2865) is a frequently-used authentication server protocol.

Authenticator: The authenticator is the device which acts as a server in the EAPOL (EAP over LAN) protocol, and is a client of the authentication server. The authenticator is responsible for passing EAP messages between the supplicant and the authentication server, and for ensuring that only authenticated supplicants gain access to the network.

https://datatracker.ietf.org/doc/html/rfc5216 (EAP-TLS) is a mature and widely-used protocol for network authentication, for IoT and IT equipment. IEEE 802.1x defines the encapsulation of EAP over LAN

access technologies, like IEEE 802.11 wireless and IEEE 802.3 ethernet. RADIUS is a protocol and server technology frequently used for supporting the server side of EAP-TLS authentication. Guidance for implementing RADIUS strongly encourages the use of a single common CA for all supplicants, to mitigate the possibility of identifier collisions across PKIs. The use of DANE for client identity can allow the safe use of any number of CAs. DNS acts as a constraining namespace, which prevents two unrelated CAs from issuing valid certificates bearing the same identifier. Certificates represented in DNS are valid, and all others are un-trusted.

4.1.12.2. RADSEC

The RADIUS protocol has a few recognized security problems. https://datatracker.ietf.org/doc/html/rfc6614 (RADSEC) addresses the challenges related to the weakness of MD5-based authentication and confidentiality over untrusted networks by establishing a TLS session between the RADIUS protocol client and the RADIUS protocol server. RADIUS datagrams are then transmitted between the authenticator and authentication server within the TLS session. Updating the RADSEC standard to include the use of DANE for client and server identity would allow a RADIUS server and client to mutually authenticate, independent of the client's and server's issuing CAs. The benefit for this use case is that a hosted RADIUS service may mutually authenticate any client device, like a WiFi access point or ethernet switch, via RADSEC, without requiring the distribution of CA certificates.

4.1.13. Example 13: Structured data messages: JOSE/COSE

JOSE and COSE provide formats for exchanging authenticated and encrypted structured data. JOSE defines the x5u field in [RFC7515], Section 4.1.5, and COSE defines a field of the same name in [I-D.ietf-cose-x509], Section 2.

However, this URL field points to where the key can be found. There is, as yet, no URI scheme which says that the key can be found via the DNS lookup itself.

In order to make use of x5u, a DANCEr would have to define a new URI scheme that explained how to get the right key from DNS.

5. Protocol implementations

For each protocol implementation, a specific usage document needs to be published. In this document, the DANCE protocol requirements and usage needs to be specified (this is refered above as the "How to DANCE" document). These documents should as a minimum contain the following sections:

* Specifics on naming: How the name of the client is defined and how this is related to the name in a DNS zone. This defines the organization of the related DNS zone. Whether a flat namespace is used, or a way to use a DNS Zone hierarchy is applied to this usage. (see notes above on DNS zone design)

* Privacy: If the subject name is a personal identifier, how to protect that name from being exposed in the DNS zone. [RFC7929] describes one way to handle privacy for personal identifiers in DNS.

* TTL: Recommended TTL settings for records in this usage

* Security: Security considerations for this usage

6. Security Considerations

6.1. Confidentiality

DNS clients should use DNS over TLS with trusted DNS resolvers to protect the identity of authenticating peers.

6.2. Integrity

The integrity of public keys represented in DNS is most important. An altered public key can enable device impersonation, and the denial of existence for a valid identity can cause devices to become untrusted by the network or the application. DNS records should be validated by the DNS stub resolver, using the DNSSEC protocol.

Compartmentalizing failure domains within an application is a well-known architectural best practice. Within the context of protecting DNS-based identities, this compartmentalization may manifest by hosting an identity zone on a DNS server which only supports the resource record types essential for representing device identities. This can prevent a compromised identity zone DNS server from presenting records essential for impersonating web sites under the organization’s domain name.

The naming pattern suggested in [I-D.ietf-dance-client-auth] includes an underscore label (_device). The underscore is not a valid character for names used in the Web PKI. This prevents the issuance of any Web PKI-validating certificates for these names.

This means that even were the authoritative DNS server compromised, it would not be possible to issue Web PKI certificates using, for instance, the [RFC8555] DNS-01 challenge.

An alternative underscore label _user separates the TLSA records with the domain CA from the TLSA records for devices.

6.3. Availability

One of the advantages of DNS is that it has more than fourty years of demonstrated scaling. It is a distributed database with a caching mechanism, and properly configured, it has proven resilient to many kinds of outages and attacks.

A key part of this availability is the proper use of Time To Live (TTL) values for resource records. A cache is allowed to hang on to the data for a set time, the TTL, after which it must do a new query to find out if the data has changed, or perhaps been deleted.

There is therefore a tension between resilience (higher TTL values), and agility (lower TTL values). A lower TTL value allows for revocation or replacement of a key to become known much faster. This allows for a more agile security posture.

The TTL value is not enforced, which may lead to unexpected responses, like a malicious server caching responses for a long time after the TTL for the record has expired. This may lead to a situation where a revocation by removing the record from DNS doesn’t come in to effect as expected.

On the other hand, lower TTLs cause the queries to occur more often, which may reveal more information to an observer about which devices are active. Encrypted transports like DoT/DoH/DoQ make these queries far less visible. In addition to the on-path observer being able to see more, the resolver logs also may be a source of information. It also allows for more opportunities for an attacker to affect the response time of the queries.

6.4. TLS Server availability

TLS servers supporting DANCE should implement a list of domains that are valid for client authentication, in order not to be open to DDOS attacks where a large number of clients force the server to do random DNS lookups. More implementation details are to be found in the protocol specific documents.

6.5. Privacy

If the DNS owner name of the identity proven by a certificate is directly or indirectly relatable to a person, privacy needs to be considered when forming the name of the DNS resource record for the certificate. This privacy is implied for domain users inasfar as the domain CA does not mention users. When creating the DNS owner name, effects of DNS zone walking and possible harvesting of identities in the DNS zone will have to be considered. The DNS owner name may not have to have a direct relation to the name of the subject or the subjectAltName of the certificate. If there is such a relation, a DANCEr may specify support for CA certificates, stored under a wildcard in DNS.

Further work has do be done in this area.

6.5.1. DNS Scalability

In the use case for IoT an implementation must be scalable to a large amount of devices. In many cases, identities may also be very short lived as revocation is performed by simply removing a DNS record. A zone will have to manage a large amount of changes as devices are constantly added and de-activated.

In these cases it is important to consider the architecture of the DNS zone and when possible use a tree-like structure with many subdomain parts, much like reverse DNS records or how telephone numbers are represented in the ENUM standard (RFC 6116).

If an authoritative resolver were configured to respond quite slowly using TCP, (for instance, a [slowloris] attack), it possible that this would cause a TLS server to exhaust all it’s TCP sockets.

The availability of a client identity zone is essential to permitting clients to authenticate. If the DNS infrastructure hosting client identities becomes unavailable, then the clients represented by that zone cannot be authenticated.

6.5.2. Change of ownership for IoT devices

One of the significant use cases is where the devices are identified by their manufacturer assigned identities. A significant savings was that enterprises would not have to run their own (private) PKI systems, sometimes even one system per device type. But, with this usage style for DANCE there is no private PKI to run, and as a result there is no change of ownership required. The device continues to use the manufacturer assigned identity.

The device OwnerOperator is therefore at risk if the device's manufacturer goes out of business, or decides that they no longer wish to manufacturer that device. Should that happen then the OwnerOperator of the device may be in trouble, and may find themselves having to replace the devices.

[RFC8995], Section 10.4 (BRSKI) deals with concerns about manufacturers influence on devices. In the case of BRSKI, the concern was limited to when the device ownership transfer was performed (the BRSKI transaction itself). There was no concern once the OwnerOperator had taken control over the device through an [RFC8366] voucher.

In the case of DANCE, the manufacturer is continuously involved with the day to day operation of the device.

If this is of concern, then the OwnerOperator should perform some kind of transfer of ownership, such as using DPP, [RFC8995](BRSKI), [RFC9140](EAP-NOOB), and others yet to come.

The DANCE method of using manufacturer assigned identities would therefore seem to be best used for devices which have a short lifetime: one much smaller than the uncertainty about the anticipated lifespan of the manufacturer. For instance, some kind of battery operated sensor which might be used in a large quantity at a construction site, and which can not be recharged.

7. IANA Considerations

This document has no IANA actions.

8. References

8.1. Normative References

[I-D.ietf-dance-client-auth]
Huque, S. and V. Dukhovni, "TLS Client Authentication via DANE TLSA records", Work in Progress, Internet-Draft,

draft-ietf-dance-client-auth-06, 7 November 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-dance-client-auth-06>.

[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/rfc/rfc2119>.

[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/rfc/rfc8174>.

[RFC9525] Saint-Andre, P. and R. Salz, "Service Identity in TLS", RFC 9525, DOI 10.17487/RFC9525, November 2023, <https://www.rfc-editor.org/rfc/rfc9525>.

8.2. Informative References

[I-D.hong-t2trg-iot-edge-computing]
Hong, J., Hong, Y., de Foy, X., Kovatsch, M., Schooler, E., and D. KUTSCHER, "IoT Edge Challenges and Functions", Work in Progress, Internet-Draft, draft-hong-t2trg-iot-edge-computing-05, 13 July 2020, <https://datatracker.ietf.org/doc/html/draft-hong-t2trg-iot-edge-computing-05>.

[I-D.ietf-cose-x509]
Schaad, J., "CBOR Object Signing and Encryption (COSE): Header Parameters for Carrying and Referencing X.509 Certificates", Work in Progress, Internet-Draft, draft-ietf-cose-x509-09, 13 October 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-x509-09>.

[I-D.ietf-dance-tls-clientid]
Huque, S. and V. Dukhovni, "TLS Extension for DANE Client Identity", Work in Progress, Internet-Draft, draft-ietf-dance-tls-clientid-04, 7 November 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-dance-tls-clientid-04>.

[I-D.ietf-madinas-use-cases]
Henry, J. and Y. Lee, "Randomized and Changing MAC Address: Context, Network Impacts, and Use Cases", Work in Progress, Internet-Draft, draft-ietf-madinas-use-cases-19, 20 December 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-madinas-use-cases-19>.

[I-D.johansson-sipcore-dane-sip]
Johansson, O. E., "TLS sessions in SIP using DNS-based Authentication of Named Entities (DANE) TLSA records", Work in Progress, Internet-Draft, draft-johansson-sipcore-dane-sip-00, 6 October 2014, <https://datatracker.ietf.org/doc/html/draft-johansson-sipcore-dane-sip-00>.

[pkiiot] Balakrichenan, S., Ayoub, I., and B. Ampeau, "PKI for IoT using the DNS infrastructure", IEEE, 2022 IEEE International Conference on Public Key Infrastructure and its Applications (PKIA) pp. 1-8, DOI 10.1109/pkia56009.2022.9952253, September 2022, <https://doi.org/10.1109/pkia56009.2022.9952253>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/rfc/rfc3748>.

[RFC4186] Haverinen, H., Ed. and J. Salowey, Ed., "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)", RFC 4186, DOI 10.17487/RFC4186, January 2006, <https://www.rfc-editor.org/rfc/rfc4186>.

[RFC4255] Schlyter, J. and W. Griffin, "Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints", RFC 4255, DOI 10.17487/RFC4255, January 2006, <https://www.rfc-editor.org/rfc/rfc4255>.

[RFC4422] Melnikov, A., Ed. and K. Zeilenga, Ed., "Simple Authentication and Security Layer (SASL)", RFC 4422, DOI 10.17487/RFC4422, June 2006, <https://www.rfc-editor.org/rfc/rfc4422>.

[RFC4519] Sciberras, A., Ed., "Lightweight Directory Access Protocol (LDAP): Schema for User Applications", RFC 4519, DOI 10.17487/RFC4519, June 2006, <https://www.rfc-editor.org/rfc/rfc4519>.

[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/rfc/rfc5280>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/rfc/rfc7515>.

[RFC7593] Wierenga, K., Winter, S., and T. Wolniewicz, "The eduroam Architecture for Network Roaming", RFC 7593, DOI 10.17487/RFC7593, September 2015, <https://www.rfc-editor.org/rfc/rfc7593>.

[RFC7817] Melnikov, A., "Updated Transport Layer Security (TLS) Server Identity Check Procedure for Email-Related Protocols", RFC 7817, DOI 10.17487/RFC7817, March 2016, <https://www.rfc-editor.org/rfc/rfc7817>.

[RFC7929] Wouters, P., "DNS-Based Authentication of Named Entities (DANE) Bindings for OpenPGP", RFC 7929, DOI 10.17487/RFC7929, August 2016, <https://www.rfc-editor.org/rfc/rfc7929>.

[RFC8366] Watsen, K., Richardson, M., Pritikin, M., and T. Eckert, "A Voucher Artifact for Bootstrapping Protocols", RFC 8366, DOI 10.17487/RFC8366, May 2018, <https://www.rfc-editor.org/rfc/rfc8366>.

[RFC8376] Farrell, S., Ed., "Low-Power Wide Area Network (LPWAN) Overview", RFC 8376, DOI 10.17487/RFC8376, May 2018, <https://www.rfc-editor.org/rfc/rfc8376>.

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <https://www.rfc-editor.org/rfc/rfc8555>.

[RFC8995] Pritikin, M., Richardson, M., Eckert, T., Behringer, M., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructure (BRSKI)", RFC 8995, DOI 10.17487/RFC8995, May 2021, <https://www.rfc-editor.org/rfc/rfc8995>.

[RFC9140] Aura, T., Sethi, M., and A. Peltonen, "Nimble Out-of-Band Authentication for EAP (EAP-NOOB)", RFC 9140, DOI 10.17487/RFC9140, December 2021, <https://www.rfc-editor.org/rfc/rfc9140>.

[RFC9598] Melnikov, A., Chuang, W., and C. Bonnell, "Internationalized Email Addresses in X.509 Certificates", RFC 9598, DOI 10.17487/RFC9598, May 2024, <https://www.rfc-editor.org/rfc/rfc9598>.

[slowloris]
"Slowloris Attack", 15 August 2024,
<https://en.wikipedia.org/wiki/
Slowloris_(computer_security)>.

Acknowledgments

TODO acknowledge.

Authors' Addresses

Ash Wilson
Valimail
Email: ash.d.wilson@gmail.com

Shumon Huque
Salesforce
Email: shuque@gmail.com

Olle Johansson
Edvina.net
Email: oej@edvina.net

Michael Richardson
Sandelman Software Works Inc
Email: mcr+ietf@sandelman.ca