Network Working Group
Internet-Draft
Intended status: Informational
Expires: 6 December 2025

P. M. Hallam-Baker
ThresholdSecrets.com

4 June 2025

Using TLS Client Authentication with DNS Handles
draft-hallambaker-dns-dance-00

Abstract

This is a discussion document prepared for the DANCE Working Group presenting possible convergence between the DNS Handles authentication approach based on OAUTH being used in the ATmosphere and the DANCE approach.

User experience and architectural requirements for using TLS Client Authentication in a DNS Handles based environment are discussed.

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 6 December 2025.

Table of Contents

1. Introduction

This is a discussion document prepared for the DANCE Working Group presenting possible convergence between the DNS Handles authentication approach based on OAUTH being used in the ATmosphere and the DANCE approach.

The ATmosphere has demonstrated a remarkably fast growth rate, adding 35 million users in a year. Since both approaches make use of the DNS as the basis for user authentication, it is natural to look at the possibility of convergence.

OAUTH is a mechanism for exchange of authentication and authorization data, the means by which the authentication is performed is outside the scope of OAUTH but a necessary part of any system deployment. The use of TLS Client Authentication aligned with the DNS Handles approach is clearly an option for authenticating the user to the OAUTH IdP service and with some additional infrastructure, could stand as an alternative to the OAUTH approach.

Note that the term ’DNS Handles’ is used with full knowledge of Bob Khan’s work described in [RFC3650]. The use of the term ’handles’ to refer to people in the CB radio and other applications predates that work by at least a decade. Handle is the term used by the Atmosphere community to describe their user identifiers. Adding the prefix DNS provides for disambiguation.

1.1. TLS Client Authentication

TLS Client Authentication has been a part of the protocol since SSL 3.0 and remains one of the few standard approaches to achieving client authentication against a cryptographic credential. Despite this, use has been limited because of the need to provision cryptographic credentials to users and because of the severe usability issues associated with its use.

In the earliest versions of the protocol, users were presented with a dialog asking them to choose a certificate with essentially no context to guide the selection. While TLS/1.3 [RFC8446] provides a mechanism that in theory avoids the need for the user to face a ’pick a certificate’, feedback from deployment suggests this approach is underspecified for use as a ubiquitous authentication mechanism replacing passwords

A second and currently unsolved limitation in current implementation is the issue of the cryptographic credentials themselves which is typically achieved through proprietary and out of band mechanisms. As a result, use of TLS Client Authentication in Web browsers is typically limited to enabling authentication by means of a cryptographic token such as a card or fob.

A third and entirely avoidable usability limitation is the demand that users transfer their TLS Client Authentication credential from one device to another. Secure transfer of cryptographic keys between devices is a complex and demanding task which should be avoided wherever possible. Since the purpose of TLS Client Authentication is authentication, the requirement to allow use of multiple devices is much better satisfied by issue of separate credentials to each device.

1.1.1. Captive Browser and Non-Browser

Despite falling short as a ubiquitous Web authentication approach, TLS Client Authentication is widely used in non-browser and captive browser applications. TLS Client Authentication is a widely supported approach for two Web Services to authenticate each other without introducing shared secrets. TLS Client Authentication is also used in a large number of smartphone applications where it is part of the captive browser platform around which they are built.

1.2. DNS Handles

A DNS Handle is a DNS name that is used to obtain some form of assertion relating to the user of that handle. Unlike traditional Internet account addresses using the form <username>@<service>, a DNS Handle is bound directly to a DNS label rather than an Internet service. This provides the user with considerably greater autonomy than the traditional model since they can register an Internet name, bind it to the service provider of their choice and if desired, rebind it to another service provider in the future.

Use of DNS Handles provides ’telephone number portability’ for Internet accounts.

Using alice@example.commakes Alice subordinate to her service provider example.com. Registering alice.example and using the handle @alice.example gives Alice the option of exiting her service provider at any time.

The premise of this document is essentially:

* The DNS is the naming system of the Internet

* Internet users need names

* We should use the DNS to provide Internet users with names.

* We should adapt every IETF user facing protocol to use DNS Handles

* TLS Client Authentication is one of the authentication protocols that should plug into the DNS Handles 'backplane'.

* This is an organic process that is already underway. If we want it to function correctly and well, we need to think about how to apply it now, _before_ there is a legacy userbase rather than wait for incompatible approaches to emerge.

While this suggestion has drawn criticism from people insisting 'DNS will not scale', those people do not give specifics and were saying the same when the DNS had fewer than a million registered names. The scope of DNS was originally limited to host discovery and has expanded organically to service discovery. The ATmosphere has expanded use of DNS to user discovery. If there are scaling issues, we are going to need to understand and fix them because the core principle of the Internet remains permissionless innovation.

1.2.1. A Hailing Address, not a Credential

One important caveat on current use of DNS Handles is that the handle is a means of locating a permanent unique user identifier rather than the user identifier itself. As Wheeler observed, _Any problem in computer science can be solved with another level of indirection_.

While the architecture of DNS Handles is currently in flux, the permanent unique user identifiers used in all the DNS Handle based schemes proposed to date are fingerprints of a public key used to authenticate a binding assertion. Thus, at least in theory, Alice can maintain control over her accounts even if she loses control over her DNS Handle, provided that she has control of the corresponding public key.

This approach allows Alice to use multiple DNS Handles as aliases for the same account. She can also transfer control of her DNS domain registration @alice.example without transferring control of the accounts bound to that name.

One issue arising from this approach is that there is the potential for divergence between the identity resolved through different protocols.

1.2.2. OAUTH Authentication

The ATmosphere realization of DNS Handles demonstrates a high degree of user acceptability by the fact that the scheme has gained little or no comment at all. Users are completely comfortable with the notion of an account identifier in the form @alice.example.com.

While the ATmosphere implementation is essentially a profile of OAUTH, it does currently employ a DID format that relies on a new registry 'PLC'. Use of DNS Handles as a ubiquitous authentication approach requires some adjustments to the approach so that the authentication service advertisement in the DNS allows a relying party to discover the user's IdP directly rather than having to query the PLC registry.

For example, the following DNS entry binds the handle @bsky.app to the DID did:plc:z72i7hdynmk6r22z27h6tvu__via the PLC registry.

```
_atproto.bsky.app TXT did=did:plc:z72i7hdynmk6r22z27h6tvur
```

We can eliminate the need for indirection by specifying the IdP in the TXT record:

```
_atproto.bsky.app TXT oauth=example.com \

       did=did:plc:z72i7hdynmk6r22z27h6tvur
```

Once the user is authenticated, their permanent user identifier is the DID did:plc:z72i7hdynmk6r22z27h6tvu, not the DNS Handle. It is therefore important to ensure that when DNS Handles are being used in multiple authentication mechanisms, they all refer to the same individual.

Since the string z72i7hdynmk6r22z27h6tvur is the fingerprint of a public key, we can achieve this effect by either using the same key in both contexts or by creating cross certification between the authentication keys or by introducing a primary permanent user identifier as the user's root of trust from which trust flows.

1.2.3. JSContact exchange

JSContact is an update to the venerable vCard format currently being developed by the CALSIFY working group. A proposal is currently before DISPATCH to extend the format to improve specification of public keys used in applications . [draft-hallambaker-jscontact]

We can bind a JSContact to a DNS Handle in a similar fashion to binding the OAUTH authentication provider. This approach uses the EARL Encrypted Authenticating Resource Locator scheme which ensures that the security context in which the URI is passed is preserved. If the DNS Handle is protected by DNSSEC, so is the integrity of the JSContact.

```
_jscontact.phill.hallambaker.com TXT \
        "jscontact=jscontact://example.com/ej4a-o4bp-oeel-5r5m-nb3l-b
            sae-zevq"
```

This format can be used to exchange user addresses and public key material for any Internet application protocol. A lightweight assertion format is used to support digitally signed updates. [draft-hallambaker-earl]

If the EARL is passed in person by means of a QR code, the direct trust established in that context is preserved in the application addresses and keys presented in the original context and future updates.

There is currently no alternative contact exchange scheme offering the same level of security and ease of use. Traditional contacts don’t even update, let alone update securely. Once a user adopts the JSContact/EARL approach, their DNS Handle naturally becomes their principal account handle for *every Internet service*. It is the only contact address Alice will ever need to share.

There are of course security issues implicit in ’oversharing’ contact information, these are addressed in the drafts. In brief, Alice might well decide to leave her SMTP email address out of her public contact details. Fortunately, the DNS Handle approach opens the door to deployment of next generation protocols in which every message is signed by the originator (not just the service) and only messages from authorized senders are accepted.

2. DNS Handle Providers

Currently, DNS Handles are only used in the ATmosphere and related projects. While many ATmosphere participants have registered custom handles, this is currently a minority interest.

Even with this limited scope, specialist DNS Handle Providers are already in operation. Increases in the number of users and the functionality provided by the handles will naturally increase interest in service provision, particularly if the features provided by DNS Handles compliment those of existing service providers such as password managers, virtual private network provision, Web site hosting, etc.

While TLS Client Authentication is functional without the introduction of DNS handles, the current incarnation of the technology presents many rough edges that mar the user experience.

The introduction of DNS Handle Providers presents an opportunity to introduce network services that allow the rough edges to be smoothed away providing a seamless and irritation free user experience.

While most Internet users will choose to outsource their DNS handle provision, the services required are straightforward and self-operation requires only modest DNS experience, a DNS registration and the ability to publish the authoritative zone records.

2.1. Deployment concerns

Developments in Web authentication have traditionally been left to the providers of general purpose Web browsers, the expectation being that the technologies developed for general Web authentication will trickle down to other applications.

In practice, this approach has not worked because while the circle of Web browser providers is small and the number of independent platforms even smaller, the number of Web sites is vast. Most have neither the time, nor the resources, nor the motivation to deploy new authentication technologies, particularly those which threaten to disrupt their business model.

Fortunately, the open Internet is not the only context in which Web authentication is used. TLS Client Authentication has established a modest but real application as an enterprise authentication technology where the client and server are under control of the same party and the user is paid to cater to their authentication preference.

Another area where deployment may prove more tractable is in the IoT space. TLS Client Authentication is one of the few alternatives to password authentication that supports offline use. An IoT device that requires a cloud based authentication technology is clearly going to be unsatisfactory as the basis for a ’smart home’ technology in the case of an Internet outage.

2.1.1. Hybrid Approach

The combination of OAUTH and TLS Client Authentication provide an attractive staged deployment strategy.

* In the first stage of deployment, the DHP provides an OAUTH IdP providing only password-based authentication.

* In the second stage of deployment, the DHP establishes a CA with a separate certificate signing certificate for each user under which separate TLS client authentication certificates can be issued to devices on which the user wishes to use TLS Client Authentication to authenticate to the IdP.

* In the third stage of deployment, the user adopts a browser that provides whatever usability sugar is necessary to make TLS Client Authentication an acceptable alternative as a direct authentication mechanism.

* In the fourth stage of deployment, the device credentials established to support TLS client authentication could be applied at the transaction layer to achieve controls such as non-repudiation.

Each stage represents an increase in security assurances provided to the relying party while the user experience is maintained throughout. As far as the user is concerned, they authenticate by means of their DNS Handle. The only point at which a change in the user experience would be necessary is in the final fourth stage if in addition to requiring user authentication, a transactional system also required express user intent.

Rather than presenting OAUTH and TLS Client Authentication as oppositional, the DNS Handles approach provides an approach in which both can contribute to the same end. As the use of URIs in the Web demonstrates, adoption of a consistent address scheme across protocols allows them to become interchangeable which in turn allows applications to chose the best approach for the circumstances.

This document is intended to encourage discussion of the ’usability sugar’ referred to in stage three. How to we make the user experience absolutely seamless? The techniques used would likely involve the use of ACME for certificate enrolment under some suitable profile. Applications requiring a high degree of assurance could make use of threshold techniques to enforce separation of roles and mitigate the risk of the DHP defecting.

3. Definitions

This section presents the related specifications and standard, the terms that are used as terms of art within the documents and the terms used as requirements language.

3.1. Requirements Language

This document is not normative and contains no normative text.

3.2. Defined Terms

TBS

3.3. Related Specifications

The Transport Layer Security (TLS) Protocol Version 1.3 [RFC8446]. D escribes the existing TLS Client Authentication scheme

Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile [RFC5280]. Describes the use of the X.509v3 Certificate in TLS and other widely used specifications.

JSContact: A JSON Representation of Contact Data [RFC9553]. Describe s the format used for contact data interchange.

3.4. Implementation Status

This is a discussion document only, no prototype code has been developed.

4. Authentication Concerns

We consider the points of comparison between alternative authentication schemes.

4.1. User Acceptance

In spite of psychological acceptability being one of Saltzer and Schroeder’s principles of security design, user effort and convenience are frequently overlooked in the deployment of authentication schemes.

A glaring example of the carelessness of developers is the persistence of the ’capital letter and special character’ password requirements which were introduced on 4^th November 1991 in response to the release of the Crack v4.0a password vulnerability checker.

Before the release of Crack v4.0a, the lack of any protection on the UNIX system password file was widely considered to be a demonstration of the superiority of a system that rejected ’security through obscurity’. Afterwards, there was a panicked deployment of shadow passwords and users were told to add numbers and special letters to defeat dictionary based attacks.

Modern password crackers use brute force on massively parallel machines such as the one I am typing at now which has a graphics card that can exhaustively test every 8 character password combination under a widely used digest mechanism in approximately 40 hours. The special character requirements have been irrelevant for over a decade but they are still widely enforced.

The Crack incident exposes one of the factors that makes preserving security usability so difficult, security mechanisms are frequently introduced as exceptional measures without planning or analysis. Once established, security mechanisms are very difficult to change.

4.1.1. Remembering passwords is expensive

Using passwords securely requires a herculean amount of user effort or the use of a password manager, the security of which have been brought into question by repeated breaches.

Most people find remembering things difficult. I spent over twenty minutes looking for a tool I had used only an hour earlier yesterday, so how am I supposed to remember a unique password for a Web site I last visited five years ago?

It is time for engineers to stop blaming users for sharing passwords across sites. I use strong, memorable passwords for the very small number of services that protect assets belonging to myself or that I am being paid to use. I do not waste my time creating or attempting to remember passwords to protect assets belonging to anyone else and I am pretty certain nobody else does either.

Rather than presenting users with an authentication technology that is obviously unfit for purpose and then blaming the users for ’not using it correctly’, Internet sites looking to protect their assets need to adopt technologies that don’t make absurd demands of the user’s time and effort.

4.1.2. Hardware tokens are not there when needed

Hardware tokens deliver proven security advantages in enterprise applications where the user is being paid to carry the token around with them. For most users and most uses, they are a serious inconvenience.

I do not carry keys on my person in the house, I paid a significant premium on my car so I can start it with my watch or my phone. I am not going to be carrying a hardware token about so I can log in to an online video game. I am certainly not going to be carrying about a keyring with a dozen tokens from different sources.

Use of a device the user customarily carries with them such as a watch or a smartphone provides clearly superior convenience. While the possibility of endpoint compromise is certainly a concern in some applications, it is not a

4.1.3. Multiple Devices

For modern users, use of multiple devices is the normal state of affairs. Even rural farmers making a subsistence living in developing countries commonly have multiple devices and for good reason: Their livelihoods depend on them.

Contrary to the assertions of certain parties promoting authentication technologies to replace passwords, use of multiple devices is not a strange or fringe use case. It is an absolute and essential requirement for any technology that is going to serve as a 'password replacement'.

One of the principal reasons for the continued popularity of passwords is that they work across multiple devices without the need for the user to carry an additional hardware authentication token.

4.2. Offline Use

Offline use is an important requirement for many application areas, in particular device control. A person whose Internet connection has been severed by a hurricane does not want to be locked out of their house because their 'Smart' home is unable to reach a cloud-based authentication system.

4.3. Privacy

Authentication requirements are frequently in conflict with privacy requirements.

4.3.1. User Tracking

User tracking is a key privacy concern in the modern Web and a major criticism of ’single account’ systems is that they greatly assist in user tracking. The use of separate credentials to authenticate to different sites is built into the fabric of some authentication schemes, notably Passkeys.

While systems such as passkeys don’t contribute to the tracking problem making it worse, they fall far short of a solution. User tracking technologies are built to use multiple inputs to build user profiles, including the source IP address of HTTP requests, HTTP cookies, tracking codes in URI and DNS indicators.

By itself, the use of separate credentials for separate sites provides no additional security to the user and arguably makes the user less secure by instilling a false assumption of security.

4.3.2. Desirable Tracking

The assumption that no user ever wants to be tracked across sites must be considered skeptically. One of the chief advantages of the major social media providers is precisely the fact that a user can express a single identity across a large number of disparate interest groups.

Rather than assuming that the user’s privacy requirements are straightforward and can be met by a single technological fix, we must accept that these requirements are complex, nuanced and in most cases contradictory.

4.3.3. Isolated Profiles

Privacy is a difficult problem that we cannot and must not expect to be solved by merely ’not making things worse’. The only way to achieve real isolation between user activities on the net is to build applications that are specifically designed to provide that isolation.

A starting point for such isolation is the ability to create independent profiles with independent cookie, history and authentication stores in some current browsers. This isolation could in principle be further improved by layering HTTP requests under separate profiles over separate VPN circuits.

The chief limitation to this approach is the cognitive demands placed on the user.

The introduction of DNS Handles potentially allows the browser to allow users to create a manageable number of isolated identities for different purposes without overburdening the user. Such a browser would create a separate, independent profile for each independent DNS Handle.

4.3.4. Relying Party Privacy

It is not just the privacy interests of the users that are frequently violated in authentication systems. The requirements of relying parties are often violated as well.

One of the principal weaknesses of the OpenID is that in practice, the system is operated by a closed cartel of four operators, two of which are the primary competitors of the content providers using the system for access control.

5. Gap Analysis

As currently specified, the capabilities of TLS Client Authentication are closely aligned with those of DNS Handle authentication but with gaps.

Some of these gaps are easily filled by allocation of code points such as a DNS resource record or service prefix. Other gaps are conceptual and require us to understand and document the differences between the mode of use assumed in the original design of SSL 3.0 Client Authentication in 1995 and the proposed use to support the use of DNS Handles.

5.1. Certificate Attestation

In the traditional Kohnfelder PKI model, the Certificate Authority validates the credentials presented by an applicant requesting a certificate and only issues a certificate binding the public key to a name if the validation criteria are met.

In the DNS Handles approach, the public key is initially attested by the act of resolving the DNS Handle itself and subsequently on the basis of a public key returned in the initial exchange.

These are different approaches leading to very different assurances to the relying party. PKIX is designed to provide validation of a user identity established in some external context. That context might be a name supported by government issued credentials or merely an identity established through employment in a company. In the DNS Handles approach, the only essential requirement is that the party establishing their identity through a permanent identifier in subsequent exchanges is the same as the party that established it in the first.

Provided we retain the use of PKIX client certificates as the basis for credential exchange, we have the option of making use of any degree of additional user validation in the context of DNS Handles. But this is an option, not a requirement.

There is also the possibility of making use of a DNS name path (optionally secured by DNSSEC) to provide additional information to a relying party. A company offering free versions of its software for educational use might simply assume any handle in the domain *.example.edu to be issued to a student or staff.

5.2. Certificate Issue

PKIX is an infrastructure designed to provide a large degree of flexibility allowing it to support a large number of applications. The cost of flexibility is complexity. If we are going to make use of DNS Handles on an extended basis, we need to specify a profile of the PKIX that takes as many choices that don’t matter as possible.

For example, rather than treating single device and multiple device operation as two different things, we should consider only multiple device operation with the single device as a special case.

The certificate issue process requires:

* A mechanism for provisioning certificates to multiple devices.

* A mechanism for publishing a root of trust

* Since migrating credentials across devices is a complex and risky operation, the simplest approach is to establish a Certificate Signing Certificate (CSC) for each separate user identity and to provision end entity certificates signed under the user’s CSC for each device. This approach neatly separates the requirement for publishing the root of trust and provisioning across multiple devices.

* Performing this in a seamless manner suggests that we would want to implement at least part of the ACME protocol in the client with appropriately chosen validation mechanisms.

5.3. Certificate Selection

In traditional TLS Client Authentication, the server returns a list of X.500 Distinguished Names identifying the Certificate Authorities whose certificates it accepts. The browser uses the list to filter the list of options presented to the user.

In the DNS Handles application, the server requires a certificate bound to a DNS handle rather than a certificate issued by a particular CA.

5.4. Account Binding Interaction

Most Web sites allow anonymous users to access at least some content. A mechanism through which an unauthenticated user is authenticated is therefore required. Achieving the highest levels of usability is somewhat challenging as the task of handling authentication credentials in the Web is typically performed through a HTML mediated interaction under control of the party requesting the authentication rather than the browser mediated HTTP mechanism originally anticipated.

One option is for the user to click on a ’Login’ button redirecting them to an authentication resource. If the service knows that the user prefers DNS Handle based authentication, they can return a 403 Not Authenticated response and request authentication by means of a DNS Handle in the WWW-Authenticate header.

5.5. Transaction Authentication

TLS Client Authentication does not support non-repudiation or authentication of individual transactions. Applications requiring these capabilities will require the use of a different protocol but not necessarily different credentials.

6. Strawman Architecture

The gap analysis suggests that the requirements for seamless TLS Client Authentication could be met by the following architecture:

6.1. Personal PKI

A DHP supporting the TLS Client Authentication mechanism maintains a personal PKI on behalf of each user consisting of:

* An individual Certificate Signing Certificate (CSC) for each separate user identity conforming to some PKIX profile to be determined.

* An ACME service issuing end entity client certificates conforming to some PKIX profile to be determined.

* Publishing a DNS TXT or TSLA record accrediting the CSC under the handle.

In an enterprise environment, the DHP would typically be the enterprise itself or some outsourced third-party CA service provider. The DNS Handles infrastructure providing better usability for existing TLS Client Authentication based services rather than entirely new functionality.

It is likely most consumers will be willing to trust an outsourced DHP to do the right thing on their behalf. Those that are not can run their own DHP giving them full control.

If the approach is widely adopted, a mechanism for retaining control over an outsourced DHP through separation of roles that is enforced cryptographically through threshold cryptography.

6.2. Client Features

While existing Web browsers support TLS Client Authentication, the user experience provided by this approach is less than satisfactory being ’good enough for Enterprise use’ but not intended as a mechanism for typical Web users.

6.2.1. TLS Feature Advertisement

A signaling mechanism to allow a TLS client to notify a server that it supports the DNS Handles profile for client authentication is required.

The existing TLS client feature support extension supports this requirement.

6.2.2. Authentication Mechanism Selection

A mechanism to select the DNS Handle under which to authenticate to a given site is required.

The fluency of the user interaction will depend greatly on the affordances provided by the browser for management of separate user profiles. Ideally, a user should at most be presented with a dialog asking them which DNS Handle they are going to use at a specific site the first time they interact with that site.

6.3. Server Validation Module

Services supporting DNS Handles will require some form of authentication and authorization plug in.

The deployment strategy anticipates a transition from OAUTH2 mediated authentication to direct authentication through TLSA. It is therefore likely the developers of OAUTH2 modules to support DNS Handles with OAUTH will add support for DNS Handles with TLS Client Authentication if a well-defined specification is available.

7. Security Considerations

This document is for discussion purposes and has not been subjected to a full security review.

The approach to applying TLS Client Authentication does not propose any new cryptographic extensions to DNS or TLS. The difference is in the context in which the protocol feature is used. The following issues will require further review:

Trustworthiness of the DHP managing the user’s personal PKI on their behalf. It is very likely that the demand for technical measures such as deployment of threshold cryptography to achieve separation of roles will be driven by DHPs looking to mitigate their liability exposure rather than the end users themselves.

Security of the certificate issue mechanism In particular, the mechanism by which a user authorizes provision of credentials to additional devices.

The use of multiple authentication mechanisms under a single DNS Handle. In particular, the case in which use of a DNS handle has changed from one user to another. Covering this case will at minimum require any permanent identifier established in the OAUTH authentication use case to be bound to the certificate and a prop of right under that identifier to be verified.

The use of TLS Client Authentication credentials by other protocols. The client authentication credentials established through

the mechanism intended to support TLS Client Authentication could be used for other purposes. For example, S/MIME messaging and transactional authentication. The consequences of this approach need to be thought through.

8. IANA Considerations

This document does not specify any actions for IANA.

9. Acknowledgements

TBS

10. Normative References

[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>.

[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/rfc/rfc8446>.

[RFC9553] Stepanek, R. and M. Loffredo, "JSContact: A JSON Representation of Contact Data", RFC 9553, DOI 10.17487/RFC9553, May 2024, <https://www.rfc-editor.org/rfc/rfc9553>.

11. Informative References

[draft-hallambaker-earl] Hallam-Baker, P., "Encrypted Authenticated Resource Locator", Work in Progress, Internet-Draft, draft-hallambaker-earl-00, 10 April 2025, <https://datatracker.ietf.org/doc/html/draft-hallambaker-earl-00>.

[draft-hallambaker-jscontact] Hallam-Baker, P., "JSContact Cryptographic Key Extensions", Work in Progress, Internet-Draft, draft-hallambaker-jscontact-04, 10 May 2025, <https://datatracker.ietf.org/doc/html/draft-hallambaker-jscontact-04>.

[RFC3650] Sun, S., Lannom, L., and B. Boesch, "Handle System Overview", RFC 3650, DOI 10.17487/RFC3650, November 2003, <https://www.rfc-editor.org/rfc/rfc3650>.

Author’s Address

Phillip Hallam-Baker
ThresholdSecrets.com
Email: phill@hallambaker.com