Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: 27 November 2026

D. Yu, Ed.
S. Johnson, Ed.
LinuxMagic

26 May 2026

IMAP Service Extension for Client Identity
draft-yu-imap-client-id-16.txt

Abstract

Multi-Factor Authentication has rapidly become a driving requirement for any internet based technology that requires authentication. While a large number of initiatives are active for providing solutions to this requirement for Web Browser based applications that can generally support real time human interaction for providing a secondary method of identification, legacy protocols such as [IMAP] have not yet been revised to provide such support despite being a high-risk target for business email compromise, possibly as a result of [IMAP] activity generally expecting to be non-interactive in nature outside of Webmail logins.

This document defines an extension to the [IMAP] service protocol called "CLIENTID" that an [IMAP] client can provide an additional unique identification token prior to standard credentials authentication that the server may then apply as an identity verification method in a similar manner to other Multi-Factor authentication techniques.

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 27 November 2026.

Copyright Notice

Copyright (c) 2026 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

The [IMAP] protocol and its extensions describe methods whereby a client may provide identity and/or authentication information to an IMAP server, traditionally in the form of a [username] or an [email address] in combination with a [password] string. Historically, this was the primary form of security controlling who could access the IMAP resources. While newer forms of communication may have more complex forms of authentication, and web browser applications have received heightened attention to multi-factor authentication, legacy email protocols are limited in how they can change, while still offering a means for staged adoption via backwards compatibility with all the existing email clients.

The approaches taken for other protocols do not lend themselves well to the [IMAP] protocol, and general usage. IMAP users expect their client to automatically log in, fetch their messages, and notify them of new messages in a non-interactive procedure.

The challenges with introducing novel methods of enhanced security for IMAP without upsetting the non-interactive flow of usage has limited the ability for IMAP to address the security concerns with the traditional use of username/password combinations.

A core of the problem is determining the ’identity’ of the client or person presenting authentication information in a manner that does not require active participation on the part of the user. There are many ways in which a persons authentication credentials can be obtained fraudulently; for instance accidental exposure in plain text across insecure or compromised networks, reuse of the same credentials across multiple services (password reuse) where the theft or data breach at one service allows access to the rest of the services tied to that email identifier, or theft of stored credentials in browsers and password managers.

Existing methods, such as the use of [IP], [ID], Geolocation and other tools available in the standard [IMAP] protocol currently are subject to limitations and none offer a way to sufficiently identify the source of the IMAP client with confidence. This has led to an environment of extreme risk of threat actors accessing IMAP resources, which can lead to serious financial impact and losses. Over 14 Billion email credentials have been exposed or stolen, and without methods to identify who or what is presenting authentication credentials, there is little defense against malicious or unauthorized use of those credentials.

This document defines an IMAP service extension to provide an additional identity token which can help represent the IMAP client with a higher degree of certainty than normally available in the current IMAP standards and protocols when accessing the IMAP server.

It should be noted that other proposed alternative methods of improving the situation, and the resulting threats and risks, such as [oAUTH] have been examined as alternatives, (see discussion) but did not satisfy the requirements that led to CLIENTID implementations.

Using the CLIENTID extension, an IMAP client can provide an additional identity token to the server called its "client identity". The client identity can provide unique characteristics about the client accessing the IMAP service and may be combined with existing identification mechanisms in order to identify the client. An IMAP server may then apply additional security policies using this identity such as restricting use of the service to clients presenting recognized client identities or only allowing use of authorized identities that match previously established client identities.

The CLIENTID extension is present in any IMAP implementation that returns "CLIENTID" as one of the supported capabilities to the CAPABILITY command.

1.1. Conventions Used in This Document

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.

2. CLIENTID

2.1. CLIENTID Command

Arguments: client identity type
client identity token

Responses: no specific responses for this command

Result: OK - clientid completed, client identity stored
BAD - command unknown or arguments invalid

Note that a valid CLIENTID command will never return the NO result because heuristics MUST NOT be applied to the CLIENTID arguments at this stage. Instead the client identity information SHOULD be stored within the session and passed along to any and all [SASL] authentication mechanisms.

2.2. CLIENTID Arguments

The CLIENTID command takes the following two arguments:

1. client identity type: A string identifying the identity type the client is providing. It MUST be between 1 and 16 alphanumeric and dash characters.

2. client identity token: A string identifying the client. It MUST be between 1 and 128 printable characters.

The IMAP server MUST reject any CLIENTID command with badly formatted arguments. The IMAP server MUST accept the arguments from a valid CLIENTID command and SHOULD store it at the minimum for the remaining duration of the IMAP connection.

2.3. Advertising the CLIENTID capability

The CLIENTID capability is used to tell the IMAP client that the IMAP server supports the CLIENTID extension. However, certain conditions MUST be met before the IMAP server advertises the CLIENTID capability.

1. The IMAP server and IMAP client MUST negotiate encryption via STARTTLS/SSL or some other secure mechanism.

2. The IMAP server MUST be in the non-authenticated state.

3. The IMAP server MUST have the CLIENTID extension support enabled.

When all the conditions are met, the IMAP server MUST advertise the CLIENTID capability in all proceeding CAPABILITY commands for as long as all conditions are met.

2.4. Restrictions on the CLIENTID command

Under certain circumstances, the use of the CLIENTID command will be restricted:

1. Before the CLIENTID capability has been advertised, the IMAP server MUST reject any issued CLIENTID command and the IMAP client MUST NOT issue the CLIENTID command.

2. Outside of the non-authenticated state, the IMAP server MUST reject any CLIENTID command issued by the IMAP client and the IMAP client MUST NOT issue the CLIENTID command.

3. Once a valid CLIENTID command has been issued, the IMAP server MUST reject any further CLIENTID command issued by the IMAP client and the IMAP client MUST NOT issue any subsequent CLIENTID commands.

3. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form notation as specified in [ABNF]. [IMAP] defines the non-terminals "capability" and "command-nonauth".

Except as noted otherwise, all alphabetic characters are case-insensitive. The use of upper or lower case characters to define token strings is for editorial clarity only. Implementations MUST accept these strings in a case-insensitive fashion.

```
 capability      =/ "CLIENTID"

 command-nonauth =/ client-id

 client-id       = "CLIENTID" SP client-id-type SP client-id-token

 client-id-type  = 1*16 ALPHA / DIGIT / "-"
                 ;; alphanumeric with dash character

 client-id-token = 1*128 VCHAR
                 ;; any printable US-ASCII character
```

4. Discussion

4.1. Background

The historical standard of using the user and password combination as a means of authentication is no longer effective in this day and age with recent developments in the world.

1. ISPs transitioning to Carrier-grade NAT due to IPv4 address exhaustion placing multiple devices behind the same IP address.

2. Numerous large scale data breaches exposing billions of user and password combinations.

3. Continued propensity for user to use same simple passwords across multiple accounts and services.

4. Botnets growing larger and more sophisticated due to the proliferation of IoT devices.

As a result, brute force attacks against internet services have become increasingly effective as malicious actors have easy access to billions of email addresses, commonly used passwords and massive botnets while the safety practices of users have not improved.

The traditional methods of defending against these types of attacks such as tracking the rate of failed password attempts and subsequently blocking by IP address are no longer viable without collateral damage as thousands of devices could potentially be behind the same IP address as more ISPs adopt the CGN/LSN/NAT444 standard, i.e. blocking an IP address due to the actions of a single malicious actor bears the risk of blocking legitimate users.

By introducing CLIENTID as another non-public factor to be used in tandem with the user and password combination, authentication becomes much more resilient against brute force attacks. The email addresses and passwords exposed from the data breaches will no longer be sufficient to authenticate. Security and mitigation methods such as limiting the rate in which authentication attempts can be made from a single IP address can be carried out by the CLIENTID identifier instead of the IP address, reducing the risk of falsely blocking traffic for clients behind a CGN/LSN/NAT444 connection. CLIENTID would also be backwards compatible with existing authentication protocols encouraging adoption in a viable manner.

4.2. Applying heuristics to CLIENTID

This section discusses the possible heuristics that can be applied to the information that is presented via the CLIENTID command. This information includes whether a valid CLIENTID command was issued, the client identity type and the client identity token.

1. The IMAP server MAY choose to require that a successful CLIENTID command be issued or that a particular client identity type be presented before processing or accepting an authentication request.

2. The IMAP server MAY reject any authentication request not preceded with a client identity type that matches ACL’s or rules as defined in the IMAP server.

3. An IMAP server MAY reject any authentication request preceded by a CLIENTID command that contains a client identity type or client identity token that the server chooses not to accept for any reason such as by policy.

4. An IMAP server MAY reject any authentication request preceded by a CLIENTID command that contains a client identity type or client identity token that the server has chosen to disable or revoke use of either temporarily or permanently.

The IMAP server SHOULD only ever reject an IMAP client based on CLIENTID information during or after the authentication process/ handler. In the interest of limiting the amount of information being revealed, the rejection message SHOULD be as generic as possible and SHOULD NOT reveal any information on the heuristics.

Even if the client identity type and/or client identity token are not recognized, supported or permitted by the server and/or the owner of the authentication credentials, the presented information may still be useful for analysis.

4.3. Utility of CLIENTID

Regardless of how frowned upon, users commonly reuse authorization information (like the username and password pair) across multiple services. When one service is compromised, malicious actors can also gain access to other services where the user also used the same credentials. Based on this representative problem alone, the utility of CLIENTID as an additional layer of determining the rights to present such authorization information becomes quickly apparent.

The utility of CLIENTID may be seen by considering the following:

1. An IMAP server could recognize a device not historically known to have presented the authentication credentials before.

2. An IMAP server could restrict authentication from actors not presenting a valid CLIENTID, or an account holder that the IMAP server provides service for could restrict authentication to only those devices that present valid CLIENTID.

3. An IMAP server could restrict authentication to only devices which present a CLIENTID containing a client type identifier which the account holder or operator of the server deems to be permitted. (e.g. Only allow vendor A’s devices)

4. An IMAP server could alert an account holder that an attempt to present their authorization credentials came from an unknown, unrecognized, or different device.

However, this extends beyond just the restriction of authentication. While it might be argued that this can be served as a special form of SASL, by implementing this in the IMAP service itself, the IMAP

service can choose before allowing a connection to be passed to a SASL implementation, allowing it to perform other heuristics, such as brute force attacks, more efficiently.

While ’forgery’ and/or the use of random client identifier is possible, such behavior is also more readily detectable when a device identifier is presented.

1. The IMAP server, when faced with hundreds of devices behind the same IP address, during an attack can restrict authentication attempts to only connections presenting a valid client identifier token.

2. The IMAP server, during an attack, can restrict authentication to only historically known devices.

3. The IMAP server can differentiate between many different devices behind the same IP, and apply maximum connections per device, rather than maximum connections per IP.

4. While a person may present authentication credentials from many different geographical locations, e.g. home, office, and travel, a single device will not in general be able to be in two geographical locations at the same time. The IMAP server will have new information to apply to threat detection heuristics, i.e. to treat the use of the same client identifier token from two locations, as a possible brute force or forgery situation.

4.4. Use Cases of CLIENTID

With CLIENTID the IMAP server has additional information it may use in its interactions with the client. It may:

1. Restrict use of an authorization tokens to a set of client identity token identities, thereby offering an added level of security. For example the use of authorization credentials may only be accompanied by a specified set of CLIENTID tokens and/or types for a specific account holder, or set of account holders

2. Identify that the same CLIENTID token is used to access multiple authorized identities, and restrict access to the IMAP service. For example a malicious client that has attempted to gain access using multiple authorization tokens may be identified through its unusual behavior.

3. Retain knowledge of CLIENTID tokens previously presented with specific authorization credentials, and if the token has not been previously seen, restrict access to the IMAP service.

4. Require that the IMAP client present a token such as a license key established outside of the IMAP session in order to make use of any authorized identity.

5. Apply different security policies to clients that provide a CLIENTID token versus those which do not. For example, provide clients providing such an identity with additional trust.

6. Ability to rate limit or block based on the presented client-identifier-token, when multiple devices use a shared IP address, without affecting other devices.

7. Ability to detect distributed and localized dictionary attacks and brute force attacks.

8. Use the client-identifier-token as a third factor to be passed to authentication methods eg. [SASL]

4.5. Other IMAP Client Identifiers

The [IMAP] protocol and its extensions describe methods whereby an IMAP client may provide identity information to an IMAP server. Some of these identifiers are listed for contrast:

1. The client connection provides a source IP address associated with the IMAP session. This may be accompanied by a PTR record and/or GeoIP information. This information is growing increasingly unreliable as IMAP Client softwares are increasingly configured to route traffic through 3rd party networks such as Virtual Private Networks.

2. The AUTHENTICATE and LOGIN command allows the client to present a user and/or password/authentication mechanism for an IMAP session.

3. The IMAP4 [ID] extension allows the server and client to exchange identification information for bug reports and usage statistics purposes. This specification however has limitations for the use of acting as an additional identification token for security:

* The command is valid in any state - non encrypted, authenticated, or other.

* Popular client side implementations of the [ID extension] implement the action after authentication is already complete - for ID to act as a security identifier with authentication would require a critical change to the specification that could cause conflict with existing implementations.

4.6. Alternative Considerations

Discussion on why [OAUTH] is not a consideration vs CLIENTID. Some critics have suggested that instead of CLIENTID, that the use of [OAUTH] should be improved or considered to address the problems surrounding authentication in legacy protocols. The authors of this draft considered that carefully, and researched why more adoption wasn't supported by those in the industry.

The main objections stem from two new perceived problems that [OAUTH] is seen to create. One is the requirement of an [OAUTH] service, separate from the client -> server used in traditional protocols, which is perceived to be complicated, and introduces another potential point of failure in the process. Email operators, even large ones such as ISPs and Telcos are reluctant to build out and maintain [OAUTH] services.

The alternative, eg using 3rd party [OAUTH] services is also seen to be a concern, due to adding extra risks to a stable environment, which may not be under their control, as well as concern over privacy problems when using 3rd party services for something like authentication. Concern also that the main providers of [OAUTH] services currently perform data collection as part of their business model, and the use of those services for [OAUTH] would share customer behavior inappropriately.

CLIENTID is seen to be easier to implement at the client and server level, more transparent, and less of a risk of sharing customer behavior outside of the client->server relationship. CLIENTID support is also seen to require less change in customer behavior compared to legacy usage, allows for easier transition to a more secure environment transparently.

4.7. Future Considerations

In the future there may be a demand for being able to provide multiple CLIENTID commands with different client identity types. For instance, it may be desirable for a device to identify itself, both with a hardware device identifier, and a software identifier. We believe this to be out of scope, and can be accommodated with a special client-identifier-token which encapsulates both.

In future, there may be a demand to formalize the methods to describe how the CLIENTID command should be handled for any given client identity type. We believe this to currently be out of scope, but have created a successful implementation in which a server associates a set of flags to describe how it should behave:

1. Handled but treat as not presented (ignored, no persistence)

2. Store in SMTP session but treat as not presented (for debug)

3. Store in the SMTP session, so it is available to System log

4. Store in the SMTP session, so it is available to User log

5. Use for authentication

6. Use for alert when authentication fails

7. Use for alert when authentication succeeds

8. Unused

5. Client Identity Types

This document does not specify any CLIENTID identity type that MUST be supported. The client identity type is meant to be defined by the client implementation that is designed to access the IMAP server and protocol. For instance, many IMAP client software implementations already create a distinct Universally Unique Identifer [UUID] for each account. Some commercial email clients have a license key. Some physical devices that need to interact with IMAP might have a unique hardware ID. While there is no pre-defined list of client identity type defined by this RFC, and all IMAP servers should be prepared to accept any form of client identity type that conforms to the definition, it is suggested that IMAP client developers carefully consider the name of the client identity type. For example, rather that using a client identity type of UUID, consider the advantages of making it more distinct, e.g. "<product_short_code>UUID". This way the IMAP server can better record histories, e.g. the difference between say a Thunderbird generated unique id, and a Mutt generated unique id.

Some examples of identity type might be UUID, LICENSE, DEVICE_ID, and/or COOKIE. It is expected that the most common types might be related to distinct UUID, LICENSEKEY, or HARDWAREID.

An IMAP server SHOULD NOT reject an unidentified CLIENTID type, except for specific policy use cases.

It is envisioned that in the future it will be useful to propose a set of standardized client-identity-type to help with validation, or to allow the IMAP server to apply ACL rules on expected types, this would be an extension to this RFC.

1.UUID
UUID is a common practice to represent either a individual user, hardware device or software installation associated with a specific individual. The support of UUID enables existing UUID implementations to be used to semi-uniquely identify a device associated with an individual. A definition of the format should be considered. Otherwise non-standard UUID might be a separate type specific to the software implementation, for instance TBIRD-UUID.

2.LICENSE
An IMAP client may find it useful to identify the license key of software it is using. Such licenses are typically crafted such that they are unique and useful to identify a software installation. This is more normally suited for a software designed for a single-user. While LICENSE could be standard type again, it might more more helpful to specify a vendor specific type such as BBLICENSEKEY.

3.DEVICE_ID
Many hardware devices are designed to be used by a single individual and already have an associated hardware device id. While a standard type might be defined, it also might be more helpful to use a vendor specific type, such as ATOM-DEVICEID.

4.COOKIE
While not guaranteed to be consistent many web applications are designed to access IMAP directly and may need to have a semi-unique identifier available as part of the web based transaction. It is assumed that COOKIE encompasses the group of web based tokens known to persist from session to session. A specific web based application can provide sufficient information in the actual client-identifier-token to differentiate between applications and or websites, and are convenient as they can be related to very specific domains, and are universally available to web application designers.

As a reminder, an IMAP server SHOULD NOT retain and/or store the CLIENTID information WITH authentication credentials or authentication systems directly, but the IMAP service MAY associate the CLIENTID with a specific account holder, e.g. to create a history file of known CLIENTID tokens associated or permitted to access or present authentication credentials for that account holder.

6. Examples

6.1. UUID as Client Identity

```
C: [connection established over a plaintext connection]
C: a001 CAPABILITY
S: * CAPABILITY IMAP4rev1 STARTTLS AUTH=GSSAPI LOGINDISABLED
S: a001 OK CAPABILITY completed
C: a002 STARTTLS
S: a002 OK STARTTLS completed
<TLS negotiation, further commands are under [TLS] layer>
C: a003 CAPABILITY
S: * CAPABILITY IMAP4rev1 AUTH=GSSAPI AUTH=PLAIN CLIENTID
S: a003 OK CAPABILITY completed
C: a004 CLIENTID UUID 23bf83be-aad7-46aa-9e0f-39191ccf402f
S: a004 OK CLIENTID completed
C: a005 LOGIN joe password
S: a005 OK LOGIN completed
```

6.2. Malformed CLIENTID Command

```
C: [connection established over a plaintext connection]
C: a001 CAPABILITY
S: * CAPABILITY IMAP4rev1 STARTTLS AUTH=GSSAPI LOGINDISABLED
S: a001 OK CAPABILITY completed
C: a002 STARTTLS
S: a002 OK STARTTLS completed
<TLS negotiation, further commands are under [TLS] layer>
C: a003 CAPABILITY
S: * CAPABILITY IMAP4rev1 AUTH=GSSAPI AUTH=PLAIN CLIENTID
S: a003 OK CAPABILITY completed
C: a004 CLIENTID UUID
S: a004 BAD Error in IMAP command received by server
```

The IMAP server rejects the CLIENTID command as it is not well formed due to there being only a single parameter provided.

6.3. Client Identity without TLS/SSL Session

```
C: [connection established over a plaintext connection]
C: a001 CAPABILITY
S: * CAPABILITY IMAP4rev1 STARTTLS AUTH=GSSAPI LOGINDISABLED
S: a001 OK CAPABILITY completed
C: a002 CLIENTID UUID 23bf83be-aad7-46aa-9e0f-39191ccf402f
S: a002 BAD Unknown IMAP command received by server
```

The IMAP server rejects use of the CLIENTID command as the CLIENTID capability had not been advertised because no encryption was negotiated between the IMAP server and IMAP client.

6.4. Client Identity Leading to Rejection

```
C: [connection established over a plaintext connection]
C: a001 CAPABILITY
S: * CAPABILITY IMAP4rev1 STARTTLS AUTH=GSSAPI LOGINDISABLED
S: a001 OK CAPABILITY completed
C: a002 STARTTLS
S: a002 OK STARTTLS completed
<TLS negotiation, further commands are under [TLS] layer>
C: a003 CAPABILITY
S: * CAPABILITY IMAP4rev1 AUTH=GSSAPI AUTH=PLAIN CLIENTID
S: a003 OK CAPABILITY completed
C: a004 CLIENTID UUID 23bf83be-aad7-46aa-9e0f-39191ccf402f
S: a004 OK CLIENTID completed
C: a005 LOGIN joe password
S: a005 BAD Failed to authenticate
```

The IMAP server rejects use of the system during the LOGIN command after deciding that the provided client identity does not establish sufficient privileges. Note that the error message that's returned to the client is very generic and does not reveal any information about CLIENTID and/or the existence of 'joe' and/or the validity of the password.

7. IANA Considerations

The IANA is requested to add CLIENTID to the "IMAP 4 Capabilities" registry, http://www.iana.org/assignments/imap4-capabilities.

8. Security Considerations

As this extension provides an additional means of communicating information from a client to a server, it is clear that there is additional information divulged to the server. This may have privacy considerations depending on the client identity type or its contents. For example, it may reveal a MAC address of the device used to communicate with a server that would not previously have been revealed. While it has been useful to use identifier such as email address for authentication, it is easy for these authentication tokens to be shared and/or reused and/or be publicly available for other purposes. An IMAP server and/or its operators SHOULD NOT share any CLIENTID information presented with a third party as it may represent or be linked to an individual and SHOULD never be shared in association with authentication tokens.

In essence, this provides a transparent method of multi-factor authentication requiring no modification to the IMAP resource of the client, where the traditional username and password along with any one of the unique identifiers can be used to identify a device you 'own'. However, great care should be taken by the client when

deciding on the unique identifier to use and select one that cannot be easily discovered. For example, one could use the MAC address but such an identifier may be elementary to discover and be forged by another device. While any compromise of a device MAY reveal the unique identifier, that problem is beyond the scope of the problem that this RFC is designed to solve. Consider using a different unique identifier for each service to avoid having a compromised service expose identifiers that can then be used to access another service.

Also, while this service extension requires that the identity information only be transmitted over an encrypted channel to reduce the risk of eavesdropping, it does not specify any policies or practices required in the establishment of such a channel, and so it is the responsibility of the client and the server to determine that the communication medium meets their requirements.

An example of service specific device identifiers can be seen in the implementation of CLIENTID in the Thunderbird email client.

9. References

9.1. Normative References

[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>.

[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>.

[ABNF] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 5234, January 2008, <https://www.rfc-editor.org/rfc/rfc5234>.

[IMAP] Melnikov, A. and B. Leiba, "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev2", RFC 9051, August 2021, <https://www.rfc-editor.org/rfc/rfc9051>.

[OAUTH] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <https://www.rfc-editor.org/rfc/rfc6749>.

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

[TLS] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8846, August 2018, <https://www.rfc-editor.org/rfc/rfc8846>.

Appendix A. Appendix A. CLIENTID Product Support

Since publishing the IMAP Client Identity RFC draft, multiple email server and client vendors have implemented CLIENTID support into their products, e.g. MailEnable, MagicMail, SaneBox, BlueMail, emClient, and Thunderbird.

Given the current usage and adoption of CLIENTID in the public domain, this RFC should be considered for a Standards track.

Contributors

Michael Peddemors
LinuxMagic

Authors' Addresses

Deion Yu (editor)
LinuxMagic
#405 - 860 Homer St.
Vancouver British Columbia V6B 2W5
Canada
Email: deiony@linuxmagic.com

Shaun Johnson (editor)
LinuxMagic
#405 - 860 Homer St.
Vancouver British Columbia V6B 2W5
Canada
Email: shaun@linuxmagic.com