INTERNET-DRAFT
<draft-storey-smtp-client-id-09.txt>
Intended Status: Standards Track
Expires December 4, 2020

W. Storey
LinuxMagic

June 4, 2020

SMTP Service Extension for Client Identity
<draft-storey-smtp-client-id-09.txt>

Abstract

This document defines an extension for the Simple Mail Transfer Protocol (SMTP) called "CLIENTID" to provide a method for clients to indicate an identity to the server.

This identity is an additional token that may be used for security and/or informational purposes, and with it a server may optionally apply heuristics using this token.

Table of Contents

1. Introduction

The [SMTP] protocol and its extensions describe methods whereby an SMTP client may provide identity and/or authentication information to an SMTP server. However, these existing methods are subject to limitations and none offer a way to identify the SMTP client with absolute confidence. This document defines an SMTP service extension to provide an additional identity token which can represent the SMTP client with a higher degree of certainty when accessing the SMTP server.

Typically SMTP clients are identified by establishing an authorized connection using the [AUTH] SMTP extension. SMTP servers are often subject to malicious clients attempting to use authorized identities not intended for their use (often referred to as a brute-force attack). When such an attack is attempted, the SMTP server may be unable to identify the impersonation and restrict such an unintended use by someone other than the authorized user of said credentials. While there are ways to identify the source of the SMTP client such as its IP address or EHLO identity, it would be useful if there was an additional way to uniquely identify the client in a method solely available across an encrypted channel.

Using the CLIENTID extension, an SMTP 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 SMTP service and may be combined with existing identification mechanisms in order to identify the client. An SMTP

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 key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [KEYWORDS].

2. The CLIENTID Service Extension

The following SMTP service extension is hereby defined:

1. The name of this [SMTP] service extension is "Client Identity".

2. The EHLO keyword value associated with this extension is "CLIENTID".

3. The CLIENTID keyword has no parameters.

4. A new [SMTP] verb "CLIENTID" is defined.

5. No parameter is added to any SMTP command.

6. This extension is appropriate for the submission protocol [SUBMIT].

3. The CLIENTID Keyword of the EHLO Command

The CLIENTID keyword is used to tell the SMTP client that the SMTP server supports the CLIENTID service extension. Though certain conditions must be met before the CLIENTID keyword can be advertised.

1. An SMTP server MUST NOT advertise the CLIENTID keyword in any EHLO responses if the CLIENTID extension support is not enabled.

2. An SMTP server MUST NOT advertise the CLIENTID keyword in any EHLO response if the connection is not encrypted.

3. An SMTP server MUST advertise the CLIENTID keyword in all EHLO responses after the connection is successfully encrypted (if CLIENTID is supported).

4. The CLIENTID Command

The format for the CLIENTID command is:

CLIENTID client-id-type client-id-token

Arguments:

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

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

Restrictions:

An SMTP client MUST NOT issue a CLIENTID command unless a TLS/SSL session has been negotiated as described in [STARTTLS] or through other means such as over a historical SMTP-SSL connection. An SMTP server MUST reject any CLIENTID command sent before establishing an encrypted connection with a 500 reply.

An SMTP client MUST only issue the CLIENTID command after the SMTP server advertises the CLIENTID keyword via an EHLO command. An SMTP server MUST reject a CLIENTID command prior to advertsing the CLIENTID keyword via an EHLO command.

An SMTP server MUST reject any CLIENTID command that is not well formatted with a 501 reply.

An SMTP client MUST NOT issue any subsequent CLIENTID commands after a successful CLIENTID command in the same session. An SMTP server MUST reject any subsequent CLIENTID commands after a successful CLIENTID command in the same session with a 503 reply.

An SMTP client MUST issue any CLIENTID commands prior to issuing an [AUTH] command. An SMTP server MUST reject any CLIENTID command after receiving an [AUTH] command with a 503 reply.

Several SMTP service extensions such as [AUTH] require that an SMTP session be reset to an initial state under conditions such as after applying a security layer. An SMTP server MUST discard any CLIENTID information after such a reset.

5. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form notation as specified in [ABNF]. Non-terminals referenced but not defined below are as defined by [ABNF].

Except as noted otherwise, all alphabetic characters are case-insensitive.

```
client-id-type-char = ALPHA / DIGIT / "-"
                    ;; alphanumeric and dash character

client-id-type      = 1*16 client-id-type-char
```

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

6. Discussion

6.1 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. An SMTP server MAY choose to require that a successful CLIENTID command be issued, or that a particular client type be presented before processing or accpeting an authentication request.

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

3. An SMTP 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 SMTP 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.

5. An SMTP server MAY reject any authentication request where the provided client identity is not on the list of permitted clients for the account holder.

The SMTP server SHOULD only ever reject an SMTP 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 or rules on which it bases it's decisions.

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.

6.2 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 SMTP client may be present on a device that does not have a useful domain name or network address, such as a mobile device, so its EHLO identity may be ambiguous.

2. An SMTP client may utilize the same SMTP server with multiple different authorized identities, so an identity that persists across authorized identities is lacking.

3. An authorized identity may make use of multiple discrete devices over different SMTP sessions, so an identity persisting on one device is lacking.

4. The SMTP DATA payload does not need to be inspected for this identity.

5. Connection information, a type of identity, such as network address frequently changes.

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

The recent evolution of the internet as a whole has brought about large scale data breaches, compromised botnets comprising of millions of nodes, the transition to Carrier Grade NAT and the flourishing of IoT devices means traditional methods of protecting against brute force attacks have become much more difficult. Traditional methods such as rate limiting and/or blocking access by IP are no longer viable without introducing collateral effects, such as either blocking legitimate user, or creating the conditions that allow DOS (Denial of Service) to legitimate users.

Historically, SMTP and other services used what is technically a two factor, the email/user and password, albeit not effective in that the email is a KNOWN value. And with the propensity for users to use a simple password, and the hundreds of millions of email addresses exposed in data breaches, or available by other means the ability to brute force is quite simple. While of course it is recommended that users use longer and more secure passwords, this is not the de facto situation, and the threats when credentials get compromised are significant. And with 'botnet' operators able to engage millions of IoT in a distributed brute force, the status quo is in danger. Adding another non-public factor to be used as part of access control adds a strength against brute force by many factors and accomplishes it in a backwards compatible fashion, encouraging adoption.

Under brute force attacks, rate limiting or blocking by IP Address was possible with little damage. But with the proliferaction of IoT devices, smart phones, and the run-out of IP space, we have conditions where thousands of devices could be behind an IP Address, or IP(s) that are dynamic with devices changing IP(s) in minutes,

blocking or rate limiting an IP bears risks of blocking legitimate users. By implementing a level of uniqueness to a connecting device, it introduces the ability to restrict or block a subset from connecting to the service for either brute force or dictionary attacks, while still allowing other devices to continue to be able to present authentication successfully.

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 SMTP 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 SMTP server, during an attack, can restrict authentication to only historically known devices.

3. The SMTP 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, eg, home, office, and travel, a single device will not in general be able to be in two geographical locations at the same time. The SMTP server will have new information to apply to threat detection heuristics, ie to treat the use of the same client identifier token from two locations, as a possible brute force or forgery situation.

6.3 Use Cases of CLIENTID

The SMTP server may use the additional information from CLIENTID with its interactions with SMTP clients in the following manner:

1. Restrict use of an authorized identity to a set of client identities, thereby offering an added level of security. For example, the use of an authorized identity may only be permitted from a single device using the client identity as a form of whitelisting.

2. Identify that the same client identity is used to access multiple authorized identities and restrict access to the SMTP service. For example, a client that has successfully gained access to many authorized identities may be identified through its use of a shared client identity.

3. Retain knowledge of client identities previously presented with an authorized identity and if an identity not previously seen is used

restrict access to the SMTP service.

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

any authorized identity;

5. Apply different security policies to clients that provide a client identity 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. [SASL]

6.4. Other SMTP Client Identifiers

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

1. The client connection source provides an IP address associated with the SMTP session. This may be accompanied by a PTR record and/or GeoIP information.

2. The EHLO command allows a client to identify itself with a domain or address for an SMTP session.

3. The [AUTH] SMTP extension allows the client to establish an authorized identity for an SMTP session.

4. The MAIL command identifies a specific sender for a mail transaction.

6.5. 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 accomodated with a special client identifier token which encapsulates both.

7. 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 SMTP server and protocol. For instance, many SMTP client software implementations

already create a distinct UUID for each account. Some commercial email clients have a license key. Some physical devices that need to of client identity type that conforms to the definition, it is interact with SMTP might have a unique hardware ID or MAC Address.

While there is no pre-defined list of client identity type defined by this RFC, and all SMTP servers should be prepared to accept any form suggested that SMTP 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, eg "<product_short_code>UUID". This way the SMTP server can better record histories, eg 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, MAC and/or COOKIE. It is expected that the most common types might be related to distinct UUID, LICENSEKEY, or HARDWAREID.

An SMTP 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 SMTP 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 SMTP 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. MAC

The MAC address traditionally was used as a worldwide identifier both of the unique device, as well as it's vendor and product category, however this is not always the case anymore, in the case

of it's usage in 'virtual' devices. But for many hardware devices which are required to access a defined SMTP resource, the MAC address may still be a simple unique identifier. MAC should NOT be used, unless this is a MAC address that can be associated to a vendor using standard MAC registration information as defined or set by the IEEE Standards Association and is meant to represent a unique device.

5. COOKIE

While not guranteed to be consistent many web applications are designed to access SMTP 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 SMTP server SHOULD NOT retain and/or store the CLIENTID information WITH authentication credentials or authentication systems directly, but the SMTP service MAY associate the CLIENTID with a specific account holder, eg to create a history file of known CLIENTID tokens associated or permitted to access or present authentication credentials for that account holder.

This document recommends that a server associates a set of flags that describes how the CLIENTID command should be handled for any given client identity type.

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

8. Examples

8.1 UUID Address as Client Identity

C: [connection established]
S: 220 server.example.com ESMTP ready
C: EHLO client.example.net
S: 250-server.example.com

```
   S: 250-STARTTLS
   S: 250 AUTH LOGIN
   C: STARTTLS
   S: 220 Go ahead
   C: <starts TLS negotiation>
```

```
C & S: <negotiate a TLS session>
C & S: <check result of negotiation>
C: EHLO client.example.net
S: 250-server.example.com
S: 250-AUTH LOGIN
S: 250 CLIENTID
C: CLIENTID UUID 23bf83be-aad7-46aa-9e0f-39191ccf402f
S: 250 OK
C: AUTH LOGIN dGVzdAB0ZXN0ADEyMzQ=
S: 235 Authentication successful
C: MAIL FROM:<sender@example.net>
S: 250 OK
C: RCPT TO:<receiver@example.com>
S: 250 OK
C: DATA
S: 354 Ready for message content
C: <body>
C: .
S: 250 OK
C: QUIT
S: 221 server.example.com Service closing transmission channel
```

8.2 Client Identity Without a TLS/SSL Session

```
C: [connection established over a plaintext connection]
S: 220 server.example.com ESMTP ready
C: EHLO client.example.net
S: 250-server.example.com
S: 250 STARTTLS
C: CLIENTID MAC 08:9e:01:70:f6:46
S: 500 Syntax error, command unrecognised
C: MAIL FROM:<sender@example.net>
S: 250 OK
C: QUIT
S: 221 server.example.com Service closing transmission channel
```

The server rejects use of the CLIENTID command as no TLS/SSL session was yet established.

8.3 Client Identity Leading to Rejection

```
C: [connection established over a plaintext connection]
S: 220 server.example.com ESMTP ready
C: EHLO client.example.net
S: 250-server.example.com
S: 250 STARTTLS
C: STARTTLS
S: 220 Go ahead
C: <starts TLS negotiation>
```

```
C & S: <negotiate a TLS session>
C & S: <check result of negotiation>
C: EHLO client.example.net
S: 250-server.example.com
S: 250 CLIENTID
```

```
C: CLIENTID MAC 08:9e:01:70:f6:46
S: 250 OK
C: AUTH LOGIN dGVzdAB0ZXN0ADEyMzQ=
S: 235 Authentication successful
S: 550 Server policy does not permit your use of this mail system
C: QUIT
S: 221 server.example.com Service closing transmission channel
```

The server rejects use of the mail system after deciding that the provided client identity does not establish sufficient privileges.

8.4 Malformed CLIENTID Command

```
C: [connection established over a plaintext connection]
S: 220 server.example.com ESMTP ready
C: EHLO client.example.net
S: 250-server.example.com
S: 250 STARTTLS
C: STARTTLS
S: 220 Go ahead
C: <starts TLS negotiation>
C & S: <negotiate a TLS session>
C & S: <check result of negotiation>
C: EHLO client.example.net
S: 250-server.example.com
S: 250 CLIENTID
C: CLIENTID MAC
S: 501 Syntax error in parameters or arguments
C: QUIT
S: 221 server.example.com Service closing transmission channel
```

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

9. Security Considerations

As this extension provides an additional means of communicating information from a client to a server it is clear 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 authetication tokens to be shared and/or reused and/or be publically available for other purposes. An SMTP 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.

As well, 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.

10. IANA Considerations

10.1 SMTP Extension Registration

Section 2.2.2 of [SMTP] sets out the procedure for registering a new SMTP extension.

This extension will need to be registered.

11. References

11.1. Normative References

[ABNF] Crocker, D., Ed., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <http://www.rfc-editor.org/info/rfc5234>.

[AUTH] Siemborski, R., Ed., and A. Melnikov, Ed., "SMTP Service Extension for Authentication", RFC 4954, DOI 10.17487/RFC4954, July 2007, <http://www.rfc-editor.org/info/rfc4954>.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[SMTP] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <http://www.rfc-editor.org/info/rfc5321>.

[STARTTLS] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, DOI 10.17487/RFC3207, February 2002, <http://www.rfc-editor.org/info/rfc3207>.

[SUBMIT] Gellens, R. and J. Klensin, "Message Submission for Mail", STD 72, RFC 6409, DOI 10.17487/RFC6409, November 2011, <http://www.rfc-editor.org/info/rfc6409>.

Contributors

Michael Peddemors
LinuxMagic

Authors' Addresses

William Storey
LinuxMagic
#405 - 860 Homer St.
Vancouver, British Columbia

CA V6B 2W5

EMail: william@linuxmagic.com

Deion Yu
LinuxMagic
#405 - 860 Homer St.
Vancouver, British Columbia
CA V6B 2W5

EMail: deiony@linuxmagic.com