Simple Mail Transfer Protocol

The **Simple Mail Transfer Protocol** (**SMTP**) is a communication protocol for electronic mail transmission. As an Internet standard, SMTP was first defined in 1982 by RFC 821 (https://tools.ietf.org/html/rfc821), and updated in 2008 by RFC 5321 (https://tools.ietf.org/html/rfc5321) to Extended SMTP additions, which is the protocol variety in widespread use today. Mail servers and other message transfer agents use SMTP to send and receive mail messages. SMTP servers commonly use the Transmission Control Protocol on port number 25.

User-level email clients typically use SMTP only for sending messages to a mail server for relaying, and typically submit outgoing email to the mail server on port 587 or 465 per RFC 8314. For retrieving messages, IMAP and POP3 are standard, but proprietary servers also often implement proprietary protocols, e.g., Exchange ActiveSync.

Contents

History

Various forms of one-to-one electronic messaging were used in the 1960s. Users communicated using systems developed for specific mainframe computers. As more computers were interconnected, especially in the U.S. Government's ARPANET, standards were developed to permit exchange of messages between different operating systems. SMTP grew out of these standards developed during the 1970s.

SMTP traces its roots to two implementations described in 1971: the Mail Box Protocol, whose implementation has been disputed,[1] but is discussed in RFC 196 (https://tools.ietf.org/html/rfc196) and other RFCs, and the SNDMSG program, which, according to RFC 2235 (https://tools.ietf.org/html/rfc2235), Ray Tomlinson of BBN invented for TENEX computers to send mail messages across the ARPANET.[2][3][4] Fewer than 50 hosts were connected to the ARPANET at this time.[5]

Further implementations include FTP Mail[6] and Mail Protocol, both from 1973.[7] Development work continued throughout the 1970s, until the ARPANET transitioned into the modern Internet around 1980. Jon Postel then proposed a Mail Transfer Protocol in 1980 that began to remove the mail's reliance on FTP.[8] SMTP was published as RFC 788 (https://tools.ietf.org/html/rfc788) in November 1981, also by Postel.

The SMTP standard was developed around the same time as Usenet, a one to many communication network with some similarities.

SMTP became widely used in the early 1980s. At the time, it was a complement to Unix to Unix Copy Program (UUCP) mail, which was better suited for handling email transfers between machines that were intermittently connected. SMTP, on the other hand, works best when both the sending and receiving machines are connected to the network all the time. Both use a store and forward mechanism and are examples of push technology. Though Usenet's newsgroups are still propagated with UUCP between servers,[9] UUCP as a mail transport has virtually disappeared[10] along with the "bang paths" it used as message routing headers.[11]

Sendmail, released with 4.1cBSD in 1982, soon after RFC 788 (https://tools.ietf.org/html/rfc788) was published in November 1981, was one of the first mail transfer agents to implement SMTP.[12] Over time, as BSD Unix became the most popular operating system on the Internet, Sendmail became the most common MTA (mail transfer agent).[13] Some other popular SMTP server programs include Postfix, qmail, Novell GroupWise, Exim, Novell NetMail, Microsoft Exchange Server and Oracle Communications Messaging Server.

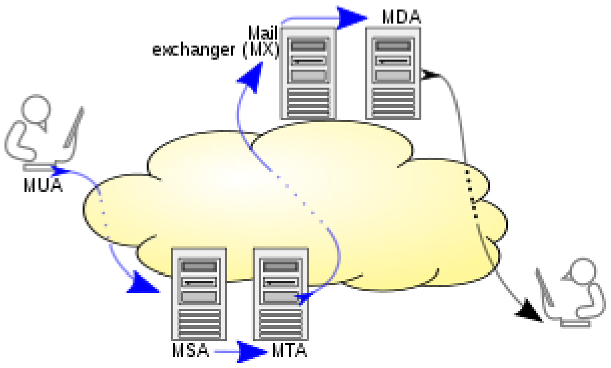
Message submission (RFC 2476 (https://tools.ietf.org/html/rfc2476)) and SMTP-AUTH (RFC 2554 (https://tools.ietf.org/html/rfc2554)) were introduced in 1998 and 1999, both describing new trends in email delivery. Originally, SMTP servers were typically internal to an organization, receiving mail for the organization *from the outside*, and relaying messages from the organization *to the outside*. But as time went on, SMTP servers (mail transfer agents), in practice, were expanding their roles to become message submission agents for Mail user agents, some of which were now relaying mail *from the outside* of an organization. (e.g. a company executive wishes to send email while on a trip using the corporate SMTP server.) This issue, a consequence of the rapid expansion and popularity of the World Wide Web, meant that SMTP had to include specific rules and methods for relaying mail and authenticating users to prevent abuses such as relaying of unsolicited email (spam). Work on message submission (RFC 2476 (https://tools.ietf.org/html/rfc2476)) was originally started because popular mail servers would often rewrite mail in an attempt to fix problems in it, for example, adding a domain name to an unqualified address. This behavior is helpful when the message being fixed is an initial submission, but dangerous and harmful when the message originated elsewhere and is being relayed. Cleanly separating mail into submission and relay was seen as a way to permit and encourage rewriting submissions while prohibiting rewriting relay. As spam became more prevalent, it was also seen as a way to provide authorization for mail being sent out from an organization, as well as traceability. This separation of relay and submission quickly became a foundation for modern email security practices.

As this protocol started out purely ASCII text-based, it did not deal well with binary files, or characters in many non-English languages. Standards such as Multipurpose Internet Mail Extensions (MIME) were developed to encode binary files for transfer through SMTP. Mail transfer agents (MTAs) developed after Sendmail also tended to be implemented 8-bit-clean, so that the alternate "just send eight" strategy could be used to transmit arbitrary text data (in any 8-bit ASCII-like character encoding) via SMTP. Mojibake was still a problem due to differing character set mappings between vendors, although the email addresses themselves still allowed only ASCII. 8-bit-clean MTAs today tend to support the 8BITMIME extension, permitting some binary files to be transmitted almost as easily as plain text (limits on line length and permitted octet values still apply, so that MIME encoding is needed for most non-text data and some text formats). Recently the SMTPUTF8 extension was created to support UTF-8 text, allowing international content and addresses in non-Latin scripts like Cyrillic or Chinese.

Many people contributed to the core SMTP specifications, among them Jon Postel, Eric Allman, Dave Crocker, Ned Freed, Randall Gellens, John Klensin, and Keith Moore.

Mail processing model

Email is submitted by a mail client (mail user agent, MUA) to a mail server (mail submission agent, MSA) using SMTP on TCP port 587. Most mailbox providers still allow submission on traditional port 25. The MSA delivers the mail to its mail transfer agent (mail transfer agent, MTA). Often, these two agents are instances of the same software launched with different options on the same machine. Local processing can be done either on a single machine, or split among multiple machines; mail agent processes on one machine can share files, but if processing is on multiple machines, they transfer messages between each other using SMTP, where each machine is configured to use the next machine as a smart host. Each process is an MTA (an SMTP server) in its own right.



Blue arrows depict implementation of SMTP variations.

The boundary MTA uses DNS to look up the MX (mail exchanger) record for the recipient's domain (the part of the email address on the right of @). The MX record contains the name of the target MTA. Based on the target host and other factors, the sending MTA selects a recipient server and connects to it to complete the mail exchange.

Message transfer can occur in a single connection between two MTAs, or in a series of hops through intermediary systems. A receiving SMTP server may be the ultimate destination, an intermediate "relay" (that is, it stores and forwards the message) or a "gateway" (that is, it may forward the message using some protocol other than SMTP). Each hop is a formal handoff of responsibility for the message, whereby the receiving server must either deliver the message or properly report the failure to do so.[14]

Once the final hop accepts the incoming message, it hands it to a mail delivery agent (MDA) for local delivery. An MDA saves messages in the relevant mailbox format. As with sending, this reception can be done using one or multiple computers, but in the diagram above the MDA is depicted as one box near the mail exchanger box. An MDA may deliver messages directly to storage, or forward them over a network using SMTP or other protocol such as Local Mail Transfer Protocol (LMTP), a derivative of SMTP designed for this purpose.

Once delivered to the local mail server, the mail is stored for batch retrieval by authenticated mail clients (MUAs). Mail is retrieved by end-user applications, called email clients, using Internet Message Access Protocol (IMAP), a protocol that both facilitates access to mail and manages stored mail, or the Post Office Protocol (POP) which typically uses the traditional mbox mail file format or a proprietary system such as Microsoft Exchange/Outlook or Lotus Notes/Domino. Webmail clients may use either method, but the retrieval protocol is often not a formal standard.

SMTP defines message *transport*, not the message *content*. Thus, it defines the mail *envelope* and its parameters, such as the envelope sender, but not the header (except *trace information*) nor the body of the message itself. STD 10 and RFC 5321 (https://tools.ietf.org/html/rfc5321) define SMTP (the envelope), while STD 11 and RFC 5322 (https://tools.ietf.org/html/rfc5322) define the message (header and body), formally referred to as the Internet Message Format.

Protocol overview

SMTP is a connection-oriented, text-based protocol in which a mail sender communicates with a mail receiver by issuing command strings and supplying necessary data over a reliable ordered data stream channel, typically a Transmission Control Protocol (TCP) connection. An *SMTP session* consists of commands originated by an SMTP client (the initiating agent, sender, or transmitter) and corresponding responses from the SMTP server (the listening agent, or receiver) so that the session is opened, and session parameters are exchanged. A session may include zero or more SMTP transactions. An *SMTP transaction* consists of three command/reply sequences:

1. **MAIL** command, to establish the return address, also called return-path,[15] reverse-path,[16] bounce address, mfrom, or envelope sender.
2. **RCPT** command, to establish a recipient of the message. This command can be issued multiple times, one for each recipient. These addresses are also part of the envelope.
3. **DATA** to signal the beginning of the *message text*; the content of the message, as opposed to its envelope. It consists of a *message header* and a *message body* separated by an empty line. DATA is actually a group of commands, and the server replies twice: once to the *DATA command* itself, to acknowledge that it is ready to receive the text, and the second time after the end-of-data sequence, to either accept or reject the entire message.

Besides the intermediate reply for DATA, each server's reply can be either positive (2xx reply codes) or negative. Negative replies can be permanent (5xx codes) or transient (4xx codes). A **reject** is a permanent failure and the client should send a bounce message to the server it received it from. A **drop** is a positive response followed by message discard rather than delivery.

The initiating host, the SMTP client, can be either an end-user's email client, functionally identified as a mail user agent (MUA), or a relay server's mail transfer agent (MTA), that is an SMTP server acting as an SMTP client, in the relevant session, in order to relay mail. Fully capable SMTP servers maintain queues of messages for retrying message transmissions that resulted in transient failures.

A MUA knows the *outgoing mail* SMTP server from its configuration. A relay server typically determines which server to connect to by looking up the MX (Mail eXchange) DNS resource record for each recipient's domain name. If no MX record is found, a conformant relaying server (not all are) instead looks up the A record. Relay servers can also be configured to use a smart host. A relay server initiates a TCP connection to the server on the "well-known port" for SMTP: port 25, or for connecting to an MSA, port 587. The main difference between an MTA and an MSA is that connecting to an MSA requires SMTP Authentication.

SMTP vs mail retrieval

SMTP is a delivery protocol only. In normal use, mail is "pushed" to a destination mail server (or next-hop mail server) as it arrives. Mail is routed based on the destination server, not the individual user(s) to which it is addressed. Other protocols, such as the Post Office Protocol (POP) and the Internet Message Access Protocol (IMAP) are specifically designed for use by individual users retrieving messages and managing mail boxes. To permit an intermittently-connected mail server to *pull* messages from a remote server on demand, SMTP has a feature to initiate mail queue processing on a remote server (see Remote Message Queue Starting below). POP and IMAP are unsuitable protocols for relaying mail by intermittently-connected machines; they are designed to operate after final delivery, when information critical to the correct operation of mail relay (the "mail envelope") has been removed.

Remote Message Queue Starting

Remote Message Queue Starting enables a remote host to start processing of the mail queue on a server so it may receive messages destined to it by sending a corresponding command. The original `TURN` command was deemed insecure[17] and was extended in RFC 1985 (https://tools.ietf.org/html/rfc1985) with the `ETRN` command which operates more securely using an authentication method based on Domain Name System information.[18]

Outgoing mail SMTP server

An email client needs to know the IP address of its initial SMTP server and this has to be given as part of its configuration (usually given as a DNS name). This server will deliver outgoing messages on behalf of the user.

Outgoing mail server access restrictions

Server administrators need to impose some control on which clients can use the server. This enables them to deal with abuse, for example spam. Two solutions have been in common use:

- In the past, many systems imposed usage restrictions by the *location* of the client, only permitting usage by clients whose IP address is one that the server administrators control.

Usage from any other client IP address is disallowed.
- Modern SMTP servers typically offer an alternative system that requires authentication of clients by credentials before allowing access.

Restricting access by location

Under this system, an ISP's SMTP server will not allow access by users who are outside the ISP's network. More precisely, the server may only allow access to users with an IP address provided by the ISP, which is equivalent to requiring that they are connected to the Internet using that same ISP. A mobile user may often be on a network other than that of their normal ISP, and will then find that sending email fails because the configured SMTP server choice is no longer accessible.

This system has several variations. For example, an organisation's SMTP server may only provide service to users on the same network, enforcing this by firewalling to block access by users on the wider Internet. Or the server may perform range checks on the client's IP address. These methods were typically used by corporations and institutions such as universities which provided an SMTP server for outbound mail only for use internally within the organisation. However, most of these bodies now use client authentication methods, as described below.

Where a user is mobile, and may use different ISPs to connect to the internet, this kind of usage restriction is onerous, and altering the configured outbound email SMTP server address is impractical. It is highly desirable to be able to use email client configuration information that does not need to change.

Client authentication

Modern SMTP servers typically require authentication of clients by credentials before allowing access, rather than restricting access by location as described earlier. This more flexible system is friendly to mobile users and allows them to have a fixed choice of configured outbound SMTP server. SMTP Authentication, often abbreviated SMTP AUTH, is an extension of the SMTP in order to log in using an authentication mechanism.

Open relay

A server that is accessible on the wider Internet and does not enforce these kinds of access restrictions is known as an open relay. This is now generally considered a bad practice worthy of blacklisting.

Ports

Communication between mail servers generally uses the standard TCP port 25 designated for SMTP.

Mail *clients* however generally don't use this, instead using specific "submission" ports. Mail services generally accept email submission from clients on one of:

- 587 (Submission), as formalized in RFC 6409 (https://tools.ietf.org/html/rfc6409) (previously RFC 2476 (https://tools.ietf.org/html/rfc2476))
- 465 This port was deprecated after RFC 2487 (https://tools.ietf.org/html/rfc2487), until the issue of RFC 8314.

Port 2525 and others may be used by some individual providers, but have never been officially supported.

Most Internet service providers now block all outgoing port 25 traffic from their customers as an anti-spam measure.[19] For the same reason, businesses will typically configure their firewall to only allow outgoing port 25 traffic from their designated mail servers.

SMTP transport example

A typical example of sending a message via SMTP to two mailboxes (*alice* and *theboss*) located in the same mail domain (*example.com* or *localhost.com*) is reproduced in the following session exchange. (In this example, the conversation parts are prefixed with *S:* and *C:*, for *server* and *client*, respectively; these labels are not part of the exchange.)

After the message sender (SMTP client) establishes a reliable communications channel to the message receiver (SMTP server), the session is opened with a greeting by the server, usually containing its fully qualified domain name (FQDN), in this case *smtp.example.com*. The client initiates its dialog by responding with a `HELO` command identifying itself in the command's parameter with its FQDN (or an address literal if none is available).[20]

```
S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.com
S: 250 smtp.example.com, I am glad to meet you
C: MAIL FROM:<bob@example.com>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: From: "Bob Example" <bob@example.com>
C: To: Alice Example <alice@example.com>
C: Cc: theboss@example.com
C: Date: Tue, 15 Jan 2008 16:02:43 -0500
C: Subject: Test message
C:
C: Hello Alice.
C: This is a test message with 5 header fields and 4 lines in the message body.
C: Your friend,
C: Bob
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
{The server closes the connection}
```

The client notifies the receiver of the originating email address of the message in a `MAIL FROM` command. This is also the return or bounce address in case the message cannot be delivered. In this example the email message is sent to two mailboxes on the same SMTP server: one for each recipient listed in the `To` and `Cc` header fields. The corresponding SMTP command is `RCPT TO`. Each successful reception and execution of a command is acknowledged by the server with a result code and response message (e.g., 250 Ok).

The transmission of the body of the mail message is initiated with a `DATA` command after which it is transmitted verbatim line by line and is terminated with an end-of-data sequence. This sequence consists of a new-line (<CR><LF>), a single full stop (period), followed by another new-line. Since a message body can contain a line with just a period as part of the text, the client sends *two* periods every time a line starts with a period; correspondingly, the server replaces every sequence of two periods at the beginning of a line with a single one. Such escaping method is called *dot-stuffing*.

The server's positive reply to the end-of-data, as exemplified, implies that the server has taken the responsibility of delivering the message. A message can be doubled if there is a communication failure at this time, e.g. due to a power shortage: Until the sender has received that `250` reply, it must assume the message was not delivered. On the other hand, after the receiver has decided to accept the message, it must assume the message has been delivered to it. Thus, during this time span, both agents have active copies of the message that they will try to deliver.[21] The probability that a communication failure occurs exactly at this step is directly proportional to the amount of filtering that the server performs on the message body, most often for anti-spam purposes. The limiting timeout is specified to be 10 minutes.[22]

The `QUIT` command ends the session. If the email has other recipients located elsewhere, the client would `QUIT` and connect to an appropriate SMTP server for subsequent recipients after the current destination(s) had been queued. The information that the client sends in the `HELO` and `MAIL FROM` commands are added (not seen in example code) as additional header fields to the message by the receiving server. It adds a `Received` and `Return-Path` header field, respectively.

Some clients are implemented to close the connection after the message is accepted (`250 Ok: queued as 12345`), so the last two lines may actually be omitted. This causes an error on the server when trying to send the `221` reply.

Extended Simple Mail Transfer Protocol

Extended SMTP (**ESMTP**), sometimes referred to as **Enhanced SMTP**, is a definition of protocol extensions to the Simple Mail Transfer Protocol (SMTP) standard. ESMTP was defined in November 1995 in IETF publication RFC 1869 which established a general structure for all existing and future extensions. ESMTP defines consistent and manageable means by which ESMTP clients and servers can be identified and servers can indicate supported extensions. The original SMTP protocol supported only unauthenticated unencrypted ASCII text communications susceptible to a man-in-the-middle attack, spoofing, and spamming, and requiring any binary data to be encoded to readable text before transmission. A number of optional extensions specify various mechanisms to address these problems.

Optional extensions discovery

Clients learn a server's supported options by using the `EHLO` greeting, as exemplified below, instead of the original `HELO` (example above). Clients fall back to `HELO` only if the server does not support SMTP extensions.[23]

Modern clients may use the ESMTP extension keyword `SIZE` to query the server for the maximum message size that will be accepted. Older clients and servers may try to transfer excessively sized messages that will be rejected after consuming network resources, including connect time to network links that is paid by the minute.[24]

Users can manually determine in advance the maximum size accepted by ESMTP servers. The client replaces the `HELO` command with the `EHLO` command.

```
S: 220 smtp2.example.com ESMTP Postfix
C: EHLO bob.example.com
S: 250-smtp2.example.com Hello bob.example.org [192.0.2.201]
S: 250-SIZE 14680064
S: 250-PIPELINING
S: 250 HELP
```

Thus *smtp2.example.com* declares that it can accept a fixed maximum message size no larger than 14,680,064 octets (8-bit bytes).

In the simplest case, an ESMTP server declares a maximum `SIZE` immediately after receiving an `EHLO`. According to RFC 1870 (https://tools.ietf.org/html/rfc1870), however, the numeric parameter to the `SIZE` extension in the `EHLO` response is optional. Clients may instead, when issuing a `MAIL FROM` command, include a numeric estimate of the size of the message they are transferring, so that the server can refuse receipt of overly-large messages.

Binary data transfer

Original SMTP supports only a single body of ASCII text, therefore any binary data needs to be encoded as text into that body of the message before transfer, and then decoded by the recipient. Binary-to-text encodings, such as uuencode and BinHex were typically used.

The 8BITMIME command was developed to address this. It was standardized in 1994 as RFC 1652 (https://tools.ietf.org/html/rfc1652)[25] It facilitates the transparent exchange of e-mail messages containing octets outside the seven-bit ASCII character set by encoding them as MIME content parts, typically encoded with Base64.

Mail delivery mechanism extensions

On-Demand Mail Relay

On-Demand Mail Relay (**ODMR**) is an SMTP extension standardized in RFC 2645 (https://tools.ietf.org/html/rfc2645) that allows an intermittently-connected SMTP server to receive email queued for it when it is connected.

Internationalization extension

Original SMTP supports email addresses composed of ASCII characters only, which is inconvenient for users whose native script is not Latin based, or who use diacritic not in the ASCII character set. This limitation was alleviated via extensions enabling UTF-8 in address names. RFC 5336 (https://tools.ietf.org/html/rfc5336) introduced experimental[26] `UTF8SMTP` command and later was superseded by RFC 6531 (https://tools.ietf.org/html/rfc6531) that introduced `SMTPUTF8` command. These extensions provide support for multi-byte and non-ASCII characters in email addresses, such as those with diacritics and other language characters such as Greek and Chinese.[27]

Current support is limited, but there is strong interest in broad adoption of RFC 6531 (https://tools.ietf.org/html/rfc6531) and the related RFCs in countries like China that have a large user base where Latin (ASCII) is a foreign script.

Extensions

Like SMTP, ESMTP is a protocol used to transport Internet mail. It is used as both an inter-server transport protocol and (with restricted behavior enforced) a mail submission protocol.

The main identification feature for ESMTP clients is to open a transmission with the command `EHLO` (Extended HELLO), rather than `HELO` (Hello, the original RFC 821 standard). A server will respond with success (code 250), failure (code 550) or error (code 500, 501, 502, 504, or 421), depending on its configuration. An ESMTP server returns the code 250 OK in a multi-line reply with its domain and a list of keywords to indicate supported extensions. A RFC 821 compliant server returns error code 500, allowing ESMTP clients to try either `HELO` or `QUIT`.

Each service extension is defined in an approved format in subsequent RFCs and registered with the Internet Assigned Numbers Authority (IANA). The first definitions were the RFC 821 optional services: `SEND`, `SOML` (Send or Mail), `SAML` (Send and Mail), `EXPN`, `HELP`, and `TURN`. The format of additional SMTP verbs was set and for new parameters in `MAIL` and `RCPT`.

Some relatively common keywords (not all of them corresponding to commands) used today are:

- `8BITMIME` – 8 bit data transmission, RFC 6152
- `ATRN` – Authenticated `TURN` for On-Demand Mail Relay, RFC 2645
- `AUTH` – Authenticated SMTP, RFC 4954
- `CHUNKING` – Chunking, RFC 3030
- `DSN` – Delivery status notification, RFC 3461 (See Variable envelope return path)
- `ETRN` – Extended version of remote message queue starting command `TURN`, RFC 1985
- `HELP` – Supply helpful information, RFC 821
- `PIPELINING` – Command pipelining, RFC 2920
- `SIZE` – Message size declaration, RFC 1870
- `STARTTLS` – Transport Layer Security, RFC 3207 (2002)
- `SMTPUTF8` – Allow UTF-8 encoding in mailbox names and header fields, RFC 6531
- `UTF8SMTP` – Allow UTF-8 encoding in mailbox names and header fields, RFC 5336 (deprecated[28])

The ESMTP format was restated in RFC 2821 (superseding RFC 821) and updated to the latest definition in RFC 5321 in 2008. Support for the `EHLO` command in servers became mandatory, and `HELO` designated a required fallback.

Non-standard, unregistered, service extensions can be used by bilateral agreement, these services are indicated by an `EHLO` message keyword starting with "X", and with any additional parameters or verbs similarly marked.

SMTP commands are case-insensitive. They are presented here in capitalized form for emphasis only. An SMTP server that requires a specific capitalization method is a violation of the standard.

8BITMIME

At least the following servers advertise the 8BITMIME extension:

- Apache James (since 2.3.0a1)[29]
- Citadel (since 7.30)
- Courier Mail Server
- Gmail[30]
- IceWarp

- IIS SMTP Service
- Kerio Connect
- Lotus Domino
- Microsoft Exchange Server (as of Exchange Server 2000)
- Novell GroupWise
- OpenSMTPD
- Oracle Communications Messaging Server
- Postfix
- Sendmail (since 6.57)

The following servers can be configured to advertise 8BITMIME, but do not perform conversion of 8-bit data to 7-bit when connecting to non-8BITMIME relays:

- Exim and qmail do not translate eight-bit messages to seven-bit when making an attempt to relay 8-bit data to non-8BITMIME peers, as is required by the RFC.[31] This does not cause problems in practice, since virtually all modern mail relays are 8-bit clean.[32]
- Microsoft Exchange Server 2003 advertises 8BITMIME by default, but relaying to a non-8BITMIME peer results in a bounce. This is allowed by RFC 6152 section 3 (http://tools.ietf.org/html/rfc6152#section-3).

SMTP-AUTH

The SMTP-AUTH extension provides an access control mechanism. It consists of an authentication step through which the client effectively logs into the mail server during the process of sending mail. Servers that support SMTP-AUTH can usually be configured to require clients to use this extension, ensuring the true identity of the sender is known. The SMTP-AUTH extension is defined in RFC 4954.

SMTP-AUTH can be used to allow legitimate users to relay mail while denying relay service to unauthorized users, such as spammers. It does not necessarily guarantee the authenticity of either the SMTP envelope sender or the RFC 2822 "From:" header. For example, spoofing, in which one sender masquerades as someone else, is still possible with SMTP-AUTH unless the server is configured to limit message from-addresses to addresses this AUTHed user is authorized for.

The SMTP-AUTH extension also allows one mail server to indicate to another that the sender has been authenticated when relaying mail. In general this requires the recipient server to trust the sending server, meaning that this aspect of SMTP-AUTH is rarely used on the Internet.

SMTPUTF8

Supporting servers include:

- Postfix (version 3.0 and later)[33]
- Momentum (versions 4.1[34] and 3.6.5, and later)
- Sendmail (under development)
- Exim (experimental as of the 4.86 release)
- CommuniGate Pro as of version 6.2.2[35]
- Courier-MTA as of version 1.0[36]
- Halon as of version 4.0[37]

- Microsoft Exchange Server as of protocol revision 14.0[38]
- Haraka and other servers.[39]
- Oracle Communications Messaging Server as of release 8.0.2.[40]

Security extensions

Mail delivery can occur both over plain text and encrypted connections, however the communicating parties might not know in advance of other party's ability to use secure channel.

SMTP Authentication

SMTP Authentication, often abbreviated SMTP AUTH, describes a mechanism for a client to log in using any authentication mechanism supported by the server. It is mainly used by submission servers, where authentication is mandatory. Multiple RFCs exist that provide different variations of the mechanism and update each other.

STARTTLS or "Opportunistic TLS"

SMTP extensions describe STARTTLS command that enables server to tell client that it supports encrypted communications and client to request an upgrade to a secure channel. STARTTLS is effective only against passive observation attacks, since the STARTTLS negotiation happens in plain text and an active attacker can trivially remove STARTTLS command, such attack is sometimes called STRIPTLS (client would think that server did not send STARTTLS header so does not support STARTTLS, while server would think that client did not respond to STARTTLS header and thus does not support STARTTLS).[41] Note that STARTTLS is also defined for IMAP and POP3 in other RFCs, but these protocols serve different purposes: SMTP is used for communication between message transfer agents, while IMAP and POP3 are for end clients and message transfer agents.

Electronic Frontier Foundation maintains a "STARTTLS Everywhere" list that similarly to "HTTPS Everywhere" list allows relying parties to discover others supporting secure communication without prior communication.[42]

RFC 8314 (https://tools.ietf.org/html/rfc8314) officially declared plain text obsolete and recommend always using TLS, adding ports with implicit TLS.

SMTP MTA Strict Transport Security

A newer 2018 RFC 8461 (https://tools.ietf.org/html/rfc8461)called "SMTP MTA Strict Transport Security (MTA-STS)" aims to address the problem of active adversary by defining a protocol for mail servers to declare their ability to use secure channels in specific files on the server and specific DNS TXT records. The relying party would regularly check existence of such record, and cache it for the amount of time specified in the record and never communicate over insecure channels until record expires.[41] Note that MTA-STS records apply only to SMTP traffic between mail servers while communications between end client and the mail server are protected by HTTPS, HTTP Strict Transport Security.

In April 2019 Google Mail announced support for MTA-STS.[43]

SMTP TLS Reporting

A number of protocols allows secure delivery of messages, but they can fail due to misconfigurations or deliberate active interference, leading to undelivered messages or delivery over unencrypted or unauthenticated channels. RFC 8460 (https://tools.ietf.org/html/rfc8460) "SMTP TLS Reporting" describes a reporting mechanism and format for sharing statistics and specific information about potential failures with recipient domains. Recipient domains can then use this information to both detect potential attacks and diagnose unintentional misconfigurations.

In April 2019 Google Mail announced support for SMTP TLS Reporting.[43]

Spoofing and spamming

The original design of SMTP had no facility to authenticate senders, or check that servers were authorized to send on their behalf, with the result that email spoofing is possible, and commonly used in email spam and phishing.

Occasional proposals are made to modify SMTP extensively or replace it completely. One example of this is Internet Mail 2000, but neither it, nor any other has made much headway in the face of the network effect of the huge installed base of classic SMTP.

Instead, mail servers now use a range of techniques, such as stricter enforcement of standards such as RFC 5322 (https://tools.ietf.org/html/rfc5322),[44][45] DomainKeys Identified Mail, Sender Policy Framework and DMARC, DNSBLs and greylisting to reject or quarantine suspicious emails.[46]

Implementations

There is also SMTP proxy implementation as for example nginx.[47]

Related requests for comments

- RFC 1123 (https://tools.ietf.org/html/rfc1123) – Requirements for Internet Hosts—Application and Support (STD 3)
- RFC 1870 (https://tools.ietf.org/html/rfc1870) – SMTP Service Extension for Message Size Declaration (obsoletes: RFC 1653 (https://tools.ietf.org/html/rfc1653))
- RFC 2505 (https://tools.ietf.org/html/rfc2505) – Anti-Spam Recommendations for SMTP MTAs (BCP 30)
- RFC 2821 (https://tools.ietf.org/html/rfc2821) – Simple Mail Transfer Protocol
- RFC 2920 (https://tools.ietf.org/html/rfc2920) – SMTP Service Extension for Command Pipelining (STD 60)
- RFC 3030 (https://tools.ietf.org/html/rfc3030) – SMTP Service Extensions for Transmission of Large and Binary MIME Messages
- RFC 3207 (https://tools.ietf.org/html/rfc3207) – SMTP Service Extension for Secure SMTP over Transport Layer Security (obsoletes RFC 2487 (https://tools.ietf.org/html/rfc2487))
- RFC 3461 (https://tools.ietf.org/html/rfc3461) – SMTP Service Extension for Delivery Status Notifications (obsoletes RFC 1891 (https://tools.ietf.org/html/rfc1891))
- RFC 3463 (https://tools.ietf.org/html/rfc3463) – Enhanced Status Codes for SMTP (obsoletes RFC 1893 (https://tools.ietf.org/html/rfc1893), updated by RFC 5248 (https://tools.ietf.org/html/rfc5248))
- RFC 3464 (https://tools.ietf.org/html/rfc3464) – An Extensible Message Format for Delivery Status Notifications (obsoletes RFC 1894 (https://tools.ietf.org/html/rfc1894))

- RFC 3798 (https://tools.ietf.org/html/rfc3798) – Message Disposition Notification (updates RFC 3461 (https://tools.ietf.org/html/rfc3461))
- RFC 3834 (https://tools.ietf.org/html/rfc3834) – Recommendations for Automatic Responses to Electronic Mail
- RFC 3974 (https://tools.ietf.org/html/rfc3974) – SMTP Operational Experience in Mixed IPv4/v6 Environments
- RFC 4952 (https://tools.ietf.org/html/rfc4952) – Overview and Framework for Internationalized Email (updated by RFC 5336 (https://tools.ietf.org/html/rfc5336))
- RFC 4954 (https://tools.ietf.org/html/rfc4954) – SMTP Service Extension for Authentication (obsoletes RFC 2554 (https://tools.ietf.org/html/rfc2554), updates RFC 3463 (https://tools.ietf.org/html/rfc3463), updated by RFC 5248 (https://tools.ietf.org/html/rfc5248))
- RFC 5068 (https://tools.ietf.org/html/rfc5068) – Email Submission Operations: Access and Accountability Requirements (BCP 134)
- RFC 5248 (https://tools.ietf.org/html/rfc5248) – A Registry for SMTP Enhanced Mail System Status Codes (BCP 138) (updates RFC 3463 (https://tools.ietf.org/html/rfc3463))
- RFC 5321 (https://tools.ietf.org/html/rfc5321) – The Simple Mail Transfer Protocol (obsoletes RFC 821 (https://tools.ietf.org/html/rfc821) aka STD 10, RFC 974 (https://tools.ietf.org/html/rfc974), RFC 1869 (https://tools.ietf.org/html/rfc1869), RFC 2821 (https://tools.ietf.org/html/rfc2821), updates RFC 1123 (https://tools.ietf.org/html/rfc1123))
- RFC 5322 (https://tools.ietf.org/html/rfc5322) – Internet Message Format (obsoletes RFC 822 (https://tools.ietf.org/html/rfc822) aka STD 11, and RFC 2822 (https://tools.ietf.org/html/rfc2822))
- RFC 5504 (https://tools.ietf.org/html/rfc5504) – Downgrading Mechanism for Email Address Internationalization
- RFC 6409 (https://tools.ietf.org/html/rfc6409) – Message Submission for Mail (STD 72) (obsoletes RFC 4409 (https://tools.ietf.org/html/rfc4409), RFC 2476 (https://tools.ietf.org/html/rfc2476))
- RFC 6522 (https://tools.ietf.org/html/rfc6522) – The Multipart/Report Content Type for the Reporting of Mail System Administrative Messages (obsoletes RFC 3462 (https://tools.ietf.org/html/rfc3462), and in turn RFC 1892 (https://tools.ietf.org/html/rfc1892))
- RFC 6531 (https://tools.ietf.org/html/rfc6531) – SMTP Extension for Internationalized Email Addresses (updates RFC 2821 (https://tools.ietf.org/html/rfc2821), RFC 2822 (https://tools.ietf.org/html/rfc2822), RFC 4952 (https://tools.ietf.org/html/rfc4952), and RFC 5336 (https://tools.ietf.org/html/rfc5336))
- RFC 8314 (https://tools.ietf.org/html/rfc8314) – Cleartext Considered Obsolete: Use of Transport Layer Security (TLS) for Email Submission and Access

List of supporting servers

- IceWarp
- Postfix – no patches needed for RFC 6531..RFC 6533.
- Sendmail – source code patch necessary for SMTPUTF8 support
- HMailServer – free mail server for Windows
- Exim
- MailEnable – support only in Enterprise Edition
- MagicMail – pipe-lining is intentionally not supported

List of supporting clients

- nmh (from version 1.7)

- Mozilla Thunderbird (from version 82.0) [48]

List of supporting content filters

- Amavis (SMTP and LMTP) since version 2.10.0[49]

See also

- Bounce address
- CRAM-MD5 (a SASL mechanism for ESMTPA) RFC 2195
- Email
  - Email encryption
- DKIM
- Ident
- List of mail server software
- List of SMTP server return codes
- POP before SMTP / SMTP after POP
- RFC 3516, Internet Message Access Protocol Binary Content Extension
- Sender Policy Framework (SPF)
- Simple Authentication and Security Layer (SASL) RFC 4422
- SMTP Authentication
- Variable envelope return path

Notes

1. The History of Electronic Mail (http://www.multicians.org/thvv/mail-history.html), *Tom Van Vleck*: *"It is not clear this protocol was ever implemented"*
2. *The First Network Email* (https://openmap.bbn.com/~tomlinso/ray/firstemailframe.html), Ray Tomlinson, BBN
3. Picture of "The First Email Computer (https://openmap.bbn.com/~tomlinso/ray/ka10.html)" by Dan Murphy, a PDP-10
4. Dan Murphy's TENEX and TOPS-20 Papers (http://www.opost.com/dlm/tenex/) Archived (https://web.archive.org/web/20071118204016/http://www.opost.com/dlm/tenex/) November 18, 2007, at the Wayback Machine
5. RFC 2235 (https://tools.ietf.org/html/rfc2235)
6. RFC 469 (https://tools.ietf.org/html/rfc469) – Network Mail Meeting Summary
7. RFC 524 (https://tools.ietf.org/html/rfc524) – A Proposed Mail Protocol
8. RFC 772 (https://tools.ietf.org/html/rfc772) – Mail Transfer Protocol
9. Tldp.org (http://tldp.org/HOWTO/Usenet-News-HOWTO/x64.html)
10. draft-barber-uucp-project-conclusion-05 – The Conclusion of the UUCP Mapping Project (https://tools.ietf.org/html/draft-barber-uucp-project-conclusion-05)
11. The article about sender rewriting contains technical background info about the early SMTP history and source routing before RFC 1123 (https://tools.ietf.org/html/rfc1123).
12. Eric Allman (1983), *Sendmail – An Internetwork Mail Router* (https://docs.freebsd.org/44doc/smm/09.sendmail/paper.pdf) (PDF), BSD UNIX documentation set, Berkeley: University of California, retrieved June 29, 2012

13. Craig Partridge (2008), *The Technical Development of Internet Email* (https://web.archive.org/web/20110512165437/http://www.ir.bbn.com/~craig/email.pdf) (PDF), IEEE Annals of the History of Computing, **30**, IEEE Computer Society, pp. 3–29, doi:10.1109/MAHC.2008.32 (https://doi.org/10.1109%2FMAHC.2008.32), S2CID 206442868 (https://api.semanticscholar.org/CorpusID:206442868), archived from the original (http://www.ir.bbn.com/~craig/email.pdf) (PDF) on May 12, 2011
14. John Klensin (October 2008). "Basic Structure" (https://tools.ietf.org/html/rfc5321#section-2.1). *Simple Mail Transfer Protocol* (https://tools.ietf.org/html/rfc5321). IETF. sec. 2.1. doi:10.17487/RFC5321 (https://doi.org/10.17487%2FRFC5321). RFC 5321 (https://tools.ietf.org/html/rfc5321). Retrieved January 16, 2016.
15. "The MAIL, RCPT, and DATA verbs" (http://cr.yp.to/smtp/mail.html), [D. J. Bernstein]
16. RFC 5321 (https://tools.ietf.org/html/rfc5321) Section-7.2
17. RFC 1985 (https://tools.ietf.org/html/rfc1985), *SMTP Service Extension for Remote Message Queue Starting*, J. De Winter, The Internet Society (August 1996)
18. Systems, Message. "Message Systems Introduces Latest Version Of Momentum With New API-Driven Capabilities" (https://www.prnewswire.com/news-releases/message-systems-introduces-latest-version-of-momentum-with-new-api-driven-capabilities-277568381.html). *www.prnewswire.com*. Retrieved July 19, 2020.
19. Cara Garretson (2005). "ISPs Pitch In to Stop Spam" (http://www.pcworld.com/article/116843/article.html). *PC World*. Retrieved January 18, 2016. "Last month, the Anti-Spam Technical Alliance, formed last year by Yahoo, America Online, EarthLink, and Microsoft, issued a list of antispam recommendations that includes filtering Port 25."
20. RFC 5321 (https://tools.ietf.org/html/rfc5321), *Simple Mail Transfer Protocol*, J. Klensin, The Internet Society (October 2008)
21. RFC 1047 (https://tools.ietf.org/html/rfc1047)
22. rfc5321#section-4.5.3.2.6 (https://tools.ietf.org/html/rfc5321#section-4.5.3.2.6)
23. John Klensin; Ned Freed; Marshall T. Rose; Einar A. Stefferud; Dave Crocker (November 1995). *SMTP Service Extensions* (https://tools.ietf.org/html/rfc1869). IETF. doi:10.17487/RFC1869 (https://doi.org/10.17487%2FRFC1869). RFC 1869 (https://tools.ietf.org/html/rfc1869).
24. "MAIL Parameters" (https://www.iana.org/assignments/mail-parameters/mail-parameters.txt). IANA. Retrieved April 3, 2016.
25. Which was obsoleted in 2011 by RFC 6152 (https://tools.ietf.org/html/rfc6152) corresponding to the then new STD 71
26. "MAIL Parameters" (https://www.iana.org/assignments/mail-parameters/mail-parameters.txt). November 15, 2018.
27. Jiankang Yao (December 19, 2014). "Chinese email address" (http://www.ietf.org/mail-archive/web/ima/current/msg05395.html). *EAI* (Mailing list). IETF. Retrieved May 24, 2016.
28. "SMTP Service Extension Parameters" (https://www.iana.org/assignments/mail-parameters/mail-parameters.txt). IANA. Retrieved November 5, 2013.
29. James Server - ChangeLog (http://james.apache.org/server/2.3.0/changelog.html). James.apache.org. Retrieved on 2013-07-17.
30. 8BITMIME service advertised in response to EHLO on gmail-smtp-in.l.google.com port 25, checked 23 November 2011
31. Qmail bugs and wishlist (https://archive.is/20120630212728/http://home.pages.de/~mandree/qmail-bugs.html). Home.pages.de. Retrieved on 2013-07-17.
32. The 8BITMIME extension (http://cr.yp.to/smtp/8bitmime.html). Cr.yp.to. Retrieved on 2013-07-17.
33. "*Postfix SMTPUTF8 support is enabled by default*" (http://www.postfix.org/SMTPUTF8_README.html), February 8, 2015, postfix.org

34. "Message Systems Introduces Latest Version Of Momentum With New API-Driven Capabilities" (http://www.prnewswire.com/news-releases/message-systems-introduces-latest-version-of-momentum-with-new-api-driven-capabilities-277568381.html) (Press release).
35. "Version 6.2 Revision History" (https://communigate.com/Communigatepro/History62.html#6.2.2). *CommuniGate.com*.
36. Sam Varshavchik (September 18, 2018). "New releases of Courier packages" (https://sourceforge.net/p/courier/mailman/message/36417878/). *courier-announce* (Mailing list).
37. changelog (https://github.com/halon/changelog)
38. "MS-OXSMTP: Simple Mail Transfer Protocol (SMTP) Extensions" (https://interoperability.blob.core.windows.net/files/MS-OXSMTP/%5bMS-OXSMTP%5d-180724.docx). July 24, 2018.
39. "EAI Readiness in TLDs" (https://uasg.tech/wp-content/uploads/2019/02/UASG021D-EN-EAI-Readiness-in-TLDs.pdf) (PDF). February 12, 2019.
40. "Communications Messaging Server Release Notes" (https://docs.oracle.com/communications/E72263_01/doc.802/e72267/msvrn.htm#BAJGIBHB). *oracle.com*. October 2017.
41. "Introducing MTA Strict Transport Security (MTA-STS) | Hardenize Blog" (https://www.hardenize.com/blog/mta-sts). *www.hardenize.com*. Retrieved April 25, 2019.
42. "STARTTLS Everywhere" (https://starttls-everywhere.org/). EFF. Retrieved August 15, 2019.
43. Cimpanu, Catalin. "Gmail becomes first major email provider to support MTA-STS and TLS Reporting" (https://www.zdnet.com/article/gmail-becomes-first-major-email-provider-to-support-mta-sts-and-tls-reporting/). *ZDNet*. Retrieved April 25, 2019.
44. Message Non Compliant with RFC 5322 (https://support.google.com/mail/?p=RfcMessageNonCompliant)
45. Message could not be delivered. Please ensure the message is RFC 5322 compliant. (https://answers.microsoft.com/en-us/outlook_com/forum/oemail-ocompose/message-could-not-be-delivered-please-ensure-the/87a52762-7d08-467e-85a2-120721c2dd8e?auth=1)
46. Why are the emails sent to Microsoft Account rejected for policy reasons? (https://answers.microsoft.com/en-us/outlook_com/forum/oemail-osend/why-are-the-emails-sent-to-microsoft-account/b64e3e4a-0d93-40c8-8e28-4be849012f9c)
47. "NGINX Docs | Configuring NGINX as a Mail Proxy Server" (https://docs.nginx.com/nginx/admin-guide/mail-proxy/mail-proxy/).
48. https://bugzilla.mozilla.org/show_bug.cgi?id=1563891
49. Martinec, Mark. "ANNOUNCE: amavisd-new-2.10.0 has been released" (https://lists.amavis.org/pipermail/amavis-users/2014-October/003252.html). Retrieved October 1, 2019.

References

- Hughes, L (1998). *Internet E-mail: Protocols, Standards and Implementation*. Artech House Publishers. ISBN 978-0-89006-939-4.
- Hunt, C (2003). *sendmail Cookbook*. O'Reilly Media. ISBN 978-0-596-00471-2.
- Johnson, K (2000). *Internet Email Protocols: A Developer's Guide*. Addison-Wesley Professional. ISBN 978-0-201-43288-6.
- Loshin, P (1999). *Essential Email Standards: RFCs and Protocols Made Practical*. John Wiley & Sons. ISBN 978-0-471-34597-8.
- Rhoton, J (1999). *Programmer's Guide to Internet Mail: SMTP, POP, IMAP, and LDAP*. Elsevier. ISBN 978-1-55558-212-8.
- Wood, D (1999). *Programming Internet Mail* (https://archive.org/details/livesofcaptivere00petz). O'Reilly. ISBN 978-1-56592-479-6.

External links

- IANA registry of mail parameters (https://www.iana.org/assignments/mail-parameters) includes service extension keywords
- RFC 1869 (http://www.ietf.org/rfc/rfc1869.txt) SMTP Service Extensions
- RFC 5321 (http://www.ietf.org/rfc/rfc5321.txt) Simple Mail Transfer Protocol
- RFC 4954 – SMTP Service Extension for Authentication (obsoletes RFC 2554)
- RFC 3848 – SMTP and LMTP Transmission Types Registration (with ESMTPA)
- RFC 6409 – Message Submission for Mail (obsoletes RFC 4409, which obsoletes RFC 2476)