Email Fundamentals – Systems Architecture

This article describes the high-level "architecture" of systems for handling Internet mail, and does not discuss the protocols used in a message transfer [Klensin08], the format of the messages [Resnick08], or any of the protocols and services at the "relay-level" of a mail handling system [Crocker08]. The cited references offer a very complete discussion of these related topics. Good elementary discussions can be found in most texts on computer networks [PnD07].

Internet mail has evolved without much central planning to a collection of very diverse and astonishingly complex systems. Like the Internet itself, it is helpful to study these systems the way a biologist would study an organism, or a social scientist the behavior of a group. Who are the Actors in a typical email system? What are their roles and responsibilities in handling the mail? What are their relationships with each other and with users? What are their motivations? How can we build better security systems?

A typical mail handling system has a network of Relays[1], each temporarily storing the message, performing some specialized function, and passing it on to the next Relay using the SMTP protocol. You can tell how many Relays handled a message by looking at the Received lines in the message header. There should be one for each Relay.

Figure 1 shows a typical system with the Relays grouped into functional blocks. In this diagram, we have named the blocks by the role they play in processing a message, and assigned each role to a different Actor (User or Agent). However, each Actor can have multiple blocks, each block can have multiple hosts, and each host can have multiple Relays running as independent daemon processes. A Transmitter might have a dozen Relays, operating in parallel to handle a large mailflow, or widely dispersed to serve users all over the world. An MDA might have a process dedicated to managing a large mailstore, another running a POP/IMAP server, and another providing a webmail interface.

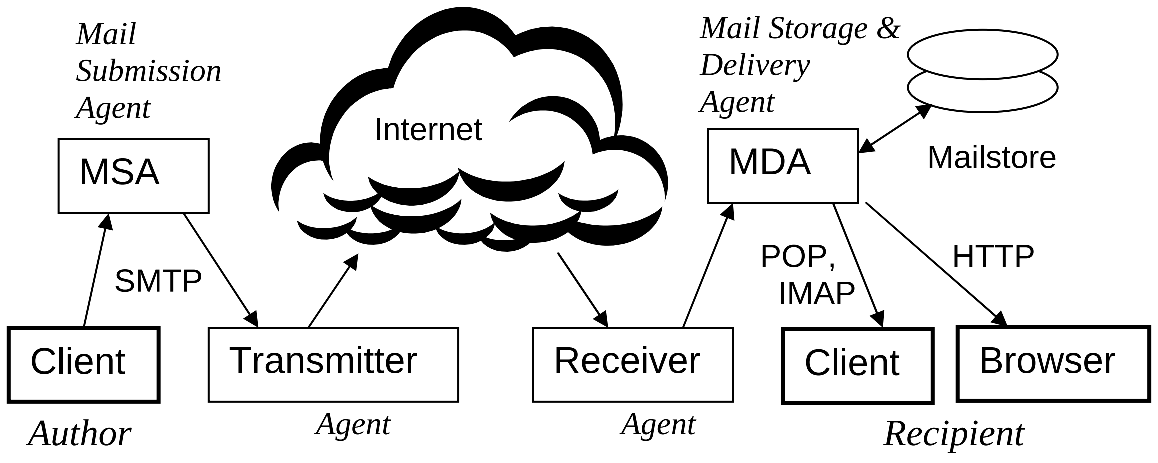

Figure 1 Actors (Users and Agents) and roles (functional blocks) in a typical email system.

To understand a mail handling system, including its security vulnerabilities, we need to focus on the roles and responsibilities of each Actor and the relationships between them. Figure 1 is a simplified model of just one system. There are many other possibilities. We might add a Forwarder between the Receiver and the MDA, or an Open Relay floating in the cloud. We

[1] Do not confuse SMTP Relays with routers or packet switches. Relays use SMTP/TCP/IP, and the functionality of routers is entirely encapsulated within the IP layer of this protocol stack. We can ignore routers in this discussion. They are "transparent" to SMTP.

might join two blocks under one Agent. We might add another layer of organization, showing a group of Actors organized as a Mail Receiving Network [Moore05], or an ADMD (Administrative Management Domain) [Crocker08]. Yet another layer could be shown by grouping the Relays according to who controls the equipment.[2]

A diagram like Figure 1 could get quite complex. A shorthand notation [MacQuigg08] will allow us to show the relevant networks, actors, roles, and relationships. Here is a basic system with four Actors (two Users and two Agents), organized as two networks[3]:

```
  |--- Sender's Network ---|           |-- Recipient's Network -|
                                  /
  Author ==> MSA/Transmitter --> / --> Receiver/MDA ==> Recipient
                                /
                             Border
```

The double arrow shows a direct relationship between Actors (e.g. a contract between the Author and his Email Service Provider (**ESP**)). The single arrow shows only the direction of mail flow. There is no relationship between Agents across the Border to the open Internet. The / shows multiple roles being played by one Actor. Using these diagrams, we can model almost any system, and include a lot of detail on relationships, but not lose the simplicity of Figure 1. The elements of the model (Actor's roles) are the fundamental building blocks.[4]

Here is an extension of the basic system, adding a Forwarder role, played by the same Actor as the Receiver. Both the Receiver/Forwarder and the MDA have a direct relationship with the Recipient, so they have an indirect relationship with each other. These details are important in discussions of authentication protocols.

```
                |-------- Recipient's Network ---------|
           /
      --> / --> Receiver/Forwarder ~~> MDA ==> Recipient
         /
       Border
```

If we wonder why email continues to be such an insecure system, we can study this last example. Authentication protocols that try to correlate the Transmitter's domain name to the connecting IP address can fail when a Forwarder is involved. We cannot just dismiss Forwarding as an "edge case", however. It is important for a user who changes jobs or ESPs, and would like to continue receiving mail at his old address.

Let's follow a message from start to finish. The scenario begins with an Author composing a message using his mail client. There are countless mail clients available, just like there are many web browsers to choose from. In fact, most web browsers now include a mail client, or at least a mechanism to invoke the user's preferred client when he clicks a mailto: link in a webpage.

When the Author clicks SEND, his mail client connects to an MSA at his ESP, and the message is transferred using SMTP. A key responsibility of the MSA is to authenticate the Author. This can

[2] These are the "Autonomous Systems" of the physical network. As with routers, however, it is much simpler to think of this layer as transparent to the level we are modeling (even if the Actors in our model happen to be also network owners). This is a frequent point of confusion, particularly with people who would like to hold network owners responsible for the content of the messages they carry.
[3] These "networks" are of Actors and their relationships, not routers and data links. To avoid confusion we could use the more precise, but less familiar terminology proposed by Crocker – ADMD.
[4] The roles of Transmitter, Receiver, and Forwarder can be abbreviated as XMTR, RCVR, and FWDR. This will make the diagrams more compact, and avoid any confusion between roles and actors. In this discussion, however, there is little confusion, and the precise terminiology is too awkward. Thus, when we say "authenticate the Transmitter" we mean "authenticate the Agent playing the role of XMTR".

be done with a password, by assigning the client a static IP address, or by having the client on the MSA's local network, not directly connected to the Internet.

Most large ESPs operate their own transmitter Relays, but smaller companies, and organizations with a lot of bulk mail, often subcontract this specialized role to another Agent. The Transmitter's responsibilities include prevention of outgoing spam, and providing some means to prove their identity to unrelated Receivers. It isn't enough to say "HELO, this is trustme.com". Any spammer can do that. The Transmitter must provide some "out-of-band" data using a service like DNS that is more trusted than email.

??? <delete> Email authentication methods fall into two categories. Methods like SPF, SenderID, and CSV rely on the fact that certain IP addresses are firmly under the control of a sender (an individual or organization identified by its domain name). Methods like DKIM rely on a digital signature applied to the entire message and most of its headers. Both depend on the security of DNS. Only the domain owner has access to the DNS records under his name. With IP-based methods, the sender publishes in DNS the IP addresses authorized to use his domain name. With signature-based methods, the sender publishes a public key. IP methods can be very efficient, rejecting an entire session without transferring any messages. End-to-end signature methods can be very secure, even with an un-trusted Forwarder in the middle. </delete>

The Receiver's responsibilities include a number of functions we might call "Border defense" – blocking a DoS attack, authenticating the sender, and various spam-blocking strategies, including whitelisting, blacklisting, statistical analysis of message content, and use of heuristic rulesets that have proven effective in separating spam from legitimate mail. Border defense should be done at the Border. Loss of mail due to violations of this principle is common. A forwarded message can look like a forgery, and then the MDA has a tough choice – drop the message with no notice to the alleged sender, or send the notice and risk being reported for "bounce spam".

The problems with mis-configured mailsystems can be avoided if all Actors understand how the system works. When a Recipient sets up forwarding from his old Receiver/Forwarder to his new MDA, he should make sure that the Forwarder is whitelisted by the MDA. Forwarders should make sure that Recipients (non-expert users) understand this. MDAs should understand that forwarding is a common need, and make it easy for Recipients to whitelist at the domain level. Simple models will help reduce the confusion.

References

[Crocker08] D. Crocker, "Internet Mail Architecture", 2008, http://tools.ietf.org/html/draft-crocker-email-arch-10.

[Klensin08] J. Klensin, ed, "Simple Mail Transfer Protocol", RFC-5321, 2008, http://tools.ietf.org/html/rfc5321.

[MacQuigg08] D. MacQuigg, "Models for Mail Handling Systems", 2008, http://open-mail.org/MHSmodels.html

[Moore05] K. Moore, "Recommendations for Submission of Email and Relaying of Email Between Mail Networks", 2005, http://www.cs.utk.edu/~moore/opinions/email-submission-recommendations.html

[PnD07] L. Peterson, B. Davie, "Computer Networks: A Systems Approach", 4th ed. 2007, Sect. 9.1.1 "Electronic Mail".

[Resnick08] P. Resnick, ed. "Internet Message Format", RFC-5322, 2008, http://tools.ietf.org/html/rfc5322