Email: SMTP

Let's now discuss some important protocols that make email what it is.

WE'LL COVER THE FOLLOWING

Introduction

Email has been **a key application** of the Internet since its early days.

Today, almost **all formal or official correspondences occur on email**. In fact, a lot of major corporations communicate primarily through email both externally and internally despite the advent of instant messaging.

Let’s see how this evergreen application really works!

SMTP

There are many protocols associated with email. One popular choice is a combination of `POP3` and `SMTP`. One is used to send emails that are stored in a user’s inbox and the other is used to retrieve emails sent to you. However, the very core of electronic mail is the **Simple Mail Transfer Protocol (SMTP)**.

SMTP uses TCP, which means that transfers are **reliable**. The connection is established at `port 25`.

> **Note** A good mnemonic to remember what SMTP does is **S**ending **M**ail **T**o **P**eople.

Also, for ease and consistency, we are defining **User Agents** as agents that allow users to compose, view, delete, reply to, and forward emails. Applications such as Apple Mail, Microsoft Outlook, and Gmail's webmail are examples of user agents.

History of SMTP

Let's delve a bit into the history of SMTP which is incredibly important to understand why it's designed the way it is.

SMTP predates HTTP by quite a margin and therefore has some antiquated design properties.

For example, **all SMTP messages *have* to be encoded into 7-bit ASCII**. This made sense in the early days when computer networks did not have the capacity to email large images, audio, or videos, and when email was primarily text that could fit in ASCII characters. We needed 7-bit encoding and decoding because mostly US-ASCII characters were being used in which the MSB (most significant bit) of the byte was 0, so there was no point in sending 8 bits per character. Instead 7 bits per character were transmitted.

Sending text with characters that require a greater number of bits per character, or binary data became challenging. Therefore, all email attachments are encoded into 7-bit ASCII even today when sending, and then decoded upon receiving. This requires additional computational work.

How SMTP Works

Let's look at how this ubiquitous protocol works.

1. When an email is sent, its sent to the sender's **SMTP server** using the SMTP protocol.
    - The SMTP server is configured in your email client. The general

format of the domain of the SMTP server is `smtp.example.com` where

the main email address of the sender is `user@example.com`. But it's not mandatory to adhere to this format. We could set up, say, `zeus.example.com` to serve as our SMTP server, if we wanted. From a security point of view, it is probably a good idea, since people are unlikely to guess it as easily.

2. The **email is now placed on a message queue in the sending SMTP server**.

3. Then, the SMTP server initiates a connection with the recipient server and will conduct an initial SMTP handshake.

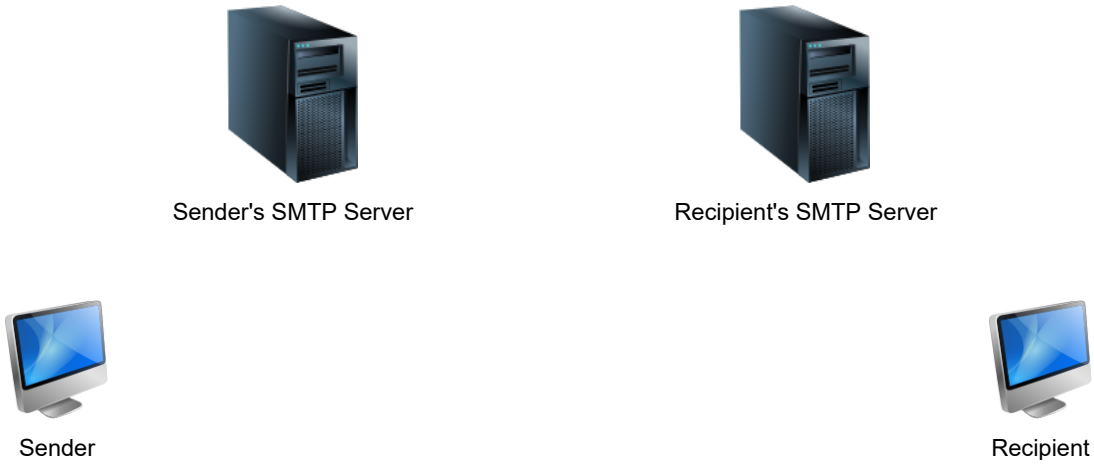
    - If the recipient is on the same SMTP server as the sender (for instance alice@gmail.com sending to bob@gmail.com), then the SMTP server doesn't need to connect to the recipient's server.

4. The SMTP server will finally send the message to the recipient's email server.

5. The **email is then downloaded from the recipient's SMTP server** using other protocols when the recipient logs in **to their email account or 'user agent**.' In other words, the recipient's SMTP server copies the email to the recipient's mail-box.

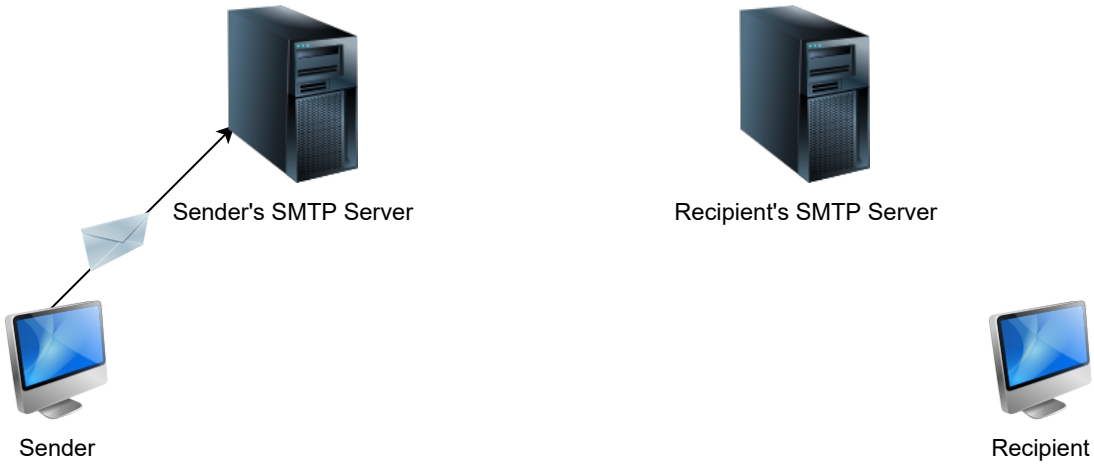
>  **Note SMTP is a push protocol** because the email client sends the email out to the server when it needs to. Which means it only *sends* data to servers. Other protocols called **Mail Access Protocols** such as **POP** and **IMAP** are used for *getting* email from a server and are called *pull* protocols because the client asks their POP/IMAP server if they have any new messages whenever they feel like.

Have a look at the following slides to get a clearer picture.



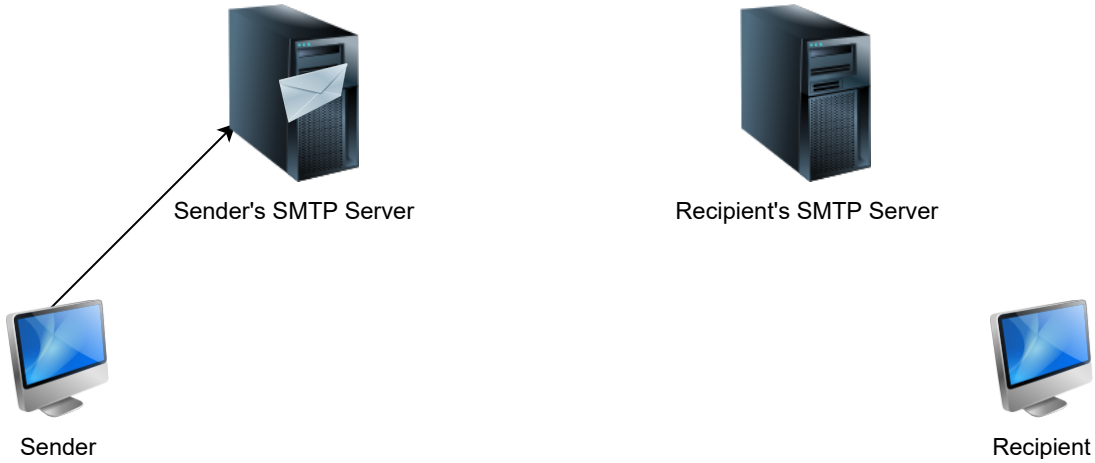
The sender wants to send the receiver an email

1 of 7

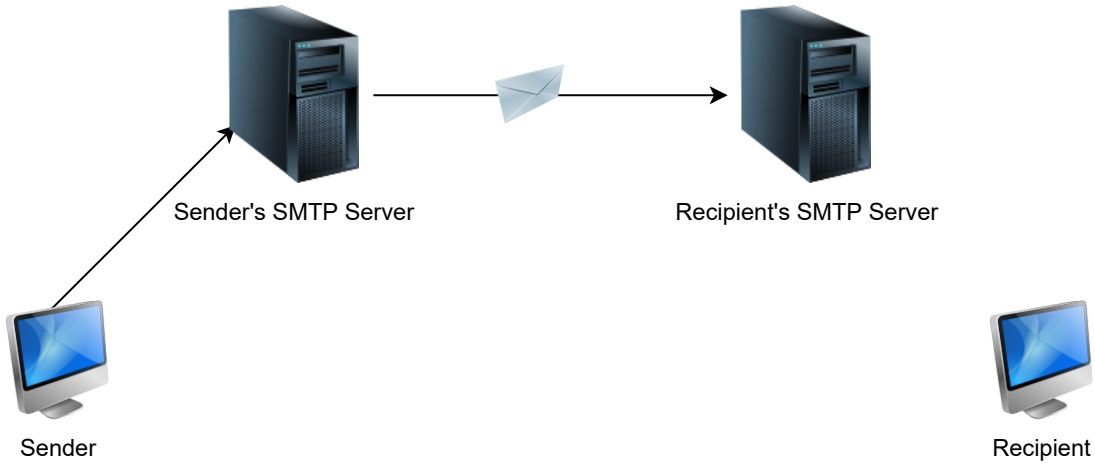


The sender composes the email and sends it

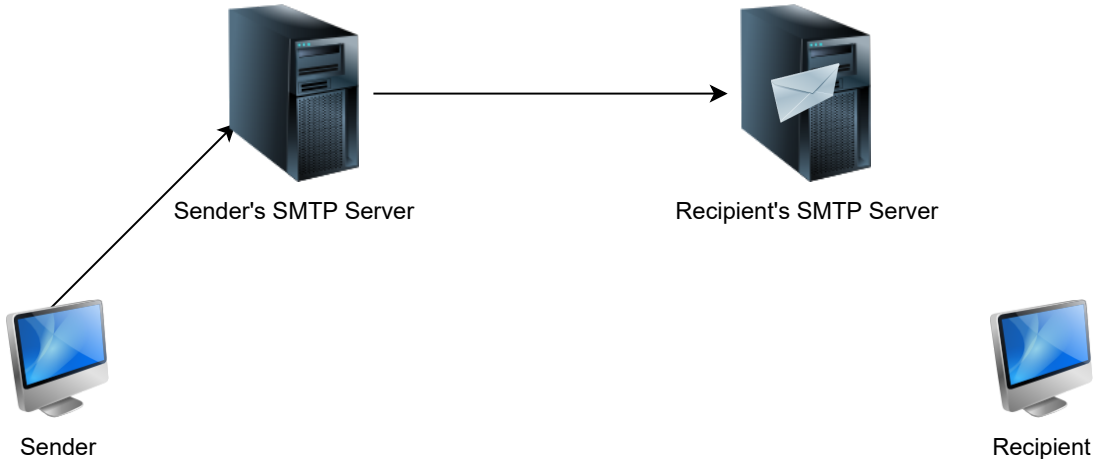
2 of 7



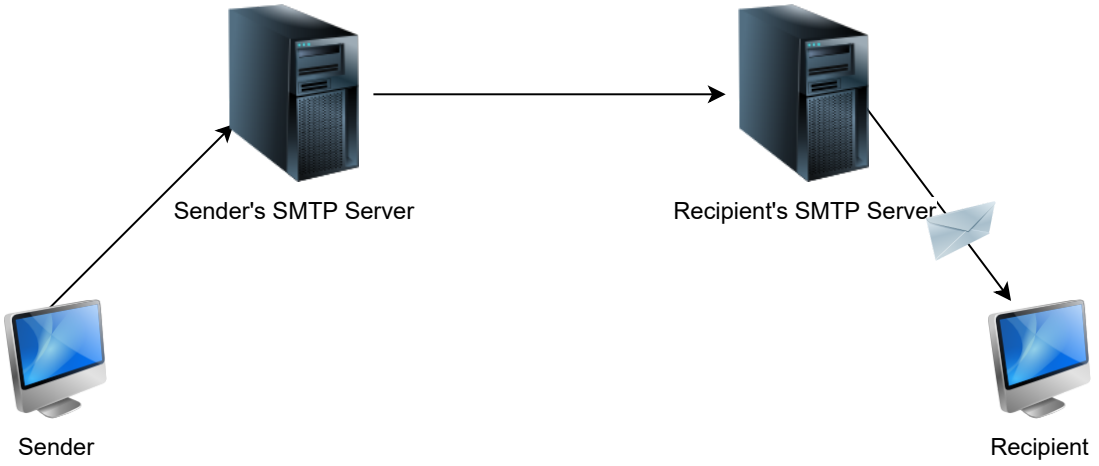
The email resides in a queue on the sender's SMTP server.



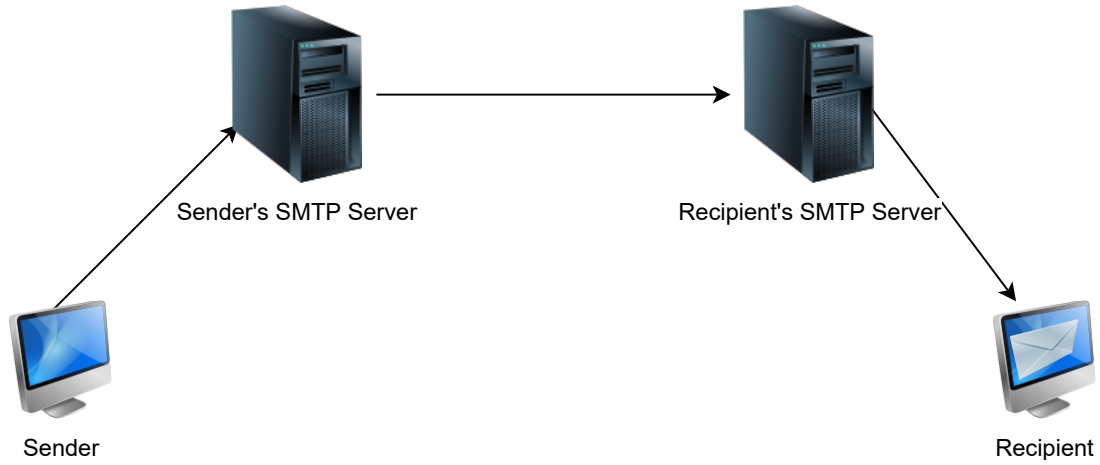
After opening an initial connection and conducting an SMTP handshake with the receipient's server, the sender's server sends the email.



The email resides on the recipient's server in the recipient’s inbox. The email will be downloaded to the recipient's user agent once they log in.



The recipient logs in and receives the email.



The recipient logs in and receives the email.

7 of 7

Error Handling

There are many scenarios where sending an email may fail. Here are a few.

- **If the email is not sent** for any reason such as a misspelled recipient address, the email is returned to the sender with a **“not delivered” message**.
- **If the recipient SMTP server is offline**, the sending SMTP server **keeps trying to send** the email, say, every 30 minutes or so. It **stops trying after a few days and alerts the sender about the mail not delivered error**.
- Also, SMTP does not use any intermediate servers. So, even if an attempt to send an email fails because of any reason such as the receiving server being down, the email won’t be stored on an intermediary server. It will be stored on the sending SMTP server.

Quick Quiz!

1 SMTP is a pull protocol

COMPLETED 0%

1 of 2

Let’s do an exercise with another popular command-line tool to check what your domain’s mail server is in the next lesson!