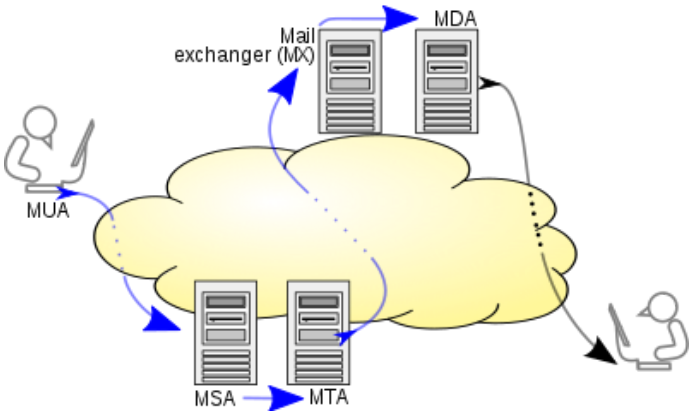
Protocolo simple de transferencia de correo(SMTP)

Ignacio Borassi y Valentino Aprea

En 1982 se diseñó el primer sistema basado en SMTP para intercambiar correos electrónicos en ARPANET,
Su estructura se basa en el modelo cliente-servidor, donde un cliente envía un mensaje a uno o varios servidores para recibir una respuesta. La comunicación entre el cliente y servidor consiste enteramente en líneas de texto compuestas por caracteres Unicode, antes estaba compuesto por caracteres ASCII.
Las respuestas del servidor constan de un código numérico de tres dígitos, seguido de un texto explicativo. El número va dirigido a un procesado automático de la respuesta por autómata, mientras que el texto permite que un humano interprete la respuesta.
En el protocolo SMTP todas las órdenes, réplicas o datos son líneas de texto, son delimitadas por el carácter <CRLF>. Todas las réplicas tienen un código numérico al comienzo de la línea central.

Modelo de procesamiento de correo



El correo electrónico es presentado por un cliente de correo(MUA, agente de usuario de correo) a un servidor de correo(MSA, agente de sumisión de correo) usando SMTP a través del puerto 587. Una gran parte de los proveedores de correo todavía permiten el envío a través del puerto 25. El MSA entrega el correo a su agente de transferencia(MTA). Estos dos agentes son casos diferentes pero provienen del mismo software.
El procesamiento local que se presenta puede ser realizado en una máquina o partido entre varias aplicaciones, en el segundo caso, los procesos complicados pueden compartir archivos. Aquí SMTP es usado para la transferencia de mensajes internamente, con cada uno de los hosts configurados para usar la siguiente aplicación como un anfitrión elegante. Para lograr la localización del servidor objetivo, el MTA divisorio tiene que usar DNS para lograr la búsqueda del registro interno de cambiado de correo conocido como registro MX para la esfera del recipiente. En ese instante es cuando el registro de MX devuelto contiene el nombre del anfitrión objetivo.
Luego el MTA se une al servidor de cambio como un cliente SMTP. Una vez que MX acepta el mensaje entrante, este se lo da a un agente de entrega de correo(MDA) para después ser llevado a la entrega de correo local. El MDA además de entregar mensajes también es capaz de guardar mensajes en un buzón de formato, y la recepción de correo puede ser realizada usando muchas computadoras. Hay 2 formas en que un MDA puede entregar mensajes: ya sea enviandolos directamente al almacenamiento, o expedirlos sobre una red usando SMTP. Una vez entregado al servidor de correo local, dicho correo es almacenado

para la recuperación. Su recuperación se logra a través de aplicaciones de usuario final, conocidas como cliente de correo electrónico, usando el protocolo de acceso de mensaje de internet(IMAP), este protocolo que facilita tanto el acceso para enviar, como el manejo de correo almacenado.

Protocolo

El protocolo simple de transferencia de correo (SMTP) es un protocolo TCP/IP que se utiliza para enviar y recibir correo electrónico. Normalmente se utiliza con **POP3** o con el protocolo de acceso a mensajes de Internet (**IMAP**) para guardar mensajes en un buzón del servidor y descargarlos periódicamente del servidor para el usuario.
Este protocolo pertenece al nivel de aplicación del modelo TCP/IP, utiliza el protocolo de la capa de transporte TCP, y hace uso de diferentes puertos dependiendo de si el tráfico va cifrado o no va cifrado:

- Puerto 25 TCP para tráfico sin cifrar.
- Puerto 465 TCP para tráfico cifrado SSL (SMTPS).
- Puerto 587 TCP como puerto alternativo para SMTPS con TLS.

Actualmente la gran mayoría de proveedores de servicios de correo electrónicos disponen de soporte para SSL/TLS, con la finalidad de cifrar y proteger todos los datos que se envían en el correo electrónico, por tanto, casi siempre vamos a hacer uso de los puertos 465 y 587, dependiendo de cómo esté configurado el servidor de correo. Por ejemplo, en el caso de Gmail no se permite utilizar el puerto 25 TCP porque no tiene ningún tipo de cifrado, sin embargo, soporta tanto el puerto 465 para conexiones SSL como el puerto 587 para conexiones TLS.
El protocolo SMTP está orientado al envío de correos electrónicos, subida o saliente, desde el cliente del correo local hasta el servidor del correo, para después enviarlo a su destinatario final.
También permite realizar la autenticación con usuario/contraseña en texto a partir del puerto 25, pero pocos servicios soportan este protocolo sin utilizar ningún tipo de cifrado, con el objetivo de proporcionarnos esta confidencialidad de autenticación y también al correo electrónico enviado. Con las últimas versiones de SMTP se pueden usar los puertos 465 o 587 para tener cifrado de datos, autenticidad y comprobación de la integridad de los mensajes enviados.

Funcionamiento e intercambio de mensajes.

El funcionamiento del protocolo SMTP está orientado a conexión basado en texto, es decir, es un protocolo fiable al utilizar una capa de transporte TCP. El cliente de correo se comunica con el servidor a través de una serie de secuencias de comandos para realizar la autenticación, el envío de mensajes y cerrar la conexión. Y el servidor le contestara a través de una serie de secuencia de comandos. En la misma sesión de SMTP se pueden incluir 0 o más transacciones, en cada una de estas tendremos un total de 3 secuencias de comandos/respuestas:

- MAIL: Establece la dirección de retorno.
- RCPT: Establece un destinatario del mensaje, puede emitirse varias veces dependiendo la cantidad de destinatarios.

- DATA: Es el contenido del propio correo electrónico, Se compone por una cabecera y cuerpo de mensaje. Este en realidad en un grupo de comandos, y el servidor responde dos veces: una vez para el comando de datos adecuado, para reconocer que está listo para recibir el texto, y la segunda vez después de la secuencia final de datos, para aceptar o rechazar todo el mensaje.

Una vez que configuramos el cliente de correo correctamente, el email se redactará directamente en el propio cliente de correo, cuando se le da al botón enviar es cuando empieza el proceso.

1. El cliente establece la conexión con el servidor SMTP esperando contestación de HELO para recibir la identificación del servidor.
2. El cliente empezará la comunicación con la orden MAIL FROM con la dirección de email y el servidor comprobará que el origen es válido.
3. El cliente le enviará un mensaje RCPT TO incorporando el email de destino del correo electrónico, dependiendo de los destinatarios tendremos un RCPT TO o varios. Después se envía una orden DATA para avisar que viene el cuerpo del mensaje línea a línea.
4. Si el cliente no va a enviar más emails. enviará una orden QUIT para cerrar la sesión SMTP.
5. En caso de enviar otro se repetiría todo este proceso.

Comandos

- HELO, para abrir una sesión con el servidor
- EHLO, para abrir una sesión, en el caso de que el servidor soporte extensiones definidas en el RFC 1651
- MAIL FROM, para indicar quien envía el mensaje
- RCPT TO, para indicar el destinatario del mensaje
- DATA, para indicar el comienzo del mensaje, éste finalizará cuando haya una línea únicamente con un punto.
- QUIT, para cerrar la sesión
- RSET Aborta la transacción en curso y borra todos los registros.
- SEND Inicia una transacción en la cual el mensaje se entrega a una terminal.
- SOML El mensaje se entrega a un terminal o a un buzón.
- SAML El mensaje se entrega a un terminal y a un buzón.
- VRFY Solicita al servidor la verificación de todo un argumento.
- EXPN Solicita al servidor la confirmación del argumento.
- HELP Permite solicitar información sobre un comando.
- NOOP No decir nada, se emplea para mantener la sesión abierta
- TURN Solicita al servidor que intercambien los papeles.

De los tres dígitos del código numérico, el primero indica la categoría de la respuesta, estando definidas las siguientes categorías:

- 2XX, la operación solicitada mediante el comando anterior ha sido concluida con éxito
- 3XX, la orden ha sido aceptada, pero el servidor está pendiente de que el cliente le envíe nuevos datos para terminar la operación
- 4XX, para una respuesta de error, pero se espera a que se repita la instrucción
- 5XX, para indicar una condición de error permanente, por lo que no debe repetirse la orden

Seguridad y spam

Una de las limitaciones del SMTP original es que no facilita métodos de autenticación a los emisores, así que se definió la extensión SMTP-AUTH en RFC 2554.
A pesar de esto, el spam es aún el mayor problema. No se cree que las extensiones sean una forma práctica para prevenirlo.

Los RCF asociados

- RFC 1123 – Requirements for Internet Hosts—Application and Support (STD 3)
- RFC 1870 – SMTP Service Extension for Message Size Declaration (deja obsoleto a: RFC 1653)
- RFC 2505 – Anti-Spam Recommendations for SMTP MTAs (BCP 30)
- RFC 2920 – SMTP Service Extension for Command Pipelining (STD 60)
- RFC 3030 – SMTP Service Extensions for Transmission of Large and Binary MIME Messages
- RFC 3207 – SMTP Service Extension for Secure SMTP over Transport Layer Security (deja obsoleto a RFC 2487)
- RFC 3461 – SMTP Service Extension for Delivery Status Notifications (deja obsoleto a RFC 1891)
- RFC 3463 – Enhanced Status Codes for SMTP (deja obsoleto a RFC 1893)
- RFC 3464 – An Extensible Message Format for Delivery Status Notifications (deja obsoleto a RFC 1894)
- RFC 3798 – Message Disposition Notification
- RFC 3834 – Recommendations for Automatic Responses to Electronic Mail
- RFC 4952 – Overview and Framework for Internationalized E-mail
- RFC 4954 – SMTP Service Extension for Authentication (deja obsoleto a RFC 2554)
- RFC 5068 – E-mail Submission Operations: Access and Accountability Requirements (BCP 134)
- RFC 5321 – The Simple Mail Transfer Protocol (deja obsoleto a RFC 821 aka STD 10, RFC 974, RFC 1869, RFC 2821)
- RFC 5322 – Internet Message Format (deja obsoleto a RFC 822 aka STD 11, and RFC 2822)
- RFC 5336 – SMTP Extension for Internationalized Email Addresses (actualiza RFC 2821, RFC 2822 y RFC 4952)
- RFC 5504 – Downgrading Mechanism for Email Address Internationalization
- RFC 6409 – Message Submission for Mail (deja obsoleto a RFC 4409, RFC 2476)
- RFC 6522 – The Multipart/Report Content Type for the Reporting of Mail System Administrative Messages (deja obsoleto a RFC 3462, y a su vez a RFC 1892)

Implementación:

Nosotros creamos dos maneras para levantar o utilizar un servidor SMTP.

En la primera creamos un código de Python que utiliza cualquier servidor público de SMTP, por ejemplo el de Google (Véase el archivo de Python adjuntado en el repositorio). En ese caso utilizamos las funciones de Python de la Biblioteca: MIME. Aclaramos la dirección de Mail que se va a utilizar, el asunto, la contraseña y el destinatario. El código va a enviarle al servidor y puerto que aclaremos la información para que este correo se envíe.

Y también levantamos un servidor en sí mismo, no solo la forma de usarlo. Usamos Ubuntu y SquirrelMail.

sudo su
apt-get install apache2 *1
apt-get install mailutils *2
{configurar nombre del servidor de correo} —- Aca pondremos lo que va después del @
apt-get install vim — Vim es un editor de texto, podríamos haber usado NANO
vim /etc/postfix/main.cf

{ultima linea modificación = "ipv4"
home_mailbox=Maildir/ tecla escape :wq} Aca configuramos donde vamos a recibir los mails, ponemos MailDir así llegan a esa dirección que es un formato de spool de correo electrónico que no bloquea los ficheros para mantener la integridad del mensaje.

/etc/init.d/postfix reload {reiniciar postfix para aplicar cambios}
apt-get install courier-pop { instalacion de protocolo pop - "no"}
apt-get install courier-imap {protocolo - imap para poder acceder desde cualquier lugar a mi servidor de correo (Dentro de la misma red) }
apt-get install squirrelmail {Aplicación para tener la webapp de mail, nos creara toda la interfaz}

squirrelmail-configure
opción D
"courier"
opción 2
opción 1

"nombre del servidor de correo ejemplo nachito.com" (Nachito yo, no vos porfe)
tecla S para guardar
tecla Q para cerrar

ln -s /usr/share/squirrelmail/ /var/www/webmail {crear directorio de acceso directo a nuestro webmail}
ls /var/www/ {verificar}

vim /etc/apache2/sites-available/000-default.conf {cambiar directorio root de apache}
Línea DocumentRoot /var/www/html {se elimina html tecla esc :wq}

/etc/init.d/apache2 restart { reiniciar nuestro servicio de apache para aplicar cambios}
{Verificamos si nuestra aplicación de correo funciona ingresando a un navegador localhost/webmail}

maildirmake Maildir
adduser user1
añadir contraseña para el usuario
usuario1

Bibliografía

https://www.redeszone.net/tutoriales/internet/que-es-protocolo-smtp-email-configuracion/
https://es.wikipedia.org/wiki/Protocolo_para_transferencia_simple_de_correo
https://www.ibm.com/docs/es/i/7.3?topic=information-smtp
https://www.youtube.com/watch?v=r4HG462inEI

*1 Apache es un servidor web de código abierto, multiplataforma y gratuito.
Este webserver es uno de los más utilizados en el mundo, actualmente el 43% de los sitios webs funcionan con él. En este caso lo usamos para hostear nuestra aplicación de mail para que todos los usuarios en nuestra red puedan usarlo.

*2 Mailutils es una navaja suiza del manejo de correo electrónico. Ofrece un amplio conjunto de utilidades y herramientas para procesar el correo electrónico.

Todos los programas de Mailutils pueden operar en buzones de correo de cualquier formato existente, desde buzones UNIX estándar, pasando por maildir y hasta buzones remotos, a los que se accede de forma remota mediante IMAP4, POP3 y SMTP. Mailutils será la manera que usaremos para configurar el servidor SMTP en sí.