

17. Correos electrónicos

Wednesday, November 4, 2020 8:16 AM

En ocasiones, para algunas tareas en Ciberseguridad y en TI en general, requerimos contar con herramientas para el manejo automatizado de correo electrónico.

Para el envío de correo electrónico usaremos el protocolo SMTP, el cual a través de su respectiva librería de Python, nos permite realizar tareas de forma automatizada. Para revisar el correo recibido se usa el protocolo IMAP, pero el aprendizaje del módulo imaplib excede los objetivos de este curso.

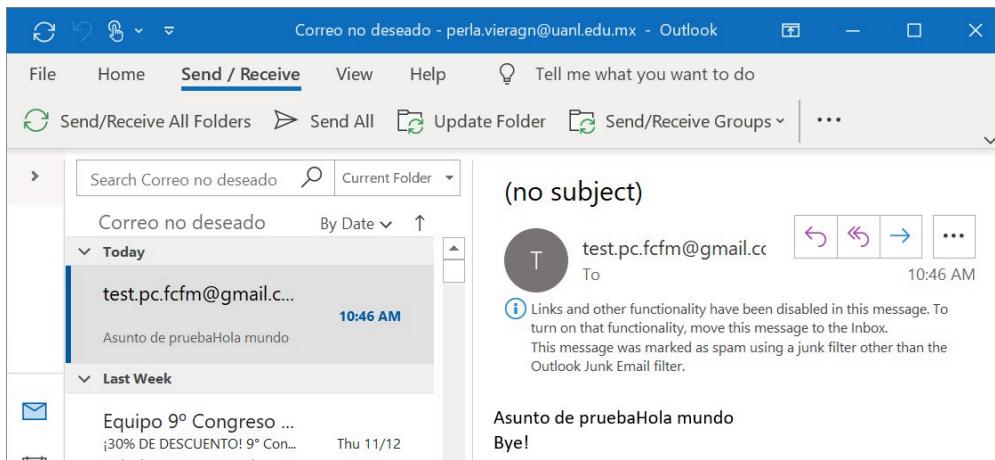
IMPORTANTE: Además de los protocolos SMTP e IMAP, la mayoría de los proveedores de servicio de correo electrónico utilizan otras medidas para protegerse contra el spam, phishing y otros usos maliciosos del correo electrónico. Dichas medidas de seguridad previenen que los scripts inicien sesión en cuentas de correo con los módulos que analizaremos.

SMTP - smtplib

El protocolo SMTP, *Simple Mail Transfer Protocol*, es usado para el envío de correo electrónico ya que indica el formato que deben tener los correos electrónicos, como deben encriptarse, y como debe ser retransmitido entre los servidores de correo y todos los demás detalles que manejan el correo después de hacer clic en Enviar.

Envío de correo electrónico

```
C:\Users\marle\Downloads\Mails>python
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct  5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import smtplib
>>> import getpass
>>> smtpObject = smtplib.SMTP('smtp.gmail.com', 587)
>>> smtpObject.ehlo()
(250, b'smtp.gmail.com at your service, [2806:2f0:4060:f9cf:41b:dcaa:1d04:12c0]\nSIZE 35882577\n8BITMIME\nSTARTTLS\nENHANCEDSTATUSCODES\nPIPELINING\nCHUNKING\nSMTPUTF8')
>>> smtpObject.starttls()
(220, b'2.0.0 Ready to start TLS')
>>> password = getpass.getpass()
Password:
>>> account = 'test.pc.fcfm@gmail.com'
>>> smtpObject.login(account,password)
(235, b'2.7.0 Accepted')
>>> to = 'perla.vieragn@uanl.edu.mx'
>>> subject = 'Asunto de prueba'
>>> cuerpo = subject+"Hola mundo\nBye!"
>>> smtpObject.sendmail(account,to,cuerpo)
{}
>>> smtpObject.quit()
(221, b'2.0.0 closing connection t126sm3789334oih.51 - gsmtp')
>>>
```



Conexión con un servidor SMTP

El servicio de envío de e-mail se da a través del protocolo SMTP, para lo cual es necesario conocer el nombre del servidor SMTP (generalmente el nombre de tu proveedor de servicio con `smtp.` al inicio del nombre) y el puerto a través del cual trabaja, que en la mayoría de los casos es el 25. También se usa el 465 y el 587.

Para la realización de la conexión, se requieren los siguientes pasos:

1. Crear un objeto para conectarnos con el servidor

```
>>> smtpObject = smtplib.SMTP('smtp.gmail.com', 587)
>>> type(smtpObject)
<class 'smtplib.SMTP'>
```

2. Si la creación del objeto no fue exitosa, es probable que el servidor no soporte conexiones TLS (puerto 587), por lo que debemos probar SSL (puerto 465), para esto se cambia la función SMTP por SMTP_SSL.
3. Enviar un mensaje de saludo (Hello) a través de la función ehlo(). Si el primer elemento de la tupla es 250, significa que la comunicación con éxito.
4. Si lograste la comunicación a través del protocolo 587, ahora se debe llamar al método starttls(), si te conectaste a través de 465, no necesitas ningún otro método, ya se estableció la encriptación. Para el método starttls(), el código 220 significa que la conexión fue un éxito.
5. Después, con el método login(), enviando tu usuario y contraseña, se inicia la sesión en el servidor. El código 235 significa que se logró el inicio de sesión.
6. Para el envío de correo se usa el método sendmail(), el cual requiere como argumentos el correo del remitente (previamente loggeado) destinatario y el cuerpo del correo.
7. Finalmente, el método quit() desconectará el programa del servidor SMTP.

Conexión SSL con SMTP

```
import smtplib
import ssl
import getpass

port = 465 # For SSL
password = getpass.getpass()
message = "Hello world!"

# Create a secure SSL context
context = ssl.create_default_context()

with smtplib.SMTP_SSL("smtp.gmail.com", port, context=context) as server:
    server.login("test.pc.fcfm@gmail.com", password)
    server.sendmail(sender_email, receiver_email, message)
```

Conexión TLS con SMTP

```
import smtplib
import ssl
import getpass

smtp_server = "smtp.gmail.com"
port = 587 # For starttls
sender_email = "test.pc.fcfm@gmail.com"
password = getpass.getpass()
message = "Hello world!"

# Create a secure SSL context
context = ssl.create_default_context()

# Try to log in to server and send email
try:
    server = smtplib.SMTP(smtp_server, port)
    server.ehlo() # Can be omitted
    server.starttls(context=context) # Secure the connection
    server.ehlo() # Can be omitted
    server.login(sender_email, password)
    server.sendmail(sender_email, receiver_email, message)
except Exception as e:
```

```
# Print any error messages to stdout
print(e)
finally:
    server.quit()
```

Otra versión

```
import smtplib
import ssl
import getpass

port = 587 # For starttls
smtp_server = "smtp.gmail.com"
sender_email = "test.pc.fcfm@gmail.com"
receiver_email = "perla.vieragn@uanl.edu.mx"
password = getpass.getpass()
message = """\
Subject: Hi there

This message is sent from Python."""

context = ssl.create_default_context()
with smtplib.SMTP(smtp_server, port) as server:
    server.ehlo() # Can be omitted
    server.starttls(context=context)
    server.ehlo() # Can be omitted
    server.login(sender_email, password)
    server.sendmail(sender_email, receiver_email, message)
```

Envío de correo con formato

```
import smtplib
from email.mime.text import MIMEText
import getpass

def sendMail(user, pwd, to, subject, text):
    msg = MIMEText(text)
    msg['From'] = user
    msg['To'] = to
    msg['Subject'] = subject
    try:
        smtpServer = smtplib.SMTP('smtp.gmail.com', 587)
        print("[+] Connecting To Mail Server.")
        smtpServer.ehlo()
        print("[+] Starting Encrypted Session.")
        smtpServer.starttls()
        smtpServer.ehlo()
        print("[+] Logging Into Mail Server.")
        smtpServer.login(user, pwd)
        print("[+] Sending Mail.")
        smtpServer.sendmail(user, to, msg.as_string())
    except Exception as e:
        print("[-] Error sending mail: ", e)
```

```

        smtpServer.quit()
    except:
        print("[+] Mail Sent Successfully.")
    print("[-] Sending Mail Failed.")

user = 'test.pc.fcfm@gmail.com'
pwd = getpass.getpass()
sendMail(user, pwd, 'perla.vieragn@uanl.edu.mx',
          'Re: Important', 'Test Message')

```

Si quieras agregar formato al texto de tus correos electrónicos, así como imágenes, hipervínculos o contenido responsive, es recomendable usar tu correo. Actualmente, el tipo más común de correos electrónicos es MIME, *Multipurpose Internet Mail Extensions*, lo que permite combinar texto plano con HTML.

En el ejemplo previo vemos como se estructura en general un correo, aunque solo maneja texto plano, que utiliza MIME.

Ahora vamos a ver un correo electrónico que tenga opción HTML y texto plano.

```

import smtplib
import ssl
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
import getpass

sender_email = "test.pc.fcfm@gmail.com"
receiver_email = "perla.vieragn@uanl.edu.mx"
password = getpass.getpass()

message = MIMEMultipart("alternative")
message["Subject"] = "multipart test"
message["From"] = sender_email
message["To"] = receiver_email

# Create the plain-text and HTML version of your message
text = """\
Hi,
How are you?
Real Python has many great tutorials:
www.realpython.com"""

html = """\
<html>
  <body>
    <p>Hi,<br>
      How are you?<br>
      <a href="http://www.realpython.com">Real Python</a>
      has many great tutorials.
    </p>
  </body>
</html>
"""

# Turn these into plain/html MIMEText objects
part1 = MIMEText(text, "plain")
part2 = MIMEText(html, "html")

```

```
# Add HTML/plain-text parts to MIME-Multipart message
# The email client will try to render the last part first
message.attach(part1)
message.attach(part2)

# Create secure connection with server and send email
context = ssl.create_default_context()
with smtplib.SMTP("smtp.gmail.com", 587) as server:
    server.ehlo()
    server.starttls(context=context)
    server.login(sender_email, password)
    server.sendmail(
        sender_email, receiver_email, message.as_string()
    )
```

Scripts



01_TestM...



02_TestM...



03_TestM...



04_TestM...



05_TestM...

