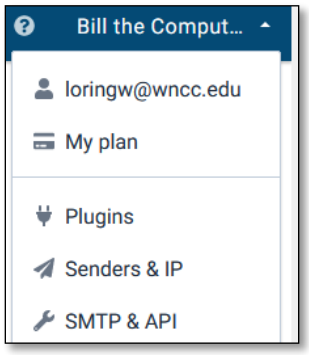
Email Spoofing

Contents

Time required: 60 minutes

Spoofing an email address happens all the time. Just because a message says it is from someone, doesn't mean is.

NOTE: Don't use your **wncc.edu** email account for this lab.



Tutorial 1: Setup Brevo Account

https://www.brevo.com/ is an email marketing platform. They offer a Free plan with a daily sending limit of 300 emails. That is plenty for our email spoofing campaign.

1. Go to https://www.brevo.com/ and create a free account.
2. Go to your Account Logon → Click your **Account name** → **SMTP and API**
3. Record your **SMTP Server**, **Port**, **Login** and **SMTP KEY VALUE**.

Tutorial 2: Send Spoofed Email in Python

Sending an email with the Brevo SMTP approach adds some code to the message to show if someone has opened the message or not.

Create a Python program named: **brevo_credentials.py** We will import this information into our email program.

```python
#!/usr/bin/env python3
"""
    Name: brevo_credentials.py
    Author:
    Created:
    Purpose: brevo email credentials
"""
SMTP_SERVER = "smtp-relay.brevo.com"
PORT = 587

# ---------- REPLACE WITH YOUR INFORMATION ------------#
LOGIN = ""
PASSWORD = ""
# -----------------------------------------------------#
```

Create a Python program named: **brevo.py**

```python
#!/usr/bin/env python3
"""
    Name: python.py
    Author:
    Created:
    Purpose: Send email through Python with Brevo
    300 emails per day
"""
from email.message import EmailMessage  # Create email message
import smtplib  # Send email through an smtp server
import ssl  # Create secure context to send mail
import brevo_credentials  # Import credentials
```

This code imports the libraries needed to send a secure email. Everything is encrypted in transit.

```python
def main():
    """Main function to execute sending an email."""
    msg = create_message()
    send_message(msg)

```

This is the main function definition which calls the create_message function to create a msg object. The msg object is passed to the send_message function.

```python
# --------------- CREATE MESSAGE ----------------------------------------#
CodiumAI: Options | Test this function
def create_message():
    """Create an email message with specific content and recipients."""

    # ------------- REPLACE BELOW WITH YOUR INFORMATION -------------------#
    # A list containing one or more email addresses
    email_from = "William Loring <williamloring@hotmail.com>"
    email_dst = [
        "williamaloring@gmail.com",
        "williamloring@hotmail.com"
    ]
    content = """\
    This test message is sent from Python Bill."""
    subject = "Email from Ethical Hacking Class from Brevo"
    # ------------- REPLACE ABOVE WITH YOUR INFORMATION -------------------#

    msg = EmailMessage()
    msg["From"] = email_from
    msg.set_content(content)
    msg["Subject"] = subject
    msg["To"] = email_dst

    return msg
```

The **create_message** function creates the email message object we are going to send. creates an email message with specific content and recipients. We return that object to the main function.

- Replace the **email_from**, **email_dst**, and **content** variables with your information.
- Edit the subject as needed.

```python
# --------------- SEND MESSAGE ------------------------------------------#
CodiumAI: Options | Test this function
def send_message(msg):
    """Send the prepared email message."""
    # Set email server object
    server = smtplib.SMTP(
        brevo_credentials.SMTP_SERVER,
        brevo_credentials.PORT
    )

    # Show all communication with the server
    server.set_debuglevel(True)

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

    # ---------------------- SEND EMAIL ---------------------------------#
    try:
        # Start an encrypted TLS session
        # This will encrypt our password and message
        server.starttls(context=context)

        # Login to the mail server
        server.login(brevo_credentials.LOGIN, brevo_credentials.PASSWORD)

        # Send the email message
        server.send_message(msg)

        print("*****************************************")
        print("*    Email message successfully sent.   *")
        print("*****************************************")
    except Exception as e:
        print(e)
    finally:
        # Quit from server
        print(server.quit())


# Call the main function
if __name__ == "__main__":
```

The **send_message(msg)**: function sends a prepared email message.

- It establishes a connection with the Brevo SMTP server using smtplib.

- It creates a secure TLS connection using ssl for secure communication.
- It logs in to the server with your Brevo credentials.
- It sends the email message and prints a confirmation message.

Send an email to one of your email accounts.

Read the message.

Go back to www.Brevo.com Go to **Transactional** tab → **Statistics**.

Scroll down to recent activity. You can see the messages that are sent and opened. An html tag is inserted in the message to track whether the end user received the email message or not.

Take a screenshot of your successfully sent email message to be submitted with this assignment.

Tutorial 3: Send Spoofed Email with sendemail with Kali

sendemail with Kali allows us to better spoof a target.

1. Set the Kali Linux virtual machine to a Bridged Adapter.
2. At a terminal prompt type: **sendemail** Look through the help information.
3. At a terminal prompt → **geany send_email.sh**

NOTE: Put everything on one line as this is a shell script. The example below is separated onto separate lines for easier reading.

4. Replace the generic information with your Brevo information.
5. **CTRL-S** will save the file. **CTRL-X** will exit nano.

```
sendemail -o tls=no
-xu brevo_login
-xp brevo_password
-s smtp-relay.brevo.com:587
-f "from_email_address"
-t "to_email address"
-u "A test of this stuff"
-m "This is an email test of send in blue"
-o message-header="From: Spoofed Name <Spoofed Email address>"
-v
```

6. We need to change the permissions of this file to be an executagle file.
7. At the terminal → **chmod 700 send_email.sh**
8. At the terminal → **./send_email.sh**
9. The email should be sent.

Explanation of program and switches.

sendemail	**Send email program in Kali**
-xp	Username for SMTP authentication
-s	SMTP mail server (relay)
-f	From (sender) email address
-t	To (recipient) email address
-u	Message subject
-m	Message body
-o	Advanced option, create a spoofed header
-v	Verbose mode for debugging

Example run:

```
Feb 17 14:57:08 kalibill sendemail[11126]: SUCCESS ⇒ Received:         334 UGFzc3dvcmQ6
Feb 17 14:57:08 kalibill sendemail[11126]: INFO ⇒ Sending:     UzN3Vk1rRzVUcGZuUkVkTg==
Feb 17 14:57:08 kalibill sendemail[11126]: SUCCESS ⇒ Received:         235 2.0.0 Authentica
tion succeeded
Feb 17 14:57:08 kalibill sendemail[11126]: DEBUG ⇒ User authentication was successful (Meth
od: LOGIN)
Feb 17 14:57:08 kalibill sendemail[11126]: INFO ⇒ Sending:     MAIL FROM:<iltfm42@hotmail.c
om>
Feb 17 14:57:08 kalibill sendemail[11126]: SUCCESS ⇒ Received:         250 2.0.0 Roger, acc
epting mail from <iltfm42@hotmail.com>
Feb 17 14:57:08 kalibill sendemail[11126]: INFO ⇒ Sending:     RCPT TO:<williamloring@hotma
il.com>
Feb 17 14:57:08 kalibill sendemail[11126]: SUCCESS ⇒ Received:         250 2.0.0 I'll make
sure <williamloring@hotmail.com> gets this
Feb 17 14:57:08 kalibill sendemail[11126]: INFO ⇒ Sending:     DATA
Feb 17 14:57:08 kalibill sendemail[11126]: SUCCESS ⇒ Received:         354 Go ahead. End yo
ur data with <CR><LF>.<CR><LF>
Feb 17 14:57:08 kalibill sendemail[11126]: INFO ⇒ Sending message body
Feb 17 14:57:08 kalibill sendemail[11126]: Setting content-type: text/plain
Feb 17 14:57:08 kalibill sendemail[11126]: SUCCESS ⇒ Received:         250 2.0.0 OK: queued
as <332975.487780885-sendEmail@kalibill>
Feb 17 14:57:08 kalibill sendemail[11126]: Email was sent successfully!  From: <iltfm42@hotm
ail.com> To: <williamloring@hotmail.com> Subject: [A test of this stuff] Server: [smtp-relay
.brevo.com:587]
```

Assignment

1. Send yourself and your instructor a spoofed email using each method.
2. Take a screenshot of the received email.
3. Compare the two email messages.

Assignment Submission

Attach the complete program, script, and screenshot of each spoofed email to the assignment in Blackboard.