Sometimes I need to test if a particular machine is able to send e-mail via an SMTP server. Using telnet or openssl is a great way to test and debug connection issues.

We use Sendgrid for sending mails in most of our web applications, so I'll use their SMTP server as an example.

Requirements

Before we can get started, make sure you have telnet or openssl installed. Both come installed by default on most Linux distro's. If it isn't installed you can grab it using your default package manager:

```
sudo apt-get install telnet openssl  # or  sudo yum install telnet openssl
```

On Mac, you can install it with Homebrew :

```
brew install telnet openssl
```

Connecting to the SMTP server

To connect to smtp.sendgrid.com, we issue the following command:

```
$ telnet smtp.domain.com 25
```

If the connection is initiated, you'll get output similar to the following back:

```
Trying 159.122.219.43…
Connected to smtp.sendgrid.net.
Escape character is '^]'.
220 SG ESMTP service ready at ismtpd0001p1lon1.sendgrid.net
```

It's important to note that we have initiated an unencrypted connection. This means that all information is being sent to the server in plaintext. Because we need to authenticate and send passwords, I recommend to connect using SSL or TLS if your server supports it.

To connect using the TLS protocol on port 587, use:

```
$ openssl s_client -starttls smtp -connect smtp.sendgrid.com:587
```

To use SSL on port 465:

```
$ openssl s_client -connect smtp.sendgrid.com:465
```

You'll get a lot of output concerning the SSL session and certificates used, but afterwards you'll see a similar confirmation as with the telnet command (a 220 or 250 status code with a message). For example:

```
220 SG ESMTP service ready at ismtpd0002p1lon1.sendgrid.net
```

Initiate the conversation

Now we need to identify ourself and initiate the SMTP conversation with the EHLO command. This command takes an IP address or domain name as argument:

```
EHLO stevenrombauts.be
```

On most SMTP servers you will get a list of commands back when using the *Extended Hello* (EHLO) command:

```
250-smtp.sendgrid.net
 250-8BITMIME
 250-PIPELINING
 250-SIZE 31457280
 250-AUTH PLAIN LOGIN
 250 AUTH=PLAIN LOGIN
```

Authenticate yourself

Time to log in! The previous output listed the AUTH command and the available mechanisms. In this case PLAIN and LOGIN.

Using PLAIN

The PLAIN mechanism expects a base64 encoded string containing both username and password, each prefixed with a NULL byte. To generate this string using the base64 binary, run this command:

$ echo -ne "□username□password" | base64 AHVzZXJuYW1lAHBhc3N3b3Jk

Note in Sendgrid, the username should be apikey and the password is the API key you generated (as described in their documentation).

Now you can pass this base64 encoded string to the AUTH command in your openssl or telnet session:

AUTH PLAIN AHVzZXJuYW1lAHBhc3N3b3Jk

The result should be:

235 Authentication successful

Using LOGIN

The LOGIN mechanism also expects base64 encoded username and password, but separately. First, generate the base64 encoded strings:

$ echo -ne "username" | base64 dXNlcm5hbWU= $ echo -ne "password" | base64 cGFzc3dvcmQ =

and authenticate with the SMTP server:

AUTH LOGIN

You will be prompted for the username first, then the password. The entire conversation will look like this:

334 VXNlcm5hbWU6
dXNlcm5hbWU=
334 UGFzc3dvcmQ6
cGFzc3dvcmQ=
235 Authentication successful

Send an e-mail

Now we get the good stuff! We need at least these details to be able to send an e-mail:

- The sender (MAIL FROM)
- The recipient (RCPT TO)
- and the message body (DATA)

You must always start with the MAIL FROM command, as this tells the SMTP server that a new mail transaction is started.

We follow that up by the recipient's address and finally the message subject and body. Both the subject header and body are passed via the DATA command. I also recommend to always include the From: header again in the DATA command.

Once we are ready to send our message, we end with a single dot (.) character. Here's how that looks if you put it all together:

MAIL FROM: from@domain.com 250 Sender address accepted rcpt to: john@doe.com 250 Recipient address accepted DATA 354 Continue From: from@domain.com Subject: Test message! Hi, This is a test message! Best, Steven . 250 Ok: queued as bazLUK4DEBqH25dH6iZuNg

You should receive a confirmation (250 Ok) at the end that the message was accepted.

Note: if you connected with openssl instead of telnet, you have to make sure to type the rcpt to command in **lowercase**. Pressing R in the client session instructs openssl to renegotiate the TLS connection.

Type QUIT to close the session.