

SEND AUTHENTICATED EMAILS DIRECTLY FROM THE PLC USING TCPCONNECT

INTRODUCTION

Previously there were two ways to send emails from TRI PLCs, which were to use the built in <EMAIL> tag to send non-authenticated emails directly from the PLC, or to use the remote file services (REMOTEFSS) method to login to TLServer running on a PC and indirectly send emails with or without authentication from TLServer (requires configuration in TLServer).

It is also possible to send authenticated emails directly from the PLC by logging in to your SMTP mail server and sending standard commands over the Internet to configure and send an email. This is a good alternative for users that need to send authenticated emails (no encryption) and are unable to or don't want to have TLServer running remotely in order to do so. This document will explain how this works and how it is done with the PLC. Two sample programs are included with this document that can be integrated into your current PLC program to perform an email function.

Note that the method described and the sample programs that implement this method are not guaranteed to work for every user because some SMTP servers require encryption, which is not possible to achieve from the PLC, and a different method of authentication may be used.

HOW EMAILS CAN BE SENT FROM THE PLC

The built-in TCPCONNECT command that is available in the TRiLOGI programming software can be used to connect to your SMTP mail servers IP address and Port number. Once connected the PLC program can send a series of standard commands recognized by SMTP servers using the built-in PRINT#4 command to send data out of the PLCs Ethernet port. This is actually what the <EMAIL> tag does transparently except some commands are slightly varied and authentication is not implemented.

Not all SMTP mail servers will support all the potential commands for configuring email settings and data, but those that require authentication should at least have the following commands available :

EHLO	Tell the SMTP server an email is being sent.
AUTH LOGIN	Tell server you are logging in with authentication.
MAIL FROM:	Email address of sender (typically can be anything).
RCPT TO:	Email address of recipient (must be correct).

DATA	Indicate beginning of email body
To:	Formatting tag for the recipient email that will be displayed (can be different from the actual recipient address). <i>Optional</i>
From:	Formatting tag for the sender email that will be displayed (can be different from the actual sender address). <i>Optional</i>
Subject:	Formatting tag for the email subject that will be displayed. <i>Optional</i>
•	A period indicates the email is finished and the server will attempt to send it.

The EHLO command is typically sent with the domain name of the ISP that the SMTP mail server is registered to as a parameter, but it is not usually required and can be sent by itself.

The AUTH LOGIN command doesn't require any parameters and should be followed by the username and password that is configured for your SMTP authenticated login. The username and password are typically encoded in a BASE64 format, which is explained in the next section.

The MAIL FROM: command is sent with the senders email address as the parameter.

The RCPT TO: command is sent with the recipients email address as the parameter.

The DATA command has no parameter and is either followed directly by the body content of the email or by the TO: / FROM: / SUBJECT: tags, which each have a single parameter for what is to be displayed in the To: / From: / Subject: headings of the email.

EXAMPLE

Here is the minimum PLC code required to connect to an SMTP mail server and send an email.

```
PRINT#4 "<TCPCONNECT 11.22.33.44:25>"
PRINT#4 "EHLO www.domainname.com"
PRINT#4 "AUTH LOGIN"
PRINT#4 "username"
PRINT#4 "password"
PRINT#4 "MAIL FROM: sender@emailaddress.com"
PRINT#4 "RCPT TO: recipient@emailaddress.com"
PRINT#4 "DATA"
PRINT#4 "To: Jane Doe"
PRINT#4 "From: John Doe"
PRINT#4 "Subject: Email Subject"
PRINT#4 "Email Body Line 1 of x total lines"
PRINT#4 "."
PRINT#4 "</>"
```

AUTHENTICATION

Authentication is basically a way for the SMTP mail server to verify that the user sending an email through the server is registered in order to prevent malicious activity from random people or spam bots. It is usually not required when you are sending emails from an account provided by the ISP you are registered with if you are sending the email from the location that you are registered to (ie. Not a public location). However, that is not always the case and it is not always possible to send emails from the location registered with the ISP, so many users now have the need to provide authentication.

If your SMTP server that your email account uses requires authentication, you should already have a username and password registered with your account. However, the username and password typically must be encoded in BASE64 format before they are sent to the SMTP server for verification. Therefore, if your username is 'myusername', it must be converted to 'bXl1c2VybmFtZQ==' in BASE64. In the above example code the username and password should both be encoded in BASE64 when sent by the PLC using PRINT#4.

The purpose of BASE64 encoding is to standardize username/password strings so that they are made up of the universal set of ASCII characters to make it easier for SMTP mail servers when dealing with uncommon characters and different languages.

BASE64 encoding is essentially a 64-bit representation of the characters in your username/password when they are strung together sequentially in a binary format. There is a consistent method for converting the username/password into BASE64 format, which can be accomplished by the PLC program.

One of the sample programs provided will take the original version of your username/password and automatically encode them in BASE64 format and the other program requires you to provide the username/password already in BASE64 format.

Here is a link to an online tool that allows you to convert a standard string into BASE64 format, which is very useful if you want to use the basic version of the sample program.

<http://www.hcidata.info/base64.htm>

ENCRYPTION

Some SMTP mail servers require the authentication process and email data transfer to be encrypted. The two main types of encryption are TLS (Transport Layer Security) and SSL (Secure Sockets Layer). Unfortunately TRI PLCs are currently unable to perform any encryption and therefore won't be able to interact with those SMTP mail servers that require it.

SAMPLE PROGRAM

There are two sample programs packaged with this application that can perform an authenticated login to an SMTP server and allow emails to be sent directly from the PLC. These are the included programs :

1. EMAIL BASIC AUTH-REV1.PC6
2. EMAIL FULL AUTH-REV1.PC6

These programs can be used on their own or embedded in your own program as an email function.

Both of these programs will make a connection to an SMTP server, perform an authenticated login, and send an email based on parameters that must be configured prior to PLC program download.

The first program is a basic version that does not perform any BASE64 conversion on the username/password for authentication, and requires that you provide the username and password in BASE64 format. The username/password can be obtained in BASE64 format using the online tool mentioned above in the Authentication section. The converted username/password can be copied and pasted into the appropriate location in the sample program.

The second program allows you to provide the username and password in standard form since it will be converted to BASE64 format automatically within the program.

In both programs, all of the configuration is done in the “Send_Email” function, and the SMTP server communication and actual email sending is managed in the “TCP_Email” function.

Here is a screenshot from the “Send_Email” function of the full version with BASE64 conversion :

```

Custom Function #1 - Send_Email

'This function sets the Email parameters such as the SMTP server, send/receive Email addresses
'and Email body. You also need to define the number of body lines.
'The function also calls the TCP_Email function, which formats and sends the Email.

A$ SMTP_Server = "xxx.xxx.xxx.xxx:25" ' Your SMTP server and port (default is 25)
B$ Send_Email = "FName LName <email_id@email_address.com>" ' Email address of the sender
C$ Receive_Email = "FName LName <email_id@email_address.com>" ' Email address of the Recipient
D$ Subject = "Test Email Subject" ' Email subject
E$ Domain = "www.your_domain.COM" ' local domain name - where the Email is sent from
' May not be required

V$ = "" ' Temp variable for username and password when converted to BASE64
' *** V$ should not be edited ***
W$ = "Username" ' Username, which will be converted using "Base 64 Encoding"
X$ = "Password" ' Password, which will be converted using "Base 64 Encoding"

F$ Body_Line1 = "Test Email line1"
G$ Body_Line2 = "Test Email line2"
H$ Body_Line3 = "Test Email line3"
I$ Body_Line4 = "Test Email line4"

Z_Number_Of_Lines = 4 ' The number of lines in your Email body
IF Z > 16 ' Max number of lines before authentication strings are overwritten.
Z = 16
ENDIF

CLRIO EMAIL_SENT
CLRIO EMAIL_FAILED

CALL TCP_Email

```

The only settings that need to be configured in order for the email to be sent are :

1. The SMTP server address and port number, which is stored in the A\$ variable.
2. The Recipient email address, which is stored in the C\$ variable.
3. The Username and Password, which are stored in the W\$ and X\$ variables (If authentication is required).

The other settings can be left in the default configuration shown above or changed to something different (whether it is real or not). Although the email will be more standard and complete looking if all the settings are configured properly.

Once the settings have been configured and the program has been transferred to the PLC, an email will be sent each time input #1 is activated, which can be done through the physical input directly on the PLC or through online monitoring from the TRiLOGI programming software.

Either of the sample programs can be freely used and modified as needed for your own application.