SMTP-Send™ for z/OS

Release 6.0

Installation and Reference Guide

SMTP-Send and OfficePath/SMTP-Send for z/OS

Trademarks
Product names may be trademarks or registered trademarks of their respective owners.
OS/390® and z/OS® are registered trademarks of IBM Corporation. Other company, product, and service names may be trademarks or service marks of others.

Edition Information

January 2016

This edition applies to Release 6 of SMTP-Send and OfficePath/SMTP-Send. It also applies to subsequent releases until otherwise indicated in new editions.
Inquiries may be directed to:

TBS Software Inc.
2201-278 Bloor St E.
Toronto, ON
Canada
M4W 3M4

Phone:	905 940-9373
Fax:	905 940-9376
Web:	www.tbssoft.com
eMail:	support@tbssoft.com

Table of Contents

Introduction

Have you ever wanted to distribute information such as reports and status notifications to your end users directly from z/OS, OS/390 (MVS) batch, TSO and/or CICS functions?

While simple solutions using file-transfer technologies may at first seem attractive, they create new challenges of their own. For example, you could use FTP to transfer data sets to a LAN. The transferred files then need to be addressed and distributed via traditional e-mail. It is possible to create distribution requests to be processed by IBM's SMTP server task, but these require detailed knowledge of a number of Internet protocols.

SMTP-Send for OS/390 and z/OS solves these problems by allowing your MVS batch, CICS and TSO processes to communicate using industry-standard SMTP. Your applications and operations staff can quickly begin mail-enabling your line of business and operations processes from within the 'glass house' without getting lost in arcane protocols such as TCP/IP, SMTP, POP3, IMAP, etc.

Since all of the processing takes place on the MVS platform you don't need to learn about PC or UNIX-based mail systems. Your operations staff can control the processing using tools they're already familiar with.

SMTP-Send is a one-way communications process from MVS to the outside world. This allows you to easily position the MVS SMTP server behind your firewall to preserve system integrity and prevent viruses entering your secure MVS environment.

Highlights

- sends e-mail from z/OS and OS/390 batch, TSO and CICS applications
- uses text control statements to provide simple access to Internet protocols
- connects to all SMTP-compliant e-mail systems including Lotus Notes/Domino and Microsoft Exchange
- easy to install, configure and administer
- central source for distribution lists
- central log of all distributions

Benefits

- easily send e-mail from z/OS and OS/390 with no programming
- uses industry-standard SMTP

Other books you may need

Title
SMTP-Send for z/OS User Guide
IBM TCP/IP for MVS Customization and Administration Guide
z/OS or OS/390 Communications Server IP Configuration Guide

Note: Throughout this book you will see sections prefixed with **OP/SS**. These sections apply only to OfficePath/SMTP-Send. If you are using SMTP-Send, then you may ignore them. Where the term SMTP-Send is used in this book, it also applies to OfficePath/ SMTP.

OP/SS Overview of OfficePath/SMTP

Many IBM OfficeVision/MVS (OV/MVS) customers have mail-enabled line-of-business (LOB) applications that use the underlying e-mail network to distribute information such as reports and status notifications from MVS batch, TSO and/or CICS functions to end-users. OV/MVS traditionally uses a SNADS-based e-mail network, either IBM's DISOSS or TBS Software's OfficePath/SNADS.

Today even as OV/MVS customers migrate their e-mail users to LAN-based systems such as Lotus Notes/Domino and Microsoft Exchange their LOB applications continue to run on MVS. However, once all of their OV/MVS users have been migrated, the cost of maintaining a SNADS network just for this purpose becomes a significant cost issue.

While alternate solutions like file-transfer may at first seem attractive, they create new challenges of their own. For example, customers who use OfficePath/Batch-TSO and/or OfficePath/CICS (or their RAPID equivalents) will need to identify all e-mail enabled LOB applications, then change them to use FTP instead of SNADS. Further, FTP only transfers the files to a LAN. The files still need to be addressed and distributed via traditional e-mail.

OfficePath/SMTP-Send (OP/SS) solves these problems by replacing your SNADS e-mail network with industry-standard SMTP. Because it is designed to work with OfficePath or RAPID, no changes are required to your existing mail-enabled LOB applications. Simply replace OfficePath/SNADS or RAPID/Base with OP/SS and map SNADS addresses to SMTP mail addresses.

Additional Highlights

- replaces the SNADS Send services of DISOSS and OfficePath/SNADS
- no changes required to existing e-mail enabled LOB applications
- SNADS to SMTP address mapping using tables or external directory (LDAP)

Additional Benefits

- preserves your investment in OfficePath/Batch, TSO, CICS and RAPID/Batch, TSO, CICS
- replaces SNADS mail infrastructure with industry-standard SMTP
- reduces overall cost of MVS-to-LAN e-mail connectivity through lower software licensing fees and administrative costs

Other books you may need

OfficePath	RAPID
Installation and Administration Guide	Installation and Administration Guide
Reference and Diagnosis	User Guide
Programming Interfaces	Problem Determination Guide

Soft*Switch Application Toolkit compatibility

You can replace the Soft*Switch Application Toolkit using OfficePath/SMTP-Send and the SMTP-Send SSWDA feature. This feature provides a replacement SSWDA program that accepts Application Toolkit input and converts this input into OfficePath/ SMTP-Send statements and automatically calls OfficePath/SMTP-Send.

Installing SMTP-Send

The install process involves downloading the appropriate datasets from the TBS Software FTP site. Download the README.TXT and PDF documentation files to a PC for review. Then download the file, DOWNLOAD.JCL to your z/OS system. This dataset is a JCL job which will download all other datasets necessary for the installation. Among these datasets will be RECEIVE.JCL. This is a JCL job that performs a RECEIVE on each of the TRANSMIT datasets downloaded. You will now have all of the required datasets on your system.

Requirements

- any z/OS release
- IBM TCP/IP
- under CICS, the TCP/IP socket interface must be active
- **OP/SS** requires at least one of:
  - OfficePath/Batch-TSO
  - OfficePath/CICS
  - RAPID/Batch-TSO
  - RAPID/CICS
- the userid whose credentials are used to run SMTP-Send must be authorized to use z/OS UNIX System Services (the userid must have an OMVS segment). This is required to allow use of the TCP/IP socket interface utilized by SMTP-Send. Refer to the UNIX System Services Planning documentation for further information.

Product authorization codes

SMTP-Send requires a product authorization code that reflects the type of license you have. This code lets SMTP-Send operate in one of the modes described below. Without an authorization code, the product will operate in Demonstration mode.

Demonstration mode

Demonstration mode does not require a product authorization code. In this mode, SMTP-Send will send the entire message, but the program will insert data into the subject of the message indicating that demonstration mode is in effect. The program will end with a return code of 16 in Demonstration mode.

Demonstration mode allows you to install, test and verify SMTP-Send without having to begin a trial.

Trial mode

Trial mode allows you to evaluate SMTP-Send for a reasonable period of time. When you decide that SMTP-Send is of value to you, you can contact TBS Software to obtain a license and a product authorization code.

Throughout the trial period, SMTP-Send outputs a message at startup that tells you when the trial period will end. In trial mode, SMTP-Send will distribute any text, but as the end of the trial appears it will add to the subject an indication that the product is in trial mode and the number of days until the trial expires. Immediately prior to expiration of the trial, SMTP-Send will return code 4 upon completion to indicate the trial is about to end.

After the trial period ends, SMTP-Send issues a warning message, and within 7 days of expiry, the program ends with return code 6. Following 7 days after expiry, the program reverts to demonstration mode and ends with return code 12.

Rental mode

Rental mode allows you to use SMTP-Send for a specific period, usually one year. Throughout the rental period, SMTP-Send outputs a message at startup that tells you when the rental period will expire.

Beginning one month before the rental period expires, SMTP-Send issues a warning message in its SYSPRINT or SYSOUT output data set and will return code 4 upon completion.

After the rental period expires, SMTP-Send continues to operate for a limited grace period. During this grace period, SMTP-Send issues a warning message and ends with return code 6.

After the grace period expires, SMTP-Send issues a warning message, reverts to demonstration mode and ends with return code 12.

Permanent mode

Permanent mode allows you unrestricted use of SMTP-Send. Once SMTP-Send verifies your permanent mode authorization code, no further expiration checking is done. This gives you full use of SMTP-Send without a time limitation.

HTML Mail Merge feature

HTML Mail Merge allows the creation of custom emails from generic templates. This feature requires a separate authorization code. This feature operates in the same modes as the base product, Demonstration, Trial, Rental and Permanent.

Distribution package contents

- Installation datasets via FTP
- Documentation via FTP
- Product authorization code (not required for demonstration mode)

Distribution data sets

Data Set Name	Description
Documentation…	
Readme.txt	
SMTPSEND.R60.Release Notes.pdf	
SMTPSEND.R60.User Guide.pdf	
SMTPSEND.R60.Installation & Reference Guide.pdf	
General Datasets…(required)	
SMTPSEND.R60.INSTLIB	installation & customization jobs; CICS resources; samples
SMTPSEND.R60.DOWNLOAD.JCL	customizable installation JCL
SMTPSEND.R60.RECEIVE.JCL	customizable installation JCL
ISPF Datasets…(required only to use the ISPF dialogs)	
SMTPSEND.R60.ISPPLIB	ISPF dialog panels
SMTPSEND.R60.ISPCLIB	ISPF dialog CLISTs
SMTPSEND.R60.ISPMLIB	ISPF dialog messages
Batch LOADLIBs…(select the appropriate one)	
SMTPSEND.R60.LOADLIB	load modules for the Generic Edition
SMTPSEND.R60.CLASY.LOADLIB	load modules for the CLASY Edition
SMTPSEND.R60.OFP. LOADLIB	load modules for the OfficePath Edition and the RAPID Editions
SMTPSEND.R60.OVFAX. LOADLIB	load modules for the OV/Fax Edition
SMTPSEND.R60.SSWDA. LOADLIB	load modules for the SSWDA Edition
CICS LOADLIBs…(select the appropriate one)	
SMTPSEND.R60.CICSLOAD	CICS load modules for the Generic Edition
SMTPSEND.R60.OFP.CICSLOAD	CICS load modules for the OfficePath Edition and RAPID Editions
SMTPSEND.R60.OVFAX.CICSLOAD	CICS load modules for the OV/Fax Edition

Installation

Before you start the installation

Please review the Readme.txt file (from the FTP site). The only file you need to download to your zOS system, is the DOWNLOAD.JCL file. The documentation uses a dataset name prefix of, SMTPSEND.R60. You can use a different value to suit your environment. This can be changed in DOWNLOAD.JCL

You need to edit DOWNLOAD.JCL, before submitting it for execution:

- Review the comments in the JCL
- Add the accounting information required for your system.
- Obtain the FTP username & password from TBS, and replace the placeholder values.
- Select the edition of SMTP-Send to install, by uncommenting the appropriate lines.
- If you are changing the dataset name prefix, then replace the string ‘SMTPSEND.R60’ on each of the ‘get’ statements

The job should be executed, and end with a zero condition code.

Some of the installation jobs provided in INSTLIB are specific to particular editions of SMTP-Send. For example, when you run OPSI04, you will find several versions of this job, with a different letter suffix. Use the one corresponding to your SMTP-Send edition.

Job suffix	*SMTP-Send edition*
G	Standard edition
O	Officepath edition (includes RAPID)
F	OVFax edition
S	SSWDA edition

To install SMTP-Send

Step	Procedure

1 Download, and edit the DOWNLOAD.JCL dataset from the FTP site, before submitting it for execution. It will download the required .XMI datasets (TSO TRANSMIT format).

- Review the comments in the JCL
- Add the accounting information required for your system.
- Obtain the FTP username & password from TBS, and replace the placeholder values.
- Select the edition of SMTP-Send to install, by uncommenting the appropriate lines.
- If you are changing the dataset name prefix, then replace the string 'SMTPSEND.R60' on each of the 'get' statements

The job should be executed, and end with a zero condition code.

2 Edit the SMTPSEND.R60.RECEIVE.JCL dataset. This was created by the DOWNLOAD.JCL job. It will perform a TSO RECEIVE for the downloaded datasets, and then delete the .XMI datasets that were downloaded.

- Review the comments in the JCL
- Add the accounting information required for your system.
- Uncomment the appropriate lines, so that you perform a RECEIVE against each dataset downloaded
- If you are changing the dataset name prefix, then replace the string 'SMTPSEND.R60'

The job should be executed, and end with a zero condition code.

3 If SMTP-Send will be used under CICS, run installation job OPSI03 to add required CICS file definitions and update program definitions. Add CICSLOAD, to your DFHRPL dataset concatenation.

4 Run installation job OPSI04 to create the system configuration data set (ddname OPSCONFG). If SMTP-Send will be used under CICS and you do not use RDO to dynamically allocate the file, add the file to your CICS JCL.
Refer to CONFIGURING SMTP-SEND, for a description of the configuration parameters.

5 Verify your SMTP-Send installation with job OPSI05. Specify your e-mail destination address according to the commented instructions in the JCL. Use the OPST transaction to test your CICS installation.

6 **OP/SS** Run installation job OPSI06 to compile and link the name of the system configuration data set in load module OPSCFDSN. If OP/SS will be used under CICS, make sure the configuration data set name is specified for the OPSCONFG CICS file definition via either RDO or JCL.

7 **OP/SS** Run installation job OPSI07 to rename your existing OfficePath or RAPID load modules and copy the OP/SS modules into your current OfficePath or RAPID data set. When OPSI07 finishes you can use the PIPT (OfficePath) or ZAPT (RAPID) transaction to verify OP/SS under CICS.

8 If you plan to offer SMTP-Send support to your ISPF users, you may want to install the SMTP-Send ISPF libraries into your ISPF environment, however, this is not necessary, as the dialogs can be used without installation. See CUSTOMIZING ISPF PANEL SUPPORT for more details.

> **OP/SS** Be sure to change the OPSSNADS variable to Y if you want to allow users to address messages using SNADS format addresses such as *userid* or *userid.address*.

CICS installation

If you install one of the CICS editions, make sure that the CICS TCP/IP socket interface is active. See IBM book IP CICS Sockets Guide, (SC31-8518), for information on starting the CICS Socket interface. (If you are using TCP/IP for MVS 3.2, see CICS TCP/IP Socket Interface Guide and Reference, (SC31-7131).)

APF authorization

If you plan to use MVS security services in an SMTP-Send addressing exit or to log distributions to System Management Facility (SMF) from batch or TSO then SMTP-Send must be APF authorized. You must (a) relink the SMTP-Send load module with the AC(1) attribute and

(b) authorize the SMTP-Send load library. INSTLIB member, OPSI11 contains JCL to relink the batch load module with the AC(1) attribute. If you will be calling SMTP-Send from TSO and require APF authorization, add OPSGS3BA to the AUTHTSF section of SYS1.PARMLIB member IKJTSO00 and call OPSGS3BA instead of OPSGS3B.

[OP/SS For OfficePath or RAPID use PIPGS3BA or ZAPGS3BA respectively.]

It is not possible to APF authorize the SMTP-Send load module under CICS. If you are using RACF (or equivalent) security, then use the CICS-supplied security functions as described in the IBM book CICS-RACF Security Guide.

For more information about APF authorization, see the IBM book MVS/ESA Initialization and Tuning Guide. Remember that all other libraries that are concatenated with the SMTP-Send load library must also be APF-authorized.

Soft*Switch Application Toolkit Feature

The Soft*Switch Application Toolkit feature provides a program to replace the SSWDA program provided by SoftSwitch. During initial testing of this feature, it is advisable to create testing JCL based upon your existing JCL and to update only the testing JCL to prevent any interference with your production jobstreams.

If you have never used OfficePath previously

If OfficePath has never been installed by your organization, your distribution materials will be specially packaged to support the distribution of items via the Soft*Switch Toolkit API in the absence of a previous installation of OfficePath. The additional required OfficePath modules will have been already included in the LOADLIB libraries shipped or downloaded.

Testing the Application Toolkit Feature

To test the feature, skip installation step 7, 'Update your existing OfficePath libraries'. Instead add the SMTPSEND.Rxx.SSWDA.LOADLIB to your Toolkit API JCL either before or in replacement of, your existing Soft*Switch Toolkit API libraries. In addition, add the SMTP-Send configuration file to your Toolkit API programs by adding the file with DDname OPSCONFG to your JCL. This will insure that the replacement Toolkit API program is used instead of the modules shipped by Soft*Switch.

Installing SMTP-Send in production without JCL changes

To avoid any JCL changes for production usage, perform one of the following.

- rename the SSWDA program in your Soft*Switch Toolkit API LOADLIB and copy the OP/SS modules into your Soft*Switch Toolkit API LOADLIB

or

- rename the Soft*Switch Toolkit API and OP/SS LOADLIB datasets so that the OP/SS library replaces your Soft*Switch Toolkit API LOADLIB.

Then run INSTLIB member, OPSI06, to inform OP/SS of the data set name of your SMTP-Send configuration file.

Configuring SMTP-Send

SMTP-Send can be configured to communicate directly with an SMTP relay server (for example, Microsoft Exchange, or IBM Domino), or alternatively with either IBM's SMTP Server or TBS's SMTP Spool Server , both of which run on z/OS. Refer to the Customization or Adminstration guide for your release of TCP/IP for information on how to configure and start IBM's SMTP server.

Creating the OPSCONFG file

The OPSCONFG file contains system defaults for SMTP-Send as well as optional distribution list information. The OPSCONFG file is a VSAM KSDS file. This file is created by the OPSMKCFG program using input from two DD statements, CONFIG for general configuration information, DISTLIST to define distribution lists. The VSAM KSDS must be empty for the OPSMKCFG program to successfully store the information. The INSTLIB member, OPSI04G, deletes and defines a new cluster before running OPSMKCFG.

OP/SS The OPSCONFG file contains system defaults for OP/SS as well as tables to map your existing 8.8 SNADS-format user addresses to SMTP format. The OPSCONFG file is a VSAM KSDS file. This file is created by the OPSCONFG program using input from three DD statements, CONFIG for general configuration information, USERS to map user names, and NODES to map node names. The VSAM KSDS must be empty for the OPSCONFG program to successfully store the information. The INSTLIB member, OPSI04O, deletes and defines a new cluster before running OPSCONFG.

Syntax conventions

Each input file consists of a series of control statements, one statement per line.

Control statements consist of two fields, Field1 and Field2. Field1 must begin in the first column. Field2 begins at the first non-blank position following Field1 and is limited to 256 characters in length.

A control statement that begins with an asterisk (*) is a comment and is not processed. Likewise, blank control statements are ignored.

All Field1 values must be in upper-case. Field2 values are case-insensitive.

To continue a statement, place the first part of Field2 in quotes immediately followed by a plus sign (+). Each continued statement must begin with the same Field1 followed by the next part of Field2.

For example:

```
COMMAND "parm1,"+
COMMAND '"parm value 2"'+
COMMAND ",parm3"
```

is equivalent to:

```
COMMAND parm1,"parm value 2",parm3
```

Creating the OPSCONFG VSAM data set

```
XXOPSI04G JOB (accounting),CLASS=A
// EXEC PGM=OPSMKCFG,REGION=7192K
//STEPLIB  DD DSN=OPS.vers.LOADLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//CONFIG   DD DSN=OPS.vers.INSTLIB(CONFIG),DISP=SHR
//DISTLIST DD DSN=OPS.vers.INSTLIB(DISTLSTS),DISP=SHR
//OPSCONFG DD DSN=OPS.vers.OPSCONFG,DISP=SHR
//*
```

Typically you would create one OPSCONFG file for production usage and one or more variants for testing.

Configuration control statements

Setting the default domain name

The first control statement specifies the SMTP domain name for your enterprise. This domain name will be appended to all sender and recipient name that do not specify a domain name. You must include a DEFAULT_DOMAIN control statement.

DEFAULT_DOMAIN

DEFAULT_HOST

Purpose: specifies the default SMTP domain name for your enterprise

OP/SS DEFAULT_HOST serves a similar purpose to the OfficePath or RAPID "DEFAULT NODE" commands.

Syntax: `DEFAULT_DOMAIN domain`

Example: `DEFAULT_DOMAIN FOO.COM`

Setting the authorization code

SMTP-Send requires a 12-digit product authorization code that reflects each optional product feature that you have licensed. You can set the authorization code values using the following control statements:

BATCH_AUTHCODE

Purpose: specifies the SMTP-Send Batch feature authorization code

Syntax: `BATCH_AUTHCODE nnnn,nnnn,nnnn`

Default: none, SMTP-Send Batch operates in "demonstration" mode

Example: `BATCH_AUTHCODE 1234,5678,4321`

CICS_AUTHCODE

Purpose: specifies the SMTP-Send CICS feature authorization code

Syntax: `CICS_AUTHCODE nnnn,nnnn,nnnn`

Default: none, SMTP-Send CICS operates in "demonstration" mode

Example: `CICS_AUTHCODE 1234,5678,4321`

HTML_AUTHCODE

Purpose: specifies the SMTP-Send HTML Substitution feature authorization code

Syntax: `HTML_AUTHCODE nnnn,nnnn,nnnn`

Default: none, SMTP-Send HTML operates in "demonstration" mode

Example: `HTML_AUTHCODE 1234,5678,4321`

For more information on authorization codes, see the cover letter that came with your SMTP-Send distribution package.

Setting Time parameters

Specifying the SMTP Server timezone offset

All times passed to the SMTP server must specify the offset of the host system from Univeral Time (also known as GMT). The default value is "-500" or Eastern Standard time. If your location observes Daylight Savings Time, be sure to specify the offset for Standard time. SMTP-Send will automatically adjust for Daylight Savings Time.

TIMEZONE

Purpose: specifies the offset from UT of your host system from which you are sending items

Syntax: `TIMEZONE {+|-}HHMM`

Default: `TIMEZONE -0500`

Example: `TIMEZONE +0100`

Specifying observance of Daylight Savings Time

To correctly create the time zone offset for times passed to the SMTP server, SMTP-Send needs to know if the location of your SMTP server observes Daylight Savings Time.

DST

Purpose: specifies the offset from SMTP domain name of the host system from which you are sending items

Syntax: **`DST Yes|No`**

Default: **`DST Yes`**

Example: **`DST Yes`**

Specifying Daylight Savings Time rules

When Daylight Savings Time (DST) is being observed, SMTP-Send needs to know the rules for determining when a date/time value is within the DST period or outside of that period. The rules are specified by supplying the month, week and day of the week when DST begins and ends.
By default, the DST period is defined as the second Sunday in March to the first Sunday in November. (This is the period during which DST is observed in the United States.) The variables *begin_month*, *begin_week*, and *begin_day*, define the month, week, and day, respectively, when DST begins. The variables *end_month*, *end_week*, and *end_day* define when DST ends. Months are 1 (January) through 12 (December); weeks are 1 through 4; days are 1 (Sunday) through 7 (Saturday). You can use negative numbers to specify the weeks, where -1 is the last week of the month, -2 is the second to last week, and so on.

DSTLAW

Purpose: specifies the period for Daylight Savings Time
Syntax: **DSTLAW "*begin_month begin_week begin_day end_month end_week end_day*"**
Default: **DSTLAW "3 2 1 11 1 1"**
Example: **DSTLAW "4 -2 1 10 -1 1"**

Changing the SMTP Server information & behaviour

By default, SMTP-Send sends text message items by writing output to a SYSOUT external writer on SPOOL. The default external writer name is "SMTP" and the default output class is "B." When running under CICS, the JES node name is also required, the default used is "N1."

Binary items such as PC files, HTML, RFT-DCA are sent using MIME-encodings directly to an SMTP relay server using TCP/IP as the transport mechanism. The default SMTP Server IP address is built by appending the default domain to the string '"SMTP." The default TCP/IP port number for SMTP is 25. You can change the port number used by appending the ':' character followed by the desired number to the address value.

You can change these values using the following control statements:

ATSIGN

Purpose: identifies the character that separates the local part of an e-mail address from the domain portion. The ATSIGN configuration statement was introduced by IBM APAR PQ36249. Specify the same value as in your SMTP Server configuration data set (DDname CONFIG in the SMTP started task).
If ATSIGN is not specified in your SMTP Server configuration data set then you do not need to specify it for SMTP-Send.
Syntax: `ATSIGN char`
Default: `ATSIGN @`
Example: `ATSIGN #`

SetTCPFailureRC

Purpose: specifies the return code to be returned when SMTP-Send is unable to connect to both the primary and secondary SMTP server to send a message via TCP/IP.
Syntax: `SetTCPFailureRC retcode`
Default: `SetTCPFailureRC 4`
Example: `SetTCPFailureRC 8`

SMTP_RELAY_ADDRESS

Purpose: specifies the primary SMTP relay server that is used to send e-mail via TCP/IP sockets.
Syntax: `SMTP_RELAY_ADDRESS {Server|IPaddress}[:port]`
Default: `SMTP_RELAY_ADDRESS 127.0.0.1:25`
Example: `SMTP_RELAY_ADDRESS smtp.foo.com:725`

SMTP_RELAY_ADDRESS_ALTERNATE

Purpose: specifies the alternate SMTP relay server if a connection to the primary server cannot be established.
Syntax: `SMTP_RELAY_ADDRESS_ALTERNATE {Server|IPaddress}[:port]`

SMTP_SERVER_CLASS

Purpose: specifies the MVS SMTP server external writer class.
Syntax: `SMTP_SERVER_CLASS classname`
Default: `SMTP_SERVER_CLASS B`
Example: `SMTP_SERVER_CLASS X`

SMTP_SERVER_NAME

Purpose: specifies the MVS SMTP server external writer name. Use the name of the SMTP started task.
Syntax: `SMTP_SERVER_NAME name`
Default: `SMTP_SERVER_NAME SMTP`
Example: `SMTP_SERVER_NAME TBSSMTP`

SMTP_SERVER_NODE

Purpose: specifies the JES node that the MVS SMTP server runs on.
Syntax: `SMTP_SERVER_NODE name`
Default: `SMTP_SERVER_NODE N1`
Example: `SMTP_SERVER_NODE N2`

WTO_onTCPFailure

Purpose: specifies that SMTP-Send should issue a WTO when unable to connect to both the primary and secondary SMTP server to send a message via TCP/IP. The text of the message '`OPS126E FAILURE RELAYING TCPIP DATA` 'is defined in the SMTP-Send message table OPSSTRT0 as string number 126.
Syntax: `WTO_onTCPFailure {No|Yes}`
Default: `WTO_onTCPFailure No`
Example: `WTO_onTCPFailure Yes`

Controlling the SMTP transmission

By default, SMTP-Send sends messages in the MIME format using TCPIP as the transport mechanism. Attachments may be sent as binary data or translated from EBCDIC to ASCII.

You can change this behavior using the following control statements:

DEFAULT_ATTACHMENT

Purpose: defines the formatting for text attachments. Binary attachments are not affected by this setting. To override this setting for a specific SEND operation, specify `OUTTYPE RTF` or ATTACH attachment `TRANSLATE RTF`
Syntax: `DEFAULT_ATTACHMENT {RTF|NORTF}`
Default: `DEFAULT_ATTACHMENT NORTF`
Example: `DEFAULT_ATTACHMENT RTF`

MAXMAILBYTES

Purpose: limits the size of e-mails that are passed to the outbound SMTP server. The value should be the same as specified in your SMTP server configuration. For the MVS SMTP server this value is specified via the MAXMAILBYTES statement of the SMTP Server Configuration data set (DDname CONFIG in the SMTP started task). Refer to the documentation for your SMTP server if you use a different server for TCP/IP connections. The minimum value is 1 and the maximum value is 4 gigabytes. Do not use a decimal point in the value, rather than 1.5G you should use 1536M.
A value of 0 (zero) means no size checking will be performed.
Syntax: `MAXMAILBYTES 0|`*`nnnnnnnnnn`*`|`*`nnnnnn`*`K|`*`nnnn`*`M|nnG`
Default: `MAXMAILBYTES 4G`
Example: `MAXMAILBYTES 512M`

SEND_TEXT

Purpose: indicates the preferred route for simple outbound SMTP messages. Specifying TCPIP results in delivery of e-mail via a direct TCP/IP socket interface to an SMTP server capable of relaying the message. If unable to contact both the primary and secondary SMTP server via TCP/IP, SMTP-Send continue as if SPOOL was specified for that distribution only.
Specifying SPOOL results in delivery of e-mail to an MVS-based SMTP relay server via the JES SPOOL interface. The IBM-supplied SMTP Server started task or the SMTP-Send Spool Server are both examples of supported SMTP relay servers.
Syntax: `SEND_TEXT {TCPIP|SPOOL}`
Default: `SEND_TEXT TCPIP`
Example: `SEND_TEXT SPOOL`

Handling empty Message or Attachment(s)

These parms allow you to control the program behavior when the message data or one or more attachment datasets are empty. You can set the program return code, and determine if empty attachments are to be included in the email. If some or all of the attachments are empty, you can control the return code with the parms below.
A message or attachment dataset can be considered empty in 2 ways, a) exists and is empty, or b) doest not exist at all (a non-existent PDS member, or an ATTACH DSN:*non-existent dsname*). In either case, SMTP-Send will handle both conditions as an empty dataset.

RC_on_All_Attach_Empty

Purpose: specifies the return code for a send where all of the attachment files are empty.
Syntax: `RC_on_All_Attach_Empty {`*`number`*`}`
Default: `RC_on_All_Attach_Empty 4`
Example: `RC_on_All_Attach_Empty 8`

RC_on_Any_Attach_Empty

Purpose: specifies the return code for a send where any of the the attachment files are empty..
Syntax: `RC_on_Any_Attach_Empty {number}`
Default: `RC_on_Any_Attach_Empty 0`
Example: `RC_on_Any_Attach_Empty 4`

RC8_on_NoData

Purpose: Overrides the default return code (RC=4) returned when the input MESSAGE dataset is empty and no attachments are requested, or they are empty. The email is not sent when a NoData condition is encountered.
If there are any attachments, the NoData condition will not occur, and the email will be sent.
Syntax: `RC8_on_NoData {Yes|No}`
Default: `RC8_on_NoData No`
Example: `RC8_on_NoData Yes`

Include_Attach_Empty

Purpose: specifies whether or not to include empty attachment files in the current email. This statement is a replacement for the older statement, `SendEmptyAttachments.`
Syntax: `Include_Attach_Empty {Yes|No}`
Default: `Include_Attach_Empty Yes`
Example: `Include_Attach_Empty No`

SkipSend_on_All_Attach_Empty

Purpose: specifies whether or not to skip send of the current email, if all attachment files are empty.
Syntax: `SkipSend_on_All_Attach_Empty {Yes|No}`
Default: `SkipSend_on_All_Attach_Empty No`
Example: `SkipSend_on_All_Attach_Empty Yes`

SendIfOnlyMSG

Purpose: specifies whether or not the email is sent, when a NoData condition exists, but there is data provided in the MESSAGE control statement. The default is to send. This configuration value can be overridden for any job, by using the *SendIfOnlyMSG* control statement. Please see Special Conditions, NoData, in the SMTP-SEND FOR Z/OS USER GUIDE for further details.
Syntax: `SendIfOnlyMSG {Yes|No}`
Default: `SendIfOnlyMSG Yes`
Example: `SendIfOnlyMSG No`

Highlighting Notices in Microsoft Outlook

You can highlight Notices that are received by users of Microsoft Outlook. This feature works as if the sender had used Outlook's Flag for Follow Up option in the Actions menu. This feature may also work with other e-mail client software that recognizes the X-Message-Flag parameter in the SMTP header.

MSG_IN_XMESSAGEF

Purpose: specifies that Notices are highlighted using Outlook's Flag for Follow Up feature.
Syntax: `MSG_in_XMESSAGEF [Yes|No]`
Default: `MSG_in_XMESSAGEF No`
Example: `MSG_in_XMESSAGEF Yes`

Redirecting output

Normally you would want SMTP-Send to send items to recipients using your SMTP server, however you can optionally redirect the transmission. When running in Batch, redirection is to a sequential file. When running under CICS, redirection is to a CICS Temporary Storage Queue. This feature is useful when you want to test an e-mail enabled application without actually sending items to recipients. Redirection can be specified for both Batch and CICS in the same CONFIG file.

TESTOUTDSN

Purpose: redirects batch output to a sequential data set
Syntax: `TESTOUTDSN datasetname[,keywords...]`
Example: `TESTOUTDSN TBS123.OPSMTP.OUTPUT,DISP=SHR`

SMTP-Send always supplies DDNAME, DSORG, RECFM, LRECL and BLKSIZE attributes. The value of DDNAME is `DATAOUT`. If required you can supply additional attributes using standard MVS JCL keywords or through dynamic allocation. See "Dynamic allocation keywords" for a list of valid keywords.

TESTOUTTSQ

Purpose: redirects CICS output to a CICS Temporary Storage Queue
Syntax: `TESTOUTTSQ 8-character-tsqueue`
Example: `TESTOUTTSQ PIPTSQO`

Note: CICS Temporary Storage queue names are case sensitive. Normally you would not include TESTOUTDSN or TESTOUTTSQ in your production OPSCONFG file. Instead create a special OPSCONFG file for running tests that contains a TESTOUTDSN or TESTOUTTSQ statement.

Handling addressing errors

If an addressing error occurs for one or more recipients, SMTP-Send sends the message to the remaining recipients. SMTP-Send automatically eliminates some forms of addressing errors by append your domain name to an address without one. You can turn this behavior off and create additional addressing errors with the `Append_Domain_Name` statement

If you want to stop SMTP-Send when it encounters an addressing error, set `FAIL_ON_ERROR` to YES. Set `MSG_ON_ERROR` to YES to return the message to the sender's REPLY-TO address.

When `FAIL_ON_ERROR` is set to NO, SMTP-Send sends the message as long as the sender and one or more recipient addresses are valid. If there are no valid addresses, SMTP-Send ends with a non-zero return code.

To enhance security, you can also instruct SMTP-Send to terminate with a return code of 32 when an address is denied by your address validation exit by setting `Fail_on_Validation_Error` to YES

OP/SS To maintain compatibility with the behavior of the original OfficePath and RAPID software, even when OP/SS is unable to map or locate at least one recipient address, it ends execution with a condition code of zero. You can override this characteristic using the `ADDRESS_MAP_FAIL_OK` statement.

APPEND_DOMAIN_NAME

Purpose: controls whether or not the default domain name is added to a TO, CC or BCC recipient value when no domain has been specified.
Syntax: `APPEND_DOMAIN_NAME {Yes|No}`
Default: `APPEND_DOMAIN_NAME Yes`
Example: `APPEND_DOMAIN_NAME No`

FAIL_ON_ERROR

Purpose: controls the SMTP-Send actions when addressing errors are encountered
Syntax: `FAIL_ON_ERROR {Yes|No}`
Default: `FAIL_ON_ERROR No`
Example: `FAIL_ON_ERROR Yes`

FAIL_ON_VALIDATION_ERROR

Purpose: controls the SMTP-Send actions when addresses fail validation
Syntax: `FAIL_ON_VALIDATION_ERROR {Yes|No}`
Default: `FAIL_ON_VALIDATION_ERROR No`
Example: `FAIL_ON_VALIDATION_ERROR Yes`

MSG_ON_ERROR

Purpose: controls the sending of error notifications when addressing errors are encountered
Syntax: `MSG_ON_ERROR {Yes|No}`
Default: `MSG_ON_ERROR Yes`
Example: `MSG_ON_ERROR No`

ERROR_MSG_TO_NAME

Purpose: specifies the SMTP address to which error messages will be sent if a reply address cannot be obtained from either a REPLY- TO or FROM statement. If no domain is specified, the default_domain value will be appended.
Syntax: `ERROR_MSG_TO_NAME name`
Default: `ERROR_MSG_TO_NAME postmaster@default_domain`
Example: `ERROR_MSG_TO_NAME postmaster@foo.com`

ERROR_MSG_FROM_NAME

Purpose: specifies the SMTP address from which error messages will be sent. If no domain is specified the default_domain value will be appended.

Syntax: `ERROR_MSG_FROM_NAME name`
Default: `ERROR_MSG_FROM_NAME OPSend@default_domain`
Example: `ERROR_MSG_FROM_NAME batch [or batch@tbssoft.com]`

SEND_NDR

Purpose: specifies the format of non-delivery notices. *Yes,* selects the new NDR notice which allows the recipient to resend the original message. *No*, preserves the existing text notice.
Syntax: `SEND_NDR {Yes|No}`
Default: `SEND_NDR Yes`
Example: `SEND_NDR No`

EXPAND_DIST_LIST

Purpose: controls the expansion of distibution lists in the message headers. Default value `YES` includes the e-mail address of each member of a distribution list in the message headers. Value `NO` suppresses this information.
Syntax: `EXPAND_DIST_LIST {Yes|No}`
Default: `EXPAND_DIST_LIST Yes`
Example: `EXPAND_DIST_LIST No`

UseDLAddress

Purpose: controls the expansion of Distribution lists during LDAP processing. By default, SMTP-Send will not expand a Distribution List with an email address associated with it, but instead will merely send the email to that address, letting the receiving email system expand the address. This is a change from the default behavior in releases prior to Release 4.2.13 which can be restored by specifying `UseDLAddress No`
Syntax: `UseDLAddress {Yes|No}`
Default: `UseDLAddress Yes`
Example: `UseDLAddress No`

Enabling user exits

You can further customize SMTP-Send by writing user exits. The following statements tell SMTP-Send which exits you want to use. For information on how to code these exits, see CUSTOMIZING USER EXITS.

VALIDATEPGM

Purpose: specifies a program to validate addresses and optionally to reformat messages for particular addresses such as pagers. The program name is limited to 7 characters as SMTP-Send appends the character 'B' or 'C' to the name when running in Batch/TSO or CICS respectively.
Syntax: `VALIDATEPGM program[,variables]`
Default: none
Example: `VALIDATEPGM VALPROG`

DISTLISTPGM

Purpose: specifies a program to expand distribution lists. The program name is limited to 7 characters as SMTP-Send appends the character 'B' or 'C' to the name when running in Batch/TSO or CICS respectively.
Syntax: see: USING LDAP

LDAP_LOOKUP_PGM

Purpose: specifies a program to look up e-mail addresses. The program name is limited to 6 characters as SMTP-Send may append additional characters such as 'B' or 'C' to the name when running in Batch/TSO or CICS respectively.
Syntax: see: USING LDAP

LDAP_DISTLIST_PGM

Purpose: specifies a program to expand e-mail distribution lists. The program name is limited to 6 characters as SMTP-Send may append additional characters such as 'B' or 'C' to the name when running in Batch/TSO or CICS respectively.
Syntax: see: USING LDAP

DATAEXITPGM

Purpose: specifies a program to manipulate the message data before it is passed to SMTP for distribution. The program name is limited to 7 characters as SMTP-Send appends the character 'B' or 'C' to the name when running in Batch/TSO or CICS respectively. This definition does not cause the exit program to be invoked. You must request the data exit be invoked either through the `ALWAYSCALLDATAEXIT` configuration statement or from the address validation exit.

Syntax: `DATAEXITPGM program[,variables]`
Default: none
Example: `DATAEXITPGM DATAXT`

ALWAYSCALLDATAEXIT

Purpose: specifies whether the data manipulation exit is invoked for each SMTP-Send request. The invocation of the data exit can also be controlled from the address validation exit.
Syntax: `ALWAYSCALLDATAEXIT {Yes|No}`
Default: `ALWAYSCALLDATAEXIT No`
Example: `ALWAYSCALLDATAEXIT Yes`

EXIT_PASSWORD

Purpose: Provides the password used in the MAPPINGPGM exit. This value is encrypted in the OPSCONFG VSAM file. For the DISTLISTPGM exit program, the LDAP_Password value is used when the exit program name begins with LDAP and the Exit_Password value is used otherwise.
Syntax: `Exit_Password value`
Default: none

LDAP_PASSWORD

Purpose: Provides the password value to be encrypted in the OPSCONFG VSAM file. Specify a * in actual LDAP statements where the password is to be replaced
Syntax: `LDAP_Password value`
Default: none

LDAP_OPENRETRY

Purpose: number of times to attempt to connect to LDAP server
Syntax: `LDAP_OpenRetry count`
Default: `LDAP_OpenRetry 3`

LDAP_OPENDELAY

Purpose: delay between connection attempts to LDAP server
Syntax: `LDAP_OpenDelay seconds`
Default: `LDAP_OpenRetry 5`

LDAP_TIMEOUT

Purpose: time to wait before aborting LDAP search
Syntax: `LDAP_Timeout seconds`
Default: `LDAP_Timeout 60`

Controlling the look of the message

CREATE_MAILTO_LINKS

Purpose: controls whether or not email addresses and URLs are converted into HTML links when encountered in the subject or the body of a text message.
Syntax: `Create_MailTo_Links {Y|N}`
Default: `Create_MailTo_Links Y`

BODY_FONT-SIZE

Purpose: specifies the font size used in the Cascading Style Sheet specifications for the body of the text.
The receiving email client will use the first value it recognizes.
There are two types of font-family values:
font-family: The name of a font-family, like "verdana" or "arial"
generic-family: The name of a generic font-family, like "serif" or "sans-serif"
Syntax: `BODY_Font-Size {pointsize |`
`{xx-small|x-small|small|medium|large|x-large|x-large|xx-large}}`
Default: `BODY_Font-Size medium`
Example: `BODY_Font-Size 10px`

BODY_FONT-FAMILY

Purpose: specifies the Font family used in the Cascading Style Sheet specifications for the body of the text.
Syntax: `BODY_Font-Family {font [,font...}`
Default: `BODY_Font-Family none`

Example: `BODY_Font-Family sans-serif`

BODY_COLOR

Purpose: specifies the color used for the foreground and background of the body of the text.
Syntax: `BODY_Color {background_color[,foreground_color]}`
where color choices are:

Aqua	BLAck	BLUe	Fuschia
GRAy	GREen	Lime	Maroon
Navy	Olive	Purple	Red
Silver	Teal	White	Yellow

Default: `BODY_Color {White,Black}`

HTMLFONT

Purpose: defines the relative font size of preformatted text to be defined using the HTML <FONT=value> tag. The default value of none, indicates that no HTML <FONT> tags are to be included with the text. HTMLFONT is ignored if the data is not being converted from plain text to HTML
Syntax: `HTMLFONT {|1|2|3|4|5|6|7}`
Default: `HTMLFONT none`

HTMLTEXTHEADER

Purpose: specifies HTMLtag used to indicate the start of preformatted text when 1403/text data is being converted to HTML.
Syntax: `HTMLTextHeader text`
Default: `HTMLTextHeader <PRE>`
Example: `HTMLTextHeader "<font=Helv SIZE=2><PRE>`

HTMLTEXTTRAILER

Purpose: specifies HTMLtag used to indicate the end of preformatted text when 1403/text data is being converted to HTML
Syntax: `HTMLTextTrailer text`
Default: `HTMLTextTrailer </PRE>`
Example: `HTMLTextTrailer "</PRE></FONT>`

MAPHTMLINTEXT

Purpose: specifies whether the characters '<', '&' and '>' within the message text should be mapped into HTML symbolics to prevent them from being interpreted as HTML.
Syntax: `MapHTMLinText {Y|N}`
Default: `MapHTMLinText No`
Example: `MapHTMLinText Yes`

ORIENT

Purpose: specifies the default orientation used for RTF attachments when no value is specified in the `RTF` subparameter of `OUTTYPE`
Syntax: `ORIENT {Portrait | Landscape}`
Default: `ORIENT Landscape`

PITCH

Purpose: specifies the default point size used for RTF attachments when no value is specified in the `RTF` subparameter of `OUTTYPE` or a `PITCH` control statement
Syntax: `PITCH pointsize`
Default: `PITCH 12`

REMOVETRAILINGBLANKS

Purpose: controls whether or not SMTP-Send should remove trailing blanks from each line of data before sending
Syntax: `RemoveTrailingBlanks {Yes|No}`
Default: `RemoveTrailingBlanks` **Y**

REPLACELFWCRLF

Purpose: specifies whether LineFeed control characters are converted to CarriageReturn LineFeed control sequences in text HFS files.
Syntax: `ReplaceLFwCRLF {Y|N}`
Default: `ReplaceLFwCRLF Y`

SENSITIVITY_MSG

Purpose: location of the sensitivity text. The actual text is controlled through the OPSCONFG statements `SENSITIVITY_TXT`

Syntax: `Sensitivity_Msg {None | Top | Bottom}`
Default: `Sensitivity_Msg None`

SENSITIVITY_TXT

Purpose: code as many SENSITIVITY_TXT statements as required for your text. HTML formatting tags may be used in the text. See the example in INSTLIB(CONFIG).
Personal Sensitivity (Type=1)
Private Sensitivity (Type=2)
Confidential Sensitivity (Type=3)
Company Confidential Sensitivity (Type=4)

Syntax: `Sensitivity_Txt type,text`
Default: `Sensitivity_Txt None`
Example:
```
Sensitivity_Txt 1,'<FONT COLOR="Blue"><B>'
Sensitivity_Txt 1,"This E-mail is Personal \n\"
Sensitivity_Txt 1,"only for the use of the recipient(s)\n"
Sensitivity_Txt 1,'</B></FONT>'
```

Adding a standard Notice to each message

You can add standardized text referred to as a NOTICE, at the bottom of some or all messages. You can define up to 1001 different NOTICEs in the Configuration file. Each NOTICE is identified by a 3 digit id. A blank NOTICE id can be used to specify the default NOTICE, otherwise the lowest id value becomes the default.

Each SMTP-Send job will be given the NOTICE text according to the following hierarchy:

1. jobs that include a 'NOTICE xxx' control statement will get NOTICE xxx

   …if the job does not specify a NOTICE control statement, or that NOTICE is not defined in the configuration file, then it will get

2. the NOTICE that matches the language-specific codepage for that job. This codepage may either be specified using the CODEPAGE control statement, or default to US English (037). This allows you to automatically include NOTICE text matching the language of the message.

   …if the language specific NOTICE set is not defined in the configuration file, then it will get,

3. the default NOTICE text.

   …if there is no default NOTICE text, then no NOTICE text is added.

If you want most or all messages to have a NOTICE added, then define your default NOTICE with the desired text. To send 'exception' messages without any NOTICE, define another NOTICE with a single blank character, and add this NOTICE id to the 'exception' jobs control statements.

If you want most messages to have no NOTICE added, then define your default NOTICE with a single blank character. To send 'exception' messages with a NOTICE, define another NOTICE with the desired text, and add this NOTICE id to the 'exception' jobs control statements.

NOTICE text can be any length, but the text in each NOTICE statement must be no more than 128 characters.

Use the string "\n" to insert a line break. You can also use standard HTML tags.

NOTICE text is not automatically included in HTML mail merge template merges. If you want to include notice text, use the keyword symbol ?SS_NOTICE or XML entity &SS_NOTICE; to indicate where to insert the notice text in the template.
NOTICE text is also not added to INTYPE HTML messages, in order to preserve the formatting of the HTML.

NOTICE

Purpose: adds a standard notice at the bottom of each message.
Syntax: `NOTICE [nnn,]"text"`
Example:
```
*-- Default NOTICE (has no id)
NOTICE    '<P STYLE="FONT-FAMILY:TREBUCHET;FONT-SIZE:90%;COLOR:SEAGREEN">'
NOTICE    "-----------------------------------------------------------------\n"
NOTICE    "*** DEFAULT NOTICE DEFAULT NOTICE DEFAULT NOTICE ***\n"
NOTICE    "This default NOTICE text is applied when no NOTICE matches"
NOTICE    " the CodePage, and no other NOTICE was specified."
NOTICE    '</P>'
*-------------------------------------------------------------
*---NOTICE, for "empty" notice---------------------------------
```

```
NOTICE    100,' '
*-------------------------------------------------------------
*---NOTICE, for SPANISH codepage-------------------------------
NOTICE    284,'<P STYLE="FONT-FAMILY:TREBUCHET;FONT-SIZE:90%;COLOR:SEAGREEN">'
NOTICE    284,"-------------------------------------------------------------\n"
NOTICE    284,"*** NOTICE 284   NOTICE 284   NOTICE 284 ***\n"
NOTICE    284,"This SPANISH NOTICE, used for CodePage 284 emails, "
NOTICE    284,'unless another NOTICE is explicitly specified \n'
NOTICE    284,</P>"
*-------------------------------------------------------------
*---NOTICE, for US ENGLISH codepage----------------------------
NOTICE    037,'<P STYLE="FONT-FAMILY:TREBUCHET;FONT-SIZE:90%;COLOR:SEAGREEN">'
NOTICE    037,"-------------------------------------------------------------\n"
NOTICE    037,"*** NOTICE 037   NOTICE 037   NOTICE 037 ***\n"
NOTICE    037,'US English NOTICE, used for all CodePage 037 emails, '
NOTICE    037,'unless another NOTICE is explicitly specified </P>'
*-------------------------------------------------------------
```

USE_CSS_STYLES

Purpose: specifies that HTML messages are formatted using Cascading Style Sheets. The `BODY_xxxx` statements control statements control the style characteristics.
A useful primer for CSS usage can be found at http://www.w3.org/Style/Examples/011/firstcss
Syntax: `Use_CSS_Styles {Y|N}`
Default: `Use_CSS_Styles Y`

USECUSTOMCODEPAGE

Purpose: controls whether or not SMTP-Send should attempt to load an external character translation table, named XLT*nnn* where *nnn* is the value specified on a CODEPAGE control statement. The XLT*nnn* module is created using INSTLIB member OPSI12
Syntax: `UseCustomCodepage {Y|N}`
Default: `UseCustomCodepage` **N**

Controlling conversion to Adobe PDF format

PDFBIAS

Purpose: Screen causes SMTP-Send to create PDF documents suitable for screen display. Text is rendered in 12pt Courier, with page length and width calculated from the lines per page and characters per line, Top, bottom, left and right margins are all set to zero.
Print causes SMTP-Send to use the PDF defaults as below or their overridden values.
Syntax: `PDFBIAS {Screen | Print}`
Default: `PDFBIAS Screen`

PDFPITCH

Purpose: specifies the font size to use for text in the PDF document
Syntax: `PDFPITCH` *`pointsize`*
Default: `PDFPITCH 12`

PDFORIENT

Purpose: specifies the orientation to use for text in the PDF document
Syntax: PDFORIENT {Portrait | Landscape}
Default: PDFORIENT Landscape

PDFPAPER

Purpose: specifies common page sizes (height and width may be overridden by PDFWIDTH and PDFHEIGHT statements)
Syntax: `PDFPAPER {LETter | LEGal | LEDger | A3 |A4}`
Default: `PDFPAPER LETTER`

PDFWIDTH

Purpose: specifies the page width in inches (the default) or millimeters
Syntax: PDFWIDTH inches | millimetersmm
Default: PDFWIDTH 8.5

PDFHEIGHT

Purpose: specifies the page height in inches (the default) or millimeters
Syntax: PDFHEIGHT inches | millimetersmm
Default: PDFHEIGHT 11

PDFTOPM

Purpose: specifies the top margin in inches (the default) or millimeters
Syntax: PDFTOPM inches | millimetersmm
Default: PDFTOPM 8.5

PDFBOTM

Purpose: specifies the bottom margin in inches (the default) or millimeters
Syntax: PDFBOTM inches | millimetersmm
Default: PDFBOTM 8.5

PDFLM

Purpose: specifies the left margin in inches (the default) or millimeters
Syntax: PDFLM inches | millimetersmm
Default: PDFLM 8.5

PDFRM

Purpose: specifies the right margin in inches (the default) or millimeters
Syntax: PDFRM inches | millimetersmm
Default: PDFRM 8.5

Notes for PDF configuration statements:

- leading zeros are required for decimal inch values less than 1, for example PDFTOPM 0.5 is valid and PDFTOPM .5 is not.
- there is no space between the millimeters value and the mm indicator, for example PDFWIDTH 216mm

Debugging SMTP-Send operation

DEBUG

Purpose: shows each sender or recipient address (a) as coded in CONTROL control statements, (b) as altered by an optional customer address mapping exit, if one exists.
Syntax: `DEBUG Yes|No|Verbose`
Default: `DEBUG No`
Example: `DEBUG Yes`
Note: to enable DEBUG only for a particular job, add `PARM='=SETDEBUG=YES'` on the `EXEC PGM=OPSGS3B` JCL statement

Sample CONFIG file

Here is an example of the CONFIG section of an OPSCONFG file that shows how the above control statements are coded:

```
DEFAULT_DOMAIN       TBSSOFT.COM
AUTHCODE             1234,5678,1234
TIMEZONE             -0500
DST                  yes
DSTLAW               "4 -2 1 10 -1 1"
SMTP_SERVER_CLASS    X
SMTP_SERVER_NAME     TBSSMTP
SMTP_SERVER_NODE     N5
SMTP_RELAY_ADDRESS   123.210.123.210:25
ATSIGN               #
TESTOUTDSN           OPS.TESTOUT,disp=old,ndisp=catlg
TESTOUTTSQ           OPSTSQO
FAIL_ON_ERROR        Yes
ERROR_MSG_FROM_NAME  Postmaster@tbssoft.com
ERROR_MSG_TO_NAME    Errors_go_here@tbssoft.com
MSG_ON_ERROR         yes
EXPAND_DIST_LIST     No
VALIDATEPGM          VALIDP
DATAEXITPGM          DATAXT
ALWAYSCALLDATAEXIT   Yes
DEBUG                VERBOSE
```

```
LDAP_Password        LDAPPW
LDAP_OpenRetry       3
LDAP_OpenRetry       5
LDAP_Timeout         60
Exit_Password        "Exit Pw"
Sensitivity_Msg      Bottom
Sensitivity_Msg      None
ReplaceLFwCRLF       Y
Use_CSS_Styles       Y
BODY_Font-Size       medium
BODY_Font-Family     Helv
BODY_Color           {Black,White}
HTMLFONT             5
HTMLTextHeader       "<font=Helv SIZE=2><PRE>
HTMLTextTrailer      "</PRE></FONT>
PDFBIAS              Print
PDFPITCH             12
PDFORIENT            Landscape
PDFWIDTH             9
PDFHEIGHT            11
PDFTOPM              0.5
PDFBOTM              0.25
PDFLM                25mm
PDFRM                25mm
```

Distribution list control statements

SMTP-Send allows you to group recipients together into distribution lists. These lists can be managed by SMTP-Send or by an external directory manager.

Distribution list control statements are specified in the OPSCONFG file via the DISTLIST DDname. They consist of two components, the distribution list name and the list member. The list name can be up to 18 characters in length and begins in column 1. The list member value can be up to 60 characters in length and does not need to contain the domain name, which will be automatically appended by SMTP-Send if not present.

Repeat the list name component on subsequent control statements to specify additional members of the distribution list. All members of a distribution list must be specified sequentially, one after the other.

Here is an example of the DISTLIST section of an OPSCONFG file that shows a couple of sample distribution lists:

```
MANAGERS             alberta@foo.com
MANAGERS             bernie@foo.com
MANAGERS             charley@foo.com
MANAGERS             denise@foo.com
MANAGERS             emily@foo.com
VENDORS              support@tbssoft.com
VENDORS              sales@tbssoft.com
```

When you specify a recipient address that has no domain name, for example, sales, SMTP-Send first checks to see if it is a distribution list name. If it is not a distribution list then SMTP-Send appends the default domain name to create a complete address, for instance, sales@tbssoft.com.

OP/SS OfficePath/SNADS and RAPID RECIPIENT statements are eligible for distribution list processing only if a single UserID value and no Address value is specified on the statement.

OP/SS NODES and USERS address mapping control statements

OP/SS uses address mapping control statements to specify how 8.8 SNADS-format addresses should be mapped to their SMTP-format (RFC 822) counterparts. SNADS format addresses consist of two components, each up to 8 characters, separated by a period. The first component specifies the name of the message recipient or user, and the second component specifies the host address or node. These are the same values that appear on OfficePath or RAPID RECIPIENT or USERNAME control statements.

SMTP format addresses have the form user@domain.com. For example TBS Software's technical support address is support@tbssoft.com.

The OPSCONFG file has two address mapping tables. The NODES table maps SNADS node names to SMTP domain names and the

USERS table maps SNADS user names to SMTP user names.

OP/SS searches the mapping tables for Field1 values that match SNADS user and node values. When OP/SS finds a match, it uses the value of Field2 to determine the corresponding SMTP value. If OP/SS cannot find a match for the node it substitutes the value specified by DEFAULT_HOST. Finally OP/SS concatenates the user and domain values of with the standard "@" separator.

For example, if your OfficePath or RAPID application includes these CARDFILE control statements:

```
USER NAME TSO
RECIPIENT USER2    LOCATN2
RECIPIENT TOUSER   PIPHOST1
```

and your OPSCONFG file includes these statements:

CONFIG DD:
```
DEFAULT_HOST tbssoft.com
```

NODES DD:
```
PIPHOST1     keane.uk.com
LOCATN2      tbs2.com
```

USERS DD:
```
USER2        sales
TOUSER       support
TSO      tso
```

then the message item will be sent using this header information:
```
FROM:   tso@tbssoft.com
TO:     sales@tbs2.com
TO:     support@keane.uk.com
```

OP/SS

Configuring and customizing Soft*Switch Application Toolkit Feature

The Soft*Switch Application Toolkit feature program, SSWDA can be configured to change the default behavior when certain types of data are being sent. Some customizations are performed using the OPSCONFG control statements and others are performed by altering the SSWDA program itself.

SSWDA OPSCONFG control statements

SSWMSGEncode

Purpose: specifies the manner in which text from the OBJECT dataset is to be displayed when the MSG= statement is specified in the //COMMAND statements. No specifies that OBJECT text be sent instream. Yes specifies that OBJECT text be sent as an attachment . Both specifies that the OBJECT text be included both instream and as an attachment.
Syntax: `SSWMSGEncode {No|Yes|Both}`
Default: `SSWMSGEncode No`
Example: `SSWMSGEncode Both`

SSW_Header_Compatibility

Purpose: specifies the header text inserted before an attachment. Yes instructs SSWDA to
a) convert the DOCUMENTNAME value into a MS-DOS compatible where the 1st eight characters are taken from the 1st eight non-blank and non-punctation characters
b) to create a multi-line message description instead of SSWENCODE 1.0;ASCI.
Syntax: `SSW_Header_Compatibility {No|Yes}`
Default: `SSW_Header_Compatibility No`
Example: `SSW_Header_Compatibility Y`

SSWDA Program customization

The SSWDA program dynamically allocates the SYSOUT dataset used by OP/SS for its output as SYSOUT=A. You can change the JES SYSOUT class used. You can also customize SSWDA to force
a) all messages with DOCUMENTTYPE=FINL to be sent as attachments or
b) all messages with DOCUMENTCLASS=FINALDOC to be sent as attachments

These customizations are performed by updating the SSWOPTS member in INSTLIB and then running the INSTLIB job OPSISSWD.

Customizing user exits

Performing your own distribution list processing

SMTP-Send allows you to use distribution lists either through the Configuration file or using LDAP. If you need to perform your own expansion of distribution lists, you can also do so using the distribution list expansion user exit.

You specify the expansion exit program name via the DISTLISTPGM Configuration statement. As each e-mail address is encountered in the input control statements it is inspected for the @ character that separates the host from the domain. If no @ is found, the e-mail address is eligible for distribution list processing and is passed to the expansion exit program for expansion. Each address returned by the distribution list exit program will still be passed to the address validation exit for validation.

The expansion exit program name on the DISTLISTPGM statement is restricted to 7 characters. When SMTP-Send runs in MVS Batch or TSO the character 'B' is appended to the program name. Under CICS, the character 'C' is appended. For example, under CICS the statement DISTLISTPGM DISTLUX causes program DISTLUXC to be used to expand distribution lists.

You can pass parameters to your expansion exit by adding them to the DISTLISTPGM Configuration statement following the program name. Enclose the entire value in double quotes as follows:

```
DISTLISTPGM "DISTLUX,IPAddress=127.0.0.0"
```

The address of the string `IPAddress=127.0.0.0` will be passed to the expansion exit in the passed data structure along with the length of the string. If the length is zero, no parameters were supplied.

The parameters passed to the DISTLISTPGM program are stored in plain text in the Configuration file. If you need to pass a password value to your exit program, you can code an asterisk character, *, in place of the password in your parameter string. SMTP-Send will use the value from the `Exit_Password` or `LDAP_Password` configuration statements as appropriate in place of the * when your exit is invoked.

Sample S/390 Assembler expansion exit programs for Batch and CICS are in INSTLIB members DISTLUXB and DISTLUXC.

Parameter List

The exit program is passed the following data structure:

Field	Description
address[4]	Contains a pointer to a 12-byte workarea that is returned after each call to the user exit. This can be used to retain information between calls.
int[4]	Contains the return code from the user exit. This is in addition to the value returned via R15, since CICS applications are unable to return values in registers.
address[4]	Contains a pointer to any parameters that may have been supplied on the DISTLISTPGM control statement. If no parameters were provided, the pointer is zero.
byte[1]	Contains the length of the parameters supplied on the DISTLISTPGM control statement. If no parameters were provided, the length is zero.
byte[1]	Contains the length of a message returned by the user exit. If no message is returned this value should be set to zero. The maximum length that can be displayed by SMTP-Send is 125 characters.
int[2]	Contains a binary number with the sequence number of the member within the distribution list.
character[125]	Contains a message from the user exit to be displayed by SMTP-Send.
byte[1]	On entry to the user exit contains the length of the SMTP name passed to the user exit. On return it contains the length of the SMTP name returned by the user exit.
character[128]	Contains the distribution list name. The user exit should use this field to return a member of the distribution list.
byte[1]	Reserved.
byte[1]	x'00'or x'40' indicates V1 format, C'1' indicates V2 format and presence of DLDATA extension
int[2]	time in seconds to wait for a response from a LDAP search request
int[2]	number of times to attempt to connect to the LDAP server
int[2]	time in seconds to wait between LDAP connection attempts

DLDATA Extension

address[4]	pointer to encryption routine supplied by SMTP-Send. This routine can be used by your exit to encrypt a password value before calling LDAP.
address[4]	pointer to decryption routine. Supplied by SMTP-Send on calls to LDAP or by user exits when calling LDAP from within an exit. Zero indicates that the password below is in plain text.
	When calling encryption or decryption programs, pass the address of the following structure in General Register 1
byte[1]	length of plain text password before encryption or after decryption.
byte[1]	length of encrypted password after encryption or before decryption.
character[128]	plain text password value. Place value to be encrypted here before calling encryption program. Value is replaced by decryption program
byte	reserved
character[128]	encrypted text password value. Place value to be decrypted here before calling decryption program. Value is replaced by encryption program. If the name of the expansion exit program name on the DISTLISTPGM statement begins with LDAP, the LDAP_Password is copied into this field; otherwise, the Exit_Password value is copied here.
byte	reserved

You need to check if the distribution list name passed to your exit is that of an existing distribution list. If it is then the user corresponding to the sequence number, i.e. 1, 2, etc, should be returned. The exit is called repeatedly until a non-zero return code is returned.

Under MVS/Batch or TSO, the data structure is passed by address. SMTP-Send will LINK to the exit program with Register 1 pointing to the address of an address that points to a 2-Byte length followed by a 9-Character string representation of the data structure address in decimal.
For example, if the data structure is located at X'201616A8' (538318504 in decimal). SMTP-Send will build the following string: `X'0009F5F3F8F3F1F8F5F0F4'` and store the address of that string in its working storage and then load the value of that working storage location into Register 1.

Under CICS, the data structure is passed as a CICS Commarea.

Return Codes

The called program should return:

Return Code	Meaning
0	a name is returned for the given distribution list and the user exit should be called again
4	name passed is not a distribution list or no more members exist in the list

Performing your own address validation

SMTP-Send allows you to perform address validation in an exit program. You can use this exit to prevent unauthorized users from pretending they are someone else.

You specify the exit program name via the VALIDATEPGM Configuration statement. When a program name has been specified, every prospective SMTP address is passed to your exit program along with additional descriptive information before it is used. You can accept the address passed by SMTP-Send, replace the SMTP name with a new value or reject the SMTP name.

The validation exit program name on the VALIDATEPGM statement is restricted to 7 characters. When SMTP-Send runs in MVS Batch or TSO the character 'B' is appended to the program name. Under CICS, the character 'C' is appended. For example, under CICS a VALIDATEPGM VALIDP statement causes the program VALIDPC to be used to provide address validation logic.

You can pass parameters to your validation exit by adding them to the VALIDATEPGM Configuration statement following the program name. Enclose the entire value in double quotes as follows:

```
VALIDATEPGM "VALIDP,IPAddress=127.0.0.0"
```

The address of the string `IPAddress=127.0.0.0` will be passed to the expansion exit in the passed data structure along with the length of the string. If the length is zero, no parameters were supplied.

The parameters passed to the VALIDATEPGM program are stored in plain text in the Configuration file. If you need to pass a password value to your exit program, you can code an asterisk character, *, in place of the password in your parameter string. SMTP-Send will use

the value from the `Exit_Password` configuration statement in place of the * when your exit is invoked.

Sample S/390 Assembler validation exit programs for Batch and CICS are in INSTLIB members VALIDPB and VALIDPC.

If required you can call the TBS LDAP routines from your validation exit to extract any additional data you might need to validate the address. Refer to Using LDAP for more information.

A note on APF authorization

If you plan to use security services that require APF authorization under MVS Batch or TSO, then you must (a) relink the SMTP- Send load module with the AC(1) attribute and (b) authorize the SMTP-Send load library. See APF authorization in the Installing section.

Parameter List

The exit program is passed the following data structure (mapped by INSTLIB member VALIDATA) either via standard OS linkage (pointed to by R1) or as a CICS commarea:

Field	Description
int[4]	Contains the return code from the user exit. This is in addition to the value returned via R15, since CICS applications are unable to return values in registers.
address[4]	Contains a pointer to a 12-byte workarea that is returned after each call to the user exit. This can be used to retain information between calls.
address[4]	Contains a pointer to any parameters that may have been supplied on the VALIDATEPGM control statement. If no parameters were provided, the pointer is zero.
byte[1]	Contains the length of any parameters that may have been supplied on the VALIDATEPGM control statement. If no parameters were provided, the length is zero.
character[8]	The current jobname.
character[8]	RACF or CICS userid of the current user.
character[80]	Default domain specified during SMTP-Send customization. The user exit can use this value to create addresses that use the default domain when no domain can be determined from the input data.
byte[1]	Contains the length of a message returned by the user exit. If no message is returned this value should be set to zero. The maximum length that can be displayed by SMTP-Send is 125 characters.
character[125]	Contains a message returned from the user exit to be displayed by SMTP-Send.
character[1]	Type of SMTP address: 'T' - TO address 'S' - sender's address 'C' - CC address 'B' - BCC address 'R' – REPLY-TO address
character[4]	Reserved. OP/SS pointer to DGN (address) value.
character[4]	Reserved. OP/SS length of DGN.
character[4]	Reserved. OP/SS pointer to DEN (user) value.
character[4]	Reserved. OP/SS length of DEN.
character[128]	Original recipient data as entered on control card; the user exit can use this data to validate the address.
byte[1]	On entry to the user exit, contains the length of the SMTP name passed to the user exit. On return, it contains the length of the SMTP name returned by the user exit.
character[128]	Contains the proposed SMTP name. The user exit can use this field to return an alternate SMTP address.
byte[1]	Filler byte used to terminate the proposed SMTP name for C and C++ callers.
byte[1]	Indicates that the userid of the sender has been obtained from RACF/SAF during address lookup due to a configuration option.
int[4]	Contains the length of the Original recipient data.

Field	Description
address[4]	Contains a pointer to the LDAP exit program parameter string passed on the LDAP_LOOKUP_PGM statement. Contains zero if no parm or LDAP exit was specified.
byte[1]	Contains the length of the LDAP exit parameter or zero if no LDAP exit parm was specified.
byte[1]	Flags to indicate the results of address lookups for this address: x'01' - no lookup performed x'02' - LDAP lookup failed x'04' - address mapping exit lookup failed x'08' - default domain has been appended to address
byte[1]	Controls subsequent invocation of data manipulation exit. Any non-zero value indicates the exit will be called. X'FF' indicates that the ALWAYSCALLDATAEXIT statement in the SMTP-Send Configuration dataset requests that the exit always be called. You can clear this field if you do not want the exit to be called.
address[4]	Contains a pointer to a 12-byte workarea that is passed between the validation and data manipulation exits. This can be used to retain information between calls.
character[6]	Contains the 6 character prefix of the LDAP lookup program from the LDAP_LOOKUP_PGM OPSCONFG statement padded on the right with nulls.
byte[1]	reserved

Under MVS/Batch or TSO, the data structure is passed by address. SMTP-Send will LINK to the exit program with Register 1 pointing to the address of an address that points to a 2-Byte length followed by a 9-Character string representation of the data structure address in decimal.
For example, if the data structure is located at X'201616A8' (538318504 in decimal). SMTP-Send will build the following string: `X'0009F5F3F8F3F1F8F5F0F4'` and store the address of that string in its working storage and then load the value of that working storage location into Register 1.

Under CICS, the data structure is passed as a CICS Commarea.

Return Codes

The called program should return:

Return Code	Meaning
0	if the proposed name is acceptable
4	the original SMTP value was replaced by a new value in the SMTP name field
8	if the proposed name is declined by the exit and must not be used
16	if the new name will not fit in the 128 character destination field

Reformatting data before it is sent via SMTP

SMTP-Send allows you to manipulate the data assembled by SMTP-Send before it is sent. You can use this exit to perform any additional data substitution you might require.

You specify the exit program name via the DATAEXITPGM Configuration statement. When a program name has been specified, the exit is called immediately before the data is sent. You cannot change the mail headers from this exit; the From, To, Cc, Bcc and Subject information is read-only.

The exit program name on the DATAEXITPGM statement is restricted to 7 characters. When SMTP-Send runs in MVS Batch or TSO the character 'B' is appended to the program name. Under CICS, the character 'C' is appended. For example, under CICS,the statement, `DATAEXITPGM DATAXTP` causes the program `DATAXTPC` to be used to provide data manipulation logic.

You can pass parameters to your exit by adding them to the DATAEXITPGM Configuration statement following the program name. Enclose the entire value in double quotes as follows:

```
DATAEXITPGM "DATAXT,IPAddress=127.0.0.0"
```

The address of the string `IPAddress=127.0.0.0` will be passed to the expansion exit in the passed data structure along with the length of the string. If the length is zero, it means that no parameters were supplied. The data is contained in a dataset for which the DDname is passed in the exit data structure. You must open the dataset and read the data in your exit. You can replace the data by allocating a new dataset and returning the new DDname to be used.

Sample MVS Assembler data manipulation exit programs for Batch and CICS are in INSTLIB members DATAXTB and DATAXTC. Refer to them for details on accessing the header information.

A note on APF authorization

If you plan to use services that require APF authorization under MVS Batch or TSO, then you must (a) relink the SMTP-Send load module with the AC(1) attribute and (b) authorize the SMTP-Send load library. See APF authorization in the Installing section.

Parameter List

The exit program is passed the following data structure (mapped by INSTLIB member DATACOMM) either via standard OS linkage (pointed to by R1) or as a CICS commarea:

Field	Description
int[4]	Contains the return code from the user exit. This is in addition to the value returned via R15, since CICS applications are unable to return values in registers.
address[4]	Contains a pointer to a 12-byte workarea that is returned after each call to the user exit. This can be used to retain information between calls.
address[4]	Contains a pointer to any parameters that may have been supplied on the DATAEXITPGM control statement. If no parameters were provided, the pointer is zero.
byte[1]	Contains the length of any parameters that may have been supplied on the DATAEXITPGM control statement. If no parameters were provided, the length is zero.
character[8]	The current jobname.
character[8]	RACF or CICS userid of the current user.
address[4]	Contains a pointer to the linked list of From information after all address lookups and validation have completed.
address[4]	Contains a pointer to the linked list of To information after all address lookups and validation have completed. Zero if no To information has been provided.
address[4]	Contains a pointer to the linked list of Cc information after all address lookups and validation have completed. Zero if no Cc information has been provided.
address[4]	Contains a pointer to the linked list of Bcc information after all address lookups and validation have completed. Zero if no Bcc information has been provided.
address[4]	Contains a pointer to the linked list of Subject information after all address lookups and validation have completed. Zero if no Subject information has been provided.
byte[1]	On entry to the user exit, contains the length of the SMTP name passed to the user exit. On return, it contains the length of the SMTP name returned by the user exit.
character[8]	Contains the DDname for the dataset or 8-byte CICS Temporary Storage Queue name containing the data to be sent.
character[1]	Indicates whether the data is in a MVS file 'F' or CICS TS Queue 'T'.
byte[1]	Contains the length of a message returned by the user exit. If no message is returned this value should be set to zero. The maximum length that can be displayed by SMTP-Send is 125 characters.
character[125]	Contains a message returned from the user exit to be displayed by SMTP-Send.
address[4]	Contains a pointer to the LDAP exit program parameter string passed on the LDAP_LOOKUP_PGM statement. Contains zero if no parm or LDAP exit was specified.
byte[1]	Contains the length of the LDAP exit parameter or zero if no LDAP exit parm was specified.
byte[1]	Indicates how data exit was invoked. X'FF' indicates data invoked due to ALWAYSCALLDATAEXIT in the SMTP-Send Configuration dataset. Any other value is set by the address validation exit.
address[4]	Contains a pointer to a 12-byte workarea that is passed between the validation and data manipulation exits. This can be used to retain information between calls.
character[6]	Contains the 6 character prefix of the LDAP lookup program from the LDAP_LOOKUP_PGM OPSCONFG statement padded on the right with nulls.
byte[1]	reserved

Under MVS/Batch or TSO, the data structure is passed by address. SMTP-Send will LINK to the exit program with Register 1 pointing to the address of an address that points to a 2-Byte length followed by a 9-Character string representation of the data structure address in decimal.

For example, if the data structure is located at X'201616A8' (538318504 in decimal). SMTP-Send will build the following string: `X'0009F5F3F8F3F1F8F5F0F4'` and store the address of that string in its working storage and then load the value of that working storage location into Register 1.

Under CICS, the data structure is passed as a CICS Commarea.

Return Codes

The called program should return:

Return Code	Meaning
0	the data should be sent as is
8	the distribution should be aborted and the data should not be sent

Performing your own address mapping

OP/SS allows you to perform your own address mapping in an exit program. You can replace the address mapping logic supplied by OP/SS or supplement it by adding your own mapping logic, for example to interrogate an external directory.

You specify the address mapping exit program name via the MAPPINGPGM Configuration statement. When a program name is specified, every SMTP address is passed to your exit program along with additional descriptive information before it is used. You can lookup the SMTP name based upon the input value or accept the address suggested by OP/SS.

The mapping exit program name on the MAPPINGPGM statement is restricted to 7 characters. When OP/SS runs in MVS Batch or TSO the character 'B' is appended to the program name before it is used. Under CICS, the character 'C' is appended. For example, under CICS the statement MAPPINGPGM LOOKUP causes program LOOKUPC to be used to provide custom address mapping logic.

You can pass parameters to your address mapping exit by adding them to the MAPPINGPGM Configuration statement following the program name. Enclose the entire value in double quotes as follows:

```
MAPPINGPGM "LOOKUP,IPAddress=127.0.0.0"
```

The address of the string `IPAddress=127.0.0.0` is passed to the address mapping exit in the passed data structure along with the length of the string. A length of zero indicates that no parameters were supplied.

You can pass an encrypted password value to your exit by specifying an `Exit_Password` OPSCONFG control statement. The address of a decryption routine will be passed along with the password in the MAPPRDAT data structure.

Sample mapping validation programs written in S/390 Assembler can be found in INSTLIB members LOOKUPB and LOOKUPC. You can also call the TBS LDAP routines from your validation exit to extract any additional data you might need to lookup the address. Refer to Using LDAP for more information.

Parameter List

The exit program is passed the following data structure either via standard OS linkage (pointed to by R1) or as a CICS commarea:

Field	Description
int[4]	Contains the return code from the user exit. This is in addition to the value returned via R15, since CICS applications are unable to return values in registers.
address[4]	Contains a pointer to a 12-byte workarea that is returned after each call to the user exit. This can be used to retain information between calls.
address[4]	Contains a pointer to any parameters that may have been supplied on the MAPPINGPGM control statement. If no parameters were provided, the pointer is zero.
byte[1]	Contains the length of any parameters that may have been supplied on the MAPPINGPGM control statement. If no parameters were provided, the length is zero.
character[8]	The current jobname.
character[8]	RACF or CICS userid of the current user.
character[80]	Default domain specified during OP/SS customization. The user exit can use this value to create addresses that use the default domain when no domain can be determined from the input data.
byte[1]	Contains the length of a message returned by the user exit. If no message is returned this value should be set to zero. The maximum length that can be displayed by OP/SS is 125 characters.
character[125]	Contains a message returned from the user exit to be displayed by OP/SS.
character[1]	Intent for `DGN.DEN` value in parms 2 and 3 , `'T'` - TO address, `'S'` - sender's address, `'C'` - CC address, `'B'` - BCC address, `'R'` – REPLY-TO address
character[4]	pointer to DGN (address) value.
character[4]	length of DGN.

Field	Description
character[4]	pointer to DEN (user) value.
character[4]	length of DEN.
character[128]	Original recipient data as entered on control card. The user exit can use this data to assist in the address mapping.
byte[1]	On entry to the user exit, contains the length of the SMTP name passed to the user exit. On return, contains the length of the SMTP name returned by the user exit.
character[128]	Contains the proposed SMTP name. The user exit can use this field to return an alternate SMTP address.
byte[1]	Filler byte used to terminate the proposed SMTP name for C and C++ callers.
byte[1]	x'00'or x'40' indicates V1 format, C'1' indicates V2 and presence of extension
int[4]	Contains the length of the original recipient data.
address[4]	Contains a pointer to the LDAP exit program parameter string passed on the `LDAP_LOOKUP_PGM` statement. Contains zero if no parm or LDAP exit was specified
Byte[1]	Length of LDAP parameter string, or zero if none was specified.
Start of V2 Extension	
address[4]	pointer to encryption routine supplied by SMTP-Send. This routine can be used by your exit to encrypt a password value before calling LDAP.
address[4]	pointer to decryption routine. Supplied by SMTP-Send on calls to LDAP or by user exits when calling LDAP from within an exit. Zero indicates that the password below is in plain text.
When calling encryption or decryption programs, pass the address of the following structure in General Register 1	
byte[1]	length of plain text password before encryption or after decryption.
byte[1]	length of encrypted password after encryption or before decryption.
character[128]	plain text password value. Place value to be encrypted here before calling encryption program. Value is replaced by decryption program
byte	reserved
character[128]	encrypted text password value. Place value to be decrypted here before calling decryption program. Value is replaced by encryption program. The `Exit_Password` value is copied here.
byte	reserved
byte	reserved
character[6]	Contains the 6 character prefix of the LDAP lookup program from the `LDAP_LOOKUP_PGM` OPSCONFG statement padded on the right with nulls.
byte	reserved

Under MVS/Batch or TSO, the data structure is passed by address. SMTP-Send will LINK to the exit program with Register 1 pointing to the address of an address that points to a 2-Byte length followed by a 9-Character string representation of the data structure address in decimal.

For example, if the data structure is located at X'201616A8' (538318504 in decimal). SMTP-Send will build the following string: `X'0009F5F3F8F3F1F8F5F0F4'` and store the address of that string in its working storage and then load the value of that working storage location into Register 1.

Under CICS, the data structure is passed as a CICS Commarea.

Return Codes

The called program should return:

Code	Meaning
0	if the proposed name is acceptable
4	the original SMTP value was replaced by a new value in the SMTP name field
8	if an SMTP name could not be determined from the DGN and DEN values
16	if the new name will not fit in the 128 character destination field

Changing SMTP-Send messages

Translating messages into a different language

SMTP-Send messages are stored in a message table in INSTLIB. The message table is named OPSSTRT0. To translate the SMTP-Send messages into your own language run INSTLIB job OPSCUSTR. This job creates the appropriate load module.

How SMTP-Send messages are structured

SMTP-Send messages consist of:

Fixed text parts Fixed text parts stay the same each time SMTP-Send generates the message. They consist of letters and numbers that you can change using the message definition macro #TBSSTRD. This macro is described below.

Variable text parts Variable text parts may differ each time SMTP-Send generates the message. These parts depend on the information that SMTP-Send is processing when the message is generated.

Insertion points Insertion points indicate where variable text parts will be inserted in the message text.

The message definition macro #TBSSTRD

Each text message that SMTP-Send generates is defined using the #TBSSTRD macro. To change a text message, edit the #TBSSTRD macro statement for that message. Follow the syntax shown below to code the messages you want to change.

```
[label] #TBSSTRD num,'text',MAXLEN=(ml,l1,...,l4)
```

`label` (optional) A unique label for the #TBSSTRD macro. Specify one to eight alphanumeric characters, beginning in column one. The first character must be a letter.

`num` The message number. Do not change this value.

`text` The message text, which consists of fixed text parts and up to four insertion points. You can change this text.

`(ml,l1,...,l4)` The maximum length for the message.. If the changed message will exceed this length, shorten the message. Do not change the value of MAXLEN.

The value of `ml` is the total maximum length. If the message contains insertion points (described below), these insertion points are represented by `l1` to `l4`. A 79-character message with two insertion points, one of 5 characters and one of 3 characters, is specified as:

```
MAXLEN=(79,5,3)
```

If there are no insertion points, you do not need the brackets or the values l1 to l4. A 79-character message with no insertion points is specified as:

```
MAXLEN=79
```

Coding message text

Quotation marks

Message text consists of a character message enclosed in single quotation marks ('). Because the single quotation mark is used to enclose text, you must use two single quotation marks ('') where you want a single quotation mark to appear within the text.

Insertion points

Message text may also include insertion points specified as &?. These insertion points tell SMTP-Send where to insert variable information. The value of # ranges from 1 to 4. You can change the text around an insertion point, or the location of an insertion point, but do not change the value of #.

The number of insertion points must be less than or equal to the number of insertion points in the original text. If you use fewer insertion points than were in the original text, the message will lose the information conveyed by the omitted insertion point(s).

Long messages

Do not put message text in columns 72 to 80, because this text is ignored. If a message will not fit in a single line, put any character in column 72, then begin the next line in column 16.

Special characters in languages other than English

SMTP-Send assumes that all text in message tables is in code page 500. If you are translating messages to a language other than English, make sure that you use code page 500 for special characters such as #, \ and Ö

Customizing ISPF panel support

SMTP-Send includes ISPF dialogs that enable TSO/ISPF users to invoke SMTP-Send from ISPF. The SMTP-Send dialogs consist of the 3 datasets, ISPCLIB, ISPMLIB & ISPPLIB.

SMTP-Send can also be invoked directly from a REXX EXEC, allowing you to email enable your TSO/ISPF application. A sample REXX EXEC, OPSREXX, included in INSTLIB, illustrates how this can be done. This EXEC does not demonstrate all of the functionality of SMTP-Send but can be used as a starting point to create your own EXEC that will meet your needs.

Installing the SMTP-Send dialogs using standard ISPF libraries

The ISPF installation includes ISPF panels, messages and CLIST libraries. These files must be accessible to users before they can run the SMTP-Send dialogs.

The simplest way to use the dialog, is to run *'TSO EX prefix.ISPCLIB(OPSMENU)'* from any ISPF panel. No installation is required to run this way. You can even wrap this command into a CLIST within the SYSPROC concatenation, to enable a single word invocation.

To install the SMTP-Send ISPF dialogs either concatenate the SMTP-Send ISPF libraries with the standard ISPF libraries OR copy the SMTP-Send ISPF library members into the corresponding standard ISPF libraries. For example, copy the members in *prefix.ISPPLIB* into the standard ISPPLIB library.

Once the members have been copied into the appropriate libraries, you can invoke SMTP-Send using the command *TSO OPSMENU*. You can also invoke the SMTP-Send ISPF panels by updating an existing ISPF panel as follows:

Step	Procedure
1	Modify the panel to add a menu item for SMTP-Send. For example: `15 - Send mail to the Internet`

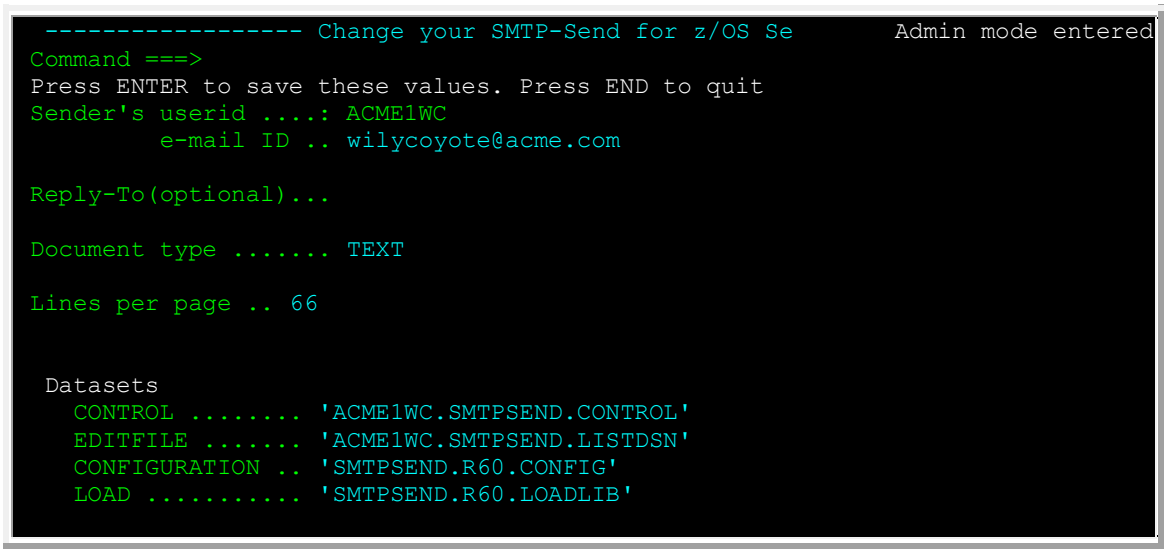
2	Add the SMTP-Send panel to the selection list. For example:
	`&ZSEL = TRANS ( TRUNC (&ZCMD,'.')`
	`.`
	`.`
	`15,'CMD(OPSMENU) NEWAPPL(OPS)'`
	`.`

Customizing the SMTP-Send dialogs

The CLIST OPSDEFLT sets the various default values used by SMTP-Send. Although dialogs are provided for users to alter these values themselves, it is recommended that you update this CLIST to set the correct values at installation time. Values you need to specify include the data set names of your SMTP-Send Configuration file and LOADLIB and the domain name for your enterprise.

Changing the Datasets

In the settings panel, a list of datasets used to process the send is shown. For normal usage these do not need to be changed by users of the panels. An administrator can change these dataset names by entering the ADMIN command. The following image shows the settings panel after the ADMIN command, with the dataset name fields now unprotected.
The CONTROL and EDITFILE datasets are dynamically allocated for each user, the first time a user runs the Dialogs.



Using the TBS SMTP Spool Server

SMTP-Send comes with a Spool Server (OPSSSRVR) that relays SMTP data from the JES2 spool to an SMTP relay server. This spool data can be written either by SMTP-Send, or by other applications such as SMTPNOTE or XMITIP, or a customer application. SMTP-Send can be explicitly configured to direct its output to the JES spool, but will also do so when it is unable to complete a relay to a receiving SMTP server via TCP/IP.

The SMTP Spool Server is a functional replacement for the outbound capabilities of IBM's SMTP server started task. It selects non-held JES2 datasets matching the external writer name, and class parameters that were specified in its configuration file. These parameters can also be changed on the fly, using MODIFY commands.

If communications fail with the receiving SMTP relay server(s), OPSSSRVR waits on a timer for the interval specified in the configuration file. Failure results in the distributions being left on the JES SPOOL until such time as an SMTP server becomes available or the retry count is exceeded. In the event the retry count is exceeded, an operator message is written to the console requesting a reply to either repeat the retry logic or to terminate OPSSSRVR. If no SMTP servers will be available for some time, the operator should terminate OPSSSRVR. Once the receiving SMTP server is available, the operator should restart OPSSSRVR.
If the receiving SMTP server rejects the incoming message, the SMTP data stream is sent as an attachment in a non-delivery message to the configured recipient.

When there is no work to process, OPSSSRVR waits until one of the following events occurs:

1. A SYSOUT dataset becomes available that matches the input selection criteria.
2. A system command is received from the console operator.
3. Retry mode is in effect due to communications failure with the SMTP server(s) and the delay interval timer expires, meaning it's time to try again.

OPSSSRVR supports single or multiple distributions in a single SYSOUT dataset

OPSSSRVR supports JES Network format data such as that produced by IBM's SMTPNOTE REXX commands. Use the configuration parameters to select SYSOUT datasets produced by SMTPNOTE.

Send details may be logged to the MVS SMF datasets or to an optional user supplied sequential dataset. Details logged include the original job name and Userid, the document size, date, sender, reply-to, subject, all recipients, and any attachment names if applicable.

Note: you can use a different SMF ID to combine your OPSSSRVR log records with those from SMTP-Send.

A user customizable message table, OPSMSGTx is used, similar to that used by SMTP-Send.

LOG messages are written to a common file named OPSSLOG and other messages such as debugging messages are written to OUTPUT. The Modify operator command can be used to change a number of application control parameters. The START operator command can be used to initiate OPSSSRVR provided the proper JCL has been added to PROCLIB, and the STOP operator command can be used to terminate OPSSSRVR.

OPSSSRVR normally connects to the SMTP server specified by the SMTP_RELAY_ADDRESS parameter. Should this server fail to respond within SERVER_TIMEOUT seconds, the alternate server will be contacted, as specified by the SMTP-RELAY_ADDRESS_ALTERNATE parameter. Once this occurs, the primary SMTP server will again be contacted, after RETRY_PRIMARY minutes. This enables a faster primary server to be used whenever possible.
Each of these parameters can also be changed for the duration of the task, using the MODIFY commands described in *Console Operator Commands*.

Installation Considerations

The Server mainline program, OPSSSRVR, must be linked SETCODE AC(1) and reside in an APF authorized library in order to read the JES SPOOL using the Subsystem Interface. Note that all other libraries used in STEPLIB must also be APF authorized.

Example: **`SETPROG APF,ADD,DSN=OPS.LOADLIB,VOLUME=`*`vvvvvv`**

If not APF-authorized, you will receive the message, **`OPS104E JES Subsystem Interface error. RC=0072, RSN=0000,`** and SMTP- Send will terminate with a return code of 8,

Configuration Parameters

The following configuration parameters are used to control various aspects of OPSSSRVR as indicated. Where a control statement is common between SMTP-Send and the Spool Server the complete definition is not repeated in this section.

ATSIGN

Same as SMTP-Send ATSIGN Configuration statement

BATCH_AUTHCODE

Same as SMTP-Send BATCH_AUTHCODE Configuration statement

DEBUG

Purpose: Messages with debugging information are written to the OPSSLOG and OUTPUT datasets if this value is coded. YES provides some detail, VERBOSE significantly more, while TRACE provides the most amount of extra detail regarding current processing.
Syntax: `DEBUG {YES|NO|VERBOSE|TRACE}`
Default: `DEBUG NO`
Example: `DEBUG YES`
Note: to enable DEBUG only for a particular job, add `PARM='value'` on the `EXEC PGM=OPSGS3B JCL` statement

DEFAULT_DOMAIN

Same as SMTP-Send DEFAULT_DOMAIN Configuration statement

ERROR_MSG_FROM_NAME

Same as SMTP-Send ERROR_MSG_FROM_NAME Configuration statement

ERROR_MSG_TO_NAME

Same as SMTP-Send ERROR_MSG_TO_NAME Configuration statement

FAIL_ON_ERROR

Same as SMTP-Send FAIL_ON_ERROR Configuration statement

FIXSPOOLLEADINGDOTS

Purpose: Instructs OPSSSRVR to apply the SMTP leading dot transparency feature to data. When a receiving SMTP server encounters a line with a length greater than 1 that begins with a `'.'` character it will remove the character, decreasing the length of the incoming line by 1. When `FixSpoolLeadingDots` is set to `Yes`, OPSSSRVR will insert an extra `'.'` character at the beginning of each line longer than 1 character that begins with a `'.'` More information on the SMTP leading dot transparency feature can be found in the SMTP specification, RFC 2821 Section 4.5.2
NOTE: This specification should only be used when SMTP streams are created outside of the SMTP-Send environment and may contain `'.'` characters at the beginning of lines, such as used by some users for document formatting. A specification of `All` will cause an extra `'.'` to be inserted even for lines containing only a single `'.'` character. This should only be used where the SMTP input streams contain single distributions as single `'.'` intended to terminate a distribution will instead be considered to be data
Syntax: `FixSpoolLeadingDots {YES|NO|ALL}`
Default: `FixSpoolLeadingDots NO`

LOGGING

Same as SMTP-Send LOGGING Configuration statement

RELAY_RETRY_COUNT

Purpose: Number of times OPSSSRVR will attempt to resend a distribution before requesting operator intervention
Syntax: `RELAY_RETRY_COUNT [nnn|3]`
Default: `RELAY_RETRY_COUNT 3`
Example: `RELAY_RETRY_COUNT 5`

RELAY_RETRY_DELAY

Purpose: Time in minutes to wait before attempting to resend a distribution or look on spool for another message to be relayed
Syntax: `RELAY_RETRY_DELAY [nnn|10]`
Default: `RELAY_RETRY_DELAY 10`
Example: `RELAY_RETRY_DELAY 5`

RELAY_LOG_DETAILS

Purpose: tells OPSSSRVR to display the sender and recipient names of relayed e-mails in the OPSSLOG data set. The name of the data set from the JES Spool is also displayed.
Syntax: `Relay_Log_Details [YES|NO]`
Default: `Relay_Log_Details NO`
Example: `Relay_Log_Details YES`

RELAY_SMF_TYPE

Purpose: specifies the SMF record type (or number) that contains log data. This value is used when logging to SMF from the TBS SMTP Spool Server. This is a value between 128 and 255. Check with your MVS Systems Programmer to determine which SMF type should be used.
Note: SMF records produced by Batch and CICS jobs, use the record type specified by the SMF_TYPE parameter. The default values of RELAY_SMF_TYPE, and SMF_TYPE are different, allowing for separate SMF record types for Spool Server activity, although you can make these the same.
In batch, this is a value between 128 and 255.
Syntax: `Relay_SMF_TYPE [nnn]`
Default: `Relay_SMF_TYPE 129`
Example: `Relay_SMF_TYPE 255`

REMOVETRAILINGBLANKS

Same as SMTP-Send REMOVETRAILINGBLANKS Configuration statement

RETRY_PRIMARY

Purpose: This parameter is used after the SERVER_TIMEOUT interval has been exceeded. OPSSSRVR will then use the SMTP_RELAY_ADDRESS_ALTERNATE. After the RETRY_PRIMARY interval (in minutes), it will once again attempt to use the primary SMTP server.
Syntax: `Server_Timeout [nnn | 10]`
Default: `Server_Timeout 10`
Example: `Server_Timeout 5`

SERVER_TIMEOUT

Purpose: specifies the number of seconds to wait for a connection to the SMTP_RELAY_ADDRESS. If this timeout is exceeded, OPSSSRVR will attempt to connect to the SMTP_RELAY_ADDRESS_ALTERNATE.
Syntax: `Server_Timeout [nnn | 30]`
Default: `Server_Timeout 30`
Example: `Server_Timeout 60`

SMTP_RELAY_ADDRESS

Same as SMTP-Send SMTP_RELAY_ADDRESS Configuration statement

SMTP_RELAY_ADDRESS_ALTERNATE

Same as SMTP-Send SMTP_RELAY_ADDRESS_ALTERNATE Configuration statement

SMTP_SERVER_CLASS

Same as SMTP-Send SMTP_SERVER_CLASS Configuration statement

SMTP_SERVER_NAME

Same as SMTP-Send SMTP_SERVER_NAME Configuration statement

SMTP_SERVER_NODE

Same as SMTP-Send SMTP_SERVER_NODE Configuration statement

USECUSTOMCODEPAGE

Same as SMTP-Send USECUSTOMCODEPAGE Configuration statement

WTO_onTCPFailure

Same as SMTP-Send WTO_onTCPFailure Configuration statement, but the default value is different for the Spool Server.
Default: `WTO_onTCPFailure Yes`

Running under MVS

The following JCL runs the SMTP Spool Server, and can be found in the INSTLIB member, OPSSSRVR.

```
//OPSSSRVR JOB
//STEP1  EXEC PGM=OPSSSRVR
//STEPLIB  DD DISP=SHR,DSN=OPS.LOADLIB
//OPSCONFG DD DSN=OPS.CONFIG,DISP=SHR
//OPSSLOG   DD SYSOUT=*
//OUTPUT    DD SYSOUT=*
```

Add the following DD statement if you want to see the incoming SMTP data. You might find this option useful during testing.

```
//DATAIN  DD SYSOUT=*
```

EXEC runtime parameter

Override the DEBUG control statement in the CONFIG file using one of the **NODEBUG, DEBUG, VERBOSE** or **TRACE** values for the OPSSSRVR EXEC parameter. This parameter can also be controlled during execution using the Modify operator command as described below.

```
//STEP1   EXEC PGM=OPSSSRVR,PARM=DEBUG
```

Logging

Processing messages are written to a sequential file named OPSSLOG.

Each message is preceded by the Julian date YYDDD followed by the time HH:MM:SS

The message number ends with a type (column 23) which indicates one of the following:

I Informational message

W Warning message

E Error message

D debug message

Sample OPSSLOG:

```
02227 14:45:04 OPSSSRVR (c) COPYRIGHT 2002,2005 BY TBS SOFTWARE INC., ALL RIGHTS RESERVED
02227 14:45:04 OPS001I OPSSSRVR Initialization - Version R5.1 PTF 1A
02227 14:45:04 OPS006I External writer name is SMTP
02227 14:45:04 OPS016I External writer class is X
02227 14:45:04 OPS007I Retry delay interval is 005 minutes
02227 14:45:04 OPS008I Retry count is 02
02227 14:45:04 OPS009I SMTP relay address is 127.0.0.1:8026
02227 14:45:04 OPS010I SMTP alternate relay address is 127.0.0.1:25
02227 14:45:04 OPS024I NDR's sent to: POSTMASTER@TBSSOFT.COM
02227 14:45:04 OPS025I NDR's sent from: SMTPSEND@TBSSOFT.COM
02227 14:45:04 OPS027I Default Domain: TBSSOFT.COM
02227 14:45:04 OPS019I AtSign Character is: @
02227 14:45:04 OPS012I SMF logging to record type 130 is active
02227 14:45:04 OPS028I GetHostByName is: 0
02227 14:45:04 OPS029I FailOnError is: 0
02227 14:45:04 OPS036I RemoveTrailingBlanks is 1
02227 14:45:04 OPS032I Relay_Log_Details is active
02227 14:45:04 OPS001I OPSSSRVR  14.42  11/30/05 Initialized
02227 15:45:43 OPS004I Processing dataset TSSJES1.JOB04887.D0000103.?
02227 15:45:44 OPS020I Received Note from <demo@tbssoft.com>
02227 15:45:44 OPS021I Delivering Note to <support@tbssoft.com>
02227 15:45:44 OPS021I Delivering Note to <demo@tbssoft.com>
02227 15:45:45 OPS005I Dataset sent successfully
02227 15:48:38 OPS003I Received STOP operator command
02227 15:48:38 OPS002I OPSSSRVR is terminating
```

Console Operator Commands

Initiate OPSSSRVR using the START Operator command:
example: **S OPSSSRVR**

Note: this assumes a valid member named OPSSSRVR exists in SYS1.PROCLIB or other such library used for starting jobs.

Shutdown OPSSSRVR using the STOP Operator command:
example: **P OPSSSRVR**

To change execution parameters using the MODIFY command, refer to the following table.

Terminate processing	**F OPSSSRVR,QUIT**
Turn ON basic debug messages	**F OPSSSRVR,DEBUG**
Turn OFF debug messages	**F OPSSSRVR,NODEBUG**
Turn ON verbose debug messages	**F OPSSSRVR,VERBOSE**
Turn ON debug trace-level messages	**F OPSSSRVR,TRACE**
Display configuration setting	**F OPSSSRVR,SHOWCFG**
Change the Spool Input class	**F OPSSSRVR,CLASS=c**
Toggle the SMTP leading dot transparency feature setting off or on (only on lines with >1 character)	**F OPSSSRVR,FIXDOTS**
Toggle the SMTP leading dot transparency feature setting off or on (for all data lines)	**F OPSSSRVR,FIXALLDOTS**
Change the amount of time OPSSSRVR waits between retries or requests to JES for additional messages	**F OPSSSRVR,SLEEPINT=nnn**
Change the primary SMTP server to the specified IP name or address	**F OPSSSRVR,SMTP=primary server**
Change the alternate SMTP server to the specified IP name or address	**F OPSSSRVR,ALTSMTP=alternate server**
Change the timeout for connection to the primary SMTP server	**F OPSSSRVR,SERVTIMEOUT=nnn (seconds)**
Change the interval to retry using the primary SMTP server (after a SERVTIMEOUT)	**F OPSSSRVR,RETRYPRIMMIN=nnn (minutes)**
Change the Spool External Writer name	**F OPSSSRVR,WTR=nnnnnnnn**

Messages and codes

Return Codes

Code	Meaning
0000	OPSSSRVR terminated without errors.
0004	Check the OPSSLOG for warning messages.
0008	Check the OPSSLOG for error messages.
0016	A severe error occurred and OPSSSRVR was terminated. Check OPSSLOG for details.
0032	The operator terminated OPSSSRVR when asked to reply, since there are no SMTP servers currently available.
0068	Error encountered loading TSSSEND module
0128	Internal logic error encountered

User Abends

U0001 Application is unable to write an operator message to the system console.

In the event OPSSSRVR is unable to communicate with the SMTP server(s), and the maximum retry count has been exceeded, an operator message is written to the console requesting what action to take.

Using LDAP

SMTP-Send allows you to expand distribution lists into e-mail addresses using LDAP (**L**ightweight **D**irectory **A**ccess **P**rotocol) compliant directories. LDAP is an Internet standard for accessing directory information using TCP/IP. Many popular e-mail systems such as Lotus Notes and Microsoft Exchange expose their directory information via LDAP, as does Active Directory.

SMTP-Send provides two types of LDAP lookup. You can choose to use either or both.

1. Expansion of distribution lists into one or more e-mail addresses
2. E-mail address lookup, using other attributes of a user.

Expansion of distribution lists

Distribution lists contain a group of e-mail addresses and possibly other distribution lists. There are 2 forms of distribution lists, the client side list, and the server side list.

With the client side list, SMTP-Send uses LDAP to obtain each of the e-mail addresses within the distribution list and then adding it to the appropriate addressing header. Duplicate e-mail addresses are removed from the expanded list.

With a server side list, the distribution list has an e-mail address of its own. SMTP-Send obtains this address and adds it to the appropriate addressing header. When the server receives this email, it will obtain each of the members within the server side distribution list, and distribute the email to each address.

SMTP-Send will by default treat any distribution list that has its own email address attribute, as a server side list, as described above. You can force SMTP-Send to expand the list (as if it were a client side list), by setting the ***UseDLAddress*** configuration parameter to ***NO***. This behavior can be set for the configuration file, and cannot be overridden on a job by job basis.

E-mail address lookup

E-mail address lookup lets you obtain an e-mail address from your LDAP directory, for instance by looking up a Microsoft Exchange alias or Lotus Notes shortname. Unlike distribution list processing, e-mail address lookup is intended to return only a single address. If LDAP returns multiple entries, SMTP-Send considers this to be an error, displays an appropriate message, and skips the lookup for that e-mail address.
You can also call SMTP-Send's LDAP address lookup routines from your own address validation, address lookup and data manipulation exit programs.

OP/SS With OP/SS, LDAP e-mail address lookups take one of two forms, userid and SNADS.

- Userid lookups are only performed for addresses specified using the Addressing extensions, FROM, TO, CC or BCC. Userid lookups will be performed for any e-mail address that does not contain a domain part (the part of the address following the @ character). The specified e-mail address will be used to perform a uid search as described below to obtain a complete e-mail address.
- SNADS address lookups are specified using a search filter. The search filter that OP/SS uses works with LDAP servers where the LDAP directory's uid field contains the SNADS user ID. If your LDAP directory stores SNADS user IDs in another field you can specify it using the search filter template that is described below. When you provide a search filter template, it is the only search filter that SMTP-Send uses.

Requirements and limitations

SMTP-Send requires OS/390 Release 5 or later in order to use LDAP.

SMTP-Send works with LDAP V2 and LDAP V3 servers.

Currently SMTP-Send supports only simple authentication. It does not support SSL binds to the LDAP server. If SMTP-Send connects to a V3 LDAP server, you can omit the bind name and password to perform an unauthenticated lookup.

LDAP client API

The SMTP-Send LDAP programs include their own LDAP client API. This provides 2 benefits, first, the API works within the CICS environment, and second, the IBM LDAP client API is not required.

Testing your LDAP environment

IBM ships a MVS-specific version of the standard LDAP utility program, ldapsearch, you can use to test your LDAP client API implementation under TSO before you activate the LDAP support in SMTP-Send. The TSO command

```
ldapsrch -h IP_address_of_LDAP_Server -V 3 -s base -b "" "objectclass=*"
```

provides information about the LDAP server named in the command. The result should look similar to this:

```
objectclass=top
subschemasubentry=cn=schema
namingcontexts=NOT Printable
supportedextension=1.3.6.1.4.1.1466.20037
supportedextension=LanguageCodes
supportedsaslmechanisms=EXTERNAL
supportedldapversion=3
supportedldapversion=2
```

LDAPSRCH is described in the IBM book LDAP Server Administration and Usage Guide, (SC24-5861), section 'Running the LDAP Operation Utilities in TSO.' Use this book and the LDAP book referenced earlier in this chapter to diagnose any errors issued by this command.

Before you can run LDAPSRCH under TSO you must ensure that the following environment is set up:

1. PDS (GLDHLQ.SGLDLDAP), where the LDAP Server load modules were installed, must be specified in one of LINKLIB, LPALIB or STEPLIB.
2. PDS (GLDHLQ.SGLDEXEC), which contains the CLISTs needed to run LDAPSRCH, must be available in SYSEXEC.

Activating LDAP lookups

LDAP processing is activated via SMTP-Send configuration statements. If migrating from a release prior to R5.2.0, refer to the Migrating section below.
To activate distribution list expansion, use the following SMTP-Send configuration statement:

```
LDAP_DISTLIST_PGM 'LDAPDL,IP address[:port|389] [,bind name][,{bind_password|"bind
     password" |*}][,D]'
```

To activate e-mail address lookup, use the following SMTP-Send configuration statement:

```
LDAP_LOOKUP_PGM 'LDAPAE,IP address[:port|389] [,bind name][,{bind_password|"bind password"
     |*}],[D],[template],[source_attribute]'
```

where:

Parameter	Description
IP Address	specifies the IP address of the LDAP server followed by the TCP/IP port number the LDAP server listens on. The IP address is a mandatory field and must be specified.
Bind name	specifies the bind distinguished name that is required to authenticate with the LDAP server. If no authentication is required you can skip this parameter and code only the comma. If the name contains spaces or commas, enclose it in double quotes.
Bind password	specifies the bind password for the bind name that is required to authenticate with the LDAP server. If no password is required you can skip this parameter and code only the comma. If the password contains spaces or commas, enclose it in double quotes. To avoid coding passwords in clear text in the configuration file, code an asterisk instead of the password. The LDAP exit will obtain the password from the `LDAP_PASSWORD` configuration statement which is stored in encrypted form.
DEBUG	The character '`D`'causes the LDAP routines to produce diagnostic information. Data is written to the LDAPDBUG DD, usually a SYSOUT dataset. Specifying '`V`' instead of '`D`' will produce verbose debugging information.
LDAP search filter template	SMTP-Send uses the filter template to perform lookups in custom fields that differ from the default UID template. Placeholders are used to describe the position where various runtime data will be inserted. `%JOB` – The jobname `%JUID` – The job or CICS transaction's User ID `%DOMAIN` – The default domain `%DATA` – The name from the addressing statement **OP/SS** P/SS uses additional placeholders when a template is specified. These placeholders are replaced by values from the address lookup parameter list. The placeholders `%DGN` and `%DEN` indicate where the SNADS DGN and DEN (Address and Userid) values should be substituted. For example, a template of `otherMailbox=SNADS$%DGN(%DEN)` could be used to

Parameter	Description
	lookup e-mail address for SNADS users in a Microsoft Exchange directory. The result is used as the LDAP search argument. If multiple fields are specified, enclose each one in parentheses, with no commas in between.
source LDAPattribute	(optional, address lookup only) This specifies the LDAP attribute (aka directory field name), containing the data returned on address lookups. The default is 'mail'.

The total LDAP configuration information is limited to 256 characters in length by the OPSCONFG program.

The LDAP lookup is limited by the `LDAP_Timeout` SMTP-Send configuration statement. The LDAP server must respond within the specified number of seconds or the lookup will fail. The LDAP server may be too busy to accept requests when SMTP-Send makes its connection request. SMTP-Send will retry a finite number of times as requested via the `LDAP_OpenRetry` and `LDAP_OpenDelay` SMTP-Send configuration statements. `LDAP_OpenRetry` specifies the number of times to attempt to connect and `LDAP_OpenDelay` specifies the number of seconds to wait between attempts. If the connection to the LDAP server cannot be made, SMTP-Send ends with a return code of 16.

If the LDAP lookup for an email address fails after the LDAP server has been successfully contacted, SMTP-Send ends with a return code based on your configuration settings. Return code 6 means a message was sent because `FAIL_ON_ERROR` is `NO`. Return code 7xx means a message was not sent because `FAIL_ON_ERROR` is `YES`. The xx indicates which addressing component was in error.

CICS

To use LDAP functions under CICS you must define programs LDAPAEC and LDAPDLC. INSTLIB member OPSI03 contains the CICS definition statements that you will need.

Additional LDAP Configuration Statements

LDAP_OpenRetry {3 | count}

Purpose: Specifies the number of times the LDAP exit should attempt to connect to an LDAP server.

LDAP_OpenDelay {5 | seconds}

Purpose: Specifies the number of seconds to delay between attempts to connect to an LDAP server.

LDAP_Timeout {60 | seconds}

Purpose: Specifies the number of seconds to wait, before considering the LDAP request to have timed-out.

LDAP_Password value

Purpose: Specifies the password value to be encrypted and stored in the OPSCONFG VSAM file. Specify a '`*`' character in the actual LDAP statements where the password is to be used.

Calling LDAP from your lookup, validation or data exit

SMTP-Send supports both name and directory lookups using LDAP. Only the name lookup function can be called from an exit. If you need to find data in LDAP that the base SMTP-Send code doesn't provide, you can call the SMTP-Send LDAP code to do the lookup for you. The simplest LDAP search is on the UID (UserID) attribute returning the mail (Internet e- mail) attribute value that matches the UID. You will probably request that SMTP-Send use a different LDAP attribute as the search target through a template. The LDAP lookup code will substitute supplied simple name values into the template, where substitution symbols are specified.

You can supply data for 3 substitution variables, `%1`, `%2`, `%DATA` to be substituted in your template. In addition, the current jobname, requesting user and default domain can be substituted via the `%JOB`, `%JUID` and `%DOMAIN` symbols respectively as described in the LDAP Lookup parameter list.

Calling the LDAP lookup program

The first 6 characters of the LDAP lookup program name is passed to your exit via the LDAPPGMP variable in the commarea passed to your exit. The default value is `LDAPAE`. In the MVS/Batch and TSO environment this name must be `LDAPAE`. You can alter this name in the CICS environment as described below.

Calling SMTP-Send's LDAP lookup program from MVS/Batch or TSO

When SMTP-Send runs in MVS Batch or TSO the character `'B'` is appended to the LDAP lookup program name prefix automatically when it is called by SMTP-Send. Your exit code must call the program `LDAPAEB`.

Calling SMTP-Send's LDAP lookup program from CICS

When SMTP-Send runs in CICS, the character 'C' is appended to the LDAP lookup program name prefix automatically when it is called by SMTP-Send. Since your exit program is called with a CICS LINK the runtime environment cannot be shared with the SMTP-Send mainline.

Your exit code must call the SMTP-Send LDAP exit with an EXEC CICS LINK command. The program name for the LINK will be the LDAP program prefix with the characters 'CX' appended to it. E.g. if the LDAP_LOOKUP_PGM program name is LDAPAE, you should issue EXEC CICS LINK PROGRAM(LDAPAECX) with the LDAPDATA structure as described below as the COMMAREA.

Passing Data to LDAP Lookup Program

Data is passed to the LDAP lookup program (called LDAPAE hereafter) in the LDAPDATA structure. The structure allows data to be communicated between your exit program and LDAPAE. Both data used in searching and data returned are specified in the LDAPDATA structure. In addition, LDAPAE requires information about the location of your LDAP server and signon information (if required by your server). This data is passed in the LDAP parameter list string pointed to by the LDAPDATA structure as described below. If you already use LDAP for address lookups, then you can use the same data that was specified on the LDAP_LOOKUP_PGM configuration statement. A pointer to this parameter list and its length is passed to your exit. You can override the address template and the LDAP attribute from which data is extracted by coding overrides in the LDAPDATA structure.

LDAPDATA layout

The data structure LDAPDATA in INSTLIB describes the layout of the data structure passed to LDAPAE.

Parameter	Description
int[4]	Contains the return code from the user exit. This is in addition to the value returned via R15, since CICS applications are unable to return values in registers.
address[4]	Contains a pointer to a 12-byte workarea used by the LDAP routines. This value should not be altered between LDAP calls.
address[4]	Contains a pointer to the LDAP program parameter list. See below for a description of this parameter list. The parameter string specified on the LDAP_LOOKUP_PGM configuration statement and its length is passed to the validation and address mapping exits in their respective data structures and can be used to build this string.
byte[1]	Contains the length of the LDAP program parameter list (1-255 characters)
character[8]	The current jobname, substituted into %JOB template variable.
character[8]	Requesting user. RACF or equivalent or CICS userid of the user requesting the send operation. Substituted into %JUID template variable.
character[80]	Default domain specified during SMTP-Send customization, substituted into %DOMAIN variable.
byte[1]	Contains the length of a message returned by LDAP. If no message is returned this value will be set to zero. The maximum length that can be displayed by SMTP-Send is 125 characters.
character[125]	Contains a message returned from LDAP.
character[1]	If non-zero, indicates that the name of a source field is specified. Zero indicates that the default, MAIL will be used.
address[4]	pointer to field2 value, substituted into %2 template variable.
int[4]	length of field2 value, must be between 1-128.
address[4]	pointer to field1 value, substituted into %1 template variable
int[4]	length of field1 value, must be between 1-128.
character[128]	Data for substitution into %DATA template variable.
byte[1]	length of returned data
character[128]	Returned data.
byte[1]	reserved
byte[1]	x'00'or x'40' indicates V1 format, C'1' indicates V2 and presence of LDAPCOMX
byte[1]	reserved
int[4]	length of %data variable, must be between 1-128.
address[4]	optional pointer to name of the source field LDAP data will be extracted from.
int[4]	length of optional addressing template. Zero indicates no template was supplied, template in parameter list, if any, will be used.
address[4]	pointer to optional addressing template. Presence of a template overrides any template specified in the parameter list.
int[2]	time in seconds to wait for a response from a LDAP search request
int[2]	number of times to attempt to connect to the LDAP server
int[2]	time in seconds to wait between connection attempts
byte[2]	reserved
LDAPCOMX	
	Start of V2 extension to LDAPDATA
address[4]	pointer to encryption routine supplied by SMTP-Send

address[4]	pointer to decryption routine. Supplied by SMTP-Send on calls to LDAP or by user exits when calling LDAP from within an exit. Zero indicates that the password described below is in plain text. When calling encryption or decryption programs, pass the address of the following structure in General Register 1
byte	length of plain text password before encryption or after decryption.
byte	length of encrypted password after encryption or before decryption.
character[128]	plain text password value. Place value to be encrypted here before calling encryption program. Value is replaced by decryption program
byte	reserved
character[128]	encrypted text password value. Place value to be decrypted here before calling decryption program. Value is replaced by encryption program
byte	reserved

The LDAP program will return:

Return Code	Meaning
-1	no entry matched
0	normal, returned requested value in returned data
4	multiple matches found. Message area contains as many copies of returned data as will fit
8	invalid or missing parameters given
12	LDAP Bind failed
16	LDAP Open failed

By default, LDAPAE returns the value from the LDAP `'MAIL'` attribute when a matching entry is found. You can change the LDAP attribute used by either a) adding the name of the field as the 6th parameter as shown below or b) passing the address of the LDAP attribute used to extract data from, in the optional LDAPDATA source field. If the LDAPDATA source field contains an address, that value will be used and any 6th parameter if present is ignored.

```
"IPaddress:389,bind user,bind pwd, ,(Hostemail=%DATA),sourceattribute"
```

With this parameter string, when searching for a matching entry for the %DATA variable "FRED", SMTP-Send uses the following LDAP search filter:

```
search base:          default
search filter:        (&(Hostemail=FRED))
attributes to return: sourceattribute
```

Examples:

```
ldap.bigfoot.com
ldap.bigfoot.com:389
ldap.bigfoot.com,binduser,bindpwd
127.0.0.1:389,"Acme Admin",password
127.0.0.1,"cn=user,cn=domain",pass,D

127.0.0.1,,,D
127.0.0.1:389,,,D,otherMailbox=SNADS$%2(%1)
127.0.0.1:389,,,D,"(uid=%1)(proposedaltorgunit=%2),e_mail"
```

Searching using LDAP Lookups

UID search

In the absence of an address template passed in the LDAP Program parameter list, LDAP lookup performs a UID search. This searches for the `%1` value (or the `%DATA` value, if `%1` is not passed), in the LDAP UID attribute (**Note:** when connected to a Microsoft Active Directory LDAP server, SMTP-Send will automatically search the SAMACCOUNTNAME attribute instead of the UID attribute). The search filter will select only person entries. The search base is left to default. For example, consider a request with only the symbol `%DATA` defined with a value of `FRED`. The resulting LDAP search filter is:

```
search base:          default
search filter:        (&(objectclass=person)(uid=FRED))
attributes to return: mail
```

Template search

The template search is performed when the installation needs to map simple name values, searching either an attribute other than UID or multiple attributes. When the caller passes an addressing template, the LDAP lookup substitutes simple name values into the template, where the symbols, `%1, %2, %DATA, %JOB, %JUID` or `%DOMAIN` are specified. The resulting string is then used to build the actual LDAP search filter. The search filter will select only entries with a LDAP objectclass of person. The search base is left to default. For example, consider a request for a simple name of `FRED` and the following addressing template, `Hostuserid=%DATA`. The resulting LDAP search filter is:

```
search base:          default
search filter:        (Hostuserid=FRED)
attributes to return: mail
```

Template search using multiple attributes

A template search can be used to match on multiple attributes. Since the template is used as the LDAP search filter, it must be specified with a valid LDAP format. Each attribute=value should be enclosed within parentheses. For example, the template `(Hostemail=Y)(userid=%JUID)(mail=%DATA)` where the job was submitted by `Sam` and `foo@bar.com` was passed in the DATA field with a source field of `location` in the LDAP parm string will result in the following LDAP search filter:

```
search base:          default
search filter:        (&(Hostemail=Y)(userid=Sam)(mail=foo@bar.com))
attributes to return: location
```

A sample parm string using this template would be:

```
"IPaddress:389,bind user,bind pwd, ,(Hostemail=Y)(userid=%JUID)(mail=%DATA),location"
```

Ambiguous search results

When either a template or UID search returns more than 1 result, LDAPAE returns the ambiguous result code back to the caller. The message buffer in the caller's commarea contains as many of the resultant attribute values as will fit.

Migrating from a release prior to Release 5.2.0

Release 5.2.0 introduced some significant changes in performance to SMTP-Send's LDAP routines. To provide these improvements, the linkage used to call the SMTP-Send provided LDAP exits changed with this release. If you do not call SMTP-Send's LDAP routines from a SMTP-Send exit or provide your own LDAP routines, you can run SMTP-Send with a pre-Release 5.2.0 OPSCONFG file but you will not get any improvements when expanding distribution lists. You must specify the LDAP distribution lookup information with the `LDAP_DISTLIST_PGM` OPSCONFG statement to see the improvements.

The statements `LDAP_LOOKUP_PGM`, and `LDAP_DISTLIST_PGM` both cause SMTP-Send to use the 'new' style linkage, and are not compatible with a customer written exit, unless the exits are updated.

Migrating custom LDAP exits

A customer written program used to call LDAP without using routines supplies with SMTP-Send must be called using one of the following OPSCONFG statements:

```
MAPPINGPGM   custom_addressing_exit_name,parms
DISTLISTPGM custom_distribution_exit_name,parms
```

These programs will be called as separate runtime environments or LE enclaves from the SMTP-Send mainline and other than the 12 byte token passed to these programs, SMTP-Send makes no effort to retain any data or environmental state from one call to the next.

Customers calling SMTP-Send's LDAP exits from their own lookup, data or address validation exit.

Calling SMTP-Send's LDAP exit from a customer written exit in MVS/Batch or TSO

This is supported without any changes required, and with the full benefits of the enhanced LDAP exits.

In addition, the customer exit can now use a different LDAP session from the one used by SMTPSend. These LDAP session parameters are passed by the customer exit using the same interface (i.e. they are not specified to SMTPSend). Specifically, this means the customer exit can query a different LDAP server, using a different userid & password, than what is specified in the CONFIG file.

Calling SMTP-Send's LDAP exit from a customer written exit in CICS

Changes are required to call LDAP from your CICS exit and the performance improvements of the enhanced LDAP exits are not available. You must recode your exit to call the 'CX' suffixed version of LDAPAE instead of the 'C' suffixed version as in earlier releases. The calling conventions and data passed remain the same as in previous releases.

Supplementary LDAP reading

The LDAP protocol is defined by a number of Internet Engineering Task Force (IETF) request for comments (RFCs.) The IBM redbook, Understanding LDAP, (SG24-4986), provides a good introduction to LDAP, with many diagrams to assist you in understanding the concepts behind the RFC contents.

You may also need to refer to these RFCs if you customize the LDAP search filter.

- RFC 1777 Lightweight Directory Access Protocol
- RFC 1778 The String Representation of Standard Attribute Syntaxes
- RFC 1779 A String Representation of Distinguished Names
- RFC 1960 A String Representation of LDAP Search Filters

Logging distributions

SMTP-Send can log details of e-mail distributions, thereby creating an audit trail of its activity. Log records can be written either to, a dataset (batch only), a CICS journal (CICS only) or the System Management Facility (SMF, batch & CICS). These features are enabled using parameters in the SMTP-Send configuration file. The TBS SMTP Spool server can perform the same logging as a batch SMTP-Send job.Distributions that cannot be sent successfully are also logged, along with the return code. This allows for analysis of send jobs having distribution problems. Logging distributionsLogging distributionsLogging distributionsLogging distributionsLogging distributions

Logging from batch to SMF

You must ensure that SMTP-Send is APF authorized as described in APF authorization.

Logging from batch to a sequential file

If you decide to log to a sequential file then create an empty dataset using JCL similar to the following:

```
//ALLOC  EXEC PGM=IEFBR14
//USERLOG DD DSN=logdatasetname,DISP=(NEW,CATLG),
//  UNIT=3380,VOL=SER=vvvvvv,SPACE=(CYL,(n,n),
//  DCB=(DSORG=PS,RECFM=VB,LRECL=32756)
```

You would then reference this dataset, using the LOGFILE DD statement, in all SMTP-Send JCL procedures, for example:

```
//LOGFILE  DD DSN=logdatasetname,DISP=MOD
```

Note: SMTP-Send does not manage the sequential file to prevent x37 abends for out-of-space conditions.

Logging from CICS

Logging to a sequential file is not available in CICS. If you want to log distributions from CICS you must define an appropriate *journalmodel* resource, which identifies a CICS journal name (e.g. DFHJ10), and relates this journal either to the MVS logstream, or to SMF. Journalmodel definitions may be created and updated online using the CEDA transaction.
Select TYPE=MVS if you want CICS to direct the named journal (e.g. DFHJ10) to an MVS logstream that you've defined.
Select TYPE=SMF to write to SMF. The data will be written as a journaling subtype CICS SMF record (SMF type 110), containing the identifier being the SMF_TYPE value described below.
The following CICS manuals provide details regarding the creation & use of user (general) journals.

- *CICS System Definition Guide…* information about JournalModels for general CICS journals.
- *CICS Operations & Utilities Guide…* information about reading & dumping general CICS journals (DFHJUP).
- *CICS Customization Guide…* describes CICS SMF records.

To log to a CICS journal or to use SMF, once your system programmer has enabled the feature, no explicit specification is required in batch jobs or CICS transactions.

Logging configuration statements

If you decide to have SMTP-Send create a log of e-mail distributions, you must specify where and how to write log records. Use the following configuration statements to set up your logging.

LOGGING

Purpose: indicates if SMTP-Send generates log records, and if so, specifies where to write them.
`YES` - In batch, writes log records to the sequential dataset defined by the LOGFILE DD statement in the SMTP-Send JCL. In CICS, writes log records to the user-defined journal indicated by the LOG_JNUM value.
`SMF` – Log records are formatted for the SMF log. From batch jobs, log records of type SMF_TYPE are written to SMF. In CICS, log records containing the identifier SMF_TYPE are written to the CICS SMF record (journal subtype).
Note: In CICS, all log records are written to the journal name indicated by the LOG_JNUM value. See the following description of LOG_JNUM for details. Based on the Journamodel definition, CICS will in turn direct these writes either to the associated MVS logstream, or to its SMF record, as a journal subtype.

Syntax: `LOGGING [No|Yes|SMF]`
Default: `LOGGING No`
Example: `LOGGING SMF`

LOG_JNUM

Purpose: (CICS only) indicates which CICS journal to write log records. The journal number must be a value between 2 and 99. This results in the traditional journal names in the format, DFHJnn.

Syntax: `LOG_JNUM [nn]`
Default: `LOG_JNUM 2`
Example: `LOG_JNUM 3`

SMF_TYPE

Purpose: Specifies the SMF record type (or number) that contains log data. This value is used when logging to SMF from either batch or CICS. This is a value between 128 and 255. Check with your MVS Systems Programmer to determine which SMF type should be used. If you are running the TBS SMTP Spool Server, please see, RELAY_SMF_TYPE.
Note: in CICS, SMF writes are directed to the CICS SMF record (type 110), journal subtype. The SMF_TYPE value will appear as an identifier within the data.
Syntax: `SMF_TYPE nnn`
Default: `SMF_TYPE 128`
Example: `SMF_TYPE 255`

Note: In order to write log data to SMF from batch, SMTP-Send must be APF authorized. Refer to APF AUTHORIZATION

Sample configurations

Logging distributions from batch jobs to a sequential file.

```
LOGGING YES
```

Logging distributions from batch to SMF.

This example would write type 130 SMF records.

```
LOGGING SMF
SMF_TYPE 130
```

Logging distributions from batch and CICS jobs to SMF.

In batch, a type 130 SMF record will be created.
In CICS, a CICS SMF journaling record (type 110, subtype x'0000') is created, containing the ID 130

```
LOGGING SMF
LOG_JNUM 3  (a JOURNALMODEL resource definition exists with values, JOURNALNAME=DFHJ03,
TYPE=SMF)
SMF_TYPE 130
```

Logging distributions from CICS jobs to CICS journal DFHJ05

```
LOGGING YES
LOG_JNUM 5  (a JOURNALMODEL resource definition exists with values, JOURNALNAME=DFHJ05,
TYPE=MVS)
```

Log record format

When logging to SMF or to CICS journals, the standard SMF or CICS headers, precede the SMTP-Send log record. The user data that SMTP-Send logs, consists of three sections:

1. offset/length values for variable fields (third section)
2. fixed length fields
3. variable length fields

Offset/length values

The first section consists of a series of 8 offset/length value pairs for the variable fields. These values appear in the order FROM:, REPLY-TO:, SUBJECT:, TO:, CC:, BCC:, ATTACH and PATHINFO. The offset and length values apply to the field data only and do not include the character introducers. Offsets are halfword values that define the beginning of the field relative to the beginning of the user data area. Lengths are halfword values that define the length of the corresponding field.

Fixed length fields

The data in each field is left justified, blank padded with and ends with one null (x'00') character.
The fixed length data area is formatted as follows:

Field	Description

VERSION[2]	X'01' Product and PATHOFF/PATHLEN information available X'02' Jobid is available
PRODUCT[1]	`G -OPSGS3B, H -OPSGS3` (SMTP-Send Batch, SMTP-Send CICS) `O -PIPGS3B, P -PIPGS3` (OP/SMTP-Send Batch, SMTP-Send CICS, OfficePath syntax) `R -RAPIDGS3, S -APIGS3` (OP/SMTP-Send Batch, SMTP-Send CICS, RAPID syntax) `Q -SSWDA + PIPGS3B` (OP/SMTP-Send Batch for Softswitch, OfficePath syntax) `T -SSWDA + RAPIDGS3` (OP/SMTP-Send Batch for Softswitch, RAPID syntax) `J -OPSSSRVR` (TBS SMTP Spool Server)
RSVD[3]	reserved
EXTOFFST[2]	offset to future Log record extensions
JOBNAME[9]	name of the job
JOBUSERID[9]	name of the user who submitted job
RETURNCODE[5]	job return code
TOTAL SIZE SENT[9]	number of bytes of data sent
DATE[32]	date and time sent, including GMT offset Example: Thu, 22 Nov 2001 11:51:44 –0500 Note: for items relayed by Spool Server may contain the value from the Date: header in the message.
JOBID[9]	Job identifier, JOBnnnnn for batch, TSUnnnnn for TSO.

Variable length fields

The remainder of the log record is the variable length fields. Each field is prefixed by a character introducer like FROM:, TO:, etc. Fields with multiple values are separated by commas. Only TO: and FROM: are required; the other fields are optional.

Field	Description
FROM:	FROM address or USER NAME
REPLY-TO:	REPLY-TO address
SUBJECT:	message subject (up to 256 characters)
TO:	one or more TO addresses separated by commas
CC:	one or more CC addresses separated by commas
BCC:	one or more BCC addresses separated by commas
ATTACH:	one or more attachment file names separated by commas
DATAPATH:	Information as to the delivery path used for the message: `TCP(`*`ip_address`*`)` - Primary or secondary SMTP relay server `SPOOL(node,wtrid,class)` - SMTP Server STC information `DDName(ddname)` - debugging redirection to external file `TSQ(8-char-CICS-TSQid)` - debugging redirection to CICS TSQ `Fax(Applid,TSQ: 8-char-CICS-TSQid, DDNAME:filename)` - send to OV/Fax V3 `Unknown` - message was not delivered

Sample log record

Here are some example log records in dump format.

This is a sample record from the LOGFILE DD file. It begins with the 8 offset/length value pairs, as described above.

```
LOGFILE record
0000: 00780013 00960013 00B30021 00D90010   *.....o.......R..*
0010: 00000000 00000000 00F20019 01160015   *.........2......*
0020: 0002C700 00000000 E35BD3D6 C7404040   *..G.....T$LOG   *
0030: 00E3C2E2 F0F0D3C4 4000F0F0 F0F000F0   *.TBS00LD .0000.0*
0040: F0F0F0F8 F3F8F000 D496956B 40F1F340   *0008380.Mon, 13 *
0050: D481A840 F2F0F1F3 40F1F97A F4F97AF1   *May 2013 19:49:1*
0060: F24060F0 F4F0F000 D1D6C2F0 F0F2F1F7   *2 -0400.JOB00217*
0070: 0040C699 96947A40 D6D7E2D4 E3D7E27C   *. From: OPSMTPS@*
0080: A382A2A2 9686A34B 83969440 D9859793   *tbssoft.com Repl*
0090: A860E396 7A40D6D7 E2D4E3D7 E27CA382   *y-To: OPSMTPS@tb*
00A0: A2A29686 A34B8396 9440E2A4 82918583   *ssoft.com Subjec*
00B0: A37A40C9 9393A4A2 A39981A3 8540C281   *t: Illustrate Ba*
00C0: A3838840 93968787 89958740 A3964086   *tch logging to f*
00D0: 8993854B 40E3967A 40C4C5D4 D67CE3C2   *ile. To: DEMO@TB*
00E0: E2E2D6C6 E34BC3D6 D440C1A3 A3818388   *SSOFT.COM Attach*
00F0: 7A40E3C2 E2F0F04B D6D7E24B C4C1E3C1   *: TBS00.OPS.DATA*
0100: 4BE5C24D C1E2C1E3 E7E35D40 C481A381   *.VB(ASATXT) Data*
0110: D781A388 7A40E3C3 D74DA294 A3974BA3   *Path: TCP(smtp.t*
0120: 82A2A296 86A34B83 96945D              *bssoft.com)     *
```

Here is a sample logged to SMF (type 130). The SMF header occupies the first 18 bytes.

```
SMF Log record
0000: 01280000 1E82006C 03900113 133FD4E5   *.....b.%......MV*
0010: E2C40078 00130000 00000095 003100CB   *SD.........n....*
0020: 00300000 00000000 00000000 00000118   *................*
0030: 00100002 D6000000 0000E35B E2D4C640   *....O.....T$SMF *
0040: 404000E3 C2E2F0F0 D3C44000 F0F0F0F0   *  .TBS00LD .0000*
0050: 00F0F0F0 F0F1F3F5 F900D496 956B40F1   *.00001359.Mon, 1*
0060: F340D481 A840F2F0 F1F340F1 F97AF3F9   *3 May 2013 19:39*
0070: 7AF4F740 60F0F4F0 F000D1D6 C2F0F0F2   *:47 -0400.JOB002*
0080: F1F50040 C6999694 7A40C6D9 D6D4E4E2   *15. From: FROMUS*
0090: D97CA382 A2A29686 A34B8396 9440E2A4   *R@tbssoft.com Su*
00A0: 82918583 A37A40E3 5BE2D4C6 4DD1D6C2   *bject: T$SMF(JOB*
00B0: F0F0F2F1 F55D40E3 85A2A340 D3C4C1D7   *00215) Test LDAP*
00C0: 40A689A3 8840D686 86898385 D781A388   * with OfficePath*
00D0: 40A58199 898195A3 40E3967A 4040D3C5   * variant To:  LE*
00E0: D6D56094 A2A77C94 A2A74BA3 82A2A296   *ON-msx@msx.tbsso*
00F0: 86A34B83 96946B40 40938489 81948195   *ft.com,  ldiaman*
0100: A37CA382 A2A29686 A34B8396 9440C481   *t@tbssoft.com Da*
0110: A381D781 A3887A40 C4C4D581 94854DC4   *taPath: DDName(D*
0120: C1E3C1D6 E4E3405D                     *ATAOUT )        *
```

Dynamic allocation keywords

The following are the dynamic allocation keywords that SMTP-Send recognizes:

Identifier	JCL Equivalent	Format	Description
blocklen	SPACE=(len,..)	int	Average block length for space allocation
cdisp	DISP=(s,n,cd)	KEEP | CATLG | DELETE	Conditional (ABEND) disposition
cyl	SPACE=(CYL,...)	none	Allocate space in cylinders
dataclas	DATACLAS=	char[9]	SMS data class
dcbddname	DCB=*	char[9]	Copy DCB information from DD statement
dcbdsname	DCB=ds	char[47]	Copy DCB information from dataset
defer	UNIT=(,,DEFER)	none	Defer mounting until open
dest	DEST=	char[9]	SYSOUT destination node
destuser	DEST=(n,user)	char[9]	SYSOUT destination userid
dir	SPACE=(u,(p,s,d))	int	Number of directory blocks
disp	DISP=	NEW | OLD | MOD	Dataset allocation status
dsname	DSN=	char[47]	Name of dataset to allocate
expdt	LABEL=EXPDT=	char[5] | char[7] [1]	Expiration date
hold	HOLD=YES	none	Hold SYSOUT for later processing
labelseq	LABEL=seq	int	File sequence number
like	LIKE=	char[47]	DSname of a model dataset (SMS)
member	DSN=ds(mem)	char[9]	PDS member name
mgmtclas	MGMTCLAS=	char[9]	SMS management class
ndisp	DISP=(s,n)	KEEP|CATLG|DELETE	Normal Disposition
password	none	char[9]	Dataset password
private	VOL=(PRIVATE,)	none	Private volume required
protect	PROTECT=YES	none	Protect dataset with RACF
refdd	REFDD=	char[27]	Copy DCB attributes from DDname
retention	LABEL=RETPD=	int	Retention period
rlse	SPACE=(,,RLSE)	none	Release unused space
round	SPACE=(,,,,ROUND)	none	Round up space request
secmodel	SECMODEL=	char[47]	SMS security model dataset (2)
space1	SPACE=(u,(s1,s2))	int	Primary space allocation
space2	SPACE=(u,(s1,s2))	int	Secondary space allocation
torclas	STORCLAS=	char[9]	SMS storage class
sysout	SYSOUT=	char[1]	Sysout class
sysoutform	SYSOUT=(c,,f)	char[5]	SYSOUT forms
sysoutwtr	SYSOUT=(c,p)	char[9]	SYSOUT writer program name
tracks	SPACE=(TRK,...)	none	Allocate space in tracks
volcount	VOL=(,,,count)	int	Maximum number of volumes
volref	VOL=REF=	char[46]	Request same volume as another dataset
volseq	VOL=(,,seq)	int	Sequence number of first volume to mount
volume	VOL=SER=	char[7]	Multiple Names of required volumes

[1] An expiration date may be specified in one of three formats. In a five-digit expiration date, the first two digits are interpreted as the years since 1900. In a seven-digit expiration date, the first four digits are the entire year. The format "yyyy/ddd", as used with the JCL EXPDT keyword, is also supported.