CA OPS/MVS® Event Management and Automation

Parameter Reference

Release 12.2

ca technologies

CA Technologies Product References

This document references the following CA products:

- CA 7™ Workload Automation (CA 7)
- CA ACF2™ for z/OS (CA ACF2)
- CA Automation Point
- CA Jobtrac® Job Management (CA Jobtrac)
- CA Librarian® Base for z/OS (CA Librarian Base for z/OS)
- CA MII Data Sharing (CA MII)
- CA MIM™ Resource Sharing (CA MIM)
- CA Netman™ (CA Netman)
- CA NetMaster® Network Automation (CA NetMaster)
- CA NSM
- CA NSM System Status Manager CA OPS/MVS® Option (CA NSM SSM CA OPS/MVS Option)
- CA OPS/MVS® Event Management and Automation (CA OPS/MVS)
- CA PDSMAN® PDS Library Management (CA PDSMAN)
- CA Scheduler® Job Management (CA Scheduler)
- CA Service Desk (CA Service Desk)
- CA SYSVIEW® Performance Management (CA SYSVIEW)
- CA Top Secret® for z/OS (CA Top Secret)

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Documentation Changes

The following documentation updates have been made since the last release of this documentation:

Note: In PDF format, page references identify the first page of the topic in which a change was made. The actual change may appear on a later page.

- Added the GLOBALTEMPWSGCIV Parameter (see page 235) section.
- Added the USSSECURITY Parameter (see page 333) section.
- Added the SSMGARESTXTUPDT Parameter (see page 255) section.

Contents

Chapter 7: Display-only Parameters 385

Chapter 1: Using This Reference

This section contains the following topics:

About This Guide

This guide describes the CA OPS/MVS parameters that affect the operation of the base CA OPS/MVS product, its facilities, and its optional interfaces.

Characters Used In z/OS Commands

Following is a list of the valid characters that can be used in z/OS commands:

Character Set: Alphanumeric

Contents:

- Alphabetic, Uppercase A through Z
- Numeric, 0 through 9

Character Set: National (see *Note*)

Contents:

- At sign @
- Dollar sign $
- Pound sign #

These characters can be represented by hexadecimal values X'7C', X'5B', and X'7B'.

Character Set: Special

- Comma ,
- Period .
- Slash /
- Apostrophe '
- Left parenthesis (
- Right parenthesis)
- Asterisk *
- Ampersand &
- Plus sign +
- Hyphen -
- Equal sign =

- Cent sign ¢
- Less than sign <
- Vertical bar |
- Exclamation point !
- Semicolon ;
- Percent sign %
- Underscore _
- Greater than sign >
- Question mark ?
- Colon :
- Quotation marks “

Guidelines:

- Do not use a prefix that is an abbreviation for an already defined process. For example, a prefix such as D conflicts with z/OS commands such as DISPLAY.
- Do not define a prefix that is either a subset or a superset of an existing prefix with the same first character. For example, if command prefix $ABC already exists, command prefixes $, $A, and $AB are subsets of, and conflict with, the original prefix. Similarly, prefixes $ABC1 and ABC also conflict with existing prefix $ABC because they are supersets with the same first character. You can, however, define command prefixes ABC, BC, or C, because they do not start with the same first character as the existing prefix. You can see which prefixes exist by issuing the DISPLAY OPDATA z/OS command or by using the OPSCFP OPS/REXX function.

Note: The system recognizes the following hexadecimal representations of the U.S. National characters: @ as X'7C', $ as X'5B', and # as X'7B'. In countries other than the U.S., the U.S. National characters represented on terminal keyboards might generate a different hexadecimal representation and cause an error. For example, in some countries, the $character may generate X'4A'.

Chapter 2: Parameters

This section contains the following topics:

Introduction to Parameters

This chapter describes the parameters that allow you to tailor CA OPS/MVS operations to your requirements, to monitor the performance of CA OPS/MVS, and to monitor the results of your automation rules and procedures. It also explains how to change or display the values of CA OPS/MVS parameters.

The CA OPS/MVS product has two types of parameters:

- Parameters governing the operation of CA OPS/MVS
- Parameters governing CA OPS/MVS facilities and interfaces

These two types of parameter allow you to do the following:

- Tailor the operation of CA OPS/MVS to your requirements.
- Look at internal fields and counters that reflect the results of the operation of CA OPS/MVS and your automation rules and procedures.

Parameters Governing Operation

These parameters affect how CA OPS/MVS:

- Browses OPSLOG contents. OPSLOG parameters determine what information about system events you can see when you browse the OPSLOG.
- Processes commands and messages. Command parameters govern how CA OPS/MVS processes commands, and message parameters affect the CA OPS/MVS message processing.
- Uses CPU memory and storage. Memory/storage parameters determine how CA OPS/MVS acquires and allocates CPU memory.
- Responds to internal abends. Abend parameters determine, and keep track of, how many times CA OPS/MVS recovers from internal abends.

CA OPS/MVS also provides a set of parameters you can use, with help from CA Customer Support, to resolve CA OPS/MVS operating problems.

Parameters Governing Facilities and Interfaces

These parameters, which control the operation of CA OPS/MVS facilities and the interface of the product to other software or subsystems, include:

- Parameters that affect the CA OPS/MVS AOF, ECF, and OSF facilities
- Parameters affecting rules, global variable, and OPS/REXX processing
- Parameters controlling how CA OPS/MVS interfaces with basic components such as JES and TSO
- Parameters controlling how CA OPS/MVS interfaces with optional components such as MSF and CA Event Manager

This chapter describes parameters that affect CA OPS/MVS message and command processing, memory and storage usage, and responses to abends. For information on parameters related to CA OPS/MVS facilities, see the chapter “Parameters for Facilities.” For descriptions of parameters for basic CA OPS/MVS interfaces, see the chapter “Parameters for Basic Interfaces.” For descriptions of parameters for optional CA OPS/MVS interfaces, see the chapters “Parameters for Optional Interfaces” and “Parameters for Optional Interfaces for Separately Licensed Features.”

When to Change Parameter Values

You can change the CA OPS/MVS parameters only while CA OPS/MVS is active. Some parameters need to be changed when you initialize CA OPS/MVS; other parameters can be changed anytime. The descriptions in this manual indicate the appropriate time to change the value of each parameter.

Obsolete Parameters

Typically, when a parameter becomes obsolete, it remains in the product for two releases. This occurs to give you time to make any necessary modifications to your automation.

After a parameter has been obsolete for two releases, it is removed from CA OPS/MVS. Attempting to use a parameter that has been removed from the product causes an error.

About Display-only Parameters

For descriptions of some of the most important display-only parameters, see the chapter "Display-only Parameters." If you need information about display-only parameters not documented in this section, use OPSVIEW option 4.1.1. See the documentation for option 4.1.1 in the *OPSVIEW User Guide*.

YES|NO and ON|OFF

The values YES|NO and ON|OFF may be used interchangeably for CA OPS/MVS parameters.

Setting or Displaying Parameter Values

This section discusses setting up and displaying parameter values.

OPSPRM and OPSPARM Function

You use the OPSPRM function of OPS/REXX or the CA OPS/MVS OPSPARM TSO command to set or display the values of CA OPS/MVS parameters. The OPSPRM function is used by OPSVIEW option 4.1.1, Set/Display CA OPS/MVS parameters. The OPSPRM function processes parameter change or display requests more rapidly than the OPSPARM command, so use OPSPRM if faster performance is important to you. In addition, you can use OPSPRM directly in an AOF rule.

To display parameters, use the following syntax:

var = OPSPRM("SHOW","*parmname*"[,["INFO"][,["NAMES"][,"system"]]])

To set parameters, use the following syntax:

var = OPSPRM("SET","*parmname*","*value*"[,,"*system*"])

OPSPRM Arguments

The OPSPRM function has the following arguments:

parmname

Specifies the name of the CA OPS/MVS parameter (for example, OSFCPU) to be displayed or set. This name can contain no more than 50 characters.

When used to display parameter values (SHOW argument), *parmname* can be any of the following:

- *name*-Displays the value of the named parameter or the values of all parameters in the named group.
- GROUPS-Displays a list of the parameter groups.
- ALL-Displays all parameter values.

value

Specifies the new value you are assigning to a parameter. For example, setting *value* to YES changes the value of the parameter to YES. OPS/REXX requires this argument only when you are setting or resetting parameters.

INFO

(Optional) Use the INFO argument with the SHOW argument to display the possible values the parameter can have.

NAMES

(Optional) Use the NAMES argument with the SHOW argument to display the names and modifiability indicators of individual parameters.

system

(Optional) Provides CA OPS/MVS with the capability to execute the OPSPRM function on external systems that have been defined to the MSF. The MSF establishes VTAM sessions between copies of CA OPS/MVS, permitting any copy to issue a command on any other copy and to receive its response. Specify any of these values for *system*:

- The name of an individual system on which the function is to execute
- EXT to indicate that the function should execute on all active systems other than the local system
- ALL to indicate that the function should execute on all active systems

As referred to here, *active* is the status of the MSF systems through which the communication takes place.

Note: A response to the OPSPRM function is returned only if you specify an individual system name for *system*.

Examples: Invoke OPSPRM

- To display the address of a module, invoke the OPSPRM function as follows:

```
RetCode = OPSPRM("SHOW","OPITQWFU")
Say "OPSPRM() return code is:" RetCode
Do While Queued() <> 0
 Pull Data
 Say Data
END
```

In response, the following information is displayed:

```
OPS1000I OPSPRM() return code is:     0
OPS1000I ADDRESS OF MODULE OPITQWFU X'051AC000'
```

- To display the address of a module with information and name, invoke OPSPRM as follows:

```
RetCode = OPSPRM("SHOW","OPITQWFU","INFO","NAMES")
do while QUEUED() > 0
 pull data
 say data
end
```

In response, the following information displays:

```
OPS0996I ADDRESS OF MODULE OPITQWFU                X'0B3AF000'
OPS0996I OPITQWFU                             N PRODMODULES
OPS0996I MODULE ORIGINAL ADDRESS                   X'0B3AF000'
OPS0996I MODULE FINAL ADDRESS                      X'0B3AF000'
OPS0996I MODULE VECTOR TABLE ENTRY ADDRESS         X'0979BA30'
OPS0996I MODULE SIZE                          6880 BYTES
OPS0996I MODULE ORIGINAL LOCATION                  EPRIVATE
OPS0996I MODULE FINAL LOCATION                     EPRIVATE
OPS0996I MODULE PROTECT KEY                        CODE (2)
OPS0996I MODULE AMODE                         31
OPS0996I MODULE VERSION                       rr.rr.sp
OPS0996I MODULE PROGRAMMER NAME                    OPSASM
OPS0996I MODULE ASSEMBLY DATE                      07/21/07
OPS0996I MODULE ASSEMBLY TIME                      21.50
OPS0996I MODULE IS ELIGIBLE FOR RELOAD             YES
```

For MODULE VERSION, *rr.rr* refers to the release of CA OPS/MVS and *sp* refers to the service pack number.

Values that OPSPRM Returns

The OPSPRM function returns one of these codes:

Return Code	Description
0	The OPSPRM function executed successfully.
4	N/A
8	The function cannot be used before CA OPS/MVS becomes active.
12	You omitted the SET argument or the SHOW argument.
16	You specified no new parameter value with the SET argument.
20	The named parameter cannot be changed after initialization.
24	You specified an output-only field.
28	You specified an input-only field.
32	The new parameter value you specified is invalid.
36	An authorization check failed.
44	You specified an argument that cannot be used with the SET or GROUPS arguments.
56	No parameter information is available.
92	A master control block error occurred.
96	OPS/REXX cannot find the master control block.
100	An authorization exit abend failure occurred.
108	OPS/REXX cannot find the control block.
112	The master and local version codes do not match.
120	The system move data routine failed.
124	The service routine failed.
128	The cross-system is not active.
132	Message queue allocation failed.
136	The last message was never received.
140	Command output error.
144	Cross-system send error.
148	The action command was ignored, because the remote system that sent the command is not secure.

OPSPARM Command

You can issue the OPSPARM command:

- When initializing CA OPS/MVS, through the OPSTART1 CLIST or the OPSSPA00 member of the Logical Parmlib Concatenation that OPSTART1 invokes; this is the *only* way to set the parameters described elsewhere in this chapter as parameters to set at initialization
- Using option 6 of the ISPF main menu to display or change parameter values
- Anywhere a TSO command processor or OPS/REXX program can execute

Syntax of OPSPARM for Changing Parameters

To change parameters, use the following syntax:

OPSPARM
{SET(*parmname*)}
{VALUE(*parmvalue*)}
[SUBSYS(*ssid*)]
[SYSTEM(*sysname*|ALL|EXT)]
[SYSWAIT(*seconds*)]

SET and VALUE

Always specify the SET and VALUE keywords together to identify the name of the parameter to be changed and the new value for the parameter. For example:

```
OPSPARM SET(OSFMAX) VALUE(3)
OPSPARM SET(SYSID) VALUE(PRODSYS)
```

If you specify an invalid *parmname*, the OPSPARM command processor issues an error message and a return code of 60.

SUBSYS

(Optional) When you reset a parameter, the SUBSYS keyword identifies the specific copy of CA OPS/MVS whose parameter this OPSPARM command will change. You must use the SUBSYS keyword when resetting parameters, except in these situations:

When the OPSPARM request is addressed to the copy of CA OPS/MVS using the default z/OS subsystem identifier (usually OPSS)

When the OPSPARM request is issued by a CLIST running in an address space belonging to a specific copy of CA OPS/MVS.

Note: This address space can be the OPSMAIN address space itself, one of its OPSOSF address spaces, or an OPSECF address space it started.

SYSTEM Keyword (Used with SET)

(Optional) The SYSTEM keyword provides CA OPS/MVS with the capability to execute the OPSPARM command on external systems that have been defined to the MSF. The MSF establishes VTAM sessions between copies of CA OPS/MVS, permitting any copy to issue a command on any other copy and to receive its response.

When used with the SET keyword, the SYSTEM keyword indicates one of the following:

- The name of the system on which the SET command is to execute; to indicate this, specify a value for *sysname*
- The SET command should execute on all active systems other than the local system; to indicate this, specify EXT
- The SET command should execute on all active systems; to indicate this, specify ALL

Note: As referred to here, *active* is the status of the MSF systems through which the communication takes place.

SYSWAIT Keyword (Used with SET)

(Optional) Use the SYSWAIT keyword to specify the number of seconds that CA OPS/MVS waits for a response from a remote system. You can specify a value from 1 to 300; the default is the MSFSYSWAIT value.

Examples: SYSTEM Keyword Used with SET

1. In this OPSPARM command, the SYSTEM keyword specifies that the OCWAIT parameter is to be set to 7 on system SYSA:

   ```
   OPSPARM SET(OCWAIT) VALUE(7) SYSTEM(SYSA)
   ```

2. In this example, the SYSTEM keyword specifies that the OCWAIT parameter is to be set to 9 on all systems except the local system:

   ```
   OPSPARM SET(OCWAIT) VALUE(9) SYSTEM(EXT)
   ```

3. This example indicates that the OCWAIT parameter is to be set to 8 on all systems including the local one:

   ```
   OPSPARM SET(OCWAIT) VALUE(8) SYSTEM(ALL)
   ```

Syntax of OPSPARM for Displaying Parameters

To display parameters, use this form of the OPSPARM TSO command:

```
OPSPARM
{SHOW(name|GROUPS|ALL)}
[INFO|NAMES|CLIST]
[SYSTEM(sysname)]
[SYSWAIT(seconds)]
```

SHOW

The SHOW keyword, used when you display parameter values, specifies which parameters to display. Specify one of these options:

name

Display the value of the named parameter or the values of all parameters in the named group.

GROUPS

Display a list of the parameter groups.

ALL

Display all parameter values.

INFO

Use the INFO keyword with the SHOW keyword to display the possible values the parameter can have. For example, to see all possible values of the ECFSECURITY parameter, issue this command:

```
OPSPARM SHOW(ECFSECURITY) INFO
```

NAMES

Use the NAMES keyword with the SHOW keyword to display the names and modifiability indicators of individual parameters. You cannot use NAMES with the SHOW(GROUPS) keyword.

To see the names of the parameters in the PRODACTIVITY group, issue this command:

```
OPSPARM SHOW(PRODACTIVITY) NAMES
```

CLIST

The CLIST keyword creates CLIST variables based on CA OPS/MVS parameters. These variables have the same names and corresponding values as the parameters.

The CLIST keyword is especially useful for obtaining the values of fields such as SYSID, which you can use in CLISTs for conditional processing based on the system where the CLIST runs.

SYSTEM Keyword (Used With SHOW)

The SYSTEM keyword provides CA OPS/MVS with the capability to execute the OPSPARM command on external systems.

When used with the SHOW keyword, the SYSTEM keyword indicates the name of the system on which the OPSPARM command is to execute. The response to the OPSPARM SHOW command will be returned only if the command is routed to an individual system.

The SYSTEM keyword in this command indicates that SYSA is the name of the system on which the command is to execute and from which the response is to be returned:

```
OPSPARM SHOW(OCWAIT) SYSTEM(SYSA)
```

SYSWAIT Keyword (Used With SHOW)

Use the SYSWAIT keyword to specify the number of seconds CA OPS/MVS waits for a response from a remote system. You can specify a value from 1 to 300; the default is the MSFSYSWAIT value.

Examples: SHOW Keyword

1. This example displays the value of the ECFSECURITY parameter:

   ```
   OPSPARM SHOW(ECFSECURITY)
   ```

2. This example displays the values of all parameters in the PRODACTIVITY parameter group:

   ```
   OPSPARM SHOW(PRODACTIVITY)
   ```

OPSPARM Command Return Codes

The OPSPARM command produces the following return codes and messages:

Return Code	Number	Message Text
0	OPS1250	IS THE ACTUAL VALUE OF THE PARAMETER ITSELF
4	n/a	CLIST KEYWORD USED OUTSIDE OF A CLIST
8	OPS1252	CANNOT BE USED BEFORE PRODUCT ACTIVE
12	OPS1253	SET/SHOW KEYWORDS MISSING
16	OPS1254	NO VALUE KEY ENTERED WITH SET
20	OPS1255	PARM CANNOT BE CHANGED AFTER INIT
24	OPS1256	OUTPUT ONLY FIELD
28	OPS1257	INPUT ONLY FIELD

Return Code	Number	Message Text
32	OPS1258	SET DATA VALUE IS INVALID
36	OPS1259	AUTHORIZATION CHECK FAILED
40	OPS1260	'ALL' OR GROUP NAME CANNOT BE USED WITH SET
44	OPS1261	KEYWORD CANNOT BE USED WITH SET/GROUPS
None for this message	OPS1262	USED INTERNALLY TO CREATE CLIST VARIABLES
56	n/a	NO PARAMETER INFORMATION AVAILABLE
60	n/a	INVALID PARAMETER NAME
80	OPS1270	TSO/E IS NOT INSTALLED
84	OPS1271	COMMAND BUFFER PARSE ERROR
88	OPS1272	COMMAND NOT ISSUED WITH AUTHORIZATION
92	OPS1273	MASTER CONTROL BLOCK ERROR
96	OPS1274	MASTER CONTROL BLOCK NOT FOUND
100	OPS1275	AUTHORIZATION EXIT ABEND FAILURE
104	OPS1276	SUBSYSTEM DOES NOT EXIST
108	OPS1277	CONTROL BLOCK NOT FOUND
112	n/a	MASTER/LOCAL VERSION CODE MISMATCH
116	OPS1279	CLIST VARIABLE UPDATE ERROR
None for this message	OPS1280	WRITE CURRENT PARM SET/SHOW REQUEST ONTO SYSLOG
120	n/a	SYSTEM MOVE DATA ROUTINE FAILED
124	n/a	SERVICE ROUTINE FAILED
128	n/a	CROSS SYSTEM NOT ACTIVE
132	n/a	MESSAGE QUEUE ALLOCATION FAILED
136	n/a	LAST MESSAGE NEVER RECEIVED
140	n/a	COMMAND OUTPUT ERROR
144	n/a	CROSS SYSTEM SEND ERROR
148	OPS3533	SECURE/NOSECURE ACTION COMMAND

Permanent Parameters

Permanent CA OPS/MVS parameters are stored in the CA OPS/MVS Permanent Product data area. The parameter values remain in effect between executions of CA OPS/MVS, but you can change permanent parameter values only while CA OPS/MVS is active.

To view or change these parameters, you choose the appropriate parameter group from OPSVIEW option 4.1.1. For details about option 4.1.1, see the *OPSVIEW User Guide*.

List of Permanent Parameters

The following table lists the permanent parameters for CA OPS/MVS:

Parameter Name	Belongs to Parameter Group
NMEPSRECID Parameter	Miscellaneous Operating Parameters group
OCCONINT Parameter	Programmable Operator Interface (POIPARMS) group
OCCONTIME Parameter	POIPARMS
OCINTERVAL Parameter	POIPARMS
OCMAXMSG Parameter	POIPARMS
OCWTORINT Parameter	POIPARMS
OCWTORTIME Parameter	POIPARMS
QUICKREFDBASE Parameter	Permanent Product (PRODPERM) group
REXXMAXCLAUSES (see OPS/REXX Parameters in the chapter “Parameters for Facilities”)	POIPARMS
REXXMAXCOMMANDS (see OPS/REXX Parameters in the chapter “Parameters for Facilities”)	POIPARMS
REXXMAXPGMSIZE (see OPS/REXX Parameters in the chapter “Parameters for Facilities”)	POIPARMS
REXXMAXQUEUE (see OPS/REXX Parameters in the chapter “Parameters for Facilities”)	POIPARMS
REXXMAXSAYS (see OPS/REXX Parameters in the chapter “Parameters for Facilities”)	POIPARMS
REXXMAXSECONDS (see OPS/REXX Parameters in the chapter “Parameters for Facilities”)	POIPARMS

Parameter Name	Belongs to Parameter Group
REXXMAXSTRINGLENGTH (see OPS/REXX Parameters in the chapter “Parameters for Facilities”)	POIPARMS
VIO Parameter	General System Information (SYSTEMINFO) group

OPSLOG Parameters

The parameters described in the following sections affect the CA OPS/MVS OPSLOG.

ALLOWNOOPSLOG Parameter

Overrides the value of the NOOPSLOG keyword operand in the)MSG statement of AOF message rules.

This parameter is useful for auditing purposes. Before assigning a value to ALLOWNOOPSLOG, decide whether you want to permit rule writers at your site to exclude certain messages from the OPSLOG.

Default value: YES

This value gives rule writers the ability to exclude messages from the OPSLOG.

Other possible values: NO

This value causes CA OPS/MVS to ignore the NOOPSLOG operand.

Set or modify this parameter...

Anytime

Example: ALLOWNOOPSLOG

This function overrides the NOOPSLOG keyword operand.

```
OPSPRM('SET','ALLOWNOOPSLOG','NO')
```

ARCHIVETRIGGER Parameter

This parameter sets the number of messages that CA OPS/MVS accumulates before issuing message OPS4403O, which you can use to trigger the archive job or restart a failed archive (through a message rule).

Note: Use this formula to determine the approximate storage capacity (in bytes) that is required to contain an archived OPSLOG:

The value of ARCHIVETRIGGER * 375

This value may be too low if most of the messages in the OPSLOG are long (close to 128 bytes in length).

Default value: 0

Note: If you use the default, the OPS4403O message never appears.

Other possible values:

Any number between 0 and 1000000000

This value should be smaller than (about half the size of) the value of the BROWSEMAX parameter so that the archive has time to complete before the OPSLOG wraps.

Set or modify this parameter...

Anytime

Example: ARCHIVETRIGGER

This function triggers the archive job after 50000 messages.

```
OPSPRM('SET','ARCHIVETRIGGER','50000')
```

BROWSEACTIVEMAX Parameter

Limits the number of concurrent active OPSLOGs.

Default value: 0

The value of 0 indicates no limit has been set. All defined OPSLOGs can be active concurrently.

Other possible values:

Any number from 1 and 32.

Set or modify this parameter...

Anytime

The new value applies to future activation attempts. If set to a value less than the existing active count, all active OPSLOGs will continue to run but activation attempts will fail until enough OPSLOGs become inactive.

Example: BROWSEACTIVEMAX

This function sets a limit of 12 concurrent active OPSLOGs.

```
OPSPRM('SET','BROWSEACTIVEMAX ','12')
```

BROWSEAPI Parameter

Specifies whether OPSLOG includes messages reporting API events.

Default value: YES

Other possible values: NO

Set or modify this parameter...

Anytime

Example: BROWSEAPI

This function includes API events in the OPSLOG.

```
OPSPRM('SET','BROWSEAPI','YES')
```

BROWSEARCHIVEDSN Parameter

Specifies a default name for the OPSLOG archive data set.

Default value

No default

Other possible values

Any valid data set name

Set or modify this parameter...

Anytime

Example: BROWSEARCHIVEDSN

This function names the OPSLOG archive OPSLOG.ARCHV.

```
OPSPRM('SET','BROWSEARCHIVEDSN','OPSLOG.ARCHV')
```

BROWSEARCHIVEUNIT Parameter

Specifies a default unit type for the OPSLOG archive.

Default value

No default

Other possible values

Any valid unit name

Set or modify this parameter...

Anytime

Example: BROWSEARCHIVEUNIT

This function names SYSDA as the default unit type.

```
OPSPRM('SET','BROWSEARCHIVEUNIT','SYSDA')
```

BROWSECA7 Parameter

Allows the OPSLOG to include message events created by the CA7 WA log component. Only messages where OPSINFO('EXITTYPE') = 'CA7' are affected.

Default value

YES

Other possible values

NO

Note: Selected CA7 messages can still be included in OPSLOG using the OPSSEND('*','B') Rexx function in an AOF rule.

Set or modify this parameter

Anytime

Example: BROWSECA7

This function excludes CA7 messages from the OPSLOG.

```
OPSPRM('SET','BROWSECA7','NO')
```

BROWSECOF Parameter

Allows the OPSLOG to include message events created by the COF facility. Only messages where OPSINFO('EXITTYPE') = 'CICS' are affected.

Default value

YES

Other possible values

NO

Note: Selected COF messages can still be included in OPSLOG using the OPSSEND('*','B') Rexx function in an AOF rule.

Set or modify this parameter

Anytime

Example: BROWSECOF

This function excludes COF messages from the OPSLOG.

```
OPSPRM('SET','BROWSECOF','NO')
```

BROWSECMD Parameter

Allows the OPSLOG to include messages reporting command events.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: BROWSECMD

This function excludes command events from OPSLOG.

```
OPSPRM('SET','BROWSECMD','NO')
```

BROWSEDIS Parameter

Allows the OPSLOG to include messages reporting disable events.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEDIS

This function includes disable messages in the OPSLOG.

```
OPSPRM('SET','BROWSEDIS','YES')
```

BROWSEDIV Parameter

Specifies whether a data-in-virtual data set backs up OPSLOG on disk.

Default value

YES

Other possible values

NO

This value makes the OPSLOG memory-resident only, so that it is not retained after CA OPS/MVS terminates.

Set or modify this parameter...

At initialization

Example: BROWSEDIV

This function sets the data-in-virtual flag off.

```
OPSPRM('SET','BROWSEDIV','NO')
```

BROWSEDOM Parameter

Specifies whether OPSLOG includes Delete Operator Message (DOM) events.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEDOM

This function includes DOM events in the OPSLOG.

```
OPSPRM('SET','BROWSEDOM','YES')
```

BROWSEENA Parameter

Specifies whether OPSLOG includes messages reporting enable events.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEENA

This function includes enable messages in the OPSLOG.

```
OPSPRM('SET','BROWSEENA','YES')
```

BROWSEEOJ Parameter

Specifies whether OPSLOG includes messages reporting an end-of-job (EOJ) event.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEEOJ

This function includes EOJ events in the OPSLOG.

```
OPSPRM('SET','BROWSEEOJ','YES')
```

BROWSEEOM Parameter

Specifies whether OPSLOG includes messages reporting end-of-memory (EOM) events.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEEOM

This function includes EOM events in the OPSLOG.

```
OPSPRM('SET','BROWSEEOM','YES')
```

BROWSEEOS Parameter

Specifies whether OPSLOG includes messages reporting an end-of-step (EOS) event.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEEOS

This function includes EOS events in the OPSLOG.

```
OPSPRM('SET','BROWSEEOS','YES')
```

BROWSEFINDLIM Parameter

This parameter determines how many lines the FIND command will search by default in OPSLOG Browse. The value you specify remains in effect across CA OPS/MVS sessions.

Default value

5000

Note: If the parameter display value is 0, then the default limit of 5000 is used.

Other possible values

Any number greater than 1; however, since you cannot FIND more lines than the number of lines existing in OPSLOG, the practical maximum value is the value of BROWSEMAX

Set or modify this parameter...

Anytime

Example: BROWSEFINDLIM

This function allows you to search up to 500 lines on subsequent FIND commands.

```
OPSPRM('SET','BROWSEFINDLIM','500')
```

BROWSEGLV Parameter

Specifies whether OPSLOG includes messages reporting global variable events.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEGLV

This function includes global variable event messages in the OPSLOG.

```
OPSPRM('SET','BROWSEGLV','YES')
```

BROWSEIDFORMAT Parameter

Determines how CA OPS/MVS displays the two-character ID of the system (SYSID column in OPSLOG Browse) that generated the events. You specify which two characters are extracted from the internal system name field and displayed in the OPSLOG Browse SYSID column.

The BROWSEIDFORMAT parameter and the SYSID column in OPSLOG Browse are primarily intended for use by JES3 sites, but in some cases may also be useful to JES2 sites. For additional information about the columns displayed in OPSLOG Browse, see the *OPSVIEW User Guide*.

Default value

The first two characters of the JES3 system ID, in hexadecimal format: X'0001'

Other possible values

Any other two bytes in the system ID character string, expressed in hexadecimals. Each hexadecimal value in this string must be in the range 00 to 07 for JES3 sites, and in the range 00 to 03 for JES2 sites.

Note: These values are zero-offset based. The first character in the string is represented by hexadecimal value 00, the second character by 01, the third by 02, and so on.

Set or modify this parameter...

Anytime

Examples: BROWSEIDFORMAT

This function displays the second and third characters of the system ID.

```
OPSPRM("SET","BROWSEIDFORMAT","X'0102'")
```

This function displays the third and sixth characters of the system ID.

```
OPSPRM("SET","BROWSEIDFORMAT","X'0205'")
```

BROWSEINTERVAL Parameter

Determines how often the OPSLOG message area is saved to the OPSLOG DIV data set.

Default value

15 (seconds)

Other possible values

Any number of seconds between 1 and 300

Set or modify this parameter...

Anytime

Example: BROWSEINTERVAL

This function saves OPSLOG contents every 30 seconds.

```
OPSPRM('SET','BROWSEINTERVAL','30')
```

BROWSELXC Parameter

The BROWSELXC parameter must be set to YES in order to include the Linux Connector component unsolicited message events passed to CA OPS/MVS in OPSLOG.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSELXC

This function ensures that Linux Connector unsolicited message events are included in OPSLOG.

```
OPSPRM('SET','BROWSELXC','YES')
```

BROWSEMAX Parameter

Determines the maximum number of messages in the OPSLOG message area for any OPSLOG. The OPSLOG message area resides in a data space owned by the CA OPS/MVS main address space. This is only the default value and can be overridden on any ADDRESS OPSCTL “OPSLOG DEFINE” or “OPSLOG ACTIVATE” host command for any OPSLOG.

Default value

400000

Other possible values

Any number greater than or equal to 1000 but less than or equal to 4925000

Set or modify this parameter...

At initialization.

However, you cannot change the BROWSEMAX value if you initialize CA OPS/MVS with an existing OPSLOG; attempting to do so generates message OPS0163W. To change the BROWSEMAX value in this case, you must:

1. Stop CA OPS/MVS.
2. Delete the OPSLOG DIV data set.
3. Allocate a new OPSLOG data set.
4. Restart CA OPS/MVS, using a different BROWSEMAX value.

Example: BROWSEMAX

This function limits the number of messages to 100000.

```
OPSPRM('SET','BROWSEMAX','100000')
```

BROWSEMESSAGES Parameter

This parameter determines which copies of a message and a DOM go into OPSLOG.

Note: In a sysplex environment, the value of BROWSEMESSAGES affects messages and DOMs from other systems in the sysplex.

Default value

MVS

CA OPS/MVS processes z/OS messages and DOMs from the SSI.

In a sysplex environment, OPSLOG receives and processes messages and DOMs from this specific system only.

Other possible values

- NONE

  OPSLOG processes no messages or DOMs.

- MVSGLOBAL

  This value is valid only for sites that are running in a sysplex environment or importing messages to CA OPS/MVS through CA MIC.

  If your site runs JES3, you must specify MVSGLOBAL on the JES3 global processor in order for OPSLOG to receive the reissued messages from the JES3 local processors. In a sysplex environment, specifying MVSGLOBAL causes OPSLOG to receive and process messages from other systems in the sysplex.

  In an environment in which CA OPS/MVS receives imported messages from CA MIC, specifying MVSGLOBAL causes OPSLOG to receive and process the imported messages.

  Note: If AOFMESSAGES is set to MVSLIST and BROWSEMESSAGES is set to MVSGLOBAL, OPSLOG will receive local messages and messages imported from systems specified on the MVSSYS*n* parameters. Consult the documentation for the AOFMESSAGES parameter for more information.

Set or modify this parameter...

Anytime

Example: BROWSEMESSAGES

This function allows messages and DOMs from other systems in the sysplex to be recorded in OPSLOG.

```
OPSPRM('SET','BROWSEMESSAGES','MVSGLOBAL')
```

More information

AOFMESSAGES Parameter (see page 142)

Usage Recommendations

CA recommends the following regarding the use of the BROWSEMESSAGES parameter:

- JES2 sites should use a value of MVS.
- JES3 sites should use a value of MVSGLOBAL on the JES3 global processor and a value of MVS on the JES3 local processors.

Important!

- Do not change the value of the BROWSEMESSAGES parameter without first considering the impact that the change will have on your ability to determine what is happening on your system.
- If you use the MVSGLOBAL value in a sysplex environment, changes to the MSCOPE of your consoles may affect which messages are logged in OPSLOG and processed by the AOF.

Note: To avoid confusion between what appears in the OPSLOG and which events are processed by the AOF, give the BROWSEMESSAGES and AOFMESSAGES parameters identical values. However, the MVSLIST option is not available on the BROWSEMESSAGES parameter. If you specify MVSLIST for AOFMESSAGES, it is recommended that you specify MVSGLOBAL for BROWSEMESSAGES. This will cause OPSLOG to show local messages and messages imported from the systems specified on the MVSSYS*n* parameters.

BROWSEOMG Parameter

This parameter determines whether OMEGAMON event messages appear in the OPSLOG.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEOMG

This function excludes OMEGAMON messages from OPSLOG.

```
OPSPRM('SET','BROWSEOMG','NO')
```

BROWSEPRINTLIM Parameter

This parameter sets the maximum number of lines you can print using the PP line command in OPSLOG Browse. The value you specify remains in effect across CA OPS/MVS sessions.

Default value

5000

Note: If the parameter display value is 0, then the default limit of 5000 is used.

Other possible values

Any number greater than 1; however, since you cannot PRINT more lines than the number of lines existing in the OPSLOG, the practical maximum value is the value of BROWSEMAX

Set or modify this parameter...

Anytime

Example: BROWSEPRINTLIM

This function allows you to print up to 100,000 lines.

```
OPSPRM('SET','BROWSEPRINTLIM','100000')
```

BROWSEPROFPROMPT Parameter

This parameter determines whether CA OPS/MVS prompts you to set a profile filter before it gives you access to OPSLOG Browse.

If you allow the value of the BROWSEPROFPROMPT parameter to default to NO, OPSLOG Browse viewing is not affected. Setting the value of the BROWSEPROFPROMPT parameter to YES affects OPSLOG Browse viewing *only* when both of these statements are true:

- You set a profile filter for OPSLOG Browse during a previous OPSLOG Browse session.
- When you set your profile filter, you entered or changed values for the JOBNAME, MSGID, RULESET, COLOR, SYSNAME, USER, or MSGevent options.

Default value

NO

Other possible values

YES

Set or modify this parameter…

Anytime

Example: BROWSEPROFPROMPT

This function causes the OPSLOG Browse Profile panel to appear when you select OPSLOG Browse, giving you a chance to change the filtering before you actually enter OPSLOG Browse.

```
OPSPRM('SET','BROWSEPROFPROMPT','YES')
```

BROWSEREQ Parameter

This parameter determines whether request event messages appear in the OPSLOG.

Default value

NO

Other possible values

YES

Set or modify this parameter…

Anytime

Example: BROWSEREQ

This function includes request events in the OPSLOG.

```
OPSPRM('SET','BROWSEREQ','YES')
```

BROWSESCR Parameter

Determines whether screen event messages appear in the OPSLOG.

Default value

NO

Other possible values

YES

Set or modify this parameter…

Anytime

Example: BROWSESCR

This function includes screen event messages.

```
OPSPRM('SET','BROWSESCR','YES')
```

BROWSESEC Parameter

Determines whether security event messages appear in the OPSLOG.

Default value

NO

Other possible values

YES

Set or modify this parameter…

Anytime

Example: BROWSESEC

This function includes security events in the OPSLOG,

```
OPSPRM('SET','BROWSESEC','YES')
```

BROWSETLM Parameter

Specifies whether OPSLOG includes messages reporting time limit excession (TLM) events.

Default value

NO

Other possible values

YES

Set or modify this parameter…

Anytime

Example: BROWSETLM

This function includes TLM events in OPSLOG.

```
OPSPRM('SET','BROWSETLM','YES')
```

BROWSETOD Parameter

Determines whether the OPSLOG includes time-of-day (TOD) event messages.

Default value

NO

Other possible values

YES

Set or modify this parameter…

Anytime

Example: BROWSETOD

This function includes time-of-day messages.

```
OPSPRM('SET','BROWSETOD','YES')
```

BROWSEUSS Parameter

Specifies whether OPSLOG includes messages reporting a USS message event.

Default value

NO

Other possible values

YES

Set or modify this parameter…

Anytime

Example: BROWSEUSS

This function includes USS message events in OPSLOG.

```
OPSPRM('SET','BROWSEUSS','YES')
```

BROWSEUSSPROC Parameter

The BROWSEUSSPROC parameter must be set to YES in order to include the USS process event messages in OPSLOG.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: BROWSEUSSPROC

This function ensures that USS process message events are included in OPSLOG.

```
OPSPRM('SET','BROWSEUSSPROC','YES')
```

MVSSYSn Parameter

There are thirty one MVSSYS*n* parameters named MVSSYS1 to MVSSYS31. Each MVSSYSn parameter can specify an eight-character system name. When the AOF MESSAGES parameter is set to MVSLIST, CA OPS/MVS will receive and process local messages and messages from systems specified on the MVSSYS*n* parameters. See the AOFMESSAGES parameter for more information.

Default value: ' '

The default value is a blank which indicates that no system name is specified on this parameter.

Other possible values

8-character system name

This value indicates that CA OPS/MVS should receive and process messages imported from the specified system if the AOFMESSAGES parameter is set to MVSLIST.

Set or modify this parameter...

Anytime

Example: MVSSYS*n* Parameter

These functions tell CA OPS/MVS to receive and process messages imported from MVSSYSAA and MVSSYSBB if AOFMESSAGES is set to MVSLIST:

```
OPSPRM('SET','MVSSYS1','MVSSYSAA')

OPSPRM('SET','MVSSYS2','MVSSYSBB')
```

Usage Recommendations

CA recommends the following regarding the use of the MVSSYS*n* parameters:

- Do not 'skip' MVSSYS*n* parameters when filling in system names. For example, if you are specifying three system names, use the MVSSYS1, MVSSYS2, and MVSSYS3 parameters. While the function will still work correctly if you skip parameters (use MVSSYS1, MVSSYS10, and MVSSYS20 for example), the best performance will be achieved by filling the MVSSYS*n* parameters consecutively.
- Use the AOFMESSAGES parameter to toggle system filtering on and off. The system names specified on the MVSSYS*n* parameters persist regardless of the AOFMESSAGES value. However, the system names are not used to filter message processing unless AOFMESSAGES is set to MVSLIST. The MVSSYS*n* parameters are ignored for any other AOFMESSAGE value.

Important! If you specify a value of MVSLIST for AOFMESSAGES and do not specify any system names on MVSSYS*n* parameters, CA OPS/MVS will not receive and process imported messages. Only local messages will be processed by CA OPS/MVS.

OMGCICS Parameter

Sets the default display color of OMEGAMON CICS messages in OPSLOG browse.

Default value

NONE (no color)

Other possible values

BLUE, GREEN, PINK, RED, TURQ, WHITE, or YELLOW

Set or modify this parameter…

Anytime

Example: OMGCICS

This function causes OMEGAMON CICS messages to display in pink.

```
OPSPRM('SET','OMGCICS','PINK')
```

OMGDB2 Parameter

Sets the default display color of OMEGAMON DB2 messages in OPSLOG browse.

Default value

NONE (no color)

Other possible values

BLUE, GREEN, PINK, RED, TURQ, WHITE, or YELLOW

Set or modify this parameter…

Anytime

Example: OMGDB2

This function causes OMEGAMON DB2 messages to display in red.

```
OPSPRM('SET','OMGDB2','RED')
```

OMGIMS Parameter

Sets the default display color of OMEGAMON IMS messages in OPSLOG browse.

Default value

NONE (no color)

Other possible values

BLUE, GREEN, PINK, RED, TURQ, WHITE, or YELLOW

Set or modify this parameter…

Anytime

Example: OMGIMS

This function causes OMEGAMON IMS messages to display in white.

```
OPSPRM('SET','OMGIMS','WHITE')
```

OMGMVS Parameter

Sets the default display color of OMEGAMON MVS messages in OPSLOG browse.

Default value

NONE (no color)

Other possible values

BLUE, GREEN, PINK, RED, TURQ, WHITE, or YELLOW

Set or modify this parameter…

Anytime

Example: OMGMVS

This function causes OMEGAMON MVS messages to display in blue.

```
OPSPRM('SET','OMGMVS','BLUE')
```

Command-related Parameters

The parameters described in the following sections affect CA OPS/MVS command processing.

BYPASSCMDECHO Parameter

This parameter determines whether message rules execute when operators issue commands.

Default value

NO

Using this setting, an operator command executes a message rule.

Other possible values

YES

Using this setting, operator commands do not execute message rules.

Set or modify this parameter...

Anytime

Example: BYPASSCMDECHO

This function prevents AOF message rules from executing as a result of an operator command.

```
OPSPRM('SET','BYPASSCMDECHO','YES')
```

CMDSECREPLYTEXT Parameter

This parameter dictates whether command rules can see the text from replies to security WTORs.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: CMDSECREPLYTEXT

This function prevents command rules from seeing WTOR reply text.

```
OPSPRM('SET','CMDSECREPLYTEXT','NO')
```

COMMANDHIGH Parameter

This parameter displays the high watermark that the command rate control mechanism has reached during the current life of the product or since this parameter value was last reset. This is the highest value that the COMMANDCURRENT parameter has ever reached since that time. Compare the COMMANDHIGH value to the value specified on the COMMANDMAX parameter to determine how close CA OPS/MVS has come to the point at which it would shut itself down due to exceeding the COMMANDMAX limit.

Note that resetting this parameter affects the data recorded in the product SMF records and the reports produced by the Automation Measurement Environment (AME).

Default value

No default

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: COMMANDHIGH

This function resets the COMMANDHIGH value to zero.

```
OPSPRM('SET','COMMANDHIGH',0)
```

COMMANDMAX Parameter

This parameter dictates the maximum number of commands CA OPS/MVS can issue per second. When this number equals the value of the COMMANDCURRENT counter, CA OPS/MVS terminates and issues message OPS3146S. Setting the COMMANDMAX parameter to its maximum possible value prevents CA OPS/MVS from ever shutting down due to an excessive command rate.

Default value

200

Other possible values

Any number between 100 and 10000

Set or modify this parameter...

Anytime

Example: COMMANDMAX

This function limits CA OPS/MVS to issuing 2000 commands.

```
OPSPRM('SET','COMMANDMAX','2000')
```

COMMANDRATE Parameter

This parameter sets the rate by which CA OPS/MVS decreases the COMMANDCURRENT counter in every elapsed second.

Default value

3

Other possible values

Between 1 and 100 per second

Set or modify this parameter...

Anytime

Example: COMMANDRATE

This function decreases the COMMANDCURRENT counter by 1 per second.

```
OPSPRM('SET','COMMANDRATE','1')
```

EXTCONSPREFIX Parameter

This parameter specifies a one- to six-character prefix for CA OPS/MVS to use when it generates extended console names. The prefix must begin with an uppercase alphabetic character or one of the following special characters: #, $, or @. The second through last characters of the prefix must be alphanumeric characters or any of these special characters: #, $, or @.

As many as 99 extended consoles may be active; the setting of the EXTENDEDCONSOLES parameter determines the number of extended consoles CA OPS/MVS activates at startup. To create a unique name for each extended console, CA OPS/MVS adds a two-digit suffix (01-99) to the prefix. For example, if the value of EXTCONSPREFIX is OPSEXT and the value of EXTENDEDCONSOLES is 3, CA OPS/MVS activates these consoles:

- OPSEXT01
- OPSEXT02
- OPSEXT03

When you specify a value for the EXTCONSPREFIX parameter, make sure that the value does not conflict with the values of the EXTRAEXTPREFIX parameter. If CA OPS/MVS detects a conflict, it reverts to its default prefix values.

Default value

CA OPS/MVS uses this pattern to generate a default prefix:

- Character 1 is E, to indicate extended.
- The next 1 to 4 characters (characters 2 through 5) are the 1- to 4-byte SMFID of the current system. The length of this string depends on the length of the SMFID.

  Because some SMFIDs contain characters that are not allowed in extended console names, in certain cases you will not be able to use the default prefix; instead, specify a value for the EXTCONSPREFIX parameter.
- The next character is the fourth character of the subsystem ID.
- The last two characters are a number from 01 through 99, depending upon the number of consoles requested.

Other possible values

Any valid one- to six-character prefix having these characteristics:

- The prefix must begin with an uppercase alphabetic character or any of these special characters: #, $, or @.
- The second through last characters must be alphanumeric characters or any of these special characters: #, $, or @.

Set or modify this parameter

At initialization

Example: EXTCONSPREFIX

This function sets the extended console prefix to EXTCON.

```
OPSPRM('SET','EXTCONSPREFIX','EXTCON')
```

Note: For more information about the CA OPS/MVS support for extended consoles, see the *Administration Guide*.

EXTENDEDCONSOLES Parameter

This parameter specifies the number of extended consoles (without MIGIDs) that CA OPS/MVS activates at startup. The recommended value for EXTENDEDCONSOLES is five more than the maximum number of all OSF servers (five plus the sum of OSFMAX, OSFTSLMAX, and OSFTSPMAX).

Important: Allocating an excessive number of extended, migration, and extra extended consoles may impact the amount of time it takes to process messages.

Note: For more information about MIGIDs and the CA OPS/MVS support for extended consoles, see the *Administration Guide*.

Default value

15

This default value of 15 is equal to five plus the sum of the defaults for OSFMAX, OSFTSLMAX, and OSFTSPMAX.

Other possible values

Any number from 0 to 99

Set or modify this parameter…

At initialization

Example: EXTENDEDCONSOLES

This function causes CA OPS/MVS to activate 22 extended consoles at startup.

```
OPSPRM('SET','EXTENDEDCONSOLES','22')
```

EXTRAEXTCONSOLES Parameter

This parameter specifies the number of extra extended consoles that CA OPS/MVS activates at startup. The EXTRAEXTCONSOLES and EXTRAEXTPREFIX parameters enable you to allocate a set of extended consoles for the exclusive use of particular pieces of CA OPS/MVSS. For example, perhaps your site uses an application that issues CICS commands, and all commands must be issued through the same console.

It is the responsibility of the client to manage these consoles in the appropriate manner. The OPSCMD command processor selects an extra extended console only if the console name is explicitly specified through the OPSCMD NAME or CONNAME keyword.

Important! Allocating an excessive number of extended, migration, and extra extended consoles may impact the amount of time it takes to process messages.

For more information about the CA OPS/MVS support for extended consoles, see the *Administration Guide*.

Default value

0

Other possible values

Any number from 0 to 99

Set or modify this parameter…

At initialization

Example: EXTRAEXTCONSOLES

This function causes CA OPS/MVS to activate 12 extra extended consoles at startup.

```
OPSPRM('SET','EXTRAEXTCONSOLES','12')
```

EXTRAEXTPREFIX Parameter

Specifies a one-to six-character prefix for CA OPS/MVS to use when it generates extra extended console names. The prefix must begin with an uppercase alphabetic character or one of the following special characters: #, $, or @. The second through last characters of the prefix must be alphanumeric characters or any of these special characters: #, $, or @.

Along with the EXTRAEXTCONSOLES parameter, the EXTRAEXTPREFIX parameter enables you to allocate a set of extended consoles for the exclusive use of particular pieces of CA OPS/MVS. For example, perhaps your site uses an application that issues CICS commands, and all commands must be issued through the same console.

As many as 99 extended consoles may be active; the setting of the EXTRAEXTCONSOLES parameter determines the number of extra extended consoles CA OPS/MVS activates at startup. To create a unique name for each extra extended console, CA OPS/MVS adds a two-digit suffix (01-99) to the prefix. For example, if the value of EXTRAEXTPREFIX is EXTEXT and the value of EXTRAEXTCONSOLES is 3, CA OPS/MVS activates these consoles:

- EXTEXT01
- EXTEXT02
- EXTEXT03

When you specify a value for the EXTRAEXTPREFIX parameter, make sure that the value does not conflict with the values of the EXTCONSPREFIX parameter. If CA OPS/MVS detects a conflict, it reverts to its default prefix values.

Default value

CA OPS/MVS uses this pattern to generate a default prefix:

- Character 1 is X, to indicate extra.
- Characters 2-5 are the 4-byte SMFID of the current system. Because some SMFIDs contain characters that are not allowed in extended console names, in certain cases you will not be able to use the default prefix; instead, specify a value for the EXTRAEXTPREFIX parameter.
- Character 6 is the fourth character of the subsystem ID.
- Characters 7-8 are a number from 01 to 99, depending upon the number of consoles requested.

Other possible values

Any valid one- to six-character prefix having these characteristics:

- The prefix must begin with an uppercase alphabetic character or any of these special characters: #, $, or @.
- The second through last characters must be alphanumeric characters or any of these special characters: #, $, or @.

Set or modify this parameter…

At initialization

Example: EXTRAEXTPREFIX

This function sets the extra extended console prefix to EXTRAC.

```
OPSPRM('SET','EXTRAEXTPREFIX','EXTRAC')
```

OCCONINT Parameter

Specifies how long CA OPS/MVS waits before retrying to allocate a console that it will use to issue operating system commands (through the OPSCMD command processor or ADDRESS OPER host command environment). This process occurs only when all consoles are busy.

Default value

25 (centiseconds)

Other possible values

Any value between 1 and 100 centiseconds

Set or modify this parameter…

Anytime

Example: OCCONINT

This function sets the console retry interval to 50 centiseconds.

```
OPSPRM('SET','OCCONINT','50')
```

OCCONSOLENAME Parameter

Specifies the console name that the CA OPS/MVS OPSCMD command processor and ADDRESS OPER commands use to issue commands when the NOOUTPUT keyword is either implied or explicitly specified.

We strongly recommend that you specify one of the following values for this parameter:

- A sysplex-unique value for the OCCONSOLENAME parameter. An extended console by that name will be allocated automatically during product initialization.
- The name of one of the extended consoles allocated in product initialization through the EXTRAEXTPREFIX and EXTRAEXTCONSOLES parameters.

Default value

A null string

Other possible values

Any valid 2- to 8-character console name; for information about valid console names, see the definition of CONSOL*xx* in the IBM documentation.

Set or modify this parameter…

Anytime

Example: OCCONSOLENAME

This function sets the value of OCCONSOLENAME to ZOSCD20.

```
OPSPRM('SET','OCCONSOLENAME','ZOSCD20')
```

OCCONTIME Parameter

Specifies how long CA OPS/MVS retries allocating a console that it will use to issue operating system commands (through the OPSCMD command processor or ADDRESS OPER host command environment). This process occurs only when all consoles are busy.

Default value

10 (seconds)

Other possible values

Any number of seconds between 1 and 60

Set or modify this parameter…

Anytime

Example: OCCONTIME

This function sets console retry time to 20 seconds.

```
OPSPRM('SET','OCCONTIME','20')
```

OCCONTYPE Parameter

Determines the default value that is used for the CONTYPE keyword of the OPSCMD command processor and the ADDRESS OPER host command environment. If the CONTYPE keyword is not specified for a cross-system command, then the value of OCCONTYPE on the target system is used.

Default value: ANY

Any console type may be used.

Other possible values

- EXTCONS

  Indicates that only extended consoles should be allocated.

- SSCONS

  Indicates that only subsystem consoles should be allocated.

Set or modify this parameter:

Anytime

Example: OCCONTYPE

This function uses only extended consoles for OPSCMD without the CONTYPE keyword specified.

```
OPSPRM('SET','OCCONTYPE','EXTCONS')
```

OCINTERVAL Parameter

Specifies how often, in hundredths of a second (centiseconds), CA OPS/MVS checks for output from an operator command issued through the OPSCMD command processor or the ADDRESS OPER host command environment.

For commands to the local system issued through OPSCMD:

- If a value for the WAIT keyword is specified, that value determines how long OPSCMD waits, *unconditionally*, to receive all output from the current command. In such cases, the value of OCINTERVAL is irrelevant.
- If the WAIT keyword is not specified, OPSCMD uses the value of the OCWAIT parameter to determine how long it should wait for a response. If it receives a response in the time specified by OCWAIT, CA OPS/MVS checks for additional output at the interval specified by OCINTERVAL.

For commands to the local system issued through ADDRESS OPER, the value of the OCWAIT parameter is always the maximum wait time. If a response is received in the time specified by OCWAIT, it checks for additional output at the interval specified by OCINTERVAL.

For cross-system commands, CA OPS/MVS looks for command output until the wait time (either the value specified by the SYSWAIT keyword on the command or the default MSFSYSWAIT value) expires. If a response is received in the specified time, CA OPS/MVS checks for additional output at the interval specified by OCINTERVAL.

Default value

90 (centiseconds)

Other possible values

Any number of centiseconds between 10 and 300

Set or modify this parameter…

Anytime

Example: OCINTERVAL

This function tells CA OPS/MVS to seek command output every half second.

```
OPSPRM('SET','OCINTERVAL','50')
```

OCMAXMSG Parameter

Specifies the maximum number of output lines a command issued in CA OPS/MVS can have. This value limits the output of commands issued through either the OPSCMD command or the ADDRESS OPER command.

Default value

2000 (lines of output)

Other possible values

Any number of lines between 100 and 32767

Set or modify this parameter…

Anytime

Example: OCMAXMSG

This function sets the maximum number of command output lines to 5000.

```
OPSPRM('SET','OCMAXMSG','5000')
```

OCWAIT Parameter

This parameter specifies how many seconds an OPSCMD command or ADDRESS OPER command will wait for command output. If used, the WAIT keyword of the OPSCMD command overrides the OCWAIT value.

Default value

10 (seconds)

Other possible values

Any number of seconds between 2 and 60

Set or modify this parameter…

Anytime

Example: OCWAIT

This function sets the maximum wait time to 10 seconds.

```
OPSPRM('SET','OCWAIT','10')
```

OCWTORINT Parameter

This parameter specifies, in centiseconds, how often CA OPS/MVS tries to find an outstanding WTOR message issued in response to any of the following:

- An OPSREPLY command
- An ADDRESS OPER host command for which the REPLY keyword or the IMSID keyword was specified
- An OPSCMD command processor for which the REPLY keyword or the IMSID keyword was specified

CA OPS/MVS continues searching for the WTOR until the OCWTORTIME interval expires.

Default value

25 (centiseconds)

Other possible values

Any number of centiseconds between 1 and 100

Set or modify this parameter…

Anytime

Example: OCWTORINT

This function tells CA OPS/MVS to look for a WTOR every 20 centiseconds.

```
OPSPRM('SET','OCWTORINT','20')
```

OCWTORTIME Parameter

This parameter limits the total number of seconds that CA OPS/MVS uses to locate outstanding WTOR messages issued in response to any of the following:

- An OPSREPLY command
- An OPSCMD command processor for which the REPLY keyword or the IMSID keyword was specified
- An ADDRESS OPER host command for which the REPLY keyword or the IMSID keyword was specified

For example, if the OCWTORINT parameter is set to 25 centiseconds and the OCWTORTIME parameter is set to 10 seconds, CA OPS/MVS looks for the reply ID a total of 40 times or four times per second (100 centiseconds / 25 centiseconds = 4 * 10 = 40).

Default value

10 (seconds)

Other possible values

Any number of seconds between 1 and 60

Set or modify this parameter…

Anytime

Example: OCWTORTIME

This function limits the total time CA OPS/MVS can use to locate a WTOR to 30 seconds.

```
OPSPRM('SET','OCWTORTIME','30')
```

OPSCMD Parameter

This parameter determines whether the OPSCMD command processor and the ADDRESS OPER host command environment are active.

Default value

YES (enables them)

Other possible values

NO (disables them)

Set or modify this parameter…

Anytime

Example: OPSCMD

This function disables the OPSCMD command processor and the ADDRESS OPER host command environment.

```
OPSPRM('SET','OPSCMD','NO')
```

QUICKREFCMD Parameter

This parameter provides OPSVIEW with the name of the MVS/QuickRef CLIST command invocation string if the installation does not use the default command of %QW as supplied by Chicago Soft, Ltd.

Default value

A string of 9 blanks. This tells OPSVIEW to use the default MVS/QuickRef command invocation string %QW.

Other possible values

Any 1 to 9-character TSO command string.

Note: If the first character is not a percent sign (which indicates that the command is a CLIST or REXX EXEC), then the string should contain a 1- to 8-character TSO command.

Set or modify this parameter…

Anytime

Example:

```
OPSPRM('SET','QUICKREFCMD','QWIKR')
```

SSICMD Parameter

This parameter determines whether CA OPS/MVS processes commands before or after subsystems such as JES, DB2, and NetView. CA recommends that you set the value of the SSICMD parameter to YES.

- When setting SSICMD do review this information.
- If the subsystem processes the command before CA OPS/MVS, then command text changes specified in a command rule are ignored.
- If CA OPS/MVS processes the command first, the command that the subsystem processes is the changed version. That is, it is the command as modified by the rule.
- If SSICMD is set to YES, CA OPS/MVS processes commands first.
- If SSICMD is set to NO, CA OPS/MVS processes commands after nearly all of the other subsystems have done so (depending on the order of the subsystems).
- To effectively intercept a command before any other subsystem processes it, perform:

Follow these steps:

1. Set the SSICMD parameter to YES.
2. Within the CMD rule logic, set the cmd.text environmental variable to null (cmd.text='') before exiting the rule with a RETURN 'ACCEPT.' For an example of intercepting and processing a JES2 command, see the AOF rule sample member JES2$TJ.

For a sample program that displays the names of all subsystems in use, see the SCANSSCT REXX program, which is distributed with CA OPS/MVS.

Default value

YES

This value causes CA OPS/MVS to process commands first. YES is the recommended setting.

Other possible values

NO

This value causes CA OPS/MVS to process commands after other subsystems.

Set or modify this parameter...

At initialization

Example: SSICMD

This function enables CA OPS/MVS command rules to process commands after other subsystems.

```
OPSPRM('SET','SSICMD','NO')
```

SYSPLEXSCOPE Parameter

This parameter determines whether CA OPS/MVS command processors thatuse the SYSTEM(ALL) or SYSTEM(EXT) parameters:

- Are delivered to all MSF connected systems or
- Just those that are in the same SYSPLEX as the local system.

Default value

NO

This value causes CA OPS/MVS to process the SYSTEM(ALL) and SYSTEM(EXT) keywords on command processors. This value always delivers the command to all MSF connected systems.

Other possible values

YES

This value causes CA OPS/MVS to limit the systems that receive commands through the SYSTEM(ALL) or SYSTEM(EXT) keywords to those systems which MSF is connected. These systems are in the same SYSPLEX as the local system.

Set or modify this parameter at the initialization and at any time.

Example: SYSPLEXSCOPE

This function enables CA OPS/MVS command rules to process commands after other subsystems.

```
OPSPRM('SET','SYSPLEXSCOPE','YES')
```

SUBSYSDEFAULT Parameter

The SUSYSDEFAULT parameter defines how many subsysstem consoles CA OPS/MVS will try to acquire. CA OPS/MVS uses subsystem consoles to submit commands and to retrieve command output. For more information, see the *Installation Guide*.

Default value

2 (consoles)

Other possible values

Any number of consoles between 0 and 99

Set or modify this parameter…

At initialization

Example: SUBSYSDEFAULT

This function allocates 20 subsystem consoles for CA OPS/MVS.

```
OPSPRM('SET','SUBSYSDEFAULT','20')
```

SYSTEMCPF Parameter

This parameter allows a site to add a sysplex-wide CPF prefix to the system for general-purpose use. This facility allows console operators and automation code to use the CPF prefix as a short form of the z/OS ROUTE command to issue commands to this system from any other system in the sysplex.

For example, if you set this parameter to S1 for system SYS1, your console operators can issue the abbreviated command:

```
S1 D A,OPSMAIN
```

instead of:

```
ROUTE S1 D A,OPSMAIN
```

from anywhere in the sysplex that includes SYS1. The CPF prefix is available on all systems in the sysplex. Therefore, you must use unique values on each system in a sysplex.

Once defined, the SYSTEMCPF prefix value remains available even if CA OPS/MVS is terminated. If you restart CA OPS/MVS with a different SYSTEMCPF parameter value, the old CPF prefix is deleted prior to defining the new one.

Note: The difference between using a CPF prefix and the ROUTE command-the ROUTE command causes a CMD event for the ROUTE command on the local system.

Default value

A string of 8 blanks. This blank value indicates that no system CPF value is set.

Other possible values

Any 1- to 8-character alphanumeric string.

Important! The SYSTEMCPF parameter can only contain characters from the table Characters That Can Be Used in z/OS Commands shown in the chapter “Using This Reference.”

Set or modify this parameter…

At initialization

Examples: SystemCPF

- This example sets the CPF prefix for this system to the value of the SMFID of the system:

  ```
  smfid = OPSINFO("SMFID")

  T = OPSPRM("SET","SystemCPF",smfid)
  ```

- You could also use the following, or another unique value in place of OPSINFO("SMFID"):

```
OPSINFO("SYSNAME")
```

Memory/Storage Parameters

The parameters described in the following sections affect the CA OPS/MVS use of CPU memory and storage.

CSALIMIT Parameter

Limits the amount of common service area (CSA) below the 16 MB line that CA OPS/MVS can acquire. Resetting this parameter to meet a need for more CSA in CA OPS/MVS does not guarantee that CA OPS/MVS will be able to acquire the additional storage; the available CSA storage of the operating system is still the main factor. For more information, see the description of the IEASYS*xx* member-CSA parameter in the IBM documentation.

Default value

15360 (bytes); 15 KB

Other possible values

Any amount of storage between 1024 and 2097152 bytes

Set or modify this parameter…

Anytime

Example: CSALIMIT

This function limits CA OPS/MVS to acquiring 15 KB of CSA storage.

```
OPSPRM('SET','CSALIMIT','15K')
```

ECSALIMIT Parameter

Limits the amount of extended common service area (ECSA) above the 16-megabyte line that CA OPS/MVS can acquire. Resetting this parameter to meet a need for more ECSA in CA OPS/MVS does not guarantee that CA OPS/MVS will be able to acquire the additional storage; the available ECSA storage of the operating system is still the main factor. For more information, see the description of the IEASYS*xx* member-ECSA parameter in the IBM documentation.

WARNING! If you set the value of the ECSALIMIT parameter too low, CA OPS/MVS may not be able to start or the module reload facility may not be able to function.

Default value

4096 KB

Other possible values

Any amount of storage between 256 KB and 16384 KB

Set or modify this parameter…

Anytime

Example: ECSALIMIT

This function limits CA OPS/MVS to acquiring 700 KB of ECSA storage.

```
OPSPRM('SET','ECSALIMIT','700K')
```

EPRIVLIMIT Parameter

Limits the amount of private storage area, above the 16-megabyte line, that CA OPS/MVS can acquire. Even when you increase this parameter value, the amount of storage that CA OPS/MVS can acquire depends on what the operating system allows.

Note: Expansion of the common storage area may affect user private storage.

Default value

Determined by CA OPS/MVS based on the maximum possible size of the extended private area

Other possible values

Any amount of storage between 1024 KB and 2 GB

Note: The value that you specify may be overridden if it is too large.

Set or modify this parameter...

Anytime

Example: EPRIVLIMIT

This function limits CA OPS/MVS to acquiring 1 GB of private storage above the 16 MB line.

```
OPSPRM('SET','EPRIVLIMIT','1048576K')
```

GENERICPOOLSIZE Parameter

Determines the size of the storage pool used by a number of basic and optional CA OPS/MVS facilities (for example, ADDRESS USS and ADDRESS NETMAN).

Default value

1024 KB

Other possible values

Any amount of storage between 524288 bytes (512 KB) and 67108864 (64 MB) bytes

Set or modify this parameter...

At initialization

Example: GENERICPOOLSIZE

This function limits the generic storage pool to 512 KB.

```
OPSPRM('SET','GENERICPOOLSIZE','512K')
```

MSFPOOLSIZE Parameter

This parameter determines the size of the storage pool used by MSF to contain data being passed between CA OPS/MVS systems through MSF.

Default value

1024 KB

Other possible values

Any amount of storage between 524288 bytes (512 KB) and 67108864 (64 MB) bytes

Set or modify this parameter…

Anytime

Note: After changing this parameter you must recycle the MSF component with a MODIFY OPSx,RESTART(MSF) command for the change to be effective.

Example: MSFPOOLSIZE

This function limits the MSF storage pool to 2 MB.

```
OPSPRM('SET','MSFPOOLSIZE','2048K')
```

PRIVLIMIT Parameter

This parameter limits the amount of private storage area, below the 16-megabyte line, that CA OPS/MVS can acquire. Even when you increase this parameter value, the amount of storage that CA OPS/MVS can acquire depends on the limits of the operating system (set by the REGION parameter on the EXEC JCL card, through the IEFUSI exit, or through the IEALIMIT routine).

Default value

12288 KB

Other possible values

Any amount of storage between 128 KB and 12288 KB (12 MB)

Set or modify this parameter

Anytime

Example: PRIVLIMIT

This function limits CA OPS/MVS to acquiring 256 KB of private storage below the 16 MB line.

```
OPSPRM ('SET','PRIVLIMIT','256K')
```

PROCESS Parameter

This parameter determines how many process blocks are allocated in the extended private area of the CA OPS/MVS main address space when the CA OPS/MVS address space initializes.

Note: The value of the SSEXEXITHICOUNT parameter indicates the high water mark for the number of these process blocks.

Allocating the right number of process blocks is critical. The number cannot be too low, because each event processed by CA OPS/MVS requires its own process block. However, setting the value too high has its own implications; the number of process blocks you specify may use so much virtual storage that CA OPS/MVS fails to function correctly.

Important! For these reasons, always discuss the PROCESS parameter with CA Customer Support before setting it to a value higher than 150.

Note: During initialization, CA OPS/MVS may increase the value of the PROCESS parameter if it is set too low for the current environment.

Default value

30 (blocks)

Other possible values

Any number of blocks between 10 and 350

Set or modify this parameter…

At initialization

Example: PROCESS

This function allocates 50 process blocks in the CA OPS/MVS main address space.

```
OPSPRM('SET','PROCESS','50')
```

SQLPOOLSIZE Parameter

This parameter determines the size of the storage pool used by the RDF to contain SQL data.

The size of this pool must be at least as large as the amount of data returned by the largest SQL request.

Default value

1024 KB

Other possible values

Any amount of storage between 524288 bytes (512 KB) and 67108864 (64 MB) bytes

Set or modify this parameter…

At initialization

Example: SQLPOOLSIZE

This function limits the MSF storage pool to 2 MB.

```
OPSPRM('SET','SQLPOOLSIZE','2048K')
```

STACKERROR Parameter

This parameter sets the size of the error stack space CA OPS/MVS uses.

Important! Do not modify this value except at the request of a CA Customer Support representative.

Default value

216 KB

Other possible values

Any amount of stack space

Set or modify this parameter…

At initialization

Example: STACKERROR

This function allocates 220 KB of error stack space.

```
OPSPRM('SET','STACKERROR','220K')
```

STACKMAIN Parameter

This parameter sets the size of the primary stack space CA OPS/MVS uses.

Important! Do not modify this value except at the request of a CA Customer Support representative.

Default value

512 KB

Other possible values

Any amount of stack space between 100 KB and 1024 KB

Set or modify this parameter…

At initialization

Example: STACKMAIN

This function allocates 640 KB of primary stack space.

```
OPSPRM('SET','STACKMAIN','640K')
```

STACKRESERVED Parameter

This parameter sets the size of the reserve stack space CA OPS/MVS uses.

Important! Do not modify this value except at the request of a CA Customer Support representative.

Default value

256 bytes

Other possible values

Any amount of stack space, specified in units of 1 KB (the default, even though it is specified in bytes, is valid and appears as 0 KB)

Set or modify this parameter…

At initialization

Example: STACKRESERVED

This function allocates 1 KB of reserve stack space.

```
OPSPRM('SET','STACKRESERVED','1')
```

Message-related Parameters

The parameters described in the following sections affect the CA OPS/MVS message processing.

MESSAGEHIGH Parameter

Displays the high watermark that the message rate control mechanism has reached during the current life of the product or since this parameter value was last reset. This is the highest value that the MESSAGECURRENT parameter has ever reached since that time. Compare the MESSAGEHIGH value to the value specified on the MESSAGEMAX parameter to determine how close CA OPS/MVS has come to the point at which it would shut itself down due to exceeding the MESSAGEMAX limit.

Note: Resetting this parameter affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default value

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter…

Anytime

Example: MESSAGEHIGH

This function resets the MESSAGEHIGH value to zero.

```
OPSPRM('SET','MESSAGEHIGH',0)
```

MESSAGEMAX Parameter

This parameter limits the number of write-to-operator messages that CA OPS/MVS can generate in a given second.

Note: By default, SAY and TRACE statements from rules or from REXX programs running in a server are written to OPSLOG only; they are not WTOed.

The MESSAGEMAX parameter dictates the maximum value of the message counter kept by the MESSAGECURRENT parameter. When the value of MESSAGECURRENT reaches the value of MESSAGEMAX, CA OPS/MVS terminates and issues message OPS3146S. Setting the MESSAGEMAX parameter to its maximum possible value prevents CA OPS/MVS from ever shutting down due to an excessive message rate.

Default value

3000

Other possible values

Any number of messages between 100 and 100000

Set or modify this parameter…

Anytime

Example: MESSAGEMAX

This function limits CA OPS/MVS to generating 250 WTO messages.

```
OPSPRM('SET','MESSAGEMAX','250')
```

MESSAGERATE Parameter

This parameter sets the rate by which CA OPS/MVS decrements the value of the MESSAGECURRENT counter every second.

Default value

100

Other possible values

Any number between 1 and 1000

Set or modify this parameter…

Anytime

Example: MESSAGERATE

This function decrements the MESSAGECURRENT counter by 5 each second.

```
OPSPRM('SET','MESSAGERATE','5')
```

MSGCOLOR Parameter

This parameter sets the default display color for messages in OPSLOG browse for which the MSG.COLOR attribute variable was not set.

Note: You set the MSG.COLOR variable in AOF message rules.

Default value

NONE

When the MSGCOLOR parameter has a value of NONE, the message descriptor code (if any) determines the message display color. Otherwise, the color specified on MSGCOLOR overrides the color set by the message descriptor code.

Other possible values

BLUE, GREEN, PINK, RED, TURQ (turquoise), WHITE, or YELLOW

Set or modify this parameter…

Anytime

Example: MSGCOLOR

This function sets the default message display color to green.

```
OPSPRM('SET','MSGCOLOR','GREEN')
```

MSGDRAINRATE Parameter

This parameter with the MSGTHRESHOLD parameter, detects address spaces that issue excessive numbers of messages. To understand what these parameters do, consider this analogy: each address space is like a bathtub with a depth set by MSGTHRESHOLD. Each time the address space issues a message, it flows into this bathtub, which has a drain through which the MSGDRAINRATE number of messages flows each second. When the message bathtub overflows, CA OPS/MVS issues message OPS4402O.

To prevent messages from overflowing an address space and causing system problems due to looping, you can create a message rule. This rule might cancel the looping address space or place it in a penalty performance group.

Default value

10 (messages)

Other possible values

Any number of messages between 1 and 32767

Set or modify this parameter…

Anytime

Example: MSGDRAINRATE

This function sets the message drain rate to 5000.

```
OPSPRM('SET','MSGDRAINRATE','5000')
```

MSGTHRESHOLD Parameter

This parameter, with the MSGDRAINRATE parameter, detects address spaces that generate excessive numbers of messages. For more information, see MSGDRAINRATE Parameter in this chapter.

Default value

3000

Other possible values

Any number between 10 and 32767

Set or modify this parameter…

Anytime

Example: MSGTHRESHOLD

This function sets the threshold for messages in an address space to 10000.

```
OPSPRM('SET','MSGTHRESHOLD','10000')
```

PROPAGATEATTR Parameter

This determines whether CA OPS/MVS propagates back to the console display color changes for messages that the z/OS subsystem interface intercepts, when a CA OPS/MVS rule modified the color attribute.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: PROPAGATEATTR

This function tells CA OPS/MVS to propagate message color changes.

```
OPSPRM('SET','PROPAGATEATTR','YES')
```

SSIMSG Parameter

This parameter determines whether CA OPS/MVS processes messages before or after JES2 processes them.

Default value

NO

This value causes CA OPS/MVS to process messages after JES2 has seen them. Changes made to the messages are not reflected in the job log messages of an individual, but the changes will appear in the system SYSLOG.

Other possible values

YES

This value causes CA OPS/MVS to process messages before JES2 does so, preventing the CA OPS/MVS MSG.JOBNM environmental variable from being populated.

Set or modify this parameter...

At initialization

Example: SSIMSG

This function tells CA OPS/MVS to process messages before JES2 sees them.

```
OPSPRM('SET','SSIMSG','YES')
```

SSIWTL Parameter

This parameter determines whether CA OPS/MVS processes Write-to-Log (WTL) messages.

Default value

NO

This value prevents CA OPS/MVS from processing WTL messages.

Other possible values

YES

This value causes CA OPS/MVS to treat WTL messages like other messages, except that it deliberately skips the WTL done by JES2 or CONSOLE to put the message on the SYSLOG.

Set or modify this parameter...

At initialization

Example: SSIWTL

This function prevents CA OPS/MVS from processing WTL messages.

```
OPSPRM('SET','SSIWTL','NO')
```

WTODEFAULTROUTE Parameter

This parameter specifies the single z/OS route code to add to any WTO issued by OPSWTO or ADDRESS WTO when no route codes are specified, no console is specified, and the MCSFLAG is not set to HRDCPY only. This parameter supersedes any z/OS default route code and reduces the number of undeliverable messages.

Default value

0

No default route code is defined.

Other possible values

A single integer route code between 0 and 128

Set or modify this parameter...

Anytime

Example: WTODEFAULTROUTE

This function sets the default route code to 24.

```
OPSPRM('SET','WTODEFAULTROUTE','24')
```

Security Parameters

The parameters described in the following sections affect CA OPS/MVS security.

AUTHSTRING Parameter

This parameter allows testing of the CA OPS/MVS security user exit.

Typically, CA OPS/MVS loads the OPUSEX module in CSA. Specifying a user ID allows a particular user ID to load the security exit from a steplib or joblib for testing purposes. To permit this, you set the AUTHSTRING parameter to the literal TRIAL plus the user ID to be used. The resulting parameter value is TRIAL*userid*.

Default value

NONE

Other possible values

TRIAL*userid*

userid is the user ID that will load the security exit. You must specify both the literal TRIAL, and the *userid*.

Set or modify this parameter...

Anytime

Example: AUTHSTRING

This function allows user XYZ123 to test the security user exit (OPUSEX) from a steplib or joblib.

```
OPSPRM('SET','AUTHSTRING','TRIALXYZ123')
```

AUTHVALUE Parameter

This parameter is reserved for future use.

EXTSECCLASS Parameter

Specifies the resource class name for your site.

Default values

Defaults to FACILITY if the host z/OS is running IBM RACF.

Defaults to FAC if the host z/OS is running CA ACF2.

Defaults to FACILITY if the host z/OS is running CA Top Secret.

Other possible values

The value you specify depends on the external security manager running on your host system. Set this parameter to one of the following values:

- IBMFAC (if your external security manager is CA Top Secret).
- FAC (if your external security manager is CA ACF2).
- FACILITY (if your external security manager is IBM RACF).

You can create and use another resource class name that is based on the security package you have installed on the target z/OS system.

Set or modify this parameter...

At initialization

Example: EXTSECURITY

Specifies the new resource class name OPSCLS on the EXTSECCLASS parameter.

```
EXTSECCLASS(OPSCLS)
```

For more information, see the *CA OPS/MVS Security Guide*.

EXTSECPREFIX Parameter

Specifies the prefix for all security resource names that CA OPS/MVS defined. Use this value as the first or highest level qualifier for all resource names that your external security uses.

Default value

OP$MVS

Other possible values

You can override the prefix value with the EXTSECPREFIX parameter to meet local naming standards.

Set or modify this parameter...

At initialization

Example: EXTSECURITY

Grant user OPSUSR UPDATE access to the external security resource:

```
OP$MVS.OPSPARM
```

For more information, see the *CA OPS/MVS Security Guide.*

EXTSECSHOW Parameter

Sends trace messages to the OPSLOG through message OPS2109T. Specify ON to turn on trace messages. The trace messages show event checking information from SAF.

For more information, see the *CA OPS/MVS Security Guide* and the *CA OPS/MVS Message Guide.*

EXTSECSQLSUFFIX Parameter

This parameter specifies the suffix for all security resource names that CA OPS/MVS defined for the SQL security event.

Thus, you can choose between two formats for the SQL security event:

EXTSECSUFFIX=TBL - table name only

<prefix>.SQL.<table>

Or

EXTSECSUFFIX=TBL.CMD - table name and a command type

<prefix>.SQL.<table>.<cmdtype>

Where <prefix> is the specified or defaulted parameter EXTSECPREFIX value.

Default value

TBL

Other possible values

TBL.CMD

Set or modify this parameter...

At initialization

Example: EXTSECSQLSUFFIX=TBL

Then, you attempt to create a table named 'NEWTAB':

The SQL resource name generated in the CA OPS/MVS security event is:

<prefix>.SQL.NEWTAB

with an access type of UPDATE

Then, you insert data into the table 'NEWTAB'; the resource name is the same:

<prefix>.SQL.NEWTAB

with an access type of UPDATE

Now, suppose you select from data from the table 'NEWTAB'; the resource name is the same:

<prefix>.SQL.NEWTAB

The access type is now READ.

Example: EXTSECSQLSUFFIX=TBL.CMD

Then, you attempt to create a table named 'NEWTAB':

The SQL resource name generated in the CA OPS/MVS security event is:

<prefix>.SQL.NEWTAB.CT

with an access type of UPDATE

Then, you insert data into the table ‘NEWTAB’; the resource name is the same:

<prefix>.SQL.NEWTAB.IN

with an access type of UPDATE

Now, suppose you select from data from the table ‘NEWTAB’; the resource name is the same:

<prefix>.SQL.NEWTAB.SE

The access type is READ.

Where <prefix> is the specified or defaulted parameter EXTSECPREFIX value.

Possible CMD type values:

Value	SQL command
CT	Create table
IN	Insert rows
UP	Update rows
SE	Select
DE	Delete rows
DC	Declare cursor
OP	Open cursor
FE	Fetch
CL	Close cursor
DT	Drop table
CA	Alter table add column
CD	Alter table drop column
CI	Create index
DI	Drop index

For more information, see the *CA OPS/MVS Security Guide*.

EXTSECURITY Parameter

Turns on or off external security. Specify ON to turn on external security.

Default value

OFF

Other possible values

ON

Set or modify this parameter...

At initialization

Example: EXTSECURITY

Specify ON to turn on external security.

```
EXTSECURITY=ON
```

For more information, see the *CA OPS/MVS Security Guide*.

PASSWORDTEXTn Parameter

This parameter determines a set of character strings representing character sequences that precede password text in commands or command echoes. When CA OPS/MVS detects a match, the subsequent text (the password) is blanked out in both the OPSLOG and SYSLOG.

The *n* value is a number from 1 to 9.

Default value

NONE

Other possible values

Any valid character string

Set or modify this parameter...

Anytime

Example: PASSWORDTEXT*n*

This function sets the string to PASSWORD=.

```
OPSPRM('SET','PASSWORDTEXT5','PASSWORD=')
```

SECRULEFAILURE Parameter

This parameter sets the return code for security rules that fail because of coding errors or because they exceed rule execution limits.

Default value

NOACTION

This value causes CA OPS/MVS to ignore the failed security rule, allowing the OPUSEX security routine to control access to CA OPS/MVS.

Note: As distributed, the OPUSEX routine requires the issuer to have TSO OPER authority. Therefore, if you modify OPUSEX or all your TSO users have OPER authority, a security rule failure can allow a user improper access to a resource.

Other possible values

REJECT

This value causes the secured function to fail, so it provides maximum security. However, if you use this option and a security rule controlling the ADDRESS AOF DISABLE command contains an error, you may not be able to disable any rules. You can recover from this situation using a command rule that allows secured MCS consoles to issue AOF commands. The OPSAOF rule in the OPS.SAMPLE.RULES library is an example of this type of rule.

Set or modify this parameter...

Anytime

Example: SECRULEFAILURE

This function causes the secured function to fail.

```
OPSPRM('SET','SECRULEFAILURE','REJECT')
```

SECRULEFAILURE Parameter

This parameter sets the return code for security rules that fail because of coding errors or because they exceed rule execution limits.

Default value

NOACTION

This value causes CA OPS/MVS to ignore the failed security rule, allowing the OPUSEX security routine to control access to CA OPS/MVS.

Note: As distributed, the OPUSEX routine requires the issuer to have TSO OPER authority. Therefore, if you modify OPUSEX or all your TSO users have OPER authority, a security rule failure can allow a user improper access to a resource.

Other possible values

REJECT

This value causes the secured function to fail, so it provides maximum security. However, if you use this option and a security rule controlling the ADDRESS AOF DISABLE command contains an error, you may not be able to disable any rules. You can recover from this situation using a command rule that allows secured MCS consoles to issue AOF commands. The OPSAOF rule in the OPS.CCLXSAMP library is an example of this type of rule.

Set or modify this parameter...

Anytime

Example: SECRULEFAILURE

This function causes the secured function to fail.

```
OPSPRM('SET','SECRULEFAILURE','REJECT')
```

SECURITYLOG Parameter

This parameter determines whether CA OPS/MVS logs failed requests, made through OPSECURE('R') for resources in either the CA ACF2 or RACF environment. Logging of these requests always occurs outside the rule environment unless the OPS/REXX compiler has APF authority.

Default value

YES

CA OPS/MVS logs security failures.

Other possible values

NO

CA OPS/MVS does not log security failures.

Set or modify this parameter...

Anytime

Example: SECURITYLOG

This function prevents CA OPS/MVS from logging failed resource requests.

```
OPSPRM('SET','SECURITYLOG','NO')
```

SECURITYRULESET Parameter

This parameter defines the rule set name of the library from which security rules can be activated. When you set this parameter, the specified library becomes the *only* library from which you can activate security rules, preventing unauthorized users from using another rule set to override the security rule that your data center has implemented.

For more information on rule sets, see the *Installation Guide*. For more information on security rules, see the *AOF Rules User Guide*.

Default value

No default

Other possible values

The name of any security rule set

Set or modify this parameter...

At initialization

Example: SECURITYRULESET

This function defines SYSSECUR as the security rule set to activate.

```
OPSPRM('SET','SECURITYRULESET','SYSSECUR')
```

Miscellaneous Operating Parameters

The parameters described in the following sections affect miscellaneous areas of basic CA OPS/MVS operation.

AAMSGTBLENTRIES Parameter

This parameter specifies the number of unique message IDs that the Automation Analyzer allows in its message table.

Default value

15,000

Other possible values

Any number of message entries between 100 and 32767

Set or modify this parameter...

Anytime

Example: AAMSGTBLENTRIES

This function sets the number of entries in the Automation Analyzer message table to 20000.

```
OPSPRM('SET','AAMSGTBLENTRIES','20000')
```

Note: When the value of this parameter is not set (for example, when a value of 0 is displayed), the Automation Analyzer program uses a default value of 15,000 table entries.

Important! If you specify larger values on this parameter, the TSO or batch address space that is performing the analysis requires more virtual storage. Some installations limit the size of the “above-the-line” region. Thus, if you increase the value of this parameter in these cases, it may result in storage-related abends.

ABENDHIGH Parameter

This parameter displays the high watermark that the abend rate control mechanism has reached during the current life of the product or since this parameter value was last reset. Compare its value to the value specified on the ABENDMAX parameter to determine how close CA OPS/MVS has come to the point at which it would shut itself down due to exceeding the ABENDMAX limit.

Note that resetting this parameter affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default value

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: ABENDHIGH

This function resets the ABENDHIGH value to zero.

```
OPSPRM('SET','ABENDHIGH',0)
```

ABENDMAX Parameter

This parameter dictates the maximum number of abends CA OPS/MVS can take per second. Setting the ABENDMAX parameter to its maximum possible value prevents CA OPS/MVS from ever shutting down due to an excessive abend rate.

Default value

100

Other possible values

Any number of abends between 0 and 1000

Set or modify this parameter...

Anytime

Example: ABENDMAX

This function sets the number of abends per second to 500.

```
OPSPRM('SET','ABENDMAX','500')
```

ABENDRATE Parameter

This parameter determines the rate by which CA OPS/MVS decreases the ABENDCURRENT counter in every elapsed second.

Default value

0.1 (per second)

Other possible values

Between 0.0 and 1000 abends per second

Set or modify this parameter...

Anytime

Example: ABENDRATE

This function tells CA OPS/MVS to decrement the counter by 0.2.

```
OPSPRM('SET','ABENDRATE','0.2')
```

BYPASSDESC Parameter

This parameter, whose default value is equivalent to descriptor code 16, specifies the descriptor code used for internal CA OPS/MVS WTO messages. These messages will not execute rules or be placed in OPSLOG.

For example, when the CA OPS/MVS IOF facility issues a WTO request for an IMS message, IOF has already passed the message it intercepted to AOF for processing. IOF then sets the special descriptor code, so when the WTO is issued, CA OPS/MVS does not reprocess it through AOF. The CA OPS/MVS COF facility also does the same processing to insure that duplicate processing is not done for COF messages.

Although you can bypass AOF and OPSLOG processing for other messages, you cannot use AOF to change the descriptor codes of the messages because AOF already will have received those messages. You will need to use other methods to change the descriptor codes.

WARNING! When specifying a value for BYPASSDESC, do not specify a descriptor code that is used by any other type of message.

Default value

X'0001'

Other possible values

Any descriptor code, expressed in hexadecimals

Set or modify this parameter...

Anytime

Example: BYPASSDESC

This function sets the value of BYPASSDESC to “X'0008'” (which is equivalent to a descriptor code of 13).

```
OPSPRM("SET","BYPASSDESC","X'0008'")
```

LOGRECHIGH Parameter

This parameter displays the high watermark that the LOGREC record rate control mechanism has reached during the current life of the product or since this parameter value was last reset. This is the highest value that the LOGRECCURRENT parameter has ever reached since that time. Compare the LOGRECHIGH value to the value specified on the LOGRECMAX parameter to determine how close CA OPS/MVS has come to the point at which it would shut itself down due to exceeding the LOGRECMAX limit.

Note: Resetting this parameter affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default value

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter…

Anytime

Example: LOGRECHIGH

This function resets the LOGRECHIGH value to zero.

```
OPSPRM('SET','LOGRECHIGH',0)
```

LOGRECMAX Parameter

This parameter sets the maximum number of LOGREC records that CA OPS/MVS can write. Setting the LOGRECMAX parameter to its maximum possible value prevents CA OPS/MVS from ever shutting down due to an excessive log record rate.

Default value

50

Other possible values

Any number between 10 and 1000

Set or modify this parameter…

Anytime

Example: LOGRECMAX

This function limits the number of LOGREC records to 100.

```
OPSPRM('SET','LOGRECMAX','100')
```

LOGRECRATE Parameter

This parameter determines how many LOGREC records CA OPS/MVS can write in a given second.

Default value

0.01 (per second)

Other possible values

Any number between 0 and 100 per second

Set or modify this parameter…

Anytime

Example: LOGRECRATE

This function limits CA OPS/MVS to writing 4 records per second.

```
OPSPRM('SET','LOGRECRATE','4')
```

MAINPRM Parameter

This parameter supplies a parameter string for CA OPS/MVS. You can supply this parameter string using the OPSPRM() REXX function, the OPSPRM or OPSPARM command, or through the MAINPRM substitutional parameter in the CA OPS/MVS started-task JCL. In addition, you can fetch the value of this parameter from a rule or a REXX EXEC through the OPSINFO REXX function. For a description of the OPSINFO function, see the *Command* and *Function Reference*.

Although the entire main product parameter string (up to 100 characters) is available through the OPSINFO('MAINPRM') REXX function, when using the OPSPRM() REXX function, the OPSPRM or OPSPARM command processor, or OPSVIEW option 4.1.1, you are restricted to setting and displaying the first 50 characters of the string.

Default value

NONE

Other possible values

Any valid parameter string, with a maximum length of 50 bytes

Set or modify this parameter…

Anytime

Example: MAINPRM

This function defines TESTMODE as the CA OPS/MVS main parameter string.

```
OPSPRM('SET','MAINPRM','TESTMODE')
```

NAME Parameter

The product security interface uses the value of the NAME parameter as the application name (for RACF security) or the source name (for CA ACF2 security).

Note: Because you may change the value of the NAME parameter at any time, you may set it to PAUTOMAT in your startup parameters to maintain compatibility with your existing environment.

Default value

OPS/MVS

Other possible values

Any name you want to use, with a maximum of eight characters

Set or modify this parameter…

Anytime

Example: NAME

This function assigns the name PAUTOMAT to CA OPS/MVS.

```
OPSPRM('SET','NAME','PAUTOMAT')
```

NMEPSRECID Parameter

This parameter determines the name of the CA NetMaster EPS receiver ID that receives the alerts generated by ADDRESS NETMASTR host commands.

Note: When this parameter is not set (or contains only blanks or nulls), ADDRESS NETMASTR uses the default CA NetMaster alert receiver ID of $AMALERT. You should only set this parameter if you have changed the name of the default CA NetMaster alert receiver.

Default value

None

Other possible values

Any valid CA NetMaster EPS receiver ID

Set or modify this parameter…

Anytime

Example: NMEPSRECID

This function sets the CA NetMaster alert receiver ID name to $USALERT.

```
OPSPRM('SET','NMEPSRECID','$USALERT')
```

OPSTART1RESVAL Parameter

This parameter enables the CA OPS/MVS initialization CLIST (OPSTART1), initialization REXX program (OPSSPA00), or both, to pass a result value back to CA OPS/MVS.

If you specify SHUTDOWN (in uppercase characters) as the first eight characters of the value of the OPSTART1RESVAL parameter, CA OPS/MVS issues message OPS0032I and terminates immediately after the completion of the initialization CLIST or REXX program. Message OPS0032I includes the last eight characters of the OPSTART1RESVAL parameter value, which you can set to indicate the reason for the CA OPS/MVS termination.

For example, in case the allocation of the SYSCHK1 data set in OPSSPA00 ever fails, you can set the OPSTART1RESVAL parameter to SHUTDOWNSYSCHK1 to prevent CA OPS/MVS from initializing without the necessary global variables and relational tables, and to identify SYSCHK1 as the problem.

Note: The default OPSSPA00 has been changed to do this.

You can also use the OPSTART1RESVAL parameter to set an arbitrary value during initialization that you can use during later processing.

Default value

Blanks

Other possible values

Any 1- to 16-character string

Set or modify this parameter…

At initialization

Example: OPSTART1RESVAL

This function causes CA OPS/MVS to issue message OPS0032I and terminate immediately after the completion of the initialization CLIST or REXX program. Message OPS0032I contains a reference to SYSCHK1.

```
OPSPRM('SET','OPSTART1RESVAL','SHUTDOWNSYSCHK1')
```

PERMDEBUG Parameter

This parameter is a debugging tool. It is not intended for general customer use; rather, it should only be used under the direction of a CA Customer Support representative.

Note: The value of the PERMDEBUG parameter will not change for the life of the IPL (even if the product terminates) unless it is explicitly changed through the OPSPRM or OPSPARM function.

Default value

X'00000000'

Other possible values

Any valid four-byte hexadecimal value

Set or modify this parameter…

Anytime

Example: PERMDEBUG

This function sets PERMDEBUG to its default value.

```
OPSPRM("SET","PERMDEBUG","X'00000000'")
```

PRODUCTNAME Parameter

This parameter is the literal name of CA OPS/MVS. Its value is returned by the OPSINFO('PRODUCT') REXX function, and it is substituted into the text of product messages as the CA OPS/MVS name.

Note: Because you may change the value of the PRODUCTNAME parameter at any time, you may set it to OPS/MVS or PREVAIL/XP-AUTOMATION in your startup parameters to maintain compatibility with your existing environment.

Default value

CA-OPS/MVS

Other possible values

Any name that you want to use that contains no more than 21 characters

Set or modify this parameter…

Anytime

Example: PRODUCTNAME

This function assigns the name OPS/MVS to CA OPS/MVS.

```
OPSPRM('SET','PRODUCTNAME','OPS/MVS')
```

QUICKREFDBASE Parameter

This parameter sets the default name of the Chicago-Soft Ltd. MVS/QuickRef product database. Use this parameter only if your site uses the MVS/QuickRef product and you have not specified the database name in the MVS/QuickRef module (QWIKOPTS). For instructions on modifying the options table, see the *MVS/QuickRef User's Guide*. An individual MVS/QuickRef user can override the QUICKREFDBASE parameter by allocating a different database to QWREFDD, through either JCL in the logon procedure or through the TSO ALLOCATE command. For more information, see your MVS/QuickRef documentation.

Important! We strongly recommend that you allow the QUICKREFDBASE parameter to be set to its default value.

The CA OPS/MVS interface to MVS/QuickRef uses the ISPF BROWSE interface service. This means you can use any ISPF BROWSE command, such as FIND, when viewing MVS/QuickRef data.

Default value

Binary zeros (indicating that no database name is specified)

Other possible values

The name you use for the MVS/QuickRef database

Set or modify this parameter…

Anytime

Example: QUICKREFDBASE

This function sets the database name to CHSFT.QUIKRFDB.

```
OPSPRM('SET','QUICKREFDBASE','CHSFT.QUIKRFDB')
```

QUICKREFTYPE Parameter

This parameter determines the location from which MVS/QuickRef extracts information about the CA OPS/MVS own messages.

The QUICKREFTYPE parameter was designed because the MVS/QuickRef database almost never corresponds exactly to the particular release of CA OPS/MVS that you are running. Therefore, it does not contain current product message information. The TSO HELP file, which contains a HELP member for each product message, is kept current with each product maintenance tape, and is therefore a more accurate source of information about the product messages.

We strongly recommend that you allow the value of the QUICKREFTYPE parameter to default.

The output that is displayed will be slightly different depending on which value you specify.

Default value

NONE

This value indicates that the message information should be extracted from TSO HELP file of the product.

Other possible values

MSG

This value indicates that the message information should be extracted from the MVS/QuickRef database.

TSOHELP

This value indicates that the message information should be extracted from the TSO HELP file of the product.

Set or modify this parameter…

Anytime

Example: QUICKREFTYPE

This function indicates that the information should come from the MVS/QuickRef database.

```
OPSPRM('SET','QUICKREFTYPE','MSG')
```

SMFRECORDING Parameter

This parameter determines whether SMF creates records.

Default value

YES

This value indicates that SMF records will be created once the SMFRECORDNUMBER parameter is set to a valid value.

Other possible values

NO, ON, OFF

Set or modify this parameter...

Anytime

Example: SMFRECORDING

This function prevents CA OPS/MVS from creating SMF records, even after setting SMFRECORDNUMBER to a valid value.

```
OPSPRM('SET','SMFRECORDING','NO')
```

SMFRECORDNUMBER Parameter

This parameter determines SMF record types that CA OPS/MVS components can create when:

- CA OPS/MVS stops.
- An OSF server terminates.
- An internal IMS BMP terminates.
- An AOF rule or rule set is disabled. For more information, see SMFRULEDISABLE Parameter in the chapter “Parameters for Facilities.”
- After each OSF transaction completes. For more information, see OSFTRANSSMFREC Parameter in the chapter “Parameters for Facilities.”

Default value

0

The default value of 0 indicates that no SMF records will be created; however, 0 is not a valid value when you are setting the parameter. To stop CA OPS/MVS from creating SMF records after this parameter is set to a valid value, set SMFRECORDING to NO.

Other possible values

A value between 128 and 255 (creates standard SMF subtype records)

Set or modify this parameter...

Anytime

Example: SMFRECORDNUMBER

This function tells CA OPS/MVS to create SMF records of type 128.

```
OPSPRM('SET','SMFRECORDNUMBER','128')
```

SQLABEND Parameter

The SQLABEND parameter determines whether the RDF SQL processor will intentionally abend with an S0C3 abend code when an internal SQL logic error is detected.

If the SVCDUMP parameter is also set to ON, an SVCDUMP will be requested.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime after SQL initialization is complete. It cannot be set in OPSSPA00.

Example: SQLABEND

```
OPSPRM('SET','SQLABEND','NO')
```

SVCDUMP Parameter

This parameter provides diagnostic data for problem resolution by initiating an SVC dump for the first occurrence of an unexpected abend in the OPS/MVS address space. You can set this parameter dynamically using the OPSPRM REXX function or the OPSPARM POI command at any time.

Default value

ON

Initiates an SVC dump when an unexpected abend occurs in any OPS/MVS task or in cross-memory processing routines, such as AOF rule processing, REXX functions, and Address environments.

Other possible values

OFF

Suppresses automatic SVC dumps.

Set or modify this parameter...

Anytime

Example: SVCDUMP

Enables SVCDUMP for the next unexpected abend.

```
rc=OPSPRM('SET','SVCDUMP','ON')
```

VIO Parameter

This parameter supplies the name in the Eligible Device Table (EDT) that allows VIO. This name is used in processing ADDRESS TSO clauses to trap TSO command output.

Default value

The first VIO eligible name in the EDT

Other possible values

Any generic or esoteric device name in the EDT that allows VIO

Set or modify this parameter...

Anytime

Example: VIO

Specifies SYSVIO as the device that allows VIO.

```
OPSPRM('SET','VIO','SYSVIO')
```

Parameters for Resolving Problems

We provide a set of special parameters, as listed, for use in diagnosing CA OPS/MVS operating problems.

Important! Set these parameters only when a CA Customer Support representative asks you to do so.

- DEBUG xxxx parameters

  These parameters set debugging flags for various CA OPS/MVS routines. The *xxxx* is a set of characters specifying which routine to trace. For example, the DEBUGSSRW parameter allows tracing for the CA OPS/MVS SSRW routine or DEBUGEXTSEC parameter allows tracing for CA OPS/MVS external security checks.

- EPIDEBUG parameter

  This parameter allows the CA OPS/MVS EPI facility to trace VTAM exits.

- IMSRELxCHANGEID, IMSRELxCHANGELEN, IMSRELxCHANGEOFF, IMSRELxLEN, IMSRELxOFF, IMSRELxOFFSET, IMSRELxSTRING, and IMSRELxSVC parameters

  These parameters provide information about CA OPS/MVS processing for IMS SVCs.

- OCDEBUG parameter

  This parameter activates CA OPS/MVS messages that can help to determine throughput from executing commands.

- TRACE*xxxx* parameters

  These parameters activate tracing for various CA OPS/MVS entries and exits. The *xxxx* value represents the name of the item to be traced. For example, the TRACE3X parameter controls tracing of the UX31 exit.

For more information about these parameters, ask your CA Customer Support representative.

Chapter 3: Parameters for Facilities

This section contains the following topics:

Parameter Groups

This chapter describes the parameters that allow you to tailor the operation of CA OPS/MVS facilities such as rules, OPS/REXX, the AOF, OSF, and so on.

The groups of CA OPS/MVS parameters discussed in this chapter control the operation of these basic CA OPS/MVS facilities:

- AOF
- ECF
- EPI
- OSF
- Rules
- Global variables
- OPS/REXX
- System State Manager
- Automate-to-CA OPS/MVS rules translation

Change or Display Parameter Values

To change the values of parameters controlling CA OPS/MVS facilities or display those values, use either the OPSPRM function of OPS/REXX (the recommended method) or the OPSPARM command processor. For a description of OPSPRM and OPSPARM and the syntax of each, see Setting or Displaying Parameter Values in the chapter “Parameters.”

AOF Parameters

The parameters described in the following sections allow you to tailor AOF operation.

AOFACTIVE Parameter

The AOFACTIVE parameter activates or deactivates the CA OPS/MVS Automated Operations Facility.

Note: The AOFACTIVE parameter must be set to ON for the AOFINITREXX parameter to take effect. For details, see AOFINITREXX Parameter in this chapter.

Default value

ON

Other possible values

OFF

Set or modify this parameter...

Anytime

Example: AOFACTIVE

This function deactivates the AOF.

```
OPSPRM('SET','AOFACTIVE','OFF')
```

AOFACTIVEREXX Parameter

The AOFACTIVEREXX parameter specifies the name of an OPS/REXX program that you have written to validate your AOF environment and to perform any processing that you want after the auto enable process has completed and before the OPx0123O message is issued indicating that the AOF is active. One purpose of this OPS/REXX EXEC is to validate that those AOF rule sets and rules, which are absolutely required for your automation to function correctly, are enabled.

The program that you specify as the value of AOFACTIVEREXX must be in a PDS library that is allocated to the OPSMAIN STC JCL under the SYSEXEC ddname.

Important! For the AOFACTIVEREXX parameter to take effect, the value of the AOFACTIVE parameter must be ON. For details, see AOFACTIVE Parameter in this chapter.

Note: Using the ADDRESS AOF ENABLE command in the AOFACTIVEREXX program to enable individual rules is slower than using the AUTOENABLE process, which processes an entire rule set at a time.

REXX execs specified in AOFINITREXX and AOFACTIVEREXX will execute in the OPSMAIN address space. They cannot execute any Address OSF or TSO commands at AOF initialization time since the TMPs are not yet active.

For details on tailoring CA OPS/MVS startup procedures, see the *Administration Guide*.

Default value

Blanks (indicating that there is no OPS/REXX program to execute)

Other possible values

Any valid OPS/REXX program name

Set or modify this parameter...

At initialization

Example: AOFACTIVEREXX

This function names AOFACTV as the OPS/REXX program that CA OPS/MVS executes to validate the AOF environment, perform post auto-enablement processing, or both.

```
OPSPRM('SET','AOFACTIVEREXX','AOFACTV')
```

AOFDEFAULTADDRESS Parameter

The AOFDEFAULTADDRESS parameter defines the default host command environment for all AOF rules except request (REQ) rules.

Default value

MESSAGE

Other possible values

Any valid OPS/REXX host environment

Set or modify this parameter...

Anytime

Example: AOFDEFAULTADDRESS

This function sets MESSAGE as the default host command environment for AOF rules.

```
OPSPRM('SET','AOFDEFAULTADDRESS','MESSAGE')
```

Any misspelled OPS/REXX command can cause an unrecoverable loop, resulting in CA OPS/MVS shutdown. Setting the AOFDEFAULTADDRESS value to MESSAGE prevents looping because it forces you to send commands explicitly to the OPER host environment.

AOFDELETE Parameter

The AOFDELETE parameter determines how CA OPS/MVS processes the return values DELETE and DISPLAY in message rules. For details about message rule return values, see the *AOF Rules User Guide*.

Default value

YES

A return value of DELETE suppresses a message entirely (that is, the message appears neither on the console nor in the system log); a return value of DISPLAY prevents a message from appearing in the system log, but the message appears on the console.

Other possible values

NO

CA OPS/MVS processes the DELETE value as though it were SUPPRESS; it processes the DISPLAY value as though it were NORMAL.

Set or modify this parameter...

Anytime

Example: AOFDELETE

This function allows the AOF to suppress messages entirely.

```
OPSPRM('SET','AOFDELETE','YES')
```

AOFDESC Parameter

The AOFDESC parameter specifies the default descriptor codes of messages generated in AOF rules.

You can set these descriptor codes two ways because the descriptor codes that you specify for AOFDESC are the same as the descriptor codes specified for bytes 7 and 8 of the AOFDEST parameter value. For example, if you set a value for AOFDEST, and then set a value for AOFDESC *specifying* different *descriptor codes*, the codes set by AOFDESC override the earlier settings. The opposite is also true-if you set the AOFDESC parameter and then later specify different descriptor codes through the AOFDEST parameter, the AOFDEST settings override the AOFDESC settings.

The following table will help you set the AOFDESC parameter:

- Descriptor codes 1 through 11
- The hexadecimal representation of each code
- The corresponding OPSBITS() character string of each code
- A description of each code

Code	Hex Representation	OPSBITS() String	Description
1	X'8000'	SYSFAIL	System failure
2	X'4000'	IMEDACTN	Immediate action required
3	X'2000'	EVENACTN	Eventual action required
4	X'1000'	SYSSTAT	System status
5	X'0800'	IMEDCMD	Immediate command response
6	X'0400'	JOBSTAT	Job status
7	X'0200'	APPLPRGM	Application program
8	X'0100'	OOLMSG	Out-of-line message
9	X'0080'	OPERREQ	Request of the operator
10	X'0040'	DYNSTAT	TRACK command response
11	X'0020'	CRITEVET	Critical eventual action required

You can combine two or more hexadecimal representations to specify multiple descriptor codes, but keep in mind that descriptor codes 1-6 and descriptor code 11 are all mutually exclusive.

Consider these examples:

To represent descriptor codes...	Specify this value for AOFDESC...
6	X'0400'
2 and 7	X'4200'
5, 8, and 9	X'0980'

For more information about descriptor codes, see the IBM documentation.

Default value

X'0000'

Other possible values

Any valid descriptor codes, expressed in hexadecimals

Set or modify this parameter...

Anytime

Example: AOFDESC

This function sets the default descriptor codes to codes 2 and 7 (X'4000' + X'0200' = X'4200').

```
OPSPRM("SET","AOFDESC","X'4200'")
```

AOFDEST Parameter

The AOFDEST parameter specifies the default destination of messages (such as SAY and TRACE statements) generated by AOF rules. AOFDEST is valid only when the message has one of these severity levels: I (informational messages), W (warnings), E (error messages), or S (severe error messages).

Default Value: X'C300000000000000'

Other possible values

Any valid destination, expressed in hexadecimals. To specify a value for AOFDEST, it may be helpful for you to think of the value as 8 bytes of information. These 8 bytes break down as follows:

- The first byte can be either C3, which is the hexadecimal representation of the character C; or C2, which is the hexadecimal representation of the character B.

  If you specify C3, the messages are sent to the specified console (that is, the console specified by the fourth byte of the AOFDEST value) and also to OPSLOG Browse.

  If you specify C2, the messages are sent to OPSLOG Browse only, and the second through eighth bytes of the AOFDEST value are ignored.

- The second byte of the AOFDEST value is reserved; this value should always be 00.

- The third byte of the AOFDEST value indicates the MCS flags code. There are two values you can specify for this byte: 00 (for X'00') indicating that the messages should be queued for the console and hard copy; and 02 (for X'02') indicating the messages should be queued for hard copy only.

  Note: The integer value of X'02' is 7; the OPSBITS() character string that corresponds to this value is HRDCPY.

- The fourth byte indicates the default console ID. You can specify a console ID from 00 to 99, but you must represent it in hexadecimals (the decimal values 00 through 99 are equivalent to the hexadecimal values X'00' through X'63').

- The fifth and sixth bytes indicate the default routing codes for the messages. The routing codes that you specify here are the same as the routing codes that you may specify for the AOFROUTE parameter. For more information, see the description of the AOFROUTE Parameter in this chapter.

  So, if you use the AOFROUTE parameter to set default routing codes for AOF messages and then later use bytes 5 and 6 of the AOFDEST value to set different routing codes, the values you set last override the earlier settings. The opposite is also true. For a list of routing codes and their descriptions, see AOFROUTE Parameter in this chapter.

- The seventh and eighth bytes indicate the default descriptor codes for the messages. The descriptor codes that you specify here are the same as those that you may specify for the AOFDESC parameter. For details, see the description of the AOFDESC Parameter in this chapter. In other words, if you use the AOFDESC parameter to set default descriptor codes for AOF messages and then later use bytes 7 and 8 of the AOFDEST value to set *different descriptor codes*, the values you set last override the earlier settings. The opposite is also true. For a list of descriptor codes and their descriptions, see the list of descriptor codes in AOFDEST Parameter.

Set or modify this parameter...

Anytime

Example: AOFDEST

This function sends messages generated by AOF rules to console 5.

```
OPSPRM('SET','AOFDEST','C3000005000000000')
```

AOFEDQHIGH Parameter

The AOFEDQHIGH parameter displays the highest number of entries ever queued in any of the external data queues that are used by AOF rules for the current life of the product or since the parameter value was last reset. Compare its value to the value specified on the AOFMAXQUEUE parameter.

Note: Resetting these parameters affects the data recorded in the product SMF records and the reports produced by the Automation Measurement Environment (AME).

Default value

No default

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: AOFEDQHIGH

This function resets the AOFEDQHIGH value to zero.

```
OPSPRM('SET','AOFEDQHIGH',0)
```

AOFEDQWARNTHRESH Parameter

The AOFEDQWARNTHRESH parameter sets the threshold percentage for the use of any AOF rule's external data queue (EDQ), which when exceeded issues a trace message to the OPSLOG. This message (OPx4115T) will only be issued once for each rule even if this parameter is modified. Only disabling and reenabling the rule or restarting the product can ever allow the message to be issued again. When the value of this parameter is set to zero (the default value), AOF EDQ monitoring is disabled and the trace messages are not issued.

Note: The ADDRESS AOF LISTINST host command can be used to retrieve the AOF EDQ high used values for any enabled AOF rule irrespective of this parameter setting.

Default value

0

Other possible values

Any value between 0 and 100 percent.

Set or modify this parameter...

Anytime

Example: AOFEDQWARNTHRESH

This function resets the AOFEDQWARNTHRESH value to 80%.

```
OPSPRM('SET','AOFEDQWARNTHRESH',80)
```

AOFFIRELIMIT Parameter

This parameter limits the number of times that a single AOF rule can execute per minute.

- If AOFLIMITDISABLE is set to YES, the rule is disabled if it exceeds this limit.
- With AOFLIMITDISABLE set to NO, the rule is logically disabled until the next minute boundary if it exceeds this limit.

To override this execution limit for an individual rule, use the FIRELIMIT keyword of the OPS/REXX OPTIONS statement, and place that statement in the initialization section of the rule. This is the best method to use if you plan to set the limit only once. For details, see OPTIONS Instruction in the chapter “Using OPS/REXX” in the *User Guide*.

Using the CA OPS/MVS LIMITCOUNT parameter, you can display the total number of AOF events that the limit set by AOFFIRELIMIT prevented from executing. The AOF section of the CA OPS/MVS summary SMF record also contains this information.

Default value

10000

Other possible values

Any positive integer greater than 0

Set or modify this parameter...

Anytime

Example: AOFFIRELIMIT

This function sets the maximum number of rule executions per minute to 50.

```
OPSPRM('SET','AOFFIRELIMIT','50')
```

AOFFORNSSI Parameter

The AOFFORNSSI parameter tells AOF message processing whether to honor the setting of the .FORNSSI statement in the currently active MPFLST*xx* parmlib member.

Notes:

- This parameter is only applicable if the SSIMSG parameter is set to YES. For details, see SSIMSG Parameter in the Message-related Parameters section.

Support for .FORNSSI was added through IBM APAR OA08482.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: AOFFORNSSI

This function deactivates the AOF.

```
OPSPRM('SET','AOFFORNSSI','YES')
```

AOFINITOPEN Parameter

The AOFINITOPEN parameter determines how the AOF rule data sets are opened during product initialization processing. In order to improve performance, the AOF rule data sets are opened only once during AOF initialization processing. Therefore, it is necessary to open them for the least restrictive processing mode that might be needed.

Some installations do not allow the CA OPS/MVS main address to have UPDATE authority to some rule data sets (usually the data set designated by the SECUITYRULESET parameter). In this case, you may set the AOFINITOPEN parameter to INPUT, and design your automation so that no update processing through ADDRESS AOF host commands is attempted, to any rule data set, during product initialization. AOF host commands that require UPDATE access to the rule data sets are SETAUTO, RESETAUTO, COMPILE, and DELCOMP.

Important! CA **strongly** recommends that this parameter be left at its default value of UPDATE. If you set the value to INPUT, but attempt to update any rules through some AOF action during initialization, abends occur and initialization fails.

Default value

UPDATE

Other possible values

INPUT

Set or modify this parameter...

At initialization

Example: AOFINITOPEN

Open the rule data sets for INPUT processing only during AOF initialization.

```
OPSPRM('SET','AOFINITOPEN','INPUT')
```

AOFINITREXX Parameter

This parameter specifies the name of an OPS/REXX program that you have written to set up your AOF environment. Creating such a program allows you to logically control your AOF environment, including controlling the sequence in which AOF rules are enabled.

The program you specify as the value of AOFINITREXX must be in a PDS library that is allocated to the OPSMAIN STC JCL under the SYSEXEC ddname.

For the AOFINITREXX parameter to take effect, the value of the AOFACTIVE parameter must be ON. For details, see the AOFACTIVE Parameter in this chapter.

Note: Using the ADDRESS AOF ENABLE command in the AOFINITREXX program to enable individual rules is slower than using the AUTOENABLE process, which processes an entire rule set at a time.

REXX execs specified in AOFINITREXX and AOFACTIVEREXX will execute in the OPSMAIN address space. They cannot execute any Address OSF or TSO commands at AOF initialization time since the TMPs are not yet active.

For details about tailoring CA OPS/MVS startup procedures, see the *Administration Guide*.

Default value

Blanks (indicating that there is no special program to execute)

Other possible values

Any valid OPS/REXX program name

Set or modify this parameter...

At initialization

Example: AOFINITREXX

This function names AOFINITS as the special OPS/REXX program that CA OPS/MVS executes to set up the AOF environment.

```
OPSPRM('SET','AOFINITREXX','AOFINITS')
```

AOFLIMITDISABLE Parameter

This parameter determines whether CA OPS/MVS disables AOF rules that exceed the limit on executions per minute set by the AOFFIRELIMIT parameter or the FIRELIMIT keyword on an OPS/REXX OPTIONS statement.

Default value

NO

This value issues message OPS4111O and disables a rule logically until the current minute has expired.

Other possible values

YES

This value issues message OPS4110O and disables a rule after it exceeds the limit.

Set or modify this parameter...

Anytime

Example: AOFLIMITDISABLE

This function disables rules that exceed the execution limit.

```
OPSPRM('SET','AOFLIMITDISABLE','YES')
```

AOFMAXCLAUSES Parameter

This parameter determines the maximum number of REXX clauses that AOF rules (except request rules) can execute. The REXXMAXCLAUSES parameter limits the number of REXX clauses in request rules. For more information, see REXXMAXCLAUSES Parameter in this chapter.

You can override this limit for an individual rule using the OPS/REXX OPTIONS statement. For details, see the chapter “OPS/REXX Built-in Functions” in the *Command and Function Reference*.

Default value

10000

Other possible values

Any number of clauses greater than 1

Set or modify this parameter...

Anytime

Example: AOFMAXCLAUSES

This function sets the maximum number of REXX clauses to 100000.

```
OPSPRM('SET','AOFMAXCLAUSES','100000')
```

AOFMAXCOMMANDS Parameter

This parameter defines the maximum number of host commands that AOF rules (except for request rules) can execute. Host commands are commands issued in all valid ADDRESS environments. To limit the number of host commands in request rules, use the REXXMAXCOMMANDS parameter. For more information, see the REXXMAXCOMMANDS Parameter in this chapter.

You can override this limit for an individual rule using the OPS/REXX OPTIONS statement. For details, see the chapter “OPS/REXX Built-in Functions” in the *Command and Function Reference*.

Default value

100

Other possible values

Any number of commands greater than 0

Set or modify this parameter...

Anytime

Example: AOFMAXCOMMANDS

This function limits the number of host commands to 50.

```
OPSPRM('SET','AOFMAXCOMMANDS','50')
```

AOFMAXPGMSIZE Parameter

This parameter sets the maximum size, in bytes, of a compiled OPS/REXX rule section or subroutine in virtual storage.

Default value

1048616 (bytes)

Other possible values

Any number of bytes greater than 32768

Set or modify this parameter...

Anytime

Example: AOFMAXPGMSIZE

This function sets the maximum program size to 2000000 bytes.

```
OPSPRM('SET','AOFMAXPGMSIZE','2000000')
```

AOFMAXQUEUE Parameter

This parameter defines the maximum number of output lines an AOF rule can have in its external data queue (EDQ).

Default value

5000

Other possible values

Any number from 1 to 32768

Set or modify this parameter...

At initialization

Example: AOFMAXQUEUE

This function sets the number of output lines to 6000.

```
OPSPRM('SET','AOFMAXQUEUE','6000')
```

Note: Although the maximum value is 32768, if you set this parameter to a high value, CA OPS/MVS may terminate during product initialization and issue a MESSAGE QUEUE GETMAIN FAILED error message. This is due to the many factors that influence the maximum value that can be used on any given system. The maximum values depend on the size of the extended private region in the system and on the value of other CA OPS/MVS parameters. For example, the BROWSEMAX and PROCESS parameters have a big impact on the actual limit value for the AOFMAXQUEUE parameter.

AOFMAXSAYS Parameter

This parameter limits the number of REXX SAY statements AOF rules (except for request rules) can issue. For request rules, the limiting parameter is REXXMAXSAYS. For more information, see the REXXMAXSAYS Parameter in this chapter.

You can override this limit for an individual rule using the OPS/REXX OPTIONS statement.

Default value

1000

Other possible values

Any number of SAY statements greater than or equal to 0

Set or modify this parameter...

Anytime

Example: AOFMAXSAYS

This function limits the number of SAY statements to 500.

```
OPSPRM('SET','AOFMAXSAYS','500')
```

AOFMAXSECONDS Parameter

This parameter defines the maximum number of seconds of elapsed time an AOF rule (except for request rules) can execute. To limit execution for request rules, use the REXXMAXSECONDS parameter. For more information, see REXXMAXSECONDS Parameter in this chapter.

Default value

10 (seconds)

Other possible values

Any number greater than or equal to 1

Set or modify this parameter...

Anytime

Example: AOFMAXSECONDS

This function limits rule execution time to five minutes.

```
OPSPRM('SET','AOFMAXSECONDS','300')
```

AOFMESSAGES Parameter

This parameter names the source of messages and DOMs that the AOF processes.

Note: In a sysplex environment, the value of AOFMESSAGES affects messages from other systems in the sysplex.

Default value

MVS

This value indicates that AOF processes z/OS messages and DOMs from the subsystem interface.

In a sysplex environment, CA OPS/MVS receives and processes messages and DOMs from this system only.

Other possible values

- NONE

  The NONE value indicates that AOF processes no messages or DOMs.

- MVSGLOBAL

  The MVSGLOBAL value is valid only for sites that are running in a sysplex environment or routing messages to CA OPS/MVS using CA MIC.

  If your site runs JES3, you can specify MVSGLOBAL on the JES3 global processor in order to automate the reissued messages being imported from the JES3 local processors.

  In a sysplex environment, specifying MVSGLOBAL causes CA OPS/MVS to receive and process messages and DOMs being imported from other systems in the sysplex.

  If messages are being imported to CA OPS/MVS using CA MIC, specifying MVSGLOBAL causes AOF to receive and process the imported messages.

- MVSLIST

  This value is valid only for sites that are running in a sysplex environment or routing messages to CA OPS/MVS using CA MIC. When MVSLIST is specified, CA OPS/MVS will receive and process messages from the systems specified on the CA OPS/MVS MVSSYS*n* parameters as well as local messages.

  In JES3 environments, the MVSGLOBAL value would normally be specified on the JES3 global system in order to automate the reissued messages being imported from the JES3 local processors. However, the MVSLIST option can be used in conjunction with the MVSSYS*n* parameters on the JES3 global if there is a need to specify the individual systems from which messages should be processed. For example, the MVSLIST option can be used to specify only the JES3 local systems in a mixed sysplex containing both JES3 and JES2 systems, effectively filtering out messages imported from the JES2 systems.

In a sysplex environment, specifying MVSLIST in conjunction with MVSSYS*n* systems causes CA OPS/MVS to receive and process messages imported from the specified system as well as the local system.

If messages are being imported to CA OPS/MVS using CA MIC, specifying MVSLIST along with MVSSYS*n* systems causes AOF to receive and process messages imported from the specified systems as well as local messages.

Set or modify this parameter...

Anytime

Example: AOFMESSAGES

This function selects the SSI as the message and DOM source. Messages from any system in the sysplex are eligible to trigger rules.

```
OPSPRM('SET','AOFMESSAGES','MVSGLOBAL')
```

Usage Recommendations

CA recommends the following regarding the use of the AOFMESSAGES parameter:

- JES2 sites should use a value of MVS
- JES3 sites should use a value of MVSGLOBAL on the JES3 global processor and a value of MVS on the JES3 local processors

WARNINGS!

- Do not change the value of the AOFMESSAGES parameter without first considering the impact that the change may have on your AOF rules.
- If you use the MVSGLOBAL or MVSLIST values in a sysplex environment, changes to the MSCOPE of your consoles may affect which messages are logged in OPSLOG and processed by the AOF.

Note: To avoid confusion between what appears in the OPSLOG and which events are processed by the AOF, give the BROWSEMESSAGES and AOFMESSAGES parameters identical values. However, the MVSLIST option is not available on the BROWSEMESSAGES parameter. If you specify MVSLIST for AOFMESSAGES, it is recommended that you specify MVSGLOBAL for BROWSEMESSAGES. This will cause OPSLOG to show local messages and messages imported from the systems specified on the MVSSYS*n* parameters.

AOFMPFBYPASS Parameter

The AOFMPFBYPASS parameter enables you to use the MPF list as a high-speed suppression mechanism.

When the value of AOFMPFBYPASS is YES, messages that have been suppressed by the z/OS MPF facility are not inserted into OPSLOG, nor are they eligible for AOF rule processing. Setting the value to YES reduces the value of OPSLOG, because it no longer contains a complete log of all system messages. However, it can result in a small reduction in CPU overhead when processing MPF-suppressed messages.

Since the small amount of CPU savings is secondary to the issue of a complete OPSLOG and the ability to consolidate all message automation in AOF rules, we recommend that you set AOFMPFBYPASS to NO.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: AOFMPFBYPASS

Using this function means that MPF-suppressed messages do not appear in OPSLOG, nor are they eligible for rule processing.

```
OPSPRM('SET','AOFMPFBYPASS','YES')
```

AOFMSGIDn Parameter

The AOFMSGID*n* parameters can specify a 1- to 10-character AOF special character message ID pattern. There are three AOFMSGIDn parameters named AOFMSGID1 to AOFMSGID3.

All messages processed by AOF message rules contain a match specification that:

- Must be matched by the derived message ID of the message event.
- This message ID is normally the first word within the first ten characters of the message.
- If any special delimiter characters (.-><=) are encountered in the first 10 characters, the message ID is truncated to the preceding characters unless a message ID pattern that includes the special character is specified.

An internal table of known patterns such as PDSM**- for the CA PDSMAN product is already included in the product. These parameters are provided for accommodation of any local or new commercial software that may use special characters in the message ID.

Note: Notify CA Support of any commercial software products that require the use of these parameters for CA OPS/MVS automation so that they can add an internal table entry to CA OPS/MVS by normal product maintenance.

Default value

DUMMY

The default value does not contain any special characters and will not be matched by any special character message ID.

Other possible values

A 1- to 10-character message pattern that contains at least one special character. An asterisk in the message pattern is considered a single non-delimiter wild card character.

Set or modify this parameter...

Anytime

Example: AOFMSGID*n* Parameter

This message pattern value prevents message IDs that begin with CAI-MSG from being truncated to CAI. The dash is treated as a valid message ID character.

```
OPSPRM('SET','AOFMSGID1','CAI-MSG')
```

Usage Recommendations

These parameters replace the need to apply an SMP/E ++USERMOD to modify the internal CA OPS/MVS special character message ID pattern table.

AOFNIPMESSAGES Parameter

The AOFNIPMESSAGES parameter determines whether CA OPS/MVS allows message rules to process NIP messages. Since this parameter is only effective prior to starting the primary job entry subsystem, we recommend that you set this parameter during initialization only.

Default value

NO

NIP messages are recorded in OPSLOG but are not processed by AOF message rules.

Other possible values

YES

NIP messages are recorded in OPSLOG and are eligible to be processed by AOF message rules.

Note: These message rules are driven at the time the NIP messages are being written to the SYSLOG rather than at the time the messages are being generated.

Set or modify this parameter...

Anytime

Example: AOFNIPMESSAGES

This function allows the AOF to process NIP messages at the time they are written to the SYSLOG.

```
OPSPRM('SET','AOFNIPMESSAGES','YES')
```

Note:

- This parameter applies to z/OS 1.4.2 or higher operating environments that have applied the required fixes to support IBM APAR OA10401.
- Since OPSINFO("EXITTYPE") returns a value of NIP when processing NIP messages, you can use it to write automation that is unique to NIP messages.
- You cannot change the attributes of NIP messages or suppress them in AOF rules.

AOFPRECOMPILED Parameter

This parameter determines whether CA OPS/MVS can compile rules without executing them.

Note: Setting the parameter to NO frees data sets containing precompiled rules. For details about precompiling rules, see How to Maintain the AOF Production Compiled Rules Library in the chapter "Using the OPSVIEW Control Option" in the *OPSVIEW User Guide*.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: AOFPRECOMPILED

This function activates the CA OPS/MVS facility for compiling rules and rule sets.

```
OPSPRM('SET','AOFPRECOMPILED','YES')
```

AOFPRECOMPILEDDSN Parameter

This parameter defines the data set containing compiled rules. CA OPS/MVS requires you to set both this parameter and the AOFPRECOMPILED parameter if you want to precompile rules; you cannot allocate a data set to store the compiled rules.

Note: Member names in precompiled data sets are encrypted. You can access those members only through OPSVIEW option 4.5.2.

For details about precompiling rules, see How to Maintain the AOF Production Compiled Rules Library in the chapter “Using the OPSVIEW Control Option” in the *OPSVIEW User Guide*.

Default value

No default

Other possible values

Any valid data set name

Set or modify this parameter...

Anytime

Example: AOFPRECOMPILEDDSN

This function specifies MY.COMP.RULES as the name for a precompiled data set.

```
OPSPRM('SET','AOFPRECOMPILEDDSN','MY.COMP.RULES')
```

AOFROUTE Parameter

The AOFROUTE parameter determines the default routing codes of messages generated in AOF rules.

There are actually two ways to set these routing codes, because the routing codes that you specify for AOFROUTE are the same as those specified for bytes 5 and 6 of the AOFDEST parameter value. For example, if you set a value for AOFDEST, and then set a value for AOFROUTE *specifying different* routing *codes*, the codes set by AOFROUTE override the earlier settings. The opposite is also true-if you set the AOFROUTE parameter and then later specify different routing codes through the AOFDEST parameter, the AOFDEST settings override the AOFROUTE settings.

To help you to set the AOFROUTE parameter, the following table lists this information:

- Routing codes 1 through 12
- The hexadecimal representation of each code
- The corresponding OPSBITS() character string of each code
- A description of each code

Code	Hex Representation	OPSBITS() String	Description
1	X'8000'	MSTRACTN	Master console action
2	X'4000'	MSTRINFO	Master console information
3	X'2000'	TAPEPOOL	Tape pool
4	X'1000'	DASDPOOL	Direct access pool
5	X'0800'	TAPELIB	Tape library
6	X'0400'	DISKLIB	Disk library
7	X'0200'	UR	Unit record pool
8	X'0100'	TP	Teleprocessing control
9	X'0080'	SECURITY	System security
10	X'0040'	SYSERROR	System/Error maintenance
11	X'0020'	PGMRINFO	Programmer information
12	X'0010'	EMULATOR	Emulator

You can combine two or more hexadecimal representations in order to specify multiple routing codes. Consider these examples:

To represent routing codes...	Specify this value for AOFROUTE...
2	X'4000'
1 and 2	X'C000'
9 and 11	X'00A0'

If you need more detailed information about routing codes, see the IBM documentation.

Default value

X'0000'

Other possible values

Any valid routing codes, expressed in hexadecimals

Set or modify this parameter...

Anytime

Example: AOFROUTE

This function sets the default routing codes to 9 and 11 (X'0080' + X'0020' = X'00A0').

```
OPSPRM("SET","AOFROUTE","X'00A0'")
```

AOFSOURCETEXT Parameter

The AOFSOURCETEXT parameter determines whether the source code of a rule is saved when the rule is activated.

Default value

YES

Other possible values

NO

If AOFSOURCETEXT has the value NO, then:

- The TRACE command is not operative.
- Should an error occur, CA OPS/MVS indicates the line number but cannot provide the source text.
- The REXX SOURCELINE() function does not display source text.
- CA OPS/MVS uses less storage than it does when AOFSOURCETEXT has the value YES.

Set or modify this parameter...

Anytime

Example: AOFSOURCETEXT

This function tells CA OPS/MVS to save the source text of a rule when it becomes active.

```
OPSPRM('SET','AOFSOURCETEXT','YES')
```

AOFSIZE Parameter

The AOFSIZE parameter determines the size of the AOF workspace (used to store REXX variables).

Using several new CA OPS/MVS parameters (AOFWSHIGHUSED, AOFWSHIGHUSEDPGM, AOFWSHIGHUSEDRULE, AOFWSHIGHUSEDDATE, and AOFWSHIGHUSEDTIME), you can trace the maximum amount of workspace that rules have used since CA OPS/MVS was last started. After tracing the levels of workspace use, you may wish to decrease the value of the AOFSIZE parameter if it is set larger than the value of the AOFWSHIGHUSED parameter.

Note: The AOFWSHIGHUSED, AOFWSHIGHUSEDPGM, AOFWSHIGHUSEDRULE, AOFWSHIGHUSEDDATE, and AOFWSHIGHUSEDTIME parameters are display-only parameters.

Default value

307200 (bytes)

Other possible values

Any number of bytes greater than 32768

Set or modify this parameter...

At initialization

Example: AOFSIZE

This function sets the AOF work space to 500000 bytes.

```
OPSPRM('SET','AOFSIZE','500000')
```

More information:

Display-only Parameters (see page 385)

AOFUSEOJOBNAME Parameter

The AOFUSEOJOBNAME parameter determines whether CA OPS/MVS substitutes the values in MSG.OJOBNAME and MSG.OASID for MSG.JOBNAME and MSG.ASID for branch entry WTOs.

Note: Setting the parameter to YES substitutes the values in MSG.OJOBNAME and MSG.OASID for MSG.JOBNAME and MSG.ASID for branch entry WTOs.

Note: The JOBNAME and ASID fields in the OPSLOG also reflects this change.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: AOFUSEOJOBNAME

```
OPSPRM('SET','AOFUSEOJOBNAME', 'YES')
```

AOFWSHIGHUSED Parameter

The AOFWSHIGHUSED parameter displays the largest amount of REXX workspace storage ever used by any rule (and its external subroutines) for the current life of the product or since the parameter value was last reset. Compare this value to the value of the AOFSIZE parameter to check whether the AOF REXX workspaces are either over or under allocated. Over allocating is not critical, but it wastes virtual storage that could be used for some other purpose. Under allocating is a serious problem that results in rule failures.

When you reset this value to zero, the following also happens:

- The AOFWSHIGHUSEDPGM and AOFWSHIGHUSEDRULE display-only parameters are reset to blanks.
- The AOFWSHIGHUSEDDATE and AOFWSHIGHUSEDTIME display-only parameters are reset to zero.

Note: Resetting these values to zero affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: AOFWSHIGHUSED

This function resets the AOFWSHIGHUSED value to zero.

```
OPSPRM('SET','AOFWSHIGHUSED',0)
```

ARMELEMASSOC Parameter

The ARMELEMASSOC parameter specifies the element name of another Automatic Restart Management (ARM)-registered copy of CA OPS/MVS running on the same system that the current copy is temporarily replacing for restart purposes. This feature prevents a restart of the first copy of CA OPS/MVS when a backup or override copy of the product is also registered and running.

Default value

None

Other possible values

1- to 16-character ARM element name

Set or modify this parameter...

At initialization

Example: ARMELEMASSOC

This function overrides the current ARM restart registration of OPSMVSSYSA with the ARM element name specified on the ARMELEMNAME parameter of this copy of CA OPS/MVS, which also has a different subsystem name.

```
OPSPRM('SET','ARMELEMASSOC','OPSMVSSYSA')
```

ARMELEMNAME Parameter

The ARMELEMNAME parameter specifies the ARM element name that CA OPS/MVS uses to register itself with ARM. This name must be unique across the sysplex.

Default value

None

CA OPS/MVS will not register with ARM.

Other possible values

1- to 16-character unique ARM element name that CA OPS/MVS uses to register itself with ARM at initialization. You may define the restart criteria for this element in the ARM policy data set.

Set or modify this parameter...

At initialization

Example: ARMELEMNAME

This function causes CA OPS/MVS to register with ARM at product startup using a sysplex-unique element name that is based on the SMF ID of the current system.

```
OPSPRM('SET','ARMELEMNAME','OPSMVS||OPSINFO('SMFID')')
```

ARMELEMTYPE Parameter

The ARMELEMTYPE parameter specifies the ARM element type that is to relate the ARM element name to a restart order name in the ARM policy. The ARM policy, in turn, determines the sequence in which the elements are restarted. The element type is optional because it is a shorthand method for defining the restart order for groups of element names.

Default value

None

Other possible values

1- to 8-character ARM element type

Set or modify this parameter...

At initialization

Example: ARMELEMTYPE

This function assigns the OPSMVS type to the element name specified by ARMELEMNAME.

```
OPSPRM('SET','ARMELEMTYPE','OPSMVS')
```

ARMRULES Parameter

The ARMRULES parameter determines whether ARM events are evaluated by AOF ARM rules.

Note: The ARMRULES parameter is valid only when the INITARM parameter is set to YES.

Default value

NO

MVS ARM restart events trapped in the IXC_ELEM_RESTART MVS exit are not to be routed to the AOF for ARM rule evaluation.

Other possible values

YES

MVS ARM restart events are to be evaluated by AOF ARM rules.

Set or modify this parameter...

Anytime

Example: ARMRULES

This function activates AOF ARM rule processing.

```
OPSPRM('SET','ARMRULES','YES')
```

CATCHUPREPLYWAIT Parameter

The CATCHUPREPLYWAIT parameter determines the maximum number of seconds AOF initialization is to wait for a reply to either of the catchup prompts - messages OPx0142R or OPx0143R. If more time passes than the interval specified on the CATCHUPREPLYWAIT parameter while waiting for your reply, the default action of NO is taken.

Default value

120

Other possible values

Any value between 1 and 3600 seconds.

Set or modify this parameter...

At initialization

Example: CATCHUPREPLYWAIT

This function set the maximum catchup reply wait time to 300 seconds.

```
OPSPRM('SET','CATCHUPREPLYWAIT','300')
```

EOJRULES Parameter

This parameter determines whether EOJ events are evaluated by AOF EOJ rules.

Note: The EOJRULES parameter is only valid when INITSMF is set to YES. Additionally, the EOSRULES parameter must be set to YES to allow EOJ rules to populate the highest non-zero return code that occurred during execution of the batch job. Additionally, the EOSRULES parameter must be set to YES to allow EOJ rules to populate the highest non-zero return code that occurred during the execution of the batch job.

Default value

NO

End-of-job events trapped by the SMF IEFACTRT exit are not forwarded to AOF for rule evaluation.

Other possible values

YES

End-of-job events are evaluated by AOF EOJ rules.

Set or modify this parameter...

Anytime

Example: EOJRULES

This function activates AOF EOJ rule processing.

```
OPSPRM('SET','EOJRULES','YES')
```

EOSRULES Parameter

This parameter determines whether EOS events are evaluated by AOF EOS rules.

Note: The EOSRULES parameter is only valid when INITSMF is set to YES.

Default value

NO

End-of-step events trapped by the SMF IEFACTRT exit are not forwarded to AOF for rule evaluation.

Other possible values

YES

End-of-step events are evaluated by AOF EOS rules.

Set or modify this parameter...

Anytime

Example: EOSRULES

This function activates AOF EOS rule processing.

```
OPSPRM('SET','EOSRULES','YES')
```

EXECQUE Parameter

The EXECQUE parameter determines the size, by number of queue elements, of the execution queue of the AOF. CA OPS/MVS makes an entry in this queue whenever an ADDRESS AOF command is issued. For more information on ADDRESS AOF commands, see the *AOF Rules User Guide*.

Default value

256

Other possible values

Any number from 1 to 32768

Set or modify this parameter...

At initialization

Example: EXECQUE

This function sets the size of the execution queue to 250 elements.

```
OPSPRM('SET','EXECQUE','250')
```

INITARM Parameter

The INITARM parameter determines whether CA OPS/MVS installs the z/OS dynamic exit module OPMVAREX at the IXC_ELEM_RESTART MVS exit point to intercept ARM restart events and to send the events to all copies of CA OPS/MVS (on the same system) that have the ARMRULES parameter set to YES.

Default value

NO

MVS ARM events are not to be trapped and AOF ARM rules are not to be executed.

Other possible values

YES

MVS ARM restart events are to be trapped and sent to all copies of CA OPS/MVS that have the ARMRULES parameter set to YES.

Set or modify this parameter...

At initialization

Example: INITARM

This function installs the z/OS dynamic exit module to enable the trapping of ARM restart events.

```
OPSPRM('SET','INITARM','YES')
```

INITSMF Parameter

This parameter determines whether CA OPS/MVS SMF exits required to generate TLM, EOS, and EOJ AOF events are installed at product initialization. One copy of each SMF exit services all copies of CA OPS/MVS on a system. The exits are installed dynamically using the z/OS dynamic exit facility.

Default value

NO

The product SMF exits are not installed. TLM, EOS, and EOJ events are not generated unless another copy of the product has already installed the SMF exits.

Other possible values

YES

Install the product SMF exits if they are not already installed.

Set or modify this parameter...

At initialization

Example: INITSMF

This function installs the CA OPS/MVS dynamic SMF exits at product initialization.

```
OPSPRM('SET','INITSMF','YES')
```

INITWME Parameter (Obsolete)

The INITWME parameter is obsolete, and will be removed in a future release.

LXCRULES Parameter

This parameter determines whether Linux Connector component unsolicited Linux and VM message events are evaluated by AOF API rules.

Note: This parameter is only valid when the INITLXC parameter is set to YES.

Default value

NO

Linux Connector unsolicited message events are not forwarded to the AOF for rule evaluation.

Other possible values

YES

Linux Connector unsolicited message events are evaluated by AOF API rules.

Set or modify this parameter...

Anytime

Example: LXCRULES

This function activates AOF API rule processing Linux Connector messages.

```
OPSPRM('SET','LXCRULES','YES')
```

MVSSYSn Parameter

There are thirty one MVSSYS*n* parameters named MVSSYS1 to MVSSYS31. Each MVSSYSn parameter can specify an eight-character system name. When the AOF MESSAGES parameter is set to MVSLIST, CA OPS/MVS will receive and process local messages and messages from systems specified on the MVSSYS*n* parameters. See the AOFMESSAGES parameter for more information.

Default value: ' '

The default value is a blank which indicates that no system name is specified on this parameter.

Other possible values

8-character system name

This value indicates that CA OPS/MVS should receive and process messages imported from the specified system if the AOFMESSAGES parameter is set to MVSLIST.

Set or modify this parameter...

Anytime

Example: MVSSYS*n* Parameter

These functions tell CA OPS/MVS to receive and process messages imported from MVSSYSAA and MVSSYSBB if AOFMESSAGES is set to MVSLIST:

```
OPSPRM('SET','MVSSYS1','MVSSYSAA')

OPSPRM('SET','MVSSYS2','MVSSYSBB')
```

Usage Recommendations

CA recommends the following regarding the use of the MVSSYS*n* parameters:

- Do not 'skip' MVSSYS*n* parameters when filling in system names. For example, if you are specifying three system names, use the MVSSYS1, MVSSYS2, and MVSSYS3 parameters. While the function will still work correctly if you skip parameters (use MVSSYS1, MVSSYS10, and MVSSYS20 for example), the best performance will be achieved by filling the MVSSYS*n* parameters consecutively.
- Use the AOFMESSAGES parameter to toggle system filtering on and off. The system names specified on the MVSSYS*n* parameters persist regardless of the AOFMESSAGES value. However, the system names are not used to filter message processing unless AOFMESSAGES is set to MVSLIST. The MVSSYS*n* parameters are ignored for any other AOFMESSAGE value.

Important! If you specify a value of MVSLIST for AOFMESSAGES and do not specify any system names on MVSSYS*n* parameters, CA OPS/MVS will not receive and process imported messages. Only local messages will be processed by CA OPS/MVS.

SSEXEXITHICOUNT Parameter

The SSEXEXITHICOUNT parameter displays the highest number of process blocks from the main process block pool (see the PROCESS parameter) that were simultaneously in use since this CA OPS/MVS subsystem was last started or since this parameter value was last reset.

When you reset this value to zero, the SSEXEXITHIDATE and SSEXEXITHITIME display-only parameters are reset to zero.

Note: Resetting these values to zero affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default value

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: SSEXEXITHICOUNT

This function resets the SSEXEXITHICOUNT value to zero.

```
OPSPRM('SET','SSEXEXITHICOUNT',0)
```

TLMRULES Parameter

This parameter determines whether TLM events are evaluated by AOF TLM rules.

Default value

NO

Time limit excession events trapped by the SMF IEFUTL exit are not forwarded to AOF for rule evaluation.

Other possible values

YES

Time limit excession events are evaluated by AOF TLM rules.

Set or modify this parameter...

Anytime

Example: TLMRULES

This function activates AOF TLM rule processing.

```
OPSPRM('SET','TLMRULES','YES')
```

USSRULES Parameter

This parameter determines whether USS message events are evaluated by AOF USS rules.

Note: This parameter is only valid when the INITUSS parameter is set to YES.

Default value

NO

UNIX System Services (USS) message events trapped by CA Common Services (CCS) for z/OS are not forwarded to the AOF for rule evaluation.

Other possible values

YES

UNIX System Services message events are evaluated by AOF USS rules.

Set or modify this parameter...

Anytime

Example: USSRULES

This function activates AOF USS rule processing.

```
OPSPRM('SET','USSRULES','YES')
```

ECF Parameters

The parameters described in the following sections control Enhanced Console Facility (ECF) operation.

ECFCHAR Parameter

The ECFCHAR parameter defines the one- to three-character string that the operator uses to prefix commands that are to be sent to the ECF from a console. For example, if the value of the ECFCHAR parameter were ECF, the following commands issued at an MCS console would be routed to the ECF:

```
ECFLOGON USER01/PASSWORD
ECFLOGOFF
```

Note: When logging on to the ECF, the LOGON keyword must immediately follow the ECF command prefix (that is, the value of ECFCHAR). Intervening blanks are not allowed.

For information about how CA OPS/MVS defines ECFCHAR to the z/OS Command Prefix Facility (CPF), see the *Administration Guide*. For general rules for defining CPF strings, see Characters That Can Be Used in z/OS Commands in the chapter "Using This Reference."

Note: CA OPS/MVS automatically defines the command prefix to the CPF during product initialization and attempts to delete the prefix from the CPF during termination. If some other subsystem is already using this prefix, then the prefix is not defined to the CPF and the following message is issued during product initialization:

```
OPS0100W CPF DEFINE OF ECFCHAR FAILED, RC=X'00000008', REASON CODE=X'00000008'
```

Reason codes of X'0000000C' and X'00000010' are also indicators of this type of problem since the prefix specified may be either a subset or superset of a prefix that is already defined.

If you change the value of ECFCHAR while CA OPS/MVS is active, then the CPF prefixes are **not** updated and the following messages are issued during product termination. Therefore, we strongly recommend that you do not modify ECFCHAR after product initialization.

Note: In a future release, ECFCHAR may change so that it can be set only during product initialization.

```
OPS0100W CPF DELETE OF ECFCHAR FAILED, RC=X'00000008', REASON CODE=X'00000004'
```

Important! The ECFCHAR parameter can only contain characters from the table shown in Characters That Can Be Used In z/OS Commands in the chapter "Using This Reference."

Default value

The question mark (?)

Other possible values

Any unique, 1- to 3-character string

Specify a value other than the default if you already use the question mark for some other purpose, or to allow the operator to talk to a test version of ECF.

Set or modify this parameter...

Anytime

Important! We strongly recommend that you do not modify ECFCHAR after product initialization.

Example: ECFCHAR

This function sets the ECF command prefix to ECF.

```
OPSPRM('SET','ECFCHAR','ECF')
```

ECFLOGON Parameter

The ECFLOGON parameter determines whether CA OPS/MVS accepts ECF logon requests.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: ECFLOGON

This function prevents users from logging onto ECF.

```
OPSPRM('SET','ECFLOGON','NO')
```

ECFMSTR Parameter

The ECFMSTR parameter determines whether CA OPS/MVS allows ECF logons to the master subsystem.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: ECFMSTR

This function prevents users from logging on to the master subsystem through ECF.

```
OPSPRM('SET','ECFMSTR','NO')
```

ECFOUTLIM Parameter

The ECFOUTLIM parameter sets the maximum number of messages the ECF can generate during every interval specified by the ECFWAIT parameter. For more information, see the description of the ECFWAIT Parameter in this chapter.

Default value

100 (message lines)

Other possible values

Any number of message lines

Set or modify this parameter...

Anytime

Example: ECFOUTLIM

This function sets the number of messages per ECFWAIT interval to 1.

```
OPSPRM('SET','ECFOUTLIM','1')
```

With ECFOUTLIM set as shown above and with the ECFWAIT parameter set to 100 centiseconds, the OPSECF address space sends only one command output message every second. ECFOUTLIM also lets you tune how ECF displays console messages; for instance, in this example, the console may show messages from other address spaces interspersed with the ECF output.

ECFPARM Parameter

The ECFPARM parameter adds ECF parameter strings to the internal start commands of the address space. ECFPARM also allows you to set an ECF parameter string to a JCL keyword=value string to change the JCL for the ECF started task without modifying the actual procedure.

Default value

A null string

Other possible values

Any valid ECF parameter string

Set or modify this parameter...

Anytime

Example: ECFPARM

This function adds the string UNIT=3390 to internal start commands.

```
OPSPRM('SET','ECFPARM','UNIT=3390')
```

ECFSECURITY Parameter

The ECFSECURITY parameter determines the level of security for ECF.

Default value

CHECKPASSWORD

This value causes CA OPS/MVS to check both the TSO user ID and the password specified in the ECF logon command.

Other possible values

- NOSECURITY

  This value makes any console eligible to log onto ECF; CA OPS/MVS performs no TSO user ID or password checking.

- CHECKUSERID

  This value causes CA OPS/MVS to verify the TSO user ID specified in the ECF logon command. The specified password is ignored.

Notes:

- The CHECKUSERID and default CHECKPASSWORD settings are valid only when you start the ECF under JES or when a security product is active on your system.
- If you specify either CHECKPASSWORD or CHECKUSERID, the ECF session has only the privileges of the TSO user ID.

Set or modify this parameter...

At initialization

Example: ECFSECURITY

This function allows any user ID and password to log onto ECF.

```
OPSPRM('SET','ECFSECURITY','NOSECURITY')
```

ECFSTC Parameter

The ECFSTC parameter names the started task that supports an ECF user. Use ECFSTC to rename the ECF started task to meet your initialization naming conventions or to activate a test copy of ECF.

Default value

OPSECF

Other possible values

Any valid started task name

Set or modify this parameter...

Anytime

Example: ECFSTC

This function sets the ECF started task name to ECFTEST.

```
OPSPRM('SET','ECFSTC','ECFTEST')
```

ECFWAIT Parameter

The ECFWAIT parameter specifies, in centiseconds, how long ECF pauses after reaching the message limit set by the ECFOUTLIM parameter before it resumes message processing. For more information, see the description of the ECFOUTLIM Parameter in this chapter.

Default value

50 (centiseconds)

50 centiseconds = 1/2 second

Other possible values

Any number of centiseconds

Set or modify this parameter...

Anytime

Example: ECFWAIT

This function sets the wait interval to 30 centiseconds.

```
OPSPRM('SET','ECFWAIT','30')
```

EPI Parameters

The EPICMDLOGGING and EPIEXTENDEDATTR parameters affect the CA OPS/MVS EPI operations.

EPICMDLOGGING Parameter

This parameter determines whether CA OPS/MVS logs all EPI commands to the OPSLOG or suppresses them from appearing in OPSLOG.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: EPICMDLOGGING

This function suppresses the logging of EPI commands.

```
OPSPRM('SET','EPICMDLOGGING','NO')
```

EPIEXTENDEDATTR Parameter

This parameter specifies whether CA OPS/MVS calls the EPI extended attribute routine, which allows users to modify the data stream before EPI processes it.

Default value

OFF

Using the default value conserves system overhead.

Other possible values

ON

Set or modify this parameter...

Anytime

Example: EPIEXTENDEDATTR

This function enables CA OPS/MVS to call the extended attribute routine.

```
OPSPRM('SET','EPIEXTENDEDATTR','ON')
```

OSF Parameters

The parameters described in the following sections affect operation of the CA OPS/MVS OSF feature.

BEGINCMD Parameter

This parameter specifies the text of the first OSF TSO command that CA OPS/MVS sends to the OSF TSO execute queue after it completes its internal initialization (this does not mean that the AOF is completely active). The command is processed when the first OSF TSO server is initialized and ready to execute transactions.

Default value

OI OPSTART2

This value indicates the name of the default CA OPS/MVS startup OPS/REXX program supplied in the SYS1.OPS.CCLXEXEC data set.

Other possible values

Any valid TSO command, expressed as a text string containing up to 128 bytes

Set or modify this parameter...

At initialization

Example: BEGINCMD

This function causes CA OPS/MVS to execute a startup REXX EXEC called KICKOFF.

```
OPSPRM('SET','BEGINCMD','KICKOFF')
```

OSFALLOWED Parameter

The OSFALLOWED parameter determines whether a source other than an MCS console can use the OSF command character prefix specified by the OSFCHAR parameter to send a command to the OSF server for execution. An OSF server is a started task that can execute a TSO transaction, not unlike a TSO logon ID. This parameter applies only to OSF TSO servers.

For more information about OSF servers, consult:

- The comment section of the OPSOSF STC JCL member of the CA OPS/MVS CNTL library
- The *Administration Guide*
- The description of the OSFCHAR Parameter in this chapter

Default value

YES

This value allows any address space to use the OSF command character to send a command to the OSF server.

Other possible values

NO

The OSF command character can be used to send a command to the OSF server only from an MCS console.

Set or modify this parameter...

Anytime

Example: OSFALLOWED

This function allows the OSF server to receive only commands issued from a console.

```
OPSPRM('SET','OSFALLOWED','NO')
```

OSFALLOWRESTART Parameter

When an OSF server fails during initialization, the setting of the OSFALLOWRESTART parameter determines whether CA OPS/MVS attempts to restart it. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value

NO

CA OPS/MVS does not attempt to restart servers that failed during initialization.

Other possible values

YES

At 30-second intervals, CA OPS/MVS attempts to restart servers that failed during initialization.

Set or modify this parameter...

Anytime

Example: OSFALLOWRESTART

This function tells CA OPS/MVS to automatically attempt to restart a failed OSF server.

```
OPSPRM('SET','OSFALLOWRESTART','YES')
```

OSFCHAR Parameter

The OSFCHAR parameter defines the one- to three-character string that you use to prefix commands that are to be issued from a console directly to an OSF TSO server. For example, if the value of the OSFCHAR parameter were OSF, the following commands issued at an MCS console would be routed to the OSF:

```
OSFTIME
OSFLISTD 'SYS1.PROCLIB'
OSFOI PROGRAMA ARG1
```

The OSFALLOWED parameter determines whether a source other than an MCS console can use the OSF command prefix to send a command to the OSF server for execution.

For information about how CA OPS/MVS defines OSFCHAR to the z/OS Command Prefix Facility (CPF), see the *Administration Guide*. You should also see OSFSYSPLEXCMD Parameter in this chapter. For general rules for defining CPF strings, see Characters That Can Be Used in z/OS Commands in the chapter “Using This Reference.”

Note: CA OPS/MVS automatically defines the command prefix to the CPF during product initialization and attempts to delete the prefix from the CPF during termination. If some other subsystem is already using this prefix, then the prefix is not defined to the CPF and the following message is issued during product initialization:

```
OPS0100W CPF DEFINE OF OSFCHAR FAILED, RC=X'00000008', REASON CODE=X'00000008'
```

Reason codes of X'0000000C' and X'00000010' are also indicators of this type of problem since the prefix specified may be either a subset or superset of a prefix that is already defined.

If you change the value of OSFCHAR while CA OPS/MVS is active, then the CPF prefixes will *not* be updated and the following message will be issued during product termination. Therefore, we strongly recommend that you do not modify OSFCHAR after product initialization.

Note: In a future release, OSFCHAR may change so that it can be set only during product initialization.

```
OPS0100W CPF DELETE OF OSFCHAR FAILED, RC=X'00000008', REASON CODE=X'00000004'
```

Important! The OSFCHAR parameter can only contain characters from the table shown in Characters That Can Be Used In z/OS Commands in the chapter “Using This Reference.”

Default value

The exclamation point (!)

Other possible values

Any unique, 1- to 3-character string

Change the value of OSFCHAR if you already use the ! character for something else, or if you want to allow the operator to talk to a test version of the server.

Set or modify this parameter...

Anytime

Important! We strongly recommend that you do not modify OSFCHAR after product initialization.

Example: OSFCHAR

This function changes the OSF command prefix to OSF.

```
OPSPRM('SET','OSFCHAR','OSF')
```

OSFCONSOLE Parameter

The OSFCONSOLE parameter names the user ID that OSF TSO servers use for commands issued with the OSF command character.

Note: The value of the OSFCONSOLE parameter is valid only if a security product is active on your system, and the value of the OSFSECURITY parameter is CHECKUSERID.

Default value

OPSOSF

Other possible values

Any valid user ID

Set or modify this parameter...

Anytime

Example: OSFCONSOLE

This function changes the user ID to OSFCDCON.

```
OPSPRM('SET','OSFCONSOLE','OSFCDCON')
```

OSFCPU Parameter

The OSFCPU parameter limits the amount of CPU time that a single OSF TSO server transaction can use.

Default value

15 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFCPU

This function allows a server transaction to use up to 30 seconds of CPU time.

```
OPSPRM('SET','OSFCPU','30')
```

Note: Setting OSFCPU to a value higher than OSFRUN will in effect negate the use of this parameter. For a discussion on the practical limits for OSF TSO transactions, see OSFRUN Parameter in this chapter.

OSFDORM Parameter

The OSFDORM parameter determines, in seconds, how long OSF TSO servers can remain dormant before they are automatically terminated. This parameter takes effect only if the number of active OSF TSO servers is greater than the value of the OSFMIN parameter. This parameter then decrements the servers one per OSFDORM dormant period. For more information, see OSFMIN Parameter (see page 184).

Default value

60

Other possible values

Any number of seconds from 60 through 16777216

Set or modify this parameter...

Anytime

Example: OSFDORM

This function allows 300 seconds, or 5 minutes, of OSF TSO server inactivity before a server is terminated.

```
OPSPRM('SET','OSFDORM','300')
```

OSFEPRIV Parameter

The OSFEPRIV parameter allows you to control the extended private area region limit for OSF server address spaces. By default this value is zero, which results in the OSF region size being controlled by your system installation exit (IEFUSI), or the IBM default value 32 MB. This parameter applies to OSF TSL, TSO, and TSP servers.

Changing the value of this parameter does not affect the region size for OSF address spaces that are already active. If you change this value after product initialization, you must terminate the servers to have the new value take effect.

Default value

0 KB

Other possible values

32768 KB

Set or modify this parameter...

Anytime

Example: OSFEPRIV

Set the extended private area limit for all OSF servers that start after this change to 32 MB.

```
OPSPRM('SET','OSFEPRIV','32768K')
```

OSFGETJOBID Parameter

The OSFGETJOBID parameter determines whether servers started under the master subsystem try to obtain a job ID from JES when JES starts. Before JES is active, you cannot issue a valid SUBMIT command because JES SYSOUT data sets cannot be allocated. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: OSFGETJOBID

This function prevents OSF servers from obtaining job IDs.

```
OPSPRM('SET','OSFGETJOBID','NO')
```

OSFJOBLOG Parameter

The OSFJOBLOG parameter determines whether servers started under the master subsystem create a JES job log. If you change the value of OSFJOBLOG from the default (NO) to YES, a job log is created the next time a server obtains a new JES job ID. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value

NO

Do not allocate a JES job log.

Other possible values

YES

Set or modify this parameter...

Anytime

Example: OSFJOBLOG

This function enables OSF servers to create job logs.

```
OPSPRM('SET','OSFJOBLOG','YES')
```

OSFLOGONSOURCE Parameter

The OSFLOGONSOURCE parameter allows an installation to override the security product source (or terminal ID) string with which OSF TSO servers log on to the security system.

Default value

SYSTEM

Other possible values

Any valid 1- to 8-character security product source (terminal ID) string

Set or modify this parameter...

Anytime

Example: OSFLOGONSOURCE

This function sets the OSF TSO security source string to OPSSYST.

```
OPSPRM('SET','OSFLOGONSOURCE','OPSSYST')
```

OSFMAX Parameter

The OSFMAX parameter sets the maximum number of OSF TSO servers that can be active at any time.

Default value

10

Other possible values

Any amount of OSF TSO servers up to 30, but at least as many as the number of servers specified for the OSFMIN parameter. If you attempt to set the OSFMAX parameter value lower than the OSFMIN value, CA OPS/MVS sets OSFMAX equal to OSFMIN.

Set or modify this parameter...

Anytime

Example: OSFMAX

This function sets 12 as the maximum number of OSF servers.

```
OPSPRM('SET','OSFMAX','12')
```

OSFMIN Parameter

The OSFMIN parameter sets the minimum number of OSF TSO servers that can be active at any time. If the OSFMIN value exceeds the current number of active OSF TSO servers, CA OPS/MVS automatically starts additional servers.

We strongly recommend that OSFMIN be set to a value that is high enough to process almost all (about 99 percent) of your typical asynchronous automation work. For more information, see Regulating OSF Servers in the chapter “Technical Notes” in the *Administration Guide*.

Default value

4

Other possible values

Any amount of servers up to 30, but fewer than the number of servers specified for the OSFMAX parameter. If you set the OSFMIN parameter value higher than the OSFMAX value, CA OPS/MVS sets OSFMIN equal to OSFMAX.

Set or modify this parameter...

Anytime

Example: OSFMIN

This function sets 7 as the minimum number of OSF servers.

```
OPSPRM('SET','OSFMIN','7')
```

OSFMSGID Parameter

The OSFMSGID parameter lets you specify whether the default for server address spaces is PROFILE MSGID or PROFILE NOMSGID. This value takes effect when a rule or a non-TSO address space sends work to a server; when a TSO address space sends work, the PROFILE settings of the TSO user are copied to the server. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value

NO

Equivalent to PROFILE NOMSGID

Other possible values

YES

Equivalent to PROFILE MSGID

Set or modify this parameter...

Anytime

Example: OSFMSGID

This function specifies PROFILE MSGID as the default.

```
OPSPRM('SET','OSFMSGID','YES')
```

OSFOUTLIM Parameter

The OSFOUTLIM parameter limits the number of console messages a transaction running under an OSF TSO server can produce. These include:

- SAY statements from REXX programs
- Output from the CLIST command WRITE
- Output of an OPSCMD command processor issued outside a REXX program

Default value

1000 (lines)

Other possible values

Any number of lines greater than or equal to 0

Set or modify this parameter...

Anytime

Example: OSFOUTLIM

This function limits the number of output lines to 500.

```
OPSPRM('SET','OSFOUTLIM','500')
```

OSFPARM Parameter

The OSFPARM parameter lets you add a keyword=option string that CA OPS/MVS includes in the z/OS START command for the OSF started task. OSFPARM also allows you to override the corresponding substitution parameter in the cataloged procedure. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value

No default

Other possible values

Any valid keyword=option string

Set or modify this parameter...

Anytime

Example: OSFPARM

This function adds the string UNIT=3390 to the START command.

```
OPSPRM('SET','OSFPARM','UNIT=3390')
```

OSFPRODUCT Parameter

The OSFPRODUCT parameter defines the user ID for ADDRESS TSO commands generated in AOF rules (except for request rules). This parameter applies to OSF TSL, TSO, and TSP servers.

Depending on the level of privileges assigned to the user ID, the ADDRESS TSO commands may fail.

Note: The value of the OSFPRODUCT parameter is valid only if a security product is active on your system, and the value of the OSFSECURITY parameter is CHECKUSERID.

Default value

OPSOSF

Other possible values

Any valid user ID for OSF

Set or modify this parameter...

Anytime

Example: OSFPRODUCT

This function sets the user ID for ADDRESS TSO commands to OSF.

```
OPSPRM('SET','OSFPRODUCT','OSF')
```

OSFQADD Parameter

The OSFQADD parameter sets the threshold CA OPS/MVS uses to determine whether more OSF TSO servers are started. The algorithm that is used to add servers depends on the OSFQADDPROCESS parameter. If OSFQADDPROCESS=AVAIL a new server is added, when the number of commands in the OSF TSO execution queue is higher than OSFQADD and OSFMIN but lower than OSFMAX. When OSFQADDPROCESS= QADD a new server is added when the queue depth equals OSFQADD+1 + (n * OSFQADD) until OSFMAX is reached. Where ‘n’ equals the number of dynamically added servers.

Default value

8

Other possible values

Any number of queued commands greater than or equal to 0

Set or modify this parameter...

Anytime

Example: OSFQADD

This function sets the OSF TSO execution queue threshold to five commands.

```
OPSPRM('SET','OSFQADD','5')
```

OSFQADDPROCESS Parameter

The OSFQADDPROCESS parameter determines what process is used to add servers.

Default value

QADD

Add servers every time the queue depth is OSFQADD+1 + (n * OSFQADD) until OSFMAX is reached. Where 'n' equals the number of dynamically added servers. For example: If OSFMIN=2 OSFQADD=2 and OSFMAX=5 a new server would be added when the queue reached a depth of 3, 5, and 7. This algorithm is used to optimize performance.

Other possible values

AVAIL

Add servers for every request in the queue that exceeds the OSFQADD depth, until OSFMAX is reached. When the system is under heavy workload, coding this parameter could result in performance issues.

Set or modify this parameter...

Anytime

Example: OSFQADDPROCESS

This function request sets the OSFQADDPROCESS parameter.

```
OPSPRM_SET("OSFQADDPROCESS","QADD")
```

OSFQUE Parameter

The OSFQUE parameter specifies the maximum number of queued commands the OSF TSO execute queue can hold. This queue is where CA OPS/MVS sends TSO commands to be executed in OSF TSO servers. The OSF execute processor dispatches these commands to OSF TSO servers as the servers become available to process work.

Default value

1700 (commands)

Other possible values

Any number from 1 to 32768

Set or modify this parameter...

At initialization

Example: OSFQUE

This function sets the size of the OSF TSO execution queue to 2,000 commands.

```
OPSPRM('SET','OSFQUE','2000')
```

OSFRECYCLE Parameter

The OSFRECYCLE parameter determines the number of completed OSF server transactions that can run in a server before CA OPS/MVS recycles the server. This parameter applies to OSF TSL, TSO, and TSP servers.

Typically, you should let the value of the OSFRECYCLE parameter default to 0, so that no recycling is performed. However, under some conditions, private area storage in the server address space is not correctly freed at the end of each server transaction. When this happens, storage-related abends in other transactions may result. Under these conditions, you may decide to set the OSFRECYCLE parameter to a reasonable value that causes CA OPS/MVS to recycle the servers after a particular number of transactions. Doing so alleviates the residual storage problems.

Default value

0

Other possible values

Any positive integer value between 1 and 2147483647

Set or modify this parameter...

Anytime

Example: OSFRECYCLE

This function causes CA OPS/MVS to recycle OSF servers after each 1000 transactions.

```
OPSPRM('SET','OSFRECYCLE','1000')
```

OSFRUN Parameter

The OSFRUN parameter determines how long CA OPS/MVS allows a TSO transaction to execute in an OSF TSO server. This is an elapsed time limit; in comparison, the OSFCPU parameter limits execution time.

When any OSF TSO server transaction exceeds the time limit set by OSFRUN, CA OPS/MVS terminates the server with the hung or looping transaction and starts another server to accept new commands.

Default value

120 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFRUN

This function sets a 60-second time limit for TSO transactions.

```
OPSPRM('SET','OSFRUN','60')
```

Note: The OSF TSO servers are intended to execute relatively short pieces of asynchronous automation. While this parameter does not enforce a maximum transaction runtime value, we strongly recommend that this value not be set higher than 300 seconds (5 minutes). Long-running transactions should be run in OSF TSL servers or in a separate address spaces (either started tasks or batch jobs). Using the OSF TSO servers to execute long running transactions may result in other automation that is time-critical to not execute in a timely fashion, since all of the OSF TSO servers may be busy executing long-running work. It may also result in the OSF TSO execute queue filling up, which in turn may result in transactions being lost.

OSFSECURITY Parameter

The OSFSECURITY parameter determines whether the OSF operates with or without security checking. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value:

NOSECURITY

This value allows the OSF to operate without security checking in a trial or test situation. If your system has no security product installed, CA OPS/MVS ignores the OSFSECURITY parameter setting and defaults it to NOSECURITY.

Other possible values:

CHECKUSERID

Most sites use this value for the OSFSECURITY parameter. If you specify this value, CA OPS/MVS checks that the issuer has been defined to the security product on the system.

CA OPS/MVS uses the security privileges assigned to the issuer of the command, as follows:

- For work sent to servers from rules (other than request rules), the security privileges are those assigned to the user ID that is specified as the value of the OSFPRODUCT parameter. For a description of OSFPRODUCT, see OSFPRODUCT Parameter in this chapter.
- For work sent to servers from request rules, the security privileges are those assigned to the TSO user ID.
- For work sent from consoles using the OSF command character (as defined by the OSFCHAR parameter), the security privileges are those assigned to the user ID that is specified as the value of the OSFCONSOLE parameter. For more information, see OSFCONSOLE Parameter in this chapter. For a description of the OSFCHAR parameter, see OSFCHAR Parameter in this chapter.
- For work sent to servers from TSO users using the ADDRESS OSF host command environment, the security privileges are those assigned to the TSO user ID.

Set or modify this parameter:

At initialization

Example: OSFSECURITY

This function tells CA OPS/MVS to operate the OSF with security checking.

```
OPSPRM('SET','OSFSECURITY','CHECKUSERID')
```

OSFSTC Parameter

The OSFSTC parameter specifies the name of the OSF cataloged procedure or started task, allowing you to use the naming conventions of your site. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value

OPSOSF

Other possible values

Any valid started task name

Set or modify this parameter...

At initialization

Example: OSFSTC

This function specifies OSFTASK as the name of the OSF started task.

```
OPSPRM('SET','OSFSTC','OSFTASK')
```

OSFSWAPPABLE Parameter

The OSFSWAPPABLE parameter determines whether all servers started after this parameter is set are swappable. This parameter applies to OSF TSL, TSO, and TSP servers.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: OSFSWAPPABLE

This function makes OSF servers non-swappable.

```
OPSPRM('SET','OSFSWAPPABLE','NO')
```

OSFSYSPLEXCMD Parameter

The OSFSYSPLEXCMD parameter allows you to specify whether the OSFCHAR prefix string should be defined to the z/OS Command Prefix Facility (CPF) with a scope of SYSTEM or SYSPLEX.

Default value

NO

The OSFCHAR prefix is defined to the CPF on the local system only.

Other possible values

YES

The OSFCHAR prefix is defined with a scope of all systems in the sysplex and should, therefore, be a unique value in the sysplex.

Note: Specifying a value of YES on a system that is not part of the sysplex is meaningless and is equivalent to a value of NO.

Set or modify this parameter...

At initialization

Example: OSFSYSPLEXCMD

This function defines the OSFCHAR prefix string to the CPF with a scope of SYSPLEX.

```
OPSPRM('SET','OSFSYSPLEXCMD','YES')
```

OSFTRANSSMFREC Parameter

The OSFTRANSSMFREC parameter determines whether SMF records are created to record each individual OSF transaction. This parameter applies to OSF TSL, TSO, and TSP servers.

When the value of the OSFTRANSSMFREC parameter is YES, a new SMF record (of subtype 7) is created to record the activity of each individual server transaction. Among other things, the SMF record contains this information:

- The length of time the transaction spent on the OSF execute queue prior to being dispatched to a server
- The length of time the transaction spent in the server
- The CPU time (TCB + SRB), I/O count, and number of output lines for the transaction

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: OSFTRANSSMFREC

This function causes SMF records to be created for OSF transactions.

```
OPSPRM('SET','OSFTRANSSMFREC','YES')
```

OSFTSLCPU Parameter

The OSFTSLCPU parameter limits the amount of CPU time that a single OSF TSL server transaction can use.

Note: TSL servers are intended for longer running transactions and therefore the default value is set higher than for TSO and TSP servers.

Default value

150 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFTSLCPU

This function allows a TSL server transaction to use up to 30 seconds of CPU time.

```
OPSPRM('SET','OSFTSLCPU','30')
```

Note: Setting OSFTSLCPU to a value higher than OSFTSLRUN will in effect negate the use of this parameter. For a discussion on the practical limits for OSF TSL transactions, see OSFTSLRUN Parameter in this chapter.

OSFTSLDORM Parameter

The OSFTSLDORM parameter determines, in seconds, how long OSF TSL servers can remain dormant before they are automatically terminated. This parameter takes effect only if the number of active OSF TSL servers is greater than the value of the OSFTSLMIN parameter. This parameter then decrements the servers one per OSFTSLDORM dormant period. For more information, see OSFTSLMIN Parameter (see page 198).

Default value

60

Other possible values

Any number of seconds from 60 through 16777216

Set or modify this parameter...

Anytime

Example: OSFTSLDORM

This function allows 300 seconds, or 5 minutes, of OSF TSL server inactivity before a server is terminated.

```
OPSPRM('SET','OSFTSLDORM','300')
```

OSFTSLMAX Parameter

The OSFTSLMAX parameter sets the maximum number of OSF TSL servers that can be active at any time.

Default value

0

Other possible values

Any amount of OSF TSL servers up to 30, but at least as many as the number of servers specified for the OSFTSLMIN parameter. If you attempt to set the OSFTSLMAX parameter value lower than the OSFTSLMIN value, CA OPS/MVS sets OSFTSLMAX equal to OSFTSLMIN.

Set or modify this parameter...

Anytime

Example: OSFTSLMAX

This function sets 12 as the maximum number of OSF TSL servers.

```
OPSPRM('SET','OSFTSLMAX','12')
```

OSFTSLMIN Parameter

The OSFTSLMIN parameter sets the minimum number of OSF TSL servers that can be active at any time. If the OSFTSLMIN value exceeds the current number of active OSF TSL servers, CA OPS/MVS automatically starts additional servers.

We strongly recommend that OSFTSLMIN be set to a value that is high enough to process almost all (about 99 percent) of your typical long-running asynchronous automation work. For more information, see Regulating OSF Servers in the chapter "Technical Notes" in the *Administration Guide*.

Default value

0

Other possible values

Any amount of OSF TSL servers up to 30, but fewer than the number of servers specified for the OSFTSLMAX parameter. If you set the OSFTSLMIN parameter value higher than the OSFTSLMAX value, CA OPS/MVS sets OSFTSLMIN equal to OSFTSLMAX.

Set or modify this parameter...

Anytime

Example: OSFTSLMIN

This function sets 2 as the minimum number of OSF TSL servers.

```
OPSPRM('SET','OSFTSLMIN','2')
```

OSFTSLOUTLIM Parameter

The OSFTSLOUTLIM parameter limits the number of console messages a transaction running under an OSF TSL server can produce. These include:

- SAY statements from REXX programs
- Output from the CLIST command WRITE
- Output of an OPSCMD command processor issued outside a REXX program

Default value

5000 (lines)

Other possible values

Any number of lines greater than or equal to 0

Set or modify this parameter...

Anytime

Example: OSFTSLOUTLIM

This function limits the number of output lines to 500.

```
OPSPRM('SET','OSFTSLOUTLIM','500')
```

OSFTSLQADD Parameter

The OSFTSLQADD parameter sets the threshold CA OPS/MVS uses to determine whether more OSF TSL servers are starting. The algorithm that is used to add servers depends on the OSFQADDPROCESS parameter. If OSFQADDPROCESS=AVAIL a new server is added when, the number of commands in the OSF TSL execution queue is higher than OSFQADD and OSFMIN but lower than OSFMAX. If OSFQADDPROCESS= QADD a new server is added when the queue depth equals OSFTSLQADD+1 + (n * OSFTSLQADD) until OSFTSLMAX is reached. Where 'n' equals the number of dynamically added servers.

Default value

0

Other possible values

Any number of queued commands greater than or equal to 0

Set or modify this parameter...

Anytime

Example: OSFTSLQADD

This function sets the OSF TSL execution queue threshold to 5 commands.

```
OPSPRM('SET','OSFTSLQADD','5')
```

OSFTSLQUE Parameter

The OSFTSLQUE parameter specifies the maximum number of queued commands the OSF TSL execute queue can hold. This queue is where CA OPS/MVS sends TSO commands to be executed in OSF TSL servers. The OSF TSL execute processor dispatches these commands to OSF TSL servers as the servers become available to process work.

Default value

1700 (commands)

Other possible values

Any number from 1 to 32768

Set or modify this parameter...

At initialization

Example: OSFTSLQUE

This function sets the size of the OSF TSL execution queue to 2,000 commands.

```
OPSPRM('SET','OSFTSLQUE','2000')
```

OSFTSLRUN Parameter

The OSFTSLRUN parameter determines how long CA OPS/MVS allows a TSO transaction to execute in an OSF TSL server. This is an elapsed time limit; in comparison, the OSFTSLCPU parameter limits execution time.

When any OSF TSL server transaction exceeds the time limit set by OSFTSLRUN, CA OPS/MVS terminates the server with the hung or looping transaction and starts another server to accept new commands.

Note: TSL servers are intended for longer running transactions and therefore the default value is set higher than for TSO and TSP servers.

Default value

1200 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFTSLRUN

This function sets a 60-second time limit for TSL transactions.

```
OPSPRM('SET','OSFTSLRUN','60')
```

OSFTSLWAIT Parameter

The OSFTSLWAIT parameter sets the maximum time, in seconds, which a transaction can wait for input while in an OSF TSL server. CA OPS/MVS sets the server address space wait time limit based on the value of OSFTSLWAIT. The operating system checks only every 100 seconds for any waiting address space that has exceeded the wait time limit. As a result, the exact time when a server is terminated is unpredictable.

Keep this information in mind when using any CA OPS/MVS command processors that use wait times, such as OPSCMD, OPSWAIT, and OPSWTO.

Note: TSL servers are intended for longer running transactions and therefore the default value is set higher than for TSO and TSP servers.

Default value

1200 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFTSLWAIT

This function limits the wait time of a transaction to 60 seconds.

```
OPSPRM('SET','OSFTSLWAIT','60')
```

OSFTSO Parameter

This parameter determines whether you can issue TSO OSF commands from the console.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: OSFTSO

This function prevents you from issuing TSO OSF commands from the console.

```
OPSPRM('SET','OSFTSO','NO')
```

OSFTSPCPU Parameter

The OSFTSPCPU parameter limits the amount of CPU time that a single OSF TSP server transaction can use.

Default value

15 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFTSPCPU

This function allows an OSF TSP server transaction to use up to 30 seconds of CPU time.

```
OPSPRM('SET','OSFTSPCPU','30')
```

Note: Setting OSFTSPCPU to a value higher than OSFTSPRUN will in effect negate the use of this parameter. For a discussion on the practical limits for OSF TSP transactions, see OSFTSPRUN Parameter in this chapter.

OSFTSPDORM Parameter

The OSFTSPDORM parameter determines, in seconds, how long OSF TSP servers can remain dormant before they are automatically terminated. This parameter takes effect only if the number of active OSF TSP servers is greater than the value of the OSFTSPMIN parameter. This parameter then decrements the servers one per OSFTSPDORM dormant period. For more information, see OSFTSPMIN Parameter (see page 205).

Default value

60

Other possible values

Any number of seconds from 60 through 16777216

Set or modify this parameter...

Anytime

Example: OSFTSPDORM

This function allows 300 seconds, or 5 minutes, of OSF TSP server inactivity before a server is terminated.

```
OPSPRM('SET','OSFTSPDORM','300')
```

OSFTSPMAX Parameter

The OSFTSPMAX parameter sets the maximum number of OSF TSP servers that can be active at any time.

Default value

0

Other possible values

Any amount of OSF TSP servers up to 30, but at least as many as the number of OSF TSP servers specified for the OSFTSPMIN parameter. If you attempt to set the OSFTSPMAX parameter value lower than the OSFTSPMIN value, CA OPS/MVS sets OSFTSPMAX equal to OSFTSPMIN.

Set or modify this parameter...

Anytime

Example: OSFTSPMAX

This function sets 12 as the maximum number of OSF TSP servers.

```
OPSPRM('SET','OSFTSPMAX','12')
```

OSFTSPMIN Parameter

The OSFTSPMIN parameter sets the minimum number of OSF TSP servers that can be active at any time. If the OSFTSPMIN value exceeds the current number of active OSF TSP servers, CA OPS/MVS automatically starts additional servers.

We strongly recommend that OSFTSPMIN be set to a value that is high enough to process almost all (about 99 percent) of your typical high-priority asynchronous automation work. For more information, see Regulating OSF Servers in the chapter "Technical Notes" in the *Administration Guide*.

Default value

0

Other possible values

Any amount of servers up to 30, but fewer than the number of servers specified for the OSFTSPMAX parameter. If you set the OSFTSPMIN parameter value higher than the OSFTSPMAX value, CA OPS/MVS sets OSFTSPMIN equal to OSFTSPMAX.

Set or modify this parameter...

Anytime

Example: OSFTSPMIN

This function sets 2 as the minimum number of OSF TSP servers.

```
OPSPRM('SET','OSFTSPMIN','2')
```

OSFTSPOUTLIM Parameter

The OSFTSPOUTLIM parameter limits the number of console messages a transaction running under an OSF TSP server can produce. These include:

- SAY statements from REXX programs
- Output from the CLIST command WRITE
- Output of an OPSCMD command processor issued outside a REXX program

Default value

1000 (lines)

Other possible values

Any number of lines greater than or equal to 0

Set or modify this parameter...

Anytime

Example: OSFTSPOUTLIM

This function limits the number of output lines to 500.

```
OPSPRM('SET','OSFTSPOUTLIM','500')
```

OSFTSPQADD Parameter

The OSFTSPQADD parameter sets the threshold CA OPS/MVS uses to determine whether more OSF TSP servers are started. The algorithm that is used to add servers depends on the OSFQADDPROCESS parameter. IF OSFQADDPROCESS=AVAIL a new server is added, when the number of commands in the OSF TSP execution queue is higher than OSFQADD and OSFMIN but lower than OSFMAX. When OSFQADDPROCESS=QADD a new server is added and the queue depth equals OSFTSPQADD+1 + (n * OSFTSPQADD) until OSFTSPMAX is reached.

Default value

0

Other possible values

Any number of queued commands greater than or equal to 0

Set or modify this parameter...

Anytime

Example: OSFTSPQADD

This function sets the OSF TSP execution queue threshold to 5 commands.

```
OPSPRM('SET','OSFTSPQADD','5')
```

OSFTSPQUE Parameter

The OSFTSPQUE parameter specifies the maximum number of queued commands the OSF TSP execute queue can hold. This queue is where CA OPS/MVS sends TSO commands to be executed in OSF TSP servers. The OSF TSP execute processor dispatches these commands to TSP servers as the servers become available to process work.

Default value

1700 (commands)

Other possible values

Any number from 1 to 32768

Set or modify this parameter...

At initialization

Example: OSFTSPQUE

This function sets the size of the OSF TSP execution queue to 2000 commands.

```
OPSPRM('SET','OSFTSPQUE','2000')
```

OSFTSPRUN Parameter

The OSFTSPRUN parameter determines how long CA OPS/MVS allows a TSO transaction to execute in an OSF TSP server. This is an elapsed time limit; in comparison, the OSFTSPCPU parameter limits execution time.

When any server transaction exceeds the time limit set by OSFTSPRUN, CA OPS/MVS terminates the server with the hung or looping transaction and starts another OSF TSP server to accept new commands.

Default value

120 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFTSPRUN

This function sets a 60-second time limit for TSP transactions.

```
OPSPRM('SET','OSFTSPRUN','60')
```

OSFTSPWAIT Parameter

The OSFTSPWAIT parameter sets the maximum time, in seconds, which a transaction can wait for input while in an OSF TSP server. CA OPS/MVS sets the server address space wait time limit based on the value of OSFTSPWAIT. The operating system checks only every 100 seconds for any waiting address space that has exceeded the wait time limit. As a result, the exact time when a server is terminated is unpredictable.

Keep this information in mind when using any CA OPS/MVS command processors that use wait times, such as OPSCMD, OPSWAIT, and OPSWTO.

Default value

120 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFTSPWAIT

This function limits the wait time of a transaction to 60 seconds.

```
OPSPRM('SET','OSFTSPWAIT','60')
```

OSFWAIT Parameter

This parameter sets the maximum time, in seconds, which a transaction can wait for input while in an OSF TSO server. CA OPS/MVS sets the server address space wait time limit based on the value of OSFWAIT. The operating system checks only every 100 seconds for any waiting address space that has exceeded the wait time limit. As a result, the exact time when an OSF TSO server is terminated is unpredictable.

Keep this information in mind when using any CA OPS/MVS command processors that use wait times, such as OPSCMD, OPSWAIT, and OPSWTO.

Default value

120 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: OSFWAIT

This function limits the wait time of a transaction to 60 seconds.

```
OPSPRM('SET','OSFWAIT','60')
```

Rule-related Parameters

The parameters described in the following sections affect the CA OPS/MVS rules processing.

CATCHUPGLVPREFIX Parameter

The CATCHUPGLVPREFIX parameter is used internally for catch-up rule processing.

Important! Unless you are already using the default stem prefix (GLOBAL1.CATCHUP.) for other variable processing, we strongly recommend that you do not modify the value of this parameter. Do not use this global variable prefix for any other purpose. Do not assign values to the prefix itself or create any other subnodes under this prefix for any other purpose. **Failure to heed this warning results in abends in the AOF subtask during product initialization.**

Modifying the value of CATCHUPGLVPREFIX causes all CATCHUPYES and CATCHUPMAN rules that were active during the previous iteration of CA OPS/MVS to not perform catch-up processing.

Default value

GLOBAL1.CATCHUP

Other possible values

Any prefix of the form GLOBAL*n.tail.*, where *n* can be either 0-9 or A-Z. However, we strongly recommend that you avoid using the characters A-Z for the value of *n*, as they may result in the unwanted execution of GLV rules. The value of *tail* may be any combination of the characters A-Z and 0-9. The maximum length of the prefix is 33 characters; the minimum is 10.

Set or modify this parameter...

At initialization

Example: CATCHUPGLVPREFIX

This function sets the prefix to GLOBAL1.OTHER.

```
OPSPRM('SET','CATCHUPGLVPREFIX','GLOBAL1.OTHER.')
```

NORULESxxBOUND Parameter

This parameter is the time boundary, in microseconds, for the *xx* counter that keeps track of system events for which no AOF rule executes. The *xx* value is a number from 01 to 21. Such an event is counted through the NORULES*xx*COUNT parameter if its processing time falls above the previous NORULES*xx*BOUND and below the current NORULES*xx*BOUND.

Default value

No default

Other possible values

Any number of microseconds

Set or modify this parameter...

Anytime

Example: NORULES*xx*BOUND

This function sets the time boundary to 50 microseconds.

```
OPSPRM('SET','NORULESxxBOUND','50')
```

NORULESxxCOUNT Parameter

This parameter counts the number of system events for which no AOF rule executes. CA OPS/MVS keeps track of the time stamp when it detects an event for processing and the time stamp when it finishes processing the event. The difference between these time stamps is recorded as the total time CA OPS/MVS used to process the event.

An event is counted through the NORULES*xx*COUNT parameter if it falls in the time boundary set by the NORULES*xx*BOUND parameter. The *xx* is the progressive number (01 to 21) that separates the counters. The separating factor is the NORULES*xx*BOUND parameter.

Default value

No default

Other possible values

Any number

Set or modify this parameter...

Anytime

Example: NORULES*xx*COUNT

This function sets the counter to 1.

```
OPSPRM('SET','NORULESxxCOUNT','1')
```

NORULESxxMEAN Parameter

This parameter is the highest recorded processing time among all events counted in the time boundary set by NORULES*xx*BOUND.

Default value

No default

Other possible values

Any number of seconds

Set or modify this parameter...

Anytime

Example: NORULES*xx*MEAN

This function sets the mean processing time to 10 seconds.

```
OPSPRM('SET','NORULESxxMEAN','10')
```

RULEALTFIX Parameter

This parameter sets the high level qualifiers of the alternate rule sets. You can specify only the highest-level qualifiers in the list. If the high level qualifier specified by the RULEPREFIX parameter has multiple levels, the alternate rules data set names must have the same secondary levels. For example, if the RULEPREFIX parameter is set to SYS1.OPS and the RULEALTFIX parameter to SYS2, the secondary rules data should be named SYS2.OPS, not SYS2.TECH.

The RULEALTFIX value must have the same length as the highest-level qualifier of the RULEPREFIX value; both must contain seven or fewer characters, because the secondary rule set qualifiers must begin with the RULEALTFIX value.

For help in naming your rules data sets, see the *Installation Guide*.

Default value

No default

Other possible values

Any valid high level data set qualifier containing up to seven characters

Set or modify this parameter...

At initialization

Example: RULEALTFIX

This function sets the qualifier to MYRULES.

```
OPSPRM('SET','RULEALTFIX','MYRULES')
```

RULEPREFIX Parameter

This parameter sets the high level qualifier (up to 10 levels and a maximum of 26 characters) of the primary rule sets. For more information about RULEPREFIX, see the description of the RULEALTFIX parameter.

Default value

SYS1.OPS

Other possible values

Any valid high level data set qualifier

Set or modify this parameter...

At initialization

Example: RULEPREFIX

This function sets the high level qualifier for rules data sets to TEST1.OPS.

```
OPSPRM('SET','RULEPREFIX','TEST1.OPS')
```

RULEPREFIX2 Parameter

This parameter sets the high level qualifier (up to 10 levels and a maximum of 26 characters) of the secondary rule sets. This parameter differs from the RULEALTFIX parameter in that it specifies an independent prefix without the length constriction of RULEALTFIX. This parameter is mutually exclusive with the RULEALTFIX parameter and is ignored if RULEALTFIX is specified. This parameter will require more overhead than using RULEPREFIX alone, or RULEPREFIX with RULEALTFIX, but adds flexibility to the rules data set naming conventions. A duplicate rule set found using RULEPREFIX2 is ignored if a rule set with the same name is found using RULEPREFIX.

Note: This parameter is mutually exclusive with parameter RULEALTFIX and is ignored if RULEALTFIX is specified.

Default value

No default

Other possible values

Any valid high level data set qualifier

Set or modify this parameter...

At initialization

Example: RULEPREFIX2

This function sets the high level qualifier for rules data sets to SYS3.OPSMVS.

```
OPSPRM('SET','RULEPREFIX2','SYS3.OPSMVS')
```

RULEPREFIX3 Parameter

This parameter sets the high level qualifier (up to 10 levels and a maximum of 26 characters) of the tertiary rule sets. This parameter allows for a third level of ruleprefix, and should only be specified if RULEPREFIX and RULEPREFIX2 are already in use. This parameter will require more overhead than using RULEPREFIX alone, or RULEPREFIX with RULEPREFIX2 alone, but adds flexibility to the rules data set naming conventions. A duplicate rule set found using RULEPREFIX3 is ignored if a rule set with the same name is found using RULEPREFIX or RULEPREFIX2.

Note: This parameter is mutually exclusive with parameter RULEALTFIX and is ignored if RULEALTFIX is specified.

Default value

No default

Other possible values

Any valid high level data set qualifier

Set or modify this parameter...

At initialization

Example: RULEPREFIX3

This function sets the high level qualifier for rules data sets to SYS4.OPSMVS.

```
OPSPRM('SET','RULEPREFIX3','SYS4.OPSMVS')
```

RULESxxBOUND Parameter

This parameter is the time boundary, in microseconds, for the *xx* counter that keeps track of system events for which an AOF rule executes. Such an event is counted through the RULES*xx*COUNT parameter if its processing time falls above the previous RULES*xx*BOUND and just below the current RULES*xx*BOUND.

Default value

No default

Other possible values

Any number of microseconds

Set or modify this parameter...

Anytime

Example: RULES*xx*BOUND

This function sets the time boundary to 50 microseconds.

```
OPSPRM('SET','RULESxxBOUND','50')
```

RULESxxCOUNT Parameter

This parameter counts the number of system events for which an AOF rule executes. CA OPS/MVS keeps track of the time stamp when it detects an event for processing and the time stamp when it finishes processing the event. The difference between these time stamps is recorded as the total time CA OPS/MVS used to process the event.

An event is counted through the RULES*xx*COUNT parameter if it falls in the time boundary set by the RULES*xx*BOUND parameter. The *xx* is the progressive number that separates the counters. The separating factor is the RULES*xx*BOUND parameter.

Default value

No default

Other possible values

Any number

Set or modify this parameter...

Anytime

Example: RULES*xx*COUNT

This function sets the counter to 1.

```
OPSPRM('SET','RULESxxCOUNT','1')
```

RULESxxMEAN Parameter

This parameter is the highest recorded processing time among all events counted in the time boundary set by RULES*xx*BOUND.

Default value

No default

Other possible values

Any number of seconds

Set or modify this parameter...

Anytime

Example: RULES*xx*MEAN

This function sets the mean processing time to 10 seconds.

```
OPSPRM('SET','RULESxxMEAN','10')
```

RULESUBSYS Parameter

This parameter identifies the subsystem under which rule sets and REXX programs are allocated. Although this parameter is optional, sites using CA Librarian Base for z/OS or similar products can use the RULESUBSYS parameter to identify affected AOF data sets. For example, LAM often identifies data sets under CA Librarian Base for z/OS control.

Default value

No default

Other possible values

Any 1- to 4-character subsystem ID

Set or modify this parameter...

At initialization

Example: RULESUBSYS

This function identifies LAM as the subsystem for rules data sets and REXX programs.

```
OPSPRM('SET','RULESUBSYS','LAM')
```

RULESUFFIX Parameter

This parameter sets the lowest level qualifier (only one is allowed) commonly shared by all rule sets.

Default value

RULES

Other possible values

Any valid data set suffix

Set or modify this parameter...

At initialization

Example: RULESUFFIX

This function sets the suffix for rules data sets to MYRULES.

```
OPSPRM('SET','RULESUFFIX','MYRULES')
```

RULETRACE Parameter

This parameter determines whether CA OPS/MVS inserts a message in OPSLOG each time a rule executes for a given message. These OPSLOG messages can help you to resolve problems caused by multiple rules executing for a message.

Default value

OFF

This value deactivates tracing for rules processing.

Other possible values

ON

This value turns tracing on for rules processing.

Set or modify this parameter...

Anytime

Example: RULETRACE

This function turns tracing off for rules processing.

```
OPSPRM('SET','RULETRACE','OFF')
```

SMFRULEDISABLE Parameter

This parameter creates an SMF record (specified by the value of the SMFRECORDNUMBER parameter) when a user disables a rule, a rule set, or both. CA OPS/MVS produces SMF records only if the rule or rule set is in the production AOF environment, not the AOF test environment.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: SMFRULEDISABLE

This function instructs CA OPS/MVS to create SMF records when an AOF rule is disabled.

```
OPSPRM('SET','SMFRULEDISABLE','YES')
```

Global Variable Parameters

The parameters described in the following sections control how CA OPS/MVS processes global variables.

GLOBALBACKUPDSN Parameter

Use this parameter if you are using CA OPS/MVS to schedule a backup and restore of your global variable database (as described in the *Administration Guide*).

This parameter names the GDG (generation data group) data set to be used for the backup.

Note: All global variables and SQL tables are included in the backup data set.

Default value

A null string

Other possible values

Any valid GDG data set name

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPDSN

This function tells CA OPS/MVS to use OPS.GLOBKP for the GDG.

```
OPSPRM('SET','GLOBALBACKUPDSN','OPS.GLOBKP')
```

GLOBALBACKUPINTVAL Parameter

Use this parameter if you are using CA OPS/MVS to schedule a backup and restore of your global variable database.

This parameter specifies, in minutes, how long CA OPS/MVS waits between backup cycles of global variables. If you specify a value for the GLOBALBACKUPINTVAL parameter, you must also specify a value for the GLOBALBACKUPPROC parameter.

Default value

0

Other possible values

Any number of minutes between 1 and 32767. Setting this value to 0 (zero) prevents any further backups from being scheduled.

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPINTVAL

This function tells CA OPS/MVS to start the global variable backup cycle every four hours.

```
OPSPRM('SET','GLOBALBACKUPINTVAL','240')
```

GLOBALBACKUPMDSCB Parameter

Use this parameter if you are using CA OPS/MVS to schedule a backup and restore of your global variable database (as described in the *Administration Guide*). This parameter specifies the name of the model (or pattern) DSCB for the GDG.

For sample JCL for creating a GDG data set, see the GVBKGDG member in the SYS1.OPS.CCLXCNTL data set.

For sites using SMS, it may not be necessary to create the pattern DSCB or set this parameter.

Default value

A null string

Other possible values

Any valid model GDG data set name

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPMDSCB

This function names OPS.GLVBK.MODEL as the model DSCB.

```
OPSPRM('SET','GLOBALBACKUPMDSCB','OPS.GLVBK.MODEL')
```

GLOBALBACKUPMGCLASS Parameter

This parameter specifies the SMS management class for the global variable backup data set (for example, OPSMGMT).

Default value

None

Other possible values

Any valid SMS management class name

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPMGCLASS

This function names OPSMGMT as the management class for the global variable backup.

```
OPSPRM('SET','GLOBALBACKUPMGCLASS','OPSMGMT')
```

Note: This parameter, if used, may be used in conjunction with the GLOBALBACKUPSTCLASS parameter. When this parameter is specified both GLOBLALBACKUPUNIT and GLOBALBACKUPVOLSER are ignored.

GLOBALBACKUPPROC Parameter

Use this parameter if you are using CA OPS/MVS to schedule a backup and restore of your global variable database (as described in the *Administration Guide*).

This parameter specifies the name of the backup procedure. It must be a started task procedure in either SYS1.PROCLIB or any other PROCLIB that is defined to the JES subsystem and is eligible for started tasks. If you specify a value for the GLOBALBACKUPPROC parameter, you must also specify a value for the GLOBALBACKUPINTVAL parameter.

Default value

OPSSGVBK

Other possible values

Any valid started task name. Setting this parameter value to blanks prevents any further backups from being scheduled.

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPPROC

This function names GVBKUP as the started task that backs up the global variable database.

```
OPSPRM('SET','GLOBALBACKUPPROC','GVBKUP')
```

GLOBALBACKUPSTCLASS Parameter

This parameter specifies the SMS storage class for the global variable backup data set (for example, OPSPOOL, WORKPOOL).

Default value

None

Other possible values

Any valid SMS storage class name

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPSTCLASS

This function names WORKPOOL as the storage class default for the backup.

```
OPSPRM('SET','GLOBALBACKUPSTCLASS','WORKPOOL')
```

Note: This parameter overrides both GLOBLALBACKUPUNIT and GLOBALBACKUPVOLSER.

When this parameter is specified, the backup data set is allocated with a block size of zero. This allows System Determined Blocksize to select an optimum block size for the backup data set.

GLOBALBACKUPUNIT Parameter

Use this parameter if you are using CA OPS/MVS to schedule a backup and restore of your global variable database (as described in the *Administration Guide*).

This parameter specifies the generic device name for the backup data set (for example, 3390, SYSDA, and so on).

Default value

None

Other possible values

Any valid disk, tape, generic, or esoteric device name

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPUNIT

This function names SYSDA as the generic device name for the backup.

```
OPSPRM('SET','GLOBALBACKUPUNIT','SYSDA')
```

GLOBALBACKUPVOLSER Parameter

This parameter specifies the VOLUME serial of the device for the backup data set (for example, WORK01, OPS002).

Default value

None

Other possible values

Any valid DASD volume serial

Set or modify this parameter...

Anytime

Example: GLOBALBACKUPVOLSER

This function names OPS002 as the generic device name for the backup.

```
OPSPRM('SET','GLOBALBACKUPVOLSER,'OPS002')
```

This parameter must be specified in conjunction with the GLOBALBACKUPUNIT parameter. When GLOBALBACKUPSTCLASS is specified, this parameter is ignored.

GLOBALDIV Parameter

This parameter determines whether CA OPS/MVS checkpoints changes to global variables using a data-in-virtual (DIV) data set allocated to the SYSCHK1 ddname. For more information, see the *Administration Guide*.

Default value

YES

Other possible values

NO

Important! This value causes global variables to be lost when CA OPS/MVS terminates.

Set or modify this parameter...

At initialization

Example: GLOBALDIV

This function tells CA OPS/MVS not to use the global variable checkpoint data set.

```
OPSPRM('SET','GLOBALDIV','NO')
```

GLOBALINTERVAL Parameter

This parameter determines the number of seconds after which CA OPS/MVS checkpoints all changes to global variables, including the creation or deletion of variables. At each checkpoint, CA OPS/MVS saves all changes to global variables to the DIV checkpoint data set, SYSCHK1.

Default value

15 (seconds)

Other possible values

Any number of seconds between 1 and 300

Set or modify this parameter...

Anytime

Example: GLOBALINTERVAL

This function sets the checkpointing interval for global variables to 60 seconds.

```
OPSPRM('SET','GLOBALINTERVAL','60')
```

GLOBALMAX Parameter

This parameter determines the maximum number of global variable blocks that CA OPS/MVS can create.

Sizes of global variables vary, with some global variables taking up more than one block. Typically, the number of global variables in use at a site is less than the number of global variable blocks specified by GLOBALMAX.

When determining the value of the GLOBALMAX parameter, be sure to take into account these CA OPS/MVS facilities and the global variables they create:

- AOF-Consider all TOD rules that require catch-up processing. This is important because for each CATCHUPYES or CATCHUPMAN rule, CA OPS/MVS creates a variable with the value of the CATCHUPGLVPREFIX parameter as its stem.

  Note: For a description of the CATCHUPGLVPREFIX parameter, see CATCHUPGLVPREFIX Parameter in this chapter.

- RDF-The CA OPS/MVS RDF and System State Manager features require space for global variables. CA OPS/MVS implements its relational tables as a set of global variables. One global variable stores one row of a relational table, which you access through SQL statements.

Note: If you are using CA OPS/MVS to schedule a backup and restore of your global variable database, make sure you take a new backup immediately after changing the value of the GLOBALMAX parameter. For more information, see the *Administration Guide.*

Default value

10,000 (variable blocks)

Other possible values

Any number greater than 1, but the value must be higher than the current high used block in the database; if it is not, then CA OPS/MVS issues a warning message (OPS0185W) at startup. To decrease the size of the database:

1. Stop CA OPS/MVS.
2. Delete the global variable DIV data set.
3. Allocate a new global variable data set.
4. Restart CA OPS/MVS and use a smaller GLOBALMAX value.

You can increase the value of GLOBALMAX if the DIV data set contains enough space.

Set or modify this parameter...

At initialization

Example: GLOBALMAX

This function limits CA OPS/MVS to generating 3000 global variable blocks.

```
OPSPRM('SET','GLOBALMAX','3000')
```

GLOBALREBUILD Parameter

Use this parameter to instruct CA OPS/MVS to rebuild its global variable database at the next checkpoint interval (or at startup time, if you set this parameter before restarting CA OPS/MVS). At times, CA OPS/MVS itself may set this parameter value internally to COMP; for example, if the product detects integrity problems in the global variable database.

Regardless of whether you set the GLOBALREBUILD parameter or it is set internally, CA OPS/MVS automatically resets the parameter value to NONE after the rebuild operation is complete.

When invalid variables are detected during a complete rebuild, limited information regarding the discarded data is written to the OPSLOG.

Important! Use the GLOBALREBUILD parameter only when a CA Customer Support representative tells you to do so.

Default value

NONE

Other possible values

TREE, to request an AVL tree rebuild; and COMP, to request a complete rebuild

Set or modify this parameter...

Anytime

Example: GLOBALREBUILD

This function requests a complete rebuild.

```
OPSPRM('SET','GLOBALREBUILD','COMP')
```

GLOBALTEMPMAX Parameter

This parameter determines the maximum number of *temporary* global variables that CA OPS/MVS can create. The number of GLVTEMP*n*., GLVEVENT., and GLVJOBID. variables in use influences the amount of storage that must be allocated during CA OPS/MVS processing. Therefore, take into account the number of expected GLVTEMP*n*., GLVEVENT., and GLVJOBID. variables when you set the GLOBALTEMPMAX parameter. You should also consider the use of the GLOBAL TEMPORARY operand on CREATE TABLE statements. Use of the GLOBAL TEMPORARY operand causes the creation of a temporary table to store relational data.

Default value

5000 (variables)

Other possible values

Any number greater than or equal to 0

Set or modify this parameter...

At initialization

Example: GLOBALTEMPMAX

This function limits CA OPS/MVS to generating 3000 temporary global variables.

```
OPSPRM('SET','GLOBALTEMPMAX','3000')
```

GLOBALTEMPWARNIV Parameter

This parameter determines how often CA OPS/MVS issues a warning message when the *temporary* global variable database is becoming full.

CA OPS/MVS issues warning message OPS4290O:

- When database usage reaches the level set by the GLOBALTEMPWARNTH parameter
- At the interval specified with GLOBALTEMPWARNIV
- Each time database usage increases by 5 percent above the threshold (for instance, at 85 percent, 90 percent, and 95 percent of capacity)

Note: When the threshold for the temporary global variable database is exceeded due to a Relational Data Framework-related SQL operation, certain conditions prevent the OPS4290O message from being issued synchronously. In such cases CA OPS/MVS records the threshold exceeded condition and issues the message when the next global variable operation (either reference or update) occurs. If no global variable operations are performed, the database may completely fill up due to SQL INSERT, UPDATE, CREATE TABLE, and ALTER TABLE operations prior to message OPS4290O being issued.

Default value

5 (minutes)

Other possible values

Any number of minutes between 1 and 32767

Set or modify this parameter...

Anytime

Example: GLOBALTEMPWARNIV

This function causes CA OPS/MVS to issue warnings every three minutes.

```
OPSPRM('SET','GLOBALTEMPWARNIV','3')
```

GLOBALTEMPWARNTH Parameter

This parameter determines how full the *temporary* global variable database becomes before CA OPS/MVS starts issuing warnings.

Default value

80 (80 percent of blocks in use)

Other possible values

Any percentage between 1 and 100

Set or modify this parameter...

Anytime

Example: GLOBALTEMPWARNTH

This function causes CA OPS/MVS to issue warnings when the database is 75 percent full.

```
OPSPRM('SET','GLOBALTEMPWARNTH','75')
```

GLOBALTEMPWSGCIV Parameter

This parameter determines how often CA OPS/MVS schedules the Garbage-Collection REXX to delete temporary global variables that the osfrexx web service uses.

Notes:

- If you use the osfrexx web service to start OSF/REXX programs and those programs use the OPWSOUT function to store their responses, temporary global variables with prefix GLVTEMP0.#OPWEBSVC# are used to store the responses.
- Since the nature of the OSF servers is asynchronous, it is up to your site to determine how long those responses are saved. If you never use the response feature (for example, you never call OPWSOUT), then there is no need for this parameter. But if you do use OPWSOUT, you *must* determine how long you wish those responses to remain in temporary variables before they are deleted.
- You must strike a balance between leaving them for long enough for your web client applications to retrieve them but not so long that they use up too much of your global variable space and potentially disrupt other automation also using global variables space.

Default value

30 (minutes)

Other possible values

Any number of minutes between 5 and 32767

Set or modify this parameter...

Anytime

Example: GLOBALTEMPWSGCIV

```
OPSPRM('SET','GLOBALTEMPWSGCIV','60')
```

GLOBALWARNINTVAL Parameter

This parameter determines how often CA OPS/MVS issues a warning message when the global variable database is becoming full. CA OPS/MVS issues warning message OPS4290O when:

- Database usage reaches the level set by GLOBALWARNTHRESH
- The interval specified with GLOBALWARNINTVAL is reached
- Each time database usage increases by 5 percent above the threshold (for instance, at 85 percent, 90 percent, and 95 percent of capacity)

Note: When the threshold for the global variable database is exceeded due to a Relational Data Framework-related SQL operation, certain conditions prevent the OPS4290O message from being issued synchronously. In such cases CA OPS/MVS records the threshold exceeded condition and issues the message when the next global variable operation (either reference or update) occurs. If no global variable operations are performed, then the database may completely fill up due to SQL INSERT, UPDATE, CREATE TABLE, and ALTER TABLE operations prior to message OPS4290O being issued.

Default value

5 (minutes)

Other possible values

Any number of minutes between 1 and 32767

Set or modify this parameter...

Anytime

Example: GLOBALWARNINTVAL

This function causes CA OPS/MVS to issue warnings every three minutes.

```
OPSPRM('SET','GLOBALWARNINTVAL','3')
```

GLOBALWARNTHRESH Parameter

This parameter determines how full the global variable database becomes before CA OPS/MVS starts issuing warnings.

Default value

80 (80 percent of blocks in use)

Other possible values

Any percentage between 1 and 100

Set or modify this parameter...

Anytime

Example: GLOBALWARNTHRESH

This function causes CA OPS/MVS to issue warnings when the database is 75 percent full.

```
OPSPRM('SET','GLOBALWARNTHRESH','75')
```

GLVCHAINMAX Parameter

This parameter sets the maximum number of global variable events that can execute in response to an original global variable event. GLVCHAINMAX and the GLVPENDINGMAX parameter together prevent runaway recursion of global variable events. If the number of events exceeds the value of GLVCHAINMAX, CA OPS/MVS issues message OPS4401E.

Default value

1000

Other possible values

Any number of events between 1 and 32767

Set or modify this parameter...

Anytime

Example: GLVCHAINMAX

This function tells CA OPS/MVS to allow only 500 global variable events to result from another global variable event.

```
OPSPRM('SET','GLVCHAINMAX','500')
```

GLVDELETERULES Parameter

The GLVDELETERULES parameter activates global variable rule execution for the three delete variable function codes of OPSVALUE. The delete function codes are R (remove), 4 (delete by name mask) and 6 (single variable delete). The OPSDELV command or function implicitly uses the 4 and 6 codes. To execute a GLV delete rule, at least one variable must be successfully deleted. The new GLV.FUNCTION built-in variable contains the OPSVALUE function code of the variable operation. This variable can be used to distinguish a delete operation from an update operation.

Default

NO

This value prevents global variable delete events.

Other possible values

YES

This value activates global variable rule execution for the variable deletes.

Set or modify this parameter...

Anytime

Example: GLVDELETERULES

This function specifies that global variable rules for variable delete functions are executed.

```
OPSPRM('SET','GLVDELETERULES','YES')
```

GLVNOTIFYRULES Parameter

The GLVNOTIFYRULES parameter controls global variable rule execution for SYSPLEX variables that are managed by the OPSVASRV command. Three subfunctions of OPSVASRV drive a GLV rule when this parameter is set to “YES”. They are: CREATE, UPDATE, and DELETE. Other subfunctions do not drive a GLV rule.

Default

NO

This value prevents global variable events for SYSPLEX variables.

Other possible values

YES

This value activates global variable rule execution for the SYSPLEX variable creation, deletion, and updates.

Set or modify this parameter...

Anytime

Example: GLVNOTIFYRULES

This function specifies that global variable rules for SYSPLEX variable functions CREATE, DELETE, and UPDATED are executed.

```
OPSPRM('SET','GLVNOTIFYRULES','YES')
```

GLVPENDINGHIGH Parameter

The GLVPENDINGHIGH parameter displays the highest number of entries ever queued in any of the global variable pending queues used by AOF rules during the current life of the product or since this parameter value was last reset. Compare this value to the value specified on the GLVPENDINGMAX parameter.

Note: Resetting these values to zero affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: GLVPENDINGHIGH

This function resets the GLVPENDINGHIGH value to zero.

```
OPSPRM('SET','GLVPENDINGHIGH',0)
```

GLVPENDINGMAX Parameter

This parameter sets the maximum number of global variable events that can be pending at any time for a rule. When a global variable rule changes a global variable, the event goes onto a pending queue until the rule finishes its processing. This queue and the GLVCHAINMAX parameter help to prevent event recursion. If the number of pending events exceeds the GLVPENDINGMAX value, the rule fails and CA OPS/MVS issues message OPS4401E.

Default value

100

Other possible values

Any number of events between 1 and 32767

Set or modify this parameter...

At initialization

Example: GLVPENDINGMAX

This function limits the number of pending global variable events to 40.

```
OPSPRM('SET','GLVPENDINGMAX','40')
```

GLVSHAREDDD Parameter

The GLVSHAREDDD parameter relates to the OPSHFI command processor and shared file support. The OPSHFI command processor provides you with the capability to store global variable records on an external VSAM key-sequenced data set that can be shared among systems. Using various OPSHFI keywords, you can WRITE variable records to the VSAM file, or READ or DELETE variable records from the VSAM file.

The GLVSHAREDDD parameter specifies the z/OS ddname that you want CA OPS/MVS to use when it allocates this shared VSAM file.

CA OPS/MVS uses the value you specify to form the z/OS RESERVE enqueue name, which is used to serialize access to a file that is shared among systems. All systems that share the VSAM file should use the same ddname.

For related information, see the description of the OPSHFI command processor in the *Command and Function Reference*.

Default

OPAMSVDB

Other possible values

Any valid 1- to 8-character z/OS ddname

Set or modify this parameter...

Anytime

Example: GLVSHAREDDD

This function specifies ATMSVDB as the z/OS ddname to be used for allocating the VSAM file.

```
OPSPRM('SET','GLVSHAREDDD','ATMSVDB')
```

GLVSHAREDFILE Parameter

The GLVSHAREDFILE parameter relates to the OPSHFI command processor and shared file support. The OPSHFI command processor provides you with the capability to store global variable records on an external VSAM key-sequenced data set that can be shared among systems. Using various OPSHFI keywords, you can WRITE variable records to the VSAM file, or READ or DELETE variable records from the VSAM file.

The GLVSHAREDFILE parameter specifies the name of the VSAM key-sequenced file that CA OPS/MVS should use for OPSHFI requests. The VSAM file must not be in the load state.

For a sample IDCAMS definition of the file, see the OPAMSVDB member of the SYS1.OPS.CCLXCNTL data set. The key of the VSAM file is:

This part of the key…	Contains…
Beginning with position 1 of the key and continuing for the entire key length, minus the last 4 characters of the key	The variable name
The last 4 characters of the key	Either the SMF ID of the owning system, or blank for variables that are shared across systems

Note: If you need to perform emergency maintenance on or a reorganization of the real VSAM file, change the value of the GLVSHAREDFILE parameter to NULLFILE or DUMMY. Then use the OPSHFI command processor to issue a dummy file request. Doing this deallocates the VSAM file so that you can repair or reorganize it. When the file is ready to use again, reset the GLVSHAREDFILE parameter to the name of the VSAM file.

For related information, see the description of the OPSHFI command processor in the *Command and Function Reference*.

Default

NULLFILE

All file requests become null operations and process no variables.

Other possible values

The z/OS cluster name of the VSAM file

DUMMY

All file requests become null operations and process no variables.

Set or modify this parameter...

Anytime

Example: GLVSHAREDFILE

This function specifies that all OPSHFI requests should become null operations and process no variables.

```
OPSPRM('SET','GLVSHAREDFILE','NULLFILE')
```

GLVSHAREDQUE Parameter

The GLVSHAREDQUE parameter specifies the maximum number of shared variable VSAM file requests that may be waiting to be processed. Since real VSAM file input and output is being performed by the service task, this parameter may need to be adjusted upward for heavy usage of this facility.

Default

100 requests

Other possible values

Any number from 50 to 10000

Set or modify this parameter...

At initialization

Example: GLVSHAREDQUE

Change the size of the VSAM shared file request queue to 200.

```
OPSPRM('SET','GLVSHAREDQUE','200')
```

GLVSHAREDRESERVE Parameter

The GLVSHAREDRESERVE parameter relates to the OPSHFI command processor and shared file support. The OPSHFI command processor provides you with the capability to store global variable records on an external VSAM key-sequenced data set that can be shared among systems. Using various OPSHFI keywords, you can WRITE variable records to the VSAM file, or READ or DELETE variable records from the VSAM file.

The GLVSHAREDRESERVE parameter determines the number of seconds that CA OPS/MVS should wait to acquire control of the shared VSAM file using a z/OS RESERVE request.

If you specify 0, no RESERVE is performed. In this case, the file must not be accessed simultaneously by another system. If the RESERVE request times out due to contention with other systems, the file request that is being processed fails.

If the system uses a product such as CA MII to convert RESERVE requests to global enqueue requests, define the ddname used for shared file allocation to the product as the RESERVE major name. Consult your local systems programming group for the required system changes.

The z/OS RESERVE enqueue name for the shared VSAM file is:

Name	Description
Major name	The value of the GLVSHAREDDD parameter
Minor name	SVDB

For more information about the VSAM file used to store global variables, see the description of the OPSHFI command processor in the *Command and Function Reference*.

Default

120

Other possible values

Any number of seconds from 0 to 600

Set or modify this parameter...

Anytime

Example: GLVSHAREDRESERVE

This function specifies that CA OPS/MVS should wait 60 seconds to acquire control of the shared VSAM file.

```
OPSPRM('SET','GLVSHAREDRESERVE','60')
```

GLVSHAREDRLS Parameter

The GLVSHAREDRLS parameter allows CA OPS/MVS to use VSAM RLS for the shared VSAM file feature instead of using hardware reserve to serialize access to the file. z/OS, with DFSMS/MVS 1.3 and coupling facility hardware, supports VSAM record level sharing (RLS) through the coupling facility. For information about the availability and use of this feature at your site, see your systems programming group. Also, for information about VSAM RLS recovery considerations and the cleanup of residual record locks for failures, see the appropriate IBM manuals regarding DFSMS/MVS 1.3 data sharing using RLS.

To use VSAM RLS, the shared VSAM file must be managed by SMS and defined with the IDCAMS parameter LOG(NONE). All systems that share the file must also use RLS. Using both RLS and hardware reserve causes open failures.

To use VSAM RLS, all the following must be true:

- All the systems in the parallel sysplex that share the file must also use RLS
- The VSAM data set must be managed by SMS; therefore, SMS must be active
- The SMSVSAM started task must be active

Default

NO

This value prevents CA OPS/MVS from using VSAM RLS for file serialization.

Other possible values

YES

This value causes CA OPS/MVS to use VSAM RLS for file serialization.

Set or modify this parameter...

Anytime

Example: GLVSHAREDRLS

This function specifies that the VSAM shared file I/O feature is to use VSAM RLS for serialization.

```
OPSPRM('SET','GLVSHAREDRLS','YES')
```

GLVSHAREDTASK Parameter

The GLVSHAREDTASK parameter controls the availability and features of the VSAM shared file support task. This task is attached at product initialization. Both the OPSHFI POI command and OPS/REXX function queue shared file requests to this task and, optionally, wait for a result. If the shared file task is not used, you can save a considerable amount of virtual storage overhead by not starting this task or by prohibiting the execution of GLV AOF rules by the READ function of the OPSHFI command.

Default

GLVRULES

The shared file task is attached, and it can execute GLV rules. GLVRULES requires a large amount of virtual storage, more than any other value.

Other possible values

- NOTASK

  The shared file task is not attached. Any OPSHFI command that is issued produces a return code of 8. NOTASK eliminates the overhead associated with unwanted features.

- NOGLVRULES

  The shared file is attached, but it cannot invoke GLV rules. The system ignores any SVRULES(YES) keyword specified for the OPSHFI command. NOGLVRULES prevents allocating all the OPS/REXX virtual storage that the execution of GLV rules requires.

Set or modify this parameter...

At initialization

Example: GLVSHAREDTASK

This function specifies that the OPSHFI command should not be used by this copy of the product.

```
OPSPRM('SET','GLVSHAREDTASK','NOTASK')
```

OPS/REXX Parameters

The parameters described in the following sections control OPS/REXX.

REXXDDNAME Parameter

This parameter enables you to modify the ddname that CA OPS/MVS uses to locate the source REXX program when using OPS/REXX command processors. These command processors include the OPS/REXX compile command processors (OICOMP and OXCOMP) as well as other OPS/REXX command processors (OI, OPSIMEX, OX, OPSEXEC, and their aliases).

If you set the value of the REXXDDNAME parameter to a null value or a blank string, CA OPS/MVS treats it as though you set it to the default value of SYSEXEC. If you set the value to an invalid ddname, all attempts to use the OPS/REXX command processors fail.

The value of the REXXDDNAME parameter is effective even when CA OPS/MVS is down, as long as CA OPS/MVS has been started at least once since the last system IPL, and the parameter value was set. If CA OPS/MVS has never been started, the default value of SYSEXEC is always used.

If you are using subsystems other than OPSS, always use the keyword format of the OPS/REXX command processors, using SUBSYS(OPS*x*), to reference the correct REXXDDNAME parameter value for that subsystem or allocate the OPS$OPS*x* ddname.

For more information about running multiple copies of CA OPS/MVS, see the *Administration Guide*.

Default value

SYSEXEC

Other possible values

Any 1- to 8-character string that is a valid ddname

Set or modify this parameter...

Anytime; however, we strongly recommend that you modify this parameter only during product initialization, and that you do not change it thereafter.

Example: REXXDDNAME

This function defines SYSOEXEC as the ddname CA OPS/MVS uses to locate the source REXX program.

```
OPSPRM('SET','REXXDDNAME','SYSOEXEC')
```

REXXDEFAULTADDRESS Parameter

This parameter sets the default host command environment for OPS/REXX programs and AOF request (REQ) rules.

Default value

TSO

Has the same effect as having ADDRESS TSO as the first command issued in an OPS/REXX program.

Other possible values

Any valid OPS/REXX host environment

Set or modify this parameter...

Anytime

Example: REXXDEFAULTADDRESS

This function defines OSF as the default host environment for OPS/REXX.

```
OPSPRM('SET','REXXDEFAULTADDRESS','OSF')
```

REXXMAXCLAUSES Parameter

This parameter defines the maximum number of REXX clauses that OPS/REXX programs or request rules can execute.

Default value

1000000

Other possible values

Any number of clauses between -1 and 999999 (a value of -1 indicates that there is no limit; a value of 0, while not prohibited, has no meaning and causes unpredictable results)

Set or modify this parameter...

Anytime

Example: REXXMAXCLAUSES

This function sets the maximum number of clauses to 500000.

```
OPSPRM('SET','REXXMAXCLAUSES','500000')
```

REXXMAXCOMMANDS Parameter

This parameter determines the maximum number of host commands that OPS/REXX programs or request rules can execute. Host commands are commands issued in all valid ADDRESS environments.

Default value

100000

Other possible values

Any number of commands between -1 and 99999 (a value of -1 indicates that there is no limit)

Set or modify this parameter...

Anytime

Example: REXXMAXCOMMANDS

This function limits OPS/REXX programs or request rules to issuing 20000 host commands.

```
OPSPRM('SET','REXXMAXCOMMANDS','20000')
```

REXXMAXPGMSIZE Parameter

This parameter determines the maximum size, in bytes, of a compiled OPS/REXX program in virtual storage.

Default value

1048616 (bytes)

Other possible values

Any number of bytes greater than 32768

Set or modify this parameter...

Anytime

Example: REXXMAXPGMSIZE

This function sets the maximum OPS/REXX program size to 750000 bytes.

```
OPSPRM('SET','REXXMAXPGMSIZE','750000')
```

REXXMAXQUEUE Parameter

This parameter sets the maximum number of output lines that OPS/REXX programs can have in their external data queues.

Default value

3000 (lines)

Other possible values

Any number of output lines between 1 and 32768. The maximum value you can specify is dependent upon your region. When considering the amount of storage that is necessary, bear in mind that every three messages take up 1 KB of storage.

Set or modify this parameter...

Anytime

Example: REXXMAXQUEUE

This function sets the maximum queue size to 2000 lines.

```
OPSPRM('SET','REXXMAXQUEUE','2000')
```

REXXMAXSAYS Parameter

This parameter sets the maximum number of SAY statements that OPS/REXX programs or request rules can execute.

Default value

100000

Other possible values

Any number of statements between -1 and 99999 (a value of -1 indicates that there is no limit)

Set or modify this parameter...

Anytime

Example: REXXMAXSAYS

This function limits the number of SAY statements to 30000.

```
OPSPRM('SET','REXXMAXSAYS','30000')
```

REXXMAXSECONDS Parameter

This parameter sets the maximum number of seconds that OPS/REXX programs or request rules can execute.

Default value

No default

Other possible values

Any number of seconds greater than or equal to -1 (a value of -1 indicates that there is no limit; a value of 0, while not prohibited, has no meaning and causes unpredictable results)

Set or modify this parameter...

Anytime

Example: REXXMAXSECONDS

This function allows OPS/REXX programs or request rules to execute for up to 600 seconds (ten minutes).

```
OPSPRM('SET','REXXMAXSECONDS','600')
```

REXXMAXSTRINGLENGTH Parameter

This parameter sets the maximum length of text strings in an OPS/REXX program. The limit set by REXXMAXSTRINGLENGTH is approximate; you cannot set an exact limit on string length.

Default value

32000

Other possible values

Any number of characters greater than 128 but not more than 32000

Set or modify this parameter...

Anytime

Example: REXXMAXSTRINGLENGTH

This function limits text string length to 1048 characters.

```
OPSPRM('SET','REXXMAXSTRINGLENGTH','1048')
```

System State Manager Parameters

The parameters described in the following sections control the CA OPS/MVS System State Manager (SSM) feature.

SSMACTIVEGLOBAL Parameter

The SSMACTIVEGLOBAL parameter sets the SSM global application manager status for the local system and immediately transmits the status to all systems that have an MSF connection to the local system. Every CA OPS/MVS system has its own setting for this parameter. To programmatically retrieve the status of all systems, issue the ADDRESS OPSCTL MSF LIST command.

This setting is intended for use by SSMGA. However, customers who do not use SSMGA can use this setting as a configurable option for their own application that applies coordinated automation procedures to multiple connected systems.

Note: If you are using SSMGA, do not use this parameter. The setting of this parameter is automatically maintained by SSMGA.

Default value

NO

Other possible values

- YES

  System is an active global status manager.

- NO

  System is not an active global status manager.

Set or modify this parameter...

Anytime

Example: SSMACTIVEGLOBAL

This function sets the local SSM system as an active global status manager.

In the SSMGA application, the SSMGAGST REXX program would set this parameter to 'YES' when the local SSM system has the highest SSMPRIORITY value of all other MSF connected systems belonging to the same SSMplex:

```
OPSPRM('SET','SSMACTIVEGLOBAL','YES')
```

SSMAUDIT Parameter

The SSMAUDIT parameter controls whether changes to SSM resource tables and the directory table are logged to OPSLOG using the trace message OPS7914T. This message includes the SQL operation, the table name, key, and the job name and program name of the address space that issued the SQL command. For SQL updates, the first monitored column name changed and the first 16 characters of the new value are also displayed.

Default value

YES - Produce audit messages.

Other possible values

NO - Suppress audit messages.

Set or modify this parameter...

Anytime

Example: SSMAUDIT

This function deactivates SSM audit message generation for SSM resource and directory table changes.

```
OPSPRM('SET','SSMAUDIT','NO')
```

SSMAUTOHOME Parameter

Use the parameter SSMAUTOHOME to enable the AUTOHOME functionality to move the resource back to their home system once this system becomes active.

Default value

NO

AUTOHOME functionality disabled.

Other possible values

YES

AUTOHOME functionality enabled.

Set or modify this parameter...

Anytime

Example: SSMAUTOHOME

Activate the AUTOHOME functionality:

```
OPSPRM('SET','SSMAUTOHOME','YES')
```

SSMAUXTBLPREFIX Parameter

The SSMAUXTBLPREFIX parameter specifies an RDF table name prefix that causes the SSM engine to awaken from a wait state whenever an SQL command external to the SSM subtask makes a change to any RDF table whose table name begins with the specified prefix. Table names that begin with this prefix should be reserved for use as auxiliary status tables for resources that are not defined on the local system or some similar support function related to SSM.

Default value

Blank

Other possible values

A 1- to 6-character prefix that complies with the RDF table naming restrictions (A-Z, 0-9, @, #, $, and _, where the first character is any letter from A-Z)

Set or modify this parameter...

Anytime

Example: SSMAUXTBLPREFIX

This function assigns an SSM table prefix for SSM auxiliary tables.

```
OPSPRM('SET','SSMAUXTBLPREFIX','SSM#E')
```

SSMDEBUG Parameter

The SSMDEBUG parameter causes the SSM engine to produce additional diagnostic messages for problem determination. Use this parameter only under the direction of CA Customer Support. Changes to this parameter take effect immediately.

Default value

NO

Other possible values

- YES, produce diagnostic messages.
- NO, suppress diagnostic messages.

Set or modify this parameter...

Anytime

Example: SSMDEBUG

This function activates additional diagnostic message generation for all SSM resource processing:

```
OPSPRM('SET','SSMDEBUG','YES')
```

SSMGARESTXTUPDT Parameter

The optional SSMREXSTXTUPDT parameter controls the updating of the RESOURCE_TEXT column. Set the SSMRESTXTUPDT parameter to NO to prevent the updating of the resource text in the local table by SSM resource updates. Keep the SSMRESTXTUPDT parameter set to Yes to keep the RESOURCE_TEXT column synchronized across all inactive copies of the resource. The active copy will not be propagated to unsynchronized copies.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: SSMGARESTXTUPDT

This function disables SSM in order to prevent SSM resource updates making changes in the RESOURCE_TEXT column in the local table:

```
OPSPRM('SET','SSMRESTXTUPD','NO')
```

SSMGLOBALEXITTBL Parameter

The SSMGLOBALEXITTBL parameter specifies the RDF table name of the SSM global exit action table. This action table allows for the specification of the BEGIN GLOBAL process and the SSM actions to insert, update, and delete SSM table resource row events detected by the RDF engine. The structure of this action table is slightly different from the conventional SSM action table. Only the event type, table name, and resource name form the key of this table for action selection. If the table name specified does not exist, then the SSM engine creates an empty table of that name. Rows may then be added to this table using the RDF table editor. This parameter is only relevant when the SSMGLOBALEXITS parameter is set to YES.

Default value

SSMV2_GBLEXIT_TBL

Other possible values

Any valid 1- to 18-character RDF table name

Set or modify this parameter...

Anytime

Example: SSMGLOBALEXITTBL

This function specifies an alternate SSM global exit facility action table name:

```
OPSPRM('SET','SSMGLOBALEXITTBL','SSMV2_GLOBAL_TBLEX')
```

SSMGLOBALEXITS Parameter

The SSMGLOBALEXITS parameter determines whether the global exit facility of SSM is active while SSM executes. The global exit facility works in conjunction with RDF SQL exits to track row inserts, updates to monitored columns, and deletes of SSM resource table rows. When such changes occur, the table name and resource name are recorded by the RDF and asynchronously processed by the SSM engine during usual processing. A unique global exit action table specified by the SSMGLOBALEXITTBL parameter is used to specify SSM actions to take in response to the detected table changes.

Default value

NO

Other possible values

- YES

  Activate the SSM global exit facility.

- NO

  Deactivate the SSM global exit facility.

Set or modify this parameter...

Anytime

Example: SSMGLOBALEXITS

This function activates the global exit facility in the SSM System State Manager task:

```
OPSPRM('SET','SSMGLOBALEXITS','YES')
```

SSMGAPREREQCHK Parameter

Use the parameter SSMGAPREREQCHK to enable the premove prerequisite validation for system recovery and resource movement commands.

Default value

NO

Prerequisite validation disabled.

Other possible values

YES

Prerequisite validation enabled.

Set or modify this parameter...

Anytime

Example: SSMPREREQCHK

Activate the prerequisite validation process:

```
OPSPRM('SET','SSMGAPREREQCHK','YES')
```

SSMGLVPREFIX Parameter

The SSMGLVPREFIX parameter specifies a global variable name prefix that can be automatically added as a prefix to variables in action table text. The SSM engine will replace a variable of the format &.var1 with the value of a global variable whose full name is ssmglvprefix.var1. This allows for much shorter action text strings when global variables with a common prefix are used in the action text as substitutable items.

Default value

GLOBAL0.#SSMVARS

Other possible values

A 7- to 30-character valid global variable name that does not end with a period

Set or modify this parameter...

Anytime

Example: SSMGLVPREFIX

This function specifies an alternate SSM global variable prefix value that is eligible for GLV rule processing:

```
OPSPRM('SET','SSMGLVPREFIX','GLOBALB.#SSMEVAR#')
```

SSMPLEXNAME Parameter

The SSMPLEXNAME parameter sets a visible SSMplex name for the local system and immediately transmits the name to all systems that have an MSF connection to the local system. Every CA OPS/MVS system has its own setting for this parameter. To programmatically retrieve the SSMplex name of all systems, issue the ADDRESS OPSCTL MSF LIST command.

This setting is intended for use by SSMGA. However, customers who do not use SSMGA can use this setting as a configurable option for their own application that applies coordinated automation procedures to multiple connected systems

Default value

NONE

Other possible values

Any 1- to 8-character name

Set or modify this parameter...

Anytime

Example: SSMPLEXNAME

This function assigns an SSM group name to the local SSM task:

```
OPSPRM('SET','SSMPLEXNAME','PROD001')
```

SSMMONITOREDCOLn Parameter

There are five of these parameters named SSMMONITOREDCOL1 to SSMMONITOREDCOL5. Each SSMMONITOREDCOL*n* parameter can specify an eighteen-character SQL column name. Columns specified within these parameters are actively monitored by System State Manager. User-defined exits may be written to react to the global events generated by changes to these columns. Changes to these columns may also be logged to the OPSLOG using the trace message OPS7914T if SSMAUDIT is set to YES.

Default value

Blank ' '

The default value of blank indicates that no column name is specified on this parameter.

Other possible values:

Up to eighteen character column name.

This value indicates that System State Manager should monitor changes within the column specified in all SSM-managed tables on this subsystem.

Important! Consider your choice of additional monitored columns carefully. Monitoring a column involves waking up the main SSM processing cycle each time a change is detected. Frequently modified columns are unsuitable for monitoring and may result in poor performance. Examples of inappropriate SSM columns to monitor include MISSING_PREREQ and INTERNAL_DATA

SSMOPSWEB Parameter

This parameter controls whether System State Manager sends internal resource update notifications to its associated WebCenter component. These notifications are only sent when the WebCenter component has been detected by SSM.

Default value

YES

If it is active, resource update notifications are sent to WebCenter.

Other possible values

NO

Resource update notifications are not sent to WebCenter.

Set or modify this parameter...

Anytime

Example: SSMOPSWEB

This function tells CA OPS/MVS to prohibit all SSM resource update notifications to the WebCenter component.

```
OPSPRM('SET','SSMOPSWEB','NO')
```

SSMPRIORITY Parameter

The SSMPRIORITY parameter sets a global priority for the local system and immediately transmits the value to all systems that have an MSF connection to the local system. Every CA OPS/MVS system has its own setting for this parameter. To programmatically retrieve the priority of all systems, issue the ADDRESS OPSCTL MSF LIST command.

This setting is intended for use by SSMGA. However, customers who do not use SSMGA can use this setting as a configurable option for their own application that applies coordinated automation procedures to multiple connected systems

Default value

0

Other possible values

An integer value between 0 and 1000

Set or modify this parameter...

Anytime

Example: SSMPRIORITY

This function assigns an active global priority of 100 to the local SSM system:

```
OPSPRM('SET','SSMPRIORITY','100')
```

Note: Setting the priority to 1000 causes the number 1000 to be added to the current 1–999 value of SSMPRIORITY. This feature forces the normal highest priority selection scheme to select a specific system. SSMGA uses this capability for designation of a global system by operator command. Once the system becomes the SSMGA global, the system priority is reset to its original value by subtracting 1000. Thus it is possible that priority values in the range 1001-1999 will be displayed as the value of this parameter. However the directly assignable values of the parameter using OPSPRM remain 0–1000.

SSMSUBPREFIX Parameter

The SSMSUBPREFIX parameter specifies an RDF table name prefix that causes the SSM subtask manager to awaken from a wait state whenever an SQL command that did not originate from the local SSM subtask or SSM main task makes a change to any RDF table whose table name begins with the specified prefix. Table names that begin with this prefix should be reserved for exclusive use by the SSM subtask manager.

Default value

Blank

Other possible values

A 1- to 6-character prefix that complies with the RDF table naming restrictions (A-Z, 0-9, @, #, $, and _, where the first character is any letter from A-Z)

Set or modify this parameter...

Anytime

Example: SSMSUBPREFIX

This function assigns an SSM table prefix for SSM subtask manager tables:

```
OPSPRM('SET','SSMSUBPREFIX','SSMSU_')
```

SSMSUBRULE Parameter

The SSMSUBRULE parameter specifies the name of an enabled AOF request rule that will be executed by the SSM subtask manager whenever the subtask manager is awakened by an external SQL change to an RDF table whose table name begins with the SSMSUBPREFIX value, an explicit OPSSMTBL POST(SSMSUB) command, or changes in SSMPLEXNAME, SSMGLOBALACTIVE, or SSMPRIORITY values in any MSF connection. This parameter also controls the starting and stopping of the SSM subtask manager task. The default value of 'NONE' prevents the start of or causes the termination of the SSM subtask manager. Any other value activates the SSM subtask manager task.

Default value

NONE

Other possible values

Any valid 1-8 character AOF request rule name

Set or modify this parameter...

Anytime

Example: SSMSUBRULE

This function starts the SSM subtask manager if not already active and instruct the subtask manager to invoke the AOF request rule SSMSUB each time a wake-up event occurs:

```
OPSPRM('SET','SSMSUBRULE','SSMSUB')
```

SSMVERSION Parameter

The SSMVERSION parameter determines what version of the SSM engine will be started when SSM initializes during product startup or when the SSM engine subtask restarts. At this time, only the SSM version 2 engine is supported.

Default value

2

Other possible values

None

Set or modify this parameter...

Anytime; however, it will not take effect until the following command is issued:

F OPSS,RESTART(STATEMAN)

Example: SSMVERSION

This function uses version 2 of the SSM engine for SSM processing:

```
OPSPRM('SET','SSMVERSION','2')
```

STATEGROUPMAN Parameter

The STATEGROUPMAN parameter determines whether System State Manager automatically maintains Group Manager tables.

If you change the value of the STATEGROUPMAN parameter from NO to YES and System State Manager is not inactive, System State Manager builds and updates the Group Manager tables with the most current resource status information.

Default value

NO

System State Manager does not build or update the Group Manager tables. If you do not use the Group Manager, you can save a significant amount of resources by leaving this parameter set to NO.

Other possible values

YES

System State Manager builds and updates the Group Manager tables with the current statuses of all monitored resources. The STATEGROUPMAN parameter must be set to YES for you to be able to use the OPSVIEW Group Manager option.

Set or modify this parameter...

Anytime

Example: STATEGROUPMAN

This function indicates that you want System State Manager to automatically maintain Group Manager tables:

```
OPSPRM('SET','STATEGROUPMAN','YES')
```

STATEIGNOREMPRE Parameter

The STATEIGNOREMPRE parameter controls whether nonexistent SSM resource prerequisites are considered missing when prerequisite processing is performed by the SSM engine task.

When the value of this parameter is YES, all undefined prerequisites that prevent SSM actions from performing are ignored and omitted from the missing prerequisite list.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: STATEIGNOREMPRE

This function tells SSM to ignore undefined prerequisite resources:

```
OPSPRM('SET','STATEIGNOREMPRE','YES')
```

STATEMAN Parameter

This parameter determines the executing mode for System State Manager.

Note: If System State Manager is installed at your site but you do not use it, add an OPSPRM statement to your OPSTART1 initialization CLIST that sets the value of the STATEMAN parameter to INACTIVE:

```
abc = Opsprm('SET','STATEMAN','INACTIVE')
```

Default value

ACTIVE

This value means System State Manager is fully active, monitoring the state of system resources and doing automation tasks when the state of a resource changes.

Other possible values

- INACTIVE

  System State Manager is installed but is not monitoring resources or responding to changes in resource status.

- NONE

  System State Manager is not installed.

- NOPREREQ

  System State Manager is active, but is not checking the status of prerequisites for system resources.

- PASSIVE

  System State Manager is active but will not change the status of a resource at startup time. This setting is useful when you change the status of System State Manager from inactive to active (such as when you restart it).

Set or modify this parameter...

Anytime (unless the value of STATEMAN is NONE because the System State Manager is not installed)

Example: STATEMAN

This makes an installed System State Manager passive:

```
OPSPRM('SET','STATEMAN','PASSIVE')
```

STATEMATCHPREFIX Parameter

The STATEMATCHPREFIX parameter controls whether SSM prerequisite and subrequisite processing uses the defined UP and DOWN states for a resource table as states that must match exactly or as prefix values that must match only up to the length of the state names.

If the current state of a prerequisite resource is ACTIVEX and the defined SSM UP state for the resource is ACTIVE, then a NO value for STATEMATCHPREFIX means the prerequisite resource is not in its UP state (ACTIVEX ¬= ACTIVE). If the value of STATEMATCHPREFIX is YES, the prerequisite resource is in its UP state since ACTIVE contains the first six letters of ACTIVEX.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: STATEMATCHPREFIX

This functions tells SSM to use prerequisite and subrequisite current state matching:

```
OPSPRM('SET','STATEMATCHPREFIX','YES')
```

STATEMAXACTION Parameter

This parameter determines how many actions System State Manager can perform on resources in a given minute.

Default value

10

Other possible values

Any number of actions between 0 and 32767

Set or modify this parameter...

Anytime

Example: STATEMAXACTION

This function limits System State Manager to 200 actions per minute:

```
OPSPRM('SET','STATEMAXACTION','200')
```

STATEMAXWAIT Parameter

This parameter determines how many seconds System State Manager waits between checks of the status of resources it is monitoring.

Default value

120 (seconds)

Other possible values

Any number of seconds between 0 and 32767

Set or modify this parameter...

Anytime

Example: STATEMAXWAIT

This function tells System State Manager to check resource status every six seconds:

```
OPSPRM('SET','STATEMAXWAIT','6')
```

STATENOACTMSG Parameter

This parameter controls whether System State Manager issues the OPS7902H message when it detects a mismatched state for which no action appears in the action table.

Default value

YES

Message OPS7902H is produced even when no action is found for a mismatched state.

Other possible values

NO

Eliminates the OPS7902H message for the NO ACTION FOUND condition. This may be useful for reducing system overhead when large resource tables are in use.

Set or modify this parameter...

Anytime

Example: STATENOACTMSG

This function tells CA OPS/MVS to eliminate message OPS7902H for the NO ACTION FOUND condition:

```
OPSPRM('SET','STATENOACTMSG','NO')
```

STATESCHEDEXCLUDE Parameter

The STATESCHEDEXCLUDE parameter specifies the resource effective mode or modes that are to be excluded from desired state updates by the System State Manager Schedule Manager RESET command. A RESET command occurs whenever a schedule is loaded or during the next scheduled change event. Because updating the desired states of inactive or passive mode resources does not result in the initiation of any SSM actions for the resources, the setting of the desired states can be safely bypassed. The benefit of using this parameter is the elimination of mismatched states that can cause the System State Manager Schedule Group Manager to indicate erroneous group status values for groups that contain inactive resources and exception condition indicators in the SSM Resource Status Monitor. The effective mode of a resource is the mode determined by the hierarchical merging of the global System State Manager mode, the resource table mode, and the individual resource mode. The most restrictive mode of any of the above three modes is the effective mode of a resource.

The Schedule Manager RESET command will issue informational messages indicating that blocks of resources have been bypassed for desired-state updating when the global System State Manager mode or a resource table mode cause all System State Manager Schedule resources or a table of resources to be bypassed.

Default value

NONE

Other possible values

INACTIVE, PASSIVE, BOTH (inactive and passive)

Set or modify this parameter...

Anytime

Example: STATESCHEDEXCLUDE

This function sets the Schedule Manager to skip the setting of desired states for resources with an effective mode of inactive.

```
OPSPRM('SET','STATESCHEDEXCLUDE','INACTIVE')
```

STATETBL Parameter

This parameter identifies a resource directory table containing the names of resource tables that you want System State Manager to manage.

Default value

SSM_MANAGED_TBLS

Other possible values

Any valid table name

Set or modify this parameter...

Anytime

Example: STATETBL

This function identifies the directory table as TABLLIST:

```
OPSPRM('SET','STATETBL','TABLLIST')
```

STATETBLLOG Parameter

The STATETBLLOG parameter determines whether messages are produced in OPSLOG when a change is made to the active SSM directory table using the OPSSMTBL command.

These messages include the MSF system and job names of the OPSMMTBL command issuer and the new values of the directory table entry that was changed.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: STATETBLLOG

This function uses the OPSSMTBL command and OPSVIEW option 4.11.1 to log all changes to the SSM directory table made by users:

```
OPSPRM('SET','STATETBLLOG','YES')
```

Automate-related Parameters

The parameters described in the following sections support Automate.

ATMCMDCHAR Parameter

The ATMCMDCHAR parameter specifies the command character used to issue Automate commands to CA OPS/MVS. The OPAMCMCH rule, which is enabled during CA OPS/MVS initialization, intercepts and performs these Automate commands.

For information about how CA OPS/MVS defines ATMCMDCHAR to the z/OS Command Prefix Facility (CPF), see the *Administration Guide*. For general rules for defining CPF strings, see Characters That Can Be Used in z/OS Commands in the chapter "Using This Reference."

Note: CA OPS/MVS automatically defines the command prefix to the CPF during product initialization and attempts to delete the prefix from the CPF during termination. If some other subsystem is already using this prefix, the prefix will not be defined to the CPF and the following message will be issued during product initialization:

```
OPS0100W CPF DEFINE OF ATMCMDCHAR FAILED, RC=X'00000008', REASON CODE=X'00000008'
```

Reason codes of X'0000000C' and X'00000010' are also indicators of this type of problem since the prefix specified may be either a subset or superset of a prefix that is already defined.

If you change the value of ATMCMDCHAR while CA OPS/MVS is active, the CPF prefixes will *not* be updated and the following message will be issued during product termination:

```
OPS0100W CPF DELETE OF ATMCMDCHAR FAILED, RC=X'00000008', REASON CODE=X'00000004'
```

Notes:

- The value you specify for the ATMCMDCHAR parameter must be different than the Automate command character specified on the Automate CMDCHAR parameter, unless Automate is no longer active on the same z/OS image. Using the same value for both parameters causes unpredictable results.
- If you have not converted from Automate or if you no longer use Automate operator commands, you should set the ATMCMDCHAR parameter to a single blank to avoid wasting a z/OS command prefix (for example, OPSPRM('SET','ATMCMDCHAR',' ').

Important! The ATMCMDCHAR parameter can only contain characters from the table shown in Characters That Can Be Used In z/OS Commands in the chapter "Using This Reference."

Default value

(

Other possible values

Any valid z/OS command recognition character that is not being used by another subsystem; if necessary, you may specify a 2-byte character.

Set or modify this value...

At initialization

Example: ATMCMDCHAR

This function sets the ATMCMDCHAR to !:

```
OPSPRM('SET','ATMCMDCHAR','!')
```

ATMCOMMENTPOS Parameter

When committed Automate-format rules have been recently added to the commit list or have changed, CA OPS/MVS dynamically translates them during initialization and then enables them. To provide all user comments from the original source rules in the output, translated rules, you need to use the ATMCOMMENTPOS parameter to indicate the position of comments relative to the major keyword of each rule.

Default value

BEFORE

Comments for a rule may immediately precede the rule major keyword and be intermixed in rule text. Comments are to be saved in the output translated rules.

Other possible values

- AFTER

  Comments for a rule follow the rule major keyword and may be intermixed in rule text. Comments are to be saved in the output translated rules.

- NONE

  Any source rule comments are not to be retained in the output, translated rules. At enablement time, this parameter setting can cause insignificant storage use.

Set or modify this parameter...

Anytime

Example: ATMCOMMENTPOS

This function indicates that no source rule comments should be retained in the output, translated rules:

```
OPSPRM('SET','ATMCOMMENTPOS','NONE')
```

ATMCOMMIT Parameter

The ATMCOMMIT parameter determines whether Automate-format rules in the commit list are enabled during product initialization.

Default value

MEMLIST

Enable all committed rules during product initialization.

Other possible values

- NONE

  Do not enable any committed rules during product initialization.

- FASTMEMLIST

  Bypass comparison check of source rule and its associated translated CA OPS/MVS rule.

Note: Any committed rule that is modified while CA OPS/MVS is not running must be replaced or retranslated when you start CA OPS/MVS.

Set or modify this parameter...

At initialization

Example: ATMCOMMIT

This function indicates that you do not want any committed rules to be enabled during CA OPS/MVS initialization:

```
OPSPRM('SET','ATMCOMMIT','NONE')
```

ATMLOCALSCOPEn Parameter

The ATMLOCALSCOPE*n* (where *n* is 03-32) parameter determines the set of local variable scope prefixes that are used at your site. Set the value of this parameter to the value specified in the Automate LOCALSV parameter.

Note: Do *not* set ATMLOCALSCOPE01 and ATMLOCALSCOPE02, which are reserved for the @ and LOCAL_ default local prefixes.

Default value

Null string

Other possible values

Alphanumeric characters or any of these special characters: #, $, @, or _

Set or modify this parameter...

Anytime

Examples: ATMLOCALSCOPE*n*

The first example sets ATMLOCALSCOPE03 to the value LOCSCOPE1_, and the second example sets ATMLOCALSCOPE04 to the value LOCSCOPE2_:

```
OPSPRM('SET','ATMLOCALSCOPE03','LOCSCOPE1_')

OPSPRM('SET','ATMLOCALSCOPE04','LOCSCOPE2_')
```

ATMSHAREDSCOPEn Parameter

The ATMSHAREDSCOPE*n* (where *n* is 02-32) parameter determines the set of shared variable scope prefixes in use at your site. Set the value of this parameter to the value specified in the Automate SHAREDSV parameter.

Note: Do *not* set ATMSHAREDSCOPE01, which is reserved for the SHARED_ default shared prefix.

Default value

Null string

Other possible values

Alphanumeric characters or any of these special characters: #, $, @, or _

Set or modify this parameter...

Anytime

Examples: ATMSHAREDSCOPE*n*

The first example sets ATMSHAREDSCOPE02 to the value SHARESCOPE1_, and the second example sets ATMSHAREDSCOPE03 to the value SHARESCOPE2_:

```
OPSPRM('SET','ATMSHAREDSCOPE02','SHARESCOPE1_')

OPSPRM('SET','ATMSHAREDSCOPE03','SHARESCOPE2_')
```

ATMTEMPSCOPEn Parameter

The ATMTEMPSCOPE*n* (where *n* is 02-32) parameter determines the set of temporary variable scope prefixes in use at your site. Set the value of this parameter to the value specified in the Automate TEMPSV parameter.

Note: Do *not* set ATMTEMPSCOPE01, which is reserved for the TEMP_ default temporary prefix.

Default value

Null string

Other possible values

Alphanumeric characters or any of these special characters: #, $, @, or _

Set or modify this parameter...

Anytime

Examples: ATMTEMPSCOPE*n*

The first example sets ATMTEMPSCOPE02 to the value TEMPSCOPE1_, and the second example sets ATMTEMPSCOPE03 to the value TEMPSCOPE2_:

```
OPSPRM('SET','ATMTEMPSCOPE02','TEMPSCOPE1_')

OPSPRM('SET','ATMTEMPSCOPE03','TEMPSCOPE2_')
```

Chapter 4: Parameters for Basic Interfaces

This section contains the following topics:

Using OPSPARM to Control Interaction with Other Software

This chapter describes the parameters that allow you to tailor how CA OPS/MVS interfaces to other software such as JES and TSO.

Using the OPSPARM command processor, you can set parameters to control how CA OPS/MVS interacts with JES and TSO.

Changing or Displaying Parameter Values

To change the values of parameters controlling CA OPS/MVS interfaces or display those values, use the OPSPRM function of OPS/REXX (preferable) or the OPSPARM command processor (an alternative). For a description of OPSPRM and OPSPARM and their syntax, see Setting or Displaying Parameter Values in the chapter “Parameters.”

JES-related Parameters

The parameters in the following sections affect the CA OPS/MVS interaction with JES.

JES2CHECKUPCOMMAND Parameter (for JES2 only)

This parameter specifies the command that CA OPS/MVS issues to determine if JES2 is active and processing commands.

Default value:

$DCONDEF

Other possible values

Any other valid JES2 command

Set or modify this parameter:

Anytime

Example: JES2CHECKUPCOMMAND

This function issues the JES2 $DQ command:

```
OPSPRM('SET','JES2CHECKUPCOMMAND','$DQ')
```

JES2OFFSETSUFFIX Parameter (for JES2 only)

This parameter lets users share a common CA OPS/MVS library among multiple CPUs with different releases of JES2 by specifying a suffix for the CA OPS/MVS OPJ2CB module.

Note: After CA OPS/MVS starts and you change the value of the JES2OFFSETSUFFIX parameter, the OPJ2CB module is automatically reloaded.

Default value

If you specify no suffix, CA OPS/MVS uses the OPJ2CB module assembled at your site.

Other possible values

Any two-character suffix

Besides specifying a suffix, you need to:

- Assemble and link the OPJ2CB module with that suffix.
- Modify the linkage editor control cards as follows:

```
PUNCH 'ENTRY OPJ2CB'
PUNCH 'NAME  OPJ2CBxx(R)
```

The *xx* value is the suffix.

Note: If a value is specified for JES2OFFSETSUFFIX, then the CA OPS/MVS load library must contain the OPJ2CB module that is included on the CA OPS/MVS distribution media and the OPJ2CBxx members that are created.

Set or modify this parameter:

Anytime

Example: JES2OFFSETSUFFIX

This function specifies 42 as the suffix for the OPJ2CB module:

```
OPSPRM('SET','JES2OFFSETSUFFIX','42')
```

JES3SYSID Parameter

This parameter determines the MSF system ID of the system, normally a JES3 global system, to which you want to route JES3 commands, from a JES3 local system, that are not routed by using a JES3 PLEXSYN (sysplex scope) CPF prefix. Normally you should allow sysplex to route JES3 commands. However, it may be desirable to use MSF to route commands from a JES3 local system in one JESPLEX to a JES3 global system in another JESPLEX in the following cases:

- When the JESPLEXes are in different sysplexes
- When the JESPLEXes are in the same sysplex and there are no sysplex scope JES3 PLEXSYN prefixes to do the command routing

Only commands that begin with one of the characters specified in the JESCHAR parameter are ever routed in this way.

The copy of CA OPS/MVS that is running in a JES3 local processor uses this value to determine the JES3 global MSF ID. Except for a few special JES3 commands (*CALL DSI, *CANCEL DSI, *DUMP, *RETURN, and *START DSI), all JES3 commands issued by CA OPS/MVS running in the local processor are sent to the global processor through the MSF.

Default value

A null string

Other possible values

A valid MSF system ID (should be the MSFID of some JES3 global system in the MSF complex)

Set or modify this parameter...

Anytime

Example: JES3SYSID

This function sets the JES3 system ID to JESS:

```
OPSPRM('SET','JES3SYSID','JESS')
```

JESCHAR Parameter

This parameter defines the JES command character.

Notes:

- In a JES2 environment where multiple copies of JES2 are running, the command character for the primary JES2 system must be the first one listed.
- In a JES3 environment the first character of JESCHAR is the character you will use when writing AOF CMD rules to intercept JES3 commands. You only need to write a single CMD rule for JES3 commands, no matter which command prefix (system or sysplex scope) was used when the command was issued. CA OPS/MVS translates the actual prefix used, to the first character specified in JESCHAR, for AOF rule purposes. For more information on writing CMD rules, see the *AOF Rules User Guide*. Specifically see the variables CMD.JES3PREFIX, CMD.JES3PLEXSYN and CMD.JES3SYN.
- In a JES3 environment where the character 8 is allowed as a system command prefix, you must specify 8 as the second character in the JESCHAR string. If not, only specify the first character of JESCHAR.

Important! The JESCHAR parameter can only contain characters from the table shown in Characters That Can Be Used In z/OS Commands in the chapter "Using This Reference."

Default value

$ for JES2 sites; *8 for JES3 sites

Other possible values

Any two-character string (based on the notes above). Typically, a one-character value is specified in JES2 sites.

Set or modify this parameter...

Anytime

Example: JESCHAR

This function sets the JES command character to #:

```
OPSPRM('SET','JESCHAR','#')
```

JESNAME Parameter

This parameter specifies the name of the address space for the primary JES running on your system, allowing CA OPS/MVS to monitor JES startups and terminations.

Default value

JES2 or JES3 (CA OPS/MVS determines and sets the default automatically at initialization)

Other possible values

Any other valid started task name for JES

Note: The only time that you should change the default value of the JESNAME parameter is if one of these conditions is true:

- You use JES2, but the started task name and the subsystem name are not JES2 (for example, JESA).
- You use JES3, but you have modified the JES3 source code so that both the started task name and the subsystem name are not JES3 (for example, JES9).

Set or modify this parameter...

Anytime

Example: JESNAME

This function identifies JESA as the name of the primary JES on the system:

```
OPSPRM('SET','JESNAME','JESA')
```

TSO-related Parameters

The parameters described in the following sections allow you to tailor the CA OPS/MVS handling of TSO messages.

TSODESC Parameter

This parameter specifies the default descriptor codes of messages generated by commands sent to the server through ADDRESS TSO commands.

There are actually two ways that you can set these descriptor codes, because the descriptor codes that you specify for TSODESC are the same as the descriptor codes specified for bytes 7 and 8 of the TSODEST parameter value. For example, if you set a value for TSODEST, and then set a value for TSODESC specifying different descriptor codes, the codes set by TSODESC override the earlier settings. The opposite is also true-if you set the TSODESC parameter and then later specify different descriptor codes through the TSODEST parameter, the TSODEST settings override the TSODESC settings.

Note: For a description of the TSODESC parameter, see TSODESC Parameter in this chapter.

To help you to set the TSODESC parameter, the following table lists this information:

- Descriptor codes 1 through 11
- The hexadecimal representation of each code
- The corresponding OPSBITS() character string of each code
- A description of each code

Code	Hex Representation	OPSBITS() String	Description
1	X'8000'	SYSFAIL	System failure
2	X'4000'	IMEDACTN	Immediate action required
3	X'2000'	EVENACTN	Eventual action required
4	X'1000'	SYSSTAT	System status
5	X'0800'	IMEDCMD	Immediate command response
6	X'0400'	JOBSTAT	Job status
7	X'0200'	APPLPRGM	Application program
8	X'0100'	OOLMSG	Out-of-line message
9	X'0080'	OPERREQ	Request of the operator
10	X'0040'	DYNSTAT	TRACK command response
11	X'0020'	CRITEVET	Critical eventual action required

You can combine two or more hexadecimal representations to specify multiple descriptor codes, but keep in mind that descriptor codes 1-6 and descriptor code 11 are all mutually exclusive.

Consider these examples:

To represent descriptor codes...	Specify this value for TSODESC...
6	X'0400'
2 and 7	X'4200'
5, 8, and 9	X'0980'

For more information about descriptor codes, see the IBM documentation.

Default value

X'0000'

Other possible values

Any valid descriptor codes, expressed as hexadecimal

Set or modify this parameter...

Anytime

Example: TSODESC

This function sets the default descriptor codes for TSO messages to codes 2 and 7 (X'4000' + X'0200' = X'4200'):

```
OPSPRM("SET","TSODESC","X'4200'")
```

TSODEST Parameter

This parameter specifies the default destination of messages generated by commands sent to servers through ADDRESS TSO clauses in AOF rules. TSODEST is valid only when the CA OPS/MVS message has one of these severity levels: I (informational), W (warning), E (error), or S (severe error).

Default value:

X'0000000000000000'

Other possible values:

Any valid destination, expressed as hexadecimals

To specify a value for TSODEST, it may be helpful for you to think of the value as 8 bytes of information.

These 8 bytes break down as follows:

- The first byte can be either C3, which is the hexadecimal representation of the character C; or C2, which is the hexadecimal representation of the character B.
  - If you specify C3, the messages are sent to the specified console (that is, the console specified by the fourth byte of the TSODEST value) and also to OPSLOG Browse.
  - If you specify C2, the messages are sent to OPSLOG Browse only, and the second through eighth bytes of the TSODEST value are ignored.
- The second byte of the TSODEST value is reserved; this value should always be 00.
- The third byte of the TSODEST value indicates the MCS flags code. There are two values you can specify for this byte: 00 (for X'00') indicating that the messages should be queued for the console and hard copy; and 02 (for X'02') indicating the messages should be queued for hard copy only.

  Note: The integer value of X'02' is 7; the OPSBITS() character string that corresponds to this value is HRDCPY.
- The fourth byte indicates the default console ID. You can specify a console ID from 00 to 99, but you must represent it as hexadecimals (the decimal values 00 through 99 are equivalent to the hexadecimal values X'00' through X'63').
- The fifth and sixth bytes indicate the default routing codes for the messages. The routing codes that you specify here are the same as the routing codes that you may specify for the TSOROUTE parameter. For more information, see TSOROUTE Parameter in this chapter. In other words, if you use the TSOROUTE parameter to set default routing codes for TSO messages and then later use bytes 5 and 6 of the TSODEST value to set *different routing codes*, the values you set last override the earlier settings. The opposite is also true. For a list of routing codes and their descriptions, see TSOROUTE Parameter in this chapter.

- The seventh and eighth bytes indicate the default descriptor codes for the messages. The descriptor codes that you specify here are the same as those that you may specify for the TSODESC parameter. For more information, see TSODESC Parameter in this chapter. In other words, if you use the TSODESC parameter to set default descriptor codes for TSO messages and then later use bytes 7 and 8 of the TSODEST value to set *different descriptor codes*, the values you set last override the earlier settings. The opposite is also true. For a list of descriptor codes and their descriptions, see TSODESC Parameter in this chapter.

Set or modify this parameter: Anytime

Example: TSODEST

This function sends messages resulting from commands sent to TSO to console 5:

```
OPSPRM('SET','TSODEST','C300000500000000')
```

TSOROUTE Parameter

This parameter determines the default routing codes of messages generated by commands sent to the server through ADDRESS TSO clauses.

There are actually two ways to set these routing codes, because the routing codes that you specify for TSOROUTE are the same as those specified for bytes 5 and 6 of the TSODEST parameter value. For a description of the TSODEST parameter, see TSODEST Parameter in this chapter. For example, if you set a value for TSODEST, and then set a value for TSOROUTE specifying different routing codes, the codes set by TSOROUTE override the earlier settings. The opposite is also true-if you set the TSOROUTE parameter and then later specify different routing codes through the TSODEST parameter, the TSODEST settings override the TSOROUTE settings.

To help you to set the TSOROUTE parameter, the following table lists this information:

- Routing codes 1 through 12
- The hexadecimal representation of each code
- The corresponding OPSBITS() character string of each code
- A description of each code

Code	Hex Representation	OPSBITS() String	Description
1	X'8000'	MSTRACTN	Master console action
2	X'4000'	MSTRINFO	Master console information
3	X'2000'	TAPEPOOL	Tape pool

Code	Hex Representation	OPSBITS() String	Description
4	X'1000'	DASDPOOL	Direct access pool
5	X'0800'	TAPELIB	Tape library
6	X'0400'	DISKLIB	Disk library
7	X'0200'	UR	Unit record pool
8	X'0100'	TP	Teleprocessing control
9	X'0080'	SECURITY	System security
10	X'0040'	SYSERROR	System/Error maintenance
11	X'0020'	PGMRINFO	Programmer information
12	X'0010'	EMULATOR	Emulator

You can combine two or more hexadecimal representations to specify multiple routing codes. Consider these examples:

To represent routing codes...	Specify this value for TSOROUTE...
2	X'4000'
1 and 2	X'C000'
9 and 11	X'00A0'

If you need more detailed information about routing codes, see the IBM documentation.

Default value

X'0000'

Other possible values

Any valid routing codes, expressed as hexadecimals

Set or modify this parameter...

Anytime

Example: TSOROUTE

This function sets the default routing codes to 1 and 2 (X'8000' + X'4000' = X'C000'):

```
OPSPRM("SET","TSOROUTE","X'C000'")
```

Chapter 5: Parameters for Optional Interfaces

This section contains the following topics:

How OPSPARM Controls Interaction with Other Software

This chapter describes the parameters that allow you to tailor how CA OPS/MVS interfaces to other optional software such as CA Automation Point, CA 7, CA Netman, CA Network and Systems Management System Status Manager CA OPS/MVS Option, UNIX System Services (USS), and CA Service Desk.

Note: These interfaces are CA OPS/MVS-base product features, but can optionally be turned off at initialization.

For a list of parameters for optional interfaces that are *not* part of the CA OPS/MVS base product, see the chapter “Parameters for Optional Interfaces for Separately Licensed Features.”

Using the OPSPARM command processor, you can set parameters to control how the CA OPS/MVS product interacts with optional software such as CA 7, CA Netman, CA Network and Systems Management System Status Manager CA OPS/MVS Option, and USS.

Change or Display Parameter Values

To change or display the values of parameters controlling CA OPS/MVS interfaces, use the OPSPRM function of OPS/REXX (preferable) or the OPSPARM command processor (an alternative). For a description of OPSPRM and OPSPARM and their syntax, see Setting or Displaying Parameter Values in the chapter “Parameters.”

Application Program Interface Parameters

The parameters described in the following sections determine how CA OPS/MVS interacts with the Application Program Interface (API).

APIACTIVE Parameter

The APIACTIVE parameter enables or disables the interface that applications outside of CA OPS/MVS can use to provide information to CA OPS/MVS rules or read information from CA OPS/MVS.

Default value

YES

This value allows applications outside of CA OPS/MVS to provide information to CA OPS/MVS rules or read information from CA OPS/MVS.

Other possible values

NO

This value prevents applications outside of CA OPS/MVS from providing information to CA OPS/MVS rules or reading information from CA OPS/MVS.

Set or modify this parameter...

Anytime

Example: APIACTIVE

This function allows applications outside of CA OPS/MVS to provide information to CA OPS/MVS rules or read information from CA OPS/MVS:

```
OPSPRM('SET','APIACTIVE','YES')
```

DEBUGAPI Parameter

The DEBUGAPI parameter determines whether CA OPS/MVS generates API-related trace messages.

Default value

NO

This value prevents CA OPS/MVS from generating API-related trace messages.

Other possible values

YES

This value causes CA OPS/MVS to generate API-related trace messages.

Set or modify this parameter...

Anytime

Example: DEBUGAPI

This function causes CA OPS/MVS to generate API-related trace messages:

```
OPSPRM('SET','DEBUGAPI','YES')
```

CA 7-related Parameters

The parameters described in the following sections determine how CA OPS/MVS interacts with CA 7.

INITCA7 Parameter

The INITCA7 parameter determines whether CA OPS/MVS detects CA 7 browse ENF events.

Default value

OFF|NO

This value prevents CA OPS/MVS from detecting CA 7 browse ENF events.

Other possible values

ON|YES

This value enables CA OPS/MVS to detect CA 7 browse ENF events.

Set or modify this parameter...

Anytime

Example: INITCA7

This function causes CA OPS/MVS to detect CA 7 browse ENF events:

```
OPSPRM('SET','INITCA7','YES')
```

Note: Changes take effect only the next time the CA OPS/MVS CAIENF interface task is recycled through the F OPSx,RESTART(CA7) command.

If the INITCA7 parameter is set to a value of ON and CAIENF is not available, the highlighted OPx0301S error message will be issued at regular intervals until CAIENF is started. If you want to avoid these error messages, we suggest that rather than unconditionally setting this parameter to a value of ON in the initial REXX exec, you provide automation to set this parameter appropriately and restart the CAIENF interface when CAIENF is available. Sample rules CAS9200I and CAS9300E are provided to do this.

CAIENF-related Parameters

The parameters described in the following sections determine how CA OPS/MVS interacts with CAIENF.

CAIENFHIGH Parameter

This parameter displays the high watermark that the CAIENF rate control mechanism has reached during the current life of the product or since this parameter value was last reset. This is the highest value that the CAIENFCURRENT parameter has ever reached since that time. Compare the CAIENFHIGH value to the value specified on the CAIENFMAX parameter to determine how close CA OPS/MVS has come to the point at which it would shut itself down due to exceeding the CAIENFMAX limit.

Note: Resetting this parameter affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default value

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: CAIENFHIGH

This function resets the CAIENFHIGH value to zero:

```
OPSPRM('SET','CAIENFHIGH',0)
```

CAIENFMAX Parameter

This parameter limits the number of CAIENF events that CA OPS/MVS can receive in a given second before it shuts down the CAIENF interface. This parameter is designed to prevent a loop in CAIENF processing from disabling all of your automation.

Note: The CAIENF task is activated and CAIENF events are passed to CA OPS/MVS only if the INITCA7 or INITCPM parameters are set to either ON or YES.

The CAIENFMAX parameter dictates the maximum value of the CAIENF counter kept by the CAIENFCURRENT parameter. When the value of CAIENFCURRENT reaches the value of CAIENFMAX, the CA OPS/MVS ENF interface terminates but can be restarted with a MODIFY command. Setting the CAIENFMAX parameter to its maximum possible value prevents CA OPS/MVS from ever shutting down the ENF interface task due to an excessive CAIENF rate. Doing so could impact your other automation.

Default value

1000 (if the CAIENF interface is active)

Other possible values

Any number of CAIENF events between 100 and 100000

Set or modify this parameter...

Anytime

Example: CAIENFMAX

This function sets the CAIENF event limit to 500 events:

```
OPSPRM('SET','CAIENFMAX','500')
```

Note: If the CAIENF interface is inactive and has never been activated, the default value will be zero if it was not previously set.

CAIENFRATE Parameter

This parameter sets the rate by which CA OPS/MVS decrements the value of the CAIENFCURRENT counter every second.

Default value

50 (if the CAIENF interface is active)

Other possible values

Any number between 1 and 1,000

Set or modify this parameter...

Anytime

Example: CAIENFRATE

This function decrements the CAIENFCURRENT counter by 5 each second:

```
OPSPRM('SET','CAIENFRATE','5')
```

Note: If the CAIENF interface is inactive and has never been activated, the default value will be zero if it was not previously set.

DEBUGENF Parameter

The DEBUGENF parameter determines whether CA OPS/MVS generates ENF-related trace messages. ENF is the acronym for Event Notification Facility.

Default value

OFF|NO

This value prevents CA OPS/MVS from generating ENF-related trace messages.

Other possible values

ON|YES

This value causes CA OPS/MVS to generate ENF-related trace messages.

Set or modify this parameter...

Anytime

Example: DEBUGENF

This function causes CA OPS/MVS to generate ENF-related trace messages:

```
OPSPRM('SET','DEBUGENF','YES')
```

CA Netman-related Parameters

The parameters described in the following sections determine how CA OPS/MVS interacts with CA Netman.

INITNETMAN Parameter

Initializes the CA Netman interface and issues a NETMAN API SIGNON at the startup of OPSMAIN.

Default value

NO

This value prevents CA OPS/MVS from initializing the CA Netman interface.

Other possible values

YES

This value causes CA OPS/MVS to initialize the CA Netman interface.

Set or modify this parameter...

At initialization

Example: INITNETMAN

This function causes CA OPS/MVS to initialize the CA Netman interface:

```
OPSPRM('SET','INITNETMAN','YES')
```

NETMANDATABASE Parameter

Sets the CA Netman database that CA OPS/MVS updates.

Default value

No default

Other possible values

Any valid CA Netman database

Set or modify this parameter...

At initialization

Example: NETMANDATABASE

This function causes CA OPS/MVS to update the PRODDATA CA Netman database:

```
OPSPRM('SET','NETMANDATABASE','PRODDATA')
```

NETMANAPIID Parameter

Sets the requestor ID and password that CA OPS/MVS will use when signing on to CA Netman. To designate a password that is different from the requestor ID, use OPSVIEW option 4.8 to set the global variable GLOBAL1.NETMAN.PASSWORD.

Default value

No default

Other possible values

Any valid CA Netman requestor ID and password

Set or modify this parameter...

At initialization

Example: NETMANAPIID

This function causes CA OPS/MVS to set the requestor ID and password to M/G:

```
OPSPRM('SET','NETMANAPIID','M/G')
```

CA NSM SSM CA OPS/MVS Option-related Parameters

The parameters described in the following sections determine how CA OPS/MVS interacts with the CA Network and Systems Management System Status Manager CA OPS/MVS Option.

CACPMTABLE Parameter

Defines the System State Manager (SSM) resource table name used to contain the job flow information that CA OPS/MVS obtains from its interface to CA CPM Version 3. That information is transmitted to the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation to support the Critical Path Monitoring (CPM) feature of that product. Only one CPM table per system is allowed. The CPM table is also added to the System State Manager directory table.

Note: The CPM table is built automatically by the interface to CA CPM Version 3. Before changing the value of this parameter, please see the chapter "Critical Path Monitoring" in the *Administrators Guide*.

Default value

CPM_SSM_TABLE

Other possible values

Any valid relational table name of 1-18 characters

Set or modify this parameter...

At initialization

Example: CACPMTABLE

This function sets the CPM job flow table name to a name more descriptive of the system on which it is being used:

```
OPSPRM('SET','CACPMTABLE','CPM_SYS_A001')
```

The CACPMTABLE parameter is required to activate the interface to CA CPM Version 3. For detailed information about the interface, see the chapter “Critical Path Monitoring” in the *Administration Guide*.

CAUNIALLOWSET Parameter

Determines whether requests to alter selected values in System State Manager resource tables are allowed. The alter value request originates from the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation product as an SNMP set request. If this parameter is set to NO, the workstation does not permit the requests to be issued. If it is set to YES, the requests are sent by the workstation and allowed to change a value in an SSM resource table.

Default value

NO|OFF

Other possible values

YES|ON

Set or modify this parameter...

Anytime

Example: CAUNIALLOWSET

This function permits workstation System State Manager resource value changes:

```
OPSPRM('SET','CAUNIALLOWSET','YES')
```

CAUNICONFIGSET Parameter

The CAUNICONFIGSET parameter is the PDS member name of the Agent Technologies configuration set that defines the MIB, CA Network and Systems Management System Status Manager CA OPS/MVS Option workstations, and SNMP community names to be used by Agent Technologies for SNMP communication with the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstations (see the sample configuration file in CNTL(CFGSSMO)). The member name specified must be specifically configured for the CA Network and Systems Management System Status Manager CA OPS/MVS Option product only.

Default value

OPSCNFG

Other possible values

Any valid PDS member name of a defined Agent Technologies configuration set

Set or modify this parameter...

Anytime

Example: CAUNICONFIGSET

This function sets the CA Agent Technologies configuration set name to an alternate name:

```
OPSPRM('SET','CAUNICONFIGSET','ALTCNFG')
```

Note: If the CA Network and Systems Management System Status Manager CA OPS/MVS Option agent task of the product is already active, any parameter changes to CAUNICONFIGSET will not take effect until the CA Network and Systems Management System Status Manager CA OPS/MVS Option task is restarted.

CAUNICONNECTWAIT Parameter

Determines how many minutes the CA OPS/MVS subtask running in the CA OPS/MVS address space waits between re-attempts to connect to the CA Agent Technologies interface (AWS) subtask running on the same z/OS image.

Default value

0 (no attempt to connect is made)

Other possible values

Any number from 0 to 120

Note: The recommended value is 2.

Set or modify this parameter...

Anytime

Example: CAUNICONNECTWAIT

This function sets the number of minutes that the CA OPS/MVS subtask waits to 2:

```
OPSPRM('SET','CAUNICONNECTWAIT','2')
```

The CAUNICONNECTWAIT parameter is required to activate the interface to CA CPM Version 3. For detailed information about that interface, see the chapter “Critical Path Monitoring” in the *Administration Guide*.

CAUNIDEBUG Parameter

Debugs the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component. We usually recommend that you use this parameter *only* at the direction of CA Customer Support.

Default value

NO|OFF

This value indicates that you do *not* want to debug the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component.

Other possible values

YES|ON

This value indicates that you want to debug the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component. All SNMP trap data is logged in the OPSLOG. It can then be displayed using OPSLOG Browse.

Set or modify this parameter...

Anytime

Example: CAUNIDEBUG

This function enables you to debug the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component:

```
OPSPRM('SET','CAUNIDEBUG','YES')
```

The CAUNIDEBUG parameter may also be used to debug the interface to CA CPM Version 3. For detailed information about that interface, see the chapter “Critical Path Monitoring” in the *Administration Guide*.

CAUNITRACE Parameter

Controls the generation of trace messages by the SNMP AWS subagent for SNMP trap requests issued by the workstation task in the CA OPS/MVS address space.

Regardless of the setting for this parameter, default trace output is directed to the OPSALOG and OPSBLOG DD files, which must be allocated to the CA OPS/MVS address space before you can use the ca Network and Systems Management System Status Manger CA OPS/MVS Option and the interface to CA CPM Version 3.

If CA OPS/MVS is started under the Master subsystem (SUB=MSTR), then the OPSALOG and OPSBLOG files must not be allocated to SYSOUT=outputclass data sets. Instead, real data sets must be allocated. Sample JCL is provided in the .CCLXiCNTL install library, member name ALLOCAWS, to allocate real data sets for this trace output.

We suggest that you add code similar to the following to OPS*x*PA00, which is called by the OPSTART1 list:

```
If Substr (Opsinfo('JOBID'),1,1) = 'S' Then
 Do
  Address TSO "Allocate FI(OPSALOG) SYSOUT(H)"
  Address TSO "Allocate FI(OPSBLOG) SYSOUT(H)"
 End
Else
 Do
  Address TSO "Allocate FI(OPSALOG)
              DSN('SYS1.OPS.OPSALOG') SHR"
  Address TSO "Allocate FI(OPSBLOG)
              DSN('SYS1.OPS.OPSBLOG') SHR"
 End
```

Default value

X'00'

Other possible values

X'01' through X'7F'

Set or modify this parameter...

Anytime, but only at the direction of CA Customer Support.

Example: CAUNITRACE

This function activates SNMP AWS subagent tracing:

```
OPSPRM('SET','CAUNITRACE',"X'3F"')
```

Note: Changes take effect when the CA Network and Systems Management System Status Manager CA OPS/MVS Option subtask is restarted.

APIACTIVE Parameter

The APIACTIVE parameter enables or disables the interface that applications outside of CA OPS/MVS can use to provide information to CA OPS/MVS rules or read information from CA OPS/MVS.

Default value

YES

This value allows applications outside of CA OPS/MVS to provide information to CA OPS/MVS rules or read information from CA OPS/MVS.

Other possible values

NO

This value prevents applications outside of CA OPS/MVS from providing information to CA OPS/MVS rules or reading information from CA OPS/MVS.

Set or modify this parameter...

Anytime

Example: APIACTIVE

This function allows applications outside of CA OPS/MVS to provide information to CA OPS/MVS rules or read information from CA OPS/MVS:

```
OPSPRM('SET','APIACTIVE','YES')
```

DEBUGAPI Parameter

The DEBUGAPI parameter determines whether CA OPS/MVS generates API-related trace messages.

Default value

NO

This value prevents CA OPS/MVS from generating API-related trace messages.

Other possible values

YES

This value causes CA OPS/MVS to generate API-related trace messages.

Set or modify this parameter...

Anytime

Example: DEBUGAPI

This function causes CA OPS/MVS to generate API-related trace messages:

```
OPSPRM('SET','DEBUGAPI','YES')
```

INITCA7 Parameter

The INITCA7 parameter determines whether CA OPS/MVS detects CA 7 browse ENF events.

Default value

OFF|NO

This value prevents CA OPS/MVS from detecting CA 7 browse ENF events.

Other possible values

ON|YES

This value enables CA OPS/MVS to detect CA 7 browse ENF events.

Set or modify this parameter...

Anytime

Example: INITCA7

This function causes CA OPS/MVS to detect CA 7 browse ENF events:

```
OPSPRM('SET','INITCA7','YES')
```

Note: Changes take effect only the next time the CA OPS/MVS CAIENF interface task is recycled through the F OPSx,RESTART(CA7) command.

If the INITCA7 parameter is set to a value of ON and CAIENF is not available, the highlighted OPx0301S error message will be issued at regular intervals until CAIENF is started. If you want to avoid these error messages, we suggest that rather than unconditionally setting this parameter to a value of ON in the initial REXX exec, you provide automation to set this parameter appropriately and restart the CAIENF interface when CAIENF is available. Sample rules CAS9200I and CAS9300E are provided to do this.

CAIENFHIGH Parameter

This parameter displays the high watermark that the CAIENF rate control mechanism has reached during the current life of the product or since this parameter value was last reset. This is the highest value that the CAIENFCURRENT parameter has ever reached since that time. Compare the CAIENFHIGH value to the value specified on the CAIENFMAX parameter to determine how close CA OPS/MVS has come to the point at which it would shut itself down due to exceeding the CAIENFMAX limit.

Note: Resetting this parameter affects the data recorded in the product SMF records and the reports produced by the AME.

Default value

No default value

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: CAIENFHIGH

This function resets the CAIENFHIGH value to zero:

```
OPSPRM('SET','CAIENFHIGH',0)
```

CAIENFMAX Parameter

This parameter limits the number of CAIENF events that CA OPS/MVS can receive in a given second before it shuts down the CAIENF interface. This parameter is designed to prevent a loop in CAIENF processing from disabling all of your automation.

Note: The CAIENF task is activated and CAIENF events are passed to CA OPS/MVS only if the INITCA7 or INITCPM parameters are set to either ON or YES.

The CAIENFMAX parameter dictates the maximum value of the CAIENF counter kept by the CAIENFCURRENT parameter. When the value of CAIENFCURRENT reaches the value of CAIENFMAX, the CA OPS/MVS ENF interface terminates but can be restarted with a MODIFY command. Setting the CAIENFMAX parameter to its maximum possible value prevents CA OPS/MVS from ever shutting down the ENF interface task due to an excessive CAIENF rate. Doing so could impact your other automation.

Default value

1000 (if the CAIENF interface is active)

Other possible values

Any number of CAIENF events between 100 and 100000

Set or modify this parameter...

Anytime

Example: CAIENFMAX

This function sets the CAIENF event limit to 500 events:

```
OPSPRM('SET','CAIENFMAX','500')
```

Note: If the CAIENF interface is inactive and has never been activated, the default value will be zero if it was not previously set.

CAIENFRATE Parameter

This parameter sets the rate by which CA OPS/MVS decrements the value of the CAIENFCURRENT counter every second.

Default value

50 (if the CAIENF interface is active)

Other possible values

Any number between 1 and 1,000

Set or modify this parameter...

Anytime

Example: CAIENFRATE

This function decrements the CAIENFCURRENT counter by 5 each second:

```
OPSPRM('SET','CAIENFRATE','5')
```

Note: If the CAIENF interface is inactive and has never been activated, the default value will be zero if it was not previously set.

DEBUGENF Parameter

The DEBUGENF parameter determines whether CA OPS/MVS generates ENF-related trace messages. ENF is the acronym for Event Notification Facility.

Default value

OFF|NO

This value prevents CA OPS/MVS from generating ENF-related trace messages.

Other possible values

ON|YES

This value causes CA OPS/MVS to generate ENF-related trace messages.

Set or modify this parameter...

Anytime

Example: DEBUGENF

This function causes CA OPS/MVS to generate ENF-related trace messages:

```
OPSPRM('SET','DEBUGENF','YES')
```

INITNETMAN Parameter

Initializes the CA Netman interface and issues a NETMAN API SIGNON at the startup of OPSMAIN.

Default value

NO

This value prevents CA OPS/MVS from initializing the CA Netman interface.

Other possible values

YES

This value causes CA OPS/MVS to initialize the CA Netman interface.

Set or modify this parameter...

At initialization

Example: INITNETMAN

This function causes CA OPS/MVS to initialize the CA Netman interface:

```
OPSPRM('SET','INITNETMAN','YES')
```

NETMANDATABASE Parameter

Sets the CA Netman database that CA OPS/MVS updates.

Default value

No default

Other possible values

Any valid CA Netman database

Set or modify this parameter...

At initialization

Example: NETMANDATABASE

This function causes CA OPS/MVS to update the PRODDATA CA Netman database:

```
OPSPRM('SET','NETMANDATABASE','PRODDATA')
```

NETMANAPIID Parameter

Sets the requestor ID and password that CA OPS/MVS will use when signing on to CA Netman. To designate a password that is different from the requestor ID, use OPSVIEW option 4.8 to set the global variable GLOBAL1.NETMAN.PASSWORD.

Default value

No default

Other possible values

Any valid CA Netman requestor ID and password

Set or modify this parameter...

At initialization

Example: NETMANAPIID

This function causes CA OPS/MVS to set the requestor ID and password to M/G:

```
OPSPRM('SET','NETMANAPIID','M/G')
```

CACPMTABLE Parameter

Defines the System State Manager (SSM) resource table name used to contain the job flow information that CA OPS/MVS obtains from its interface to CA CPM Version 3. That information is transmitted to the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation to support the Critical Path Monitoring (CPM) feature of that product. Only one CPM table per system is allowed. The CPM table is also added to the System State Manager directory table.

Note: The CPM table is built automatically by the interface to CA CPM Version 3. Before changing the value of this parameter, please see the chapter "Critical Path Monitoring" in the *Administrators Guide*.

Default value

CPM_SSM_TABLE

Other possible values

Any valid relational table name of 1-18 characters

Set or modify this parameter...

At initialization

Example: CACPMTABLE

This function sets the CPM job flow table name to a name more descriptive of the system on which it is being used:

```
OPSPRM('SET','CACPMTABLE','CPM_SYS_A001')
```

The CACPMTABLE parameter is required to activate the interface to CA CPM Version 3. For detailed information about the interface, see the chapter “Critical Path Monitoring” in the *Administration Guide*.

CAUNIALLOWSET Parameter

Determines whether requests to alter selected values in System State Manager resource tables are allowed. The alter value request originates from the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation product as an SNMP set request. If this parameter is set to NO, the workstation does not permit the requests to be issued. If it is set to YES, the requests are sent by the workstation and allowed to change a value in an SSM resource table.

Default value

NO|OFF

Other possible values

YES|ON

Set or modify this parameter...

Anytime

Example: CAUNIALLOWSET

This function permits workstation System State Manager resource value changes:

```
OPSPRM('SET','CAUNIALLOWSET','YES')
```

CAUNICONFIGSET Parameter

The CAUNICONFIGSET parameter is the PDS member name of the Agent Technologies configuration set that defines the MIB, CA Network and Systems Management System Status Manager CA OPS/MVS Option workstations, and SNMP community names to be used by Agent Technologies for SNMP communication with the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstations (see the sample configuration file in CNTL(CFGSSMO)). The member name specified must be specifically configured for the CA Network and Systems Management System Status Manager CA OPS/MVS Option product only.

Default value

OPSCNFG

Other possible values

Any valid PDS member name of a defined Agent Technologies configuration set

Set or modify this parameter...

Anytime

Example: CAUNICONFIGSET

This function sets the CA Agent Technologies configuration set name to an alternate name:

```
OPSPRM('SET','CAUNICONFIGSET','ALTCNFG')
```

Note: If the CA Network and Systems Management System Status Manager CA OPS/MVS Option agent task of the product is already active, any parameter changes to CAUNICONFIGSET will not take effect until the CA Network and Systems Management System Status Manager CA OPS/MVS Option task is restarted.

CAUNICONNECTWAIT Parameter

Determines how many minutes the CA OPS/MVS subtask running in the CA OPS/MVS address space waits between re-attempts to connect to the CA Agent Technologies interface (AWS) subtask running on the same z/OS image.

Default value

0 (no attempt to connect is made)

Other possible values

Any number from 0 to 120

Note: The recommended value is 2.

Set or modify this parameter...

Anytime

Example: CAUNICONNECTWAIT

This function sets the number of minutes that the CA OPS/MVS subtask waits to 2:

```
OPSPRM('SET','CAUNICONNECTWAIT','2')
```

The CAUNICONNECTWAIT parameter is required to activate the interface to CA CPM Version 3. For detailed information about that interface, see the chapter “Critical Path Monitoring” in the *Administration Guide*.

CAUNIDEBUG Parameter

Debugs the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component. We usually recommend that you use this parameter *only* at the direction of CA Customer Support.

Default value

NO|OFF

This value indicates that you do *not* want to debug the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component.

Other possible values

YES|ON

This value indicates that you want to debug the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component. All SNMP trap data is logged in the OPSLOG. It can then be displayed using OPSLOG Browse.

Set or modify this parameter...

Anytime

Example: CAUNIDEBUG

This function enables you to debug the CA Network and Systems Management System Status Manager CA OPS/MVS Option workstation component:

```
OPSPRM('SET','CAUNIDEBUG','YES')
```

The CAUNIDEBUG parameter may also be used to debug the interface to CA CPM Version 3. For detailed information about that interface, see the chapter “Critical Path Monitoring” in the *Administration Guide*.

CAUNIUSERCURRENT Parameter

Specifies the current state value that must be matched by an active SSM resource to be assigned the user status and icon in the CA NSM map. If left defaulted to blank and CAUNIUSERDESIRED is specified, only the desired state is used to test the resource for user status assignment.

Default value

No default

Other possible values

Any System State Manager current state value

Set or modify this parameter...

Anytime

Example: CAUNIUSERCURRENT

This function sets the current state value for the user defined Unicenter status assignment:

```
OPSPRM('SET','CAUNIUSERCURRENT','DOWN')
```

CAUNIUSERDESIRED Parameter

Specifies the desired state value that must be matched by an active SSM resource to be assigned the user status and icon in the CA NSM map. If left defaulted to blank and CAUNIUSERCURRENT is specified, only the current state is used to test the resource for user status assignment.

Default value

No default

Other possible values

Any System State Manager desired state value

Set or modify this parameter...

Anytime

Example: CAUNIUSERDESIRED

This function sets the desired state value for the user defined Unicenter status assignment:

```
OPSPRM('SET','CAUNIUSERDESIRED','DOWN')
```

INITAWS Parameter

Determines whether the CA Network and Systems Management System Status Manager CA OPS/MVS Option interface is initialized. If you have not licensed this Option, specify NO.

Default value

NO|OFF

Other possible values

YES|ON

Sites that have licensed the CA Network and Systems Management System Status Manager CA OPS/MVS Option should set INITAWS to YES.

Set or modify this parameter...

At initialization

Example: INITAWS

This function prevents CA OPS/MVS from initializing the CA Network and Systems Management System Status Manager CA OPS/MVS Option:

```
OPSPRM('SET','INITAWS','NO')
```

Note: Changes take effect only after the CA NSM interface task of CA OPS/MVS is recycled through the F OPSx,RESTART(SSMO) command.

The INITAWS parameter is required to activate the interface to CA CPM Version 3. For detailed information about that interface, see the chapter “Critical Path Monitoring” in the *Administration Guide.*

INITCPM Parameter

You must set the parameter INITCPM to YES in the OPSSPA00 (or equivalent) member before starting the interface to CA CPM Version 3.

For detailed information about that interface, see the chapter “Critical Path Monitoring” in the *Administration Guide*.

If the INITCPM parameter is set to a value of ON and CAIENF is not available, the highlighted OPx0301S error message will be issued at regular intervals until CAIENF is started. If you want to avoid these error messages, we suggest that rather than unconditionally setting this parameter to a value of ON in the initial REXX exec, you provide automation to set this parameter appropriately and restart the CAIENF interface when CAIENF is available. Sample rules CAS9200I and CAS9300E in library OPS.CCLXSAMP are provided to do this.

Default value

NO|OFF

Other possible values:

YES|ON

Set or Modify

Anytime

Example: INITCPM

The following activates the CA OPS/MVS CPM ENF event gathering feature:

```
OPSPRM("SET","INITCPM","YES")
```

Note: Changes take effect only the next time the CA OPS/MVS CAIENF interface task is recycled through the F OPSx,RESTART(CPM) command.

CA Automation Point-related Parameters

The parameters described in the following sections determine how CA OPS/MVS interacts with the CA Automation Point.

APDEFAULTUSERID Parameter

The APDEFAULTUSERID parameter must be set to a valid security USERID defined to the z/OS security package (CA ACF2, CA Top Secret, or RACF).

Default value

None

Other possible values

None

Set or modify this parameter...

Anytime

Example: SECUID28

UNIX System Services-related Parameters

The parameters described in the following sections determine how CA OPS/MVS interacts with UNIX System Services.

INITUSS Parameter

The INITUSS parameter determines whether the product activates the optional USS command and USS rule interfaces. If USS support is not desired, leave this parameter set to NO to reduce overhead. The INITUSS parameter must be set to YES for most of the other USS parameters to have an effect. The exception is for the INITUSSPROC related parameters. This includes INITUSSPROC, USSPROCRULES and BROWSEUSSPROC. The INITUSS parameter should only be set to YES if you intend to use the USS interfaces in CA OPS/MVS that deal with the use of the USS servers

Default value

NO

Other possible values

YES

Set or modify this parameter...

At initialization

Example: INITUSS

This function activates support for USS commands and rules:

```
OPSPRM('SET','INITUSS','YES')
```

INITUSSPROC Parameter

Determines whether the USS process dynamic exit module, OPUSPREX, is installed in the z/OS USS process exit list.

Default value

NO

Other possible values

YES

Set or modify this parameter...

At initialization

Example: INITUSSPROC

This function tells CA OPS/MVS to install OPUSPREX in the z/OS process exit list:

```
OPSPRM('SET','INITUSSPROC','YES')
```

Note: Once the exits have been installed by setting INITUSSPROC=YES, they can only be removed by a system IPL or by issuing the following z/OS commands:

```
SETPROG EXIT,DELETE,EXITNAME=BPX_POSPROC_INIT,MODNAME=OPUSPREX
SETPROG EXIT,DELETE,EXITNAME=BPX_IMAGE_INIT,MODNAME=OPUSPREX
SETPROG EXIT,DELETE,EXITNAME=BPX_PREPROC_TERM,MODNAME=OPUSPREX
```

Setting INITUSSPROC=NO will not cause the exits to be de-installed.

USSACTIVE Parameter

Specifies whether USS commands are allowed to execute. This parameter should be set to ON at initialization if USS command processing is being used.

If a problem arises, USS command processing can be dynamically suspended and then resumed.

Default value

OFF

Other possible values

ON

Set or modify this parameter...

Anytime

Example: USSACTIVE

This function allows USS command processing to proceed:

```
OPSPRM('SET','USSACTIVE','ON')
```

USSALLOWRESTART Parameter

Determines whether the product restarts USS servers that fail during initialization. When the USS server task is not initializing properly under any circumstances, a slow restart loop occurs if this parameter is set to YES.

This parameter is useful when Open Edition (OMVS) is not immediately available at system initialization or for some short period of time.

Default value

NO

Other possible values

YES

Note: At 30 second intervals, the product attempts to restart USS servers that failed during initialization.

Set or modify this parameter...

Anytime

Example: USSALLOWRESTART

This function sets automatic restart of USS servers that fail during initialization to YES.

```
OPSPRM('SET','USSALLOWRESTART','YES')
```

USSDORM Parameter

Determines, in seconds, how long OSF USS servers can remain dormant before they are automatically terminated. This parameter takes effect only if the number of active OSF USS servers is greater than the value of the USSMIN parameter.

Default value

60

Other possible values

Any number of seconds between 60 and 16777216

Set or modify this parameter...

Anytime

Example: USSDORM

This function allows 300 seconds, or 5 minutes, of OSF USS server inactivity before a USS server is terminated.

```
OPSPRM('SET','USSDORM','300')
```

USSMAX Parameter

Sets the maximum number of OSF USS servers that can be active at any time. If the USSMAX value exceeds the current number of active USS servers, CA OPS/MVS automatically stops the additional servers.

We strongly recommend that USSMAX be left at the default value.

Default value

8

Other possible values

Any number of USS servers between 1 and 30, but at least as many as the number of servers specified for the USSMIN parameter. If you attempt to set the USSMAX parameter value lower than the USSMIN value, CA OPS/MVS sets the USSMAX equal to USSMIN

Set or modify this parameter...

Anytime

Example: USSMAX

This function sets 6 as the maximum number of OSF USS servers:

```
OPSPRM('SET','USSMAX','6')
```

USSMIN Parameter

Sets the minimum number of OSF USS servers that can be active at any time. If the USSMIN value exceeds the current number of active USS servers, CA OPS/MVS automatically starts additional servers.

We strongly recommend that USSMIN be left at the default value. Do not set this parameter to a value of 1. When the INITUSS parameter is set to NO, setting this value has no effect. For an example of how to terminate all the OSF USS servers, see the sample STOPUSS REXX EXEC.

Default value

2

Other possible values

Any number of USS servers between 0 and 30, but fewer than the number of servers specified for the USSMAX parameter. If you set the USSMIN parameter value higher than the USSMAX value, CA OPS/MVS sets USSMIN equal to USSMAX.

Set or modify this parameter...

Anytime

Example: USSMIN

This function sets 0 as the minimum number of OSF USS servers:

```
OPSPRM('SET','USSMIN','0')
```

Note: The sample STOPUSS REXX program demonstrates how to stop all the OSF USS servers. You may need to use this procedure if you plan to shut down the Job Entry Subsystem (JES) prior to shutting down CA OPS/MVS.

USSOUTLIM Parameter

Limits the number of server and command messages that are written to the syslog as requested by the log keyword.

Default value

1000 (lines)

Other possible values

Any number of lines greater than zero

Set or modify this parameter...

Anytime

Example: USSOUTLIM

This function limits the number of output lines to 500:

```
OPSPRM('SET','USSOUTLIM','500')
```

USSPARM Parameter

Lets you add a keyword=option string that CA OPS/MVS includes in the z/OS START command for the USS started task. USSPARM also allows you to override the corresponding substitution parameter in the cataloged procedure.

Default value

No default

Other possible values

Any valid keyword=option string

Set or modify this parameter...

Anytime

Example: USSPARM

This function adds the string VER=450 to the START command for OSF USS servers:

```
OPSPRM('SET','USSPARM','VER=450')
```

USSPROCRULES Parameter

Determines whether USS process events are processed using USS AOF rules.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: USSPROCRULES

This function ensures that USS process events are processed using USS AOF rules:

```
OPSPRM('SET','USSPROCRULES','YES')
```

USSQADD Parameter

Sets the threshold CA OPS/MVS uses to determine whether more OSF USS servers need to be started. When the number of commands in the OSF USS execution queue exceeds the value of the USSQADD parameter, CA OPS/MVS checks to see if the current number of servers is less than the value of the USSMAX parameter and equal to or greater than the value of the USSMIN parameter. If so, then CA OPS/MVS starts another USS server.

Default value

20

Other possible values

Any number of queued USS commands greater than or equal to 0

Set or modify this parameter...

Anytime

Example: USSQADD

This function sets the OSF USS execution queue threshold to 5 commands:

```
OPSPRM('SET','USSQADD','5')
```

USSQUE Parameter

Specifies the maximum number of queued OSF USS commands. This queue is where CA OPS/MVS sends USS commands to be executed in servers. The OSF execute processor dispatches these commands to OSF USS servers as the servers become available to process work.

Default value

1700 (commands)

Other possible values

Any number from 1 to 32768

Set or modify this parameter...

At initialization

Example: USSQUE

This function sets the size of the OSF USS execution queue to 1024:

```
OPSPRM('SET','USSQUE','1024')
```

USSRUN Parameter

Determines how long CA OPS/MVS allows a USS transaction to execute in an OSF USS server.

When any USS server transaction exceeds the time limit set by USSRUN, CA OPS/MVS terminates the server with the hung or looping transaction and starts another USS server to accept new commands.

Default value

600 (seconds)

Other possible values

Any number of seconds between 1 and 604800 (7 days), inclusive

Set or modify this parameter...

Anytime

Example: USSRUN

This function sets a 60-second time limit for USS transactions:

```
OPSPRM('SET','USSRUN','60')
```

USSSECURITY Parameter

The USSSECURITY parameter determines which type of security check the OSF USS server performs.

Default value

NOSECURITY

This value allows the OSF USS server to operate with the security privileges assigned to the user ID that is associated with OPSUSS started task.

Other possible values

CHECKUSERID

If you specify this value, CA OPS/MVS uses the security privileges assigned to the issuer of the command, as follows:

- For work sent to servers from rules, the security privileges are those assigned to the user ID that is associated with OPSUSS started task.
- For work sent to servers from TSO users using the ADDRESS USS host command environment, the security privileges are those assigned to the TSO user ID.

Set or modify this parameter

At initialization

Example: USSSECURITY

The following function tells CA OPS/MVS to operate the OSF USS server under the authority of the Address USS requester.

```
OPSPRM('SET','USSSECURITY','CHECKUSERID')
```

USSSTC Parameter

Specifies the name of the started task catalogued procedure to be used for USS servers.

A sample procedure named OPSUSS is provided with the product.

Default value

OPSUSS

Other possible values

Any valid started task name

Set or modify this parameter...

At initialization

Example: USSSTC

This function sets the name of the USS server task to OPSUNIX:

```
OPSPRM('SET','USSSTC','OPSUNIX')
```

USSSWAPPABLE Parameter

Specifies whether USS servers are set as non-swappable by the operating system at server initialization. Setting this parameter to YES should only be necessary if USS commands are heavily used in a continuous mode. The overhead of a swap event is avoided at the expense of some real storage frames.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: USSSWAPPABLE

This function makes USS servers that start after this command is issued non-swappable:

```
OPSPRM('SET','USSSWAPPABLE','NO')
```

Hardware Services (HWS) -related Parameters

The parameters described in the following sections control Hardware Services (HWS).

INITHWS Parameter

The INITHWS parameter determines whether the product activates the optional HWS interface. If hardware services are not desired, leave this parameter set to NO to reduce overhead. The INITHWS parameter must be set to YES for the other hardware service parameters to have an effect. This includes HWSRULES and the Address HWS host command environment. The INITHWS parameter should only be set to YES if you intend to use the CA OPS/MVS hardware services.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

If you are changing the value of INITHWS from NO to YES after CA OPS/MVS initialization, you must issue the z/OS command: MODIFY OPSS,RESTART(HWS) for the new value to take effect (where OPSS is the CA OPS/MVS subsystem name).

Example: INITHWS

This function activates support for HWS:

```
OPSPRM('SET','INITHWS','YES')
```

HWSRULES Parameter

The HWSRULES parameter determines whether the product activates the optional hardware event notification and associated rule interfaces. If hardware event notification support is not desired, leave this parameter set to NO to reduce overhead. The INITHWS parameter must also be set to YES if HWSRULES is set to YES in order for HWSRULES to take effect. The HWSRULES parameter should only be set to YES if you intend to use the hardware event notification function of HWS.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: HWSRULES

This function activates support for hardware event notification and associated rules:

```
OPSPRM('SET','HWSRULES','YES')
```

Note: Setting HWSRULES to YES activates the OPS hardware event notification function. When this function is activated, OPS receives hardware events as API events that can be automated through OPS)API rules.

For detailed information on the hardware event types and associated variables see, "Hardware Event API Rules" in the *CA OPS/MVS AOF Rules User Guide.*

Since hardware events are presented as OPS)API events, the OPS API interface must also be activated to receive the hardware events. To activate the OPS API interface, set the OPS APIACTIVE parameter to YES:

```
OPSPRM('SET','APIACTIVE','YES')
```

For more information on the APIACTIVE parameter, see Application Programming Interface Parameters and for information on coding API rules and specific information for coding hardware event API rules, see "Generic Event Application Program Interface" in the *CA OPS/MVS AOF Rules User Guide.*

DEBUGHWS Parameter

The DEBUGHWS parameter determines whether CA OPS/MVS generates HWS-related trace messages.

Default value

OFF

This value prevents CA OPS/MVS from generating HWS-related trace messages.

Other possible values

ON

This value causes CA OPS/MVS to generate HWS-related trace messages.

Set or modify this parameter...

Anytime

Example: DEBUGHWS

This function causes CA OPS/MVS to generate HWS-related trace messages:

```
OPSPRM('SET','DEBUGHWS','ON')
```

Hardware Interface Service Configuration

The Hardware Interface Service provides CA Technologies products with a common interface/API for accessing hardware services. CA OPS/MVS uses the Hardware Interface Service to implement HWS. Therefore, the Hardware Interface Service must be configured and started on the system where CA OPS/MVS is running in order for HWS to provide its hardware functions.

For more information see, "Configure Hardware Services (HWS)" in the *CA OPS/MVS Installation Guide*.

CA Service Desk-related Parameters

The parameter described in the following section determines how CA OPS/MVS interacts with CA Service Desk.

INITSD Parameter

Determines whether CA OPS/MVS attempts to open CA Service Desk requests for a variety of problems detected internally by CA OPS/MVS.

Default value

OFF|NO

This value prevents CA OPS/MVS from making requests to CA Service Desk.

Other possible values

ON|YES

This value enables CA OPS/MVS to make requests to CA Service Desk.

Set or modify this parameter...

Anytime

Example: INITSD

This function enables CA OPS/MVS to create CA Service Desk requests:

```
OPSPRM('SET','INITSD','YES')
```

Note: Changes take effect only the next time the CA OPS/MVS CA Service Desk interface task is recycled through the following command:

```
F OPSx,RESTART(ServDesk)
```

Linux Connector Interface Related Parameters

The parameters described in the following section determine how CA OPS/MVS interfaces with the Linux Connector component.

INITLXC Parameter

Determines whether CA OPS/MVS attempts to connect to the Linux Connector component in order to receive unsolicited VM and Linux messages and issue commands to connected VM and Linux system.

Default value

OFF|NO

This value prevents CA OPS/MVS from connecting to the Linux Connector.

Other possible values

ON|YES

This value allows CA OPS/MVS to connect to the Linux Connector.

Set or modify this parameter...

Anytime

Example: INITLXC

This function enables CA OPS/MVS to connect to the Linux Connector:

```
OPSPRM('SET','INITLXC','YES')
```

Note: Changes only take effect the next time the CA OPS/MVS Linux Connector interface task is recycled by the following command:

```
F OPSx,RESTART(LXC)
```

DEBUGLXC Parameter

The DEBUGLXC parameter determines whether CA OPS/MVS generates Linux Connector interface related trace messages.

Default value

OFF

This value prevents CA OPS/MVS from generating Linux Connector interface related trace messages.

Other possible values

ON

This value causes CA OPS/MVS to generate Linux Connector interface related trace messages.

Set or modify this parameter...

Anytime

Example: DEBUGLXC

This function causes CA OPS/MVS to generate Linux Connector interface trace messages:

```
OPSPRM('SET','DEBUGLXC','ON')
```

LXCONMSG Parameter

The LXCONMSG parameter contains the z/OS name/token pair name whose token value is the IP port number that CA OPS/MVS uses to connect to the Linux Connector component unsolicited message server. The value of this parameter must match the MSGTOKEN parameter of the desired Linux Connector component.

Default value

CALINUXUNSOLMSG:

This is a common value for the MSGTOKEN parameter.

Other possible values

Any other 1-16 character string that matches the MSGTOKEN parameter of the Linux Connector component.

Set or modify this parameter

Anytime

Example: LXCONMSG

This function causes CA OPS/MVS to use an alternate Linux Connector component z/OS Name/Token designated IP port number.

```
OPSPRM('SET','LXCONMSG','CALXCONMSGTOKEN2')
```

Note: Changes only take effect the next time the CA OPS/MVS Linux Connector interface task is recycled by the following command:

```
F OPSx,RESTART(LXC)
```

LXCONCMD Parameter

The LXCONCMD parameter contains the z/OS name/token pair name whose token value is the IP port number that CA OPS/MVS uses to connect to the Linux Connector component command server. The value of this parameter must match the CMDTOKEN parameter of the desired Linux Connector component.

Default value

CALINUXCOMMANDS:

This is a common value for the CMDTOKEN parameter.

Other possible values

Any other 1-16 character string that matches the CMDTOKEN parameter of the Linux Connector component.

Set or modify this parameter...

Anytime

Example: LXCONCMD

This function causes CA OPS/MVS to use an alternate Linux Connector component z/OS Name/Token designated command IP port number.

```
OPSPRM('SET','LXCONMSG','CALXCONCMDTOKEN2')
```

LXCONHIGH Parameter

This parameter displays the high watermark that the Linux Connector interface rate control mechanism has reached during the current life of the product or since this parameter value was last reset. This value is the highest value that the LXCONCURRENT parameter has ever reached since the last reset. Compare the LXCONHIGH value to the LXCONMAX parameter to determine how close CA OPS/MVS has come to shutting down the Linux Connector interface due to exceeding the LXCONMAX limit.

Default value

No default value

Other possible values

The only possible value you can set this parameter to is zero.

Set or modify this parameter...

Anytime

Example: LXCONHIGH

This function resets the LXCONHIGH value to zero:

```
OPSPRM('SET','LXCONHIGH',0)
```

LXCONMAX Parameter

This parameter limits the number of LXCON events that CA OPS/MVS can receive in a given second before it shuts down the Linux Connector interface. This parameter is designed to prevent a loop in the Linux Connector interface processing from disabling all of your automation.

Note: The Linux Connector interface task is activated and LXCON events are passed to CA OPS/MVS only if the INITLXC parameter is set to either ON or YES.

The LXCONMAX parameter dictates the maximum value of the LXCON counter kept by the LXCONCURRENT parameter. When the value of LXCONCURRENT reaches the value of LXCONMAX, the Linux Connector interface terminates but can be restarted with a MODIFY command. Setting the LXCONMAX parameter to its maximum value prevents CA OPS/MVS from ever shutting down the Linux Connector interface task due to an excessive LXCON rate. Doing so could impact your other automation.

Default value

1000 (if the LXCON interface is active).

Other possible values

Any number of LXCON events from 100 through100000

Set or modify this parameter...

Anytime

Example: LXCONMAX

This function sets the LXCON event limit to 500 events:

```
OPSPRM('SET','LXCONMAX','500')
```

Note: If the parameter is not set before the initial start of the Linux Connector Interface, the value is zero.

LXCONRATE Parameter

This parameter sets the rate by which CA OPS/MVS decrements the value of the LXCONCURRENT counter every second.

Default value

50 (if the LXCON interface is active).

Other possible values

Any number from 1 through 1,000

Set or modify this parameter...

Anytime

Example: LXCONRATE

This function decrements the LXCONCURRENT counter by 5 each second:

```
OPSPRM('SET','LXCONRATE','5')
```

Note: If the parameter is not set before the initial start of the Linux Connector Interface, the value is zero.

Chapter 6: Parameters for Optional Interfaces for Separately Licensed Features

This section contains the following topics:

Using OPSPARM to Control Interaction with Other Software

This chapter describes the parameters that allow you to tailor the CA OPS/MVS optional components CICS, ESI, IMS, and MSF.

Important! The interfaces discussed in this chapter are not part of the CA OPS/MVS base product-they are separately licensed features.

Each of the optional interfaces described in this chapter has an associated parameter of the form INIT*xxx* that controls whether the optional interface is to be activated during CA OPS/MVS initialization. In order to set any of these parameters to the YES value, you must be a licensed user of the interface, and you must have already obtained and set up an LMP key. For additional information on LMP keys, see the *Administration Guide*.

Using the OPSPARM command processor, you can set parameters to control how CA OPS/MVS interacts with optional components such as CICS, ESI, IMS, and MSF.

Changing or Displaying Parameter Values

You can change the values of parameters controlling CA OPS/MVS interfaces or display those values using the following methods:

- The OPSPRM function of OPS/REXX (preferable)
- The OPSPARM command processor (an alternative)

For a description of OPSPRM and OPSPARM and their syntax, see Setting or Displaying Parameter Values in the chapter “Parameters.”

CAICCI-related Parameters

CAICCI itself is not a separately licensed feature. It is a sub-component of the Multi-System Facility (MSF), which is a separately licensed feature. If you are not licensed to use MSF, you cannot use CAICCI.

INITCCI Parameter

Determines whether the CAICCI interface is to be activated when a remote MSF system is defined as a CCI type.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: INITCCI

This function sets the INITCCI parameter to YES:

```
OPSPRM('SET','INITCCI','YES')
```

Note: If you change the value of this parameter, you must restart MSF in order for the changes to take effect. For information on restarting MSF, see the *User Guide*.

CCIMSGQSZ Parameter

Determines the size of the CCI message queue.

Default value

1024

Other possible values

Any number from 0 to 10,000

Set or modify this parameter...

Anytime

Example: CCIMSGQSZ

This function sets the size of the message queue to 5,000:

```
OPSPRM('SET','CCIMSGQSZ','5000')
```

Note: If you change the value of this parameter, you must restart MSF in order for the changes to take effect. For information on restarting MSF, see the *User Guide*.

CCITRACE Parameter

Turns the tracing of CAICCI calls on or off. Do *not* modify the value of the CCITRACE parameter unless you are instructed to do so by a CA Customer Support representative.

Default value

OFF

Other possible values

ON

Set or modify this parameter...

Anytime

Example: CCITRACE

This function sets tracing ON:

```
OPSPRM('SET','CCITRACE','ON')
```

CICS-related (COF) Parameters

The parameters described in the following sections allow you to tailor the CA OPS/MVS CICS interface.

Note: The CICS Operations Facility (COF) is not part of the CA OPS/MVS base product-it is a separately licensed feature.

CICSAOF Parameter

Determines whether connections between CA OPS/MVS and active CICS regions that are using the CICS global exit (XTDOUT) are permitted.

When the value of the CICSAOF parameter is NO, CICS transient data messages are not sent to the AOF for processing. However, any existing connections to the CICS region remain in place, and you can reactivate them by setting CICSAOF to YES. The setting of CICSAOF has no effect on the availability of the ADDRESS OPSCTL COF host command environment.

Note: If your system runs more than one copy of CA OPS/MVS, you can use the CICSAOF parameter to limit which copy processes CICS messages, and as a switch to turn off CICS message traffic while AOF rules are being modified.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: CICSAOF

This function causes connections to the COF interface to be permitted:

```
OPSPRM('SET','CICSAOF','YES')
```

CICSCONSNAME Parameter

Identifies the console that CA OPS/MVS uses to issue WTO messages for messages that the CICS Operations Facility (COF) generates. CA OPS/MVS uses the CICSCONSNAME parameter value only when the WTOCICS parameter is set to YES. For more information, see the description of the WTOCICS parameter in this chapter.

Default value

Blanks

Other possible values

Any valid 2- to 8-character console name. For information about valid console names, see the definition of CONSOLxx in the IBM documentation.

Set or modify this parameter...

Anytime

Example: CICSCONSNAME

This function sets the ID for the CICS console to CONS01:

```
OPSPRM('SET','CICSCONSNAME','CONS01')
```

CICSDELETE Parameter

Determines whether CICS messages that are trapped by the XTDOUT CICS exit may be deleted from the appropriate CICS transient data queue by AOF rules.

Note: When you are writing rules for CICS messages and the COF interface is active, you can determine the exit type by using the OPS/REXX OPSINFO function as follows:

var = OPSINFO('EXITTYPE')

This function returns the type of the exit in which the AOF captured the event that triggered the rule. If the exit type is CICS, the message is from the COF interface. If the exit type is MVS, either CICS issued the message through a WTO, or the COF interface issued a WTO because the WTOCICS parameter is set to YES. When the exit type is CICS and the CICSDELETE parameter is set to YES, an AOF message rule return value of SUPPRESS or DELETE has the same effect, deleting the message from the CICS transient data queue. If CICSDELETE is set to NO, the SUPPRESS or DELETE value is ignored. When the exit type is MVS, the SUPRESS and DELETE return values operate as with any z/OS message and have no effect on the CICS transient data queue.

Default value

NO

CA OPS/MVS will not allow suppression of transient data queue messages by AOF rules. CA OPS/MVS will not honor a return value of either SUPPRESS or DELETE from an AOF rule, and the CICS message appears in the CICS transient data queue.

Other possible values

YES

CA OPS/MVS will allow suppression of transient data queue messages by AOF rules. CA OPS/MVS honors the return value from the AOF rule.

Set or modify this parameter...

Anytime

Example: CICSDELETE

This function permits AOF suppression of transient data queue messages:

OPSPRM('SET','CICSDELETE','YES')

CICSDESC Parameter

This parameter, valid only if the WTOCICS parameter is set to YES, determines the descriptor code that CA OPS/MVS uses to issue WTOs for COF-generated messages.

Default value

x'0000'

Other possible values

Any valid descriptor code

Set or modify this parameter...

Anytime

Example: CICSDESC

This function sets the descriptor code to 7 for WTO messages issued in response to CICS messages:

```
OPSPRM('SET','CICSDESC','7')
```

CICSMSGID Parameter

Use the CICSMSGID parameter to force a default message ID for AOF processing of CICS messages when a transient data queue message does not begin with an ID of either DFH or OPS342. Setting this parameter has no effect on the message text. You can write a single rule to process the unidentified messages.

Default value

No default

Other possible values

Any one- to eight-character message ID

Set or modify this parameter...

Anytime

Example: CICSMSGID

This function sets CICSMSG as the CICS default message ID for AOF processing:

```
OPSPRM('SET','CICSMSGID','CICSMSG')
```

CICSROUTE Parameter

This parameter, valid only if the WTOCICS parameter is set to YES, supplies the routing code that CA OPS/MVS uses for WTOs for COF-generated messages.

Default value

x'0000'

Other possible values

Any valid routing code, expressed as hexadecimals

Set or modify this parameter...

Anytime

Example: CICSROUTE

This function sets the default routing code to 12:

```
OPSPRM('SET','CICSROUTE','12')
```

CICSTIMER Parameter

Determines whether CA OPS/MVS automates message OPS3420O, which reports the status of CICS operations. If you specify YES, CA OPS/MVS generates message OPS3420O every 15 minutes.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: CICSTIMER

This function prevents CA OPS/MVS from automatically issuing message OPS3420O:

```
OPSPRM('SET','CICSTIMER','NO')
```

CICSTIMERDEST Parameter

Use the CICSTIMERDEST parameter to determine the name of the CICS destination that receives the CICS status message (message OPS3420O). The CICS XTDOUT exit intercepts this message. You can use an AOF rule to suppress the OPS3420O message.

If more than one copy of CA OPS/MVS is active, the copy with OPSS as its subsystem name controls the destination of the message. If the OPSS subsystem is inactive, the subsystem that appears first on the z/OS subsystem control table chain controls the destination of the message.

Default value

CSMT

Other possible values

Any one- to four-character CICS destination name

Set or modify this parameter...

Anytime

Example: CICSTIMERDEST

This function specifies that the OPS3420O message is to be sent to the CSCS destination:

```
OPSPRM('SET','CICSTIMERDEST','CSCS')
```

CICSTIMERINTERVAL Parameter

Determines the number of minutes that elapse between the issuance of the CICS status message, which is the OPS3420O message.

If more than one copy of CA OPS/MVS is active, the copy with OPSS as its subsystem name controls the frequency of the message. If the OPSS subsystem is inactive, the subsystem that appears first on the z/OS subsystem control table chain controls the frequency of the message.

Note: You can use the OPS3420O message as a monitoring tool for a particular CICS region. Since the message should appear at the interval specified by the CICSTIMERINTERVAL parameter if the region is operational, you can write an AOF rule that records the last time the message appears and places the time value in a global variable. A subsequent time-of-day (TOD) rule can periodically check the time of the last message and notify you if the region is no longer producing the message as scheduled.

Default value

15

Other possible values

Any number of minutes between 1 and 5,999

Set or modify this parameter...

Anytime

Example: CICSTIMERINTERVAL

This function specifies that the OPS3420O message should be issued every 60 minutes:

```
OPSPRM('SET','CICSTIMERINTERVAL','60')
```

INITCOF Parameter

Initializes the COF interface for possible connections to active CICS regions that are using the CICS global exit (XTDOUT). When the value of the INITCOF parameter is NO:

- You cannot use the ADDRESS OPSCTL COF host command environment.
- CICS transient data messages are not sent to the AOF for processing.

Default value

NO

Other possible values

YES

Note: Only those customers who have licensed the COF can set the value of the INITCOF parameter to YES.

Set or modify this parameter...

Anytime

Example: INITCOF

This function initializes the COF interface:

```
OPSPRM('SET','INITCOF','YES')
```

WTOCICS Parameter

Determines whether CA OPS/MVS issues WTOs for the CICS messages generated by the COF. Allowing the COF to generate these messages has these benefits:

- Operators can see the CICS messages without looking at a CICS terminal.
- The CICS messages can be in the system log.
- CICS messages are available in OPSLOG on a JES3 global system even if CICS runs in the JES3 local system.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: WTOCICS

This function prevents CA OPS/MVS from issuing WTOs for COF-generated CICS messages:

```
OPSPRM('SET','WTOCICS','NO')
```

ESI-related Parameters

The Expert Systems Interface (ESI) is not part of the CA OPS/MVS base product-it is a separately licensed feature.

INITESI Parameter

Determines whether the Expert Systems Interface (ESI) is initialized.

Default value

NO

Other possible values

YES

Sites that have licensed the ESI should set INITESI to YES.

Set or modify this parameter...

Anytime

Example: INITESI

This function prevents CA OPS/MVS from initializing the ESI:

```
OPSPRM('SET','INITESI','NO')
```

IMS-related (IOF) Parameters

The parameters described in the following sections tailor the operation of the CA OPS/MVS IMS Operations Facility (IOF).

Note: The IMS Operations Facility (IOF) is not part of the CA OPS/MVS base product-it is a separately licensed feature.

DEBUGBMP Parameter

Together with the IMS*n*INITBMP, IMS*n*PSBNAME, and IMS*n*TRANNAME parameters, the DEBUGBMP parameter helps you to control the activation or deactivation of a BMP region that the IOF can use to process IMS commands.

This parameter generates trace messages for BMP execution that are useful for debugging and gathering statistics.

Default values

OFF

Other possible values

ON

Set or modify this parameter...

Anytime

Example: DEBUGBMP

This function turns on tracing for BMP execution:

```
OPSPRM('SET','DEBUGBMP','ON')
```

DEBUGIMSU Parameter

The DEBUGIMSU parameter is used to trace installation and user exit problems. It is recommended that you do not enable this parameter unless directed to do so by CA Customer Support.

Default values

OFF

Other possible values

ON

Set or modify this parameter...

Anytime

Example: DEBUGIMSU

This function turns on tracing for IMS user exits:

```
OPSPRM('SET','DEBUGIMSU','ON')
```

IMSnBMPSTC Parameter

The IMS*n*BMPSTC parameter identifies the IOF BMP started task JCL. For details, see the *Administration Guide.*

Default values

No default

Other possible values

Any 1- to 8-character PDS member name of the IOF BMP started task (STC) JCL

Set or modify this parameter...

Anytime

Example: IMS1BMPSTC

This function specifies IMSBATCH as the member name for the BMP:

```
OPSPRM('SET','IMS1BMPSTC','IMSBATCH')
```

IMSnCHAR Parameter

Defines the command character for an IMS control region that CA OPS/MVS processes through its IMS Operations Facility (IOF). The *n* value is the region number. CA OPS/MVS allows up to 32 IMS control regions in the IOF, each having its own unique command character.

By prefixing the IMS command with this command character, you can send a command to the *n* IMS region without using a REPLY command. CA OPS/MVS automatically determines the WTO request of the IMS region and responds to it with the intended command.

Important! The IMS*n*CHAR parameter can only contain characters from the table shown in Characters That Can Be Used In z/OS Commands in the chapter "Using This Reference." This parameter may be set to a single or double quote (x'7D' or x'7F') by supplying the value as a hexadecimal.

Note: This parameter does not modify the IMS CMDCHAR value defined in the IMS system generation. It is only used as an alias value, to allow an operator to issue a command to IMS*n* without knowing the IMS*n* WTOR r*eplyid*.

Default values

- The / character

  This value is for IMS regions 1 through 32.

- A blank

  This value is for IMS regions 2 through 32.

Other possible values

Any character, as long as each IMS region has a different command character

Set or modify this parameter...

Anytime

Example: IMS7CHAR

This function sets the command character for IMS region 7 to @:

```
OPSPRM('SET','IMS7CHAR','@')
```

IMSnCOLOR Parameter

Defines the display color for messages from each IMS region in the IOF. The value of *n* is the number (1 to 32) of an IMS control region. The color you specify applies only to messages displayed in OPSLOG on color terminals.

Default value

- TURQ (turquoise)

  This value is for IMS region 1.

- NONE

  This value is for all IMS regions 2 through 32

Other possible values:

BLUE, GREEN, PINK, RED, WHITE, YELLOW

Note: Each color is valid for any IMS region.

Set or modify this parameter...

Anytime

Example: IMS4COLOR

This function causes messages from IMS region 4 to display in green:

```
OPSPRM('SET','IMS4COLOR','GREEN')
```

IMSnDROPDUPLICATE Parameter

Determines whether CA OPS/MVS attempts to automatically eliminate duplicate IMS messages from the OPSLOG and AOF rules. The value *n* is a number (in the range of 1 to 32) of an IMS control region.

Many IMS messages are simply passed to the IMS exit (IOF), while many are passed to the IOF *and* WTOd by IMS. You can differentiate between these two cases in your AOF rules by using the OPSINFO("EXITTYPE") function of OPS/REXX. If the message is simply passed to the IOF, the function returns the string IMS. If the message is also WTOd by IMS, the function returns the string MVS.

The EXITTYPE column in the OPSLOG provides the same information as discussed above. The algorithm for eliminating duplicate messages is very simplistic to avoid adding too much overhead to the process. Due to its simplicity, the algorithm is unlikely to work well in an environment that has heavy IMS message traffic. Therefore, we recommend that you leave this parameter set to the default value (NO) and use the OPSINFO("EXITYPE") function in your AOF MSG rules to avoid processing messages from both the IMS and MVS exits.

Default value

NO

This value is for all IMS regions. Use this value when most of the rules automating your IMS messages simply suppress duplicate IMS WTOs. However, when automation rules do more than just suppression and the IMS*n*DROPDUPLICATE parameter is set to NO, the rules must take responsibility for avoiding duplicate processing.

Other possible values

YES

When the parameter value is YES, CA OPS/MVS does not automate the duplicate IMS message.

Set or modify this parameter...

Anytime

Example: IMS*n*DROPDUPLICATE

This function prevents automation for duplicate IMS WTOs from IMS region 5:

```
OPSPRM('SET','IMS5DROPDUPLICATE','YES')
```

IMSnID Parameter

Defines the IMS ID of each region under the control of the IOF. The *n* value is the number (1 to 32) of an IMS region.

If you nullify this parameter, CA OPS/MVS automatically sets it to the ID of the first valid IMS control region it finds active. CA OPS/MVS sees a region as valid when its IMS SVC number is valid.

Default value

No default

Other possible values

Any valid IMS region ID

Set or modify this parameter...

At initialization

Example: IMS*n*ID

This function sets the IMS ID of IMS region 6 to IMS6:

```
OPSPRM('SET','IMS6ID','IMS6')
```

IMSnINITBMP Parameter

Together with the DEBUGBMP, IMS*n*PSBNAME, and IMS*n*TRANNAME parameters, the IMS*n*INITBMP parameter helps you to control the activation or deactivation of a BMP region that the IOF can use to process IMS commands. The value of *n* is a number (1 to 32) of an IMS control region.

This parameter starts or terminates a BMP region. When IMS*n*INITBMP is set to YES, the IOF sends a subset of IMS commands to the BMP for processing. However, the following commands will still be sent to the IMS WTOR: CANCEL, CHECKPOINT, END, ERESTART, EXCLUSIVE, EXIT, FORMAT, HOLD, IAM, LOCK, MODIFY, MSVERIFY, NRESTART, RCLSDST, and RCOMPT.

When terminating the IMS control region, you must set the IMS*n*INITBMP parameter to NO to terminate the BMP-dependent region CA OPS/MVS started for this application. The control region cannot terminate until any dependent regions (including the BMP region) have terminated.

You can automate terminating the BMP region by writing a command rule that responds to IMS termination commands by calling the OPSPRM function of OPS/REXX to set the IMS*n*INITBMP parameter to NO.

Default values

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: IMS*n*INITBMP

This function starts a BMP region that may be used for process IMS commands:

```
OPSPRM('SET','IMS27INITBMP','YES')
```

IMSnPSBNAME Parameter

Together with the DEBUGBMP, IMS*n*INITBMP, and IMS*n*TRANNAME parameters, the IMS*n*INITPSB parameter helps you to control the activation or deactivation of a BMP region that the IOF can use to process IMS commands. The value of *n* is a number (1 to 32) of an IMS control region.

This parameter specifies the program specification block (PSB) name for this BMP region. The name must match the value specified in the PSB=*name* keyword of the APPLCTN macro for the BMP.

The APPLCTN macro must specify these values:

```
PGMTYPE=BATCH, FPATH=NO, SCHDTYPE=PARALLEL
```

Note: These values are the basic requirements for the BMP.

The PSB also requires a PSBGEN/ACBGEN job that includes the following values in the PSBGEN statement:

```
LANG=ASSEM, MAXQ=0, CMPAT=YES, IOASIZE=132
```

No other PSBGEN statements are needed.

Default values

Blanks

Other possible values

Any 1- to 8-character PSB name; PSB naming conventions are set by your installation

Set or modify this parameter...

Anytime

Example: IMS*n*INITPSB

This function specifies IOFBMP as the PSB name of the BMP:

```
OPSPRM('SET','IMS5PSBNAME','IOFBMP')
```

IMSnTRANNAME Parameter

Together with the DEBUGBMP, IMS*n*INITBMP, and IMS*n*PSBNAME parameters, the IMS*n*TRANNAME parameter helps you to control the activation or deactivation of a BMP region that the IOF can use to process IMS commands. The value of *n* is a number (1 to 32) of an IMS control region.

This parameter specifies the transaction name that CA OPS/MVS uses for the BMP region. The name must match the CODE value you specify for the TRANSACT macro that follows the APPLCTN macro mentioned in the description of the IMS*n*PSBNAME parameter.

The TRANSACT macro can include these values:

```
PROCLIM=(10,10), INQUIRE=(YES,RECOVER), SCHD=1
```

Note: The value of the IMS*n*TRANNAME parameter may be the same as the value of the IMS*n*PSBNAME parameter.

Default values

Blanks

Other possible values

Any 1- to 8-character transaction name; transaction naming conventions are set by your installation

Set or modify this parameter...

Anytime

Example: IMS*n*TRANNAME

This function specifies IOFBMP as the transaction name for the BMP:

```
OPSPRM('SET','IMS2TRANNAME','IOFBMP')
```

IMSAOIxOFFSET Parameter

Defines which of 16 possible offsets (indicated by *x*) that CA OPS/MVS uses to determine the address of the IMS AOI exit modules, allowing the CA OPS/MVS AOI exit to take control before the original IMS AOI exit. The offset points to the VCON pointer that IMS module DFSQUE10 uses to link to the AOI exit, DFSAOUE0.

Default value

X'0000'

Other possible values

Any hexadecimal number

Set or modify this parameter...

At initialization

Example: IMSAOI*x*OFFSET

This function sets the offset to X'1234':

```
OPSPRM("SET","IMSAOI14OFFSET","X'1234'")
```

IMSBMPAGN Parameter

Lets you use IMS Application Group Names, or AGNs, for the IOF BMP application. You must use the same AGN for every IMS system on which you plan to run the IOF BMP.

Note: This parameter is only for use by IMS users that must have AGNs in their system generation; for more information, consult with your systems programmer.

Default value

Null

Other possible values

Any eight-character AGN

Set or modify this parameter...

Anytime

Example: IMSBMPAGN

This function sets the AGN to OPSBMP:

```
OPSPRM('SET','IMSBMPAGN','OPSBMP')
```

IMSCMDxOFFSET Parameter

Defines which of 16 possible offsets (indicated by *x*) that CA OPS/MVS uses to determine the address of the IMS command analyzer routine, allowing the CA OPS/MVS CMD exit to take control before the original IMS CMD exit. The offset points to the VCON pointer that IMS module DFSICIO0 uses to link to the AOI exit, DFSICLP0.

Default value

X'0000'

Other possible values

Any four-digit hexadecimal number

Set or modify this parameter...

At initialization

Example: IMSCMD*x*OFFSET

This function sets the offset to X'1234':

```
OPSPRM("SET","IMSCMD3OFFSET","X'1234'")
```

IMSCONSNAME Parameter

Determines the console name that CA OPS/MVS uses to issue WTOs for IMS messages. CA OPS/MVS uses the IMSCONSNAME parameter value only when the WTOIMS parameter is set to YES. For more information, see the description of the WTOIMS parameter in this chapter.

Default value

Blanks

Other possible values

Any valid 2- to 8-character console name. For information about valid console names, see the definition of CONSOLxx in the IBM documentation.

Set or modify this parameter...

Anytime

Example: IMSCONSNAME

This function sets the IMS console ID to CONS01:

```
OPSPRM('SET','IMSCONSNAME','CONS01')
```

IMSDESC Parameter

Supplies the descriptor code that CA OPS/MVS uses to issue WTOs for the IMS messages processed through its AOI exit.

Default value

X'0000'

Other possible values

Any valid descriptor code, expressed as hexadecimals

Set or modify this parameter...

Anytime

Example: IMSDESC

This function sets the descriptor code to X'0001':

```
OPSPRM("SET","IMSDESC","X'0001'")
```

IMSMTO Parameter

The IMSMTO parameter:

- Suppresses all IMS exit type messages destined for the MTO if it is set to ON.
- Affects ALL IMS systems that are using the OPS/MVS IOF interface.

No message suppression rules are needed for IMS exit messages when IMSMTO is set to ON. The OPSLOG disposition for the IMS exit message will not show SUP (for suppressed) if IMSMTO is only doing the suppression. This is because the IMSMTO setting does not require the AOF rules engine to do the suppression. AOF rules used for IMS exit messages will still be processed.

Default value

OFF

Other possible values

ON

Set or modify this parameter...

Anytime

Example: IMSMTO

This function turns on tracing for IMS user exits:

```
OPSPRM('SET','IMSMTO','ON')
```

IMSNONE Parameter

Sets the default value of the OPSLOG browse field IMSID for non-IMS messages, and also the value of the MSG.IMSID variable in a message rule when the current message is not an IMS message.

Default value

NONE

Other possible values

Any 1- to 4-character value

Set or modify this parameter...

Anytime

Example: IMSNONE

This function sets the default IMS ID for non-IMS messages to IMSX:

```
OPSPRM('SET','IMSNONE','IMSX')
```

IMSOUTPUT Parameter

Determines whether CA OPS/MVS includes the IMS WTOR message and any subsequent message lines when retrieving output from an IMS command.

Note: Although most IMS commands produce no additional output after redisplaying the IMS WTOR, a few commands, such as /DBR DB BI21PART, do generate such output.

Default value

YES

This value excludes the WTOR and any subsequent lines.

Other possible values

NO

This value includes the WTOR and additional output lines.

Set or modify this parameter...

Anytime

Example: IMSOUTPUT

This function tells CA OPS/MVS to include the WTOR and any additional output lines:

```
OPSPRM('SET','IMSOUTPUT','NO')
```

IMSROUTE Parameter

Defines the routing code that CA OPS/MVS uses when issuing WTOs for the IMS messages processed through its AOI exit.

Default value

X'0000'

Other possible values

Any valid routing code, expressed as hexadecimals

Set or modify this parameter...

Anytime

Example: IMSROUTE

This function sets the routing for IMS messages processed through the AOI exit to X'0001':

```
OPSPRM("SET","IMSROUTE","X'0001'")
```

IMSSVCx Parameter

Lets you predefine the IMS SVC numbers used in your system. Through IMSSVC*x*, you supply the number of the one of 16 possible IMS SVCs that CA OPS/MVS supports, with *x* representing the number of the IMS SVC found in the SVC table of the operating system. The *x* value does not correlate to the number of IMS control regions.

If you do not set the IMSSVC*x* parameter, CA OPS/MVS scans the SVC table of the system at startup and sets the parameter automatically.

Default value

No default

Other possible values

Any valid IMS SVC number

Set or modify this parameter...

Anytime

Example: IMSSVC*x*

This function sets the IMS SVC for the system to 6:

```
OPSPRM('SET','IMSSVC4','6')
```

IMSSVCxOFFSET Parameter

Defines the offset into the SVC table of the system at which the SVC indicated by *x* can be found, where *x* is any number between 1 and 16. CA OPS/MVS scans the SVC table at startup and sets this parameter automatically. To control which SVC is used, use the IMSSVC*x* parameter.

Default value

Null (the recommended value)

Other possible values

Any valid hexadecimal number

Set or modify this parameter...

Anytime

Example: IMSSVCxOFFSET

This function identifies the offset for the IMS SVC as X'0001':

```
OPSPRM("SET","IMSSVC3OFFSET","X'0001'")
```

IMSxINSTALLEXIT Parameter

Lets you optionally install the IOF AOI message and command exits. For more details, see the *Administration Guide*.

Default value

YES

Other possible values

NO

Set or modify this parameter...

Anytime

Example: IMSxINSTALLEXITS

This function installs the IOF exits:

```
OPSPRM(“SET”,”IMSxINSTALLEXITS”,”YES”)
```

INITIMS Parameter

Determines whether CA OPS/MVS dynamically installs its own IMS AOI and CMD exits.

If you are a customer who has licensed the IOF but have z/OS images where IMS is never used, there is a performance advantage (CPU and storage) in setting the INITIMS parameter to NO on those systems.

When the INITIMS parameter is set to NO, the output of the OPSPARM SHOW(ALL) command processor does not include IMS parameters. The NO setting also inhibits the displays of IMSPARMS when you use OPSVIEW option 4.1.1. This characteristic reduces storage and improves performance. If you set the value of INITIMS to YES, all displays are available.

Default value

NO

Other possible values

YES

Note: Only those customers who have licensed the IOF component should set this parameter value to YES.

Set or modify this parameter...

At initialization

Example: INITIMS

This function causes CA OPS/MVS to initialize its IMS exits:

```
OPSPRM('SET','INITIMS','YES')
```

WTOIMS Parameter

Determines whether IMS messages destined only for the MTO terminal will also be WTOd on the system console. See also IMSDESC Parameter and IMSROUTE Parameter in this chapter.

Default value

NO

Other possible values

YES

Set or modify this parameter...

Anytime

Example: WTOIMS

This function permits IMS messages that the AOI exit processes to receive WTO requests:

```
OPSPRM('SET','WTOIMS','YES')
```

MSF Parameters

The Multi-System Facility (MSF) is not part of the CA OPS/MVS base product-it is a separately licensed feature.

INITMSF Parameter

Initializes the MSF interface. If you have not licensed the MSF component, specify NO. If you are using ADDRESS MQ, INITMSF must be set to YES.

Default value

NO

Other possible values

YES

Sites that have licensed the MSF should set INITMSF to YES. If you are using ADDRESS MQ, you must set INITMSF to YES.

Set or modify this parameter...

At initialization

Example: INITMSF

This function prevents CA OPS/MVS from initializing the MSF:

```
OPSPRM('SET','INITMSF','NO')
```

MSFDELAY Parameter

Sets the default delay time for all remote MSF systems. CA OPS/MVS uses this parameter when the OPSCTL MSF DEFINE command, used to define a remote system, does not include the DELAY keyword. For information about the OPSCTL MSF DEFINE command, see the *Command and* Function *Reference*.

Default value

1 (second)

Other possible values

Any number of seconds

Set or modify this parameter...

Anytime

Example: MSFDELAY

This function sets 10 seconds as the delay time for remote MSF systems:

```
OPSPRM('SET','MSFDELAY','10')
```

MSFLOGMODE Parameter

Specifies the default VTAM LOGMODE name for all MSF APPC sessions.

Default value

LU62

Other possible values

Any valid VTAM LOGMODE name

Set or modify this parameter...

Anytime

Example: MSFLOGMODE

This function assigns the VTAM LOGMODE name LU4 to your MSF APPC sessions:

```
OPSPRM('SET','MSFLOGMODE','LU4')
```

MSFNONVTAMONLY Parameter

Determines whether MSF uses only non-VTAM based communication protocols.

To use MSF when VTAM is *not* active, set the value of MSFNONVTAMONLY to YES. If MSFNONVTAMONLY is set to NO, VTAM must be active to use MSF.

The MSFNONVTAMONLY parameter lets you either use non-VTAM MSF connections before VTAM has completely initialized or not use VTAM at all. In the first case, if you need such non-VTAM MSF connections, and also want to use VTAM MSF connections after VTAM has initialized, you can start CA OPS/MVS with the value of MSFNONVTAMONLY set to YES, and use a message rule to set the value of MSFNONVTAMONLY to NO and activate VTAM MSF connections after VTAM has initialized.

Default value

NO

This value indicates that VTAM must be active to use MSF.

Other possible values

YES

This value indicates that MSF will not attempt to use VTAM.

Set or modify this parameter...

Anytime

Example: MSFNONVTAMONLY

This function indicates that MSF will not attempt to use VTAM:

```
OPSPRM('SET','MSFNONVTAMONLY','YES')
```

MSFRESTARTREXX Parameter

Specifies the name of an OPS/REXX program that you have written to set up your MSF environment after the Multi-System Facility (MSF) has been restarted. The same OPS/REXX EXEC can and should be called from the REXX designated by the BEGINCMD parameter to set up MSF during product initialization.

The program you specify as the value of MSFRESTARTREXX must be in a PDS library that is allocated to the OPSMAIN STC JCL under the SYSEXEC ddname.

Important! For the MSFRESTRATREXX parameter to take effect, the value of the INITMSF parameter must be YES. For a description, see INITMSF Parameter in this chapter.

For details about tailoring CA OPS/MVS startup procedures, see the *Administration Guide*.

Default value

Blanks (indicating that there is no MSF restart program to execute)

Other possible values

Any valid OPS/REXX program name

Set or modify this parameter...

Anytime

Example: MSFRESTARTREXX

This function names MSFINIT as the MSF restart OPS/REXX program:

```
OPSPRM('SET','MSFRESTARTREXX','MSFINIT')
```

MSFSYSWAIT Parameter

Specifies a default wait time for CA OPS/MVS components that use the MSF.

For example, if you issue the ADDRESS AOF ENABLE command but fail to specify a value for the SYSWAIT keyword, CA OPS/MVS uses the wait time specified on the MSFSYSWAIT parameter. On the other hand, if your ADDRESS AOF ENABLE command includes a value for the SYSWAIT keyword, that value overrides the value of the MSFSYSWAIT parameter.

Default value

10 (seconds)

Other possible values

Any number of seconds from 1 to 300

Set or modify this parameter...

Anytime

Example: MSFSYSWAIT

This function sets the MSF wait time to 60 seconds:

```
OPSPRM('SET','MSFSYSWAIT','60')
```

SENDQUE Parameter

Sets the default size of the MSF send queue. The MSF send queue stores messages and commands sent across systems.

Default value

1024

Other possible values

You can send between 0 and 10,000 queued messages to an MSF node

Set or modify this parameter...

Anytime

Example: SENDQUE

This function sets the default size of the MSF send queue to 5,000 messages:

```
OPSPRM('SET','SENDQUE','5000')
```

Note: If you change the value of this parameter, you must restart MSF for the changes to take effect. For information on restarting MSF, see the *User Guide*.

SYSID Parameter

Defines the name of the local system in a Multi-System Facility, or MSF, network.

If possible, we recommend that you specify either the SMF ID of your system or the SYSNAME from IEASYS*xx* as the value of the SYSID parameter.

Typically, you should change the value of the SYSID parameter only prior to defining and starting your MSF network.

Default value

The SMF ID of your system. If the SMF ID does not begin with an alphabetic or national character ($, #, or @), then the SMF ID will be prefixed by the character string @@@@. For example, if the SMF ID of your system is 1234, the Default value of the SYSID parameter is @@@@1234.

Other possible values

Any valid one- to eight-character MSF ID. The first character must be alphabetic or national. The remaining characters may be alphabetic, national, or numeric. Imbedded blanks are not permitted.

Set or modify this parameter...

Anytime

Example: SYSID

This function sets the system ID toZOS9:

```
OPSPRM('SET','SYSID','ZOS9')
```

VTAMQUE Parameter

Determines the maximum number of cross-system messages and commands that can be queued for sending across MSF communication links.

Default value

1024

This value should be sufficient in most cases.

Other possible values

Any number from 0 to 10,000

Set or modify this parameter...

At initialization

Example: VTAMQUE

This function indicates that as many as 5000 messages and commands can be queued:

```
OPSPRM('SET','VTAMQUE','5000')
```

MLWTO Parameters

The parameters described in the following sections control multiple line messages (MLWTOs).

MLWTOMAXLINES Parameter

The MLWTOMAXLINES parameter determines the maximum lines that can be used for input. This parameter can be changed in the OPSSPAxx member.

Default value

1024

Other possible values

0-10000

Set or modify this parameter...

At initialization

Example: MLWTOMAXLINES

This function changes the default to 100 lines.

```
OPSPRM('SET','MLWTOMAXLINES','100')
```

MLWTOFAILCOUNT Parameter

The MLWTOFAILCOUNT parameter for output only, indicates how many times:

- The 64-bit buffer that contains the MLWTO lines was full.
- An MLWTO message with an associated rule was received but the rule did not fire due to the 64-bit buffer being full.

Note: Regardless of the number of buffers, a rogue product can overrun the buffers issuing an erroneous MLWTO message.

A rogue product is a product that uses up all available storage when issuing a multiline MLWTO message.

Default value

0

Other possible values

0-9999999

Example: Erroneous MLWTO messages

Because of the way that MLWTO messages are issued, it is possible for the first line of an MLWTO to be seen by CA OPS/MVS, where subsequent lines are not. A product can also issue many lines of a multiline wto but never issue an end line.

MLWTOFAILDATE Parameter

The MLWTOFAILDATE parameter indicates the date when the last failure occurred. The MLWTOFAILCOUNT parameters must be greater than zero for a date to be written in the MLWTOFAILDATE parameter.

Default value

None

Other possible values

Any valid date of the format yyyy/mm/dd.

Example: MLWTOFAILDATE

2011/12/19

MLWTORULEFIRECOUNT Parameter

The MLWTORULEFIRECOUNT parameter indicates the number of rules that fired on MLWTOs from startup.

Default value

0

Other possible values

Any whole number value.

Example: MLWTORULEFIRECOUNT

555

MLWTOFAILTIME Parameter

MLWTOFAILTIME the time where the last failure occurred indicated by MLWTOFAILCOUNT>0.

Default value

None

Other possible values

Any valid timestamp.

Example: MLWTOFAILTIME

15:38:19

Chapter 7: Display-only Parameters

This section contains the following topics:

Overview of the Display-only Parameters

CA OPS/MVS provides numerous display-only parameters. You cannot change the values of display-only parameters, but they can provide you with useful and important information.

Some of the most important display-only parameters are described in this chapter. For information about display-only parameters not described here, see the description of OPSVIEW option 4.1.1 in the *OPSVIEW User Guide*.

Some of the CA OPS/MVS most important display-only parameters are listed alphabetically in the following sections.

ABENDCURRENT Parameter

The ABENDCURRENT parameter keeps a count of the number of times CA OPS/MVS has abended.

AOFEXECUTESTATUS Parameter

The AOFEXECUTESTATUS parameter displays the current state of the AOF Execute Processor. The AOF Execute Processor may be in one of the following states:

State	Description
WAITING	Waiting for work (the most common state).
NORMAL	Normal AOF Execute processing.
INIT	AOF is initializing.

State	Description
TOD	AOF is performing Time-Of-Day event processing (a TOD rule is probably executing). If this parameter remains in the TOD status for a long period of time, check the AOFEXECUTETOD parameter for the name of the TOD rule that is monopolizing AOF processing. Do not code TOD rules that wait for long periods of time, since they prevent other AOF activity from occurring. Such processing should be done asynchronously in OSF TSO servers.
AOFCMD	AOF is processing an ADDRESS AOF host command.
TERM	AOF is terminating.

AOFEXECUTETOD Parameter

The AOFEXECUTETOD parameter displays the ruleset.rulename combination of either the currently executing or last executed TOD rule. When the AOFEXECUTESTATUS parameter shows TOD, this parameter almost always shows the name of the executing TOD rule.

AOFRECURSIONDATE Parameter

The AOFRECURSIONDATE parameter displays the date the AOF recursion was detected. The date is in the yyyy/mm/dd format.

AOFRECURSIONTIME Parameter

The AOFRECURSIONTIME parameter displays the time the AOF recursion was detected.

AOFRECURSIONRULE Parameter

The AOFRECURSIONRULE parameter displays the name of the rule (in ruleset.rulename format) that was running when the AOF recursion was detected.

AOFRULESTORAGE Parameter

The AOFRULESTORAGE parameter displays the number of bytes of extended private virtual storage currently being used by AOF rules.

AOFWSHIGHUSEDDATE Parameter

The AOFWSHIGHUSEDDATE parameter displays the date the AOF REXX workspace high watermark was reached. The date is in the *yyyy*/*mm*/*dd* format.

AOFWSHIGHUSEDPGM Parameter

The AOFWSHIGHUSEDPGM parameter displays the name of the external subroutine (or rule) that caused the AOF REXX workspace high watermark to be reached.

AOFWSHIGHUSEDRULE Parameter

The AOFWSHIGHUSEDRULE parameter displays the name of the rule (in *ruleset.rulename* format) that caused the AOF REXX workspace high watermark to be reached.

AOFWSHIGHUSEDTIME Parameter

The AOFWSHIGHUSEDTIME parameter displays the time the AOF REXX workspace high watermark was reached.

BROWSEARCHNEEDED Parameter

The BROWSEARCHNEEDED parameter contains the OPSLOG message sequence number of the message that caused the OPS4403O message to be issued in order to start an OPSLOG archive process. The frequency of the OPSLOG archive cycle is controlled by the ARCHIVETRIGGER parameter.

BROWSEBLOCKS Parameter

The BROWSEBLOCKS parameter displays the total number of z/OS pages included in the OPSLOG data area that were written to the VSAM checkpoint data set as a result of the checkpointing process. The checkpoint process writes out all changed pages periodically and all pages at termination.

BROWSEBYTES Parameter

The BROWSEBYTES parameter displays the number of bytes of above-the-line private area storage required to contain the OPSLOG data and its associated overhead areas. The size of this data area is primarily determined by the value of the BROWSEMAX parameter.

BROWSECHECK Parameter

The BROWSECHECK parameter displays the total number of OPSLOG checkpoint requests that successfully wrote data to the OPSLOG checkpoint VSAM data set. If no new event data is received during a checkpoint interval, the checkpoint is bypassed.

BROWSECHECKNUM Parameter

The BROWSECHECKNUM parameter displays the highest OPSLOG message sequence number that was included in the last OPSLOG checkpoint. When this number is less than the current message sequence number, new data has been added to OPSLOG and it must be checkpointed.

BROWSEDATE Parameter

The BROWSEDATE parameter displays the local date (in *yyyy/mm/dd* format) of the last OPSLOG checkpoint in which changed and new data in OPSLOG was written to the OPSLOG checkpoint VSAM data set.

BROWSELASTARCH Parameter

The BROWSELASTARCH parameter displays the highest OPSLOG message sequence number that has been successfully archived. The OPSLOG archive program, OPARLGCR, sets this value after successfully completing an archive request.

BROWSEMAXINUSE Parameter

Displays the maximum number of messages in the OPSLOG message area for the current in-use OPSLOG. The OPSLOG message area resides in a data space owned by the CA OPS/MVS main address space.

The value of the BROWSEMAXINUSE parameter is determined by one of the following:

- The BROWSEMAX parameter
- The default or override value specified on either of the following host commands for the current OPSLOG:

  ADDRESS OPSCTL “OPSLOG DEFINE”

  ADDRESS OPSCTL “OPSLOG ACTIVATE”

BROWSEMSGS Parameter

The BROWSEMSGS parameter displays the total number of error messages that would have been produced from errors in OPSLOG processing. When this number exceeds an internal threshold value, error messages are suppressed to avoid flooding the system consoles and OPSLOG with excessive error messages.

BROWSEPAGES Parameter

The BROWSEPAGES parameter displays the number of z/OS 4 KB page frames of above-the-line private area storage required to contain the OPSLOG data and its associated overhead areas. The size of this data area is primarily determined by the BROWSEMAX parameter. This number may be used to estimate the required size of the OPSLOG VSAM linear data set, which is composed of 4 KB control intervals that are mapped to z/OS pages.

BROWSESEQ Parameter

The BROWSESEQ parameter displays the OPSLOG message sequence number of the most recent message sent to OPSLOG. This number is increased incrementally by 1 for each new message.

BROWSETIME Parameter

The BROWSETIME parameter displays the local time (in *hh:mm:ss* format) of the last OPSLOG checkpoint in which changed and new data in OPSLOG was written to the OPSLOG checkpoint VSAM data set. The checkpoint frequency is determined by the value of the BROWSEINTERVAL parameter.

CAIENFCOUNT Parameter

The CAIENFCOUNT parameter keeps a count of the CAIENF events that CA OPS/MVS receives.

CAIENFCURRENT Parameter

The CAIENFCURRENT parameter keeps a count of the number of CAIENF events that CA OPS/MVS receives each second. For every elapsed second, CA OPS/MVS decreases this counter by the value of the CAIENFRATE parameter.

CAIENFDATE Parameter

The CAIENFDATE parameter displays the date (in *yyyy*/*mm*/*dd* format) of the last CAIENF event that CA OPS/MVS received.

CAIENFTIME Parameter

The CAIENFTIME parameter displays the time of the last CAIENF event CA OPS/MVS received.

CMDACCEPT Parameter

The CMDACCEPT parameter keeps a count of the number of times the final command rule disposition was ACCEPT after all AOF rules processed the command event.

CMDECF Parameter

The CMDECF parameter keeps a count of the number of commands that were directed to the ECF because they were prefixed by the value of the ECFCHAR parameter.

CMDNOACTION Parameter

The CMDNOACTION parameter keeps a count of the number of times the final command rule disposition was NOACTION after all AOF rules processed the command event.

CMDOSF Parameter

The CMDOSF parameter keeps a count of the number of commands that were directed to the OSF because they were prefixed by the value of the OSFCHAR parameter.

CMDREJECT Parameter

The CMDREJECT parameter keeps a count of the number of times the final command rule disposition was REJECT after all AOF rules processed the command event.

COMMANDCOUNT Parameter

The COMMANDCOUNT parameter keeps a count of commands that CA OPS/MVS has generated.

COMMANDCURRENT Parameter

The COMMANDCURRENT parameter keeps a count of how many commands CA OPS/MVS issues per second. For every elapsed second, CA OPS/MVS decreases this counter by the value of the COMMANDRATE parameter. CA OPS/MVS calculates the COMMANDCURRENT value only when a command is issued in CA OPS/MVS.

COMMANDDATE Parameter

The COMMANDDATE parameter displays the date (in *yyyy*/*mm*/*dd* format) of the last command issued by CA OPS/MVS.

COMMANDSUPPRESSED Parameter

The COMMANDSUPPRESSED parameter keeps a count of the commands that were destined to be issued by CA OPS/MVS, but were suppressed. CA OPS/MVS suppresses the generation of a command if it will cause the COMMANDCURRENT value to exceed the COMMANDMAX value.

COMMANDTIME Parameter

The COMMANDTIME parameter displays the time of the last command issued by CA OPS/MVS.

CSA Parameter

The CSA parameter displays the amount of common service area (CSA) storage below the 16 MB line that CA OPS/MVS has directly acquired. Compare this value to the limit defined by the CSALIMIT parameter. Note that CSA storage may be indirectly acquired by operating system services used by CA OPS/MVS. Any storage acquired in this way is not included in this value or in the ECSALIMIT value.

ECSA Parameter

The ECSA parameter displays the amount of extended common service area (ECSA) storage above the 16 MB line that CA OPS/MVS has directly acquired. Compare this value to the limit defined by the ECSALIMIT parameter. Note that ECSA storage may be indirectly acquired by operating system services used by CA OPS/MVS. For example, the IBM Communication Server (VTAM) acquires ECSA storage when MSF APPC connections are used. Any storage acquired in this way is not included in this value or in the ECSALIMIT value.

ENFEXECUTESTATUS Parameter

The ENFEXECUTESTATUS parameter displays the current state of the ENF Execute Processor. The ENF Execute Processor may be in one of the following states:

State	Description
NEVACT	The ENF interface has not been activated since the product started.
INIT	The ENF interface is initializing.
LISTEN	The ENF interface is active and is listening for ENF events (the most common state).
TERM	The ENF interface is terminating or has already terminated. It can be restarted via the MODIFY command.

ENQ Parameter

The ENQ parameter specifies the ENQ major name (QNAME) used by CA OPS/MVS. This ENQ major name guarantees that only one OPS/MVS subsystem with a unique OPS/MVS subsystem name (for example: OPSS) is active at any time on each copy of the operating system. If you start a second copy, it is suspended on the ENQ until the first copy terminates and releases its ENQ.

GLOBALALLOC Parameter

The GLOBALALLOC parameter displays the high-used watermark in the global variable workspace. Space is always allocated from the workspace using a first fit algorithm to keep the working set to a minimum. The value of the GLOBALALLOC parameter is the sum of the values of the GLOBALBLOCKSUSED parameter and the GLOBALFREE parameter.

GLOBALBACKUPCOUNT Parameter

The GLOBALBACKUPCOUNT parameter displays the number of global variable and Relational Data Framework data backups that have been completed since starting CA OPS/MVS.

GLOBALBACKUPEND Parameter

The GLOBALBACKUPEND parameter displays the time at which the last global variable and Relational Data Framework data backup was completed.

GLOBALBACKUPNEXT Parameter

The GLOBALBACKUPNEXT parameter displays the time at which the next global variable and Relational Data Framework data backup is scheduled to start.

GLOBALBACKUPSTART Parameter

The GLOBALBACKUPSTART parameter displays the time at which the last global variable and Relational Data Framework data backup began.

GLOBALBLOCKSUSED Parameter

The GLOBALBLOCKSUSED parameter displays the number of global variable blocks in use. The global variable database is divided into 256-byte segments. Each global variable and relational table row requires at least one block and may require a number of additional contiguous blocks to hold all of the data. Most of the first block is not used to contain data, but instead contains information associated with the variable (for example, the name of the variable). The value of the GLOBALBLOCKSUSED parameter is greater than or equal to the value of the GLOBALUSED parameter.

GLOBALCHECK Parameter

The GLOBALCHECK parameter displays the number of successful SYSCHK1 checkpoint requests. During a successful checkpoint operation, CA OPS/MVS writes all changed global variable and Relational Data Framework data to the SYSCHK1 linear VSAM (DIV) data set. If no changes are made to the database during a checkpoint interval, which is defined by the GLOBALINTERVAL parameter, the checkpoint operation is bypassed.

GLOBALCHECKUPDATES Parameter

The GLOBALCHECKUPDATES parameter displays the number of global variable and Relational Data Framework updates that occurred from the time CA OPS/MVS was last started to the time the last SYSCHK1 checkpoint operation was completed.

GLOBALDATE Parameter

The GLOBALDATE parameter displays the local date, in *yyyy*/*mm*/*dd* format, of the last checkpoint operation in which all changed pages were written to the SYSCHK1 linear VSAM (DIV) data set. The value of the GLOBALINTERVAL parameter determines the checkpoint frequency. See also GLOBALTIME Parameter in this chapter.

GLOBALFREE Parameter

The GLOBALFREE parameter displays the total number of free blocks in all of the free areas. For related information, see GLOBALFREEAREAS Parameter in this chapter. This number does not include the never allocated blocks above the high-used watermark. For related information, see GLOBALALLOC Parameter in this chapter. The value of the GLOBALFREE parameter is greater than or equal to the value of the GLOBALFREEAREAS parameter.

GLOBALFREEAREAS Parameter

The GLOBALFREEAREAS parameter displays the total number of free areas in the allocated space. For related information, see GLOBALALLOC Parameter in this chapter. Contiguous free blocks are glued together into free areas, so each free area may contain one or more free blocks.

GLOBALMSGS Parameter

The GLOBALMSGS parameter displays the total number of error messages that would have been produced from global variable and Relational Data Framework checkpoint processing. When this number exceeds an internal threshold value, error messages are suppressed to avoid flooding the system consoles and OPSLOG with excessive error messages. Under normal conditions, the value of the GLOBALMSGS parameter is 0.

GLOBALPAGES Parameter

The GLOBALPAGES parameter displays the size, in pages, of the above-the-line private storage area that is required to contain the global variable and Relational Data Framework data, as well as the associated overhead areas. The value of the GLOBALMAX parameter determines the size of this data area.

GLOBALRESTORETIME Parameter

The GLOBALRESTORETIME parameter displays the time at which the last global variable and Relational Data Framework data restore operation were completed.

GLOBALRETRY Parameter

The GLOBALRETRY parameter displays the number of times that a SYSCHK1 checkpoint operation had to be retried. If the update counter changed while a checkpoint was in progress, the checkpoint operation is retried in order to guarantee the integrity of the DASD copy of the database. For more information about the update counter, see GLOBALDATE Parameter in this chapter.

GLOBALSIZE Parameter

The GLOBALSIZE parameter displays the size of the above-the-line private storage area that is required to contain the global variable and Relational Data Framework data, as well as the associated overhead areas. The value of the GLOBALMAX parameter determines the size of this data area.

GLOBALTEMPALLOC Parameter

The GLOBALTEMPALLOC parameter displays the high-used watermark in the temporary global variable workspace. Space is always allocated from the workspace using a first fit algorithm to keep the working set to a minimum. The value of the GLOBALTEMPALLOC parameter is equal to the sum of the values of the GLOBATEMPBLKSUSED parameter and the GLOBALTEMPFREE parameter.

GLOBALTEMPBLKSUSED Parameter

The GLOBALTEMPBLKSUSED parameter displays the number of temporary global variable blocks in use. The temporary global variable database is divided into 256-byte segments. Each temporary global variable and each row of a relational table that was defined as a GLOBAL TEMPORARY table requires at least one block and may require a number of additional contiguous blocks to hold all of the data. Most of the first block is not used to contain data, but instead contains information associated with the variable (for example, the name of the variable). The value of the GLOBALTEMPBLKSUSED parameter is greater than or equal to the value of the GLOBALTEMPUSED parameter.

GLOBALTEMPFREE Parameter

The GLOBALTEMPFREE parameter displays the total number of free blocks in all of the free areas. For related information, see GLOBALTEMPFREEAREAS Parameter in this chapter. This number does not include the never allocated blocks above the high-used watermark. For related information, see GLOBALTEMPALLOC Parameter in this chapter. The value of the GLOBALTEMPFREE parameter is greater than or equal to the value of the GLOBALTEMPFREEAREAS parameter.

GLOBALTEMPFREEAREAS Parameter

The GLOBALTEMPFREEAREAS parameter displays the total number of free areas in the allocated space. For more information, see GLOBALTEMPALLOC Parameter in this chapter. Contiguous free blocks are glued together into free areas, so each free area may contain one or more free blocks.

GLOBALTEMPUPDATE Parameter

The GLOBALTEMPUPDATE parameter displays the number of updates that were made, since CA OPS/MVS was started, to temporary global variables and to relational tables that were defined with the GLOBAL TEMPORARY operand.

GLOBALTEMPUSED Parameter

The GLOBALTEMPUSED parameter displays the number of temporary global variables in use. This number includes all of the data related to relational tables that were defined with the GLOBAL TEMPORARY operand. Each row in a relational table that has been designated as GLOBAL TEMPORARY counts as a single temporary global variable. For more information, see GLOBALTEMPMAX Parameter in the chapter “Parameters for Facilities.”

GLOBALTIME Parameter

The GLOBALTIME parameter displays the local time, in *hh*:*mm*:*ss* format, of the last checkpoint operation in which all changed pages were written to the SYSCHK1 linear VSAM (DIV) data set. The value of the GLOBALINTERVAL parameter determines the checkpoint frequency. See also GLOBALDATE Parameter in this chapter.

GLOBALUPDATE Parameter

The GLOBALUPDATE parameter displays the number of global variable and Relational Data Framework updates that have occurred since CA OPS/MVS was started. The value of the GLOBALUPDATE parameter is greater than or equal to the value of the GLOBALCHECKUPDATES parameter.

GLOBALUSED Parameter

The GLOBALUSED parameter displays the number of global variables in use. This number includes all of the data related to relational tables. Each row in a relational table counts as a single global variable. For more information, see GLOBALMAX Parameter in the chapter “Parameters for Facilities.”

JESPVER Parameter

The JESPVER parameter, which is automatically set by CA OPS/MVS at startup time, is used to determine the levels of certain operating system facilities. The value of the JESPVER parameter reflects the JESSVERS field of the JESCT extension control block and should contain a value between 1 and 255. The value affects several CA OPS/MVS functions, including its WTO message processing and whether it can access DIV. The JESPVER parameter is not intended for customer use.

JOBID Parameter

The JOBID parameter displays the JES job ID of the product if it was started under JES (for example, STC01234), or it displays the product job name if it was started under the master subsystem.

LOGRECCOUNT Parameter

The LOGRECCOUNT parameter keeps a count of LOGREC records that CA OPS/MVS has generated.

LOGRECCURRENT Parameter

The LOGRECCURRENT parameter counts the number of LOGREC records CA OPS/MVS writes. CA OPS/MVS creates these records when recoverable abends occur during CA OPS/MVS processing, placing them in the SYS1.LOCREC data set. A CA Customer Support representative may ask you to print these records for diagnostic purposes. If you have the CA SYSVIEW product, you can browse and print these records online.

LOGRECDATE Parameter

The LOGRECDATE parameter displays the date (in *yyyy*/*mm*/*dd* format) of the last LOGREC record created by CA OPS/MVS.

LOGRECSUPPRESSED Parameter

The LOGRECSUPPRESSED parameter displays the count of LOGREC records that were going to be generated by CA OPS/MVS, but were suppressed. CA OPS/MVS suppresses the generation of a LOGREC record if the record will cause the LOGRECCURRENT value to exceed the LOGRECMAX value.

LOGRECTIME Parameter

The LOGRECTIME parameter displays the time of the last LOGREC record created by CA OPS/MVS.

LXCONCOUNT Parameter

The LXCONCOUNT parameter keeps a count of the Linux Connector events that CA OPS/MVS receives.

LXCONCURRENT Parameter

The LXCONCURRENT parameter keeps a count of the number of Linux Connector events that CA OPS/MVS receives each second. For every elapsed second, CA OPS/MVS decreases this counter by the value of the LXCONRATE parameter.

LXCONDATE Parameter

The LXCONDATE parameter displays the date (in yyyy/mm/dd format) of the last Linux Connector event that CA OPS/MVS received.

LXCONTIME Parameter

The LXCONTIME parameter displays the time of the last Linux Connector event CA OPS/MVS received.

MEMBER Parameter

The MEMBER parameter displays the value of the MEMBER substitutional JCL parameter of the OPSMAIN started task procedure.

MESSAGECOUNT Parameter

The MESSAGECOUNT parameter keeps a count of the messages that CA OPS/MVS generates.

MESSAGECURRENT Parameter

The MESSAGECURRENT parameter keeps a count of the number of WTO messages that CA OPS/MVS issues each second. For every elapsed second, CA OPS/MVS decreases this counter by the value of the MESSAGERATE parameter.

MESSAGEDATE Parameter

The MESSAGEDATE parameter displays the date (in *yyyy*/*mm*/*dd* format) of the last message that CA OPS/MVS issued.

MESSAGESUPPRESSED Parameter

The MESSAGESUPPRESSED parameter keeps a count of the messages that were destined to be issued by CA OPS/MVS, but were suppressed. CA OPS/MVS suppresses the generation of one of its messages if generating the message would cause the value of the MESSAGECURRENT parameter to exceed the value of the MESSAGEMAX parameter.

MESSAGETIME Parameter

The MESSAGETIME parameter displays the time of the last message CA OPS/MVS issued.

MSGDELETE Parameter

The MSGDELETE parameter keeps a count of how many messages were targeted for deletion through the RETURN "DELETE" statement. This parameter is useful for gathering CA OPS/MVS performance statistics.

MSGDISPLAY Parameter

The MSGDISPLAY parameter keeps a count of how many messages were targeted, through the RETURN "DISPLAY" statement, for display only (that is, not for inclusion in the SYSLOG). This parameter is useful for gathering CA OPS/MVS performance statistics.

MSGMPFBYPASS Parameter

The MSGMPFBYPASS parameter keeps a count of the number of messages that were not processed by the AOF because the AOFMPFBYPASS parameter was set to YES and these messages had been suppressed by MPF.

MSGNOOPSLOG Parameter

The MSGNOOPSLOG parameter keeps a count of how many messages were suppressed from OPSLOG through message rules using the NOOPSLOG option. This parameter is useful for gathering CA OPS/MVS performance statistics. See the description of the NOOPSLOG option in the *AOF Rules* User *Guide*.

MSGNORMAL Parameter

The MSGNORMAL parameter keeps a count of how many messages were targeted for normal processing through the MSG.DISP variable. This parameter is useful for gathering CA OPS/MVS performance statistics. For a description of the MSG.DISP variable, see the *AOF Rules User* Guide.

MSGSUPPRESS Parameter

The MSGSUPPRESS parameter keeps a count of how many messages were targeted for suppression through the MSG.DISP variable. This parameter is useful for gathering CA OPS/MVS performance statistics. For a description of the MSG.DISP variable, see the *AOF Rules User* Guide.

OCCONSALLOC Parameter

The OCCONSALLOC parameter keeps a count of the number of times that a product console was successfully allocated as a result of the use of either the OPSCMD command processor or the ADDRESS OPER host command environment. CA OPS/MVS will not increase the counter incrementally if, in the text of the OPSCMD command processor or ADDRESS OPER host command, you specify either the NOOUTPUT keyword or the name or ID of a console that is not owned by CA OPS/MVS.

OCCONSFAIL Parameter

The OCCONSFAIL parameter keeps a count of the number of times a request to allocate a product console failed immediately without waiting. This condition typically occurs when the CONTYPE keyword on an OPSCMD command processor or ADDRESS OPER host command specifies a type of console that is not allocated to the product.

OCCONSWAITALLOC Parameter

The OCCONSWAITALLOC parameter keeps a count of the number of times that a product console was successfully allocated only after waiting for one of the product consoles to be released. This count is included in the count kept by the OCCONSALLOC parameter.

OCCONSWAITFAIL Parameter

The OCCONSWAITFAIL parameter keeps a count of the number of times a timeout occurred while CA OPS/MVS was waiting for a product console to become available for allocation. A timeout occurs when all of the consoles of the type indicated by the CONTYPE keyword (specified on an OPSCMD command processor or ADDRESS OPER host command) were busy at each interval defined by the OCCONINT parameter for the duration defined by the OCCONTIME parameter. For more information, see the descriptions of the OCCONINT parameter and the OCCONTIME parameter.

SECURITYPACKAGE Parameter

The SECURITYPACKAGE parameter displays the name of the security package that CA OPS/MVS has determined is being used on this system. Possible values are limited to RACF, CA ACF2, TSS, and NONE.

SECURITYSTRING Parameter

The SECURITYSTRING parameter displays the version and release level of RACF or CA ACF2 that is being used on this system (for example, 2.60 for RACF 2.60).

SQLALTERTABLE Parameter

The SQLALTERTABLE parameter keeps a count of the number of times CA OPS/MVS has issued an SQL ALTER TABLE statement since product initialization. When CA OPS/MVS is restarted, the value of SQLALTERTABLE is reset to 0.

SQLCLOSE Parameter

The SQLCLOSE parameter keeps a count of the number of times CA OPS/MVS has issued an SQL CLOSE statement to close a relational table since product initialization. When CA OPS/MVS is restarted, the value of SQLCLOSE is reset to 0.

SQLCOMPILATIONS Parameter

The SQLCOMPILATIONS parameter keeps a count of the number of SQL statements CA OPS/MVS has compiled since product initialization. When CA OPS/MVS is restarted, the value of SQLCOMPILATIONS is reset to 0.

SQLCOMPILEERRORS Parameter

The SQLCOMPILEERRORS parameter keeps a count of the number of SQL statement compiling errors that have occurred since product initialization. When CA OPS/MVS is restarted, the value of SQLCOMPILEERRORS is reset to 0.

SQLCREATETABLE Parameter

The SQLCREATETABLE parameter keeps a count of the SQL CREATE TABLE statements that CA OPS/MVS has processed since product initialization. When CA OPS/MVS is restarted, the value of SQLCREATETABLE is reset to 0.

SQLDECLARECURSOR Parameter

The SQLDECLARECURSOR parameter keeps a count of the SQL DECLARE CURSOR statements that CA OPS/MVS has processed since product initialization. When CA OPS/MVS is restarted, the value of SQLDECLARECURSOR is reset to 0.

SQLDELETE Parameter

The SQLDELETE parameter keeps a count of the SQL DELETE statements that CA OPS/MVS has processed since product initialization. When CA OPS/MVS is restarted, the value of SQLDELETE is reset to 0.

SQLDELETIONS Parameter

The SQLDELETIONS parameter keeps a count of the SQL record statements that CA OPS/MVS has deleted since product initialization. When CA OPS/MVS is restarted, the value of SQLDELETIONS is reset to 0.

SQLDIRECTREADS Parameter

The SQLDIRECTREADS parameter keeps a count of the number of times that CA OPS/MVS read directly from its SQL database since initialization. When CA OPS/MVS is restarted, SQLDIRECTREADS is reset to 0.

SQLDROPTABLE Parameter

The SQLDROPTABLE parameter keeps a count of the number of SQL DROP TABLE instructions that CA OPS/MVS has processed since initialization. When CA OPS/MVS is restarted, SQLDROPTABLE is reset to 0.

SQLEXECUTIONERRORS Parameter

The SQLEXECUTIONERRORS parameter keeps a count of the number of errors that CA OPS/MVS made in executing SQL statements since initialization. When CA OPS/MVS is restarted, the value of SQLEXECUTIONERRORS is reset to 0.

SQLEXECUTIONS Parameter

The SQLEXECUTIONS parameter keeps a count of the number of SQL statements that CA OPS/MVS has executed since product initialization. When CA OPS/MVS is restarted, SQLEXECUTIONS is reset to 0.

SQLFETCH Parameter

The SQLFETCH parameter keeps a count of the number of SQL FETCH statements that CA OPS/MVS has processed since initialization. When CA OPS/MVS is restarted, the value of SQLFETCH is reset to 0.

SQLINSERT Parameter

The SQLINSERT parameter keeps a count of the number of SQL INSERT statements that CA OPS/MVS has processed since product initialization. When CA OPS/MVS is restarted, SQLINSERT is reset to 0.

SQLINSERTIONS Parameter

The SQLINSERTIONS parameter keeps a count of the number of records inserted into the CA OPS/MVS SQL database since CA OPS/MVS initialization. When CA OPS/MVS is restarted, the value of SQLINSERTIONS is reset to 0.

SQLLOGICERRORS Parameter

The SQLLOGICERRORS parameter keeps a count of the internal SQL errors that CA OPS/MVS has incurred since product initialization. When CA OPS/MVS is restarted, SQLLOGICERRORS is reset to 0.

SQLOPEN Parameter

The SQLOPEN parameter keeps a count of the SQL OPEN instructions that CA OPS/MVS has processed since product initialization. When CA OPS/MVS is restarted, SQLOPEN is reset to 0.

SQLSELECT Parameter

The SQLSELECT parameter keeps a count of the SQL SELECT instructions that CA OPS/MVS has processed since initialization. When CA OPS/MVS is restarted, the value of SQLSELECT is reset to 0.

SQLUPDATE Parameter

The SQLUPDATE parameter keeps a count of the SQL UPDATE instructions that CA OPS/MVS has processed since initialization. When CA OPS/MVS is restarted, SQLUPDATE is reset to 0.

SQLUPDATES Parameter

The SQLUPDATES parameter keeps a count of the number of records updated in the CA OPS/MVS SQL database since product initialization. When CA OPS/MVS is restarted, SQLUPDATES is reset to 0.

SSEXEXITHIDATE Parameter

The SSEXEXITHIDATE parameter displays the local date (in *yyyy/mm/dd* format) on which the SSEXEXITHICOUNT value was first reached.

SSEXEXITHITIME Parameter

The SSEXEXITHITIME parameter displays the local time (in *hh:mm:ss* format) at which the SSEXEXITHICOUNT was first reached.

SSEXEXITFAILURES Parameter

The SSEXEXITFAILURES parameter displays the number of failed attempts to allocate a process block from the process block pool because there were no process blocks available. For more information, see the description of the PROCESS parameter in the chapter “Parameters.”

SSEXEXITFAILDATE Parameter

The SSEXEXITFAILDATE parameter displays the local date (in *yyyy/mm/dd* format) on which the last process block allocation failure occurred.

SSEXEXITFAILTIME Parameter

The SSEXEXITFAILTIME parameter displays the local time (in *hh:mm:ss* format) at which the last process block allocation failure occurred.

SSPCCOUNT Parameter

The SSPCCOUNT parameter displays the numbers of times the space switch PC was unable to process a request due to some kind of error.

SSPCTEXT Parameter

The SSPCTEXT parameter displays a text string indicating the cause of the last space switch PC error. The most common error is the inability to allocate a process block because there are no free process blocks. In this case, the text string contains the string 'PRCS BLOCK POOL EMPTY'.

SSPCDATE Parameter

The SSPCDATE parameter displays the local date (in *yyyy/mm/dd* format) on which the last switch PC error occurred.

SSPCTIME Parameter

The SSPCTIME parameter displays the time local time (in *hh:mm:ss* format) at which the last space switch PC error occurred.

STARTDATE Parameter

The STARTDATE parameter displays the local date (in *yyyy/mm/dd* format) on which CA OPS/MVS was started.

STARTTIME Parameter

The STARTTIME parameter displays the local time (in *hh:mm:ss* format) at which CA OPS/MVS was started.

SUBSYS Parameter

The SUBSYS parameter displays the z/OS subsystem name under whose control CA OPS/MVS was started. The value will be the master subsystem (MSTR) if the product was started with SUB=MSTR on the z/OS START command, or the name of the JES subsystem (usually JES2 or JES3).

SUBSYSCONSOLES Parameter

The SUBSYSCONSOLES parameter displays the number of z/OS non-extended subsystem consoles that have been successfully acquired by CA OPS/MVS. The related parameter, SUBSYSDEFAULT, determines the number of consoles CA OPS/MVS tries to acquire at CA OPS/MVS initialization.

UX18EXIT Parameter

The UX18EXIT parameter keeps a count of the number of times the CA OPS/MVS IATUX18 exit, which is the JES3 command exit, passed a command to CA OPS/MVS.

Index

A

B

C

D

E

G

I

J

L

M

N

O

P

R

S

T

U

V

W