precisely

Ironstream for Microfocus Universal Discovery for IBM Z®

Administration Guide

Version 7.3.0

Ironstream for Micro Focus Universal Discovery for IBM Z: Administration Guide

Version 7.3.0

Last Update: 16 September 2021

Contents

THIS PAGE IS INTENTIONALLY LEFT BLANK

Preface

Audience

This document is intended for licensed Ironstream for Micro Focus Universal Discovery for IBM Z administrators and users. It describes configuring and operating the product and assumes it been installed as described in the *Installation Guide*.

Please note: *Ironstream for Micro Focus Universal Discovery for IBM Z®* was formerly *EView/390z Mainframe Discovery for Micro Focus Universal Discovery (UD) for UCMDB*. Precisely is in the process of re-branding all EView products to Ironstream.

Precisely Support

To contact Precisely Support, please visit https://support.precisely.com/.

Related Resources

Ironstream for Micro Focus Universal Discovery for IBM Z Administration Guide provides manuals to help you use the product and understand the underlying concepts. All product documentation is available at https://support.precisely.com/.

- *Installation Guide*

  Explains how to install, de-install, and configure Ironstream for Micro Focus Universal Discovery for IBM Z. Also includes how to upload installation files from the Discovery Probe server and start and stop Ironstream processes.

In addition to Ironstream documentation, related Micro Focus UCMDB products provide a comprehensive set of manuals that help you use the products and improve your understanding of the underlying UCMDB concepts.

Revision History

This manual's title page contains the following identifying information:

- *Version number, which indicates the software version.*
- *Print date, which changes each time the document is updated.*

This table indicates changes made to this document since the last released edition.

Date	**Description**
September 2021	Rebranding

Configuring Ironstream

This chapter describes how to configure Ironstream for Micro Focus Universal Discovery for IBM Z. This chapter assumes that you have already followed the product installation instructions in the *Installation Guide* document.

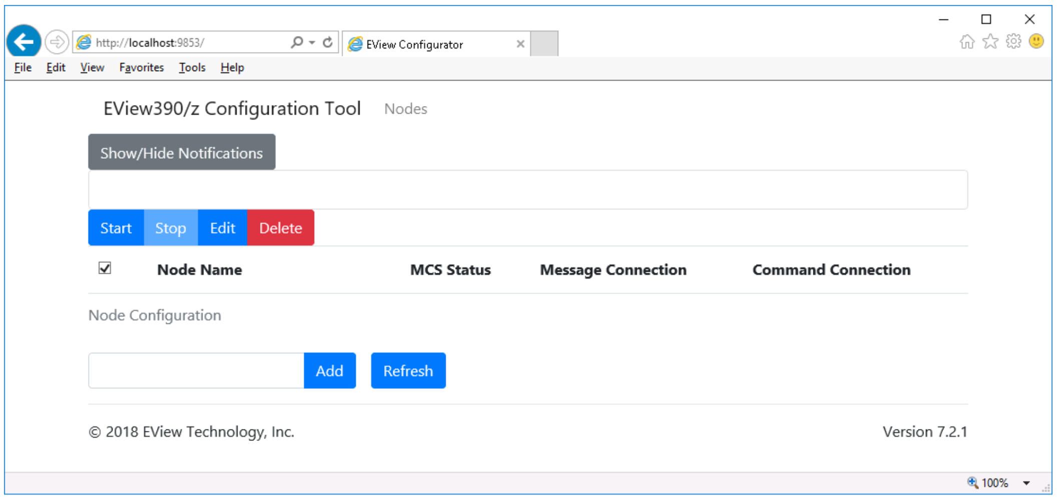
Phase 1: Configuring the Ironstream Mainframe Agent

Customization of the Ironstream agent job is accomplished by editing the appropriate parameter datasets on the IBM Z mainframe, then restarting the started task. Details for mainframe customization are given in the *Installation Guide*. Follow the instructions in the *Installation Guide* and start the job on the mainframe agent before continuing with the configuration on the Discovery Probe server.

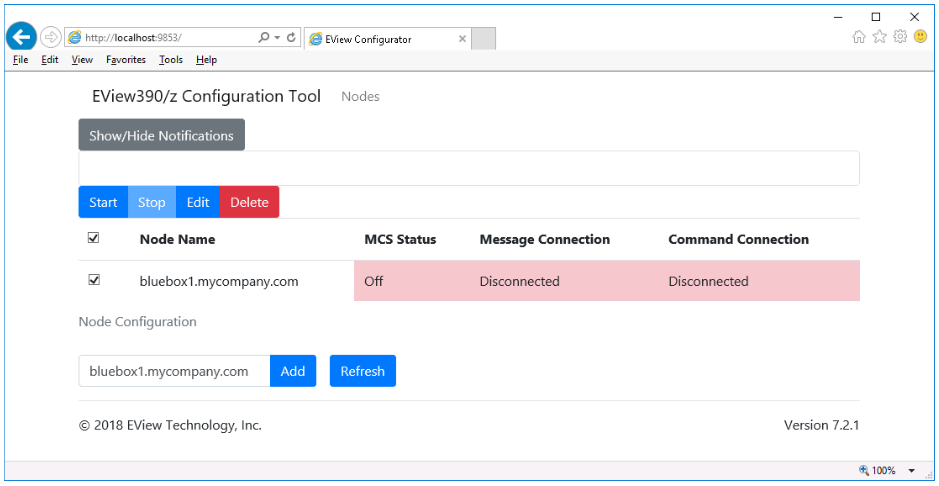
Phase 2: Adding IBM Z Systems to the Ironstream Configuration Tool

Start the Ironstream for Micro Focus Universal Discovery for IBM Z Configuration Tool web interface by entering the following from a web browser on the Ironstream UCMDB Probe server:

```
http://localhost:9853
```



1. Enter a fully qualified domain name for the mainframe LPAR in the New node name box to register the new node. The new node will be added to the list of Ironstream processes.


http://localhost:9853/ EView Configurator
File Edit View Favorites Tools Help

EView390/z Configuration Tool Nodes

Show/Hide Notifications

Start Stop Edit Delete

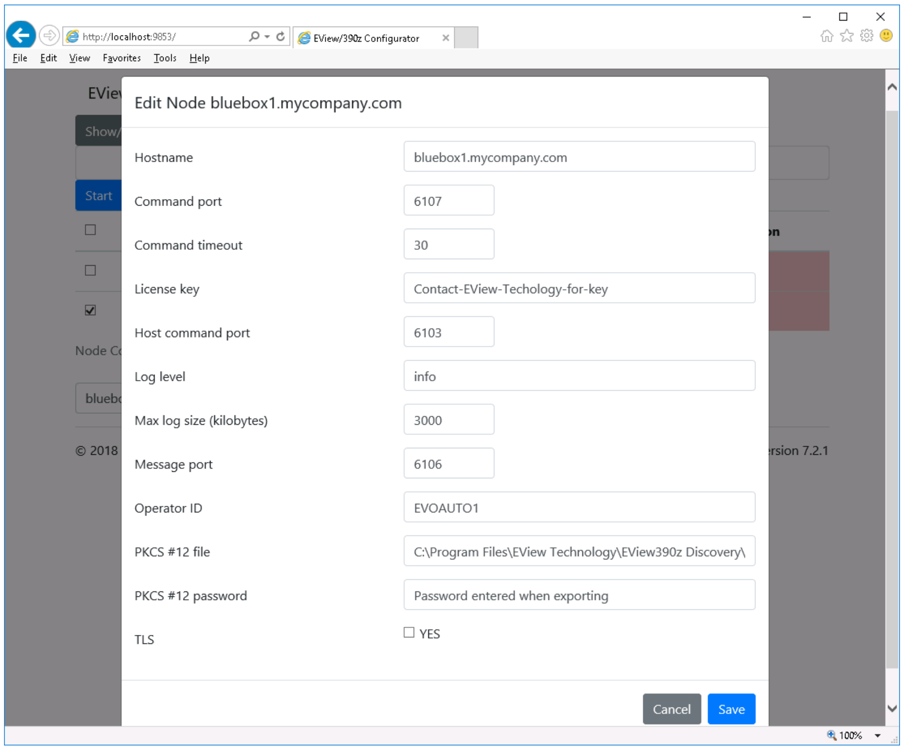
☑	Node Name	MCS Status	Message Connection	Command Connection
☑	bluebox1.mycompany.com	Off	Disconnected	Disconnected

Node Configuration

bluebox1.mycompany.com Add Refresh

© 2018 EView Technology, Inc. Version 7.2.1

2. Check the box on the left of the Node Name and Click the **Edit** button to modify the parameters for the processes which will run for this new node. The default parameter values will allow an Ironstream connection to all necessary processes, but you may change these values if there is a conflict with other software running on your system.


http://localhost:9853/ EView/390z Configurator
File Edit View Favorites Tools Help

Edit Node bluebox1.mycompany.com

Hostname	bluebox1.mycompany.com
Command port	6107
Command timeout	30
License key	Contact-EView-Techology-for-key
Host command port	6103
Log level	info
Max log size (kilobytes)	3000
Message port	6106
Operator ID	EVOAUTO1
PKCS #12 file	C:\Program Files\EView Technology\EView390z Discovery\
PKCS #12 password	Password entered when exporting
TLS	☐ YES

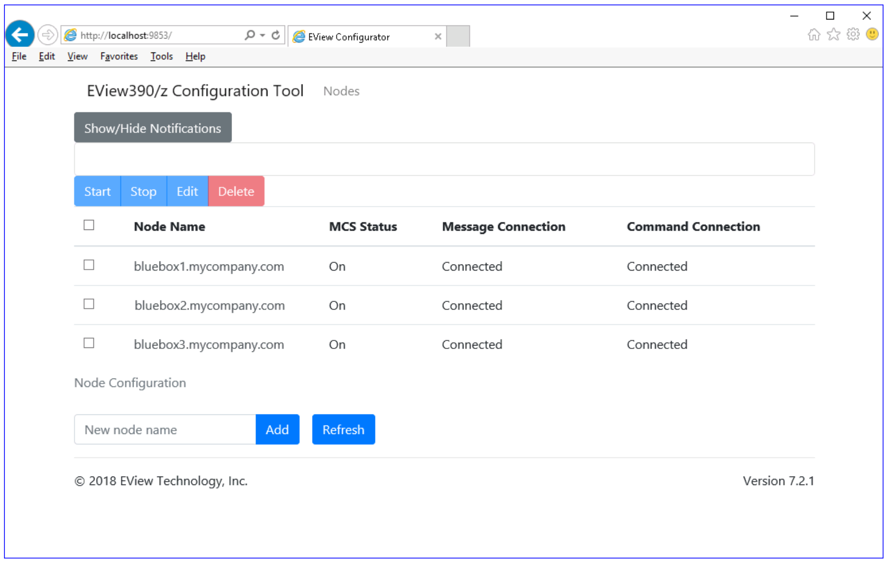
Cancel Save

Parameter	**Description**
Hostname	The network name of the managed IBM Z LPAR node. The name may need to be fully qualified, depending on your DNS setup. Create a separate node for each LPAR.

Host command port	A port reserved for Ironstream inter-process communications on the UCMDB Probe server. The default port is 6101, and must be unique on the UCMDB Probe server. It will increase automatically to avoid port conflicts as other IBM Z nodes are added. **Valid Value**: Any unused port number on the UCMDB Probe server **Note**: The port must be open for and outbound traffic.
Message port	The TCP/IP port on the IBM Z LPAR that the UD Probe server connects to, to receive messages. The default port number is 6106. **Valid Value**: Available mainframe TCP/IP port (greater than 1024). **Note**: The port must be in a Listening status on the IBM Z LPAR. **Note**: This port number **MUST** match the first number on the TCP parameter card for the started task on the mainframe agent. (See the TCP card definition in the *Installation Guide*.)
Command port	The TCP/IP port on the IBM Z LPAR that the Ironstream UD Probe server connects to, to send commands. The default port number is 6107. **Valid Value**: Available mainframe TCP/IP port (greater than 1024). **Note**: The port must be in a Listening status on the IBM Z LPAR. **Note**: This port number MUST match the second number on the TCP parameter card for the started task on the mainframe agent. (See the TCP card definition in the *Installation Guide*.)
Command timeout	The amount of time (in seconds) to wait for a response from a IBM Z command. The default is 30 seconds.
Operator ID	Defines the name of the NetView/390 autotask ID under which commands may be issued. If NetView/390 is in use on the mainframe, this name must match the name of the autotask defined in the NetView/390 DSIPARM(DSIOPF) member. The default is EVOAUTO1.

License key	The license key for this mainframe node. Each mainframe managed node requires a unique key to connect to the Ironstream management server, although the same key may be used for multiple LPARs of the same physical IBM Z system. See the *Installation Guide* for information on acquiring a license key.
Max log size (kilobytes)	Maximum size (in Kilobytes) of any file which is created on the Discovery probe server by Ironstream for logging or debug purposes. The default is 3000 (3MB).
Log level	Granularity of log tracing. Valid values are: • info • debug • warning • error The default is info.
TLS	If selected, enables TLS encryption between the server and the mainframe agent. See the *Installation Guide* for instructions on setting up a TLS encrypted connection.
PKCS 12 file	The full path and filename of the PKCS #12 file that was exported from the mainframe when using TLS encryption. See the *Installation Guide* for instructions on setting up a TLS encrypted connection.
PKCS 12 password	The password of the PKCS #12 file that was exported from the mainframe when using TLS encryption. See the *Installation Guide* for instructions on setting up a TLS encrypted connection.

3. Click **Save** to close the Node Editor window, then select the added node and click the **Start** button to start the Ironstream processes for the node.



EView390/z Configuration Tool Nodes

Show/Hide Notifications

Start Stop Edit Delete

☐	Node Name	MCS Status	Message Connection	Command Connection
☐	bluebox1.mycompany.com	On	Connected	Connected
☐	bluebox2.mycompany.com	On	Connected	Connected
☐	bluebox3.mycompany.com	On	Connected	Connected

Node Configuration

New node name Add Refresh

© 2018 EView Technology, Inc. Version 7.2.1

Phase 3: Starting and Stopping the Ironstream Processes

Use the Ironstream Configuration Tool to start and stop the processes for each defined IBM Z node.

- To *start* the Ironstream processes, click each line of processes listed in the Ironstream Configuration Tool and click the **Start** button.
- To *stop* the Ironstream processes, click each line of processes listed in the Ironstream Configuration Tool and click the **Stop** button.

Sending Commands to the Agent

The Mainframe Discovery Adapter utilizes the UD Probe server interface to communicate requests to the Ironstream agent on the IBM Z systems. The UD Probe server interface provides the capability to request the agent to execute z/OS console commands, subsystem commands, reply to outstanding WTORs, execute REXX programs and requests other information via the Ironstream mainframe agent.

The `ev390hostcmd` is the interface to the Ironstream UD Probe server for executing mainframe commands, subsystem commands, REXX programs or requesting information via the Discovery agent. The format of `ev390hostcmd` is:

```
ev390hostcmd <type> <command>.<zOS_system_name>
```

where:

`<type>`	Specifies a code to direct where the command should be executed on the mainframe. Three codes are valid: **40** = z/OS (MVS) commands. The command is sent to a MCS console defined for VP390 on the mainframe. The VP390 mainframe job must have a CMD subtask defined. **45** = z/OS (MVS) commands that do not return a response. This is the same as a Type 40 code, except the `ev390hostcmd` does not wait for a response message with this option. **46** = z/OS system information commands. The command instructs the VP390 mainframe task to gather specific z/OS system information such as CPU usage or JES2 job queue contents. See section "Sending Commands to the Agent" for the syntax of a type 46 command. The VP390 mainframe job must have an OSINFO subtask defined.
`<command>`	The command text, syntax dependent on the type. The first period (.) encountered is used to mark the end of the command. If the command text has a period in it, enter two periods to signify that it is not the end of the command. See the example below.
`<zOS_system>`	The IBM Z system on which the command is to be executed. Use the IP name of the mainframe domain. This name must match the name of the IBM Z system that was configured in the Ironstream Configuration Tool.

Examples:

- Send an MVS command to the mainframe named myhost.mysite.com to display the system time:

```
ev390hostcmd 40 D T.myhost.mysite.com
```

- Send an MVS command to the mainframe named myhost.mysite.com to start a job named MYJOB with a job identifier of MYIDENT (note that the period between MYJOB and MYIDENT must be doubled to signify that it is not the end of the command):

  ```
  ev390hostcmd 45 S MYJOB..MYIDENT.myhost.mysite.com
  ```

Using OSINFO System Information API Commands

The OSINFO subtask of the VP390 agent task will gather various z/OS operating system statistics and present the data in a format that can be parsed by a script. The OSINFO subtask can also be requested to run REXX programs or list information from physical sequential or partitioned datasets. OSINFO data are requested using type 46 of the `ev390hostcmd` utility. (See section "Sending Commands to the Agent" for syntax of `ev390hostcmd`.)

The OSINFO data are requested by specifying a two-digit code followed by a vertical bar and additional parameter information depending on the selected code. For example, to gather DASD information (code 01) for a volume named "disk99" on mainframe "s390.mysite.com," the `ev390hostcmd` syntax is:

```
ev390hostcmd 46 "01|DISK99.s390.mysite.com"
```

Because the Windows/DOS shell interprets the vertical bar as a pipe symbol, the vertical bar will need to be escaped by enclosing everything after the last space inside double quotation marks.

Output lines for requests other than REXX programs or dataset listings will be returned with values separated by a vertical bar. One line will be generated for each record found, representing one job, device, etc. The last line will be the text "EOF".

Note: Codes 06, 07, and 08 require SDSF to be running on the mainframe agent and will require the extra DD cards ISFIN and ISFOUT to be uncommented in the VP390 startup JCL

Note: Codes 110, 111, 112, 113, and 114 require a temporary dataset to hold the TSO output. The dataset should be created with parameters DCB=(DSORG=PS,RECFM=FB,LRECL=1028,BLKSIZE=6144),SPACE=(TRK,(1,1,0)). This dataset can be defined in three ways and will be searched in the following order:

1. The dataset name is defined in the SYSIN parameter cards on a card labeled "TSOUTPUT". The dataset must exist and be cataloged before the VP390 job is started.
2. The dataset is defined in the VP390 startup JCL with a DDname of "TSOUTPUT".
3. A dataset named "EV390.TSOUTPUT" will be created when a 110-114 command is called and will persist after the VP390 job has ended.

TSO options (110-114) require that the "NETSTAT" command is not in the list of TSO authorized commands. (Verify that "NETSTAT" is not listed in the AUTHCMD section of the IKJTSOxx member of PARMLIB.)

The available OSINFO codes are:

01 DASD Utilization Statistics

Description	Collects DASD volume statistics. The DASD must be online at the time of the request.

Parameters	DASD volume name, or a regular expression to look for multiple volumes, or * for all volumes
Output	One line for each DASD volume found, in the format: VolSer | Number of tracks | Tracks per cylinder | Free extents | Free tracks | Largest free extent | Percent used | DSCBs
Sample Command	**`ev390hostcmd 46 01\\|O..*.s390.mysite.com`**

Sample Output	`OS390M1|50085|15|8|3374|1230|93|1364`
	`WORK01|50085|15|23|16450|15928|67|3704`
	`EOF`

02 RMF Address Space Resource Statistics

Description	Collects statistics from RMF for a specified address space(s). RMF must be running on the system for this option to collect.
Parameters	Address space name, or a prefix of address space with an * to find multiple address spaces with the same starting characters.
Output	One line for each address space found in the format: Job Name | Device connect time in milliseconds |Number of fixed frames located below the 16M real line | Number of non-LSQA fixed frames | LSQA pages in real storage | Total TCB time for this step in milliseconds | Total CPU time consumed on behalf of this address space in milliseconds | EXCP count for this step
Sample Command	**`ev390hostcmd 46 02\\|VTAM.s390.mysite.com`**

Sample Output	`VTAM     |4589|0|29|66|333806|411134|4234`
	`EOF`

03 Current CPU Snapshot for System and Specific Address Space

Description	Collects CPU and memory usage for the system and a specific address space by scheduling an SRB to execute in the target address space.

Parameters	Address space name.
Output	One line of values in the format: Current total LPAR CPU utilization percentage | Percentage of CPU used by specified address space | Total CPU time used by address space in seconds | Real storage used by address space in kilobytes | Extended stage used by address space in kilobytes | Region size requested in kilobytes | Private storage allocated under the 16M line | Private storage allocated above the 16M line | Private storage used under the 16M line | Private storage used above the 16M line
Sample Command	**`ev390hostcmd 46 03\\|LLA.s390.mysite.com`**

Sample Output	`4.14|  0.00|  7.53|  1776|  464|   0|  940|   21424|`
	`849|  21115`
	`EOF`

04 Current Active Jobs

Description	Collects a list of active address spaces.
Parameters	Regular expression filter of address space names to be displayed, or "*" for all
Output	One line for each address space found, in the format: Job name | Step name | Proc step | Job ID | Owner | Position | Performance Group number | Priority | Current real storage usage (in frames)
Sample Command	**`ev390hostcmd 46 04\\|^V..*.s390.mysite.com`**

Sample Output	`VLF     |VLF    |VLF     |        |       |N/S|0|FE|3679`
	`VTAM    |VTAM   |VTAM    |STC00766|VTAM   |N/S|0|FE|835`
	`VMCF    |VMCF   |IEFPROC |        |       |N/S|0|FE|35`
	`VP390V55|STEP1  |        |STC01381|IBMUSER|N/S|0|FE|1702`
	`VP390   |VP390  |VP390   |STC01104|IBMUSER |N/S|0|FE|1262`
	`EOF`

05 System statistics from RMF

Description	Collects current system statistics as reported by RMF type 79 subtype 3, subtype 4, and subtype 9 records. RMF must be running to get a valid output.
Parameters	none
Output	One line of output in the format: System CPU utilization percentage | System demand paging rate | Number of system common (LPA+CSA) pages in | Number of swaps (out) | Number of pages swapped in | Number of pages swapped out | Number of private pages swapped in | Number of private pages swapped out | High UIC count | System LPA pages in | Number of pages to extended storage | Number of extended storage slots available and not in use | Number of pages migrated from extended storage to auxiliary storage | Number of available frames | I/O activity rate: average I/O requests per second | I/O response time: average milliseconds needed to complete an I/O request | ISOQ time: average milliseconds an I/O request must wait on an IOS queue | Number of fixed SQA frames | Number of common (LPA+CSA) frames | Number of private non-LSQA fixed frames | Number of address spaces in storage | Number of total LPA frames | Number of total CAS frames | Number of LPA fixed frames | Number of CSA fixed frames | Number of fixed LSQA frames | Number of address spaces logically swapped out|Current time in seconds from the beginning of the 1970 epoch
Sample Command	**`ev390hostcmd 46 05.s390.mysite.com`**

Sample Output	`5|0|40672|3762|159116|148700|286378|216962|254|26391|8772189|`
	`6352|2058999|328|0|3|0|4491|581|1455|52|3281|2052|68|513|4968`
	`|9|1089391325`
	`EOF`

06 JES2 Input Queue

Description	Collects a list of jobs on the JES2 Input Queue. See the note in section " Using OSINFO System Information API Commands".
Parameters	Job name, or a prefix of a job name with an * to find multiple jobs with the same starting characters.
Output	One line for each job found, in the format: Job name | Job ID | Owner | JES2 input queue priority | JES2 input class | Position within JES2 input queue class | Print designating name | Print routing | Print node | System affinity (if any)

Sample Command	**`ev390hostcmd 46 06|*.s390.mysite.com`**

Sample Output	`COPYJOB  |JOB01817|USER1    |   9|A|     |LOCAL    |      |      |`
	`EOF`

07 JES2 Output Queue

Description	Collects a list of jobs on the JES2 Output Queue. See the note in section " Using OSINFO System Information API Commands" for extra SDSF requirements to run this option.
Parameters	Job name, or a prefix of a job name with an * to find multiple jobs with the same starting characters.
Output	One line for each job found, in the format: Job name | Job ID | Owner | JES2 output group priority | JES2 output class | Output form number | Print destination name | Output total record count (lines) | Output creation due
Sample Command	**`ev390hostcmd 46 07|*.s390.mysite.com`**

Sample Output	`SDSF    |STC00024|START2   |144|A|STD    |LOCAL    |223       |05/13/2004`
	`SMFDUMP |JOB00091|USER42   |144|A|STD    |LOCAL    |50     |05/14/2004`
	`SYSLOG  |STC01405|+MASTER+| 96|L|STD    |LOCAL    |20682     |06/09/2004`
	`COMPRESS|JOB00166|IBMUSER |144|T|STD    |LOCAL    |6283      |10/19/2004`
	`EOF`

08 JES2 Held Queue

Description	Collects a list of jobs on the JES2 Held Queue. See the note in section " Using OSINFO System Information API Commands for extra SDSF requirements to run this option.
Parameters	Job name, or a prefix of a job name with an * to find multiple jobs with the same starting characters.
Output	One line for each job found, in the format:

	Job name | Job ID | Owner | JES2 output group priority | JES2 output class | JES2 output disposition | Print destination name | Output total record count (lines) | Output creation date
Sample Command	**`ev390hostcmd 46 08|T*.s390.mysite.com`**

Sample Output	`TCPIP    |STC00577|TCPIP   |144|K|HOLD |LOCAL   |22       |10/19/2004`
	`TSO      |STC00803|++++++++|144|K|HOLD |LOCAL   |12       |10/19/2004`
	`EOF`

10 Dataset Display

Description	Displays the contents of a sequential dataset, dataset member, or HFS file. If a partitioned dataset name is given without a member name, this command will return a list of all members in the dataset. This command cannot display VSAM datasets or datasets with unformatted records.
Parameters	Dataset or file name. For PDS members, specify the member name in parentheses. For HFS files, give the full path to the filename. (If the first character of the name is a forward slash (/), it is assumed that an HFS file is being described.) Remember that the syntax of the `ev390hostcmd` requires two consecutive periods to denote one period in any parameter. Optionally, specify "`maxsize=n`" as a second parameter to limit the number of lines of output. The default maxsize is 5000 lines. If the maximum number of lines is exceeded, an EVO140 message will be written to the output.
Output	One line for each line of the file.

Sample Command	1. Display all the lines of the /etc/hosts HFS file:
	`ev390hostcmd 46 10|/etc/hosts.s390.mysite.com`
	2. Display all the lines of the VP390 member in the USER.PROCLIB dataset:
	`ev390hostcmd 46 10|USER..PROCLIB(VP390).s390.mysite.com`
	3. Display all the members of the USER.PROCLIB partitioned dataset:
	`ev390hostcmd 46 10|USER..PROCLIB.s390.mysite.com`
	4. Display all the lines of the THOMAS.DAILY.LOG sequential dataset:
	`ev390hostcmd 46 10|THOMAS..DAILY..LOG.s390.mysite.com`
	5. Display only the first 4 lines of the THOMAS.DAILY.LOG sequential dataset:
	`ev390hostcmd 46`
	`10|THOMAS..DAILY..LOG|maxsize=4.s390.mysite.com`

Sample Output	`Line 1 of log`
	`Line 2 of log`
	`Line 3 of log`
	`Line 4 of log`
	`EVO140  Maximum lines of output exceeded (4)`
	`EOF`

12 Execute REXX Program

Description	Run the named REXX program. The named program must reside in a dataset identified by the SYSEXEC DD in the VP390 startup JCL.
Parameters	The first parameter is the name of the REXX program to run, followed by any parameters that are to be passed to the REXX program. Separate the program name from the program parameters with a vertical bar and separate the program parameters with white space. Up to 50 program parameters may be specified. Optionally, specify "`maxsize=n`" as the last parameter to limit then number of bytes of output that are returned. The default maxsize is 5000 bytes.
Output	The value of the REXX `return` statement will be sent as the output. Normally this will be one line, but multiple lines can be created by inserting carriage return (x'0A') characters into the value that is returned.

Sample Command	The following REXX program is saved as "TEST" in a SYSEXEC DD library:
	`/* REXX program that accepts two whole numbers and returns     */`
	`/* two lines of output: the numbers and the sum of the numbers */`
	`parse upper arg parm1 parm2 .`
	`if DATATYPE(parm1,'W') \= 1 | DATATYPE(parm2,'W') \= 1 then`
	`  do`
	`  n = 'Inputs must be whole numbers'`
	`  return n`
	`  exit`
	`  end`
	`linefeed = X2C('0A')`
	`n = 'Input numbers:' parm1 parm2 linefeed 'sum is:' parm1 + parm2`
	`return n`
	`exit`
	
	Run the program with the parameters "111" and "222" with a maximum output of 6000 bytes. Note that the X2C('0A') function inserts a line break into the output:
	`ev390hostcmd 46 12\\|TEST\\|111 222\\|maxsize=6000.s390.mysite.com`

Sample Output	`Input numbers: 111 222`
	` sum is: 333`
	`EOF`

20 Machine Type and LPAR Capacity

Description	Displays the mainframe machine type and the LPAR capacity of the LPAR where the mainframe agent is running.
Parameters	None
Output	One line in the format: Machine type | Machine Model ID | Machine Capacity | LPAR Name | LPAR ID | LPAR Capacity The “Machine Capacity” and “LPAR Capacity” values represent the maximum service rates in millions of service units per hour.
Sample Command	**`ev390hostcmd 46 "20.s390.mysite.com"`**

Sample Output	`2098|D02                |11|ZOS2    |2|11`
	`EOF`

40 Active Jobs and Program Name

Description	Collects a list of active address spaces including the program name of the current step.
Parameters	Regular expression filter of job names to be displayed, or "*" for all.
Output	One line for each address space found, in the format: Job name | Step name | Proc step | Job ID | Owner | Position | Performance Group number | Priority | Current real storage usage in frames | Program Name If Program Name is not available, the field is filled in with “*NA”.
Sample Command	**`ev390hostcmd 46 "40|A.s390.mysite.com"`**

Sample Output	`*MASTER*|         |         |STC05258|+MASTER+|N/S|0|FF|295|IEEMB860`
	`PCAUTH  |PCAUTH   |         |         |         |N/S|0|FB|40|*NA`

	```
	RASP    |RASP    |        |        |        |N/S|0|FF|161|*NA
	TRACE   |TRACE   |        |        |        |N/S|0|FB|41|*NA
	IOSAS   |IOSAS   |IEFPROC |        |        |N/S|0|FF|1013|IOSVROUT
	LLA     |LLA     |LLA     |        |        |N/S|0|FE|865|CSVLLCRE
	VTAM    |VTAM    |VTAM    |STC05255|START1  |N/S|0|FE|1483|ISTINM01
	RACF    |RACF    |RACF    |STC05270|START2  |N/S|0|FE|58|IRRSSM00
	JES2AUX |JES2AUX |        |        |        |N/S|0|FB|50|*NA
	PORTMAP |PORTMAP |PMAP    |STC05368|START2  |OUT|0|FF|330|PORTMAP
	RMFGAT  |RMFGAT  |IEFPROC |STC05358|START2  |N/S|0|FE|9695|ERB3GMFC
	EOF
	```

41 Display IMS Subsystem Name

Description	Displays the IMS subsystem name for the given job.
Parameters	Regular expression filter of job names to be displayed, or "*" for all.
Output	One line for each address space found, in the format: Job name | Subsystem name If the given job is not an IMS job (that is, the job is not running program DFSMVRC0), the subsystem name field is filled in with "*NA".
Sample Command	**`ev390hostcmd 46 "41|^IMS.s390.mysite.com"`**

Sample Output	
	```
	IMS10RL1|*NA
	IMS10CR1|IVP1
	IMS10DL1|IVP1
	IMS10RC1|IVP1
	EOF
	```

42 Display Program Name and PARM Value

Description	Displays the program name and any PARM value from the startup JCL of the named job.

Parameters	Regular expression filter of job names to be displayed.
Output	One line for each address space found, in the format: Job name | Program name | PARM value
Sample Command	**`ev390hostcmd 46 "42|^CICS.s390.mysite.com"`**

Sample Output	`CICSA    |DFHSIP  |START=INITIAL,SYSIN`
	`EOF`

43 Display DDNames of Job

Description	Displays the DD names and associated dataset names for the named job.
Parameters	Regular expression filter of job names to be displayed.
Output	One line for each dataset name found, in the format: Job name | DDName | Dataset name DD names with several datasets concatenated will be displayed with a separate line for each dataset name. DD names that point to partitioned dataset members will show the member name in parentheses immediately after the dataset name.
Sample Command	**`ev390hostcmd 46 "43|VTAM.s390.mysite.com"`**

Sample Output	`VTAM    |VTAMLST |USER.VTAMLST`
	`VTAM    |VTAMLST |SYS1.VTAMLST`
	`VTAM    |VTAMLIB |USER.VTAMLIB`
	`VTAM    |VTAMLIB |SYS1.VTAMLIB`
	`VTAM    |SISTCLIB|SYS1.SISTCLIB`
	`VTAM    |ACYGDMO |SYS1.SISTGDMO(ACYGDMO)`
	`VTAM    |SYSABEND|START1.VTAM.STC06385.D0000101.?`
	`VTAM    |TRSDB   |SYS1.TRSDB`
	`EOF`

44 Display CICS Group Lists

Description	Executes a WebSphere MQ command and displays the command output.
Parameters	None
Output	One line for each address space found, in the format: Job name | Group List 1|Gropup List 2|Group List 3| Group List 4 Group list names will be padded out to 8 characters.
Sample Command	**`ev390hostcmd 46 "44|^CICS.s390.mysite.com"`**
Sample Output	`CICS1 |DFHLIST	|DFHPGADX|XYZLIST` `EOF`

50 Execute MQ Series Command

Description	Executes a WebSphere MQ command and displays the command output.
Parameters	MQ Manager name followed by the WebSphere MQ command to be executed. Optionally, specify "`maxsize=n`" as a third parameter to limit the number of lines of output. The default maxsize is 64000 bytes. If the maximum number of bytes is exceeded, an EVO141 message will be written to the output.
Output	One line for each line of command output.

Sample Command	**`ev390hostcmd 46 "50|CSQ7|DISPLAY CHANNEL(*).s390.mysite.com"`**
	`ev390hostcmd 46 "50|CSQ7|DISPLAY CHANNEL(*)|maxsize=70000.s390.mysite.com"`

Sample Output	`CSQN205I   COUNT=      13, RETURN=00000000, REASON=00000000`
	`CSQM410I %CSQ7 CHANNEL(mars.to.venus) CHLTYPE(SDR) QSGDISP(QMGR)`
	`CSQM412I %CSQ7 CHANNEL(venus.to.mars) CHLTYPE(RCVR) QSGDISP(QMGR)`
	`CSQM417I %CSQ7 CHANNEL(SYSTEM.DEF.CLUSRCVR) CHLTYPE(CLUSRCVR) QSGDISP(QMGR)`
	`CSQM418I %CSQ7 CHANNEL(SYSTEM.DEF.CLUSSDR) CHLTYPE(CLUSSDR) QSGDISP(QMGR)`
	`CSQM412I %CSQ7 CHANNEL(SYSTEM.DEF.RECEIVER) CHLTYPE(RCVR) QSGDISP(QMGR)`

	`CSQM413I %CSQ7 CHANNEL(SYSTEM.DEF.REQUESTER) CHLTYPE(RQSTR) QSGDISP(QMGR)`
	`CSQM410I %CSQ7 CHANNEL(SYSTEM.DEF.SENDER) CHLTYPE(SDR) QSGDISP(QMGR)`
	`CSQM411I %CSQ7 CHANNEL(SYSTEM.DEF.SERVER) CHLTYPE(SVR) QSGDISP(QMGR)`
	`CSQM415I %CSQ7 CHANNEL(SYSTEM.DEF.SVRCONN) CHLTYPE(SVRCONN) QSGDISP(QMGR)`
	`CSQM412I %CSQ7 CHANNEL(VENUS.TO.MARS) CHLTYPE(RCVR) QSGDISP(QMGR)`
	`CSQ9022I %CSQ7 CSQMDRTS ' DISPLAY CHANNEL' NORMAL COMPLETION`
	`EOF`

60 Display TCP/IP Connections

Description	Displays "netstat" type information from a mainframe TCP/IP stack.
Parameters	[stack|*][|protocol[,protocol]...][|stat][,stat]...][|verbosity] where: *stack*: An 8-character TCP/IP jobname, or "*" for first active stack (default=*) `protocol`: TCP|UDP|ALL Display TCP only, UDP only, or both (default=ALL) `stat`: CONN|LISTEN|ALL Display connected or listening ports, or both (default=ALL) `verbosity`: SHORT|LONG The LONG output requires slightly more processing time to gather the additional information (default=SHORT)
Output	One line for each active port, in the format (depending on the verbosity requested): SHORT Output: Protocol | IP Family | Local Address | Local Port | Remote Address | Remote Port | Bytes Received | Bytes Sent | State LONG Output:

	Protocol | IP Family | Local Address | Local Port | Remote Address | Remote Port | Bytes Received | Bytes Sent | State |Connection Time (seconds) | Idle Time (seconds) | Owning Jobname

Sample Command	**`ev390hostcmd 46 "60|TCPIP.s390.mysite.com"`**
	`ev390hostcmd 46 "60|*|ALL|ALL|LONG.s390.mysite.com"`

Sample Output	`TCP|IPV4|192.168.0.210	6106|192.168.0.174 |64940| 0|`
		`5404|ESTABLISH |203|3|EV390V70`
		`TCP|IPV4|192.168.0.210
			6107|192.168.0.174 |64939| 367|`
			`15|ESTABLISH |206|0|EV390V70`
			`TCP|IPV6|::ffff:192.168.0.210|   23|::ffff:192.168.1.128|52017|   4949|`
			`159675|ESTABLISH|11866| 160|TN3270`
			`UDP|IPV4|192.168.0.222 |12000|0.0.0.0
				0| 0|`
				`0|UDP|1370650|1370650|VTAM`
				`UDP|IPV4|192.168.0.222 |12004|0.0.0.0
					0| 0|`
					`0|UDP|1370650|1370650|VTAM`
					`TCP|IPV6|::
						23|::	0| 0|`
							`0|LISTEN|1371721|2091|TN3270`
							`TCP|IPV6|::
								21|::	0| 0|`
									`0|LISTEN|1371713|736247| FTPD1`
									`UDP|IPV4|0.0.0.0
										111|0.0.0.0	0| 0|`
											`0|UDP|1370642|1370642|PORTMAP`
											`EOF`

61 Display Network Interfaces

Description	Displays list of defined interfaces and devices for a TCP/IP stack.
Availability	Agent running z/OS V1R12 and later.
Parameters	Regular expression filter of job names to be displayed, or “*” for all.
Output	One line for each interface found, in the format: Interface Name | Interface Associated Name | MAC Address | PhysVirt | Status | Interface Type The PhysVirt value will be either “P” to indicate that the MAC address listed is a Physical MAC address, or ”V” to indicate a configured or OSA-generated VMAC address.
Sample Command	**`ev390hostcmd 46 "61|*.s390.mysite.com"`**

Sample Output

```
LOOPBACK        |LOOPBACK       |00-00-00-00-00-00|P|ACTIVE  |LOOPBACK     |IPV4|127.0.0.1|
LOOPBACK6       |               |00-00-00-00-00-00|P|ACTIVE  |LOOPBACK     |IPV6|::1||LOOPBACK|
VLINK2          |VIPA2          |00-00-00-00-00-00|P|ACTIVE  |STATIC-VIRT
|IPV4|192.168.0.222|PRIMARY
OSDL            |ADM1ETP        |00-14-5E-B8-81-95|P|ACTIVE  |ETHERNET-OSD |IPV4|192.168.0.218|
OSDL6           |ADM1ETP        |00-14-5E-B8-81-95|P|ACTIVE  |ETHERNET-OSD
|IPV6|fe80::14:5e00:1b8:8195| AUTOCONFIG|LINK_LOCAL|
OSDL6           |ADM1ETP        |00-14-5E-B8-81-95|P|ACTIVE  |ETHERNET-OSD
|IPV6|fda5:3ad7:3471:5::218| |GLOBAL|
EZASAMEMVS      |IUTSAMEH       |00-00-00-00-00-00|P|ACTIVE  |MPC-P2P-SAME |IPV4|192.168.0.212|
EZAXCF2A        |Z113SSCP       |00-00-00-00-00-00|P|ACTIVE  |MPC-P2P-XCF  |IPV4|192.168.0.212|
VIPLC0A801DB    |VIPLC0A801DB   |00-00-00-00-00-00|P|ACTIVE  |DYNAMIC-VIRT  IPV4|192.168.0.219|
EOF
```

110 TSO-Generated Netstat Connections

Description	Displays the network configuration and status of a TCP/IP stack via a "netstat" request to TSO. This option requires a temporary dataset to hold the TSO output (see the in the note in the section: " Using OSINFO System Information API Commands").
Parameters	An 8-character TCP/IP jobname, or "*" for first active stack (default=*) (Optional) "APPLDATA" to show detailed information concerning application data for TCP connections.
Output	TSO-styled "NETSTAT ALLConn" output. Note that the format of the output changed after z/OS V1R10.

Sample Command	`ev390hostcmd 46 "110|*.s390.mysite.com"`
	 `ev390hostcmd 46 "110|*|APPLDATA.s390.mysite.com"`

Sample Output	
	```
	MVS TCP/IP NETSTAT CS V2R1       TCPIP Name: TCPIP         22:03:36
	User Id  Conn     State
	-------  ----     -----
	BPXOINIT 00000024 Listen
	  Local Socket:   0.0.0.0..10007
	  Foreign Socket: 0.0.0.0..0
	CICSWUIP 00000050 Listen
	  Local Socket:   ::..12345
	  Foreign Socket: ::..0
	  Application Data:  DFHIWUINCM01CWXNHTTP    EYUWUI
	CICSWUIP 00000052 Listen
	  Local Socket:   ::..12346
	  Foreign Socket: ::..0
	  Application Data:  DFHIWUINCM01CWXNHTTP    EYUCMCIT
	EV390V70 0004ED53 Establish
	  Local Socket:   192.168.1.210..6107
	  Foreign Socket: 192.168.1.174..64939
	EV390V70 0004ED55 Establish
	  Local Socket:   192.168.1.210..6106
	  Foreign Socket: 192.168.1.174..64940
	FTPD1    00000022 Listen
	  Local Socket:   ::..21
	  Foreign Socket: ::..0
	  Application Data:  EZAFTP0D
	EOF
	```

111 TSO-Generated Netstat Home Addresses

Description	Displays the IP address and associated links or interface name via a "netstat" request to TSO. This option requires a temporary dataset to hold the TSO output (see the in the note in the section: " Using OSINFO System Information API Commands").
Parameters	An 8-character TCP/IP jobname, or "*" for first active stack (default=*)
Output	TSO-styled "NETSTAT HOME" output. Note that the format of the output changed after z/OS V1R10.
Sample Command	**`ev390hostcmd 46 "111|TCPIP.s390.mysite.com"`**

Sample Output	
	```
	MVS TCP/IP NETSTAT CS V2R1       TCPIP Name: TCPIP          15:25:19
	Home address list:
	LinkName:   VLINK2
	  Address:  192.168.1.222
	    Flags:  Primary
	LinkName:   EZASAMEMVS
	  Address:  192.168.1.212
	    Flags:
	LinkName:   VIPLC0A801DB
	  Address:  192.168.1.219
	    Flags:
	LinkName:   EZAXCF2A
	  Address:  192.168.1.212
	    Flags:
	LinkName:   LOOPBACK
	  Address:  127.0.0.1
	    Flags:
	IntfName:   OSDL
	  Address:  192.168.1.210
	    Flags:
	IntfName:   OSDL6
	  Address:  fda5:3ad7:3471:5::210
	    Type:   Global
	    Flags:
	  Address:  fe80::14:5e00:1b8:8195
	    Type:   Link_Local
	    Flags:  Autoconfigured
	IntfName:   LOOPBACK6
	  Address:  ::1
	    Type:   Loopback
	    Flags:
	EOF
	```

112 TSO-Generated Netstat Routing Table

Description	Displays the TCP/IP routing table for the named TCP/IP stack via a "netstat" request to TSO. This option requires a temporary dataset to hold the TSO output (see the in the note in the section: " Using OSINFO System Information API Commands").
Parameters	An 8-character TCP/IP jobname, or "*" for first active stack (default (Optional) "APPLDATA" to show detailed information concerning application data for TCP connections.
Output	TSO-styled "NETSTAT ARP" output. Note that prior to z/OS V1R12, only IPV4 destinations are displayed. Beginning with z/OS V1R12, both IPV4 and IPV6 destinations are listed.

Sample Command	**`ev390hostcmd 46 "112|*.s390.mysite.com"`**
	`ev390hostcmd 46 "112|*|APPLDATA.s390.mysite.com"`

Sample Output	see below

```
MVS TCP/IP NETSTAT CS V2R1       TCPIP Name: TCPIP         15:28:34
IPv4 Destinations
Destination         Gateway         Flags    Refcnt     Interface
-----------         -------         -----    ------     ---------
Default             192.168.1.1     UGS      0000000000 OSDL
127.0.0.1/32        0.0.0.0         UH       0000000000 LOOPBACK
192.168.1.0/24      0.0.0.0         US       0000000007 OSDL
192.168.1.210/32    0.0.0.0         UH       0000000000 OSDL
192.168.1.211/32    0.0.0.0         UHS      0000000000 EZAXCF2A
192.168.1.212/32    0.0.0.0         UH       0000000000 EZASAMEMVS
192.168.1.212/32    0.0.0.0         UH       0000000000 EZAXCF2A
192.168.1.219/32    0.0.0.0         UH       0000000000 VIPLC0A801DB
192.168.1.222/32    0.0.0.0         UH       0000000000 VLINK2
IPv6 Destinations
DestIP:   Default
  Gw:     fda5:3ad7:3471:5::1
  Intf:   OSDL6             Refcnt:  0000000000
  Flgs:   UGS               MTU:     1500
DestIP:   ::1/128
  Gw:     ::
  Intf:   LOOPBACK6         Refcnt:  0000000002
  Flgs:   UH                MTU:     65535
DestIP:   fda5:3ad7:3471:5::/64
  Gw:     ::
  Intf:   OSDL6             Refcnt:  0000000000
  Flgs:   US                MTU:     1500
DestIP:   fda5:3ad7:3471:5::210/128
  Gw:     ::
  Intf:   OSDL6             Refcnt:  0000000000
  Flgs:   UH                MTU:     1500
DestIP:   fe80::14:5e00:1b8:8195/128
  Gw:     ::
  Intf:   OSDL6             Refcnt:  0000000000
  Flgs:   UH                MTU:     1500
EOF
```

113 TSO-Generated Netstat ARP Cache

Description	Displays the IPV4 ARP cache for the named TCP/IP stack via a “netstat” request to TSO. This option requires a temporary dataset to hold the TSO output (see the in the note in the section: “ Using OSINFO System Information API Commands”).
Parameters	An 8-character TCP/IP jobname, or “*” for first active stack (default=*)

	(Optional) Display the ARP cache for a specific network address (in the format nnn.nnn.nnn.nnn) or "ALL". "ALL" is the default.
Output	TSO-styled "NETSTAT ARP" output

Sample Command	**`ev390hostcmd 46 "113|*.s390.mysite.com"`**
	`ev390hostcmd 46 "113|192..168..1..102.s390.mysite.com"`
	(Remember that period characters must be doubled in the command portion of an `ev390hostcmd` to differentiate it from the destination agent name.)

Sample Output	`MVS TCP/IP NETSTAT CS V2R1       TCPIP Name: TCPIP         15:37:48`
	`Querying ARP cache for address 192.168.1.52`
	`Interface: OSDL              ETHERNET: 00145EB88185`
	
	`Querying ARP cache for address 192.168.1.53`
	`Interface: OSDL              ETHERNET: 00145EB88185`
	
	`Querying ARP cache for address 192.168.1.54`
	`Interface: OSDL              ETHERNET: 00145EB88185`
	
	`Querying ARP cache for address 192.168.1.55`
	`Interface: OSDL              ETHERNET: 00145EB88185`
	
	`Querying ARP cache for address 192.168.1.1`
	`Interface: OSDL              ETHERNET: 00A0CC65D8A2`
	
	`EOF`

114 TSO-Generated Netstat Device Links

Description	Displays the devices, links, and interfaces defined to a TCP/IP stack via a "netstat" request to TSO. This option requires a temporary dataset to hold the TSO output (see the in the note in the section: " Using OSINFO System Information API Commands").
Parameters	An 8-character TCP/IP jobname, or "*" for the first active stack (default=*)

Output	TSO-styled “NETSTAT DEVLINKS” output
Sample Command	**`ev390hostcmd 46 "114|*.s390.mysite.com"`**

Sample Output

```
MVS TCP/IP NETSTAT CS V2R1       TCPIP Name: TCPIP          15:51:23
DevName: LOOPBACK          DevType: LOOPBACK
  DevStatus: Ready
  LnkName: LOOPBACK          LnkType: LOOPBACK   LnkStatus: Ready
    ActMtu: 65535
  Routing Parameters:
    MTU Size: n/a               Metric: 00
    DestAddr: 0.0.0.0           SubnetMask: 0.0.0.0
  Multicast Specific:
    Multicast Capability: No
  Link Statistics:
    BytesIn                           = 119485313
    Inbound Packets                   = 1896986
    Inbound Packets In Error          = 0
    Inbound Packets Discarded         = 0
    Inbound Packets With No Protocol  = 0
    BytesOut                          = 119485313
    Outbound Packets                  = 1896986
    Outbound Packets In Error         = 0
    Outbound Packets Discarded        = 0

IntfName: OSDL              IntfType: IPAQENET   IntfStatus: Ready
    PortName: ADM1ETP   Datapath: 0402     DatapathStatus: Ready
    CHPIDType: OSD      SMCR: Disabled (GLOBALCONFIG NOSMCR)
    PNetID: *None*
    Speed: 0000000100
    IpBroadcastCapability: No
    CfgRouter: Non                   ActRouter: Non
    ArpOffload: Yes                  ArpOffloadInfo: Yes
    CfgMtu: None                     ActMtu: 1492
    IpAddr: 192.168.1.210/24
    VLANid: None                     VLANpriority: Disabled
    ReadStorage: GLOBAL (4096K)
    InbPerf: Balanced
    ChecksumOffload: Yes             SegmentationOffload: No
    SecClass: 255                    MonSysplex: No
    Isolate: No                      OptLatencyMode: No
  Multicast Specific:
    Multicast Capability: Yes
    Group             RefCnt        SrcFltMd
    -----             ------        --------
    224.0.0.1         0000000001    Exclude
      SrcAddr: None
  Interface Statistics:
    BytesIn                           = 5900185389
    Inbound Packets                   = 13211764
    Inbound Packets In Error          = 15390
    Inbound Packets Discarded         = 0
    Inbound Packets With No Protocol  = 0
    BytesOut                          = 1089099540
    Outbound Packets                  = 10967466
    Outbound Packets In Error         = 0
    Outbound Packets Discarded        = 0

DevName: VIPDC0A801DB      DevType: VIPA
  DevStatus: Ready
  LnkName: VIPLC0A801DB      LnkType: VIPA       LnkStatus: Ready
  Routing Parameters:
    MTU Size: n/a               Metric: 01
    DestAddr: 0.0.0.0           SubnetMask: 255.255.255.0
  Multicast Specific:
    Multicast Capability: No
```

```
IPv4 LAN Group Summary
LanGroup: 00002
  Name              Status      ArpOwner          VipaOwner
  ----              ------      --------          ---------
  OSDL              Active      OSDL              Yes
IPv6 LAN Group Summary
LanGroup: 00001
  Name              Status      NDOwner           VipaOwner
  ----              ------      -------           ---------
  OSDL6             Active      OSDL6             Yes

OSA-Express Network Traffic Analyzer Information:
No OSA-Express Network Traffic Analyzer interfaces are defined
EOF
```

REXX Functions Provided by Ironstream

Ironstream provides the following commands for use in REXX programs. These are located in the Ironstream LOAD library and can be used when constructing automation scripts which will be called and run in the Ironstream address space (outside of a TSO/E environment). A sample program which uses these commands can be found in the SAMP dataset member JOBSUB.

EVORXALO - Allocate/Concatenate DD Names

Description	This function allocates and concatenates the dataset name(s) listed to the DD name provided. If the 'ddname' DD is currently allocated, then datasets will be concatenated to the existing allocation. If the named dataset is already part of the DD's concatenated list, no action will be taken.
Syntax	`EVORXALO('`*`ddname`*`','`*`dsname`*`'[,'`*`dsname`*`'...])` where: *`ddname`* A 1 to 8 character symbolic DD name *`dsname`* A pre- existing and cataloged dataset name
Return Value	A variable can be assigned to the command to hold one of the following return texts: `OK` — Allocations and/or concatenations were successful `ERROR IN SPECIFYING FUNCTION` — Incorrect parameter(s) specified `ALLOCATE FAILED; RC - `*`rc`*`, REASON - `*`reas`* — See the "DYNALLOC" entry in *IBM z/OS MVS Programming: Authorized Assembler Services Guide* for explanation of the return code and reason code `CONCATENATE ERROR; RC - `*`rc`*`, REASON - `*`reas`* — See the "DYNALLOC" entry in *IBM z/OS MVS Programming: Authorized Assembler Services Guide* for explanation of the return code and reason code `DYNAMIC ALLOCATION INFORMATION PROBLEM WITH DDNAME` — The *`ddname`* was freed prior to the addition of the *`dsname`*. Retry the allocation command.

REXX Example	`x = EVORXALO('MYLIB','TEST.JCL')`
	`x = EVORXALO('DD1','TEST.LOADLIB1','TEST.LOADLIB2')`

EVORXCON - Issue Console Command

Description	The EVORXCON command is used to establish an extended console session using MVS console services. This session allows you to enter MVS system commands (or subsystem commands) from the REXX program.
Syntax	`EVORXCON('`*`cmd`*`','`*`var`*`'[,'HC'][,'`*`consname`*`'][,'NR'][,'MAXSIZE=`*`max`*`'][,'RCA'])` where: *`command`* — An MVS console command *`returnmsg`* — A 1 to 17 character variable name that will be used as a compound variable containing any response message(s) to the command *`'HC'`* — "Hard Copy": An optional third parameter which will cause the MVS command and response to be written to the hardcopy log. *`consname`* — An optional name to be used when initializing the extended console. The name must be 1-8 uppercase alphanumeric characters. If no name is given or an invalid name is specified, the name is set to the default name of "EVORXCON". If this parameter is to be specified, then the third parameter must also be specified, even if only with a null value (two consecutive commas). See the second example below. `'NR'` — "No Response": An optional parameter which will cause the EVORXCON to return without waiting for any response message(s) from the command. `'MAXSIZE=`*`max`*`'` — An optional parameter to define the maximum memory size (in bytes) to be allocated to hold the command's response messages. If not specified, the default memory allocation is 56 Kbytes.

	`'RCA'`	"Route Codes All": An optional parameter which specifies that the messages making up the response can accept all MVS route codes. If not specified, command responses are, by default, expected to have no route codes. (Note that using this option may result in receiving message responses that are not associated with the original command if other messages with route codes are generated at the same time that the command is being processed. This option is primarily intended to be used for receiving responses to a WTOR reply.)
Return Value	A variable can be assigned to the command to hold one of the return text: OK - The command completed	

REXX Example	Issue the "D A,L" command to display the names of all active address spaces on the mainframe. The response lines are printed by reading the RMSG compound variable.
	`/*REXX*/`
	`x = EVORXCON('D A,L','RMSG')`
	`if x = 'OK' then`
	`  do`
	`    do i = 1 to RMSG.0`
	`      say RMSG.i`
	`    end`
	`  end`
	`exit`
	Issue the "D R,U" command to display the devices that require operator intervention. The response lines are printed by reading the RMSG compound variable. Use the console name "REXXCON", and do not display the command/response on the hardcopy log.
	`/*REXX*/`
	`x = EVORXCON('D R,U','RMSG',,'REXXCON')`
	`if x = 'OK' then`
	`  do`
	`    do i = 1 to RMSG.0`
	`      say RMSG.i`
	`    end`
	`  end`
	`exit`	

EVORXDIR - Read PDS Directory

Description	Reads the partitioned dataset directory of the pre-allocated ddname provided, and returns the member names in a REXX compound variable. The 0 stem of the variable will contain the total number of members returned.
Syntax	`EVORXDIR('ddname','variable'[,'count'][,'directory'][,'prefix'])` where: *ddname* — An existing symbolic DD name (1 to 8 characters) *variable* — A 1 to 17 character name used to build a compound variable containing the member names in the PDS. *count* — (Optional) The maximum number of names returned. The default maximum is 1000 names. *directory* — Optional) Either 'YES' or 'NO', indicating whether to return the directory user data, the approximately 60 bytes of user halfwords (e.g., ISPF or link-edit information). When used, the member name will be the first word of the compound variable, followed by the user data. *prefix* — (Optional) Filter the output so that only member names with this prefix are returned.
Return Value	A variable can be assigned to the command to hold one of the following return texts: `OK` — Read of the dataset was successful `ERROR IN SPECIFYING FUNCTION` — An incorrect parameter was passed, either the *variable* name is too long, or the count exceeds the maximum size allowed `ERROR OPENING PDS DATASET` — The named *ddname* failed to open. `ERROR READING PDS DATASET` — There was a failure reading the *ddname* `STORAGE REQUEST FAILED` — The function was unable to allocate enough memory to hold the complete list of dataset members.

REXX Example	`EVORXDIR('LOADLIB','MEMBER')`
	`EVORXDIR('LOADLIB','MEMBER','9999','YES')`

EVORXFRE - Free Allocated DDs

Description	Dynamically frees a dataset or a DD name and its associated datasets.
Syntax	`EVORXFRE('key=name')` where: *`key`* - `DDN`" if freeing a DD name, or "`DSN`" if freeing a specific dataset name *`name`* - An existing symbolic DD name (1 to 8 characters) or a dataset name (up to 44 characters)
Return Value	A variable can be assigned to the command to hold one of the following return texts: `OK` — The free was successful `NO PARAMETER SPECIFIED` — The function was called without a valid parameter `INCORRECT PARAMETER SPECIFIED` — The *`key`* must be specified as "`DDN`" or "`DSN`"

REXX Example	`x = EVORXFRE('DDN=MYLIB')`
	`x = EVORXFRE('DSN=USER.JCL.CNTL')`

EVORXGET - Read a PDS Member

Description	This function will read a member of a PDS and return the records in the 'variable_name' specified as a compound variable (e.g. PDSRECD.1). The 'ddname' must be pre-allocated prior to invocation of the function. The '0' stem of the ‘variable_name' will contain the number of records read. The 'count' field is optional and will default to a maximum of 9,999 records.
Syntax	`EVORXGET('`*`member`*`','`*`ddname`*`','`*`returnmsg`*`'[,`*`linecount`*`])` where:

	member The member name of a partitioned dataset (1 to 8 characters) *ddname* An existing symbolic DD name (1 to 8 characters) *returnmsg* A 1 to 17 character variable name that will be used as a compound variable containing the lines (records) read from the dataset *linecount* The maximum number of records to be read. The default is 9999 records.
Return Value	A variable can be assigned to the command to hold one of the following return texts: `OK` The read was successful `ERROR IN SPECIFYING READ FUNCTION` Invalid parameter(s) specified `ERROR OPENING PDS DATASET` The *ddname* pointed to a non-partitioned dataset `STORAGE REQUEST FAILED` Temporary memory allocation failed `MEMBER REQUESTED NOT_FOUND` *member* was not found in the *ddname* dataset
REXX Example	This example opens a dataset member and reads JCL records. The records are written to an allocated internal reader to allow the JCL to be submitted as a job.

```
/*REXX*/
/* Allocate my JCL dataset to the "INPUT" DDname */
if EVORXALO('INPUT','USER.JCL.CNTL') = 'OK' then
  do
  /* Allocate the internal reader */
  if EVORXINT('JCLOUT') = 'OK' then
    do
    /* Read the PDS member and write it to the internal reader
*/
    if EVORXGET('IEFBR14','INPUT','PDSRECD') = 'OK' then
      do
      "EXECIO  0  DISKW  JCLOUT (OPEN"
      "EXECIO  *  DISKW  JCLOUT (STEM PDSRECD. FINIS"
      say 'IEFBR14 submitted'
      end
    /* Free the allocated dataset */
    x = EVORXFRE('JCLOUT')
    end
  x = EVORXFRE('INPUT')
  end
exit
```

EVORXINT - Allocate an Internal Reader (INTRDR)

Description	This function will allocate an INTRDR to the DD name specified.	
Syntax	`EVORXINT('`*`ddname`*`')` where:	
	`ddname`	A symbolic DD name (1 to 8 characters)
Return Value	A variable can be assigned to the command to hold one of the following return texts:	
	`OK`	Read of the dataset was successful
	`ERROR IN SPECIFYING READ FUNCTION`	An incorrect parameter was passed, either the *`variable`* name is too long, or the count exceeds the maximum size allowed
	`ERROR OPENING PDS DATASET`	The named *`ddname`* failed to open.
	`ERROR READING PDS DATASET`	There was a failure reading the *`ddname`*

	`STORAGE REQUEST FAILED`	The function was unable to allocate enough memory to hold the complete list of dataset members.
REXX Example	See the example for EVORXGET below to demonstrate the usage of EVORXINT.	

EVORXSYS - Display Users of Highest System Resources

Description	This function will return several lines of output explaining which address spaces are using the most mainframe system resources.
Syntax	`EVORXSYS('variable')` where: *`variable`* — A 1 to 8-character name for a compound variable that will be built to hold the returned messages. If not specified, the default stem variable "SYS" will be used. The stem ("SYS.0") value will hold the number of lines of output.
Return Value	A variable can be assigned to the command to hold one of these possible return texts: OK — The wait has returned successfully

REXX Example	`/*REXX*/`
	`x = EVORXSYS('DATA')`
	`if x = 'OK' then`
	`  do`
	`    do I = 1 to DATA.0`
	`      say DATA.I`
	`    end`
	`  end`
	`exit`

REXX Example Output	`EVORXSYS(1): -Highest CPU user at    .9% is RMFGAT`
	`EVORXSYS(2): -Max number of pages fixed below 16MB (000132 frames is BBOS001`
	`EVORXSYS(3): -Largest user of VSTOR (0006698 frames) is GRS`
	`EVORXSYS(4): -Highest user of total SRM service is ROYM`
	`EVORXSYS(5): -System CPU usage=  6%, Demand page rate=00000 pages/sec`

EVORXWAT - Wait/Sleep

Description	This function will suspend the processing in the REXX program for the specified number of seconds. If no parameter or a non-numeric parameter is specified, the default is five seconds. The maximum wait time is 999 seconds; any input larger than 999 is truncated to 999. While processing a Wait, all other Ironstream REXX automation processing is suspended, which should be taken into consideration when choosing a wait time.
Syntax	`EVORXWAT(`*`seconds`*`)` where: *`seconds`* Number of seconds to wait; whole numbers 0-999
Return Value	A variable can be assigned to the command to hold one of these possible return texts: OK The wait has returned successfully

REXX Example	`x = EVORXWAT(30)`
	`x = EVORXWAT()      /* Wait for five seconds */`

EVORXWTO - Issue a Write to Operator (WTO)

Description	This function will issue a WTO with the default route code to the z/OS operator console.
Syntax	`EVORXWTO('message'[,'{ROLL|NOROLL}'])` where: *`message`* The text message to be sent to the console, between 1 and 126 characters. A zero length text message or one that is greater than 126 characters will result in an error response. `ROLL|NOROLL` An optional parameter which, when set to "NOROLL", can set a WTO "critical message" descriptor flag to prevent the message from rolling off the console display. "ROLL" is the default. Use caution when using the NOROLL option, as overuse of this option can cause the master console to fill up and prevent any new messages from being displayed
Return Value	A variable can be assigned to the command to hold one of these possible return texts:

	`OK` — WTO processed successfully `ERROR IN SPECIFYING WTO FUNCTION` — error in the command parameters

REXX Example	`x = EVORXWTO('This WTO message will roll off the console')`
	`x = EVORXWTO('This message will not roll off the console','NOROLL')`

Troubleshooting

This chapter describes how to troubleshoot problems with Ironstream.

General Troubleshooting

Before you troubleshoot a particular problem that you run into when installing, configuring, or using Ironstream, you should verify that your Ironstream environment is correctly installed and configured.

Correct installation and configuration of ensures, among other things, that messages are processed correctly:

Ironstream UD Probe Server Components

ev390mcs (Windows)

The Master Configuration Server (or MCS) process – initiates two TCP connections to the agent using the Command port and Message port parameters in the Node configuration. It receives message data and commands or API requests from the agent. Message data can be system messages, performance data, resource status change messages, command responses, or responses to API requests.

There should be one ev390mcs process for each active IBM Z system. When there is no activity on the TCP connections, the ev390mcs process expects to receive a periodic heartbeat message from the agent (by default every 30 seconds, but this can be changed with the "HB" option on the TCP card in the mainframe agent's `SYSIN` parameter cards). If the heartbeat is not received, the connection is closed, and the connection process is re-initiated.

Different levels of program tracing is available by modifying the "HCI" value in the `parm/evodebug.parm` file.

Ironstream Mainframe Agent

The Ironstream agent runs as a started task on the IBM Z system. Task startup example follows:

```
S VP390.VP390
```

("VP390" being the started task name defined by the JCL procedure).

If the agent does not initialize after entering this command, then it is likely that not all installation/configuration steps have been completed. For the agent to start correctly, the following configuration file member must exist in the Ironstream `parmlib` dataset defined as input to the SYSIN DD statement:

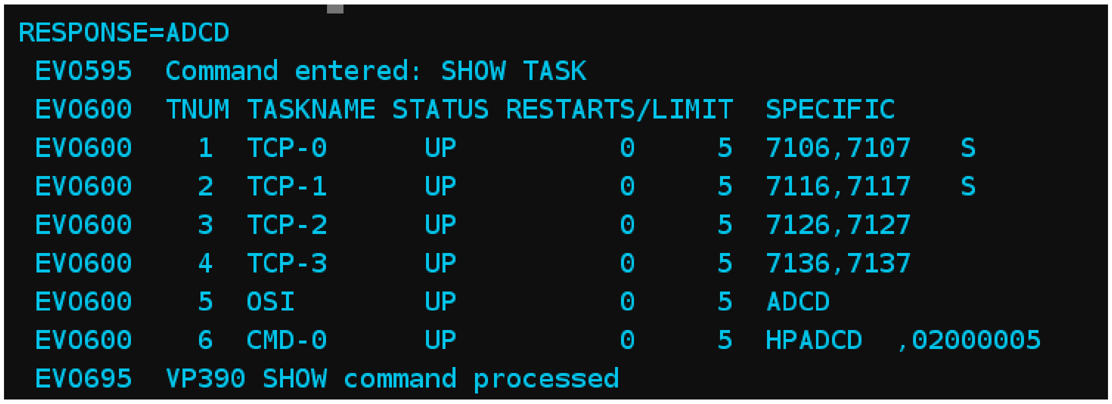
```
DDMPARM
```

The DDMPARM member contains initialization statements/parameters for the started task.

Error messages and initialization messages are issued during startup and can be found in the VP390 joblog and the syslog.

After the agent has been started the status of all subtasks can be displayed by issuing the following command:

```
F VP390,SHOW TASK
```



```
RESPONSE=ADCD
 EV0595  Command entered: SHOW TASK
 EV0600  TNUM TASKNAME STATUS RESTARTS/LIMIT  SPECIFIC
 EV0600    1  TCP-0      UP          0     5  7106,7107   S
 EV0600    2  TCP-1      UP          0     5  7116,7117   S
 EV0600    3  TCP-2      UP          0     5  7126,7127
 EV0600    4  TCP-3      UP          0     5  7136,7137
 EV0600    5  OSI        UP          0     5  ADCD
 EV0600    6  CMD-0      UP          0     5  HPADCD   ,02000005
 EV0695  VP390 SHOW command processed
```

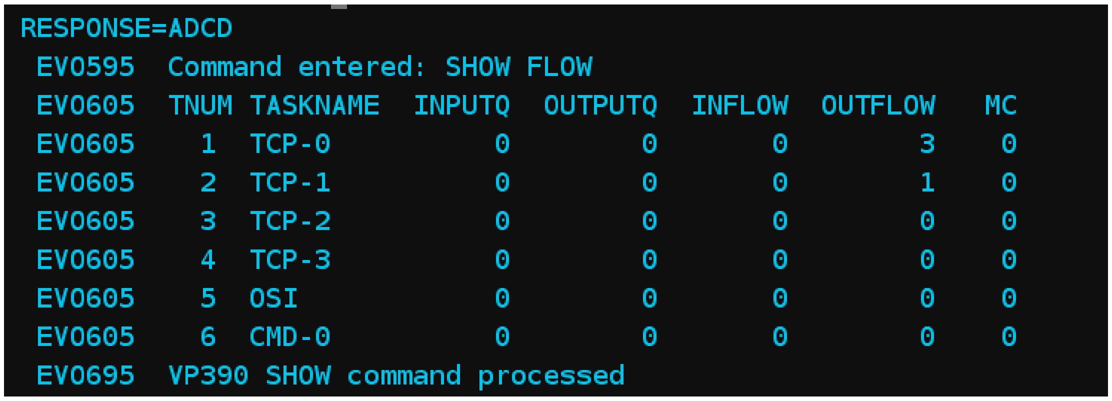
Check the command output to see that all required subtasks and all optional subtasks for your configuration are active. Also, note the status and compare to the expected status shown in the following list of subtasks.

Subtasks

- The **OSI** subtask will accept various requests from the Ironstream UD Probe server for information concerning the system, including CPU and job queue usage.
  - Optional Task, but required if any vp390hostcmd type 46 commands will be issued.
  - Status should be UP if configured.
- The **TCP** subtask is used to connect the agent to the Master Message Server and the Command Server processes on the Ironstream UCMDB Probe server. Multiple TCP subtasks are allowed.
  - Required Task
  - Status should be UP. A capital "S" appears at the end of the SHOW STATUS line for "TCP" if both port connections are established.
- The **CMD** subtask sets up the extended console used for MVS command inputs to VP390.
  - Optional Task, but required if any commands are going to be sent from the Ironstream UD Probe server back to the mainframe.
  - Status should be UP if configured.

To determine if messages/commands are flowing from or to the individual subtasks issue the following command:

```
F VP390,SHOW FLOW
```



```
RESPONSE=ADCD
 EV0595  Command entered: SHOW FLOW
 EV0605  TNUM TASKNAME  INPUTQ  OUTPUTQ  INFLOW  OUTFLOW   MC
 EV0605     1  TCP-0          0        0       0        3    0
 EV0605     2  TCP-1          0        0       0        1    0
 EV0605     3  TCP-2          0        0       0        0    0
 EV0605     4  TCP-3          0        0       0        0    0
 EV0605     5  OSI            0        0       0        0    0
 EV0605     6  CMD-0          0        0       0        0    0
 EV0695  VP390 SHOW command processed
```

The INFLOW and OUTFLOW columns can be monitored to check if messages are flowing into the agent or out of the agent subtasks. Note that the OUTFLOW number for the TCP subtask(s) will increment each time a heartbeat message is sent.

By observing the Inflow and Outflow values of the subtasks, you can detect if the agent is receiving messages or commands and forwarding the data over to the Ironstream UCMDB Probe server.

The "MC" column indicates how many memory allocations are outstanding for the subtask. It may show a positive number when the subtask is actively processing messages or commands, but should return to "0" when finished.

To further help in debugging a problem with the agent a DEBUG command is available to turn tracing on or off for the individual subtasks.

The command is entered as follows:

```
F VP390,DEBUG taskname level
```

where *level* can be 1 or 2 or 4 or any combination up to 7. For example, a 7 would include the results from 1 plus 2 plus 4.

- Trace level of 1 shows basic message flow in and out of the subtask.
- Trace level of 2 shows values of internal variables within subtask.
- Trace level of 4 includes hexadecimal dumping of control blocks.
- Trace level of 0 turns the trace off.

Note: The output of the trace data goes to the SYSPRINT DD statement. Leaving DEBUG tracing active for long periods of time could fill the output queue, especially when using the value of 7.

Example:

```
F VP390,DEBUG TCP-0 1
```

```
EVO595  Command entered: DEBUG TCP-0 1
EVO217  Debug for TCP-0 changed from 0 to 1
```

Example of data produced by the DEBUG command:

```
11/05 15:02:56 TCP-0 writing 14 bytes of type 25 info to 192.168.1.99
11/05 15:02:56 TCP-0 writing 14 bytes of type 27 info to 192.168.1.99
11/05 15:03:26 TCP-0 writing 14 bytes of type 27 info to 192.168.1.99
11/05 15:03:56 TCP-0 writing 14 bytes of type 27 info to 192.168.1.99
11/05 15:04:16 TCP-0 writing message to 192.168.1.99:
 SGMAIN ROYM 2006/07/19 11:58 F8 0 0 00000000 70 50 MAIN STORAGE GROUP
11/05 15:04:16 TCP-0 writing message to 192.168.1.99:
 VIO ROYM 2006/07/20 08:16 00 1 2000000 F3F3F9F0 0 0 VIO STORAGE GROUP
11/05 15:04:16 TCP-0 writing message to 192.168.1.99:
EOF
11/05 15:04:18 TCP-0 writing message to 192.168.1.99:
 EVWRK1 SGMAIN ROYM 2006/07/19 11:59 56664 00F3F640 2707 1043 930 00 01
11/05 15:04:18 TCP-0 writing message to 192.168.1.99:
 EVWRK2 SGMAIN ROYM 2006/07/19 11:59 56664 00F3F6D0 2707 756 364 00 01
```

Specific Troubleshooting

This section explains how to solve specific problems you may encounter when using Ironstream.

Failure of the Discovery Jobs

Symptom:

Discovery jobs are not able to complete command requests.

Troubleshooting Steps:

Use the Ironstream Configuration Tool to verify that the Message and Command processes are running for each IBM Z system.

Actions:

Restart the processes using the **Stop** and **Start** buttons on the Ironstream Configuration Tool.

TCP/IP connection problems

For proper operation, the client component must be able to connect to the IBM Z agent on both the message port and command port. After verifying that all required client component processes are running and all required agent subtasks are running on the IBM Z system, perform the following steps to verify TCP/IP communication between the server component and agent.

1. Check the status of the message port (default 6106) and command port (default 6107) on the Discovery Probe client using the command:

   ```
   netstat –a|grep 6106
   netstat –a|grep 6107
   ```

   If you have changed the default ports, then use the port numbers that are configured. If there is a connection from the Ironstream UD Probe server component, then the ports will show a state of “Established”.

2. Check the TCP port status on the IBM Z agent with the following `TSO NETSTAT` command:

```
NETSTAT CONN (PORT 6106 6107
```

If communication is working properly, then the state should have “Establsh”. Below is an example of a NETSTAT CONN output from a normal state:

```
EZZ2587I VP390    0017D2FA 192.168.1.117..6106    192.168.1.174..41245   Establsh
EZZ2587I VP390    0017D2FC 192.168.1.117..6107    192.168.1.174..41248   Establsh
```

3. Check the ev390mcs log on the Discovery Probe for an indication of a problem. Look for connection messages in the log.
    - In the connection message if you see a connection failure message with a result of “Connection Refused”, then the TCP stack on the Ironstream UD Probe server is getting a result back that indicates the mainframe port is not in a “Listen” state.
    - If the mainframe ports are in “Listen” state and are still seeing a “Connection Refused”, then check to see if there is a firewall in place between the Ironstream UD Probe server and the IBM Z system, and if so make sure it has rules to allow bi-directional communication between the Discovery Probe and IBM Z system.
    - If you are seeing a connection failure with the result of “Connection timed out”, this indicates a network routing error between the Ironstream UD Probe server and the IBM Z system.
4. If you see that the server component makes a successful connection, but the connection is closed 30 seconds later and then reconnects immediately, this indicates a port conflict. In this case, change the default ports of 6106 and 6107 to a different range, for example, 6116 and 6117.

    Please note the change must be made on:

    - Ironstream UD Probe server in the EVOMF_HCI_AGENT_PORT and EVOMF_CMDS_AGENT_PORT parameters.
    - The "TCP" parameter card in the mainframe SYSIN parameters).

    You will need to restart both the client component processes and agent for the change to take effect.

Appendix A: z/OS Console Commands

This appendix explains Ironstream z/OS console commands that enable operators to display and change maintenance information about the present mainframe job. Commands are sent from a z/OS console to the Ironstream job using the `MODIFY` command.

If the Ironstream job name is `VP390`, the syntax for a console command is:

`MODIFY VP390,`*`command`*

This appendix explains the following types of z/OS commands:

- SHOW commands
- Subtask control commands
- FILTER commands
- SUPPRESS commands
- PERF commands

SHOW Commands

SHOW commands display the requested information in a formatted table.

SHOW TASK

Displays each of the defined subtask, their status, number of times the subtask was restarted, maximum number of automatic restart attempts for the subtask, and any unique information for the subtask.

Subtask Status

UP	Subtask is active and can accept messages.
DOWN	Subtask is down and is not restarting.
DOWNR	Subtask is down but is restarted after a delay.
INIT	Subtask is initializing.
QUIES	Subtask is in a quiescent state, cleaning up outstanding allocated memory before going into the DOWN or DOWNR state.

Example

`MODIFY VP390,SHOW TASK`

```
EVO595 Command entered: SHOW TASK
EVO600 TNUM TASKNAME STATUS RESTARTS/LIMIT  SPECIFIC
EVO600    6  TCP-0      UP          0   100  6106,6107   S
EVO600    7  OSI        UP          1   100  BLUEBOX
EVO600    9  CMD-0      UP          0     5  EVOCONSL,01000002
EVO695  VP390 SHOW command processed
```

SHOW ADDR

Displays the memory address of each defined subtask internal header control block, subtask control block, z/OS Task Control Block, and CPU usage in milliseconds for each subtask. This information is useful if you anticipate making an address space dump.

Parameters

None

Example

`MODIFY VP390,SHOW ADDR`

```
EVO595  Command entered: SHOW ADDR
EVO603  TNUM TASKNAME  ADDRESS    HEADER     TCB      CPU USE
EVO603    0  MAINTASK  00000000  05A350C8  00000000  52.3643
EVO603    1  TCP-0     05A1C014  05A7B808  008CDE88  10.6746
EVO603    2  OSI       05A1C068  05A837C8  008C5C58  3.9319
EVO603    3  CMD-0     05A1C0BC  05A8B788  008BDA28  8.2409
EVO695  VP390 SHOW command processed
```

SHOW VERSION

Displays the version of Ironstream running and the compile date of each subtask.

Parameters

None

Example

`MODIFY VP390,SHOW VERSION`

```
EVO595  Command entered: SHOW VERSION
EVO607  EView/390z V7.3 Copyright 2020 EView Technology, Inc.
EVO608  TASKNAME     DATE         TIME
EVO608  MAINTASK  Feb 15 2010   06:20:00
EVO608  TCP-0     Feb 15 2010   06:20:00
EVO608  OSI       Feb 15 2010   06:20:00
EVO608  CMD-0     Feb 15 2010   06:20:00
EVO695  VP390 SHOW command processed
```

SHOW FLOW

Displays the number of messages for each subtask on the input and output queues, the total number of messages that flowed in and out of the subtask, and the number of memory allocations currently outstanding.

Parameters

None

Example

`MODIFY VP390,SHOW FLOW`

```
EVO595  Command entered: SHOW FLOW
EVO605  TNUM TASKNAME  INPUTQ  OUTPUTQ  INFLOW  OUTFLOW   MC
EVO605    5  TCP-0          0        0      11      249    0
EVO605    6  TCP-1          0        0       0        0    0
EVO605    8  CMD-0          0        0       0        0    0
EVO695  VP/390 SHOW command processed
```

SHOW SUPPRESS

Displays a list of VP390 message Ids that were suppressed from printing using the SUPPRESS SYSIN command or the SUPPRESS Modify command.

Parameters

None

Example

`MODIFY VP390,SHOW SUPPRESS`

```
EVO595  Command Entered: SHOW SUPPRESS
EVO615  Suppressed message IDs:
EVO615  002, 902, 905
```

Subtask Control Commands

Subtask control commands allow you to manually control the status of a subtask. Ironstream subtasks start automatically when the job is started, and the subtasks restart automatically if brought down by some anomaly.

Note: For more information on automatic subtask restarts, see the description of the DELAY and RESTART input parameter cards in the *Ironstream for Micro Focus Universal Discovery for IBM Z Installation Guide*.

INIT

Activates a defined subtask that is in a `DOWN` state. This command can also be used when a subtask is in the `DOWNR` state to skip the rest of the timed delay and force the re-initialization to continue immediately. The INIT command can only activate tasks that are listed in the `SHOW TASK` table.

Parameters

`subtaskname`

Example

`MODIFY VP390,INIT TCP-0`

```
EVO595  Command entered: INIT TCP-0
EVO002  TCP subtask initialized for 6106,6107
```

KILL

Forces the termination of a defined subtask. When a subtask is terminated with this command, it does not attempt any automatic restarts. The command resets the count of number of automatic restarts that are attempted. The command can also be used to stop a subtask in the `DOWNR` state from attempting any more restarts.

Parameters

`Subtaskname`

Example

`MODIFY VP390,KILL SPO-1`

```
EVO595  Command entered: KILL SPO-1
EVO902  SPO-1 subtask terminated, RC = 0
```

TERM

Stops all subtask and then stops the main task, terminating the VP390 job. This command is equivalent to the z/OS `STOP` command.

Parameters

None

Example

`MODIFY VP390,TERM`

```
EVO595  Command entered: TERM
EVO690  VP390 STOP Command accepted
EVO901  Stopping subtask #1: TCP-0
EVO901  Stopping subtask #2: OSI
EVO901  Stopping subtask #3: CMD-0
EVO695  VP390 STOP command processed
EVO902  OSI subtask terminated, RC = 0
EVO902  CMD-0 subtask terminated, RC = 0
EVO902  TCP-0 subtask terminated, RC = 0
EVO904  All VP390 subtasks complete
IEF404I VP390 - ENDED - TIME=17.30.08
$HASP395 VP390 ENDED
```

FILTER Commands

The FILTER commands listed below make use of the Ironstream agent feature that restricts incoming z/OS commands from the server.

SHOW FILTER

Displays all commands in the command filter table. If the table has any entries, then incoming commands are checked against the table's regular expressions, and only those commands that have a match in the table will be executed. If the command table has no entries, then all commands are executed.

Parameters

None

Example

`MODIFY VP390,SHOW FILTER`

```
EVO595  Command entered: SHOW FILTER
```

```
EVO612  No message filters defined
EVO612  No alert filters defined
EVO609  Command filters:
EVO609  ¬D IPLINFO$
EVO609  ¬D NET,MAJNODES$
EVO609  ¬V NET,ACT,ID=*.
EVO695  VP390 SHOW command processed
```

FILTER ADD

Adds a command to the command filter table. By default, the command table holds up to 200 command expressions. Command expressions are in the format of Unix-style regular expressions. (Note that some terminal emulators may not be able to enter certain regular expression characters, such as the caret or square brackets. In these cases, add the filter entry as a FILTER card in Ironstream's SYSIN parameters, and restart the Ironstream job.)

Syntax

```
FILTER ADD CMD regularexpression
```

Example

* Permit the console to issue a Display Time command:

```
MODIFY VP390,FILTER ADD CMD D T$
EVO595  Command entered: FILTER ADD CMD D T$
EVO610  Command filter D T$ added
```

FILTER DEL

Deletes a command from the command filter table. Specifying ALL deletes all filters from the command table.

Syntax

```
FILTER DEL CMD regularexpression

FILTER DEL ALL
```

Examples

```
MODIFY VP390,FILTER DEL CMD D T$
EVO595  Command entered: FILTER DEL CMD D T$
EVO610  Command filter  deleted
```

```
MODIFY VP390,FILTER DEL ALL
EVO595  Command entered: FILTER DEL ALL
EVO613  All message and alert and command filters deleted
```

Appendix B: VP390 Mainframe Messages

This appendix describes all messages generated by the Ironstream job running on the mainframe. The default name for the mainframe task is “VP390”.

EVO002 type subtask initialized for feature

Message Variables

`type`	Type of subtask
`feature`	A specific attribute that this subtask is initialized for:

Subtask	**Attribute Description**
CMD	Extended MCS console name
NOMATCH	Dataset name to be written to
MVS	Extended MCS console name
OSI	z/OS system name
OPC	Initialized TCP/IP Port number
PERF	z/OS system name
PPI	"PPI"
PPO	VTAM resource contacted
RMA	DD name of REXX programs' dataset
SEC	Defined security application name
SPO	VTAM resource contacted
TCP	Initialized TCP/IP port numbers

Message Description
The VP390 subtask is successfully initialized. This message will be issued for each of the defined subtasks of the VP390 main task.

System Action
Processing continues.

User Action
None.

EVO007 Invalid size of parameter 'parm' on line number

Message Variables

`parm`	Character string in `SYSIN` line
`number`	Line number of `SYSIN`

Message Description
A parameter or option for a `SYSIN` line was found to be of an invalid length.

System Action
The invalid card is skipped. Processing continues with the next `SYSIN` card.

User Action
Correct the input card on the given line number of `SYSIN`. Valid values for `SYSIN` cards are listed in the *Ironstream for Micro Focus Universal Discovery for IBM Z Installation Guide*. If a system symbol was used in the line, verify that the combination of the symbol's length and any other character concatenations do not result in an invalid size for the parameter.

EVO008 Invalid input parameter card on line number

Message Variables

`number`	Line number of `SYSIN`

Message Description
The VP390 job read a line from `SYSIN` that it did not understand.

System Action
The invalid card is skipped. Processing continues with the next `SYSIN` card.

User Action
Correct the input card on the given line number of SYSIN. Valid syntax for SYSIN cards are listed in the *Ironstream for Micro Focus Universal Discovery for IBM Z Installation Guide*. All other lines must begin with an asterisk (*) to denote a comment line.

EVO009 Duplicate subtask card on line number ignored

Message Variables

`subtask`	Type of subtask
`number`	Line number of `SYSIN`

Message Description
The VP390 job read a definition card from `SYSIN` for a subtask that has already been defined.

System Action
The invalid card is skipped. Processing continues with the next `SYSIN` card.

User Action
Correct or remove the input card on the given line number of SYSIN. For names of input parameter cards that may be defined multiple times, see the Ir*onstream for Micro Focus Universal Discovery for IBM Z Installation Guide*.

EVO010 Maximum number of subtask cards reached; ignoring line number

Message Variables

`subtask`	Type of subtask, or `subtask`
`number`	Line number of `SYSIN`

Message Description
The VP390 job has reached the maximum number of subtasks of the type named. The definition card on the named line is not processed. If *`subtask`* is "`subtask`", VP390 has reached the maximum number of total subtasks that can be defined, and all `SYSIN` parameter cards from the current line number forward are ignored.

System Action
The parameter cards are skipped and processing continues.

User Action
Decrease the number of `SYSIN` parameter cards of the type named.

EVO011 Maximum number of type filter entries reached; new entry ignored

Message Variables

`type`	Type of filter entry

Message Description
The VP390 job has reached the maximum number of filter entries allowed. By default, VP390 will accept up to 2000 message ID entries and 200 command filter entries.

System Action
The new filter entry is discarded and processing continues.

User Action
Decrease the number of filter entries, possibly by combining multiple entries using wildcard characters, or use the FILTERTABLE parameter card to increase the size of the filter table.

EVO012 Unable to allocate name filter table

Message Variables

`name`	"message" or "alert"

Message Description
A memory allocation failure has occurred while either (1) attempting to allocate the filter table, or (2) attempting to add attributes (jobname, jobid) to an existing filter table entry.

System Action
The new filter entry is discarded and processing continues.

User Action
Allocate more memory for the VP390 job in the startup JCL. If this message appears during startup, use the FILTERTABLE parameter card to decrease the initial allocation size of the message filter table.

EVO018 VTAM ACB generation for subtask acb failed, RC = rcnumber

Message Variables

`subtask`	Type of subtask
`acb`	Name of failing ACB
`rcnumber`	Return code from the Get VTAM ACB routine

Message Description
An attempt by an initializing subtask to get a VTAM ACB failed.

System Action
The VP390 subtask terminates with a condition code 8.

User Action
Verify that the ACB is available. Use the INIT command to restart the subtask.

EVO019 VTAM subtask open for acb failed, RC = rcnumber, error = enumber

Message Variables

`subtask`	Type of subtask
`acb`	Name of failing ACB
`rcnumber`	Return code from the Open VTAM ACB routine
`enumber`	Error code within ACB

Message Description
An attempt by an initializing subtask to open a VTAM ACB failed.

System Action
The VP390 subtask terminates with a condition code 8.

User Action
Verify that the ACB name acb is correctly defined. If `rcnumber` = 8, then the subtask may be restarted using the INIT command. If rcnumber = 12, then there is a serious VTAM error which will not allow a re-issue of the ACB Open command; check the status of VTAM and recycle the VP390 job. If `enumber` = 88, then resource `acb` is already in use by another program. (Remember that the PPO subtask should not be used if NetView is running.) If e`number` = 36, verify that `acb` does not have a password requirement or other RACF restriction. If `enumber` = 90, verify that the VTAMLST APPL entry for `acb` is coded correctly and the APPL is active. For descriptions of other error codes, see the section for the OPEN macroinstruction in the IBM manual VTAM Programming.

EVO020 subtask is currently in use

Message Variables

subtask	Type of subtask

Message Description
This message follows immediately after the EVO019 message if an exclusive subtask ACB is already in use by another program.

System Action
The VP390 subtask terminates with a condition code 8.

User Action
Verify that the ACB is not taken by another program on the mainframe, such as NetView/390 or SOLVE:NETMASTER.

EVO021 Unsolicited msgtype data is unavailable

Message Variables

msgtype	Type of message

Message Description
This message follows the `EVO019` message to alert you that the VP390 job is not able to receive unsolicited data because it was unable to access an ACB.

System Action
The VP390 subtask terminates with a condition code 8.

User Action
Correct the problem identified by the EVO019 message, then restart the subtask.

EVO026 Unexpected subtask return code, RC = rcnumber

Message Variables

subtask	Type of subtask
rcnumber	Return code from Receive routine

Message Description
The subtask Receive routine received an unexpected return code while attempting to receive messages.

System Action
The VP390 subtask terminates with a condition code 9.

User Action
Check the mainframe job output log for additional messages. Use the INIT command to restart the subtask.

EVO033 VP390 COMMAND = command

Message Variables

command	Command text

Message Description
The command issued through the VP390 job is logged to `SYSLOG`.

System Action
Processing continues.

User Action
None.

EVO034 Initialization of SPO name failed in reqtype processing, RC1 = addr RC2 = size

Message Variables

name	Name of the SPO subtask
reqtype	Type of request being processed
addr	Returned address from Get RPL routine
size	Returned size from Get RPL routine

Message Description
The SPO subtask failed calling the VTAM RPL routine.

System Action
The VP390 SPO subtask terminates with a condition code 13.

User Action
Use the INIT command to recover subtask.

EVO035 SPO Warning: Failure retrieving command responses, max retries reached.

Message Variables
None.

Message Description
The VP390 Secondary Program Operator interface subtask encountered a failure while attempting to retrieve the command responses from an issued VTAM command. Not all responses were retrieved.

System Action
Processing continues.

User Action
Re-issue the VTAM command. If the proper responses are still not returned, contact Precisely support.

EVO036 SPO command queue depth exceeded maximum

Message Variables
None.

Message Description
A VTAM SPO command could not be placed on the VP390 queue of waiting SPO commands because that queue has reached its maximum size.

System Action
The command is discarded.

User Action
Re-issue the VTAM command. If this message appears frequently, consider defining additional SPO subtasks to handle the load (the VP390 job allows up to ten SPO subtasks to be defined in the SYSIN cards).

EVO038 subtask command support unavailable

Message Variables

subtask	name of unavailable subtask

Message Description
The mainframe task is not able to process a command because the necessary subtask is not running.

System Action
The command is discarded.

User Action
Use the SHOW TASK console command (see Appendix A) to check the status of the VP390 subtasks. If *subtask* is in the list of subtasks but does not have an "UP" status, use the INIT command to restart the subtask. If *subtask* is not in the list of subtasks, then add it to the `SYSIN` deck and restart the VP390 job.

EVO039 Unable to route message (type=type)

Message Variables

type	Invalid message type

Message Description
The VP390 job was unable to route an incoming message to any of its subtasks because the message type was unrecognized.

System Action
The invalid message is dumped to `SYSPRINT` immediately after this message.

User Action
Capture the job's `SYSPRINT` information and contact Precisely support.

EVO040 Authorization lock timeout for subtask

Message Variables

subtask	Ironstream subtask

Message Description
The VP390 job was unable to secure a semaphore lock in 20 seconds before issuing and APF-authorized command. The lock is requested to ensure that an S047 abend is avoided.

System Action
The system command is executed, but an S047 abend may be generated if the job is not reset to allow APF authorized commands

User Action
Check if any outstanding commands from the UD Probe server to the agent are unresolved to determine which command did not release the lock.

EVO091 PPI initialization failed, step = stepnum RC = rcnumber

Message Variables

stepnum	Initialization step that failed:	
	1	SSI not running
	2	Attempt to get ASCB value failed
	3	Attempt to register receiver failed
rcnumber	Return code from call to CNMNETV	

Message Description
An attempt by the PPI subtask to access the CNMNETV module failed.

System Action
The PPI subtask terminates with a condition code 6.

User Action
If *stepnum* = 1, check the status of the SSI address space. If *stepnum* = 2, use the NetView DISPLAY PPI modiFy command to verify that the NetView program-to-program interface is active. If *stepnum* = 3, verify that no other application is attached to the NetView/390 or NETMASTER PPI.

EVO095 VP390 PPI buffer size error, RC = rcnumber

Message Variables

rcnumber	Return code from PPI call

Message Description
A Receive request for the PPI failed because the allocated buffer size was not large enough to hold the incoming data.

System Action
The VP390 PPI subtask terminates with a condition code 31.

User Action
Use the `INIT` command to restart the subtask.

EVO096 VP390 PPI interface failed, ID = requestid, RC = rcnumber

Message Variables

`requestid`	ID of task request
`rcnumber`	Return code from PPI call

Message Description
A Receive request for the `PPI` failed.

System Action
The VP390 PPI subtask terminates with a condition code 11.

User Action
For explanations of return codes, see the *TME 10 NetView for OS/390 Application Programmer's Guide*. If *`requestid`* = 22 and *`rcnumber`* = 25, then add "`BUFLEN=40`" to the `PPI` card in `SYSIN`.

EVO119 count messages queued on subtask. Command rejected: cmd

Message Variables

`count`	Number of messages
`subtask`	Subtask name
`cmd`	Command entered

Message Description
Subtask *`subtask`* does not process the command issued from the workstation because there is a backlog of *`count`* messages waiting to be sent to the workstation.

System Action
The command *`cmd`* is discarded. Processing continues on the remaining messages in the subtask queue.

User Action
Wait until the existing backlog of messages is processed, then re-issue the command. Use the mainframe VP390 modiFy command SHOW `TASK` to view the number of messages in the Output Queue of the subtask.

EVO121 MVS console name could not obtain a migration ID

Message Variables

`name`	Name of console to be defined

Message Description
The MVS console being defined requested a one-byte migration ID, but the console initialization routine was unable to provide one.

System Action
Initialization of the console continues.

User Action
None.

EVO122 type console name initialization failed, RC = rc,reas

Message Variables

`type`	Subtask type ("MVS" or "CMD")
`name`	Name of console to be defined
`rc`	Return code from initialization routine, in hexadecimal
`reas`	Reason code from initialization routine, in hexadecimal

Message Description
The initialization of the console failed.

System Action
The VP390 subtask terminates with a condition code 8.

User Action
Verify that all the parameters on the `type` SYSIN card conform to the syntax rules. If `rc`=4, then a console name is already running. If you are running multiple Ironstream agents on mainframes or LPARs in a sysplex, then one mainframe image may be able to see another's consoles. Use a unique name for each agent's MVS and CMD card in its `SYSIN` deck. If `rc` =10, verify that `name` conforms to the rules for console names. If `rc` =C, the VP390 task does not have the necessary READ access to the OPERCMDS resource name MVS.MCSOPER.`name`. Enter the RACF command to allow this READ access for the user ID under which the VP390 job is running.

EVO126 Unable to open MSGCATLG message file

Message Variables
None.

Message Description
The VP390 main task could not find or open the messages file, which is identified by the MSGCATLG DD card in the VP390 startup JCL.

System Action
The VP390 task terminates.

User Action
Verify that the MSGCATLG DD card is defined in the VP390 started task JCL and points to a readable message file. Restart the VP390 job.

EVO127 Too many messages in MSGCATLG message file

Message Variables
None.

Message Description
The VP390 messages file, identified by the MSGCATLG DD card in the VP390 startup JCL, contained more lines than expected for a valid messages file.

System Action
The VP390 task terminates.

User Action
Verify that the MSGCATLG file does not contain extra non-blank lines which could be misinterpreted for message lines. Comment lines beginning with an asterisk and blank lines in the file are ignored. Restart the VP390 job.

EVO128 Unable to find message ID msg in MSGCATLG file

Message Variables

msg	Message ID to be written

Message Description
VP390 attempted to issue a message with the message ID `msg` but could not find this message ID in the MSGCATLG file.

System Action
Processing continues.

User Action
Verify that the file identified by the MSGCATLG DD in the VP390 startup JCL contains message text for the ID *msg*. In the MSGCATLG file, message IDs must start in the first column of each line. Restart the VP390 job to re-read the messages file.

EVO130 Unrecognized command option: code

Message Variables

code	Option number

Message Description
The `ev390hostcmd` utility on the OVOW server sent a type 46 command with an option code that the mainframe OSINFO subtask did not recognize.

System Action
Processing continues.

User Action
Consult the earlier section for valid options for OSINFO system information and correct syntax of the `ev390hostcmd` utility.

EVO131 Query failed, error code = code

Message Variables

code	Error code

Message Description
The `ev390hostcmd` utility on the OVOW server sent a type 46 command requesting information that could not be supplied by the OSINFO subtask on the mainframe.

System Action
Processing continues.

User Action
The code can have different meanings depending on the type 46 option that was requested. Identify what command request is being issued and contact Precisely support.

EVO132 Query returned no lines

Message Variables
None.

Message Description
The `ev390hostcmd` utility on the OVOW server sent a type 46 command that returned no output. This can be caused by improper syntax on the 46 command, or by specifying a non-existent task name or DASD volume.

System Action
Processing continues.

User Action
Check the syntax and parameters of the `ev390hostcmd` which was sent to the mainframe.

EVO133 Unable to collect queue queue data: error accessing source, rc=code

Message Variables

queue	Queue name to gather information from: "INPUT", "OUTPUT", or "HELD"
source	Resource that could not be accessed: "ISFIN", "ISFOUT", or "SDSF"
code	Return code

Message Description
The `ev390hostcmd` utility on the OVOW server sent a type 46 command requesting information from one of the JES2 queues that could not be supplied.

System Action
Processing continues.

User Action
If *`source`* is "ISFIN" or "ISFOUT", verify that the `ISFIN` and `ISFOUT DD` cards are correctly defined in the VP390 startup JCL. The code can have different meanings depending on the type 46 option that was requested. Identify what command request is being issued, and contact Precisely support.

EVO134 Error in function call for value: code

Message Variables

function	Name of program function where error occurred
value	The user-requested parameter that was being queried
code	Return code from the *function*

Message Description
The `ev390hostcmd` utility sent a type 46 command requesting information that could not be supplied, usually due non-availability of certain information from certain resources.

System Action
Processing continues.

User Action
Verify that the *value* is valid for the requested output. For example, the type 42 option (program name and PARM value) will not succeed if the "CONSOLE" address space is used for the *value* . The *code* can have different meanings depending on the type 46 option that was requested. Identify what command request is being issued and contact Precisely support.

EVO135 Dataset read error: text

Message Variables

text	Error description

Message Description
An error occurred while attempting to read a file or dataset for the ev390hostcmd 46 type 10 command. (See "

10 Dataset Display".) If *text* is "Unexpected file read error", then the error occurred when attempting to read an HFS file. If *text* is "Unexpected dataset read error", then the error occurred when attempting to read a sequential or partitioned dataset.

System Action
The file is closed and the command is canceled.

User Action
Verify that the named file or dataset exists and the appropriate authority exists to read it.

EVO137 Rexx command error: error

Message Variables

error	Error code or IRX error number

Message Description
An error occurred while attempting to execute a REXX program using the ev390hostcmd 46 type 12 command. (See "12 Execute REXX Program".) The values of *error* are:

Error Value	Description

could not initialize REXX	Ironstream could not initialize the REXX interface module IRXEXEC. Verify that this module is available in the mainframe LPALIST.
invalid script member name	The member name of the REXX program is has a length of 0 or is greater than 8. Verify the program name given in the first parameter of the type 12 option.
20	The member name of the REXX program was not found in the SYSEXEC library. Verify that the name is spelled correctly and the SYSEXEC DD is pointing to the desired partitioned dataset.
32	An error occurred in the call to the REXX IRXEXEC interface. Record the syntax of the command being sent and contact Precisely support.
100	A system abend occurred during the execution of the REXX program. Check the mainframe syslog for information.
104	A user abend occurred during the execution of the REXX program. Check the mainframe syslog for information.
IRX*xxxx*I	A syntax error was detected in the REXX program. This IRX message is the message ID of the detected error. Consult the IBM REXX documentation for the description of this error. Additional information may appear in the mainframe syslog.

System Action
The command is canceled.

User Action
Verify that the program call has correct syntax and parameters. For IRXnnnnI message IDs, see the system console for the full message text, and refer to the IBM "TSO/E Messages" manual for description.

EVO140 Maximum lines of output exceeded (linecount)

Message Variables

`linecount`	Number of lines printed

Message Description
While attempting to read a file or dataset using the ev390hostcmd 46 type 10 command, the maximum number of output lines was exceeded. (The default maximum is 5000 lines.)

System Action
The dataset/file is closed.

User Action
If more lines of output are desired, use the "`maxsize=`*`n`*" option at the end of the ev390hostcmd to specify a larger maximum. See "10 .Dataset Display" for syntax.

EVO141 Output size exceeded (bytecount)

Message Variables

`bytecount`	Number of bytes allocated for output

Message Description
While attempting to run an MQ query using the ev390hostcmd 46 type 50 command, the maximum number of bytes was exceeded. (The default maximum is 64000 bytes.)

System Action
The command output is discarded.

User Action
Use the "`maxsize=`*`n`*" option at the end of the ev390hostcmd to specify a larger maximum. See "50 Execute MQ Series Command" for syntax.

EVO144 TCP/IP query failed: reason

Message Variables

`reason`	Reason for command failure

Message Description
An error was encountered while attempting to issue an ev390hostcmd 46 command to gather TCP/IP stack information, either because of the syntax usage of the command or a response from a system API.

System Action
The error message is returned to the calling server process.

User Action
Validate the syntax of the command issued. If the *`reason`* indicates a system error, check the mainframe syslog and/or the JES output of the Ironstream job for more information.

EVO150 TCP/IP communications: function for workstation component agent failed with errno value

Message Variables

function	Failing communication function
component	Workstation component that detected the failure
value	Integer error value

Message Description
A TCP/IP communications error occurred. The error could have occurred while TCP/IP communication was being established or while a message was sent or received by the mainframe or specified agent.

System Action
The VP390 TCP subtask terminates with a condition code 1.

User Action
Verify the availability of TCP/IP communications between the workstation and the mainframe, and verify the mainframe TCPIP job's high-level qualifier is specified correctly on the TCP card in the VP390 SYSIN deck. Use the INIT command to recover the TCP subtask, or recycle the VP390 job if the `SYSIN` needs modification.

EVO151 VP390 failure in communication to TCP/IP

Message Variables
None.

Message Description
The VP390 job received an error while attempting to receive data from a TCP/IP socket or ECB.

System Action
The TCP subtask terminates.

User Action
Use the `INIT` command to recover the subtask.

EVO152 Default TCP/IP function failed

Message Variables

function	Failing communication function

Message Description
The setup of a default TCP/IP environment failed while performing *function*.

System Action
Processing continues, but initialization of subsequent `TCP` subtasks may fail.

User Action
Verify the mainframe TCPIP job's high-level qualifier is specified correctly on the TCP card in the VP390 `SYSIN` deck. Recycle the VP390 job if the `SYSIN` needs modification.

EVO153 Message length exceeds send buffer allocation

Message Variables
None.

Message Description
The TCP subtask could not send out a block of data because it was longer than the standard VP390 data buffer could hold.

System Action
The message is discarded.

User Action
Note the system message and alert activity at the time this message was issued, and contact Precisely support.

EVO154 server Server connection lost on port number

Message Variables

`server`	Ironstream server process on the UD Probe server
`number`	Port number

Message Description
The mainframe agent lost its connection to the UCMDB Probe server.

System Action
The port number is reset to allow re-connections. If message buffering is active, mainframe messages will be written to the buffer file until the connection to the UD Probe server is re-established.

User Action
Use the Ironstream Configuration Tool on the UD Probe server to verify the Ironstream processes are running.

EVO155 server Server connection established on port number

Message Variables

`server`	Ironstream server process on the UD Probe server
`number`	Port number

Message Description
The mainframe agent has made a connection to the server process on the UCMDB Probe server.

System Action
Processing continues.

User Action
None.

EVO156 Invalid connection attempt from different servers

Message Variables
None.

Message Description
Two Ironstream UD Probe server components attempted to connect to the agent's TCP/IP ports, with one server taking the Message port and the other taking the Command port. The Ironstream design requires that both ports communicate with server processes on the same Ironstream UCMDB Probe server.

System Action
The TCP subtask terminates both TCP connections and resets. If the server conflict continues for more than the number of restarts allowed for the TCP subtask, then the TCP subtask will shut down completely, requiring a manual restart using the INIT console command, or restarting the mainframe job.

User Action
The mainframe task's SYSPRINT will give a detailed message identifying the source of the two server connection attempts. Terminate the Ironstream processes on one of the servers. If multiple Ironstream clients are desired to connect to the same mainframe agent, then add another TCP subtask card to the SYSIN deck with different port numbers, and refer to that new set of port numbers in the EVOMF_HCI_AGENT_PORT and EVOMF_CMDS_AGENT_PORT fields in the mainframe node configuration file on the Ironstream UD Probe server.

EVO157 Unable to convert segment text due to NLS error

Message Variables

`segment`	Portion of the message that failed

Message Description
A failure occurred either when converting a message from the local codeset to UTF-8 for delivery to the OM server or when converting an incoming command from UTF-8 to the local codeset.

System Action
The message/command is dropped.

User Action
Record the hexadecimal message dump that appears in the SYSPRINT and contact Precisely support.

EVO158 Invalid connection attempt from address:rport to port lport

Message Variables

`address`	Remote IP address
`rport`	Remote port
`lport`	Local port

Message Description
A connection was attempted to a port from an address that was restricted by the SERVERIP option on the TCP subtask definition card.

System Action
The connection is closed and the `lport` returns to a Listen status. If more than five distinct invalid *address*es are received before a valid connection is made, the TCP subtask is stopped and will require a manual subtask restart command to be entered.

EVO160 Console command return code = rcnumber

Message Variables

rcnumber	Return code from command Send subroutine

Message Description
An MVS command request completed with a non-zero return code.

System Action
Processing continues.

User Action
If expected command response is not received, record the return code and contact Precisely support

EVO161 No match for console command in command filter table

Message Variables
None.

Message Description
A z/OS command issued to the CMD console failed to match any of the entries in the command filter table.

System Action
The command is not executed.

User Action
Add appropriate FILTER CMD entries (either as cards in the SYSIN deck, or dynamically using the “F VP390,FILTER ADD CMD” command) to allow the desired command to be executed.

EVO162 No valid DD names for message logging subtask

Message Variables
None.

Message Description
No valid log file DD names were specified for the NOMATCH subtask.

System Action
The `NOMATCH` subtask is terminated.

User Action
Add appropriate DD names to the `NOMATCH` line in SYSIN, and verify that the DD names are defined in the VP390 startup JCL. Recycle the VP390 job.

EVO163 Unable to open message logging file ddname

Message Variables

ddname	DD name of the file

Message Description
The NOMATCH subtask was unable to open the logging dataset *ddname* named on the SYSIN card for the NOMATCH initialization.

System Action
The NOMATCH subtask attempts to open the next dataset in the list.

User Action
Verify that the DD name given on the SYSIN card has a matching DD card in the VP390 startup JCL. Verify that the dataset named for that DD name is defined with the DCB values stated in the *Ironstream for Micro Focus Universal Discovery for IBM Z Installation Guide*.

EVO164 Message logging is closing dataset

Message Variables

dataset	Log dataset name. If the log is a PDS member, the member name will be appended to the dataset name in parentheses.

Message Description
The NOMATCH subtask is closing the dataset logging dataset, either because of subtask termination or because an attempt to write to the dataset failed (usually because the dataset has been filled.)

System Action
If the dataset closing was due to a write failure, the NOMATCH subtask attempts to open the next dataset in its list of defined DDs.

User Action
None.

EVO165 Message logging is wrapping to the first file

Message Variables
None.

Message Description
The NOMATCH subtask has reached the end of its list of valid logfile DD names.

System Action
The NOMATCH subtask wraps back to re-open the first DD in the list. The existing data in that logfile will be purged and overwritten.

User Action
None.

EVO170 Unable to open message buffering file ddname

Message Variables

`ddname`	DD name of the file

Message Description
The message buffering facility was unable to open the dataset `ddname` for buffering messages while the TCP/IP connection to the OVOW server is down.

System Action
No message buffering will occur while the TCP/IP connection is down.

User Action
Verify that the DD name on the TCP SYSIN card for message buffering has a matching DD card in the VP390 startup JCL. Verify that the dataset named for that DD name is defined with the DCB values stated in the *Ironstream for Micro Focus Universal Discovery for IBM Z Installation Guide*. Recycle the VP390 job if any changes are made to the `SYSIN` cards or the startup JCL.

EVO205 MVS console name reached memory limit. Data lost

Message Variables

`name`	VP390 console name

Message Description
The extended console defined for the VP390 job has filled all available cells in the data space. The incoming message is not queued.

System Action
Processing continues.

User Action
Check the status of the extended console with the `DISPLAY CONSOLES,CN=name` command. If messages do not resume queuing to the extended console, recycle the VP390 job, making sure the console shuts down without any problems. You may need to define a new console with a larger message data space.

EVO206 MVS console name reached queue limit, data lost

Message Variables

`name`	VP390 console name

Message Description
The extended console defined for the VP390 job reached its maximum queue depth.

System Action
The incoming message is not queued. Processing continues.

User Action
Check the status of the extended console with the `DISPLAY CONSOLES,CN=`*`name`* command. If messages do not resume queuing to the extended console, recycle the VP390 job, making sure the console shuts down without any problems. Use the QL parameter on the `MVS SYSIN` card to increase the queue size of the console. See the definition of the MVS Parameter Card in the *Ironstream for Micro Focus Universal Discovery for IBM Z Installation Guide*.

EVO207 MVS console name stopped by internal error

Message Variables

name	VP390 console name

Message Description
The extended console defined for the VP390 job received an error while processing its message queues.

System Action
VP390 deactivates the console and stops the MVS subtask.

User Action
Recycle the subtask, then issue a `DISPLAY CONSOLES,CN=`*`name`* command to check the status of the name console.

EVO208 MVS console name reached alert percentage

Message Variables

name	VP390 console name

Message Description
The number of messages queued to the extended console reached a pre-specified alert percentage of the maximum queue depth.

System Action
Processing continues.

User Action
Verify that desired MVS messages are being sent to the Discovery probe client. Check the status of the extended console with the command: `DISPLAY CONSOLES,CN=`*`name`* If the queue shortage is not relieved shortly, recycle the VP390 job, making sure the console shuts down without any problems. Use the QL parameter on the MVS SYSIN card to increase the queue size of the console. See the definition of the MVS parameter card in the *Ironstream for Micro Focus Universal Discovery for IBM Z Installation Guide*.

EVO209 MVS console name suspended by request

Message Variables

name	VP390 console name

Message Description
A condition developed in the extended console defined for the VP390 job that caused the operating system to request console deactivation.

System Action
VP390 deactivates the console and stops the MVS subtask.

User Action
Recycle the subtask, then issue the command: `DISPLAY CONSOLES,CN=`*`name`* to check the status of the name console.

EVO210 MVS console name alert ECB posted for unknown reason

Message Variables

name	VP390 console name

Message Description
The extended console defined for the VP390 job is posted with an alert indicating a problem, but no error flags are set in the console status area.

System Action
Processing continues.

User Action
Check the condition of the console with the command: `DISPLAY CONSOLES,CN=`*`name`*

EVO211 DOM source key

Message Variables

source	message deletion type, either "`MSGKEY`" or "`TOKEN`"
key	identifying number of the original message

Message Description
The operating system has issued a Delete Operator Message notification that a previous message (identified by a MSGKEY) or group of messages (identified by a TOKEN) have been deleted from the console.

System Action
Processing continues.

User Action
If DOM processing is active in the VP390 job (activated by the "DOM" option on the MVS parameter card in `SYSIN`), then this message will be sent to the OM server. It can be used for automatically acknowledging an existing message on the OM browser. See "Using DOM Information" in Chapter 3 for more information.

EVO214 DOM flag updated

Message Variables
None.

Message Description
In response to a `MODIFY` command, the VP390 job has changed its processing of operating system DOM messages. See "About DOM Commands" in Appendix A.

System Action
Processing continues.

User Action
None.

EVO215 PERF parameter updated

Message Variables
None.

Message Description
In response to a `MODIFY` command, the VP390 job has updated its timing intervals for performance data gathering. See "About PERF Commands" in Appendix A.

System Action
Processing continues.

User Action
None.

EVO216 SMF buffer size changed from old to new

Message Variables

`old`	Previous size of the SMF buffer, in bytes
`new`	Updated size of the SMF buffer, in bytes

Message Description
In response to a `MODIFY` command, the VP390 job has changed the size of the SMF data collection buffer. See the description of the SMFBUFFER command of "About PERF Commands" in Appendix A.

System Action
Processing continues.

User Action
None.

EVO217 Debug for task changed from old to new

Message Variables

`task`	Subtask name
`old`	Previous debug value
`new`	Updated debug value

Message Description
In response to a `MODIFY` command, VP390 has updated the debugging value for the specified subtask `task`. The amount of debug information collected varies by subtask, with "0" indicating no debugging. Debug information is written to the SYSYPRINT DD of the VP390 job. Debug information should only be collected at the request of Precisely support.

System Action
Processing continues.

User Action
None.

EVO230 Unable to initialize RMA Rexx environment

Message Variables
None.

Message Description
The RMA subtask could not find the IRXEXEC program, which is necessary for initiating Rexx programs from VP390.

System Action
The RMA subtask terminates.

User Action
Verify that the IRXEXEC program exists in the system LPALST.

EVO302 name : VP390 PPI TASK INITIALIZED

Message Variables

`name`	Name of NetView/390 PPI subtask

Message Description
The program-to-program interface subtask for the VP390 job is successfully initialized in the NetView/390 address space.

System Action
Processing continues.

User Action
None.

EVO303 name : VP390 PPI TASK TERMINATED

Message Variables

`name`	Name of NetView/390 PPI subtask

Message Description
The program-to-program interface task for the VP390 job is terminated in the NetView/390 address space.

System Action
Processing continues, but VP390 no longer receives unsolicited VTAM messages from NetView/390.

User Action
Restart NetView/390 if it is terminated. If only the PPI subtask is terminated, restart the subtask from a NetView/390 operator session with the `START TASK=`*`name`* command.

EVO304 name : DSIFRE FAILED FOR USER STORAGE

Message Variables

`name`	Name of NetView/390 PPI subtask

Message Description
The VP390 PPI program running in the NetView address space received an error return code from the NetView/390 macro DSIFRE while attempting to free the 4K work area of memory during subtask shutdown.

System Action
Subtask shutdown processing continues.

User Action
Notify the system programmer that a potential memory leak exists in the currently running NetView/390.

EVO305 name : DSIFRE FAILED FOR QUEUED STORAGE

Message Variables

`name`	Name of NetView/390 PPI subtask

Message Description
The VP390 PPI program running in the NetView address space received an error return code from NetView/390 macro DSIFRE while attempting to free all remaining subtask memory during subtask shutdown.

System Action
Subtask shutdown processing continues.

User Action
Notify the system programmer that a potential memory leak exists in the currently running NetView/390.

EVO306 name : DSIFRE FAILED FOR MQS BUFFER

Message Variables

name	Name of NetView/390 PPI subtask

Message Description
The VP390 PPI program running in the NetView address space received an error return code from NetView/390 macro `DSIFRE` while attempting to free the memory allocated for the private message queue.

System Action
Processing continues.

User Action
Notify the system programmer that a potential memory leak exists in the currently running NetView/390.

EVO307 name : DSIGET FAILED FOR USER STORAGE

Message Variables

name	Name of NetView/390 PPI subtask

Message Description
The VP390 PPI program running in the NetView address space failed to get a 4K block of memory for use during processing.

System Action
Task termination flag is set.

User Action
Notify the system programmer that a potential memory shortage exists in the currently running NetView/390. The region size of the NetView/390 address space may need to be increased.

EVO308 name : ENQ ERROR

Message Variables

name	Name of NetView/390 PPI subtask

Message Description
An ENQ on the NetView/390 TVB chain failed.

System Action
If not already in termination processing, the task termination flag is set.

User Action
Notify the system programmer. Restart the subtask.

EVO309 name : DEQ ERROR

Message Variables

`name`	Name of NetView/390 PPI subtask

Message Description
A DEQ on the NetView/390 TVB chain failed.

System Action
If not already in termination processing, the task termination flag is set.

User Action
Notify the system programmer. Restart the subtask.

EVO310 name : TASK ALREADY EXISTS

Message Variables

`Name`	Name of NetView/390 PPI subtask

Message Description
The VP390 PPI subtask attempted to add itself to the NetView/390 TVB chain, but found another task with the same name already on the chain.

System Action
The task termination flag is set.

User Action
Verify that another instance of the subtask is not already running under this NetView/390. Restart the subtask.

EVO311 name : LOAD OF CNMNETV COMPLETE

Message Variables
name Name of NetView/390 PPI subtask

Message Description
The loading of the CNMNETV module into NetView virtual storage completed successfully.

System Action
Processing continues.

User Action
None.

EVO312 name : UNABLE TO LOAD CNMNETV

Message Variables
name Name of NetView/390 PPI subtask

Message Description
The loading of the CNMNETV module into virtual storage failed.

System Action
The subtask terminates.

User Action
Verify that load module CNMNETV exists in a NetView/390 STEPLIB dataset. Restart the subtask.

EVO313 name : NETVIEW INTERFACE FAILURE, RC=rcnumber

Message Variables

name	Name of NetView/390 PPI subtask
rcnumber	Hexadecimal return code from CNMNETV call

Message Description
A call to the CNMNETV interface routine failed.

System Action
The message is discarded.

User Action
For explanations of return codes, see the TME 10 *NetView for OS/390 Application Programmer's Guide.*

EVO314 name : NETVIEW COMMAND RECEIVED

Message Variables

name	Name of NetView/390 PPI subtask

Message Description
A message was successfully received from the PPI interface routine. This message is used for debugging purposes only. It is not displayed unless the subtask is reassembled with the CMDREC lines uncommented.

System Action
Processing continues.

User Action
None.

EVO315 autotask COMMAND EXECUTION FAILED

Message Variables

autotask	Name of NetView/390 autotask that executes the command

Message Description
A failure occurred in a command that was to be executed under NetView/390 on behalf of Ironstream.

System Action
The command is discarded.

User Action
Verify that the autotask defined under NetView/390 during Ironstream installation is active. Verify that the `NTICMD` and `NTIMVS` command lists are present in a NetView/390 `DSICLD` dataset. Verify that the name in the `EVOCMD_OPERATOR` field on the OVOW server (which was filled in while running the Add Node function) matches the autotask name defined under NetView/390.

EVO351 region TRANSACTION trans RESPONSE TIME OF used SECS. EXCEEDED THRESHOLD OF thresh SECS.

Message Variables

region	Name of CICS region
trans	Four-character transaction ID
used	Number of seconds used
thresh	Maximum number of seconds defined in threshold table

Message Description
A CICS transaction *trans* running under the CICS region *region* has completed, but the transaction time *used* exceeded the number of clock seconds *thresh* that was defined in the EVTHRTBL table.

System Action
Processing continues.

User Action
This message is sent to the system console. If desired, add the "EVO351" message ID to the VP390 filter table to forward it to the server as an alert.

EVO352 NO RESOURCE RECORD

Message Variables
None.

Message Description
The EVXMNOUT exit was called by CICS after the completion of a CICS transaction, but there was no monitoring resource record created for the XMNOUT exit program.

System Action
Processing continues.

User Action
Check with the system programmer that there are no errors in the CICS region.

EVO595 Command entered: cmdtxt

Message Variables

cmdtxt	Text of command entered

Message Description
The VP390 job received a command from a console.

System Action
Processing continues with the execution of the command.

User Action
None.

EVO600 TNUM TASKNAME STATUS RESTARTS/LIMIT SPECIFIC

Message Variables
None.

Message Description
This message is the header of a table which is generated in response to a `SHOW TASK` console command. Additional EVO600 messages will follow with data for each subtask.

System Action
Processing continues.

User Action
None.

EVO603 TNUM TASKNAME ADDRESS HEADER TCB

Message Variables
None.

Message Description
This message is the header of a table which is generated in response to a SHOW ADDR console command. Additional EVO603 messages will follow with data for each subtask.

System Action
Processing continues.

User Action
None.

EVO605 TNUM TASKNAME INPUTQ OUTPUTQ INFLOW OUTFLOW MC

Message Variables
None.

Message Description
This message is the header of a table which is generated in response to a SHOW `FLOW` console command. Additional EVO605 messages will follow with data for each subtask.

System Action
Processing continues.

User Action
None.

EVO608 TASKNAME DATE TIME

Message Variables
None.

Message Description
This message is the header of a table which is generated in response to a `SHOW VERSION` console command. Additional EVO608 messages will follow with data for each subtask.

System Action
Processing continues.

User Action
None.

EVO609 type filters:

Message Variables

`type`	Filter type, "Message"

Message Description
This message is the start of a list of filter table entries which is generated in response to a `SHOW FILTER` console command. Additional EVO609 messages will follow with lists of filter table entries. Message IDs will be listed four per line after the EVO609.

System Action
Processing continues.

User Action
None.

EVO610 type filter data action

Message Variables

`type`	Filter type, "Message"
`data`	User-entered data
`action`	Command action, either "added" or "deleted"

Message Description
Verification message to indicate that the message of filter table action entered from a `VP390 MODIFY` command has completed successfully.

System Action
Processing continues.

User Action
None.

EVO611 type filter data not found

Message Variables

type	Filter type, either "`Message`", "`JOBNAME`", or "`JOBID`"
data	User-entered data

Message Description
A `VP390 MODIFY` command could not find the data entry when attempting to delete it from the message table.

System Action
Processing continues.

User Action
Use the `SHOW FILTER` command to see the names of the currently defined message filters.

EVO612 No type filters defined

Message Variables

type	Filter type, "message" or "alert" or "command"

Message Description
A VP390 `MODIFY` command could not any filters of the type to display.

System Action
Processing continues.

User Action
None.

EVO613 All type filters deleted

Message Variables
type Filter type: "message and alert and command"

Message Description
A FILTER DEL ALL command has successfully deleted all message filter table entries.

System Action
Processing continues.

User Action
None.

EVO614 No suppressed messages

Message Variables
None.

Message Description
The VP390 message suppression table has no entries to display as a result of a `SHOW SUPPRESS` command.

System Action
Processing continues.

User Action
None.

EVO615 Suppressed message IDs:

Message Variables
None.

Message Description
This message is the header of a table which is generated in response to a `SHOW SUPPRESS` console command. Additional EVO615 messages will follow with a list of VP390 message IDs, eight per line that should not be sent to the console.

System Action
Processing continues.

User Action
None

EVO616 action suppression of msgid

Message Variables

action	Suppression action, either "Added" or "Removed".
msgid	VP390 message ID

Message Description
Verification message to indicate that the action to suppress or unsuppress a VP390 message ID from printing on the system console has completed successfully.

System Action
Processing continues.

User Action
None.

EVO617 Message ID msgid not found in suppression table

Message Variables

msgid	VP390 message ID

Message Description
An attempt to `UNSUPPRESS` a message ID in the VP390 message suppression table failed. The message ID given was not found in the table.

System Action
Processing continues.

User Action
Use the `SHOW SUPPRESS` command to see the list of message IDs currently in the table. Use only the 3-digit suffix of the message ID when issuing an `UNSUPPRESS` command.

EVO690 VP390 STOP Command accepted

Message Variables
None.

Message Description
The VP390 task has received a `STOP` command.

System Action
Processing continues with shutdown of any active subtasks, then ends the main task.

User Action
None.

EVO695 VP390 cmdtype command processed

Message Variables

cmdtype	Command type

Message Description
The VP390 job completed the initial processing of a console command. Additional messages may be sent, depending on whether additional work is being done by subtasks.

System Action
None.

User Action
None.

EVO698 Subtask task is already status

Message Variables

`task`	Subtask name
`status`	Current subtask status, either "active" or "inactive"

Message Description
A request to activate or deactivate a VP390 subtask was not processed because the subtask is already in that state.

System Action
None.

User Action
Use the `SHOW TASK` command to verify the status of the VP390 subtasks.

EVO699 Invalid operator command entered

Message Variables
None.

Message Description
An invalid `MODIFY` command was sent to the VP390 task.

System Action
None.

User Action
See Appendix A for syntax rules of MODIFY commands.

EVO701 Starting subtask #idnum for info

Message Variables

idnum	Numerical ID for the newly started subtask
info	Information sent to the ATTACH macro

Message Description
VP390 attached a subtask with the information provided in *info*.

System Action
Processing continues with the ATTACH attempt.

User Action
None.

EVO702 Buffer size = sizeM, Queue depth = totalmsg, Maximum = maxmsg

Message Variables

size	Size (in megabytes) allocated for messages
totalmsg	Total message queue depth
maxmsg	Maximum message queue depth permitted

Message Description
A message queuing problem occurred for an MCS console defined for VP390. This message will be displayed only in the VP390 job log. Additional message(s) giving more detailed information about the problem may appear on the system console at the same time.

System Action
Processing continues. The MCS console may be terminated, depending on the severity of the queuing problem.

User Action
Monitor the VP390 job log and system console for the next message and necessary action.

EVO703 Console name is utilizing pct% of message queue

Message Variables

name	Name of defined extended console
pct	Percentage of console queue in use

Message Description
This message is generated when the extended console for gathering MVS messages has a backlog of messages on its queue to be processed by the VP390 task. pct tells what percentage of the console's queue is in use. This message is only generated when using the QLP option of the MVS SYSIN card.

System Action
Processing continues.

User Action
The extended console name may need to be re-defined with a larger queue size. See the QL and QLP options of the MVS parameter card in the *Installation Guide*.

EVO704 Console name queue backlog has been relieved

Message Variables

name	Name of defined extended console

Message Description
This message is generated after an `EVO703` message is issued to announce that the console message queue shortage has been relieved. This message is only generated when using the QLP option of the `MVS SYSIN` card.

System Action
Processing continues.

User Action
The extended console name may need to be re-defined with a larger queue size. See the QL and QLP options of the MVS parameter card in the *Installation Guide* This message can be used for automatically acknowledging an existing `EVO703` message on the OM browser.

EVO778 RMF data not available, rc=code

Message Variables

code	Return code

Message Description
The VP390 job encountered an error while attempting to collect system data from the mainframe Resource Measurement Facility (RMF) for an `ev390hostcmd` 46 option 02 call.

System Action
The OSINFO subtask will send an `EVO131` error message in response to the `ev390hostcmd` explaining that the command had failed to complete.

User Action
The meaning of the return code can be looked up in Chapter 1 of the *IBM Resource Measurement Facility Programmer's Guide* under the section of "Return Codes" for the ERBSMFI command.

EVO801I ERROR ACTIVATING CONSOLE

Message Variables
None.

Message Description
An error was detected when attempting to activate an EMCS console named EVORXCON.

System Action
The command ends.

User Action
Use the `DISPLAY CONSOLES,CN=EVORXCON` command to verify that an EMCS console with that name does not already exist.

EVO802I ERROR TRYING TO GET A MESSAGE

Message Variables
None.

Message Description
An error was detected when attempting to retrieve console messages.

System Action
The command ends.

User Action
Look for previous errors that may have caused this condition.

EVO805I ERROR DEACTIVATING CONSOLE

Message Variables
None.

Message Description
An error was detected while attempting to deactivate an EMCS console.

System Action
Processing continues.

User Action
Look for previous errors that may have caused the condition. Deactivate the console before issuing the EVORXCON command again.

EVO901 Stopping subtask #number: name

Message Variables

number	Subtask number
name	Subtask name

Message Description
This message is issued in response to a STOP command. One message is issued for each VP390 subtask.

System Action
A termination command is sent to each of the existing subtasks.

User Action
None.

EVO902 name subtask terminated, RC = rcnumber

Message Variables

name	Name of subtask
rcnumber	Return code from termination call

Message Description
The named subtask is terminated.

System Action
Any queues or memory allocated for the subtask are freed.

User Action
None.

EVO903 name type queue freed, RC = rcnumber

Message Variables

name	Name of subtask
type	Queue type, either "Input" or "Output"
rcnumber	Return code from `Free` call

Message Description
An allocated message queue for the named subtask has been cleared during subtask termination.

System Action
Processing continues.

User Action
None.

EVO904 All VP390 subtasks completed

Message Variables
None.

Message Description
The VP390 job completed the shutdown of all subtasks.

System Action
Processing continues with main task shutdown.

User Action
None.

EVO905 Restart #num of subtask name will be attempted in sec seconds

Message Variables

num	Count of number of restarts for this subtask
name	Name of subtask
sec	Number of seconds until next automatic restart attempt

Message Description
The subtask name has been terminated, but will be automatically restarted in sec seconds.

System Action
Processing continues.

User Action
None.

EVO906 No auto restart for name - Use INIT command to restart

Message Variables

name	Name of subtask

Message Description
The subtask *name* has terminated and will not restart because it has exceeded the number of automatic restarts allowed.

System Action
Processing continues.

User Action
Use the console `INIT` command to restart the subtask. See Appendix A for the syntax of the INIT command. Use the console command `SHOW TASK` to see how many restarts are allowed for each subtask. To change the number of automatic restarts that a subtask is allowed, add a RESTART card to the `SYSIN` deck just prior to the name subtask parameter card. See the "RESTART Parameter Card" in the *Installation Guide* for the syntax of the RESTART card.