CA Workload Automation Agent for i5/OS

Implementation Guide

r11.3, Second Edition

CA Technologies Product References

This document references the following CA Technologies products:

- CA Process Automation
- CA Workload Automation AE
- CA Workload Automation Agent for Application Services (CA WA Agent for Application Services)
- CA Workload Automation Agent for Databases (CA WA Agent for Databases)
- CA Workload Automation Agent for i5/OS (CA WA Agent for i5/OS)
- CA Workload Automation Agent for Linux (CA WA Agent for Linux)
- CA Workload Automation Agent for Micro Focus (CA WA Agent for Micro Focus)
- CA Workload Automation Agent for Microsoft SQL Server (CA WA Agent for Microsoft SQL Server)
- CA Workload Automation Agent for Oracle E-Business Suite (CA WA Agent for Oracle E-Business Suite)
- CA Workload Automation Agent for PeopleSoft (CA WA Agent for PeopleSoft)
- CA Workload Automation Agent for Remote Execution (CA WA Agent for Remote Execution)
- CA Workload Automation Agent for SAP (CA WA Agent for SAP)
- CA Workload Automation Agent for UNIX (CA WA Agent for UNIX)
- CA Workload Automation Agent for Web Services (CA WA Agent for Web Services)
- CA Workload Automation Agent for Windows (CA WA Agent for Windows)
- CA Workload Automation CA 7 Edition
- CA Workload Automation DE
- CA Workload Automation Desktop Client (CA WA Desktop Client)
- CA Workload Automation ESP Edition
- CA Workload Control Center

Contact CA Technologies

Contact CA Support

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

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

Providing Feedback About Product Documentation

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

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

Contents

Chapter 1: Introduction

This section contains the following topics:

Intended Audience

This document is for system administrators who are responsible for upgrading, installing, and configuring agents.

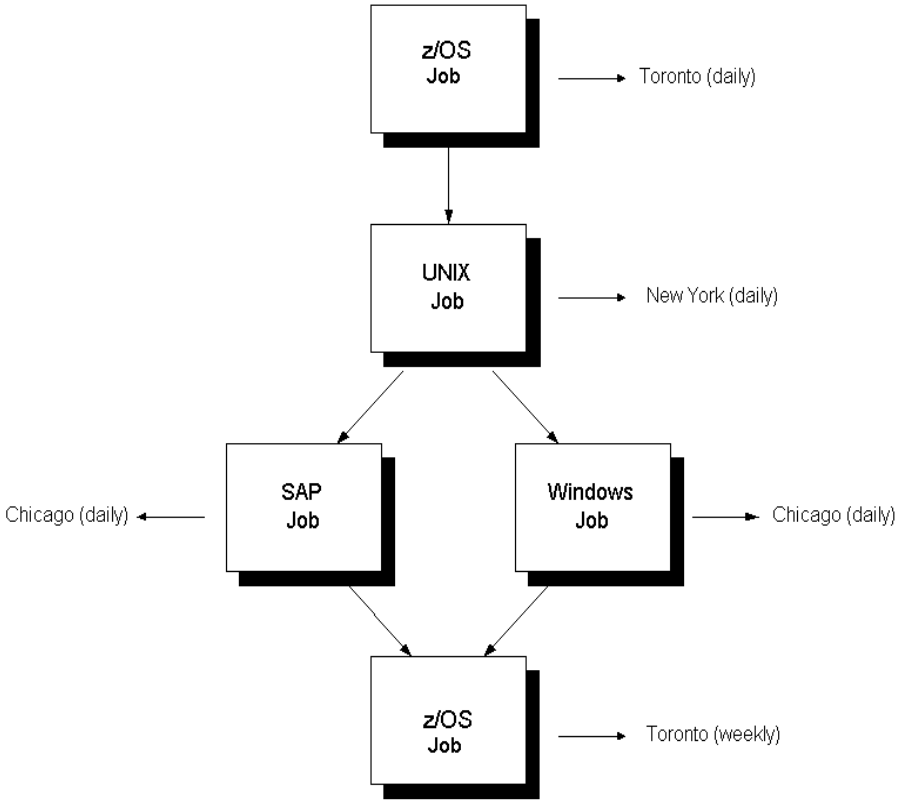
To use this guide, you must be familiar with the operating system where the agent is installed and any third-party products or software technology that the agent uses.

Agents

Agents are the key integration components of CA's workload automation products. Agents let you automate, monitor, and manage workload on all major platforms, applications, and databases.

Example: Run Workload with Different Types of Jobs

The following workload contains z/OS jobs, a UNIX job, an SAP job, and a Windows job, running on different computers, in different locations, and at different times:



CA WA Agent for i5/OS

CA WA Agent for i5/OS runs on the i5/OS operating system enabling the scheduling manager to submit and run workload on the i5/OS platform. You can run workload from the following file systems:

- Root file system
- Open systems file system (QOpenSys)
- Library file system (QSYS)

You can schedule most UNIX workload, such as UNIX scripts, in the PASE environment on the i5/OS operating system.

Using the agent, you can automate and manage your i5/OS environment from one central point of control. For example, you can do the following:

- Process jobs on an i5/OS platform with predecessor or successor jobs running on a z/OS mainframe or on other platforms
- Run workload across one or more i5/OS systems connected to a central scheduling manager site
- Pass information between the scheduling manager and the agent

You can also use the agent to automate FTP transfers using FTP jobs. An FTP job can use an existing FTP server or the agent as an FTP server. The FTP job always acts as an FTP client. You can set up the agent to run as an FTP client, FTP server, or both. You can define SCP and SFTP jobs to securely transfer binary files between an agent computer and a remote computer. You can upload to or download data from a remote server. The data is encrypted during the transfer. Other functionality includes the ability to monitor the disk, CPU and files on the system, as well as respond to jobs in a message wait (MSGW) state on the i5 system, and retrieve the spool file(s) for a job.

Job Types Supported by CA WA Agent for i5/OS

With CA WA Agent for i5/OS you can define an i5/OS job that lets you run a program or issue a command on an i5/OS system. You can run i5/OS jobs in the following file systems:

- Root file system
- Open systems file system (QOpenSys)
- Library file system (QSYS)

Within an i5/OS job definition, the environment option lets you set the following details:

- Library name, library list, or current library for running a program
- Job specifications to define the i5/OS job name, options under which the job will run, where it will run, and which user will run it
- Ending exit value of the program, such as a severity code

How a Scheduling Manager and Agents Communicate

Agents receive and respond to commands sent by the scheduling manager and transmit data and messages back to the scheduling manager. A scheduling manager and agents communicate by sending Automated Framework Messages (AFMs) to each other. Communication is asynchronous using message queues through TCP/IP ports. For example, while the scheduling manager is sending a new job request to an agent, that agent can be sending completion status for another job.

The following table summarizes the relationship between the scheduling manager and agents:

Scheduling Manager	Agent
Is aware of the entire network	Is aware of the local environment
Sends commands and parameters to the agents	Responds to commands and parameters sent by the scheduling manager
Receives data from the agents	Transmits data to the scheduling manager
Makes decisions	Takes direction from the scheduling manager
Schedules jobs	Runs jobs on different platforms

Receiver Ports

A scheduling manager and agents each have TCP/IP ports to receive messages. The receiver listens on its designated port for messages from one or more senders. When the sender has messages to transmit, it connects to the port of the receiver, sends the messages, and closes the connection.

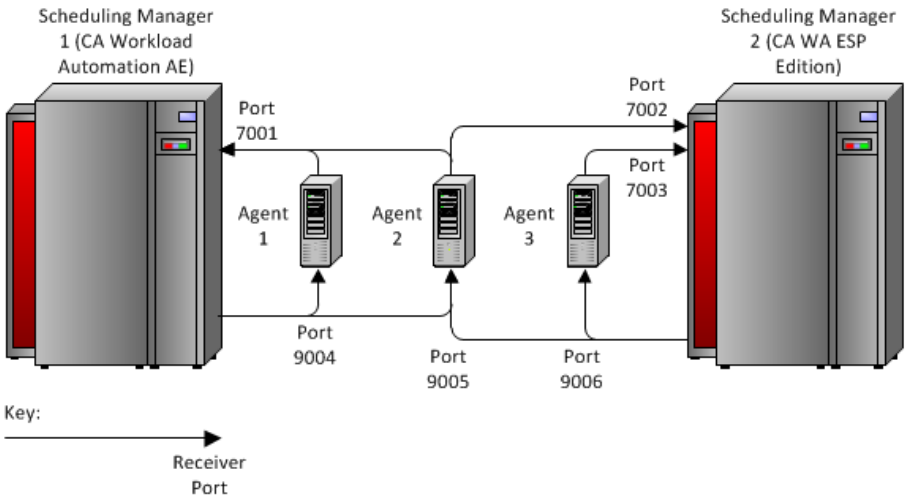
Receiver port configuration is restricted as follows:

- CA Workload Automation AE and CA Workload Automation CA 7 Edition have only one receiver port. The receiver port can receive messages from multiple agents.
- CA Workload Automation ESP Edition can have multiple receiver ports (for example, to separate encrypted and unencrypted message traffic). Each of these ports can receive messages from multiple agents.
- An agent has only one receiver port. This port can receive messages from multiple scheduling managers.

Communication Configuration Example

The following diagram shows some possible communication configurations between two scheduling managers and agents.

- Scheduling Manager 1 (CA Workload Automation AE) communicates with Agent 1 and Agent 2 and receives messages from both agents through port 7001. Scheduling Manager 1 sends messages to port 9004 on Agent 1 and port 9005 on Agent 2.
- Scheduling Manager 2 (CA Workload Automation ESP Edition) communicates with Agent 2 and Agent 3 and receives messages from Agent 2 through port 7002 and from Agent 3 through port 7003. Scheduling Manager 2 sends messages to port 9005 on Agent 2 and port 9006 on Agent 3.



Chapter 2: Implementation Checklist

This section contains the following topics:

How to Install and Configure the Agent

You can install the agent using an interactive program or using a non-interactive command-based silent installer. If you are installing multiple agents, the silent installer allows you to automate the installation process. After you install the agent, you can configure it to change your settings or to implement additional features. You also set up security features after the agent is installed.

To install and configure the agent, follow these steps:

1. Review the system requirements in the *CA Workload Automation Agent for i5/OS Release Notes*.
2. Collect information about the scheduling manager (see page 18).
3. Review the agent installation program options (see page 20).
4. Create an i5/OS user profile for the agent (see page 25).
5. Install JRE 1.5, if required (see page 25).
6. Change the JAVA_HOME environment variable, if required (see page 25).
7. Install the agent using one of these methods:
   - Install the agent using an interactive program (see page 26).
   - Install the agent using a silent installer (see page 27).
8. Start the subsystem and the agent (see page 35).
9. (Optional) Change the password for the agent user profile (see page 36).

10. Configure the scheduling manager to work with the agent:
    - Define the agent on the scheduling manager.
    - (Optional) Define a user on the scheduling manager.
    - Configure security profiles on the scheduling manager.
    - Verify that the agent works with the scheduling manager.

    Note: For detailed instructions to complete the above steps, refer to the documentation for your scheduling manager (see page 151).
11. (Optional) Configure the agent (see page 49).
12. Configure security features (see page 83).

Collecting Information about the Scheduling Manager

During the agent installation you are prompted for information about your scheduling manager. Speak to your administrator and collect the following information:

- Scheduling Manager Name—Corresponds to the Scheduling Manager ID required in the agent installation program
- IP address—Corresponds to the scheduling manager Address required in the agent installation program
- Port number—Corresponds to the Scheduling Manager Port required in the agent installation program

Chapter 3: Installing the Agent

This section contains the following topics:

Installation Consideration for CA Workload Automation AE

To run both native and UNIX jobs on the same i5/OS computer, you must install two i5/OS agents on that computer. This requirement only applies to agents configured to work with CA Workload Automation AE.

On the agent that runs native i5/OS jobs, set the following parameter in the agentparm.txt file:

```
oscomponent.targetenvironment=I5
```

On the agent that runs UNIX jobs, set the following parameter in the agentparm.txt file:

```
oscomponent.targetenvironment=UNX
```

Note: For more information about UNIX workload that can run in the PASE environment, see the IBM i5/OS documentation.

Installing Multiple Agents on a Single Computer

You can install multiple agents on a single computer. This configuration lets you do the following:

- Distribute the load of the jobs across multiple agents. For example, you can run different jobs for different business applications on the same computer. To run this workload, you can install an agent for one business application and an agent for the other business application and provide access at the agent level.
- Test maintenance applied to an agent before applying maintenance to the production agent.

Important! If a computer with multiple agents is not available, all workload scheduled on that computer is impacted. To avoid a single point of failure, we recommend that you install agents across multiple computers.

Agent Installation Options

The interactive agent installation program prompts you for the following information:

Installation Path

Specifies the path to the location where you want to install the agent program files. The specified location must be empty.

Default: ~/CA/ CA_WA_Agent_R11_3 where ~ is the user home directory that is installing the agent

Library

Defines the name of the library that contains the objects the agent requires.

Default: CAWAGNT113

Notes:

- The library can exist, but the library must be empty.
- This library cannot be the same library used for another agent installed on the same i5/OS system.

AgentParm File Conversion

Indicates whether or not the installation program preserves settings for a r7.0 agent by converting the existing agentparm.txt file.

- Yes—Preserves the parameter settings from the previous release of the agent
- No—Does not preserve the parameter settings from the previous release of the agent

Default: No

Path to agentparm.txt file

Specifies the path to an existing agent installation directory for a r7.0 agent. When you specify this path, the installation program converts the agentparm.txt file in this directory to an R11.3 version.

Default: /home/Cybermation/ESP_System_Agent_R7

Agent Name

Defines the agent name. You need the agent name when you configure the scheduling manager to work with the agent.

Default: AGENT

Limits: Up to 16 characters; the first character must be a letter

Notes:

- Agent names must begin with an alphabetic character and can contain any alphanumeric characters and the special characters @, $, and underscore (_). Because the scheduling manager uses agent names as file names, use standard file-naming conventions for your operating system.
- For CA Workload Automation DE, the agent name must be in uppercase.

Input Port

Specifies the main port number the agent uses to listen for incoming messages from the scheduling manager. You need this port when you configure the scheduling manager to work with the agent.

Default: 7520

Limits: 1024-65534

Note: On UNIX, ports 1–1023 are reserved ports and require root access.

Number of Managers

Specifies the number of scheduling managers you want to configure to work with the agent.

Default: 1

Manager *n* ID

Specifies the name of the scheduling manager instance that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured.

Default: CENTRAL_MANAGER

Example: MYSERVER

Note: You can configure the agent to work with multiple scheduling managers by adding additional scheduling manager definitions in the agentparm.txt file.

Manager *n* Address

Specifies the address of the scheduling manager that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the IP address in the connection details for the scheduling manager. You can specify a list of addresses for the scheduling manager.

Example: 172.24.36.107 (IPv4) or 0:0:0:0:0:FFFF:192.168.00.00 (IPv6)

Notes:

- You can specify a DNS name instead of the IP address for the scheduling manager. However, your agent computer must be able to resolve the DNS name at all times. If there is a DNS outage and your agent computer cannot resolve DNS names, the agent cannot communicate with the scheduling manager.
- If the scheduling manager address never changes, enter the DNS name for the scheduling manager in the hosts file for your agent computer. This entry helps ensure that the IP address can be resolved after DNS disruptions.

Manager *n* Port

Specifies the port that the scheduling manager listens on for communication from agents, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the port number in the connection details for the scheduling manager.

Default: 7507

Limits: 1024-65534

Cipher Algorithm

Specifies the type of cipher algorithm the agent uses to encrypt and decrypt messages sent to the scheduling manager. The agent supports the following types:

- AES—Advanced Encryption Standard that uses a 32-character encryption key. AES is the algorithm required by U.S. Government organizations to protect sensitive (unclassified) information (FIPS-140-2 compliance).
- BLOWFISH—A license-free encryption algorithm that uses an encryption key of 32 to 64 even-numbered characters.
- DES—Data Encryption Standard that uses a 16-character encryption key.
- DESEDE—Triple Data Encryption Algorithm that applies the DES algorithm three times to each data block.

Default: DES

Encryption Key

Defines the encryption key the agent uses to communicate with the scheduling manager. The encryption key must be prefixed with 0x and followed by the number of characters required for the chosen cipher algorithm:

- AES—32 hexadecimal character encryption key.

  Note: If you omit the 0x prefix, the keygen utility interprets the inputted value as a 16-character passphrase and not as a hexadecimal number. If you enter less than 16 characters, the keygen utility appends the passphrase with spaces for the missing number of characters. The keygen utility will internally encode the 16 character passphrase into a 32 hexadecimal character AES encryption key.

- Blowfish—32-64 even-numbered hexadecimal character encryption key
- DES—16 hexadecimal character encryption key
- DESEDE—48 hexadecimal character encryption key

Local Security Option

Specifies whether local security on the agent is enabled or disabled. Local security on the agent controls which scheduling manager user IDs can perform certain actions, for example, which user IDs can issue CONTROL messages to the agent. If you enable local security, define security rules in a security.txt file.

Default: disabled

Management Connector Option

Enables the following management connectors:

SNMP Connector

Lets you use an SNMP Manager to monitor and control the agent.You can connect the agent to any SNMP Manager that supports SNMP v1, v2, and v3. This option requires the SNMP Manager address and User Datagram Protocol (UDP) port.

Default: disabled

JMX Connector

Lets you use a JMX console to monitor and control the agent. You can use any JMX console that implements JSR-160. This option requires the input port number for the JMX console.

Default: disabled

Remote SNMP Manager Trap Listener Host

Specifies the SNMP trap receiver host name.

Default: localhost

Note: This value applies to the SNMP management connector option.

SNMP Agent Port

Specifies the SNMP GET/SET listening port.

Default: 161

Limits: 1-65535

Note: This value applies to the SNMP management connector option.

JMX Communication Port

Specifies the input port number for the JMX console.

Default: 1099

Note: This value applies to the JMX management connector option.

Enable FTP Plug-in

Enables the FTP plug-in on the agent, which lets you configure FTP client and FTP server options.

Default: disabled (unselected)

FTP Client Information

Specifies whether the agent can act as an FTP client using Regular or Secure Sockets Layer (SSL) encryption for FTP transfers.

Default: Regular Client Transfer

FTP Server Information

Specifies whether the agent can act as an FTP server using regular or Secure Sockets Layer (SSL) encryption for FTP transfers.

Default: Disable FTP Server

FTP Server Port

Specifies the port number for the agent to act as an FTP server.

Default: 21

Limits: 1-65534

FTP User ID

Specifies the FTP user ID required to access the FTP server.

FTP Password

Specifies the password corresponding to the FTP user ID.

Limits: case-sensitive

Verify FTP Password

Confirms the FTP password.

SNMP Job Type Support Configuration

Enables the agent to act as an SNMP Manager to emit and listen for SNMP traps. This option lets users define and run SNMP job types. The agent supports SNMP v1, v2, and v3.

Default: disabled (unselected)

SNMP Trap Listener Port

Specifies the agent port listening for trap information.

Default: 162

Limits: 1-65535

Create an i5/OS User Profile for the Agent

Prior to installing the agent, you must create an i5/OS user profile that will run the agent. This user profile must have the following authorities:

- *ALLOBJ—Allows access to monitored objects, job queues, job definitions, and user profiles for job submission.
- *JOBCTL—Allows the agent to control jobs, including set priorities, cancel jobs, and respond to messages.
- *SPLCTL—Allows the agent to access the spool files for the jobs.

Notes:

- Instead of granting *ALLOBJ to the user profile, you can grant specific authorities that allow access to required objects. The user profile must be enabled and have a password.
- You can use the same user profile for multiple agents installed on the same i5/OS system.

Install JRE 1.5

To install the agent on i5/OS systems, you must have J2SE 5.0 32-bit installed. This JRE is option 8 of product 5722-JV1 as shown on the DSPLICPGM command.

Change the JAVA_HOME Environment Variable

If you are installing the agent on an i5/OS system and the JAVA_HOME environment variable is not jdk15, you must change JAVA_HOME, as follows:

```
JAVA_HOME=/QIBM/ProdData/Java400/jdk15
```

Install the Agent Using an Interactive Program

You can install the agent using an interactive program that lets you change and review your settings prior to starting the installation process. The installation program installs a packaged Java Virtual Machine (JVM) for the agent.

Note: You can also use a silent installation program which lets you automate the installation. When you have multiple agents to install, a silent installation is useful.

To install the agent using an interactive program

1. Copy the setup file from the product CD or download a zip file from the CA Support Online website, found at http://ca.com/support.
2. Log on to the i5/OS system using the i5/OS user profile you created for the agent.
3. FTP the setup file in binary mode to a directory in the root file system.
4. Open a PASE terminal session (see page 41).
5. Change to the directory where the setup file was uploaded.
6. Enter the following command:

   ```
   export JAVA_HOME=/QOpenSys/QIBM/ProdData/JavaVM/jdk50/32bit
   ```

   The default JRE is set to Java 5.0.
7. Type the following command to start the installation:

   ```
   java –jar setup.jar
   ```

   Note: If you are using a 5250 emulator to install the agent and want to configure the agent FTP server, you must use the –Dterminal.CanMask=false option. The 5250 emulator does not allow the masking of the password required when configuring the FTP server. In this case, type the following command to start the installation:

   ```
   java –Dterminal.CanMask=false –jar setup.jar
   ```

   The agent installation program opens.
8. Press Enter to review each page of the license agreement.

   A prompt appears to accept the license agreement after you review all pages.
9. Type **y** to accept the license agreement.
10. Continue with the installation by entering the required information (see page 20).

    Note: To comply with U.S. Government encryption standard FIPS 140-2, select AES when you are prompted for the cipher algorithm.
11. Review your selections. To return to a previous option, type **back**.
12. Press **Enter** to exit the installation program.

    The agent is installed and the settings are stored in the agentparm.txt file located in the agent installation directory.

How to Install the Agent Using a Silent Installer

A silent installer lets you automate the installation of multiple agents. You can configure a properties file for each agent and then run a silent installer instead of using an interactive program to install each agent.

To install the agent using a silent installer

1. Configure the installer.properties file (see page 27).
2. Run the silent installer (see page 34).
3. Check the generated log file (see page 34).

Configure the installer.properties File

You configure the installer.properties file as the first step in performing a silent installation for one or more agents. We recommend that you keep a copy of this file to use as a template.

To configure the installer.properties file

1. Open the installer.properties file, which is available on the product CD or CA Support Online website, found at http://ca.com/support.
2. Edit the properties for the agent. Remove the # sign to uncomment each property line.
3. Save the file.

   The properties are set in the installer.properties file.

Silent Installer Properties

The installer.properties file contains the following properties for the agent:

USER_INSTALL_DIR

Specifies the path to the location where you want to install the agent program files. The specified location must be empty.

USER_INSTALL_LIBRARY

Specifies the library on the system where the native i5/OS objects will be stored. The specified library must be empty.

AGENT_INFO_1

Defines the agent name. You need the agent name when you configure the scheduling manager to work with the agent.

Default: AGENT

Limits: Up to 16 characters; the first character must be a letter

Notes:

- Agent names must begin with an alphabetic character and can contain any alphanumeric characters and the special characters @, $, and underscore (_). Because the scheduling manager uses agent names as file names, use standard file-naming conventions for your operating system.
- For CA Workload Automation DE, the agent name must be in uppercase.

AGENT_INFO_2

Specifies the main port number the agent uses to listen for incoming messages from the scheduling manager. You need this port when you configure the scheduling manager to work with the agent.

Default: 7520

Limits: 1024-65534

NUM_MANAGER_N=N

Specifies the number of scheduling managers (N) the agent works with.

Default: NUM_MANAGER_1=1

MANAGER_*n*_INFO_1

Specifies the name of the scheduling manager instance that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured.

Default: CENTRAL_MANAGER

Example: MYSERVER

MANAGER_*n*_INFO_2

Specifies the address of the scheduling manager that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the IP address in the connection details for the scheduling manager. You can specify a list of addresses for the scheduling manager.

Example: 172.24.36.107 (IPv4) or 0:0:0:0:0:FFFF:192.168.00.00 (IPv6)

Notes:

- You can specify a DNS name instead of the IP address for the scheduling manager. However, your agent computer must be able to resolve the DNS name at all times. If there is a DNS outage and your agent computer cannot resolve DNS names, the agent cannot communicate with the scheduling manager.
- If the scheduling manager address never changes, enter the DNS name for the scheduling manager in the hosts file for your agent computer. This entry helps ensure that the IP address can be resolved after DNS disruptions.

MANAGER_*n*_INFO_3

Specifies the port that the scheduling manager listens on for communication from agents, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the port number in the connection details for the scheduling manager.

Default: 7507

Limits: 1024-65534

STRONG_ENCRYPTION_CIPHER

Specifies the type of cipher algorithm the agent uses to encrypt and decrypt messages sent to the scheduling manager. The agent supports the following types:

- AES—Advanced Encryption Standard that uses a 32-character encryption key. AES is the algorithm required by U.S. Government organizations to protect sensitive (unclassified) information (FIPS-140-2 compliance).
- BLOWFISH—A license-free encryption algorithm that uses an encryption key of 32 to 64 even-numbered characters.
- DES—Data Encryption Standard that uses a 16-character encryption key.
- DESEDE—Triple Data Encryption Algorithm that applies the DES algorithm three times to each data block.

Default: DES

STRONG_ENCRYPTION_KEYGEN

Defines the encryption key the agent uses to communicate with the scheduling manager. The encryption key must be prefixed with 0x and followed by the number of characters required for the chosen cipher algorithm:

- AES—32 hexadecimal character encryption key.

  Note: If you omit the 0x prefix, the keygen utility interprets the inputted value as a 16-character passphrase and not as a hexadecimal number. If you enter less than 16 characters, the keygen utility appends the passphrase with spaces for the missing number of characters. The keygen utility will internally encode the 16 character passphrase into a 32 hexadecimal character AES encryption key.

- Blowfish—32-64 even-numbered hexadecimal character encryption key
- DES—16 hexadecimal character encryption key
- DESEDE—48 hexadecimal character encryption key

LOCAL_SECURITY

Specifies whether local security on the agent is enabled or disabled. Local security on the agent controls which scheduling manager user IDs can perform certain actions, for example, which user IDs can issue CONTROL messages to the agent. If you enable local security, define security rules in a security.txt file.

- off—Disables local security
- on—Enables local security

Default: off

The following properties apply if you want to connect the agent to an SNMP manager.

SNMP_MGMT_CONN

Enables an SNMP connector that lets you use an SNMP Manager to monitor and control the agent. The agent supports SNMP v1, v2, and v3. This option requires the SNMP Manager address and UDP port.

- 0—Disables the SNMP connector
- 1—Enables the SNMP connector

MGMT_SNMP_HOST

Identifies the SNMP Manager IP address or DNS name. Your SNMP administrator can provide the host name.

Default: localhost

Example: 172.24.36.107

MGMT_CONN_AGENT_PORT

Specifies the SNMP GET/SET listening port.

Default: 161

Limits: 1-65535

JMX_PLUGIN

Enables a JMX connector that lets you use a JMX console to monitor and control the agent. You can use any JMX console that implements JSR-160.

- 0—Disables the JMX connector
- 1—Enables the JMX connector

JMX_CONNECTOR_PORT

Specifies the port where the JMX connector listens.

Default: 1099

Limits: 1-65534

FTP_PLUGIN

Enables the FTP plug-in on the agent, which lets you configure FTP client and FTP server options.

- 0—Disables the FTP plug-in
- 1—Enables the FTP plug-in

Default: 0 (disabled)

FTP_SSL_CLIENT_ENABLED

Specifies whether the agent can act as an FTP client using Regular or Secure Sockets Layer (SSL) encryption for FTP transfers.

- true—Enables SSL encryption
- false—Enables regular encryption

Default: false

FTP_NO_SERVER

Sets whether the agent can act as an FTP server.

- true—Disables FTP server
- false—Enables FTP server

Default: true

FTP_SSL_SVR_ENABLED

Specifies whether the agent can act as an FTP server using regular or Secure Sockets Layer (SSL) encryption for FTP transfers.

- true—Enables SSL encryption
- false—Enables regular encryption

Default: false

FTP_SVR_PORT

Specifies the port number for the agent to act as an FTP server.

Default: 21

Limits: 1-65534

FTP_USER_ID

Specifies the FTP user ID required to access the FTP server.

FTP_PASSWORD

Specifies the password corresponding to the FTP user ID.

Limits: case-sensitive

FTP_PASSWORD_V

Confirms the FTP password.

SNMP_PLUGIN

Enables the agent to act as an SNMP Manager to emit and listen for SNMP traps. This option lets users define and run SNMP job types. The agent supports SNMP v1, v2, and v3.

- 0—Disables the SNMP plug-in
- 1—Enables the SNMP plug-in

Default: 0 (disabled)

SNMP_P_TRAP_PORT

Specifies the agent port listening for trap information.

Default: 162

Limits: 1-65535

NUM_MANAGER_VARS_2

Specifies the number of manager environment variables for the scheduling manager.

Limits: 0-3

MANAGER_VARS_*n*_INFO_1

Specifies the name of the specific scheduling manager the environment variables apply to, where *n* is an integer that corresponds to the scheduling manager being configured.

Example: MANAGER1_VAR

MANAGER_VARS_n_INFO_2

Specifies the path to the file that stores the environment variables, where *n* is an integer that corresponds to the scheduling manager being configured.

Example: ~/MANAGER_1_FILE.TXT

NUM_USER_VARS_2

Specifies the number of user environment variables for the scheduling manager.

Limits: 0-3

USER_VARS_*n*_INFO_1

Specifies the name of the user the environment variables apply to, where *n* is an integer that corresponds to the scheduling manager being configured.

Example: USER1

USER_VARS_*n*_INFO_2

Specifies the path to the file that defines user-specific variables.

Example: ~/USER_1_FILE.TXT

LOOKUPCMD

Determines how to specify the script or command name to run in a job definition.

- true—The script or command name can be specified without the full path in a job definition.The agent looks up the path to the script or command name for the specified user ID.
- false—The full path to the script or command name must be specified in the job definition.

Default: true

JOBLOG

Sets whether the agent creates a job log for each UNIX job that runs on the i5/OS system.

- true—Enables job logs
- false—Disables job logs

Default: false

Note: These job logs are different than the job logs the i5/OS system creates.

Run the Silent Installer

After you configure the installer.properties file for each agent you want to install, you run the silent installer to perform the installation.

To run the silent installer

Type the following command at the command prompt:

```
java -jar Setup.jar -f response file
```

response file

Specifies the full path to the file where the installation responses are written.

The agent is installed.

Review the Generated Log File

The agent installation program creates a log file, which you can review for installation errors. Review the following file located in the agent installation directory:

```
CA_Workload_Automation_Agent_R11.3_InstallLog.log
```

Agent Objects

This section lists the objects that are added by the installation program. The agent requires these objects, which are located in the agent library.

The following objects can be modified as required:

- CYBESPJOBD *JOBD—The job description used by the agent process
- CYBESPJOBS *JOBD—The default job description used for the i5/OS jobs that are submitted by the agent
- CYBESPCLS *CLS—The default class object that the agent uses when jobs are submitted
- CAWAGNT113 *SBSD—The default subsystem that the agent and its jobs run under

The following objects are internal to the agent. Do not modify these objects.

- CYBCMD *PGM
- CYBESPRUN *PGM
- CYBAGENT *CMD
- ESPMGR *CMD
- CYBESPMSGF *MSGF
- CYBESPHOME *DTAARA
- CYBESPHOLD *JOBQ
- CYBESPJOBS *JOBQ

Start the Subsystem and the Agent

To complete the agent installation, you must start the subsystem that runs the agent jobs and then start the agent.

Note: If you installed multiple agents, start each subsystem and agent.

To start the agent and the subsystem

1. Ensure you are logged on to the i5/OS system using a user profile that has authority to the agent library/objects.
2. Enter the following command:

   `CHGCURLIB libraryname`

   libraryname

   Specifies the name of the library that contains the required agent objects.

   The current library is changed to the library where the agent objects are located.
3. Enter the following command:

   `STRSBS subsystemname`

   subsystemname

   Specifies the name of the subsystem that runs the agent jobs.

   Default: The name of the library that contains the required agent objects.

   The subsystem that runs the agent jobs starts.
4. Enter the following command:

   `CYBAGENT`

   The agent starts running.

Change the Password for the Agent User Profile

For better security, we strongly recommend that you change the password for the agent user profile after the agent installation completes. The profile must remain enabled for the agent to function.

To change the password for the agent user profile

1. Log on to the i5/OS system using a user profile that has SECOFR authorities.
2. Change the password for the agent user profile using the appropriate procedure for your system.

How to Upgrade Existing Agents

To preserve your existing settings, you can upgrade an existing agent using the interactive installation program or using the command-based silent installer. If you are upgrading multiple agents, the silent installer may save you time.

Note: You can upgrade an r7.0 agent to r11.3.

To upgrade an existing agent, follow these steps:

1. Convert the existing agentparm.txt file to r11.3 using one of these methods:
   - Convert the agentparm.txt file using an interactive program (see page 37).
   - Convert the agent parm.txt file using a silent installer (see page 37).
2. (Optional) Upgrade the agent to r11.3 by configuring the agentparm.txt file for the following features:
   - JMX connector (see page 79)
   - SNMP connector (see page 80)
   - Strong encryption (see page 86)
   - Encryption standard FIPS 140-2 (see page 89)

Convert an Existing agentparm.txt File Using an Interactive Program

You can convert your r7.0 agentparm.txt file to r11.3 to preserve your existing settings. You can use the interactive installation program to do the conversion.

To convert an existing agentparm.txt file using the interactive program

1. Run the interactive installation program (see page 26).

   The agent installation program opens.

2. Accept the license agreement and enter the required information until you are prompted for the AgentParm File Conversion.
3. Select Yes.
4. Continue to the next prompt.
5. Enter the path to your existing agentparm.txt file.
6. Complete the installation.

   The agentparm.txt file is converted to r11.3.

Convert an Existing agentparm.txt File Using the Silent Installer

You can convert your r7.0 agentparm.txt file to r11.3 to preserve your existing settings. You can use the silent installer to do the conversion.

To convert an existing agentparm.txt file using the silent installer

1. Copy the r11.3 installer.properties file to your agent computer. The file is available on the product CD or CA Support Online website, found at http://ca.com/support.
2. Open the installer.properties file you copied.
3. Disable the following property by adding a comment (#) character:

   ```
   #AGENTPARM_CONVERT_2 =No
   ```

4. Enable the following property by removing the comment (#) character:

   ```
   AGENTPARM_CONVERT_1 =Yes
   ```

   Enabling this property preserves the agentparm.txt settings of your existing agent.

5. Enable and edit the following property to specify the path to the agentparm.txt file for your existing agent:

   ```
   #OLD_AGENT_PARM=C:\\Program Files\\Cybermation\\ESP System Agent R7\\agentparm.txt
   ```

   For example, if you have a Release 6 ESP System Agent installed in C:\\Program Files\\Cybermation\\ESP System Agent R6\\agentparm.txt, edit the property as follows:

   ```
   OLD_AGENT_PARM=C:\\Program Files\\Cybermation\\ESP System Agent R6\\agentparm.txt
   ```

6. Ensure all other properties are disabled by adding the comment (#) character to each property that is uncommented.
7. Save the file.

   The properties are set in the installer.properties file.

8. Run the silent installer (see page 34).
9. Check the generated log file (see page 34).

How to Remove the Agent

You may want to remove an agent when you no longer require it.

To remove the agent, follow these steps:

1. Uninstall the agent (see page 38).
2. Remove the agent from the scheduling manager.

   For detailed instructions to remove the agent from the scheduling manager, refer to the documentation for your scheduling manager.

Uninstall the Agent

You might want to uninstall the agent if you have upgraded it from a previous release or if you want to remove the agent from your system.

To uninstall the agent

1. Ensure all workload is complete.
2. Stop the agent (see page 44).
3. Ensure the current library is not the library that contains the agent objects and the current path is not the path to the agent installation directory.

4. Enter the following command from the i5/OS command line:

   ENDSBS *libraryname*

   libraryname

   Specifies the name of the library that contains the required agent objects.

   The library stops.

5. Open a PASE terminal session (see page 41).

6. Enter the following command from the PASE command line:

   `rm -rf agentdir`

   agentdir

   Specifies the directory that contains the agent program files.

   The directory is deleted.

7. Enter the following command from the i5/OS command line:

   `CLRLIB libraryname`

   The library contents are deleted.

8. Enter the following command:

   `DLTLIB libraryname`

   The library is deleted, and the agent is uninstalled.

Chapter 4: Controlling the Agent

This section contains the following topics:

Open a PASE terminal session

Some configuration scripts and utilities must be run in the Portable Application Solutions Environment (PASE) on the i5/OS operating system. You can issue commands in PASE to control the agent.

To open a PASE terminal session

1. Log on to the i5/OS system.
2. From the i5/OS command line, enter the following command:

   ```
   call qp2term
   ```

   A PASE terminal session opens.

Note: Alternatively, you can use an x-terminal session to access PASE. Do not use Qshell to operate the agent.

Starting the Agent

To start the agent, follow these steps:

1. Start the subsystem that runs the agent jobs using one of these methods:
   - Start the subsystem from the i5/OS command line (see page 42).
   - Start the subsystem from PASE (see page 43).
2. Start the agent using one of these methods:
   - Start the agent from the i5/OS command line (see page 43).
   - Start the agent from PASE (see page 44).

Start the Subsystem from the i5/OS Command Line

To start the subsystem from the i5/OS command line

From the i5/OS command line, enter the following command:

STRSBS SBSD(*libraryname/subsystemname*)

libraryname

Specifies the name of the library that contains the required agent objects.

subsystemname

Specifies the name of the subsystem that runs the agent jobs.

Default: The name of the library that contains the required agent objects.

The subsystem that runs the agent jobs starts.

Note: After the subsystem has started, if you make a change that requires you to restart the agent, you do not need to restart the subsystem unless it has stopped.

Start the Subsystem from PASE

To start the subsystem from PASE

1. Open a PASE terminal session (see page 41).
2. Enter the following command:

   `system "strsbs` *`libraryname/subsystemname`*`"`

   libraryname

   Specifies the name of the library that contains the required agent objects.

   subsystemname

   Specifies the name of the subsystem that runs the agent jobs.

   Default: The name of the library that contains the required agent objects.

   The subsystem that runs the agent jobs starts.

Start the Agent from the i5/OS Command Line

To start the agent from the i5/OS command line

From the i5/OS command line, enter the following command:

`libraryname``/CYBAGENT`

libraryname

Specifies the name of the library that contains the required agent objects.

The agent starts running.

Note: You can add this item to your system IPL Start-Up program. For more information about changing the IPL Start-Up program, see the IBM documentation.

Start the Agent from PASE

Note: Before you start the agent, ensure the cybAgent process and related Java processes from the previous run of the agent were shut down correctly.

To start the agent from PASE

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command:

   ```
   ./cybAgent
   ```

   The agent starts running.

Stopping the Agent

You can stop the agent using several methods:

- from the i5/OS system
- from CA WA server

If you stop the agent while it is processing workload, the agent shuts down but all workload continues running. However, the agent cannot track the workload states.

Stop the Agent from the i5/OS Command Line

To stop the agent from the i5/OS command line

Enter the following command:

libraryname/CYBAGENT OPTION(SHUTDOWN)

libraryname

Specifies the name of the library that contains the required agent objects.

The agent stops running.

Stop the Agent from PASE

To stop the agent from PASE

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command:

   ```
   ./cybAgent -s
   ```

   The agent stops running.

Verifying the Status of the Agent

If there are problems running workload using the agent, you can verify whether the agent is running or has stopped. You can verify the status in the following ways:

- Verify the status file—The status file shows whether the agent is running or a controlled shutdown has occurred.
- Verify the agent process—The agent process status can verify whether the agent is down, which is helpful when a controlled shutdown did not occur and the status file does not show the correct status.

Check the status File from the i5/OS Command Line

The status file describes the status of the agent core. To verify the agent is running or a controlled shutdown has occurred, check the status file.

To check the status file from the i5/OS command line

1. From the i5/OS command line, enter the following command:

   ```
   WRKLNK 'installpath'
   ```

   installpath

   Specifies the path to the agent installation directory.

   A list of files in the specified directory appears.
2. Enter the display option next to the status file name.

   The contents of the status file appear.

Example: Check the status File

The following response shows the agent is running. The number - 1507 in this example is the process ID (PID) for the agent process.

```
CA Workload Automation Agent for OS/400 PowerPC R11.3, Build 20091215
Started at:
OS component - 1507
```

The following response indicates the agent is not running.

```
Inactive
```

The agent was shut down in a controlled manner as shown in the default_Agent.log.

```
CybAgentDriver - CybAgentDriver terminated
```

Check the status File from PASE

The status file describes the status of the agent core. To verify the agent is running or a controlled shutdown has occurred, check the status file.

To check the status file from PASE

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command:

   ```
   cat status
   ```

   The contents of the status file appear.

Example: Check the status file

The following response shows the agent is running. The number - 1507 in this example is the process ID (PID) for the agent process.

```
CA Workload Automation Agent for OS/400 PowerPC R11.3, Build 20091215
Started at:
OS component - 1507
```

The following response indicates the agent is not running.

```
Inactive
```

The agent was shut down in a controlled manner as shown in the default_Agent.log.

```
CybAgentDriver - CybAgentDriver terminated
```

Check the Agent Process Status from the i5/OS Command Line

If the agent is down and a controlled shutdown did not occur, you can check the agent process to verify the status.

To check the agent process status from the i5/OS command line

1. Enter the following command from the i5/OS command line:

   `WRKACTJOB SBS(`*`subsystemname`*`)`

   subsystemname

   Specifies the name of the subsystem that runs the agent jobs.

   Default: The name of the library that contains the required agent objects.

   A list of jobs running in the agent subsystem is displayed.

2. Enter the display option next to cybAgent.bin.

   The status of the agent appears.

Check the Agent Process Status from PASE

If the agent is down and a controlled shutdown did not occur, you can check the agent process to verify the status.

To check the process status of an agent from PASE

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command:

   ```
   # ps -ef | grep PID
   ```

   PID

   Specifies the process ID (PID) of the agent.

   The status of the specified process appears.

Example: Check the Process Status Using a PID

Suppose that the agent has a PID of 13214. Enter the following command to check the process status:

```
# ps -ef | grep 13214
```

A sample response is

```
cybuser 13214 12216  0  Apr 16 ?    0:00  ./cybAgent
```

Example: Check the Process Status of Multiple Agents or an Agent with an Unknown PID

Suppose that you want to check the process status of an agent with the name cybAgent. Enter the following command:

```
# ps -ef | grep cybAgent
```

The statuses of all processes that contain cybAgent in the name appear.

Chapter 5: Configuring the Agent

This section contains the following topics:

How to Configure Agent Parameters

You configure agent parameters by editing the agentparm.txt file, located in the agent installation directory. When you install the agent, the installation program adds frequently-configured agent parameters to the file. Other agent parameters exist, which you must manually add to the agentparm.txt file to configure the agent. For any configuration changes to take effect, always stop and restart the agent. For some agent parameters, such as the agent name and communication parameters, also configure the parameters on the scheduling manager.

To configure agent parameters, do the following:

1. Configure agent parameters on the agent (see page 49).
2. Configure agent parameters on the scheduling manager (see page 50).

Configure Agent Parameters on the Agent

You may need to change an agent parameter, for example, the agentname parameter to rename the agent.

To configure agent parameters on the agent

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command to stop the agent:

```
./cybAgent -s
```

4. Open the agentparm.txt file located in the agent installation directory.
5. Edit the file to make the required changes.
6. Save and close the agentparm.txt file.
7. Start the subsystem that runs the agent if it has stopped (see page 43).
8. Enter the following command to start the agent:

   ```
   ./cybAgent
   ```

   The agent starts running and the parameters are configured on the agent.

Configure Agent Parameters on the Scheduling Manager

When you change an agent parameter in the agentparm.txt file that is also defined on the scheduling manager, such as the agent name, configure the agent parameter on the scheduling manager.

Note: For detailed instructions to configure agent parameters on the scheduling manager, see the documentation for your scheduling manager.

Agent Parameters in the agentparm.txt File

The agent parameters below appear in the agentparm.txt file in the order listed. The parameter values are set during the agent installation. You can modify these parameters as required. To configure the agent for additional functions, follow the procedures in this chapter.

log.level

Specifies the type and number of logs the agent generates. This parameter is important for troubleshooting.

- 0, 1, or 2—Creates logs for any errors including the receiver and transmitter logs. Level 2 is adequate for production, unless problems arise requiring more details for troubleshooting.
- 3—Adds queues. If this value is specified, the agent ignores the log.maxsize parameter.
- 4 or 5—Adds debugging information. Use log level 5 for setup and initial testing.
- 6-8—Adds tracing information to diagnose a problem. These levels are not intended for continuous use.

Default: 5

Example: log.level=2

log.archive

Defines the log archiving options:

- 0—Appends current date and time to the log file.
- 1—Renames to logfile.archive and starts a new file.
- 2—Removes current file.
- 3—Appends new log entries to the current logs.

Default: 0

log.maxsize=*maximum size*[B|K|M|G]

Specifies the maximum log size. When the log file exceeds the specified size, the agent archives it and starts a new log file. If the log.archive parameter is set to three, the agent ignores this parameter. The agent does not create an archive file, but it does append all logs. You can specify the following optional modifiers:

- B—Specifies the size in bytes.
- K—Specifies the size in kilobytes.
- M—Specifies the size in megabytes.
- G—Specifies the size in gigabytes.

Note: The default (no modifier) size is in bytes.

Limits: 2G

Default: 1M

agentname

Defines the agent name. You need the agent name when you configure the scheduling manager to work with the agent.

Default: AGENT

Limits: Up to 16 characters; the first character must be a letter

Notes:

- Agent names must begin with an alphabetic character and can contain any alphanumeric characters and the special characters @, $, and underscore (_). Because the scheduling manager uses agent names as file names, use standard file-naming conventions for your operating system.
- For CA Workload Automation DE, the agent name must be in uppercase.

os400.product.library

Specifies the library in which the native i5/OS objects are installed

Default: CAWAGNT113

communication.inputport

Specifies the main port number the agent uses to listen for incoming messages from the scheduling manager. You need this port when you configure the scheduling manager to work with the agent.

Default: 7520

Limits: 1024-65534

Note: On UNIX, ports 1–1023 are reserved ports and require root access.

communication.receiver.socket.main

Specifies the type of socket the agent uses for its main port. The value of this parameter must be different from the communication.receiver.socket.aux parameter. You can specify the following socket types:

- plain
- dylan

Default: plain

Notes:

- CA Workload Automation DE does not require this parameter.
- i5/OS systems use plain socket types only.

communication.managerid_*n*

Specifies the name of the scheduling manager instance that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured.

Default: CENTRAL_MANAGER

Example: MYSERVER

communication.manageraddress_*n=address 1;...;address_m*

Specifies the address of the scheduling manager that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the IP address in the connection details for the scheduling manager. You can specify a list of addresses for the scheduling manager.

Example: 172.24.36.107 (IPv4) or 0:0:0:0:0:FFFF:192.168.00.00 (IPv6)

Notes:

- You can specify a DNS name instead of the IP address for the scheduling manager. However, your agent computer must be able to resolve the DNS name at all times. If there is a DNS outage and your agent computer cannot resolve DNS names, the agent cannot communicate with the scheduling manager.
- If the scheduling manager address never changes, enter the DNS name for the scheduling manager in the hosts file for your agent computer. This entry helps ensure that the IP address can be resolved after DNS disruptions.

communication.managerport_*n*

Specifies the port that the scheduling manager listens on for communication from agents, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the port number in the connection details for the scheduling manager.

Default: 7507

Limits: 1024-65534

communication.monitorobject_*n*

Specifies the monitor object for the scheduling manager that is used in agent alive ping.

communication.socket_*n*

Defines the socket type the agent and scheduling manager use for communication, where *n* is an integer starting at one that corresponds to the scheduling manager being configured. The following socket types are available:

- plain
- dylan

Default: plain

Note: i5/OS systems use plain socket types only.

security.filename

Specifies the path to the security file that contains the security rules that define local security on the agent.

Default: *agentinstallDir*/security.txt

security.level

Specifies whether local security on the agent is enabled or disabled. Local security on the agent controls which scheduling manager user IDs can perform certain actions, for example, which user IDs can issue CONTROL messages to the agent. If you enable local security, define security rules in a security.txt file.

security.cryptkey

Specifies the path to the text file that stores the encryption key for the agent.

Default: /CA/WA_Agent_R11_3/cyrptkey.txt

initiators.class_n=*jobclass*,*number of initiators*

Describes job classes and the number of initiators that can process jobs that are assigned a particular job class. Use a new line for each initiators.class_*n* parameter, where *n* is an integer starting at the value 1. By controlling the type and number of initiators, you can have greater control over the initiation of jobs and manually balance the loads on system resources.

The parameter initiators.afmjobclassmap_*n* relates to this parameter. However, the value of *n* does not have to match in both parameters.

For UNIX workload, depending on the number of initiators you assign, you may need to increase the number of threads that can be run per process on your operating system.

Examples:

initiators.class_1=Default,1000

initiators.class_2=POJO,100

core.health.monitor.enable

Specifies whether resource usage information within the Java Virtual Machine (JVM), such as memory usage and threads information, should be logged.

- false—Does not log resource usage information within the JVM.
- true—Logs the resource usage information within the JVM to a file called simple_health_monitor.log in the log folder.

Default: true

Note: The log.level parameter must be set to 5 or greater for this information to be logged.

spooldir

Specifies the path to the spool file directory.

Default: spool subdirectory of the agent installation directory

oscomponent.javapath

Specifies the full path to the directory where Java resides.

oscomponent.jvm

Specifies the Java virtual machine (JVM) to use.

plugins.start_internal_*n*

Specifies the agent plug-in to start by the core Java agent.

n

Denotes an integer assigned to the agent plug-in, starting at 1. The *n* suffix must increase sequentially for each agent plug-in.

oscomponent.classpath

Specifies the path to jar files required by the agent.

management.snmp.mibfile

Specifies the path to the MIB file that describes the metrics and SNMP traps for the agent.

Default: *agentinstalldir*/cybermation.mib

management.snmp.host

Identifies the SNMP Manager IP address or DNS name. Your SNMP administrator can provide the host name.

management.snmp.port

Specifies the SNMP Manager UDP port. Your SNMP administrator can provide this port number.

Default: 162

Limits: 1-65535

management.snmp.community

Specifies the type of network the SNMP traps are sent across for SNMP v1 or v2 only. Your SNMP administrator can provide the type.

- public—Identifies an unsecured network, for example, the Internet.
- private—Identifies a secure network, for example, a local area network.

Default: public

ftp.noserver

Specifies whether the agent FTP server is enabled or disabled. If ftp.noserver is set to false, the FTP server is enabled. If the ftp.noserver is set to true, the FTP server is disabled.

Default: true

ftp.serverport

Specifies the port number for the agent to act as an FTP server.

Default: 21

Limits: 1-65534

ftp.client.ssl

Specifies whether all FTP jobs on the agent computer automatically use SSL communication.

- false—Disables SSL communication
- true—Enables SSL communication

ftp.client.ssl.truststore

Specifies the full path name of the truststore file. The default file name is cacerts. You can use keytool, provided with the JRE, to create your own truststore.

ftp.client.ssl.truststore.password

Specifies the encrypted password for the client truststore file, for example cacerts, that contains some common CA X509 certificates.

Default: changeit (encrypted)

Note: You can use the agent password utility to encrypt your password before using it in the agentparm.txt file.

ftp.server.ssl

Specifies that the FTP server handles both non-SSL and SSL FTP.

ftp.server.ssl.keystore.password

Specifies the encrypted password for the server keystore that contains an X509 certificate. This password is sent to the client during the handshake process.

Default: cyberuser (encrypted)

management.connector_*n*

Identifies the type of management connector the agent uses to connect to an external application, where *n* is an integer starting from 1. You can specify the following types of connectors:

- JMX—Specifies a JMX connector, built into the agent, that lets you use a JMX console to monitor and control the agent.
- SNMP—Specifies an SNMP connector, built into the agent, that lets you use an SNMP manager to monitor and control the agent.

management.jmx.port

Specifies the port where the JMX connector listens.

Default: 1099

Limits: 1-65534

oscomponent.loginshell

Indicates how to invoke the Shell program when executing a script.

- false—The agent ignores the shell as a login shell.
- true—The agent invokes the shell as a login shell if you specify true. The shell program looks in the directory specified by the HOME environment variable and tries to execute the user's login scripts (in addition to the .cshrc script).

Default: false

Note: For most systems, this parameter affects only the C and Korn shells. The Bourne shell ignores the oscomponent.loginshell parameter.

oscomponent.defaultshell

Identifies the shell in which scripts are run on the system.

Default: /bin/sh

oscomponent.validshell

Identifies the full path and name of every shell that is valid for use on the agent. Separate each shell with a comma.

Default: /usr/bin/sh,/bin/csh,/bin/ksh,/bin/sh,/bin/bash

Note: This parameter is checked when the oscomponent.checkvalidshell parameter is set to true (the default). If the shell used in a job definition or script is not specified in this parameter, the job fails.

oscomponent.checkvalidshell

Determines whether the agent checks valid shells.

- false—The agent bypasses the valid shell check.
- true—The agent checks valid shells. All shells that jobs use must be specified in the oscomponent.validshell parameter.

Default: true

oscomponent.lookupcommand

Determines how to specify the script or command name to run in a job definition.

- false—The full path to the script or command name must be specified in the job definition.
- true—The script or command name can be specified without the full path in a job definition. The agent looks up the path to the script or command name for the specified user ID.

Default: true

Note: If set to true, ensure that the agent on UNIX is running under the root account.

oscomponent.joblog

Sets whether the agent creates a job log for each UNIX job that runs on the i5/OS system.

- false—Disables job logs
- true—Enables job logs

Default: false

Note: These job logs are different than the job logs the i5/OS system creates.

Configure Communication with a Scheduling Manager

You can change your scheduling manager connection information or add a connection to a different scheduling manager. If you are using the agent with two scheduling managers that require different socket types for communication, you can specify a main and auxiliary socket for the agent.

To configure communication with a scheduling manager, configure the following agent parameters on the agent:

communication.inputport

Specifies the main port number the agent uses to listen for incoming messages from the scheduling manager. You need this port when you configure the scheduling manager to work with the agent.

Default: 7520

Limits: 1024-65534

communication.inputport.aux

Optional. Specifies the auxiliary port number the agent uses to listen for incoming messages from the scheduling manager.

communication.manageraddress_*n*

Specifies the address of the scheduling manager that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the IP address in the connection details for the scheduling manager. You can specify a list of addresses for the scheduling manager.

Example: 172.24.36.107 (IPv4) or 0:0:0:0:0:FFFF:192.168.00.00 (IPv6)

Notes:

- You can specify a DNS name instead of the IP address for the scheduling manager. However, your agent computer must be able to resolve the DNS name at all times. If there is a DNS outage and your agent computer cannot resolve DNS names, the agent cannot communicate with the scheduling manager.
- If the scheduling manager address never changes, enter the DNS name for the scheduling manager in the hosts file for your agent computer. This entry helps ensure that the IP address can be resolved after DNS disruptions.

communication.managerid_*n*

Specifies the name of the scheduling manager instance that the agent works with, where *n* is an integer that corresponds to the scheduling manager being configured.

Default: CENTRAL_MANAGER

Example: MYSERVER

communication.managerport_*n*

Specifies the port that the scheduling manager listens on for communication from agents, where *n* is an integer that corresponds to the scheduling manager being configured. This value corresponds to the port number in the connection details for the scheduling manager.

Default: 7507

Limits: 1024-65534

communication.monitorobject_*n*

Specifies the monitor object for the scheduling manager that is used in agent alive ping.

communication.receiver.socket.aux

Specifies the type of socket the agent uses for its auxiliary port. The value of this parameter must be different from the communication.receiver.socket.main parameter. You can specify the following socket types:

- plain
- dylan

Note: CA Workload Automation DE does not require this parameter.

communication.receiver.socket.main

Specifies the type of socket the agent uses for its main port. The value of this parameter must be different from the communication.receiver.socket.aux parameter. You can specify the following socket types:

- plain
- dylan

Default: plain

Note: CA Workload Automation DE does not require this parameter.

communication.socket_*n*

Defines the socket type the agent and scheduling manager use for communication, where *n* is an integer starting at one that corresponds to the scheduling manager being configured. The following socket types are available:

- plain
- dylan

Default: plain

Note: CA Workload Automation DE does not require this parameter.

Note: You can configure the agent to work with multiple scheduling managers by adding additional definitions in the agentparm.txt file.

Example: Configure the Agent to Communicate with a Scheduling Manager

In this example, the following configuration parameters are set in the agentparm.txt file for a scheduling manager running under the instance “ACE” at address 130.200.146.134. The scheduling manager listens for incoming messages from the agent on port 49155:

```
communication.inputport=7520
communication.managerid_1=ACE
communication.manageraddress_1=130.200.146.134
communication.managerport_1=49155
communication.monitorobject_1=CAEWA_AGENT/AGENTMON1.0/MAIN
communication.receiver.socket.main=plain
communication.socket_1=plain
```

Configure the Agent for Internet Protocol Version 6 (IPV6) Communication

If your scheduling manager uses Internet Protocol Version 6 (IPV6), configure the agent for this protocol also.

To configure the agent for IPV6 communication, configure the following agent parameters on the agent:

```
java.net.preferIPv6Addresses=true
java.net.preferIPv4Stack=false
```

More information:

Configure Agent Parameters on the Agent (see page 49)

Specify Job Classes and Number of Initiators

The initiator.class parameter sets the maximum number of active jobs of a particular class allowed by the agent. By default, the agent allows up to 1000 jobs of the default class at a given time. You can further control jobs the agent allows by setting up additional initiator classes and indicating which job types they control. For example you can have an i5 class allowing only 100 active i5/OS jobs at the same time.

Note: Depending on the number of initiators you assign, you might need to increase the number of threads that can run per process on your operating system.

To specify job classes and number of initiators, edit the following parameters in the agentparm.txt file and restart the agent:

initiators.class_*n*

Describes job classes and the number of initiators that can process jobs that are assigned a particular job class. Each pair of <jobclass, number of initiators> should be typed on a different line, where *n* is an integer starting at the value 1. By controlling the type and number of initiators, you can have greater control over the initiation of jobs and manually balance the loads on system resources.

The parameter initiators.afmjobclassmap_*n* relates to this parameter. However, that parameter's value of *n* does not relate to this parameter.

For UNIX workload, depending on the number of initiators you assign, you may need to increase the number of threads that can be run per process on your operating system.

Examples:

initiators.class_1=Default,1000

initiators.class_2=POJO,100

initiators.afmjobclassmap_*n*

Maps verb and subverb combinations of a job request to a job class. When the agent sees an AFM containing a defined pair of verb and subverb, it will assign the specified job class to that job.

The defined pair must be a valid verb and subverb combination. You must write a separate instance of this parameter for each pair. For some job types, you can also specify a job class in a job definition.

Example: Setting the Job Class and Number of Initiators

Suppose that you want to limit the number of i5/OS jobs the agent initiates to 100. In this example, the job class is defined as i5. The agent receiver log below contains the verb and subverb: RUN and '.' (dot).

```
OS400AGNT CENTRAL_MANAGER TEST/ I5OS_NATIVE.2/MAIN RUN . Data(CCExit=*SEVERITY)
Command(CHGCURLIB) Parameters('QGPL')
```

Note: The verb and subverb in the agent receiver log follow the job ID, I5OS_NATIVE.2/MAIN.

To set the job class and number of initiators, you configure the following parameters to the values shown.

```
initiator.class_1=Default,1000
initiator.class_2=i5,100
initiators.afmjobclass_1=RUN,'.',i5
```

More information:

Configure Agent Parameters on the Agent (see page 49)

Define a Default User ID

To run a job under a user ID, you usually specify the user ID in the job definition. You can also define a default user ID in the agentparm.txt file so that all jobs on the agent computer run under the default user ID. If another user ID is specified in the job definition, the agent ignores the default user value and the job runs under the user ID specified in the job definition.

To define a default user ID

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command to stop the agent:

   ```
   ./cybAgent -s
   ```
4. Open the agentparm.txt file.
5. Enter a value for the following parameter:

   oscomponent.default.user

   Specifies the default user ID.

6. Save and close the agentparm.txt file.
7. Start the subsystem that runs the agent if it has stopped (see page 43).
8. Enter the following command:

   ```
   ./cybAgent
   ```

   The agent starts running. The default user ID is defined.

Note: Unlike user IDs you specify in job definitions, you do not need to define the default user ID on on the scheduling manager. However, if the same user ID is used in a job definition, you must define the user ID on the scheduling manager.

How to Set Up Environment Variables

You can define environment variables on the agent that jobs submitted by the agent can access. You can define the following types of environment variables:

- Agent-wide variables that are available for every job from every scheduling manager on behalf of every user.
- Manager-specific variables that are available for every job from a specific scheduling manager on behalf of every user. These variables override agent-wide variables.
- User-specific variables that are available for every job from a specific user. These variables override agent-wide variables and manager-specific variables.

To set up environment variables, do the following:

1. Define environment variables in a file (see page 63).
2. Specify the path to the environment variables (see page 64).

Defining Environment Variables in a File

You define each environment variable on a single line in a text file as a variable=value pair. You must create a different file for each type of environment variable. For example, you might create a file named agent_vars.txt to store all your agent-wide environment variables and a file named mgr_stress_vars.txt to store all your manager-wide environment variables.

Example: Defining Environment Variables

A text file contains the following two manager-specific environment variables required for CA Workload Automation AE.

```
AUTOSYS=C:\Program Files\CA\EWA\autosys
AUTOROOT=C:\Program Files\CA\EWA
```

Using the $EWAGLOBALPROFILE environment variable for UNIX Workload

For agents running on UNIX, you can specify the $EWAGLOBALPROFILE environment variable to specify the path to a file or script that defines global variables.

Note: To use the $EWAGLOBALPROFILE environment variable, run the agent as root and set the following parameters in the agentparm.txt file:

- oscomponent.loginshell=true
- oscomponent.lookupcommand=true

Example: Using the $EWAGLOBALPROFILE environment variable

In this example, the $EWAGLOBALPROFILE environment variable is included in a file that defines environment variables. $EWAGLOBALPROFILE specifies the path to the var_ewa_global.txt UNIX script.

```
EWAGLOBALPROFILE=u1/envar/var_ewa_global.txt
```

Specify the Path to the Environment Variables

After you have defined your agent-wide, manager-specific, or user-specific environment variables in separate text files, configure the agent to specify the location of the files.

To specify the path to the text files, configure the following agent parameters on the agent:

Note: If you omit the path, the agent uses the profiles/filename subdirectory of the agent installation directory as the default path.

- To specify the path to agent-wide variables, configure the following parameter:

  oscomponent.environment.variable

  Specifies the path to the file that defines agent-wide variables.

  Example: C:\MyVars\agent_vars.txt

- To specify the path to manager-specific variables, configure the following parameter:

  oscomponent.environment.variable_manager_*managerid*

  Specifies the path to the file that defines manager-specific variables.

  managerid

  Specifies the name of the specific scheduling manager the environment variables apply to.

  Example: C:\MyVars\mgr_stress_vars.txt

- To specify the path to user-specific variables, configure the following parameter:

  oscomponent.environment.variable_user_*userid*

  Specifies the path to the file that defines user-specific variables.

  userid

  Specifies the name of the user the environment variables apply to.

  Example: C:\MyVars\usr_abc.txt

Set PAM Parameters for User Authentication on UNIX Systems

PAM (Pluggable Authentication Modules) is used for security to check whether a service should be used. A service is a program that provides a function that requires authentication. Examples of services are login, sshd, pam, and sudo.

To set PAM parameters for user verification on UNIX systems, configure the following agent parameters on the agent:

oscomponent.auth.pam.svc

Specifies the default PAM service the agent uses for login authentication. The list of available PAM services for your system is located in the /etc/pam.conf or /etc/pam.d/ file.

Default: login

Note: You can use the chkusr utility provided with the agent to test a PAM service being used to authenticate a user and password.

oscomponent.auth.pam.lib

Optional. Specifies the path to the PAM shared library.

Note: We recommend that you specify the full path to the library file.

How to Set Up Wake On LAN (WOL)

You can save energy using the Wake on LAN (WOL) feature to automate the startup of your computers. Setting up WOL lets you define and schedule WOL jobs to send a signal to a server to turn it on. When the server is no longer needed, you can schedule a different job to power it down.

Wake on LAN (WOL) is a hardware and software solution that lets you wake up a computer remotely. The solution requires an ACPI-compliant computer and a special software program that sends a signal to the network card of the computer to wake it up. The agent provides the AMD magic packet to broadcast the signal to a computer that has been soft-powered-down (ACPI D3-warm state). You can configure the agent for how many times it broadcasts the signal and the amount of time it waits between broadcasts.

Note: Not all scheduling managers support Wake on LAN. Consult the *Release Notes* for your scheduling manager to determine whether this enhancement is supported.

To set up Wake on LAN (WOL)

1. Collect the MAC address (see page 66).
2. Configure WOL properties on the agent (see page 66).
3. Define a WOL job (see page 68).

Collecting the MAC Address

You require the Media Access Control (MAC) address of the computer you want to receive the Wake on LAN (WOL) signal. The MAC address is burned into the Ethernet card (NIC) of the motherboard.

Configure WOL Properties on the Agent

To run Wake on LAN (WOL) jobs, you must define specific parameters to enable communication. You can set up how often the agent sends the magic packet to the WOL-enabled computer.

To configure WOL properties on the agent

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command to stop the agent:

```
./cybAgent -s
```

4. Open the agentparm.txt file located in the agent installation directory.
5. Set the following parameter:

   `plugin.start_internal_N=management`
6. Define the following parameters:

   management.wol.nudges

   Specifies the number of times the agent broadcasts the magic packet.

   Default: 10

   management.wol.nudges.sleep

   Specifies the amount of time, in milliseconds, between broadcasts of the magic packet.

   Default: 1000 (ms)

   management.wol.ports

   Specifies the port that the magic packet will be sent to.

   Default: 6
7. Save and close the agentparm.txt file.
8. Start the subsystem that runs the agent if it has stopped (see page 43).
9. Enter the following command to start the agent:

   `./cybAgent`

   The agent is configured for WOL jobs.

Defining a WOL Job

To define a WOL job, you require the following information:

- The broadcast address where the packet must be broadcasted
- The target computer MAC address. Represented in a '-' or ':' separated list of six octets (bytes) in hexadecimal format.
- The optional IP address to which the agent attempts the connection to verify that the target host is up.
- The optional list of ports to which the agent attempts the connection to verify that the target host is up.

  Defaults: 21, 22, 23, 80, 111, 135, 139, 445
- The space or comma-separated list of ports. Optional. In case none are specified, 0 is assumed.
- The WOL password. Must be 4 or 6 '.', '-', or ':' separated octets (bytes) in hexadecimal format.

For detailed instructions to define a WOL job, see the documentation for your scheduling manager.

Configure the Agent to Run File Triggers as Separate Processes

You can configure the agent to run file triggers as separate processes, which lets the agent do the following:

- Scan files using a specified user ID on a UNIX system, letting the agent scan NFS file systems that the local root user does not have access to.
- Resolve file names to be scanned on UNIX.

To configure the agent to run file triggers as separate processes

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Stop the agent.
4. Open the agentparm.txt file.
5. Set the following parameters:

   ```
   oscomponent.loginshell=true
   oscomponent.checkvalidshell=true
   oscomponent.lookupcommand=true
   filemonplugin.runexternal=true
   ```

   These settings let the agent start as an external process and communicate using SysV IPC (UNIX/Linux) queues.

6. Specify the following parameter to set a default user for submitting jobs:

   oscomponent.default.user

   Specifies the default operating system user name.

   Note: The user specified in a job definition overrides this value.

7. Save and close the agentparm.txt file.
8. Start the subsystem that runs the agent if it has stopped (see page 43).

   The agent submits each file trigger as a separate process.

Important! When the filemonplugin.runexternal parameter is set to true, the user name in an i5/OS job definition must be eight characters or less instead of ten characters. If the user name exceeds eight characters, you may receive an error ("User does not exist in the system") even though the user does exist. This user name restriction is an i5/OS system limitation.

Enable Operating System Reporting in the Agent Status

You can enable operating system errors corresponding to the script or binary exit codes sent from the agent to the scheduling manager.

To enable operating system reporting in the agent status, configure the following agent parameter on the agent:

oscomponent.lookuposerror

Enables operating system errors to pass from the agent to the scheduling manager.

- false—Disables the operating system errors.
- true—Enables the operating system errors.

Default: false

Chapter 6: Configuring the Agent as an SNMP Manager

This section contains the following topics:

Configure the Agent as an SNMP Manager

You can configure an SNMP agent plug-in, packaged with the agent, to act as an SNMP manager to emit and listen for SNMP traps. The SNMP agent plug-in supports SNMP V1, V2, and V3. Once configured, users can define and run SNMP job types.

Note: Not all scheduling managers support the SNMP manager functionality. Consult the *Release Notes* for your scheduling manager to determine whether this enhancement is supported.

To configure the agent as an SNMP manager, configure the following agent parameters on the agent:

snmp.response.translate

Sets whether the agent translates Object Identifiers (OIDs).

- false—Disables translation.
- true—Enables translation.

Default: false

snmp.response.translate.full

Sets whether the agent translates the Object Identifiers (OIDs) from the numeric format to the string format.

- false—Disables full-name translation.
- true—Enables full-name translation.

Default: false

snmp.request.timeout

Defines the time-out, in milliseconds (ms), when the agent requests SNMP trap information.

Default: 2000 (ms)

snmp.request.retries

Defines the maximum number of times the agent requests SNMP trap information. Zero indicates one attempt.

Default: 0

More information:

Configure Agent Parameters on the Agent (see page 49)

Configure the SNMP Trap Listener for SNMP Subscribe Jobs

To configure the SNMP trap listener, configure the following agent parameters on the agent:

snmp.trap.listener.version

Specifies the SNMP version of the SNMP manager you want the agent to connect with.

- 1—Specifies SNMP v1.
- 2—Specifies SNMP v2.
- 3—Specifies SNMP v3.

Default: 2

snmp.trap.listener.host

Specifies the IP address of the agent listening for trap information.

snmp.trap.listener.port

Specifies the agent port listening for trap information.

Default: 162

Limits: 1-65535

snmp.trap.listener.community

Specifies the v1 or v2 SNMP trap community. The SNMP trap listener ignores traps that do not match this community type.

Default: public

snmp.trap.listener.v3.auth.password_*n*

Specifies the encrypted authentication password for the SNMP v3 user, where *n* is an integer starting from 1.

Note: All parameters ending with the same value of n belong to the same group. This parameter applies only to SNMP v3.

snmp.trap.listener.v3.auth.protocol_*n*

Specifies the authentication protocol of the SNMP trap listener, where *n* is an integer starting from 1.

- MD5—Specifies the Message Digest 5 Algorithm.
- SHA—Specifies the Secure Hash Algorithm.

Note: All parameters ending with the same value of n belong to the same group. This parameter applies only to SNMP v3.

snmp.trap.listener.v3.engine_*n*

Specifies the agent engine ID that sends trap information, where *n* is an integer starting from 1. All parameters ending with the same value of n belong to the same group. This parameter applies only to SNMP v3.

Default: AGENT_ENGINE

snmp.trap.listener.v3.priv.password_*n*

Specifies the encrypted privacy password for the SNMP v3 user, where *n* is an integer starting from 1.

Note: All parameters ending with the same value of n belong to the same group. This parameter applies only to SNMP v3.

snmp.trap.listener.v3.priv.protocol_*n*

Specifies the privacy protocol for the SNMP v3 user, where *n* is an integer starting from 1.

- AES—Specifies the Advanced Encryption Standard that uses a 32-character encryption key. AES is the algorithm required by U.S. Government organizations to protect sensitive (unclassified) information (FIPS-140-2 compliance).
- DES—Specifies the Data Encryption Standard that uses a 16-character encryption key.

Note: All parameters ending with the same value of n belong to the same group. This parameter applies only to SNMP v3.

snmp.trap.listener.v3.user_*n*

Specifies the user authorized to communicate with the SNMP v3 agent, where *n* is an integer starting from 1.

Note: All parameters ending with the same value of n belong to the same group. This parameter applies only to SNMP v3.

Chapter 7: Configuring Agent Aliases for Clustered Environments

This section contains the following topics:

How to Configure Agent Aliases for Clustered Environments

If a node fails or is down for maintenance, cluster management software migrates application packages from the inactive node to an active node in the cluster. The agent aliases let the agent accept and respond to automated framework messages (AFMs) for migrated packages. The packages are necessary to continue workload processing when the node where the workload was running experiences failover.

The scheduling manager administrator configures each application package as an alias agent in the server. This configuration lets the scheduling manager redirect AFMs to the appropriate node where the package is currently running. If the agent on the node the package was migrated to is aware of the package through the alias, when failover occurs, the agent can respond to AFMs for the migrated package.

Although the agent responds to AFMs for one of its aliases, it does not respond to the scheduling manager as the agent on whose behalf the agent responded. In all communications with the scheduling manager, the agent correctly identifies itself using the agentname parameter in the agentparm.txt file.

You can use the agent aliases in clustered environments, such as HACMP/6000 for IBM AIX, MC/ServiceGuard on HP-UX and Linux, and VERITAS Cluster Service on Windows.

To configure agent aliases for clustered environments, follow these steps:

1. Enable aliasing on the agent (see page 76).
2. Enable the agent aliasing on the scheduling manager (see page 76).

Enable Aliasing on the Agent

Agent aliasing is part of the process you can use to set up failover for the agent.

To enable aliasing on the agent, do the following:

1. Install the agent on a partition mounted locally on each node of the cluster.
2. Verify that each agent has its own copy of the agentparm.txt file, log directory, and log files. These files must reside on a locally mounted partition.
3. Configure the spooldir parameter in the agentparm.txt file to help ensure that the agents of the cluster share a common spool directory.

   Note: Sharing a common spool directory ensures that, when failover occurs and a workload object restarts on another node, the agent can retrieve the spool file and continue updating it as required.
4. Configure the communication.alias_n parameter on the agent of each node to enable each agent to respond to AFMs for all alias agents in the cluster.

Example: Configure Agent Aliases for a Clustered Environment

The following example shows how to configure the agentparm.txt file in a two-node clustered environment with two application packages.

Parameter	Node A	Node B
Agentname	AGENTA	AGENTB
Communication.alias_1	PKG1	PKG1
Communication.alsias_2	PKG2	PKG2
Persistence.coldstart	FALSE	FALSE
Spooldir	shareddisk/dir/spool	shareddisk/dir/spool

Enable the Agent Aliasing on the Scheduling Manager

Once you set up aliasing on the agent, enable the agent aliasing on the scheduling manager.

To enable the agent aliasing on the scheduling manager

1. Configure each application package as an alias agent in addition to configuring each agent on the scheduling manager.
2. Select Keep alive only for the physical agent, not for the alias agents you configured for application packages.

Considerations for Alias-Enabled Agents in Clustered Environments

When working with alias-enabled agents in clustered environments, consider the following:

- If you shut down the agent in an alias-enabled clustered environment for maintenance, the agent resumes monitoring as usual when you restart it.
- In job definitions, schedulers must refer to aliased agents, not physical agents.
- To avoid error messages when you restart an alias-enabled agent in a clustered environment, always use a cold start.

Chapter 8: Connecting the Agent to External Applications

The agent has built-in management connectors that let third-party tools monitor and control the agent.

This section contains the following topics:

Configure the Agent to Connect with a JMX Console

A JMX connector, built into the agent, lets you use a JMX console to monitor and control the agent. You can use any JMX console that implements JSR-160 to perform the following tasks on the agent:

- Discover metrics
- Query and modify values of various metrics
- Discover and invoke various functions
- Discover, subscribe, and receive notifications

To configure the agent to connect to a JMX console, configure the following agent parameters on the agent:

management.connector_*n*=jmx

Identifies the type of management connector the agent uses to connect to an external application, where *n* is an integer starting from 1.

Specify jmx to allow a JMX console to monitor and control the agent.

management.jmx.host

Specifies the host name or IP address where the JMX connector listens.

management.jmx.port

Specifies the port where the JMX connector listens.

Default: 1099

Configure the Agent to Connect with an SNMP Manager

An SNMP connector, built into the agent, lets you use an SNMP manager to monitor and control the agent. You can use any SNMP manager that supports SNMP v1, v2, or v3 to perform the following tasks on the agent:

- Discover metrics
- Query and modify values of various metrics
- Subscribe and receive notifications through SNMP traps

The cybermation.mib file, located in the agent installation directory, is a Management Information Base (MIB) file that describes all the metrics and SNMP traps for the agent.

To configure the agent to connect to an SNMP manager, configure the following agent parameters on the agent:

management.connector_*n*=snmp

Identifies the type of management connector the agent uses to connect to an external application, where *n* is an integer starting from 1.

Specify snmp to use an SNMP manager to monitor and control the agent.

management.snmp.agent.version

Specifies the SNMP version of the SNMP manager you want the agent to connect with.

- 1—Specifies SNMP v1.
- 2—Specifies SNMP v2.
- 3—Specifies SNMP v3.

Default: 2

management.snmp.mibfile

Specifies the path to the MIB file that describes the metrics and SNMP traps for the agent.

Default: *agentinstalldir*/cybermation.mib

management.snmp.host

Identifies the SNMP Manager IP address or DNS name. Your SNMP administrator can provide the host name.

management.snmp.port

Specifies the SNMP Manager UDP port. Your SNMP administrator can provide this port number.

Default: 162

Limits: 1-65535

management.snmp.community

Specifies the type of network the SNMP traps are sent across for SNMP v1 or v2 only. Your SNMP administrator can provide the type.

- public—Identifies an unsecured network, for example, the Internet.
- private—Identifies a secure network, for example, a local area network.

Default: public

management.snmp.agent.community.read

Specifies the SNMP read community. This parameter applies only to SNMP v1 and v2.

- public—Specifies read-only access.
- private—Specifies read/write access.

management.snmp.agent.community.write

Specifies the SNMP write community. This parameter applies only to SNMP v1 and v2.

- public—Specifies read-only access.
- private—Specifies read/write access.

management.snmp.agent.trapsink.host

Specifies the host name or the IP address of the SNMP listener that receives trap information. The management connector uses this host to send the trap.

management.snmp.agent.trapsink.port

Specifies the port of the SNMP listener that receives trap information. The management connector uses this port to send the trap.

Default: 162

management.snmp.agent.trapsink.community

Specifies the SNMP community that receives trap information.

- public—Specifies read-only access.
- private—Specifies read/write access.

Default: public

management.snmp.agent.trapsink.user

Specifies the user authorized to receive trap information.

Configure Connection with a Version 3 SNMP Manager

To configure the agent to connect with a Version 3 SNMP Manager, configure the following agent parameters on the agent:

management.snmp.agent.user

Specifies the user authorized to communicate with the SNMP agent plug-in.

Example: MBAGENT

management.snmp.agent.user.auth.protocol

Specifies the authentication protocol the agent uses. Supported protocols are SHA and MD5.

Example: SHA

management.snmp.agent.user.auth.password

Specifies the encrypted authentication password for the user authorized to communicate with the SNMP agent plug-in.

management.snmp.agent.user.priv.protocol

Specifies the privacy protocol the agent uses. Supported protocols are AES and DES.

Example: AES

management.snmp.agent.user.priv.password

Specifies the encrypted privacy password for the user authorized to communicate with the SNMP agent plug-in.

Chapter 9: Setting Up Security

This section contains the following topics:

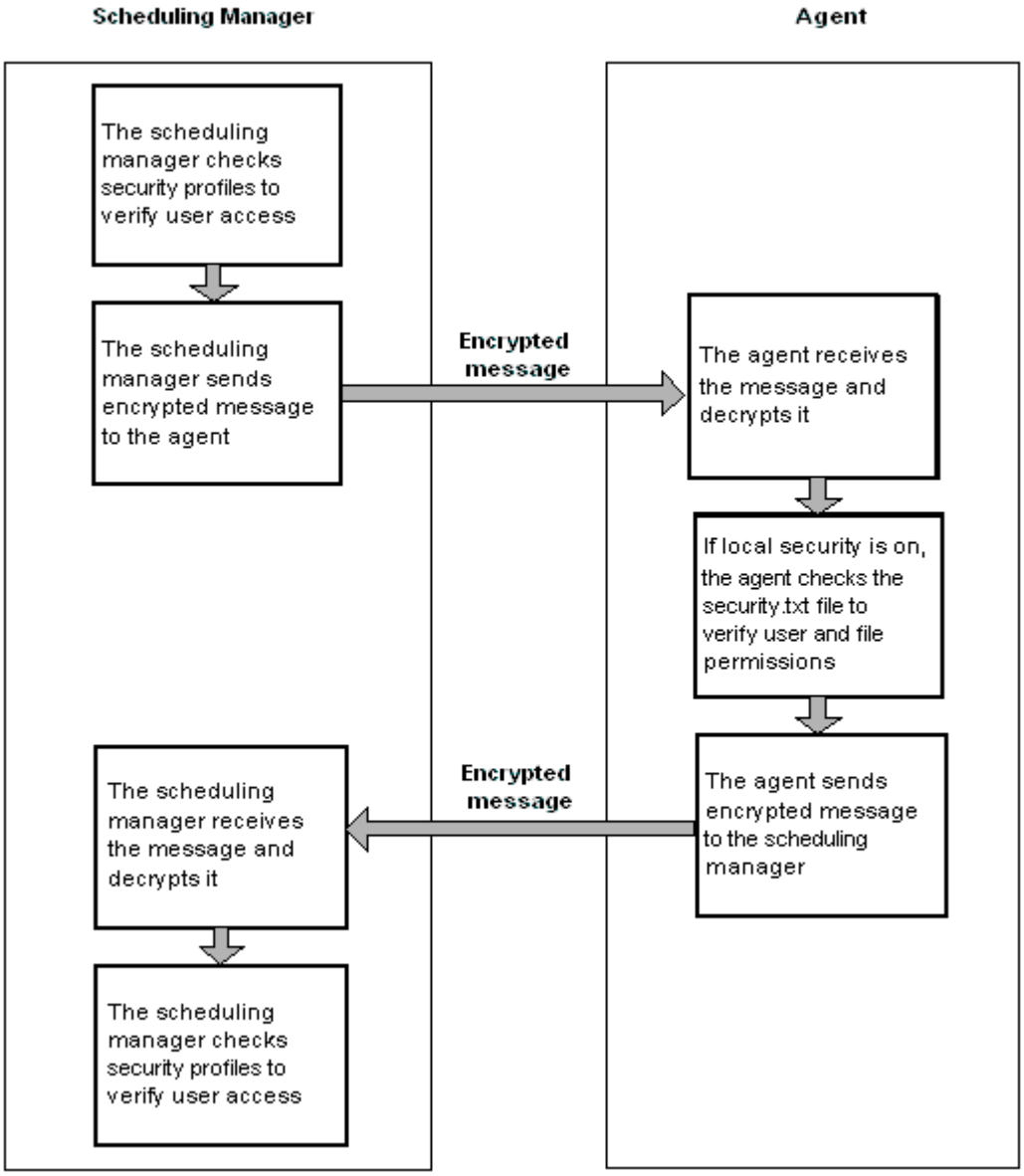
Types of Security

At a minimum, security is set between the agent and the scheduling manager using an encryption key. The agent requires encrypted communication with the scheduling manager. The encryption key is set when you install the agent and when you configure the scheduling manager to work with the agent.

You can also set up local security on the agent to control the following:

- Which scheduling manager user IDs can submit jobs under a specific agent user ID, from a specific directory
- Which FTP user IDs can issue FTP-related commands to files in directories
- Which scheduling manager user IDs can issue control commands and send messages to an agent

Example: Security between the agent and the scheduling manager



The scheduling manager verifies the access privileges of the user it has even before sending workload to the agent. The scheduling manager also verifies security when receiving messages from the agent. The agent also has its own security verifications it performs when it receives instructions from the scheduling manager.

How to Set Up Security between the Agent and the Scheduling Manager

Encryption is a mandatory security feature that safeguards communication between the agent and the scheduling manager. Your scheduling manager administrator must complete configuration tasks so that the agent and the scheduling manager can communicate with message encryption.

To set up security between the agent and the scheduling manager, follow these steps:

1. Set up security permissions on the scheduling manager (see page 85).
2. Set the encryption on the agent (see page 86).
3. Set the encryption key on the scheduling manager (see page 88).
4. Restart the agent (see page 89).
5. Run a test job to test the security.

   For detailed instructions to run a test job, see the documentation for your scheduling manager.

Security Permissions on the Scheduling Manager

Your scheduling manager administrator must set up the following security permissions on the scheduling manager to control agent access:

- Permission to run work on the agent
- Permission to run a job on the agent under a user ID
- Permission for the agent to issue control commands

Note: For more information about security permissions, see the documentation for your scheduling manager.

Set the Encryption on the Agent Using the Keygen Utility

You can install the agent with one of four types of encryption: AES, Blowfish, DES, or DESEDE. The encryption key is specified during the agent installation, but you can change it any time using this procedure.

The keygen utility provided with the agent lets you encrypt a key. By default, the encryption key is stored in the cryptkey.txt file located in the agent installation directory. You can replace the encryption key in this file or specify a different file to store it.

Note: Make a note of the encryption key as you will need to set the same value on the scheduling manager.

To set the encryption on the agent using the keygen utility

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command at the command prompt:

   keygen 0x*key cipher destination*

   key

   Defines the encryption key the agent uses to communicate with the scheduling manager. The encryption key must be prefixed with 0x and followed by the number of characters required for the chosen cipher algorithm:

   - AES—32 hexadecimal character encryption key.

     Note: If you omit the 0x prefix, the keygen utility interprets the inputted value as a 16-character passphrase and not as a hexadecimal number. If you enter less than 16 characters, the keygen utility appends the passphrase with spaces for the missing number of characters. The keygen utility will internally encode the 16 character passphrase into a 32 hexadecimal character AES encryption key.

   - Blowfish—32-64 even-numbered hexadecimal character encryption key
   - DES—16 hexadecimal character encryption key
   - DESEDE—48 hexadecimal character encryption key

   Limits: 16-64 alphanumeric characters (any digits and letters A-F only)

cipher

Specifies the type of cipher algorithm the agent uses to encrypt and decrypt messages sent to the scheduling manager. The agent supports the following types:

- AES—Advanced Encryption Standard that uses a 32-character encryption key. AES is the algorithm required by U.S. Government organizations to protect sensitive (unclassified) information (FIPS-140-2 compliance).
- BLOWFISH—A license-free encryption algorithm that uses an encryption key of 32 to 64 even-numbered characters.
- DES—Data Encryption Standard that uses a 16-character encryption key.
- DESEDE—Triple Data Encryption Algorithm that applies the DES algorithm three times to each data block.

Default: DES

destination

(Optional). Specifies the name of a text file that stores the encryption key.

Default: cryptkey.txt

Note: If you specify a new text file, you must update the security.cryptkey parameter in the agentparm.txt file.

The encryption key is replaced by the keygen utility.

Example: Encrypt a Key

This example encrypts the key 0x1020304050607080 for 16-character (DES) encryption.

```
keygen 0x1020304050607080 DES
```

Disable Encryption on the Agent

You can configure the agent for no encryption.

To disable encryption on the agent

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Stop the agent.
4. Open the agentparm.txt file located in the agent installation directory.
5. Set the security.cryptkey parameter to no value, as follows:

   ```
   security.cryptkey=
   ```
6. Save and close the agentparm.txt file.
7. Start the subsystem that runs the agent if it has stopped (see page 43).
8. Start the agent.

   The encryption is disabled on the agent.

Set the Encryption Key on the Scheduling Manager

The scheduling manager and the agent must have the same encryption key to communicate. The encryption key for the agent is stored in a text file. The security.cyrptkey parameter in the agentparm.txt file sets the path to the text file. After you set the encryption key on the agent, set the same key on the scheduling manager. If the keys are different, the agent and scheduling manager cannot communicate and an AGENTDOWN state occurs when you try to run workload.

Note: For detailed instructions to set the encryption key on the scheduling manager, see the documentation for your scheduling manager.

Restart the Agent

After you have set up 256-bit encryption on the agent and enabled 256-bit encryption on the scheduling manager, you must restart the agent to complete the configuration.

To restart the agent

1. Open a PASE terminal session (see page 41).
2. Ensure you are in the agent installation directory.
3. Enter the following command to stop the agent:

   ```
   ./cybAgent -s
   ```

4. Start the subsystem that runs the agent if it has stopped (see page 43).
5. Enter the following command:

   ```
   ./cybAgent
   ```

   The agent restarts.

Configure the Agent for Encryption Standard FIPS 140-2

The U.S. Government encryption standard FIPS 140-2 requires a FIPS-certified library and FIPS-certified cipher algorithm. To comply with the standard, the agent provides the following:

- RSA BSAFE Crypto-J library
- Advanced Encryption Standard (AES) cipher algorithm

If you did not select the AES cipher algorithm when you installed the agent, you can configure the agent to comply with encryption standard FIPS 140-2.

To configure the agent for encryption standard FIPS 140-2

1. Change to the agent installation directory.
2. Stop the agent.
3. Open the agentparm.txt file.
4. Edit the following parameter to specify the encryption key:

   security.cryptkey

   Defines the encryption key the agent uses to communicate with the scheduling manager. The encryption key must be prefixed with 0x and followed by the number of characters required for the chosen cipher algorithm:

   - AES—32-character encryption key
   - DESEDE—48-hexadecimal encryption key

   Note: Encrypt the encryption key using the keygen utility.
5. Set the following parameter for the agent to use the FIPS-certified library and cipher algorithm.

   ```
   security.jce.fips=true
   ```

   Note: Setting this parameter can impact workload that uses SSL, for example, FTP jobs where the servers do not use the same cipher suites.
6. Save and close the agentparm.txt file.
7. Start the agent.

More information:

Set the Encryption on the Agent Using the Keygen Utility (see page 86)
Configure Agent Parameters on the Agent (see page 49)

How to Set Up Local Security on the Agent

The agent has its own security verification it performs when it receives instructions from the scheduling manager. Security rules on the agent define the local security verification. Local security on the agent controls which scheduling manager user IDs can perform the following actions:

- Submit jobs run under a specific agent user ID.
- Issue CONTROL messages to the agent.
- Perform FTP transfers under a specific agent user ID.

Note: Agent security rules do not override permissions set at the operating system level.

To set up local security on the agent, follow these steps:

1. Enable local security (see page 91).
2. Configure the security.txt file (see page 92).
3. Refresh the security.txt file (see page 99).

Enable Local Security

Local security must be enabled for the agent to perform its own security checks.

To enable local security

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Stop the agent.
4. Open the agentparm.txt file located in the agent installation directory.
5. Define the following parameter:

   `security.level=`**`on`**
6. Save and close the agentparm.txt file.
7. Start the subsystem that runs the agent if it has stopped (see page 43).
8. Start the agent.

   Local security is enabled on the agent.

Configure the security.txt File

The security.txt file contains the rules that allow or deny scheduling manager user IDs the authority to issue control commands to the agent.

Note: If the security.txt file does not exist, default security rules apply.

To configure the security.txt file

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Stop the agent.
4. Open the security.txt file, or create one if it does not exist.

   Note: The security.txt file must reside in the agent installation directory.
5. Define security file rules in the security.txt file (see page 93).
6. Save and close the file.
7. Start the subsystem that runs the agent if it has stopped (see page 43).
8. Start the agent.

   Security rules are set for the agent.

More information:

Additional Formats for Security File Rules (see page 97)

Security File Rules

The security file contains three types of rules: x, f, and c type rules as described below:

`x a | d` *`CAWA_userID agent_userID path`*

Defines a rule that allows or denies scheduling manager user IDs from submitting jobs that run under a specific user ID. This type of rule also controls access to native i5/OS objects and specific directories. These rules begin with the letter x.

x

Identifies a rule controlling execution of programs and commands.

a | d

Specifies whether access is allowed or denied.

- a indicates permission is allowed.
- d indicates permission is denied.

CAWA_userID

Defines the scheduling manager's Manager name or the scheduling manager user ID this rule applies to.

agent_userID

Defines the user ID on the agent computer under which the job runs.

path

Defines the IFS path that the scheduling manager is allowed to submit jobs from, using the user ID identified by a*gent_userID*. Paths are case sensitive.

`f a | d` *`FTP_userID operation path`*

Defines a rule that allows or denies FTP user IDs from issuing FTP-related commands to files in specified directories or members of i5/OS file objects. These rules begin with the letter f.

f

Identifies FTP commands.

a | d

Specifies whether access is allowed or denied.

- a indicates permission is allowed.
- d indicates permission is denied.

FTP_userID

Defines the FTP user ID this rule applies to.

operation

Specifies the FTP command. Valid commands are

- list—Changes directory and list files (CD, LIST, NLST)
- read—Retrieves the file (RETR)
- write—Stores the file or makes a directory (STOR, STOU, RNFR, RNTO, MKD)
- delete—Deletes the file or directory (DELE, RMD)

The above commands apply to the agent as FTP server. For FTP jobs, only read and write apply.

path

Specifies the path that the scheduling manager is allowed to submit jobs from, using the user ID identified by Agent_UserID. Paths are case sensitive.

`c a | d CAWA_userID CONTROL command`

Defines a rule that allows or denies scheduling manager user IDs the authority to issue control commands to the agent. These rules begin with the letter c.

c

Identifies a rule controlling operational commands to an agent.

a | d

Specifies whether access is allowed or denied.

- a indicates permission is allowed.
- d indicates permission is denied.

CAWA_UserID

Defines the scheduling manager's Manager name or the scheduling manager user ID this rule applies to.

command

Specifies the control command. Valid commands are: shutdown, refresh, and clrfiles. You can also specify an asterisk (*) for all commands.

Note:

- Specify at least one rule of each type (x, f, and c) in the security.txt file.
- If security.txt does not exist, default security rules apply.
- Agent security rules do not override permissions set at the operating system level.
- To specify an f rule that restricts access to a directory itself (not the contents in the directory), the directory path must end with a forward slash.

Example: CA WA (x) Security File Rules

The following rule allows any scheduling manager user to submit jobs that use any i5/OS object or files from any directory. The user can submit the jobs under any user ID on the agent computer.

```
x a * * +
```

The following rule denies any scheduling manager user from submitting jobs that use any i5/OS object or files from any directory under the QSECOFR profile on the agent computer.

```
x d * QSECOFR +
```

The following rule allows any scheduling manager user to submit jobs that use any object in any library on the system. The user can submit the jobs under any user profile that starts with the characters ADMIN on the agent computer.

```
x a * ADMIN /QSYS.LIB/+
```

The following rule allows any scheduling manager user to submit jobs that use any object whose name starts with F in the USER library. The user can submit the jobs under any user profile that starts with the characters USR on the agent computer. However, members of file objects whose names start with F are excluded.

```
x a * USR* /QSYS.LIB/USER.LIB/F*
```

The following rule allows any scheduling manager user to submit jobs that use any object whose name starts with F in the USER library. The user can submit the jobs under any user profile that starts with the characters JO on the agent computer. Members of file objects whose names start with F are included.

```
x a * JO* /QSYS.LIB/USER.LIB/F*
```

The following rule denies any scheduling manager user from submitting jobs that use /QSYS.LIB/MLIB.LIB/DEPT.FILE/ PAYROLL.MBR on the agent computer.

```
x d * * /QSYS.LIB/MLIB.LIB/DEPT.FILE/PAYROLL.MBR
```

Example: FTP (f) Security File Rules

The following rule denies all users from using any FTP operations in any directory. To allow specific FTP access, this general rule is overridden by the FTP rules that follow.

```
f d * * +
```

The following rule allows all users to list the files in /pub/ftp and its subdirectories.

```
f a * list /pub/ftp/+
```

The following rule allows all users to store files, rename files, and make directories in /pub/ftp/upload and its subdirectories.

```
f a * write /pub/ftp/upload/+
```

The following rule allows all users to read files from /pub/ftp/download and its subdirectories.

```
f a * read /pub/ftp/download/+
```

Example: Command (c) Security File Rule

The following rule allows all users to issue control commands to the agent.

```
c a * * *
```

Additional Formats for Security File Rules

When defining security rules in the security.txt file, you can use the following additional formats:

Wildcards

The scheduling manager name, user IDs, object names, paths, verbs, and subverbs can contain a single wildcard character at the end of the value only.

The following wildcards are valid:

- Asterisk (*)-Represents zero or more character matches in the current directory only.
- Plus sign (+)-Represents zero or more character matches in the current directory and all subdirectories. For a FILE object, + applies to the members within it.

Start point and spacing

Every security rule starts in column 1. The items on a line are separated by one or more blanks or tab characters and end with a new-line character.

Comment lines

The file can contain comment lines. An asterisk (*) or a number sign (#) in column 1 identifies comment lines.

Security Rule Interpretation

For a rule to match, three components of a rule have to match. If two or more rules match, the closest match overrides the others, as follows.

Interpretation	Explanation
A specific rule overrides a generic rule. A generic rule is a rule that contains wildcards.	/u1/jsmith overrides /u1/jsmith* CYBDL01 overrides *CYB
If both rules are generic, the more specific one overrides the other.	/u1/jsmith/scripts/* overrides /u1/jsmith* /u1/jsmith/scripts/a* overrides /u1/jsmith/scripts*
The scheduling manager User ID takes precedence over the Agent User ID, and the Agent User ID takes precedence over the directory or object name.	A rule is considered a closer match if the CA WA server User ID is a closer match. If the scheduling manager User IDs of two rules are the same, the rule with the closest matching Agent User ID overrides the other.

Interpretation	Explanation
If there is still ambiguity after these rules have been applied, a deny rule overrides an allow rule.	c d USR* * *overrides c a USR* * *

How Local Security Works

This section describes how the agent determines what to validate when it receives instructions from a scheduling manager. By understanding how local security works, you can decide how to configure local security for your system.

When the agent starts, it verifies local security and does the following:

- If local security is enabled, the agent then looks for the security.txt file.
  - If the security.txt file does not exist, default security rules apply.
  - If the security.txt file exists, the agent uses the rules defined in the file. The agent does not use the default security rules. If a request does not have a match in the security file, the agent denies the request.
- If local security is not enabled, the agent does not verify security.

Default Security Rules

The agent uses the default security rules when local security is enabled, but the security.txt file does not exist. Default security rules allow all users to execute control commands.

The following rules allow all users to issue control commands, disables FTP, and disables execution of workload by any user:

c a * * *

f d * * +

x d * * +

Format for Defining an i5/OS Object in an Execution Rule

To define an i5/OS object in an execution rule, you must use the path file format only. No other naming convention is supported in the security rules. The path naming format has the form:

`/QSYS.LIB/`*`libraryname`*`.LIB/`*`objectname`*`.`*`type`*

libraryname

Defines the name of the library that contains the object. The value can be

- A specific library name
- %LIBL%—Lets the agent validate the permission against a library list that is resolved by the i5/OS system when the library is not specified in the CLPNAME, COMMAND, AS400FILE, or AS400LIB statement.

Limits: Up to 10 characters

objectname

Defines the object name.

Limits: Up to 10 characters

type

Defines the object type.

Limits: CMD, PGM, or FILE

Note: For *FILE objects, you can specify the member name as follows:

`/QSYS.LIB/`*`libraryname`*`.LIB/`*`objectname`*`.FILE/`*`membername`*`.MBR`

membername

Defines the member name for the file object.

Limits: Up to 10 characters

Refresh an Agent Security File

Refresh the agent security.txt file for any changes to take effect.

To refresh an agent security file

From the command prompt, enter the following command:

```
cybAgent -r.
```

Test the Encryption between the Agent and the Scheduling Manager

To test the encryption between the agent and the scheduling manager, run and monitor a job.

Note: For more information about defining jobs, see the documentation for the scheduling manager.

Encrypting and Changing Passwords

When you change a password, encrypt it before entering it in the agentparm.txt file. To encrypt a password, use the Password utility that is provided with the agent.

The agent handles FTP user IDs and passwords separately.

Encrypt a Password Using the Password Utility

When you change a password, you will need to encrypt it for inclusion in the agentparm.txt file. To encrypt a password, use the Password utility provided with the agent.

To encrypt a password

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory
3. Enter the following command:

   ```
   password.sh
   ```

   A command prompt appears asking you to enter your password.
4. Enter your password.

   The utility responds with your encrypted password as in the following example:

   ```
   **** PASSWORD ENCRYPTION ****
   Enter your password: Lisa
   Encrypted password: 1FF8897FCDDF8A62
   ```
5. Make note of the encrypted password as you will need to enter it in the agentparm.txt file.
6. Exit the utility.

Chapter 10: Setting Up and Running FTP Workload

This section contains the following topics:

FTP Client and FTP Server

Using your agent, you can automate FTP transfers with FTP jobs. An FTP job can use an existing FTP server or the agent as an FTP server. The FTP job always acts as an FTP client.

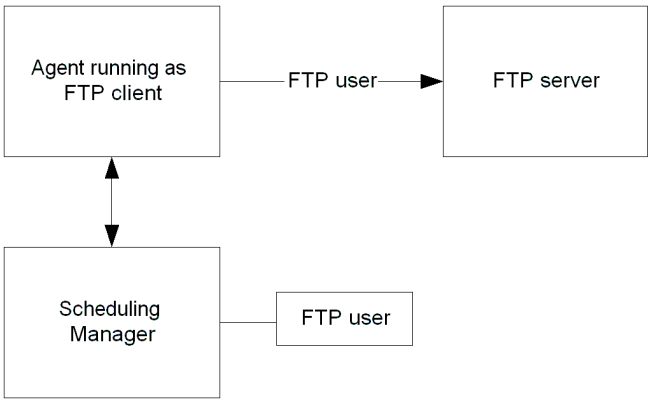
Use an FTP job to automate

- Downloading an ASCII, binary or EBCDIC file from a remote FTP server to your agent computer
- Uploading an ASCII, binary or EBCDIC file from your agent machine to a remote FTP server

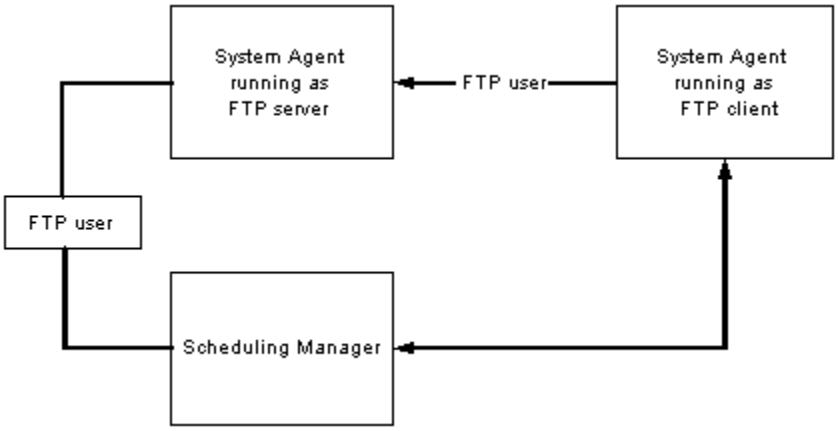
Note: For the QSYS file system on i5/OS systems, you can only transfer Save files or members of other FILE objects. For more information about FTP restrictions on i5/OS systems, see the IBM FTP documentation.

You can set up the agent to run as an FTP client, FTP server, or both.

The following diagram shows you the relationships between the agent running as an FTP client, the scheduling manager, and an FTP server.



The following diagram shows you the relationships between the agent running as an FTP server, the scheduling manager, and another agent running as an FTP client.



How to Set Up the Agent as an FTP Client

When you set up the agent as an FTP client, it can log in to remote FTP servers, download files from those servers, and upload files to those servers.

Note: When the agent runs as an FTP client only, other FTP clients (such as other agents) cannot log in to the agent to FTP files. To allow other FTP clients to log in and transfer files, you also need to set up the agent to run as an FTP server.

To set up the agent as an FTP client, follow these steps:

1. Configure the agent as an FTP client (see page 103).
2. Define FTP rules for local security on the agent (see page 106).

   This step only applies if local security is enabled on the agent.
3. Define the FTP User on the scheduling manager (see page 106).

Note: You can also set up the agent to use Secure (SSL) FTP.

Configure the Agent as an FTP Client

You can configure the agent as an FTP client after installation using the following procedure.

To configure the agent as an FTP client

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command to stop the agent:

   ```
   ./cybAgent -s
   ```

   The agent stops running.
4. Open the agentparm.txt file.
5. Remove the pound sign (#) from the following parameter:

   plugins.start_internal_*N*

   Specifies that the FTP plug-in is enabled. *N* is an integer, assigned to the plug-ins, starting at 1. The N suffix must increment sequentially for each plug-in.

   Default: plugins.start_internal_1=os400ftp

   The FTP plug-in parameter is uncommented.

6. (Optional) Define the following parameters.

ftp.ascii.ccsid

Defines the Coded Character Set Identifier (CCSID) to use for ASCII file transfers. If the file to be transferred already exists on the target computer, the file is written using the encoding of the existing file.

Default: 819

ftp.client.updatemsg

Defines the frequency interval in milliseconds in which the status information for an FTP job in EXEC state is updated.

Default: 30 000 (30 sec)

ftp.data.compression

Specifies whether data for all FTP jobs on this agent should be compressed for transfer. The value ranges from zero (0) for no compression to nine (9) for the best compression. If the compression level is also specified in the job definition, the ftp.data.compression value is ignored, and the data is compressed using the level specified in the job definition. To use FTP data compression, both FTP client and FTP server must be run by the agent software.

Default: 0 (no compression)

ftp.download.owner

Specifies a default user ID on the computer where the agent is installed. This user ID determines the access permissions of a downloaded file on the agent computer. When the file is downloaded, the file is created with this user as the file owner.

ftp.ebcdic.ccsid

Defines the Coded Character Set Identifier (CCSID) to use for EBCDIC file transfers. If the file to be transferred already exists on the target computer, the file is written using the encoding of the existing file.

Default: 37

ftp.passive

Specifies whether the agent FTP client uses a passive mode connection, as follows:

- false—The agent uses an active mode connection.
- true—The agent uses a passive mode connection.

Default: false

Note: We recommend you set the value to true under any of the following conditions: the agent uses IPV4 and the FTP server uses IPV6 for communication, the FTP server resides beyond the firewall, and/or the FTP server opens a listening port for the data channel.

7. Save and close the agentparm.txt file.
8. Start the subsystem that runs the agent if it has stopped (see page 43).
9. Enter the following command:

   ```
   ./cybAgent
   ```

   The agent starts running.

Configure the Agent FTP Client to Use Secure Copy Protocol (SCP)

You can configure the agent to act as an FTP client that uses the secure copy protocol (SCP) to transfer binary files.

To configure the agent for secure copy file transfers, configure the following agent parameters on the agent:

ftp.client.updatemsg

Specifies the status update interval in milliseconds (ms).

Default: 30000 (ms)

ftp.download.owner

Specifies a default user ID on the computer where the agent is installed. This user ID determines the access permissions of a downloaded file on the agent computer. When the file is downloaded, the file is created with this user as the file owner.

Notes:

- This parameter only applies to agents installed on UNIX systems.
- The local user ID does not require a password to be stored on the scheduling manager.

ftp.scp.sshd.timeout

Controls the timeout, in milliseconds (ms), for SCPv2.

Default: 30000 (ms)

ftp.scp.debug.enable

Sets whether debugging of the secure copy protocol (SCP) sessions is enabled. The output is stored in the ftp_scp_debug.log.

- false—Disables debugging.
- true—Enables debugging.

Default: false

More information:

Configure Agent Parameters on the Agent (see page 49)

Define FTP Rules for Local Security on the Agent

By default, the agent's security file denies all users from issuing FTP-related commands on the agent computer. To allow users to issue FTP-related commands while local security is enabled, you must define FTP rules in the security.txt file.

Note: Local security is enabled on the agent if the security.level parameter is set to on in the agentparm.txt file.

You can create a rule in the security.txt file to set a root directory for the FTP server so that FTP users can only access directories that are within this root directory.

Example: Restrict FTP access to files in a specified directory

In this example, the root directory is set to /local/pub/. The command “cd /” on the FTP client changes directories to /local/pub/. FTP users that are logged in to the FTP server can only access /local/pub/ and its subdirectories, if permitted.

```
f a * * /local/pub
```

Example: Restrict FTP access to files in a specified library

The following rule allows all users to store and rename any FILE object in the USER library whose name starts with F. Members of those file objects are included.

```
f a * write /QSYS.LIB/USER.LIB/F+
```

More information:

How to Set Up Local Security on the Agent (see page 91)

Define the FTP User on the Scheduling Manager

To use the agent as an FTP client, your scheduling manager administrator must define each FTP user on the scheduling manager.

Note: Passwords are case sensitive.

How to Set Up the Agent as an FTP Server

The agent supports a built-in FTP server capability. You can enable the agent to act as a generic FTP server in addition to its other roles. This server comes under the security rules established for the agent.

To set up the agent as an FTP server, follow these steps:

1. Configure the agent as an FTP server (see page 107).
2. Set up local security on the agent (see page 91).

   Note: Local security must be enabled (the security.level parameter in the agentparm.txt file must be set to on). If the agent runs as an FTP server, clients can log in to the agent and transfer files.
3. Define the FTP user on the agent (see page 109).

   Note: The FTP user ID used to connect to the agent running as an FTP server must be defined on that agent and the scheduling manager.

If you configure the agent as an FTP server, the agent can handle ASCII, binary, and EBCDIC file transfers, wildcard requests, simple GET and PUT requests for single files, and MGET and MPUT requests for multiple files. The agent has a secure store of FTP server user IDs and associated passwords. The ftpusers.txt file, located in the directory that contains the agent program files, stores these user IDs and their corresponding hashed passwords.

The agent running as an FTP server does not support anonymous FTP requests. For audit purposes, the agent provides a detailed log of all FTP requests.

Configure the Agent as an FTP Server

You can configure the agent as an FTP server while installing the agent or after installation. However, to set up optional FTP server features, you must modify the agentparm.txt file after installation.

To set up the agent as an FTP server

1. Open a PASE terminal session (see page 41).
2. Change to the directory that contains the agent program files.
3. Enter the following command:

   ```
   ./cybAgent -s
   ```

   The agent stops running.

4. Open the agentparm.txt file.
5. Remove the pound sign (#) from the following parameter:

   plugins.start_internal_N

   Specifies that the FTP plug-in is enabled. N is an integer, assigned to the plug-ins, starting at 1. The N suffix must increment sequentially for each plug-in.

   Default: plugins.start_internal_1=os400ftp

   The FTP plug-in parameter is uncommented.
6. Define the following parameters:

   ftp.noserver

   Specifies whether the agent FTP server is enabled or disabled. If ftp.noserver is set to false, the FTP server is enabled. If the ftp.noserver is set to true, the FTP server is disabled.

   Default: true

   ftp.serverport

   Defines the port number for the agent FTP server. If ftp.serverport is not defined and the FTP server is enabled, the operating system will try to use its default FTP port.

   Default: 1234
7. (Optional) Define the following parameters:

   ftp.ascii.ccsid

   Defines the Coded Character Set Identifier (CCSID) to use for ASCII file transfers. If the file to be transferred already exists on the target computer, the file is written using the encoding of the existing file.

   Default: 819

   ftp.ebcdic.ccsid=ccsid_for_EBCDIC

   Defines the Coded Character Set Identifier (CCSID) to use for EBCDIC file transfers. If the file to be transferred already exists on the target computer, the file is written using the encoding of the existing file.

   Default: 37
8. Save and close the file.
9. Start the subsystem that runs the agent if it has stopped (see page 43).
10. Enter the following command:

    ```
    ./cybAgent
    ```

    The agent starts running.

Set Up Local Security on the Agent

To use the agent as an FTP server, you must set up local security on the agent.

To set up local security on the agent, follow these steps:

1. Enable local security (see page 91).
2. Configure the security.txt file (see page 92).
3. Refresh the security.txt file (see page 99).

Define the FTP User on the Agent

To run FTP workload through an agent operating as an FTP server, you must define the FTP user ID and the corresponding password on the agent. The FTP user ID belongs to the user authorized to make the file transfer.

To define FTP users on the agent, run the ftpusrcfg utility located in the agent installation directory.

Note: If you set up the agent as an FTP server during installation, you defined one FTP user ID and password. Use the ftpusrcfg utility to define additional FTP users or change the password of an FTP user.

FTP Server Maintenance

The agent FTP server is a fully functional FTP server with user authentication support. To maintain the FTP server, manage the FTP users file and configure local security on the agent.

The ftpuser.txt file, located in the directory that contains the agent program files, stores FTP user IDs and passwords. The ftpusers.txt file uses one line for each entry with the user ID in the first position followed by the hashed password.

Manage FTP User IDs and Passwords

To run FTP workload on an agent operating as an FTP server, you must define the FTP user ID and password on the agent. You use the ftpusrcfg command in the PASE environment to add, delete, and change FTP user IDs and passwords. Changes made with the command update the ftpusers.txt file. You must refresh the agent security files for the changes to take effect.

To manage FTP user IDs and passwords

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.

3. Enter the following command from a command line on the same computer as the agent:

   `ftpusrcfg.sh -a|-d|-m|-l userID password`

   -a

   Adds a new user ID. Use with the userID and password parameters. Enter the userID first followed by the password.

   -d

   Deletes the specified user ID. Use with the userID parameter.

   -m

   Changes the password for the specified user ID. Use with the userID and password parameters. Enter the userID first followed by the password.

   -l

   Lists all entries in the ftpuser.txt file. The utility does not show passwords in plain text.

   userID

   Specifies the FTP user ID you want to add, change or delete.

   password

   Specifies the password corresponding to the FTP user ID. Passwords are case sensitive.

   Note: Issuing the ftpusrcfg command without a parameter displays a list of options.

4. Enter the following command:

   `./cybAgent -r`

   The agent security files are refreshed and the FTP user ID and password are changed accordingly. The ftpusrcfg command only returns a response if it detects an error. The command hashes passwords written to the ftpusers.txt file.

Example: Add a New FTP User ID

The following command adds the FTP user ID P01Prod01 with the password cyber.

```
ftpusrcfg.sh -a P01Prod01 cyber
```

Example: Change an Existing FTP User's Password

The following command changes the password for the FTP user ID P01Prod01 from cyber to r6ut09.

```
ftpusrcfg.sh -m P01Prod01 r6ut09
```

Configuring SSL FTP

To run FTP workload using Secure Sockets Layer (SSL) communication, enable and configure SSL on the FTP server and the FTP client. When you select to enable SSL FTP during the agent installation process, the installation program does the following:

- Defines default SSL server and client parameters in the agentparm.txt file.
- Adds cacerts, serverkeystore default certificates, a password, and a customized java.security file to the agent installation directory.

You can use the default certificates and settings to configure SSL FTP. You can also generate a certificate and apply your own settings to configure SSL FTP.

Configure SSL FTP Using the Default Certificates and Settings

The default certificates and settings provided during the agent installation let you set up SSL FTP without generating your own certificates and settings.

To set up SSL FTP using the default certificates and settings

1. Ensure that the following parameter is defined in the QIBM/ProdData/Java400/jdk14/lib/security/java.security file on your i5/OS operating system:

   ```
   security.overridePropertiesFile=true
   ```

   The JVM uses the customized java.security file that is installed with the agent. This value allows the JVM to use the java.security provided with the agent for its own communications, but does not affect other instances of the JVM for other applications on the i5/OS system. Without defining this property the i5/OS agent will not be able to use SSL.

2. In the SSL FTP server directory, export the certificate used by the SSL FTP server, for example

   ```
   jre\bin\keytool -export -alias agent -file key.cer -keystore serverkeystore
   ```

   You are prompted for a password. The default password is cyberuser.

   Note: You require the alias “agent” to ensure you use the certificate provided by the agent.

3. Copy the created file, in this case, key.cer, to the SSL FTP client directory if it is different from the server directory.

4. Import the created file, in this case, key.cer, into the truststore file supplied by Sun (cacerts), for example

```
D:\agent\R6SP2>jre\bin\keytool -import -file key.cer -keystore cacerts
Enter keystore password: changeit
Owner: CN=cyberuser, OU=ESP System Agent, O=Cyber, L=Markham, ST=Ont, C=CA
Issuer: CN=cyberuser, OU=ESP System Agent, O=Cyber, L=Markham, ST=Ont, C=CA
Serial number: 4152d5dc
Valid from: Thu Apr 21 09:55:40 EDT 2005 until: Mon Jun 05 09:55:40 EDT 2006
Certificate fingerprints:
MD5: 74:F2:17:20:B6:B0:10:AE:AC:88:9A:BA:AA:3A:6D:73
SHA1:4C:88:B6:39:64:65:98:AD:3E:1E:33:05:12:13:9C:4A:F4:4E:E7
:FA
Trust this certificate? [no]: yes
Certificate was added to keystore
```

5. Start the agent acting as an FTP server.

   Note: If another agent acts as an FTP client, start that agent as well.

6. Run a test upload and download job to verify the setup.

How to Configure SSL FTP Using a Generated Certificate

This process configures SSL FTP using user-generated certificates and settings.

To configure SSL FTP using a generated certificate, follow these steps:

1. Generate a server keystore (see page 113).
2. Verify the server keystore (see page 114).
3. Encrypt the server keystore password (see page 114).
4. Configure an SSL-enabled FTP server on the agent (see page 115).
5. Add the certificate to the client keystore on the agent (see page 116).
6. Configure an SSL-enabled FTP client on the agent (see page 119).

Note: After you configure SSL FTP, you can enable and disable it as required. If SSL is disabled on the FTP client after configuration and you want to run SSL FTP workload, you can specify SSL in the job definition instead.

Generate a Server Keystore Using a Generated Certificate

To configure an SSL-enabled FTP server using user-generated certificates and settings, you need to generate a keystore. You can generate your own keystore using the keytool utility provided with the JRE. The utility is located in the JRE/bin directory.

Note: Add the path to the keytool to your path variable.

To generate a server keystore

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command:

   ```
   keytool -genkey -alias agent -keystore ./serverkeystore
   ```

   If you do not assign an alias, the default alias, mykey, is used.
4. Follow the prompts.

   Note: You will need to encrypt the keystore password you enter.

   The keystore is generated.

Example: Generate a Server Keystore

The following example shows sample keytool prompts and values:

```
/home/ESP_System_Agent_R7>keytool -genkey -alias agent -keystore ./serverkeystore
Enter keystore password:  123456
What is your first and last name?
  [Unknown]:  Cyberuser
What is the name of your organizational unit?
  [Unknown]:  agent
What is the name of your organization?
  [Unknown]:  Cybermation
What is the name of your City or Locality?
  [Unknown]:  Markham
What is the name of your State or Province?
  [Unknown]:  Ontario
What is the two-letter country code for this unit?
  [Unknown]:  CA
Is CN=Cyberuser, ESPSystemAgent, O=Cybermation, L=Markham, ST=Ontario, C=CA correct?
  [no]:  yes
Enter key password for [set AGENT value for your book]
        (RETURN if same as keystore password):
```

Verify a Server Keystore

You can verify the keystore you generated using the keytool utility to ensure its accuracy.

To verify a server keystore

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command:

   ```
   keytool -list -v -keystore serverkeystore
   ```

4. Follow the prompts.

   Information about the server keystore is displayed.

Example: Verify a Server Keystore

The following example shows sample keytool prompts and values:

```
/home/ESP_System_Agent_R7>keytool -list -v -keystore serverkeystore
Enter keystore password:  123456
Keystore type: jks
Keystore provider: SUN
Your keystore contains 1 entry
Alias name: agent
Creation date: Apr 21, 2005
Entry type: keyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=Cyberuser , ESPSystemAgent, O=Cybermation, L=Markham, ST=Ontario, C=CA
Issuer: CN=Cyberuser, ESPSystemAgent, O=Cybermation, L=Markham, ST=Ontario, C=CA
Serial number: 4123a631
Valid from: Thu Apr 21 09:55:40 EDT 2005 until: Mon Jun 05 09:55:40 EDT 2006
Certificate fingerprints:
MD5:  39:D8:D9:4F:50:1C:43:A2:27:4D:50:75:32:E9:9D:40
SHA1: 98:30:54:C0:F7:4E:34:FF:DC:0A:85:D8:F7:98:D6:B7:41:7D:E7:58
```

Encrypt a Password for the Server Keystore

You require an encrypted keystore password to configure an SSL-enabled FTP server on the agent. Use the Password utility provided with the agent to encrypt the keystore password you used when you generated the keystore.

Change an SSL FTP Server Keystore Password

To change an SSL FTP server keystore password

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Enter the following command:

   ```
   ./cybAgent -s
   ```

   The agent stops running.
4. Open the agentparm.txt file.
5. Define the following parameter:

   ftp.server.ssl.keystore.password

   Specifies the new encrypted password.
6. Save and close the file.
7. Start the subsystem that runs the agent if it has stopped (see page 43).
8. Enter the following command:

   ```
   ./cybAgent
   ```

   The agent starts running. The password is changed.

Configure an SSL-enabled FTP Server on the Agent

You use the generated server keystore and its encrypted password to configure an SSL-enabled FTP server on the agent.

To configure an SSL-enabled FTP Server on the agent

1. Ensure that the following parameter is defined in the QIBM/ProdData/Java400/jdk14/lib/security/java.security file on your i5/OS operating system:

   ```
   security.overridePropertiesFile=true
   ```

   The JVM uses the customized java.security file that is installed with the agent. This value allows the JVM to use the java.security provided with the agent for its own communications but does not affect other instances of the JVM for other applications on the i5/OS system. Without defining this property the i5/OS agent will not be able to use SSL.
2. Open a PASE terminal session (see page 41).

3. Change to the agent installation directory.
4. Enter the following command:

   ```
   ./cybAgent -s
   ```

   The agent stops running.
5. Open the agentparm.txt file.
6. Set the following parameters:

   ```
   security.level=on
   ```

   ```
   ftp.noserver=false
   ```

   ```
   ftp.server.ssl=true
   ```
7. Specify the following parameters:

   ftp.server.ssl.keystore

   Specifies the full path of the keystore file. The default file name is serverkeystore. You can use keytool, provided with the JRE, to create your own keystore.

   Example: ftp.server.ssl.keystore=/R7/serverkeystore

   ftp.server.ssl.keystore.password

   Specifies the encrypted password for the server keystore that contains an X509 certificate. This password is sent to the client during the handshake process. The default password is cyberuser (encrypted).

   Note: You can use the agent password utility to encrypt your password before using it in the agentparm.txt file.
8. Save and close the agentparm.txt file.
9. Start the subsystem that runs the agent if it has stopped (see page 43).
10. Enter the following command:

    ```
    ./cybAgent
    ```

    The agent starts running and the FTP server on the agent is SSL-enabled.

How to Add a Certificate to the Client Keystore on the Agent

You add a certificate to the client keystore to configure an SSL-enabled FTP client on the agent.

To add a new certificate to the client keystore on the agent, follow these steps:

1. Export the certificate from the server keystore (see page 117).
2. Import the certificate to the client keystore on the agent (see page 118).
3. Verify the client keystore on the agent (see page 119).

Export the Certificate from the Server Keystore

You can export the certificate from the server keystore using the keytool utility provided with the JRE.

Note: Add the path to keytool to your path variable.

To export the certificate from the server keystore

1. Open a PASE terminal session.
2. Change to the directory that contains the agent program files.
3. Enter the following command:

   ```
   keytool -export -file key.cer -keystore serverkeystore
   ```

   Note: To export the certificate generated with an alias, you must include the same alias in the export command. For example, suppose a certificate was generated with the following command:

   ```
   keytool -genkey -alias agent -keystore ./serverkeystore
   ```

   To export that certificate, use the following command:

   ```
   keytool -export -alias agent -file key.cer -keystore serverkeystore
   ```

4. Follow the prompts.

   The server keystore certificate is exported.

Example: Export the Certificate from the Server Keystore

The following example shows sample keytool prompts and values:

```
/home/ESP_System_Agent_R7>keytool -export -file key.cer -keystore serverkeystore
Enter keystore password:  654321
Certificate stored in file <key.cer>
```

Import a Certificate to the Client Keystore on the Agent

You can import the server keystore certificate to the client keystore on the agent using the keytool utility provided with the JRE.

Note: Add the path to keytool to your path variable.

To import a certificate to the client keystore on the agent

1. Open a PASE terminal session.
2. Change to the directory that contains the agent program files.
3. Enter the following command:

   ```
   keytool -import -file key.cer -keystore cacerts
   ```

   To import a certificate that was exported with an alias, you must include the same alias in the import command. For example, suppose a certificate was exported with the following command:

   ```
   keytool -export -alias agent -file key.cer -keystore serverkeystore
   ```

   To import that certificate, use the following command:

   ```
   keytool -import -alias agent -file key.cer -keystore cacerts
   ```

4. Follow the prompts.

   The certificate is imported to the agent client keystore.

Example: Import a Certificate to the Client Keystore on the Agent

The following example shows sample keytool prompts and values:

```
C:\Program Files\Cybermation\ESP System Agent>keytool -import -file key.cer
-keystore cacerts
Enter keystore password:  changeit
Owner: CN=Cyberuser C, OU=ESPSystemAgent, O=r, L=g, ST=d, C=ca
Issuer: CN=Cyberuser C, OU=ESPSystemAgent, O=r, L=g, ST=d, C=ca
Serial number: 41239e39
Valid from: Thu Apr 21 09:55:40 EDT 2005 until: Mon Jun 05 09:55:40 EDT 2006
Certificate fingerprints:
MD5:  31:CC:29:0F:B6:C8:E9:3C:70:C7:6B:6C:AD:B7:00:38
SHA1:9D:86:A7:51:15:9E:B1:D3:E7:3B:59:C6:B2:E0:E0:3F:3D:C6:97:6
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

Verify a Client Keystore on the Agent

You can verify the keystore on the agent using the keytool utility provided with the JRE.

To verify a server keystore

1. Open a PASE terminal session.
2. Change to the directory that contains the agent program files.
3. Enter the following command:

   ```
   keytool -list -v -keystore cacerts
   ```

4. Follow the prompts.

   Information about the clientkeystore is displayed.

Example: Verify a Client Keystore

The following example shows sample keytool prompts and values:

```
C:\Program Files\Cybermation\ESP System Agent>keytool -list -v -keystore cacerts
Enter keystore password:  changeit
Keystore type: jks
Keystore provider: SUN
Your keystore contains 26 entries
Alias name: equifaxsecureebusinessca1
Creation date: Apr 21, 2005
Entry type: trustedCertEntry
Owner: CN=Equifax Secure eBusiness CA-1, O=Equifax Secure Inc., C=US
Issuer: CN=Equifax Secure eBusiness CA-1, O=Equifax Secure Inc., C=US
Serial number: 4
```

Configure an SSL-enabled FTP Client on the Agent

If you use the agent FTP client to connect to the SSL-enabled FTP server on the agent, you must configure the FTP client for SSL communication as well.

To configure an SSL-enabled FTP client on the agent

1. Ensure that the following parameter is defined in the QIBM/ProdData/Java400/jdk14/lib/security/java.security file on your i5/OS operating system:

   ```
   security.overridePropertiesFile=true
   ```

   The JVM uses the customized java.security file that is installed with the agent. This value allows the JVM to use the java.security provided with the agent for its own communications but does not affect other instances of the JVM for other applications on the i5/OS system. Without defining this property the i5/OS agent will not be able to use SSL.

2. Open a PASE terminal session (see page 41).
3. Change to the agent installation directory.
4. Enter the following command:

   ```
   ./cybAgent -s
   ```

   The agent stops running.
5. Open the agentparm.txt file.
6. Set the following parameter:

   ```
   ftp.client.ssl=true
   ```
7. Specify the following parameter:

   ftp.client.ssl.truststore

   Specifies the full path name of the truststore file. The default file name is cacerts. You can use keytool, provided with the JRE, to create your own truststore.
8. Save and close the agentparm.txt file.
9. Start the subsystem that runs the agent if it has stopped (see page 43).
10. Enter the following command:

    ```
    ./cybAgent
    ```

    The agent starts running and the FTP client on the agent is SSL-enabled.

Chapter 11: Maintaining Spool and Log Files

This section contains the following topics:

Spool File Maintenance

Depending on the type of workload the agent runs, the spool files are stored in and accessed from different locations. The output for native i5/OS workload is stored in spool files that the i5/OS system generates. The output for UNIX workload is stored in spool files that the agent software generates.

We recommend that you clear both types of spool files regularly to maintain storage space.

Spool Files for i5/OS Workload

For i5/OS workload, the spool files are native objects on the i5/OS system. Spool files can be created while a job executes and a spool file is created from a job's job log after the job completes. Unlike spool files for UNIX workload, i5/OS job spool files are not stored in a spool directory.

We recommend you use your routine maintenance procedures to clear spool files on the i5/OS system.

You can schedule workload using the scheduling manager and the agent to automate the clearing of the spool files.

Spool Files for UNIX Workload

For UNIX workload on the i5/OS system, the spool files are created by the agent and stored in a spool directory. By default, these spool files do not clear automatically. You should clear these spool files periodically. If the file system where a spool file resides reaches its maximum size, the agent cannot continue to run.

The agent does not limit the size of the spool files. Spool files are limited in size by the available space on the file system where they reside.

The frequency for clearing the spool files varies, depending on your installation. There are several ways to clear the spool files:

- Configure the agent to clear the UNIX spool files automatically.
- Use the clearspool and deldirifempty scripts to clear UNIX spool files automatically or manually.

Configure the Agent to Clear Spool Files Automatically

You can configure the agent to automatically clear the UNIX workload spool files by modifying the agentparm.txt file. You can also set parameters to specify a file expiration time and sleep time.

Note: The agent logs the spool-file cleanup activity in the runner_spool_cleaner.log log, located in the agent's log directory.

To configure the agent to clear spool files automatically

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Stop the agent.
4. Open the agentparm.txt file located in the agent installation directory.

5. Edit the the following parameter:

   `runnerplugin.spool.clean.enable=true`

6. (Optional) Edit the following additional parameters:

   runnerplugin.spool.expire=*n* D|H|M|S

   Specifies the file expiration time. The agent deletes spool files that are older than this value.

   n

   Specifies the time period.

   D

   Specifies the time period unit as days.

   H

   Specifies the time period unit as hours.

   M

   Specifies the time period unit as minutes.

   S

   Specifies the time period unit as seconds.

   Default: 10D (10 days)

   runnerplugin.spool.sleep=*n* D|H|M|S

   Specifies the sleep interval. At every interval, the agent checks for spool files that meet the expiration time and deletes them.

   Default: 1D (1 day)

7. Save and close the agentparm.txt file.
8. Start the subsystem that runs the agent if it has stopped (see page 43).
9. Start the agent.

   The agent is configured to clear spool files automatically.

Example: Delete Spool Files Older Than 10 Days

Suppose that you want to configure the agent to check the spool files every 36 hours and delete spool files that are older than 10 days.

Add the indicated values to the following parameters in the agentparm.txt file:

```
runnerplugin.spool.clean.enable=true
runnerplugin.spool.expire=10D
runnerplugin.spool.sleep=36H
```

The agent deletes spool files that are older than 10 days.

Example: Check Spool Files When the Sleep Interval Is Greater Than the File Expiration Time

Suppose that you want to configure the agent to check the spool files every 50 minutes and delete spool files that are older than 50 minutes as specified by runnerplugin.spool.expire.

Add the indicated values to the following parameters in the agentparm.txt file:

```
runnerplugin.spool.clean.enable=true
runnerplugin.spool.expire=50M
runnerplugin.spool.sleep=2H
```

The agent ignores the two hour sleep interval set by runnerplugin.spool.sleep.

Clear UNIX Spool Files Using Scripts

You can clear UNIX spool files periodically using the clearspool and deldirifempty (delete directory if empty) scripts.

1. Create the clearspool and deldirifempty scripts.
2. Schedule the clearspool script using CA WA server or run it manually.

   The clearspool script deletes files that meet certain modification time criteria. If the spool files are completely cleared, the clearspool script calls deldirifempty. The deldirifempty script deletes empty directories within the spool directory.

   If you run clearspool from a Telnet session, ensure you switch to the directory containing the spool files.

   If you used the defaults when installing the agent, the spool directory is called spool.

   The clearspool script assumes the spool directory is called spool in the current directory. If not, supply the full directory path name in the environment variable SPOOL.

Example: Create the clearspool and deldirifempty Scripts

The following are sample scripts. You can have other file maintenance procedures.

1. Create a script called clearspool that contains the following code:

```
#! /bin/ksh
  if [[ -z $SPOOL ]]
    then
      SPOOL=./spool
  fi
find $SPOOL -type f -mtime +n -exec rm {} \;
find $SPOOL -depth -type d -exec /bin/ksh /script_path/deldirifempty {} \;
```

mtime n

Specifies the age of the files to be deleted.

- +n — deletes files last modified more than n days.
- n — deletes files last modified exactly n days ago.
- -n — deletes files last modified less than n days ago.

Note: Put this script in the same directory as the cybAgent binary. Otherwise, specify the full path for SPOOL. You cannot specify a symbolic-linked directory for the SPOOL path.

2. Create the script called deldirifempty that contains the following code:

```
#!/bin/ksh
Dir=$(ls -A $1)
 if [[ -z $Dir ]]
  then
   echo  "deleting directory $1"
   rmdir $1
  else
   echo "$1 is not empty"
 fi
```

Example: Deleting Files Modified Yesterday or Earlier

In the following example, mtime is specified as +1 to delete files that were last modified at least one day ago. The clearspool script then calls the deldirifempty script, which deletes any empty spool subdirectories.

```
#!/bin/ksh

  if [[ -z $SPOOL ]]

    then

     SPOOL=/AgentDirectory/spool

  fi

find $SPOOL –type f –mtime +1 –exec rm {} \;

find $SPOOL –depth –type d -exec /bin/ksh  /script_path/

deldirifempty {} \;
```

Note: $SPOOL cannot be symbolic-linked directories.

Log File Maintenance

The agent keeps a set of logs that you must clear periodically to maintain disk space availability. The log files contain records of all messages between the agent and the scheduling manager, and internal messages. These files are located in the log directory by default and are updated continually while the agent is running. The types and number of logs that are generated depend on the log.level parameter set in the agentparm.txt file.

You can configure agent log file properties that control the log file size, the types and number of log files that are generated, and how the agent archives the log files. Depending on your scheduling manager, you can also clear log files manually.

Configure the Agent to Clear Log Files Automatically

The agent has a housekeeping function that automatically removes all existing files with the extension .log that reach a certain size. You can configure the agent to automatically clear the log files by modifying the agentparm.txt file.

To configure the agent to clear log files automatically

1. Open a PASE terminal session (see page 41).
2. Change to the agent installation directory.
3. Stop the agent.
4. Open the agentparm.txt file located in the agent installation directory.
5. Edit the following parameter to specify the maximum log size (in bytes).

   `log.maxsize`

   When the log file exceeds the specified size, the agent archives it and starts a new log file.
6. Edit the the following parameter to specify the log archiving options:

   `log.archive`

   Note: The agent ignores the log.maxsize value if the log.archive parameter is set to 3. The agent does not create an archive file, but appends new log entries to the current logs.
7. Edit the following parameter to specify the types of logs and number of logs to generate:

   `log.level`

   Note: Level 2 is adequate for general, initial testing, and level 0 is adequate for production unless problems arise requiring more details for troubleshooting.
8. Save and close the agentparm.txt file.
9. Start the subsystem that runs the agent if it has stopped (see page 43).
10. Start the agent.

    The agent is configured to clear log files automatically.

    Note: In some combinations of log.level and log.archive settings, a new file is generated (runner_plugin_transmitter_queue.log).

More information:

Agent Parameters in the agentparm.txt File (see page 50)

Enable or Disable Job Logs

By default, the agent creates a job log for every script or binary request that runs on the system it manages. The job log contains environment and other diagnostic information that you can use to debug failed jobs. These job logs are different than the job logs the i5/OS system creates.

To enable or disable job logs, edit the following parameter in the agentparm.txt file and restart the agent:

oscomponent.joblog

Sets whether the agent creates a job log for each job that runs.

- false—Disables job logs
- true—Enables job logs

Default: true

Note: The agent stores the job logs in the spool file directory, which you will need to clear periodically depending on the volume of your workload.

Chapter 12: Troubleshooting

This section contains the following topics:

Contacting Product Support Services

The sections in this chapter can help you perform basic troubleshooting procedures.

Review any error resolution with your i5/OS or UNIX system administrator. If this information does not help you resolve the problem, contact Product Support Services.

During Service Request investigations, Product Support Services commonly requires log files to help resolve your problem.

Collect Log Files for Agents Running on i5/OS or UNIX

This procedure makes the following assumptions:

- Your PATH environment variable defines the path to the Java bin directory.
- You know the path of the directory the agent is installed in.
- You have permissions required to traverse the directories specified in the procedure.
- The find command on your system is the standard one.

If you need the required permissions or the path to Java on your machine, see your system administrator.

This procedure writes the jar file to the /tmp directory. You can create the jar file anywhere you want.

To collect log files for agents running on i5/OS or UNIX

1. Create a temporary directory for the empty jar file. For example, if the Service Request number is 12345, type the following:
   `cat /dev/null > /tmp/sr12345_logfiles.jar`
2. Create the jar file. For example, type the following on one line:
   `find /<AgentInstallDirectory>/log -type f -mtime -2 -exec jar uvf /tmp/sr12345_logfiles.jar {} \;`

   Note: The find command above limits the amount of data included in the jar file using the -mtime switch. In the above example, -mtime -2 includes all log files whose last modified data is within the last two days.
3. FTP the jar file, using binary mode, to a machine where you can email CA.
4. Email the jar file to CA. Check that the jar file name includes the Service Request name. Identify the Service Request number on the subject line of your email.

   Note: Emails sent to CA cannot exceed 5 megabytes. If the file is greater than 5 megabytes, please contact the Product Specialist investigating your issue for assistance.

Using a Job Log to Debug a Failed Job

These job logs are different than the job logs the i5/OS system creates. The job log contains environment and other diagnostic information that you can use to debug failed jobs.

The agent stores each job log in the spool file directory using the following naming convention:

job.*hash*.joblog

job

Specifies the name of the job.

hash

Specifies the encryption key that the agent uses to encrypt messages.

Example: Using a Job Log to Debug a Failed Job

A job, running on a Windows system, fails. The agent records the following message in the job log named x.E61ADD84CD3C0864D155EEADD4EEECA6D509D23E.joblog.

```
20071125 20342154+0500 . MBAGENT X/Y/Z State SUBERROR Failed SetEnd Status(Error
creating stdin file) Cmpc(20004) JobLogId(E61ADD84CD3C0864D155EEADD4EEECA6D509D23E)
User(MBAGENT) Host(workstation)
```

More information:

Enable or Disable Job Logs (see page 128)

Agent Logs

The agent provides logging facilities to assist in testing and debugging. The logging facilities can specify logging targets, message levels, and buffering processing.

The agent supports log levels 0, 1, 2, 3, 4, and 5, where level 0 provides the least information and level 5 provides the most.

- Levels 0, 1, and 2 create logs of any errors including the receiver and transmitter logs.
- Level 3 adds queues.
- Levels 4 and 5 add debugging information.

When you install the agent, the log level is set to 0 by default.

In a standard agent installation, the agent maintains the log files in a directory called log, which resides in the agent installation directory.

Log File Structure

All log files have the following basic structure:

```
Date Time <Time Zone> <Message priority> <Thread Group>.Thread.Class.method[:line number] - <message>
```

Note: The runner_os_component.log log file has a slightly different structure.

Example: Log file structure

```
07/06/2009 10:22:32.609 EDT-0400 2 TCP/IP Controller
Plugin.Transmitter.CybTransmitter.run[:129] - Creating the processor pool[2]
```

Setting Log Levels for Troubleshooting

The log level determines the type and number of logs the agent generates and the amount of information contained in a log. To change the log level, you set the value of the log.level parameter in the agentparm.txt file. You can set the following log level values for troubleshooting:

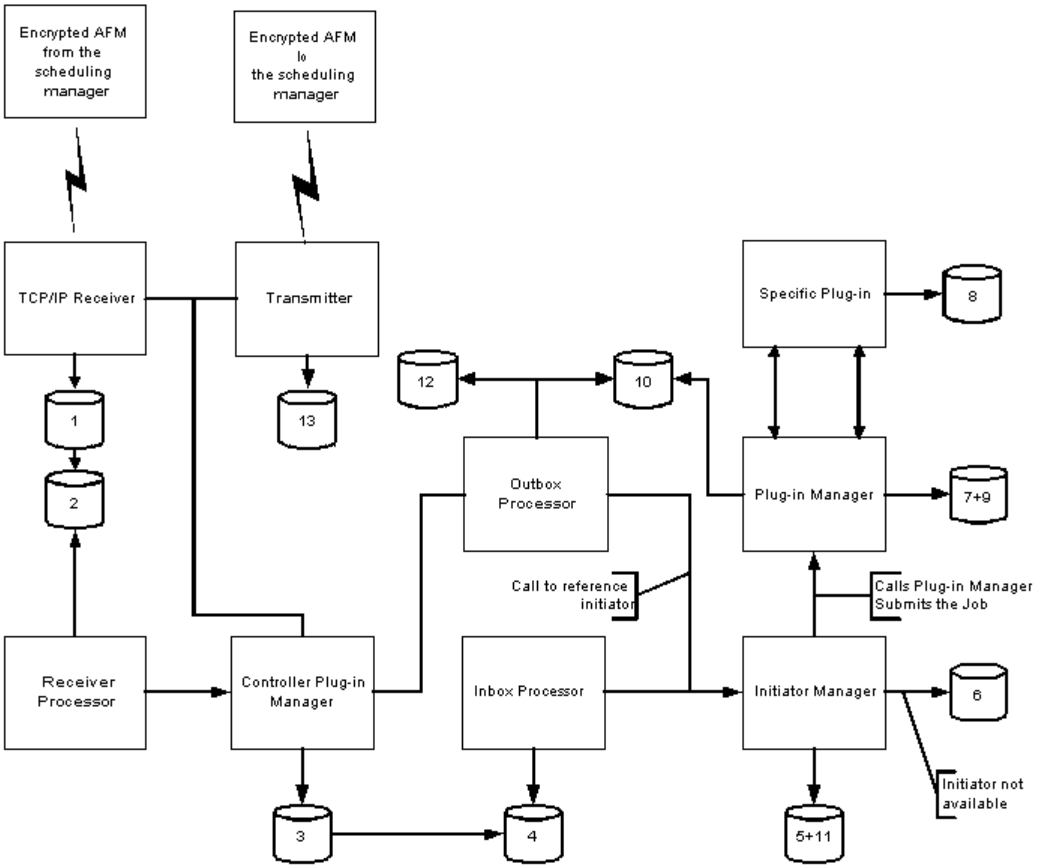
- 5—Adds debugging information. Use log level 5 for setup and initial testing, and diagnosis of problems.
- 8—Adds tracing information. Use log level 8 for troubleshooting communication problems.

Note: These levels are not intended for continuous use.

Trace an Automated Framework Message (AFM)

The agent logs the path of any AFM as it proceeds from the scheduling manager to the agent. The following dataflow diagram and tables describe how the agent logs the AFMs as it processes them.

The numbers in the data flow diagram refer to the step numbers given in the following table.



Step	Log File	Description	Log Level
1	receiver.log	Record of all successfully received AFMs.	0, 1, 2
2	queue_receiver.log	Log for the queue that holds all successfully received AFMs.	0, 1, 2
3	cybrmicontrollerpluginmanager.log	Shows TCP/IP plug-in attempts to send a message to the core.	5

Step	Log File	Description	Log Level
4	queue_inbox.log	All messages from the controller arrive here. Incoming message distributor (inbox) calls initiatormanager to process these messages.	3
5	initiatormanager.log	The initiatormanager records any exception conditions here.	5
6	initiators_waiting_<Job class>.log	If all initiators for this job class are consumed, the job is put in a queue.	3
7	rmipluginmanager.log	Logs the number of active jobs that the plug-in has.	4
8	plug-in specific ■ For runner plug-in, see Runner Plug-in AFM Processing (see page 136). ■ For file monitoring plug-in, see Filemon Plug-in AFM Processing (see page 136).	The message is sent to a plug-in; for example, runner_os_component.log. Once you have completed the trace routine in either of the two other streams, return to the next step in this stream.	-
9	rmipluginmanager.log	Shows the plug-in has attempted to send a message to the core.	4
10	queue_communicator.log	A reply is placed here.	3
11	initiatormanager.log	Shows the initiator has been released.	5
12	messagedistributoroutgoing.log	Shows the message is sent through the controller plug-in manager to the scheduling manager.	5
13	transmitter.log	Log of all sending activity and any errors discovered.	0, 1, 2

Main Stream of AFM Processing

Step	Log File	Description	Log Level
1	receiver.log	Record of all successfully received AFMs.	0, 1, 2

Step	Log File	Description	Log Level
2	queue_receiver.log	Log for the queue that holds all successfully received AFMs.	0, 1, 2
3	cybrmicontrollerpluginmanager.log	Shows TCP/IP plug-in attempts to send a message to the core.	5
4	queue_inbox.log	All messages from the controller arrive here. Incoming message distributor (inbox) calls initiatormanager to process these messages.	3
5	initiatormanager.log	The initiatormanager records any exception conditions here.	5
6	initiators_waiting_<Job class>.log	If all initiators for this job class are consumed, the job is put in a queue.	3
7	rmipluginmanager.log	Logs the number of active jobs that the plug-in has.	4
8	plug-in specific ■ For runner plug-in, see Runner Plug-in AFM Processing (see page 136). ■ For file monitoring plug-in, see Filemon Plug-in AFM Processing (see page 136).	The message is sent to a plug-in; for example, runner_os_component.log. Once you have completed the trace routine in either of the two other streams, return to the next step in this stream.	-
9	rmipluginmanager.log	Shows the plug-in has attempted to send a message to the core.	4
10	queue_communicator.log	A reply is placed here.	3
11	initiatormanager.log	Shows the initiator has been released.	5
12	messagedistributoroutgoing.log	Shows the message is sent through the controller plug-in manager to the scheduling manager.	5
13	transmitter.log	Log of all sending activity and any errors discovered.	0, 1, 2

Runner Plug-in AFM Processing

Step	Log File	Description	Log Level
	internal_plugin_queue_for_runnerplugin.log	All AFMs sent to the Runner plug-in is logged here.	3
	runner_plugin_executing_jobs_map.log	All submitted jobs are logged here.	3
	runner_plugin_transmitter_queue.log	All messages sent to OS component are logged here.	3
	runner_os_component.log	Any errors are logged here.	0, 1, 2
	runner_plugin_receiver_queue.log	All messages coming back from OS component are logged here.	3
	Sent back to core	Return to main stream of AFM processing.	

Filemon AFM Plug-in Processing

Step	Log File	Description	Log Level
1	internal_plugin_queue_for_filemonplugin.log	All AFMs sent to the Filemon plug-in are logged here.	3
2	file_mon_plugin_threads.log	All executing triggers are registered here. When a trigger is completed successfully or has failed, it is removed from this database.	3
3	Sent back to core	Return to main stream of AFM processing.	

Log Resource Usage Information within the JVM

The agent can periodically collect resource usage information within the Java Virtual Machine (JVM) such as memory usage and threads information. This data is logged in a file named simple_health_monitor.log. To log resource usage information within the JVM, edit the following parameters in the agentparm.txt file and restart the agent:

- Set the core.health.monitor.enable parameter to true
- Set the log.level parameter to 5 or greater

You can also specify the polling interval for logging the information using the core.health.monitor.interval parameter. The default is 60 000 ms (1 min). The minimum interval time is 1000 ms (1 sec). If the specified interval is less than the minimum, the agent ignores that value and logs the information at every 1000 ms.

Agent Error Messages on i5/OS

This section provides common error messages returned by the agent installed on an i5/OS system.

Command does not use shell

Reason:

Only a script can use a shell.

Action:

Remove the shell statement from the job definition.

Command or script name missing

Reason:

In the job definition, you have not defined the required command or script name.

Action:

Add the command name or script name to the job definition.

Command requires a User ID

Reason:

Each command must specify an allowable user.

Action:

Add the USER statement.

Connection aborted by peer: JVM_recv in socket input stream read

Reason:

You may need to reset the communication.timeout parameter to allow more time between a message being sent and an acknowledgement (ACK) being received. When this time is exceeded, the connection is aborted.

Action:

As a starting point, set the value in msecs to 120000 and test. Change the value as needed so this error does not occur.

Error changing directory

Reason:

In the agentparm.txt file, the parameter oscomponent.initialworkingdirectory specifies the working directory. If the path is incorrect, this error will occur.

Action:

1. Verify the directory exists.
2. Specify the correct path in the parameter.

Error closing stdout, stdout, or stderr

Reason:

The error is caused by a system problem that may be intermittent.

Action:

Try to resubmit the job. If this action does not resolve the problem, contact your system administrator.

Error creating pthread

Reason:

System resources are low.

Action:

Contact your system administrator.

Error creating spool file. Job: JOB.TXT/APPL. xxxx/MAIN, errno: 31, Reason: Too many links

Reason:

On AIX and Linux, the file system may limit the number of spool directories.

Action:

Refer to your operating system documentation for details regarding the spool directory limitations. Clear the spool directories periodically using the agent.

Error creating stdout spool file

Reason:

The user may not have the necessary permissions to create the spool file.

Action:

Check the user permissions.

Error getting owner of the script

Reason:

Failure to get the owner of the script from the system password file.

Action:

Contact your system administrator.

Error occurring during submission

Reason:

Connection error with the scheduling manager.

Action:

Check the connection to the scheduling manager.

Error opening stdin, stdout, or stderr

Reason:

The agent does not have permission to open the file. This error is caused by a system problem.

Action:

1. Check permissions. Change them as needed to allow the agent to open the file.
2. Contact your system administrator.

Error redirecting stdin, stdout, or stderr

Reason:

The agent tries to redirect an input file to another file and an error results. This file could be a spool file or some specified stdin or stdout file. For a stderr message, the cause may be a system problem that is intermittent.

Action:

Try to resubmit the job. If this action does not resolve the problem, contact your system administrator.

File not found

Reason:

The agent cannot find the file. Either the wrong path was specified in the job definition, or the file does not exist.

Action:

Verify the path and check the file exists.

Invalid command: not listed in oscomponent.validcommand

Reason:

Invalid command.

Action:

Add the command to the oscomponent.validcommand parameter in the agentparm.txt file, or contact your system administrator.

Invalid file name

Reason:

The file path is too long. This error can happen if there are too many symbolic links. The system call will return an error if the name is too long.

Action:

Relocate the files to get rid of the symbolic link.

Invalid shell error

Reason:

In the agentparm.txt file, you must specify the valid shell in the parameter oscomponent.validshell. The corresponding job definition may not have the correct shell specified in the shell statement.

Action:

1. Verify the correct shell is specified in the agentparm.txt file. Alternatively, you can set the oscomponent.checkvalidshell=false parameter in the agentparm.txt file, so that the agent will never bother to validate whether the shell is valid.
2. Verify the corresponding job definition has the correct shell specified in the shell statement.
3. In the first line of a script file, the shell and its path must match exactly what you specified in the validshell parameter.

Irregular file

Reason:

The file is not a regular file, such as an ascii or binary file. The file may be a directory file or a pipe, or a program in /bin being run under PASE.

Action:

Check the file's type using ls -l. Replace the file with a regular file.

Not a script file

Reason:

The script contains non-printable characters.

Action:

Use Command instead of Script name in the job definition.

Refused by Agent security

Reason:

The job is refused by the agent's local security.

Action:

Check the security.txt file.

Script/Command not accessible

Reason:

The symbolic link does not exist.

Action:

Check the definition of the symbolic link to see if it exists.

Script/Command not executable

Reason:

The user may not have the necessary permissions to execute the script or command.

Action:

Check the user permissions.

Script/Command not readable

Reason:

User does not have read permission.

Action:

Check the user permissions.

User does not exist in the system

Reason:

You did not define the user on the agent system.

Action:

Ensure the user exists on the agent system.

Communication Problems Between the Agent and the Scheduling Manager

If there is a communication problem between the agent and the scheduling manger, the jobs are shown in the AGENTDOWN state. The following are possible causes:

- The agent is not started.
- The scheduling manager and the agent have different encryption keys.
- The scheduling manager and the agent have different values for the parameters that must match.
- A firewall is blocking the transmission of the agent responses.
- The scheduling manager address is specified as a DNS name in the agentparm.txt file, but the DNS name resolution on the agent computer is faulty or not available.

SNMP-related Problems

This section provides error messages related to the SNMP management connector.

CybSnmpPluginDriver terminated: java.net.BindException: Permission denied

Reason:

The SNMP communication port specified by the snmp.trap.listener.port agent parameter is already in use. The agent uses ports below 1024. On UNIX, start the agent as root.

Action:

Ensure that the value specified for the SNMP communication port (snmp.trap.listener.port) is not in use by another application. The default port number is 162.

java.lang.IllegalArgumentException: Passed ip address is invalid

Reason:

SNMP v1 does not support IPv6.

Action:

To ensure that the agent uses IPv4, add the following parameters to the agentparm.txt file:

```
java.net.preferIPv4Stack=true
java.net.preferIPv6Stack=false
```

FTP Job Failure Messages

If an FTP job fails, review the job status and spool file for information. The following status messages can appear when a job fails:

Access is denied

Reason:

The FTP user ID does not have the proper permission to access the file.

Action:

Determine whether you have access to the local path specified in the job definition on the agent computer.

File not found

Reason:

The agent cannot find the file. Either the wrong path was specified in the job definition, or the file does not exist.

Action:

Check that the remote path and file name specified in the job definition exist on the remote server.

Logon unsuccessful

Reason:

A problem with the FTP user exists.

Action:

1. Check that the password of your FTP user ID is correct on the scheduling manager.
2. If the agent runs as an FTP server, check that the user ID and password are also defined on the agent.

Note: If you update the ftpusers.txt file, restart the agent for your changes to take effect.

Password is missing

Reason:

A problem with the FTP user ID exists.

Action:

1. Check that the FTP user ID specified in the job definition is correct.
2. Check that the same FTP user ID is defined on the scheduling manager.
3. If the agent runs as an FTP server, check that the user ID and password are also defined on the agent.

Note: If you update the ftpusers.txt file, restart the agent for your changes to take effect.

Please log in with USER and PASS

Reason:

The FTP server requires SSL enablement.

Action:

Check that the FTP server has SSL enabled.

The system cannot find the path specified

Reason:

The path specified in the job definition is incorrect.

Action:

Check that the local path specified in the job definition exists on the agent computer.

Unknown host

Reason:

The remote server name specified in the job definition is incorrect.

Action:

Check that the remote server name specified in the job definition is correct.

Agent Parameters used for Troubleshooting

You can add the following parameters to the agentparm.txt file, as required, to configure the agent. These parameters are generally used for troubleshooting.

communication.timeout

Specifies the time, in milliseconds (ms), for TCP/IP that can elapse between a message being sent and an acknowledgement (ACK) being received. If this time is exceeded, the connection will be aborted.

Default: 10,000 milliseconds (10 seconds)

core.health.monitor.interval

Specifies the polling interval, in milliseconds (ms), for logging the resource usage information within the Java Virtual Machine.You can set this parameter when the core.health.monitor.enable parameter is set to true.

Default: 60,000 (1 min)

Note: The minimum interval time is 1000 ms (1 sec). If the specified interval is less than the minimum, the agent ignores that value and logs the information at every 1000 ms.

filemonplugin.sleepperiod

Specifies the time, in milliseconds (ms), a Monitoring job uses as the polling interval for file monitoring. Specify no less than 1000 ms.

Default: 30,000 (30 sec)

Note: This parameter does not apply to CA Workload Automation AE.

ftp.client.separator

Specifies the character that will be used to separate multiple file entries in the LOCALFILENAME or REMOTEFILENAME statements.

ftp.ssl.provider=IbmX509

Specifies an AIX parameter that supports IBM JSSE (Java Secure Socket Extension), a Java implementation of SSL and TLS.

Note: Do not change the value.

ftp.userfile=path

Specifies the location of the FTP user ID and password file. The default file name is ftpusers.txt.

Example: /export/home/userid/WA Agent R11.3/ftpusers.txt

log.archive

Defines the log archiving options:

- 0—Appends current date and time to the log file.
- 1—Renames to logfile.archive and starts a new file.
- 2—Removes current file.
- 3—Appends new log entries to the current logs.

Default: 0

log.folder

Specifies the location of the log files. You can specify the full path to the log file directory or the folder name that stores the log files. If you specify the folder name only, the agent creates the folder in the agent installation directory.

Default: the log subdirectory contained in the agent installation directory

objmon.cpu.scalefactor

Specifies a scale factor to multiply the load averages of a CPU and allow the agent, when processing a CPU Monitoring job, to express the load average as a percentage. This scale factor is for busy machines that would otherwise always report 100 percent utilization.

Default: 100

Example: If you set the scale factor to 10, and the reported load average is 7, then the reported CPU usage would be 70 percent.

objmon.scaninterval

Specifies the interval, in milliseconds (ms), between successive scans for any Monitoring job that uses continuous monitoring.

Default: 10,000 milliseconds (10 seconds)

Note: A shorter interval puts a greater demand on system resources.

oscomponent.dumpenvironment

Specifies whether all environment variables are written to the agent spool file for every RUN job.

- false—Does not write environment variables to the spool file
- true—Writes all environment variables to the spool file

Default: false

oscomponent.libjvmpath

Specifies the path statement to the Java library location.

oscomponent.initialworkingdirectory

Specifies the default initial working directory for all scripts.

- SCRIPT—Sets the path to where the script resides
- USER—Sets the path to the home directory of the owner of the script
- PATH—Sets the path to an absolute path to where the script should run

If you do not specify a value, the parameter defaults to the path where the running cybAgent resides.

Note: You can override the InitialWorkingDirectory on a per-job basis by specifying a value for the PWD environment variable.

oscomponent.noexitcode

Specifies the exit code that tells the agent not to send a completion code to the scheduling manager host.

Limits: 1-255

Default: 255 for UNIX

oscomponent.noforceprofile

Specifies whether or not the agent allows loading a .profile/.login file based on the usual UNIX rules for sourcing .profile/.login. When set to false, if the loginshell is set to false and USER has not been specified in the job definition, no .profile/.login will be sourced.

- false—Allows the agent to load the .profile
- true—Prevents the agent from loading the .profile/.login

Default: false

Note: If oscomponent.loginshell is set to false and USER is not specified in the job definition, no .profile/.login will be sourced and oscomponent.noforceprofile is ignored.

oscomponent.noguardianprocess

Specifies whether or not the agent resumes tracking jobs that were active at the time when the agent is recycled.

- false—Returns the status of any active or inactive job when the agent restarts
- true—Does not return the status of jobs that ran at the time the agent went down. Fails UNIX jobs upon restart and returns the message "Lost Control".

Default: false

Note: To enable the default, you must also set the persistence.coldstart parameter to false or comment it out.

oscomponent.security.turbo

Specifies whether the agent loads the security.txt file into a fast-loading, binary format when the cybAgent process starts up. This setting applies to run jobs only. FTP jobs have separate security rules. When the agent checks security rules for authorizations, it will do so much more quickly when this parameter is set to true. Enabling this feature is particularly useful if you have a large, multi-line security file where the required processing may slow system operation.

- false—The agent refreshes security rules each run job.
- true—The agent loads security rules into a binary-formatted file and does not check security rules again unless a manual refresh is issued to the agent.

Default: false

oscomponent.umask

Provides support for the umask command. The three-digit octal code specifies the file and directory permissions that are turned off. The default value, 022, sets the following permissions:

File rw-r--r--

Directories rwxr-xr-x

Note: This parameter will only apply to files created by the agent such as a spool or log file.

persistence.coldstart

Specifies whether the agent performs a warm or cold start, as follows:

- false—Performs a warm start. The agent will try to use the existing databases. However, if there is sufficient damage, the agent will not start.
- true—Performs a cold start. All databases will be automatically destroyed and new ones opened. No manual intervention is required. This setting is recommended if there is extensive damage to the databases. The agent discontinues job monitoring after a cold start.

  Note: On UNIX systems, the agent continues monitoring after a cold start.

Default: false (agent performs a warm start)

persistence.gcinterval

Specifies the persistent garbage collector interval. The garbage collector will be invoked at the end of each transaction and will run at least every N milliseconds.

Default: 10 000 (10 sec)

runnerplugin.spool.clean.enable

Specifies whether or not the agent deletes spool files.

- false—Disables the spool file cleaner
- true—Enables the spool file cleaner

Default: false

Note: If enabled, the agent deletes spool files older than 10 days and checks the spool files every day by default. To specify a different file expiration value, set the runnerplugin.spool.expire parameter. To specify a different sleep interval value, set the runnerplugin.spool.sleep parameter.

Chapter 13: Related Documentation

Documentation for the agent and scheduling managers is available in PDF format at http://ca.com/support.

Note: To view PDF files, you must download and install the Adobe Reader from the Adobe website if it is not already installed on your computer.

This section contains the following topics:

CA Workload Automation AE Documentation

To work with the agent and CA Workload Automation AE, see the following documentation:

Task	Documentation
Configure the scheduling manager to work with the agent	*CA Workload Automation AE UNIX Implementation Guide* *CA Workload Automation AE Windows Implementation Guide*
Define, monitor, and control jobs	*CA Workload Automation AE Reference Guide* *CA Workload Automation AE User Guide* *CA Workload Control Center Workload Scheduling Guide*

CA Workload Automation DE Documentation

To work with the agent and CA Workload Automation DE, see the following documentation:

Task	Documentation
Configure the scheduling manager to work with the agent	*CA Workload Automation DE Admin Perspective Help*
Define jobs	*CA Workload Automation DE Define Perspective Help*
Monitor and control jobs	*CA Workload Automation DE Monitor Perspective Help*

Note: The online help is available in HTML and PDF formats.

CA Workload Automation ESP Edition Documentation

To work with the agent and CA Workload Automation ESP Edition, refer to the following documentation:

Task	Documentation
Configure the agent to work with the scheduling manager	*CA Workload Automation ESP Edition Installation and Configuration Guide*
Define jobs	*ESP System Agent for i5/OS Guide to Scheduling Workload*
Monitor and control jobs	*ESP System Agent for i5/OS Guide to Scheduling Workload* *CA Workload Automation ESP Edition Operator's Guide*

CA Workload Automation CA 7 Edition Documentation

To work with the agent and CA Workload Automation CA 7 Edition, see the following documentation:

Task	Documentation
Configure the scheduling manager to work with the agent	*CA Integrated Agent Services Implementation Guide* *CA Workload Automation CA 7 Edition Interface Reference Guide* *CA Workload Automation CA 7 Edition Systems Programming Guide*
Define, monitor, and control jobs	*CA Integrated Agent Services User Guide* *CA Workload Automation CA 7 Edition Interface Reference Guide* *CA Workload Automation CA 7 Edition Database Maintenance Guide* *CA Workload Automation CA 7 Edition Command Reference Guide*

Index

O

P

S

U

W