z/OS

APAR OA12576

z/OS

APAR OA12576

Contents

© Copyright IBM Corp. 2007

About this document

This document supports APAR OA12576 for the UNIX System Services element of z/OS®. This document is only available on the UNIX System Services Web site at: http://www.ibm.com/servers/eserver/zseries/zos/unix/pdf/OA12576.pdf.

© Copyright IBM Corp. 2007

Internationalization on z/OS

Setting up your system or user environment for internationalization on z/OS is a little different from what most users are accustomed to when setting up internationalization on ASCII platforms. On z/OS, an extra step is usually needed when changing your locale. This step involves setting the ASCII/EBCDIC coded character set conversion for the controlling terminal. This is required because most PC terminal emulators require ASCII data, but the z/OS shells use EBCDIC data.

For example, when using a PC emulator to interactively log into an ASCII UNIX® operating system, a user will:

- On the PC, change the emulator's coded character set to match the coded character set of the remote session's locale.
- In the UNIX shell, assign the environment variable LC_ALL to a new locale, where the ASCII coded character set of that locale matches the emulator's setting.

When interactively logging into an EBCDIC z/OS UNIX operating system, the user will:

- On the PC, change the emulator's coded character set to match the ASCII coded character set of the remote session's locale. For example, the user might change the translation settings in their emulator to use coded character set ISO/IEC 8859-2 (Latin-2).
- In the UNIX shell:
  - Assign the environment variable LC_ALL to a new locale, whose EBCDIC coded character set is compatible with the ASCII coded character set used in the emulator. To determine if a coded character set is compatible with a particular locale, refer to the appendix titled "Locales supplied with z/OS XL C/C++" in *z/OS XL C/C++ Programming Guide*.

    For example, a user might issue:

    ```
    export LC_ALL=Hu_HU.IBM-1165
    ```
  - If a tty is allocated, issue the **chcp** command to assign the EBCDIC and ASCII coded character sets, as appropriate. Note that the specified ASCII coded character set should match that of the client emulator's setting.

    For example, a user might issue:

    ```
    chcp -a ISO8859-2 -e IBM-1165
    ```

On z/OS, in daemons such as rlogind, telnetd, and sshd, conversion between ASCII and EBCDIC occurs in the forked daemon process which handles the user's connection. This process allocates the terminal (tty) for the end user. On ASCII platforms, no conversion is necessary.

OpenSSH and internationalization

The GA-level of OpenSSH assumes that all text data traveling across the network is encoded in ISO/IEC 8859-1 (Latin-1). Specifically, OpenSSH treats data as text and performs conversion between the ASCII Latin-1 coded character set and the EBCDIC coded character set of the current locale in the following scenarios:

- ssh login session
- ssh remote command execution
- scp file transfers
- sftp file transfers when the *ascii* subcommand is specified

© Copyright IBM Corp. 2007

With ***APAR OA12576***, the OpenSSH daemon can understand and handle non-Latin-1 coded character sets on the network for ***interactive*** sessions, specifically sessions with a tty allocated. However, not all EBCDIC coded character sets are compatible with ISO 8859-1. To determine if a coded character set is compatible with a particular locale, refer to the appendix titled "Locales supplied with z/OS XL C/C++" in *z/OS XL C/C++ Programming Guide*.

Warning: If there is no one-to-one mapping between the EBCDIC coded character set of the session data and ISO 8859-1, then nonidentical conversions may occur. Specifically, substitution characters (for example, IBM-1047 0x3F) will be inserted into the data stream for those incompatible characters. See "Considerations for configuring OpenSSH for another locale" on page 3 for more information.

Sessions which are considered interactive include:
- ssh login session when a tty is allocated. This is the default behavior.
- ssh remote command execution, when the **-t** option is used to allocate a tty.

The following scenarios are considered ***noninteractive***, and continue to interpret network data as ISO 8859-1:
- ssh login session when the **-T** option is specified (which disables tty allocation.)
- ssh remote command execution, when the **-t** option is not specified. The default behavior is not to allocate a tty for remote command execution.
- scp file transfers
- sftp file transfers when the *ascii* subcommand is specified

The support provided by APAR OA12576 is summarized in Table 1.

Table 1. Summary of support provided by APAR OA12576. The table lists the expected coded character set for the network data during both interactive and noninteractive OpenSSH sessions with various peers, when OA12576 is applied.

Scenario	Session is:	Client is running:	Server is running:	Coded character set of network data is:
1	Interactive	z/OS	z/OS	ASCII coded character set as defined by the **chcp** setting. **Restriction:** The z/OS client expects Latin-1, so the ASCII coded character set must be handled accordingly on the server side. See "Considerations for configuring OpenSSH for another locale" on page 3 for more information.
2	Interactive	Non-z/OS UNIX (such as AIX®, Linux®) or PC	z/OS	ASCII coded character set as defined by the **chcp** setting.
3	Interactive	z/OS	Non-z/OS UNIX (such as AIX, Linux) or PC	ISO 8859-1
4	Noninteractive	z/OS	z/OS	ISO 8859-1
5	Noninteractive	Non-z/OS UNIX (such as AIX, Linux) or PC	z/OS	ISO 8859-1
6	Noninteractive	z/OS	Non-z/OS UNIX (such as AIX, Linux) or PC	ISO 8859-1

Note that some OpenSSH sessions transfer data as binary. In other words, no character translation is performed. These include:
- sftp sessions (when the *ascii* subcommand is not used)
- Port-forwarded sessions
- X11-forwarded sessions

Limitation: OpenSSH on z/OS does not support multibyte locales.

Considerations for configuring OpenSSH for another locale

Configuring the OpenSSH daemon

The OpenSSH daemon (sshd) must be run in the POSIX C locale. In most cases, this occurs without any action on behalf of the user. However, an alternate locale could inadvertently be picked up through the shell profile of the user ID invoking the daemon, or through the ENVAR run-time option in CEEPRMxx member of SYS1.PARMLIB. You can enforce LC_ALL=C by using STDENV in the BPXBATCH job that starts the daemon.

For more information about the POSIX C locale, see the chapter on the definition of the S370 C, SAA® C, and POSIX C locales in *z/OS XL C/C++ Programming Guide*.

Configuring the OpenSSH client

With APAR OA12576, the OpenSSH daemon (sshd) can understand and handle non-Latin-1 coded character sets for *interactive* sessions, specifically those with a tty allocated. However, the OpenSSH client (ssh) still expects network data to be encoded in ISO 8859-1.

If the EBCDIC coded character set for your sessions are compatible with ISO 8859-1, the following setup is not required. To determine if a coded character set is compatible with a particular locale, refer to the appendix titled “Locales supplied with z/OS XL C/C++” in *z/OS XL C/C++ Programming Guide*.

If **chcp** is issued in your environment, verify that the SSH peer supports the specified ASCII coded character set.

For example, if you are using a PC to connect directly to z/OS, you issue the **chcp** command in the remote z/OS shell to assign the ASCII coded character set for the terminal to match that of the PC emulator. With APAR OA12576, the daemon will properly inherit the **chcp** setting to translate the network data accordingly. The SSH peer, the PC emulator, must also support the new ASCII coded character set. This can be determined by checking your emulator’s configuration.

If you are issuing the ssh client from z/OS to connect to a z/OS platform running in another locale, you need to verify that the ASCII coded character set of the remote session (set by **chcp**) is ISO 8859-1, which is what the z/OS ssh client expects.

Warning: If there is no one-to-one mapping between the EBCDIC coded character set of the session data and ISO 8859-1, then nonidentical conversions may occur. Specifically, substitution characters (for example, IBM-1047 0x3F) may be inserted into the data stream for those incompatible characters.

If the EBCDIC coded character set of your target locale is not compatible with ISO 8859-1, then nonidentical conversions may occur in either of these scenarios:
- You are running in the target locale when issuing the ssh command locally.

- You are running in the target locale in your remote ssh session.

To avoid nonidentical conversions, you can force the ssh client process to run in the C locale. Note also that the remote session's shell must also be configured to run in either the C locale or a locale with a coded character set that is compatible with ISO 8859-1.

To force the local ssh client process to run in a C locale, you may run ssh as follows:

```
 LC_ALL=C ssh [arguments]
```

where arguments represents the remainder of arguments passed to ssh.

You can set up a shell alias to avoid repeatedly typing the above command. For example:

```
alias ssh="LC_ALL=C ssh"
```

Configuring ssh when LC_ALL is set through shell profiles

If all of the following are true for your environment:

- Your system is configured to run in a locale other than the default C locale
- The corresponding ASCII coded character set for your locale is **not** ISO 8859-1
- You changed the system-wide locale by setting LC_ALL through shell profiles (for example, /etc/profile or $HOME/.profile.)

then perform the following steps as part of your OpenSSH system-wide setup.

If you have changed the locale at a system-wide level, consider defining this alias in an area where it can be picked up by all users and inherited by all subshells. Shell aliases are typically defined through the file named by the ENV variable of /bin/sh. Users may have defined their own ENV setting in one of their shell profiles. For this setup, the ENV variable should be exported so it is inherited by subshells.

- For /bin/sh users, this alias should be defined in the ENV file.
- For /bin/tcsh users, this alias should be defined in /etc/csh.cshrc.

Steps to follow for setting up a system-wide alias for ssh:

1. Create a UNIX file /etc/ssh/.sshalias which contains the following line:
   ```
   alias ssh="LC_ALL=C ssh"
   ```
2. Ensure that the UNIX permissions for this file are world-readable. From the UNIX prompt, issue:
   ```
   chmod 744 /etc/ssh/.sshalias
   ```
3. Notify users to either add the ssh alias to their ENV file or read in the above ENV file from their user-defined ENV file. For example, users may add to their ENV file the following line, which reads in (or "sources") the new ssh alias file using the **dot** command:
   ```
   . /etc/ssh/.sshalias
   ```
4. Verify that the ssh alias is set properly. From a ***new*** UNIX shell, issue:
   ```
   > alias ssh
   ssh="LC_ALL=C ssh"
   >
   ```

Configuring ssh when LC_ALL is set through the ENVAR run-time option in CEEPRMxx

If all of the following statements are true for your environment

- Your system is configured to run in a locale other than the default C locale

- The corresponding ASCII code page for your locale is not ISO 8859-1
- You changed the system-wide locale by setting LC_ALL through the ENVAR run-time option in a CEEPRMxx member of SYS1.PARMLIB or through the MVS™ operator command SETCEE.
  - For information about SETCEE, see *z/OS MVS System Commands*.
  - *z/OS MVS Initialization and Tuning Reference*contains information about the ENVAR run-time option for CEEPRMxx.

then perform the following steps as part of your OpenSSH system-wide setup.

Create an alias for the ssh command which forces ssh to run in a C locale. This alias should be defined in an area where it will be picked up by all users and all subshells, even when a login shell is not used. Shell aliases are typically defined through the file named by the ENV variable of /bin/sh. The ENVAR run-time option in CEEPRMxx can also be used to set a shell alias.

Steps to follow for setting up a system-wide alias for ssh through the ENVAR run-time option of CEEPRMxx:

1. Create a UNIX file /etc/ssh/.sshalias which contains the following line:
   ```
   alias ssh="LC_ALL=C ssh"
   ```
2. Ensure that the UNIX permissions for this file are world-readable. From the UNIX prompt, issue:
   ```
   chmod 744 /etc/ssh/.sshalias
   ```
3. Notify users to define this alias if they already have created their own ENV file. Users may have defined their own ENV setting in one of their shell profiles. Their ENV setting will not be inherited for remote command execution or remote ssh processes, because these are not login shells. However, ENV will be initialized to their own setting for interactive shells, where users may later be issuing the ssh command. Their ENV setting overrides the ENVAR setting through CEEPRMxx, so they need to pick up your alias for local ssh command invocations.
   - For /bin/sh users, this alias should be defined in the file specified by the ENV variable. For /bin/tcsh users, this alias should be defined in /etc/csh.cshrc.
   - For /bin/tcsh users, this alias should be defined in /etc/csh.cshrc.

   Notify users to either add the ssh alias to their ENV file or read in your ENV file from their ENV file. For example, users may add to their ENV file the following line, which reads in (or “sources”) the new ssh alias file using the **dot** command:
   ```
   .  /etc/ssh/.sshalias
   ```
4. Issue the MVS operator command SETCEE to change the CEEPRMxx setting dynamically. For example:
   ```
   SETCEE  CEEDOPT,ENVAR('LC_ALL=Hu_HU.IBM-1165','ENV=/etc/ssh/.sshalias')
   ```
5. Verify that the ssh alias is set properly. From a new UNIX shell, issue:
   ```
   > echo $ENV
   /etc/ssh/.sshalias
   > alias ssh
   ssh="LC_ALL=C ssh"
   >
   ```

Configuring sftp

By default, sftp treats files as binary. Use sftp if you do not want your data files altered. If you want your data files translated between ASCII and EBCDIC, use iconv to convert the files at the start or end of the sftp transfer.

If you have existing sftp jobs that use the *ascii* sftp subcommand: The *ascii* sftp subcommand converts between ASCII ISO 8859-1 and the EBCDIC of the current locale. If the file data on the network is in a coded character set that is ***not*** ISO 8859-1, then you must adjust existing jobs to transfer files as binary and use **iconv** for the data conversion.

Configuring scp

By default, scp treats files as text. It assumes that all data going over the network is encoded in ASCII coded character set ISO 8859-1. The EBCDIC coded character set of the current locale is used for data conversion. On the remote system, the locale of the scp process is determined by how LC_ALL is initialized on that system. If LC_ALL is set through a shell profile (for example, /etc/profile), then it will not be inherited by the remote scp process. Specifically, the remote scp process will run in a C locale. See Figure 1. If a user on Host GERMANY running in locale De_DE.IBM-273 uses scp to transfer a file to a remote host, the file contents will be converted from IBM-273 to ISO 8859-1 to go over the network, and from ISO 8859-1 to IBM-1047 on the target system.

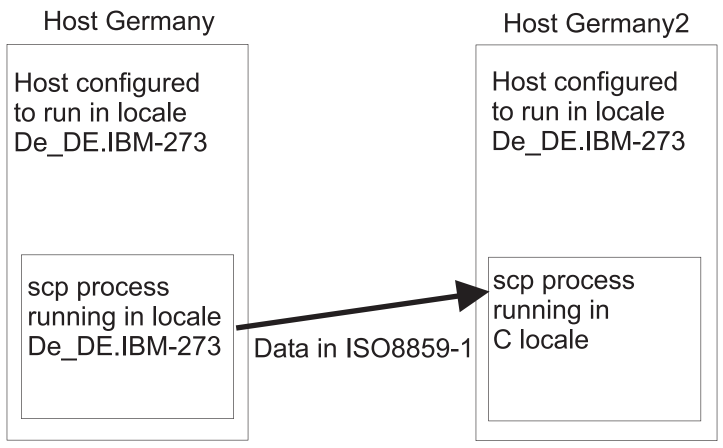
Host Germany
Host configured to run in locale De_DE.IBM-273
scp process running in locale De_DE.IBM-273
Data in ISO8859-1
Host Germany2
Host configured to run in locale De_DE.IBM-273
scp process running in C locale

Figure 1. Using scp when LC_ALL is set through shell profiles

If LC_ALL is set through the ENVAR run-time option in the CEEPRMxx member, then the new locale will be inherited by the remote scp process. Specifically, the EBCDIC coded character set of that locale will be used. See Figure 2 on page 7. If a user on Host GERMANY running in locale De_DE.IBM-273 uses scp to transfer a file to a remote host, the file contents will be converted from IBM-273 to ISO 8859-1 to go over the network, and from ISO 8859-1 to IBM-273 on the target system.

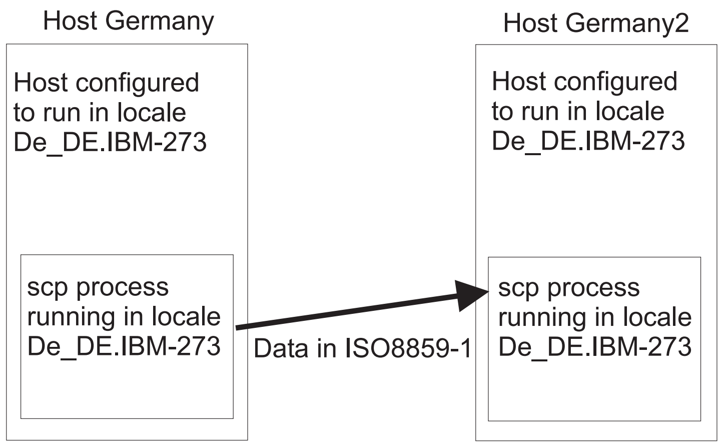
Host Germany
Host Germany2
Host configured to run in locale De_DE.IBM-273
Host configured to run in locale De_DE.IBM-273
scp process running in locale De_DE.IBM-273
Data in ISO8859-1
scp process running in locale De_DE.IBM-273

Figure 2. Using scp when LC_ALL is set through ENV in CEEPRMxx

Warning: If a file is encoded in an EBCDIC coded character set whose compatible ASCII coded character set is not ISO 8859-1, then nonidentical conversions may occur. Specifically, substitution characters (for example, IBM-1047 0x3F) may replace characters which do not have a mapping between the specified EBCDIC coded character set and ISO 8859-1. To determine if a coded character set is compatible with a particular locale, refer to the appendix titled “Locales supplied with z/OS XL C/C++” in *z/OS XL C/C++ Programming Guide*.

If the EBCDIC coded character set for your sessions are compatible with ISO 8859-1 and the above text conversions are satisfactory for your environment, the following setup is not required.

If you have existing scp jobs: If you are changing the locale on a system whose ASCII coded character set is not Latin-1 and you have existing scp jobs configured, you may:

- Convert those jobs to use sftp.
- Force scp to treat files as though they are encoded in IBM-1047, so substitution characters are not introduced. This can be done through a shell alias, as described in “Configuring scp when LC_ALL is set through shell profiles.”
- If you intend to configure a new locale through a shell profile, then continue to “Configuring scp when LC_ALL is set through shell profiles.”
- If you intend to configure a new locale using CEEPRMxx to specify run-time options, then continue to “Configuring scp when LC_ALL is set through the ENVAR run-time option in CEEPRMxx” on page 8.

Configuring scp when LC_ALL is set through shell profiles

If all the following are true for your environment:

- Your system is configured to run in a locale other than the default C locale
- The corresponding ASCII coded character set for your locale is **not** ISO 8859-1
- You changed the system-wide locale by setting LC_ALL through shell profiles (for example, /etc/profile or $HOME/.profile.
- You do not want to convert existing scp workloads to sftp workloads

then perform the following steps as part of your OpenSSH system-wide setup.

If you have changed the locale at a system-wide level, consider defining this alias in an area where it can be picked up by all users and inherited by all subshells. Shell aliases are typically defined through the file named by the ENV variable of /bin/sh.

Users may have defined their own ENV setting in one of their shell profiles. For this setup, the ENV variable should be exported so it is inherited by subshells.

- For /bin/sh users, this alias should be defined in the ENV file.
- For /bin/tcsh users, this alias should be defined in /etc/csh.cshrc.

Steps to follow for setting up a system-wide alias for scp:

1. Create a UNIX file, /etc/ssh/.sshalias, which contains the following line:
   ```
   alias scp="LC_ALL=C scp"
   ```
2. Ensure that the UNIX permissions for this file are world-readable. From the UNIX prompt, issue:
   ```
   chmod 744 /etc/ssh/.sshalias
   ```
3. Notify users to either add the scp alias to their ENV file or read in the above ENV file from their user-defined ENV file. For example, users may add to their ENV file the following line, which reads in (or “sources”) the new scp alias file using the **dot** command:
   ```
   .  /etc/ssh/.sshalias
   ```
4. Verify that the scp alias is set properly. From a *new* UNIX shell, issue:
   ```
   > alias scp
   scp="LC_ALL=C scp"
   >
   ```

Configuring scp when LC_ALL is set through the ENVAR run-time option in CEEPRMxx

If all the following are true for your environment:

- Your system is configured to run in a locale other than the default C locale
- The corresponding ASCII code page for your locale is **not** ISO 8859-1
- You changed the system-wide locale by setting LC_ALL through the ENVAR run-time option in a CEEPRMxx member or through the SETCEE operator command.
  – For information about SETCEE, see *z/OS MVS System Commands*.
  – *z/OS MVS Initialization and Tuning Reference* contains information about the ENVAR run-time option for CEEPRMxx.
- You do not want to convert existing scp workloads to sftp workloads

then perform the following steps as part of your OpenSSH system-wide setup.

Steps to follow for setting up a system-wide alias for scp through the ENVAR run-time option of CEEPRMxx:

1. Create a UNIX file /etc/ssh/.sshalias which contains the following line:
   ```
   alias scp="LC_ALL=C scp"
   ```
2. Ensure the UNIX permissions for this file are world-readable. From the UNIX prompt, issue:
   ```
   chmod 744 /etc/ssh/.sshalias
   ```
3. Notify users to define this alias if they already have created their own ENV file. Users may have defined their own ENV setting in one of their shell profiles. Their ENV setting will not be inherited for remote command execution or remote scp processes, because these are not login shells. However, ENV will be initialized to their own setting for interactive shells, where users may later be issuing the scp command. Their ENV setting overrides the ENVAR setting through CEEPRMxx, so they need to pick up your alias for local scp command invocations.

- For /bin/sh users, this alias should be defined in the file specified by the ENV variable.
- For /bin/tcsh users, this alias should be defined in /etc/csh.cshrc.

Notify users to either add the scp alias to their ENV file or read in your ENV file from their ENV file. For example, users may add to their ENV file the following line, which reads in (or “sources”) the new scp alias file using the **dot** command:

```
. /etc/ssh/.sshalias
```

4. Issue the SETCEE operator command to change the CEEPRMxx setting dynamically. For example:

```
SETCEE  CEEDOPT,ENVAR('LC_ALL=Hu_HU.IBM-1165','ENV=/etc/ssh/.sshalias')
```

5. Verify that the scp alias is set properly. From a *new* UNIX shell, issue:

```
> echo $ENV
/etc/ssh/.sshalias
> alias scp
scp="LC_ALL=C scp"
>
```

Customizing your UNIX environment to run in another locale

To configure your UNIX environment to run in another locale, see the chapter on customizing for your national code page in *z/OS UNIX System Services Planning*.

Rule: All files used by OpenSSH (such as key files and configuration files) must be in the IBM-1047 coded character set, with the exception of the **rc** files (/etc/ssh/sshrc and ~/.ssh/rc). The **rc** files are parsed by /bin/sh and should be in the coded character set of the current locale. Do not use the /etc/ssh/sshrc file if there is a possibility of the users on the system running in different locales.

Warning: While it is possible to set LC_ALL through the ENVAR run-time option of the CEEPRMxx member, configuring the locale in this way may cause unexpected results. Specifically, it is possible that daemons or long-running processes may expect to run in a C locale. Verify that all these processes support running in your alternate locale. Additionally, some system administration user IDs may need to run in a C locale, for editing configuration files which expect to be encoded in IBM-1047.

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide, Volume I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

© Copyright IBM Corp. 2007

Notices

This information was developed for products and services offered in the USA.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

AIX
MVS
SAA
z/OS

IBM, the IBM logo, and ibm.com are registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be trademarks or service marks of others.

Program Number:

Printed in USA