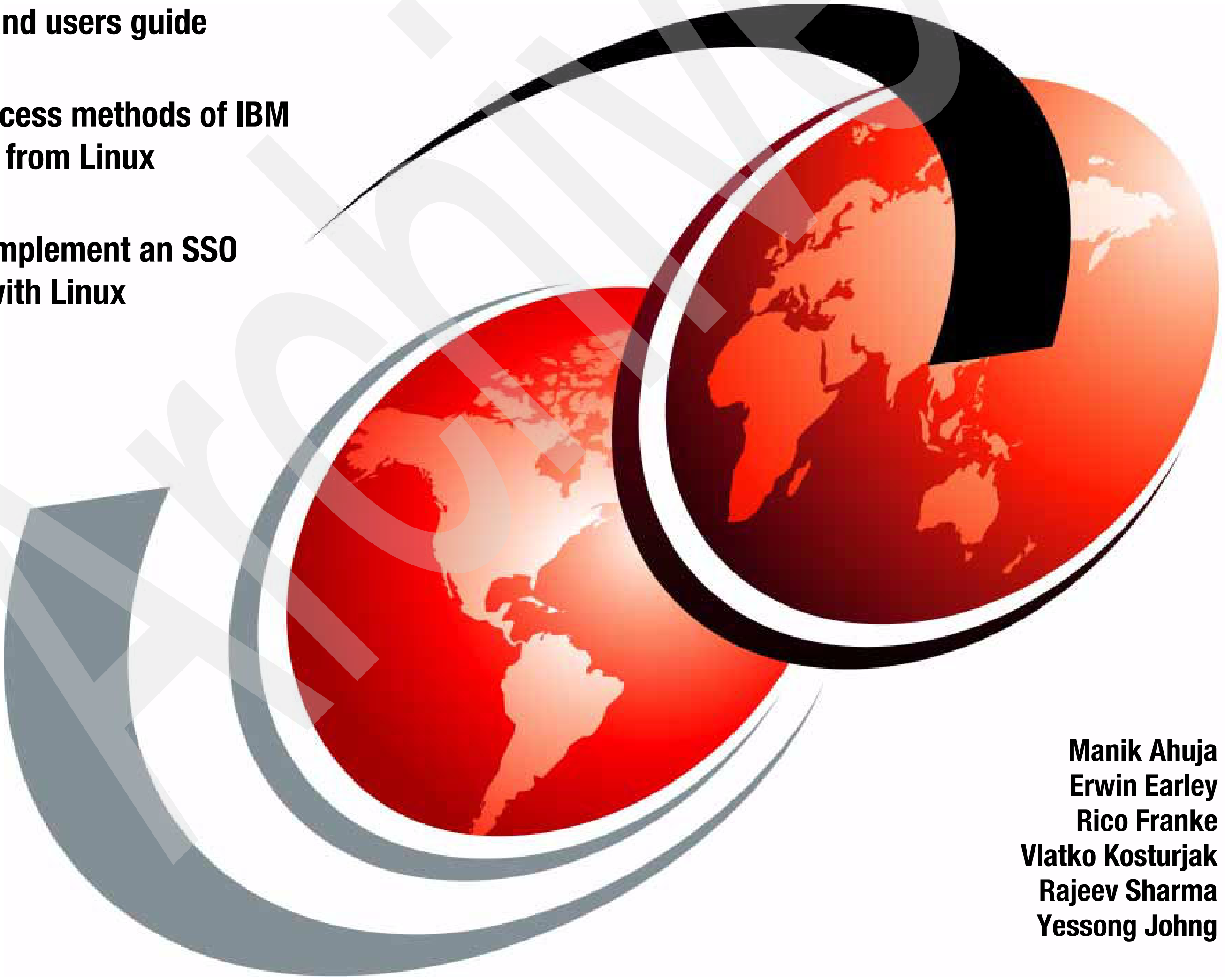
Linux Integration with IBM i5/OS

iSeries access for Linux installation, features, and users guide

Various access methods of IBM i5/OS data from Linux

Plan and implement an SSO scenario with Linux



Manik Ahuja
Erwin Earley
Rico Franke
Vlatko Kosturjak
Rajeev Sharma
Yessong Johng

Redbooks

ibm.com/redbooks

International Technical Support Organization

Linux Integration with IBM i5/OS

November 2006

SG24-6551-00

Note: Before using this information and the product it supports, be sure to read the general information in “Notices” on page vii.

First Edition (November 2006)

This edition applies to V5R4 of IBM i5/OS.

Contents

Notices

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

Advanced Function Printing™
AFP™
AIX®
AS/400®
DB2®
DB2 Universal Database™
eServer™
i5/OS®
IBM®
Infoprint®
Informix®
IPDS™
iSeries™
NetServer™
OpenPower™
OS/400®
POWER™
PowerPC®
Power PC®
pSeries®
Redbooks™
Redbooks (logo) ™
System i™
WebSphere®
xSeries®
zSeries®

The following terms are trademarks of other companies:

Java, JDBC, JDK, JSP, JVM, StarOffice, Sun, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Active Directory, Excel, Microsoft, Windows NT, Windows Server, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

i386, Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

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

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

RealVNC and the RealVNC logo are trademarks of RealVNC Ltd.

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

Preface

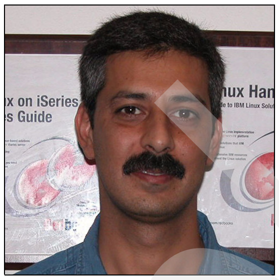
IBM® System i™ platform offers many points of integration that support Linux® applications leveraging IBM i5/OS applications and data, such as IBM eServer™ iSeries™ Access for Linux, IBM i5/OS NetServer™ with Samba support, and IBM Java™ Virtual Machine and JTOpen support.

This IBM Redbook covers each communication technique with sample code and shows you the methods that you can use to connect Linux applications with IBM i5/OS.

This book is intended to help Linux users, programmers and administrators, with an IBM i5/OS background, to connect Linux with IBM i5/OS.

The team that wrote this IBM Redbook

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.



Manik Ahuja is an IT Architect for IBM. His work involves designing IT infrastructure solutions by selecting the best combination of hardware software for each requirement. Manik holds industry recognized certifications for most operating systems and is equally proficient in both hardware and software technologies. He started his career as a computer lab assistant in 1995 and gradually moved to being a Systems Administrator and then to an IT Architect. He is an acknowledged expert at all things open source and also writes technology articles for various magazines. His areas of specialization would include designing and implementing large e-mail systems, high availability clusters, data consolidation techniques, and virtualization solutions.

Erwin Earley is an Advisory Software Engineer for IBM, assigned to the System i Technology Center in Rochester, Minnesota. In his role in the Linux Center of Competency, Erwin is responsible for technical enablement of field personnel for Linux on the IBM System i platform. Erwin has written numerous articles centered on Linux on System i and is a certified professional with the Linux Professional Institute, as well as a Red Hat Certified Technician. He can be reached at erwin.earley@us.ibm.com.

Rico Franke is an iSeries Technical Support Specialist with IBM in Germany. He has five years of experience in iSeries. His areas of expertise include iSeries Access, EIM / Kerberos, integrated IBM xSeries® solutions (IXA / IXS) and logical partition (LPAR). He provides assistance in problem determination, configuration, and usage for IBM customers. He holds a degree of engineering in Information Technology from the University of Cooperative Education Mannheim.

Vlatko Kosturjak is an IT Security Specialist with IBM Global Services Security and Privacy Services at IBM Croatia. In his practice, Vlatko specializes in ethical hacking, IT audit, security policy development according to ISO/IEC 17799 standard, consultative services for the design, implementation, and validation of enterprise-wide continuity and recovery programs, and assisting clients with developing and implementing effective security solutions for the protection of their information assets. He has extensive experience in security design for networks and server hardening on various operating systems.

Rajeev Sharma is an IT specialist in Linux with IBM Sales and Distribution at IBM India. His previous jobs include a Technical Account Manger in Linux Impact Team in Novell Bangalore and a Systems Engineer and Business Development in NetCore Solutions in Mumbai. He is a subject matter expert in consulting and solution designing and implementation of SUSE Linux Enterprise Server and SUSE Linux Open Exchange Server among other things.

Yessong Johng is an IBM Certified IT Specialist at the IBM International Technical Support Organization, Rochester Center. He started his IT career at IBM as an S/38 Systems Engineer in 1982 and has been with S/38, IBM AS/400®, and now iSeries for 20 years. He writes extensively and develops and teaches IBM classes worldwide in the areas of IT optimization whose topics include Linux, IBM AIX®, and Microsoft® Windows® implementations on iSeries. He is also interested in the e-business area, especially IBM WebSphere® implementations on iSeries.

Thanks to the following people who have contributed in various draft materials that this book is based on.

Sungsim Park was an ITSO Specialist at the International Technical Support Organization, Poughkeepsie Center, at the time of the draft materials creation. She has over 16 years of experience in working with S/36, S/38, and iSeries servers. Before joining ITSO in 2000, she taught IBM classes in all areas of iSeries as a senior education specialist in IBM-Korea and she provided technical marketing support to AS/400 sales representatives, IBM Business Partners, and customers. Her areas of expertise include server consolidation and application development.

Abhijit Chavan was a Systems Engineer in Starcom Software Pvt Ltd, Mumbai, India. Starcom Software is a Business Partner of IBM and is involved with Web Applications Development and Network Infrastructure Management on Unix/Linux systems. Abhijit has two years of working experience on Linux. His present job is as a Linux Systems Administrator and his areas of expertise include Transmission Control Protocol/Internet Protocol (TCP/IP) networking, network management, and security with services like Apache http, Sendmail, Bind, Samba, Secure Shell (SSH), and Linux Firewall. He has two years of experience in programming with C and one year with Perl and Shell. He holds a degree in Computer Engineering from the University of Mumbai (Bombay).

Jian Hui Chen was an IT specialist from IBM China. He has four years of experience in iSeries and IBM zSeries® solution development field. The last two years he supported customers/ISVs porting their applications to the iSeries platform from other DBMS, such as Oracle, Sybase, IBM Informix®. Also, he provided technical marketing support to the iSeries sales team, and holds the technical training for customers or BPs. He holds a bachelor degree in Computer Science from Beijing Polytechnic University. Before he joined IBM, he worked as a billing system developer in ChinaTelecom. His areas of expertise include IBM DB2® Universal Database™, WebSphere, and Linux OS.

Larry Dunn was an Advisory IT Specialist with IBM in the United Kingdom. He has seven years of experience with iSeries, learning Linux in his spare time. The last five years he has spent specializing in iSeries printing, rising to the role of iSeries Printing Technical Advisor within the IBM Software Support OS/400® Center. He is also a member of the IBM Technical Council in the UK. Larry holds a Master of Science (MSc) degree in Information Systems from the University of Portsmouth, a Bachelor of Arts (BA Honours) degree in Sports Science from Bedford College of Physical Education, and a Postgraduate Certificate in Education (PGCE) from Chichester Institute of Higher Education. Prior to joining IBM, Larry trained as a middle school teacher specializing in Physical Education. His areas of expertise include Linux OS, anything related to iSeries printing, LPAR, and TCP/IP. He has written extensively on iSeries printing, Facsimile Support, and TCP/IP and PSF trace interpretation.

Takaki Kimachi was a Senior Advisory IT Specialist in IBM Japan. He has 14 years experience in the iSeries and IBM S/36 area. He had been a field Systems Engineer supporting IBM AS/400 customers generally. For the past two years, he has worked as a Techline Japan specialist supporting BPs/IBMers with iSeries hardware or LPAR. He developed the prototype of LVT (LPAR Validation Tool) in 2000, which is the LPAR configuration tool used world wide today. He has been a Linux user for seven years, and on the IBM Japan iSeries Linux task team for one year. He has contributed to this project by testing Tomcat, creating sample JSP™ application, and setting up LPAR and iSeries Linux.

Alexander Marquis was a Staff Software Engineer from the USA. He has five years of experience with iSeries and has spent the last two years supporting Client Access Data Transfer, application programming interfaces (APIs), Open Database Connectivity (ODBC), and NetServer in the Rochester Support Center. He holds a degree in Interdisciplinary Studies from the University of Minnesota, Duluth, where he was introduced to POSIX-style operating systems. His areas of expertise include various Data Access methods to the iSeries, NetServer, and the Linux OS.

Patrick Wildt was a Staff Software Engineer from the USA. He has 13 years of experience on the iSeries platform with eleven years spent working in the OS/400 development lab in the Systems and Network Management area. The last two years he has spent in the IBM eServer Custom Technology Center working on various customer contracts and engagements. He holds a bachelors degree in Computer Science from North Dakota State University. His areas of expertise include systems management and communications.

Thanks to the following people from IBM Rochester for their contributions to this project:

Brent Nelson
Brian King
Carl Pecinovsky
Craig Johnson
Dave Wall
David Boutcher
David Engebretsen
Dennis Towne
Erwin Earley
Jay Bryant
Jeffrey Scheel
Kay Tate
Keith Cooper
Larry Loen

Monte Bruesewitz
Mark Vanderwiel
Pamela Bowen
Richard Johnston
Selwyn Dickey
Steven Janssen
Vess Natchev

Comments welcome

Your comments are important to us.

We want our IBM Redbooks™ to be as helpful as possible. Send us your comments about this or other IBM Redbooks in one of the following ways:

- Use the online **Contact us** review IBM Redbook form found at:

  ibm.com/redbooks

- Send your comments in an Internet note to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station PO99
  2455 South Road
  Poughkeepsie, NY 55901-7829.

1

Introduction to Linux integration with IBM i5/OS

This chapter provides a brief overview of the connection methods between Linux and IBM i5/OS. This chapter discusses the following areas:

- Advantages of integrating Linux with IBM i5/OS
- Overview of communication techniques between Linux and IBM i5/OS

1.1 Advantages of integrating Linux with IBM i5/OS

IBM is committed to supporting the Linux operating system on the iSeries platform. There is the capability of installing Linux onto iSeries partitions, Integrated xSeries Server (IXS), and Integrated xSeries Adapter (IXA) thus reducing the capital expenditure of purchasing extra hardware. This configuration is covered in detail in *Linux on the IBM eServer iSeries Server: An Implementation Guide,* SG24-6232, *Implementing Linux on Integrated xSeries Solutions for iSeries*, SG24-6379, and *Implementing Web Applications with CM Information Integrator for Content and OnDemand Web Enablement Kit*, SG24-6338.

IBM has further enhanced its support of Linux by developing several methods by which the actual data and applications running on both Linux and IBM i5/OS can be integrated. One example is iSeries Access for Linux product, which enables a Linux user to connect to an IBM i5/OS with a 5250 session or an Open Database Connectivity (ODBC) enabled applications.

The ability to connect Linux and i5/OS applications draws together the main strengths of both operating systems. For example, the database capabilities of i5/OS and the Web serving qualities of Linux, provide a powerful, consolidated computing environment.

Traditionally, the iSeries system (and its predecessor, the AS/400) has been used as a database machine by customers wishing to store and retrieve vast amounts of information relating to their business. IBM *DB2 Universal Database for iSeries* is fully integrated into the i5/OS operating system. Customers can also benefit from the single level storage, security, object, file and print management, reliability and scalability of the iSeries. Unlike some other system types, as a customer's business expands, the iSeries system hardware can be upgraded to accommodate the requirement for greater processing power or storage space.

Linux is recognized as an e-business system. At present a high percentage of business implementations of Linux are Web servers using the Apache Web Server and TomCat Web Application Server. The Linux operating system provides an excellent environment on which to run applications based on PHP Hypertext Preprocessor (PHP) and Perl.

However, the following implementation types are also popular:

- Mail Serving with products like Sendmail mail gateway
- File and Print Serving with Samba
- Firewall

Given the world-wide development community and the open source nature of Linux, new and appealing applications reflecting the ever-changing domain of e-business can be developed relatively quickly and economically. Applications requiring access to a database may now be enhanced by the ability to access an existing DB2 Universal Database running on an iSeries logical partition, thus reducing development time and costs.

1.2 Overview of Linux connection techniques

This section presents an overview of the methodologies which you can use to connect the Linux operating system with i5/OS. The techniques and methods are expanded upon in subsequent chapters of this book, in some cases in conjunction with sample program code.

1.2.1 iSeries Access for Linux

iSeries Access for Linux offers Linux-based access to i5/OS. It allows you to establish a 5250 session to an i5/OS system from a Linux system. It also includes the ODBC driver to access the DB2 Universal Database (UDB) for iSeries.

iSeries Access for Linux is available for Intel® and Power platforms. It runs on standalone PC, IBM PowerPC®, on IXA and IXS, and also on IBM eServer i5 logical partitions.

The iSeries Access for Linux product is further explained in Chapter 2, "iSeries Access for Linux" on page 5.

The following Web site presents current information about iSeries Access for Linux:

http://www.ibm.com/servers/eserver/iseries/access/linux/

1.2.2 The iSeries Access ODBC Driver for Linux

The iSeries Access ODBC Driver for Linux is a part of iSeries Access for Linux. It allows access to an IBM DB2 UDB for iSeries from a Linux application. This driver is based upon the ODBC driver shipped with the iSeries Access for Windows product 5722XE1 and similarly accesses the system using the Host Servers database interface via a socket connection.

The ODBC driver is covered in Chapter 3, "Connect IBM i5/OS with ODBC" on page 29.

1.2.3 IBM i5/OS connection using the JDBC driver

The *IBM Toolbox for Java* product 5722JC1 contains a Java Database Connectivity (JDBC™) driver which allows you to use JDBC application programming interface (API) to issue Structured Query Language (SQL) statements to and process results from the iSeries database. The Toolbox for Java also contains a set of Java classes enhancing the ease with which you can develop programs to access IBM i5/OS data and resources.

The JDBC driver is further explained in Chapter 4, "Connect IBM i5/OS with JDBC" on page 51. Sample program code demonstrating its use is also presented.

The following Web site also presents information about the JDBC driver:

http://www-03.ibm.com/servers/eserver/iseries/toolbox/

1.2.4 NetServer support for Linux clients using Samba

Support has been added to V5R1 of OS/400 allowing Linux Samba clients, using the Server Message Block (SMB) protocol. The support that is being provided here allows a Linux client running Samba to connect to iSeries Support for Windows Network Neighborhood, iSeries NetServer, through the *smbclient* and *smbmount* utilities.

The iSeries NetServer samba client support is covered in Chapter 5, "Connect iSeries NetServer with Samba" on page 67.

The following Web site presents current information about the iSeries NetServer:

http://www-1.ibm.com/servers/eserver/iseries/netserver/

1.2.5 Network File System

The Network File System (NFS) is a file system that allows users to access files located on a remote computer via an Internet Protocol (IP) network.

The nature and structure of the Linux file system lends itself to being stored in different places. The Network File System was developed to allow machines to read and write files on remote systems as though they were on a local hard drive. This facilitates the seamless sharing of files across a network.

The NFS is further explained in Chapter 6, “Connect IBM i5/OS with NFS” on page 81.

1.2.6 File Transfer Protocol

File Transfer Protocol (FTP) is the standard file transfer protocol for Transmission Control Protocol/Internet Protocol (TCP/IP). You can transfer text and binary data between Linux and i5/OS® using FTP.

FTP is discussed in further detail in Chapter 7, “Connect IBM i5/OS with FTP” on page 89.

1.2.7 Programmed socket connections

Sockets and the programs written on top of sockets are able to handle almost any integration you want to provide between Linux and i5/OS.

Chapter 8, “Connect IBM i5/OS with socket programming” on page 95, provides an example how to work with sockets and gives you an idea about how to use it for your purposes.

2

iSeries Access for Linux

iSeries Access for Linux is a new offering in the iSeries Access product line. iSeries Access for Linux allows you to access the DB2 Universal Database (UDB) for iSeries using its Open Database Connectivity (ODBC) driver and to establish a 5250 session to an IBM i5/OS system from a Linux client.

This chapter covers the following items:

- iSeries Access for Linux overview
- iSeries Access for Linux requirements
- Installation procedure
- 5250 configuration

ODBC topic is discussed in detail in Chapter 3, “Connect IBM i5/OS with ODBC” on page 29.

2.1 iSeries Access for Linux product overview

iSeries Access for Linux is an additional offering in the iSeries Access family 5722XW1. It offers Linux-based access to i5/OS systems. iSeries Access for Linux enables users to leverage business information, applications, and resources across an enterprise by extending the i5/OS resources to Linux.

2.1.1 iSeries Access for Linux product features

iSeries Access for Linux consists of the following product features:

- iSeries Access ODBC Driver

  Linux applications written to the ODBC application programming interfaces (APIs) can use this driver to access DB2 UDB for iSeries. The iSeries Access ODBC Driver is based on the ODBC driver in iSeries Access for Windows. It uses the iSeries database Host Servers as the access point to the system.

- The 5250 emulator

  It provides the function equivalent to that of an IBM 5250 terminal. The emulator also provides extended 5250 terminal functions similar to those that the IBM PC5250 emulator provides. The emulator supports the following functions:

  – Customizable multi-session support up to 99 sessions
  – Best Fit Fonts: different fonts are available and the font fit to the windows size
  – Multi language support
  – Macro record/playback
  – Copy/cut/paste
  – 24x80 and 27x132 screen size support
  – Screen print to Postscript printers
  – Hot spot highlighting option
  – Automatic, customizable, communication error recovery
  – Double-byte character set (DBCS) support is available in Version 1.8
  – Single sign-on and Kerberos support in Version 1.10

2.1.2 iSeries Access for Linux facts sheet

This section covers the topics related to this product. At the time of writing this book the actual version of iSeries Access for Linux used was 1.10.

Latest available version is 5.4.0-1.0

Platform choice

The iSeries Access for Linux product is designed to work on Linux systems based on the Intel and IBM POWER™ platform. POWER platforms include:

- Linux partitions on i5
- Linux partitions on p5
- IBM OpenPower™ Server

For Intel compatible systems:

- Integrated xSeries Server (IXS)
- xSeries servers connected via Integrated xSeries Adapter (IXA)
- Stand alone Intel PC and Server

Product architecture

The product is available in a 32-bit version as an Intel and a POWER implementation.

Latest version supports additional 64-bit platforms (PowerPC-64 and x86-64). The ODBC and Extended Dynamic Remote Support (EDRS) drivers are included in both the 32-bit and 64-bit packages. The 5250 emulator is included with the 32-bit packages but not included with the 64-bit packages.

License requirements

iSeries Access is not an Open Source product. You require a license of IBM iSeries Access Family 5722XW1 for 5250 emulation. License is not required for using the ODBC driver. Licensing rules for iSeries Access for Linux are the same as iSeries Access for Windows.

Availability

IBM iSeries Access for Linux is available as a downloadable package. Visit the following Web page for the latest version:

http://www.ibm.com/servers/eserver/iseries/access/linux/

2.2 iSeries Access for Linux requirements

This section lists the requirements for iSeries Access for Linux even before its installation. Check the i5/OS requirements first:

- For 5250 emulation, IBM iSeries Access Family 5722XW1 requires to be installed and you require a License of the product.
- Enable the QUSER user profile. Use the following procedure to verify the user profile status. On 5250 command entry screen, type:

  ```
  DSPUSRPRF USRPRF(QUSER)
  ```

  This command shows the status of QUSER, which must be *ENABLED. If the status is *DISABLED, type:

  ```
  CHGUSRPRF USRPRF(QUSER) STATUS(*ENABLED)
  ```

- Transmission Control Protocol/Internet Protocol (TCP/IP) must be up and running on your IBM i5/OS.
- Start the host servers in IBM i5/OS. To start them type:

  ```
  STRHOSTSVR *ALL
  ```

On Linux client, you require the following packages installed:

- glibc 2.2 or later

  This package is the GNU C Library. It contains the most important standards libraries as C, math, and Portable Operating System Interface (POSIX). Use the following command to check your installed version:

  ```
  linux:~> rpm -qa |grep glibc
  glibc-2.4-31.2
  glibc-64bit-2.4-31.2
  ```

- Red Hat Package Manager (RPM) 3.0 or later

  RPM manages the software packages. With it you can install, remove, and update software packages. Check the installed version:

```
linux:~> rpm --version
RPM version 4.4.2
```

- openmotif 2.0 or later

  Openmotif is a runtime environment. This software is necessary for 5250 emulator. To check if it is installed, type:

```
linux:~> rpm -qa | grep openmotif
```

  If the system returns you the lines of its package, it is installed.

- unixODBC driver manager Version 2.0.11 or later

  This software provides a standard interface between applications and ODBC driver on the Linux platforms. To check if it is installed, type:

```
linux:~> rpm -qa | grep unixODBC
```

  To get more information or download the latest level of unixODBC driver manager, go to:

  http://www.unixodbc.org

> **Note:** At the time of writing the book we noticed some problems with loading dynamic linked libraries on RHEL 3 and RHEL 4 for iSeries and IBM pSeries®. The source of the problems are the 64-bit versions of Openmotif and unixODBC, which are installed as default.
>
> You have to replace the installed packages of Openmotif and unixODBC with the 32-bit versions.
>
> ```
> linux:~> rpm -e openmotif
> linux:~> rpm -e unixODBC
> linux:~> up2date -i openmotif --arch=ppc
> linux:~> up2date -i unixODBC --arch=ppc
> ```
>
> For your information, 64-bit versions come with --arch=ppc64.

- The ibm5250 works with the default installed fonts of SUSE or Red Hat Linux but in some cases you require more font sizes to fit the 5250 screen to your display size. In order to display 5250 emulation screens properly, ibm5250 requires fixed, mono-spaced 75dpi and 100dpi fonts.

> **Note:** To check current install font for 75dpi or 100dpi, type:
>
> ```
> linux:~> rpm -qa | grep dpi
> xorg-x11-fonts-75dpi-6.9.0-50.14
> xorg-x11-fonts-100dpi-6.9.0-50.14
> ```

Some Linux distributions do not install the 100dpi fonts by default, but they provide it as an optional package.

Scalable fonts fill most of the screen but are considered by some to be fuzzy or not as sharp as fixed fonts. The emulator automatically detects and uses scalable fonts if they are available.

Note: As an X Window system, you might be using either X.org or Xfree86 implementation. Also, you may be using font server or not. Therefore, for a correct configuration, you have to check two items:

- Which implementation of X Window system you are using.
- Whether you are using font server.

To configure your system for scalable fonts follow the next steps:

a. Check which X Window system you are using. Type the commands as shown in Example 2-1.

Example 2-1 Type commands to check X Window system

```
linux:~> # X -version

X Window System Version 6.9.0
Release Date: 21 December 2005
X Protocol Version 11, Revision 0, Release 6.9
Build Operating System: SuSE Linux [ELF] SuSE
Current Operating System: Linux 2.6.16.21-0.8-ppc64 #1 SMP Mon Jul 3
18:25:39 UTC 2006 ppc64
Build Date: 17 June 2006
        Before reporting problems, check http://wiki.X.Org
        to make sure that you have the latest version.
Module Loader present
```

If you are using X.org implementation as shown in Example 2-1, you see the X Window configuration file, which is /etc/X11/xorg.conf.

For Xfree86, the file is /etc/X11/XF86Config.

b. Check whether you are using font server or not. At either of the W Window configuration files that you found in the previous step (either /etc/X11/xorg.conf file or /etc/X11/XF86Config file), search for the FontPath entry. If you find the entry and it's note commented out, that means you are using a font server. Following is an example of that entry:

```
FontPath     "Unix/:7100"
```

c. Now you are ready to edit the configuration file.

Note: Depending on what you have found, the actual configuration file to edit is different:

- If you are not using font server, edit X Window configuration file:
  – /etc/X11/xorg.conf file
  – /etc/X11/XF86Config file
- If you are using font server, edit:

  /etc/X11/fs/config

Look for the following lines in the located configuration file:

```
/usr/X11R6/lib/X11/fonts/75dpi:unscaled,
/usr/X11R6/lib/X11/fonts/100dpi:unscaled,
```

The word *unscaled* at the end of the lines means that you are using unscaled fonts.

To switch to scalable fonts, remove the `:unscaled`, so that those lines with selected dpi look as the following:

```
/usr/X11R6/lib/X11/fonts/75dpi,
/usr/X11R6/lib/X11/fonts/100dpi,
```

d. The changes take effect after restarting the X Window system.

2.3 Installing iSeries Access for Linux

The following procedure guides you to install iSeries Access for Linux:

1. Download the iSeries Access for Linux RPM package from:

   http://www.ibm.com/servers/eserver/iseries/access/linux

2. There is an option to use 32-bit with 5250 emulator and 64-bit with no 5250 emulator RPM to install the iSeries Access for Linux. We are using 32-bit IBM Power PC® with 5250 emulator RPM, on Linux terminal type for POWER platform:

   ```
   linux:~> rpm -ivh iSeriesAccess-5.4.0-X.YY.ppc.rpm
   ```

   For Intel platform:

   ```
   linux:~> rpm -ivh iSeriesAccess-5.4.0-X.YY.i386.rpm
   ```

   In these instances, the *X.YY* indicates the version level of the iSeries Access as shown in Example 2-2.

   Example 2-2 Installing the iSeries Access for Linux RPM

   ```
   linux:~> rpm -ivh iSeriesAccess-5.4.0-1.0.ppc.rpm
   Preparing...                ########################################### [100%]
      1:iSeriesAccess          ########################################### [100%]
   post install processing for iSeriesAccess 1.0...1
   iSeries Access ODBC Driver has been deleted (if it existed at all) because its
   usage count became zero
   odbcinst: Driver installed. Usage count increased to 1.
       Target directory is /etc/unixODBClinux:~>
   ```

   Make a note of the following points:

   – The files are installed in /opt/ibm/iSeriesAccess directory.

   – For logging information, run the rpm with -vv parameter at the end of the command, such as:

   ```
   linux:~> rpm -ivh iSeriesAccess-5.4.0-X.YY.ppc.rpm -vv
   ```

   – To update a former version of iSeries Access for Linux, run the **rpm** command with the -Uvh option, such as:

   ```
   linux:~> rpm -Uvh iSeriesAccess-5.4.0-X.YY.i386.rpm
   ```

   – To uninstall, run the rpm command with the -ev option, such as:

   ```
   linux:~> rpm -ev iSeriesAccess
   ```

- If you used the outdated *iSeries ODBC driver for Linux*, then you have to remove the package with the following command:

  ```
  linux:~> rpm -ev iSeriesODBC
  ```

- If the installation fails because of a dependency on libodbc.so or by another library, make sure that you have installed unixODBC driver manager. If the problem persists run the installation with the --nodeps parameter to stop the dependency checking:

  ```
  linux:~> rpm -ivh iSeriesAccess-5.4.0-X.YY.ppc.rpm --nodeps
  ```

3. Test the installation with the **cwbping** command as shown in Example 2-3. This command pings to the IBM i5/OS server, in this case as pointed by the IP address of 192.168.1.1. Successful returns of ping verifies the communication link between two systems is established.

Example 2-3 cwbping command

```
linux:~> /opt/ibm/iSeriesAccess/bin/cwbping 192.168.1.1

IBM iSeries Access for Linux
Version 5  Release 4 Level 0
Connection Verification Program
(C) Copyright IBM Corporation and Others 1984, 2003.  All rights reserved.
U.S. Government Users Restricted Rights - Use, duplication or disclosure
  restricted by GSA ADP Schedule Contract with IBM Corp.
Licensed Materials - Property of IBM

To cancel the CWBPING request, press CTRL-C or CTRL-BREAK
I - Verifying connection to system 192.168.1.1...
I - Successfully connected to server application: Central Client
I - Successfully connected to server application: Network File
I - Successfully connected to server application: Network Print
I - Successfully connected to server application: Data Access
I - Successfully connected to server application: Data Queues
I - Successfully connected to server application: Remote Command
I - Successfully connected to server application: Security
I - Successfully connected to server application: DDM
I - Successfully connected to server application: Telnet
I - Connection verified to system 192.168.1.1
linux:~>
```

Note: The **cwbping** command is installed into /opt/ibm/iSeriesAccess/bin/. To execute cwbping from any directory, your PATH shell variable requires to have an entry for /opt/ibm/iSeriesAccess/bin/ for 32-bit and /opt/ibm/iSeriesAccess/bin64/ for 64-bit installed iSeries Access for Linux.

Other useful commands, such as **ibm5250** to start a 5250 emulation session or **rmtcmd** to run i5/OS command are installed in this directory as well.

2.4 5250 emulator

The iSeries Access for Linux 5250 emulator is a client implementation. It is intended to be used on a Linux client.

This version allows multiple Virtual Network Computing (VNC) and Linux Terminal Server Project (LTSP) users to easily use the 5250 emulator.

This section covers:

- Emulator language support
- Emulator modules
- Some important emulation functions

The other major feature of iSeries Access for Linux package is the ODBC driver, which is discussed in further Detail in Chapter 3, "Connect IBM i5/OS with ODBC" on page 29.

2.4.1 Emulator languages

The main emulator window labels and helper applications are translated in many languages and it is displayed based on the $LANG system environment variable. The command line value -LANGID can be used to override the default language. See Table 2-1.

Table 2-1 Supported languages for 5250 emulator

i5/OS language	Locale	Description
2902	et	Estonian
2903	lt	Lithuanian
2904	lv	Latvian
2905	vi	Vietnamese
2906	lo	Lao
2909	en_BE	Belgian English
2911	sl	Slovenia
2912	sh	Croatian
2913	mk	Macedonia
2914	sr	Serbian
2922	pt_PT	Portuguese
2923	nl	Dutch Netherlands
2924	en	English
2925	fi	Finnish
2926	da	Danish
2928	fr	French
2929	de	German
2931	es	Spanish
2932	it	Italian

i5/OS language	Locale	Description
2933	no	Norwegian
2937	sv	Swedish
2938	en_JP	English Uppercase DBCS
2939	de_CH	German MNCS
2940	fr_CH	French MNCS
2942	it_CH	Italian MNCS
2956	tr	Turkish
2957	el	Greek
2958	is	Icelandic
2962	ja	Japanese Kanji DBCS
2963	nl_BE	Dutch in Belgium MNCS
2966	fr_BE	Belgian French MNCS
2972	th	Thai
2974	bg	Bulgarian
2975	cs	Czech
2976	hu	Hungarian
2978	pl	Polish
2979	ru	Russian (Requires iconv support CP1025)
2980	pt_BR	Brazilian Portuguese
2981	fr_CA	Canadian French MNCS
2984	en_CN	English DBCS
2986	ko	Korean DBCS
2987	zh_HK	Traditional Chinese DBCS
2989	zh	Simplified Chinese DBCS
2992	ro	Romanian
2994	sk	Slovakian
2995	sq	Albanian
2996	pt	Portuguese MNCS
2998	fa	Farsi

Note: For Russian language support, you require a version of iconv that supports CP1025. To check this, type:

```
iconv -l | grep CP1025
```

2.4.2 Starting a 5250 emulator

The emulator consists of two main binaries:

- ibm5250
- setup5250

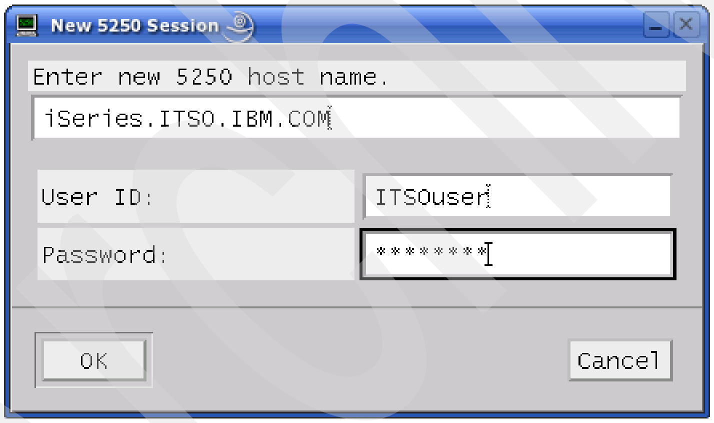
Note: We do not recommend running ibm5250 as root. The default environment variables required to run X applications are not, by default, set up correctly when switching to the root user. Running ibm5250 as a normal user after running as root can cause problems (file permissions). A common symptom is that the emulator cannot read a config file. Therefore, it reverts back to default values.

ibm5250

ibm5250 invokes the 5250 emulator and opens a graphical user interface (GUI) window that prompts you for the system name, user ID, and password. This information is used to obtain a 5722XW1 license on the system, which is required for a 5250 session to be brought up.

Type **`ibm5250`** `<`*`iSeriesname`*`>` to start the emulator or use **`ibm5250 --help`** to see command line options. The iSeries name is an optional parameter. This opens the New 5250 Session window as illustrated in Figure 2-1. Type your IBM i5/OS user profile and password and then click **OK** to open a 5250 session. If you get an error like `ibm5250: command not found` then you must add the directory /opt/ibm/iSeriesAccess/bin/ to the shell PATH variable:

```
linux:~> ibm5250
5250: [ INFORMATIONAL ]: Build Date: 13 July 2005 (1.10).
```



Enter new 5250 host name.

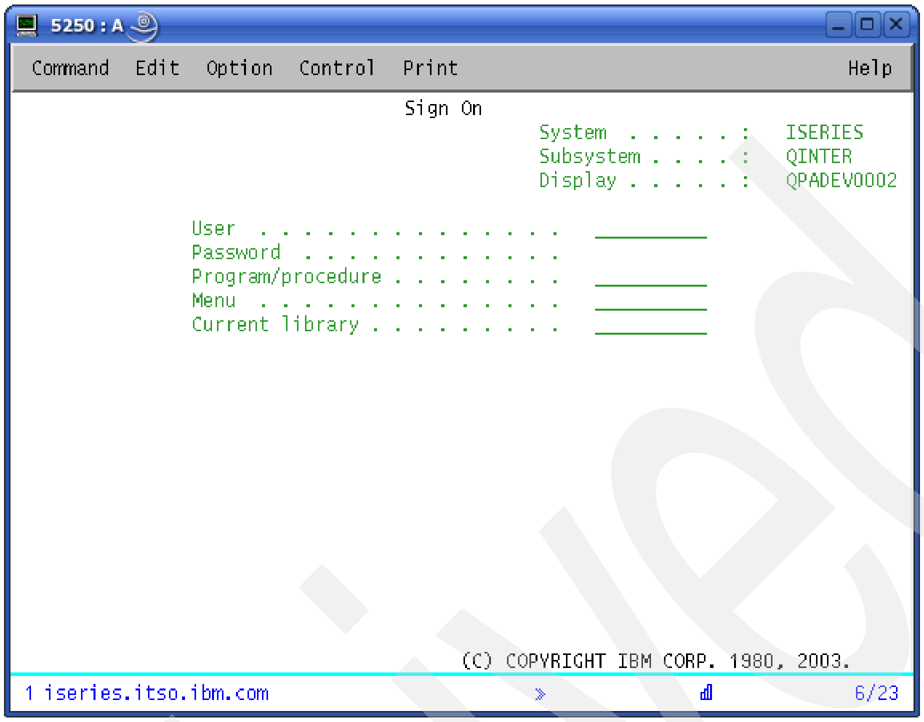
iSeries.ITSO.IBM.COM

User ID: ITSOuser

Password: ********

Figure 2-1 New 5250 session

Then the 5250 emulation is displayed as shown in Figure 2-2.



```
                              Sign On
                                          System  . . . . . :   ISERIES
                                          Subsystem . . . . :   QINTER
                                          Display . . . . . :   QPADEV0002

               User  . . . . . . . . . . . . . .    __________
               Password  . . . . . . . . . . . .
               Program/procedure . . . . . . . .    __________
               Menu  . . . . . . . . . . . . . .    __________
               Current library . . . . . . . . .    __________

                                    (C) COPYRIGHT IBM CORP. 1980, 2003.
1 iseries.itso.ibm.com                                              6/23
```

Figure 2-2 5250 session

setup5250

setup5250 is the setup program that you can use to configure a setting which applies to all 5250 sessions of the current user ID and defines multiple connections.

1. In a GUI terminal session, type **`setup5250`**:

```
linux:~> setup5250
setup5250: [ INFORMATIONAL ]: Build Date: 13 July 2005 (1.10)
setup5250: [ INFORMATIONAL ]: setup5250
```

   The main menu is displayed as illustrated in Figure 2-3.

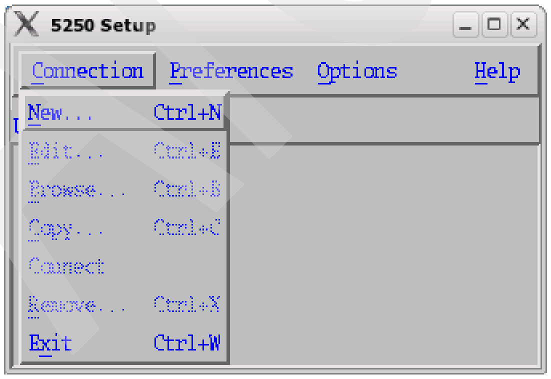


Figure 2-3 setup5250

The menu bar has four pull-down lists of Connection, Preferences, Options, and Help.

– Connection

  Use this option to create, edit, browse, copy, and remove emulator connection configurations. Also use this option to connect to any existing connection configuration.

– Preferences

  This menu allows you to turn on or off emulator options. Preference settings apply to all 5250 sessions.

– Options

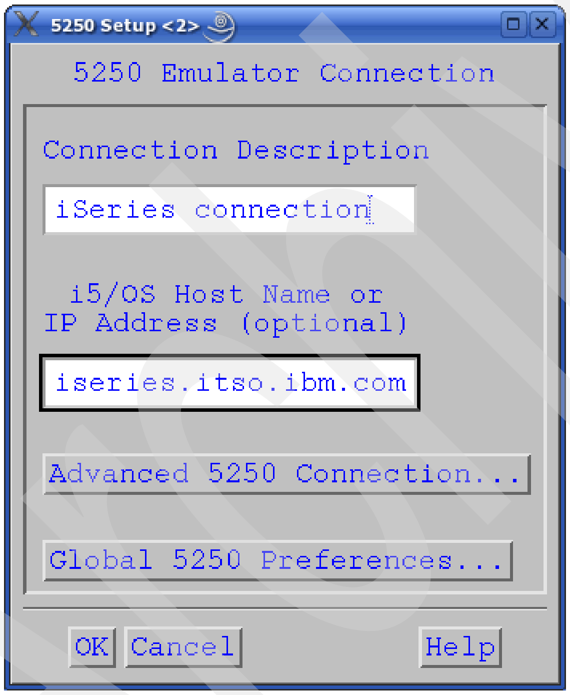
  You can define a password to prevent others from using the setup5250 utility.

– Help

  Use this menu to get a detailed description about the setup program.

2. To create a new session

   Select **New** from the Connection menu bar. Give a connection description and the Host name or IP address of the iSeries as illustrated in Figure 2-4.



Connection Description: iSeries connection

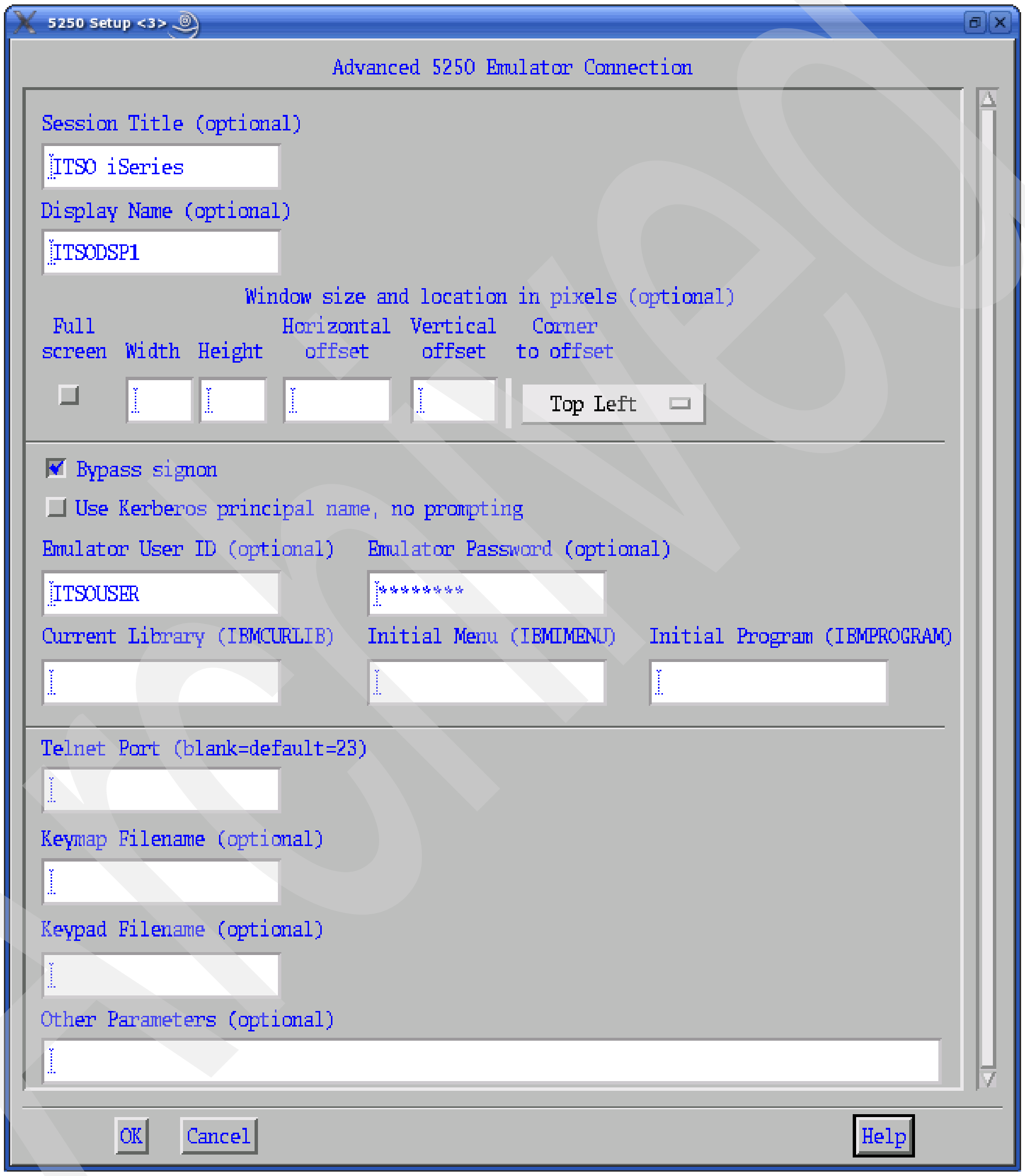
i5/OS Host Name or IP Address (optional): iseries.itso.ibm.com

Figure 2-4 5250 Emulator Connection

Then, on the same panel, click **Advanced 5250 Connection** button if you want to define additional options, as illustrated in Figure 2-5:

– Session title: This text value is displayed as the session title.
– Display name: This value determines the 5250 display device name in IBM i5/OS.
– Window size: This specifies the initial size and location of 5250 window.
  - Full screen: If this box is checked, the 5250 window uses whole screen.
  - Width: This value determines the horizontal width of the window in pixels.
  - Height: This value determines the vertical height of the window in pixels.
  - Horizontal, vertical, and corner offset specify the windows location on screen.
– Bypass signon: This option allows you to bypass the IBM i5/OS signon screen.
– Use Kerberos principal name, no prompting: This enables the Kerberos authentication.

– Emulator User ID: This value is used as IBM i5/OS user profile for bypass signon.
– Emulator password: This password is stored encoded and used for bypass signon.
– Current library: This text specifies the current library for bypass signon.
– Initial menu: Allows to specify a specific start menu after bypass signon.
– Initial program: starts the specified program after bypass signon.
– Telnet port: Specify the telnet port for your telnet server. This port is 23 by default.
– Keymap filename: Specify the full path of the keymap file.
– Keypad filename: Specify the full path of the keypad file.
– Other parameters: See Help for additional parameters.



Session Title (optional): ITSO iSeries
Display Name (optional): ITSODSP1
[✓] Bypass signon
[ ] Use Kerberos principal name, no prompting
Emulator User ID (optional): ITSOUSER
Emulator Password (optional): ********
Current Library (IBMCURLIB)
Initial Menu (IBMIMENU)
Initial Program (IBMPROGRAM)
Telnet Port (blank=default=23)
Keymap Filename (optional)
Keypad Filename (optional)
Other Parameters (optional)

Figure 2-5 Advanced 5250 emulator connection options

Click the **OK** button to create or submit your edit options.

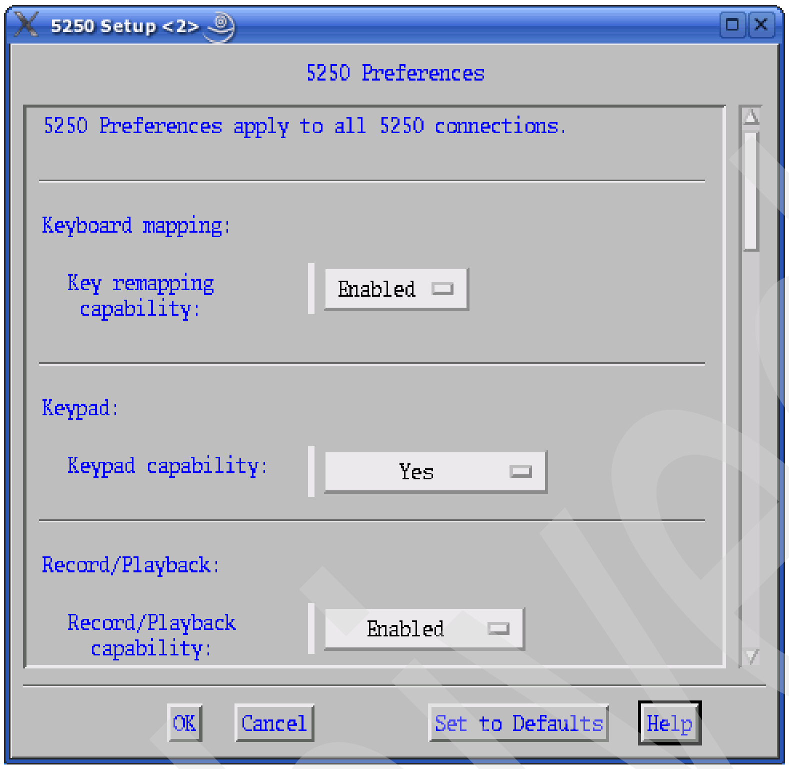
3. To customize your 5250 emulator use the Preferences pull-down menu. Preferences apply to all emulator connections, and may be overridden by individual connection configuration options.

   On the Global 5250 preferences button you can define options as illustrated in Figure 2-6 (the whole screen is not shown here):

   – Keyboard remapping: This option allows you to define your 5250 keyboard layout.
   – Keypad capability: This option allows you to create and use customized keypad buttons.
   – Record/Playback capability: This option allows you to record and later playback key sequences in the 5250 emulator.
   – Color customization capability: This option allows you to modify colors within your 5250 emulators.
   – 132 Columns: If enabled, the 27 rows by 132 columns screen size is allowed.
   – Column separator: This value enables or disables displaying column separator within certain types of 5250 entry fields.
   – Desktop file: This option determines if the 5250 emulator stores and uses the last location and font size of 5250 emulator windows.
   – Command menu: This option allows you to enable or disable the 5250 menu bar Command choice. If you disable it, you will have no access to new 5250 sessions and you will be unable to exit a 5250 session using the command choice.
   – Edit menu: This option allows you to enable or disable the Edit choice on the 5250 menu bar. If it is disabled you cannot use any copy, cut, or paste option.
   – Option menu: This option allows you to enable or disable the 5250 menu bar Option choice. If it is disabled you do not have access to fonts selection, keypad customization, record/playback selections, color mapping, or keyboard remapping.
   – Print menu: The Print menu option allows you to enable or disable the 5250 menu bar Print choice (to print the 5250 screen). “No menu, keyboard print only” means you can use a print key sequence to print the 5250 screen.
   – Help menu: Turns help menu on or off.
   – New session window: This option allows you to enable or disable starting another 5250 emulator to a different i5/OS or OS/400 host.
   – Browser hotspot: The Browser hotspot option allows you to enable or disable recognizing a URL hotspot and starting a browser. When the option is enabled and a user clicks on a valid URL hotspot, the command specified in the system environment variable $BROWSER is called with the URL passed as a parameter.

Note: For other options and detailed description refer to 5250 Preferences Help.

See Figure 2-6.



5250 Preferences apply to all 5250 connections.

Keyboard mapping:

Key remapping capability: Enabled

Keypad:

Keypad capability: Yes

Record/Playback:

Record/Playback capability: Enabled

Figure 2-6 5250 preferences

4. Options

   Use the option pull-down menu to set, change, or delete a setup program password.

5. Help

   In the Help pull-down menu bar, you can learn about the emulation setup program, and bring the program help for the emulator itself.

2.4.3 ibm5250 in a Multiuser environment

By default the ibm5250 emulator is configured to act as a single user interface on a Linux client. A Linux client is defined as a system with a single Linux user, using one Linux user ID.

The emulator may be also run in a Multiuser environment on a Linux server, which means a Linux Terminal Server Project, Virtual Network Computing, remote X Window sessions.

If more than one user is running on a single Linux system, then each user has to change his 5250 emulator to multiuser mode.

Therefore, the user can specify the -STAND_ALONE command line option at ibm5250 startup or change the connection configuration with setup5250.

In setup5250 the configuration for each session has to contain the following resource setting in the field *Other Parameters (optional)*:

```
IBM5250*STAND_ALONE:TRUE
```

When -STAND_ALONE is specified on the command line or the resource setting is IBM5250*STAND_ALONE: TRUE, then:

- The emulator hot-key switching between sessions is disabled.
- Each emulator session runs in its own process (stand alone process mode).
- Multiple users are allowed to run on the same Linux system.

When -STAND_ALONE is NOT specified on the command line or the resource setting is IBM5250*STAND_ALONE: FALSE, then:

- Only one user is allowed to run ibm5250 sessions
- If multiple users try to launch ibm5250, the emulator hangs.
- The emulator hot-key switching between sessions is allowed.
- Emulator sessions run as a single process (shared process mode).

2.4.4 Getting HMC Remote 5250 console from your Linux desktop

You can get an i5/OS 5250 console from your Linux desktop (in the same way you would get it on a Hardware Management Console (HMC) box). You get this service using the 5250 emulator of iSeries Access for Linux (ibm5250) and HMC Telnet Proxy.

To configure the HMC support for the Linux product, use the setup5250 to create a new connection. On the 5250 Emulator Connection properties page, specify the parameter. In place of "i5/OS Host Name or IP Address", give HMC's IP address or Domain Name System (DNS) name.

On the Advanced 5250 Connection properties page specify:

- Emulator User ID: Q#HMC
- Emulator Password: Q#HMC
- Telnet Port: 2300

Note: The special Q#HMC string is not used for security authentication in any way, you are prompted for additional security values by the HMC Telnet Proxy.

2.5 Imeplenting SSO using EIM and Kerberos

Starting with Version 1.10, iSeries Access for Linux includes support Enterprise Identity Mapping (EIM) and Kerberos authentication. Using these features, you can implement single sign-on (SSO) for your network. These new functions allow a user to minimize the usage of different user profiles and passwords. With single sign-on a user requires to log in only once. Kerberos and EIM allow a user to be authenticated once and get connected to every IBM i5/OS system he is allowed to. Even if he has different user names on different systems.

For detailed informations about Kerberos and EIM as well as configuration examples for iSeries Access for Windows see iSeries Information center and follow the location page to R530 and select security → Enterprise Identity Mapping (EIM) or security → single sign-on:

`http://publib.boulder.ibm.com/pubs/html/as400/`

You could also see the IBM Redbook titled *Windows-based Single Signon and the EIM Framework on the IBM eServer iSeries Server*, SG24-6975.

There are two major Kerberos implementations available for Linux: MIT and Heimdal. Most Linux distributions have at least one version of Kerberos included with them. For specific informations about Heimdal visit:

`http://www.pdc.kth.se/heimdal/`

Use the following link to visit MIT - Massachusetts Institute of Technology:

`http://web.mit.edu/kerberos/www/`

To install and configure Linux for Kerberos, see one of the many How-to's available at the internet. For example, on The Linux Documentation Project:

`http://www.tldp.org/`

2.5.1 Prerequisites and configuration for Kerberos

The following steps provide an example about Kerberos setup on a Linux workstation. We assume that Kerberos and EIM are configured and working in a Windows environment with iSeries Access for Windows. Therefore, we use the Windows Active Directory® Server as a Kerberos server.

iSeries Access for Linux requires Kerberos Version 5. You could use the Heimdal or the MIT implementation of Kerberos5. Mostly every Linux distribution includes installation packages of at least one of the implementations.

Linux also supports Kerberos as an initial logon method using the Pluggable Authentication Module (PAM) pam_krb5. In this case Kerberos is used for Linux user authentication instead of the local password file. For our example, we do not need to change the Linux user authentication to Kerberos but it might be used as an alternative Linux login method in your environment. If you plan to use pam_krb5, refer to the documentation of your distribution about the setup.

After Kerberos5 and its utilities are installed on your Linux system you must have at least the following Kerberos commands:

- kinit
- klist
- kpasswd

You must also have the Kerberos5 configuration file, which is /etc/krb5.conf.

The krb5.conf contains the basic Kerberos5 configuration and has the requirement to be customized for your environment. The following information about your Active Directory are required:

- Windows logon domain name - in Kerberos Terms: default realm
- DNS Name of Windows Server® (Domain controller), which provides the functions: Key distribution Center (KDC) and Password server

If you are not sure about this configuration then you could verify this setting in your existing Kerberos configuration on i5/OS system. On IBM i5/OS, the Kerberos service is called Network Authentication Service (NAS) and the settings are located in a similar file like on Linux:

`/QIBM/UserData/OS400/NetworkAuthentication/krb5.conf`

Because Kerberos and EIM must already be configured on IBM i5/OS, the file must resemble Example 2-4. The double question mark and the parentheses are a replacement for squared brackets, as ‘??(‘ to ‘[‘ and ‘??)’ to ‘]’, while double question mark and the arrow replaces brace, as ‘??<‘ to ‘{‘ and ‘??>’ to ‘}’.

Example 2-4 IBM i5/OS Kerberos configuration

```
??(libdefaults??)
default_keytab_name =
/QIBM/UserData/OS400/NetworkAuthentication/keytab/krb5.keytab
default_realm = ITSODOMAIN.TEST
??(realms??)
ITSODOMAIN.TEST = ??<
  kdc = master.itsodomain.test:88
  kpasswd_server = master.itsodomain.test:464
??>
??(domain_realm??)
  .itso.ibm.com = ITSODOMAIN.TEST
??(capaths??)
```

Example 2-4 use the settings shown in Table 2-2.

Table 2-2 Settings for IBM i5/OS Kerberos configuration

Description	**Parameter**	**Value**
Kerberos Realm/Name of our Active Directory	default_realm	ITSODOMAIN.TEST
KDC/our Active Directory Server	kdc	master.itsodomain.test
Password server/same as our KDC server	kpasswd_server	master.itsodomain.test

Insert your settings in the configuration file /etc/krb5.conf on your Linux system similar to Example 2-5. It is a convention that the Realm is always written in uppercase and DNS names are written in lowercase.

Example 2-5 Example of /etc/krb5.conf

```
[libdefaults]
default_realm = ITSODOMAIN.TEST
clockskew = 300

[realms]
ITSODOMAIN.TEST = {
kdc = master.itsodomain.test
admin_server = master.itsodomain.test
kpasswd_server = master.itsodomain.test
}

[logging]
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmin.log
default = FILE:/var/log/krb5lib.log

[appdefaults]
pam = {
ticket_lifetime = 1d
renew_lifetime = 1d
forwardable = true
proxiable = false
retain_after_close = false
minimum_uid = 0
}
```

If your IBM i5/OS uses a different domain name than your Windows domain, then Kerberos requires to know about this. In our example, IBM i5/OS is known in the network with the host name: *iseries.itso.ibm.com*, but our Kerberos Realm is ITSODOMAIN.TEST. Therefore, add another section to the krb5.conf file and configure Kerberos to use the ITSODOMAIN.TEST settings even if the server has a DNS extension of .itso.ibm.com:

```
[domain_realm]
  .itso.ibm.com = ITSODOMAIN.TEST
```

After the krb5.conf file is modified check the following important considerations:

- The Kerberos protocol expects that all participating computer systems have a consistent time setting. This means a proper setting of the current time and the time zone. If the time skew between two systems exceed more than 300 seconds, then an authentication is no longer possible. We recommend that you use a network time source to keep a time discrepancy as low as possible.
- For Kerberos authentication, a correct DNS resolution is also very important. To get access to a target system the Kerberos client resolves the fully qualified domain name of that target. If the Kerberos client resolves a different name than the system name known by the KDC, a Kerberos ticket for authentication cannot be created. If you use the /etc/hosts file for name resolution, then move the fully qualified domain name on first position after the IP address:

  ```
  10.10.10.1 iseries.itso.ibm.com     iseries
  ```

- iSeries Access for Linux dynamically loads the Kerberos shared library by the name. For the Heimdal implementation it uses the following name:

  ```
  /usr/lib/libgssapi.so
  ```

  For the MIT implementation:

  ```
  /usr/lib/libgssapi_krb5.so
  ```

  Not every Linux distribution provides this file, therefore you have to create a symbolic link to the appropriate Kerberos library. Verify the installed files and symbolic links with the following command:

  ```
  linux:~> ls -l /usr/lib/libgssapi*
  ```

  – For MIT Kerberos you might see these files:

    ```
    /usr/lib/libgssapi_krb5.so.2 -> libgssapi_krb5.so.2.2
    /usr/lib/libgssapi_krb5.so.2.2
    ```

    In this case you have to add the following symbolic link:

    ```
    linux:~> ln -s /usr/lib/libgssapi_krb5.so.2.2 /usr/lib/libgssapi_krb5.so
    ```

  – For Heimdal Kerberos you might see these files:

    ```
    /usr/lib/libgssapi.so.1 -> libgssapi.so.1.4.0
    /usr/lib/libgssapi.so.1.4.0
    ```

    For the Heimdal example create a symbolic link such as:

    ```
    linux:~> ln -s /usr/lib/libgssapi.so.1.4.0 /usr/lib/libgssapi.so
    ```

Note: If a correct symbolic link to the Kerberos library is not available then you get the following error:

```
CWBSY1015 - Kerberos not available on this version of the operating system
```

2.5.2 Login with Kerberos

After the configuration is completed you must be able to successfully authenticate to your Kerberos domain using the **kinit** command. As we are using a Windows Active Directory, your login name is your Windows user name. Kinit appends the default Realm to it.

```
linux:~> kinit windowsuser
Password for windowsuser@ITSODOMAIN.TEST:
```

If you get no further messages or an information about ticket lifetime, then the authentication to the Windows domain was successful. If you used the Kerberos Pluggable Authentication Module for your initial Linux login then you must already own a Kerberos ticket and do not require to do a kinit again. Use the command **klist** to verify if you got an Kerberos ticket named krbtgt:

```
linux:~> klist
Ticket cache: FILE:/tmp/krb5cc_1000
Default principal: windowsuser@ITSODOMAIN.TEST

Valid starting     Expires            Service principal
08/01/05 20:01:00 08/02/05 06:01:00 krbtgt/ITSODOMAIN.TEST@ITSODOMAIN.TEST
```

To open a 5250 session without logon you must have an EIM identifier, which maps the Windows user name to an IBM i5/OS user profile.

For opening a 5250 session use the **ibm5250** command with the parameter -kerberos -sso. The -kerberos flag authenticates the user against Host Server and the -sso switch uses this authentication to bypass the signon window:

```
/opt/ibm/iSeriesAccess/bin/ibm5250 iseries.itso.ibm.com -kerberos -sso
```

2.5.3 Kerberos support of iSeries Access for Linux

Starting with Version 1.10 all functions of iSeries Access for Linux supports Kerberos. This includes all command line utilities and also the ODBC driver and the 5250 emulation. To use Kerberos, replace the iSeries user name with *kerberos as Example 2-6 shows. Any password is ignored in this case.

Example 2-6 rmtcmd with Kerberos

```
linux:~> /opt/ibm/iSeriesAccess/bin/rmtcmd CRTLIB SSOTEST
/system:iseries.itso.ibm.com /user:*kerberos

IBM iSeries Access for Linux
Version 5  Release 2  Level 0
Remote Command utility V1.2
(C) Copyright IBM Corporation and Others 1984, 2003.  All rights reserved.
U.S. Government Users Restricted Rights - Use, duplication or disclosure
  restricted by GSA ADP Schedule Contract with IBM Corp.
Licensed Materials - Property of IBM
```

```
The remote system name is iseries.itso.ibm.com.
CPC2102 - Library SSOTEST created.
```

Even in an ODBC data source name (DSN), a user name of *Kerberos is allowed. This means every ODBC application can get the advantages of Kerberos single sign-on.

2.6 Encrypt the connection with Secure Sockets Layer

Even if you use a secured login method such as Kerberos, the 5250 Telnet connection is still unencrypted. This means that persons who have access to the network environment can easily monitor and change every input and screen data. To prevent any unauthorized data access in the network, the Secure Sockets Layer (SSL) protocol was invented. It provides data encryption, ensures data integrity, and supports authentication of client and server. iSeries Access for Linux can use the advantages of SSL by using the Linux tool **`stunnel`**.

For SSL encrypted communication you must apply certificates to each IBM i5/OS TCP/IP server or Host server that you want to use. Otherwise the SSL TCP/IP ports are not available. How to set up an own public/private key infrastructure is described in the iSeries Infocenter under Security → Secure Sockets Layer:

`http://publib.boulder.ibm.com/pubs/html/as400/`

The program **`stunnel`** establishes an SSL tunnel between a server and a client. In this case **`stunnel`** connects to the Telnet SSL Port on IBM i5/OS, transfers the encrypted data to the Linux client, decrypts it and provides the plain telnet stream to local applications. Instead of connecting IBM i5/OS directly, **`ibm5250`** connects to the end of the SSL tunnel at localhost. See Figure 2-7. After the tunnel is established every user on the Linux client can use it. The same scenario also applies to Host server connections such as ODBC.

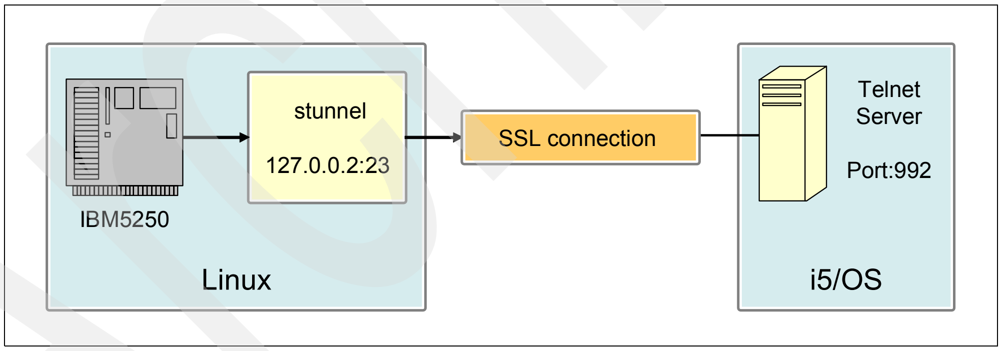


Figure 2-7 Stunnel connection

The **`stunnel`** command reads his configuration from a text file. The structure of the file is very simple. First specify that you use the client mode of stunnel. Then add a section for every SSL service you want to connect to:

- Service name: `[service]`
- Local IP address and port: `accept = localIpAdress:localPort`
- Remote IP address and port: `connect = remoteHostName:remotePort`

An example configuration is provided in Example 2-7.

Example 2-7 Stunnel configuration file: stunnel.conf

```
# Enable client mode
client = yes

# Debug and Foreground are for testing / Uncomment for debugging stunnel problems
##debug = 5
##foreground = yes

# Services: The following sections contain the port maps for iSeries Access
connections.

# Used for 5250 Telnet emulation (ibm5250)
[telnet]
accept = 127.0.0.2:23
connect = iseries.itso.ibm.com:992

# Used for license and conversion tables (ibm5250, cwbnltbl)
[as-central]
accept = 127.0.0.2:8470
connect = iseries.itso.ibm.com:9470

# Used for ODBC (rmtodbc, cwbrunsql, isql, DataManager, ...)
[as-database]
accept = 127.0.0.2:8471
connect = iseries.itso.ibm.com:9471

# Used for Remote commands (rmtcmd)
[as-rmtcmd]
accept = 127.0.0.2:8475
connect = iseries.itso.ibm.com:9475

# Used for SSO Signon
[as-signon]
accept = 127.0.0.2:8476
connect = iseries.itso.ibm.com:9476
```

For the first usage of stunnel you must uncomment `foreground = yes` line and start the SSL tunnel as root user with the command `stunnel stunnel.conf`. Now open a 5250 session with **`ibm5250 127.0.0.2`**. If all works correctly you see an output of stunnel similar to Example 2-8.

Example 2-8 Output of a successful ibm5250 connection through stunnel

```
root@linux:~> stunnel stunnel.config
2005.08.01 LOG5[]: stunnel 4.07 on i686-suse-linux-gnu PTHREAD+POLL+IPv4 with
OpenSSL 0.9.7e 25 Oct 2004
2005.08.01 LOG5[]: 500 clients allowed
2005.08.01 LOG5[]: as-central connected from 127.0.0.2:21774
2005.08.01 LOG5[]: Connection closed: 783 bytes sent to SSL, 215 bytes sent to
socket
2005.08.01 LOG5[]: telnet connected from 127.0.0.2:2218
```

This configuration establishes an encrypted connection between Linux and IBM i5/OS, but it does not prove whether the remote computer is the IBM i5/OS that you want to connect. To prevent a so called man in the middle attack, **`stunnel`** can validate the remote server with a certificate. During establishing an SSL connection, the server provides his certificate to the client in the order in which the client can prove his authenticity. On many IBM i5/OS a self signed certificate is used. This means, that IBM i5/OS is also the Certification Authority (CA).

If you use self signed certificates then the public CA key is stored on IBM i5/OS Integrated File System (IFS) in /QIBM/UserData/ICSS/Cert/CertAuth/CA.TXT. You can either download it with ftp in American Standard Code for Information Interchange (ASCII) mode or use the procedure provided in Example 2-9. The example retrieves the public CA key with **`smbclient`** and convert it with **`iconv`** from Extended Binary Coded Decimal Interchange Code (EBCDIC) encoding to ASCII.

If you have your own certification infrastructure or use a certificate signed by another company then you must use the appropriate public CA key. See Example 2-9.

Example 2-9 Install i5/OS CA certificate

```
linux:~> smbclient //iseries.itso.ibm.com/qibm
Password:
Domain=[ITSO] OS=[OS/400 V5R3M0] Server=[iSeries Support for Windows Network
Neighborhood]

smb: \> get userdata/icss/cert/certauth/ca.txt ca.ebcdic
getting file \userdata/icss/cert/certauth/ca.txt of size 765 as ca.ebcdic (249.0
kb/s) (average 249.0 kb/s)
smb: \> exit

linux:~> iconv -f IBM037 -t IBM437 ca.ebcdic -o ca.txt
linux:~> cat ca.txt
-----BEGIN CERTIFICATE-----
MIICBzCCAXCgAwIBAgIEPbUeYjANBgkqhkiG9w0BAQQFADBIMQswCQYDVQQGEwJk
ZTEMMAoGA1UECBMDcmxwMRgwFgYDVQQKEw9JQk0gSVRTIGlTZXJpZXMxETAPBgNV
BAMTCEFTNWwyLUNBMB4XDTAyMTAyMTA5NDYxMFoXDTIyMTAxNzA5NDYxMFowSDEL
MAkGA1UEBhMCZGUxDDAKBgNVBAgTA3JscDEYMBYGA1UEChMPSUJNIElUUyBpU2Vy
l9U8UTLTtXWavNCvTdtSUT4Y64rNjkk5z+S3rjcDY/uepQLPm3gGzIaUFlCdqLV0
qZfy3HBtq8z+PzzfwUswBVl0MwDF/lEJVcqeqQvk3KdJys0CAwEAATANBgkqhkiG
9w0BAQQFAAOBgQAMvToco908mYj9qgFuJOWfzwQHumHjWSKOy8S6PrlUQsAOALOm
CTxsyHt/eG1ubU7wG35Cl4+Hp9yTceOPAw2+J5O32Y7qGleJ+lhs86vTSplozlUs
wxDyIl88nflxTJOpKOGwX+uiPY1QNOOOKMmdS3tcplYF9CFbqmMGzFiOlQ==
-----END CERTIFICATE-----
```

The default directory for stunnel configuration files is /etc/stunnel/. You must place your CA key file and the stunnel.conf file there. Add the following text in front of the Services section of the stunnel.conf file:

```
# enable CA validation
CAFile = /etc/stunnel/ca.txt
verify = 2
```

To test the certificate for server authentication start **`stunnel`** and connect it to IBM i5/OS with **`ibm5250 127.0.0.2`**. This test must show a **`stunnel`** message text as follows:

```
2005.08.10 LOG5[]: telnet connected from 127.0.0.2:1039
2005.08.10 LOG5[]: VERIFY OK: depth=1, /C=us/ST=mn/O=IBM-ITSO/CN=ITSO-CA
2005.08.10 LOG5[]: VERIFY OK: depth=0, /C=us/ST=mn/O=IBM-ITSO/CN=ITSO-ISERIES
```

After you finish the tests do not forget to comment out the foreground = yes parameter in the stunnel.conf file.

For a better usability you can add alias names in your local /etc/hosts file such as:

```
127.0.0.2 SSLiSeriesA
127.0.0.3 SSLiSeriesB
```

Note: Be careful if you want to map the full IBM i5/OS DNS name to a localhost address instead of an alias. In this case you must use the IP address of IBM i5/OS in your stunnel.conf file.

If you use Kerberos in this environment then you have to map the localhost address to the fully qualified domain name of your IBM i5/OS. Otherwise the Kerberos tickets are not created for the correct principal. Therefore, all non-SSL services are not reachable with this DNS name.

2.7 Printing service

You cannot use any printing service. iSeries Access for Linux does not provide 5250 printer emulation. However, it is possible to print from an iSeries to a Linux printer. Usually the printing system on a Linux system supports to send and receive Print jobs via Line Printer Daemon (LPD), Line Print Requestor (LPR), and/or Internet Printing Protocol (IPP). Review your Linux documentation about enabling your local Linux printer for remote network access.

IBM i5/OS is able to print to a Linux client using a remote out queue with LPR/LPD. This configuration is in detail explained in the IBM Redbook titled *IBM AS/400 Printing V*, SG24-2160.

The usage of the Internet Printing Protocol is described in the IBM Redbook titled *V5 TCP/IP Applications on the IBM eServer iSeries Server*, SG24-6321.

How to use an iSeries Printer from a Linux client with Samba is discussed in 5.4, “Connect to an iSeries NetServer printer share” on page 75.

3

Connect IBM i5/OS with ODBC

The iSeries Access Open Database Connectivity (ODBC) Driver for Linux allows you to access the DB2 Universal Database (UDB) for iSeries from any Linux application written to the ODBC application programming interfaces (APIs). It is an ODBC 3.5 American National Standards Institute (ANSI) driver with the ability to store and process Unicode data. Similar to the iSeries Access for Windows ODBC Driver, it uses a Transmission Control Protocol/Internet Protocol (TCP/IP) socket connection to the database host server to access data on the IBM i5/OS.

The ODBC driver is included in iSeries Access for Linux package. For information about the installation of the product, refer to 2.3, "Installing iSeries Access for Linux" on page 10.

Some examples of using this driver for Linux applications accessing data stored in IBM i5/OS side include:

- OpenOffice applications integrating DB2 UDB for iSeries database
- DataManagerII is a graphical user interface (GUI) application to view and manage data sources.
- Command line applications such as iSQL for interactive Structured Query Language (SQL) requests
- **`cwbrunsql`** for running batch SQL scripts
- PHP Hypertext Preprocessor (PHP) applications serving various Web pages
- Simple ODBC application written in C or Perl

3.1 ODBC configuration

The installation of ODBC driver as a part of iSeries Access for Linux is discussed in Chapter 2, “iSeries Access for Linux” on page 5. Do not use the outdated *iSeries ODBC driver for Linux* Red Hat Package Manager (RPM). The superseding version of the driver is included in iSeries Access for Linux.

There are two options for configuring the iSeries Access ODBC Driver, using a graphical tool such as ODBCConfig or editing the odbc.ini configuration file. The GUI tool ODBCConfig is modeled closely on Microsoft ODBC administrator and is the best choice for those who work with ODBC in a Microsoft environment.

You must also choose the type of data source name (DSN). DSNs can be either a user DSN or a system DSN. User DSNs are stored in a ~/.odbc.ini file in the home directory of the user who created the DSN, for example, /home/itsouser/.odbc.ini. They can only be accessed by the user who created them. System DSNs may only be created by root and are stored in the file such as /etc/unixODBC/odbc.ini and are accessible by all users on the system. The file location differs on several Linux distributions. Use the following command to get information about the current unixODBC configuration files on your system:

```
linux:~> odbcinst -j
unixODBC 2.2.11
DRIVERS............: /etc/unixODBC/odbcinst.ini
SYSTEM DATA SOURCES: /etc/unixODBC/odbc.ini
USER DATA SOURCES..: /root/.odbc.ini
```

3.1.1 ODBCConfig: GUI method of ODBC configuration

A data source can be configured using the ODBC data source GUI. This GUI contains fields to set required and frequently used options. The following instructions describe how to create or configure an iSeries Access ODBC Driver - DSN.

Note: ODBCConfig, or whatever different name your Linux distribution might use, is not required but if you want to use it, you may require to install it. In case of Novell SUSE distributions the ODBCConfig tool is included in *unixODBC-gui-qt*. Red Hat packaged ODBCConfig in *unixODBC-kde*. Your distribution may have a different package name or similar tools like gODBCConfig.

Perform the following steps:

1. On a graphical session, open the Data Source Administrator with the command **`ODBCConfig`** in the Linux terminal or use **start menu** → **Run Command**:

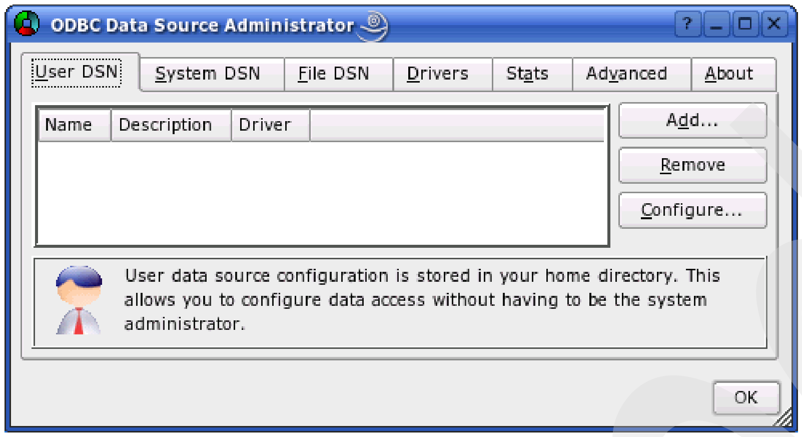
   ```
   linux:~> ODBCConfig
   ```

2. Decide what type of data source you want to use.

   User DSN are only accessible by the user who created them. System DSN are accessible by any user on the machine.

   The System DSN is configured by *root* user.

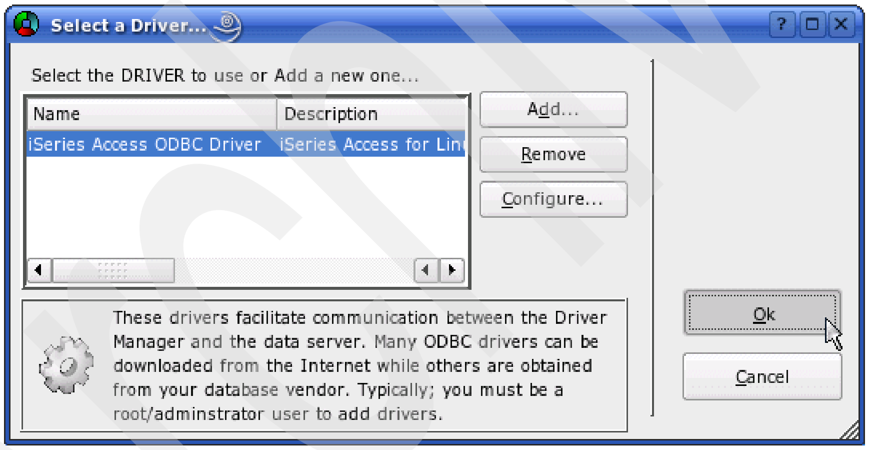
In the example used, click **User DSN** to create a new User DSN as illustrated in Figure 3-1.



User data source configuration is stored in your home directory. This allows you to configure data access without having to be the system administrator.

Figure 3-1 ODBC Data Source Administrator

3. In Figure 3-1, click **Add** to create a new data source or click **Configure** if you want to configure a data source that already exists.
4. Select an ODBC driver. Click **iSeries Access ODBC Driver** and click the **OK** button as illustrated in Figure 3-2.



Select the DRIVER to use or Add a new one...

Name	Description
iSeries Access ODBC Driver	iSeries Access for Lin

These drivers facilitate communication between the Driver Manager and the data server. Many ODBC drivers can be downloaded from the Internet while others are obtained from your database vendor. Typically; you must be a root/adminstrator user to add drivers.

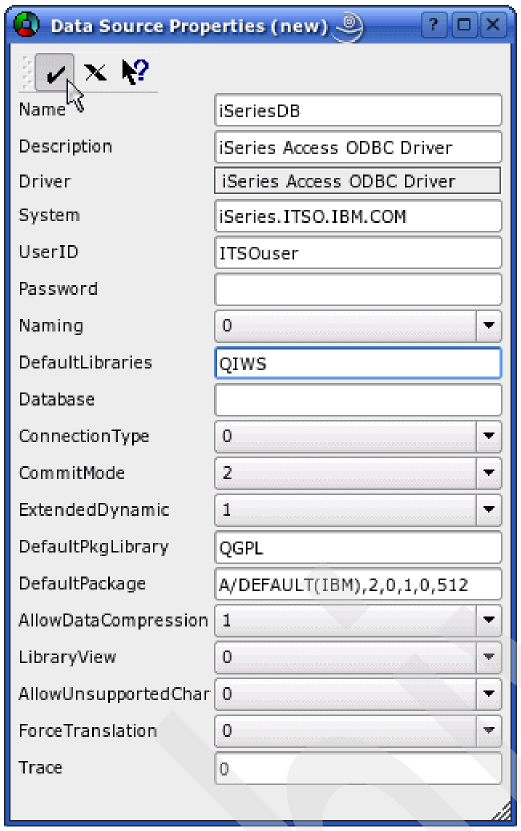
Figure 3-2 Add DSN

5. Configure your DSN properties. All the default values must work fine.

   Fill in the data source name in the Name field and your system name in the System field. All other fields are optional.

   For test purposes you can change the Default library to QIWS. QIWS provides the sample file QCUSTCDT.

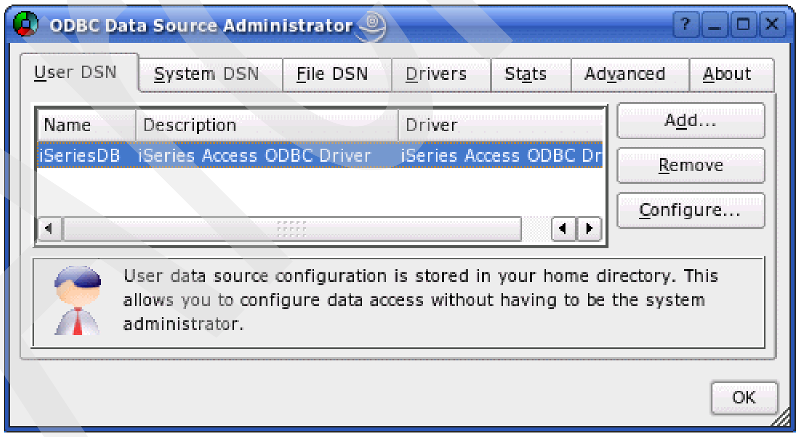
A DSN example is shown in Figure 3-3.


Data Source Properties (new)

Name	iSeriesDB
Description	iSeries Access ODBC Driver
Driver	iSeries Access ODBC Driver
System	iSeries.ITSO.IBM.COM
UserID	ITSOuser
Password	
Naming	0
DefaultLibraries	QIWS
Database	
ConnectionType	0
CommitMode	2
ExtendedDynamic	1
DefaultPkgLibrary	QGPL
DefaultPackage	A/DEFAULT(IBM),2,0,1,0,512
AllowDataCompression	1
LibraryView	0
AllowUnsupportedChar	0
ForceTranslation	0
Trace	0

Figure 3-3 ODBC config

6. Then, the Data Source Administrator shows the new user DSN as illustrated in Figure 3-4. Click the **OK** Button to exit the window.


ODBC Data Source Administrator

Name	Description	Driver
iSeriesDB	iSeries Access ODBC Driver	iSeries Access ODBC Dr

User data source configuration is stored in your home directory. This allows you to configure data access without having to be the system administrator.

Figure 3-4 ODBC Data Source Administrator: new DSN added

3.1.2 Editing odbc.ini: Text mode of ODBC configuration

Some of the driver connection options are not available via the ODBCConfig GUI. If you have the requirement to use some of the advanced DSN options then you must edit an *odbc.ini* file. The following example describes the edit of a system DSN, but the same steps (except the file location) also applies to a user DSN.

See the full list of connection DSN options in Appendix C, “ODBC connection string parameters” on page 151 or at:

`http://www.ibm.com/servers/eserver/iseries/linux/odbc/guide/odbcproperties.html`

Perform the following steps to manually add connection options to the system wide odbc.ini file:

1. An empty file odbc.ini must already exist after installation of unixODBC. Using your text editor of choice, edit this file with the contents as shown in Example 3-1. Make sure that you have root authorities for editing the system DSN file. Use the **`odbcinst -j`** command to locate your system DSN.

Example 3-1 Editing odbc.ini file

```
[iSeriesDB4allUser]
Description              = iSeries Access ODBC Driver
Driver                   = iSeries Access ODBC Driver
System                   = iseries.itso.ibm.com
UserID                   =
Password                 =
Naming                   = 0
DefaultLibraries         = QGPL
Database                 =
ConnectionType           = 0
CommitMode               = 2
ExtendedDynamic          = 1
DefaultPkgLibrary        = QGPL
DefaultPackage           = A/DEFAULT(IBM),2,0,1,0,512
AllowDataCompression     = 1
LibraryView              = 0
AllowUnsupportedChar     = 0
ForceTranslation         = 0
Trace                    = 0
```

2. You can add rows to modify the default behavior of the driver at run time. For example, the default date format is 5 which is yyyy-mm-dd or *ISO. If you want to change this format to 1 which is mm/dd/yy or *MDY, add a new row after the last entry in the data source and enter the new connection option and its value in the syntax of **`keyword = value`**. The entry this book uses is **`DFT = 1`** as shown in Example 3-2.

Example 3-2 Changing ODBC connection options

```
[iSeriesDB4allUser]
Description              = iSeries Access ODBC Driver
Driver                   = iSeries Access ODBC Driver
System                   = iseries.itso.ibm.com
UserID                   =
Password                 =
Naming                   = 0
DefaultLibraries         = QIWS
```

```
Database               =
ConnectionType         = 0
CommitMode             = 2
ExtendedDynamic        = 1
DefaultPkgLibrary      = QGPL
DefaultPackage         = A/DEFAULT(IBM),2,0,1,0,512
AllowDataCompression   = 1
LibraryView            = 0
AllowUnsupportedChar   = 0
ForceTranslation       = 0
Trace                  = 0
DFT                    = 1
```

Note: It is easy to make a mistake of keying in the same entry twice. For example, Trace=0, and somewhere else in the same file, Trace=1. Having multiple entries in the same file can lead to unpredictable behaviors.

After manually editing the odbc.ini file, you can still use **`ODBCConfig`** to configure your data source. Older versions of the unixODBC driver manager version had a problem where manually added options were removed from the odbc.ini file when **`ODBCConfig`** was used to configure the data source. If you experience this problem, install a newer version of the unixODBC driver manager.

Options specified by the application in the connection string that an application uses, override any options specified in the odbc.ini file.

3.2 Tools and utilities for data access

There are some readily available tools and utilities using iSeries Access ODBC Driver to access data stored in DB2 UDB for iSeries. This section covers the following:

- Using iSQL

  This is a small sample application that comes with unixODBC and can be useful to test queries or extract data in batch mode to a delimited file. It can also be used to generate results wrapped in Hypertext Markup Language (HTML).

- Using cwbrunsql

  iSeries Access for Linux is shipped with a tool called **`cwbrunsql`** that also allows running SQL scripts in batch mode.

- DataManagerII

  This is a GUI application which allows the user to view and manage data sources. It comes with unixODBC driver manager package.

3.2.1 iSQL

iSQL is an unixODBC program that provides an SQL Interface in interactive and batch mode. This section demonstrates how to use iSQL.

Perform the following steps:

1. Connect to the database using the Data Source Name followed by your IBM i5/OS user name and password as shown in Example 3-3.

   Example 3-3 Connecting to DB2 UDB for iSeries using iSQL

   ```
   linux:~> isql iSeriesDB ITSOuser password
   +---------------------------------------+
   | Connected!                            |
   |                                       |
   | sql-statement                         |
   | help [tablename]                      |
   | quit                                  |
   |                                       |
   +---------------------------------------+
   SQL>
   ```

2. This brings you to an `SQL>` prompt where you can execute SQL commands. Execute the following query:

   ```
   select * from qiws.qcustcdt
   ```

   See Example 3-4.

Example 3-4 Executing SQL statements using iSQL

```
SQL> select * from qiws.qcustcdt
+---------+---------+-----+-------------+-------+------+-------+-------+------+--------+------+
| CUSNUM  | LSTNAM  | INIT| STREET      | CITY  | STATE| ZIPCOD| CDTLMT|CHGCOD|BALDUE  |CDTDUE|
+---------+---------+-----+-------------+-------+------+-------+-------+------+--------+------+
| 938472  | Henning | G K | 4859 Elm Ave| Dallas| TX   | 75217 | 5000  | 3    |37.00   | 0    |
| 839283  | Jones   | B D | 21B NW 135 St| Clay | NY   | 13041 | 400   | 1    |100.00  | 0    |
| 392859  | Vine    | S S | PO Box 79   | Broton| VT   | 5046  | 700   | 1    |439.00  | 0    |
| 938485  | Johnson | J A | 3 Alpine Way| Helen | GA   | 30545 | 9999  | 2    |3987.50 |33.50 |
| 397267  | Tyron   | W E | 13 Myrtle Dr| Hector| NY   | 14841 | 1000  | 1    |0       | 0    |
| 389572  | Stevens | K L | 208 Snow Pass| Denver| CO  | 80226 | 400   | 1    |58.75   |1.50  |
| 846283  | Alison  | J S | 787 Lake Dr | Isle  | MN   | 56342 | 5000  | 3    |10.00   | 0    |
| 475938  | Doe     | J W | 59 Archer Rd| Sutter| CA   | 95685 | 700   | 2    |250.00  |100.00|
| 693829  | Thomas  | A N | 3 Dove Circle| Casper| WY  | 82609 | 9999  | 2    |0       | 0    |
| 593029  | Williams| E D | 485 SE 2 Ave| Dallas| TX   | 75218 | 200   | 1    |25.00   | 0    |
| 192837  | Lee     | F L | 5963 Oak St | Hector| NY   | 14841 | 700   | 2    |489.50  |.50   |
| 583990  | Abraham | M T | 392 Mill St | Isle  | MN   | 56342 | 9999  | 3    |500.00  | 0    |
+---------+---------+-----+-------------+-------+------+-------+-------+------+--------+-----+
SQLRowCount returns -1
12 rows fetched
SQL> quit
linux:~>
```

3. Example 3-5 shows an example of **isql** being used in batch mode. Batch mode is turned on with parameter -b and the parameter -w enables HTML output. This time the query is coming from the file batch.sql and the result is written to address.html.

Example 3-5   iSQL- Running queries in batch mode

```
linux:~> echo “select cusnum, lstnam, city from qiws.qcustcdt order by lstnam”
> batch.sql

linux:~> cat batch.sql | isql iSeriesDB ITSOuser password -b -w > address.html
linux:~>
```

4. The results of this query are formatted into an HTML table, which is sent the file called address.html. Table 3-1 illustrates the resulting table.

Table 3-1   Web browser view of address.html

CUSNUM	LSTNAM	CITY
583990	Abraham	Isle
846283	Alison	Isle
475938	Doe	Sutter
938472	Henning	Dallas
938485	Johnson	Helen
839283	Jones	Clay
192837	Lee	Hector
389572	Stevens	Denver
693829	Thomas	Casper
397267	Tyron	Hector
392859	Vine	Broton
593029	Williams	Dallas

5. In some cases you might require a Comma Separated File (CSV) for further processing with non-ODBC complaint applications. The **isql** parameter -dx exports the data with a delimiter x into a file, the column names are added with -c to the top of the file. This example uses a comma as delimiter:

```
echo “select * from qiws.qcustcdt” |isql iSeriesDB ITSOuser password -b -d, -c
>cust.csv
```

Note: For further testing of more complex queries you can use a sample database which is created on IBM i5/OS running the following SQL statement in iSQL:

```
CALL QSYS.CREATE_SQL_SAMPLE('dbsample');
```

Here you can give any name of your choice instead of <*dbsample*>. This creates a collection with sample tables, which come with DB2 UDB for iSeries itself.

3.2.2 cwbrunsql

iSeries Access for Linux is shipped with a tool called **cwbrunsql** that also allows running SQL scripts in batch mode. An interactive command shell like in iSQL is not available. You can use **cwbrunsql** with an existing DSN, where **iSeriesDB** is a User DSN and **itsouser** an IBM i5/OS user profile:

```
cwbrunsql /DSN:iSeriesDB /USER:itsouser /PASSWORD:password /I:batch.sql
```

On the other hand, you can use **cwbrunsql** without a DSN but with the IBM i5/OS host name:

```
cwbrunsql /SYSTEM:iseries.itso.ibm.com /USER:itsouser /PASSWORD:password
/I:batch.sql
```

Specify the SQL batch file with the parameter /I:. We used the same batch file as shown in Example 3-5. The result also includes additional data such as execution time and the IBM i5/OS job information as shown in Example 3-6. Due to explicit error messages the tool is also helpful for problem determination.

Example 3-6 cwbrunsql example

```
IBM iSeries Access for Linux
Version 5  Release 4Level 0
Run SQL Script V1.0
(C) Copyright IBM Corporation and Others 1984, 2003.  All rights reserved.
U.S. Government Users Restricted Rights - Use, duplication or disclosure
  restricted by GSA ADP Schedule Contract with IBM Corp.
Licensed Materials - Property of IBM

*********************************************************************************
Connected to system: ISERIES.ITSO.IBM.COM
Job information: QZDASOINITQUSER     076973
*********************************************************************************

STATEMENT 1
---------------
select cusnum, lstnam, city from qiws.qcustcdt order by lstnam

"CUSNUM", "LSTNAM", "CITY"
583990, "Abraham ", "Isle  "
846283, "Alison  ", "Isle  "
475938, "Doe     ", "Sutter"
938472, "Henning ", "Dallas"
938485, "Johnson ", "Helen "
839283, "Jones   ", "Clay  "
192837, "Lee     ", "Hector"
389572, "Stevens ", "Denver"
693829, "Thomas  ", "Casper"
397267, "Tyron   ", "Hector"
392859, "Vine    ", "Broton"
593029, "Williams", "Dallas"

12 rows fetched
*** Approximate time to run this stmt: 36 milliseconds ***
```

3.2.3 DataManagerII

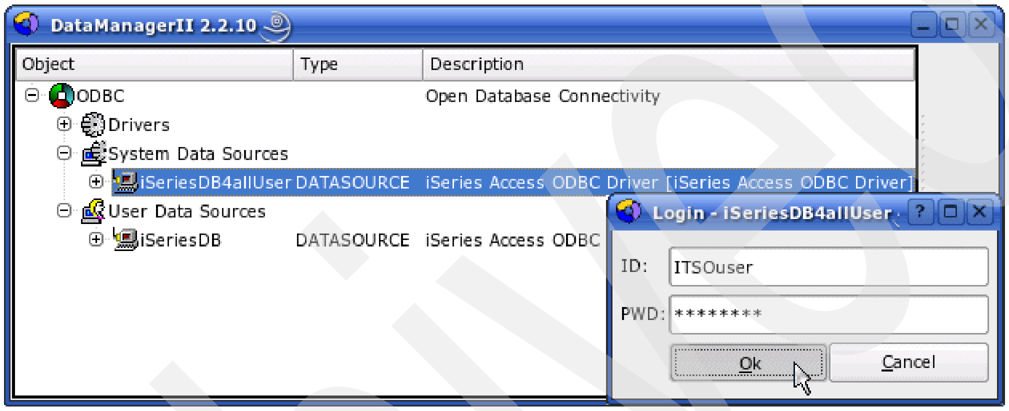
DataManagerII, which is included in unixODBC, is a graphical tool for exploring data sources. It allows to explore data sources in a similar manner as exploring file systems.

The following procedure provides the steps to work with unixODBC DataManagerII:

1. On a graphical session, open the DataManagerII in a Linux terminal or use **start menu** → **Run Command**:

   ```
   linux:~> DataManagerII
   ```

2. Open one of your existing Data Sources and log in to the IBM i5/OS as illustrated in Figure 3-5.

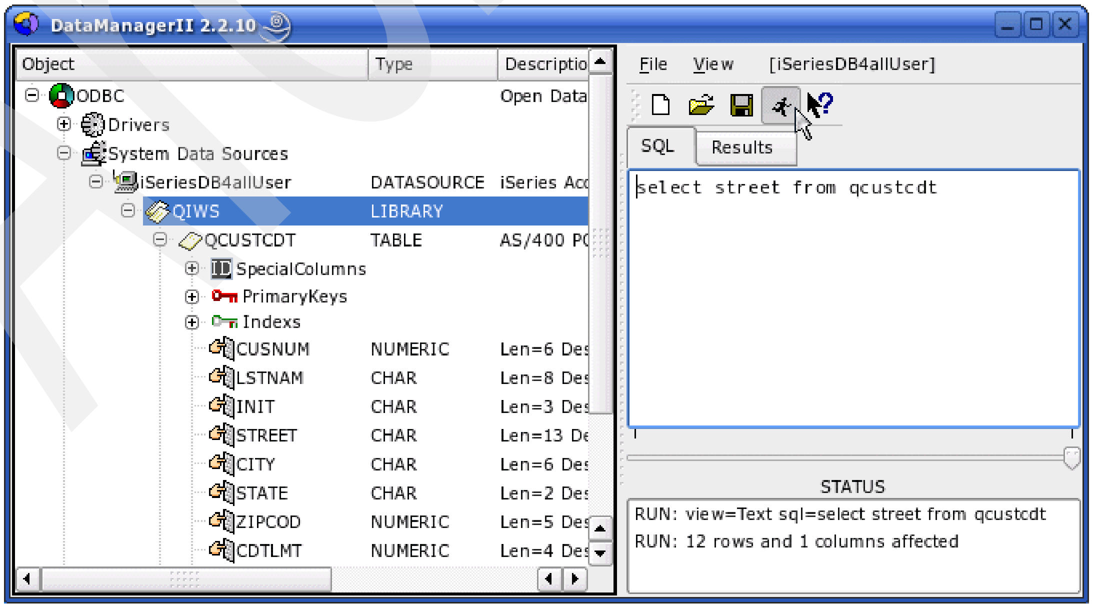


```
Object                          Type        Description
⊖ ODBC                                      Open Database Connectivity
  ⊕ Drivers
  ⊖ System Data Sources
    ⊕ iSeriesDB4allUser          DATASOURCE  iSeries Access ODBC Driver [iSeries Access ODBC Driver]
  ⊖ User Data Sources
    ⊕ iSeriesDB                  DATASOURCE  iSeries Access ODBC
```

Figure 3-5 Login to DataManagerII

3. The DataManagerII has two panes. The left pane displays a Tree view. In this view you can see elements like tables, restriction keys, indexes, and views.

   The right pane is the detail view. It shows any details that may be available for the selected item. In this view you can also execute SQL command to do queries. Write an SQL query as illustrated in Figure 3-6. Click the *runner man* to execute the query.



```
Object                          Type        Descriptio
⊖ ODBC                                      Open Data
  ⊕ Drivers
  ⊖ System Data Sources
    ⊖ iSeriesDB4allUser          DATASOURCE  iSeries Acc
      ⊖ QIWS                     LIBRARY
        ⊖ QCUSTCDT               TABLE       AS/400 P
          ⊕ SpecialColumns
          ⊕ PrimaryKeys
          ⊕ Indexs
            CUSNUM               NUMERIC     Len=6 Des
            LSTNAM               CHAR        Len=8 Des
            INIT                 CHAR        Len=3 Des
            STREET               CHAR        Len=13 De
            CITY                 CHAR        Len=6 Des
            STATE                CHAR        Len=2 Des
            ZIPCOD               NUMERIC     Len=5 Des
            CDTLMT               NUMERIC     Len=4 Des
```

```
select street from qcustcdt
```

```
STATUS
RUN: view=Text sql=select street from qcustcdt
RUN: 12 rows and 1 columns affected
```

Figure 3-6 DataManagerII: General view and SQL query execution

3.3 Accessing the iSeries DB2 UDB via OpenOffice.org

OpenOffice.org is an open source office productivity suite like Microsoft Office. The source code for the OpenOffice.org project was originally provided by the Sun™ StarOffice™ project. OpenOffice.org and Sun's StarOffice are functionally very similar and both include word processing, spreadsheet, presentation, and graphics applications. StarOffice has been successfully tested with the iSeries Access ODBC Driver for Linux but for the purpose of this demonstration, we use the open source OpenOffice.org suite. The spreadsheet application includes a great deal of the same functionality as MS Excel®, including the ability to query remote databases via ODBC. This section describes using OpenOffice.org spreadsheet application to pull data from the DB2 UDB for iSeries via ODBC.

3.3.1 Requirements

A recent version of OpenOffice.org 1.1.x or 2.x must be installed. If your Linux distribution does not include OpenOffice then you could download an installer for x86 or PPC Linux. For the installation instructions, program code, and other documentation, see:

http://www.openoffice.org/

3.3.2 Importing DB2 UDB for iSeries data in OpenOffice.org spreadsheets

OpenOffice.org has the capabilities to import data from an ODBC datasource. The following steps provide an example of the data import. We used OpenOffice.org 1.1.3.

1. On a graphical session, start OpenOffice with the command `ooffice` or use Start menu.
2. Select **File** → **New** → **Spreadsheet**.

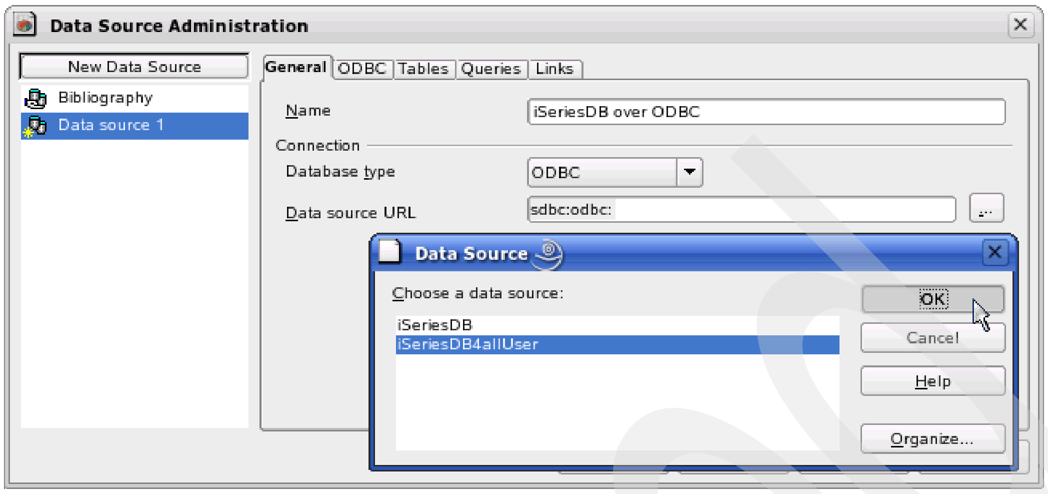
   Note: Starting with Version 2.0, OpenOffice includes its own database module. To configure an ODBC connected data source on Version 2.0, use the database wizard, **File** → **New** → **Database**.

   Again, we are using OpenOffice.org 1.1.3, but configuration steps must be similar to what you have with Version 2.0.

   This opens the spreadsheet application Calc.
3. Select **Tools** → **Data Sources**.

   The OpenOffice Data Source Administration window opens. This window helps you to configure an OpenOffice data source. This new data source is usable only by OpenOffice.

See Figure 3-7.


Data Source Administration

New Data Source

Bibliography
Data source 1

General | ODBC | Tables | Queries | Links

Name: iSeriesDB over ODBC

Connection
Database type: ODBC
Data source URL: sdbc:odbc:

Data Source

Choose a data source:
iSeriesDB
iSeriesDB4allUser

Figure 3-7 OpenOffice data source configuration

4. From the window similar to the one shown in Figure 3-7, perform the following steps:
   a. Click the **New Data Source** button in the upper-left corner.

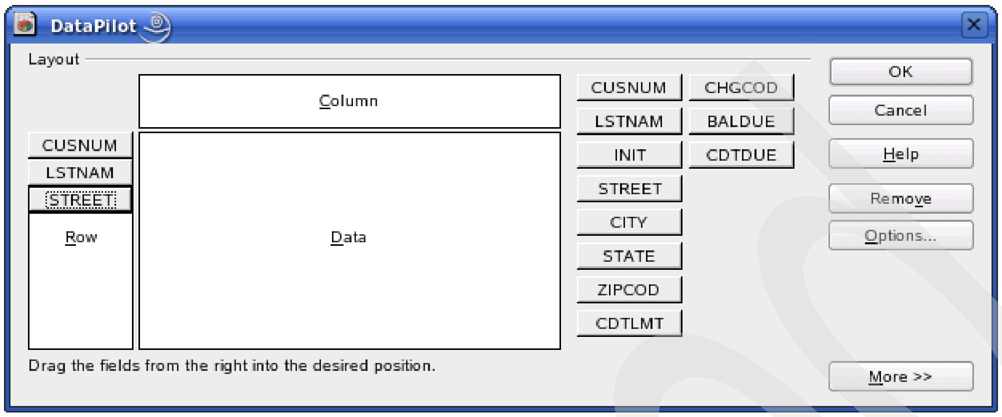
      Data source 1 is displayed in the list of OpenOffice data sources.
   b. Give the new data source a name, in the example this book uses, the name is *iSeriesDB over ODBC*. Again, this data source is only available for use by OpenOffice.
   c. Change the Database type to ODBC.
   d. In the Data source URL parameter there is a browse button represented with "...". Click it and you are presented with a list of DSN that you have configured in ODBCConfig.
   e. Select the DSN that was configured for access to your iSeries Database and click **OK**. In this example, we select the system DSN *iSeriesDB4allUser*.
   f. On the ODBC tab you must enter your i5/OS user profile and mark the check box **Password required**. You may view other configuration options available, but they are not necessary to change. Click **OK** to finish configuring the Data Source.
5. Now you are ready to bring data into the spreadsheet. Click **Data** → **DataPilot** → **Start**.
6. A Select Source window opens. Choose the option **Data source registered in OpenOffice.org** and click **OK**.
7. Change the Database to your Data Source.
8. For the Data Source parameter, select a table from the list.

   If **QCUSTCDT** is listed, select it.

> **Note:** The list of tables you see is your default collection. The default collection consists of the IBM i5/OS Library for your job (based on your iSeries user profile) and the default library you configured in the unixODBC DSN with ODBCConfig.

9. Click **OK**.

   Now you are presented with the DataPilot screen. This screen allows you to select your fields and place them into the desired positions. See Figure 3-8.



Drag the fields from the right into the desired position.

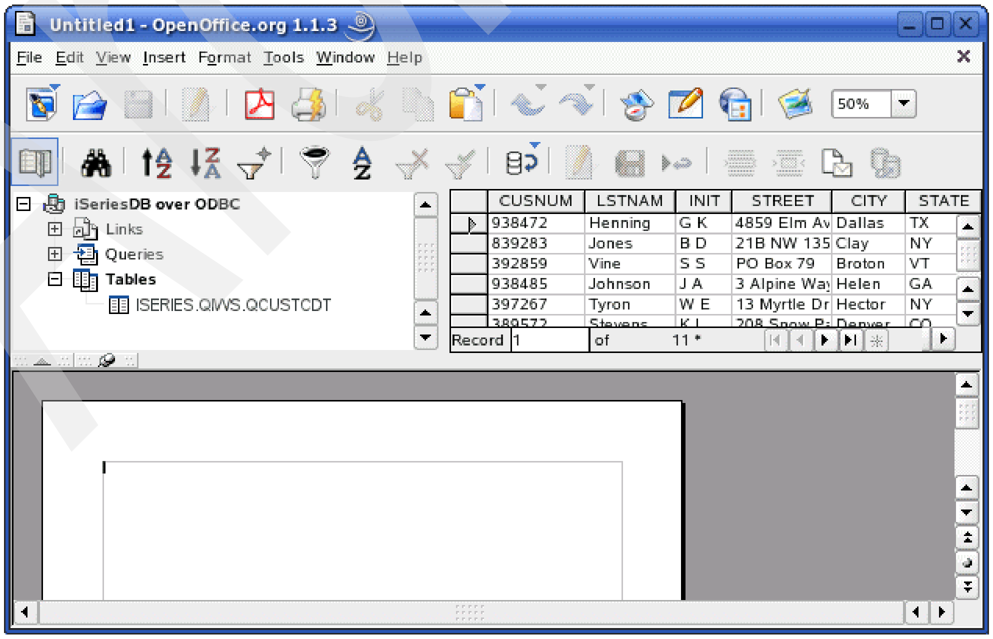
Figure 3-8 Data Pilot

   Left-click and drag the fields from the right into either the Row or Column areas. When finished, click **OK**.

   You must now have a spreadsheet with data stored in DB2 UDB for iSeries which you can save to any of the many OpenOffice.org data types.

3.3.3 Direct link to DB2 UDB for iSeries data

Another option for accessing DB2 UDB for iSeries is available in the menu **View** → **Data Sources** or press the F4 key in every OpenOffice.org application. In an additional window you get direct access to your Data Sources. That allows you to browse, sort, copy, or edit data without importing it to a spreadsheet. You can also use the Mail Merge function in menu **Tools** → **Mail Merge...** to generate personalized mails from you address database on IBM i5/OS. Figure 3-9 shows the Data Source View in Open Office Writer.


Untitled1 - OpenOffice.org 1.1.3
File Edit View Insert Format Tools Window Help
50%
iSeriesDB over ODBC
Links
Queries
Tables
ISERIES.QIWS.QCUSTCDT

CUSNUM	LSTNAM	INIT	STREET	CITY	STATE
938472	Henning	G K	4859 Elm Av	Dallas	TX
839283	Jones	B D	21B NW 135	Clay	NY
392859	Vine	S S	PO Box 79	Broton	VT
938485	Johnson	J A	3 Alpine Way	Helen	GA
397267	Tyron	W E	13 Myrtle Dr	Hector	NY
389572	Stevens	K L	208 Snow P:	Denver	CO

Record 1 of 11 *

Figure 3-9 OpenOffice Writer with Data Source View

Note: OpenOffice requires a primary key on a database file in order to edit the data. If there is no primary key the table is opened as read-only. You can add a primary key with iSeries Navigator or via SQL with the following command:

ALTER TABLE *tablename* **ADD PRIMARY KEY (***field***)**

3.4 Writing ODBC applications

There are many methods of writing your own Linux applications to access database on IBM i5/OS. This section provides an overview to ODBC programming and demonstrates the usage of unixODBC API with small sample code. The following scenarios are presented in this section:

- Client Server applications: Using C applications that can fetch an ordinary database file from a back-end i5/OS using the iSeries Access ODBC Driver for Linux.
- On three tier applications: These are common in Web pages. It uses a PHP module that provides built-in ODBC functions and access to DB2 UDB for iSeries.
- Using Perl to establish a database connection

3.4.1 Client application written in C

The ODBCexample.c is a working example of an application fetching an ordinary database file (QIWS/QCUSTCDT) from a backend IBM i5/OS using the iSeries Access ODBC Driver for Linux.

This example assumes that unixODBC and the iSeries Access ODBC Driver for Linux is installed. The target IBM i5/OS database must be registered as a data source in unixODBC. For compiling, you may also require unixODBC-devel package. The command to compile this file is:

```
linux:~> gcc -o ODBCexample -lodbc ODBCexample.c
```

The -o switch outputs the ODBCexample executable. The -lodbc switch links the ODBC libraries. The source code of ODBCexample is shown in Example 3-7.

Example 3-7 Source code of ODBCexample.c

```
/* Simple ODBC example for Linux

   Arguments: datasource (name of your System or User DSN)
              UserID
              password

   Example:   ODBCexample datasource UserID password

   Rochester, MN, July 2005 */

#include <sql.h>
#include <stdio.h>
#include <stdlib.h>

/* Simple query to DB table very likely to exist.  Takes first i5/OS "member" */
char querystring[] = "SELECT * FROM QIWS.QCUSTCDT";
```

```
void
die (char *reason, SQLRETURN code)
{
  if (code != 0)
    {
      printf ("Error code: %d\n", code);
    }
  printf ("Failure because: %s\n", reason);
  exit (1);
}

int
main (int argc, char *argv[])
{

#define MAX_CONNECTIONS 5
  char field1[255]; /*receives values from table row fetch */
  char field2[255];
  char field3[255];
  SQLINTEGER retfld1;
  /* For completelness. */
  SQLINTEGER retfld2; /* For this example, NULL for SQLBindCol */
  SQLINTEGER retfld3; /* would be enough    */
  SQLHENV henv;
  SQLHDBC hdbc[MAX_CONNECTIONS]; /* For multiple connections */
  SQLHSTMT statement;
  SQLRETURN sqe = 0;
  SQLRETURN res = 0;

  /* allocate environment */
  if (SQLAllocEnv (&henv))
    die ("Alloc Environment", 0);

  /* allocate connection */
  if ((sqe = SQLAllocConnect (henv, &hdbc[0])))
    die ("Allocate Connect", sqe);

  /* connect to data base */
  if ((sqe = SQLConnect (hdbc[0], argv[1], SQL_NTS,
          argv[2], SQL_NTS, argv[3], SQL_NTS)))
    die ("Connect", sqe);

  /* allocate statement(s) */
  if (SQLAllocStmt (hdbc[0], &statement))
    die ("Alloc stmt", 0);

  /* Do query */
  if ((sqe = SQLExecDirect (statement, querystring, SQL_NTS)))
    die ("exec direct", sqe);

  /* Bind coming SQLFetch operations to fields */
  if ((sqe = SQLBindCol (statement, 1, SQL_CHAR, field1, 255, &retfld1)))
    die ("Bind field 1", sqe);
  if ((sqe = SQLBindCol (statement, 2, SQL_CHAR, field2, 255, &retfld2)))
    die ("Bind field 2", sqe);
  if ((sqe = SQLBindCol (statement, 3, SQL_CHAR, field3, 255, &retfld3)))
    die ("Bind field 3", sqe);

  /* While loop to fetch all records */
  res = SQLFetch (statement);

  /* field1,2,3 take new values each time */
  while (SQL_SUCCEEDED (res))
    {       /* Create tab separated values */
      printf ("%s\t%s\t%s\n", field1, field2, field3);
      res = SQLFetch (statement); /* next record */
    }

  /* end actual processing */

  /* free statement(s) pretty completely  */
  if (SQLFreeStmt (statement, SQL_DROP))
    die ("Free Statement with Drop", 0);

  /* close connection */
  if ((sqe = SQLDisconnect (hdbc[0])))
    die ("Disconnect ", sqe);
  if ((sqe = SQLFreeConnect (hdbc[0])))
    die ("Connection closed", sqe);

  /* free environment */
  if (SQLFreeEnv (henv))
    die ("Free Environment", 0);

  printf ("Success!\n");

  return 0;
}
```

To run ODBC, use the following command as shown in Example 3-8, where iSeriesDB is the DSN, itsouser is the IBM i5/OS user profile and password is the related password of this user.

Example 3-8 Running testODBC program

```
linux:~> ./ODBCexample iSeriesDB itsouser password
938472  Henning         G K
839283  Jones           B D
392859  Vine            S S
938485  Johnson         J A
397267  Tyron           W E
389572  Stevens         K L
846283  Alison          J S
475938  Doe             J W
693829  Thomas          A N
593029  Williams        E D
192837  Lee             F L
583990  Abraham         M T
Success!
linux:~>
```

3.4.2 PHP for three-tier applications

PHP stands for PHP Hypertext Preprocessor. PHP is a widely-used general-purpose scripting language that is especially well-suited for Web development. It is commonly embedded into HTML.

PHP is mainly focused on server-side scripting, therefore you can do anything that any other Common Gateway Interface (CGI) programs can do, such as collecting form data, generating dynamic page content, or sending and receiving cookies.

The PHP module provides built-in ODBC functions if compiled with unixODBC support. It can access DB2 UDB for iSeries using the iSeries Access ODBC Driver for Linux.

PHP overview

Figure 3-10 describes the full process. When the Apache Web server receives a PHP request from a client Web browser, it forwards the request to the PHP module. The PHP module then finds the source code of the requested PHP page and interprets and executes that source code. If the PHP module finds a database access function in the source code, it invokes the iSeries Access ODBC Driver for Linux to access DB2 UDB for iSeries and executes the related SQL statement. The ODBC driver returns the data result set to the PHP module. The PHP module then embeds the resulting data with HTML tags and other content, and forwards it to the Apache server. Apache then sends the HTML page to the client's Web browser.

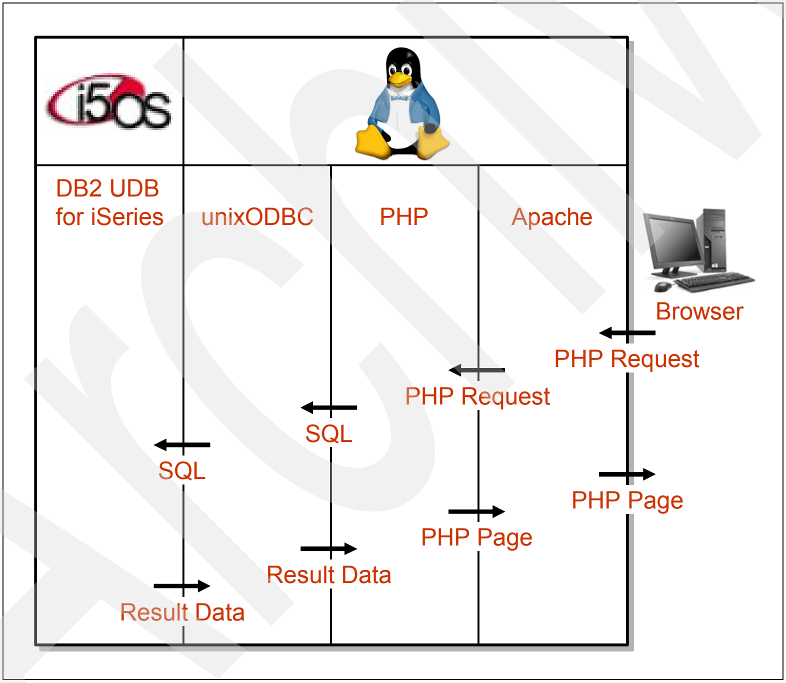


Figure 3-10 Apache and PHP with ODBC driver on a Linux system

Using PHP

The Novell SUSE Linux Enterprise Server and the Red Hat Enterprise Linux include all required software to run an Apache Web server with PHP scripting language using unixODBC for database access.

Perform the following steps:

1. Based on a default software installation of SLES 9 and SLES 10 you are required to add the following software packages to your server from your installation source:

   – apache2
   – apache2-prefork or apache2-worker
   – apache2-mod_php4
   – php4
   – php4_unixODBC for SLES9 and php5-unixODBC for SLES10

   On RHEL 3 and RHEL 4 you may add the package called php-odbc

2. Start the Web server with one of the following:

   – root@suse-linux:~> rcapache2 start
   – root@redhat-linux:~> /etc/init.d/httpd start

3. Check if the Apache server is working. Use your browser to open **`http://`*`linuxserver`*`/`**.

   As there is currently no configuration you must get an error message:

   ```
   Access forbidden: Error 403
   ```

   If you get a message similar to *destination unreachable*, then you might check if Apache is running and verify your firewall settings.

4. Test the PHP support by creating a file in the in the public_html directory of a user profile other than root:

   ```
   user@linux:~> echo “<? phpinfo() ?>” > ~/public_html/phpinfo.php
   ```

   Note: Make your PHP application available to Apache by placing PHP applications in a subfolder of the Apache documents directory.

   This example uses public_html folder in a users home directory. You can also place the files in a directory under the Apache documents directory of /www/htdocs. For example, place the PHP sample applications under /www/htdocs/phptest directory and open the Web page using:

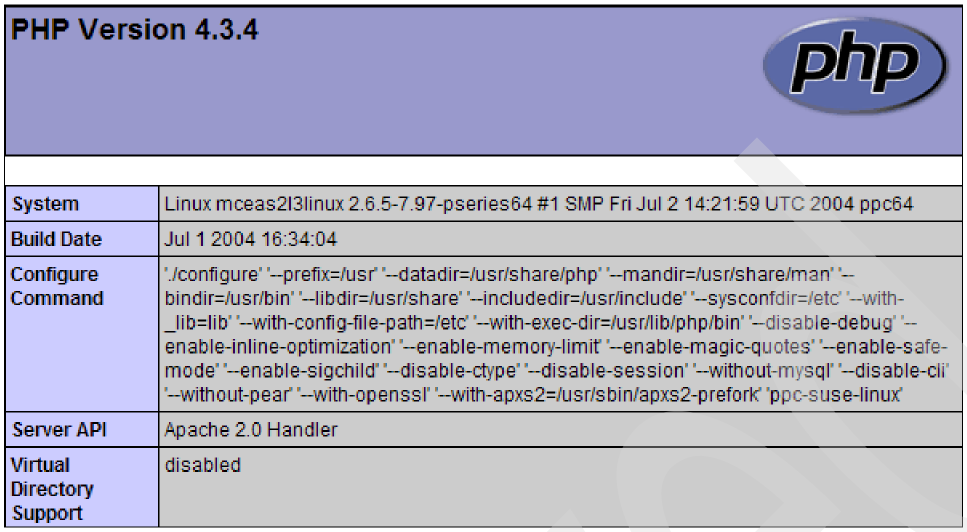
   ```
   http://linuxserver/phptest
   ```

5. Point your Web browser to the following address and replace *~itsouser* with your own Linux user name you used for creating the phpinfo file:

   ```
   http://linuxserver/~itsouser/phpinfo.php
   ```

   This PHP test side contains information about how PHP was configured and how ODBC support was compiled.

See Figure 3-11.



PHP Version 4.3.4

System	Linux mceas2l3linux 2.6.5-7.97-pseries64 #1 SMP Fri Jul 2 14:21:59 UTC 2004 ppc64
Build Date	Jul 1 2004 16:34:04
Configure Command	'./configure' '--prefix=/usr' '--datadir=/usr/share/php' '--mandir=/usr/share/man' '--bindir=/usr/bin' '--libdir=/usr/share' '--includedir=/usr/include' '--sysconfdir=/etc' '--with-_lib=lib' '--with-config-file-path=/etc' '--with-exec-dir=/usr/lib/php/bin' '--disable-debug' '--enable-inline-optimization' '--enable-memory-limit' '--enable-magic-quotes' '--enable-safe-mode' '--enable-sigchild' '--disable-ctype' '--disable-session' '--without-mysql' '--disable-cli' '--without-pear' '--with-openssl' '--with-apxs2=/usr/sbin/apxs2-prefork' 'ppc-suse-linux'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled

Figure 3-11 PHP test with phpinfo()

6. Now place the PHP script of Example 3-9 in the Apache document directory as the following:

   `/home/itsouser/public_html/ODBCexample.php`

   The PHP script is executed by Apache using a special user profile. Be aware of the different access permissions. Your User DSN is not accessible by the PHP script. Use a System DSN instead.

Example 3-9 PHP ODBC Example

```
<html>
<body>
<h1 align=center>Customer Name List</h1>
<table border=1 bgcolor='#FFFFFF' align=center>
<tr> <th>Customer Number</th>  <th>Last Name</th>  <th>Balance Due</th> </tr>
<?
$dsnname="iSeriesDB4allUser";                        //unixODBC System DSN name
$dbuser="itsouser";                                  //i5/OS username
$dbpwd="password";                                   //i5/OS password
$db=odbc_connect($dsnname,$dbuser,$dbpwd);      //connect to DB2 UDB for iSeries

if ($db==FALSE) echo "<p>Cannot connect!</p>";//test if db connection succeeded

$sql="select CUSNUM, LSTNAM, BALDUE from qiws.qcustcdt"; //query Statement
$result=odbc_exec($db,$sql);                          //execute Query statement
While (odbc_fetch_row($result)) {                         //fetch result set
   printf("<tr><td>%s</td><td>%s</td><td>%s</td></tr>",
   odbc_result($result,1),odbc_result($result,2),odbc_result($result,3));
}
odbc_close($db);                                          //close connection
?>
</table>
</body>
</html>
```

7. Use a Web browser to see the result of the php script:

   `http://linuxserver/~itsouser/ODBCexample.php`

   See Table 3-2.

Table 3-2 Web browser view of ODBCexample.php

Customer Name List		
Customer Number	**Last Name**	**Balance Due**
938472	Henning	37.00
839283	Jones	100.00
392859	Vine	439.00
938485	Johnson	3987.50
397267	Tyron	0
389572	Stevens	58.75
846283	Alison	10.00
475938	Doe	250.00
693829	Thomas	0
593029	Williams	25.00
192837	Lee	489.50
583990	Abraham	500.00

Sample PHP application for accessing the i5/OS database

IBM provides some PHP sample programs which you can download from the Web site at:

`http://www-1.ibm.com/servers/eserver/iseries/linux/odbc/guide/demoindex.html`

The application that comes with this download accesses DB2 UDB for iSeries database. An Apache Web server runs on a Linux server or on an IBM eServer i5 partition, and uses the iSeries Access ODBC Driver to retrieve data. This 3-tier application sample uses PHP and unixODBC to obtain data from IBM i5/OS.

This example extracts the sample PHP programs in /www/htdocs/IBM folder. You are free to run any of these sample programs in any method, but we provide one example of navigation.

1. Use the following URL in your Web browser:

   `http://<linux server name>/IBM`

2. Click **Start** from the main window to get the login screen. enter your iSeries name and the name of the sample database.
3. Select **employee**, then select any employee under that file.

3.4.3 Perl script example

Perl stands for Practical Extraction and Report Language. It was primarily developed for text manipulation and was rapidly grown to a powerful scripting language. One of the well-known usage is as a CGI language. Perl has an enormous number of expansion modules which extend the functionality for almost every purpose. This example script uses the perl-DBI module to establish a database connection. The source code is shown in Example 3-10.

Example 3-10 ODBCexample.perl

```
#!/usr/bin/perl
# ODBC Perl Example, KST & RF, Rochester, MN, 2005

use DBI;                # we're using perl DB Interface

$dbh = DBI->connect('dbi:ODBC:iSeriesDB', 'itsouser', 'password') or die "cannot
connect";

# connect to database
$sql = "SELECT * FROM QIWS.QCUSTCDT";           # our SQL query

$sth = $dbh->prepare($sql) or die "cannot prepare query"; # prepare SQL query or
die
$sth->execute or die "cannot execute";          # execute SQL query or die

do {
        my @row;
        while (@row = $sth->fetchrow_array()) {         # fetch each row in array
               for ($i=0;$i<$#row;$i++) { print $row[$i] }; # print each field in
a row
                print "\n";
        }
} while ($sth->{odbc_more_results});    # see if there's more records to show

$dbh->disconnect() or die "cannot disconnect";  # disconnect
```

Note: Depending on the installed versions of perl, perl-DBI, and unixODBC you might face a segmentation fault problem during the opening of a Database connection. As a circumvention you add the following shell variable:

```
linux:~> export LD_BIND_NOW=1
```

Execute the sample script on a Linux command line as shown in Example 3-11.

Example 3-11 Result of sample Perl script

```
linux:~> export LD_BIND_NOW=1
linux:~> perl ODBCexample.perl
938472Henning G K4859 Elm Ave DallasTX752175000337.00
839283Jones   B D21B NW 135 StClay  NY130414001100.00
392859Vine    S SPO Box 79    BrotonVT50467001439.00
938485Johnson J A3 Alpine Way Helen GA30545999923987.50
397267Tyron   W E13 Myrtle Dr HectorNY14841100010
389572Stevens K L208 Snow PassDenverCO80226400158.75
846283Alison  J S787 Lake Dr  Isle  MN563425000310.00
475938Doe     J W59 Archer Rd SutterCA956857002250.00
693829Thomas  A N3 Dove CircleCasperWY82609999920
593029WilliamsE D485 SE 2 Ave DallasTX75218200125.00
192837Lee     F L5963 Oak St  HectorNY148417002489.50
583990Abraham M T392 Mill St  Isle  MN5634299993500.00
linux:~>
```

4

Connect IBM i5/OS with JDBC

This chapter discusses how Linux can be integrated with IBM i5/OS using Java Database Connectivity (JDBC). The following topics are covered in this chapter:

- Introduction to JDBC
- Preparing to use JDBC
- Command line JDBC application
- Graphical JDBC application
- More Toolbox functions

4.1 Introduction to JDBC

The JDBC application programming interface (API) was designed to make coding common Structured Query Language (SQL) statements in Java simple and easy. At the same time, more complex SQL statements are possible. The typical steps to use the JDBC API are:

1. Load the JDBC driver.
2. Create the JDBC connection.
3. Create and execute your SQL statements.
4. Close the JDBC connection.

The JDBC API is part of the standard Java Development Kit (JDK™). Version 1.0 of JDBC runs on JDK 1.1.x and later. Version 2.0 of JDBC runs on JDK 1.2 and later.

4.2 Preparing to use JDBC

As you can see in 4.1, "Introduction to JDBC" on page 52, the first step in using JDBC is to load the JDBC driver. This leads to the question of which driver to use and where to get it. For DB2 Universal Database (UDB) for iSeries there are two different JDBC drivers that can be used. They are commonly known as the *native* driver and the *toolbox* driver. The native driver is shipped as part of licensed program Java Developer Kit 5722JV1. The toolbox driver is part of licensed program IBM Toolbox for Java 5722JC1. A simple comparison of the two drivers is that the native driver is faster but runs only on the IBM i5/OS Java virtual machine (JVM™), while the toolbox driver runs on any JVM. A good discussion about these drivers can be found at:

http://www-03.ibm.com/servers/eserver/iseries/toolbox/faqjdbc.html

We used the toolbox driver in the examples in this book.

4.2.1 Downloading the Java Toolbox

You can install the IBM Toolbox for Java on IBM i5/OS as product 5722JC1. After it is installed, the toolbox jar files are found in the Integrated File System (IFS) in directory /QIBM/ProdData/HTTP/Public/jt400/lib. The toolbox is also written as an open source project and the source code is available under the IBM Public license. Further information about the toolbox, a downloadable version and the documentation can be found at:

http://www.ibm.com/servers/eserver/iseries/toolbox/

To use the toolbox classes you require to download the appropriate jar file to your client and update the CLASSPATH to include the jar files. For JDBC and the simple command line JDBC application included here, only the base jt400.jar file is required.

Note: The toolbox is packaged in a ZIP file named jtopen_x_y.zip, where x_y is the version. The zip archive contains different jar files. In the example in this book, the zip is unpacked in the same directory as the java program. The jt400.jar is now located in the following path:

```
./jtopen/lib/jt400.jar
```

The Java classpath for the program must include the jt400.jar. Therefore, you must add the path to the CLASSPATH variable.

```
linux:~> export CLASSPATH=$CLASSPATH:./jtopen/lib/jt400.jar:.
```

4.2.2 Creating the sample database

All the examples in this chapter use a sample database. The creation of the sample database is done using the stored procedure CREATE_SQL_SAMPLE("*library*"), where *library* is the name of the sample library you wish to create. This stored procedure is new in version 5 release 1 and shipped with the operating system. To create the sample database perform the following steps:

1. STRSQL

   To get into interactive SQL
2. CALL QSYS/CREATE_SQL_SAMPLE('DBSAMPLE')

   To create the library (or collection in SQL terminology) with the sample tables

Creating the sample database takes a couple of minutes to complete. After it is completed you have 13 tables and numerous logical views. Sample data is also included in the tables. There are two sets of database tables in the sample schema. One set includes tables ORG, STAFF, and SALES. This set is a very simple schema with limited interaction between the tables and less complex data types within the tables. The remaining ten tables make up a more complicated database schema and makes use of more complex data types, such as BLOBS and CLOBS. These tables also interact with each other much more.

4.3 Command line JDBC application

To get started with JDBC programming on Linux, you can start with a command line approach where all input is from standard input and all the output is sent to standard out. This is a great way to test your JDBC and SQL code without the worries of the graphical interfaces. The sample command line JDBC application selects all the records out of the ORG table to get the list of departments. It then takes the department number and runs a prepared SQL statement to get the members of each department, their position, and the number of years with the company. This information is sent to standard out for each department, producing a departmental type listing.

Use **`javac`** to create the Java program, which is a program source that is shown in Example 4-1 on page 54:

```
linux:~> javac SimpleJDBC.java
```

This creates the executable Java code. To run the program, run the **`java`** command:

```
linux:~> java SimpleJDBC
```

You may have to add the current directory to your CLASSPATH to get the Java program to run. To do this issue the following command:

```
linux:~> export CLASSPATH=$CLASSPATH:./jtopen/lib/jt400.jar:.
```

Or use the -classpath parameter to include jtopen and the sample program to search for the path:

```
linux:~> java -classpath ./jtopen/lib/jt400.jar:. SimpleJDBC
```

This adds the current directory (identified by the '.') to the end of the current classpath. The output of this Java program is sent to standard out. If you want to send the data to a file, you can redirect standard out to a file use the following:

```
linux:~> java SimpleJDBC > jdbc.output
```

4.3.1 More about the program

When making the JDBC connection, we have imbedded the user profile and password that are going to be used during the connection. If a user profile and password are not provided then a prompt opens asking for profile and password.

Note: If a user and password are not configured, the Toolbox generates a graphical user interface (GUI) dialog box prompted for them. This means you require an X-Window environment to get this prompt. If you are working on the text mode screen, you see some font error messages on the command prompt, but the log in works without problems.

The program flow is to execute a query to identify all the departments from the ORG table. For each resulting department a prepared statement query is run to get all the members of the department from the STAFF table. The SQL statements can be simple as shown in Example 4-1, which shows the JDBC code using the ORG, STAFF, and SALES tables.

Example 4-1 Simple JDBC source code SimpleJDBC.java

```
import java.sql.*;

public class SimpleJDBC
{

    //! format a string to a set length
    private static String format(String s, int width)
    {
        String formattedString;

        if (s.length() < width) {
            StringBuffer buffer = new StringBuffer(s);
            for (int i = s.length(); i < width; ++i) {
                buffer.append(" ");
            }
            formattedString = buffer.toString();
        } else {
            formattedString = s.substring(0, width);
        }
        return formattedString;
    }

    public static void main(String argc[])
    {
        Connection connection = null;

        try {
            //! load the Toolbox JDBC driver
            Class.forName("com.ibm.as400.access.AS400JDBCDriver");

            //! get the connection to the iSeries database
            //! if no user or password is given a prompt will appear
            String system = "iSeriesname";
            String url = "jdbc:as400://" + system;
            String user = "Username";
            String password = "Password";
            connection = DriverManager.getConnection(url, user, password);
            DatabaseMetaData meta = connection.getMetaData();

            //! create and execute a simple SQL select statement
            Statement orgSelect = connection.createStatement();
           ResultSet orgResults = orgSelect.executeQuery("SELECT * FROM DBSAMPLE"
                                                               +
meta.getCatalogSeparator()
                                                               + "ORG");

            PreparedStatement deptSelect = connection.prepareStatement(
                                    "SELECT NAME, YEARS, JOB FROM DBSAMPLE"
                                    + meta.getCatalogSeparator()
                                    + "STAFF WHERE DEPT = ?");
            ResultSet deptResults = null;
            int deptNumb = 0;

            //! output info from each record
            while (orgResults.next()) {
                deptNumb = orgResults.getInt("DEPTNUMB");

                System.out.println("Department listing for "
                                  + orgResults.getString("DEPTNAME")
                                  + " in division "
                                  + orgResults.getString("DIVISION")
                                  + " located in "
                                  + orgResults.getString("LOCATION"));

                deptSelect.setInt(1, deptNumb);
                deptResults = deptSelect.executeQuery();

                while (deptResults.next()) {
                    System.out.println("  " +
format(deptResults.getString("NAME"), 15)
                                      + " " +
format(deptResults.getString("JOB"), 7)
                                      + " " + deptResults.getInt("YEARS"));
                }
            System.out.println();
            }
        }

        catch(Exception e) {
            System.out.println();
            System.out.println("Caught error " + e.getMessage());
            System.out.println(e);
        }

        finally {
            //! close the connection
            try {
                if (connection != null) {
                    connection.close();
                }
            }
            catch(SQLException e) {
            //! ignore since we are exiting anyway
            }
        }
    }
}
```

4.3.2 Sample output

Example 4-2 is a sample of the output generated by the simple JDBC example.

Example 4-2 Sample simple JDBC output

```
linux:~> java -classpath ./jtopen/lib/jt400.jar:. SimpleJDBC
Department listing for Head Office in division Corporate located in New York
  Molinare        Mgr     7
  Lu              Mgr     10
  Daniels         Mgr     5
  Jones           Mgr     12

Department listing for New England in division Eastern located in Boston
  Hanes           Mgr     10
  Rothman         Sales   7
  Ngan            Clerk   5
  Kermisch        Clerk   4

Department listing for Mid Atlantic in division Eastern located in Washington
  Sanders         Mgr     7
  Pernal          Sales   8
  James           Clerk   0
  Sneider         Clerk   8
```

4.4 Graphical JDBC application

JDBC can also be used in GUI applications to access data from IBM i5/OS. The following GUI program uses the same sample database as the command line JDBC application discussed in 4.3, “Command line JDBC application” on page 53. The difference here is that it presents the data to the user graphically for interaction with the user.

4.4.1 More about the program

The first thing the user is presented with is a drop down or combo box from which they can select the department they want information about. By default the first entry is selected and is blank. When this entry is selected the table and all data are cleared out. If a department is selected from the drop down box, the server is queried and the information about that department is filled. In the text field above the table is the division the department is in and the location of the department. In the table is the information about each employee, their position, and their number of years with the company.

Example 4-3 shows the graphical JDBC code.

Example 4-3 Graphical JDBC example GraphicalJDBC.java

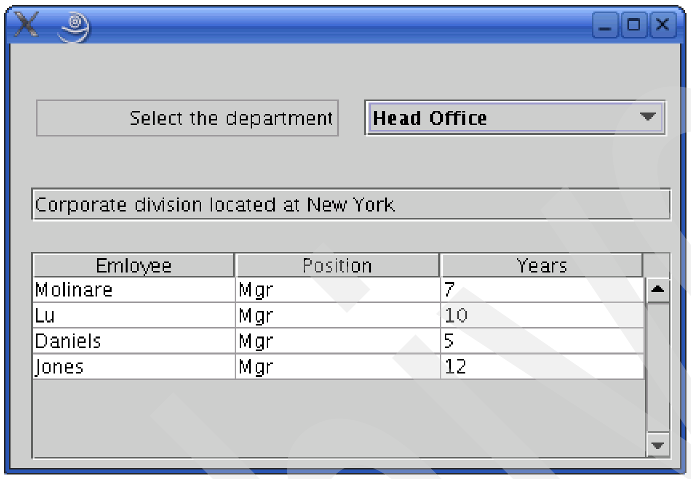
```
import javax.swing.*;
import javax.swing.table.*;
import java.util.*;
import java.sql.*;

public class GraphicalJDBC extends JDialog {
    private JPanel ivjJDialogContentPane = null;
    private JTextField ivjTFSelection = null;
    private JComboBox ivjCBDepartmentName = null;
    private JTextField ivjTFDepartmentLocation = null;
    private JScrollPane ivjSPTableScrollPane = null;
    private JTable ivjTBDepartmentTable = null;

    private Vector departments = new Vector();
    Connection connection = null;
    DatabaseMetaData meta = null;

    IvjEventHandler ivjEventHandler = new IvjEventHandler();

    class IvjEventHandler implements java.awt.event.ActionListener {
        public void actionPerformed(java.awt.event.ActionEvent e) {
            if (e.getSource() == GraphicalJDBC.this.getCBDepartmentName())
                connEtoC1(e);
        };
    };

    class DepartmentInfo extends Object{
        String name;
        String division;
        String location;
        int number;

        public DepartmentInfo() {
            super();
        }

        public String getName() {
            return name;
        }
        public String getDivision() {
            return division;
        }
        public String getLocation() {
            return location;
        }
        public int getNumber() {
            return number;
        }
        public void setName(String name) {
            this.name = name;
        }
        public void setDivision(String division) {
            this.division = division;
        }
        public void setLocation(String location) {
            this.location = location;
        }
        public void setNumber(int number) {
            this.number = number;
        }
    }

public GraphicalJDBC() {
    super();
    initialize();
}

private javax.swing.JPanel getJDialogContentPane() {
    if (ivjJDialogContentPane == null) {
        try {
            ivjJDialogContentPane = new javax.swing.JPanel();
            ivjJDialogContentPane.setName("JDialogContentPane");
            ivjJDialogContentPane.setLayout(null);
            getJDialogContentPane().add(getTFSelection(), 
getTFSelection().getName());
            getJDialogContentPane().add(getCBDepartmentName(), 
getCBDepartmentName().getName());
            getJDialogContentPane().add(getTFDepartmentLocation(), 
getTFDepartmentLocation().getName());
            getJDialogContentPane().add(getSPTableScrollPane(), 
getSPTableScrollPane().getName());
        } catch (java.lang.Throwable ivjExc) {
            handleException(ivjExc);
        }
    }
    return ivjJDialogContentPane;
}

private javax.swing.JTextField getTFSelection() {
    if (ivjTFSelection == null) {
        try {
            ivjTFSelection = new javax.swing.JTextField();
            ivjTFSelection.setName("Selection");
            ivjTFSelection.setText("Select the department");
            ivjTFSelection.setBounds(16, 35, 189, 23);
            ivjTFSelection.setEditable(false);
            ivjTFSelection.setHorizontalAlignment(javax.swing.JTextField.RIGHT);
        } catch (java.lang.Throwable ivjExc) {
            handleException(ivjExc);
        }
    }
    return ivjTFSelection;
}

private javax.swing.JComboBox getCBDepartmentName() {
    if (ivjCBDepartmentName == null) {
        try {
```

```
            ivjCBDepartmentName = new javax.swing.JComboBox();
            ivjCBDepartmentName.setName("CBDepartmentName");
            ivjCBDepartmentName.setBounds(221, 35, 189, 23);
        } catch (java.lang.Throwable ivjExc) {
            handleException(ivjExc);
        }
    }
    return ivjCBDepartmentName;
}

private javax.swing.JTextField getTFDepartmentLocation() {
    if (ivjTFDepartmentLocation == null) {
        try {
            ivjTFDepartmentLocation = new javax.swing.JTextField();
            ivjTFDepartmentLocation.setName("TFDepartmentLocation");
            ivjTFDepartmentLocation.setBackground(new
java.awt.Color(204,204,204));
            ivjTFDepartmentLocation.setBounds(13, 90, 400, 21);
        } catch (java.lang.Throwable ivjExc) {
            handleException(ivjExc);
        }
    }
    return ivjTFDepartmentLocation;
}

private javax.swing.JScrollPane getSPTableScrollPane() {
    if (ivjSPTableScrollPane == null) {
        try {
            ivjSPTableScrollPane = new javax.swing.JScrollPane();
            ivjSPTableScrollPane.setName("SPTableScrollPane");

ivjSPTableScrollPane.setVerticalScrollBarPolicy(javax.swing.JScrollPane.VERTICAL_S
CROLLBAR_ALWAYS);

ivjSPTableScrollPane.setHorizontalScrollBarPolicy(javax.swing.JScrollPane.HORIZONT
AL_SCROLLBAR_NEVER);
            ivjSPTableScrollPane.setBounds(13, 130, 400, 187);
            getSPTableScrollPane().setViewportView(getTBDepartmentTable());
        } catch (java.lang.Throwable ivjExc) {
            handleException(ivjExc);
        }
    }
    return ivjSPTableScrollPane;
}

private javax.swing.JTable getTBDepartmentTable() {
    if (ivjTBDepartmentTable == null) {
        try {
            ivjTBDepartmentTable = new javax.swing.JTable();
            ivjTBDepartmentTable.setName("TBDepartmentTable");

getSPTableScrollPane().setColumnHeaderView(ivjTBDepartmentTable.getTableHeader());
            getSPTableScrollPane().getViewport().setBackingStoreEnabled(true);
            ivjTBDepartmentTable.setModel(new
javax.swing.table.DefaultTableModel());
```

```
            ivjTBDepartmentTable.setBounds(0, 0, 200, 200);
            ivjTBDepartmentTable.getTableHeader().setReorderingAllowed(false);
            ivjTBDepartmentTable.getTableHeader().setResizingAllowed(false);

((DefaultTableModel)ivjTBDepartmentTable.getModel()).addColumn("Emloyee");

((DefaultTableModel)ivjTBDepartmentTable.getModel()).addColumn("Position");

((DefaultTableModel)ivjTBDepartmentTable.getModel()).addColumn("Years");
        } catch (java.lang.Throwable ivjExc) {
            handleException(ivjExc);
        }
    }
    return ivjTBDepartmentTable;
}

private void handleException(java.lang.Throwable exception) {

    /* Uncomment the following lines to print uncaught exceptions to stdout */
    // System.out.println("--------- UNCAUGHT EXCEPTION ---------");
    // exception.printStackTrace(System.out);
}

private void initConnections() throws java.lang.Exception {
    getCBDepartmentName().addActionListener(ivjEventHandler);
}

private void connEtoC1(java.awt.event.ActionEvent arg1) {
    try {
        this.cBDepartmentName_ActionPerformed(arg1);
    } catch (java.lang.Throwable ivjExc) {
        handleException(ivjExc);
    }
}

/**
 * Department selected.
 */
public void cBDepartmentName_ActionPerformed(java.awt.event.ActionEvent
actionEvent) {
    retrieveDepartmentInfo((String)getCBDepartmentName().getSelectedItem(),
getCBDepartmentName().getSelectedIndex());
    return;
}

private void initialize() {
    try {
        setName("GraphicalJDBC");
        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setSize(426, 366);
        setContentPane(getJDialogContentPane());
        initConnections();
    } catch (java.lang.Throwable ivjExc) {
        handleException(ivjExc);
    }
    try {
        // load the Toolbox JDBC driver
        Class.forName("com.ibm.as400.access.AS400JDBCDriver");

        // get the connection to the iSeries database
        // if no user or password is given a prompt will appear
        String system = "iSeries";
        String url = "jdbc:as400://" + system;
        String user = "ITSOUSER";
        String password = "PASSWORD";
        connection = DriverManager.getConnection(url, user, password);
        meta = connection.getMetaData();

        // retrieve the departments
        retrieveDepartments();
    }
    catch(Exception e) {
    }

}

public static void main(java.lang.String[] args) {
    try {
        GraphicalJDBC aGraphicalJDBC;
        aGraphicalJDBC = new GraphicalJDBC();
        aGraphicalJDBC.setModal(true);
        aGraphicalJDBC.addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent e) {
                System.exit(0);
            };
        });
        aGraphicalJDBC.show();
        java.awt.Insets insets = aGraphicalJDBC.getInsets();
        aGraphicalJDBC.setSize(aGraphicalJDBC.getWidth() + insets.left +
insets.right, aGraphicalJDBC.getHeight() + insets.top + insets.bottom);
        aGraphicalJDBC.setVisible(true);
    } catch (Throwable exception) {
        System.err.println("Exception occurred in main() of javax.swing.JDialog");
        exception.printStackTrace(System.out);
    }
}

private void retrieveDepartmentInfo(String department, int selectedIndex) {
    DefaultTableModel tm = (DefaultTableModel)getTBDepartmentTable().getModel();

    // remove all the rows
    for (int x = tm.getRowCount(); x > 0; --x)
        tm.removeRow(x - 1);

    // if this is the first entry the clear everything out, else fill in the table
    if (selectedIndex == 0) {
        getTFDepartmentLocation().setText("");
    } else {
        try {
            DepartmentInfo dept =
(DepartmentInfo)departments.elementAt(selectedIndex);
            getTFDepartmentLocation().setText(dept.getDivision()
                                        + " division located at "
                                        + dept.getLocation());

            PreparedStatement deptSelect = connection.prepareStatement(
                    "SELECT NAME, YEARS, JOB FROM ITSOSAMPLE"
                  + meta.getCatalogSeparator()
                  + "STAFF WHERE DEPT = ?");
            ResultSet deptResults = null;

            // add each record to the table
            deptSelect.setInt(1, dept.getNumber());
            deptResults = deptSelect.executeQuery();

            while (deptResults.next()) {
                Vector data = new Vector();
                data.add(deptResults.getString("NAME"));
                data.add(deptResults.getString("JOB"));
                data.add(new Integer(deptResults.getInt("YEARS")));
                tm.addRow(data);
            }
        }
        catch(Exception e) {
        }
    }
}

private void retrieveDepartments() {
    // create the default blank entry
    getCBDepartmentName().addItem("");
    DepartmentInfo defaultDepartment = new DepartmentInfo();
    defaultDepartment.setName("");
    defaultDepartment.setLocation("");
    defaultDepartment.setDivision("");
    defaultDepartment.setNumber(-1);
    departments.addElement(defaultDepartment);

    try {
        //! create and execute a simple SQL select statement
        Statement orgSelect = connection.createStatement();
        ResultSet orgResults = orgSelect.executeQuery("SELECT * FROM ITSOSAMPLE"
                                                  + meta.getCatalogSeparator()
                                                   + "ORG");

        while (orgResults.next()) {
            getCBDepartmentName().addItem(orgResults.getString("DEPTNAME"));
            DepartmentInfo dept = new DepartmentInfo();
            dept.setName(orgResults.getString("DEPTNAME"));
            dept.setLocation(orgResults.getString("LOCATION"));
            dept.setDivision(orgResults.getString("DIVISION"));
            dept.setNumber(orgResults.getInt("DEPTNUMB"));
            departments.addElement(dept);
        }
    }
    catch(Exception e) {
    }
}
}
```

4.4.2 Sample output

Figure 4-1 shows the output of the graphical JDBC code when run from the Linux platform using the ORG, STAFF, and SALES tables.



Corporate division located at New York

Emloyee	Position	Years
Molinare	Mgr	7
Lu	Mgr	10
Daniels	Mgr	5
Jones	Mgr	12

Figure 4-1 Graphical JDBC example

4.5 Toolbox for Java

The former examples used only a small part out of the IBM Toolbox for Java. The toolbox contains many other useful classes for accessing IBM i5/OS. You can use the toolbox to do things such as run i5/OS commands, create users and groups, call IBM i5/OS programs and APIs, and access other objects. Also included in the toolbox are graphical classes that make presenting IBM i5/OS information and objects easy and consistent. The toolbox is also written as an open source project and the source code is available under the IBM Public license. More information about the toolbox and the open source features can be found at:

`http://www.ibm.com/servers/eserver/iseries/toolbox/`

4.5.1 Running an i5/OS command

Example 4-4 is a Java program that shows just how easy it is to use the toolbox to run an i5/OS command. In this example the user is prompted to create a library name. The toolbox **`CommandCall`** object is then used to create the library. All messages generated by the command are then retrieved and shown to the user.

Note: If a user and password are not configured, the toolbox generates a graphical prompt for them. This means that you require an X-Window environment to get this prompt. You may see some font error messages on the command prompt, but the log in works without problems.

See Example 4-4.

Example 4-4 IBM Toolbox for Java command example

```
import java.io.*;
import java.util.*;
import com.ibm.as400.access.*;

public class ToolboxCommand
{
    public static void main(String[] argc) throws Exception
    {
    String systemName = "iseries.itso.ibm.com";
    String user = "itsouser";
    String password = "password";
    AS400 system = new AS400(systemName, user, password);

    BufferedReader inputStream = new BufferedReader(new
InputStreamReader(System.in), 1);
    String libraryName = null;
    System.out.println(" ");
    System.out.print("Library to create: ");
    libraryName = inputStream.readLine();

    System.out.println(" ");
    CommandCall command = new CommandCall(system);
    if (command.run("crtlib " + libraryName)) {
        System.out.println("Success");
    } else {
        System.out.println("Failure");
    }
    System.out.println(" ");

    AS400Message[] messageList = command.getMessageList();

    if (messageList.length > 0) {
        System.out.println("Messages returned are: ");
    }
```

4.5.2 Sample output

Example 4-5 shows the compiling, a successful library creation and an unsuccessful creation. In both cases the i5/OS command is run under the user profile associated with the connection (in this case ITSOUSER) and runs in a job under the QSERVER subsystem on IBM i5/OS.

Example 4-5 Toolbox command sample output

```
linux:~> javac -classpath ./jt400.jar ToolboxCommand.java

linux:~> java -classpath ./jt400.jar:. ToolboxCommand

Library to create: itsotest

Success
```

```
Messages returned are:
CPC2102: Library ITSOTEST created.
linux:~> java -classpath ./jt400.jar:. ToolboxCommand

Library to create: itsotest

Failure

Messages returned are:
CPF2111: Library ITSOTEST already exists.
linux:~>
```

5

Connect iSeries NetServer with Samba

This chapter discusses the connection of Samba client on Linux to iSeries NetServer and gives examples of how it may be used in an IBM i5/OS and Linux environment.

This chapter discusses the following:

- iSeries NetServer Samba client support
- NetServer versus Network File System (NFS)
- Walkthrough on using iSeries NetServer Samba client support
- Troubleshooting guide to iSeries NetServer Samba client support
- Printing on IBM i5/OS printers

5.1 NetServer introduction

There are some conventions that we use when discussing Samba access to iSeries NetServer. In Windows, a user *maps a network drive* to shared resources. But in Linux, a user *mounts* on an existing directory. For example, you might map a network drive to the *I:\* drive in Windows, while you would *mount* a remote iSeries NetServer share on /mnt/myshare in Linux. The ability to use any directory as a local mount point lifts the restriction on how many *mapped drives* you can have on a Linux system as you are not limited to single drive letters.

5.1.1 NetServer versus NFS

Both NetServer and NFS allow a Linux system to access the i5/OS file system. But even if the development of the NetServer or Server Message Block (SMB) and the NFS protocol is not restricted to a special operating system, you find NFS mostly in a UNIX® environment and SMB in a Windows environment.

The main differences between an NFS and NetServer connection is the user authentication. NetServer shares authenticate one user per session, only. This means that even if multiple users use the same mount of a NetServer share at the same time, all of them are using the i5/OS user profile which was specified during mounting. If every user requires to be authenticated on i5/OS NetServer with his or her own user profile, then every user requires own connection to NetServer. This behavior produces a large overhead in a multiuser environment.

NFS clients authenticate users on a server with their User ID number (UID). This means that different users can use one connection to i5/OS, but all users are working with there own user profiles. Administrator must synchronize the UID on the Client with the UID on the server.

In an NFS environment, the mount is made by an administrator and is often static/permanent. NetServer shares can also be mounted permanently by the root user, but there are tools for a temporary/dynamic file access that non administrator users can use. One of the tools is **`smbclient`**. See “Using smbclient to copy a file between iSeries NetServer and Linux” on page 72.

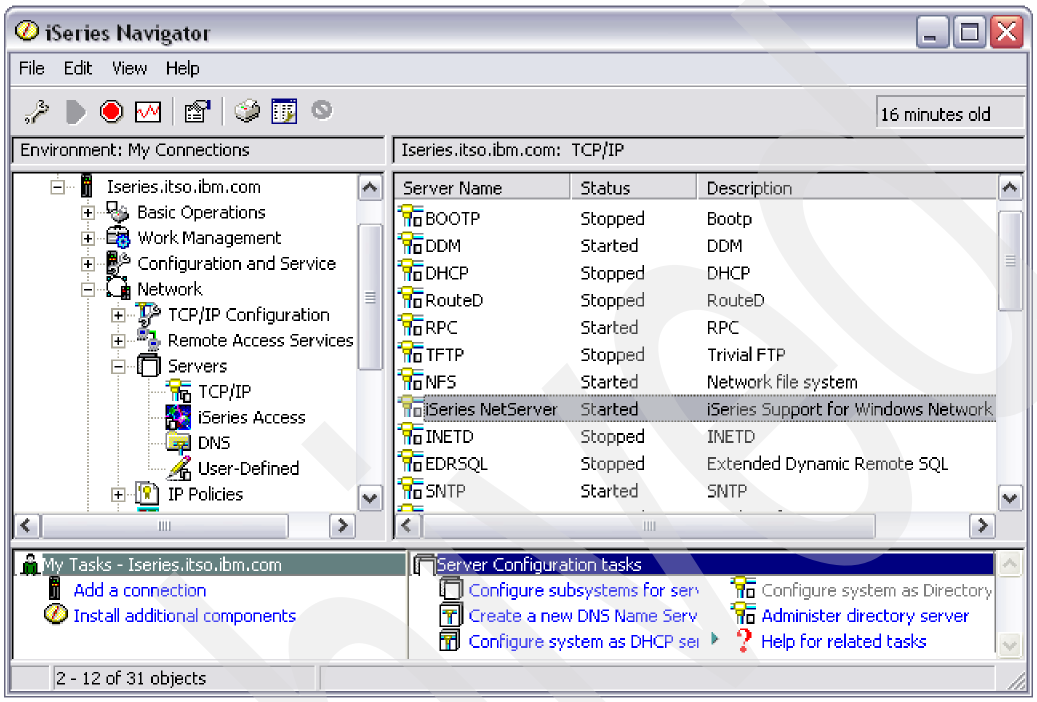
5.1.2 iSeries NetServer Samba client support

Starting with version 5 release 1 of OS/400, Linux Samba client connectivity is supported to iSeries NetServer. Thus, Linux PCs, or any Linux system for that matter, can mount network drives to iSeries NetServer shares and move files back and forth just as Windows PCs do.

Note: IBM i5/OS has an SMB client in the form of the QNTC file system. It was designed for accessing Windows NT® SMB servers and is not supported for use with Samba servers. Consider using NFS to access Linux servers from IBM i5/OS.

5.2 Requirements and installation

The iSeries NetServer is included in the IBM i5/OS operating system. The iSeries Navigator allows you to check the NetServer status, starting, stopping, and changing properties like the NetServer Network name. See Figure 5-1.



- Iseries.itso.ibm.com
  - Basic Operations
  - Work Management
  - Configuration and Service
  - Network
    - TCP/IP Configuration
    - Remote Access Services
    - Servers
      - TCP/IP
      - iSeries Access
      - DNS
      - User-Defined
    - IP Policies

Server Name	Status	Description
BOOTP	Stopped	Bootp
DDM	Started	DDM
DHCP	Stopped	DHCP
RouteD	Stopped	RouteD
RPC	Started	RPC
TFTP	Stopped	Trivial FTP
NFS	Started	Network file system
iSeries NetServer	Started	iSeries Support for Windows Network
INETD	Stopped	INETD
EDRSQL	Stopped	Extended Dynamic Remote SQL
SNTP	Started	SNTP

My Tasks - Iseries.itso.ibm.com
Add a connection
Install additional components
Server Configuration tasks
Configure subsystems for ser
Create a new DNS Name Serv
Configure system as DHCP se
Configure system as Directory
Administer directory server
Help for related tasks
2 - 12 of 31 objects

Figure 5-1 Check NetServer status with iSeries Navigator

Ensure that a current PTF level is installed. The recommended fixes can be found on iSeries NetServer home page under *Service Updates (PTFs)*:

http://www.ibm.com/servers/eserver/iseries/netserver/

The following minimum levels are required on a Linux system. Today's Linux distributions meet these criteria:

- Linux kernel version 2.4.4+
- Samba versions 2.0.7+ or 2.2+

Samba became a standard package included in mostly all Linux distributions. Check if Samba is already installed with the **rpm** command:

```
linux:~> rpm -qa |grep samba
samba-client-64bit-3.0.22-13.16
samba-64bit-3.0.22-13.16
samba-client-3.0.22-13.16
samba-3.0.22-13.16
```

If nothing is returned, you have to install at least the Samba client before mounting an iSeries NetServer share. The Samba file names may be different from the names listed previously depending on the Linux distribution you are using and the release level of Samba.

5.3 Using iSeries NetServer Samba support

This section provides an introduction to utilizing iSeries NetServer support for Linux Samba clients.

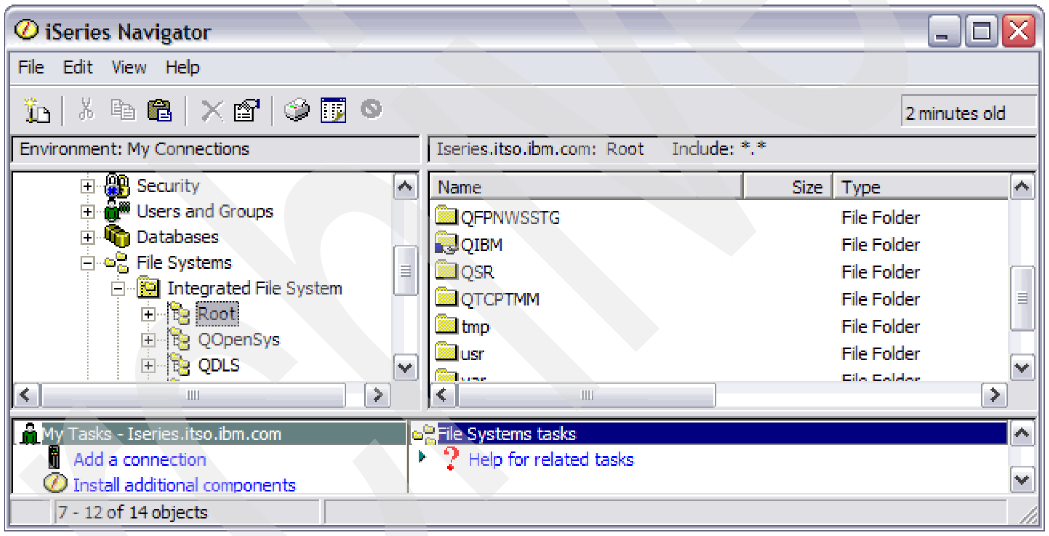
iSeries NetServer is a large topic and has its own IBM Redbook titled *The AS/400 NetServer Advantage*, SG24-5196. For details on iSeries NetServer, consult this manual. Up-to-date details on iSeries NetServer and also its support for Linux Samba clients are also available at:

http://www.ibm.com/servers/eserver/iseries/netserver/

5.3.1 iSeries NetServer configuration

Before a client can get access to IBM i5/OS file system, a share has to be defined on iSeries NetServer. This is accomplished via iSeries Navigator. Perform the following steps:

1. Open iSeries Navigator.
2. Expand sections **iSeriesname** → **File Systems** → **Integrated File System** → **Root**. See Figure 5-2.



Name	Size	Type
QFPNWSSTG		File Folder
QIBM		File Folder
QSR		File Folder
QTCPTMM		File Folder
tmp		File Folder
usr		File Folder
var		File Folder

Figure 5-2 iSeries Navigator

3. Create a new directory under the root or the /home directory. Right-click the directory and select **New Folder**.
4. Name the folder and click **OK**. The directory /root/test is the one we created in the example in this book.
5. This folder opens in the right-hand window.
6. Right-click the folder and select, **Sharing** → **New Share**. As the name of the share we use *Test*.
7. Change the access to **Read/Write**. You may also provide a description for the iSeries NetServer share. Also the share name can be different to the folder name. Click **OK**.

Note: Make a note of the small blue hand that is displayed beneath the folder. This indicates that the directory has been shared.

After these steps are completed, you are ready to access the share from a Samba client.

5.3.2 Linux Samba client configuration

There are many methods to get access to a NetServer share, on graphical user interface (GUI) and also on the console. As the GUI methods are very similar to Windows operating system, we discuss two methods of accessing SMB resources from a Linux console:

- smbmount
- smbclient

The **smbmount** command is the equivalent of mapping a drive in Windows, but instead of drive letters as with Windows, directories are mounted on a directory. These directories are called mount points. Any directory you have authorization to, can be mounted over.

Executing the **smbmount** command starts an smbmount daemon process that manages the mount point. While the smbmount daemon runs, the mount point is active.

In order to mount arbitrary file systems, you must either be signed in as the root user or the **smbmount** and **smbumount** commands must allow non-root user execution. For further information, see the man pages of mount, smbmount, and fstab. The type of file system mounted by smbmount is called smbfs.

The **smbclient** command is an interface similar to the File Transfer Protocol (FTP) interface that you can use to access remote SMB shares. It is generally used when a user only requires a short connection to a remote share or wants to see a list of the available shares.

Using smbmount to mount a network drive

Because you have just created a new folder and shared it with iSeries Navigator, you are now ready to mount that remote iSeries NetServer share. First, you require a local mount point on the Linux system. Most distributions have a directory of the root named /mnt. Generally, its purpose is to place mount directories within it. Therefore, a good place to create a mount point would be within /mnt:

1. Execute:

   ```
   linux:~> mkdir /mnt/smbtest
   ```

   It is not a requirement for the directory name to match the share name on IBM i5/OS.

2. Now, if you are signed on as root, you can mount the remote share:

   ```
   linux:~> smbmount //iseries.itso.ibm.com/test /mnt/smbtest -o username=itsouser
   Password:
   ```

   The user name parameter must be used to specify an i5/OS user name. You are prompted for a password. You may now treat the mounted directory as though it were a local file system.

3. To test, execute:

   ```
   linux:~> ls /mnt/smbtest
   ```

smbmount command details

The first argument of the **smbmount** command is the remote share being accessed in the form of //servername/sharename. The server name must be an Internet Protocol (IP) address or be resolvable to an IP address via Domain Name System (DNS), /etc/hosts, or WINS-style lookup such as /etc/samba/lmhosts. The second argument is the local mount point. The -o switch indicates optional parameter and precedes settings to be sent to the mount daemon. In this case, the only option you are sending is the i5/OS user profile you would like to sign in with. For additional smbmount command options use **man smbmount** or the following:

```
linux:~> smbmount --help
```

smbumount

The **smbumount** command is the equivalent to unmapping a network drive in Windows. It unmounts a remote share from the local file system. The only parameter for this command is the local mount point. Using the example given previously, the command is:

```
linux:~> smbumount /mnt/smbtest
```

This command fails if the actual connection no longer exists. In this case, the root user has to execute the more general **umount** command.

Using smbclient to copy a file between iSeries NetServer and Linux

As mentioned earlier, **smbclient** is an FTP-like utility that you can use to copy files to and from an SMB share on a remote system.

1. Verify what shares are available on the remote system:

   ```
   linux:~> smbclient -U username -L //systemname
   ```

   This provides you with a list of shares from the remote SMB server as shown in Example 5-1.

   Example 5-1 smbclient lists available share

   ```
   linux:~> smbclient -U itsouser -L //iseries.itso.ibm.com
   Password:
   Domain=[ITSO] OS=[OS/400 V5R3M0] Server=[iSeries Support for Windows Network
   Neighborhood]

           Sharename       Type      Comment
           ---------       ----      -------
           QIBM            Disk      IBM Product Directories
           ADMIN$          IPC       AS400 Resource
           QDIRSRV         Disk      OS/400 -- Directory Services
           IPC$            IPC       AS400 Resource
   Domain=[ITSO] OS=[OS/400 V5R3M0] Server=[iSeries Support for Windows Network
   Neighborhood]

           Server               Comment
           ---------            -------

           Workgroup            Master
           ---------            -------
   ```

2. Establish an smbclient connection to the iSeries NetServer:

   ```
   linux:~> smbclient //iSeriesname/sharename -U username
   ```

   You are prompted for a password. You are then presented with the smb prompt:

   ```
   smb: \>
   ```

3. Type **help**.

   You are presented with the list of available commands. Notice that they are similar to FTP.

4. Try **ls** is to get a file list of the share you have connected to.
5. Use **get** or **put** to copy a file from or to the NetServer.
6. To copy an entire directory, you must turn on directory recursion and disable prompting, which would otherwise prompt for every file you want to copy. The commands **recurse**

and **prompt** are toggle switches. The first usage of **recurse** turns the function on and the second turns it off. The command **prompt** acts vice versa.

To enable recursion and disable prompting, execute:

```
smb: \> recurse
smb: \> prompt
```

You may now use the **mget** and **mput** commands to copy directories to and from the iSeries NetServer share.

5.3.3 Troubleshooting SMB connectivity to iSeries NetServer

Following are common problem instances that can occur:

- `session request to <sysname> failed (Called name not present)`

  This error message occurs when the system name or IP address used to make the smbmount connection is different from the iSeries NetServer name. You can ignore it. The **smbmount** and **smbclient** commands only require the IP address of the remote system or the ability to resolve the system name to an IP address.

- `cannot mount on /<mountpoint>: Operation not permitted`

  This error message occurs because the user executing smbmount does not have authority to the mount point.

- `Input/output error`

  If you get an input/output error while trying to perform a function with a mount point or when you try to unmount (smbumount) the mount point, then this means that the SMB connection has probably been lost. This can be caused by the forced ending of the IBM i5/OS job QZLSFILE that was managing the connection, an event such as a restart of the iSeries NetServer server or a network issue, and so on. To get the Linux system to know the mount point has been lost, the root user requires to execute:

  ```
  linux:~> umount /<mountpoint>
  ```

- `umount: /<mountpoint>: device is busy` or `device is in use`

  A running process that you are unaware of may be using the file system you are trying to unmount. The first thing to check is whether your current working directory is inside the smbfs file system. The Linux command **pwd** shows your current working directory. To find users of a file system, Linux provides the very helpful command, **fuser**. In Example 5-2 the working directory is inside the smbfs, therefore you must prevent the unmount with your own bash shell.

  Example 5-2 fuser command

  ```
  linux:/mnt/smbtest> umount /mnt/smbtest/
  umount: /mnt/smbtest: device is busy
  linux:/mnt/smbtest> fuser -mauv /mnt/smbtest

                       USER        PID ACCESS COMMAND
  /mnt/smbtest         root      25582 ..c..  bash
  linux:/mnt/smbtest>
  ```

  If you have no control over the process which is still using the smbfs file system then you could use the **kill** command to end the process by providing the process identification (PID) as argument. For an immediate end of the process add the parameter -9 between command and PID:

  ```
  linux:~> kill 25582
  ```

Note: To kill all the PIDs accessing the file system, add the -k option to the **fuser** command (use with caution)!

- `tree connect failed: ERRSRV - ERRinvnetname (Invalid network name in tree connect.)`

  This error is received when an invalid remote share is specified on the **smbmount** command. Run:

  ```
  linux:~> smbclient -L //<systemame> -U <valid IBM i5/OS user profile>
  ```

  This generates a list of shares available on the SMB server specified.

- Overwriting files on a mounted netserver share fails with input/output error, even if permissions are set correctly. Copying the same files with smbclient works.

  ```
  linux:~> cp test.txt /mnt/smbtest/
  cp: cannot create regular file `/mnt/smbtest/test.txt': Input/output error
  ```

  The Linux log file /var/log/messages shows the following entries:

  ```
  kernel: smb_setup_bcc: Packet too large 4256 > 14
  kernel: smb_add_request: request [c000000039221080, mid=18] timed out!
  ```

  At the time of writing, the smbfs module of the Kernel 2.6.x shows a problem with NetServer shares if a file must be overwritten. If the file does not exist on the share then the **cp** or **mv** command works well. Linux with Kernel 2.4.x does not show this behavior.

  As a circumvention you can use the smbclient utility or delete the file on the share prior to copying the new one.

smbfs permissions

You may notice that when doing an extended listing of an SMB share with command **ls -l**, the permissions and owner of all objects are the same. The read, write, execute (rwx), and ownership properties are determined by the smbfs file system when smbmount is executed. It is possible to change the ownership and permission bits but the change is for every object in the mount point.

Note that this does not change anything on the remote share. Only the local Linux system enforces these permissions. The IBM i5/OS permissions always use object level security independent of the security mechanisms on the Linux system.

Important: The connection to the iSeries NetServer is established with the usage of one i5/OS user profile. This user profile is used on iSeries for every security validation for that particular NetServer mount, no matter which Linux user has access to the mount point.

IBM i5/OS troubleshooting

iSeries NetServer allocates a QZLSFILE job in the QSERVER subsystem for each mount point mounted from the Linux system to iSeries NetServer. To see what QZLSFILE jobs are allocated to particular users, you can use the IBM i5/OS command:

```
WRKOBJLCK OBJ(ITSOUSER) OBJTYPE(*USRPRF)
```

Example 5-3 shows a list of all jobs in use by user ITSOUSER. If you work with the QZLSFILE job, you can get information such as the IP address of the remote SMB client and other typical IBM i5/OS job information.

Example 5-3 Object Locks for a user

```
                             Work with Object Locks
                                                               System: ISERIES
 Object . . . . :   ITSOUSER             Type . . . . . :   *USRPRF
   Library  . . :     QSYS               ASP device . . :   *SYSBAS

 Type options, press Enter.
   4=End job   5=Work with job   8=Work with job locks

 Opt   Job          User         Lock      Status          Scope     Thread
       QZLSFILE     QUSER        *SHRRD     HELD           *JOB
                                 *SHRRD     HELD           *JOB
                                 *SHRRD     HELD           *JOB
                                 *SHRRD     HELD           *THREAD   00000004
```

Similar information is available in iSeries Navigator NetServer Configuration.

Open iSeries Navigator and expand sections **iSeries** → **Network** → **Servers** → **TCP/IP**. Double-click **iSeries NetServer**.

Here you can gather a great deal of information regarding iSeries NetServer shares and all the active connections to those shares.

5.4 Connect to an iSeries NetServer printer share

Linux has its own printer and spooling support through lpr, lpd, cups, and so on. However, it may be beneficial for Linux users to send files requiring printing to an IBM i5/OS printer using IBM i5/OS spooling and printer management capabilities as the print server.

5.4.1 How to configure a printer share

The following example shows you how to configure a printer share and enable users on a Linux system to share an IBM i5/OS configured printer. We assume that a printer is already configured on IBM i5/OS and you could use the printer's OUTQ for your purpose.

Important: Due to the limited printer driver support at present on Linux your IBM i5/OS printer must be able to print PCL or postscript data streams. If you intend to set up a printer share to a printer configured as IBM *IPDS™, IBM Advanced Function Printing™ (IBM AFP™) *YES then you have to use the IBM Infoprint® Server for iSeries 5722IP1 product to convert a PCL or Postscript data stream to AFP.

For further information see the IBM Redbook titled *IBM eServer iSeries Printing VI: Delivering the Output of e-business*, SG24-6250.

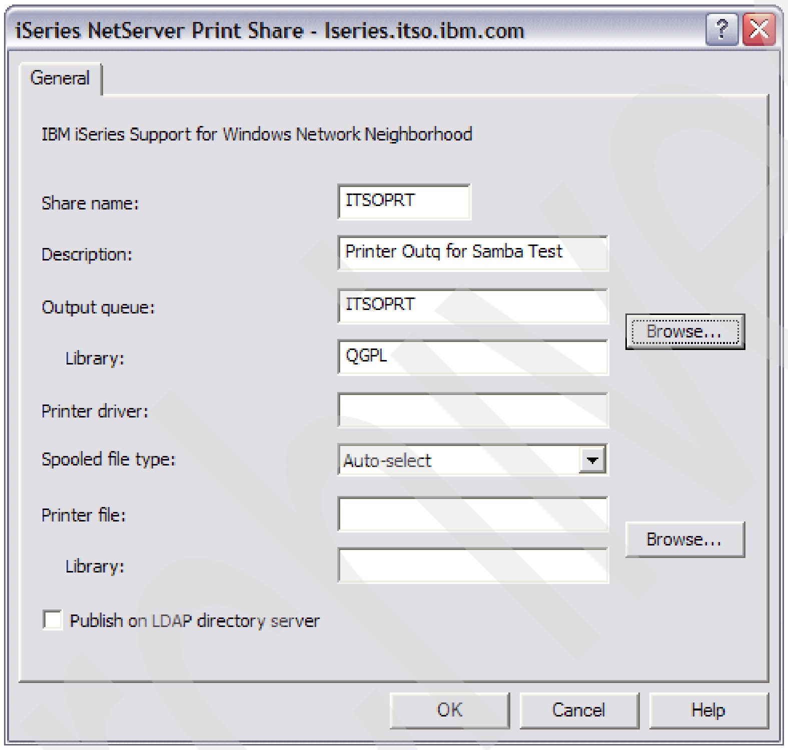
In this example, we use a Novell SUSE Linux distribution to connect to iSeries NetServer printer share. If you are using a different Linux distribution, then the steps and screen displays may be different.

Initially select a suitable OUTQ on IBM i5/OS or create one using the CRTOUTQ command. To set up a printer share on IBM i5/OS, you require the iSeries Navigator.

Create a printer share with iSeries Navigator

To share the i5/OS OUTQ, perform the following steps from the iSeries Navigator interface:

1. Select **Network** → **Servers** → **TCP/IP** and **iSeries NetServer**.
2. Expand **iSeries NetServer** and right-click **Shared Objects**. Select **New** and then **Printer**.
3. In the following screen illustrated in Figure 5-3 fill in the Share name and Output queue and library fields.


iSeries NetServer Print Share - Iseries.itso.ibm.com

IBM iSeries Support for Windows Network Neighborhood

Share name: ITSOPRT

Description: Printer Outq for Samba Test

Output queue: ITSOPRT

Library: QGPL

Printer driver:

Spooled file type: Auto-select

Printer file:

Library:

Publish on LDAP directory server

Figure 5-3 iSeries NetServer Print Share screen

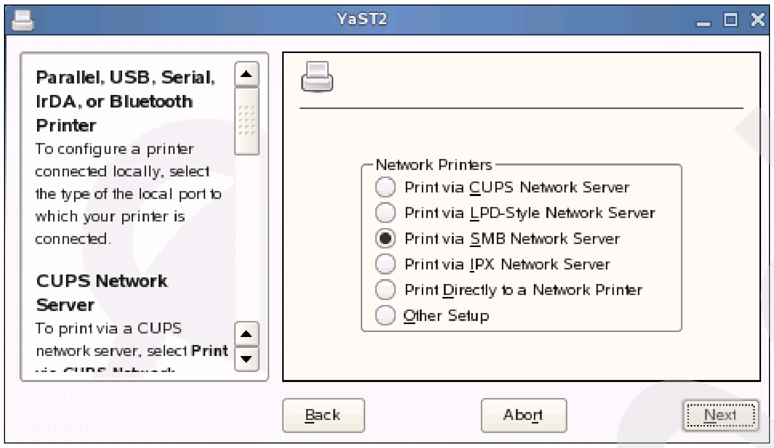
4. Select **OK**.
5. Select **Shared Objects** to check that this printer share opens in the list.

Connect to a shared printer from Linux

After an IBM i5/OS OUTQ has been shared, you can configure a Samba printer from the Linux client.

1. On a Novell SUSE Linux start the configuration utility YaST. Select the Hardware section on the left panel and click the Printer icon on the right panel.
2. YaST then initializes the printer configuration tool. When the printer setup window opens, select **Add**.
3. At the Printer Type window select **Network Printer**.

4. At the Connection for printer window, select **Print via SMB Network Server** and then select **Next**. See Figure 5-4.



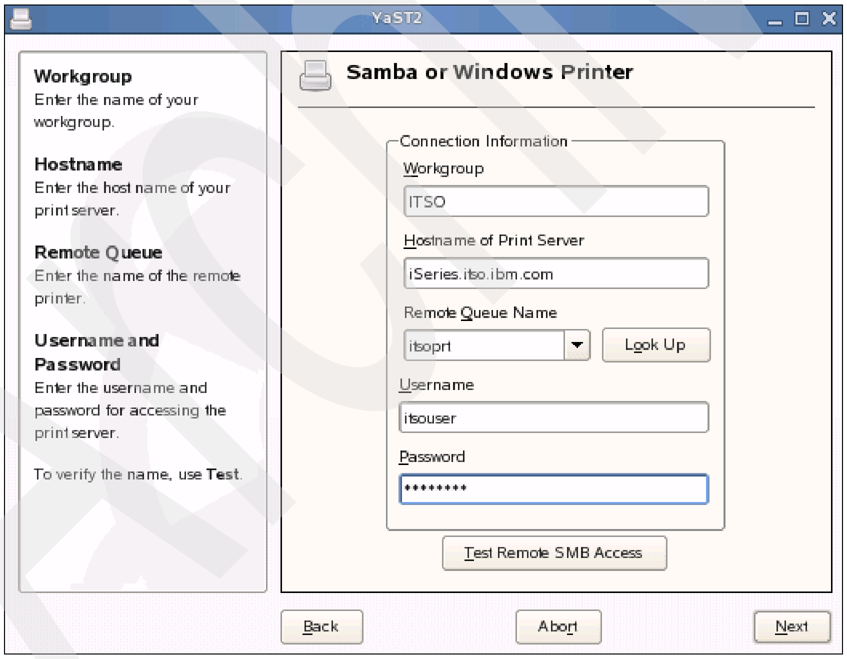
Parallel, USB, Serial, IrDA, or Bluetooth Printer
To configure a printer connected locally, select the type of the local port to which your printer is connected.

CUPS Network Server
To print via a CUPS network server, select **Print**

Network Printers
- ( ) Print via CUPS Network Server
- ( ) Print via LPD-Style Network Server
- (●) Print via SMB Network Server
- ( ) Print via IPX Network Server
- ( ) Print Directly to a Network Printer
- ( ) Other Setup

Figure 5-4 Connect to a printer share with YaST

5. The Samba/Windows printer screen opens. Enter the following information as shown in Figure 5-5 and select **Next**:
   – Host name: Enter the host name or IP address for the IBM i5/OS.
   – Remote queue: Define the printer share name.
   – User: Enter a valid IBM i5/OS user profile.
   – Password: Enter the password for this user.



Workgroup
Enter the name of your workgroup.

Hostname
Enter the host name of your print server.

Remote Queue
Enter the name of the remote printer.

Username and Password
Enter the username and password for accessing the print server.

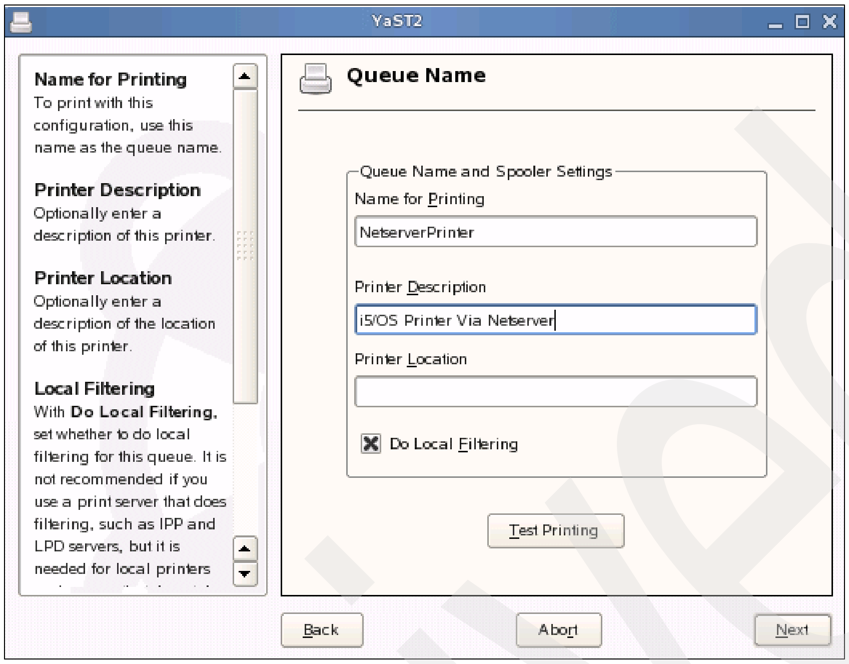
To verify the name, use **Test**.

Samba or Windows Printer

Connection Information
Workgroup: ITSO
Hostname of Print Server: iSeries.itso.ibm.com
Remote Queue Name: itsoprt
Username: itsouser
Password: ********

Figure 5-5 Samba/Windows printer window configuration in YaST

6. On the Queue name Window specify the printer name for Linux and check the option **Do Local Filtering**. See Figure 5-6.


YaST2
Name for Printing
To print with this configuration, use this name as the queue name.
Printer Description
Optionally enter a description of this printer.
Printer Location
Optionally enter a description of the location of this printer.
Local Filtering
With Do Local Filtering, set whether to do local filtering for this queue. It is not recommended if you use a print server that does filtering, such as IPP and LPD servers, but it is needed for local printers
Queue Name
Queue Name and Spooler Settings
Test Printing
Back
Abort
Next

Name for Printing: NetserverPrinter

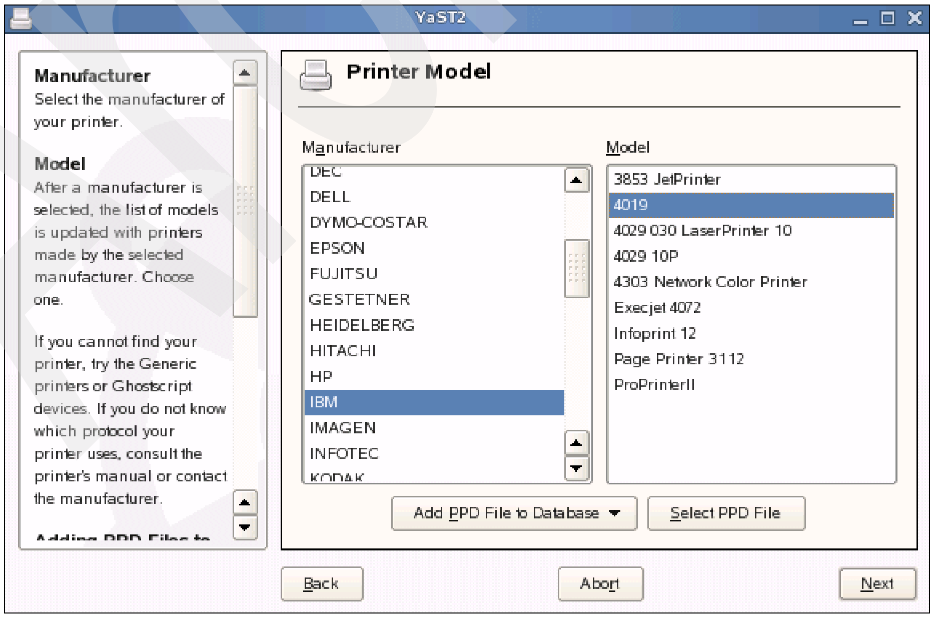
Printer Description: i5/OS Printer Via Netserver

Printer Location:

[✓ Do Local Filtering]

Figure 5-6 Specify the local name of the NetServer Printer

7. From the Manufacturer and model of the printer window select the type of your IBM i5/OS printer. See Figure 5-7. If your printer type is not available in the list but supports PCL or Postscript then select one of the generic printer types such as **PCL 5 Printer** or **Postscript level 2 Printer**.
8. The value selected in the Select model determines the exact American Standard Code for Information Interchange (ASCII) printer data stream into which a Linux file is converted and then sent to the i5/OS OUTQ.

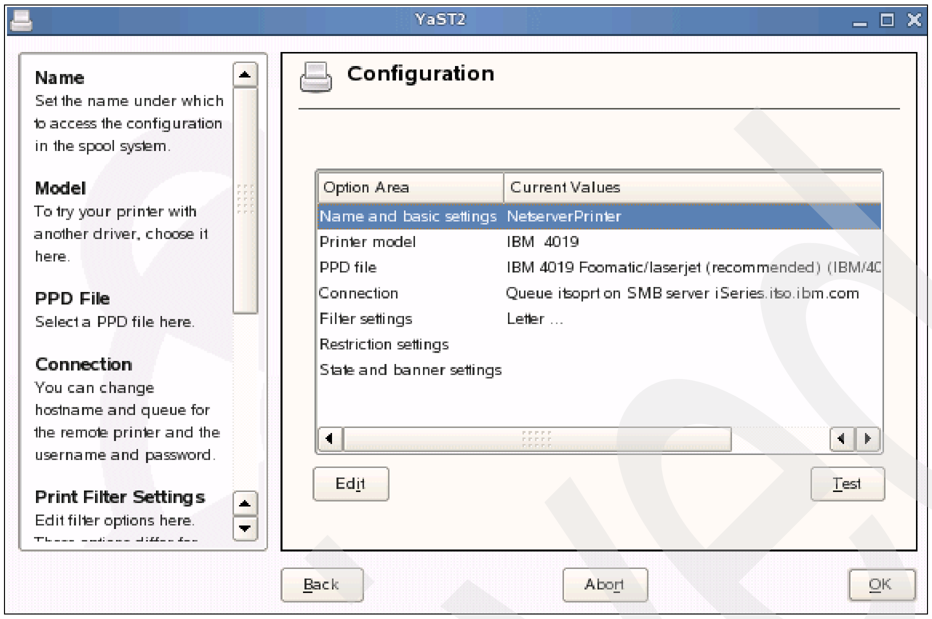

YaST2
Manufacturer
Select the manufacturer of your printer.
Model
After a manufacturer is selected, the list of models is updated with printers made by the selected manufacturer. Choose one.
If you cannot find your printer, try the Generic printers or Ghostscript devices. If you do not know which protocol your printer uses, consult the printer's manual or contact the manufacturer.
Printer Model
Add PPD File to Database
Select PPD File
Back
Abort
Next

Manufacturer: DEC, DELL, DYMO-COSTAR, EPSON, FUJITSU, GESTETNER, HEIDELBERG, HITACHI, HP, [✓ IBM], IMAGEN, INFOTEC, KODAK

Model: 3853 JetPrinter, [✓ 4019], 4029 030 LaserPrinter 10, 4029 10P, 4303 Network Color Printer, Execjet 4072, Infoprint 12, Page Printer 3112, ProPrinterII

Figure 5-7 Manufacturer and model of printer window

9. From the Printer settings window you must set your default settings such as paper size or tray and also the hardware settings such as duplex and orientation. Then click **OK**.


YaST2
Name
Set the name under which to access the configuration in the spool system.
Model
To try your printer with another driver, choose it here.
PPD File
Select a PPD file here.
Connection
You can change hostname and queue for the remote printer and the username and password.
Print Filter Settings
Edit filter options here.
Configuration
Edit
Test
Back
Abort
OK

Option Area	Current Values
Name and basic settings	NetserverPrinter
Printer model	IBM 4019
PPD file	IBM 4019 Foomatic/laserjet (recommended) (IBM/4C
Connection	Queue itsoprt on SMB server iSeries.itso.ibm.com
Filter settings	Letter ...
Restriction settings	
State and banner settings	

Figure 5-8 General Printer settings

10. From the final window, select **Finish**. This updates the CUPS configuration.

5.4.2 Points to keep in mind

There are some limitations and restrictions that you may require to consider when setting up a Samba printer share for many Linux users.

- During creation of an SMB printer you have to specify an IBM i5/OS user profile. This user is the owner of the spool files on IBM i5/OS regardless of which Linux user prints it. If you want to keep the owner of the spool file in connection to a Linux user then you have to create one printer per user on Linux.
- You can move spool files from one OUTQ to another, or you can start a different printer writer against the initial OUTQ. However, doing this ensures that the physical printer supports the same printer language as the printer the spool file was created for. The attributes of a spool file of type **USERASCII* cannot be changed after the spool file is placed on an IBM i5/OS OUTQ. For example, a *USERASCII SPLF built with the HP PCL5 data stream must print successfully on any PCL5 compliant printer. However, it would print incorrectly if sent to a printer supporting the IBM IPDS data stream.
- Some printer functions, such as correct drawer selection, staple or finisher support may not work with the generic drivers. However, there are Linux printer drivers for various IBM printers. See the following Web site for more details:

  `http://www.printers.ibm.com`

6

Connect IBM i5/OS with NFS

This chapter discusses the usage of the Network File System (NFS) in connection with IBM i5/OS. It gives examples about the configuration and how it may be used between IBM i5/OS and Linux.

This chapter discusses the following:

- Set up IBM i5/OS as NFS server
- Mount an IBM i5/OS Integrated File system (IFS) directory on a Linux system
- Security considerations

6.1 Introduction into the Network File System

The NFS is a file system which allows users to access files located on a remote computer via an Internet Protocol (IP) network.

The nature and structure of the Linux file system lends itself to being stored in different places. NFS is developed to allow machines to read and write files on remote systems as if they were on a local hard drive. This facilitates the seamless sharing of files across a network. Both Linux and IBM i5/OS support NFS.

The advantages of the NFS include:

- ► Exchanging data between IBM i5/OS and Linux
- ► Sharing the same files between multiple Linux clients and IBM i5/OS

If you have Linux partitions on your IBM eServer i5, then you can use NFS to share the same data across multiple Linux servers, thus saving disk space.

A comparison between NFS and NetServer is written in 5.1.1, “NetServer versus NFS” on page 68.

6.1.1 Running NFS server on i5/OS versus Linux

This section briefly compares the options of running NFS server either on i5/OS side or on Linux side.

Attention: Whether you run NFS server on i5/OS side or Linux side, stay aware of the security considerations regarding NFS. Keep in mind that the only security that the NFS protocol provides is based on the client IP address. The user profiles are identified by a user identification (UID) number which every Linux administrator can change at any time. No further password checks are performed.

For further discussion of security matters regarding NFS, refer to 6.4, “Advanced NFS configuration and security” on page 85.

Running NFS server on IBM i5/OS

IBM i5/OS as an NFS Server provides the complete advantages of your existing object security, backup concepts, and availability to Linux systems.

As the files are stored in the IFS of IBM i5/OS you can use your existing methods and tools for managing resources and administering the data. You can easily include the files in your IBM i5/OS backup without the requirement of additional backup devices on NFS client side.

Running NFS server on Linux

For a large amount of NFS traffic the Linux system can become the preferred choice to be an NFS server. On Linux systems you have much more influence to the runtime parameter of the operating system and the NFS server. Experienced Linux administrator have the opportunity to change a lot of performance related settings and adjust the NFS server specially to their kind of workload. This causes much more administration complexity, but may result in some performance improvements.

For more comprehensive overview and implementation guidelines about NFS and iSeries see the book titled *OS/400 Network File System Support*, SC41-5714.

6.2 Setting up NFS server on IBM i5/OS

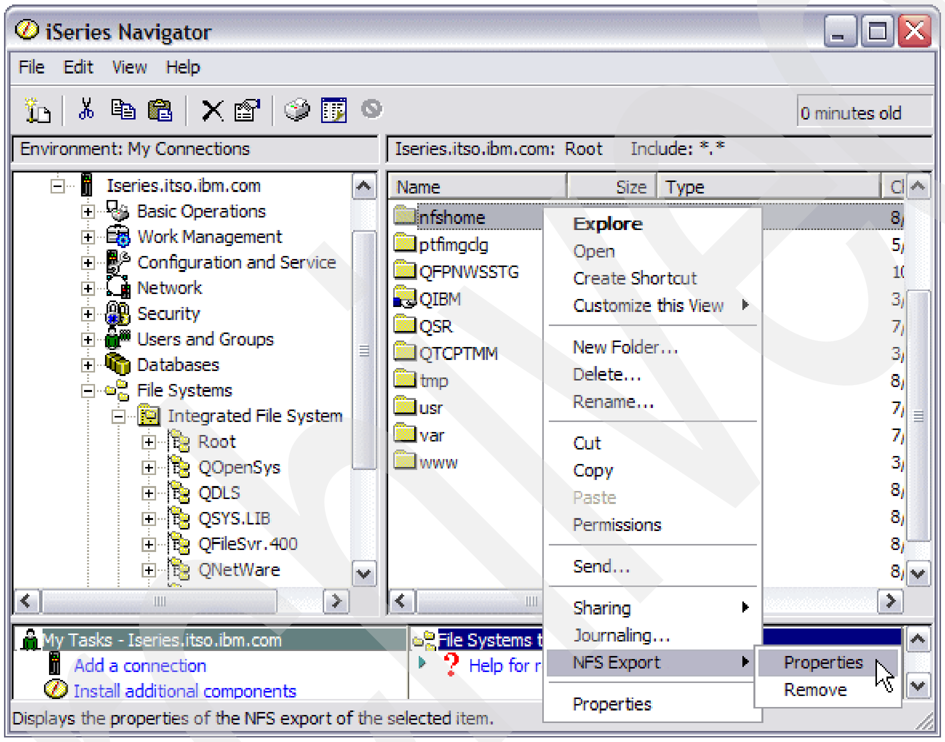
This section shows how to configure and administer NFS server on the IBM i5/OS side, using iSeries Navigator. Perform the following steps:

1. Create a directory in the IFS to hold the Linux files such as:

   ```
   MKDIR DIR('/nfshome')
   ```

2. On the iSeries Navigator select **File System** → **Integrated File System** → **Root**.

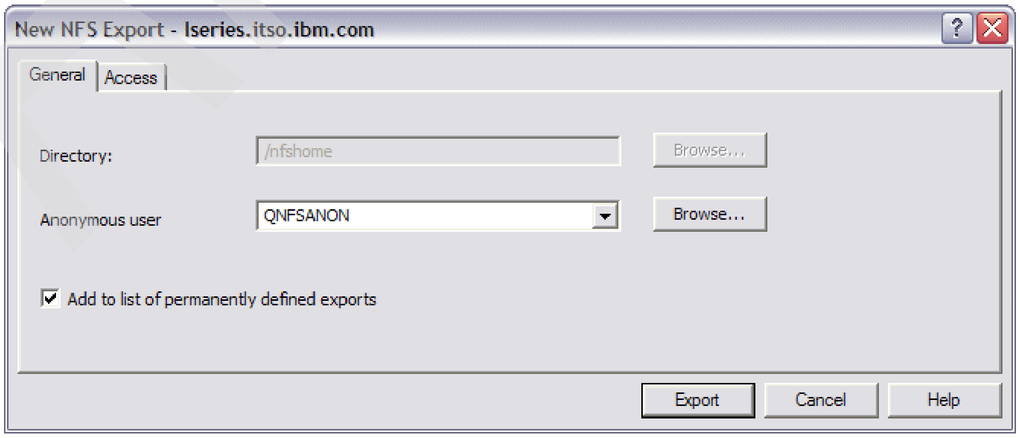
   Choose the desired directory which can hold the NFS files. Right-click, select **NFS Export** and then **Properties** as shown in Figure 6-1.



Displays the properties of the NFS export of the selected item.

Figure 6-1 Create an NFS Export with iSeries Navigator

3. Check that the details are correct (by default user profile QNFSANON is used) then select **Export** as shown in Figure 6-2.



Directory: /nfshome

Anonymous user QNFSANON

☑ Add to list of permanently defined exports

Figure 6-2 Configure the NFS Export /nfshome using iSeries Navigator

Note: If you select **Add to list of permanently defined exports**, an entry is added to the IFS /etc/EXPORTS file.

4. Now start the NFS server. Select **Network** → **Servers** → **TCP/IP**. Right-click **NFS** and then select **Start all**, as shown in Figure 6-3.

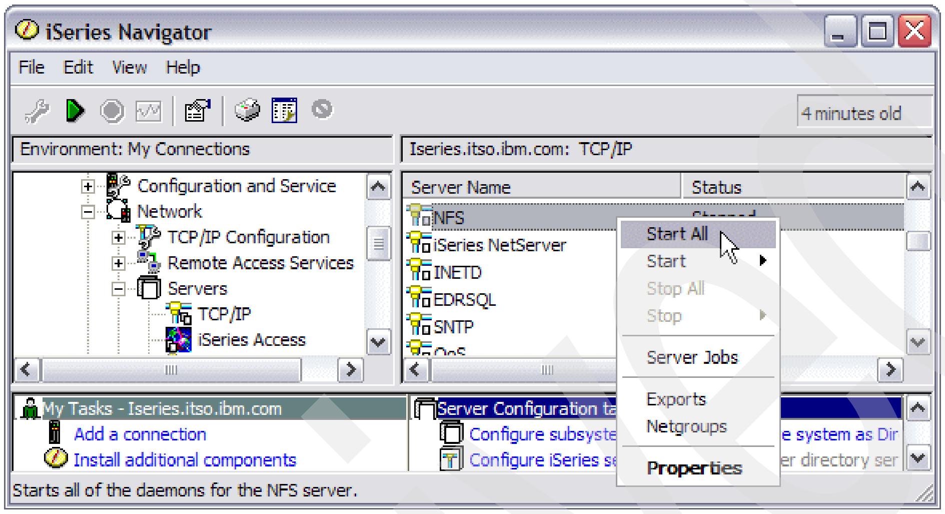


Figure 6-3 Starting NFS server

Now, NFS is ready on IBM i5/OS.

6.3 Mount IBM i5/OS NFS export

For a temporary connection to an available NFS export use the **mount** command. If you want to establish an NFS connection during boot time of Linux then modify the /etc/fstab file to mount the file system at Linux startup. Perform the following steps:

1. Log in to the Linux system as user root.
2. Create a directory as mount point, such as /mnt/iseries:

```
linux:~> mkdir /mnt/iseries
```

3. Mount the NFS export to the directory with the following parameter syntax for the remote file system **mount systemname:/exportname /localmountpoint**:

```
linux:~> mount iseries.itso.ibm.com:/nfshome /mnt/iseries
```

4. The **mount** command without a parameter shows all mounted file systems as shown in Example 6-1.

Example 6-1 All mounted files systems

```
linux:~> mount
/dev/sda3 on / type reiserfs (rw,acl,user_xattr)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
tmpfs on /dev/shm type tmpfs (rw)
devpts on /dev/pts type devpts (rw,mode=0620,gid=5)
iseries.itso.ibm.com:/nfshome on /mnt/iseries type nfs (rw,addr=10.1.1.1)
```

5. Now you can copy files from and to the NFS mounted directory:

```
echo "Test file for NFS" > /mnt/iseries/test.txt
```

6. Verify the new file on the iSeries within a 5250 session or iSeries Navigator:

```
DSPF STMF('/nfshome/test.txt')
```

7. To remove the NFS connection from the Linux system use the following command:

```
umount /mnt/iseries
```

8. If you want to mount NFS exports during system startup, then you must edit the /etc/fstab file. This file contains information about various file systems, such as which ones are mounted at boot time. Be careful with editing because a corrupted fstab file prevents your Linux from starting. See Example 6-2. Add the following information about the NFS export in the correct order:

   a. File system location: systemname:/nfsexport
   b. Local mount point: /mountdirectory
   c. File system type: nfs
   d. Mount options, which means read/write access: rw
   e. Must the file system be dumped (only used for root): 0 no
   f. Order of checking the file system by fsck: 0 no check

Example 6-2 NFS export

```
linux:~> cat /etc/fstab
/dev/sda3            /                  reiserfs   acl,user_xattr        1 1
/dev/sda2            swap               swap       pri=42                0 0
devpts               /dev/pts           devpts     mode=0620,gid=5       0 0
proc                 /proc              proc       defaults              0 0
sysfs                /sys               sysfs      noauto                0 0
iseries.itso.ibm.com:/nfshome /mnt/iseries nfs     rw                    0 0
```

9. The command **mount -a** mounts all file systems like it would be performed at boot time.

6.4 Advanced NFS configuration and security

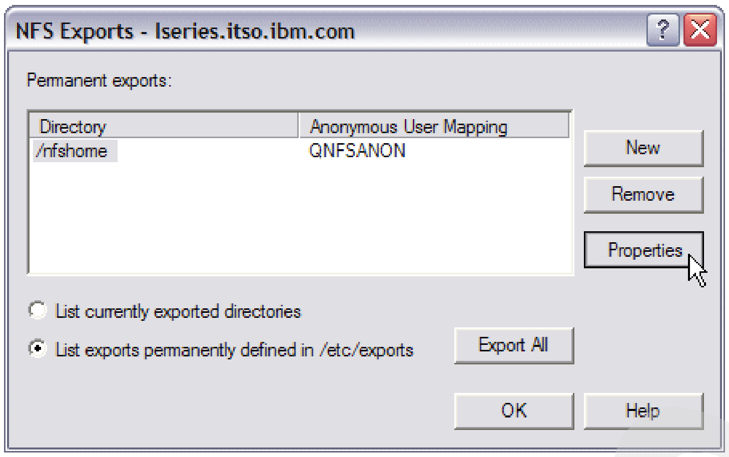
The configuration previously described only provides a simple NFS configuration. It does not contain any security options. Everybody is able to mount the exported file system on his computer because there are no restrictions configured at this time. Also the file access is performed by one user on IBM i5/OS: QNFSANON. Object level security is not possible in this case. The IBM i5/OS cannot differentiate which users access the NFS export. But there are different options to extend the security level.

6.4.1 Restrict access by host IP address

The first thing that you must do is restrict the NFS access to one Linux system or a group of computers. Every NFS export has his own access list. In the default configuration everyone (public) is allowed to read and write to the file system. Use the iSeries Navigator to change this setting:

1. Open in iSeries Navigator **Network** → **Servers** → **TCP/IP**.
2. Right-click the **NFS** server and choose **Exports**.

3. Select the export, which is /nfshome and click **Properties** as shown in Figure 6-4.



NFS Exports - Iseries.itso.ibm.com

Permanent exports:

Directory	Anonymous User Mapping
/nfshome	QNFSANON

New | Remove | Properties

( ) List currently exported directories
(●) List exports permanently defined in /etc/exports

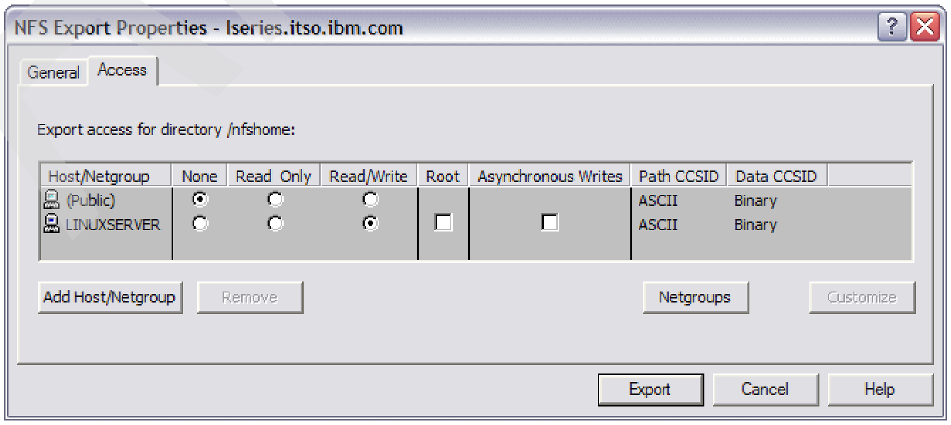
Export All | OK | Help

Figure 6-4 Open Properties of an NFS export in iSeries Navigator

> **Important:** There is a major difference between the two options on the bottom of the configuration window:
>
> - List currently exported directories: changes only the current configuration, the setting is lost after you restart NFS server.
> - List exports permanently defined in /etc/exports: makes all changes permanent.

4. Choose the **Access** tab.
5. In the window, as shown in Figure 6-5, you can configure an access control list. You can define the remote computer either by IP address or by Domain Name System (DNS) name. The available access permissions are None, Read Only, and Read / Write. The option Root is to specify whether the NFS client is allowed access to the root directory as QSECOFR user or with *ALLOBJ permissions. If the option is unchecked, QSECOFR user or users with *ALLOBJ authority are mapped to the QNFSANON profile.
6. If you check the option Asynchronous Writes, then the IBM i5/OS is allowed to cache write requests in memory and perform it at a later time. This can improve performance but may also result in corrupted data if an IBM i5/OS job accesses the file at the same time.

   If you have multiple NFS clients then you could add them into net groups for simpler management.



NFS Export Properties - Iseries.itso.ibm.com

General | Access

Export access for directory /nfshome:

Host/Netgroup	None	Read Only	Read/Write	Root	Asynchronous Writes	Path CCSID	Data CCSID
(Public)	●	○	○			ASCII	Binary
LINUXSERVER	○	○	●	☐	☐	ASCII	Binary

Add Host/Netgroup | Remove | Netgroups | Customize

Export | Cancel | Help

Figure 6-5 Restrict NFS access by host

6.4.2 Apply object level security

Object level security on an NFS exported directory means that a Linux user profile can be identified by IBM i5/OS and mapped to a local user profile. Linux and Unix systems define user by User Identifications (UID) and Group Identifications (GID). UID and GID are unique numbers on each Linux system. The numbers are assigned by the system during creation of user profiles or by an administrator.

User Identification

It is important that the UID is the same on the NFS server and client, otherwise you are connected with the permissions of an other user profile. It does not matter which names the user profile has. In IBM i5/OS, the UID can be set in the creation of a user profile (CRTUSRPRF) or can be changed (CHGUSRPRF). In IBM i5/OS, for example, the QSECOFR user profile has a UID of 0, which is convenient because in Linux the UID of root is also 0. When changing the UID, the user must not own any files in the IFS.

For the example in this book, a Linux user profile named linuxuser was created with UID 1001. You can verify the UID with the Linux command **`id`**:

```
linux:~> id linuxuser
uid=1001(linuxuser) gid=100(users) groups=100(users),14(uucp),16(dialout)
```

The IBM i5/OS user profile is called ISERIESUSR and is also defined with UID 1001:

```
CHGUSRPRF USRPRF(ISERIESUSR) UID(1001)
```

To display the current UID use the following command as shown in Example 6-3.

Example 6-3 Displaying current UID

```
DSPUSRPRF USRPRF(ISERIESUSR)

                         Display User Profile - Basic

User profile . . . . . . . . . . . . . . . :   ISERIESUSR

Object auditing value  . . . . . . . . . . :   *NONE
Action auditing values . . . . . . . . . . :   *NONE
User ID number . . . . . . . . . . . . . . :   1001
Group ID number  . . . . . . . . . . . . . :   *NONE
```

Disable the usage of the QNFSANON user in the NFS export properties:

1. Open in iSeries Navigator **Network** → **Servers** → **TCP/IP**.
2. Right-click the **NFS** server and choose **Exports**.
3. Select the exported directory, which is /nfshome and click **Properties**.
4. On the General tab change the anonymous user to NONE.
5. On the access tab as shown in Figure 6-5 you may turn off the root access.

Now you can log in with the linuxuser and create your own files and directory inside the mounted file system. See Example 6-4.

Example 6-4 Creating your own files and directory

```
linuxuser@linux:~> mkdir /mnt/iseries/linuxuser
linuxuser@mceas213linux:~> ls -l /mnt/iseries/
drwxr-xr-x  2 linuxuser root 8192 2005-08-05 00:00 linuxuser
```

```
linuxuser@linux:~> chmod 700 /mnt/iseries/linuxuser/
linuxuser@linux:~> ls -l /mnt/iseries/
drwx------  2 linuxuser root 8192 2005-08-05 00:00 linuxuser

linuxuser@linux:~> echo "secret of linuxuser" > /mnt/iseries/linuxuser/secret.txt
```

As you turn off the root access to this NFS export, the Linux root user is mapped to the QNFSANON iSeries user. It is not possible to access files with a root account anymore:

```
root@linux:~> ls -l /mnt/iseries/
/bin/ls: /mnt/iseries/: Permission denied
```

Now you are able to separate the data of different users, for example, set up users home directory on IBM i5/OS. Keep in mind that the UID is the only authentication criteria in NFS protocol. No additional user name or password check is performed. If the Linux UID is not mapped to the correct user profile on IBM i5/OS, someone can get unauthorized file access. Especially, if a user is allowed to create his own user profiles on an NFS connected Linux system then he can create any UID he wants to and get access to any files inside the NFS export.

Group Identification

The Group Identification operates in the same fashion as a UID. On a Linux system there is a unique number for every user group. GID allows to build groups of users who have the same subset of permissions.

It is not necessary that you create groups on IBM i5/OS that match groups on Linux, but it is useful for managing access permissions.

In the example mentioned previously, you did not set up any GID on IBM i5/OS. Created files have a GID of null. But on a Linux system a UID or GID of 0 means root. The **`ls -l`** command shows the permission, owner, group, size, and date of a file. The group is shown as root:

```
linuxuser@linux:~> ls -l /mnt/iseries/
drwx------  2 linuxuser root 8192 2005-08-05 00:00 linuxuser
```

To change this behavior you can create a group on IBM i5/OS with the same GID as the group of the linuxuser. To obtain the GID of the group use the Linux command **`id`**:

```
linux:~> id linuxuser
uid=1001(linuxuser) gid=100(users) groups=100(users),14(uucp),16(dialout)
```

On IBM i5/OS create a group profile with the GID of 100:

```
CRTUSRPRF USRPRF(USERSGRP) PASSWORD(*NONE) TEXT('Linux USERS group') GID(100)
```

Now the linuxuser can create a new file within the NFS mount:

```
linux:~> mkdir /mnt/iseries/linuxuser
linux:~> echo "GID TEST" > /mnt/iseries/linuxuser/gid100.txt

linux:~> ls -l /mnt/iseries/linuxuser/
-rw-r--r--  1 linuxuser users 9 2005-08-05 18:00 gid100.txt
```

7

Connect IBM i5/OS with FTP

This chapter discusses the File Transfer Protocol (FTP) method of accessing data stored on an IBM i5/OS system from a Linux operating system. Covered methods include:

- FTP operation tips
- Running FTP commands in interactive mode
- Running FTP commands in batch mode

7.1 FTP operations

FTP is the standard for Transmission Control Protocol/Internet Protocol (TCP/IP). FTP consists of both a client and a server application. A client application is included with most operating systems and is run typically in an interactive mode. The server application is also included in most operating systems. However, the server application runs in a background or batch mode and typically must be started explicitly.

7.1.1 Common FTP commands

There are two ways to transfer data in FTP. These are binary and American Standard Code for Information Interchange (ASCII) mode. In binary mode the bits of the file are copied exactly as is. No conversion is done to the data. Binary mode is used for transferring actual programs, image files, audio files, and other similar files. In non-binary mode, data may be converted between the two systems. Non-binary mode, or ASCII mode, is used for text files, program source, and other files that are read by humans. By default you start your FTP session in non-binary mode. To switch to binary mode enter FTP command **`bin`**.

To start an FTP session, enter **`FTP`** ***`server`*** where *server* is the host name or IP address of the system you want to transfer files with. After you start the FTP session, you can use the command **`get`** to get files from the remote system to the local system and the command **`put`** to put files from the local system to the remote system. You can perform many **`get`** and **`put`** commands during a single FTP session.

When you initially FTP into IBM i5/OS, you are in the QSYS file system. Files are accessed by entering library/file.member for member objects within a file or library/object for non-file objects. You can also access files in the Integrated File System (IFS). To switch to IFS mode you can enter the FTP command **`quote site namefmt 1`** or **`cd /`**. To access QSYS files you can use /qsys.lib/library.lib/file.file/member.mbr or /qsys.lib/library.lib/object.type. To access files in IFS you can use /directory1/directory2/file.name where file.name is the name of the file you want to transfer.

Other useful FTP commands are **`cd`** and **`lcd`**, which change the directory location on the remote and local systems respectively. The **`prompt`** command is useful in conjunction with the **`mget`** or **`mput`** commands. The **`prompt`** command turns off interactive prompting which you may want to do, because **`mget`** or **`mput`** allow you to **`get`** or **`put`** multiple files with a single command.

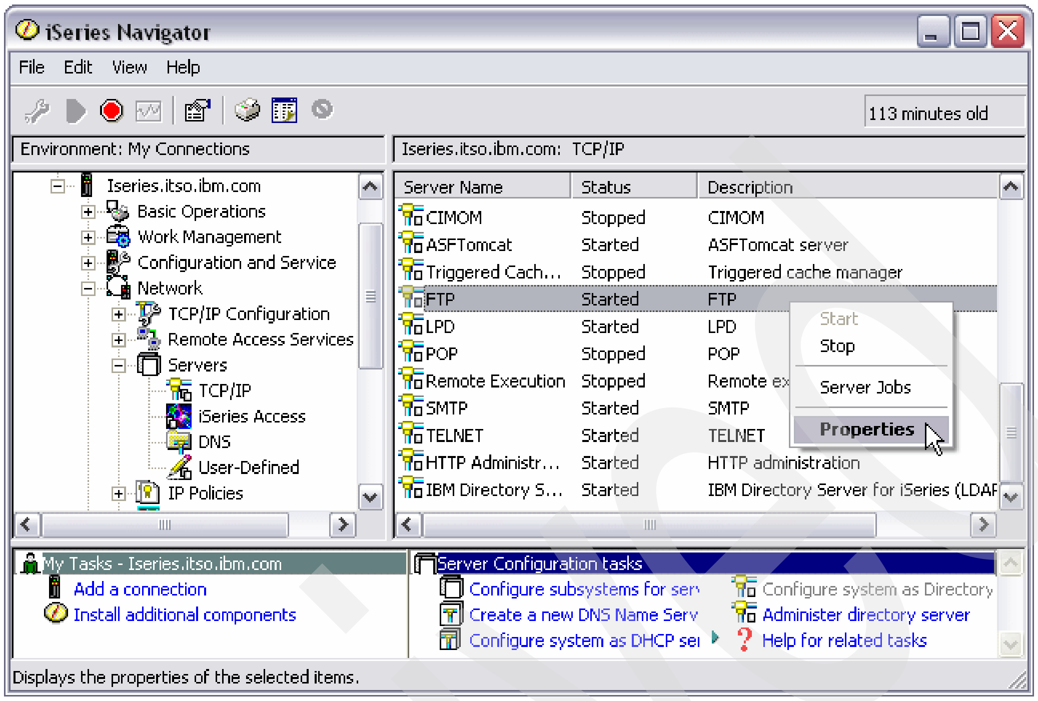
7.1.2 Managing FTP server

To configure the i5/OS FTP server use the command CHGFTPA. For starting and stopping FTP server use the commands STRTCPSVR *FTP and ENDTCPSVR *FTP.

Alternatively use the iSeries Navigator. The FTP Server is located in Network → Server → TCP/IP → FTP.

Additionally, the iSeries Navigator shows the current status of the FTP server.

See Figure 7-1.



iSeries Navigator

File Edit View Help

113 minutes old

Environment: My Connections

- Iseries.itso.ibm.com
  - Basic Operations
  - Work Management
  - Configuration and Service
  - Network
    - TCP/IP Configuration
    - Remote Access Services
    - Servers
      - TCP/IP
      - iSeries Access
      - DNS
      - User-Defined
    - IP Policies

Iseries.itso.ibm.com: TCP/IP

Server Name	Status	Description
CIMOM	Stopped	CIMOM
ASFTomcat	Started	ASFTomcat server
Triggered Cach...	Stopped	Triggered cache manager
FTP	Started	FTP
LPD	Started	LPD
POP	Stopped	POP
Remote Execution	Stopped	Remote ex
SMTP	Started	SMTP
TELNET	Started	TELNET
HTTP Administr...	Started	HTTP administration
IBM Directory S...	Started	IBM Directory Server for iSeries (LDAP

My Tasks - Iseries.itso.ibm.com
- Add a connection
- Install additional components

Server Configuration tasks
- Configure subsystems for serv
- Create a new DNS Name Serv
- Configure system as DHCP ser
- Configure system as Directory
- Administer directory server
- Help for related tasks

Displays the properties of the selected items.

Figure 7-1 Manage i5/OS FTP Server with iSeries Navigator

If you want to run an FTP server on a Linux server, refer to the Linux documentation of your distribution about the FTP management. There are different implementations of an FTP server available for Linux. Therefore, we cannot provide a general management description. Some Linux distributions have there own configuration utility or a tool such as webmin.

7.2 Using Linux as an FTP client

To transfer data from Linux to IBM i5/OS while on the Linux system, simply enter **`ftp server`** where *server* is the IP host name of the IBM i5/OS system or the IP address of the system. After this, you are prompted for a user and then a password in the FTP session. These require to be a valid IBM i5/OS user profile and password. After getting logged on you can use the **`put`** command to transfer a file from Linux to IBM i5/OS. By default you are operating in the QSYS file system. This is the traditional file system with libraries, files, and members. To move, for example, a program source file from Linux to IBM i5/OS you can issue the following command from within the FTP session:

```
put ~/client.c itso/qcsrc.client
```

This moves the file client.c from user's home directory to member CLIENT in the file QCSRC in the library ITSO on the i5/OS side. The data is converted from ASCII to Extended Binary Coded Decimal Interchange Code (EBCDIC) by the FTP server. If the file is something such as an image file that must not be translated, then you must first issue the **`bin`** FTP command.

You can also get files from IBM i5/OS to Linux using the **get** command. Example 7-1 shows how you can get files from IBM i5/OS to Linux.

Example 7-1 FTP transfer from iSeries to Linux using the mget command

```
# ftp iseries.itos.ibm.com
Connected to iseries.itos.ibm.com.
220-QTCP at ISERIES.ITSO.IBM.COM.
220 Connection will close if idle more than 5 minutes.
Name (iseries:linuxuser): ITSOUSER
331 Enter password.
Password:
230 ITSOUSER logged on.
Remote system type is .
ftp> prompt
Interactive mode off.
ftp> quote site namefmt 1
250  Now using naming format "1".
ftp> cd /itsodir/src
250 "/itsodir/src" is current directory.
ftp> mget *.c
Subcommand EPSV not valid.
local: client.c remote: client.c
227 Entering Passive Mode (9,155,34,180,220,160).
150 Retrieving file /itsodir/src/client.c
250 File transfer completed successfully.
4127 bytes received in 00:00 (94.21 KB/s)
local: server.c remote: server.c
227 Entering Passive Mode (9,155,34,180,66,131).
150 Retrieving file /itsodir/src/server.c
250 File transfer completed successfully.
4185 bytes received in 00:00 (102.92 KB/s)
ftp> quit
```

This sequence of commands retrieves all c source files out of the /itsodir/src IFS directory and places them in the current Linux directory. It uses the **prompt** command to turn off interactive prompting and then the **mget** command to get all files matching the wildcard.

7.3 Using IBM i5/OS as FTP client

Transferring data from IBM i5/OS is the same as transferring data from Linux, except the fact that i5/OS has different file systems. The FTP command is issued from an IBM i5/OS command line and the FTP server requires to be up and running on the Linux system. After logging in to a Linux FTP server you must choose your local i5/OS file system. Typically you are using the QSYS file system. If you have to change your local directory to the IFS, then use the FTP command **namefmt 1**. The following sequence of commands accomplishes the same transfer of all c source files from IBM i5/OS to Linux as in the case of Example 7-1:

```
prompt
namefmt 1
lcd /itsodir/src
cd /home/itso/src
mput *.c
```

The **lcd** command changes the directory of the local system which, in this case, is the IBM i5/OS system. The **cd** command changes the directory on the remote or Linux system. Without the **prompt** command, before each file is transferred, you are prompted whether to transfer that file.

7.4 Running FTP commands in batch mode

So far we have shown the FTP commands running interactively. While this is the common usage, it is possible to execute FTP commands in a batch or program mode. Inside a control language (CL) program in IBM i5/OS or a script file on the Linux system, you can enter the FTP commands, provide or be prompted for a user and password, and then run a number of FTP commands. You can use the FTP command **rcmd** to run a remote command.

Example 7-2 is a simple script file that connects as user ITSOUSER, prompts for the password, creates an IBM i5/OS library, creates a number of source physical files, puts a number of source files, creates one of the CL programs that was just downloaded, and runs the newly created program.

Example 7-2 Sample FTP script file: download.txt

```
itsouser

quote rcmd crtlib sysmgmt text('Systems Management Tools')

quote rcmd crtsrcpf sysmgmt/qclsrc text('CL Programs')

put ~itso/sysmgmt/chgcmddft.cl sysmgmt/qclsrc.chgcmddft
put ~itso/sysmgmt/startmsg.cl sysmgmt/qclsrc.startmsg
put ~itso/sysmgmt/startsys.cl sysmgmt/qclsrc.startsys
put ~itso/sysmgmt/startup.cl sysmgmt/qclsrc.startup
put ~itso/sysmgmt/mapkeys.cl sysmgmt/qclsrc.mapkeys
put ~itso/sysmgmt/create.cl sysmgmt/qclsrc.create
put ~itso/sysmgmt/snads.cl sysmgmt/qclsrc.snads
put ~itso/sysmgmt/ctl.cl sysmgmt/qclsrc.ctl
put ~itso/sysmgmt/dev.cl sysmgmt/qclsrc.dev
put ~itso/sysmgmt/lin.cl sysmgmt/qclsrc.lin
put ~itso/sysmgmt/pt.cl sysmgmt/qclsrc.pt
put ~itso/sysmgmt/tn.cl sysmgmt/qclsrc.tn

quote rcmd crtclpgm sysmgmt/create sysmgmt/qclsrc

quote rcmd call sysmgmt/create

quit
```

To use this FTP script file on Linux, use the following command:

```
ftp system < download.txt
```

8

Connect IBM i5/OS with socket programming

This chapter discusses a programming method of accessing data stored in IBM i5/OS from a Linux operating system. It discusses methods for processes to communicate between the two platforms.

This chapter discusses the following:

- General considerations
  - American Standard Code for Information Interchange (ASCII) and Extended Binary Coded Decimal Interchange Code (EBCDIC) character sets
  - Endian byte order
- Sockets
  - Client example
  - Server example

8.1 General considerations

Before starting with socket programming using the programming language C, there are a couple of things that you have to watch out for. First is the character set used in the different platforms and second is the byte order scheme used by the different architectures.

8.1.1 ASCII and EBCDIC character sets

You are probably aware that the IBM i5/OS runs natively in an EBCDIC character set while Windows and all flavors of UNIX (including Linux) run in an ASCII character set. This is most noticeable when you handle integration between the platforms at a lower level, because higher levels of data movement usually handle the conversion automatically. Three common mechanisms for handling the different character sets include:

- Using the capabilities in DB2 Universal Database (UDB) for iSeries

  If the data is something that you require and the integration method allows you to use database tables, then you definitely must use them. The database capabilities in IBM i5/OS easily handles the conversion between ASCII and EBCDIC (and also many other character set translations). The Open Database Connectivity (ODBC) and Java Database Connectivity (JDBC) examples use this mechanism.

- Using unicode

  Unicode is a larger character set that is available on both the IBM i5/OS and the Linux platform. This is the approach that Java uses extensively.

- Performing your own conversion

  You can perform the conversion on either the Linux or IBM i5/OS side. However, it is probably the easiest and best to perform the conversion in IBM i5/OS. There are numerous tools available for character conversion. The socket example code demonstrates the use of the *iconv()* application programming interface (API) to convert from one coded character set identifier (CCSID) to another.

The example program this book describes uses the IBM i5/OS *iconv()* API for ASCII - EBCDIC conversion. The approach we took was to always send the message in ASCII format. This means that only the program on IBM i5/OS requires to convert messages.

8.1.2 Endian byte order

The IBM eServer i5 system works with Power processors. One of the differences between the Power architecture and Intel x86 architecture is the byte ordering.

A 32-bit binary integer is always represented as four bytes in the memory of a computer. On a PowerPC System, four bytes such as 0x0A0B0C0D is stored with the 0A byte at the first in memory, 0B at the second, and so on. This is called big endian because the high order (“big”) byte is the first one in memory.

However, on an x86 processor, the representation is the other way around. It is called little endian because the low order (“little”) byte is stored at the first position in memory. Thus, in little endian, the 0D is stored at the first byte position, 0C at the second, and so on.

It is important that programmers understand the situation and code in such a way that they can avoid depending on the exact order. This way, code written on an x86 Linux system can easily be ported to run on an IBM eServer i5 Linux partition.

Table 8-1 shows several examples and the differences.

Table 8-1 Bytes ordering (endian)

Decimal value, C source code value	Big endian representation	Little endian representation
125 - 0x7D	7D	7D
500 - 0x01F4	01 F4	F4 01
200,000,000 - 0x0BEBC200	0B EB C2 00	00 C2 EB 0B
4,294,967,298 - 0x0100000002	00 00 00 01 00 00 00 02	02 00 00 00 01 00 00 00

The best thing to do to avoid byte ordering or endian problems is to minimize endian dependencies. The following are the steps you can take in C and C++ programs:

- Avoid casting smaller sized storage over larger sized ones (for example, no (short *) over (int *)).
- Avoid casting larger sized storage over smaller sized ones (for example, no (int *) over (short *)).
- Avoid casting one structure over another.
- Avoid unions with different sized items (this is almost the same as saying avoid unions).
- Avoid the use of bit variables.

If you cannot avoid these coding styles, then you must employ the use of suitable #ifdef statements with big endian and little endian versions of the code provided. This lets the compiler select the correct byte order dependent code.

8.2 Sockets programming example

Sockets and the programs written on top of sockets are able to handle almost any integration you want to provide between Linux and IBM i5/OS. In fact, most of the other methods discussed in this book use some flavors of sockets or another under the cover. Example 8-1 shows how to send a text message from one system to the other using sockets interface.

8.2.1 Client code example

The client program has two input parameters. The first parameter is the host name or Internet Protocol (IP) address of the server where the message is sent. The second parameter is the text to send. If the second parameter is left off then a default message is sent.

The client code is C program that opens a socket on a specific port to a specific host system, sends data using the socket, and finally closes the socket. It also converts from EBCDIC to ASCII when it runs on the IBM i5/OS system. Multiple #ifdef statements around the IBM i5/OS specific code allows the exact same code to compile and run on Linux and on IBM i5/OS.

To create the client program on Linux, issue the following command:

```
linux:~> gcc -o client client.c
```

See Example 8-1.

Example 8-1 Client source code client.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <errno.h>
#include <netdb.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

/*! Specify the TCP / IP port of the server pgm here*/
#define SERVER_PORT 30595
#define BUFFER_SIZE 250

/*! The following statments are used if the operating system is IBM i5/OS*/
#ifdef __OS400__
#include <iconv.h>

typedef struct {
  char name[8];
  char ccsid[5];
  char conversion[3];
  char substitute[1];
  char shift[1];
  char input[1];
  char option[1];
  char RESERVED[12];
} fromCode_t;

typedef struct {
  char name[8];
  char ccsid[5];
  char RESERVED[19];
} toCode_t;
#endif

int main(int argc, char *argv[])
{
  int rc;
  char buffer[BUFFER_SIZE],
       converted[BUFFER_SIZE],
       *inBuffer,
       *convertBuffer,
       server[255];

#ifdef __OS400__
  /*! iconv() is used for converting ASCII to EBCDIC on IBM i5/OS */
  size_t inBufferLength,
         convertBufferLength;
  fromCode_t fromCCSID;
  toCode_t toCCSID;
  iconv_t convertTable;
#endif


  /*! socket() variables */
  int descriptor,
      addressFamily,
      type,
      protocol;

  /*! sendto() variables */
  int bufferLength,
      flags,
      serverLength;
  struct sockaddr_in serverAddress;

  /*! gethostbyname() variables */
  struct hostent *host;

  /*! read the input variables*/
  if (argc > 2) {
    bufferLength = (strlen(argv[2]) < BUFFER_SIZE) ? strlen(argv[2]) :
BUFFER_SIZE;
    strncpy(buffer, argv[2], bufferLength);
  } else {
    strcpy(buffer, "Default data");
    bufferLength = strlen(buffer);
  }

  /*! open the socket descriptor */
  addressFamily = AF_INET;
  type = SOCK_DGRAM;
  protocol = IPPROTO_IP;
  if ((descriptor = socket(addressFamily, type, protocol)) < 0) {
    printf("Error getting socket descriptor\n");
    printf("errno is %i\n", errno);
    exit(-1);
  }

#ifdef __OS400__
  /*! setup the conversion table ONLY on IBM i5/OS*/
  memcpy(toCCSID.name, "IBMCCSID", 8);
  memcpy(toCCSID.ccsid, "01252", 5);
  memset(toCCSID.RESERVED, 0x00, 19);
  memcpy(fromCCSID.name, "IBMCCSID", 8);
  memcpy(fromCCSID.ccsid, "00037", 5);
  memcpy(fromCCSID.conversion, "000", 3);
  memcpy(fromCCSID.substitute, "0", 1);
  memcpy(fromCCSID.shift, "0", 1);
  memcpy(fromCCSID.input, "0", 1);
  memcpy(fromCCSID.option, "0", 1);
  memset(fromCCSID.RESERVED, 0x00, 12);
  convertTable = iconv_open((char *)&toCCSID, (char *)&fromCCSID);
  if (convertTable.return_value < 0) {
    printf("Error creating conversion table\n");
    printf("errno is %i\n", errno);
    close(descriptor);
    exit(-1);
  }

  /*! convert the data */
  convertBufferLength = inBufferLength = strlen(buffer);
  inBuffer = buffer;
  convertBuffer = converted;
  memset(converted, 0x00, sizeof(converted));
  iconv(convertTable, &inBuffer, &inBufferLength, &convertBuffer,
&convertBufferLength);
#else
  memcpy(converted, buffer, bufferLength);
#endif

  /*! get the server's address */
  memset(&serverAddress, 0x00, sizeof(serverAddress));
  strcpy(server, argv[1]);
  if ((serverAddress.sin_addr.s_addr = inet_addr(server)) == (unsigned
long)INADDR_NONE) {
    host = gethostbyname(server);
    if (host == (struct hostent *)NULL) {
      printf("Host not found\n");
      printf("h_errno is %d\n", h_errno);
      close(descriptor);
      exit(-1);
    }
    memcpy(&serverAddress.sin_addr, host->h_addr, sizeof(serverAddress.sin_addr));
  }

  /*! send the message to the server */
  flags = 0;
  serverAddress.sin_family = AF_INET;
  serverAddress.sin_port   = htons(SERVER_PORT);
  serverLength = sizeof(serverAddress);
  rc = sendto(descriptor, converted, bufferLength, flags,
              (struct sockaddr *)&serverAddress, serverLength);
  if (rc < 0) {
    printf("Error sending reqeusts\n");
    printf("errno is %i\n", errno);
    close(descriptor);
    exit(-1);
  }
#ifdef __OS400__
  /*! close the conversion table */
  iconv_close(convertTable);
#endif

  /*! close the socket */
  close(descriptor);

  exit(0);
} /*! ... The End */
```

8.2.2 Server code example

The server program is the program that waits for incoming messages. It binds a socket to port 30595 and then waits for incoming messages. If the program receives a message then it prints the message text to standard out. No parameters are required for the server program.

To create the server program on IBM i5/OS, issue the following command in this order:

1. `CRTCMOD MODULE(ITSOLIB/SERVER) SRCSTMF('/itsodir/src/server.c') TGTCCSID(37)`
2. `CRTPGM PGM(ITSOLIB/SERVER) MODULE(ITSOLIB/SERVER) BNDSRVPGM(QSYS/QTQICONV)`

Example 8-2 shows the server code. The server code waits on a specific port for a message from a client. When the message is received it is sent to standard output. The only difference between the Linux version and the IBM i5/OS version is the requirement to convert the incoming message from ASCII to EBCDIC.

Example 8-2 Server source code server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <errno.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>

/*! Specify the TCP / IP port of the server pgm here*/
#define SERVER_PORT 30595
#define BACK_LOG 20
#define BUFFER_SIZE 250

/*! The following statments are used if the operating system is IBM i5/OS*/
#ifdef __OS400__
#include <iconv.h>

typedef struct {
  char name[8];
  char ccsid[5];
  char conversion[3];
  char substitute[1];
  char shift[1];
  char input[1];
  char option[1];
  char RESERVED[12];
} fromCode_t;

typedef struct {
  char name[8];
  char ccsid[5];
  char RESERVED[19];
} toCode_t;
#endif

int main(int argc, char *argv[])
{
  int rc,
        bufferLength;
    char buffer[BUFFER_SIZE],
         converted[BUFFER_SIZE],
         *inBuffer,
         *convertBuffer;

#ifdef __OS400__
/*! iconv() is used for converting ASCII to EBCDIC on IBM i5/OS */
  size_t inBufferLength,
         convertBufferLength;
  fromCode_t fromCCSID;
  toCode_t toCCSID;
  iconv_t convertTable;
#endif

  /*! socket() variables */
  int descriptor,
      addressFamily,
      type,
      protocol;

  /*! bind() variables */
  struct sockaddr_in localAddress;
  int addressLength;

  /*! recvfrom() variables */
  int flags,
      clientLength;
  struct sockaddr_in clientAddress;

  /*! open the socket descriptor */
  addressFamily = AF_INET;
  type = SOCK_DGRAM;
  protocol = IPPROTO_IP;
  if ((descriptor = socket(addressFamily, type, protocol)) < 0) {
    printf("Error getting socket descriptor\n");
    printf("errno is %i\n", errno);
    exit(-1);
  }

  /*! bind to the socket */
  memset(&localAddress, 0x00, sizeof(localAddress));
  localAddress.sin_family = AF_INET;
  localAddress.sin_port   = htons(SERVER_PORT);
  localAddress.sin_addr.s_addr = htonl(INADDR_ANY);
  addressLength = sizeof(localAddress);
  if ((rc = bind(descriptor, (struct sockaddr *)&localAddress, addressLength)) <
0) {
    printf("Error binding to the socket\n");
    printf("errno is %i\n", errno);
    close(descriptor);
    exit(-1);
  }

#ifdef __OS400__
```

```
  /*! setup the conversion table ONLY on IBM i5/OS*/
  memcpy(toCCSID.name, "IBMCCSID", 8);
  memcpy(toCCSID.ccsid, "00037", 5);
  memset(toCCSID.RESERVED, 0x00, 19);
  memcpy(fromCCSID.name, "IBMCCSID", 8);
  memcpy(fromCCSID.ccsid, "01252", 5);
  memcpy(fromCCSID.conversion, "000", 3);
  memcpy(fromCCSID.substitute, "0", 1);
  memcpy(fromCCSID.shift, "0", 1);
  memcpy(fromCCSID.input, "0", 1);
  memcpy(fromCCSID.option, "0", 1);
  memset(fromCCSID.RESERVED, 0x00, 12);
  convertTable = iconv_open((char *)&toCCSID, (char *)&fromCCSID);
  if (convertTable.return_value < 0) {
    printf("Error creating conversion table\n");
    printf("errno is %i\n", errno);
    close(descriptor);
    exit(-1);
  }
#endif

  do {
    /*! receive any incoming client requests */
    memset(buffer, 0x00, sizeof(buffer));
    bufferLength = sizeof(buffer);
    flags = 0;
    clientLength = sizeof(clientAddress);
    rc = recvfrom(descriptor, buffer, bufferLength, flags,
                  (struct sockaddr *)&clientAddress, &clientLength);
    if (rc < 0) {
      printf("Error receiving incoming client reqeusts\n");
      printf("errno is %i\n", errno);
      close(descriptor);
      exit(-1);
    }

#ifdef __OS400__
    /*! convert the data if program runs on IBM i5/OS*/
    convertBufferLength = inBufferLength = strlen(buffer);
    inBuffer = buffer;
    convertBuffer = converted;
    memset(converted, 0x00, sizeof(converted));
    iconv(convertTable, &inBuffer, &inBufferLength, &convertBuffer,
&convertBufferLength);
#else
    memcpy(converted, buffer, sizeof(buffer));
#endif

    /*! output the data */
    printf("Received data from: %.i\n", clientAddress.sin_addr.s_addr);
    printf("'%.80s'\n\n", converted);

    /*! stop when 'STOP' is received as the first 4 characters */
  } while (strncmp(converted, "STOP", 4) != 0);
```

```
#ifdef __OS400__
  /*! close the conversion table */
  iconv_close(convertTable);
#endif

  /*! close the socket */
  close(descriptor);

  exit(0);
} /*! ...The End*/
```

Testing Socket scenario

After the client and the server programs are compiled and ready to run, you must start the server socket program to receive a message. You can run the server in either interactive mode or in batch mode. On both IBM i5/OS and Linux, the messages sent to the server are outputted to standard out. To run the server interactively on IBM i5/OS, use the command CALL ITSOLIB/SERVER. To run the server interactively on Linux you use the **`./server`** command. To run the server code in a background or batch jobs, you can use the following commands:

- On IBM i5/OS:

  ```
  SBMJOB CMD(CALL ITSOLIB/SERVER)
  ```

  This submits a batch job and by default standard out, it goes to a spooled file.

- On Linux:

  ```
  linux:~> server > output.file &
  ```

  This starts a background job and redirects standard out to file `output.file` in the current directory.

Now you can send a message to your server. To run the client program on IBM i5/OS you can use the command:

```
CALL ITSO/CLIENT PARM(Servername ‘This is my text message’)
```

Here, *Servername* is the host name or IP address of the server you want to send your message to. To end the server program send it a STOP message. On Linux you can run the program using:

```
client Servername ‘This is my text message’
```

Example 8-3 is a sample of the output generated by a server running on a Linux system.

Example 8-3 Sample server output

```
linux:~> server
Received data from: 151346265
'This is a message from iSeries'

Received data from: 151346266
'This is text from Linux'

Received data from: 151346266
'STOP'

linux:~>
```

A

Building an SSO scenario using i5/OS projected profiles

This chapter discusses the planning and implementation of a single sign-on (SSO) solution for users who log in to i5/OS, Linux, and Windows machines. The method explained here is custom solution that you can built to enable a single username or password for the users.

This chapter discusses the following:

- What is single sign-on: perspective
- The requirements and benefits of single sign-on
- The scenario for implementing single sign-on
- The strategy for implementing single sign-on
- Configuring Linux and Samba server for Windows authentication
- Linux integration with i5/OS Lightweight Directory Access Protocol (LDAP)
- Configuring Windows clients for authentication with Samba server

Scenario description

This section defines what we mean by single sign-on and gives a brief description of the scenario we are implementing.

What is single sign-on

"Single sign-on" as a term can be defined in different ways. One way of defining it is "The ability to authenticate once and subsequently access network resources throughout the session without being prompted again".

Another definition may be termed as "The ability to use the same credentials to access resources throughout the network, you may be prompted for authentication each time you access a new resource". For the purpose of this chapter we stick to the latter definition.

The requirement for single sign-on: our perspective

As much as we would have preferred to keep things simple, today's business requirements have become very complex, subsequently no single system can perform everything you may want it to do. The result is a complex IT infrastructure where we utilize various combinations of software, hardware, applications, and such technologies. This brings with it the complexity of managing and integration of these key components. One of the most painful areas in such a scenario is to manage and administer authentication and authorization systems for all these disparate processes. This is where the concept of single sign-on steps comes in.

Single sign-on, in very simple terms, is the ability of a user to utilize a single set of authentication for all these systems. The user can authenticate using username or password, biometric devices, smartcard, or any such mechanism. The underlying principle being that he must be able to access resources with the same set of credentials. Without such a mechanism it is nearly impossible for a person to keep track of the different set of credentials that the person may require to access different resources.

It is common to find a single person maintaining and remembering a dozen or so different credentials for the various systems that the person uses. This can be a big management and also a personal overhead apart from being a security issue. It is easier to protect a single username/password rather than having to protect a dozen.

The benefits for a single sign-on

With single sign-on, it aims to reduce all this complexity and try to integrate all these systems to use the same credentials. The user requires to authenticate to the different systems using the same credentials. A single username/password is sufficient for him to gain access to systems that he requires. Similarly, a single password change is enough if he requires to change his credentials.

The scenario to implement single sign-on

This section outlines the scenario that you can utilize for implementing single sign-on.

Current scenario

The following describe the current scenario:

- The name of the organization is International Technical Support Organization (ITSO)
- ITSO primarily utilizes Windows clients (Windows XP SP2, Windows 2000 Professional SP4, Windows 98, and Windows 95) and Linux clients for their users.

- ITSO has IBM i5/OS and SUSE Linux Enterprise Server 10 as their primary operating systems.
- ITSO uses Samba on Linux as their Primary Domain Controller for authenticating Windows clients. The domain name is ITSODMN.
- All users have a minimum of three username/password sets that enable them to log in to and utilize resources on i5/OS, Linux, and Samba servers respectively.

Desired scenario

The following describe the desired scenario:

- All users must have a single username/password to access all three servers.
- A single point for user creation / modification and general maintenance.
- A single password change must propagate to all three systems.

Figure A-1 helps you to understand the overall scenario that this chapter addresses.

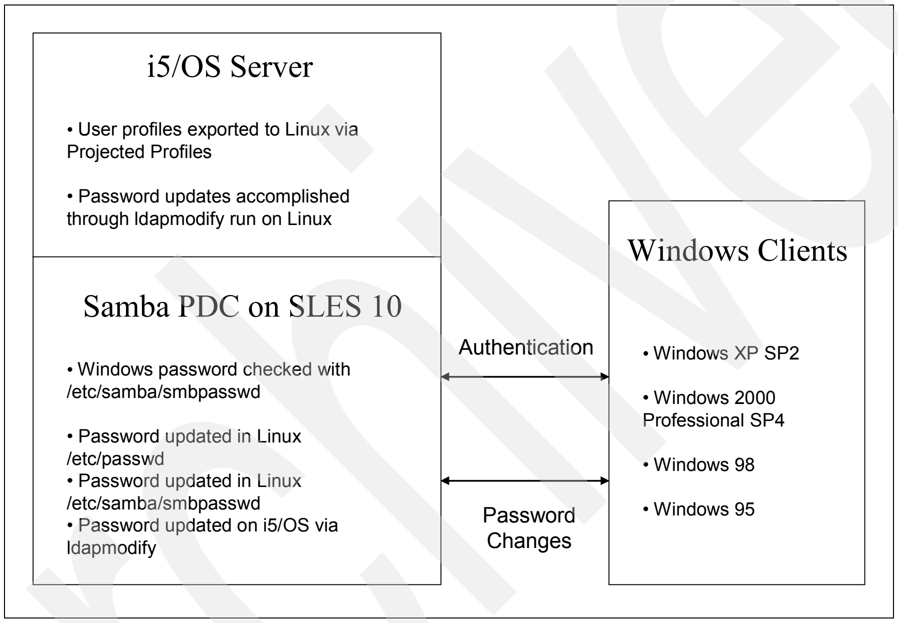

i5/OS Server
• User profiles exported to Linux via Projected Profiles
• Password updates accomplished through ldapmodify run on Linux
Samba PDC on SLES 10
• Windows password checked with /etc/samba/smbpasswd
• Password updated in Linux /etc/passwd
• Password updated in Linux /etc/samba/smbpasswd
• Password updated on i5/OS via ldapmodify
Authentication
Password Changes
Windows Clients
• Windows XP SP2
• Windows 2000 Professional SP4
• Windows 98
• Windows 95

Figure A-1 Logical overview of the scenario

The following are the steps involved in setting up the system and are discussed later in the chapter.

1. i5/OS hosts the Users Profiles. Selected User Profiles are exported to Linux.
2. Linux script is used to import Projected Profiles from i5/OS.
3. Linux and Samba User Accounts/Passwords are created during this import process.
4. Windows clients authenticate using the i5/OS Username/Password.
5. Windows clients change password and the same is synced with Linux/Samba and i5/OS.
6. Password changes to i5/OS are done using scripts on Linux (utilizing **`ldapmodify`** command).

The strategy to implement single sign-on

This section outlines the process that you must follow before a single sign-on mechanism can be implemented in an organization. These are steps that you must follow, to assess the feasibility of implementing such a solution.

Before setting out to implement the scenario, we recommend that you keep the following points in mind:

- Implementing SSO requires you to enforce a similar password policy on all effected systems. In this scenario the systems discussed are Samba (For Windows clients), SUSE Linux Enterprise Server 10, and IBM i5/OS.
- SSO is a fairly complex system to implement. It is also a custom solution that can change according to requirements. What we discuss and implement is a sample scenario for login authentication for users of Windows, Linux, and i5/OS systems.
- We integrate a system where the users use a single set of username/password to log in to any system in this network.
- All users accounts are created and maintained using scripts that run on Linux and i5/OS. These scripts propagate the changes to all systems and also keep future changes synchronized.

Initial environment setup

This section provides the information for initial setup of the scenario environment where we assume you have not implemented any type of SSO mechanism yet.

Downloading the scenario package

Download a scenario package (Scripts.zip file) for implementing the solution being discussed. The package is available in soft copy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

`ftp://www.redbooks.ibm.com/redbook/SG246551`

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the IBM Redbook form number, SG24-6551.

Use your browser and point to the previously mentioned location and save the Scripts.zip file to your Windows client PC. This scenario package contains all the files that you later require to restore/install/configure in the Linux server and the i5/OS server. The following sections explain all these procedures in detail.

Specifically the Scripts.zip archive contains the following structure as Figure A-2 illustrates.

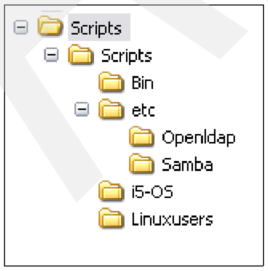


Figure A-2 Scripts.zip file contents

Setup on i5/OS side

This section discusses how to set the scenario environment on the i5/OS side.

Restoring scenario library on i5/OS

First, you have to restore the scenario library on i5/OS. *linuxusers.savf* has the i5/OS save file that contains the code to support exporting of user profiles to Linux. This file is located in the i5-OS Directory of the unzipped Scripts.zip file. See Figure A-2.

Configuring i5/OS LDAP support

Configuration of i5/OS LDAP support for Extended User Attributes requires that the linuxusers save file be restored on the i5/OS partition. You can accomplish this with the following steps:

1. First, transfer the linuxusers.savf file to the i5/OS system using ftp. On the Windows client PC, change the directory to the i5-OS directory within the hierarchy of the unzipped scripts.zip file.
2. Start an ftp session to the i5/OS system, such as:

   ```
   ftp i5-os-system
   ```
3. In the ftp session change the name format parameter to properly recognize DOS style directories:

   ```
   quote site namefmt 1
   ```
4. In the ftp session change the transfer type to properly transfer binary files:

   ```
   bin
   ```
5. In the ftp session, transfer the save-file from the PC to i5/OS:

   ```
   put linuxsers.savf linuxusers.savf
   ```
6. Quit the ftp session:

   ```
   quit
   ```

Restore the save-file

Perform the following steps to restore the save-file:

1. In i5/OS, issue the following command to ensure that the save-file was properly transferred to the i5/OS partition:

   ```
   DSPSAVF FILE(QGPL/LINUXUSERS)
   ```
2. In i5/OS, restore the save-file with the following command:

   ```
   RSTLIB SAVLIB(LINUXUSERS) DEV(*SAVF) SAVF(QGPL/LINUXUSERS)
   ```

Creating required user profiles

To provide the necessary hand-shaking between the Linux and i5/OS LDAP components, two user accounts require to be established. The first user profile (linuxsys1) is used by LDAP in Linux to access the LDAP server in i5/OS. The following command must be issued in i5/OS to create this user profile:

```
CRTUSRPRF USRPRF(LINUXSYS1) PASSWORD(ldap1acc) USRCLS(*SECADM) INLMNU(*SIGNOFF)
LMTCPB(*YES) GID(*NONE)
```

The second user profile, we call it a Linux group user profile (linuxsys1g), is used to group all of the user accounts to be given Linux access as a single group:

```
CRTUSRPRF USRPRF(LINUXSYS1G) PASSWORD(*NONE) INLMNU(*SIGNOFF) LMTCPB(*YES)
GID(*GEN)
```

Adding users to Linux group user profile

Now, add all existing i5/OS users (that you want to include in this scenario) into the linuxsys1g group profile. To accomplish the same, you have to enter the following commands on the i5/OS:

```
CHGUSRPRF USRPRF(CURUSR) GRPPRF(LINUXSYS1G)
```

Replace the CURUSR with the actual name of the user that you have to include in the linuxsys1g group. Based on your setup, repeat this command for all users that you require to configure for this scenario.

Running required commands

After all of the user profiles in i5/OS have been added to the “linuxsys1g” group, the group list in i5/OS requires to be initialized and rebuilt with the following command:

```
CHGCURLIB LINUXSERS
```

- `RMVLNXUSRS LINUXSYS(LINUXSYS1) LINUXGRP(LINUXSYS1G)`

  The RMVLNXUSRS command disables the users in Linux. Specifically, the RMVLNXUSRS command causes the list of Linux users to be cleared.
- `ADDLNXUSRS LINUXSYS(LINUXSYS1) LINUXGRP(LINUXSYS1G)`

  The ADDLNXUSRS command enables the users in Linux. Specifically, the ADDLNXUSRS command causes the list of Linux users to be rebuilt by finding all i5/OS user profiles that belong to the Linux group (in this case LINUXSYS1G) and adding them to the Linux users list.

> **Note:** Execute these commands each time a user profile in i5/OS is added to or removed from the linuxsys1g group.
>
> In addition, run the Linux commands **`doall`**/**`dosingle`**.

Setup on Linux side

This section describes the setup procedure of the scenario on Linux side.

OpenLDAP client configuration

Place the OpenLDAP client configuration file (ldap.conf) in the /etc/openldap directory and modify it to reflect the environment on which LDAP is being used.

Transfer ldap.conf file to Linux system

On the Windows client PC, change the directory to the OpenLDAP directory within the hierarchy of the unzipped scripts.zip file.

To transfer the ldap.conf file to the Linux system use the following steps (while logged in to the PC client:

1. From the Windows PC where the script is located, start an ftp session to the Linux system:

   ```
   ftp linux-system
   ```

2. After logging in to ftp, change the directory to the /usr/src/linuxusers directory:

   ```
   cd /etc/openldap
   ```

3. In the ftp session transfer the ldap.conf file

   ```
   put ldap.conf
   ```

4. Quit the ftp session

   ```
   quit
   ```

Modify the LDAP configuration file to match the environment

Change the host value in the ldap.conf file (host) to the name (or IP address) of the i5/OS partition that the user profiles are being obtained from.

In addition, change the base (base) value to match the default base DN as per your configuration. The default value used in the sample scenario is:

```
base cn=accounts,os400-sys=rchas61.rchland.ibm.com
```

Change suitably as per your setup.

Change the "rootbinddn" value to match the default base DN. Additionally, you have to specify the os400-sys profile as well. This is the user profile (linuxsys1) that is created to support the LDAP queries. This is not the group profile name.

Example A-1 shows the ldap.conf file based on the settings used in the sample.

Example: A-1 ldap.conf file based on settings in sample

```
# LDAP Defaults
#

# See ldap.conf(5) for details
# This file should be world readable but not world writable.

#BASE   dc=example, dc=com
#URI    ldap://ldap.example.com ldap://ldap-master.example.com:666

#SIZELIMIT      12
#TIMELIMIT      15
#DEREF          never

host 9.5.92.96
base cn=accounts,os400-sys=rchas61.rchland.ibm.com

rootbinddn os400-profile=linuxsys1,cn=accounts,os400-sys=rchas61.rchland.ibm.com
pam_login_attribute os400-profile
ldap_version 3
SASL_SECPROPS none

#TLS_REQCERT allow
```

Store the LDAP password

Finally, you have to store the password that is used in the LDAP queries in the /etc/ldap.secret file. Ensure that there is a newline (which means, RETURN/ENTER) at the end of the password. Also, you must change the mode (permissions) of the file so that only root can view the file (chmod 600 /etc/ldap.secret).

Samba configuration

This section discusses the steps involved for configuring Samba Server 3.0.xx on SUSE Linux Enterprise Server 10. Configure Samba to act as Windows-NT PDC and also configure it to synchronize the user passwords with the Linux password database.

The following packages are required to be installed on SUSE Linux Enterprise Version 10 for proper Samba functionality. Ensure that the following packages are installed. Perform the following steps:

1. Use the following Linux command to check for relevant packages. Log in to the root user before you issue this command.

   ```
   # rpm -qa | grep samba
   ```

   You see a list similar to the one in Example A-2. Verify if you have the samba-client-3.0.22-13.16 and samba-3.0.22-13.16 listed out in your configuration:

   Example: A-2 Checking Samba server and its version

   ```
   linux:/etc/samba # rpm -qa | grep samba
   samba-client-64bit-3.0.22-13.16
   yast2-samba-client-2.13.22-0.2
   samba-64bit-3.0.22-13.16
   samba-winbind-64bit-3.0.22-13.16
   samba-client-3.0.22-13.16
   kdebase3-samba-3.5.1-69.29
   samba-3.0.22-13.16
   samba-pdb-3.0.22-13.16
   samba-winbind-3.0.22-13.16
   yast2-samba-server-2.13.11-1.3
   ```

2. The next step is to configure the Samba Server configuration file. This file is named smb.conf and is located in the /etc/samba directory. You are required to edit the file to configure Samba server to act as a Windows NT PDC. After this is completed, you can join Windows client machines to the Samba PDC.
3. The smb.conf file has various distinct sections, the Global Section and the Shares Section and the Printers Section are some of the most commonly used ones. The actual working smb.conf is reproduced for your reference in Example A-3. Make changes specific to your environment and save the file. Give due attention to your Domain Name.

   Example: A-3 Samba configuration file contents example

   ```
   # smb.conf is the main Samba configuration file. You find a full commented
   # version at /usr/share/doc/packages/samba/examples/smb.conf.SUSE if the
   # samba-doc package is installed.
   # Date: 2006-06-17
   [global]

   #Name of the example domain that we are using for this senario, replace this
   name with
   #the correct name for your organization.
      workgroup = ITSODMN
   ```

```
   printing = cups
   printcap name = cups
   printcap cache time = 750
   cups options = raw
   map to guest = Bad User
   include = /etc/samba/dhcp.conf
   logon drive = P:
   add machine script = /usr/sbin/useradd  -c Machine -d /var/lib/nobody -s
/bin/false %m$
   domain logons = Yes
   domain master = Yes
   local master = Yes
   netbios name = ITSOPDC
   os level = 65
   passdb backend = smbpasswd
   preferred master = Yes
   security = user
   wins support = Yes
   unix password sync = Yes

   passwd program = /usr/local/bin/smbchat %u
#This above line call the smbchat script that changes the password across all
platforms. It #is installed in /usr/local/bin directory on the Linux System.
You may want to read the file #for a complete undestanding of how it works.

[homes]
   comment = Home Directories
   valid users = %S, %D%w%S
   browseable = No
   read only = No
   inherit acls = Yes
[profiles]
   comment = Network Profiles Service
   path = %H
   read only = No
   store dos attributes = Yes
   create mask = 0600
   directory mask = 0700
[users]
   comment = All users
   path = /home
   read only = No
   inherit acls = Yes
   veto files = /aquota.user/groups/shares/
[groups]
   comment = All groups
   path = /home/groups
   read only = No
   inherit acls = Yes
[printers]
   comment = All Printers
   path = /var/tmp
   printable = Yes
   create mask = 0600
   browseable = No
```

```
[print$]
   comment = Printer Drivers
   path = /var/lib/samba/drivers
   write list = @ntadmin root
   force group = ntadmin
   create mask = 0664
   directory mask = 0775
[netlogon]
   comment = Network Logon Service
   path = /var/lib/samba/netlogon
   write list = root
```

4. We have provided the smb.conf in the respective directory of the unzipped script.zip file. Feel free to utilize the same. Note that you have to copy the same to /etc/samba/ directory in the Linux system and you also require to change the workgroup to reflect your domain name.

Scenario script configuration

A number of files require to be installed (or copied) to the Linux partition. These files are part of the scripts.zip file that you have extracted earlier.

Install the doall script

Install the script in Linux with the following commands:

1. In Linux, create a directory to hold the script:

   ```
   mkdir /usr/src/linuxusers
   ```

2. On the Windows client PC, change the directory to the bin directory within the hierarchy of the unzipped scripts.zip file.
3. From the Windows PC where the script is located, start an ftp session to the Linux system:

   ```
   ftp linux-system
   ```

4. After logging into ftp, change the directory to the /usr/src/linuxusers directory:

   ```
   cd /usr/src/linuxusers
   ```

5. In the ftp session change the transfer type to properly transfer binary files:

   ```
   bin
   ```

6. In the ftp session transfer the doall script:

   ```
   put doall doall
   ```

7. Quit the ftp session:

   ```
   quit
   ```

Attention: Do not run this script if you have already run it once and imported the bulk of your users. This resets the passwords of all existing users back again. If you want to add a single i5/OS user or a group of them, run the dosingle script followed by the chguser script.

Configuring the script

There are three values (or variables) in the doall script that you have to change to match the configured ldap values as set in the ldap.conf file.

- Change the $dn value to reflect the i5/OS profile (linuxsys1) that was established earlier in support of extended user profiles (os400-profile) and the i5/OS system that the ldap users are being pulled from (os400-sys). Using the example we use in this book, and also the example ldap.conf file, the $dn value in the doall script requires to reflect the following:

  ```
  $dn = 'os400-profile=linuxsys1,cn=accounts, os400-sys=rchas61.rchland.ibmcom';
  ```

- Additionally, change the $b value to reflect the system that the ldap users are being pulled from (os400-sys). Again, using the example ldap.conf file, the $b value reflects the following:

  ```
  $b = 'os400-sys=rchas61.rchland.ibm.com’;
  ```

- Lastly, change the variable $h to reflect the host name of the system that ldap users are being pulled from. Ensure that the name can be resolved either through Domain Name System (DNS) or through the /etc/hosts file on Linux. Again using the example the $h value would reflect the following:

  ```
  $h = ‘rchas61’
  ```

Executing the doall script

You must execute the doall script after the i5/OS setup steps have been completed. It can be executed simply by entering the following command:

```
# perl /usr/src/linuxusers/doall
```

Note: This script is intended to be run the first time when importing users from i5/OS. During the import, this sets the usernames and passwords for such users to the default values. The default passwords are the same as the usernames on both the Linux and also the Samba servers.

Install the chgi5user, chguser, smbchat, and smbpass scripts

Install the rest of the scripts in the bin folder on your Windows PC into the /usr/local/bin directory on the Linux system. Give the following commands to do the same:

1. On the Windows client PC change the directory to the bin directory within the hierarchy of the unzipped scripts.zip file.
2. From the Windows PC, where the script is located, start an ftp session to the Linux system:

   ```
   ftp linux-system
   ```

3. After logging into ftp, change the directory to the /usr/local/bin directory:

   ```
   cd /usr/local/bin
   ```

4. In the ftp session change the transfer type to properly transfer binary files:

   ```
   bin
   ```

5. In the ftp session transfer the following scripts:

   ```
   – put chgi5pass
   – put chguser
   – put smbchat
   – put smbpass
   ```

6. Quit the ftp session:

   ```
   quit
   ```

Detailed logical flow

Support for projected user profiles requires a number of files to be installed in both i5/OS and Linux. As you must have realized by now this entire procedure is fairly complex and you may require to really understand how the system is working. This section explains in details the purpose and function of all the scripts and their relationships. You may require to understand this flow in case you have to change or adapt the scenario to your specific situation or requirement.

- ldap.conf

  Linux client side OpenLDAP configuration file.
- doall

  Perl script to import users with projected profiles from i5/OS. This script imports all the users with projected profiles in i5/OS and creates respective users in the Linux Samba server. It creates both Linux system users and Samba users. Intended to be run the first time when importing users from i5/OS. This resets the usernames and passwords for both Linux and Samba users to their default values. The default value of the passwords is the same as the usernames.
- smbpass

  Bash script is called by doall and it initializes the password for the new users being added to Samba. The default password is the same as the user's username. This is the script that creates and sets the Samba password for all the users. It is called automatically from doall and does not require you to run it individually.
- smbchat

  The main password synchronization script. This is called through the passwwd program parameter in smb.conf file.
- chgi5user

  Bash script that is called by Samba server through the use of passwd program parameter in smb.conf file. The purpose of this file is to look up and change the password for the user in i5/OS and the local Linux system. Samba takes care of changing the password inside the Samba password database. This script is also called by the chguser and smbchat scripts in relevant situations.

Adding new users into domain

This section provides the information for adding new users into the domain which you set up in "Initial environment setup" on page 108.

Prerequisites checkpoint

We assume that you have already installed and configured the scenario package. You can check this by executing the following commands on the i5/OS terminal:

- On the i5/OS side:

  ```
  DSPOBJD OBJ(QSYS/LINUXUSERS) TYPE(*LIB)
  ```
- On Linux side:
  - Check for the existence and correctness of the /etc/openldap/ldap.conf file
  - Check for the existence and correctness of the /etc/samba/smb.conf file.

– Check for the existence and correctness of the following files in the /usr/local/bin directory:
  - chgi5user
  - chguser
  - smbchat
  - smbpass
– Check for the existence and correctness of the following files in the /usr/src/linuxusers directory:
  - doall
  - dosingle

Follow the steps given in the following sections in case everything checks out correctly and you already ran the initial setup process earlier.

Setup on i5/OS side

This section describes the procedure of scenario setup on i5/OS side.

Adding new user to Linux group user profile

Add a new i5/OS user that you want to add as a part of this scenario. You can give the following commands in order to create the user and also to make him a member of the linux group profile (linuxsys1g) that was created in the initial environment setup:

```
CRTUSRPRF USRPRF(NEWUSR) GRPPRF(LINUXSYS1G)
```

Replace the NEWUSR variable in the previous command with the actual username that you want to create. This command creates the respective user and also makes him a member of the correct Linux Group Profile.

Building the i5/OS group list

After adding all the user profiles in i5/OS to the “linuxsys1g” group, initialize the group list in i5/OS and rebuild it with the following command:

```
CHGCURLIB LINUXSERS
```

- `RMVLNXUSRS LINUXSYS(LINUXSYS1) LINUXGRP(LINUXSYS1G)`

  The RMVLNXUSRS command disables the users in Linux. Specifically, the RMVLNXUSRS command causes the list of Linux users to be cleared.

- `ADDLNXUSRS LINUXSYS(LINUXSYS1) LINUXGRP(LINUXSYS1G)`

  The ADDLNXUSRS command enables the users in Linux. Specifically, the ADDLNXUSRS command causes the list of Linux users to be rebuilt by finding all i5/OS user profiles that belong to the Linux group (in this case LINUXSYS1G) and adding them to the Linux users list.

Note: Execute these commands each time a user profile in i5/OS is added to or removed from the linuxsys1g group.

Setup on Linux side

Run the following scripts on Linux side.

Install the dosingle script

Install the script in Linux with the following commands:

1. In Linux, create a directory to hold the script:

   ```
   mkdir /usr/src/linuxusers
   ```

2. On the Windows client PC change the directory to the linuxusers directory within the hierarchy of the unzipped scripts.zip file.
3. From the PC client where the script is located, start an ftp session to the Linux system:

   ```
   ftp linux-system
   ```

4. After logging into ftp, change the directory to the /usr/src/linuxusers directory:

   ```
   cd /usr/src/linuxusers
   ```

5. In the ftp session change the transfer type to properly transfer binary files:

   ```
   bin
   ```

6. In the ftp session transfer the doall script:

   ```
   put dosingle dosingle
   ```

7. Quit the ftp session:

   ```
   quit
   ```

Configuring the dosingle script

There are three values (or variables) in the dosingle script that you must change to match the configured ldap values as set in the ldap.conf file:

- Change the $dn value to reflect the i5/OS profile (linuxsys1) that was established earlier in support of extended user profiles (os400-profile) and the i5/OS system that the ldap users are being pulled from (os400-sys). Using the example we use in this book, and also the example ldap.conf file, the $dn value in the doall script requires to reflect the following:

  ```
  $dn = 'os400-profile=linuxsys1,cn=accounts, os400-sys=rchas61.rchland.ibmcom';
  ```

- Additionally, change the $b value to reflect the system that the ldap users are being pulled from (os400-sys). Again, using the example ldap.conf file, the $b value would reflect the following:

  ```
  $b = 'os400-sys=rchas61.rchland.ibm.com’;
  ```

- Lastly, change the variable $h to reflect the host name of the system that ldap users are being pulled from. Ensure that the name can be resolved either through DNS or through the /etc/hosts file on Linux. Again using the example the $h value would reflect the following:

  ```
  $h = ‘rchas61’
  ```

Execute the dosingle script in the /usr/local/bin directory each time you add a new user in the i5/OS system. Remember to complete the i5/OS steps for adding a user and to run the RMVLNXUSRS and ADDLNXUSRS commands before you run this script.

Execute the dosingle script after all the i5/OS setup steps previously discussed have been completed. Executed it by entering the following command:

```
# perl /usr/src/linuxusers/dosingle
```

Note: This script is intended to be run whenever you add a new user to the i5/OS and want to import the user to Linux and Samba. This script does exactly the same as the "doall" script, except that it does not create the passwords or Samba accounts for these new users. Call the "chguser" script to set the password of the users in Linux and also to create their Samba username/passwords. We recommend that you run this script for your adhoc user addition process. To import the users the first time use the "doall" script.

Execute the chguser script

After the new users are imported using the previously dosingle script, you have to now set their passwords. For the sake of simplicity, we have assumed that all i5/OS users are created with the password same as their usernames to maintain integrity while importing them into Linux. Following from this the chguser bash script is invoked with two arguments, username and password. This scripts automatically set the password for the user both in the Linux System and also the Samba password file. Repeat this step for all the users that you added in i5/OS and then imported using the dosingle script.

You can execute the chguser script as follows. On the Linux command prompt issue the following command:

```
# /usr/local/bin/chguser username password
```

Detailed logical flow

Support for projected user profiles requires a number of files to be installed in both i5/OS and Linux. As you must have realized by now this entire procedure is fairly complex and you may require to really understand how the system is working. The following sections explain in details the purpose and function of all the scripts and their relationships. You may require to understand this flow in case you have to change or adapt the scenario to your specific situation or requirement.

- dosingle

  Perl script to import new users with projected profiles from i5/OS. This script imports all the users with projected profiles in i5/OS and creates respective users in the Linux Server. It only creates the Linux System users. This script does not reset the passwords of the existing i5/OS users in Linux and Samba servers. Use the chguser script to set a Linux password and to create a Samba user and password for these new users.

- chguser

  Bash script invoked with two arguments, username and password. Each time this script is run, it sets the password of the Linux user to the value supplied. It also sets the same password for the Samba user as well. If the Samba user does not exist, it creates the Samba user. This script is intended to be used when you have already imported the bulk of your i5/OS users earlier by utilizing the "doall" script. This is a follow up action after you run dosingle script.

- chgi5user

  Bash script that is called by Samba Server through the use of passwd program parameter in smb.conf file. The purpose of this file is to look up and change the password for the user in i5/OS and the local Linux System. Samba takes care of changing the password inside the Samba password database. This script is also called by the chguser and smbchat scripts in relevant situations.

Configuring Windows clients for authentication with the Samba server

This section describes the steps that you have to follow to incorporate Windows client into the Linux/SAMBA hosted domain and file server environment. The steps presented cover the following:

1. Adding the Windows client to the domain
2. Process for users to follow when they want to change their passwords

Steps are presented for the following clients:

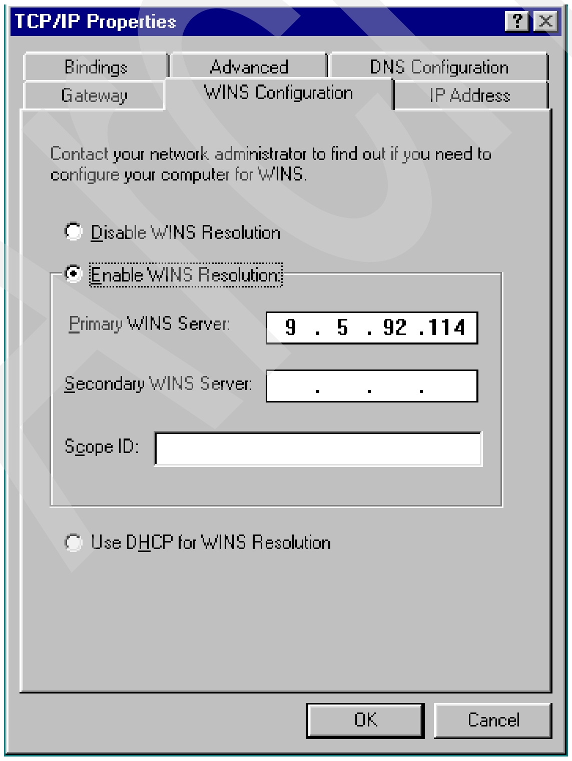
- Windows 95
- Windows 98
- Windows NT Workstation
- Windows 2000 Professional
- Windows XP Professional

You must note that Windows XP Home cannot participate in this environment. There are no mechanisms provided with Windows XP Home to allow it to join a Windows-style domain.

Windows 95: Joining the domain

Perform the following steps:

1. Go to Network Properties.
2. Select **TCP/IP** and then click the <Properties> window.
3. Click the **WINS Configuration** tab.
4. Click the radio button for **Enable WINS Resolution** and then enter the IP address for the Primary WINS Server as shown in Figure A-3. This is the IP address of the Linux server.

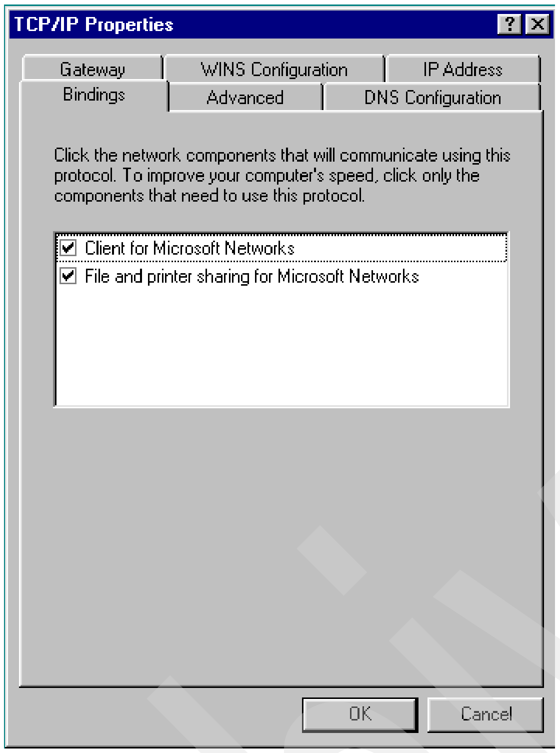


Contact your network administrator to find out if you need to configure your computer for WINS.

Disable WINS Resolution
[✓ Enable WINS Resolution]
Primary WINS Server: 9 . 5 . 92 .114
Secondary WINS Server: . . .
Scope ID:
Use DHCP for WINS Resolution

Figure A-3 Enabling WINS resolution

5. Click the **Bindings** tab from Figure A-3. Then click the radio button in front of **Client for Microsoft Networks** as shown in Figure A-4. Click **OK**.

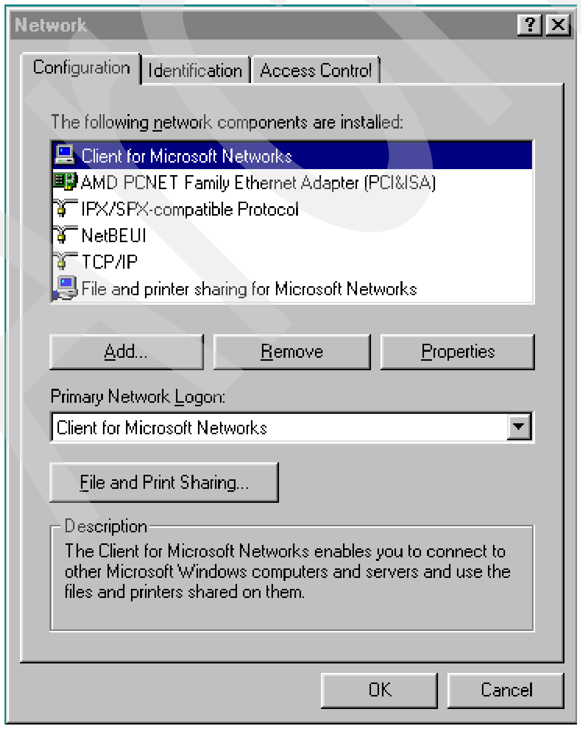


Click the network components that will communicate using this protocol. To improve your computer's speed, click only the components that need to use this protocol.

- [✓] Client for Microsoft Networks
- [✓] File and printer sharing for Microsoft Networks

Figure A-4 Check Client for Microsoft Networks

6. Select **Client for Microsoft Networks** and then click the <Properties> button as shown in Figure A-5.



The following network components are installed:

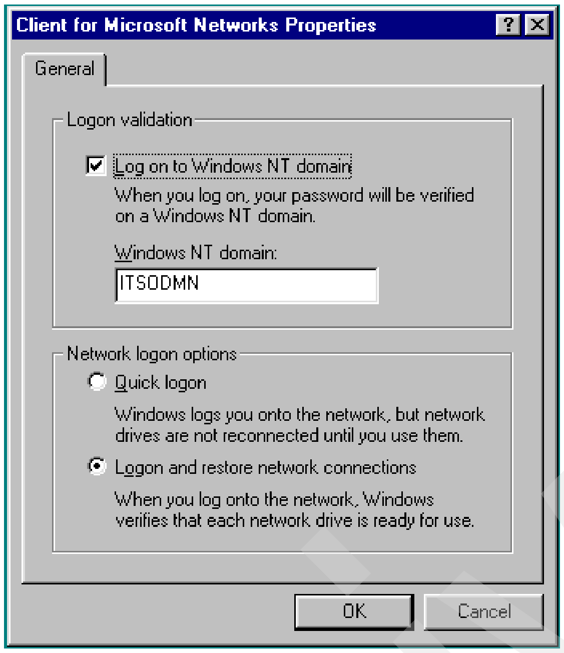
- Client for Microsoft Networks
- AMD PCNET Family Ethernet Adapter (PCI&ISA)
- IPX/SPX-compatible Protocol
- NetBEUI
- TCP/IP
- File and printer sharing for Microsoft Networks

Primary Network Logon:
Client for Microsoft Networks

Description
The Client for Microsoft Networks enables you to connect to other Microsoft Windows computers and servers and use the files and printers shared on them.

Figure A-5 Setting networks properties

7. Select the check box **Log on to Windows NT Domain** and enter `ITSODMN` for the Windows NT Domain Name, then select the **OK** button as shown in Figure A-6.



[✓] Log on to Windows NT domain

When you log on, your password will be verified on a Windows NT domain.

Windows NT domain: ITSODMN

Quick logon

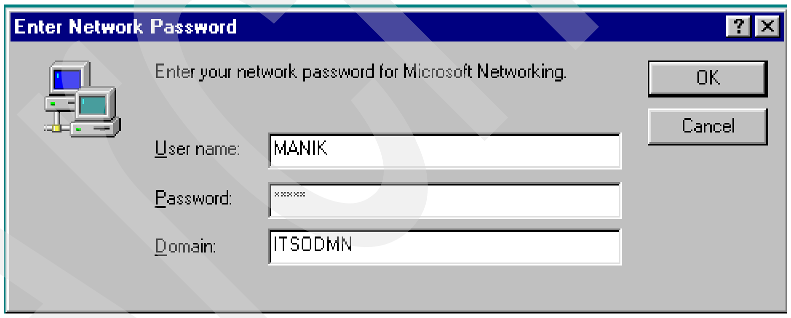
Windows logs you onto the network, but network drives are not reconnected until you use them.

[✓] Logon and restore network connections

When you log onto the network, Windows verifies that each network drive is ready for use.

Figure A-6 Giving Windows NT domain name

8. Back in the Network dialog click the **OK** button. You are prompted to restart the client system.
9. When the client restarts, you are prompted for the user name, password, and domain as shown in Figure A-7.



Enter your network password for Microsoft Networking.

User name: MANIK

Password: *****

Domain: ITSODMN

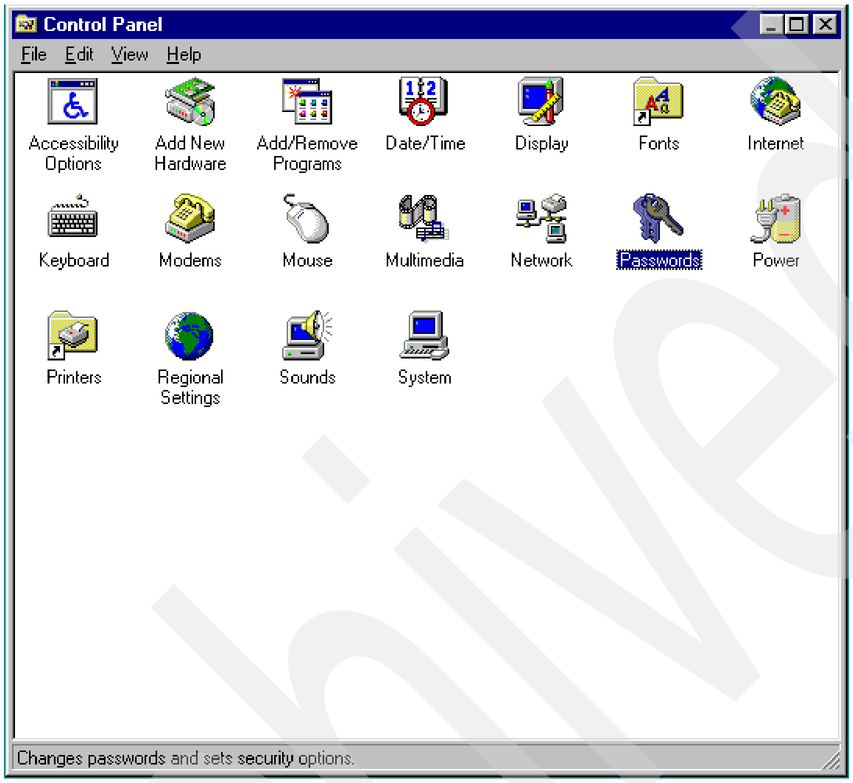
Figure A-7 Giving user name, password, and domain name

10. Enter a user name and password that is defined on the Linux system. For Domain enter `ITSODMN` for the domain name and then click the **OK** button.

Windows 95: Change the password

Perform the following steps:

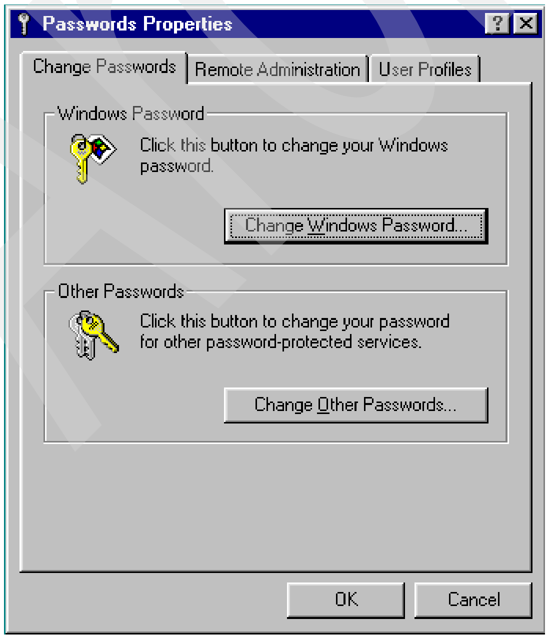
1. Open My Computer.
2. Open Control Panel as illustrated in Figure A-8.



Changes passwords and sets security options.

Figure A-8 Opening control panel

3. In the Control Panel, double click **Passwords** to bring a screen as shown in Figure A-9.

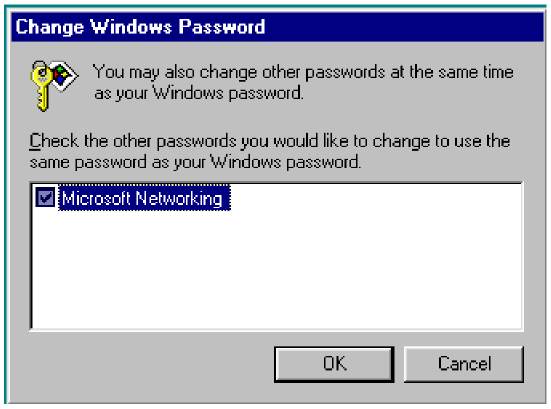


Click this button to change your Windows password.

Click this button to change your password for other password-protected services.

Figure A-9 Changing Windows password (1-2)

4. Click the **Change Windows Password** button to bring a screen as shown in Figure A-10.


Change Windows Password

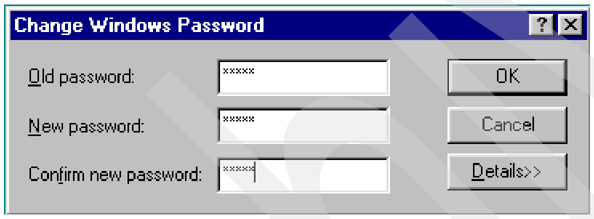
You may also change other passwords at the same time as your Windows password.

Check the other passwords you would like to change to use the same password as your Windows password.

☑ Microsoft Networking

Figure A-10 Changing Windows password (2-2)

5. On the Change Windows Password dialog box, select the radio box in front of **Microsoft Networking**. Click the **OK** button. On the Change Windows Password dialog box, provide the old password and the new password and then click the **OK** button as shown in Figure A-11.


Change Windows Password

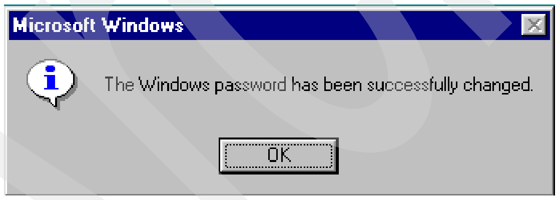
Old password: *****

New password: *****

Confirm new password: *****

Figure A-11 Changing Windows password

6. The system responds with a success dialog box after the passwords have been changed as shown in Figure A-12.


Microsoft Windows

The Windows password has been successfully changed.

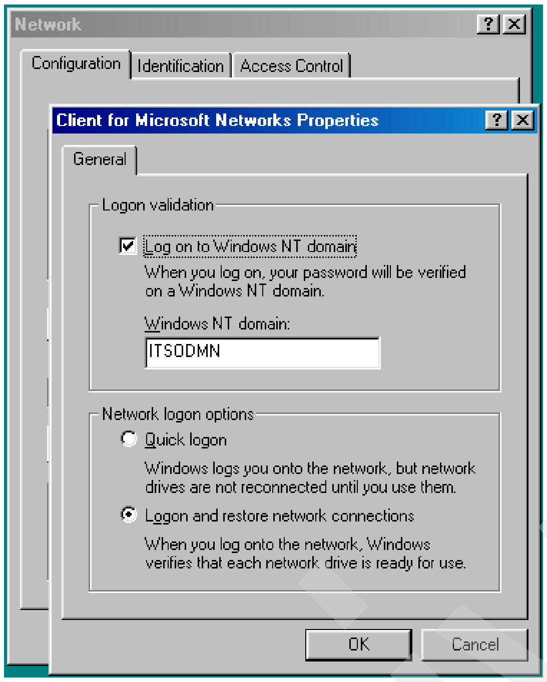
Figure A-12 Windows password change confirmation screen

Windows Workstation 98: Joining the domain

Perform the following steps:

1. Right-click Network Neighborhood and select **Properties**.
2. Click **Client for Microsoft Networks** and then click the **Properties** button.
3. Select the radio-box in front of **Log on to Windows NT domain**.

4. Specify `ITSODMN` in the Windows NT domain field as shown in Figure A-13.



Logon validation

[✓] Log on to Windows NT domain

When you log on, your password will be verified on a Windows NT domain.

Windows NT domain: ITSODMN

Network logon options

( ) Quick logon

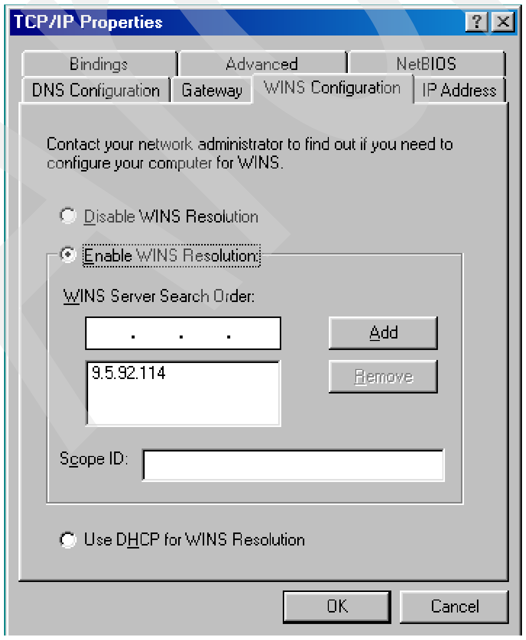
Windows logs you onto the network, but network drives are not reconnected until you use them.

(●) Logon and restore network connections

When you log onto the network, Windows verifies that each network drive is ready for use.

Figure A-13 Client for Microsoft network properties

5. Click the **OK** button.
6. Back on the Network dialog, click **TCP/IP** and then click the **Properties** button.
7. Click the **WINS Configuration** tab.
8. Select the radio button in front of **Enable WINS Resolution**.
9. Enter the IP address of the Linux server.
10. Click the **Add** button as shown in Figure A-14.



Contact your network administrator to find out if you need to configure your computer for WINS.

( ) Disable WINS Resolution

(●) Enable WINS Resolution:

WINS Server Search Order:

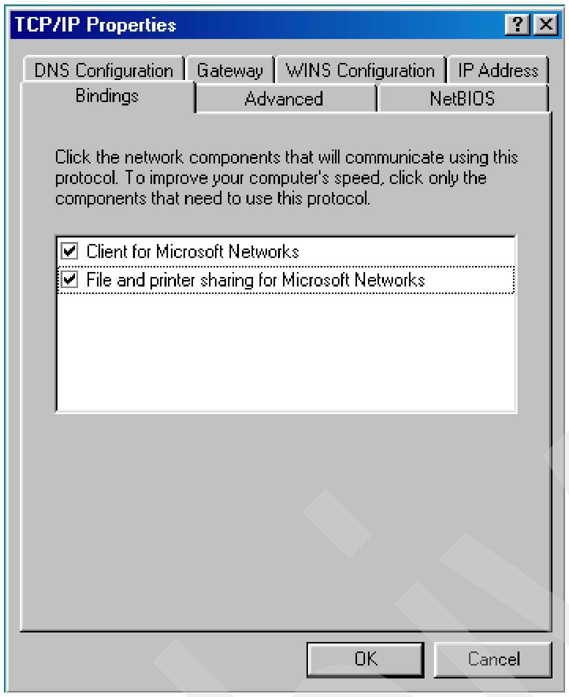
9.5.92.114

Scope ID:

( ) Use DHCP for WINS Resolution

Figure A-14 Enabling WINS resolution

11. Click the **Bindings** tab
12. Ensure that the **Client for Microsoft Networks** item is checked as shown in Figure A-15. Click the OK button.

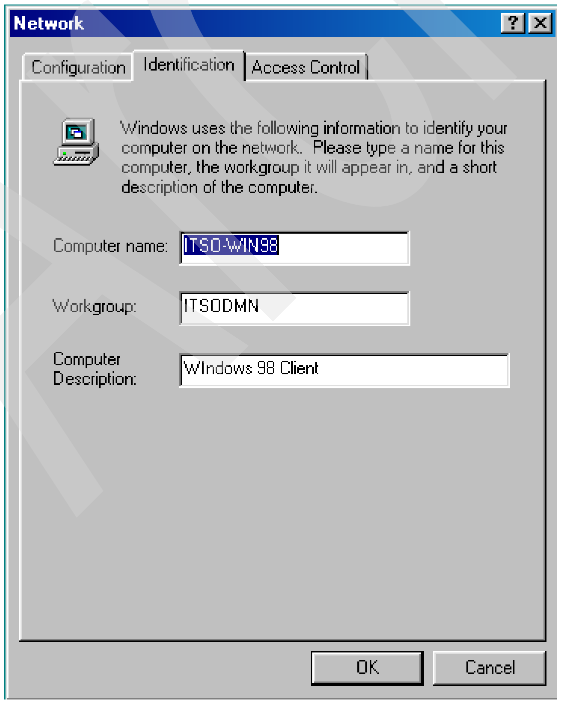


Click the network components that will communicate using this protocol. To improve your computer's speed, click only the components that need to use this protocol.

☑ Client for Microsoft Networks
☑ File and printer sharing for Microsoft Networks

Figure A-15 Setting Transmission Control Protocol/Internet Protocol (TCP/IP) properties for network

13. Back on the Network dialog, click the **Identification** tab.
14. Ensure that the Computer name is set correctly and that the Workgroup reflects the name of the domain, `ITSODMN` as shown in Figure A-16.



Windows uses the following information to identify your computer on the network. Please type a name for this computer, the workgroup it will appear in, and a short description of the computer.

Computer name: ITSO-WIN98

Workgroup: ITSODMN

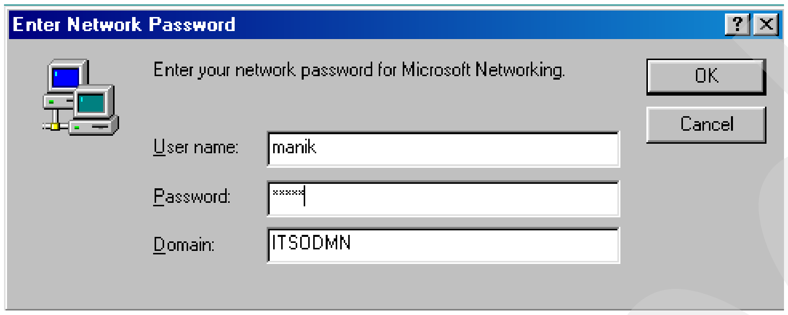
Computer Description: WIndows 98 Client

Figure A-16 Giving a domain name

15. Click the **OK** button.

16. Restart the client.

17. After the client reboots, the login box must reflect the name of the domain as shown in Figure A-17.


Enter Network Password

Enter your network password for Microsoft Networking.

User name: manik

Password: *****

Domain: ITSODMN

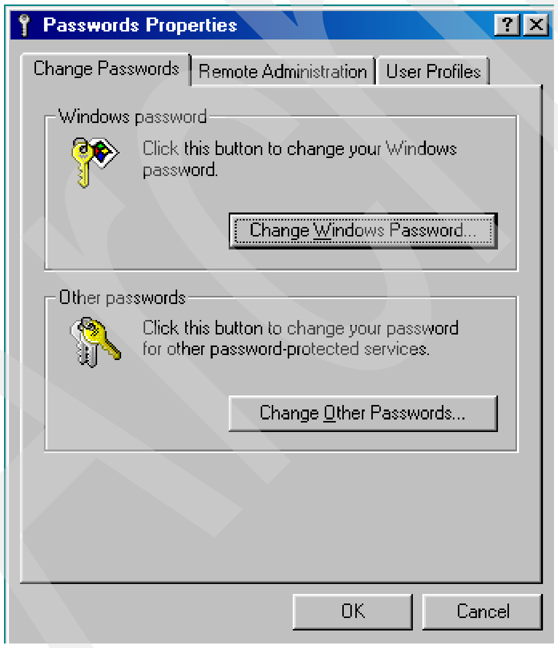
Figure A-17 Changing user name, password, and domain name

18. Enter a valid user name and password and click the **OK** button.

Windows 98: Change password

Perform the following steps:

1. Open My Computer.
2. Open Control Panel.
3. Double-click Passwords as shown in Figure A-18.


Passwords Properties

Windows password

Click this button to change your Windows password.

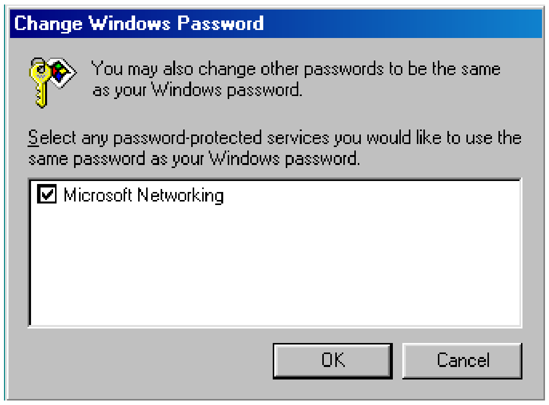
Other passwords

Click this button to change your password for other password-protected services.

Figure A-18 Changing Windows password (1-3)

4. Click the **Change Windows Password** button.

5. On the Change Windows Password dialog, ensure that **Microsoft Networking** is selected as shown in Figure A-19. Click the **OK** button.



Change Windows Password

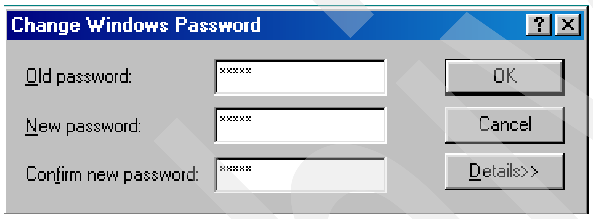
You may also change other passwords to be the same as your Windows password.

Select any password-protected services you would like to use the same password as your Windows password.

☑ Microsoft Networking

Figure A-19 Changing Windows password (2-3)

6. On the Change Windows Password, enter the old and new passwords as shown in Figure A-20. Click the **OK** button.



Change Windows Password

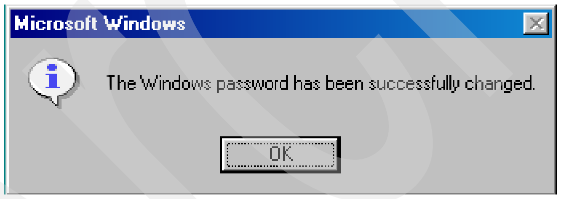
Old password: *****

New password: *****

Confirm new password: *****

Figure A-20 Changing Windows password (3-3)

7. The system responds with a success message as shown in Figure A-21. At this point all of the passwords have been changed.



Microsoft Windows

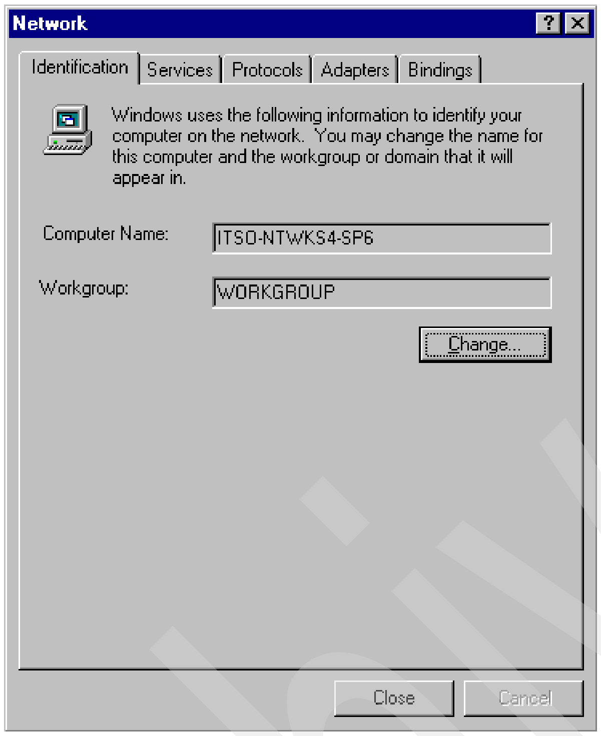
The Windows password has been successfully changed.

Figure A-21 Windows password change confirmation

Windows NT Workstation: Joining the domain

Perform the following steps:

1. Right-click Network Neighborhood and select **Properties** as shown in Figure A-22.



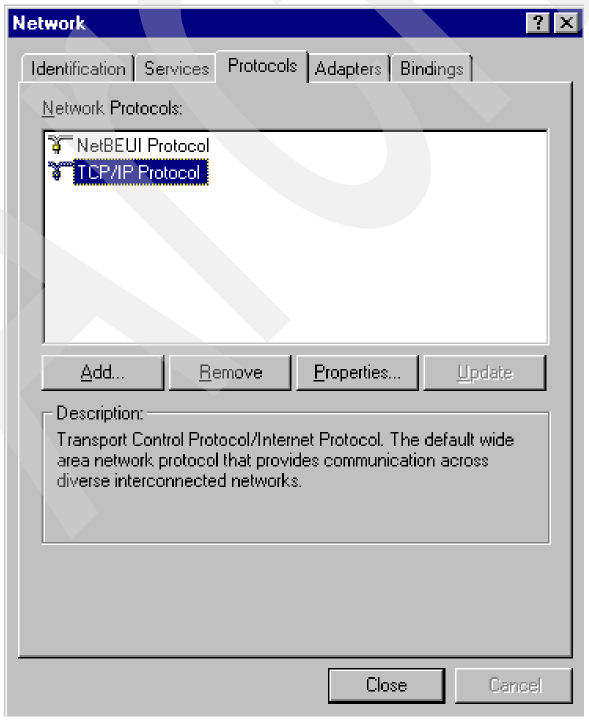
Windows uses the following information to identify your computer on the network. You may change the name for this computer and the workgroup or domain that it will appear in.

Computer Name: ITSO-NTWKS4-SP6

Workgroup: WORKGROUP

Figure A-22 NT network neighborhood properties

2. Click the **Protocols** tab as shown in Figure A-23.



Network Protocols:

NetBEUI Protocol

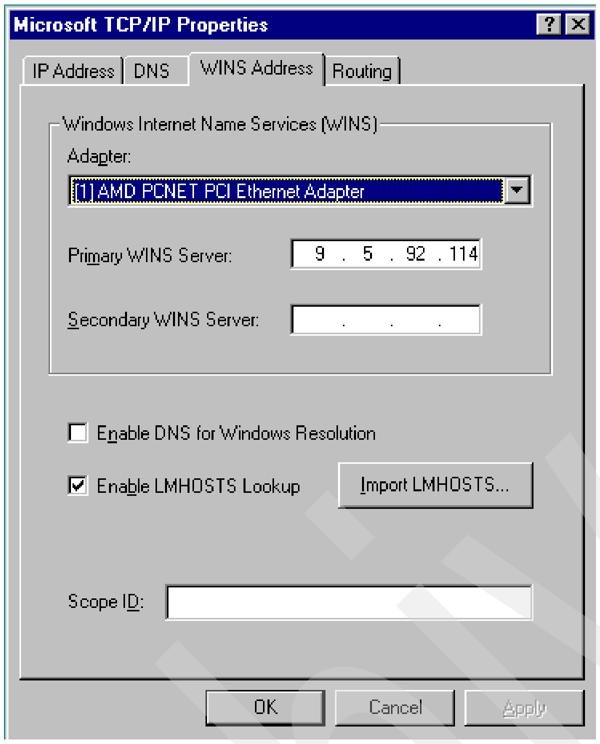
TCP/IP Protocol

Description:

Transport Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

Figure A-23 TCP/IP protocols tab

3. Click **TCP/IP Protocol** and then click the **Properties** button.
4. Click the **WINS Address** tab and then specify the IP address of the Linux partition for the Primary WINS Server as shown in Figure A-24. Click the **OK** button.



Windows Internet Name Services (WINS)

Adapter: [1] AMD PCNET PCI Ethernet Adapter

Primary WINS Server: 9 . 5 . 92 . 114

Secondary WINS Server: . . .

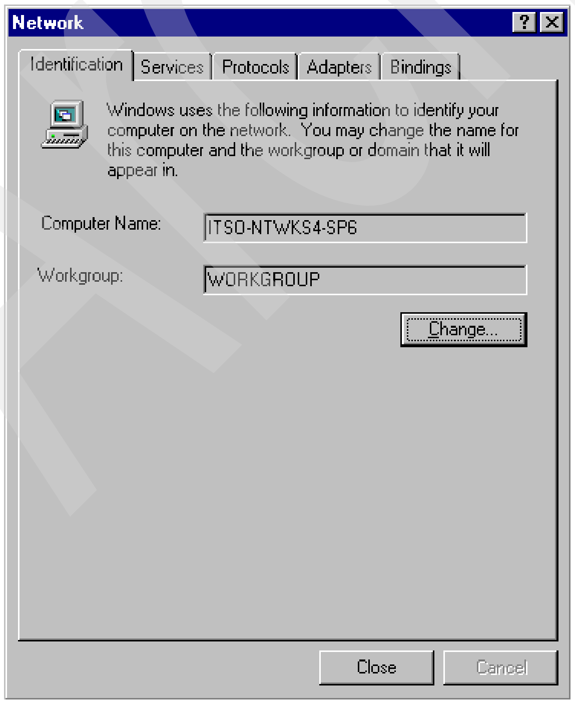
☐ Enable DNS for Windows Resolution

☑ Enable LMHOSTS Lookup

Scope ID:

Figure A-24 Giving primary WINS server address

5. Back on the Network dialog box, click the **Identification** tab as shown in Figure A-25. Click the **Change** button.



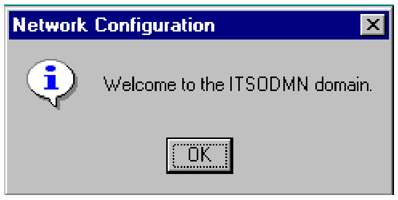
Windows uses the following information to identify your computer on the network. You may change the name for this computer and the workgroup or domain that it will appear in.

Computer Name: ITSO-NTWKS4-SP6

Workgroup: WORKGROUP

Figure A-25 Giving network identification (1-2)

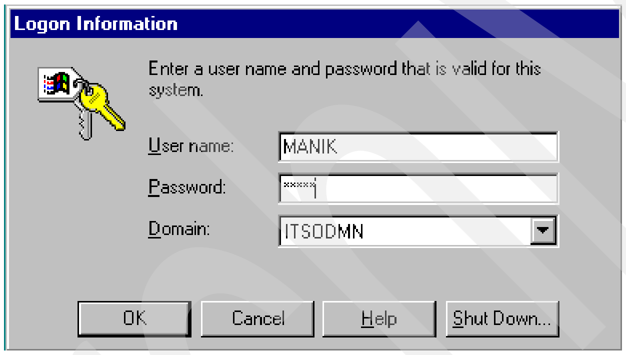
6. Click the **Domain** radio button and then specify the domain name, `ITSODMN`.
7. Click the **Create a Computer Account in the Domain** and specify a user name and password that can create domain accounts (in this case root and the root password). Click the **OK** button.
8. At this point the client requests a system account to be created on the Linux server and then the system provides a success message as shown in Figure A-26.


Network Configuration

Welcome to the ITSODMN domain.

Figure A-26 Domain name confirmation

9. Back on the Network dialog, click the **OK** button.
10. Restart the client.
11. On the logon box, ensure that the Domain reflects the domain that was just joined (ITSODMN), then provide a valid user name and password as shown in Figure A-27.


Logon Information

Enter a user name and password that is valid for this system.

User name: MANIK

Password: *****

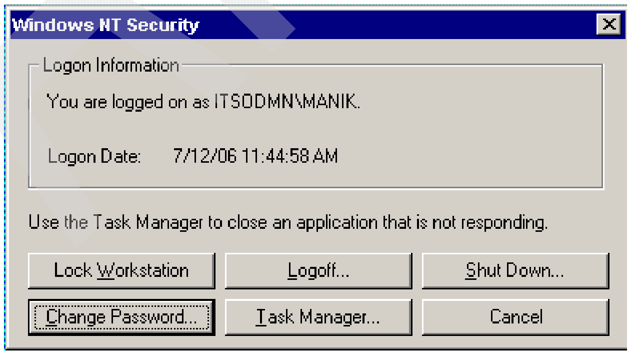
Domain: ITSODMN

Figure A-27 Validating user on the domain

Windows NT Workstation: Change password

Perform the following steps:

1. While logged in to the client, press <CTRL><ALT><DELETE> to bring up the Windows NT Security dialog box as shown in Figure A-28.


Windows NT Security

Logon Information

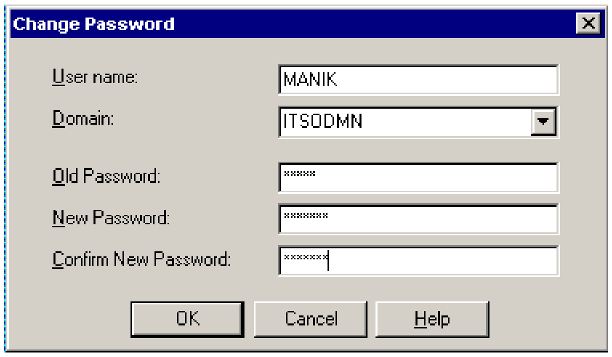
You are logged on as ITSODMN\MANIK.

Logon Date: 7/12/06 11:44:58 AM

Use the Task Manager to close an application that is not responding.

Figure A-28 Windows NT security dialog box

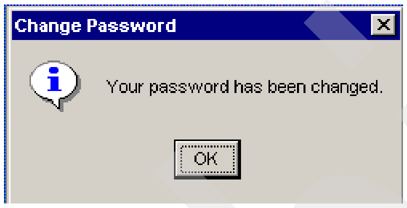
2. Click the **Change Password** button.
3. Provide the old and new passwords as shown in Figure A-29. Click the **OK** button.


Change Password

User name: MANIK
Domain: ITSODMN
Old Password: *****
New Password: *******
Confirm New Password: *******

Figure A-29 Changing password

4. The password is changed in i5/OS and also in Linux and the SAMBA file server as shown in Figure A-30.


Change Password

Your password has been changed.

Figure A-30 Confirming password change

Windows 2000 Professional: Joining the domain

Perform the following steps:

1. Right-click My Network Places and select **Properties**.
2. In the Network and Dial-up Connections window, right-click your network adapter and select **Properties**.
3. On the Local Area Connection Properties window, click **Internet Protocol (TCP/IP)** and click the **Properties** button.
4. On the Internet Protocol (TCP/IP) Properties window, click the **Advanced** button.

5. On the Advanced TCP/IP Settings window, click the WINS tab as shown in Figure A-31. Click the **Add** button.

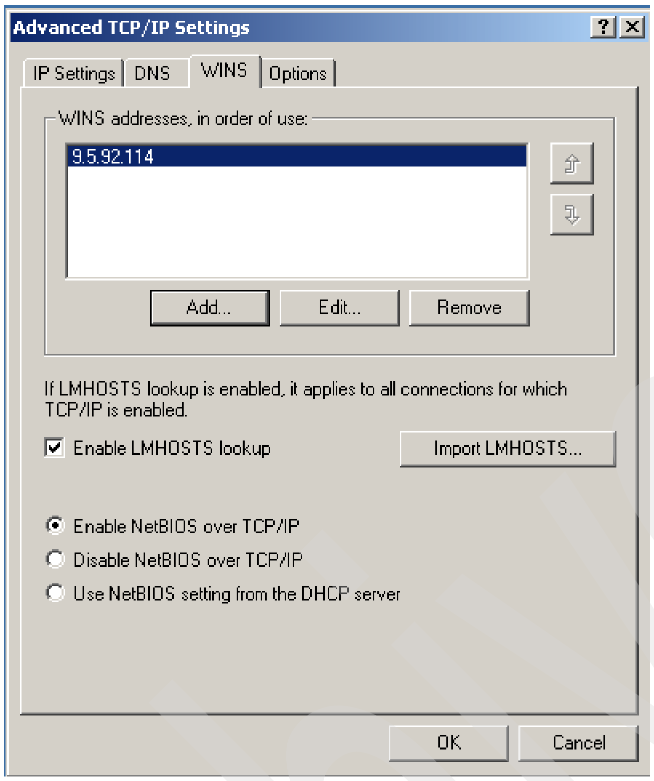
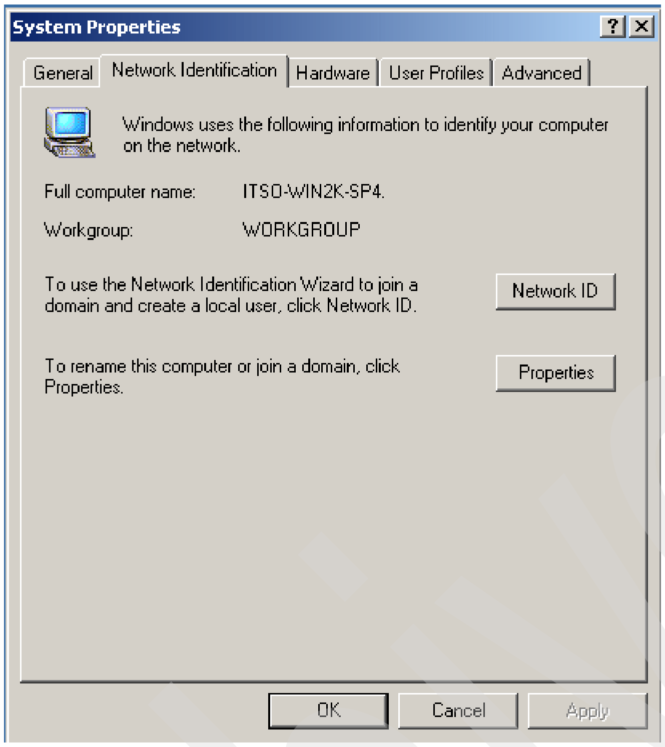

Advanced TCP/IP Settings
IP Settings
DNS
WINS
Options
WINS addresses, in order of use:
9.5.92.114
Add...
Edit...
Remove
If LMHOSTS lookup is enabled, it applies to all connections for which TCP/IP is enabled.
Enable LMHOSTS lookup
Import LMHOSTS...
Enable NetBIOS over TCP/IP
Disable NetBIOS over TCP/IP
Use NetBIOS setting from the DHCP server
OK
Cancel

Figure A-31 Giving WINS address

6. On the TCP/IP WINS SERVER dialog, specify the IP address of the Linux server and then click the **Add** button.
7. Back on the Advanced TCP/IP Settings window click the **OK** button.
8. Back on the Internet Protocol (TCP/IP) Properties window, click the **OK** button.
9. Back on the Local Area Connection Properties window click the **OK** button.
10. Right-click My Computer and select **Properties**.

11. On the System Properties window, click the **Network Identification** tab as shown in Figure A-32.


System Properties

General | Network Identification | Hardware | User Profiles | Advanced

Windows uses the following information to identify your computer on the network.

Full computer name: ITSO-WIN2K-SP4.

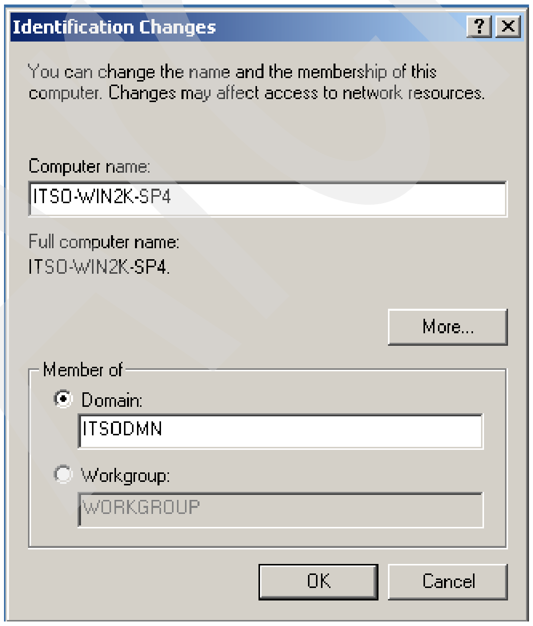
Workgroup: WORKGROUP

To use the Network Identification Wizard to join a domain and create a local user, click Network ID.

To rename this computer or join a domain, click Properties.

Figure A-32 Giving network identification

12. Click the **Properties** button to open a screen as shown in Figure A-33. Click the radio button in front of **Domain** and then provide the domain name (`ITSODMN`). Click the **OK** button.


Identification Changes

You can change the name and the membership of this computer. Changes may affect access to network resources.

Computer name: ITSO-WIN2K-SP4

Full computer name:
ITSO-WIN2K-SP4.

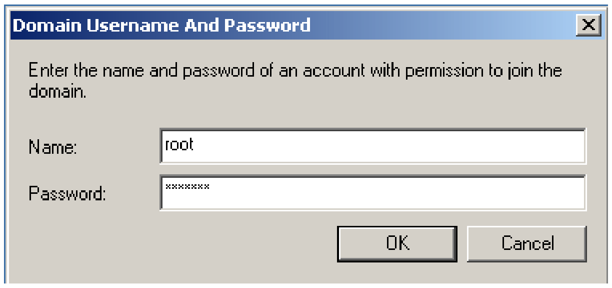
Member of

[✓ Domain:] ITSODMN

Workgroup: WORKGROUP

Figure A-33 Giving a domain name

13..The system prompts for a user-name and password that has the authority to join the client to the domain as shown in Figure A-34.



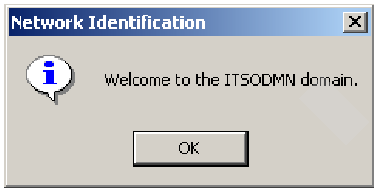
Enter the name and password of an account with permission to join the domain.

Name: root

Password: *******

Figure A-34 Testing domain setup

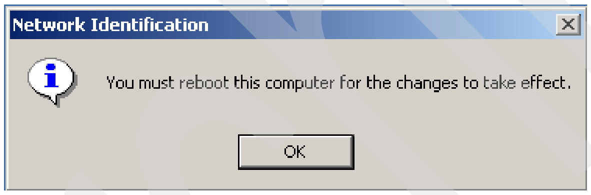
14.:Provide “root” for the name and the root password for password and then click the **OK** button. At this point the system account is added to the domain as shown in Figure A-35.



Welcome to the ITSODMN domain.

Figure A-35 Joining the domain

15. Restart the client as shown in Figure A-36.



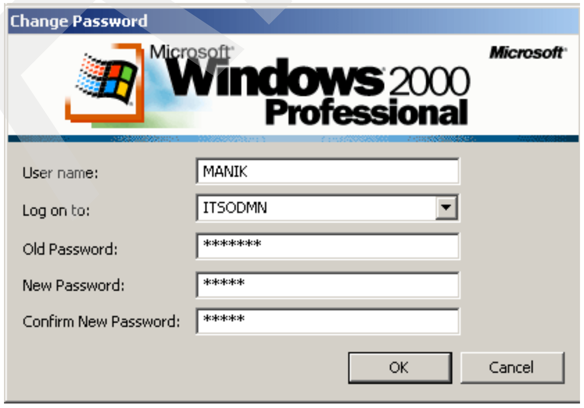
You must reboot this computer for the changes to take effect.

Figure A-36 Restart message

Windows 2000 Professional: Change the domain password

Perform the following steps:

1. While logged into the client, press <CTRL><ALT><DELETE>.
2. On the Windows Security window, click the <*Change Password*> button (Figure A-37).



User name: MANIK

Log on to: ITSODMN

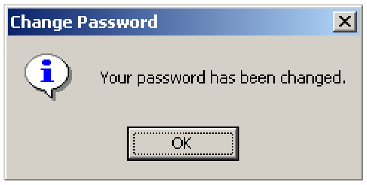
Old Password: ********

New Password: *****

Confirm New Password: *****

Figure A-37 Changing the password

3. On the Change Password dialog, provide the old and new password.
4. At this point the password is changed on the domain server, Linux, and i5/OS as shown in Figure A-38.


Change Password

Your password has been changed.

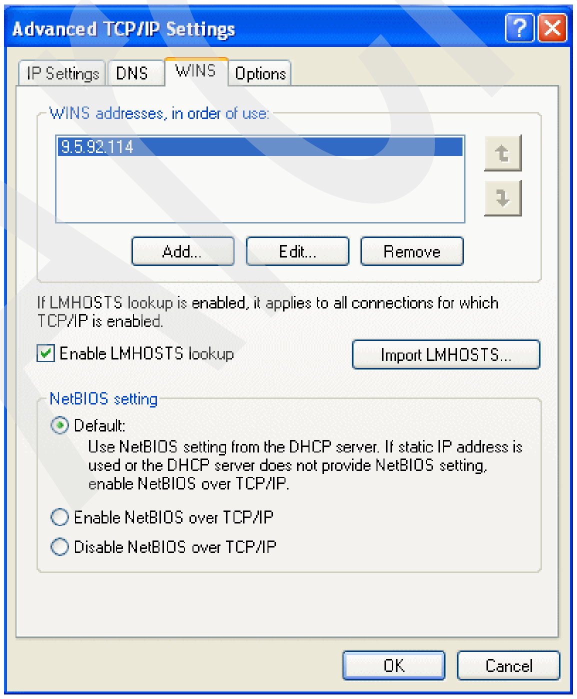
Figure A-38 Changing the domain password validation

Windows XP Professional: Joining the domain

Perform the following steps:

Note: The following steps are for Windows XP Professional only. These steps do *not* work for Windows XP Home. Windows XP Home is unable to participate in Windows-style domains.

1. Open My Computer.
2. Right-click My Network Places and select **Properties**.
3. In the Network Connections window, right-click your network adapter and select **Properties**.
4. On the Local Area Connection Properties window, click **Internet Protocol (TCP/IP)** and then click the **Properties** button.
5. On the Internet Protocol (TCP/IP) Properties window, click the **Advanced** button.
6. On the Advanced TCP/IP Settings window, click the **WINS** tab as shown in Figure A-39. Click the **Add** button.


Advanced TCP/IP Settings

WINS addresses, in order of use:

9.5.92.114

If LMHOSTS lookup is enabled, it applies to all connections for which TCP/IP is enabled.

[✓] Enable LMHOSTS lookup

NetBIOS setting

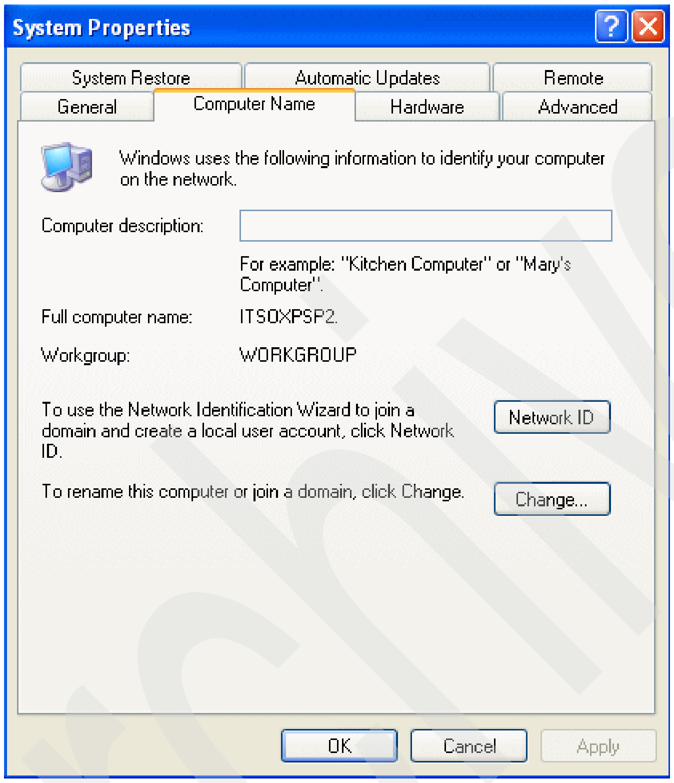
[✓ Default:]
Use NetBIOS setting from the DHCP server. If static IP address is used or the DHCP server does not provide NetBIOS setting, enable NetBIOS over TCP/IP.

Enable NetBIOS over TCP/IP

Disable NetBIOS over TCP/IP

Figure A-39 Giving WINS address

7. In the “TCP/IP WINS Server” dialog enter the IP address of the Linux server and then click the **Add** button.
8. Back on the Advanced TCP/IP Settings window, click the **OK** button.
9. Back on the Internet Protocol (TCP/IP) Properties window, click the **OK** button.
10. Back on the Local Area Connection Properties window, click the **Close** button.
11. Right-click My Computer and select **Properties**.
12. On the System Properties window, click the **Computer Name** tab as shown in Figure A-40.



Windows uses the following information to identify your computer on the network.

Computer description:

For example: "Kitchen Computer" or "Mary's Computer".

Full computer name: ITSOXPSP2.

Workgroup: WORKGROUP

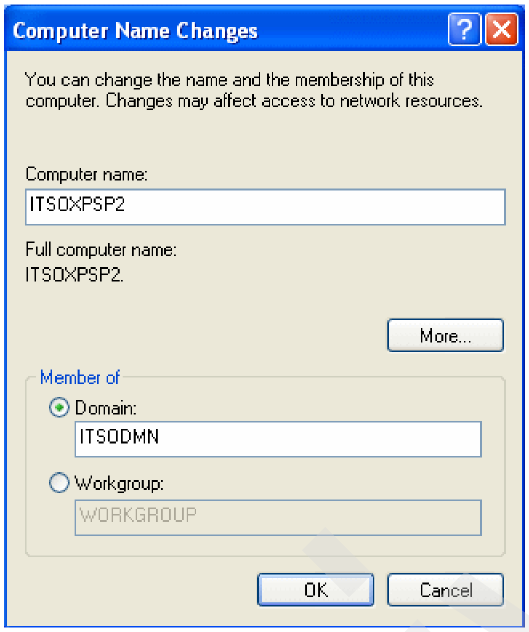
To use the Network Identification Wizard to join a domain and create a local user account, click Network ID.

To rename this computer or join a domain, click Change.

Figure A-40 Working on computer name of system properties

13. Click the **Change** button.

14. Click the radio button in front of Domain and then provide the domain name as shown in Figure A-41. Click the OK button.



Computer Name Changes

You can change the name and the membership of this computer. Changes may affect access to network resources.

Computer name:
ITSOXPSP2

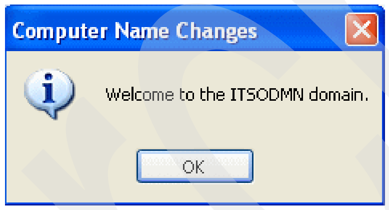
Full computer name:
ITSOXPSP2.

Member of
Domain: [✓]
ITSODMN
Workgroup:
WORKGROUP

Figure A-41 Giving a domain name

15. When prompted, provide a user name and password that has the authority to add clients to the domain (in this case 'root' and the root password).

16. The system is added to the domain as shown in Figure A-42. Restart the client.



Computer Name Changes

Welcome to the ITSODMN domain.

Figure A-42 Scenario validation

Windows XP Professional: Change password

The procedure for changing the password on a Windows XP Professional system is the same as that documented for Windows 2000 Professional. Refer to "Windows 2000 Professional: Change the domain password" on page 135.

B

Linux system message logging

This appendix describes a mechanism for logging Linux system messages in IBM i5/OS using syslog on Linux and a syslogd server program on IBM i5/OS.

All Linux system messages are logged to a file and those that are selected go as messages to QSYSOPR. This allows you to effectively monitor multiple Linux systems from an IBM i5/OS partition.

Every Linux system can send messages whether it is installed on a standalone PC, a logical partition (LPAR), or on an IXS/IXA. Topics in this appendix include:

- Introduction to syslogd
- Linux configuration
- IBM i5/OS configuration

Introduction to syslogd

Syslogd is a Linux daemon responsible for logging system messages on UNIX and Linux systems. Processing of the syslogd daemon is controlled by the */etc/syslog.conf* file and you can find detailed information about this configuration file using the following Linux command:

```
linux:~> man syslog.conf
```

The important part of the man page is the section talking about remote logging. Built in to the syslogd program is the ability to send the system messages to a remote system where it is logged. At the remote system you get a time stamp, source originating system, and the original message as follows:

```
NOTE: SLES10 also comes with syslog-ng, which is a "new-generation" syslogd
(replacement) for Unix and Unix-like systems. It tries to fill the gaps in the
original syslogd. Processing of the syslogd daemon is controlled by the
/etc/syslog-ng/syslog-ng.conf file and detailed information about this
configuration file can be found using the following Linux command:

linux:~> man syslog-ng.conf
```

We are using syslogd daemon in this book.

Linux system configuration

The configuration of syslogd is controlled by the /etc/syslog.conf file on your Linux server. There is extensive help for this configuration in the man pages. The configuration of this file allows an administrator to log certain messages (those coming from certain applications, those of a certain severity, and so on) to files or to reroute these messages to another system running the syslogd job. Typically the remote system is another Linux system, but with the i5/OS syslogd program these messages can easily be rerouted to an IBM i5/OS.

Example B-1 is a copy of the /etc/syslog.conf file from one of the example Linux systems.

Example: B-1 File /etc/syslog.conf

```
# /etc/syslog.conf - Configuration file for syslogd(8)
#
# For info about the format of this file, see "man syslog.conf".
#

#
#
#
*.emerg   *

# enable this, if you want that root is informed
# immediately, e.g. of logins
#*.alert  root


#
# all email-messages in one file
#
#mail.*  -/var/log/mail
```

```
#
# all news-messages
#
# these files are rotated and examined by "news.daily"
#news.crit-/var/log/news/news.crit
#news.err-/var/log/news/news.err
news.notice-/var/log/news/news.notice
# enable this, if you want to keep all news messages
# in one file
#news.*  -/var/log/news.all

#
# Warnings in one file
#
#*.=warn;*.=err-/var/log/warn
#*.crit   /var/log/warn

#
# save the rest in one file
#
#*.*;mail.none;news.none-/var/log/messages

#
# enable this, if you want to keep all messages
# in one file
#*.*     -/var/log/allmessages
*.*      @SYSTEM

#
# Some foreign boot scripts require local7
#
#local0,local1.*-/var/log/localmessages
#local2,local3.*-/var/log/localmessages
#local4,local5.*-/var/log/localmessages
#local6,local7.*-/var/log/localmessages

#kern.*-/var/log/firewall
```

The changes made to this file are the following:

```
#*.*        -/var/log/allmessages
*.*         @SYSTEM
```

The first line is the original line in the file which is commented out. The second line is the one added to reroute all messages to the system designated by host name SYSTEM. The “@” character before the host name indicates that the messages are rerouted to a remote system. SYSTEM can be the IP Address or the DNS name of your IBM i5/OS. If you use a DNS name then you must also add the DNS name to the hosts file to prevent connection problems.

IBM i5/OS system configuration

On IBM i5/OS, the only configuration required is to create a program that acts as a remote syslogd job and receives incoming messages. The first step is to determine how syslogd was sending messages to a remote system. To do this, look through the */etc/services* file and find an entry for syslog. This entry looks like the following:

```
syslog          514/udp
```

This told us that syslog uses IP port 514 and a protocol of User Datagram Protocol (UDP). If you want to view the /etc/services file, we recommend you to pipe the output to a search function such as **grep**, because the /etc/services file is quite large. We used the following command to limit the results to only those lines that contained the word “syslog”.

```
linux:~> cat /etc/services | grep syslog
```

After you know how syslogd is sending remote messages, it is a fairly simple step to take the socket server code from 8.2, “Sockets programming example” on page 97 and modify it to receive syslogd messages. Your initial step can be just to change the port number so that you can see what the messages look like. Based on what the messages look like, you can then further refine the server code until you get the exact behavior that you desire.

IBM i5/OS syslogd server program

The syslogd server program differs from the socket server example because the syslogd program is expected to only run on IBM i5/OS. According to this information, all the #ifdef statements are removed to clarify the code. Also, you have to get additional data such as the current time and Transmission Control Protocol/Internet Protocol (TCP/IP) host information. Finally, you have to open a file to store the incoming messages and send out selected messages to QSYSOPR.

The program flow of the syslogd program, as shown in Example B-2, is as follows:

1. Open the log file where all the messages are stored. If the file does not exist it is created. Otherwise, the file is opened in append mode. This ensures that you do not lose previous messages that were logged.
2. Now open the socket descriptor.
3. Bind port 514 to the new socket descriptor. This is the port that syslogd on Linux uses. If it is changed on the Linux side then the IBM i5/OS server program also requires to be changed.
4. Create a conversion table. All incoming data is coming from Linux machines and therefore has to be converted from American Standard Code for Information Interchange (ASCII) to Extended Binary Coded Decimal Interchange Code (EBCDIC).
5. Start the main body of the program by looping until told to stop. Inside the loop the following steps take place:
   a. Receive incoming requests from the socket.
   b. Convert the data to EBCDIC.
   c. Remove the newline character from the end of the message.
   d. Get the IP system information and add it to the front of the message.
   e. Check if this message must go to QSYSOPR and send it there if it must.
   f. Get the current time stamp.
   g. Write the message to the log file.
   h. Finally, flush the write buffer every tenth write.
6. Before exiting the program close the conversion table and socket descriptor.

To create the syslogd program run the following commands in this order:

1. `CRTCMOD MODULE(ITSOLIB/SYSLOGD) SRCSTMF('/itsodir/src/syslogd.c') SYSIFCOPT(*IFSIO) TGTCCSID(37)`
2. `CRTPGM PGM(ITSOLIB/SYSLOGD) BNDSRVPGM(QSYS/QTQICONV)`

These commands create a program called SYSLOGD in library ITSOLIB and they assume that the source for the program is in file /itsodir/src/syslogd.c in Integrated File System (IFS). You have to adjust the object names to fit your environment.

The next step required before you can run the program is to create the IFS directory structure. The program creates the file if it does not exist, but it cannot create the full directory structure if that is required. The directory you require is */var/log* and you can create it using the **`mkdir`** command on i5/OS. You also require a message file with a message description that is used for the QSYSOPR messages. In the program we depict, the message file is ITSOLIB/LINUX with a message ID of LNX0001. You can change this to meet your requirements as long as the program is also updated. These are the steps:

1. `MKDIR DIR('/var/log')`
2. `CRTMSGF MSGF(ITSOLIB/LINUX) TEXT(Message file for SYSLOGD)`
3. `ADDMSGD MSGID(LNX0001) MSGF(ITSOLIB/LINUX) MSG('&1') SECLVL('&2') FMT((*CHAR 80) (*CHAR 512))`

After the program, directory, and message file are created, you can submit the syslogd program to run in batch using the following command:

```
SBMJOB CMD(CALL SYSLOGD/SYSLOGD)
```

A batch job now runs and monitors the 514 port for incoming messages. All incoming messages are logged to file /var/log/messages and any message containing “kernel:” is also sent to the QSYSOPR message queue.

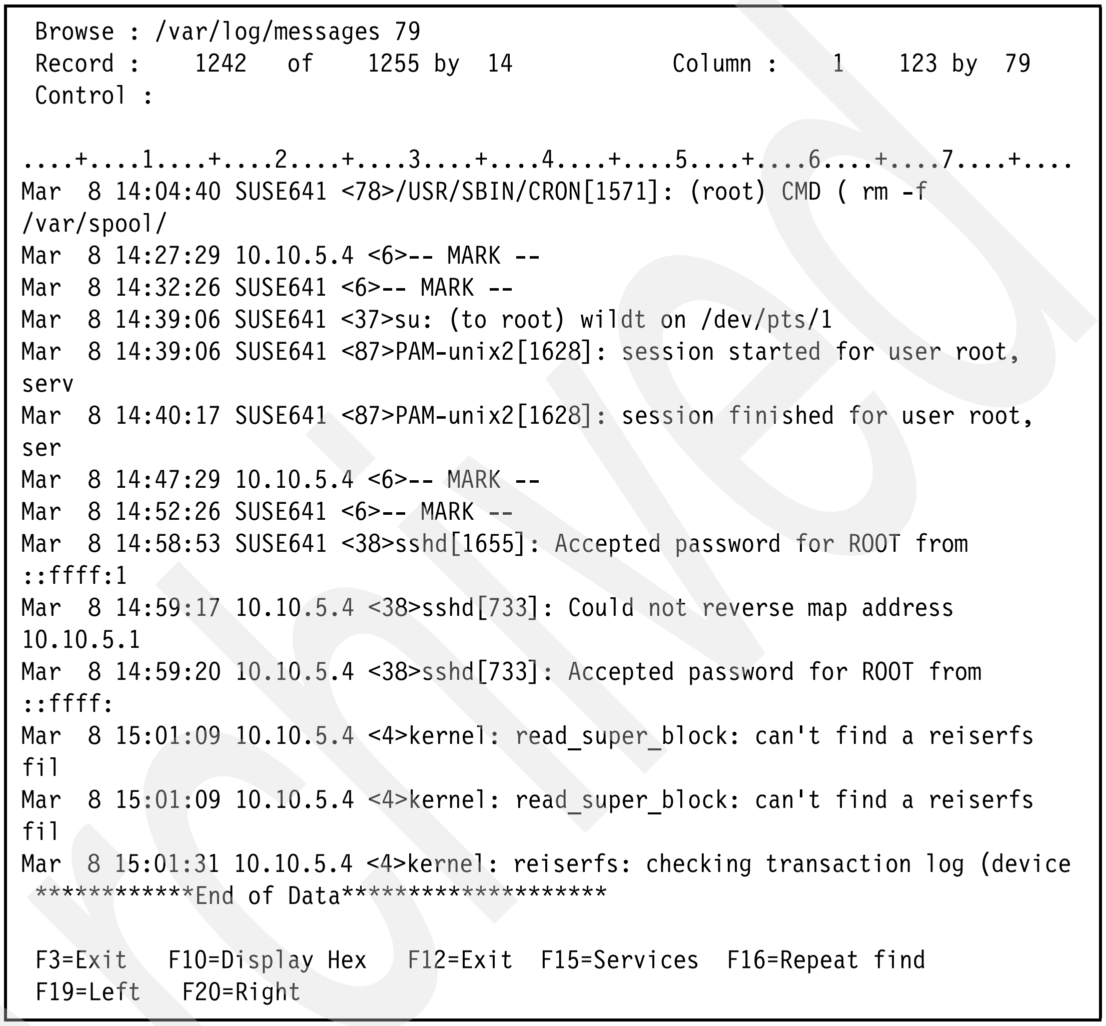
Example B-2 is the actual syslogd code running on IBM i5/OS.

Example: B-2 Syslogd source code syslogd.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

#include <errno.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#include <iconv.h>
#include <QMHSNDM.H>

#define SERVER_PORT 514
#define BACK_LOG 20
#define BUFFER_SIZE 512

typedef struct {
  char name[8];
  char ccsid[5];
  char conversion[3];
  char substitute[1];
  char shift[1];
  char input[1];
  char option[1];
  char RESERVED[12];
} fromCode_t;

typedef struct {
  char name[8];
  char ccsid[5];
  char RESERVED[19];
} toCode_t;

FILE * openFile(void);
iconv_t createConversionTable(void);
void getDotted(char *dottedName, long ipAddress);
void logMessage(char *message);

int main(int argc, char *argv[])
{
  int rc,
      bufferLength,
      recordCount = 0;
  char buffer[BUFFER_SIZE],
       converted[BUFFER_SIZE],
       message[BUFFER_SIZE],
       *inBuffer,
       *convertBuffer,
       dotted[15];
  FILE *logFile;
  time_t currentTime;

  /*! socket() variables */
  int descriptor,
      addressFamily,
      type,
      protocol;

  /*! bind() variables */
  struct sockaddr_in localAddress;
  int addressLength;

  /*! recvfrom() variables */
  int flags,
      clientLength;
  struct sockaddr_in clientAddress;

  /*! gethostbyaddr() variables */
  struct hostent *system;

  /*! iconv() variable */
  size_t inBufferLength,
         convertBufferLength;
  iconv_t convertTable;
  /*! socket() */
  /*! bind() */
  /*! recvfrom() */
  /*! close() */

  /*! open the file */
  logFile = openFile();

  /*! open the socket descriptor */
  addressFamily = AF_INET;
  type = SOCK_DGRAM;
  protocol = IPPROTO_IP;
  if ((descriptor = socket(addressFamily, type, protocol)) < 0) {
    printf("Error getting socket descriptor\n");
    printf("errno is %i\n", errno);
    exit(-1);
  }

  /*! bind to the socket */
  memset(&localAddress, 0x00, sizeof(localAddress));
  localAddress.sin_family = AF_INET;
  localAddress.sin_port   = htons(SERVER_PORT);
  localAddress.sin_addr.s_addr = htonl(INADDR_ANY);
  addressLength = sizeof(localAddress);
  if ((rc = bind(descriptor, (struct sockaddr *)&localAddress, addressLength)) <
0) {
    printf("Error binding to the socket\n");
    printf("errno is %i\n", errno);
    close(descriptor);
    exit(-1);
  }

  /*! setup the conversion table ONLY ON OS/400 SERVER*/
  convertTable = createConversionTable();

  do {
    /*! receive any incoming client requests */
    memset(buffer, 0x00, sizeof(buffer));
    bufferLength = sizeof(buffer);
    flags = 0;
    clientLength = sizeof(clientAddress);
    rc = recvfrom(descriptor, buffer, bufferLength, flags,
                  (struct sockaddr *)&clientAddress, &clientLength);
    if (rc < 0) {
      printf("Error receiving incoming client reqeusts\n");
      printf("errno is %i\n", errno);
      close(descriptor);
      exit(-1);
    }

    /*! convert the data */
    convertBufferLength = inBufferLength = strlen(buffer);
    inBuffer = buffer;
    convertBuffer = converted;
    memset(converted, 0x00, sizeof(converted));
    iconv(convertTable, &inBuffer, &inBufferLength, &convertBuffer,
&convertBufferLength);

    /*! remove the newline from the end of the string */
    converted[strlen(converted) - 1] = '\0';

    /*! get the system information and add to the message*/
    system = gethostbyaddr((char *)&clientAddress.sin_addr, 4, AF_INET);
    if (NULL == system) {
      getDotted(dotted, clientAddress.sin_addr.s_addr);
      sprintf(message, "%.15s %.512s", dotted, converted);
    } else {
      sprintf(message, "%s %.512s", system->h_name, converted);
    }

    /*! check if this message needs to go to QSYSOPR */
    if (NULL != strstr(converted, "kernel:")) {
      logMessage(message);
    }

    /*! get the current time */
    currentTime = time(&currentTime);

    /*! output the data */
    fprintf(logFile, "%.15s %.512s\n", (ctime(&currentTime)) + 4, message);

    /*! flush the write every 10th write */
    ++recordCount;
    if (recordCount > 10) {
      recordCount = 0;
      fflush(logFile);
    }

    /*! stop when 'STOP' is received as the first 4 characters */
  } while (strncmp(converted, "STOP", 4) != 0);

  /*! close the conversion table */
  iconv_close(convertTable);

  /*! close the socket */
  close(descriptor);

  exit(0);
} /*! ...end of main() */

/*!
 * ********
 * openFile
 * ********
 *
 * Open the output log file.
 */
FILE *openFile()
{
  FILE *temp;
  temp = fopen("/var/log/messages", "a, ccsid=37");
  fclose(temp);
  temp = fopen("/var/log/messages", "a");
  return temp;
}

/*!
 * *********************
 * createConversionTable
 * *********************
 *
 * Creates the conversion table for the iconv() API.
 */
iconv_t createConversionTable()
{
  size_t inBufferLength,
         convertBufferLength;
  fromCode_t fromCCSID;
  toCode_t toCCSID;
  iconv_t convertTable;

  memcpy(toCCSID.name, "IBMCCSID", 8);
  memcpy(toCCSID.ccsid, "00037", 5);
  memset(toCCSID.RESERVED, 0x00, 19);
  memcpy(fromCCSID.name, "IBMCCSID", 8);
  memcpy(fromCCSID.ccsid, "01252", 5);
  memcpy(fromCCSID.conversion, "000", 3);
  memcpy(fromCCSID.substitute, "0", 1);
  memcpy(fromCCSID.shift, "0", 1);
  memcpy(fromCCSID.input, "0", 1);
  memcpy(fromCCSID.option, "0", 1);
  memset(fromCCSID.RESERVED, 0x00, 12);
  convertTable = iconv_open((char *)&toCCSID, (char *)&fromCCSID);
  if (convertTable.return_value < 0) {
    printf("Error creating conversion table\n");
    printf("errno is %i\n", errno);
    exit(-1);
  }
  return convertTable;
}

/*!
 * *********
 * getDotted
 * *********
 *
 * Gets a string of the dotted IP address.
 */
void getDotted(char *dottedName, long ipAddress)
{
  long address = ipAddress;
  int fourth, third, second, first;

  fourth = address % 256;
  address /= 256;

  third = address % 256;
  address /= 256;

  second = address % 256;
  address /= 256;

  first = address % 256;

  sprintf(dottedName, "%i.%i.%i.%i", first, second, third, fourth);
}


/*!
 * **********
 * logMessage
 * **********
 *
 * Log the message to the QSYSOPR message queue.
 */
void logMessage(char *message)
{
  char messageId[] = "LNX0001",
       messageFile[] = "LINUX     ITSOLIB   ",
       messageData[592],
       messageType[] = "*INFO     ",
       messageQueue[] = "*SYSOPR   ",
       replyQueue[] = "                    ",
       messageKey[4];
  int lengthOfMessage = sizeof(messageData),
      numberOfQueues = 1,
      errorCode = 0,
      firstLevelLength,
      secondLevelLength;

  /*! set the message data */
  firstLevelLength = (strlen(message) < 80) ? strlen(message) : 80;
  secondLevelLength = (strlen(message) < 512) ? strlen(message) : 512;
  memset(messageData, ' ', sizeof(messageData));
  memcpy(messageData, message, firstLevelLength);
  memcpy(messageData + 80, message, secondLevelLength);

  /*! send the message */
  QMHSNDM(messageId, messageFile, messageData, lengthOfMessage, messageType, 
messageQueue,
          numberOfQueues, replyQueue, messageKey, (void *)&errorCode);
}
```

Message log on IBM i5/OS

Figure B-1 shows the bottom of the /var/log/messages file on IBM i5/OS. As you can see from the sample, you are receiving messages from at least two Linux partitions. One partition is known by the IP host name SUSE641 while the other system is not known by any host name and therefore is tracked by its IP address.

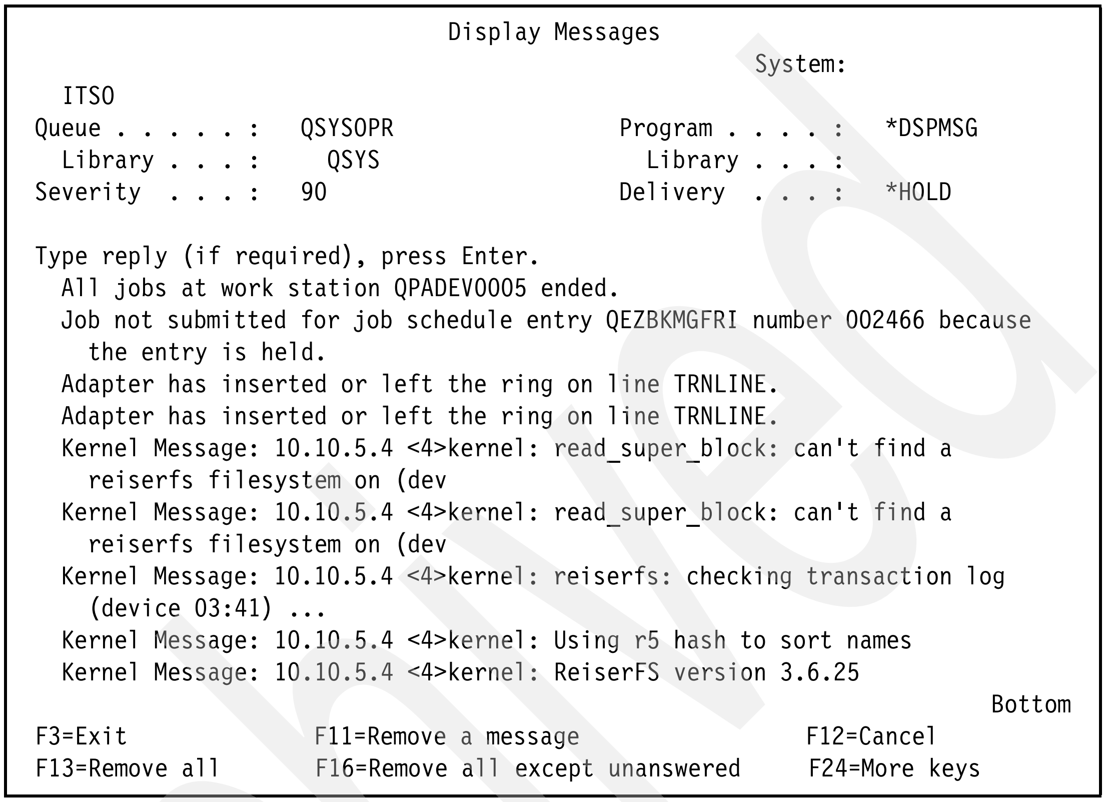


```
 Browse : /var/log/messages 79
 Record :    1242   of    1255 by  14          Column :    1    123 by  79
 Control :

....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....
Mar  8 14:04:40 SUSE641 <78>/USR/SBIN/CRON[1571]: (root) CMD ( rm -f
/var/spool/
Mar  8 14:27:29 10.10.5.4 <6>-- MARK --
Mar  8 14:32:26 SUSE641 <6>-- MARK --
Mar  8 14:39:06 SUSE641 <37>su: (to root) wildt on /dev/pts/1
Mar  8 14:39:06 SUSE641 <87>PAM-unix2[1628]: session started for user root,
serv
Mar  8 14:40:17 SUSE641 <87>PAM-unix2[1628]: session finished for user root,
ser
Mar  8 14:47:29 10.10.5.4 <6>-- MARK --
Mar  8 14:52:26 SUSE641 <6>-- MARK --
Mar  8 14:58:53 SUSE641 <38>sshd[1655]: Accepted password for ROOT from
::ffff:1
Mar  8 14:59:17 10.10.5.4 <38>sshd[733]: Could not reverse map address
10.10.5.1
Mar  8 14:59:20 10.10.5.4 <38>sshd[733]: Accepted password for ROOT from
::ffff:
Mar  8 15:01:09 10.10.5.4 <4>kernel: read_super_block: can't find a reiserfs
fil
Mar  8 15:01:09 10.10.5.4 <4>kernel: read_super_block: can't find a reiserfs
fil
Mar  8 15:01:31 10.10.5.4 <4>kernel: reiserfs: checking transaction log (device
 ***********End of Data*******************

 F3=Exit   F10=Display Hex   F12=Exit  F15=Services  F16=Repeat find
 F19=Left   F20=Right
```

Figure B-1 Bottom of /var/log/messages on IBM i5/OS

QSYSOPR message queue on IBM i5/OS

Figure B-2 shows the QSYSOPR message queue containing Linux messages from the syslogd program.



```
                              Display Messages
                                                     System:
  ITSO
Queue . . . . . :   QSYSOPR                 Program . . . . :   *DSPMSG
  Library . . . :     QSYS                    Library . . . :
Severity  . . . :   90                      Delivery  . . . :   *HOLD

Type reply (if required), press Enter.
  All jobs at work station QPADEV0005 ended.
  Job not submitted for job schedule entry QEZBKMGFRI number 002466 because
    the entry is held.
  Adapter has inserted or left the ring on line TRNLINE.
  Adapter has inserted or left the ring on line TRNLINE.
  Kernel Message: 10.10.5.4 <4>kernel: read_super_block: can't find a
    reiserfs filesystem on (dev
  Kernel Message: 10.10.5.4 <4>kernel: read_super_block: can't find a
    reiserfs filesystem on (dev
  Kernel Message: 10.10.5.4 <4>kernel: reiserfs: checking transaction log
    (device 03:41) ...
  Kernel Message: 10.10.5.4 <4>kernel: Using r5 hash to sort names
  Kernel Message: 10.10.5.4 <4>kernel: ReiserFS version 3.6.25
                                                                      Bottom
F3=Exit             F11=Remove a message                 F12=Cancel
F13=Remove all      F16=Remove all except unanswered     F24=More keys
```

Figure B-2   QSYSOPR message queue with Linux messages

C

ODBC connection string parameters

The iSeries Access Open Database Connectivity (ODBC) Driver for Linux has many connection string keywords that you can use to change the behavior of the ODBC connection. These same keywords and their values are also stored when an ODBC data source is configured. When an ODBC application makes a connection, any keywords specified in the connection string override the values specified in the ODBC data source.

The following table lists the different connection string keywords that are recognized by the driver. The current list of Parameters is available here:

`http://www.ibm.com/servers/eserver/iseries/linux/odbc/guide/odbcproperties.html`

General properties as listed in Table C-1 are always required in an ODBC connection string.

Table C-1 General properties

Keyword	Description	Choices	Default
[DSN]	Specifies the name of the ODBC data source that you want to use for the connection.	Data source name (DSN)	none
DRIVER	Specifies the name of the ODBC driver that you want to use. You must not use this if the DSN property has been specified.	"iSeries Access ODBC Driver"	none
PWD or Password	Specifies the password for connecting to the iSeries server.	iSeries password	none
SYSTEM	Specifies the name of the iSeries system that you want to connect to.	iSeries system name	none
UID or user identification	Specifies the user ID for connecting to the iSeries server.	iSeries user ID	none

Server properties are listed in Table C-2.

Table C-2 Server properties

Keyword	Description	Choices	Default
CMT or CommitMode	Specifies the default transaction isolation level.	0 = Commit immediate (*NONE) 1 = Read committed (*CS) 2 = Read uncommitted (*CHG) 3 = Repeatable read (*ALL) 4 = Serializable (*RR)	2
CONNTYPE or ConnectionType	Specifies the level of database access for the connection.	0 = Read/Write (all Structured Query Language (SQL) statements allowed) 1 = Read/Call (SELECT and CALL statements allowed) 2 = Read-only (SELECT statements only)	0
DBQ or DefaultLibraries	Specifies the iSeries libraries to add to the server job's library list. "*USRLIBL" may be used as a place holder for the server job's current library list. The library list is used for resolving unqualified stored procedure calls and finding libraries in catalog application programming interface (API) calls. If "*USRLIBL" is not specified, then the specified libraries will replace the server job's current library list.	i5/OS libraries delimited by comma or spaces *USRLIBL Note: the first library listed in this property will also be the default library, which is used to resolve unqualified names in SQL statements. To specify no default library, a comma should be entered before any libraries.	"QGPL"
NAM or Naming	Specifies the naming convention used when referring to tables.	0 = sql (as in *schema.table*) 1 = system (as in *schema/table*)	0
UNICODESQL	Specifies whether or not to send Unicode Structured Query Language (SQL) statements to the server. If set to 0, the driver sends Extended Binary Coded Decimal Interchange Code (EBCDIC) SQL statements to the server. This option is only available when connecting to servers at version 5 release 1 or later.	0 = Send EBCDIC SQL statements to the server 1 = Send Unicode SQL statements to the server	0

Data format properties are listed in Table C-3.

Table C-3 Format properties

Keyword	Description	Choices	Default
DFT or DateFormat	Specifies the date format used in date literals within SQL statements.	0 = yy/ddd (*JUL) 1 = mm/dd/yy (*MDY) 2 = dd/mm/yy (*DMY) 3 = yy/mm/dd (*YMD) 4 = mm/dd/yyyy (*USA) 5= yyyy-mm-dd (*ISO) 6 = dd.mm.yyyy (*EUR) 7 = yyyy-mm-dd (*JIS)	5

Keyword	Description	Choices	Default
DSP or DateSeparator	Specifies the date separator used in date literals within SQL statements. This property has no effect unless the DateFormat property is set to 0 (*JUL), 1 (*MDY), 2 (*DMY), or 3 (*YMD).	0 = "/" (forward slash) 1 = "-" (dash) 2 = "." (period) 3 = "," (comma) 4 = " " (blank)	1
DEC or Decimal	Specifies the decimal separator used in numeric literals within SQL statements.	0 = "." (period) 1 = "," (comma)	0
TFT or TimeFormat	Specifies the time format used in time literals within SQL statements.	0 = hh:mm:ss (*HMS) 1 = hh:mm AM/PM (*USA) 2 = hh.mm.ss (*ISO) 3 = hh.mm.ss (*EUR) 4 = hh:mm:ss (*JIS)	0
TSP or TimeSeparator	Specifies the time separator used in time literals within SQL statements. This property has no effect unless the "time format" property is set to "hms".	0 = ":" (colon) 1 = "." (period) 2 = "," (comma) 3 = " " (blank)	0

SQL package properties are listed in Table C-4.

Table C-4 Package properties

Keyword	Description	Choices	Default
DFTPKGLIB or DefaultPkgLibrary	Specifies the library for the SQL package. This property has no effect unless the XDYNAMIC property is set to 1.	Library for SQL package	"QGPL"
PKG or DefaultPackage	Specifies how the extended dynamic (package) support behaves. The string for this property must be in the following format: A/DEFAULT(IBM),x,0,y,z,0 The x, y, and z are special attributes that require to be replaced with how the package is to be used. If the package does not already exist on the server, a value of 2 must be specified for the x option. x = Specifies whether or not to add statements to an existing SQL package. y = Specifies the action to take when SQL package errors occur. When an SQL package error occurs, the driver returns a return code based on the value of this property. z = Specifies whether to cache SQL packages in memory. Caching SQL packages locally reduces the amount of communication to the server in some cases. Note, this property has no effect unless the XDYNAMIC property is set to 1.	"A/DEFAULT(IBM),x,0,y,z,0" Values for x option: 1 = Use, but do not put any more SQL statements into the package) 2 = Use/Add (Use the package and add new SQL statements into the package) Values for y option: 0 = Return error (SQL_ERROR) 1 = Return warning (SQL_SUCCESS_WITH_INFO) 2 = Return success (SQL_SUCCESS) Values for z option: 0 = Do not use local package caching 1 = Use PC memory to store package information	"A/DEFAULT (IBM),2,0,1,0, 512"

Keyword	Description	Choices	Default
XDYNAMIC or ExtendedDynamic	Specifies whether to use extended dynamic (package) support. Extended dynamic support provides a mechanism for caching dynamic SQL statements on the server. The first time a particular SQL statement is run, it is stored in a SQL package on the server. On subsequent runs of the same SQL statement, the server can skip a significant part of the processing by using information stored in the SQL package.	0 = Disable extended dynamic support 1 = Enable extended dynamic support	1

Performance related properties are listed in Table C-5.

Table C-5 Performance properties

Keyword	Description	Choices	Default
BLOCKFETCH	Specifies whether or not internal blocking will be done on fetches of 1 row. When set, the driver will try to optimize the fetching of records when one record is requested by the application. Multiple records will be retrieved and stored by the driver for later retrieval by the application. When an application requests another row, the driver does not require to send another flow to the host database to get it. If not set, blocking is used according to the application's ODBC settings for that particular statement.	0 = Use ODBC settings for blocking 1 = Use blocking with a fetch of 1 row	1
BLOCKSIZE or BlockSizeKB	Specifies the block size (in kilobytes) to retrieve from the iSeries server and cache on the client. This property has no effect unless the BLOCKFETCH property is 1. Larger block sizes reduce the frequency of communication to the server, and therefore may increase performance.	1 2 4 8 16 32 64 128 256 512	32
COMPRESSION or AllowDataCompression	Specifies whether to compress data sent to and from the server. In most cases, data compression improves performance due to less data being transmitted between the driver and the server.	0 = Disable compression 1 = Enable compression	1
CONCURRENCY	Specifies whether to override the ODBC concurrency settings by opening all cursors as updateable. Note: Setting has no effect: When you are building a SELECT SQL statement, you can add the FOR FETCH ONLY or FOR UPDATE clause. If either of these clauses are present in an SQL statement, the ODBC driver honors the concurrency that is associated with the clause. Catalog result sets are always read-only.	0 = Use ODBC concurrency settings 1 = Open all cursors as updateable	0
LAZYCLOSE	Specifies whether to delay closing cursors until subsequent requests. This increases overall performance by reducing the total number of requests. This option can cause problems, due to the cursors still holding locks on result set rows after the close request.	0 = Do not lazy close cursors 1 = Lazy close cursors	0

Keyword	Description	Choices	Default
MAXFIELDLEN or MaxFieldLength	Specifies the maximum LOB (large object) size in kilobytes that can be retrieved as part of a result set. Larger LOB thresholds reduces the frequency of communication to the server, but downloads more LOB data, even if it is not used. Smaller LOB thresholds may increase frequency of communication to the server, but only download LOB data as it is required. Setting this property to 0 forces locators to always be used.	0 - 2097152	15360
PREFETCH	Specifies whether to prefetch data upon executing a SELECT statement. This increases performance when accessing the initial rows in the ResultSet.	0 = Do not prefetch data 1 = Prefetch data	0
QUERYTIMEOUT	Specifies whether the driver disables support for the query timeout attribute, SQL_ATTR_QUERY_TIMEOUT. If disabled, SQL queries runs until they finish.	0 = Disable support for the query timeout attribute 1 = Allow the query timeout attribute to be set	1

Sort order properties are listed in Table C-6.

Table C-6 Sort properties

Keyword	Description	Choices	Default
LANGUAGEID	Specifies a three-character language ID to use for selection of a sort sequence. This property has no effect unless the SORTTYPE property is set to two.	"AFR", "ARA", "BEL", "BGR", "CAT", "CHS", "CHT", "CSY", "DAN", "DES", "DEU", "ELL", "ENA", "ENB", "ENG", "ENP", "ENU", "ESP", "EST", "FAR", "FIN", "FRA", "FRB", "FRC", "FRS", "GAE", "HEB", "HRV", "HUN", "ISL", "ITA", "ITS", "JPN", "KOR", "LAO", "LVA", "LTU", "MKD", "NLB", "NLD", "NON", "NOR", "PLK", "PTB", "PTG", "RMS", "ROM", "RUS", "SKY", "SLO", "SQI", "SRB", "SRL", "SVE", "THA", "TRK", "UKR", "URD", "VIE"	"ENU"
SORTTABLE	Specifies the library and file name of a sort sequence table stored on the iSeries server. This property has no effect unless the SORTTYPE property is set to three.	Qualified sort table name	none
SORTTYPE or SortSequence	Specifies how the server sorts records before sending them to the client.	0 = Sort based on hexadecimal values 1 = Sort based on the setting for the server job 2 = Sort based on the language set in LANGUAGEID property 3 = Sort based on the sort sequence table set in the SORTTABLE property	0

Catalog related properties are listed in Table C-7.

Table C-7 Catalog properties

Keyword	Description	Choices	Default
CATALOGOPTIONS	Specifies how catalog APIs return information about aliases.	0 = Do not return information about aliases in the SQLColumns result set. 1 = Return information about aliases in the SQLColumns result set.	1
LIBVIEW or LibraryView	Specifies the set of libraries to be searched while returning information when wildcards are used in the catalog APIs. In most cases, use the default library list option because searching all the libraries on the system takes a long time.	0 = Use default library list 1 = All libraries on the system	0
REMARKS or ODBCRemarks	Specifies the source of the text for REMARKS columns in catalog API result sets.	0 = OS/400 object description 1 = SQL object comment	0
SEARCHPATTERN	Specifies whether the driver interprets string search patterns and underscores in the library and table names as wildcards (search patterns). By default, % is treated as an 'any number of characters' wildcard, and _ is treated as a 'single character' wildcard.	0 = Do not treat search patterns as wildcards 1 = Treat search patterns as wildcards	1

Character translation properties are listed in Table C-8.

Table C-8 Translation properties

Keyword	Description	Choices	Default
ALLOWUNSCHAR or AllowUnsupportedChar	Specifies whether or not to suppress error messages which occur when characters that cannot be translated (because they are unsupported) are detected.	0 = Report error messages when characters cannot be translated 1 = Suppress error messages when characters cannot be translated	0
Coded character set identifier (CCSID)	Specifies a code page to override the default client code page setting.	Client code page setting or 0 = use default client code page setting	0
GRAPHIC	This property affects the handling of the graphic double-byte character set (DBCS) data types of GRAPHIC, VARGRAPHIC, LONG VARGRAPHIC, and DBCLOB that have a CCSID other than Unicode (13488). This property affects two different behaviors: ► Whether graphic fields have their lengths reported as a character count or byte count through the SQLDescribeCol API and SQLColAttribute API with the SQL_COLUMN_LENGTH option. ► Whether graphic fields are reported as a supported type in the SQLGetTypeInfo result set.	0 = Report character count, report as not supported 1 = Report character count, report as supported 2 = Report byte count, report as not supported 3 = Report byte count, report as supported	0

Keyword	Description	Choices	Default
TRANSLATE or ForceTranslation	Specifies whether or not to convert binary data (CCSID 65535) to text. Setting this property to 1 makes binary fields look like character fields.	0 = Do not convert binary data to text 1 = Convert binary data to text	0

Properties for diagnosis and troubleshooting are listed in Table C-9.

Table C-9 Diagnostic properties

Keyword	Description	Choices	Default
QAQQINILIB or QAQQINILibrary	Specifies a query options file library. When a query options file library is specified the driver issues the command CHGQRYA passing the library name for the QRYOPTLIB parameter. The command is issued immediately after the connection is established. You must use this option only when debugging problems or when recommended by support, because enabling it adversely affects performance.	Query options file library	none
SQDIAGCODE	Specifies DB2 Universal Database (UDB) SQL diagnostic options to be set. Use only as directed by your technical support provider.	DB2 UDB SQL diagnostic options	none
TRACE	Specifies one or more trace options. To specify multiple trace options, sum up the values for the options that you want. For example, if you want the Database Monitor and Start Debug command to be activated on the server, specify a value of 6. You must use these options only when debugging problems or when recommended by support, because they adversely affect performance.	Sum the following options together that you want: 0 = No tracing 2 = Enable Database Monitor 4 = Enable the Start Debug (STRDBG) command 8 = Print job log at disconnect 16 = Enable job trace	0

Other properties are listed in Table C-10.

Table C-10 Other properties

Keyword	Description	Choices	Default
ALLOWPROCCALLS	Specifies whether stored procedures can be called when the connection attribute SQL_ATTR_ACCESS_MODE is set to SQL_MODE_READ_ONLY.	0 = Do not allow stored procedures to be called 1 = Allow stored procedures to be called	0
DB2SQLSTATES	Specifies whether to return ODBC-defined SQL States or DB2 SQL States. Refer to the DB2 UDB SQL Reference for further details on the DB2 SQL States. You must use this option if you have the ability to change the ODBC application's source code. If not, you must leave this option set to 0 as most applications are coded only to handle only the ODBC-defined SQL States.	0 = Return ODBC-defined SQLStates 1 = Return DB2 SQL States	0

Keyword	Description	Choices	Default
DEBUG	Specifies one or more debug options. To specify multiple debug options, sum up the values for the options that you want. In most cases, you do not require to set this option.	Sum the following options together that you want: 2 = Return SQL_IC_MIXED for the SQL_IDENTIFIER_CASE option of SQLGetInfo 4 = Store all SELECT statements in the package 8 = Return zero for the SQL_MAX_QUALIFIER_NAME_LEN option of option of SQLGetInfo 16 = Add positioned UPDATEs / DELETEs into packages 32 = Convert static cursors to dynamic cursors	0
TRUEAUTOCOMMIT	Specifies whether or not to enable a true autocommit. True autocommit means that autocommit is on and is running under a isolation level other than *NONE. By default, the driver handles autocommit by running under the server isolation level of *NONE.	0 = Do not use true autocommit 1 = Use true autocommit	0

D

Additional material

This IBM Redbook refers to additional material that you can download from the Internet as described in the following sections.

Locating the Web material

The Web material associated with this IBM Redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

`ftp://www.redbooks.ibm.com/redbook/SG246551`

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the IBM Redbook titled *Linux Integration with IBM i5/OS*, SG24-6551.

Using the Web material

The additional Web material that accompanies this IBM Redbook includes the following files:

File name	*Description*
Scripts.zip	This single zip file holds all the contents required for the scenario described in this IBM Redbook.

System requirements for downloading the Web material

We recommend the following system configuration:

Hard disk space:	10 MB
Operating System:	i5/OS V5R4 and SLES 10

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this IBM Redbook.

IBM Redbooks

For information about ordering these publications, see "How to get IBM Redbooks" on page 162.

- *IBM AS/400 Printing V*, SG24-2160
- *The AS/400 NetServer Advantage*, SG24-5196
- *Linux on the IBM eServer iSeries Server: An Implementation Guide*, SG24-6232
- *IBM eServer iSeries Printing VI: Delivering the Output of e-business*, SG24-6250
- *V5 TCP/IP Applications on the IBM eServer iSeries Server*, SG24-6321
- *Implementing Web Applications with CM Information Integrator for Content and OnDemand Web Enablement Kit*, SG24-6338
- *Implementing Linux on Integrated xSeries Solutions for iSeries*, SG24-6379
- *Windows-based Single Signon and the EIM Framework on the IBM eServer iSeries Server*, SG24-6975

Referenced Web sites

These Web sites are also relevant as further information sources:

- Heimdal

  `http://www.pdc.kth.se/heimdal/`
- IBM eServer iSeries information center

  `http://publib.boulder.ibm.com/pubs/html/as400/`
- iSeries Access for Linux main site

  `http://www.ibm.com/servers/eserver/iseries/access/linux/`
- iSeries NetServer main site

  `http://www-1.ibm.com/servers/eserver/iseries/netserver/`
- Kerberos: The Network Authentication Protocol

  `http://web.mit.edu/kerberos/www/`
- Open Office

  `http://www.openoffice.org/`
- Printing Systems

  `http://www.printers.ibm.com`

- The Linux Documentation Project

  http://www.tldp.org/
- Toolbox for Java main site

  http://www.ibm.com/eservers/iseries/toolbox
  http://www-03.ibm.com/servers/eserver/iseries/toolbox/faqjdbc.html

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

IBM Redbooks

Linux Integration with IBM i5/OS

(0.2”spine)
0.17”<->0.473”
90<->249 pages

Linux Integration with IBM i5/OS

iSeries access for Linux installation, features, and users guide

Various access methods of IBM i5/OS data from Linux

Plan and implement an SSO scenario with Linux

IBM System i platform offers many points of integration that support Linux applications leveraging IBM i5/OS applications and data, such as IBM eServer iSeries Access for Linux, IBM i5/OS NetServer with Samba support, and IBM Java Virtual Machine and JTOpen support.

This IBM Redbook covers each communication technique with sample code and shows you the methods that you can use to connect Linux applications with IBM i5/OS.

This book is intended to help Linux users, programmers and administrators, with an IBM i5/OS background, to connect Linux with IBM i5/OS.

SG24-6551-00 ISBN 0738496677