

Session: EL06

LAB: Implementing Single Sign-on

ITSO iSeries Technical Forum - 2003

(c) Copyright IBM Corporation, 2003. All Rights Reserved

This publication may refer to products that are not currently available in your country.
IBM makes no commitment to make available any products referred to herein.

Agenda

Abstract	3
Prerequisites	3
Where to Find More Information	3
Software Requirements	3
Lab Information	4
Final Comments Before Starting:	5
Lab Start Up Instructions:	5
Exercise 1: Configuring iSeries to Participate in a Kerberos Realm	7
Exercise 2: Configuring iSeries to participate in an EIM domain.	8
Exercise 3: Configuring EIM Associations	11
Exercise 4: Accessing Systems Through iSeries Navigator Using Kerberos Authentication	14
Exercise 5: Accessing Systems Through QFileSvr.400 and TELNET	15
Exercise 6: Working with a Kerberos and EIM Enabled Application:	16
Lab Cleanup	20
Conclusion:	21
Appendix A: Purpose and Overview of EIM	22
Trademarks and Disclaimers	30

Abstract

In this lab you will learn how to configure and exploit a single-sign-on environment. Lab exercises will include:

1. Configuring iSeries to Participate in a Kerberos Realm
2. Configuring iSeries to participate in an EIM domain.
3. Configuring EIM associations
4. Accessing Systems Through iSeries Navigator Using Kerberos Authentication
5. Accessing systems through QFileSvr.400 and TELNET. See how EIM associations are used to determine with which profile to run.
6. Working with a Kerberos and EIM enabled application.

Where to Find More Information

EIM purpose and overview information is provided in the appendix of this document.

Web Sites

iSeries Information Center: <http://publib.boulder.ibm.com/pubs/html/as400/infocenter.html>

- Select the appropriate geography.
- Select V5R2.
- Follow the links: **Security -> Enterprise Identity Mapping (EIM)**

Software Requirements

The following software is required to perform this lab.

- V5R2 version of 5722-SS1 Base OS/400
- 5722-AC3 Crypto Access Provider
- iSeries Access

In addition, you must have a Kerberos realm that the iSeries can participate in.

Lab Information

The following preparatory steps have been completed for you:

- All the necessary SW prerequisites have been installed.
- Your PC has been configured to participate in the Kerberos realm IBMROCHESTERMN.DEMOS.COM
- The iSeries **s400a** and **s400b** have been configured to participate in the EIM domain for DEMOS.
- Lab user profiles have been created.
- Lab user profiles have been authorized to the necessary directories.
- Example data has been primed in the appropriate directories.

The iSeries systems you will use in this lab are: **s400a** and **s400b**.

The PC you are using has a unique 2-digit number assigned to it which is displayed on a sheet of paper. You will use this number in your user profile and password to maintain uniqueness between students during the lab. You will replace XX with your 2-digit number.

In addition to your workstation ID, you will be using OS/400 profiles on **s400a** and **s400b**, and a series of principals from a Kerberos realm.

The naming convention for the OS/400 profiles are listed below. You will replace XX with your 2-digit workstation number. The OS/400 user profiles have the special authorities *ALLOBJ, *SECADM, and *IOSYSCFG which are needed to run the configuration wizards.

Your OS/400 *USRPRF	prfXXos400	prf_____os400
Your OS/400 *USRPRF Password	pwdXX	pwd_____

The naming convention for the Kerberos principals for the domain IBMROCHESTERMN.DEMOS.COM are listed below. In this lab you will map wsXX to profiles on the **s400a** and **s400b** systems.

Your IBMROCHESTERMN.DEMOS.COM Kerberos Principal	wsXX	ws_____
Your principal's password	pwdXX	pwd_____

Final Comments Before Starting:

In this lab, you will use iSeries Navigator connections and OS/400 interactive session to perform the tasks. The first tasks will be performed through iSeries Navigator. You will use a “green screen” 5250 terminal session for some exercises in this lab. You will be told how to start the 5250 session during the exercise.

The lab exercises are designed to be completed in the order in which they are documented. If you see unexpected results while performing an exercise, notify the instructor of the error. Do not continue with other lab exercises until the situation is corrected. Begin with Exercise 1 and continue through all exercises.

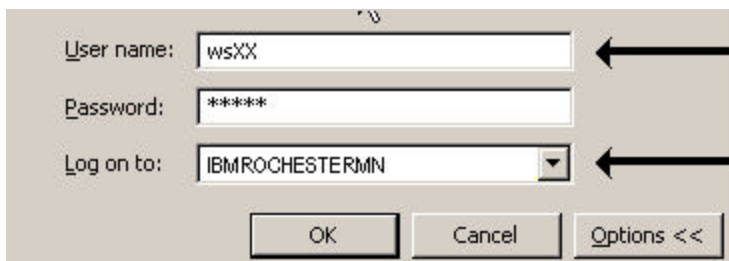
Please do NOT change or edit anything on the iSeries systems other than what you are instructed to do during the lab.

Boxes are provided by each step that you will complete in this lab. You may mark them to keep track of the steps you have completed.

Work at your own pace. If you have questions, raise your hand to get the instructor's attention.

Lab Start Up Instructions:

- ❑ Please start this lab by signing onto the workstation with the **wsXX** ID associated with your workstation. You must log on to the **IBMROCHESTERMN** domain. Your workstation sign on screen should have the following information:



Use your assigned ID and password.

Make sure you log on to the proper domain. NOTE: If the “log on to:” line does not appear on your login screen, click *Options*.

- ❑ Bring up iSeries Navigator by double clicking on the iSeries Navigator icon on the desktop.



- ❑ In the *Environment: My Connections* panel of iSeries Navigator (the upper left hand panel), move the mouse over the + character next to *My Connections* and press the left mouse button.

- Look for the names of the iSeries systems **s400a** and **s400b**. If either of the connections is missing, follow the steps listed below. If both connections are present, skip these instructions.

Instructions for adding a system connection in iSeries Navigator:

- In the *My Tasks* panel of iSeries Navigator (the lower left hand panel), move the mouse over *Add a connection* and press the left mouse button.
- On the *Add Connection - Welcome* screen, enter **s400a** or **s400b** in the *server* input box and click *Next*.
- On the *Add Connection - Signon Information* screen, do the following:
 - Select the *Use default user ID, prompt as needed* radio button.
 - Enter **prfXXos400** in the input box associated with the radio button. XX = your workstation ID.
 - Click *Next*.
- On the *Add Connection - Verify Connection* screen, click *Finish*. (It's not necessary, but you may click *Verify Connection* if you wish.)

NOTES:

- ✓ **s400a** or **s400b** should now appear in your *My Connections* list.
- ✓ Repeat the steps for the “other” system if both **s400a** and **s400b** were missing from your *My Connections* list.

Exercise 1: Configuring iSeries to Participate in a Kerberos Realm

EIM is used to determine which local ID to use after a network authentication has been performed. Kerberos is the supported network authentication method. Therefore, your iSeries must be configured to participate in a Kerberos realm. You can easily configure your iSeries to participate in a Kerberos realm using the Network Authentication Service wizard.

NOTE: The EIM configuration wizard will take you through the Network Authentication Service wizard if you have not previously configured your system to participate in a Kerberos realm.

- 1. In iSeries Navigator, click on the + next to the system named *s400a*.
- 2. Enter **prfXXos400** and **pwdXX** for the *User ID* and *Password*. Click *OK*.
- 3. Click on the + next to *Security*.
- 4. Right click on *Network Authentication Service* and select *Reconfigure...* from the context menu.
Note: The option will be *Configure* if the Network Authentication Service wizard has not been run before.
- 5. On the “welcome” screen, review the text and click *Next*.
Note: In this lab, you will get a *Network Authentication Configuration Warning* screen. This indicates that the system is already configured. Click *OK* to continue.
- 6. On the “specify realm information” screen, enter *IBMROCHESTERMN.DEMOS.COM* and click *Next*.
- 7. On the “specify KDC information” screen, enter *itsohost.ibmrochestermn.demos.com* and click *Next*.
- 8. On the “specify password server information” screen, keep all the default values and click *Next*.
- 9. On the “create keytab entry” screen, check all the boxes and click *Next*.
Note: The various kerberos enabled iSeries functions require particular service principals in the KDC. The principal passwords must be the same in the KDC and the local keytab file.
- 10. On the “create iSeries keytab entry” screen, enter *kerberos* in both password fields and click *Next*.
- 11. On the “create LDAP keytab entry” screen, enter *kerberos* in both password fields and click *Next*.
- 12. On the “create NetServer keytab entry” screen, enter *kerberos* in both password fields and click *Next*.
- 13. From the “summary” screen, click *Finish* to complete the wizard.

Global Note: Running the Network Authentication Wizard effects everyone. The wizard will not update the kerberos configuration files or the keytab file. The system is already participating in a kerberos realm.

Exercise 2: Configuring iSeries to participate in an EIM domain.

Before iSeries functions or user written applications for iSeries can take advantage of Kerberos authentication with EIM mappings, you need to configure each iSeries to participate in an EIM domain. You can easily configure your iSeries to create or join an EIM domain using the EIM configuration wizard. If your EIM domain controller is going to reside on your iSeries, you can use the wizard to create the EIM domain. If the EIM domain controller is on a remote system, you can use the wizard to join the EIM domain. The following steps will take you through creating an EIM domain on **s400a** and joining an EIM domain on **s400b**.

Create an EIM domain on **s400a**:

1. In iSeries Navigator, click on the + next to the lab system named *s400a*. This may not be necessary, **s400a** may still be open from the last exercise.
Note: If you closed iSeries Navigator and reopened it, you may be challenged for a user id and password. If you are, enter *prfXXos400* and *pwdXX* and click *OK*.
2. Click on the + next to *Network*.
3. Click on the + next to *Enterprise Identity Mapping*.
4. Right click on *Configuration* and select *Reconfigure...* from the context menu.
Note: The option will be *Configure* if there hasn't been any LDAP or EIM configuration activity on the machine prior to launching the wizard.
5. On the “welcome” screen, select the *Create and join a new domain* radio button and click *Next*.
Note: You may get an *EIM Configuration Warning* screen. Click *Yes* to ignore the message.
6. On the “specify domain” screen enter *EIM domain for DEMOS lab* for the domain name and click *Next*.
Note: It is not necessary, but you may change the description value if you wish
7. On the “specify parent DN for domain” screen, select *No* (the default) and click *Next*.
8. On the “specify user for connection” screen, specify *cn=Admin* for the distinguished name and enter *secret* in both password fields and click *Next*.
9. On the “registry information” screen, verify:
 - The *Local OS/400* and *Kerberos* check boxes are selected.
 - The *Local OS/400* value is set to the fully qualified name for **s400a** .
 - The *Kerberos* value is set to *IBMROCHESTERMN.DEMOS.COM*.
 - The *Kerberos user identities are case sensitive* box is checked by default.
Note: When you activate this at your site, select the value that is appropriate for your environment.
10. Click *Next* to exit the “registry information” screen.

- 11. On the “specify EIM system user” screen, click *Next* to move to the next screen.
Note: The distinguished name and password fields are filled in with the data you entered on the “specify user for connection” screen.

Important things to know: When you change the password of the domain controller (your LDAP administrator) ID, you need to update the EIM configuration. This is done through iSeries Navigator:

- *Network -> Enterprise Identity Mapping -> Configuration*
- Right click on *Configuration* and select *properties* from the context menu
- Update the password from the *System User* tab

- 12. From the “summary” screen, click *Finish* to complete the wizard.

Global Note: Because stopping and starting the domain controller would effect everyone in the lab, the wizard will not create an EIM domain. The domain has already been created for you.

Joining an EIM domain on **s400b**:

- 1. In iSeries Navigator, click on the + next to the lab system name: **s400b**
- 2. Enter **prfXXos400** and **pwdXX** for the *User ID* and *Password*. Click *OK*.
- 3. Click on the + next to *Network*.
- 4. Click on the + next to *Enterprise Identity Mapping*.
- 5. Right click on *Configuration* and select *Reconfigure...* from the context menu.
Note: The option will be *Configure* if there hasn't been any LDAP or EIM configuration activity on the machine prior to launching the wizard.
- 6. On the “welcome” screen, select the *Join an existing domain* radio button and click *Next*.
- 7. On the “specify domain controller” screen:
 - Change the domain controller value to the fully qualified name for **s400a.ibmrochester.mn.demos.com**
Note: The domain controller name defaults to the fully qualified name for **s400b**. You only need to change **s400b** to **s400a**. The rest of the name is fine.
 - Verify the *use secure connection (SSL or TLS)* check box is not selected.
Note: Uncheck the box if necessary.
 - Click *Next* to exit this screen.
- 8. On the “specify user for connection” screen, specify *cn=Admin* for the distinguished name and enter *secret* in both password fields and click *Next*.

- 9. On the “specify domain” screen, select (left click on) *EIM domain for DEMOS lab* and click *Next* to exit this screen.

- 10. On the “registry information” screen, verify:
 - The *Local OS/400* and *Kerberos* check boxes are selected.
 - The *Local OS/400* value is set to the fully qualified name for **s400b**.
 - The *Kerberos* value is set to *IBMROCHESTERMN.DEMOS.COM*.
 - The *Kerberos user identities are case sensitive* box is checked by default.
Note: When you activate this at your site, select the value that is appropriate for your environment.
- 11. Click *Next* to exit the “registry information” screen.
- 12. On the “specify EIM system user” screen, click *Next* to move to the next screen.
Note: The distinguished name and password fields are filled in with the data you entered on the “specify user for connection” screen.

Important things to know: When you change the password of the domain controller (your LDAP administrator) ID, you need to update the EIM configuration. This is done through iSeries Navigator:

- *Network -> Enterprise Identity Mapping -> Configuration*
- Right click on *Configuration* and select *properties* from the context menu
- Update the password from the *System User*.

- 13. From the “summary” screen, click *Finish* to complete the wizard.

Global Note: Because stopping and starting the domain controller would effect everyone in the lab, the wizard will not join the EIM domain. The system has already joined the domain

Exercise 3: Configuring EIM Associations

In this exercise you will create an EIM identifier for a person or server. You will also create three EIM associations for the identifier you create:

1. A **SOURCE** association for the principal *wsXX* in the Kerberos registry
2. A **TARGET** association for the user profile *AAxxnnnnnn* in the **s400a** registry
3. A **TARGET** association for the user profile *BBxxnnnnnn* in the **s400b** registry

Below is a pictorial view of the tasks.

Create an EIM identifier with <i>name-of-your-choice</i>		
Create SOURCE association	Create TARGET association	Create TARGET association
Registry = IBMROCHESTERMN.DEMOS.COM Registry User = wsXX	Registry = s400a Registry User = AAxxnnnnnn	Registry = s400b Registry User = BBxxnnnnnn

NOTES:

- There is only one Kerberos principal for your workstation *wsXX*, where *XX* is your workstation number.
- There are 4 user profiles on each iSeries for your use when in your TARGET associations. The available profiles are listed in the table below. In all cases, *xx* is your workstation number:

*USRPRFs for use in TARGET associations	
System s400a	System s400b
AAxxAMY	BBxxAMY
AAxxDAVE	BBxxDAVE
AAxxEMILY	BBxxEMILY
AAxxMIKE	BBxxMIKE

Create an EIM identifier:

1. In iSeries Navigator, click on the + next to the lab system named *s400a*. This may not be necessary, **s400a** may still be open from the last exercise.
Note: If you closed iSeries Navigator and reopened it, you may be challenged for a user ID and password. If you are, enter *prfXXos400* and *pwdXX* and click *OK*.
2. Click on the + next to *Network*.
3. Click on the + next to *Enterprise Identity Mapping*.
4. Click on the + next to *Domain Management*.
5. Click on the + next to *EIM Domain for DEMOS*.

Note: If the plus sign (+) is not there then you must right-click to Add a new Domain. Take all the defaults. The reason it might not be there is that this is a client (iSeries Navigator) view of the configured domains. Once this domain connection has been created on this client (for this user) you will be able to expand and see all the previous configuration.

- 6. On the “connect to EIM domain controller - S400a” screen, enter *cn=Admin* and *secret* for the password. Then click *OK*.

- 7. Double click over *Identifiers* to open the “S400a: *Identifiers*” list. (The list may be empty)
- 8. Right click on *Identifiers* and select *New Identifier* from the context menu to open the “new EIM identifier” screen.
Note: You can also open the “new EIM identifier” screen by selecting the *create a new identifier* option in the “Enterprise Identity Mapping tasks” window.
- 9. On the “new EIM identifier” screen, enter any value you want in the *identifier* input field and click *OK*. (All other fields on the screen may be left blank.)

EXAMPLES: *Workstation XX* or *Susan Jane Doe*

NOTES: Every identifier is visible to each lab participant. The identifier values must be unique.

Note your value: _____

Create the SOURCE association for wsXX:

- 10. Your EIM identifier will appear in “S400a: *identifiers*” window (the upper right hand panel) of iSeries Navigator. Right click on the identifier you just created and select *properties* from the context menu.
- 11. Select the *Associations* tab on the identities “*properties*” screen.
- 12. Click *Add...* in the “*associations*” tab.
- 13. On the *add association* screen do the following:
 - Set the *registry* field value to **IBMROCHESTERMN.DEMOS.COM**. Either type the value in or select it from a list after pressing the *Browse...* button.
 - Set the *user* field value to **wsXX**, where xx is your workstation number. Initially this field will contain the name of the profile you used to sign onto **s400a**.
 - Set the *association type* field to **source**. You can either use the pull down menu or press the “s” key until the value *source* appears.
 - Click *OK*. You will return to the identifier’s *properties* page. The SOURCE association will appear in the list.

NOTES:

- The association will not be created until you click *OK* from the identifier’s properties page.
- DON’T exit the identifier’s properties page yet. You will create two more associations.

Create a TARGET association for your identifier for s400a:

- 14. Click *Add...* in the “*associations*” tab.
- 15. On the *add association* screen do the following:
 - Set the *registry* field value to the fully qualified name for *s400a*. Either type the value in or select it from a list after pressing the *Browse...* button.
 - Set the *user* field value to any one the following values: *AAxxAMY*, *AAxxDAVE*, *AAxxEMILY*, or *AAxxMIKE*, where xx is your workstation number.
Note your value: _____
 - Set the *association type* field value to ***target***. You can either use the pull down menu or press the “*t*” key until the value *target* appears.
 - Click *OK*. You will return to the identifier’s *properties* page. The TARGET association will appear in the list.

NOTES:

- The associations will not be created until you click *OK* from the identifier’s *properties* page.
- DON’T exit the identifier’s *properties* page yet. You will create one more association.

Create a TARGET association for your identifier for s400b:

- 16. Click *Add...* in the “*associations*” tab.
- 17. On the *add association* screen do the following:
 - Set the *registry* field value to the fully qualified name for *s400b*. Either type the value in or select it from a list after pressing the *Browse...* button.
 - Set the *user* field value to any one the following values: *BBxxAMY*, *BBxxDAVE*, *BBxxEMILY*, or *BBxxMIKE*, where xx is your workstation number.
Note your value: _____
 - Set the *association type* field value to ***target***. You can either use the pull down menu or press the “*t*” key until the value *target* appears.
 - Click *OK*. You will return to the identifier’s *properties* page. The TARGET association will appear in the list.
- 19. Click *OK* on the identifier’s *properties* page. This will create all three associations.

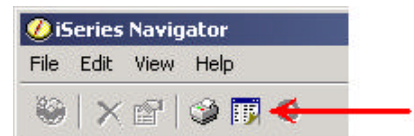
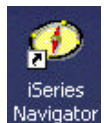
Exercise 4: Accessing Systems Through iSeries Navigator Using Kerberos Authentication

In this exercise you will use Kerberos authentication for iSeries Navigator connections to **s400a** and **s400b**. The connections will use the EIM mappings created in the last exercise.

- 1. In iSeries Navigator, move the cursor over the lab system named **s400a**. Right click to bring up the context menu and select *properties*.
Note: If you closed iSeries Navigator and reopened it, you may be challenged for a user id and password. Click *cancel* to continue.
- 2. On the “*S400a Properties*” screen, select the *Connection* tab.
- 3. On the “*Connection*” tab, select the “*use kerberos principal name, no prompting*” radio button.
- 4. On the “*S400a Properties*” screen, click *OK*.
Note: You will get a *connection* informational message that states you must exit iSeries Navigator before the change takes effect. Click *OK*.
- 5. Move the cursor over the lab system named **s400b**. Right click to bring up the context menu and select *properties*.
Note: If you closed iSeries Navigator and reopened it, you may be challenged for a user id and password. Click *cancel* to continue.
- 6. On the “*S400b Properties*” screen, select the *Connection* tab.
- 7. On the “*Connection*” tab, select the “*use kerberos principal name, no prompting*” radio button.
- 8. On the “*S400a Properties*” screen, click *OK*.
Note: You will get a *connection* informational message that states you must exit iSeries Navigator before the change takes effect. Click *OK*.
- 9. Close iSeries Navigator by selecting *Close* from the *File* pull down menu or by pressing the *X* button in the top right hand corner of the iSeries Navigator window.

Connect using Kerberos authentication and EIM mappings:

- 10 Bring up iSeries Navigator by double clicking on the iSeries Navigator icon on the desktop.
- 11. Click on the + next to **s400a**. The system should expand without any profile challenge.
- 12. Click on the + next to **s400b**. The system should expand without any profile challenge.
- 13. Click *Refresh* to update the *signed on user* field of the *My Connections* (upper right hand) panel of the iSeries Navigator window. The arrow points to the refresh button.




CONGRATULATIONS - You used kerberos authentication with EIM to connect to **s400a** and **s400b**. Notice that user profiles associated with the two connections are the OS/400 *USRPRF names you specified in the target associations you created. The profiles are **NOT** the prfXXos400 ID you originally used.

Exercise 5: Accessing Systems Through QFileSvr.400 and TELNET

In the first part of this exercise you will use QFileSvr.400 on system **s400a** to view objects on system **s400b**.

Using QFileSvr.400 to access data on s400b from s400a:

- 1. In iSeries Navigator, click on the + next to the lab system named **s400a**. This may not be necessary, **s400a** may still be open from the last exercise.
- 2. Click on the + next to *File Systems*.
- 3. Click on the + next to *Integrated File System*.
- 4. Right click over *QFileSvr.400* and select *New Folder...* from the context menu.
- 5. On the “new folder - S400a” screen, enter **s400b** and click *OK*
- 6. Click on the + next to *QFileSvr.400*. **S400B** should appear under the QFileSvr.400 directory.
- 7. Click on the + next to **S400B** to view directories on that system.
Note: You have connected to **s400b** with the user ID you specified in the “target association” for system **s400b**.
- 8. Click on the + next to *home*.
- 9. Right click over the folder icon next to the directory that has the same name as the “registry user” you specified on the “target association” for system **s400b**. Select *Explore* from the context menu. The list of objects in the file will be displayed.
Note: The authorities on the ‘/home/BBxxxxnnn’ directories are set up so that only the associated BBxxxxnnn profile can read the directory. This proves you connected through QFileSvr.400 using kerberos and EIM.

- 10. Right click over the folder icon next to any BBxxxxnnnn directory that is **NOT** the same as the “registry user” you specified on the “target association” for system **s400b**. Select *Explore* from the context menu. You will receive an iSeries Navigator error message stating you are not authorized to access the object.

Using TELNET to access an iSeries:

NOTE: To bypass the signon screen and use the EIM mapping, the system value QRMTSIGN must be *SAMEPRF or *VERIFY. This has been done on **s400a** and **s400b**.

- 11. Right click over **s400a** and select *Display Emulator...* From the context menu. Without a user ID and password challenge, you will be signed onto the system. Issue DSPJOB and verify the user ID of the job the “registry user” you specified on the “target association” for system the system you picked. **DON'T CLOSE THIS SESSION.** It will be used in the next exercise.

Exercise 6: Working with a Kerberos and EIM Enabled Application:

In this exercise, you will create a Domain user that is able to retrieve EIM mapping information. You will then work with an application that takes three parameters:

1. A Kerberos principle
2. A Kerberos domain name
3. The principal's password

The application will issue a kinit for the principal. If the kinit is successful, the program will use EIM to determine what local *USRPRF to swap to.

Create a *dn* authorized to do EIM look-ups:

The dn and password will be used in your server application. Parts of this exercise will be done from the "green screen" and parts will be done from iSeries Navigator.

- 1. The first part of this exercise is done from a 5250 session. The user must have some special authority. In the last exercise you started a device emulation session. However, your user does not have any special authority. Issue the SIGNOFF command from the device emulation session. You now have a signon screen where you can provide a more powerful user and password.
- 2. Sign onto the **s400a** emulation session with the profile **prfXXos400** and password **pwdXX**, where xx is replaced with your workstation number.
- 3. From the *command entry* screen, enter the command *wrklnk '/home/prfXXos400/crtprf*'*, and press *enter*. (XX is replaced with your workstation number.)
- 4. Select option **2** (edit) next to the file *CrtPrfXXos400DomainUser.ldf* and press *enter*. This file contains a shell script that will create the domain controller user.
- 5. From the *Edit File Control* line, enter the command *c prfxos400 prfWWos400 all* to change xx to your workstation number.

Notes:

- The underlined WW is where you put your workstation number.
 - This script file is being used to create a DN and password that the server program will use to connect to the domain controller.
 - The password for the ID is *secret*.
 - After the create is done, the dn will be added to an ACL that will only permit it to do EIM look-ups.
 - The comment line in the script file: **ldapadd -c -h opsb -D cn=admin -w secret -f '/home/prfxos400/crtprfxos400domainuser.ldf'** shows the command used to execute the script. It was put there so you could cut and past the command.
- 6. Press *F3 twice* to save and exit the file.
 - 7. From the command line enter the command *qsh* and press *enter*.

- 8. From the QSH Command Entry screen enter the command:

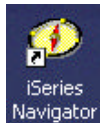
ldapadd -c -h opsb -D cn=admin -w secret -f '/home/prfxxos400/crtprfxxos400domainuser.ldf'

Where xx is replaces with your work station number. The dn *prfxxos400lookup* has been created. The next step is to attach it to an ACL that gives the dn authority to do EIM look-ups. Don't sign off the green screen, you'll be back.

Give the *dn* authority to do EIM lookup operations:

This function is done through iSeries Navigator. However, you need more authority than your AAXxxxx profile provides. The iSeries Navigator connection method will be switch to profile prompting.

- 1. In iSeries Navigator, move the cursor over the lab system named *s400a*. Right click to bring up the context menu and select *properties*.
- 2. On the “*S400a Properties*” screen, select the *Connection* tab.
- 3. On the “*Connection*” tab, select the “*prompt every time*” radio button.
- 4. On the “*S400a Properties*” screen, click *OK*.
Note: You will get a *connection* informational message that states you must exit iSeries Navigator before the change takes effect. Click *OK*.
- 5. Close iSeries Navigator by selecting *Close* from the *File* pull down menu or by pressing the *X* button in the top right hand corner of the iSeries Navigator window.
- 6. Bring up iSeries Navigator by double clicking on the iSeries Navigator icon on the desktop.
- 7. Click on the + next to *s400a*.
- 8. Enter **prfXXos400** and **pwdXX** for the *User ID* and *Password*. Click *OK*.
- 9. Click on the + next to *Network*.
- 10. Click on the + next to *Enterprise Identity Mapping*.
- 11. Click on the + next to *Domain Management*.
- 12. Right click over the *EIM domain for DEMOS* and select *Connect* from the context menu.
- 13. On the “*connect to EIM domain controller*” screen, enter **secret** for the password and click *OK*.
- 14. Right click over the *EIM domain for DEMOS* and select *Authority...* from the context menu.
- 15. On the *edit EIM authority* screen:
 - Set the *user type* field to ***Distinguished name***.
 - Set the distinguished name field to:
cn=prfXXos400lookup,cn=users,dc=opsb,dc=rchland,dc=ibm,dc=com
where xx = workstation number.
 - Click *OK*.



- 16. On the *new edit EIM authority* screen:
 - Check the EIM mapping operations box.
 - Uncheck all other EIM authorities.
 - Click *OK*.

Your dn, *prfxxos400lookup*, is now authorized to EIM lookup operations. The dn is ready for the server application.

Update a Kerberos and EIM enabled application to use the *dn* you just created.

You have been provided a sample Kerberos and EIM enabled application that takes three input parameters:

- A principal name from a Kerberos realm.
- The name of the Kerberos realm
- The password associated with the Kerberos principal

And:

- Issues a kinit for the principal.
- Use EIM to determine the target ID from the Kerberos source.
- Swap to the target profile.
- Issue a command that retrieves the active profile and sends a message with it.
- Swaps back to the the original profile.

In this exercise you will change the application to use your dn to connect to the EIM controller and remove the comments around the EIM calls so you understand the steps necessary to use EIM within an application.

This exercise will be done from the “green screen”.

Open the application source:

- 1. Right click over *s400a* and select *Display Emulator...* From the context menu. Sign onto the iSeries with the user ID **prfXXos400** and password of **pwdXX**.
- 2. From the *command entry* screen, enter the command *wrkmbpdm prfXXos400/qcsrc* and press *enter*.
- 3. Select option **2** (edit) next to member *eimXXapp* and press *enter*.

Application update 1: Put your dn into the application. It will be used to connect to the EIM domain controller:

- 4. On line 48, change "*cn=*"; to *"cn=prfxxos400lookup,cn=users,dc=opsb,dc=rchland,dc=ibm,dc=com"*;
NOTE: Don't forget to change **xx** to your work station ID.
- 5. On the SEU command line, type *save* and press *enter*.

Application update 2: Uncomment the EIM calls.

There are five EIM actions that are required:

1. Retrieve EIM configuration information. (Make sure EIM is available on the machine and get the local EIM registry name.)
2. Create an EIM connection handle.
3. Connect to the EIM domain controller. (Your dn and password are used in this operation)
4. Get the local registry name (*USRPRF name) by calling the EIM get target from source API.
5. Destroy the EIM connection handle.

All of these steps are in the *getOS400User* procedure.

- 6. Find each of the five steps listed above and remove the comments around the function calls. To quickly find the location for each item, search for the string >>>> x, where x is 1 through 5. Also, each line that needs to be removed contains the phrase *remove this line*. The open and close comment marks are */** and **/*.
- 7. After the last update is done, enter *save* on the SEU command line and press *enter*.

Compile and run the application

- 8. From the *edit* screen, press *F21* to bring up a command line.
- 9. Compile the application. Enter the command:
*crtbndc prfXXos400/eimXXapp prfXXos400/qcsrc output(*print)*
and press enter. Remember, replace XX with your work station number.
- 10. If there are no compile errors, call your application. Enter the command
call prfXXos400/eimXXapp ('wsXX' 'IBMROCHESTERMN.DEMOS.COM' 'pwdXX')
Where XX is your workstation number.
- 11. You will get a message that shows that your ID swapped the the **AAxxxxxxx** profile you specified in the “*target association*” for **s400a**.

Additional points of interest in the application:

Kerberos APIs are used to obtain initial credentials for the principal that is passed to this applications. All the Kerberos actions are in the *authenticateKerberos* function. The key actions are:

1. Call *krb5_init_context*. This API reads the Kerberos configuration file to get defaults.
2. Several APIs are called to prepare parameters for the call to the KDC which will obtain the initial credentials.
3. Call *KRB5_BUILD_PRINCIPAL_EXT*. This API builds the service name for the ticket-granting-server on the KDC.

4. Call `krb5_get_in_tkt_with_password`. This is the kinit
Note: kinit is not typically done in the server application.

Conclusion:

Kerberos authentication is widely used in the IT industry. The OS/400 implementation of Kerberos allows users to develop and port applications that require Kerberos authentication. Key functions of OS/400 are enabled to perform authentication using Kerberos service-tickets instead of the traditional user ID and password challenge. EIM allows you to manage who to become once network authentication has been done. The OS/400 implementation of Kerberos followed by the use of EIM allows iSeries to be a key participant in an Intranet or Internet environment.

Appendix A: Purpose and Overview of EIM

Purpose of EIM

When a system or server uses network authentication (i.e. Kerberos tickets) rather than a local ID and password for user authentication, the system or server must determine what local ID to use after a successful authentication. EIM was created to simplify the process of determining who to become after a successful network authentication.

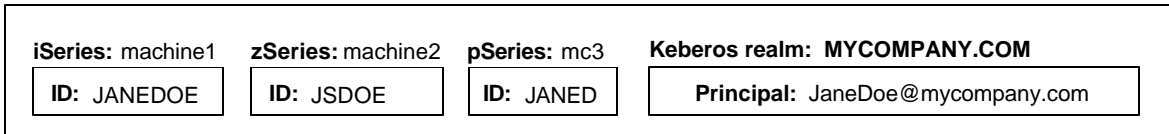
Overview of EIM

In V5R2, IBM introduced Enterprise Identity Mapping (EIM). This is a function that crosses IBM eServer platforms. EIM is a building block function. EIM doesn't do any authentication or authorization checking. EIM is a set of APIs and GUI interfaces that allow an administrator to create and maintain a list of people within an enterprise and keep track of what IDs are associated with the person on various systems in the network. EIM maps user identities between systems. Applications can use the mappings to decide what local ID to use after a network authentication. Below is an example of how the EIM directory could be set up for a user.

A company maintains a computer network with 3 machines, all of which participate in a Kerberos realm. Within the network, an employee has an ID on each system, as well as a Kerberos principal name.

Employee: Jane Doe

Company's network



In this example, you may want to create the following EIM associations for the employee Jane Doe.

List Of EIM Associations for Jane Susan Doe			
EIM Identifier	Registry Name	Registry User Name	Association Type
Jane S. Doe	MYCOMPANY.COM	JaneDoe@mycompany.com	Source
Jane S. Doe	machine1	JANEDOE	Target
Jane S. Doe	machine2	JSDOE	Target
Jane S. Doe	mc3	JANED	Target

The following terms are used by EIM:

- **EIM Association:** An association is an entry that shows what ID is assigned to a particular person or server in a registry. There is also an association type that indicates how the association is to be used.
- **EIM Identifier:** This is a unique value used to identify a particular person or server. An identifier can have any number of associations.

- **Registry Name:** A registry name refers to a place where a list of valid users or servers is kept. For example, a *registry name* may refer to a computer system, a Kerberos realm, or an LDAP directory.
- **Registry User Name:** This is the local user name assigned to a particular person or server for the registry (place) referenced in the association. For example, a *registry user name* may refer to a user profile, or Kerberos principal name, or an LDAP user.
- **Association Type:** This indicates how the association can be used.
 - **Source:** Having the value of *source* implies that authentication will be done with the registry user name. It also implies the source registry user name will be used to find the target identity for other registries (systems).
 - **Target:** Having the value of *target* implies that authentication was done with a different source registry user name and that a mapping from the source to the target registry user name exists. Generally, these mappings are used to decide what local user “to become”.
 - **Source and Target:** This value allows the association to be used for finding other associations and for deciding what local user “to become”.

Once the administrator has created and populated the EIM directory, applications can use EIM APIs to retrieve information about Enterprise users. Applications can use the mapping information to determine with which profile to run a function, or to develop tools to help manage users in a large network.

How EIM Associations Are Used After Kerberos Authentications in iSeries Functions

Some iSeries functions have to be enabled to accept Kerberos service-tickets for authentication. However, after an authentication, what local user should be used while running the job? The Kerberos-enabled functions extract the client principal name from the service-ticket. The client principal name becomes the EIM source registry user name. The Kerberos-enabled function uses the source registry user name to find the target registry user name on the local system.

Building on the example from the EIM overview section, assume iSeries Navigator is configured to use Kerberos authentication. Jane Doe starts iSeries navigator from her PC. A Kerberos service-ticket is obtained and passed to iSeries Navigator. The authentication code determines that a valid service-ticket has been received from *JaneDoe@MYCOMPANY.COM*. The iSeries Navigator code calls the appropriate EIM APIs to retrieve the target association for the iSeries based off of the source association for *JaneDoe@MYCOMPANY.COM*. The registry user name from the target association contains the OS/400 user profile name that will be used for the jobs iSeries Navigator starts for Jane. EIM did not do the authentication and it did not change the attributes of any jobs. EIM was used to determine what profile to use after authentication. This is the same method used for all the iSeries functions that support Kerberos authentication.

For more information on EIM, refer visit the web site:

<http://publib.boulder.ibm.com/series/v5r2/ic2924/index.htm>

then follow the links *Networking->Networking Security->Enterprise Identity Mapping (EIM)*.

Appendix B: Lab Application Source

```
#include <stdio.h>
#include <eim.h>
#include <qsygetph.h>
#include <qwtsetp.h>
#include <qsyrslsph.h>
#include <qusec.h>
#include <gssapi.h>
/*-----*/
/* Purpose: This program was developed for the */
/* */
/* Implementing Single Sign-on COMMON Lab */
/* */
/* This program takes three parameters: */
/* */
/* 1: A Kerberos principal name. */
/* 2: The name of the Kerberos realm. */
/* 3: The password associated with the principal. */
/* */
/* And does the following: */
/* */
/* 1: Calls the kerberos APIs necessary to do kinit to get a */
/* ticket-granting-ticket. */
/* 2: Uses EIM to find a target mapping for the Kerberos user */
/* on the local system. */
/* 3: Swaps to the target user. */
/* 4: Prints a message proving the swap occurred. */
/* 5: Swaps to the original user. */
/* */
/* The LAB purpose: */
/* */
/* Lab attendees will make the following changes to this program: */
/* */
/* 1: Add the appropriate lookup user that is used to connect to */
/* the domain controller. */
/* 2: Un-comment the 5 EIM actions being featured in this */
/* application. */
/* */
/* Points of interest to LAB Attendees may also be the Kerberos */
/* API calls made in this program.
/*****
/*****
/***** LAB ACTION REQUIRED:
/* Update the lookup user value with the dn
/* you've created.
/*****
char * lookupUser =
    "cn=";
char * lookupPassword = "secret";

/*****
/* Function: get OS400 User */
/* */
/* This routine is called AFTER the Kerberos authentication */
/* has occurred. This routine does all the EIM calls to */
/* determine what local OS/400 user is associated with the */
/* Kerberos principal passed into the program. */
/* */
/* Input: Kerberos user */
/* Kerberos registry */
/* */
```

```

/* Output: local OS400 user */
/* User will be responsible for freeing the space. */
/* */
/*-----*/
int getOS400User(char * kerberosUser,
                char * kerberosRegistry,
                char ** OS400User)
{
    int rc;
    char * msg = NULL;

    EimHandle handleData;
    EimHandle * handle;
    EimTargetIdentity * entry;

    char * localRegistry = NULL;

    /*-----*/
    /* Reuse space for both EIM configuration as well as list data */
    /* retrieved. */
    /*-----*/
    char listData[4000];
    EimConfig * cfgList = (EimConfig * ) listData;
    EimList * list = (EimList * ) listData;

    /*-----*/
    /* Connection information for the eimConnect API. */
    /*-----*/
    EimConnectInfo con;

    /*-----*/
    /* EIM error code used on all APIs. */
    /*-----*/
    char eimerr[100];
    EimRC * err;

    memset(eimerr, 0x00, 100);
    err = (EimRC *)eimerr;
    err->memoryProvidedByCaller = 100;

    /*-----*/
    /* >>>1 LAB ACTION REQUIRED: */
    /* Remove the comments around eimRetrieveConfiguration. */
    /* */
    /* Retrieve the local OS400 registry name EIM has for this */
    /* machine. */
    /*-----*/
    memset(cfgList,0x00,4000);
    /* Remove this line.
    if (0 != (rc = eimRetrieveConfiguration(4000,
                                          cfgList,
                                          0,
                                          err)))
    {
        msg = eimErr2String(err);
        printf("%s\n",msg);
        free(msg);
        return -1;
    }
    Remove this line. */

    if (0 == cfgList->localRegistry.length)
    {
        printf("Local registry not configured.\n");

```

```

    return -1;
}

/*-----*/
/* Extract the registry name. */
/*-----*/
localRegistry = (char *)malloc(cfgList->localRegistry.length + 1);
memset(localRegistry, 0x00, cfgList->localRegistry.length + 1);
memcpy(localRegistry,
        (char*)cfgList + cfgList->localRegistry.disp,
        cfgList->localRegistry.length);

/*****
/* >>>2 LAB ACTION REQUIRED: */
/* Remove the comments around eimCreateHandle. */
/* */
/* To connect to an EIM domain controller, you must have a handle.*/
/* The handle is needed on many EIM APIs. */
/*-----*/
handle = &handleData;
/* Remove this line.
if (0 != (rc = eimCreateHandle(handle,
                             NULL,
                             err)))
{
    msg = eimErr2String(err);
    printf("%s\n",msg);
    free(msg);
    free(localRegistry);
    return -1;
}
Remove this line */

/*****
/* >>>3 LAB ACTION REQUIRED: */
/* Remove the comments around eimConnect. */
/* */
/* Connect to the EIM domain controller. */
/* */
/* EIM needs to connect to ldap with a user that has the authority*/
/* to be able do mapping lookup operations. The dn you created */
/* will be used for the connect. */
/*-----*/
con.type = EIM_SIMPLE;
con.creds.simpleCreds.protect = EIM_PROTECT_NO;
con.creds.simpleCreds.bindDn = lookupUser;
con.creds.simpleCreds.bindPw = lookupPassword;
con.ssl = NULL;
/* Remove this line.
if (0 != (rc = eimConnect(handle,
                          con,
                          err)))
{
    msg = eimErr2String(err);
    printf("%s\n",msg);
    free(msg);
    free(localRegistry);
    eimDestroyHandle(handle,
                     NULL);
    return -1;
}
Remove this line */

/*****
/* >>>4 LAB ACTION REQUIRED: */

```

```

/*      Remove the comments around eimGetTargetFromSource.      */
/*      */
/* Finally, the moment we've been waiting for.  Get the target  */
/* registry user (the local *USRPRF) for the Kerberos principal. */
/*-----*/
memset(list,0x00,4000);
/* Remove this line.
if (0 != (rc = eimGetTargetFromSource(handle,
                                     kerberosRegistry,
                                     kerberosUser,
                                     localRegistry,
                                     NULL,
                                     4000,
                                     list,
                                     err)))

{
    msg = eimErr2String(err);
    printf("%s\n",msg);
    free(msg);
    free(localRegistry);
    eimDestroyHandle(handle,
                     NULL);

    return -1;
}
Remove this line */

/*****
/* >>>>5 LAB ACTION REQUIRED:      */
/*      Remove the comments around eimDestroyHandle.      */
/*      */
/* No more EIM calls will be made in this program.  Destroy the  */
/* EIM connection handle.      */
/*-----*/
if (list->entriesReturned > 1)
{
    printf("Multiple target users found\n");
    free(localRegistry);
/* Remove this line.
    eimDestroyHandle(handle,
                     NULL);

    return -1;
Remove this line */
}

/*****
/* >>>> Point of interest:      */
/*      */
/* Extract the local *USRPRF name from the data returned from  */
/* eimGetTargetFromSource      */
/*-----*/
entry = (EimTargetIdentity *)((char *)list + list->firstEntry);

*OS400User = (char *)malloc(entry->userName.length + 1);
memset(*OS400User, 0x00, entry->userName.length + 1);
memcpy(*OS400User,
      (char*)entry + entry->userName.disp,
      entry->userName.length);

free(localRegistry);
eimDestroyHandle(handle,
                 NULL);

return 0;

```

```
}
```

```

/*****
/* Function:  Authenticate Kerberos.
/*
/*      This routine issues the kinit for the Kerberos principal
/*      supplied on the program call.
/*
/* Input:   Kerberos user
/*          Kerberos registry
/*
/* Output:  local OS400 user
/*          User will be responsible for freeing the space.
/*
/*-----*/
int authenticateKerberos(char * pPrincipal,
                        char ** pRealm,
                        char * pPassword)
{
    krb5_context    krb_context = NULL; /* Kerberos context */
    krb5_error_code retval;           /* Kerberos API retval */
    krb5_principal  krb5_princ = NULL; /* Kerberos principal */
    krb5_creds      creds;            /* Kerberos credentials */
    krb5_ccache     memcache;        /* Place to store credentials
    */
    char            *pQualifiedPrincipal; /* Fully qualified
    principal */
    int             nQualifiedPrincipalLen;

    /*****
    /* >>>> Point of interest:
    /*
    /* Reads the Kerberos configuration file to get default values.
    /*-----*/
    krb5_init_context(&krb_context);

    /* If realm not specified, assume the default. */
    if (NULL == *pRealm)
        krb5_get_default_realm(krb_context, pRealm);

    /*****
    /* >>>> Point of interest:
    /*
    /* Need to build principal name in acceptable format.
    /*-----*/

    /* Set up the qualified principal name, ie gjs@IBMROCHESTERMN.DEMOS.COM.
    nQualifiedPrincipalLen = strlen(pPrincipal) + strlen(*pRealm) + 2;
    pQualifiedPrincipal = (char *)malloc(nQualifiedPrincipalLen);
    memset(pQualifiedPrincipal, 0x00, nQualifiedPrincipalLen);
    strcpy(pQualifiedPrincipal,pPrincipal);
    strcat(pQualifiedPrincipal, "@");
    strcat(pQualifiedPrincipal, *pRealm);

    /* Convert the string to a Kerberos principal. */
    krb5_parse_name(krb_context,
                   pQualifiedPrincipal,
                   &krb5_princ);

    /*****
    /* >>>> Point of interest:
    /*
    /* For this program, caching the credential in storage, not a
    /* credentials file. Get storage and initialize.
    */

```

```

/*-----*/

/* Generate a new memory based credentials cache. */
krb5_cc_generate_new(krb_context, "MEMORY", &memcache);

/* Initialize the credentials cache created above. */
krb5_cc_initialize(krb_context, memcache, krb5_princ);

/* Initialize the creds structure. */
memset(&creds, 0, sizeof(creds));
creds.client = krb5_princ;

/*****
/* >>> Point of interest: */
/*
/* Before the kinit, need the name of the ticket-granting-server */
/* for the KDC.
/*-----*/

/* We need to authenticated, so build the principal name for the
TicketGrantingServer on the KDC */
krb5_build_principal_ext(krb_context,
                        &creds.server,
                        strlen(*pRealm),
                        *pRealm,
                        KRB5_TGS_NAME_SIZE,
                        KRB5_TGS_NAME,
                        strlen(*pRealm),
                        *pRealm,
                        0);

if (0 !=
/*****
/* >>> Point of interest: */
/*
/* Issue the kinit.
/*-----*/
    (retval =
        krb5_get_in_tkt_with_password(krb_context,
                                      0, /* nothing fancy,
                                      just a TGT */
                                      NULL, /* addresses to be
                                      put in ticket */
                                      NULL, /* enctype, use
                                      default */
                                      NULL,
                                      pPassword, /* password */
                                      memcache, /* handle to place
                                      TGT */
                                      &creds, /* credential
                                      structure */
                                      NULL))) /* ignore the KDC
                                      reply */

{
    printf("\nAuthentication error, verify input. Return code = %x"
           ,retval);
    krb5_cc_destroy(krb_context, memcache);
    krb5_free_context(krb_context);
    free(pQualifiedPrincipal);
    return -1;
}

krb5_cc_destroy(krb_context, memcache);
krb5_free_context(krb_context);
free(pQualifiedPrincipal);
return 0;

```



```

}

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

    char * localOS400User;          /* Pointer to local profile*/

    char  profileHandle[12];       /* Handle for profile swap */
    char  currentHandle[12];      /* Handle for profile swap */

    char          *pPrincipal,     /* Kerberos principal */
                *pRealm,         /* Kerberos Realm name*/
                *pPassword;       /* Principal password */

    Qus_EC_t errorCode;
    errorCode.Bytes_Provided = 0;

    /*-----*/
    /* Start of executable code.                */
    /*-----*/
    /*
    Basic checking on input parameters.          */
    if (argc == 3) {
        pPrincipal = argv[1];
        pPassword  = argv[2];
        pRealm = NULL; /* Use default realm name. */
    }
    else if (argc == 4) { /* All values specified. */
        pPrincipal = argv[1];
        pRealm     = argv[2];
        pPassword  = argv[3];
    }
    else {
        printf(
"\n\nUsage: Call this program with: \n");
        printf("    parm('Principal', 'Password') or \n");
        printf("    parm('Principal', 'Realm' 'Password')\n");
        return 0;
    }

    /*-----*/
    /* Call routine to issue the kinit.          */
    /*-----*/
    if (0 != (rc = authenticateKerberos(pPrincipal,
                                        &pRealm,
                                        pPassword)))
    {
        printf("ERROR: See previous messages.\n");
        return -1;
    }

    /*-----*/
    /* Call routine that uses EIM to determine which local profile */
    /* to use.                                                    */
    /*-----*/
    if (0 != (rc = getOS400User(pPrincipal,
                               pRealm,
                               &localOS400User)))
    {
        printf("ERROR: See previous messages.\n");
        return -1;
    }
}

```

```

/*-----*/
/* Swap to the profile EIM said to use. */
/*-----*/
QSYGETPH("CURRENT ",
        "NOPWD ",
        currentHandle); /* Get current handle. */
QSYGETPH(localOS400User,
        "NOPWD ",
        profileHandle); /* Get target profile handle. */
free(localOS400User);

QWTSETP(profileHandle); /* Swap to the target profile. */
system("CALL SETUPEIMDE/DSPCURUSR"); /* Print msg with CURRENT user*/
QWTSETP(currentHandle); /* Swap back to original user. */

QSYRLSPH(profileHandle); /* release profile handles. */
QSYRLSPH(currentHandle);

return 0;
}

```

Trademarks and Disclaimers

Copyright International Business Machines Corporation 2002

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

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

AS/400, IBM Logo, AS/400e, iSeries, zSeries, pSeries, e-business logo, OS/400, IBM

Lotus, Freelance, and Word Pro are trademarks of Lotus Development Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Internet Explorer, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Sun, Java, JSP, and JavaServerPages are trademarks of Sun Microsystems in the United States, other countries, or both.

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

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information in this presentation concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of the specific Statement of Direction.

Some information in this presentation addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.

END OF DOCUMENT