PROGRESS EXCHANGE 2013

DISCOVER. DEVELOP. DELIVER.

OpenEdge Security

PROGRESS

Table of Contents

Purpose

This document provides the labs for the Exchange 2013 OpenEdge Security Workshop step-by-step.

Disclaimer

This document is not a manual. It merely gives you an overview of how to enable different security aspects in an OpenEdge environment without claiming to be complete. There are different options and paths that can be taken. More information can be found regading Security on Progress Communities (http://communities.progress.com). Progress Software cannot be held responsible for the content of this document nor for any damage that may occur to your environment.

OpenEdge Security - Introduction

Welcome to the Security Workshop. We have a very busy 3 hours set up for you. Hopefully you will be learn something new and understand better that Security, although complex, can be broken down and managed appropriately. You will learn about several different strategies to help make your OpenEdge application more secure.

A few housekeeping items before we begin:

- Have fun! These Arcade instances are set up for you to have an environment where you can play and test functionality in a place where you don't have to "worry" about breaking something. If you do break something we can always reset it or give you a new environment to play in.
- This lab will not have any scheduled breaks. Please feel free to get up and take a break at any time. There are currently 5 separate labs scheduled throughout the workshop.
- All of the Progress employees are here to make sure you are successful. Please ask questions at any time if you are stuck or do not understand a process.

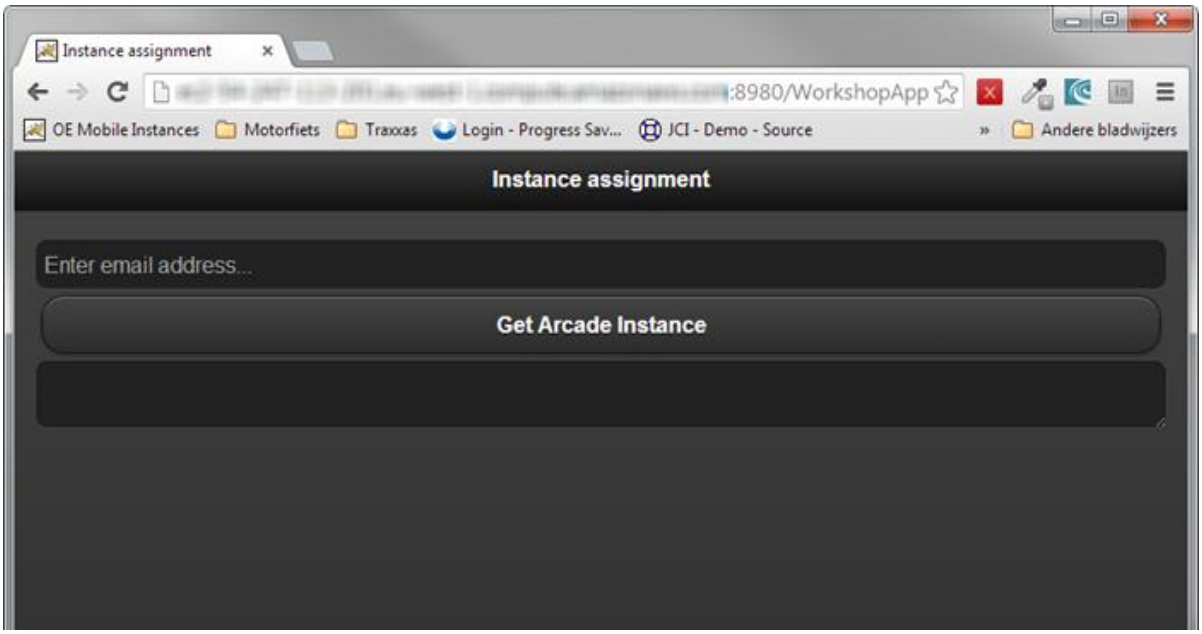
Here is a rough agenda for the day. This agenda and times may change and depending on the speed and skill set of the audience.

Topic	Type	Presenter	Approx. Length (Minutes)
Introductions / Opening	Lecture	Brian Bowman	10
A Wide Open World	Lab		10
Data In Motion	Lecture	Steve Boucher	15
Enabling SSL	Lab		15
Application Protection	Lecture	Rob Marshall	15
Client Principle	Lab		30
External Security	Lecture	Brian & Roy Ellis	10
LDAP Authentication	Lab		15
Physical Security	Lecture	Brian & Roy	10
TDE	Lab		15
Misc. Topics	Lecture	Rob & Brian	10
Tying it all together	Demo	Peter Judge	30
Total Time			170

OpenEdge Security Lab 0 – Getting Your Login

- Open a browser (Google Chrome, Mozilla Firefox or Apple Safari)
- For this workshop:
  - Morning, use this URL http://23.23.210.136:8980/WorkshopApp
  - Afternoon, use this URL http://54.225.237.144:8980/WorkshopApp
- Enter your email address
- Press the button: Get Arcade Instance

This will give you a DNS for a running Arcade instance, which you can use to connect an RDP session.



OpenEdge Security Lab 1 – A Wide Open World

Objective

The objective of this lab is to make sure you are set up and configured to be successful throughout the rest of the workshop. You will also explore your environment looking at how easy it is to access sensitive information in a typical environment.

Duration

This lab should take approximately 10 minutes to complete.

Goals

In this lab you will:

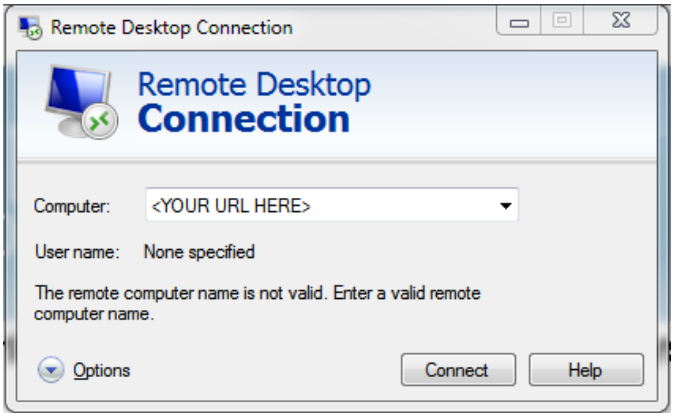
- Successfully log into your own Amazon Instance and set up your environment
- Search a database extent for customer names.
- Capture security information over the wire.

OpenEdge Security Lab 1a – Logging into the System

Instructions

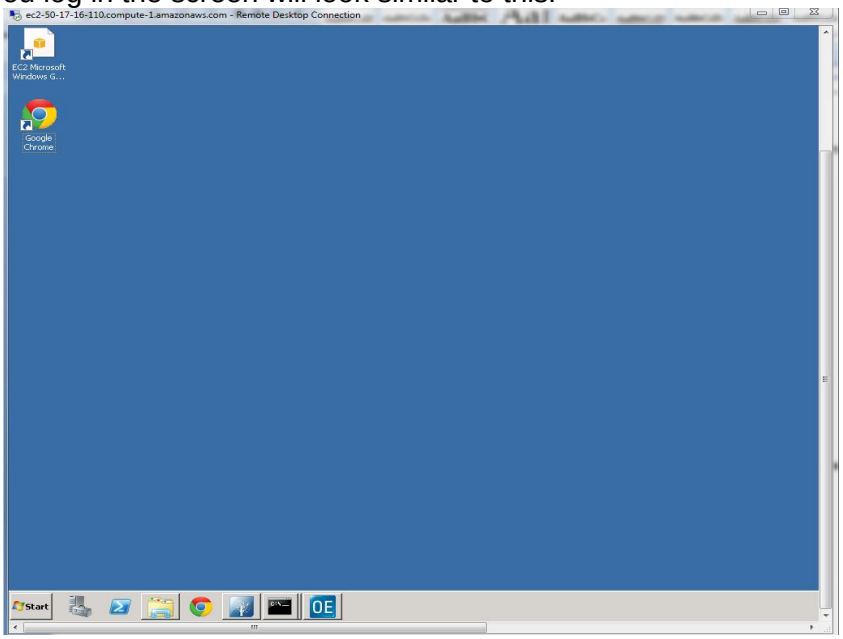
Follow these steps.

1. Connection: Connect to your Arcade instance that has been assigned to you via Windows Remote Desktop Connection (RDC):



When prompted for a login, enter \Administrator. The password is Exchange2013.

When you log in the screen will look similar to this:



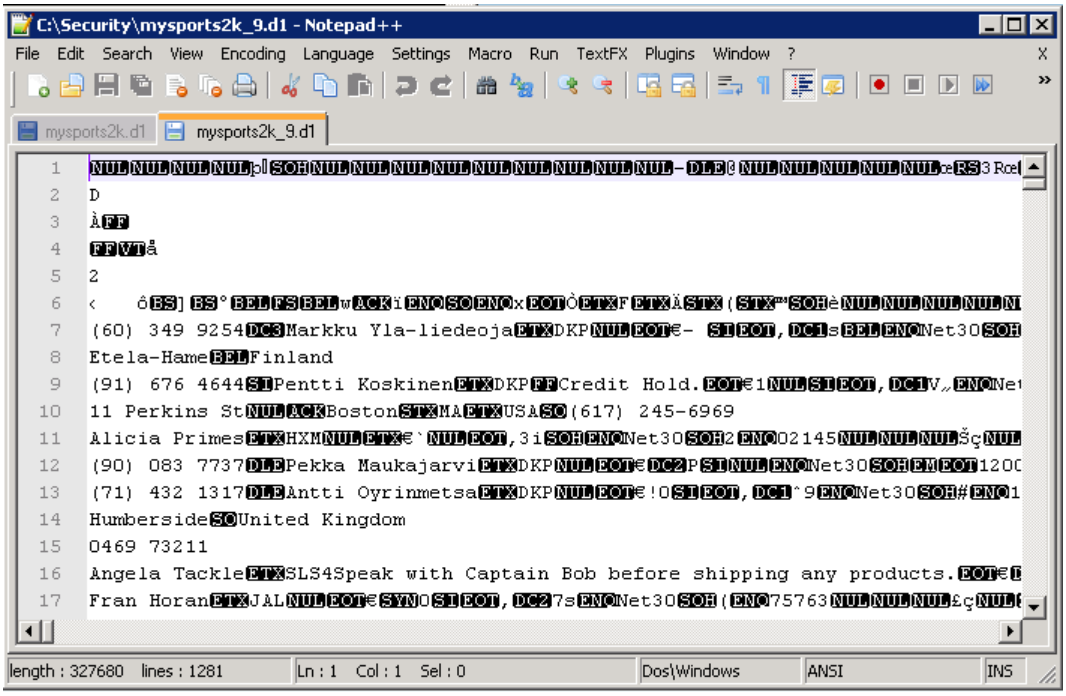
This is the environment that you will be working in for the rest of the workshop.

OpenEdge Security Lab 1b – Looking at Security Problems

Instructions

Open up the Windows explorer and navigate to the c:\Security\ directory. This directory has a database (mysports2k.*) in it for this exercise. Let's take a look at the data in the extent. To do this open the file *mysports2k_9.d1* with the editor provided to you (notepad++).

The editor should look similar to this:



As you can see all of the data is clearly easily viewable and also editable. Editing this file would probably corrupt it and force you to have to move to a backup.

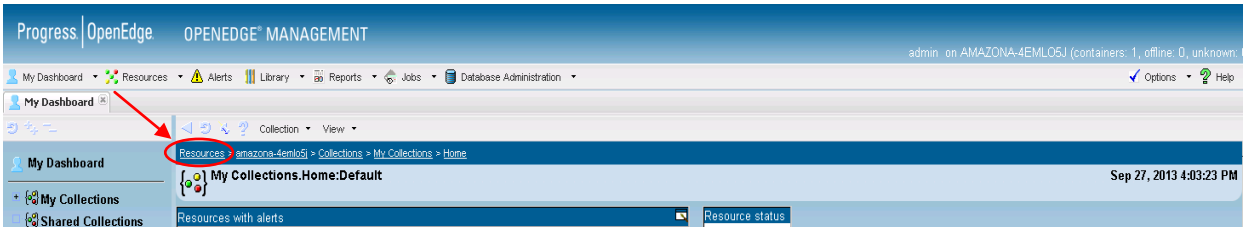
OpenEdge Security Lab 1c – Clear Text Data

Instructions

Follow these steps.

1. Launch the OpenEdge Management Console:
   Start Menu -> All Programs -> Progress -> OpenEdge 11.3 -> Management Console.
   Login prompt: username is “admin” and the password is “admin”.

2. Start the mysports2k database:
   Click the Resources link to access the list of available resources.



   In the Resource page under Database, click on <hostname>.mysports2k.
   The “Database:mysports2k” page is displayed, click Control and then click Start Database.
   Verify the database started successfully by clicking on mysports2k in: “Database: mysports2k” and confirming that the Database status is “Running”.

3. Start the AppServer asbrokerSL:
   Return to Resources view by clicking on the Resources link in “Resources -> Local -> mysports2k”.
   In the Resource page under AppServer, click on <hostname>.asbrokerSL.
   The “AppServer:asbrokerSL” page displays. Click Broker Control and then click Start AppServer.
   Verify the AppServer started successfully by checking that the Broker summary status is “Active”.
   Note that the broker port is 3092.

4. Start SmartSniff, a freeware TCP/IP packet capture tool from NirSoft:
   Start button -> All Programs -> NirSoft SmartSniff -> SmartSniff.
   Start capturing packets: File -> Start Capture.

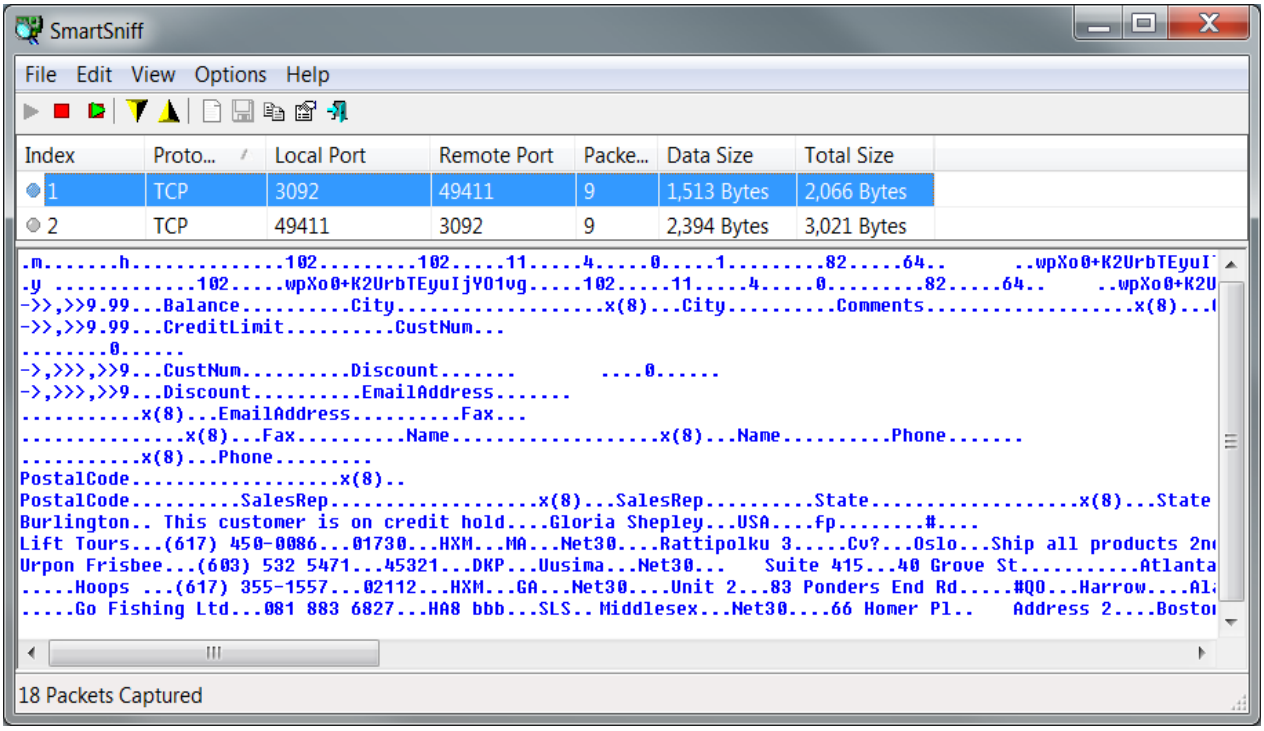
5. Start an ABL client and run the LabClient.p application:
   Start button -> All Programs -> Progress -> OpenEdge 11.3 -> Client.
   Then File -> Open, select LabClient.p -> Open button.
   Compile -> Run.
   Nothing is displayed. The LabClient.p makes an AppServer call to download a temp-table containing 5 customer records.

6. Switch back to SmartSniff to view the captured TCP/IP packets:
   Stop capturing packets: File -> Stop Capture.
   Click on the row where “Local Port” is 3092 to reveal readable customer data in ASCII text.

OpenEdge Security Lab 1c – Clear Text Data (cont.)

Instructions

The customer data transmitted through the network between the client and the AppServer is easy to intercept and read.



This completes the lab.....

NOTES

OpenEdge Security Lab 2 – Enabling SSL

Objective

This lab provides steps to walk the user through the OpenEdge Security and enabling SSL.

Duration

This lab should take approximately 15 minutes to complete.

Goals

In this lab you:

- Configure the database, AppServer and ABL client to connect using SSL
- Use the SmartSniff tool to capture the network traffic transmitted between the client and AppServer/Database
- Run an AppServer application that transmits customer data over the network
- Inspect the captured network traffic and observe that the customer data is encrypted and not human readable

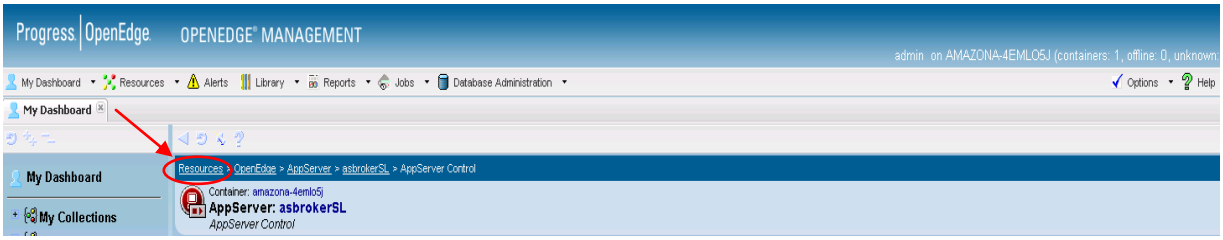
OpenEdge Security Lab 2 – Enabling SSL

Instructions

Follow these steps.

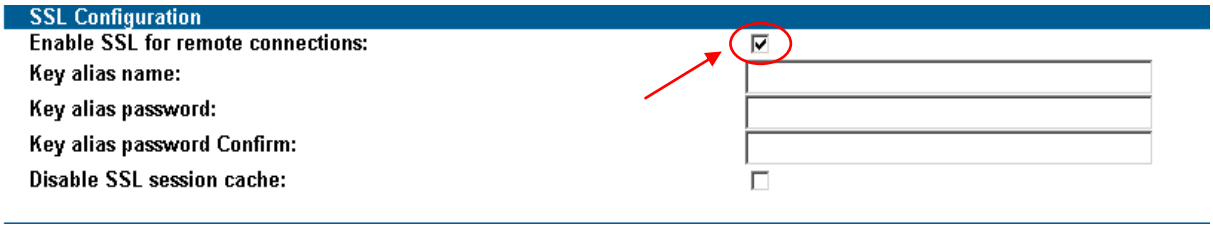
1. If not already running, launch the OpenEdge Management Console:
   Start button -> All Programs -> Progress -> OpenEdge 11.3 -> Management Console.
   Login prompt: username is “admin” and the password is “admin”.

2. Stop the mysports2k database:
   Click the Resource link to return to the list of available resources.



   In the Resource page under Database, click on <hostname>.mysports2k.
   The “Database:mysports2k” page is displayed, click Control -> Stop Database.
   Verify that the database stopped successfully by clicking on the mysports2k link in: “Database: mysports2k” and confirm that the Database status is “Not Running”.

3. Reconfigure the mysports2k database to start with SSL:
   Under Command and Control, click Configuration then under the “Configuration and Server Group Links” click on the “configuration.mysports2k.defaultconfiguration” link.
   Click the “Edit” button and scroll to the bottom of the page.
   Under “SSL Configuration” put a check in the “Enable SSL for remote connections” checkbox.



   Scroll back to the top and click the save button.
   Return to the main Database:mysports2k page by clicking on the mysports2k link in: “Database:mysports2k”.

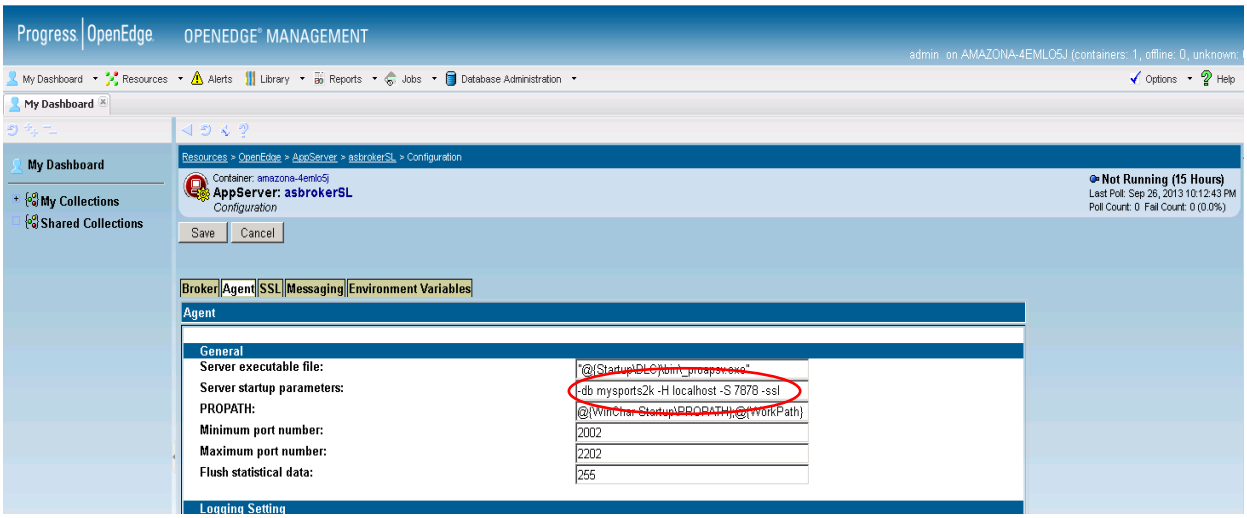
4. Restart the mysports2k database:
   Click Control and then click Start Database.

Wait a second for the command to complete and then verify that the database started successfully by clicking on the mysports2k link in: “Database: mysports2k” and confirming that the Database status is “Running”.

5. Stop the AppServer asbrokerSL:
   Return to Resources view by clicking on the Resources link in “Resources -> Local -> mysports2k”.
   In the Resource page under AppServer, click on <hostname>.asbrokerSL.
   The AppServer:asbrokerSL page is displayed, click Broker Control -> Stop AppServer.

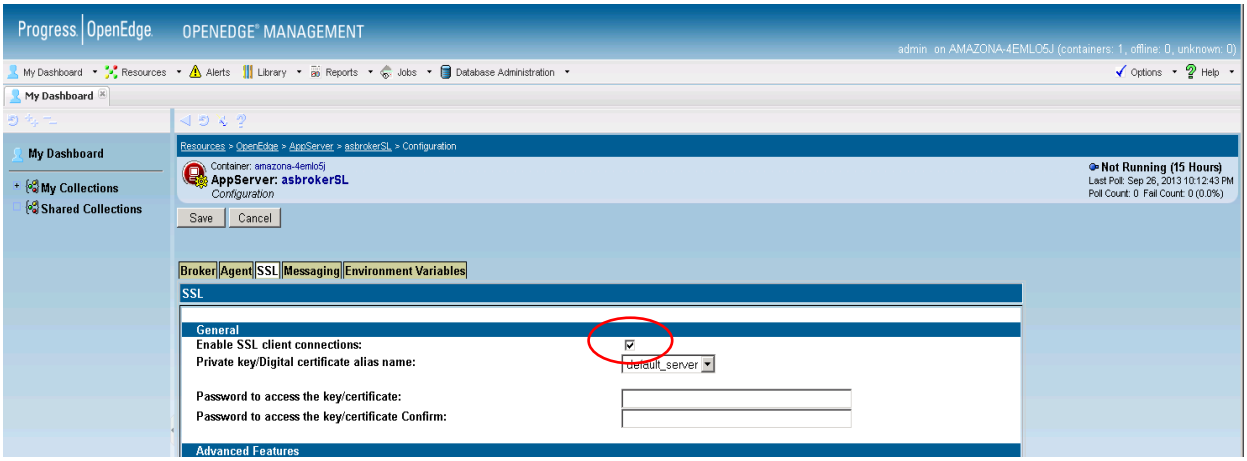
6. Configure SSL for connections between the AppServer and database:
   Return to the main AppServer:asbrokerSL page by clicking on the asbrokerSL link in: “AppServer:asbrokerSL”.
   Under Command and Control, click Configuration.
   Click the Edit button and select the Agent tab.
   Under the General section, edit the “Server startup parameters:” and add “–ssl” to the end.
   The property value is now “-db mysports2k –H localhost –S 7878 –ssl”.
   The AppServer will now connect to the database using SSL.

   The connection string to the AppServer will look like this:



7. Configure SSL for connections between the client and the AppServer:
   Select the SSL tab and put a check in the “Enable SSL client connections:” checkbox.
   Click the Save button and the AppServer will now only accept SSL client connections.

After checking the checkbox in OpenEdge Management the screen will look like this:



8. Restart the AppServer broker asbrokerSL:
   Return to the main AppServer:asbrokerSL page by clicking on the asbrokerSL link in: "AppServer:asbrokerSL".
   Under Command and Control, click Broker Control and then click Start AppServer.
   Confirm that the Appserver started successfully by watching the Broker summary Status change from "Not Running" to "ACTIVE".
   Note that the broker port is 3092.

9. If not already running, start SmartSniff:
   Start button -> All Programs -> NirSoft SmartSniff -> SmartSniff.
   Start capturing packets: File -> Start Capture.

10. If not already running, start an ABL client then open the LabClient.p application.
    Start button -> All Programs -> Progress -> OpenEdge 11.3 -> Client.
    Then File -> Open, select LabClient.p and click the Open button.

11. Modify the connection parameters and run:
    Locate the "hdl:CONNECT("-S 5162 –H localhost –AppService asbrokerSL") line.
    Append to the end of the connection string "-ssl" so that the new Connect statement is:

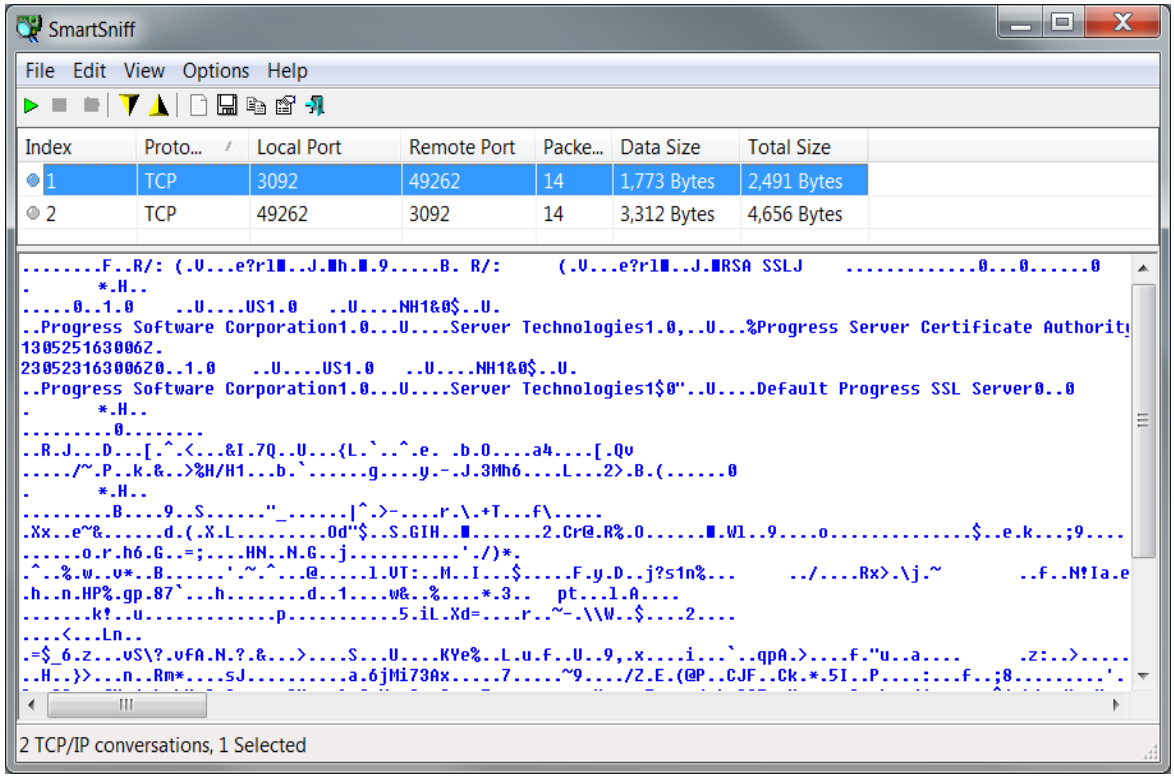
    hdl:CONNECT("-S 5162 –H localhost –AppService asbrokerSL –ssl").

    Run the LabClient.p by clicking Compile -> Run.
    Nothing is displayed. The LabClient.p makes an AppServer call to download a temp-table containing 5 customer records.

Instructions

12. Use SmartSniff to view the captured TCP/IP packets:
    Stop capturing packets: File -> Stop Capture.
    Click on the row where “Local Port” is 3092 to reveal that the customer data is encrypted.

    The customer data transmitted through the network between the client and the AppServer is easy to intercept, but with encryption, the customer data is no longer readable. Only the some of the SSL certificate information is discernable.



This completes the lab.....

NOTES

OpenEdge Security Lab 3 – Client Principal

Objective

This lab provides steps to create a Basic Client-Principal object as well as the supporting domains and authentication systems

Duration

This lab should take approximately 15 minutes to complete.

Goals

In this lab you:

- Create an 'exchange' Security Domain using the _oeusertable authentication system
- Authenticate a Client-Principal using the 'exchange' domain
- Create a new Authentication System for use with the 'exchange' domain

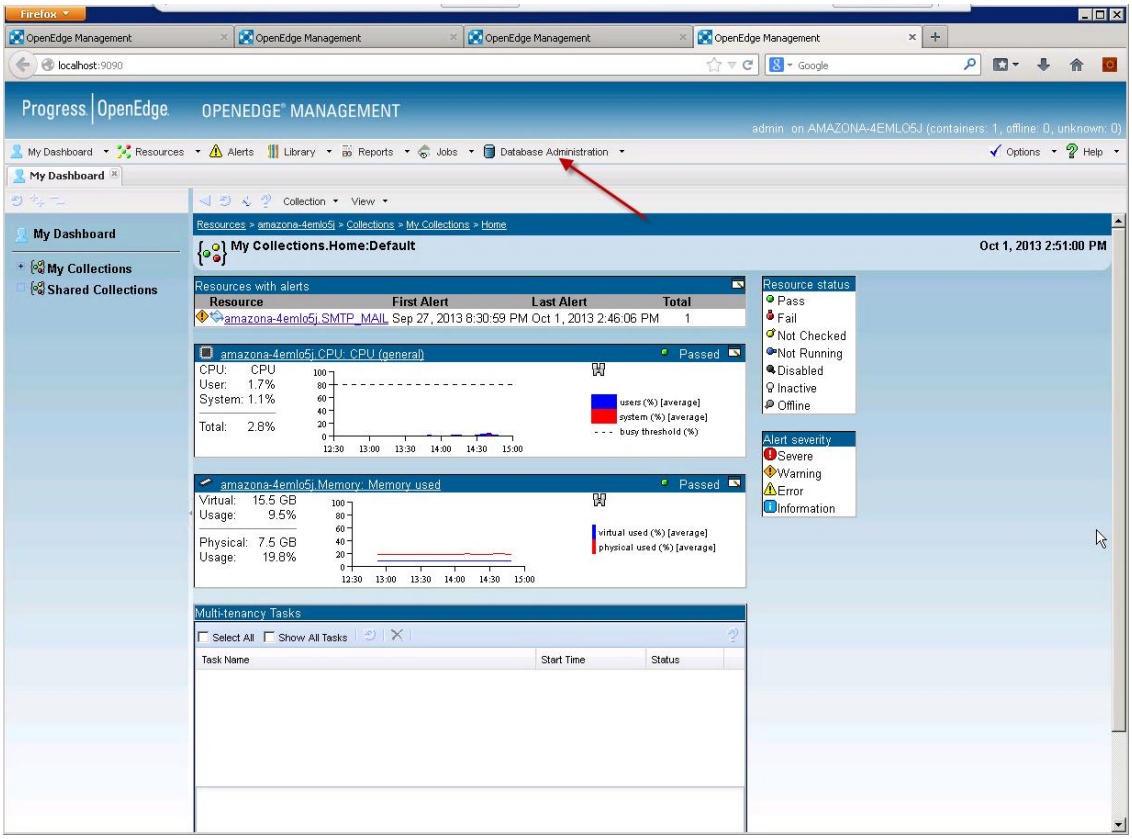
OpenEdge Security Lab 3a – Creating the 'exchange' Domain

Instructions

Follow these steps.

1. Launch OpenEdge Management and select the Database Administration tab:
   Start button -> All Programs -> Progress -> OpenEdge 113 -> OpenEdge Management

   The screen will look similar to this:



2. Select the **Identity** database for administration
   Select the Identity database from the Connections selection list on the left-hand side. The database name may be prefixed with the container name. It may look like amazona-4emlo5k.Identity, with the container name depending on the machine's name.

Instructions

3. Add the domain
   Navigate to the Domain administration page via the **Edit domains** link in the **Security Summary** section. Alternatively, select the Domains item in the selection list on the left-hand side.

   Select the **New** button and add the new domain with the following mandatory values. The other values are optional.

Field	Value	Notes
Domain Name	exchange	Make sure the name is lower-case
Authentication System	_oeusertable	
Access code	<any value>	In a production system, you would add a non-guessable, unique value, much like you would choose a (good) password.
Enabled	<checked>	

   Select the **Save** button when you're done.

4. Add users to the domain
   Navigate back to the **Security Summary** page. Select the **< amazona-4emlo5k.Identity** link next to the Save button on the top toolbar.

   In the Security Summary page, select **Create user**. Click the **New** button and specify the following mandatory values. The other values are optional.

Field	Value	Notes
User name	fred	
Domain name	Select **exchange** from the lookup ( )	
Password	LetMeIn	

   Click **Save** when you're done. You can create other users, but we only need fred for the following labs. **Note**: There's a program in the **C:\Security\Lab3\Complete** folder called **lab3a.p** which contains code to programmatically create the "exchange" domain and the "fred" user. This code is provided by way of example.

OpenEdge Security Lab 3b – Authenticating a user in ABL

Instructions

Follow these steps.

1. Launch Progress Developer Studio for OpenEdge using the desktop icon. When asked for a workspace, enter **C:\Security\Workspace** . You will see a project named **Security Lab 3**.

2. Open the procedure called **lab3b.p**.
   This program is an outline of the final program. The completed lab program is in the **Complete** folder.

3. Load the domains into the session
   You should load your domains when your session starts up; typically this would be in your AppServer's startup procedure. It can only happen **once** per session. It's included in this program file for simplicity's sake.

   Add the following line under the **Session Startup** comment.

   ```
   security-policy:load-domains('identity').
   ```

4. Create a CLIENT-PRINCIPAL and pass to the authentication systems
   In the following code, we create a CLIENT-PRINCIPAL and give it the credentials for our user, fred. This is obviously a simplified example, and typically the user's name, domain and password would be passed in to the program via input parameters or a UI.

   The code below creates an unsealed CLIENT-PRINCIPAL, with an expiry in 8 hours' time. You should add it under the **Main Block** comment.

   The INITIALIZE method allows us to pass a bunch of info in; and we can add roles too. See the Help for more details.

   ```
   cUserName   = 'fred'.
   cUserDomain = 'exchange'.
   cPassword   = 'LetMeIn'.
   cRoles      = 'demo'.

   create client-principal hClientPrincipal.
   hClientPrincipal:initialize(
       substitute('&1@&2', cUserName, cUserDomain), /* qualified username */
       guid,   /* unique session id*/
       add-interval(now, 8, 'hours'),  /* default timeout/expiration */
       cPassword).
   hClientPrincipal:roles = cRoles.
   ```

Instructions

Now that we've created the CLIENT-PRINCIPAL, we pass it off to the authentication system. We do so via the SET-CLIENT() method. We call this method after setting the roles.

```
security-policy:set-client(hClientPrincipal).
```

5. Check the state of our CLIENT-PRINCIPAL
   Notice that we have not checked the user's name or password against any values. The ABL runtime knows – via the Domain and Authentication System – where to find the credentials. If authentication passes, the CLIENT-PRINCIPAL will be sealed. If the authentication fails, we can catch the error in the CATCH block at the end of the program.

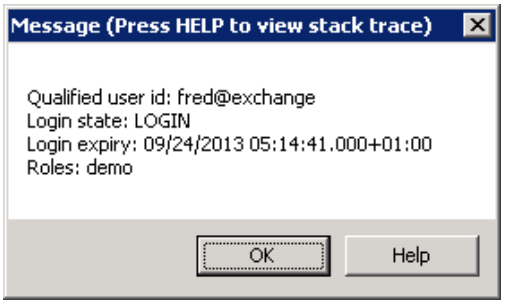
   We can check the state of the CLIENT-PRINCIPAL after login. This code is included here to illustrate how to get the asserted user's CLIENT-PRINCIPAL and query it. It should follow the SET-CLIENT call, although in a real system would probably be called elsewhere in the application.

```
/* Delete and clean up to get the asserted user from anywhere, using GET-
CLIENT() */
delete object hClientPrincipal.
hClientPrincipal = ?.
hClientPrincipal = security-policy:get-client().

message
'Qualified user id:' hClientPrincipal:qualified-user-id skip
'Login state:'       hClientPrincipal:login-state skip
'Login expiry:'      hClientPrincipal:login-expiration-timestamp skip
'Roles:' hClientPrincipal:roles
view-as alert-box.
```

   The application can now go about its business. You can modify program and pass in wrong or different values and see what happens.

6. Run the authentication program
   Run the Lab3b.p program by right-clicking in the editor and selecting **Run As > Progress OpenEdge Application**. If all goes well, you will see a message similar to the below.



OpenEdge Security Lab 3c – Authenticating users with an application database table

Instructions

Follow these steps.

1. Launch Progress Developer Studio for OpenEdge using the desktop icon
   You should see a project named **Security Lab 3**.

2. Inspect the identity table schema
   This lab has added a simple table called **Identity** to the standard Sports2000 database. This table will act as our authentication realm for this lab. The table below outlines the schema.

Field	Notes
IdentityName	The user name. *fred* for example
IdentityDomain	The user's domain: we'll only use *exchange* in this lab
IdentityKey	The password: always *LetMeIn* for this lab
IdentityRoles	User roles; these vary per user. User *fred* has a role of *demo*.

   You can also see the schema in the **identity.df** file in the Lab 3 project.

   The Identity table has already been populated with users from the Customer and Employee tables, and also including our demo **fred** user. See **Complete/load_users.p** for the code used to do this..

3. Add a new Authentication System
   Select the **Identity** database for administration in OE Management, using Step 2 in Lab 3a above.

   In the **Security Summary** screen, select the **Edit authentication systems** link. Click on the **New** link. Add the following mandatory values, and click **Save**.

Field	Value	Notes
name	DBTABLE-Identity	You can give the system any name you want; the value here simply indicates that this is a database table called Identity.
Callback	Complete/IdentityTableAuthentication.p	This is the procedure that performs the authentication.
Enable authentication	<checked>	

OpenEdge Security Lab 3c – Authenticating users with an application database table (cont.)

Instructions

4. Remove users from the database users
   Navigate back to the **Security Summary** page. Select the **< amazona-4emlo5k.Identity** link next to the Save button on the top toolbar.

   Select **Edit users** and delete the **fred** user. You will be unable to associate the new authentication system with the **exchange** domain unless you do this.

5. Update the exchange domain to use the new authentication system.
   Navigate back to the **Security Summary** page. Select the **< amazona-4emlo5k.Identity** link next to the Save button on the top toolbar.

   Select the **Edit domains** link. Select the **exchange** domain, and from the **Authentication system** dropdown, select the new DBTABLE-Identity system. Click **Save** to finish updating the domain.

6. Test the authentication
   To test the new authentication system, run the program from Lab 3b above, following Step 6 in Lab 3b. You should see exactly the same results here as in that lab.

   This highlights the flexibility that using the built-in Domains and Authentication Systems can bring to your security infrastructure: the "application code" (Lab3b.p in this case) does not have to change because we changed domains or authentication systems.

   Note: There's a program in the **C:\Security\Lab3\Complete** folder called **lab3c.p** which contains code to programmatically create the **DBTABLE-Identity** authentication system, delete the **fred** user and update the **exchange** domain. This code is provided by way of example.

This completes the lab.....

NOTES

OpenEdge Security Lab 4 – LDAP Authentication

Objective

This lab provides the basic building blocks for authenticating a user using LDAP. The sample code shows the returns to the screen in MESSAGEs. You would use a return of success (or zero) to authenticate the user.

Duration

This lab should take approximately 15 minutes to complete.

Goals

In this lab you:

- Will start the Apache Directory Studio
- Will connect to the LDAP in the Apache Directory Studio
- Will view the structure of the LDAP directory
- Will run an "ldapsearch" using OpenLDAP to verify it works
- Will open ABL code and review the values
- Will run ABL code to prove you get a successful return

OpenEdge Security Lab 4a – Exploring LDAP

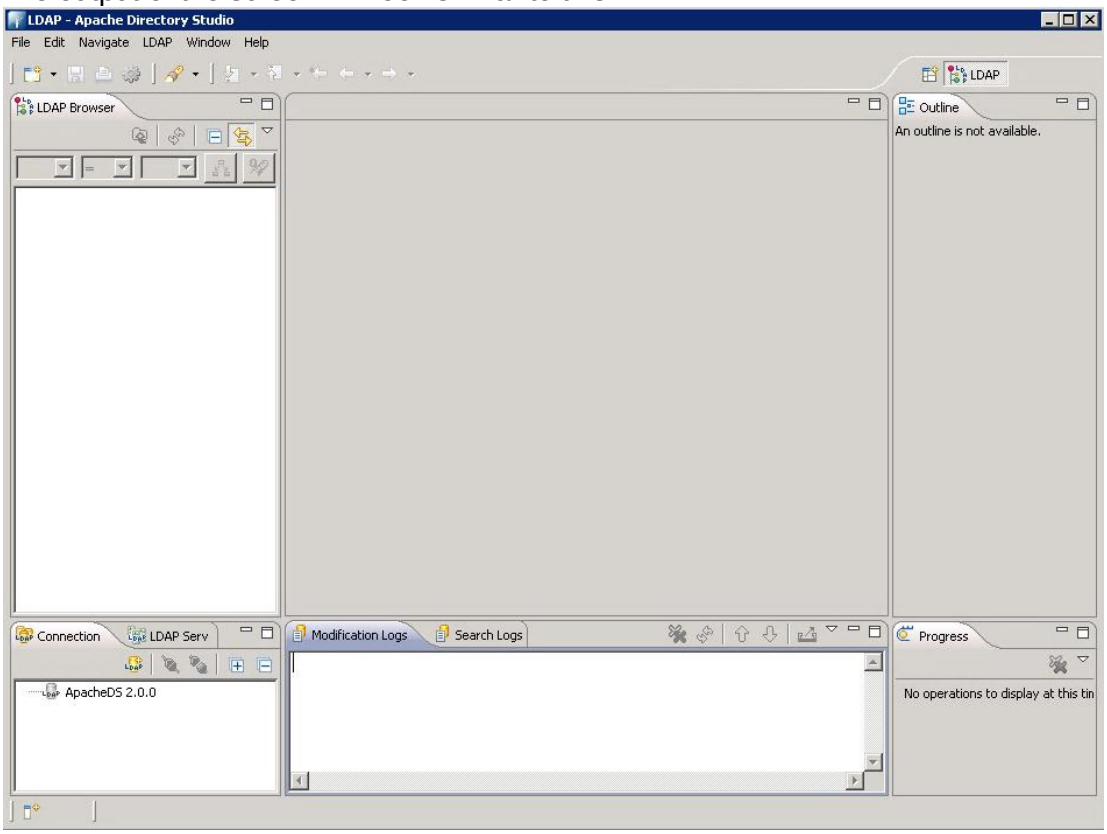
Instructions

Follow these steps.

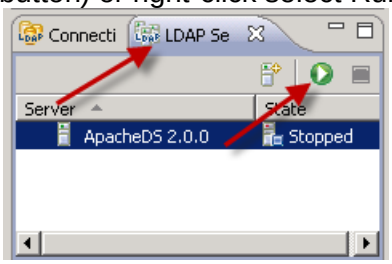
1. Start the Apache Directory Studio. Click the following:

Start → All Programs → Apache Directory Studio → Apache Directory Studio

The output of the screen will look similar to this:



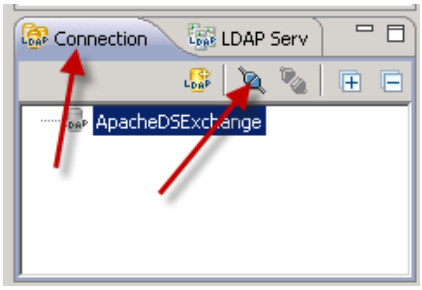
2. In the LDAP Servers pane (lower left) select the Server and click the start button (green play button) or right-click select Run.



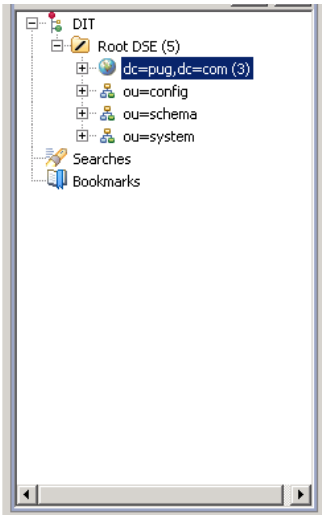
OpenEdge Security Lab 4a – Exploring LDAP (cont.)

Instructions

3. In the Connections pane (lower left) select the connection and click the connect button (2 connected plugs) or right-click select Open Connection



4. Now in the LDAP Browser (top left) double-click dc=pug,dc=com

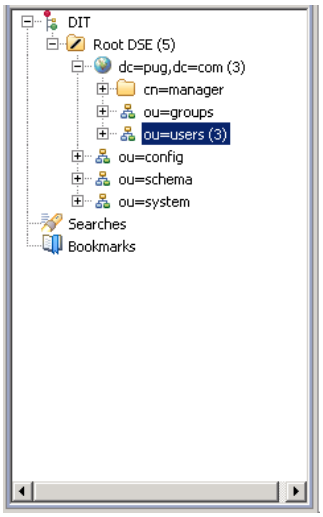


Notice the objects in the main pane.

OpenEdge Security Lab 4a – Exploring LDAP (cont.)

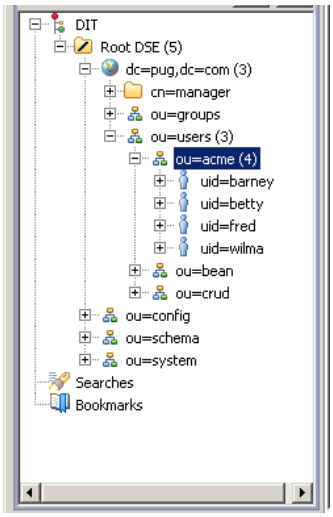
Instructions

5. Now double-click ou=users in that same section.



Again notice the objects in the right pane

6. Lastly double-click ou=acme



Notice the objects in the right pane

This is the structure and objects available in this LDAP service. This knowledge is necessary to correctly connect to and authenticate using LDAP and OpenEdge ABL.

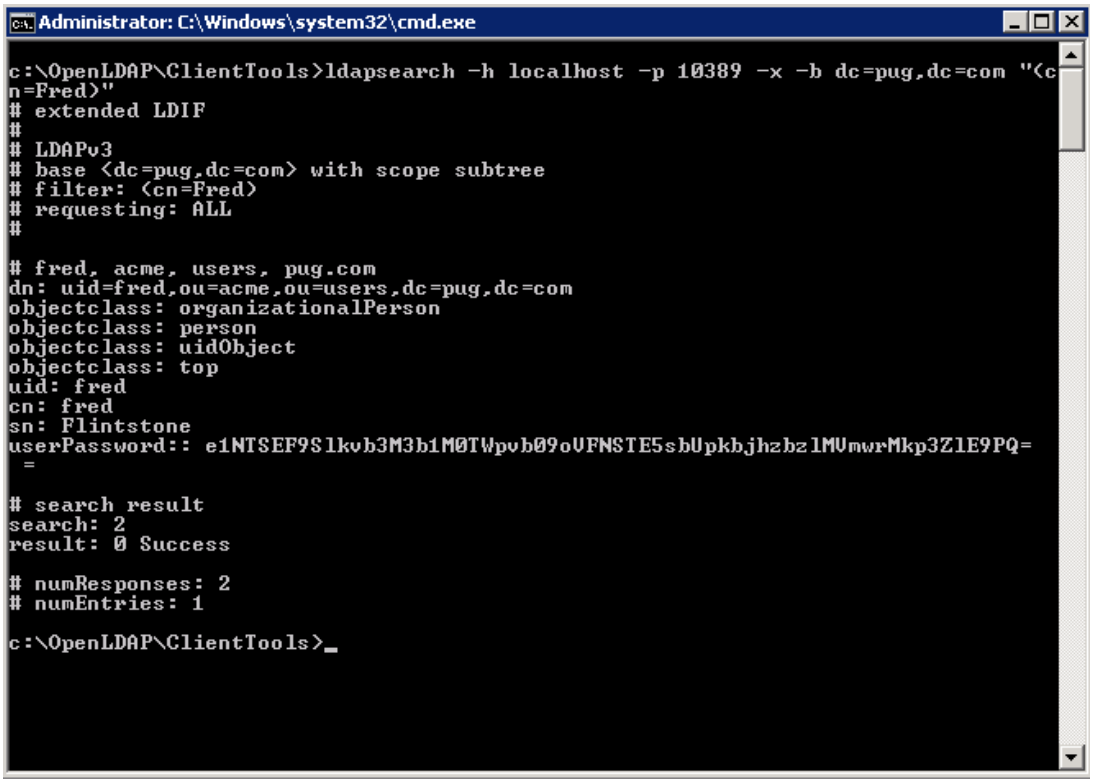
OpenEdge Security Lab 4b – Searching & Using LDAP

Instructions

1. Open a “command prompt” window.
2. Change directory to C:\OpenLDAP\ClientTools
3. Run the following command:

   ldapsearch -h localhost -p 10389 -x -b dc=pug,dc=com "(cn=Fred)"

   The output of the screen will look similar to this:



4. Notice the values returned. Now let's look at more information for setting the ABL parameters correctly.

OpenEdge Security Lab 4b – Searching & Using LDAP (cont.)

Instructions

5. Open an OpenEdge editor - Click the following:

   Start → All Programs → Progress → OpenEdge 11.3 → Client

6. Open the file LDAPAuth.p. This is located in the c:\OpenEdge\wrk (copies are available in C:\Security\LDAP).

   File → Open → LDAPAuth.p

   a. Copy the following variable values from command window returns from "ldapsearch":

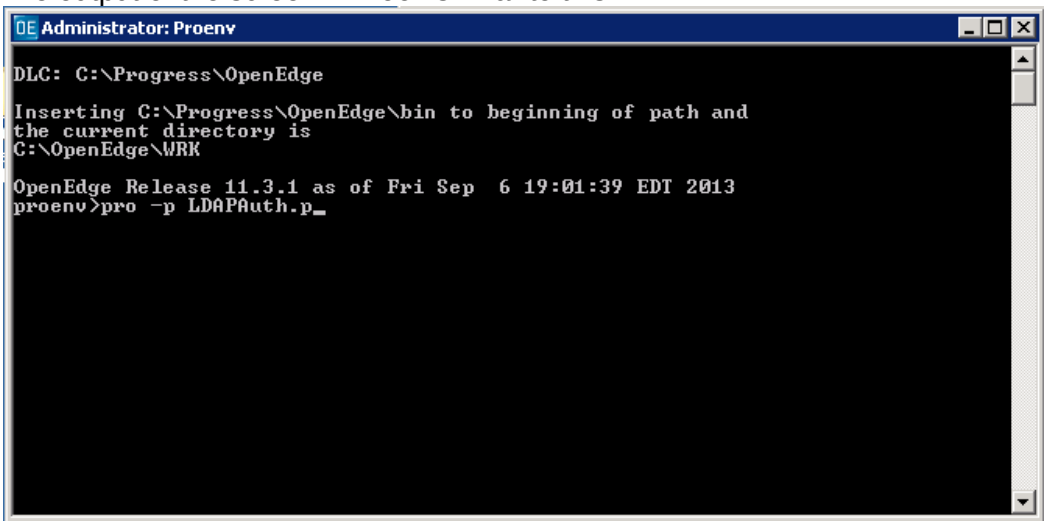
      m_cHomeDN – copy the dn: portion from ldapsearch to here
      m_cSimpleName – copy the cn: portion from ldapsearch to here (done already)
      m_cHomeRoot – copy the dn: portion from ldapsearch, minus uid=fred
      m_cHomeServer – copy the –h value from ldapsearch to here (done already)
      (the –s from ldapsearch is already set in WinLDAPAuth.p called by this program.)

7. Save the LDAPAuth.p (File → Save or F6)
8. Open the ProEnv window. Click the following:

   Start → All Programs → Progress → OpenEdge 11.3 → Proenv

9. run: pro –p LDAPAuth.p

   The output of the screen will look similar to this:

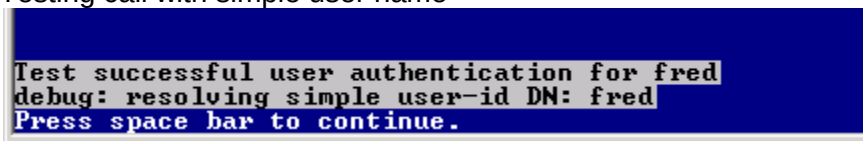


OpenEdge Security Lab 4b – Searching & Using LDAP (cont.)

Instructions

10. Now hit the “space bar” to see these messages:

Testing call with simple user name



Notice returned : 0 – Success!

Now testing the call with full DN

53 Error, why? Because this call requires the password be sent!

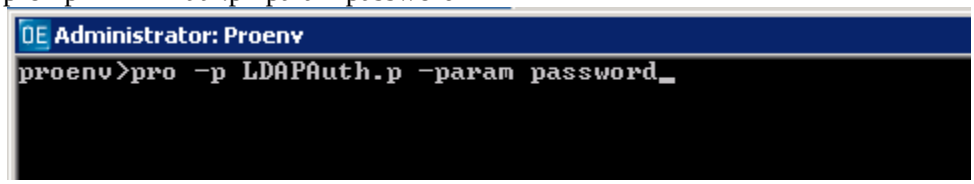
```
debug: Relasing LDAP search result
debug: LDAP simple bind returned  : 53
Press space bar to continue.

User authentication FAILURE: Unwilling To Perform
Running shutdown on LDAP authentication object ...
Procedure complete. Press space bar to continue.
```

OpenEdge Security Lab 4b – Searching & Using LDAP (cont.)

Instructions

11. To see success for the full DN, pass the password (which is password) as a –param to the command line in PROENV
pro –p LDAPAuth.p –param password



Everything will look the same

```
Starting authentication procedure WinLDAPAuth.p using server localhost
LDAP auth initialization returned with:
Press space bar to continue.
```

Testing call with simple user name

```
Test successful user authentication for fred
debug: resolving simple user-id DN: fred
Press space bar to continue.
```

Notice returned : 0 – Success!

```
debug: LDAP search returned  : 0
debug: resolved LDAP DN for user-id fred is
Press space bar to continue.
```

Now testing the call with full DN

```
uid=fred,ou=acme,ou=users,dc=pug,dc=com
debug: Relasing LDAP DN memory
Press space bar to continue.
```

Until here where now the full DN succeeds!

```
debug: Relasing LDAP search result
debug: LDAP simple bind returned  : 0
Press space bar to continue.
```

```
User authentication SUCCESS
No user attributes returned
Press space bar to continue.
```

NOTES

OpenEdge Security Lab 5 – Transparent Data Encryption

Objective

This lab provides the steps to walk through enabling OpenEdge Transparent Data Encryption (TDE) against a sports2000 database. In this lab you will see how to enable TDE for a specific table.

Duration

This lab should take approximately 15 minutes to complete.

Goals

In this lab you:

- Will learn how to work with TDE and gain an understanding of how TDE can secure your data at rest.
- Understand the encryption process and managing keystores within that process.

Notes

The directory you will be working in for this lab is: c:\Security\TDE. You will start in this directory when you double-click the TDE PROENV Icon in this directory.

The files provided for this lab are as follows:

epolicyarea.st – This file is the structure file for the encryption policy area.
mydb.st – This is a simplified structure for the sports2000 database.
type2.st – This is the structure file for the additional lab for encrypting a table in a Type II storage area.
Cheatsheet – This is a listing of all of the commands (in case you get stuck).

OpenEdge Security Lab 5a – Enabling Encryption for a Table

Instructions

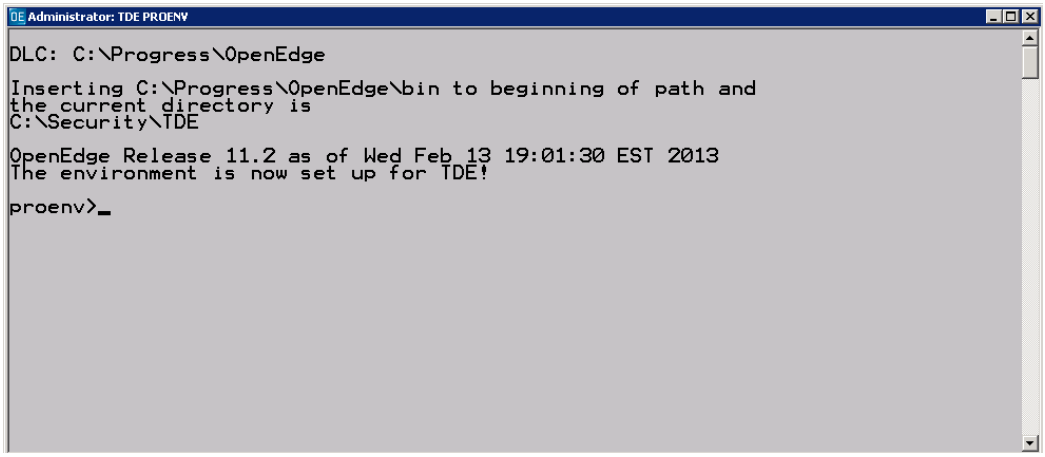
The files provided for this lab are as follows:

epolicyarea.st – This file is the structure file for the encryption policy area.
mydb.st – this is a simplified structure for the sports2000 database.
type2.st – this is the structure file for the additional lab for encrypting a table in a Type II storage area.

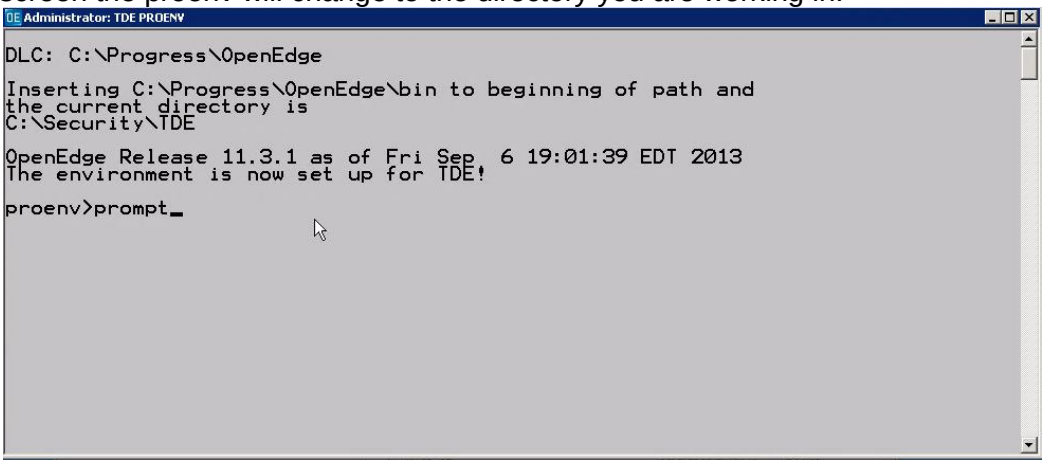
We will be encrypting the customer table. In order to do this the customer table must be an object in a type II storage area. The setup process will move the customer table to a type II area.

1. Setup
    a. Open up a TDE PROENV window. The icon should be in the c:\Security\TDE directory.

Your window should look similar to this:



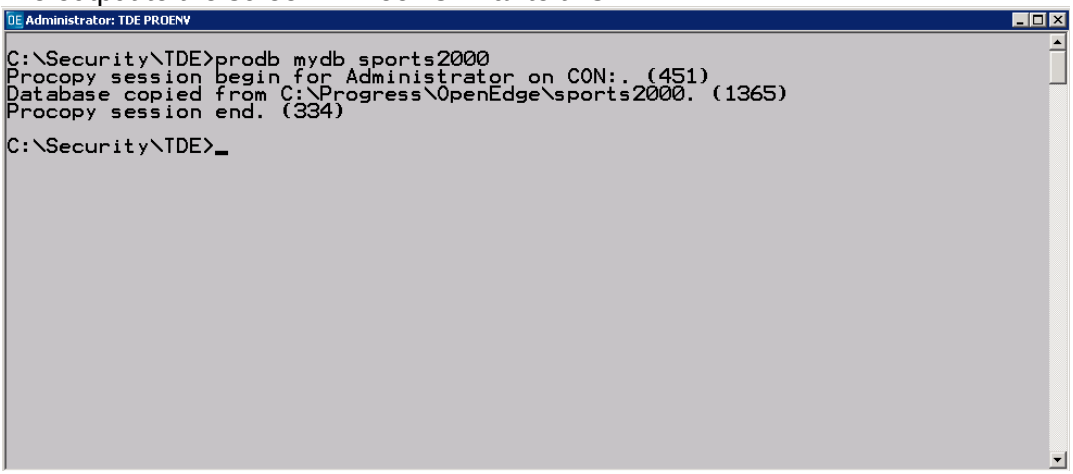
This is the window we will be working in for the rest of this lab. If you type “prompt” on the screen the proenv will change to the directory you are working in.



b. Create an OpenEdge database. For this exercise we will call it mydb. To create this database type the following command:

prodb mydb sports2000

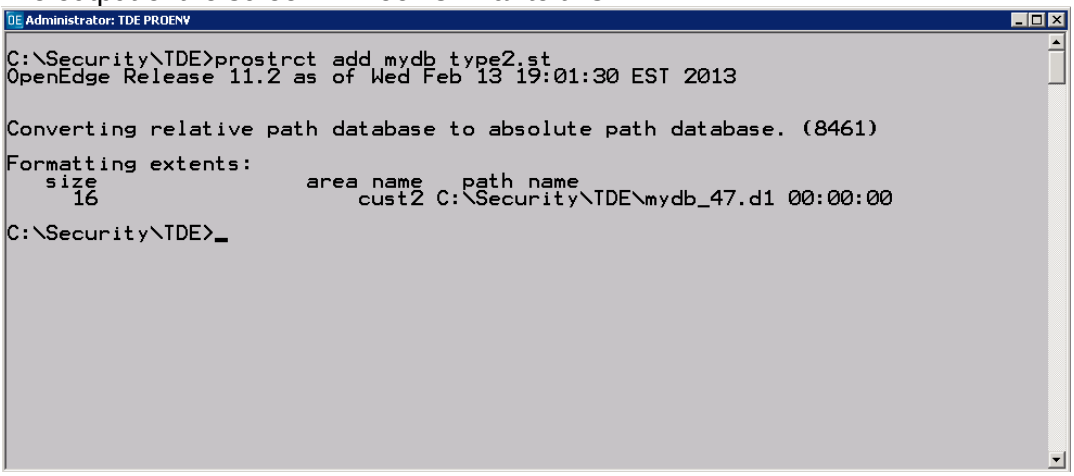
The output to the screen will look similar to this:



c. Add a type II storage area for the customer table. To do this run the following command:

prostrct add mydb type2.st

The output of the screen will look similar to this:

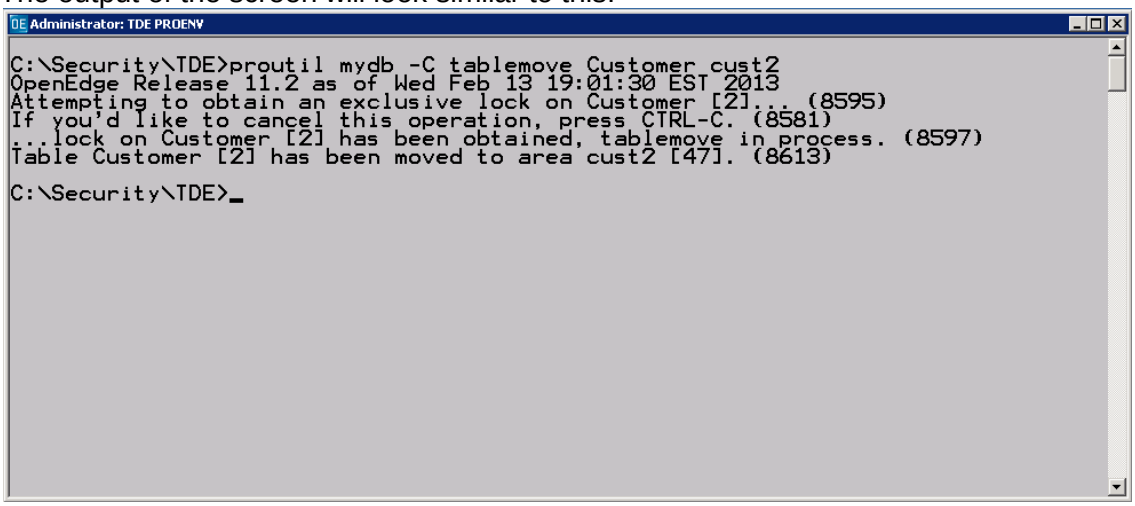


Instructions

d. Move the customer table to the new storage area created. To do this run the following command:

proutil mydb –C tablemove Customer cust2

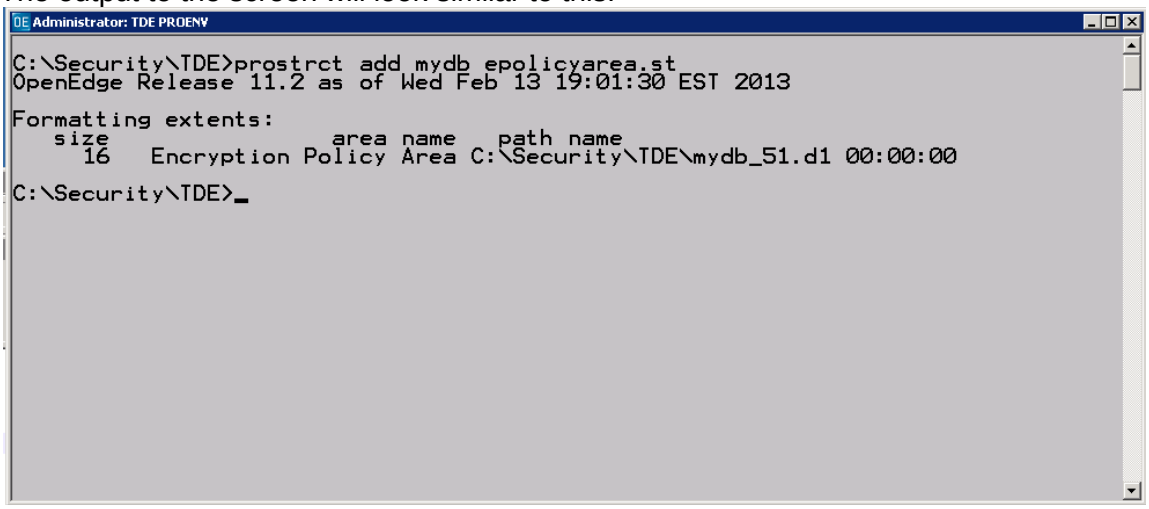
The output of the screen will look similar to this:



2. Enabling Encryption
    a. Next we will add the encryption policy area to the database. You can look at the structure file epolicyarea.st to get an idea of how to set up an encryption policy area. The command to do this is:

    prostrct add mydb epolicyarea.st

    The output to the screen will look similar to this:

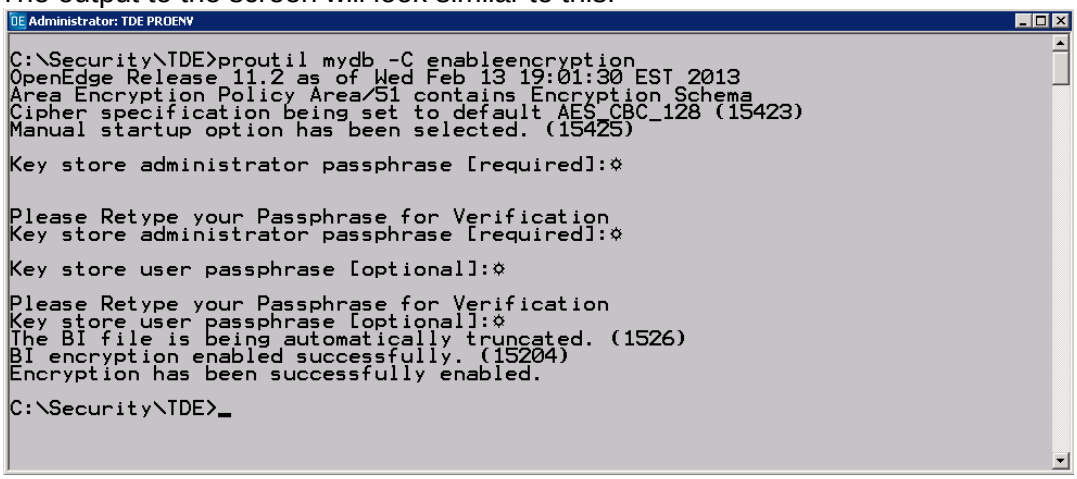


b. Now we will enable encryption for the database. This marks the database as an encryption enabled database which allows us to encrypt specific areas or objects within the database. The command to do this is:

proutil mydb –C enableencryption

There are two passphrases you must provide when enabling encryption for the database. The first is the administrator passphrase. For this exercise we will use **Exchg13!**. The second is the passphrase for the user. For this exercise we will use **User2013!**.

The output to the screen will look similar to this:

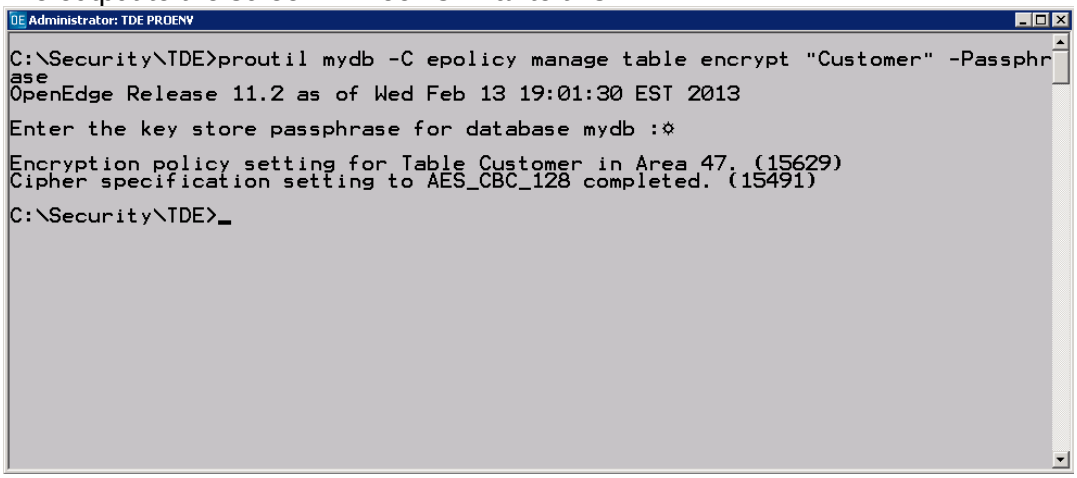


NOTE: The administrator passphrase will be needed for all of the commands in the rest of this lab. When prompted for the Key store passphrase for the database enter the passphrase that you used in the above step for the administrator!

c. Next, we will activate encryption for a specific object. For this exercise we will ecrypt an area. With Type II Storage areas you can encrypt just a table or index. The command to do this is:

proutil mydb –C epolicy manage table encrypt "Customer" –Passphrase

The output to the screen will look similar to this:

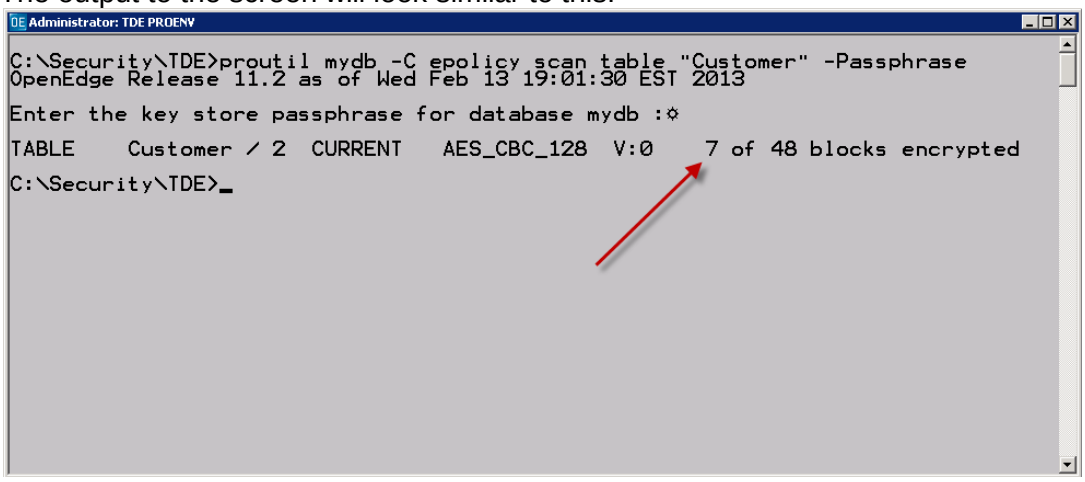


3. Scanning the Area for Encrypted Blocks

    a. Let's take a look at the area with our encryption scan utility to see how many blocs are encrypted. The command to do this is:

    proutil mydb –C epolicy scan table "Customer" –Passphrase

    The output to the screen will look similar to this:



    Notice that only 7 of 48 blocks are encrypted. The rest of the data has not been encrypted yet. We will encrypt the rest of the data soon.

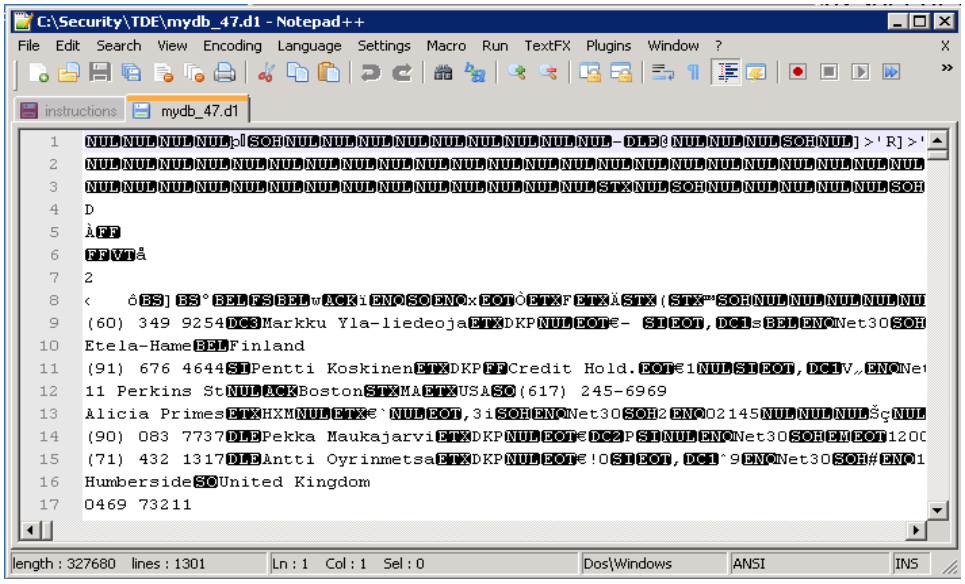
OpenEdge Security Lab 5a – Enabling Encryption for a Table (cont.)

Instructions

First let's take a look at the data in the extent. To do this open the file *mydb_47.d1* with the editor provided to you (notepad++). The command to run this is:

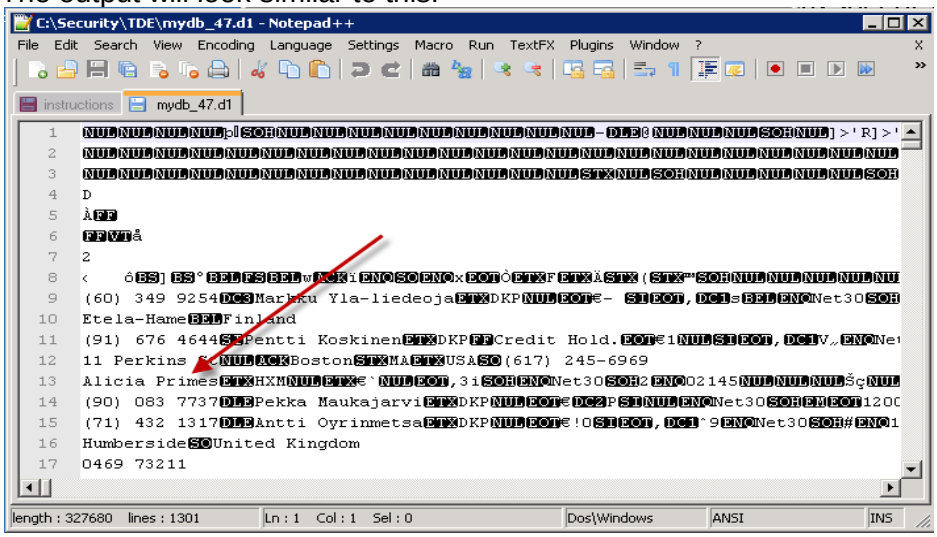
Notepad++ mydb_47.d1

The editor should look similar to this:



Next, search for "Primes". You should be able to look at the data in the database extent.

The output will look similar to this:

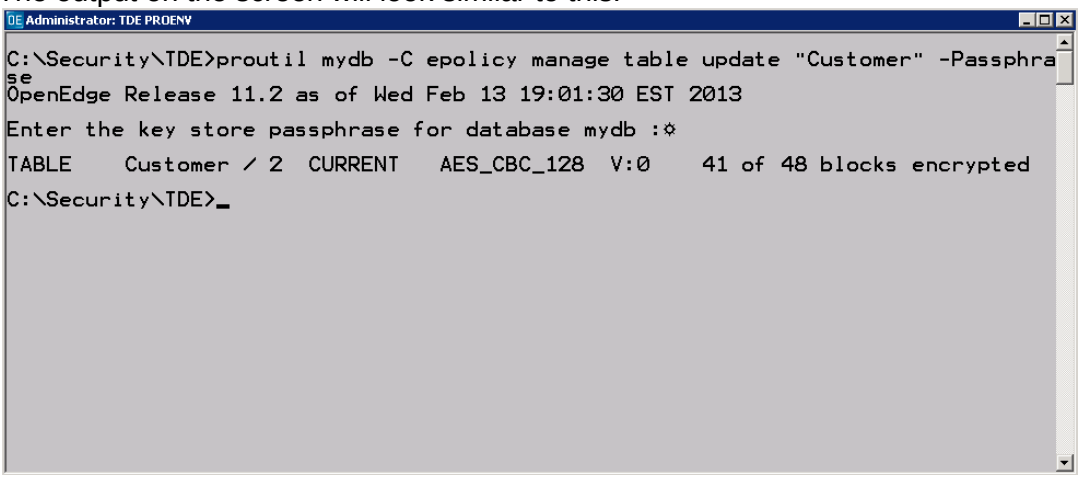


4. Encrypting the rest of the data

    a. Now we will encrypt the rest of the data in the storage area. This can be done on the fly as data is updated in the database or you can force the data to be encrypted. We will do the latter. The command to do this is:

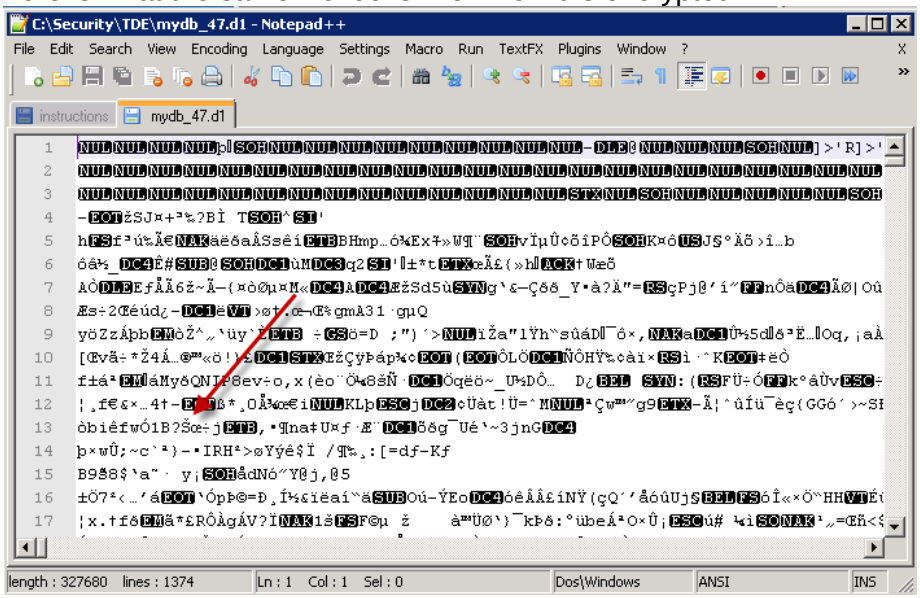
    proutil mydb –C epolicy manage table update "Customer" –Passphrase

    The output on the screen will look similar to this:



b. Now open the database extent mydb_47.d1 again with the editor provided and notice that all of the data is encrypted.

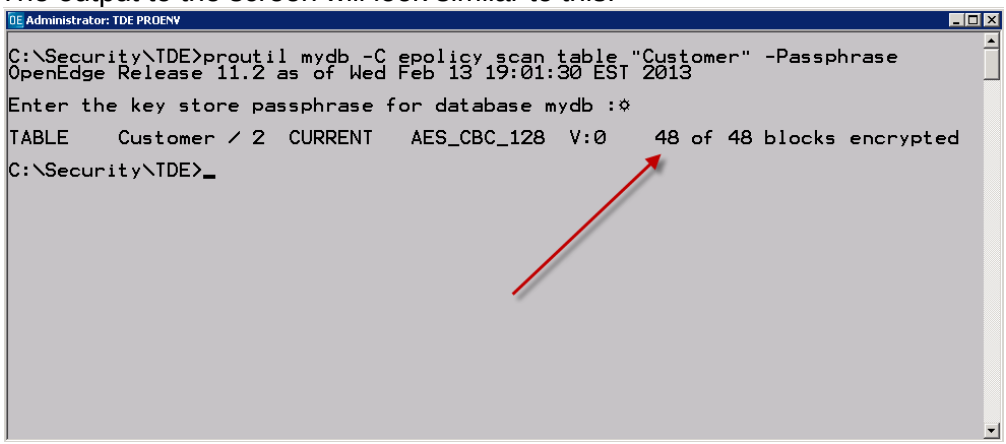
Here is what the same file looks like when it is encrypted:



c. You can now use the scan utility to see that all of the blocks in the storage area have been encrypted. The command to do this is:

proutil mydb –C epolicy scan table "Customer" –Passphrase

The output to the screen will look similar to this:



You can see that all of the blocks in the storage area are now encrypted.

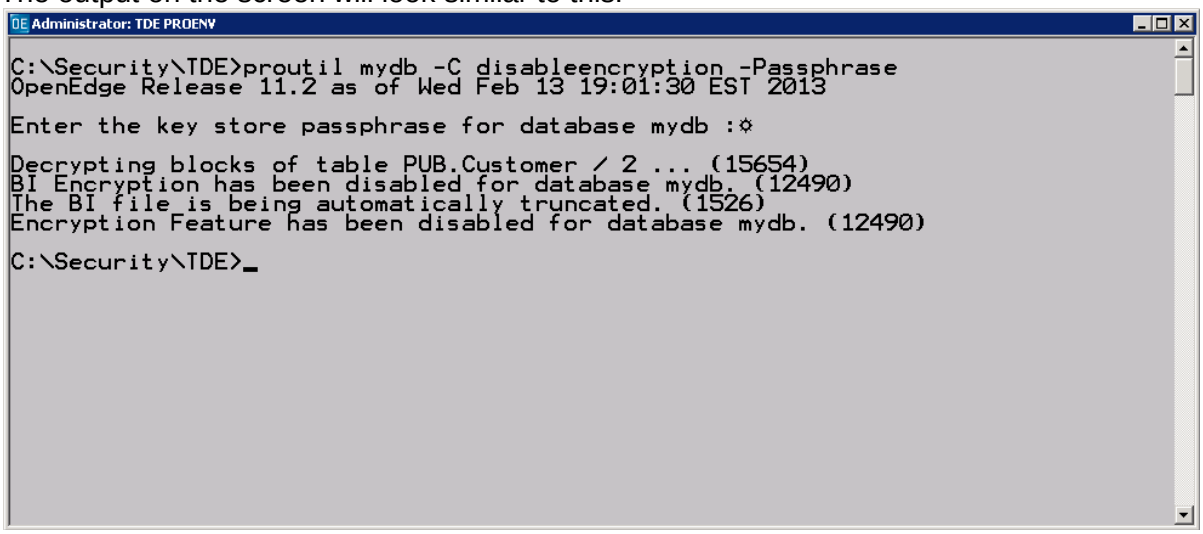
5. Disabling Encryption

   To disable encryption for the database run the following command:

   proutil mydb –C disableencryption –Passphrase

   Once this is complete encryption for the whole database will be disabled and we will be back to where we started.

   The output on the screen will look similar to this:



   This completes this part of the lab. There is an optional additional section (5B) that walks through enabling encryption for a specific table.

OpenEdge Security Lab 5b – Extra Credit: Enabling Encryption for an Area

Instructions

Follow these steps.

The files provided for this lab are the same as lab 5a. You should complete lab 5a before continuing with this lab.

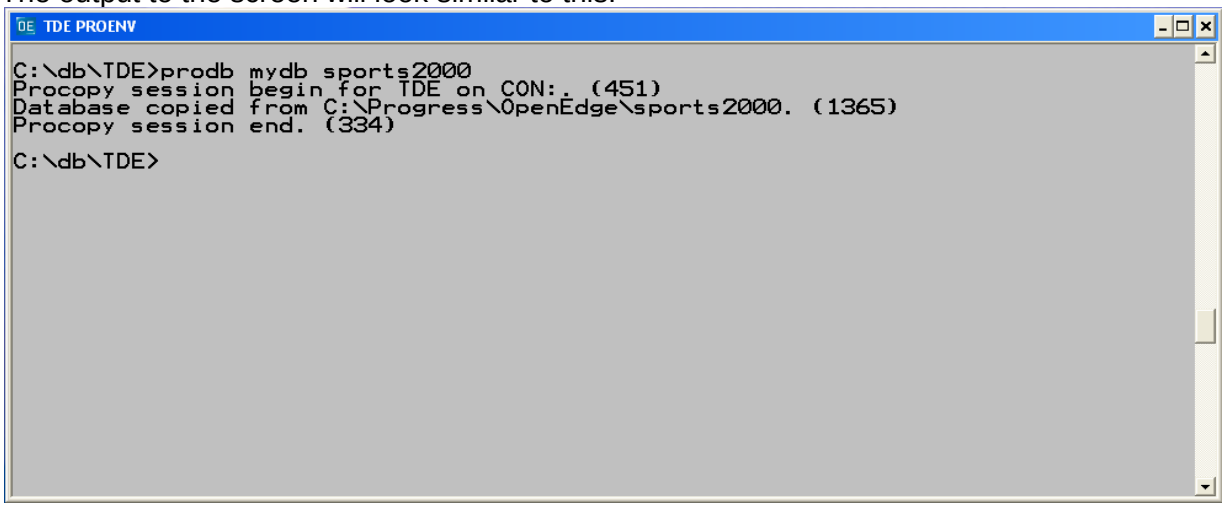
epolicyarea.st – This file is the structure file for the encryption policy area.
mydb.st – this is a simplified structure for the sports2000 database.
type2.st – this is the structure file for the additional lab for encrypting a table in a Type II storage area.

1. Setup
    a. Open up a TDE PROENV window. If the mydb still exists delete it. The easiest way to do this is to type the command: *prodel mydb*.

    b. Create an OpenEdge database. For this exercise we will call it mydb. To create this database type the following command:

    prodb mydb sports2000

    The output to the screen will look similar to this:



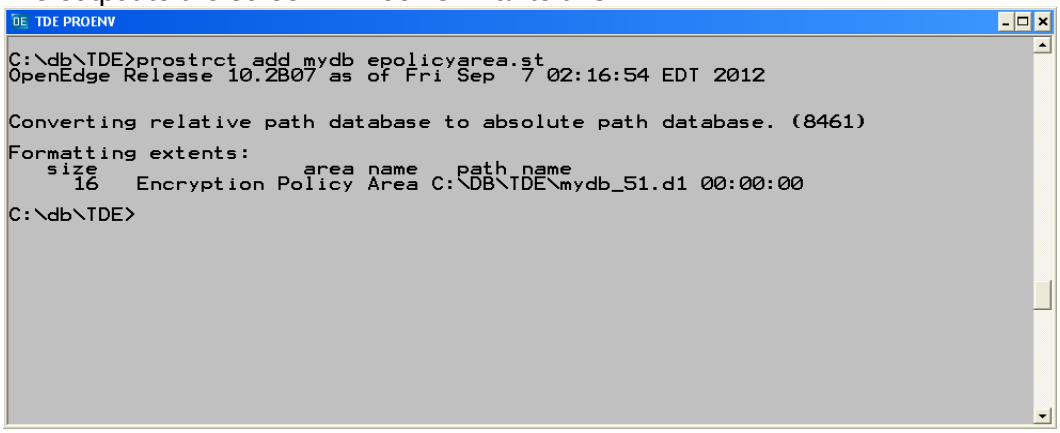
OpenEdge Security Lab 5b – Extra Credit: Enabling Encryption for an Area (cont.)

Instructions

2. Enabling Encryption
    a. Next we will add the encryption policy area to the database. You can look at the structure file epolicyarea.st to get an idea of how to set up an encryption policy area. The command to do this is:

    prostrct add mydb epolicyarea.st

    The output to the screen will look similar to this:



OpenEdge Security Lab 5b – Extra Credit: Enabling Encryption for an Area (cont.)

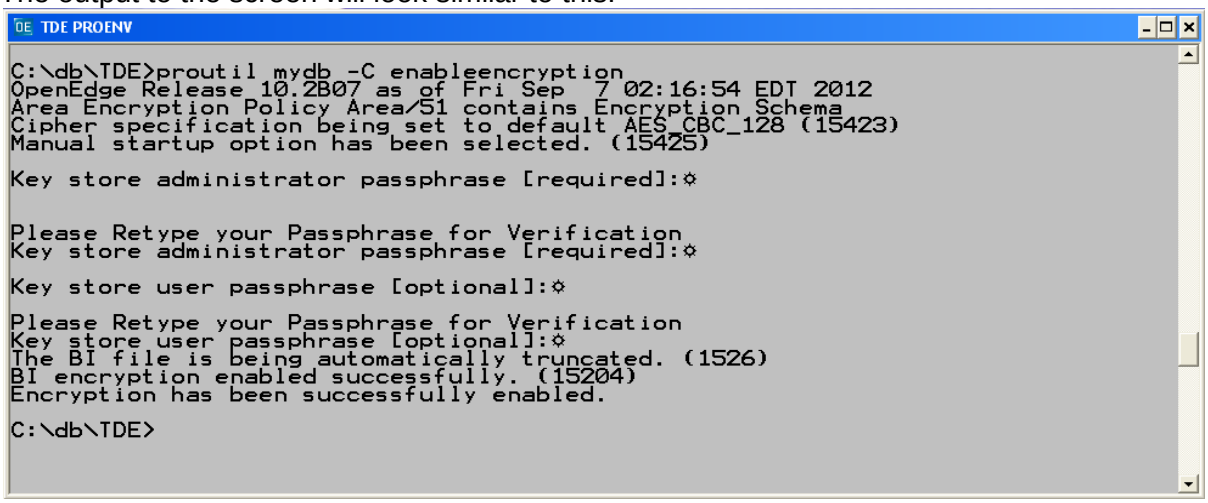
Instructions

b. Now we will enable encryption for the database. This marks the database as an encryption enabled database which allows us to encrypt specific areas or objects within the database. The command to do this is:

proutil mydb –C enableencryption

There are two passphrases you must provide when enabling encryption for the database. The first is the administrator passphrase. For this exercise we will use **Exchg13!**. The second is the passphrase for the user. For this exercise we will use **User2013!**.

The output to the screen will look similar to this:

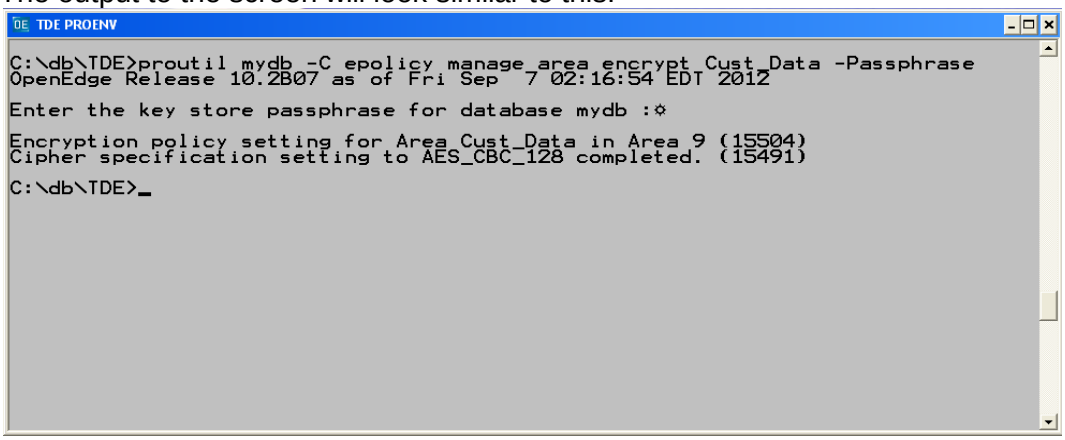


NOTE: The administrator passphrase will be needed for all of the commands in the rest of this lab. When prompted for the Key store passphrase for the database enter the passphrase that you used in the above step for the administrator!

c. Next, we will activate encryption for a specific object. For this exercise we will ecrypt an area. With Type II Storage areas you can encrypt just a table or index. The command to do this is:

proutil mydb –C epolicy manage area encrypt Cust_Data –Passphrase

The output to the screen will look similar to this:



OpenEdge Security Lab 5b – Extra Credit: Enabling Encryption for an Area (cont.)

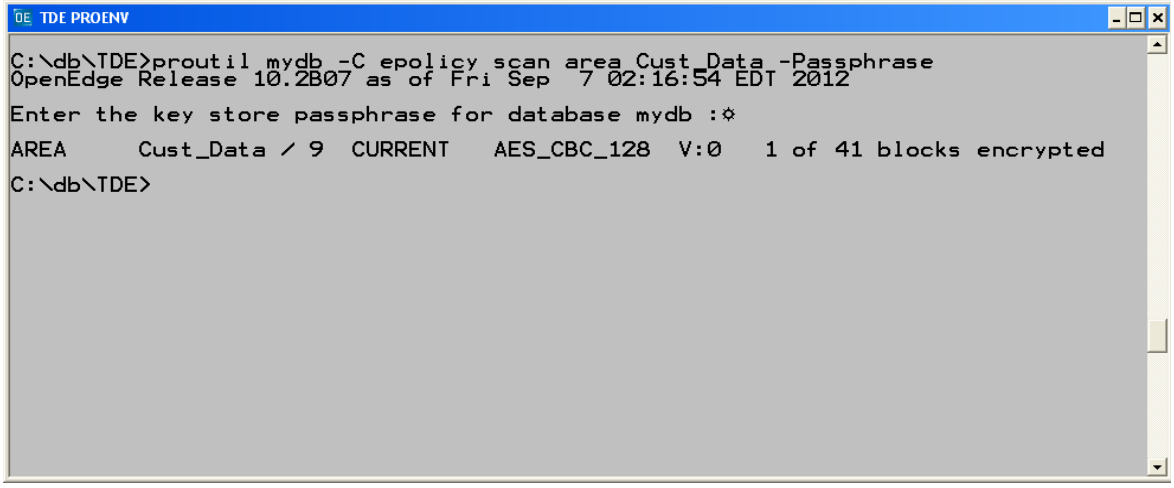
Instructions

3. Scanning the Area for Encrypted Blocks

   Let's take a look at the area with our encryption scan utility to see how many blocs are encrypted. The command to do this is:

   proutil mydb –C epolicy scan area Cust_Data –Passphrase

   The output to the screen will look similar to this:



   Notice that only 1 of 41 blocks are encrypted. The rest of the data has not been encrypted yet. We will encrypt the rest of the data soon.

   Now let's take a look at the data in the extent. To do this open the file *mydb_9.d1* with the editor provided to you. Next, search for "Primes". You should be able to look at the data in the database extent.

OpenEdge Security Lab 5b – Extra Credit: Enabling Encryption for an Area (cont.)

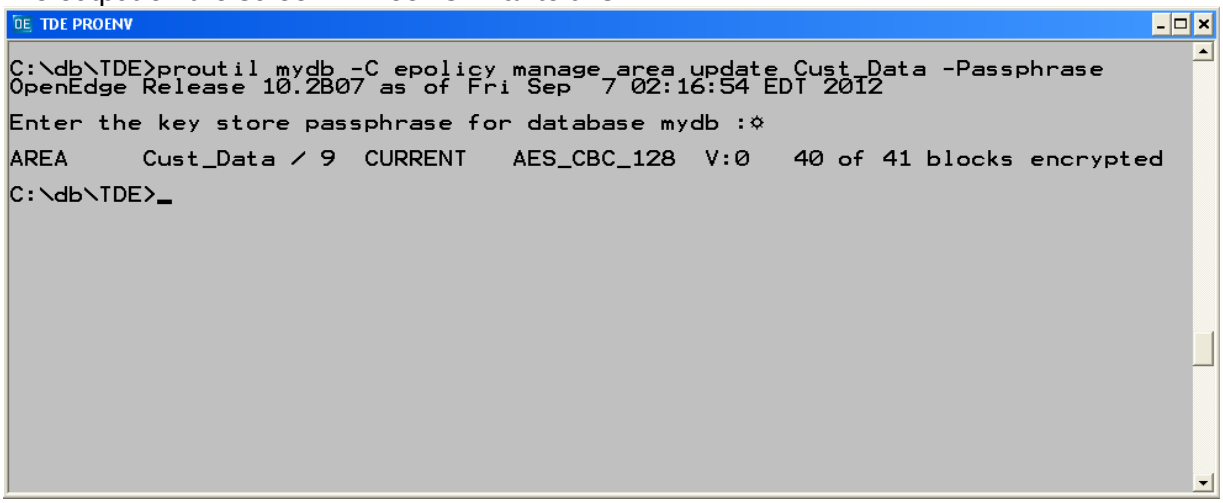
Instructions

4. Encrypting the rest of the data

    a. Now we will encrypt the rest of the data in the storage area. This can be done on the fly as data is updated in the database or you can force the data to be encrypted. We will do the latter. The command to do this is:

    proutil mydb –C epolicy manage area update Cust_Data –Passphrase

    The output on the screen will look similar to this:



    b. Now open the database extent mydb_9.d1 again with the editor provided and notice that all of the data is encrypted.

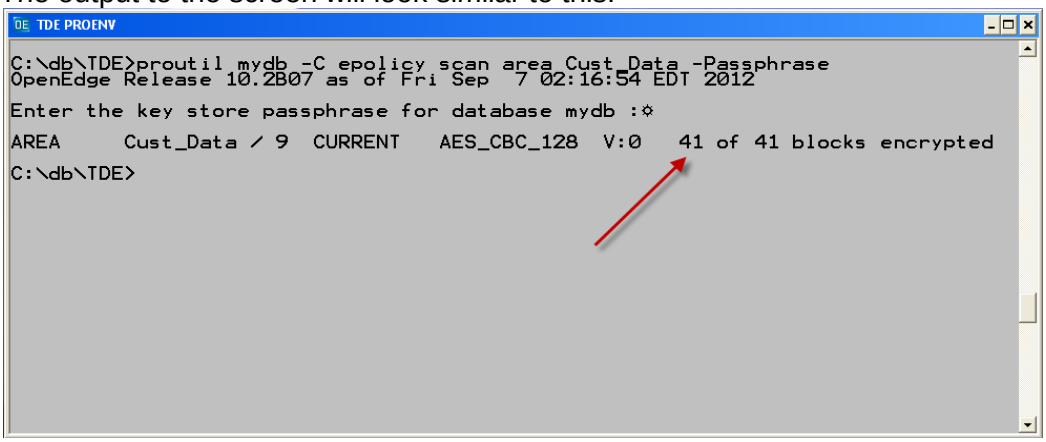
OpenEdge Security Lab 5b – Extra Credit: Enabling Encryption for an Area (cont)

Instructions

c. You can now use the scan utility to see that all of the blocks in the storage area have been encrypted. The command to do this is:

proutil mydb –C epolicy scan area Cust_Data –Passphrase

The output to the screen will look similar to this:



You can see that all of the blocks in the storage area are now encrypted.

OpenEdge Security Lab 5b – Extra Credit: Enabling Encryption for an Area (cont.)

Instructions

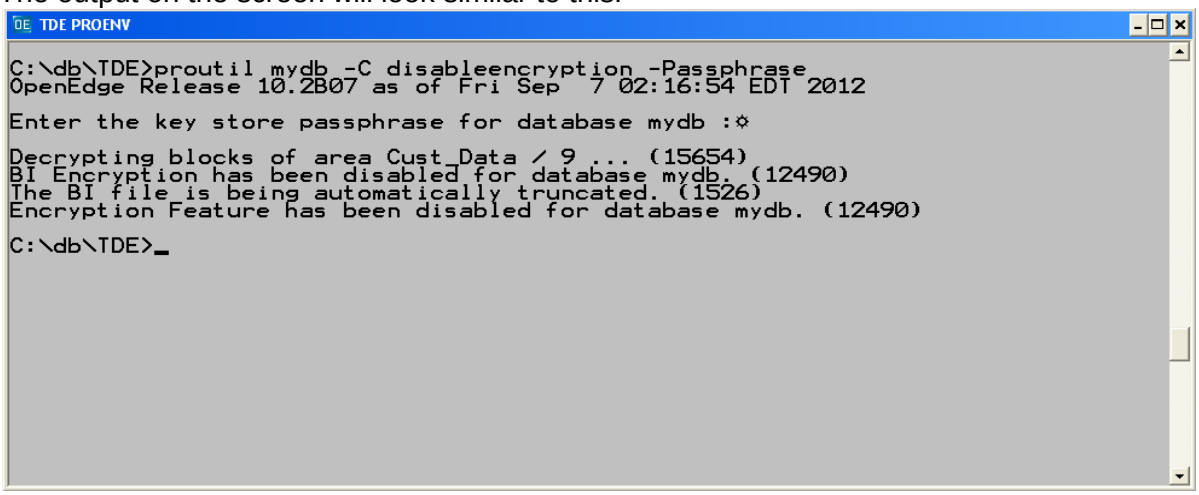
5. Disabling Encryption

To disable encryption for the database run the following command:

proutil mydb –C disableencryption –Passphrase

Once this is complete encryption for the whole database will be disabled and we will be back to where we started.

The output on the screen will look similar to this:



This completes this part of the lab. There is an optional additional section (5B) that walks through enabling encryption for a specific table.

This completes the lab.....

NOTES

OpenEdge Security Lab 6 – ODBC/JDBC

Objective

This lab provides steps to create a DBA user, create a normal user and assign / remove privelages from that user.

Duration

This lab should take approximately 15 minutes to complete.

Goals

In this lab you:

- Create a DBA user
- Create a Normal User
- Grant privelages to the user
- Revoke privelages from the user

Instructions

Follow these steps.

1. Connect to the database via SQL
   Sqlexp –db sports2000 –S 1234 –user sysprogress –password X

2. Create a new user
   CREATE USER ‘dba’, ‘dba’;
   COMMIT;

3. Make this user have the dba role
   GRANT ‘DBA” to ‘dba’;
   COMMIT;

OpenEdge Security Lab 6b -

Instructions

Follow these steps.

1. Connect to the database via SQL
   Sqlexp –db sports2000 –S 1234 –user dba –password dba

2. Create a new user
   CREATE USER ‘fred’, ‘barney’;
   COMMIT;

3. Make this user have access to the Customer and Order tables
   GRANT SELECT ON PUB.Customer to ‘fred’;
   GRANT SELECT ON PUB.Order to ‘fred’;
   COMMIT;

4. Verify this works
   Sqlexp –db sports2000 –S 1234 –user fred –password barney
   SELECT COUNT(*) FROM PUB.Customer;
   SELECT COUNT(*) FROM PUB.Order;

5. Now to remove some permissions
   Sqlexp –db sports2000 –S 1234 –user dba –password dba
   REVOKE SELECT ON PUB.Order FROM ‘fred’;
   COMMIT;

6. Verify this works
   Sqlexp –db sports2000 –S 1234 –user fred –password barney
   SELECT COUNT(*) FROM PUB.Customer;
   SELECT COUNT(*) FROM PUB.Order;

   The second select should return a Access Denied result.

This completes the lab.....

NOTES

This is the end of all of the labs…

This page left intentionally blank…
(except for the notice that this page is left intentionally blank and information at the bottom of the page)