Web Express Logon for WebSphere Host On-Demand Portal Environment

George Baker
WebSphere Worldwide Technical Sales

Tami Takamiya
Host On-Demand Development

Bijal D. Patel
Host On-Demand Testing

Introduction

In today's electronic business environments, as more businesses shift their focus from owning islands of stand alone software applications that carry out specific business tasks to integrating existing IT assets to implement business processes, the need for new software capabilities arises to deal with this new way of conducting business. The success of such solutions rests upon the ability to hide the complexity of backend IT infrastructure from the end-users. WebSphere Host On-Demand provides secure and concurrent access to multiple host applications from a Web browser. WebSphere Host On-Demand, through the use of custom portlets, allows customers to simplify user interface design and business process flows. In a secured host application environment, applications often require the user to be authenticated multiple times. For example, systems with a network security application in place require the user to be authenticated before he or she is connected to a network. After the user is connected to the network, he or she might need to enter an ID and password to connect to the sessions, depending on Host On-Demand configuration model. Finally, the user must enter a separate ID and password for each host system to which he or she connects. Current versions of Host On-Demand come with a feature called Web Express Logon, which can be used to automate the final step, the host login. Web Express Logon works with the Host On-Demand portlet, and can leverage capabilities found in WebSphere Portal Server. This paper describes how Host On-Demand supports Web Express Logon in the Portal environment and explores the implementation choices options.

This paper is divided into the following sections.

1. Web Express Logon Portal environment provides an architectural view of the Web Express Logon implementation found in Host On-Demand.
2. Credential Vault Slots provides basic information about the credential vault, the type of slot available for use with Host On-Demand, and an example of creating a vault slot for a Host On-Demand portlet.
3. Managing credentials discusses the following techniques for managing user credentials in the credential slot.
    - Using the XML Access to manage user credentials describes a more bulk method of populating and managing user credentials.
    - Using a custom portlet to manage user credentials describes how a customer could deploy a portlet for the user enabling the user to manage their own credentials.
    - Using Tivoli Products to manage user credentials describes how to leverage Tivoli products to manage the credential vault and save on user and administrator work.

Web Express Logon Portal environment

Figure 1 shows the sequence of steps involved in the Host On-Demand implementation of Web Express Logon using WebSphere Portal credential vault. The user opens a Web browser and clicks a hyperlink to load the Portlet page. They are authenticated and the Host On-Demand portlet downloads the applet to the user's machine. The Host On-Demand applet retrieves the vault slot ID and vault slot type from the portlet configuration and passes that information to the Portal Server as parameters. The diagram illustrates the Host On-Demand processing of Web Express Logon.

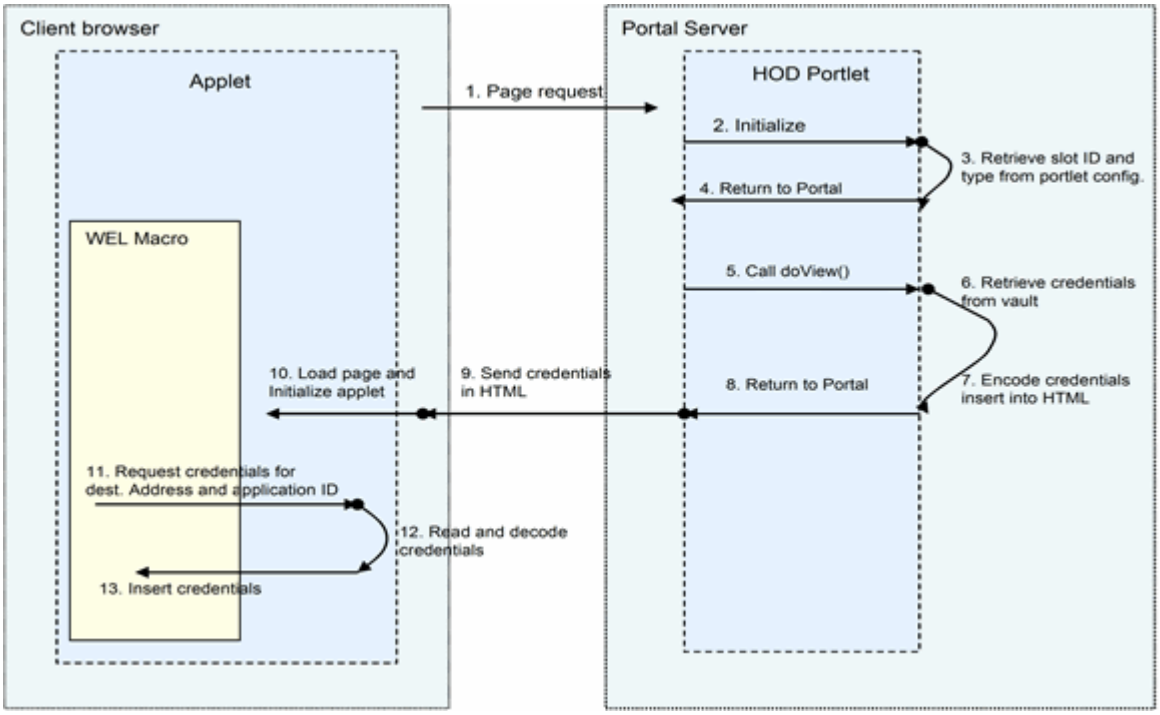

Figure 1 Web Express Logon sequence diagram

1. The user requests the portal page with Host On-Demand on it.
2. The Host On-Demand portlet initializes.
3. The slot ID and type are retrieved from the portlet configuration.
4. The information is returned to the portal.
5. The Portal calls the doView() method of the Host On-Demand portlet to refresh the portlet HTML.
6. The Host On-Demand portlet retrieves credentials from the credential vault.
7. The Host On-Demand portlet encodes the credentials and inserts them into the HTML.
8. Host On-Demand portlet returns control to WebSphere Portal
9. WebSphere Portal sends the HTML page to the client.
10. The HTML page is loaded by the client browser and the Host On-Demand applet is initialized.
11. The Web Express Logon macro requests the credentials for the user/address/application combination.
12. The Host On-Demand applet reads and decodes the credentials and returns them to the macro.
13. The Web Express Logon macro inserts the credentials into the appropriate locations in the data stream.

Though the credentials are encoded, the more secure environment would be to conduct the Portal session via HTTPS.

> **Restriction:** The Portal credential vault cannot be used as a host credential mapper database for RACF passticket generation. For the RACF passticket generation the credential mapper servlet is required.

Credential Vault Slots

In the WebSphere Portal Server implementation of Host On-Demand Web Express Logon, the credential vault APIs are used to retrieve the user credentials (host user ID and password). WebSphere Portal server supports either local databases such as Cloudscape or DB2, or Tivoli Access Manager Global Sign-on (GSO) lock box for the secure storage of credentials. The credential vault is partitioned into two types of segments – user-managed segments and administrator-managed segments. These segments are further divided into multiple slots. Slots in administrator-managed segments are created by an administrator and slots in user-managed segments are programmatically created by a user application. Host On-Demand does not provide any tools to create and manipulate user-managed slots.

Administrative slot

Credentials stored in administrative slots are not shared among users. Since Host On-Demand portlets provide access to critical host applications and data, it is considered a best practice to create an administrative slot for Host On-Demand portlets.

An administrative slot must be created by the administrator using a credential vault portlet, but the user portlet manages the credentials. Host On-Demand does not provide tools to manipulate credentials. Users can either use the XML process mentioned in Using the XML Access to manage user credentials on page 6 or create a portlet as mentioned in Using a custom portlet to manage user credentials on page 11 to manage credentials. Figure 1 shows how to create an Administrative slot using credential vault portlet.

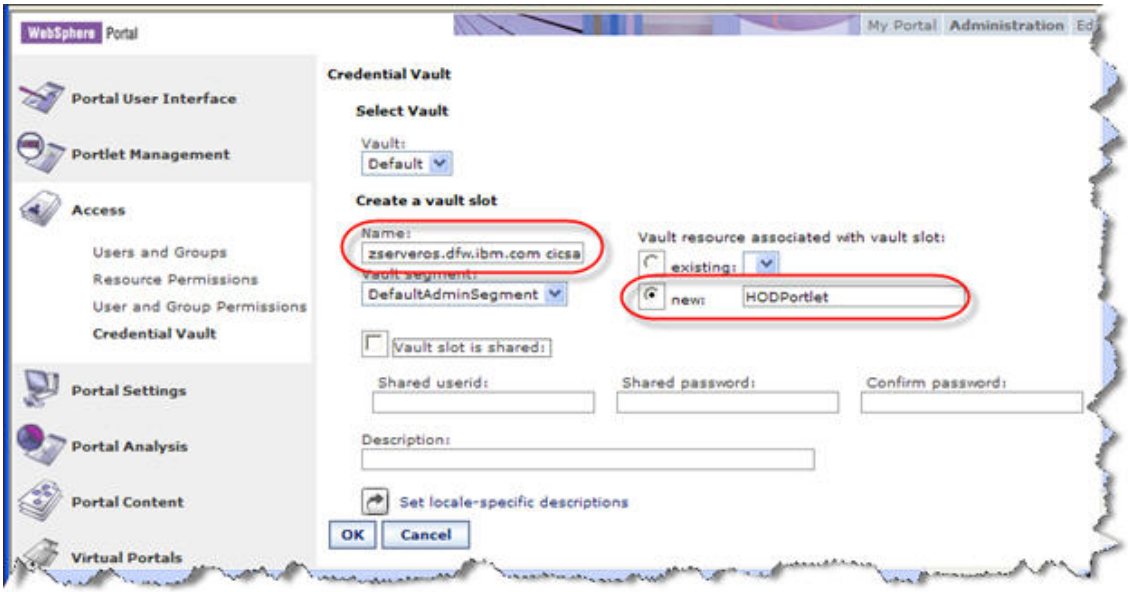

Figure 2 Add administrative slot

The **`Name`** of the slot must be provided in following format:

slot_id<blank>destination_address<blank>application_id

It is also good practice to name the resource. If you do not, you will not be able to permanently delete resources as you test or when some is removed from the system.

Note:

1. All ids are separated by blanks.
2. slot_id and application_id must be the same as the Slot ID and Application ID defined when you created the Host On-Demand portlet with the Deployment Wizard.

3. 3270 type sessions require the application_id, but 5250 sessions don't require an application_id. However, for 5250 sessions, a blank must be specified after the destination_address. (Example: slot_id<blank>destination_address<blank>)

In the example shown in Figure 2, the slot_id was **HOD**, the destination_address was **zserveros.dfw.ibm.com**, and the application_id was **cicsa.**

System slot

A system slot is use when the portlet application requires a login to the system using a common id, similar to an administrative slot, a system slot must be created by the administrator. However, credentials in system slots are managed by an administrator. Figure 3 shows how to create a system slot for a Host On-Demand portlet.

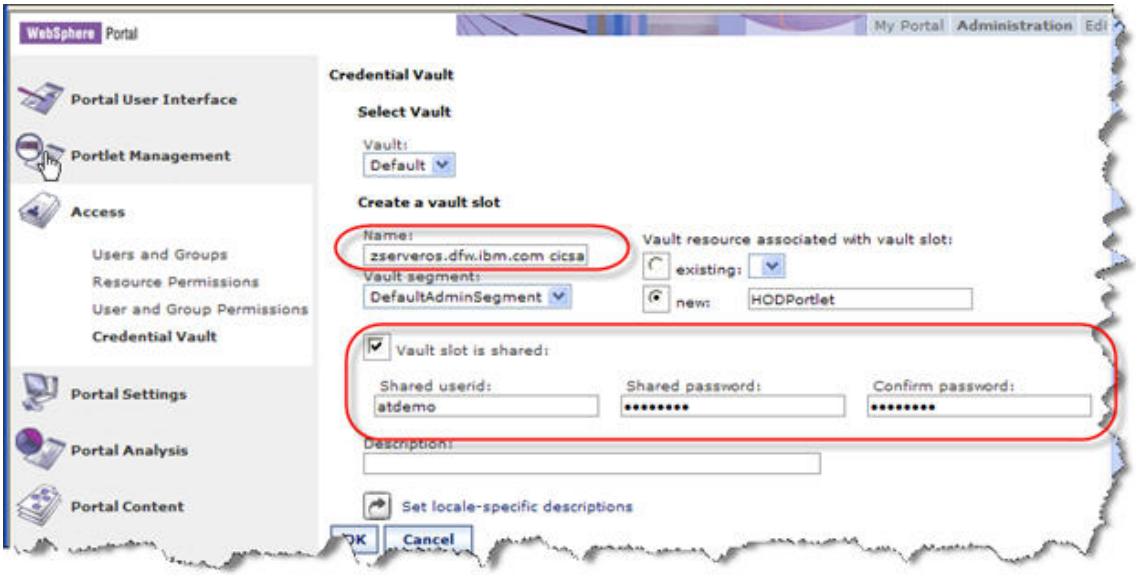

Figure 3 Add system slot

For the name of the slot, follow the conventions defined in the previous section, Administrative slot. Unlike administrative slot, the **Vault slot is shared** check box must be checked and credentials must be provided in the **Shared userid** and **Shared password** fields.

Most host systems require users to have unique credentials; therefore, it is not a good practice to use a system slot for Host On-Demand.

Shared slot

A shared slot should be created when two different applications require the same id and password. This type of slot must be created programmatically by the user, and credentials must be submitted using **edit** mode of the portlet.

Host On-Demand does not support edit mode for this function, and since a shared slot could be deleted by a user, it is not recommended to create a shared slot for Host On-Demand portlets.

Private slot

Like shared slots, private slots must be created programmatically by the user. As the name suggests, credentials stored in private slots are not shared a among user's portlets.

Note: Host On-Demand does not support private slots.

Managing credentials

Once the slot is defined and Host On-Demand is configured to retrieve the credentials from the slot there is the issue of managing credentials, which includes

1. Initially inserting the credentials into the slot
2. Managing the credentials as they change.

There are three major ways to manage credentials:

1. Have the Portal Administrator use XML Access to insert and/or manage credentials.
2. Create a portlet that lets all users manage their own credentials
3. Implement Tivoli Access Manager, Tivoli Directory Integrator, and optionally Tivoli Identity Manager to manage credentials.

The remainder of this article reviews each of these methods. Please note that all of these methods could be implemented individually or in combination.

Using the XML Access to manage user credentials

XML Access is a utility that ships with WebSphere Portal, used for updating portal configurations. It is a command line utility, but from WebSphere Portal 5.1, administrators can use it through Portal Administration Portlets.

Administrators can utilize the XML Access to manage user credentials, but there are some functional limitations, which are described in the comment lines of the `createDeploymentCredentials.xml` file. The `createDeploymentCredentials.xml` file is in the `<wps>/config/templates` directory, where `<wps>` indicates the WebSphere Portal base directory:

```
    <!-- updating the credential vault. this script creates a new segment
containing one slot in the portal credential vault.  the credentials (userid
and password pairs) that are stored in the vault cannot be accessed using
the XML configuration interface. you can only set the credentials using the
administration portlets for the credential vault. -->
```

The comment means:

1. The existing credentials cannot be read using XML Access.
2. The existing credentials can be read only by using the administration portlets
3. It is possible to replace the existing credentials with new credentials using XML Access.

That is, XML Access can be used for creating new credentials, or overriding the existing credentials with new credentials. That is what we are going to show here.

The `createDeploymentCredentials.xml` file contains the following XML segment that shows how to set the credentials using XML Access:

```
<portal action="locate">
   <credential-segment
      action="update"
      adapter-type="default"
      name="DefaultAdminSegment"
      user-mapped="false">
      <description>Default Admin Segment</description>
      <credential-slot
```

```
            action="update"
            name="deployment.user"
            active="false"
            system="true"
            resource="deployment.user"
            secrettype="userid-password">
            <localedata locale="en">
               <description>Credentials for deployment user</description>
               <keywords>APPSERVERADMIN USER</keywords>
            </localedata>
            <password-secret
               action="create"
               user="deployment.user"
               external-id="@WasUserIdXml@">
               @WasPassword@
            </password-secret>
         </credential-slot>
      </credential-segment>
   </portal>
```

This example creates a UserPasswordCredentialSecret in a system slot, but XML Access can create a UserPasswordCredentialSecret in an administrative slot also. Following is the description of the XML elements and attributes found in this example.

- *`Portal` element – Defines the action of XML Access*
    - `action` attribute – Identifies the action to be taken. It is set to "locate" for deployment.
- *`Credential-segment` element – Defines the vault segment to be updated.*
    - `Action` attribute – Identifies the action to be taken. This example sets it to "update", **but it should be set to "locate" for deployment**. If it is set to "update", XML Access generates following warning message:
      `com.ibm.wps.command.xml.XmlCommandException: XMLC0090W: It is not possible to modify credential-segment resources. Existing settings are left unchanged.`
    - `Adapter-type` attribute – Identifies the adapter-type for the vault segment. It is set to "default" if the credentials are stored in WebSphere Portal's default database vault.
    - `Name` attribute – Identifies the name of the vault segment.
    - `User-mapped` attribute – Specifies whether the vault segment is managed by users. **For the use of system slots and administrative slots, it is set to "false"**
    - `Description` element – Defines the description of the vault segment.
- *`Credential-slot` element – Defines the credential slot to be updated.*
    - `Action` attribute – Identifies the action to be taken. **It is set to "update" for deployment.**
    - `Name` attribute – Identifies the name of the credential slot.

- `Active` attribute – Sets the credential slot's active/passive flag. For UserPasswordCredential, it is set to "false".
- `System` attribute – Specifies if the credential slot reference a system credential or not. **If this slot is a system slot, set this attribute to "true". If this slot is an administrative slot, set this attribute to "false".**
- `Resource` attribute – Specifies the Resource Name for this credential slot.
- `Secrettype` attribute – Sets the credential slot's secret type. For UserPasswordCredential, it is set to "userid-password".

- *`Localedata` element – Defines Locale-dependent data for the credential slot*
  - `Locale` attribute – Identifies the Locale for this `localedata` element
  - `Description` element – Defines the description of the credential slot
  - `Keywords` element – Defines the keywords of the credential slot

- *`Password-secret` element – Defines the UserPasswordCredentialSecret to be stored.*
  - `Action` attribute – Identifies the action to be taken. **It is set to "create" for deployment.**
  - `User` attribute – Identifies the userid on WebSphere Portal.
  - `External-id` attribute – Identifies the userid on an external system, like a mainframe system. **The password on an external system is specified as the content of the `password-secret` element.**

Here is an example of how to deploy UserPasswordCredentialSecret for Host On-Demand.

Example for Host On-Demand

Let us assume that we are about to deploy UserPasswordCredentialSecret for Host On-Demand to a WebSphere Portal V5 Server with the following conditions:

Segment name	DefaultAdminSegment
Credential Slot Type	Administrative
Credential Slot Name	HOD zserveros.dfw.ibm.com cicsa
Slot Resource Name	HODPortlet
Slot Description (English)	Credentials for Host On-Demand users
Slot Keywords (English)	Host On-Demand USERS
Number of Host On-Demand Users	2
User 1's Portal Userid	wpsuser1
User 1's Host System Userid	atdemo
User 1's Host System Password	demo4you
User 2's Portal Userid	wpsuser2
User 2's Host System Userid	nsdemo

User 2's Host System Password	demo4you

The input file for XML access (HODExample.xml) will be like:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- HODExample.xml -->
<request
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="PortalConfig_1.3.xsd"
   type="update"
   create-oids="true">

   <portal action="locate">
      <credential-segment
          action="locate"
          adapter-type="default"
          name="DefaultAdminSegment"
          user-mapped="false">
         <description>Default Admin Segment</description>
         <credential-slot
            action="update"
            name="HATS hostname APPLID"
            active="false"
            system="false"
            resource="Host On-Demand"
            secrettype="userid-password">
            <password-secret
               action="create"
               user="wpsuser1"
               external-id="atdemo">
               demo4you
            </password-secret>
            <password-secret
               action="create"
               user="wpsuser2"
               external-id="nsdemo">
               demo4you
            </password-secret>
         </credential-slot>
      </credential-segment>
   </portal>
</request>
```

The steps for the deployment are:

1. Create two users, wpsuser1 and wpsuser2, using the administration portlets, if they do not currently exist.

2. Import HODExample.xml using the administration portlet. Specify the path in the entry field and click **Import**.

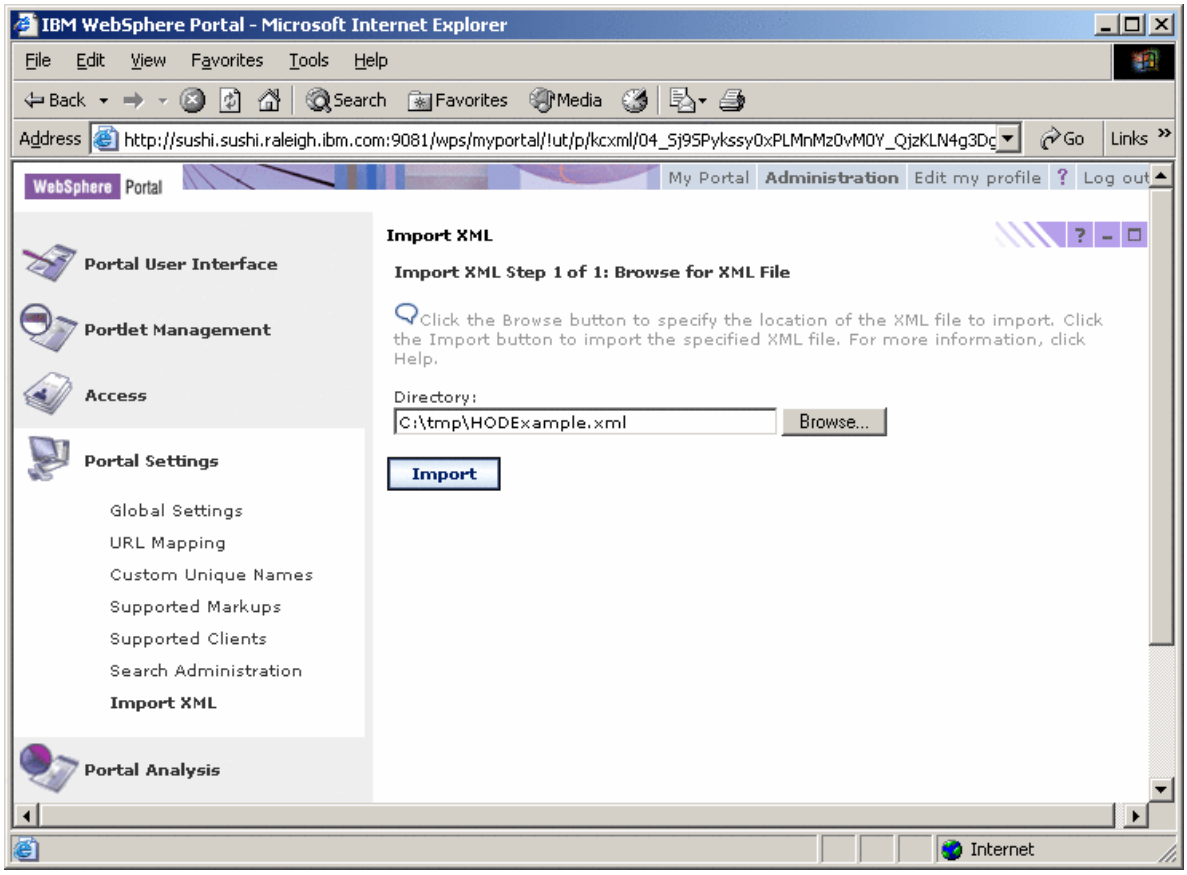

Figure 4 Import XML

3. The following message is displayed when Import XML runs successfully,

Figure 5 Import XML success

4. Clicking the View Details link will display the results:

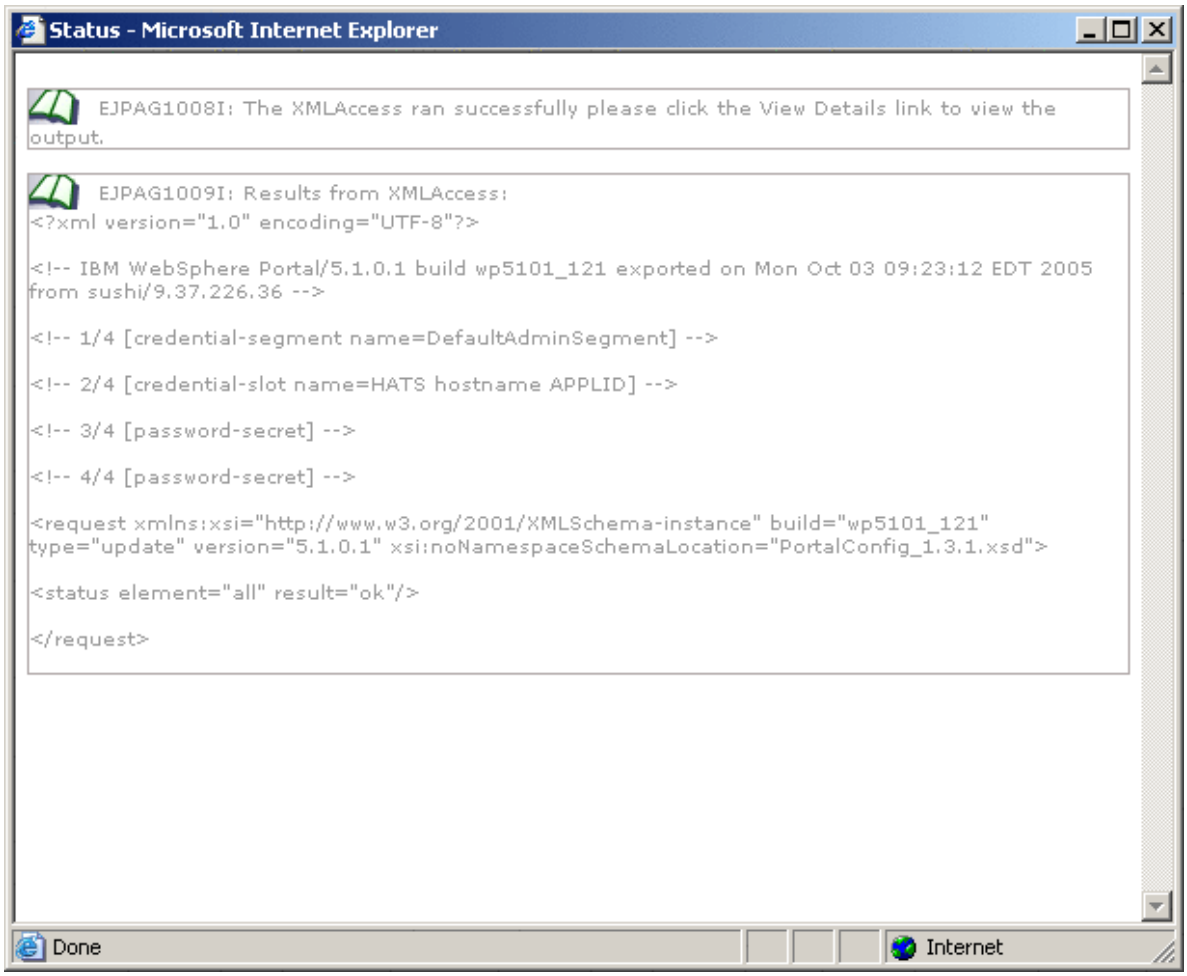

Figure 6 XML Access Results

Using a custom portlet to manage user credentials

It is easy to create a customized portlet to be placed on the user's page that allows them to update their own credentials stored in administrative slots. A sample portlet is described below that enables users to edit their IDs and passwords stored in an administrative slot. The portlet describe here is provided as collateral material.

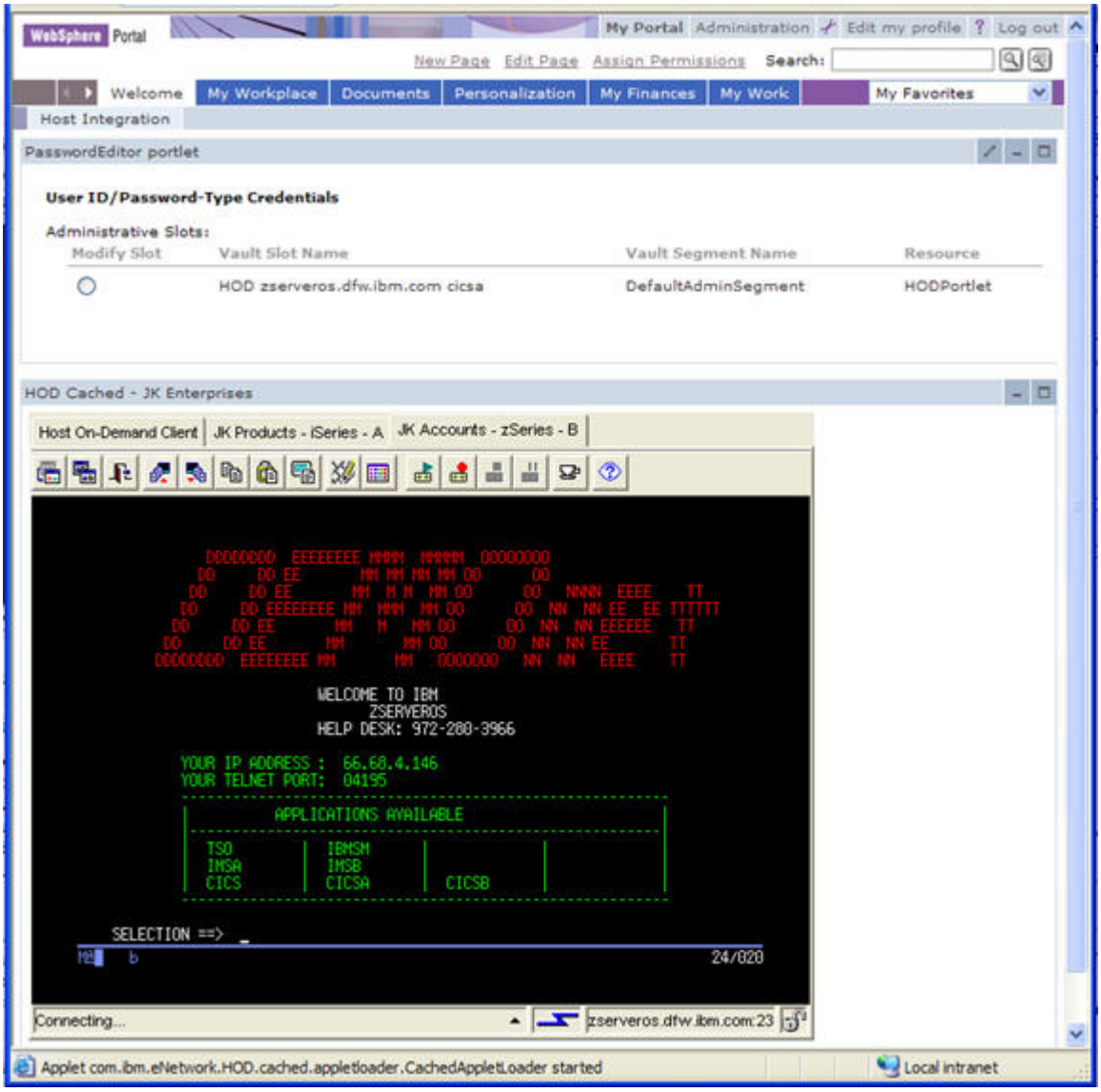

Figure 7 Password Editor and Host On-Demand portlet

In the Host On-Demand portlet page shown in Figure 7, the portlet was installed above the Host On-Demand portlet. Notice that the portlet identifies the credential vault slot that is defined, the segment and Resource name being used. This information is used by the portlet and is of value to the Portal administrator.

When the user changes their password because it expired, they forgot it and it had to be reset, they can update their credentials in the vault themselves using this portlet. After the passwords have been reset the user would select the **modify Slot** radio button. The user's credentials will be retrieved from the credential vault and presented via the portlet edit mode as show in Figure 8. The host user ID and two password fields displayed and populated with their current values. To change the credentials the user types the new credentials into the respective fields. You cannot paste the password into these fields, they must be typed. When all values are entered the user clicks **Save** and the portlet window returns

Figure 8 Password Editor, edit mode

Should the passwords not match you will receive an error as shown below in Figure 9.

Figure 9 Password mismatch

Correct the error and click **Save** to continue.

Using Tivoli Products to manage user credentials

In addition to using WebSphere Portal credential vault for storing the credentials for host applications, WebSphere Portal can be integrated with external security manager products, such as Tivoli Access Manager for e-business to provide additional features. This approach allows centralized security management of all Web based applications, including WebSphere Portal, through Tivoli Access Manager. When using Tivoli Access Manager with WebSphere Portal, the credential vault uses the Tivoli Access Manager Global Sign-on (GSO) lockbox to provide a secure repository for all credentials.

A major concern with customers when using a repository to hold host credentials for Web Express logon is how to manage changing passwords. Here we are talking about the condition where the user ID expires, is reset, or the user simply decides to change it. If all users are defined in Tivoli Access Manager as GSO users, and if the Tivoli Access Manager Global Sign-on lockbox is used as the credential vault repository, all changes to the RACF password might be automatically captured and replicated to the Tivoli Access Manager Global Sign-on lockbox using the IBM Tivoli Directory Integrator.

First, let's examine the configuration of the Portal credential vault to use the Tivoli Access Manager lockbox as its back-end repository.

Configuring the Credential Vault Adapter for Tivoli Access Manager

WebSphere includes a vault adapter to access the Tivoli Access Manager Global Sign-on (GSO) lockbox, a secure repository for all your resource credentials. Portlets that access the credential vault service can, without any

additional configuration access existing Tivoli resource credentials. The WebSphere Portal v5.1 Information Center contains information on configuring the credential vault adapter to use the Tivoli lockbox. Simply search the Information Center using the following search string Tivoli vault and look for the entry titled Configuring the Credential Vault adapter for Tivoli Access Manager. This entry provides step by step instructions on how to configure the Credential vault to use the Tivoli Access Manager lockbox as a repository for the credential vault.

IBM Tivoli Directory Integrator

IBM Tivoli Directory Integrator is a product that synchronizes data residing in directories, databases, collaborative systems, applications used for human resources (HR), customer relationship management (CRM), Enterprise Resource Planning (ERP), and other corporate applications. Through the use of plug-ins Directory Integrator can detect password changes for many products, such as IBM Directory Server, Windows, and RACF, among others. It is this ability to detect password changes for RACF, coupled with the ability to synchronize this data with the GSO lockbox, that makes this product so valuable in a Host On-Demand Web Express Logon environment.

To use this plug-in RACF must be configured to encrypt and store password changes into a z/OS LDAP store. The Directory Integrator z/OS LDAP changelog EventHandler detects that a password has changed and then synchronizes it with other identity stores, such as the Tivoli GSO lockbox.

There is an ITSO Technote, *Password Synchronization Services with Tivoli Directory Integrator and Tivoli Identity Manager* that the reader is encouraged to read. In addition, *Enterprise Security Architecture Using IBM Tivoli Security Solutions*, SG24-6014-01, an IBM Redbook, provides additional information on how Tivoli products can be used in your enterprise to enhance security and usability.

References

WebSphere Portal:

- *Using Credential Vault to Provide Single Sign-On for Portlets* http://www-128.ibm.com/developerworks/Websphere/library/techarticles/0211_konduru/konduru.html
- *Developing an XML request file for XML Access in WebSphere Portal Version 4.1* http://www-128.ibm.com/developerworks/websphere/library/techarticles/0208_konduru/konduru.html
- *Configure the Credential vault adapter for Tivoli Access Manager,* http://publib.boulder.ibm.com/pvc/wp/510/ent/en/InfoCenter/wpf/tam_vault.html *(multi-platform version)*

Host On-Demand:

- *IBM WebSphere Host On-Demand v9 Information Center, Web Express Logon Reference* *http://publib.boulder.ibm.com/infocenter/hod9help/index.jsp*
- *Web Express Logon for IBM WebSphere Host On-Demand v9* ftp://ftp.software.ibm.com/software/network/hostondemand/library/whitepapers/wel2.pdf
- *Host Access Client Package V4 Update, SG24-6182-02* http://www.redbooks.ibm.com/abstracts/sg246182.html
-

Tivoli Access Manager:

- *Password Synchronization Services with Tivoli Directory Integrator and Tivoli Identity Manager* http://www.redbooks.ibm.com/abstracts/tips0390.html?Open

- *Enterprise Security Architecture Using IBM Tivoli Security Solutions, SG24-014-01*, http://www.redbooks.ibm.com/Redbooks.nsf/RedbookAbstracts/SG246014.html
  Chapter 17 – Introducing Directory Integrator

Trademarks

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

- IBM
- z/OS
- WebSphere
- DB2
- Cloudscape
- Tivoli

Microsoft and Windows are trademarks or registered trademarks of Microsoft Corporation in the United States, other countries, or both.

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