BEATRICE CARDONA
WEN TANG

Re: Release: 1974
   Service Requests: 83181
   Error Reports: 2363
   Cobol Programs: UCPPMRT, PPMDPSR, PPCSTSPR
   Copy Members: None
   Include Members: None
   DDL Members: None
   Bind Members: None
   CICS Maps: None
   CICS Help: None
   Forms: None
   Table Updates: See install instructions for manual updates to the access and rules tables
   EAR Files: webMerit.ear
   Java Programs: ControllerServlet.java, ARSMAccessBean.java, CommonDataAccessBean.java,
   CycleIdDetailAccessBean.java, RosterDataAccessBean.java,
   AddTitleRangeObject.java, CycleBaseObject.java, CycleDetailObject.java,
   CyclePerfObject.java, DeptListObject.java, RosterBaseObject.java,
   RosterEmpDetailObject.java, RosterPageObject.java, AdminCyclePerf.java,
   AdminCycleProcessHandler.java, AdminCycleUpdate.java, CycleListHandler.java,
   DeptListHandler.java, EventHandlerBase.java, MainMenuHandler.java,
   RosterHandler.java, AccessCheckFilter.java, RedirectFilter.java,
   TokenCheckFilter.java, constant.properties, Parm.java, StringSanitizer.java,
   EmpDetailTokenTag.java, TokenTag.java
   Javascript: RosterRangeBased.js, RosterStepBased.js
   Web Pages: AdminCycleAddNew.jsp, AdminCycleEdit.jsp, AdminCycleList.jsp,
   AdminCycleUpdateStatus.jsp, AdminPerfEval.jsp, AdminPerfRatingStep.jsp,
   AdminPerfRatingStatus.jsp, AdminPerfRatingTable.jsp, AdminPerfRatingRange.jsp,
   DisplayCbuc.jsp, DisplayRangeRateTable.jsp, DisplayStepRateTable.jsp,
   DisplaySubLoc.jsp, Error.jsp, MsgDisplay.jsp,
   RosterRangeBased.jsp, RosterRangeEmpDetail.jsp, RosterStepBased.jsp,
   RosterStepEmpDetail.jsp, ViewOnlyRosterRangeBased.jsp,
   ViewOnlyRosterRangeEmpDetail.jsp, ViewOnlyRosterStepBased.jsp,
   ViewOnlyRosterStepEmpDetail.jsp
   Urgency: Urgent (see Timing of Installation below)

This release addresses the following Service Request and Error Report:

Service Request 83181

Service Request 83181 asks that view-only access be provided for the Web Merit application. Users with the new view-only access should be able to view anything within the Merit Review/Input selection on the Merit Menu but should not be able to
modify or save any data, regardless of cycle status. They should have the ability to navigate within the Merit Review/Input portion of the application and view information on pop-up windows (e.g. cycle information, title information, employee information).

Specifically, the SR asks for two types of view-only access:

- View-only access that is restricted to particular departments
- Universal view-only access

**Error Report 2363**

UCLA has reported that the Web Merit application has the following security vulnerabilities:

1. The application does not adequately scrub user submitted data which can potentially allow for cross-site scripting and cross-site request forgery.
2. The application is susceptible to cross-site request forgery (CSRF).
3. In terms of application error diagnostics, when the application encounters an error, it allows the user to see the stack trace; it should give the user a reference that the support team can use to find the stack trace in the application log. This is a minor threat but it is good practice to avoid revealing internal application details to end users.

**Summary of Changes**

To provide view-only access, modifications were made to the Cobol stored procedure which is the ARSM security driver for access to Web Merit data. Changes were also made to the stored procedure that identifies departments for the merit roster and inserts them into the merit department table. Two new rule types were defined in the UC0RUL rules table for view all access and view control point access. A new access intent for audit roster access was also defined. Also, the code was modified to return values of ‘U’ (for update), ‘R’ (for read) and ‘N’ (for no access), instead of ‘Y’ and ‘N’ in the current process.

On the java side, providing view-only access was accomplished by adding calls to the ARSM security driver in the program that handles the main menu for Web Merit to determine and save in session a user’s access. A new filter was written to map, if necessary, a view-only version of the requested page, and to check for unauthorized access attempts.

Modifications to the java code to address the security vulnerabilities outlined in Error Report 2363 were made as follows:

1. All the data entered on Web Merit pages goes through one specific routine in one program (ControllerServlet.java). Code was added at this single point to scrub all incoming data.
2. To guard against CSRF, random tokens are now added to each web page where the user can enter data. When a web page comes back to the server, a program checks that the token matches the one saved in “memory” (session). In addition, the Web Merit application was modified to use the post-redirect-get (PRG) approach for delivering web pages to the client. The PRG approach breaks post requests into 2 parts – post request incoming, then via a redirect, a get request outgoing.
3. A reference number is now generated at the time an error occurs. The reference number is written to the logs and displayed on the error page for users to relay to the system administrator.

**Cobol Programs**

**UCPPPMRT**

UCPPPMRT is a DB2 stored procedure which is the ARSM security driver for the Web Merit application. Currently, when checking for user authorization, the program returns a PERMISSION flag of “Y” or “N”. “Y” currently implies update access since that’s the only type of access available. The program will be changed to return values of “U” for update, “R” for Read-only or “N” for no access.
Also, the program currently requires that an ARSM “PERSONAL” rule be present for cycle access. Since read-only users are not required to have a PERSONAL rule, the program will be changed to bypass the check for this rule for read-only users.
There will be two new rule types in the UC0RUL table, MERITVWA for view all and MERITVWC for view from a control point. For Access Intent of Cycle Access, the two new rules will be checked for cycle status of D, O, H, S and N.
A new Access Intent value should be added to the input request code to distinguish audit roster access from regular roster access, with value ‘AUDACC’. In addition, to conform to best coding practices, the Insert SQL statement to insert into the Merit Department Table (PPPMDP) will be changed to reference individual columns of the table.

PPMDPSPR

PPMDPSPR is a stored procedure that identifies potential departments for merit roster selection and inserts them in the Merit Department table (PPPMDP).
The only change to this program is to include the two new rules MERITVWA and MERITVWC in the values for valid ARSM rules.

PPCSTSPR

PPCSTSPR is a stored procedure that is used to calculate costing for a Roster.
The change to this program is to include the two new rules MERITVWA and MERITVWC in the values for valid fund group.
Also, in order to avoid changing the PPPFND table, when the ARSM rule is MERITVWA it is substituted by MERITADM and when it is MERITVWC it is substituted by MERITCTL. This achieves the same effect when the Cost Summary button is clicked on the roster page.

Web Files

The PPS web applications are contained in runtime .ear and source archive .zip files. Details on the contents of these files can be found in the detail design document.

ControllerServlet.java

This is the main controller for Web Merit. The program loads all parameters and constants used in the Web Merit application. A call to execute a new method to load a list used for mapping view-only access was added.

This program contains a method through which all data passes. Code was added to this method to sanitize incoming data.

If an unexpected system error occurs, a reference number is now generated and written to the log along with the detailed error information. This reference number is now displayed on the error.jsp for the user to relay to the system administrator.

ARSMAccessBean.java

This program calls a COBOL stored procedure to determine the user’s access permissions. The program was modified to incorporate changes to support view-only access.

CommonDataAccessBean.java

This program gets code values used throughout the application. The code value for “ALL” sub locations and collective bargaining units is “@@”. This program was modified to change the code for ALL to “ZZ” internally since the @ sign is one of the special characters stripped from data input because of security concerns. The code will remain “@@” in the database.

CycleIdDetailAccessBean.java

This program gets the data for a particular cycle from the database. This program was modified to change the code for “ALL” for sub location and CBU to “ZZ” from “@@” for internal use.
RosterDataAccessBean.java
This program gets roster data from the roster table PPPMLA. This program was modified so search names are handled as positional parameters.

AddTitleRangeObject.java
This is the additional title code range object. This program was serialized.

CycleBaseObject.java
This is the base cycle object. This program was serialized.

CycleDetailObject.java
This is the cycle detail object. This program was serialized.

CyclePerfObject.java
This is the performance code object. This program was serialized.

DeptListObject.java
This is the department list object. This program was serialized.

RosterBaseObject.java
This is the roster base object. This program was serialized.

RosterEmpDetailObject.java
This is the roster employee detail object. This program was serialized.

RosterPageObject.java
This is the roster page object. This program was serialized.

AdminCyclePerf.java
This is the handler for the “Performance Evaluation Table” page for Merit Administration. Changes were made to support the PRG approach. These include saving objects to session and not to the request object.

AdminCycleProcessHandler.java
This is the handler for the “Merit Cycle Administration” page. Changes were made to support the PRG approach. These include saving objects to session and not to the request object.

AdminCycleUpdate.java
This is the handler for updating the status for a merit cycle. Changes were made to support the PRG approach. These include saving objects to session and not to the request object.

CycleListHandler.java
This is the handler for “Roster Cycle List” page. Changes were made to support the PRG approach. These include saving objects to session and not to the request object.

DeptListHandler.java
This is the handler for the “Roster Department Selection List” page. Modifications were made in how a user’s permission level is verified to support view-only access. In addition, changes were made to support the PRG approach. These include saving objects to session and not to the request object.
EventHandlerBase.java
This is the main event handler for all pages in the Web Merit application. The program was modified to check if the user has view-only access, and if so, map the request if necessary to a new view-only version of the requested page. A change was also made to support the PRG approach.

MainMenuHandler.java
This is the Merit Menu handler and it serves as an entry point to the application. A call to the Cobol stored procedure UCPPPMRT was added to determine if the user has view-only access when first entering the application.

RosterHandler.java
This is the handler for the roster pages. Changes were made to support the PRG approach. These include saving objects to session and not to the request object.

AccessCheckFilter.java
This is a new program written to map view-only pages and check for unauthorized access.

RedirectFilter.java
This is a new program written to support the PRG approach. The web pages are expecting data in the request object. This filter moves the objects from session (saved by specific handlers) to the request object.

TokenCheckFilter.java
This is a new program written to safeguard against CSRF attacks. The program TokenTag.java (see below) generates a hidden token on each web page that allows data entry and saves the token in session. This filter looks for the handlers of those web pages and compares the token on return to the token saved in session. If the tokens are not the same, the user is shown a blank message page.

Parm.java
This program loads parameters and constant values and was modified to build a list of valid view-only pages.

constant.properties
This program contains the constant values used in the Web Merit application. A constant identifying view-only web pages was added.

EmpDetailTokenTag.java
This is a new program written to safeguard against CSRF attacks (see TokenTag.java below). EmpDetailTokenTag.java generates a hidden token on the two employee detail pages and saves the token in session. The employee detail pages are singled out as they can be opened in a separate window while the user remains on the roster page.

TokenTag.java
This is a new program written to safeguard against CSRF attacks. TokenTag.java generates a hidden token on each web page that allows data entry and saves the token in session. TokenCheckFilter.java looks for the handlers of those web pages, and compares the token on return to the token saved in session. If the tokens are not the same, the user is shown a blank message page.

Javascript

RosterRangeBased.js
This is the javascript file for the range based rosters. Changes were made in how the employee detail screen is opened to support the PRG approach.
**RosterStepBased.js**
This is the javascript file for the step based rosters. Changes were made in how the employee detail screen is opened to support the PRG approach.

**Web Pages**

**AdminCycleAddNew.jsp**
This is the “Add New Merit Cycle” page off the Administration link on the Merit Menu. Changes were made to support the addition of tokens to plug security vulnerabilities.

**AdminCycleEdit.jsp**
This is the “Add New Merit Cycle” page off the Administration link on the Merit Menu. Changes were made to support the addition of tokens to plug security vulnerabilities.

**AdminCycleList.jsp**
This is the “Merit Cycle Administration” page. Changes were made to support the addition of tokens to plug security vulnerabilities.

**AdminCycleUpdateStatus.jsp**
This is the “Update Cycle Status” page for Merit Administration. Changes were made to support the addition of tokens to plug security vulnerabilities.

**AdminPerfEval.jsp**
This is the “Performance Evaluation Table” page for Merit Administration. Changes were made to support the addition of tokens to plug security vulnerabilities.

**AdminPerfRatingRange.jsp**
This is the “Range Performance Rating Table” page for Merit Administration. Changes were made to support the addition of tokens to plug security vulnerabilities.

**AdminPerfRatingStep.jsp**
This is the “Step Performance Rating Table” page for Merit Administration. Changes were made to support the addition of tokens to plug security vulnerabilities.

**DisplayCbuc.jsp**
This page is displayed when the user clicks “?” to display the list of code values for collective bargaining units. A change was made to change the value for “ALL” from “@@” to “ZZ”.

**DisplayRangeRateTable.jsp**
This page is displayed when the user clicks a title code link on a range based roster. Changes were made to support the addition of tokens to plug security vulnerabilities.

**DisplayStepRateTable.jsp**
This page is displayed when the user clicks a title code link on a step based roster. Changes were made to support the addition of tokens to plug security vulnerabilities.

**DisplaySubLoc.jsp**
This page is displayed when the user clicks “?” to display the list of code values for sub location. A change was made to change the value for “ALL” from “@@” to “ZZ”.


Error.jsp
When there is an unexpected system error, this page is displayed and currently has a link to see the stack trace. Changes were made to display a reference number instead and not display the stack trace.

MsgDisplay.jsp
This is the generic message display page. This page was modified to display no message if the message parameter is empty. Currently the page displays a message saying the message is null if there is no message.

RosterRangeBased.jsp
This is the “Range Based Merit Roster” page for Merit Roster Selection. Changes were made to support the addition of tokens to plug security vulnerabilities. In addition, changes were made to make search names (used for previous and next pages) positional parameters.

RosterRangeEmpDetail.jsp
This is the “Range Based Roster Employee Detail” page. Changes were made to support the addition of tokens to plug security vulnerabilities.

RosterStepBased.jsp
This is the “Step Based Merit Roster” page for Merit Roster Selection. Changes were made to support the addition of tokens to plug security vulnerabilities. In addition, changes were made to make search names (used for previous and next pages) positional parameters.

RosterStepEmpDetail.jsp
This is the “Step Based Roster Employee Detail” page. Changes were made to support the addition of tokens to plug security vulnerabilities.

ViewOnlyRosterRangeBased.jsp
This is the new view-only version of the Range Based Merit Roster.

ViewOnlyRosterRangeEmpDetail.jsp
This is the new view-only version of the Range Based Roster Employee Detail screen.

ViewOnlyRosterStepBased.jsp
This is the new view-only version of the Step Based Merit Roster.

ViewOnlyRosterStepEmpDetail.jsp
This is the new view-only version of the Step Based Roster Employee Detail screen.

Installation Instructions
A separate Installation Instructions document has been provided. Please note that as of this release the installation of Web Merit requires an upgrade of the WebSphere datasource from version 4 to version 5.

Test Plan
A separate Test Plan document has been provided.

Timing of Installation
The timing of this release is urgent.
As usual, campuses are encouraged to install this release in as timely a fashion as possible and in the normal numeric sequence. If there are any questions, please send electronic mail to Nancy.Field@ucop.edu, or call 510 987-0831.

Nancy Field