

Solution Manager Encryption/Decryption Fun

A DATA X STREAM White Paper

by

Mike Salvo

Senior Consultant

Synopsis

SAP function modules **FIEB_PASSWORD_ENCRYPT** and **FIEB_PASSWORD_DECRYPT**, built with import and export interface parameters that are based on a CHAR 32 data type, are not available in the SAP Solution Manager System. Their Solution Manager replacements, function modules **CRM_MKT_EXT_PASSWORD_ENCRYPT** and **CRM_MKT_EXT_PASSWORD_DECRYPT**, are built with import and export interface parameters that are based on a CHAR 8 data type.

This white paper shows how to extend the capability of the Solution Manager function modules to handle encryption and decryption of text up to 16 characters.

Solution Manager Encryption/Decryption Fun


I was recently working on a Solution Manager project that required me to build a character string encryption and decryption facility to be utilized for authentication purposes. Now, I was already familiar with, and had many times used the SAP function modules **FIEB_PASSWORD_ENCRYPT** and **FIEB_PASSWORD_DECRYPT**.

Here is the function module **FIEB_PASSWORD_ENCRYPT**. The import and export parameters are type **FIEB_DECRYPTED_PASSWORD** and **FIEB_ENCRYPTED_PASSWORD**. Both of these types are **CHAR 32**. It is fairly simple to use – populate the import parameter with a text string up to 32 characters, and an encrypted text string is returned in the export parameter. The decrypt function module works the same way, but in reverse.

```
FUNCTION FIEB_PASSWORD_ENCRYPT.  
*-----  
*""Lokale Schnittstelle:  
*  IMPORTING  
*    VALUE(IM_DECRYPTED_PASSWORD) TYPE  FIEB_DECRYPTED_PASSWD  
*  EXPORTING  
*    VALUE(EX_ENCRYPTED_PASSWORD) TYPE  FIEB_ENCRYPTED_PASSWD  
*-----  
  
* encrypt the decrypted password  
perform encrypt_password  
  using  
    im_decrypted_password  
  changing  
    ex_encrypted_password.  
  
ENDFUNCTION.
```

“No problem”, I thought. “I’ll have this done in no time”. Little did I know, the fun was just beginning.

It turns out, that in the SAP Solution Manager system, the function modules **FIEB_PASSWORD_ENCRYPT** and **FIEB_PASSWORD_DECRYPT** do not exist.

 Function module FIEB_PASSWORD_ENCRYPT does not exist

 Function module FIEB_PASSWORD_DECRYPT does not exist

DATASTREAM

But, with a little research, I was able to find their replacements in the SAP Solution Manager System – `CRM_MKT_EXT_PASSWORD_ENCRYPT` and `CRM_MKT_EXT_PASSWORD_DECRYPT`. The new function modules even contained explicit comments that they were copied from `FIEB_PASSWORD_ENCRYPT` and `FIEB_PASSWORD_DECRYPT` (see the screenshot below).

```
FUNCTION CRM_MKT_EXT_PASSWORD_DECRYPT.  
*-----  
*""Lokale Schnittstelle:  
*  IMPORTING  
*    REFERENCE(IM_ENCRYPTED_PASSWORD) TYPE  CRMT_MKT_EXT_PASSWORD  
*  EXPORTING  
*    REFERENCE(EX_DECRYPTED_PASSWORD) TYPE  CRMT_MKT_EXT_PASSWORD  
*-----  
*  
* coding copied from ALR  
* function module: FIEB_PASSWORD_DECRYPT  
*  
* decrypt the encrypted password  
perform decrypt_password  
  using  
    im_encrypted_password  
  changing  
    ex_decrypted_password.  
ENDFUNCTION.
```

Again, I thought, “No problem”. But it did not take me long to realize that, while the code may have been copied, the interface was not. You see, in the old familiar FIEB function modules, the import and export parameters were typed on a CHAR 32 data type (`FIEB_ENCRYPTED_PASSWORD`) – more than long enough for password encryption and decryption. But in the new CRM function modules, the import and export parameters are typed on a CHAR 8 data type (`CRM_MKT_EXT_PASSWORD`), and I needed at least 16 characters.

Aaaaaaaargh! What can I do now? How do I magically convert a CHAR 8–based encryption/decryption facility into a CHAR 16–based encryption/decryption facility?

Conceptually, I solved the problem by encrypting and decrypting the 16 character text strings in two eight character passes. The final result is the concatenation of each pass into the function module exporting parameter. This allowed me to use the Solution Manager System encryption/decryption function modules without modification.

I created two custom function modules – `Z_ENCRYPT_TEXT` and `Z_DECRYPT_TEXT`. Both of these were included in a function group named `ZCRYPTO`, to allow sharing of global data declarations at the top include.

DATA XSTREAM

Z_ENCRYPT_TEXT code highlight walkthrough

The complete code for **Z_ENCRYPT_TEXT** is shown in Appendix A. Here is a description of the code highlights:

The main body of this function module is the DO loop which is executed two times. Within this loop, the standard SAP function module CRM_MKT_EXT_PASSWORD_ENCRYPT is called.

During the first pass, and prior to dropping into the DO loop, the first 8 characters of clear_text_in are assigned to field-symbol <fs8_in>. The encrypted result is returned to the field-symbol <fs8_out>, which is assigned to 1st8_outtext.

```
assign clear_text_in(8) to <fs_8in>.  
assign 1st8_outtext to <fs_8out>.
```

During the second pass, the second 8 characters of clear_text_in are assigned to field symbol <fs8_in>. The encrypted result is returned to the field symbol <fs8_out>, which is assigned to 2nd8_out.

```
unassign: <fs_8in>, <fs_8out>.  
assign clear_text_in+8(8) to <fs_8in>.  
assign 2nd8_outtext to <fs_8out>.
```

Finally, after the two passes are complete, concatenate 1st8_outtext and 2nd8_outtext into the exporting parameter encrypted_text_out.

```
clear: encrypted_text_out.  
concatenate: 1st8_outtext 2nd8_outtext into encrypted_text_out.
```

The desired result is now resident in the exporting parameter, which is returned to the calling program.

The complete code for **Z_DECRYPT_TEXT** is shown in **Appendix B**. It works the same as **Z_ENCRYPT_TEXT**, but in reverse.

The data declarations in the top include of the function group ZCRYPTO are shown in **Appendix C**.

Let's unit test the function modules.

These tests were conducted using the SAP test facility for function modules.

In this test of the function module **Z_ENCRYPT_TEXT**, I entered the text string "MyNameIsMike" as the text to be encrypted.

The function module returned an encrypted text string.

Import parameters	Value
CLEAR_TEXT_IN	MyNameIsMike

Export parameters	Value
ENCRYPTED_TEXT_OUT	\$B2#bUEz\$2?5&4Z=

Next, I copied the encrypted text result from above, and pasted it into the import parameter of function module **Z_TEXT_DECRYPT**. The result shows the original clear text string.

Import parameters	Value
ENCRYPTED_TEXT_IN	\$B2#bUEz\$2?5&4Z=

Export parameters	Value
CLEAR_TEXT_OUT	MyNameIsMike

Well, there you have it. An encryption/decryption facility in the SAP Solution Manager System that allows up to 16 character input and output. I suppose I could have taken this concept one step further and allowed any length of input and output.

But I did not want to take away any of YOUR fun.

Enjoy!

Appendix A - Z_ENCRYPT_TEXT Source Code

FUNCTION Z_ENCRYPT_TEXT.

```
*****
**"Local Interface:
** IMPORTING
** REFERENCE(CLEAR_TEXT_IN) TYPE CHAR16
** EXPORTING
** REFERENCE(ENCRYPTED_TEXT_OUT) TYPE CHAR16
** EXCEPTIONS
** ENCRYPTION_ERROR
*****
*
field-symbols: <fs_8in> type any,
               <fs_8out> type any.
clear: 1st8_outext, 2nd8_outext, encrypted_text_out.
*
* Encrypt the 16 character input in two separate 8 character passes.
*
assign clear_text_in(8) to <fs_8in>.
assign 1st8_outext to <fs_8out>.
*
do 2 times.
*
* Encrypt each 8 character input field separately.
*
    CALL FUNCTION 'CRM_MKT_EXT_PASSWORD_ENCRYPT'
      EXPORTING
        IM_DECRYPTED_PASSWORD = <fs_8in>
      IMPORTING
        EX_ENCRYPTED_PASSWORD = <fs_8out>.
*
if sy-subrc ne 0.
  raise ENCRYPTION_ERROR.
  exit.
endif.
*
* Reassign the field symbols for the second 8 character pass
*
  unassign: <fs_8in>, <fs_8out>.
  assign clear_text_in+8(8) to <fs_8in>.
  assign 2nd8_outext to <fs_8out>.
*
enddo.
```

DATASTREAM

*

* Concatenate the two encrypted fields into a single field.

*

clear: encrypted_text_out.

concatenate: 1st8_outtext 2nd8_outtext into encrypted_text_out.

ENDFUNCTION.

Appendix B – Z_DECRYPT_TEXT Source Code**FUNCTION Z_DECRYPT_TEXT.**

```
*****
**"Local Interface:
** IMPORTING
** REFERENCE(ENCRYPTED_TEXT_IN) TYPE CHAR16
** EXPORTING
** REFERENCE(CLEAR_TEXT_OUT) TYPE CHAR16
** EXCEPTIONS
** DECRYPTION_ERROR
*****
*
field-symbols: <fs_8in> type any,
               <fs_8out> type any.
clear: 1st8_detext, 2nd8_detext, clear_text_out.
*
* Decrypt the 16 character input in two separate 8 character passes.
*
assign encrypted_text_in(8) to <fs_8in>.
assign 1st8_detext to <fs_8out>.
*
do 2 times.
*
* Decrypt the encrypted password.
*
CALL FUNCTION 'CRM_MKT_EXT_PASSWORD_DECRYPT'
EXPORTING
  IM_ENCRYPTED_PASSWORD = <fs_8in>
IMPORTING
  EX_DECRYPTED_PASSWORD = <fs_8out>.
*
if sy-subrc ne 0.
  raise DECRYPTION_ERROR.
  exit.
endif.
*
* Reassign the field symbols for the second 8 character pass
*
unassign: <fs_8in>, <fs_8out>.
assign encrypted_text_in+8(8) to <fs_8in>.
assign 2nd8_detext to <fs_8out>.
*
enddo.
```

DATA XSTREAM

*

* Concatenate the two decrypted fields into a single field.

*

concatenate: 1st8_detext 2nd8_detext into clear_text_out.

ENDFUNCTION.

Appendix C – Function Group ZCRYPTO Top Include Source Code

FUNCTION-POOL ZCRYPTO.

*

* Global data declarations

*

```
data: 1st8_intext type crmt_mkt_ext_password,  
      2nd8_intext type crmt_mkt_ext_password,  
      1st8_detext type crmt_mkt_ext_password,  
      2nd8_detext type crmt_mkt_ext_password,  
      1st8_outext type crmt_mkt_ext_password,  
      2nd8_outext type crmt_mkt_ext_password,  
      input_clear_text(16) type c,  
      output_clear_text(16) type c.
```