

REXX™-based New Password Exit (ICHPWX01)

z/OS Security Server RACF®

Author: Bruce R. Wells

brwells@us.ibm.com

Last updated: 10/23/2014

<u>Change Date</u>	<u>Change Description</u>
11/04/2008	Introduction of exit
09/25/2009	<ul style="list-style-type: none">• Documentation and code comment changes due to R11 SYSREXX enhancements• Pointer to SYSREXX PTF fixing TSO LOGON problem on R9• Documentation update to mention the R11 IRRXUTIL interface• Enhanced debugging information
10/23/2009	Fixed save area chaining bug in ICHPWX01 code
10/23/2014	<p>In ICHPWX01:</p> <ul style="list-style-type: none">• Change AXREXX macro to specify TSO=NO• Use subpool 229 for autodata• Increase timeout from two to five seconds• Add eyecatcher for use by IRRPWREX <p>In IRRPWREX:</p> <ul style="list-style-type: none">• Add special characters to list of allowable characters• Add several new checks• Add easy enablement option for STIG compliance• Add ability to list active settings using a console command

1 Introduction

In z/OS® R8, a new system component called System REXX (SYSREXX) was introduced. SYSREXX provides the ability for REXX execs to be executed outside of conventional TSO/E and Batch environments. SYSREXX was a web deliverable on R8 and first appeared in the base in R9.

In z/OS R9, RACF provided a new password phrase exit (ICHPWX11) sample which utilizes SYSREXX in order to call a REXX exec named IRRPHREX, in which password phrase quality rules can be coded.

What this download provides is a similar capability for the new password exit, ICHPWX01. A sample ICHPWX01 assembler module is provided, as well as a REXX exec IRRPWREX which it calls. Once you have modified ICHPWX01 to your liking and install it, then subsequent changes made to IRRPWREX will not require an IPL to take effect.

The ICHPWX01 exit gets control from RACF after the SETROPTS password syntax rules have been applied, and only if they are successful. It is your decision either to replace your SETROPTS password rules with ICHPWX01, or use ICHPWX01 to implement only those rules which are not possible using SETROPTS. I recommend implementing whatever rules you can using the SETROPTS rules because

1. They will perform faster, and can more quickly reject a new password value that does not meet your policy.
2. They provide a backstop set of rules in the event of a system problem (see below, where the override mechanism is described).

This download is intended to run on z/OS R9 and later releases.

Attention: PMR 29043,621,760 documents an intermittent deadlock condition when changing a password during console logon with this sample new password exit active. It only occurs under a heavy system load, and we have seen only one instance of this occurring. I have been advised that using TSO=NO on the AXREXX macro in ICHPWX01 will reduce, but not eliminate, the likelihood of the hang. Further, there will be a System REXX fix in a future release which completely solves the problem, but only for TSO=NO. Therefore, the current version of ICHPWX01 specifies TSO=NO.

2 Package Contents

This package contains

- This README file
- ICHPWX01.txt - the source code for the ICHPWX01 exit (use this file for casual viewing)
- IRRPWREX.txt – the sample REXX exec called by ICHPWX01 (use this file for casual viewing)
- ICHPWX01.xmit – Both source parts in TSO XMIT format (use this file for installing)

3 Overview

3.1 ICHPWX01

ICHPWX01 is the assembler part of the exit, and is installed in the normal RACF fashion using the instructions documented in the RACF System Programmer's Guide. It takes each of the parameters passed to it by RACF and formats it into a REXX argument to be sent to IRRPWREX. Address values are not useful within IRRPWREX at this time (Although the REXX STORAGE function is available in read-only mode, it does not support cross-memory storage references). For this reason, ICHPWX01 passes the data pointed to by the address where possible, rather than the address itself. In some cases, address values are passed, in case SYSREXX changes in the future. ICHPWREX may conceivably find the user name and the installation data from the user profile to be of use. Since these are generally contained in the ACEE, and the ACEE address is not very useful, ICHPWX01 specifically locates these fields and sends them as character arguments to IRRPWREX. In fact, from the REXX perspective, this processing improves upon the architected exit input, since in some cases the ACEE is not fully initialized. In these cases, ICHPWX01 will extract the user name and installation data from the user profile.

Note: Beginning in z/OS V1R11, IRRXUTIL provides a REXX interface to R_admin extract. IRRXUTIL can be used from IRRPWREX running under SYSREXX. As such, your password rules may use virtually any RACF profile field in order to make decisions. For example, you might choose to enforce stricter rules for users with powerful privileges such as the SPECIAL attribute.

The AXREXX macro is used to call IRRPWREX from ICHPWX01. Several keyword values are specified on this call.

- The name of the REXX exec is specified as IRRPWREX.
- A 5-second time limit is enforced on this call.
- TSO=NO is specified in order to have the request run in the AXR address space with up to 63 other concurrently running execs. The alternative is to specify TSO=YES to run the exec in its own address space. This can have implications as there are a maximum of 8 address spaces to field TSO=YES calls at a given time, and requests will be queued and run as these address spaces free up. Also, see the "Attention" box above.
- SECURITY=BYAXRUSER is specified so that the exec runs under the identity of the user ID defined in the AXRUSER parameter in the SYSREXX PARMLIB member AXRxx. Using the BYAXRUSER value has the advantage that, if you modify the REXX exec to open a dataset, for example, you need only grant the AXRUSER user ID this RACF authority, as opposed to needing to grant everyone that authority were the exec to run under the authority of the end user. The same consideration would apply to the use of the R_admin callable service if the IRRXUTIL interface is used.

Note that there is no default value for AXRUSER, so you *must* either define a value, or change ICHPWX01 to specify (or default to) SECURITY=BYUTOKEN. If you change

ICHPWX01 in this regard, keep in mind that there will be cases, such as TSO or console logon, where the current security environment is either un-initialized or is a SAF-created ACEE, and the AXREXX call will fail. In these cases, you would need to build your own UTOKEN for the appropriate user ID and pass that into AXREXX.

You may want to tailor these values, or even exploit additional facilities of AXREXX not used by ICHPWX01. For example, the REXXOUTDSN keyword provides the ability to write REXX 'say' output to a dataset (this could be used as an alternative to WTO-ing the debug data to the console, for example). If you wish to use different values than are specified, or to exploit other SYSREXX capabilities, you will need to modify the ICHPWX01 source as shipped. Remember that if you change the name of the REXX exec, ICHPWX01 will need to be modified to specify the new name.

If IRRPWREX fails the new password value, ICHPWX01 simply returns to RACF with the appropriate return code, at which time RACF will display a message to the user, such as:

```
PASSWORD CHANGE FOR 'id' REJECTED BY INSTALLATION PASSWORD EXIT
```

IRRPWREX does, however, pass back a reason code to ICHPWX01 indicating why the new password was rejected. So, ICHPWX01 can be modified to issue a more detailed message if that is what you want to do.

If the call to IRRPWREX fails, due to SYSREXX error or timeout, ICHPWX01 will by default write a message to the operator console and will fail the password change request (see the message description below). However, an override mechanism is provided, and is described in the following section.

3.1.1 System Programmer Override

Being paranoid, we like to anticipate problems that we hope you will never encounter. One such problem would be an extremely busy system, where the only one who could fix or debug such a system is not currently logged on, and whose password is expired. Such a user could get shut out by a WLM policy that gives low priority to LOGONs, which in turn will need the services of system REXX in order to check the new password value. This could cause a timeout of the AXREXX call, or an outright failure from the AXREXX call. The ICHPWX01 sample is written so that by default, any AXREXX failure results in an exit failure, because we can't be sure your password policy has been met, and the password change is rejected.

To address such a situation, if an AXREXX error is encountered when trying to invoke the REXX exec, an override mechanism is checked. The override is only checked for RACROUTE REQUEST=VERIFY/X (i.e. LOGON attempts), and not for the RACF TSO commands. The override is implemented using a resource named IRR.ICHPWX01.OVERRIDE in the FACILITY class. If a user has at least READ access to this resource, the exit returns a successful return code. This means that the user's new password will have met the rules specified on the SETROPTS command, but will not necessarily have satisfied any of the rules coded in the REXX exec.

In order to check the override, ICHPWX01 must first create an ACEE for the user, using an internal RACROUTE REQUEST=VERIFY. If this fails, then we have just suffered our second unexpected failure (AXREXX being the first one). At this point, ICHPWX01 decides to allow the password change (again, subject to the SETROPTS password rules which have already been applied), in the interest of not locking out a system programmer. If you would rather have the exit fail this request, there are comments within ICHPWX01 telling you how to make the minor change that is required.

One final note: If you have also implemented the REXX-based new password phrase exit (ICHPWX11) sample (shipped in SYS1.SAMPLIB), it has essentially the same override mechanism, except that it uses a resource name of IRR.ICHPWX11.OVERRIDE. You can implement both overrides with the same RACF profile using a generic name such as IRR.ICHPWX%1.OVERRIDE.

3.1.2 A Note on Workload Manager (WLM)

During TSO command processing, the System REXX work is charged to the TSO user from a WLM perspective. However, when ICHPWX01 is invoked during TSO LOGON, the WLM environment is not yet initialized. In this case, the work is charged to the AXR (System REXX) address space itself.

3.2 IRRPWREX

IRRPWREX, being REXX, is pretty well self-documented. Its operation is based on configuration settings towards the top of the file, which you use to enable and customize checks. You can of course add your own. It's fun! The beauty of SYSREXX is that, once you have ICHPWX01 the way you like it, and have IPLed the system so that RACF recognizes the exit, subsequent changes made to IRRPWREX take effect immediately upon saving the changes. No IPL is required!

3.2.1 Checks provided by IRRPWREX

As shipped, the exec supports the following features and checks:

- A debug mode which will write the input arguments (except the password value) to the console
- A STIG compliance variable which automatically enables the subsequent checks that are required by the DISA STIG
- A minimum length check
- An allowable characters list
- A check to enforce a minimum number of character types (numbers, lower case letters, upper case letters, and national characters) which must exist in the new password (note that this capability now exists in native RACF password syntax rules with APAR OA43999, using the MIXEDALL content keyword).
- A check that the user's name is not contained in the password
- A check that the user's user ID is not contained in the password

- A check against trivial changes with respect to the previous password
- A check to force a minimum number of different positional characters with respect to the previous password
- Similar to above, but regardless of position
- A check for repeating characters
- A check to make sure any given character in the new password is not used more than once
- A check for consecutive characters (e.g “ab” or “34”)
- A check for repeating characters
- A check against a user-specified list of restricted words
- A check against a user-specified list of restricted prefixes
- A check against a user-specified list of restricted password patterns (a.k.a “topologies”)

These checks were selected to demonstrate various REXX functions, to satisfy various requirements, and because they were fun! You will find that some checks which are impossible, or clumsy, to enforce with the RACF SETROPTS rules, or an assembler exit, are trivial in REXX.

3.2.2 Reporting options provided by IRRPWREX

System REXX provides a console command by which to invoke any exec in the REXXLIB concatenation. We exploit this capability in order to provide the ability to list the rules being enforced by IRRPWREX. The output is issued by IRRPWREX using the AXRMLWTO (multi-line WTO) function provided by System REXX. If you add or modify existing checks, make sure you keep the listing code in sync with your changes, if you wish to provide this capability.

The listing functions also verify that IRRPWREX is being called for password validation by:

1. Checking that the ICHPWX01 exit is active
2. Checking the beginning contents of ICHPWX01 to see if it contains the string ‘IRRPWREX’

A message is displayed if either of these conditions is not true. The ‘IRRPWREX’ eye-catcher string is added to ICHPWX01 as part of the 10/23/2014 functional enhancement, which requires an IPL in order to activate the change. Alternatively, you can simply ignore the message or alter IRRPWREX to not issue the message.

You do not need to be logged on to the console when issuing the listing command with the sample provided. However, if you modify the code, for example to use IRRXUTIL or to access data sets, you may need to log on to the console to issue the command.

Two versions of output are supported: human-friendly, and program-friendly.

3.2.2.1 Human-friendly listing

From a console, issue the following command to get a description of the active checks in English:

```
F AXR,IRRPWREX list
```

The “list” argument is passed to IRRPWREX. The argument is not case sensitive. In the interest of keeping down the noise, only active checks are reported, where ‘active’ means that the configuration setting was changed from the value as provided (which is designed to be functionally equivalent to native RACF).

Example 1: Issue the list command when IRRPWREX is unchanged.

```
00- SYS1 f axr,irrpwrex list
      SYS1
```

The following IRRPWREX password exit rules are in place:

There are no rules active that RACF does not already enforce

Example 2: Issue the list command after enabling STIG compliance.

```
SYS1 f axr,irrpwrex list
      SYS1
```

The following IRRPWREX password exit rules are in place:

STIG compliance is explicitly specified

The minimum password length is 8

The number of required character types is 4

The user's name cannot be contained in the password

Only 3 consecutive characters of the user's name are allowed

The minimum word length checked is 8

The user ID cannot be contained in the password

Only 3 consecutive characters of the user ID are allowed

Only 3 unchanged positions of the current password are allowed

These positions need to be consecutive to cause a failure

This check is not case sensitive

No consecutive characters (e.g. AB or 12) are allowed

This check is not case sensitive

A list of 33 restricted prefix strings is being checked:

```
APPL APR AUG ASDF BASIC CADAM DEC DEMO FEB FOCUS GAME IBM JAN JUL
JUN LOG MAR MAY NET NEW NOV OCT PASS ROS SEP SIGN SYS TEST TSO
VALID VTAM XXX 1234
```

3.2.2.2 Program-friendly listing

From a console, issue the following command to get a *name:value* listing of the defined configuration variables:

```
F AXR,IRRPWREX listconfig
```

The argument can be abbreviated as “listc”. For LISTC all defined configuration variables are displayed, even if they have not been changed from the version as provided. One variable is displayed per line. The dictionary and prefix stem variables containing the number of entries is listed, even if its value is 0, but the individual entries are printed (one per line) only if the value is greater than 0.

Example 1: Issue the listc command when IRRPWREX is unchanged.

```
SYS1 f axr,irrpwrex listc
SYS1
The following IRRPWREX configuration variables are defined:
STIG_COMPLIANT:no
PWD_MINLEN:1
PWD_REQ_TYPES:1
PWD_NAME_ALLOWED:yes
  PWD_NAME_MINLEN:3
  PWD_NAME_CHARS:4
PWD_USERID_ALLOWED:yes
  PWD_USERID_CHARS:4
PWD_TRIVIALITY:no
PWD_MIN_UNIQUE:0
  PWD_MIN_UNIQUE_UPPER:yes
PWD_MAX_UNCHANGED:8
  PWD_MAX_UNCHANGED_UPPER:yes
  PWD_MAX_UNCHANGED_CONSECUTIVE:yes
PWD_ALL_UNIQUE:no
PWD_NO_CONSECUTIVE:no
  PWD_NO_CONSECUTIVE_UPPER:yes
PWD_MIN_NEW:0
PWD_REPEAT_CHARS:7
  PWD_REPEAT_UPPER:yes
PWD_DICT.0:0
PWD_PREFIX.0:0
PWD_PATTERN.0:0
```

Example 2: Issue the list command after enabling STIG compliance.

```
SYS1 f axr,irrpwrex listc
SYS1
The following IRRPWREX configuration variables are defined:
STIG_COMPLIANT:yes
PWD_MINLEN:8
```

PWD_REQ_TYPES:4
PWD_NAME_ALLOWED:no
 PWD_NAME_MINLEN:8
 PWD_NAME_CHARS:4
PWD_USERID_ALLOWED:no
 PWD_USERID_CHARS:4
PWD_TRIVIALITY:no
PWD_MIN_UNIQUE:0
 PWD_MIN_UNIQUE_UPPER:yes
PWD_MAX_UNCHANGED:3
 PWD_MAX_UNCHANGED_UPPER:yes
 PWD_MAX_UNCHANGED_CONSECUTIVE:yes
PWD_ALL_UNIQUE:no
PWD_NO_CONSECUTIVE:yes
 PWD_NO_CONSECUTIVE_UPPER:yes
PWD_MIN_NEW:0
PWD_REPEAT_CHARS:7
 PWD_REPEAT_UPPER:yes
PWD_DICT.0:0
PWD_PREFIX.0:33
 PWD_PREFIX.1:APPL
 PWD_PREFIX.2:APR
 PWD_PREFIX.3:AUG
 PWD_PREFIX.4:ASDF
 PWD_PREFIX.5:BASIC
 PWD_PREFIX.6:CADAM
 PWD_PREFIX.7:DEC
 PWD_PREFIX.8:DEMO
 PWD_PREFIX.9:FEB
 PWD_PREFIX.10:FOCUS
 PWD_PREFIX.11:GAME
 PWD_PREFIX.12:IBM
 PWD_PREFIX.13:JAN
 PWD_PREFIX.14:JUL
 PWD_PREFIX.15:JUN
 PWD_PREFIX.16:LOG
 PWD_PREFIX.17:MAR
 PWD_PREFIX.18:MAY
 PWD_PREFIX.19:NET
 PWD_PREFIX.20:NEW
 PWD_PREFIX.21:NOV
 PWD_PREFIX.22:OCT
 PWD_PREFIX.23:PASS
 PWD_PREFIX.24:ROS
 PWD_PREFIX.25:SEP
 PWD_PREFIX.26:SIGN
 PWD_PREFIX.27:SYS

PWD_PREFIX.28:TEST
PWD_PREFIX.29:TSO
PWD_PREFIX.30:VALID
PWD_PREFIX.31:VTAM
PWD_PREFIX.32:XXX
PWD_PREFIX.33:1234
PWD_PATTERN.0:0

4 Installation Instructions

1. Download the source code (ichpwx01.xmit) to your workstation using a browser. Then transfer it to your z/OS system using FTP in binary mode. You must transfer it into a fixed-block 80 data set. For example, on Windows, this can be accomplished by specifying the following FTP client command before initiating the transfer

```
quote site lrecl=80 recfm=fb blksize=0
```

2. From a TSO session on z/OS, issue the RECEIVE command to unpackage the file. The syntax of the RECEIVE command is:

```
RECEIVE INDATASET(dsname)
```

RECEIVE prompts you for a target data set name.

Note: If you receive a message from the RECEIVE command that indicates the input data set is in an incorrect format, verify that:

- The files were FTP'd in binary format
- The input files are in fixed block format

3. Inspect the ICHPWX01 source code, and modify it if desired.
4. Assemble the ICHPWX01 source.
5. Link-edit the object code into an LPA library. For example:

```
//PWXLINK EXEC PGM=IEWL, PARM='NCAL, LIST, LET, XREF, RENT, SIZE=(300K, 30K) '  
//SYSPRINT DD SYSOUT=*  
//SYSUT1 DD DSN=&SYSUT1, UNIT=SYSDA, SPACE=(1024, (50, 20))  
//SYSLMOD DD DISP=SHR, DSN=PROD.LPALIB  
//AOSBN DD DSN=MY.ICHPWX01.OBJ, DISP=SHR  
//SYSLIN DD *  
INCLUDE AOSBN(ICHPWX01)  
ENTRY ICHPWX01  
NAME ICHPWX01(R)  
/*
```

6. Copy IRRPWREX into a library defined to the REXXLIB concatenation. Note that before SYSREXX added REXXLIB support, the exec must have resided in the SYS1.SAXREXEC library. The concatenation is defined in the active AXRxx member of SYS1.PARMLIB.
7. Re-IPL your system. Verify that ICHPWX01 appears in the ICH508I message issued at initialization to identify the active RACF exits.

It is, of course, a good idea to try this on a test system first! Change IRRPWREX to enable debug mode, change a password, and verify that IRRPWREX dumps its input arguments to the operator console.

Important Note!!! IRRPWREX is a logical extension of the ICHPWX01 new password exit, and as such, performs security-sensitive function. In fact, many of the execs in the REXXLIB concatenation will share this concern. As such, REXXLIB data sets should be protected by RACF as you would any APF authorized library! Note that the Health Check named RACF_SENSITIVE_RESOURCES will check for this.

5 Messages

NEW PASSWORD EXIT ICHPWX01 ENCOUNTERED AN UNEXPECTED ERROR: AXREXX RETURN CODE XXXXXXXX REASON CODE XXXXXXXX

Explanation: The call to IRRPWREX from ICHPWX01 failed. This does not mean that IRRPWREX rejected the new password value. It means IRRPWREX was never called, or did not return properly. The return and reason code from the AXREXX macro invocation are displayed.

System action: The password change request fails and system processing continues.

System Programmer Action: This message may indicate a problem with the System REXX environment. Look up the return and reason codes in the System REXX documentation (see the references below). Specifically, search the appropriate book for a string of the format *xxxxRSN*, where 'xxxx' is a literal string, and *RSN* is the low order halfword of the reason code displayed in the message. For example, if the following were contained in the WTO:

```
AXREXX RETURN CODE 00000008 REASON CODE 05030828
```

You would search for 'xxxx0828' and you would find, under return code 8:

Equate Symbol:

```
AXRExecSyntaxError
```

Meaning: A syntax error or another run time error was encountered during the execution of a REXX exec.

Indicating that your REXX code in IRRPWREX had a syntax error.

A return code value of X'C' coupled with a reason code value ending in X'0C0A' indicates that the REXX exec timed out. In this case, you might consider increasing the timeout value of five seconds which is coded in ICHPWX01.

NEW PASSWORD EXIT ICHPWX01 ENCOUNTERED A RACROUTE ERROR: VERIFY SAFRC XXXXXXXX RACFRC XXXXXXXX RACFREAS XXXXXXXX

Explanation: After having encountered an AXREXX error, ICHPWX01 attempted to build an ACEE for the user in order to check her authority to the override resource in RACF (IRR.ICHPWX01.OVERRIDE in the FACILITY class). However, the RACROUTE REQUEST=VERIFY was unsuccessful. The return and reason codes from the RACROUTE macro invocation are displayed.

System action: The success of the password change request is completely based on the SETROPTS password rules that are in effect.

System Programmer Action: Look up the return and reason codes in the RACROUTE documentation. When the password was being changed during LOGON, it's possible that the failing condition was simply caught by the ICHPWX01 exit before normal LOGON processing had a chance to (that is, the RACROUTE REQUEST=VERIFY issued by TSO which caused ICHPWX01 to get control in the first place).

6 References

For an overview of System REXX, see [z/OS MVS Authorized Assembler Services Guide](#).

For details on the AXREXX macro, including return and reason code values, see [z/OS MVS Authorized Assembler Services Reference \(ALE-DYN\)](#).

For documentation on installing RACF exits, see [z/OS Security Server RACF System Programmer's Guide](#).

For details on the RACROUTE macro, see [z/OS Security Server RACROUTE Macro Reference](#).

For details on the IRRXUTIL interface to retrieve RACF profile and SETROPTS data from REXX, see (the R11 or higher version of) [z/OS Security Server RACF Macros and Interfaces](#).

7 Disclaimers, etc.

This program contains code made available by IBM® Corporation on an AS IS basis. Any one receiving this program is considered to be licensed under IBM copyrights to use the IBM-provided source code in any way he or she deems fit, including copying it, compiling it, modifying it, and redistributing it, with or without modifications, except that it may be neither sold nor incorporated within a product that is sold. No license under any IBM patents or patent applications is to be implied from this copyright license.

The software is provided "as-is", and IBM disclaims all warranties, express or implied, including but not limited to implied warranties of merchantability or fitness for a particular purpose. IBM shall not be liable for any direct, indirect, incidental, special or consequential damages arising out of this agreement or the use or operation of the software.

A user of this program should understand that IBM cannot provide technical support for the program and will not be responsible for any consequences of use of the program.