

[print](#) | [close](#)

## Password Change Blocking for V5R4 and Earlier

[System iNetwork Systems Management Newsletter](#)

[Carsten Flensburg](#)

Carsten Flensburg

Wed, 07/02/2008 (All day)

I just read the latest newsletter and noted the article discussing the 6.1 QPWDCHGBLK system value and wondered what could be done for earlier releases to enforce a similar rule. The article mentions the requirement of getting at the password change date and time, and that prompted my recollection of a discovery I made some time ago: While the documentation of the [Retrieve User Information \(QSYRUSRI\) API](#) refers to the Password Change \_Date\_ as part of the USRIO100 format, the API in reality returns a system timestamp (\*DTS) value that can be converted (using f.x. the QWCCVTDT API) to a timestamp with microsecond precision.

This enables you to check how many hours have passed since a password was changed, and thereby establish a check similar to the one enforced by the new QPWDCHGBLK system value. Using the [user application APIs](#) you could even store individual PWDCHGBLK values for each user profile, if you needed that granularity.

The objective of the password validation process in this program is to prevent users from changing passwords again within a given time frame from when the password was last changed. You should change the program constant PWD\_CHGBLK\_HOURS to the number of hours for which you want to block a repeated change of a user profile's password. As mentioned below, you could also maintain user profile individual blocking values using the User Application APIs.

After proper registration of this program as a password validation exit program (see below), this program will be called whenever the Change Password (CHGPWD) command or the Change Password (QSYCHGPW) API is executed.

[Exit point documentation:](#)

Exit point registration:

```
ChgSysVal  SysVal( QPWDVLDPGM )
           Value( *REGFAC )

           AddExitPgm ExitPnt( QIBM_QSY_VLD_PASSWRD )
           Format( VLDP0100 )
           PgmNbr( 1 )
           Pgm( /CBX707 )
           Text( 'Password validation exit program' )
```

```
**  Program . . : CBX707
**  Description : Password validation exit program
**  Author . . : Carsten Flensburg
**  Published . : System iNetwork Systems Management Tips
```

```

Newsletter
**
**
**
**  Program description:
**
**    After proper registration of this program as a password
validation
**    exit program (see below), this program will be called
whenever the
**    Change Password (CHGPWD) command or the Change Password
(QSYCHGPW)
**    API is executed.
**
**    More than one program can be registered to the password
validation
**    exit point and the validation programs will be called in
turn until
**    all programs have been called - or a until a reject return
code is
**    received.
**
**    The objective of the password validation process in this
program
**    is to prevent users from changing passwords again within a
given
**    time frame from when the password was last changed.  You
should
**    change the program constant PWD_CHGBLK_HOURS to the number
of hours
**    for which you want to block a repeated change of a user
profile's
**    password.
**
**
**
**
**  Exit point registration:
**
**    ChgSysVal  SysVal( QPWDVLDPGM )
**                Value( *REGFAC )
**
**    AddExitPgm ExitPnt( QIBM_QSY_VLD_PASSWRD )
**                Format( VLDP0100 )
**                PgmNbr( 1 )
**                Pgm( /CBX707 )
**                Text( 'Password validation exit program' )
**
**
**  Compilation specification:
**
**    CrtRpgMod  Module( CBX707 )
**                DbgView( *NONE )
**                Aut( *USE )
**

```

```

**      CrtPgm      Pgm( CBX707 )
**                  Module( CBX707 )
**                  ActGrp( *NEW )
**                  Aut( *USE )
**
**      ChgPgm      Pgm( CBX707 )
**                  RmvObs( *ALL )
**
**
**-- Header specifications:
-----**
      H Option( *SrcStmt )

**-- Exit format VLDP0100:
D VLDP0100_T      Ds      65535      Qualified Based( p_T )
D  ExpNam          20a
D  ExpFmtNam       8a
D  PwdLvl          10i 0
D  UsrPrf          10a
D                  2a
D  OfsOldPwd       10i 0
D  LenOldPwd       10i 0
D  CcsOldPwd       10i 0
D  OfsNewPwd       10i 0
D  LenNewPwd       10i 0
D  CcsNewPwd       10i 0
**
D OldPwd           s      128a      Varying
D NewPwd           s      128a      Varying

**-- Global constants:
D PWD_CHGBLK_...
D  HOURS           c      24
**
D PWD_ACCEPT       c      '0'
D PWD_REJECT       c      '1'
**-- Global variables:
D DFT_RTNCOD       s      n      Inz( PWD_REJECT )
D PwdChgDts        s      z

**-- Retrieve user information:
D RtvUsrInf        Pr      ExtPgm( 'QSYRUSRI' )
D  RcvVar          32767a      Options( *VarSize )
D  RcvVarLen       10i 0 Const
D  FmtNam          10a      Const
D  UsrPrf          10a      Const
D  Error           32767a      Options( *VarSize )
**-- Convert date & time:
D CvtDtf           Pr      ExtPgm( 'QWCCVTDT' )
D  InpFmt          10a      Const
D  InpVar          17a      Const Options( *VarSize )
D  OutFmt          10a      Const
D  OutVar          17a      Options( *VarSize )
D  Error           10i 0 Const

```

```

D  InpTimZon          10a  Const  Options( *NoPass )
D  OutTimZon          10a  Const  Options( *NoPass )
D  TimZonInf          111a          Options( *VarSize:
*NoPass )
D  TimZonInfLen       10i 0 Const  Options( *NoPass )
D  PrcInd             1a  Const  Options( *NoPass )
D  InpTimInd          1a  Const  Options( *NoPass )

**-- Get user profile password system *DTS:
D  GetPwdSdt          Pr          8a
D  PxUsrPrf           10a  Value
**-- Convert system *DTS to timestamp:
D  CvtSdtDts          Pr          z
D  PxSysDts           8a  Value

**-- Parameters:
D  CBX707             Pr
D  VLDP0100                                LikeDs( VLDP0100_T )
D  PxRtnInd           1a
**
D  CBX707             Pi
D  VLDP0100                                LikeDs( VLDP0100_T )
D  PxRtnInd           1a

/Free

PxRtnInd = PWD_ACCEPT;

If  VLDP0100.ExpNam = 'QIBM_QSY_VLD_PASSWRD' And
    VLDP0100.ExpFmtNam = 'VLDP0100';

    OldPwd = %Subst( VLDP0100: VLDP0100.OfsOldPwd+1:
VLDP0100.LenOldPwd );
    NewPwd = %Subst( VLDP0100: VLDP0100.OfsNewPwd+1:
VLDP0100.LenNewPwd );

    PwdChgDts = CvtSdtDts( GetPwdSdt( VLDP0100.UsrPrf ) );

    If %Diff( %TimeStamp(): PwdChgDts: *HOURS ) *Zero;
    Return *Blanks;

When  USRI0100.PwdChgSdt = *Blanks;
    Return *Blanks;

Other;
    Return  USRI0100.PwdChgSdt;
EndSl;

/End-Free

P  GetPwdSdt          E
**-- Convert system *DTS to timestamp:
P  CvtSdtDts          B
D                                Pi          z

```

```

D   PxSysDts                                8a   Value

**-- Local constants:
D MIC_SEC                                c           '1'
**-- Local variables:
D SysDts                                s           20a
**
D TimZonInf                                Ds           Qualified
D BytRtn                                10i 0
D BytAvl                                10i 0
D TimZonNam                                10a
D                                           1a
D CurDstInd                                1a
D CurUtcOfs                                10i 0
D CurZonNam                                50a
D CurAbvNam                                10a
D CurMsgId                                7a
D CurMsgF                                10a
D CurMsgFlib                             10a
**

/Free

If PxSysDts = *Blanks;
    Return  *LoVal;

Else;
    CvtDtf( '*DTS'
            : PxSysDts
            : '*YYMD'
            : SysDts
            : *Zero
            : '*SYS'
            : '*SYS'
            : TimZonInf
            : %Size( TimZonInf )
            : MIC_SEC
            );

    Return  %Timestamp( SysDts: *ISO0 );
EndIf;

/End-Free

P CvtSdtDts                                E

```

**Source URL:** <http://iprodeveloper.com/systems-management/password-change-blocking-v5r4-and-earlier>