Paper CT-28

Encoding the Password

A low maintenance way to secure your data access

Leanne Tang, National Agriculture Statistics Services USDA, Washington DC

ABSTRACT

When users access data in a remote database through the SAS® application, the valid database user ID and password are required. One method for granting user access is to create a login ID for each user in each database. Doing so requires the maintenance of the login ID for each user on each remote database and is subject to the risk of exposing the user passwords in the SAS programs. This paper is going to present a simple solution to overcome the security concerns by using the Application Login ID for accessing databases. Then we encode the Application Password by using the PROC PWENCODE procedure. By implementing the Application Login ID on the database and encoding the Application Password in the SAS program we can eliminate the maintenance of individual user login accounts on the databases and disguise the passwords from to be revealed as clear text in the SAS programs.

INTRODUCTION

In our organization, most data are stored in the remote relational database. Users need to access the database to query data for analysis in the SAS applications. The database we need to access is an analytical database with only select access granted to the users. The database stores data from different organizational units and for different projects. In order to allow users accessing their data from the SAS application in a simple and secure way, the following four steps are carried out:

1. Create an Application Login ID and Password in the database for the SAS application.
2. Run the PROC PWENCODE procedure to encode the password and save the encoded password in a file.
3. Every time users invoke the SAS application, the encoded password is retrieved for connecting to the remote database.
4. Based on the user's organizational unit, a set of data is queried for analysis.

This paper will focus on the PROC PWENCODE procedure and its options.

PROC PWENCODE SYNTAX AND EXAMPLES

The PWENCODE procedure enables you to encode passwords. Encoded passwords can be used in place of plain text passwords in SAS programs that access remote databases and various other servers.

Here is the syntax for PROC PWENCODE from the SAS OnlineDoc:

```
PROC PWENCODE IN = 'password'
        <OUT=fileref> <METHOD=encodingmethod>;
RUN ;
```

There are two options and 3 different parameters available for the procedure. The <OUT=fileref> is used to specify a file reference for the encoded password. To capture the encoded password in an external file use the <OUT=fileref> option. Otherwise, the encoded password will be output to the SAS log.

There are three encoding methods for PROC PWENCODE listed in the table 1 below. The {sas001} and {sas002} are available with base SAS but {sas003} requires a separate license. If the method option is omitted, the default encoding method, {sas002}, is used automatically.

Table 1. The Three Encoding Methods for PROC PWENCODE from SAS Online Doc:

Encoding Method	*Description*	*Supported Data Encryption Algorithm*
sas001	Uses base64 to encode passwords	None
Sas002 (or sasenc)	Uses a 32-bit key to encode passwords	default SAS Proprietary, which is included in SAS software.
sas003	Uses a 256-bit key to encode passwords	AES (Advanced Encryption Standard), which is supported in SAS/SECURE

The following three examples are going to demonstrate the use of PROC PWENCODE and the options.

Example One, the Core Program:

In this first example, we are encoding the password, myPassword, using PROC PWENCODE with no options specified.

```
PROC PWENCODE IN='myPassword';
RUN ;
```

After executing the program, a long string of the encoded password is output to the SAS Log. Notice at the beginning of the string, {sas002}, this is a tag automatically attached to the string of the encoded password. The tag is used for SAS Server to recognize the encoded string and to identify the decoding method for interpreting the string. Since we did not define the method in the example, the default method, {sas002} is automatically applied.

Output 1. The SAS Log for Example One:

```
{sas002}68B27956211D2DB651E029C001E3D23059C7D7C8

NOTE: PROCEDURE PWENCODE used (Total process time):
    real time          0.00 seconds
    cpu time           0.00 seconds
```

Example Two, the Option for Saving the Encoded Password:

The encoded password can be saved to a file by specifying a file reference using the <OUT=> option. The saved encoded password can then be retrieved and used in other SAS programs.

```
FILENAME PwdFile "O:\EES\SAS User Groups\encodedPwd.txt" ;
PROC PWENCODE IN='myPassword' OUT= PwdFile ;
RUN ;
```

Since we directed the output to a file in this example no encoded password is output to the SAS Log.

Output 2. The SAS Log for Example Two:

```
1   FILENAME PwdFile "O:\EES\SAS User Groups\encodedPwd.txt" ;
2
3   PROC PWENCODE IN=XXXXXXXXXXXX out= PwdFile ;
4   RUN ;

NOTE: PROCEDURE PWENCODE used (Total process time):
    real time          0.01 seconds
    cpu time           0.00 seconds
```

The figure 1 below displays the saved encoded password in a text file. Same as in example one, the encoded password in the output file started with a default tag, {sas002}, because we did not specify the encoding method and default method, {sas002}, was applied.

Figure 1. The File with the Saved Encoded Password:

Example Three, Specifying the Encoding Method:

As discussed before, there are three methods to encode a password, {sas001}, {sas002}, and {sas003}. When a specific method is defined in the PROC PWENCODE procedure the correlated tag is attached at the beginning of the encoded string. In this example, we are using the encoding method {sas001}. So the {sas001} tag is attached at the beginning of the string. Also notice, the encoded string in the SAS log is different from the previous examples. Even though we are encoding the same password but the encoding method we applied here is different.

```
PROC PWENCODE IN='myPassword' METHOD=sas001;
RUN ;
```

Output 3. The SAS Log for Example Three:

```
{sas001}bXlQYXNzd29yZA==

NOTE: PROCEDURE PWENCODE used (Total process time):
    real time          0.03 seconds
    cpu time           0.03 seconds
```

USING THE ENCODED PASSWORD IN SAS PROGRAMS

After the passwords are encoded by using the PROC PWENCODE procedure the encoded passwords can be used in SAS programs to replace the clear text passwords.

Example Four, Using the Encoded Password in LIBNAME Statement:

In this example, we are using the encoded password to assign SAS library reference to a remote database using ODBC data source.

```
LIBNAME myLib ODBC
        USER=myAppID
        PW="{sas002}526FE20D05B4D88F023236E42C9D0E4520E1ECF1"
```

```
        DSN=myDatabase CONNECTION=UNIQUE ;
RUN ;
```

Output 4. The SAS Log for LIBNAME Statement:

```
1   LIBNAME myLib ODBC
2     user=myAppID
3     pw=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
4     dsn=myDatabase CONNECTION=UNIQUE ;
NOTE: Libref myLib was successfully assigned as follows:
    Engine:       ODBC
    Physical Name: myDatabase
5   RUN ;
```

Example Five, Using the Encoded Password in Proc SQL:

The following example displays the DATA step used to read in the encoded password from a file. The encoded password is then assigned to a macro variable. The macro variable can then be used in SAS programs throughout the application.

```
FILENAME  pwfile "O:\EES\SAS User Groups\encodedPwd.txt" ;
DATA  _NULL_;
        INFILE pwfile OBS=1 LENGTH=a;
        INPUT @1 myPW $varying1024. l;
        CALL SYMPUT('myPW', SUBSTR(myPW,1,a));
RUN;
```

This procedure displays how to use the encoded password in the SQL pass through facility. The password is assigned to the myPW macro variable. After connecting to the remote database through the ODBC data source, data are queried from the myDBTable table.

```
PROC SQL NOPRINT ;
        CONNECT  TO  ODBC(USER=myAppID PW="&myPW" DSN=myDatabase );
        CREATE TABLE myTable AS
                SELECT * FROM CONNECT  to  ODBC
                (SELECT * FROM myDBTable);
QUIT ;
```

ENCODING VS. ENCRYPTION

Encoding is a process of converting one set of meaningful characters into another set. By converting into a different set the characters become unreadable and the meanings of the characters are disguised from the public.

Encryption is a method to transform data from plain text to cipher text through the use of a mathematical algorithmic scheme. Any plain text through encryption process becomes cipher text and is illegible to anyone without a special key.

Though both processes involve converting data from one format to another, encoding process is designed for disguising the data to be revealed casually. Encoding and decoding processes do not require a special key. On the other hand, encryption method is used to protect data from to be revealed to anyone other than the intended recipient. In order to read the encrypted text both the encryption key and the mathematical algorithm are required.

Encoding is intended to disguise data from to be revealed in public. For our purpose, it works well to prevent casual, non-malicious viewing of password in the SAS programs. Because of the special key and mathematical algorithm involved encryption is generally more difficult to break. It is designed for maintaining data confidentiality. Encoding and encryption are developed for different purposes. One should not replace another.

CONCLUSION

The Application Login ID and password eliminate the maintenance of individual user login accounts on the database. The PROC PWENCODE procedure enables the password to be encoded and saved in a secure location. By combining the two steps we reduce the maintenance workload and steer clear the password from revealing in the SAS programs.

Additionally, since individual users don't have the database login account, users are no longer able to access database through other database query tools. The SAS application becomes the solo point of access to the remote database and allows better oversight for the control of accessing data. This is an additional bonus added to our database access security measures.

Even though the PROC PWENCODE procedure is intended for encoding passwords, technically it can be used to encode any strings up to 512 characters in length.

The SAS programs in this paper are tested using SAS 9.2.

REFERENCES

Encryption in SAS 9.2 by SAS Institute. Available at
http://support.sas.com/documentation/cdl/en/secref/62092/PDF/default/secref.pdf

SAS Online Docs, SAS Institute, Cary, NC

ACKNOWLEDGMENTS

Many thanks to Huong Luong for providing the "Application Login ID" and the endless support on the database.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Leanne Tang
National Agricultural Statistics Services, USDA
1400 Independence Ave., SW
Washington, D.C 20250-2001
202 – 720 6957
Leanne_tang@nass.usda.gov