SESUG Paper 193-2018

Password Protection of SAS® Enterprise Guide® Projects

Thomas E. Billings, MUFG Union Bank, N.A., San Francisco, California

ABSTRACT

SAS® Enterprise Guide® lets a user assign a password to a project, after which any/all users must enter the correct password to be able to open and run the project. In this paper we test the security of password protected projects against 2 hacks. We first present a sample project that has multiple types of Tasks that will be used for testing. Before assigning a password to the project, we demonstrate a simple hack that facilitates the examination of some of the internal metadata of a SAS Enterprise Guide project, i.e., a .egp file. Then we assign a password to the sample project and repeat the hack on the password-protected project, with the result that the hack does not work. Next, we try another simple hack - viewing the .egp file in a text editor - and find that this does not work either. The tests here provide independent confirmation that password-protected .egp files are secure against the 2 hacks demonstrated. We end with a brief discussion on .egp file metadata and its role in the SAS ecosystem. SAS products: SAS Enterprise Guide. User-level: beginner.

INTRODUCTION

An important security feature of SAS® Enterprise Guide® is the ability to assign a password to a project, a .egp file. To assign a password to an open project, use the menu:

1. Click: **File → Project Properties → Security**
2. Enter password
3. Click **OK**
4. Reenter password (2nd validation step; required)
5. Save the project

Once the project is password-protected, the project cannot be opened/run unless the relevant password is supplied. To remove password protection from an open, password-protected project, repeat the process above with a "null" password, i.e., don't enter a password in the password field prompts supplied by the **Project Properties** GUI.

This is an important feature as it provides an additional layer of security for SAS Enterprise Guide projects and applications. The obvious question then arises: how secure are password-protected projects? In the sections that follow, we provide a partial answer to this question by trying 2 different hacks. We begin by creating a sample project for testing the hacks.

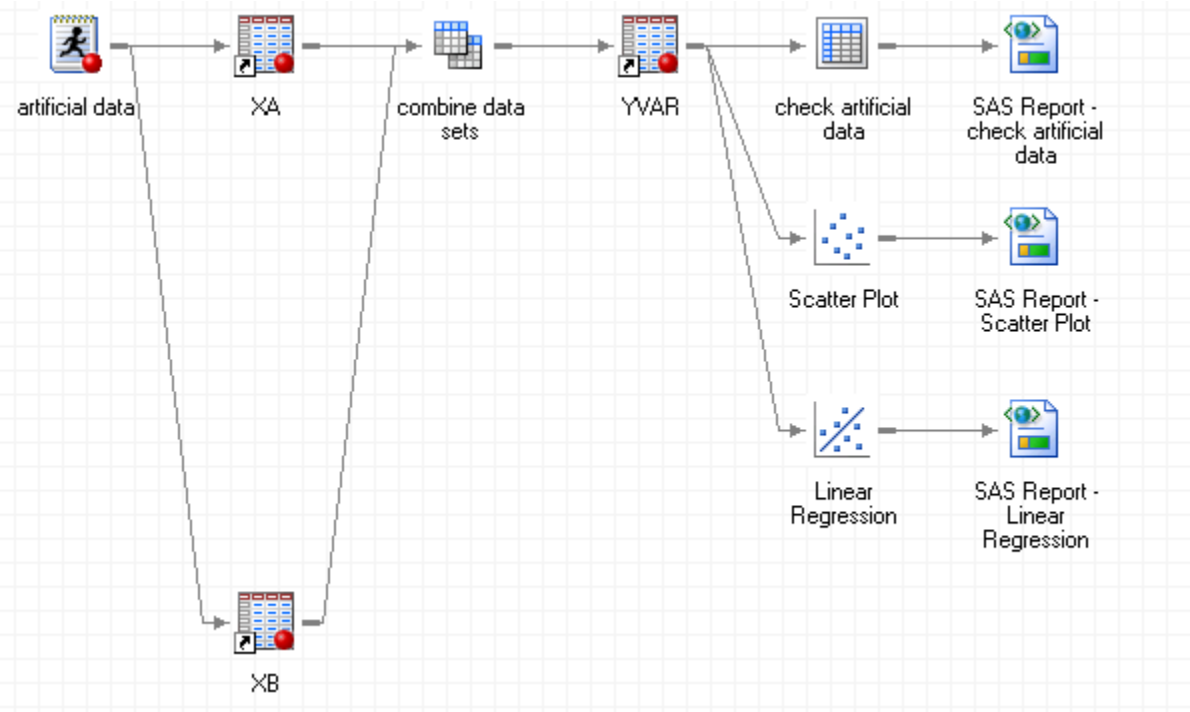
SAMPLE PROJECT

We create a sample EG project that is a regression; simulated data are created and analyzed. The data sets are created in 2 different DATA steps and they are later joined, specifically to include the Query Builder Task (a common and frequently used task) in the project.

The project has 5 tasks. This is more complex than the minimum project required for testing password protection, but the reason for doing it this way will be clear later. The 5 tasks are:

1. Program Task that creates 2 artificial data sets, XA, XB
2. Query Builder Task that joins XA, XB, and creates dependent variable Y
3. One-Way Frequencies Task with a tabulation of (artificial) variable ckval_A (created in step 1)
4. Scatter Plot Task
5. Linear Regression Task - multiple regression (simulation).

The project process flow is shown below.



The first task is a Program task that generates the test data. The code is below and is released under the open source BSD (Berkeley Systems Distribution) copyright license; see Appendix 1 for the license. Two data sets are created: XA with variables X1, E (for epsilon in regression simulation), and data set XB with variable X2.

```
*  generate artifical data for prototype project;

options nocenter dtreset;

%* macro parameters: leave as-is for reproducibility on;
%* Linux/Unix/Windows. Change to get different results;
%global num_rows random_seed1 random_seed2;
%let num_rows=150;
%let random_seed1=94101;
%let random_seed2=32935;

data XA (drop = i);
       length ckval_A $1.;
              do i = 1 to &num_rows;

       if (i=1) then
              call streaminit(&random_seed1.);
       RC = i;
```

```
        X1 = rand("uniform");
        E = rand("normal",0,1);
        ckval_A = ifc(X1 > 0.5,"U","L","L");
        output;
end;
run;

data XB (drop = i);
        length ckval_B $1.;
               do i = 1 to &num_rows;

        if (i=1) then
               call streaminit(&random_seed2.);
        RC = i;
        X2 = rand("uniform");
        ckval_B = ifc(X2 > 0.5,"U","L","L");
        output;
end;
run;
```

In the Query Builder Task (#2 above), XA and XB are joined and variable Y is created; a simulation:

Y = 10*X1 - 5*X2 + E.

The last task (#5) does a multiple regression of Y on (X1, X2).

The next step is to illustrate how hack #1 works on an unprotected project, before we try it on a password-protected project.

HACK #1 ON AN UNPROTECTED PROJECT

SAS Enterprise Guide projects can be saved on the server (often Linux or Unix systems) or on the PC where the SAS Enterprise Guide GUI/interface runs. In the discussion that follows, we assume the project files are on your PC. It is well-known that SAS Enterprise Guide projects are zipped directories that contain metadata (Hemedinger 2014). The fact that these projects are zipped files yields the obvious hack:

1. Create a separate directory (on your Windows PC) for the intended unzip
2. Make a copy of the target project (here, the sample project) in the new directory
3. Rename the *.egp file to *.zip so it can be unzipped
4. Unzip the project .zip file; extract all files.

For the sample project with 5 tasks, we get 7 subdirectories (many of which contain additional subdirectories) and a file, **project.xml**. We can open the project.xml file with a text editor, an xml editor, or in a browser (opening in a browser is the easiest – just double click the file name). Review of this file shows that it contains high-level metadata for the project.

One of the directories is for the Code Task (Program Task). It contains the user-written code in a .sas file, and a subdirectory that contains the log (.log file) from running that Task. There are other directories named for the Query Builder (that contain the log for that task), and also directories that contain .srx files (see references for an introductory link on .srx files). Review of the resultant .xml files show that they contain metadata for the sample project tasks.

We don't include any of the metadata text here, as it is extremely voluminous, detailed, a lot of it is not informative, and also it may be to subject to copyright protection.

> **Remark:** data are facts and cannot be copyrighted; this *may* apply to at least some metadata. However the arrangement of data can be copyrighted, and *presumably* the xml, srx files are copyrighted. Additionally in some countries, there are *sui*

generis rights that are similar to copyright, and which apply to data. It is safest to assume that the metadata files are copyright all rights reserved by SAS Institute.

Hack #1 lets you examine some of the metadata, but be aware of the following caveats.

- Project internals are not documented, are subject to change, and not supported for this usage.
- This hack was done in version 7.13 of SAS Enterprise Guide. This hack might not work in future releases of SAS Enterprise Guide.
- You might not see all of the metadata using this hack.
- When taken out of context, some of the metadata are not very useful or informative.

Now let's try this hack on a password-protected project file.

HACK #1 ON A PASSWORD- PROTECTED PROJECT

The relevance/application of hack #1 is: could this hack be used to expose the password of a password-protected project file? If you assign a password to a .egp file, would the password appear in the metadata that are visible using this hack? Possibly in the project.xml file?

To test this, we assign a password to the project and save it under a different name, and then follow steps #1-4 above. However, when we attempt to unzip and extract the files from the zipped directory, we are asked to supply the assigned password. After giving the password, we again get 7 subdirectories and a project.xml file. Checking the project.xml file confirms that the password does not appear in the unzipped file.

The above verifies that a project password cannot be revealed by hack #1, as the project password is required to extract the unzipped files that contain metadata. Next we check another hack that is unlikely to work for zipped files, though it sometimes works for other types of binary files so is worth trying.

HACK #2: EXAMINE THE .EGP FILE WITH A TEXT EDITOR

If you open a compiled, binary file in a text editor you may find that text constants in the associated source code program are visible in the binary file. Surprisingly, some of the source code in a compiled, encrypted SAS macro (a stored macro) may be visible by this method. This may include sensitive information if it is present in the code, e.g., passwords; see Billings (2017A, 2017B) for further discussion on this topic and 2 methods to potentially mitigate this issue for stored macros.

We open the password protected .egp files using a text editor, and search for the password. The search is unsuccessful, hence this hack does not work to reveal the password.

DISCUSSION OF RESULTS

The above shows that password protection of SAS Enterprise Guide is secure against the 2 hacks described here. Password protection of projects should be one layer of security in a multi-layer approach that includes all of the following:

- Operating system access controls
- SAS metadata roles and access controls
- Other relevant security features (non-metadata and/or SAS metadata-based)

Hemedinger (2014) suggests using a stored process on a server instead of a project, for a higher security approach. If it is feasible for the target application, replacing the project with a scheduled, production batch program (that is under formal change management control) is another high-security alternative.

CONCLUSION

The tests above provide independent confirmation that password protection of SAS Enterprise Guide projects is secure against 2 obvious hacks:

- Unzip the project file and examine the metadata components
- Opening the project file in a text editor.

Password protection of a project is only 1 layer in the multi-layer security infrastructure at your SAS site. If your project contains sensitive information, it should be password-protected.

SUPPLEMENT: COMMENTS ON SAS METADATA AND SIMULATION

Metadata. Legacy SAS users (those with experience prior to the introduction of the SAS Metadata Server) or users whose experience is limited to PC SAS and/or batch only (common on mainframes), occasionally have difficulty understanding metadata as they have worked mostly in non-metadata SAS environments. It may be informative for some legacy users to use hack #1 on a small project of their own, to see firsthand that their projects are zipped files of metadata that can be xml, srx, sas, log, and possibly other types of files.

Project metadata are used to manage and run a project, both the on-PC and on-server components. The SAS Metadata Server is a specialized server in the SAS Business Intelligence (BI) ecosystem and the entire BI ecosystem is managed using metadata. Project metadata captures the parameters/options selected for each GUI, .sas code files, .log files, and other data for the project. Kirk Lafler has a new book on metadata (Lafler 2017), and there is ample SAS documentation on metadata as well.

Simulation. This paper is not about simulation; a simulation was used only to create a more realistic sample project for testing, and to have a variety of task types in the project for review of the project metadata after unzip. However, some readers may be curious about the results of the simulation. The PROC REG output is voluminous but the most important numbers are below. The fitted equation is similar to the actual equation used to create the test data.

Linear Regression Results, Y on (X1, X2), artificial data

The REG Procedure
Model: Linear_Regression_Model
Dependent Variable: Y Y

Number of Observations Read	150
Number of Observations Used	150

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	1555.326	777.6629	741.09	<.0001
Error	147	154.2549	1.04935		

Corrected Total	149	1709.581			
Root MSE	1.02438	**R-Square**	0.9098		
Dependent Mean	2.49489	**Adj R-Sq**	0.9085		
Coeff Var	41.05913				

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	95% Confidence Limits	
Intercept	Intercept	1	0.04066	0.22971	0.18	0.8597	-0.41329	0.49462
X1		1	9.88301	0.28897	34.2	<.0001	9.31194	10.45409
X2		1	-4.9901	0.29004	-17.2	<.0001	-5.56329	-4.4169

APPENDIX 1:
BSD 2-CLAUSE COPYRIGHT LICENSE (OPEN SOURCE)

```
* All program code in this paper is released under a Berkeley Systems
  Distribution BSD-2-Clause license, an open-source license that permits
  free reuse and republication under conditions;

/*
Copyright (c) 2018, MUFG Union Bank, N.A.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are
permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of
conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list
of conditions and the following disclaimer in the documentation and/or other materials
provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY
EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT
SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED
TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY
WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH
DAMAGE.
*/
```

ACKNOWLEDGEMENTS

Thanks to Chris Hemedinger of SAS Institute, Inc., for his blog postings and helpful comments. Any errors herein are solely the responsibility of the author.

REFERENCES

Note: all URLs quoted or cited herein were accessed in August 2017.

Billings T (2017A). Secure Macro-Based Method to Assign LIBNAMEs for Databases. In press; paper accepted for presentation at 2017 *Western Users of SAS Software* conference.

Billings T (2017B). Keeping Passwords, AES Encryption Keys, and Other Sensitive Parameters Out of Source Code and Logs. In press; paper accepted for presentation at 2017 *Western Users of SAS Software* conference.

Hemedinger, C (2014). Lock down EG project? Communities post; multiple authors. URL: https://communities.sas.com/t5/SAS-Enterprise-Guide/Lock-down-EG-project/td-p/162977

Lafler, K (2017). In press.

WikiPedia (2017). Segmentation Rules eXchange. URL: https://en.wikipedia.org/wiki/Segmentation_Rules_eXchange

CONTACT INFORMATION

A list of the author's SAS-related papers, including URLs for free access, is available at the URL (hosted by Google Drive): https://goo.gl/uCUHoa

Your enterprise web filter might prevent access to this URL from work, in which case you will need to access via a personal device.

Thomas E. Billings
MUFG Union Bank, N.A.

Remote from:
Merritt Island, Florida 32952

Phone: 321-453-5694
Email: tebillings@gmail.com