Paper 1118-2017

Removal of PII

Stanley E. Legum, Ph.D., Westat

ABSTRACT

At the end of a project, many institutional review boards (IRBs) require project directors to certify that no personally identifiable information (PII) is retained by a project. This paper briefly reviews what information is considered PII and explores how to identify variables containing PII in a given project. It then shows a comprehensive way to ensure that all SAS® variables containing PII have their values set to NULL and how to use SAS to document that this has been done.

INTRODUCTION

Federal statute requires that the confidentiality of personally identifiable information (PII) in all Federal information and information systems be protected. This includes systems created by contractors as part of projects funded by the Federal government. The confidentiality and privacy of PII in these systems must be maintained throughout the project and thereafter. Corporate or client IRBs adopt rules implementing the Federal requirements. If de-identified versions of project files will be archived after a project ends, they may require a formal report when all the PII has been removed from the data.

IDENTIFYING PII

The Federal Office of Management and Budget (OMB) defines "personally identifiable information" as "information which can be used to distinguish or trace an individual's identity, such as their name, social security number, biometric records, etc. alone, or when combined with other personal or identifying information which is linked or linkable to a specific individual, such as date and place of birth, mother's maiden name, etc."[1] OMB Memorandum M-07-16 does not provide details about what might be included in "etc." Its focus is on Information Technology and security requirements and notification requirements in the case of breaches. The NIH Privacy Impact Assessment form provides more examples:

- Name
- Date of Birth
- Social Security Number (SSN)
- Photographic Identifiers
- Driver's License
- Biometric Identifiers
- Mother's Maiden Name
- Vehicle Identifiers
- Personal Mailing Address
- Personal Phone Numbers
- Medical Records Numbers
- Medical Notes
- Financial Account Information

[1] OMB Memorandum M-07-16 Safeguarding Against and Responding to the Breach of Personally Identifiable Information, footnote 1.

- Certificates
- Legal Documents
- Device Identifiers
- Web Uniform Resource Locator(s) (URL)
- Personal Email Address
- Education Records
- Military Status
- Employment Status
- Foreign Activities

The Department of Health and Human Services (HHS) in a discussion of deidentifying information for the purpose of file sharing adds address and geocoding information or other geographic subdivisions smaller than a state[2]:

- Street address
- City
- County
- ZIP Code
- Geocodes (latitude and longitude)

Note that there is an exception that allows the first 3 digits of a ZIP code if the area has more than 20,000 people.

HHS also lists dates (except year) when they are related to an individual, specifically:

- Birth date
- Admission date
- Discharge date
- Death date

In addition to birth date, ages of people over 89 are also considered PII unless they are aggregated as a category “90 or older.”

Note that these lists are not all inclusive of “information which can be used to distinguish or trace an individual's identity.” Data custodians need to use judgment in the context of their projects. For instance, an agricultural study that collected the fact that a farmer raises a rare breed of cattle such as Randall Linebacks and has some number of them (e.g., 50-99), might need to treat either the breed or the quantity as PII.

IDENTIFYING PII WITHIN A PROJECT

By the end of a large data collection project such as a survey, an epidemiological study, or a clinical trial PII may be stored in the data collection instruments, intermediate files, and final analytic files. When searching for project PII, it is useful to examine the data collection instruments to determine which ones collect PII, delivery or analytic files to determine which ones contain PII, and project processes that move PII from initial collection to final files. Project documentation that can contribute to this include:

- Data flow diagrams

[2] http://www.hhs.gov/hipaa/for-professionals/privacy/special-topics/de-identification/index.html#standard on 12/29/2015.

    - What is initial file(s) where PII was collected?
    - What intermediate files contain PII?
    - What final/delivery files contain PII?
- Codebooks / Data Dictionaries
- IRB documents
- Privacy Impact Assessment (PIA)

While reviewing these information sources, useful questions to ask are:

- Can any project directories be deleted? -- These might include directories used to import data from questionnaires or directories of intermediate files used during a geocoding step.
- Are there sets of files that can be deleted? -- If project naming conventions ensure that all files with source data include “SRC” at the start of their file names and all analysis files contain “FNL” at the start of their file names, it might be possible to decide to delete all files whose names do not start with “FNL” or “SRC.” Such identification is only practical if file naming standards are set early in a project and strictly enforced.

Any global decisions such as these, should be submitted to the Project Director and implemented only after formal approval has been obtained.

The search for PII should be comprehensive. It is unlikely that substantial amounts of PII will be found in text or word processing documents or Portable Data Format (PDF) documents, but they should not be overlooked. More common file formats that might contain PII are:

- Excel files
- Database files
    - SQL Server
    - Oracle
    - MS Access
        - ACCDB files
        - MDB files
    - dBase format (DBF) files
- SAS® files

Checking a large number of files for PII can be daunting. A useful first check is to visually inspect a sample of files. It may be possible to quickly rule out some categories of files. For example, lists of equipment used may only be indirectly linked to PII through a study identifier. As long as the PII that is directly linked to that identifier has been removed (or altered to make it non-usable for identification purposes), nothing needs to be done to the equipment file. Another useful early step in the process is to talk to the person who created or maintained some subset of the files. It is possible either that the files contain no PII or that there is no need to retain them in the archive.

It may be possible to automate PII checks using SAS. Before doing so, however, it is necessary to create a list of all directories and subdirectories that store project files.

CREATING LIST OF DIRECTORIES AND SUBDIRECTORIES

The discussion that follows makes the assumption that the project being checked has its own dedicated network directory and associated subdirectories. If the project files reside in multiple directories, it will be necessary to repeat these steps for each one. The technique using the DIR command that follows is

specific to Windows-based servers. Lists of directories and subdirectories on UNIX or Linux servers can be created using the LS command.

One quirk of the Windows DIR command is that it can only refer to a drive letter. It cannot refer to drive names using the Universal Naming Convention (UNC) form such as \\servername\sharename\path\filename or \\westat.com\DFS\.... If your high-level directory has not been mapped to a drive letter, you can do so by following the steps below:

- Click the Start button, and then click Computer
- Click the Tools menu, and then click Map Network Drive (may need to press ALT key to see Tools menu)
- In the Drive list, click a drive letter
- In the Folder box, type the path of the folder or computer (or click Browse to find the folder or computer)
- Click Finish

Once you have a drive letter assigned, you can get a list of subdirectories by running the DIR command in the COMMAND window:

- Open the COMMAND window
  - Click the Start Button
  - Type "Command" (opens Command Prompt window)
- Change directory to the mapped drive letter, for example
  CD X:
- Run the DIR command and store the results in a text file, for example:
  - DIR /AD /B /S > testlist.txt (creates text file in the X: directory)
  - DIR /AD /B /S > Y:\testlist.txt (creates text file in the Y: directory)

The DIR command switches[3] used in these examples are:

/AD = Displays directories only

/B = Uses bare format -- no heading information or summary

/S = Displays files in specified directory and all subdirectories

For the example project used in the code examples, the command

DIR /AD /B /S > FullListOfSubdirectories.txt

generated a list of 1065 directories. Figure 1 shows an extract from this list.

```
X:\PROJ2008_Archive\
X:\PROJ2008_Archive\8502.01 Management
X:\PROJ2008_Archive\8502.01 Management\Add-on coordinators
X:\PROJ2008_Archive\8502.01 Management\Experience Sheet
***
```

[3] The command DIR /? Displays the full list of available switches for DIR.

X:\PROJ2008_Archive\8502.01 Management\UsersGuides_from_2001_and_1995

\NY User's Guide\Appendices\Appendix E

X:\PROJ2008_Archive\8502.01 Management\UsersGuides_from_2001_and_1995

\NY User's Guide\Appendices\Appendix E\DBFData

***

Figure 1. Extract from Full List of Subdirectories

directories in the extract, the names of some of the subdirectories contain apostrophes. Since these directory names are later used in macro calls, it is necessary to make provisions to handle them. In the example project, this was handled during the creation of a file of LIBNAME statements by placing the character "Y" in front of the LIBNAME statements containing apostrophes. Fortunately, there were relatively few such directories and it was possible to visually determine that they contained descriptive text files without data; and hence no PII.

The file with the full directory list was used to create a new file with sequential LIBNAME statements and to flag names containing apostrophes. The code to do this is shown in

Figure 2.

```
filename dirlist
"\\Rkw6\Vol602$\PROJ2008_Archive\PII_Removal\FullListOfSubdirectories
.txt" LRECL=200 ;

data subdir ;
       infile dirlist truncover ;
       input dirname $char200. ;
       if find(dirname,"Directory of X:") ;
       dirname2 = tranwrd(dirname,"Directory of
X:","\\Rkw6\Vol602$\PROJ2008_Archive");
    run ;

data subdir ;
   set subdir ;
   seqnumN = _N_ ;
   seqnumC = put(seqnumN,z4.) ;
run ;

data LibList (keep = libref dirname2 flagged )  ;
   set subdir ;
   length libref $7 ;
   file   "\\Rkw6\Vol602$\PROJ2008_Archive\PII_Removal\
    FullListOfLibraries.txt" ;
   libref =  cats("LIB",seqnumC) ;
   if find(dirname2, "'") then flagged = "Y" ;
   put flagged libref dirname2 ;
run ;
```

Figure 2. Code Used to Create List of Libnames

In order to preserve a copy of the list of directories with apostrophes, the code in Figure 3 was run and the file FLAGGED was printed.

```
proc sql ;
   create table flagged as
   select *
   from LibList
   where  flagged = "y"
   ;
quit ;
```

Figure 3. Code to Created File FLAGGED Containing List of Directories with Apostrophes

The code in Figure 4 reads the LibList file created in Figure 2 and uses it to create and execute statements like:

```
libname LIB0047 ' \\Rkw6\Vol602$\PROJ2008_Archive\8502.02 Study
Design\Programs';
```

```
data _null_ ;
  set subdir ;
  if not find(dirname2, "'") then
  call execute("libname " || "lib" || trim(seqnumc) || " '" ||
     trim(%str(dirname2)) || "';");
run ;
```

Figure 4. Code to Read LibList File

SAS CODE TO SET FIELDS WITH PII TO NULL

The point of executing all these LIBNAME statements is that sashelp.vColumn now contains a full list of all project variables. These can be distinguished from everything else in sashelp.vColumn by the fact that all and only the project variables are in libraries whose names start with “LIB”.

To build a list of variables potentially containing PII, it is necessary to use project-specific knowledge. The example project had consistent variable labeling and it was known that after files with names and telephone numbers had been removed, the only remaining PII was in address and geocoding variables. These could be recognized by the fact that they contained the following strings in their variable labels: LATITUDE, LONGITUDE, STREET NAME or STREET NUMBER. The code in Figure 5, produces the file VarList, which contains all and only the project variables that potentially contain PII. Note that VarList also contains the LIBNAME and file name (MEMNAME) as well as the variable name (NAME) of these variables. It also contains the label, and type of the variables, which were used in the initial checks of the program.

```
proc sql ;
  create table VarList as  select libname, memname, name, label, type
  from sashelp.vColumn
  where substr(libname,1,3) = "LIB" and
```

```
        (find(upcase(label),'LATITUDE') or
         find(upcase(label),'LONGITUDE') or
         find(upcase(label),'STREET NAME') or
         find(upcase(label),'STREET NUMBER')
        )
  ;
quit ;
```

Figure 5. Code to build the list of variables containing PII

VarList contains only 59 variables. Figure 6displays the first few rows of the file.

libname	memname	name	label	type
LIB1055	ALL_LOCATIONS	PLSTNAME	Trip address - street name	char
LIB1055	ALL_LOCATIONS	PLSTNUM	Trip address - street number	char
LIB1055	BOTH	PLSTNAME	Trip address - street name	char
LIB1055	BOTH	PLSTNUM	Trip address - street number	char
LIB1055	VT_LOCATIONS	HOMELAT	HH latitude	num
LIB1055	VT_LOCATIONS	HOMELONG	HH longitude	num
LIB1055	VT_LOCATIONS	PLSTNAME	Trip address - street name	char
LIB1055	VT_LOCATIONS	PLSTNUM	Trip address - street number	char
LIB1055	VT_LOCATIONS	TRPENDLA	Trip end latitude	num
LIB1055	VT_LOCATIONS	TRPENDLO	Trip end longitude	num
LIB1055	VT_LOCATIONS	WORKLAT	Work latitude	num
LIB1055	VT_LOCATIONS	WORKLONG	Work longitude	num
LIB1055	WOOF1_NY	PLSTNUM	Travel day trip end - street number	char
LIB1055	WOOF1_NY	PLSTNAME	Travel day trip end - street name	char

Figure 6. Beginning of VarList file

To remove potential PII from these variables, all of their values were set to missing. The code to perform this task, which is relatively simple, is shown in Figure 7.

```
data _null_ ;
  set VarList ;
  call execute (
        ' proc sql ; '
        || '  update '
        ||  memname
        || '  set '
        ||    name
        ||'   = null ; '
```

```
        ||' quit ; '
        ) ;
run ;
```

Figure 7. Code to set values of variables with PII to null

For each variable in VarList, this code generates one SAS SQL update statement such as

```
proc sql ;
  update VT_LOCATIONS
  set HOMELAT = null ;
quit ;
```

Since setting a value to NULL does not need to distinguish between character and numeric variables, this code can be applied to all 59 variables without explicitly referencing their type.

Executing this code completed the PII removal task as narrowly defined. Since there are regulatory and legal reasons for performing the task, the production run also performed a number of other tasks:

1. Listed all directories searched.
2. Listed variables with values set to null.
3. Provided PROC FREQ output for each variable with values set to NULL – This documented the fact that PII values were in fact removed.
4. Created an Excel file with the lists of libraries and variables.

FOLLOW-UP STEPS

A number of additional administrative steps were completed after the code was run.

1. Cleaned up archival backup tapes
   - Created new archival backup tapes (without PII)
   - Identified backup tapes that may have PII
   - Ensured that old backup tapes were destroyed
   - When received, provided copy of certificate of destruction to Project Director
2. Wrote a memo to the project director summarizing the actions performed.

REFERENCES

Code of Federal Regulations, Title 45, Public Welfare, Department of Health and Human Services,

Part 46, Protection of Human Subjects, Revised January 15, 2009.

Privacy Act of 1974 (Privacy Act), 5 U.S.C. § 552a.

OMB Memorandum M-07-16 Safeguarding Against and Responding to the Breach of Personally Identifiable Information, https://www.whitehouse.gov/sites/default/files/omb/memoranda/fy2007/m07-16.pdf

CONTACT INFORMATION

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

Stanley Legum
Westat

1600 Research Blvd.
Rockville, MD 20850
selegum@Westat.com

DISCLAIMER

The contents of this paper are the work of the author and do not necessarily represent the opinions, recommendations, or practices of Westat.