SANDIA REPORT

SAND2016-9461
Unlimited Release
September 2016

The Regional Test Center Data Transfer System

Daniel M. Riley
Joshua S. Stein

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-mission laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: http://www.ntis.gov/search

SAND2016-9461
Unlimited Release
September 2016

The Regional Test Center Data Transfer System

Daniel M. Riley and Joshua S. Stein
Photovoltaic and Distributed Systems Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-MS0951

Abstract

The Regional Test Centers are a group of several sites around the US for testing photovoltaic systems and components related to photovoltaic systems. The RTCs are managed by Sandia National Laboratories. The data collected by the RTCs must be transmitted to Sandia for storage, analysis, and reporting. This document describes the methods that transfer the data between remote sites and Sandia as well as data movement within Sandia's network. The methods described are in force as of September, 2016.

ACKNOWLEDGMENTS

CONTENTS

FIGURES

TABLES

NOMENCLATURE

DOE	Department of Energy
FTP	File transfer protocol
FTPS	File transfer protocol over SSL
PV	Photovoltaic
RMFT	Repliweb Managed File Transfer
RNRG	Renewable NRG
RTC	Regional test center
SCP	Secure copy protocol
SNL	Sandia National Laboratories

1. INTRODUCTION

Sandia National Laboratories (SNL) manages the Regional Test Center (RTC) program which performs photovoltaic (PV) system performance testing under diverse climactic conditions. This effort necessarily requires the establishment of testing centers with diverse climates. Sandia has established and managed centers at the Sandia campus in Albuquerque, NM; Cocoa, FL; Williston, VT; and Henderson, NV.

RTC Partners work with Sandia to design a PV system or systems for testing at the RTCs. Sandia monitors the performance of these systems with on-site measurement hardware. The PV performance data generated at each site is transmitted to the Sandia campus in Albuquerque where it is stored, analyzed, and sent to the partner upon request.

This document details the transmission methods used to aggregate and disseminate the data from Sandia. Section 1 explains the basics about the RTCs and the data collection systems in place at each RTC. Section 2 explains the typical methods by which data are sent from each location to Sandia, aggregated at Sandia and stored on Sandia's network. Section 3 explains the typical methods by which the data is shared with the RTC partners and, in some cases, the National Renewable Energy Lab (NREL). Section 4 describes the full process for systems that do not follow the typical data transfer methods described in prior sections. The methods described herein are applicable to the RTCs as of September 2016.

1.1. Data Collection Background

For most PV installations at the RTCs, system performance measurements are recorded via Campbell Scientific data loggers; primarily the CR1000 and CR6 models. These loggers serve as measurement devices for analog and digital signals, collection points with nonvolatile memory, a real-time clock, and communication equipment such as Ethernet adapters. The data loggers are managed by the Campbell Scientific Loggernet software package which runs on a "host PC" at each location. The communication between the PC and the loggers is managed by the technical personnel at each site and may be different between sites.

The Loggernet software on each host PC is configured to collect data from the attached data loggers once per day, typically around 23:00 local standard time. See Figure 1 and Figure 2 for example settings within Loggernet. Upon data collection, Loggernet creates a comma separated ASCII file in the TOA5 format [1]. The files are generated with only the data since the last collection. The file names are created from a four letter prefix for the location per Table 1, then the partner name, and finally a suffix with the time and date of the first entry in the file in the YYYY_MM_DD_HHMM format. For example, the file created in Figure 2 will be called SNLA_Baseline6kW_2016_09_13_1038 if the first entry in the file were collected on September 13, 2016 10:38 MST.

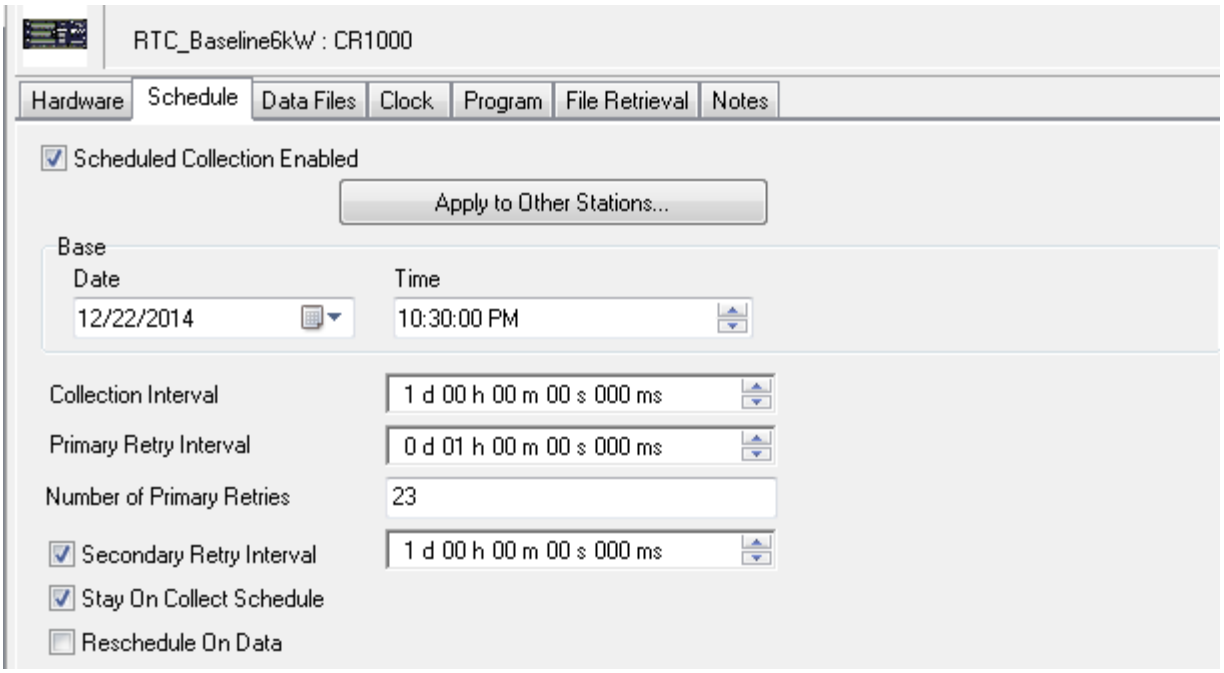


Figure 1. Example Loggernet data collection schedule

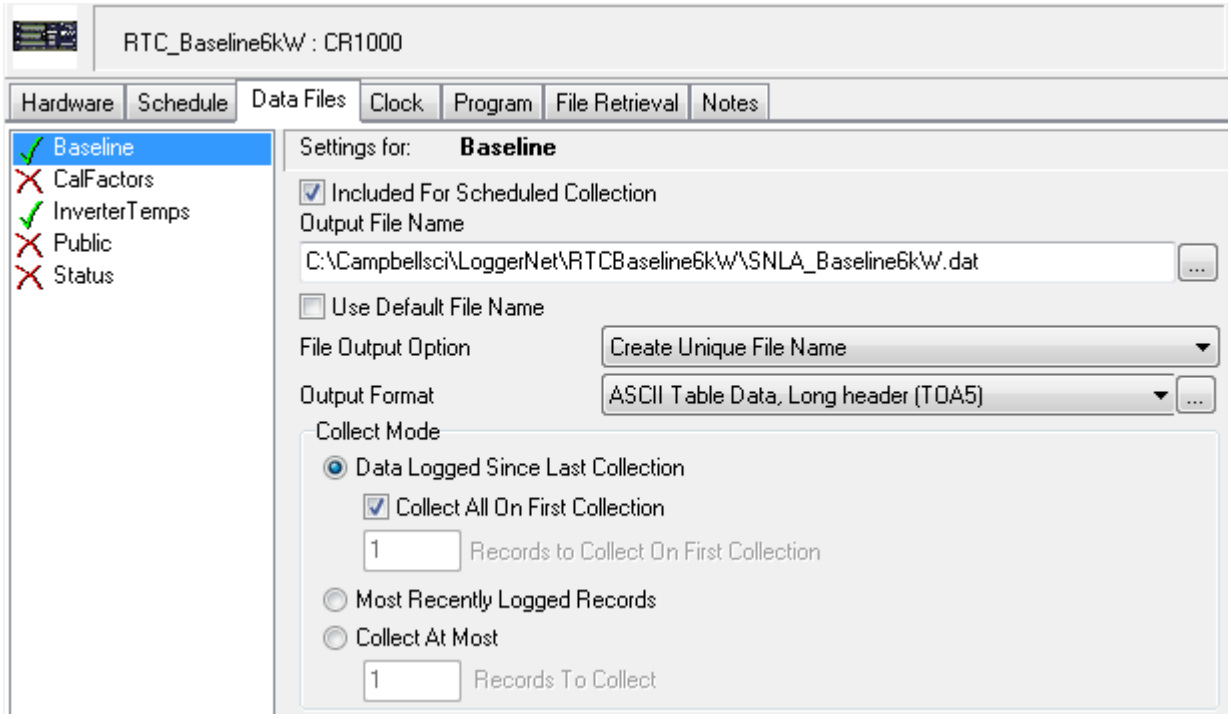


Figure 2. Example Loggernet data files

Table 1. Data file prefix designator

Prefix	Location
SNLA	Sandia National Laboratories, Albuquerque
LVRM	Henderson, NV (Las Vegas, River Mountain)
FSEC	Cocoa, FL (Florida Solar Energy Center)
IBMW	Williston, VT (IBM Williston)

2. TRANSMITTING FILES TO SANDIA

Once the files have been collected on the host PC at the remote RTC site, they must be transferred to Sandia for storage and analysis. Files generated at the Sandia location may be transferred using only the Sandia internal network. Once the data file is on Sandia's network it is kept both as the comma-separated "flat file" and it is read into the PVGRID database managed by Sandia's database team.

Figure 3 shows the general flow of information from remote RTC locations and the Sandia RTC location through Sandia's network. The steps in the process will be explained in detail in this section.

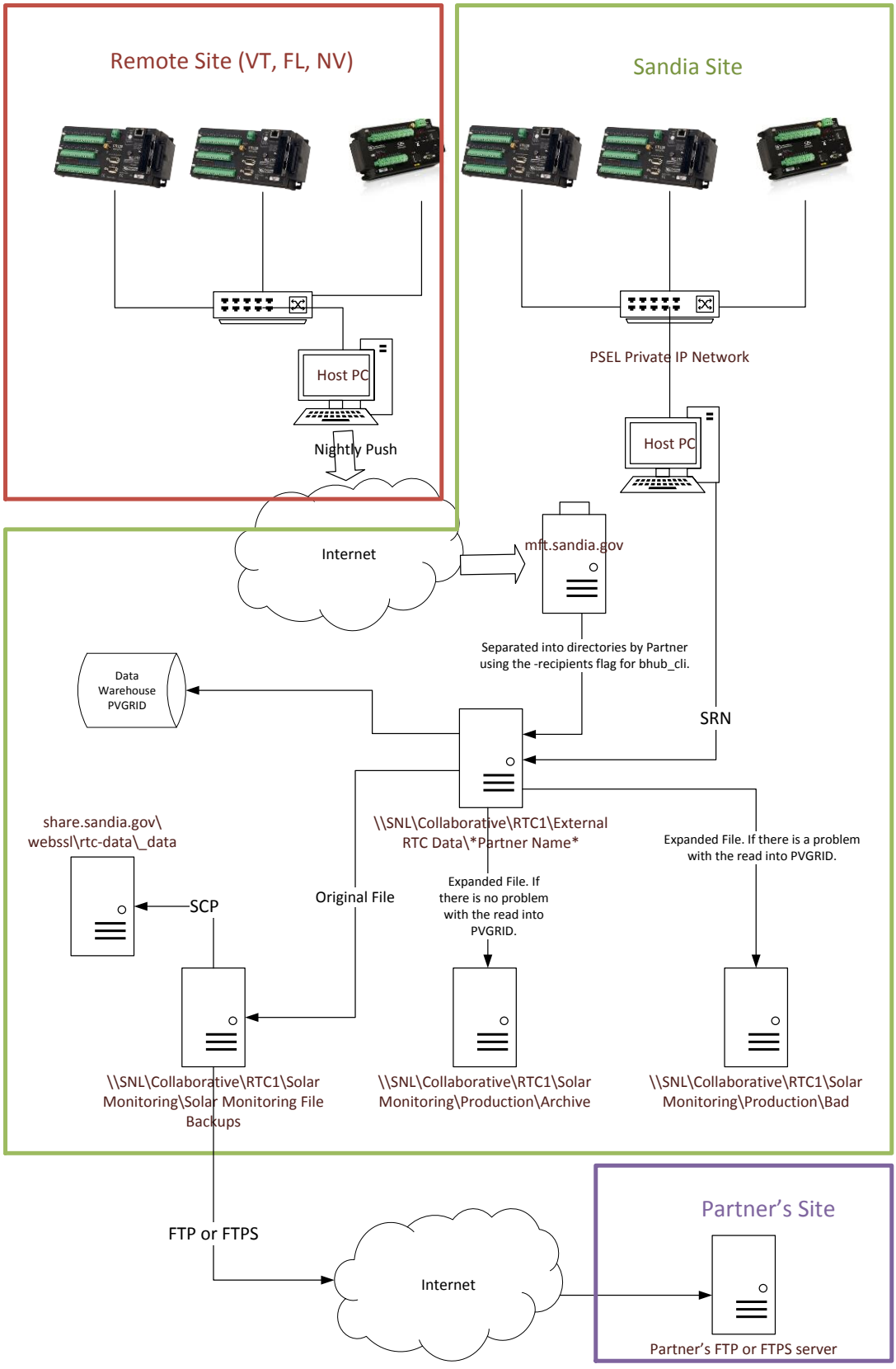


Figure 3. Information Flow of RTC Data

2.1. Host PC to SNL Collaborative Drive

The data files must first reach the RTC collaborative drive at "\\SNL\Collaborative\RTC1\External RTC Data*Partner Name*" on Sandia's network; where *Partner Name* is a directory with a name that indicates the partner's system. Since the host PC at the SNL location is already on the SNL network, the process to move the data from the host PC is straightforward. Getting the data to SNL from an outside network is more complicated.

2.1.1. At Sandia RTC Site

At Sandia, the host PC collects data from the loggers using Campbell Scientific's Loggernet software product before midnight for each RTC system. The files are created on the host PC with the file archive flag set (to true). This archive flag serves as an indicator that the files have not yet been transferred to the collaborative drive.

To transfer the files automatically to the collaborative drive, a task scheduling program such as the built-in Windows Task Scheduler or a more robust task scheduling program such as Splinterware's System Scheduler [2] can be used to move the files at a particular time or times to the collaborative drive. A sample Windows PowerShell script is provided in Appendix A. It is important that the archive flag be reset (to false) once the file is copied from the host PC to the collaborative drive to prevent the file(s) from being copied on the next scheduled copy.

Any methods for moving the files across the SNL network could be employed as long as the files are moved only once from the host PC to the collaborative drive. As of September 2016, the most common method used is a Windows PowerShell script that runs the Robocopy program as shown in Appendix A.

2.1.2. At Remote RTC Sites

At remote RTC sites, the data files must travel over the internet to reach Sandia. However, the incoming data restrictions at Sandia are stringent. The Repliweb Managed File Transfer (RMFT) service provided by Sandia is designed to handle the transfer of large files. With the help of the RMFT administrator, we are able to accept the data files from the remote host PCs and have them automatically moved onto the RTC collaborative drive.

The process begins on the remote host PC within Loggernet. The Loggernet program contains an application called Task Master which is designed to perform some action upon interaction with a data logger. In this case, Task Master should execute a batch script after successfully collecting data from a data logger. The batch script executes a command line executable to transfer data through the RMFT. An example Task Master setup screen is provided in Figure 4.

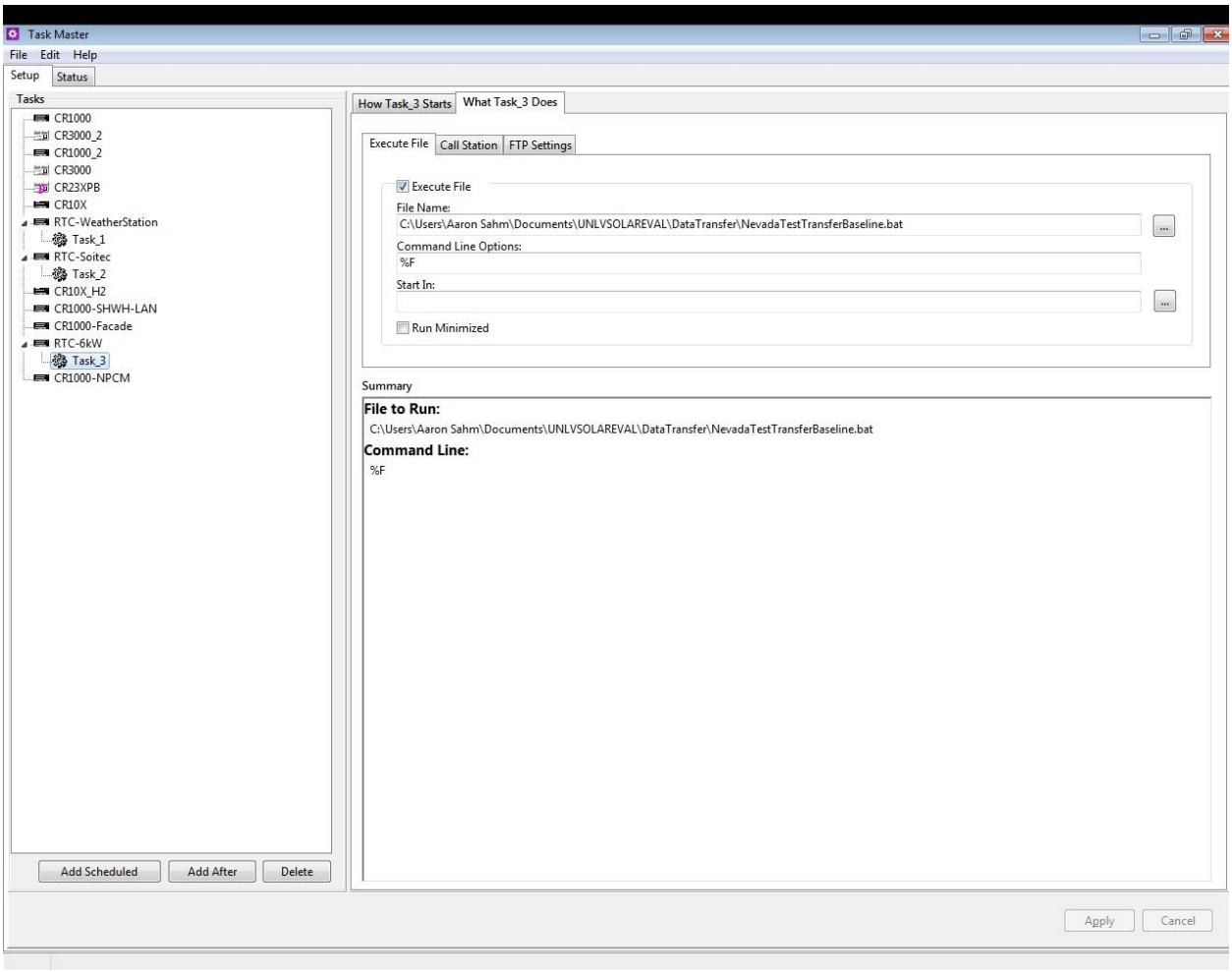


Figure 4. Example Task Master Screen

As shown in Figure 4, the "Command Line" options include "%F". This command passes the file name of the recently collected data into the batch script [3].

The batch script simply serves as a means to execute the RMFT client command line interface program. The RMFT client command line interface is an executable file which may be called with some additional fields to send data to Sandia through the RMFT [4].

The following script is designed to run using the Windows command line interface as a ".bat" file. The RMFT client executable is bhub_cli.exe and uses the "send" function. The user name and password are provided by the RMFT administrator and are unique to each RTC site for use with all of the partner data sets. The user names for each RTC site are listed in Table 2; the passwords for each user name have been omitted for security. The recipients field changes with each partner and is provided by the RMFT administrator, the list of currently active recipients is listed in Table 3 along with the subdirectory name on the RTC collaborative drive. The "%1" indicates that the script should use the filename which has been passed into the script by Loggernet's Task Master application. The final portion of the script creates a log file that documents the execution of the RMFT client.

```
C:\DataXfer\bhub_cli.exe send
-server=mft.sandia.gov -user=username -password=luggage
-recipients=snl:SunivaBaseline -files=%1
-secure >> C:\DataXfer\logfile.txt
```

Table 2. User Field in RMFT Client Command Line Interface

RTC Site	-user field
SNL, Albuquerque	N/A
Henderson, NV	lvrmrtcdata@unlv.edu
Cocoa, FL	fsecrtcdata@fsec.ucf.edu
Williston, VT	ibmwrtcdata@us.ibm.com

Table 3. Recipients Field in RMFT Client Command Line Interface and Sub-directory name on Collaborative Drive

Partner	-recipients field	Collaborative Sub-directory Name
Maxim	snl:Maxim	Maxim
Baseline	snl:BaselineSuniva	BaselineSuniva
Stion	snl:Stion	Stion
IBMW_SitePower	snl:IBMW_SitePower	IBMW_SitePower
Soitec	snl:Soitec	Soitec
Met Station	snl:MET	MET
Heliovolt	snl:HelioVolt	HelioVolt
Prism	snl:Prism	Prism
SolarCity	snl:SolarCity	SolarCity
SolarWorld	snl:SolarWorld	SolarWorld
Norwich & Chilicon & SolarWorld	snl:NorwichChiliconSolarWorld	NorwichChiliconSolarWorld
PVMC	snl:PVMC	PVMC
SunPower Bifacial	snl:SunPowerBi	SunPower_Bifacial
SunPower P-series	snl:SunPowerPSeries	SunPower_PSeries

The RMFT administrator enacts filtering instructions based on the recipients field such that the recipient field enters a particular sub-directory on the collaborative drive.

2.2. SNL Collaborative Drive to PVGRID SQL Database

Sandia's Enterprise Database Administrative (DBA) team has created the PVGRID database for housing RTC data and other data generated by the Sandia PV team. PVGRID is an SQL database which may also be colloquially known as the "data warehouse". The table of PVGRID that is relevant for RTC is the "Solar Monitoring" table which houses temporally stamped PV system

performance information. The database administrators provided by the DBA team develop of PVGRID database and the import scripts for all RTC data.

At approximately 3:00 each morning, the database administrator team's entity account picks up all data files within the PV team's collaborative drive at \\SNL\Collaborative\RTC1\ExternalRTCData and all subdirectories therein. The DBA team's scripts first create a copy of the original file to "\\SNL\Collaborative\RTC1\Solar Monitoring\Solar Monitoring File Backups". The scripts then expand the comma-separated value file by adding a large number of commas at the end of each line such that every file, regardless of the number of fields collected by the logger, has 200 commas per line. The database import software then attempts to read the expanded file into PVGRID.

If the file read into PVGRID fails the expanded file is placed into "\\SNL\Collaborative\RTC1\Solar Monitoring\Production\Bad" and the DBA team is notified of the failure. If the file read into PVGRID is successful, then the expanded file is placed into "\\SNL\Collaborative\RTC1\Solar Monitoring\Production\Archive".

2.3. SNL Collaborative Drive to Partner's Servers

In some cases, an RTC partner has asked for the data files to be provided to them each day as the files are collected rather than as a large collection on a less frequent basis. In order to accommodate this, it is simplest to have the partner establish an FTP server and provide the RTC team with a login account and password. In the event that a partner wishes more security than can be provided with a standard FTP server, it is possible to use an FTPS transfer method for additional security.

Transferring data via FTP uses Windows PowerShell to set up the files and uses the native Windows FTP client to transfer the files. A sample PowerShell script has been provided in Appendix B.

Transferring data via FTPS uses Windows PowerShell to set up the files and uses the freely available psftp client to transfer the files. A sample PowerShell script has been provided in Appendix C.

The data transfer scripts may be run automatically by a task scheduler such as the Windows Task Scheduler or System Scheduler by Splinterware. The sample scripts attempt to move the most recent 7 files to the partner FTP or FTPS server, regardless of whether the files have already been moved. This does not rely on the archival flag of each file and provides up to 7 days of "backup" whereby any missed transfers may be re-attempted during future transfers.

2.4. SNL Collaborative Drive to share.sandia.gov For Externally Accessible Data

Some of the RTC data is owned by Sandia National Laboratories and is available for the public. A publicly accessible website and underlying database has been created by a web site

administrator in the Program Communications department to provide data from Sandia-owned RTC systems to the public. Currently, the baseline PV performance and meteorological station data for all RTC locations are shared. The RTC team places the data daily onto the drive at "share.sandia.gov\webssl\rtc-data_data". The web site administrator has written scripts to import data to the database nightly from the same folder.

In order to place the data on share.sandia.gov, a secure copy (SCP) must be used. In order to perform the secure copy on a Windows operating system, the program WinSCP [5] may be used in conjunction with Windows PowerShell. A sample script has been provided in Appendix D.

3. EXCEPTIONS

The methods described in section 2 are the normal means by which files move to and within Sandia. These methods are employed when the Sandia RTC team has defined the data acquisition hardware and software based on their typical data acquisition scheme. However, there may be RTC partners or systems which do not employ the typical data acquisition scheme, and in those cases it is necessary to provide alternative means to obtain the data. Two such examples are the 60 kW_P Suniva PV system and the Renewable NRG weather station installed at the Vermont RTC.

3.1. 60 kW Suniva PV System

Sandia installed a 60 kW Suniva PV system in Vermont to showcase a larger installation with data acquisition hardware more typical to systems of its size. The SMA inverters measure the electrical parameters of the system and provide the data to an attached "Cluster Controller". The cluster controller assembles the data from each inverter and hosts an FTP server from which the data files are retrieved.

The script provided in Appendix E retrieves the files using PowerShell, Robocopy, and the native Windows FTP client and places them at "\\SNL\Collaborative\RTC1\Suniva Baseline\Vermont 60kW Cluster Controller Data". The files are not read into the PVGRID database.

3.2. Renewable NRG Weather Station

The Renewable NRG (RNRG) company installed a weather station at the Vermont RTC and requested comparison with the RTC meteorological station. The RNRG station uses proprietary data acquisition and communication equipment which is not controlled by Sandia. The data is sent to Sandia in a series of 10 emails with file attachments throughout the day. The Sandia entity account "pvtestr" receives these emails, and a rule has been created to automatically route these emails into a subdirectory of the pvtestr inbox for later collection.

The files may be retrieved from the emails manually, but this is a lengthy and tedious process. Instead, a Visual Basic macro for Microsoft Outlook can automatically retrieve all attachments. Appendix F contains the Visual Basic code as well as instructions to generate an Outlook rule to strip file attachments from a series of files.

Once the RNRG files have been removed from their emails, the binary data files must be imported into the RNRG SymphoniePRO Desktop Application software [6]. The SymphoniePRO software reads the binary files and creates a database which can be queried (through the software) to retrieve measurements over a particular date range.

Administrative practice has thus far entailed retrieving the files from the pvtestr email every 2 weeks and creating an ASCII text file of the RNRG data for each calendar month.

4. CONCLUSIONS

Transmitting RTC data from all of the various RTC sites back to SNL and then moving that data within SNL is a complicated process with many different actors, methods, and storage locations. This report details the transmission paths and methods such that the system can be fixed when problems arise or recreated if necessary.

Systems installed at SNL generate data directly onto the Sandia network and are able to move through the network with simple PowerShell scripts to move and copy files. Files generated at remote RTC sites require the use of Sandia's RMFT and a process administered by Sandia's RMFT administrator. The file moves and RMFT process should place the ASCII files generated by the Campbell Scientific data loggers at "\\SNL\Collaborative\RTC1\External RTC Data\<Partner Name>" on the RTC collaborative drive.

Once the files have arrived on the RTC collaborative drive, scripts generated by the Database administrator copy the original files to another directory on the collaborative drive, expand the ASCII files, read the expanded files, populate the SQL database PVGRID, and move the expanded files to another directory for archiving.

The original file copies are then either sent to RTC partners by FTP or FTPS or, if the data are publicly available, are moved to the share.sandia.gov server. Any files on the share.sandia.gov server are read into another database and the information becomes available by external website managed by the website administrator.

The portions of the process that are controlled by RTC team members have been explained and critical scripts are provided in the appendices. The portions of the process that are not controlled by RTC team members require the help of specialists in other portions of Sandia. Their roles and responsibilities are listed in Table 4.

Table 4. Roles and Responsibilities of Non-RTC Team SNL Personnel

Role	Responsibilities
RMFT Administrator	Create procedures to automatically move files sent via RMFT client to collaborative drive. Create users and passwords for remote RMFT clients.
Database Administrator	Create and maintain the PVGRID SQL Database. Import ASCII data files from the collaborative drive nightly, creating backup copies of the original ASCII files.
Website Administrator	Create and maintain the externally-accessible database and website. Import ASCII files from share.sandia.gov to the database

5. REFERENCES

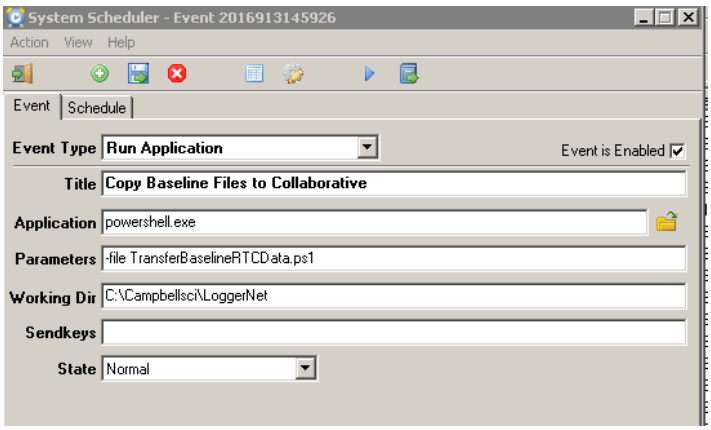
1. Campbell Scientific, Inc. *CR1000 Measurement and Control System Instruction Manual*, https://s.campbellsci.com/documents/us/manuals/cr1000.pdf, February, 2016
2. Splinterware Task Schedulersite, http://www.splinterware.com/products/scheduler.html
3. Campbell Scientific, Inc. *LoggerNet Instruction Manual*, https://s.campbellsci.com/documents/us/manuals/loggernet.pdf, April 2015
4. Repliweb, Inc., *RMFT Command Line Reference Guide*, Coconut Creek, FL, July 2011
5. WinSCP, https://winscp.net/eng/index.php.
6. Renewable NRG Systems, *SymphoniePRO Desktop Application*, https://www.renewablenrgsystems.com/services-support/resources/documentation-and-downloads/software-downloads/detail/symphoniepro-desktop-application.

APPENDIX A: SAMPLE SCRIPT FOR MOVING DATA WITHIN SNL

The following script is designed to run using Windows PowerShell and uses the built-in Robocopy program provided with Windows.

```
# Set the source directory and move to it
$SourceDirectory = "C:\Campbellsci\LoggerNet\RTCBaseline6kW"
cd $SourceDirectory

# Set the name and path to the transmission log file, start writing
$LogFileName = "$SourceDirectory\BaselineTransferLog.txt"

# Get the Current Time and Date as a string, write a human readable string to transfer log file
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $LogFileName -append
$DateStringForMachines = ((get-date).tostring('yyyyMMdd\THHmmss'))

# Set the destination (final) directory
$DestinationDirectory = "\\SNL\Collaborative\RTC1\External RTC Data\BaselineSuniva"

<# Copy the files in the source directory which have the
   archive flag set to the destination directory, remove the
   archive flag from the files in the source directory #>
$FilesToCopy = "SNLA_Baseline6kW_????_??_??_????.dat"
ROBOCOPY $SourceDirectory $DestinationDirectory $FilesToCopy /COPY:DAT /M | out-file $LogFileName -append

# Add a few new lines to the Log File for readability
"`r`n`r`n" | out-file $LogFileName -append
```

If the above script were named "TransferBaselineRTCData.ps1" located at C:\Campbellsci\LoggerNet on the host PC, then a system scheduler task such as this would run the script at a specified time (or times) each day.



APPENDIX B: SAMPLE SCRIPT FOR FTP TO A PARTNER

The following script is designed to run using Windows PowerShell and uses the built-in FTP client provided with Windows. This could be used to transfer multiple types of files such as MET, Baseline PV, and Partner PV data. This script could also transfer to multiple different FTP accounts.

```
# Set the source directory and move to it
$WorkingDirectory = "C:\WorkSpace\SendToPartner
cd $WorkingDirectory

# Get the Current Time and Date as a string, write a human readable string to transfer log file
$DateStringForMachines = ((get-date).tostring('yyyyMMdd\THHmmss'))

# Set the temporary directory and go there
$TemporaryDirectory = "$WorkingDirectory\$DateStringForMachines"
New-Item $TemporaryDirectory -ItemType Directory
cd $TemporaryDirectory

<#====================  FILE TYPE 1  ========================================================#>

# Set the name and path to the transmission log file, start writing
$LogFileName1 = "$WorkingDirectory\PartnerMetTransferLog.txt"

# Set the source directory
$SourceDirectory1 = "\\SNL\Collaborative\RTC1\Solar Monitoring\Solar Monitoring File Backups"

# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $LogFileName1 -append

# Select the most recent files to copy
$filter = "*SITEPREFIX_SUFFIX1*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter | Sort-Object LastWriteTime | select -
last 7

# Copy Files from the Source to the Temporary Directory
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append

<#====================  FILE TYPE 2  ========================================================#>

# Set the name and path to the transmission log file, start writing
$LogFileName2 = "$WorkingDirectory\PartnerTransferLog1.txt"

# Set the source directory
$SourceDirectory2 = "R:\Solar Monitoring\Solar Monitoring File Backups"

# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $LogFileName2 -append

# Select the most recent files to copy
$filter = "*SITEPREFIX_SUFFIX2*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory2 -Filter $filter | Sort-Object LastWriteTime | select -
last 7

# Copy Files from the Source to the Temporary Directory
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory | Out-File $LogFileName2 -Append


<#====================  FILE TYPE 3  ========================================================#>

# Set the name and path to the transmission log file, start writing
$LogFileName3 = "$WorkingDirectory\PartnerTransferLog1.txt"

# Set the source directory
$SourceDirectory3 = "R:\Solar Monitoring\Solar Monitoring File Backups"

# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $LogFileName3 -append

# Select the most recent files to copy
$filter = "*SITEPREFIX_SUFFIX3*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory3 -Filter $filter | Sort-Object LastWriteTime | select -
last 7

# Copy Files from the Source to the Temporary Directory
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory | Out-File $LogFileName3 -Append


<#====================  FTP Data to destination 1 ============================================#>

$FTPLogFileName1 = "$WorkingDirectory\PartnerFTPLogFile.txt"
# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $FTPLogFileName1 -append

# Set up the text for the ftp script
```

```
$FTPFileName1 = "FTPScriptName.ftp"
$FTPAddress1 = "PartnerFTPAddress.com"
$FTPUserName1 = "UserName"
$FTPPassword1 = "Password"
$OtherFTPCommands1 = "OtherCommandsLikeCD"

# Create the ftp script
"open ""$FTPAddress1""" | out-file $FTPFileName1 -encoding ascii
$FTPUserName1  | out-file $FTPFileName1 -encoding ascii -Append
$FTPPassword1 | out-file $FTPFileName1 -encoding ascii -Append
"bin" | out-file $FTPFileName1 -encoding ascii -Append
$OtherFTPCommands1  | out-file $FTPFileName1 -encoding ascii -Append
"mput *.dat" | out-file $FTPFileName1 -encoding ascii -Append
"quit" | out-file $FTPFileName1 -encoding ascii -Append

# Run the FTP script and note the results in an output file (in the source directory), delete the FTP script
ftp -i -w:32768 -s:$FTPFileName1 | out-file $FTPLogFileName1 -append
Remove-Item $FTPFileName1

# Add a few new lines to the Log File for readability
"`r`n`r`n" | out-file $FTPLogFileName1 -append

<#====================  Clean Up  ========================================================#>
# Go back to the source directory
cd $WorkingDirectory

# Delete the temporary directory
Remove-Item $TemporaryDirectory -recurse
```

APPENDIX C: SAMPLE SCRIPT FOR FTPS TO A PARTNER

The following script is designed to run using Windows PowerShell and uses the freely available psftp program. This could be used to transfer multiple types of files such as MET, Baseline PV, and Partner PV data. This script could also transfer to multiple different FTP accounts.

```
# Set the source directory and move to it
$WorkingDirectory = "C:\WorkSpace\ToPartnerFTP"
cd $WorkingDirectory

# Get the Current Time and Date as a string, write a human readable string to transfer log file
$DateStringForMachines = ((get-date).tostring('yyyyMMdd\THHmmss'))

# Set the temporary directory and go there
$TemporaryDirectory = "$WorkingDirectory\$DateStringForMachines"
New-Item $TemporaryDirectory -ItemType Directory
cd $TemporaryDirectory

<#====================  FILE TYPE 1  ========================================================#>

# Set the name and path to the transmission log file, start writing
$LogFileName1 = "$WorkingDirectory\PartnerMETTransferLog.txt"

# Set the source directory
$SourceDirectory1 = "\\SNL\Collaborative\RTC1\Solar Monitoring\Solar Monitoring File Backups"

# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $LogFileName1 -append

# Select the most recent files to copy
$filter = "*SITEPREFIX_SUFFIX1*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter | Sort-Object LastWriteTime | select -
last 7

# Copy Files from the Source to the Temporary Directory
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append

<#====================  FILE TYPE 2  ========================================================#>

# Set the name and path to the transmission log file, start writing
$LogFileName2 = "$WorkingDirectory\ PartnerMETTransferLog2.txt"

# Set the source directory
$SourceDirectory2 = "R:\Solar Monitoring\Solar Monitoring File Backups"

# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $LogFileName2 -append

# Select the most recent files to copy
$filter = "*SITEPREFIX_SUFFIX2*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory2 -Filter $filter | Sort-Object LastWriteTime | select -
last 7

# Copy Files from the Source to the Temporary Directory
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory | Out-File $LogFileName2 -Append

<#====================  FTPS Data to destination 1 ==========================================#>

$FTPLogFileName2 = "$WorkingDirectory\FTPLogFilePVMC.txt"
# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $FTPLogFileName2 -append

# Set up the text for the ftp script
$FTPFileName2 = "FTPSScriptName.ftps"
$FTPAddress2 = "PartnerFTPAddressOrIP"
$FTPUserName2 = "UserName"
$FTPPassword2 = 'Password'

# Create the ftp script
"cd foldername1" | out-file $FTPFileName2 -encoding ascii
"mput *SITEPREFIX_SUFFIX1*.dat"  | out-file $FTPFileName2 -encoding ascii -Append
"cd .." | out-file $FTPFileName2 -encoding ascii -Append
"cd foldername2" | out-file $FTPFileName2 -encoding ascii -Append
"mput *SITEPREFIX_SUFFIX2*.dat" | out-file $FTPFileName2 -encoding ascii -Append
"quit" | out-file $FTPFileName2 -encoding ascii -Append


# Run the FTP script using psftp, the output file doesn't work
Start-Process -FilePath "..\psftp.exe" -Wait -ArgumentList "-l $FTPUserName2 -pw $FTPPassword2 $FTPAddress2 -b
$FTPFileName2 -be" | out-file $FTPLogFileName2 -append
#  delete the FTP script
Remove-Item $FTPFileName2

# Add a few new lines to the Log File for readability
"`r`n`r`n" | out-file $FTPLogFileName2 -append
```

```
<#====================  Clean Up  ==========================================================#>
# Go back to the source directory
cd $WorkingDirectory

# Delete the temporary directory
Remove-Item $TemporaryDirectory -recurse
```

APPENDIX D: SAMPLE SCRIPT FOR SCP WITHIN SANDIA

The following script is designed to run using Windows PowerShell and uses the freely available WinSCP program. A subsequent WinSCP batch file is also required, shown immediately below the PowerShell script. In this case, the database creator asked to *not* copy files to the same location after those files had been copied once. Thus, we could not use the filter to select a number of most recent files and move all of them. We also could not count on the archive flag of the files to be consistently set. The script below, therefor, only copy files which have a date in the filename which corresponds to the prior day. For example, if the script is run on July 24, 2016, the only files to be copied will have the date "2016_07_23" in their file name. The script should be run periodically each day using a task scheduler.

```
<#
By: Dan Riley
2016-06-27

This script works to secure copy several files to an SNL server. It does the following:
1. Goes to a working directory set by the user
2. Creates a temporary directory with the date and time as the directory name
3. Copies a series of files from a source directory into the temporary directory.
    It only copies files with a file name that contains the prior day's date
    in the form YYYY_MM_DD.
4. Initiates a WinSCP with a script file to transfer all the files into the SNL server
5. Removes the temporary directory


To change how WinSCP works, you need to edit the script file.

#>

# Set the source directory and move to it
$WorkingDirectory = "C:\Users\pvtestr\Documents\RTCDataTransfer"
cd $WorkingDirectory

# Get the Current Time and Date as a string, write a human readable string to transfer log file
$DateStringForMachines = ((get-date).tostring('yyyyMMdd\THHmmss'))

# Get yesterday's date in the Campbell Filename/time convention in order
# to use the string for copying only files with yesterday's date
$YesterdaysDate = $YesterdaysDate = ((get-date).AddDays(-1).tostring('yyyy_MM_dd'))


# Set the temporary directory and go there
$TemporaryDirectory = "$WorkingDirectory\$DateStringForMachines"
New-Item $TemporaryDirectory -ItemType Directory
cd $TemporaryDirectory



# Set the name and path to the transmission log file, start writing
$LogFileName1 = "$WorkingDirectory\SCPFileTransferLog.txt"

# Set the source directory
$SourceDirectory1 = "R:\Solar Monitoring\Solar Monitoring File Backups"

# Create a File Name Entry
$DateStringForHumans = ((get-date).tostring('yyyy-MM-dd HH:mm:ss')) | out-file $LogFileName1 -append

<#====================  FILE TYPE 1  ========================================================#>

# Select the most recent files to copy
$filter = "*IBMW_MET_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}
<#====================  FILE TYPE 2  ========================================================#>

# Select the most recent files to copy
$filter = "*IBMW_BASE_6KW_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}
<#====================  FILE TYPE 3  ========================================================#>
```

```
# Select the most recent files to copy
$filter = "*FSEC_Baseline_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}

<#====================  FILE TYPE 4  ========================================================#>

# Select the most recent files to copy
$filter = "*FSEC_MET_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}

<#====================  FILE TYPE 5  ========================================================#>

# Select the most recent files to copy
$filter = "*LVRM_Baseline_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}

<#====================  FILE TYPE 6  ========================================================#>

# Select the most recent files to copy
$filter = "*LVRM_MET_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}

<#====================  FILE TYPE 7  ========================================================#>

# Select the most recent files to copy
$filter = "*SNLA_Baseline6kW_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}

<#====================  FILE TYPE 8  ========================================================#>

# Select the most recent files to copy
$filter = "*SNLA_MET_$YesterdaysDate*.dat"
$FilesToCopy = Get-ChildItem -Path $SourceDirectory1 -Filter $filter

# Copy Files from the Source to the Temporary Directory
If ($FilesToCopy) {
Copy-Item $FilesToCopy.FullName -Destination $TemporaryDirectory  | Out-File $LogFileName1 -Append
}


<#====================  SCP Data to destination ============================================#>

$SCPScriptFileName = "SCPTransferScript.txt"

& "C:\Program Files (x86)\WinSCP\WinSCP.com" /script="$WorkingDirectory\$SCPScriptFileName"

# Add a few new lines to the Log File for readability
"`r`n`r`n" | out-file $LogFileName1 -append

<#====================  Clean Up  ==========================================================#>
# Go back to the source directory
cd $WorkingDirectory

# Delete the temporary directory
Remove-Item $TemporaryDirectory -recurse
```

The above script calls the file “SCPTransferScript.txt” to provide commands for WinSCP. The contents of “SCPTranferScript.txt” are provided below, with the exception of the user name and password. To change the behavior of WinSCP, read the WinSCP documentation.

```
# Open an SCP session with savm0156lx.sandia.gov and go to the desired directory
open scp://SNLUSERID:PASSWORD@share.sandia.gov/webssl/rtc-data/_data/ -certificate=*

# Put all the desired files from the current local working directory into the remote computer
put *.dat

# Close the session
close

# Exit WinSCP
exit
```

APPENDIX E: SCRIPT FOR RETRIEVING FILES FROM AN FTP SERVER

The following script is designed to run using Windows PowerShell and uses the built-in FTP client provided with Windows as well as the Robocopy program provided with Windows. It will retrieve the most recent 6 days of data from the cluster controller at the Vermont site. The server IP, user name, and password have been omitted for security purposes.

```
<#
This script is made for grabbing a single day of 60kW Suniva data from the VT
RTC. The 60kW Suniva system uses 3 inverters and collects data on an SMA
cluster controller. The cluster controller has its own ftp wherein the data
is placed at /yyyy/MM/yyyyMMdd.csv where yyyy is a 4 digit year, MM is a 2
digit month, and yyyyMMdd is an 8 digit date. The script actually goes
back 6 days and retrieves each file from up to 6 days back. This is to
produce more reliability. In the event that the script is not run on one
day, the files from that day can be collected up to 6 days later. The
script should be run on the DAY AFTER the files you wish to retrieve.
For example, if you want the files from January 25, 2016, through
January 20, 2016 run the script on January 26, 2016. Note that the
VT RTC cluster controller is on Eastern Standard Time and dates the
data accordingly.
#>

# Get today's date in yyyyMMdd format
$TodaysDate = ((get-date).tostring('yyyyMMdd'))

# Set the name and path to the transmission log file, start writing
$LogFileName = "Vermont60kwSunivaTransferLog.txt"
$LogFileFullName = "$PWD\$LogFileName"

# Set the destination directory this is where the files will be placed
$DestinationDirectory = "\\SNL\Collaborative\RTC1\Suniva Baseline\Vermont 60kW Cluster Controller Data"

# Create a new directory called TempDiryyyyMMdd and move into it
New-Item TempDir$TodaysDate -type directory
cd TempDir$TodaysDate

# Set up the text for the "get files from ftp" script, this is where the file will be placed
$FTPScriptName = "60kwTransferScript.ftp"
$FTPAddress = "SERVERNAMEORIPADDRESS"
$FTPUserName = "USERNAME"
$FTPPassword = "PASSWORD"

# Create the ftp script to get the files from the FTP
"open ""$FTPAddress""" | out-file $FTPScriptName -Encoding ascii #No append flag, to create new file
$FTPUserName | out-file $FTPScriptName -Encoding ascii -Append
$FTPPassword | out-file $FTPScriptName -Encoding ascii -Append
"bin" | out-file $FTPScriptName -encoding ascii -Append

# Add more lines to the ftp script to go back 6 days and get all the .csv
# files for the last 6 days
foreach ($DayBack in -1,-2,-3,-4,-5,-6)
{
    $DayDate = ((get-date).AddDays($DayBack).tostring('yyyyMMdd'))
    $DayMonth = ((get-date).AddDays($DayBack).tostring('MM'))
    $DayYear = ((get-date).AddDays($DayBack).tostring('yyyy'))
    $FileToGetName = "$DayDate.csv"
    "cd /CSV/$DayYear/$DayMonth" | out-file $FTPScriptName -Encoding ascii -Append
    "get $FileToGetName" | out-file $FTPScriptName -Encoding ascii -Append
}
"quit" | out-file $FTPScriptName -Encoding ascii -Append

# Get the current time in yyyyMMdd\THHmmss format
$CurrentDateTime = ((get-date).tostring('yyyyMMdd\THHmmss'))

# Write the date into the log file, run the ftp script, place the output into a log file
get-date -Format "yyyy-MM-dd HH:mm:ss" | out-file $LogFileFullName -append
ftp -i -w:32768 -s:$FTPScriptName | out-file $LogFileFullName -append

# Copy the newly acquired csv files from the temporary directory to the destination directory
ROBOCOPY $PWD $DestinationDirectory *.csv /COPY:DAT /NP /XO | Out-File $LogFileFullName -Append

# Go up a directory
cd ..

# Delete the temporary directory and all of its contents
Remove-Item TempDir$TodaysDate -Recurse -Force

# Add a few new lines to the Log File for readability
"`r`n`r`n" | out-file $LogFileFullName -append
```

APPENDIX F: VISUAL BASIC CODE FOR REMOVAL OF ATTACHMENTS FROM EMAIL

The following snippet of Visual Basic code can be used to create an Outlook rule to automatically move attachments to emails into a folder, in this case C:\RNRG. The file containing this code should be called AutosaveAttachments.bas.

The Visual Basic file should be run using an Outlook rule. For example, a rule could be created to run AutosaveAttachments.bas on all unread emails in the blue category, then mark all of the emails as read, clear all category markings, and finally mark all of the emails with the green category. The blue and green categories are not important in and of themselves, but simply serve as an indicator of emails that have not yet had their attachments stripped and those that have had their attachments stripped, respectively. In order to run a rule which contains Visual Basic code, macros must be enabled within Outlook.

A user would then perform the following actions to strip the attachments from all emails. First, mark all of the new emails to be in the blue category. Then manually run the rule described above.

```
Attribute VB_Name = "AutosaveAttachments"
Public Sub saveAttachtoDisk(itm As Outlook.MailItem)
Dim objAtt As Outlook.Attachment
Dim saveFolder As String
saveFolder = "C:\RNRG"
     For Each objAtt In itm.Attachments
          objAtt.SaveAsFile saveFolder & "\" & objAtt.DisplayName
          Set objAtt = Nothing
     Next
End Sub
```

DISTRIBUTION

1	MS0951	James Stephens	6112
1	MS1033	Joshua Stein	6112
1	MS1033	Abraham Ellis	6112
1	MS0899	Technical Library	9536 (electronic copy)


Sandia National Laboratories