Last updated on 01/08/2017

Disclaimer

This script comes as-is, with no warranties either expressed or implied. I am not, and can not, be held responsible for any damages or loss of data that occurs as a result of downloading and using this script. Only continue if you completely understand this, accept these risks, have read and understand the known issues listed below, and have the knowledge and skills necessary to implement this process.

Introduction

wxStationBackup is a configurable PowerShell script designed to enable the automated data backup of a weather station being operated by the WeatherLink software. The backups, once made, are organized in to a logical directory structure, allowing you to easily locate a backup based on date/time. The directory structure created by this script is:

<DestinationDirectory\YYYY\MM\DD\wlb_hhmm
YYYY = four digit year
MM = two digit month
DD = two digit day
hh = two digit hour
mm = two digit minute

Optionally, this script can be configured to send email alerts containing results of the backup operation.

Change Log

Changes for version **1.0.3**

- Added support for 'Staged' backup. This allows us to initially backup to faster storage in the first stage, the second stage would move the backup to slower, more permanent storage.
- Added support for 7-zip, allowing us to compress our backups. Tests reveal this could reduce storage for a single backup by +80%.
- Excluded a couple of files from the backup to reduce 'file in-use' warnings during backups.
-

Getting Started

To get started with this script:

- Download and read this document in it's entirety.
- Read and agree to the statements in the disclaimer again.
- Read and pay close attention to the Known issue section above.
- Download, unzip, and place this script on the computer that is running WeatherLink. For example, create a folder called scripts on your c:\ and place the script there.
- Calculate the space requirements you will need to keep the number of backups you desire.
- Configure the script in accordance with the 'Configuring Variables' section below.
- Test run this script to make sure it runs as you expect.
- Configure an automated script to run this script at your desired frequency.

Configuring Variables

There are four variables that are require configuration in order for this script to run. Optionally, there are six variables that can be configured in order to enable email alerts.

$destDir (required)

This variable tells the script where you want you backups to be copied. For example, “e:\weatherlink_backups\” (the double quotes are required). Please take the following into consideration:

- This should be an empty folder on the drive, not the root of the drive. For example, use 'e:\backups' instead of 'e:\'
- If another drive with sufficient space is available, place the backups here and not on your c:\. This will keep you from filling up your boot volume.
- If possible, place the backups on a drive that does not include your production weather station data. This has two benefits. First, it prevents you from filling up that drive with backups. If the drive that contains your production data is full, WeatherLink will not be able to write new observation data.

Configuring Variables *(continued)*

Second, it adds another layer of redundancy. Placing the backups on a drive different than that of our production data prevents you from losing both sets of data in the event of a drive failure.

$srcDir (required)
This variable informs the script where your **production data** resides. This is the data you want to backup. Generally, this will be "c:\WeatherLink\" (the double quotes are required). If you specified a custom directory when you installed WeatherLink, that directory should be specified.

$tempDir
If you wish to use staged copy, this variable must be set. It will be used as a temporary location to store the initial backup.

$archiveInterval(required)
This variable specifies in days, how long you wish to keep backups. For example, if you wish to keep your backups for 30 days, this should be set to 30. The limiting factor to this setting is available drive space. Use this formula to calculate the necessary drive space:

RequiredDriveSpace = ((SizeOfProductionDirectory * NumberOfDailyBackups) * ArchiveInterval)

For example, if the production directory is 400Mb, you are running the script hourly, and you are keeping the backups for 30 days, your calculation will look like this:

RequiredDrivePace = ((400 * 24) * 30)
Your required drive space would be **288,000 Mb** or 281.25 Gb

$enable_email_alerts(required)
This variable will enable/disable email alerts. To turn alerts on, set this to $true. To turn alerts off, set this to $false. **Note:** Powershell version 3 is required for this functionality.

$enable_staged_copy(required)
This variable will enable/disable staged copy. The purpose of staged copy is to allow the script to initially backup to a temporary location. Then, once complete, the backup in the temporary location will be moved to the final location. The thought here is that the initial backup will be made on faster storage (SSD), then moved to slower, more abundant storage. This should reduce the amount of time we are copying the live data, thus minimizing the chances of 'file in use' message while copying. To enable staged copy, set this to $true. To disable staged copy, set this to $false.

$enable_backup_compression(required)
This variable defines whether or not the script should try to compress the backups. Keep in mind, 7zip is required for this. To enable compression, set this to $true. To disable compression, set this to $false.

$7zip_executable_path(required for compression)
If you want to use compression, you have to specify where the 7zip executable is installed. The default path is specified. If it is installed to a different location, specifiy it here.

$emailFrom(optional. Only required if $enable_email_alerts is set to $true)
This variable specifies from whom the email will come. For example, "from@example.com" (the double quotes are required).

$emailFrom(optional. Only required if $enable_email_alerts is set to $true)
This variable specifies from whom the email will come. For example, "from@example.com" (the double quotes are required).

$emailTo(optional. Only required if $enable_email_alerts is set to $true)
This variable specifies who will be receiving the notifications. For example, "to@example.com" (the double quotes are required).

Configuring Variables *(continued)*

$smtpServer(optional. Only required if $enable_email_alerts is set to $true)
This variable specifies the SMTP server that will be sending the email. I've left this defaulted to "smtp.gmail.com" (the double quotes are required).. If you have a gmail account, you too can use this too. You will have to make a setting change to your gmail account in order to use this. Copy and paste this link into your browser for more information: https://support.google.com/accounts/answer/6010255?hl=en . Only make this change if you understand what you are doing and understand what risks are involved.

$smtpPort(optional. Only required if $enable_email_alerts is set to $true)
This variable specifies a non-standard port that your server requires. Normally, this would be set to 25. However, I have left this set to "587" (the double quotes are required).

$smtpUsername(optional. Only required if $enable_email_alerts is set to $true)
If authentication is required, this variable specifies a username to be used for authentication. Gmail does require this. This will need to be set to "<your gmail username>" (the double quotes are required). Understand that your username will be stored as clear text in this file and will be visible to anyone to has access to open and read this script.

$smtpPassword(optional. Only required if $enable_email_alerts is set to $true)
If authentication is required, this variable specifies a password to be used for authenticaton. Gmail does require this. This will need to be set to "<your gmail password>" (the double quotes are required). Understand that your password will be stored as clear text in this file and will be visible to anyone to has access to open and read this script.

Setting up the Automated Task

It is important while setting up the task, that the command be structured like this:

PowerShell -ExecutionPolicy UnRestricted -File "C:\test\wxStationBackup.ps1"

Failure to do so will cause the task to fail.

Call it laziness if you will, but I do not see the point in creating documentation for doing this when this process is very well documented in other places. For example, copy and paste this link into your browser for more information: https://support.software.dell.com/appassure/kb/144451 . You need to determine how frequently you want to make backups. For example, I run the script every hour on my system. If I experience a drive failure on my WeatherLink Server, I should lose no more than an hour's worth of data at any given time. Remember, as you did with configuring the archive interval, take in to consideration drive space when determine how frequently you will run this script. Use this formula as a guide:

RequiredDriveSpace = ((SizeOfProductionDirectory * NumberOfDailyBackups) * ArchiveInterval)

For example, if the production directory is 400Mb, you are running the script hourly, and you are keeping the backups for 30 days, your calculation will look like this:

RequiredDrivePace = ((400 * 24) * 30)
Your required drive space would be **288,000 Mb** or **281.25 Gb**

For help in configuring the task, please look at the screen shots below. I used these specifications when setting up the task on my system:

Setting up the Automated Task *(continued)*

Figure 1. General Tab

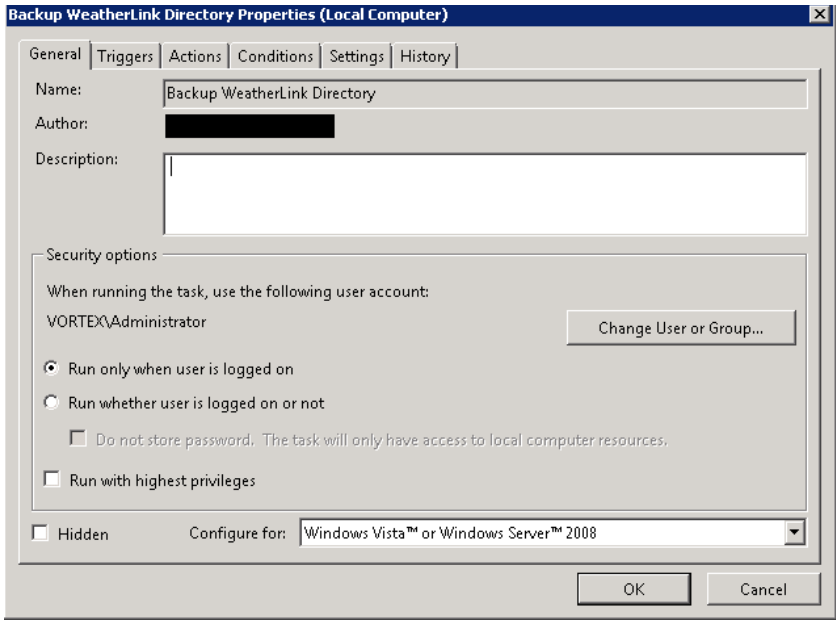
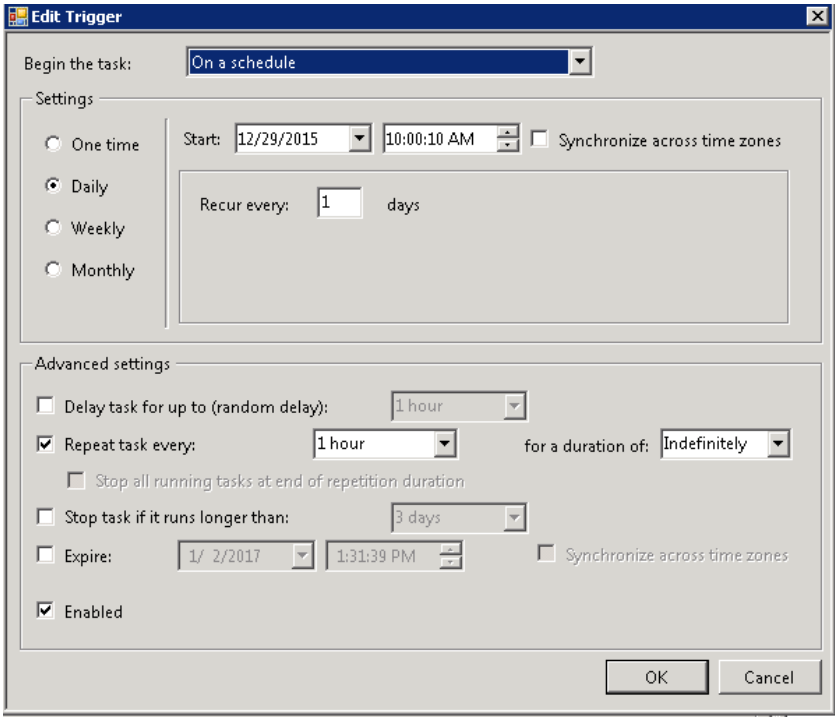


Figure 2. Trigger Details. (Note the Start Time. Please see Known issues at the beginning of this document)



Setting up the Automated Task *(continued)*

Figure 3. Actions