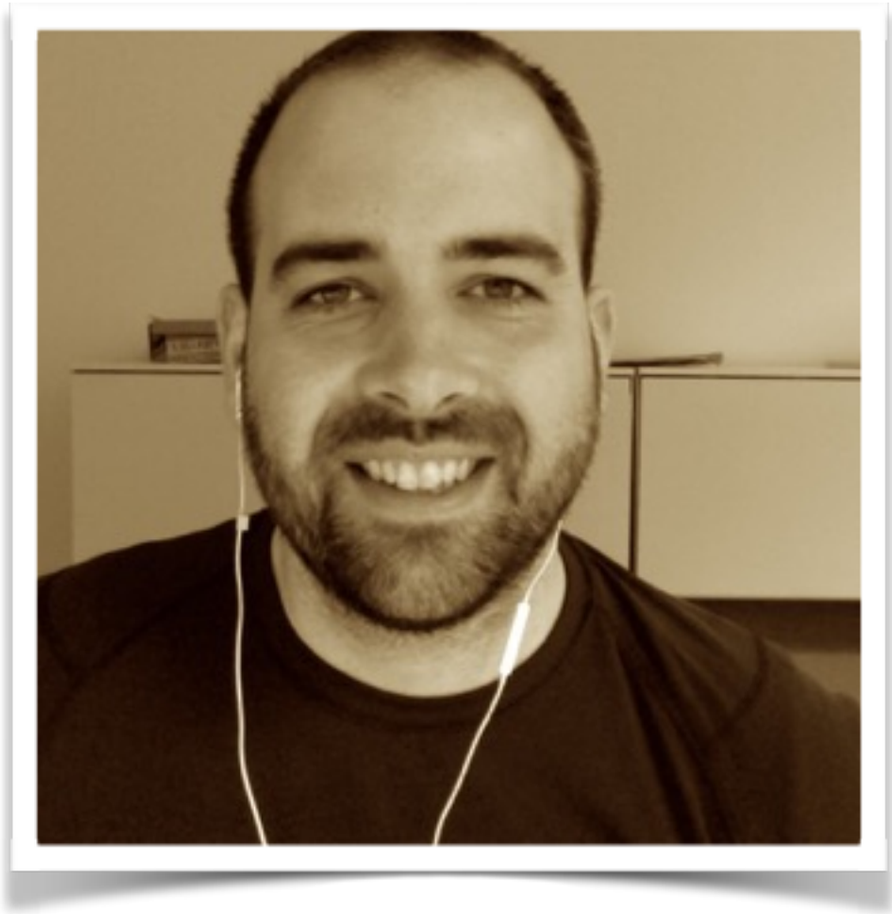
Chris Sheppard

Chris has been supporting Macs in business, education, healthcare and non-profit for 10+ years and currently resides as a Macintosh Desktop Engineer at Fermi National Accelerator Laboratory in Batavia, Illinois.

Chris, along with another individual is responsible for engineering and overseeing all aspects of the Apple infrastructure and fleet including the JAMF Casper Suite, imaging, anti-virus, reporting and compliance and much much more…



Twitter: @cshepp11

Slack: @cshepp11

Email: cshepp@fnal.gov

Backing up Cloud Servers and Remote Storage

Chris Sheppard
Fermi National Accelerator Laboratory
10 August 2016

In partnership with:

What are we going to cover?

- Servers and storage in the cloud need backup
  - What is a Cloud Server?
  - What is Cloud Storage?
  - Why do we need to backup cloud servers and storage?
- Build vs. Buy
- rsync
  - what is it and what does it do
  - rsync in OS X
  - man rsync
  - common rsync options
  - basic rsync usage
- Automation vs. user interaction
- Power of scripting
- Real examples of rsync and scripting
- Types of backups
- Q&A

Servers and storage in the cloud need backup

What is a Cloud Server?

A cloud server is a logical server that is built, hosted and delivered through a cloud computing platform over the Internet. Cloud servers possess and exhibit similar capabilities and functionality to a typical server but are accessed remotely from a cloud service provider.

A cloud server may also be called a virtual server or virtual private server.

Example Uses: Websites, Databases, Email, DNS, Etc…

techopedia. N.p., n.d. Web. 5 Aug. 2016. <https://www.techopedia.com/definition/29019/cloud-server>.

Servers and storage in the cloud need backup

What is Cloud Storage?

Cloud storage is a cloud computing model in which data is stored on remote servers accessed from the Internet, or "cloud." It is maintained, operated and managed by a cloud storage service provider on storage servers that are built on virtualization techniques.

Cloud storage is also known as utility storage - a term subject to differentiation based on actual implementation and service delivery.

Example Uses: Website Content, Data/File Sharing, FTP Sites, Etc…

techopedia. N.p., n.d. Web. 5 Aug. 2016. <https://www.techopedia.com/definition/26535/cloud-storage>.

Servers and storage in the cloud need backup

Why do we need to backup cloud servers and storage?

There are many reasons why you should back up your cloud servers and storage.

- Data Center Disaster
- Hosting Provider goes under
- Greater backup frequency
- Physical possession of data
- The list goes on…

Build vs. Buy

The debate on build vs. buy has been raging on for years. Building a backup process can have several benefits and can far outweigh buying and managing backup software/ solutions.

Build

- Completely customizable
- Low cost of ownership (free)
- Generally smaller learning curve???
- Documentation, it's everywhere!
  - About 2,740,000 rsync results
- In contrast, free software=no/little support

Buy

- Cost!!!
- Maintaining software updates, version issues, possible downtimes, etc
- Training involved, higher learning curve.

Of course, if you already have an off-the-shelf backup software/solution or there is a product that meets all of your requirements, then use it. This presentation will focus on building a backup solution using the command line tool, rsync.

rsync - What is it and what does it do?

Rsync is a fast and extraordinarily versatile file copying tool. It can copy locally, to/from another host over any remote shell, or to/from a remote rsync daemon. It offers a large number of options that control every aspect of its behavior and permit very flexible specification of the set of files to be copied. It is famous for its delta-transfer algorithm, which reduces the amount of data sent over the network by sending only the differences between the source files and the existing files in the destination. Rsync is widely used for backups and mirroring and as an improved copy command for everyday use.

Some of the additional features of rsync are:
- support for copying links, devices, owners, groups, and permissions
- exclude and exclude-from options similar to GNU tar
- a CVS exclude mode for ignoring the same files that CVS would ignore
- can use any transparent remote shell, including ssh or rsh
- does not require super-user privileges
- pipelining of file transfers to minimize latency costs
- support for anonymous or authenticated rsync daemons (ideal for mirroring)

die.net. N.p., n.d. Web. 5 Aug. 2016. <http://linux.die.net/man/1/rsync>.

Rsync in OS X

The default OS X version of rsync is old and deprecated: 2.69

```
| => rsync --version
rsync  version 2.6.9  protocol version 29
Copyright (C) 1996-2006 by Andrew Tridgell, Wayne Davison, and others.
<http://rsync.samba.org/>
Capabilities: 64-bit files, socketpairs, hard links, symlinks,
batchfiles,
              inplace, IPv6, 64-bit system inums, 64-bit internal inums

rsync comes with ABSOLUTELY NO WARRANTY.  This is free software, and you
are welcome to redistribute it under certain conditions.  See the GNU
General Public License for details.
```

The current rsync version is 3.1.2 and as of 10.11 El Capitan, /usr/bin/rsync cannot be replaced or modified due to SIP.

Let's update rsync

- Download new rsync to /usr/local/bin/rsync, build and install.

```
curl -O http://rsync.samba.org/ftp/rsync/rsync-3.1.2.tar.gz
tar -xzvf rsync-3.1.2.tar.gz
rm rsync-3.1.2.tar.gz
cd rsync-3.1.2
./prepare-source
./configure
make
sudo make install
```

- Compare:

```
| => /usr/local/bin/rsync --version
rsync  version 3.1.2  protocol version 31
```

```
| => /usr/bin/rsync --version
rsync  version 2.6.9  protocol version 29
```

Man rsync

```
| => /usr/local/bin/rsync -man
rsync  version 3.1.2  protocol version 31
Copyright (C) 1996-2015 by Andrew Tridgell, Wayne Davison, and others.
Web site: http://rsync.samba.org/

rsync is a file transfer program capable of efficient remote update
via a fast differencing algorithm.

Usage: rsync [OPTION]... SRC [SRC]... DEST
  or   rsync [OPTION]... SRC [SRC]... [USER@]HOST:DEST
  or   rsync [OPTION]... SRC [SRC]... [USER@]HOST::DEST
  or   rsync [OPTION]... SRC [SRC]... rsync://[USER@]HOST[:PORT]/DEST
  or   rsync [OPTION]... [USER@]HOST:SRC [DEST]
  or   rsync [OPTION]... [USER@]HOST::SRC [DEST]
  or   rsync [OPTION]... rsync://[USER@]HOST[:PORT]/SRC [DEST]
```

Notice that I am calling the full path of rsync above. You can edit your user shell path to include the new (updated) rsync utility so that you do not need to call the full path when using.

Common rsync options

- -P: show partial progress bar during transfer
- -h: human-readable numbers
- -a: “archive” mode copies nearly everything as-is
- -v: increase verbosity
- -z: compression
- rsync -Phavz source destination
  - Shows progress, human readable file sizes, copies mostly everything, shows every file being copied and compresses for lower bandwidth.

```
| => rsync -Phavz /Users/cshepp/Documents/mactech/MacTech\ Pics  /Volumes/Storage02
/MactechBackup
sending incremental file list
MacTech Pics/
MacTech Pics/Data-Center-Fire-Dangers.jpg
        152.98K 100%   38.21MB/s    0:00:00 (xfr#1, to-chk=5/7)
MacTech Pics/MacTech_Pro_Events-1200x630.png
         34.78K 100%    6.63MB/s    0:00:00 (xfr#2, to-chk=4/7)
MacTech Pics/Now-what.jpg
        123.17K 100%   11.75MB/s    0:00:00 (xfr#3, to-chk=3/7)
MacTech Pics/cloudserver.jpg
        397.62K 100%   12.64MB/s    0:00:00 (xfr#4, to-chk=2/7)
MacTech Pics/filestoragebackup.jpg
        307.12K 100%    6.97MB/s    0:00:00 (xfr#5, to-chk=1/7)
MacTech Pics/rsync.jpg
          4.29K 100%   99.70kB/s    0:00:00 (xfr#6, to-chk=0/7)

sent 954.28K bytes  received 134 bytes  1.91M bytes/sec
total size is 1.02M  speedup is 1.07
```

Basic rsync usage

- rsync -a SOURCE DESTINATION
- rsync -a /path/to/source /path/to/destination
  - This will copy the directory “source” to “destination”
- rsync -a source username@destination:/path/to/server/directory

```
| => rsync -a /Users/cshepp/Documents/mactech/MacTech\ Pics /Volumes/Storage02/
MactechBackup
```

```
| => rsync -a /Users/cshepp/Documents/mactech/MacTech\ Pics
cshepp@someserver.fnal.gov:/home/cshepp/tempstorage/MactechBackup
```

Classic rsync

- Copy contents of directory to destination on server:
  - rsync -avz /path/to/files/ user:password-if-you-want-it-saved@host:/path/to/target/

```
| => rsync -avz /Users/cshepp/Documents/mactech/MacTech\ Pics/
cshepp@someserver.fnal.gov:/home/cshepp/tempstorage/MactechBackup
sending incremental file list
./
.last_rsync_time
Data-Center-Fire-Dangers.jpg
MacTech_Pro_Events-1200x630.png
Now-what.jpg
cloudserver.jpg
filestoragebackup.jpg
rsync.jpg

sent 954,312 bytes  received 155 bytes  636,311.33 bytes/sec
total size is 1,019,966  speedup is 1.07
```

Please note: the trailing / on the source is important to copy the contents of the directory and not the directory itself.

Classic rsync

- Copy directory “source” to server as “destination”, showing progress in human readable format:
  - rsync -Phavz /path/to/files user:password-if-you-want-it-saved@host:/path/to/target/

```
| => rsync -Phavz /Users/cshepp/Documents/mactech/MacTech\ Pics
cshepp@someserver.fnal.gov:/home/cshepp/tempstorage/MactechBackup
sending incremental file list
MacTech Pics/
MacTech Pics/Data-Center-Fire-Dangers.jpg
        152.98K 100%   28.66MB/s    0:00:00 (xfr#1, to-chk=5/7)
MacTech Pics/MacTech_Pro_Events-1200x630.png
         34.78K 100%    6.63MB/s    0:00:00 (xfr#2, to-chk=4/7)
MacTech Pics/Now-what.jpg
        123.17K 100%   11.75MB/s    0:00:00 (xfr#3, to-chk=3/7)
MacTech Pics/cloudserver.jpg
        397.62K 100%   13.54MB/s    0:00:00 (xfr#4, to-chk=2/7)
MacTech Pics/filestoragebackup.jpg
        307.12K 100%    7.14MB/s    0:00:00 (xfr#5, to-chk=1/7)
MacTech Pics/rsync.jpg
          4.29K 100%   99.70kB/s    0:00:00 (xfr#6, to-chk=0/7)

sent 954.28K bytes  received 137 bytes  636.28K bytes/sec
total size is 1.02M  speedup is 1.07
```

“touch”

- touch /path/of/interest .last_rsync_time
  - Will either create or modify the last modified time stamp on a file
  - Can be used to mark files to backup on next run

```
| => touch filestoragebackup.jpg .last_rsync_time
| => ls -l
total 1012
-rw-r--r--@ 1 cshepp  staff  152976 Aug  8 16:28 Data-Center-Fire-Dangers.jpg
-rw-r--r--@ 1 cshepp  staff   34785 Aug  5 09:33 MacTech_Pro_Events-1200x630.png
-rw-r--r--@ 1 cshepp  staff  123175 Aug  8 13:06 Now-what.jpg
-rw-r--r--@ 1 cshepp  staff  397619 Aug  5 11:36 cloudserver.jpg
-rw-r--r--@ 1 cshepp  staff  307123 Aug  8 16:29 filestoragebackup.jpg
-rw-r--r--@ 1 cshepp  staff    4288 Aug  8 12:25 rsync.jpg
```

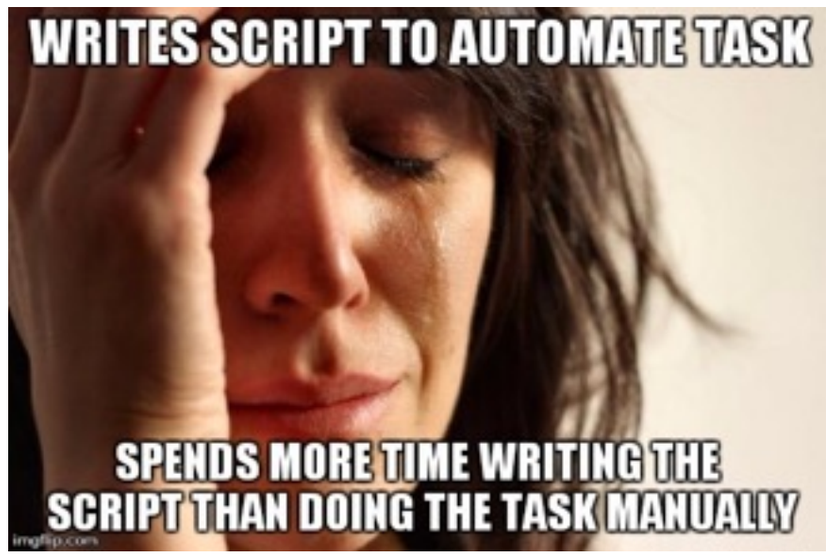
Automation vs. user interaction

Automation vs. user interaction, or manual process each have their strengths and weaknesses. For instance, automation comes with a risk of having simple mistakes in code that could have dire effects on the process. Automation could also create a scenario where the admin is not noticing ineffective or unacceptable output from otherwise successfully executed tasks.

User interaction on the other hand, has risk of inconsistent execution and/or people not doing what they committed to do.

A positive to automating a manual task is that the sysadmin's role changes from doing the task to maintaining the automation which in turn saves time, resources and in some cases, money.

Finally, assess how often a process needs to be run and if it needs to be automated.



Power of scripting

- The real power comes with stringing multiple commands together
- Allows for no dark-time between task 1 completing and the beginning of task 2
- Allows for accurate repetition of task “x” as many times as you need it to be done

```
#!/bin/sh

# This script does personal backups to a rsync backup server. You will end up
# with a 7 day rotating incremental backup. The incremental will go
# into subdirectories named after the day of the week, and the current
# full backup goes into a directory called "current"
# tridge@linuxcare.com

# directory to backup
BDIR=/home/$USER

# excludes file - this contains a wildcard pattern per line of files to exclude
EXCLUDES=$HOME/cron/excludes

# the name of the backup machine
BSERVER=owl

# your password on the backup server
export RSYNC_PASSWORD=XXXXXX


############################################################################

BACKUPDIR=`date +%A`
OPTS="--force --ignore-errors --delete-excluded --exclude-from=$EXCLUDES
      --delete --backup --backup-dir=/$BACKUPDIR -a"

export PATH=$PATH:/bin:/usr/bin:/usr/local/bin

# the following line clears the last weeks incremental directory
[ -d $HOME/emptydir ] || mkdir $HOME/emptydir
rsync --delete -a $HOME/emptydir/ $BSERVER::$USER/$BACKUPDIR/
rmdir $HOME/emptydir

# now the actual transfer
rsync $OPTS $BDIR $BSERVER::$USER/current
```

Good scripting habits

- When scripting, use a form of version control/management.
  - name/number your scripts as you develop them
  - keep notes about what each version add/edits/subtracts/etc
  - source control - upload to GitHub or BitBucket

- Output should help you track
  - have your script create and embed a header line in any output file, quoting the script version

- Consider embedding output or logs throughout your script to provide real-time feedback

- Use version/source control frequently so you can always revert to previous revision, especially if making big changes

Think like a photographer

If you want to create a perfect “image” of a dataset, then think like a photographer.

While you use your iPhone’s panoramic panning, people walk across your pan.

- If walking left to right, you end up with the same person in the picture 3 or 4 times
- If walking right to left, you end up with an unrecognizable blur or a sliver of them

Files blur too

This same effect happens with active files such as:

- large PST files or databases
- log files
- large media files

If the index you captured at blog L is out of data before you're at block N, then the restored file might not work.

Solution: Use application-aware tools or commands to 'dump' a database or a shadow file, then backup that flat/static file.

Real examples

MySQL / PostgreSQL are both great examples of the “blur” we were talking about in the prior slide

MySQL & Website - Backing it up

In order to achieve a proper backup of a MySQL database, we need to use the “mysqldump” command.

The mysqldump client is a utility that performs logical backups, producing a set of SQL statements that can be run to reproduce the original schema objects, table data, or both. It dumps one or more MySQL database for backup or transfer to another SQL server. The mysqldump command can also generate output in CSV, other delimited text, or XML format.

Learn more about mysqldump

http://dev.mysql.com/doc/refman/5.7/en/mysqldump.html

Real examples - continued

Back up everyday, keep for a week

```
#!/bin/sh

THESITE="yourwebsite.com"
THEDB="my_database_name"
THEDBUSER="my_database_user"
THEDBPW="my_database_password"
THEDATE=`date +%d%m%y%H%M`

mysqldump -u $THEDBUSER -p${THEDBPW} $THEDB | gzip > /var/www/vhosts/$THESITE/backups/files/dbbackup_${THEDB}_${THEDATE}.bak.gz

tar czf /var/www/vhosts/$THESITE/backups/files/sitebackup_${THESITE}_${THEDATE}.tar -C / var/www/vhosts/$THESITE/httpdocs
gzip /var/www/vhosts/$THESITE/backups/files/sitebackup_${THESITE}_${THEDATE}.tar

find /var/www/vhosts/$THESITE/backups/files/site* -mtime +7 -exec rm {} \;
find /var/www/vhosts/$THESITE/backups/files/db* -mtime +7 -exec rm {} \;
```

Set some variables in line 3 through 7

Line 9 running mysqldump

Line 11 archive site and store in dir

Line 14 & 15 are deleting backups older than 7 days

- We set some variables in lines 3 through 7
- Line 9 is running a mysqldump of all the data in the database named in line 4, archiving it, and storing it in the files directory using a filename that looks like dbbackup_example.com_1402120101.bak.gz.tar
- Line 11 and 12 are archiving the site's source code files in the httpdocsdirectory, and storing them in the files directory, using a filename that looks like sitebackup_example.com_1402120101.tar.
- Lines 14 and 15 are deleting any backups made more than 7 days ago.

Real examples - continued

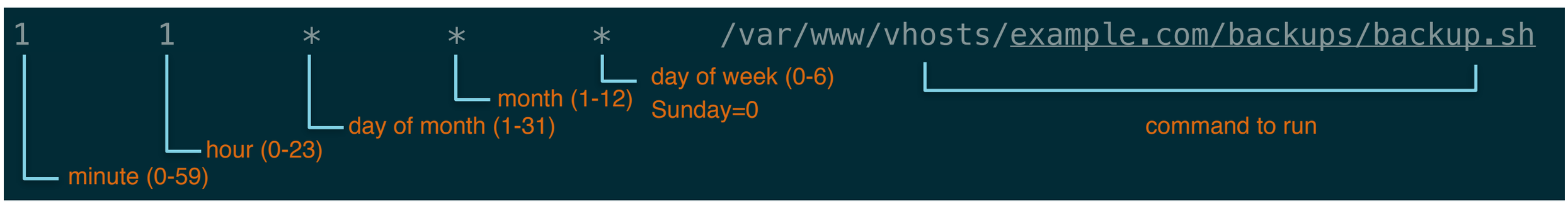
Back up everyday, keep for a week

"Cron" is a time-based job scheduler in Unix-like operating systems (Linux, FreeBSD, Mac OS etc...). And these jobs or tasks are referred to as "Cron Jobs".

There is a cron "daemon" that runs on these systems. A daemon is a program that runs in the background all the time, usually initiated by the system. This cron daemon is responsible for launching these cron jobs on schedule.

The schedule resides in a configuration file named "crontab". That's where all the tasks and their timers are listed.

Let’s create a cronjob that will execute the previous script nightly at 1:01 am.



http://linux.die.net/man/1/crontab

Real examples - continued

So now we have a crontab and a script that will automatically backup our database and website on our local machine or remote web server….but what about backing up our database and website to a different remote server?

Let's set up an rsync script and a cronjob to automate our remote backup

```
#!/bin/sh

rsync -a /var/www/vhosts/example.com/backups/files/ cshepp@someserver.fnal.gov:/your/path/to/offsitebackups/
```

You may need to generate a passwordless login or key pair

```
2        1        *        *        *      /var/www/vhosts/example.com/backups/rsync.sh
```

- 2 — minute (0-59)
- 1 — hour (0-23)
- * — day of month (1-31)
- * — month (1-12)
- * — day of week (0-6), Sunday=0
- /var/www/vhosts/example.com/backups/rsync.sh — command to run

Moving data as part of a persisting workflow

- You may want to script moving data at source, or destination
  - Example: Move some data in a source path, while leaving others (move out files older than 30 days)
- Do you want to MOVE files (remove from source) vs. COPY (leave at source)?
- Do you want to segregate or notate files at the source, if choosing to leave them on the volume (move into a ‘processed’ folder, or change filename)?

Moving files that are more than 180 days old (only) from one location to another

```
# Create list of affected files -- find all the files that are 180 days ago or more
$ find . -type f -ctime +180 -print > ${TMPLIST}

# Sync to the other box (need to define variables)
#   {EXCH_USER}  ${EXCH_HOST}   ${TARGET_DIR}
$ rsync -avz --files-from=${TMPLIST} ./ ${EXCH_USER}@${EXCH_HOST}:${TARGET_DIR}/

# Delete after successful sync
# Dangerous -- be careful
$ cat ${TMPLIST}|xargs rm -f
```

rsync’s cleanup flags

Another way to do the clean up is to use rsync’s built-in clean up flags such as:

```
-e  flags of interest...
--remove-source-files   sender removes synchronized files (non-dir)
--delete                delete extraneous files from dest dirs
--delete-before         receiver deletes before transfer (default)
--delete-during         receiver deletes during xfer, not before
--delete-after          receiver deletes after transfer, not before
--delete-excluded       also delete excluded files from dest dirs
```

So if you want to move data from one machine to another you might use:

```
rsync -avz -e --remove-source-files  --delete-after
/path/to/backupfiles user:password-if-you-want-it-saved@host:/path/to/
targetbackupfiles
```

Excluding files

- Sometimes, there are files you don't want:
  - OS Specific files (e.g., `.DS_STORE`)
  - Installation specific (e.g., configuration paths)
  - Temp or Cache files
  - Source control files (.git, .hg)
- rsync flags to the rescue, exclude any file with 'dir1':

```
rsync ––exclude 'dir1'
```

- 6 rsync Examples to Exclude Files and Folders
  - http://www.thegeekstuff.com/2011/01/rsync-exclude-files-and-folders/

```
$ rm -rf destination

$ rsync -avz --exclude 'dir1' source/ destination/
building file list ... done
created directory dest
./
file1.txt
file2.txt
dir3/
dir3/file4.txt
```

Additional considerations

- Outcome / confirmation and documentation of a script's completion
- Add logging?
- Do you want to append a date/time stamped summary line to a fixed-path log file?
- Add reporting or notification?
- Do you want the script to email you?

Trust, but verify!

- Now that you've built a script or application to move the data, you're not wrong to want to trust it. That's not a bad thing - but trust AND VERIFY.
- Take the time for periodic validation that you can successfully recover from the hypothetical threat that you seek protection from
- Establish what 'normal' behavior looks like (time to run, size of output, etc.) and determine how you'll detect off-nominal developments
- TEST A FULL RESTORE!

Test before going live

- Test regex filter syntax in a non-destructive way
- Regex Testing Tools
  - http://regexr.com
  - http://ryanswanson.com/regexp/#start
  - http://www.regexpal.com
- Additional Regex tools:
  - http://www.cheatography.com/davechild/cheat-sheets/regular-expressions/
  - http://regexlib.com/CheatSheet.aspx
  - http://www.virtuosimedia.com/dev/php/37-tested-php-perl-and-javascript-regular-expressions
  - http://code.tutsplus.com/tutorials/8-regular-expressions-you-should-know--net-6149

Incremental vs. differential

Differential backups: Differential backups are based on the last full backup performed and backup all changes since the last Full backup was performed.

Incremental backups: Incremental backups are based on the last backup performed and backup only the changes since that last backup (Full or Differential).

Incremental and Differential backups were both designed to allow you to backup only files that have changed, although they differ on which files are selected. Incremental backups will back up any changes you have made since your last backup (regardless of type), whereas Differential backups will backup all of the changes since your last Full backup.

Incremental destination file

- Monday: Full backup
- Tuesday: all changes between M and T
- Wed: all changes between T and W
- Thurs: all changes between W and Th.

Differential destination file

- Monday: Full
- Tues: all changes between M and T
- Wed: all changes between M and W
- Thur: all changes between M and Th.

- Incremental allows one restore from the date of latest incremental to get everything.
- Differential requires you restore Monday then Tues then Wed then Thurs.

Questions?

Chris Sheppard

Twitter: @cshepp11

Slack: @cshepp11

Email: cshepp@fnal.gov