installing RUBI for making backups

Note: *Currently RUBI's webpage seems to be offline. Also i found small bug which i have patched but the patched version was not yet online on the author's webpage. Here is*

my patched version of Rubi 4.0

. Always check the below linked webpage to see if maybe there is a newer version available in the meantime as i don't check this frequently.

RUBI is Beat Rubischons “Rubi’s Ultimat Backup Implementation” and it’s indeed quite a clever thing ;)

here’s what it does: every time you run rubi, it copies files from a source to a destination. therefore it creates a new folder in the destination drive containing the current date and time in the directory name. it then copies all files from the source into that new directory. but if it did just regular copies, it would cause quite some disk usage, so rubi came up with a clever solution. all the unchanged files are hardlinked to the previously downloaded copies. this way, we only need the disk space once, but we still have all those directories where we have a whole snapshot of every day. No more messing around with incremental stuff :)

if you know rsync, you might be familiar with the `link-dest` option. that is exactly what rubi does and since version 4 rubi is fully rsync based and uses this option to do all the hardlinking work :)

the only drawback i found so far.. there’s no documentation..

so here’s how i got it up and running:

first, create a new home for rubi. i suggest to install it into `/opt` then download and unpack rubi. it can be downloaded from the rubi section of beat’s webpage always check for updates before copy/pasting my lines!

```
cd /opt
wget http://www.0x1b.ch/misc/hacks/rubi/rubi-4.0.0.tar.gz
tar xzvf rubi-4.0.0.tar.gz
rm rubi-*
```

by the time I first wrote this article, the most recent version was 3.1.0. and rubi depended on many common tools that where included in most os installs. I recently updated this article and rubi is now at verison 4, where most of these dependencies have been dropped and all we need now besides bash and perl are rsync and find.

using `which` or `whereis` you can find out where these tools are located so we can tell rubi

```
which rsync
```

for example will tell you where to find rsync

note down those paths for rsync and find, we need them for the configuration of rubi

now since there is no documentation to rubi, beat has simply included a sample rubi run-script so you

can copy that one and work from there.

you find the scirpt in /opt/rubi/bin which is also the place for our final run-script of rubi too.

```
cd /opt/rubi/bin
cp rubi-sample rubi
nano rubi
```

now let's configure it the way we want it first the global settings.. just make sure all the paths are correct. usually the defaults work. make sure that PREFIX contains your rubi install directory.

you don't need to change anything to the "source shell functions" section ...

and now the most important part.. our actual backup setups.. here's an example of what it looks like:

```
# ------
# Backup of max
# ------
SRCHOST=localhost                                  # source host
SRCPART="/ /boot /export /export/home /export/data /opt/cms"    # source
partitions
BASEDST=/backup/max                                # base backup dir
DSTDIR=$BASEDST/`date +%Y.%m.%d-%H:%M`             # destination
OLDDST="`cat $BASEDST/lastdst 2>/dev/null`"        # last destination
KEEPG=15                                           # successful backups to keep
KEEPD=15                                           # min age before delete

doit | tee $DSTDIR.log
```

if you want to do several different backup sets right after each other, simply put several of these blocks at the end of your rubi file. Make sure you always include the "doit" line at the end of each block, otherwise your backup will not run.

SRCHOST is the source host name. if you do local backups, thats localhost, otherwhise it's an ip or hostname of a remot computer and you will need rsh or ssh and rsync if you have larger files than 2gig

SRCPART is a list of directories you want to backup. rubi will descend in all subdirectories of these direcotires as long as they are in the same filesystem. Since rubi won't cross filesystem borders you need to add the mount paths to your other filesystems if you want to back them up as well. Look at your `/etc/fstab` or the output of the `mount` command to see what's mounted where. For example if you backup your root filesystem `/` and you have a separate boot partition, you also need to specify `/boot` in your list of source directories. the directories are separated with a spacer and it has to be inside quotes. so it could look like this:

```
SRCPART="/dir1 /dir2 /dir3"
```

this will create a backup of `/dir1`, `/dir2` and `/dir3` into the BASEDST directory

BASEDST is the directory that will hold all those timestamped subdirecotires of each backup. You should provide a different BASEDST for each backup-set you have, because RUBI writes a file named

`lastdst` into BASEDST where it stores the name of the last complete path (including the timestamp) where the backup was made to. it will then compare the new files with those and decide if it should link or copy the files. if you have the same BASEDST for two Backup-Sets, this file will be overwritten each time a block completes and so RUBI will always compare the wrong directories…

DSTDIR is the definition of how a timestamped directory name is being put together

OLDDST usually this line can be left as it is in this example. this variable should contain the path to your last successful backup which is usually stored in the lastdst file, hence the cat command in there. so leave this the way it is unless you know better ;)

KEEPG sets how many successful backups should be kept.

KEEPD sets how old a backup should at least be before it can be deleted. so you could say that you want to keep at least 15 successful backups but you want your backups to reach a minimum age of 30 days. if you run a daily backup this would keep 30 backups. if more than half your backups fail, it will keep the last 15 backups which by then might be older than 30 days.

don’t forget to put the doit line after each config block.. that's what starts the backup

this “config” file is simply a shell script and you can put any sort of shell commands in front of and after the doit command. by doing so, you can for example mount your backup target first, then run your backup and then unmount your target again, or you can create a lvm snapshot of your storage and then run your backup from the snapshot if data consistency is important to you.

some thaughts on local backups if you run your backup locally, i.e. you backup your server to a separate harddisk attached to the same server, you should consider adding some extra commands around your doit line, to keep your backup safe:

- `rm -rf /` can erase your backup as well if it is mounted to say /backup because rm by default does not honor filesystem boundaries. in order to avoid losing your backup over a scripting mistake, you should unmount your backup after it's done!
- ransomware: ransomware encrypts all files it can find. the hackers behind it will then ask for money in order to give you the key to decrypt your data again. if you have a backup you don't need to bother paying, you simply restore and you're good. Usually ransomware is a piece of software that looks for all kind of sources for files to encrypt, recently including cloud services and network shares that are mounted to the affected machine. To protect your backup from being encrypted when your data is encrypted it is best to unmount your backup target and to not write its path into your /etc/fstab. also don't share it permanetly via samba to windows machines :)

now to run your backup simply run

```
/opt/rubi/bin/rubi
```

if you want to automatically backup your data, create a cron job for it either by using

```
crontab -e
```

or by editing the main crontab directly

```
vi /etc/crontab
```

NOTE: you need to get ssh autologin to work. so make sure your backup machine's public ssh key is in the authorized_keys file on each target machine, otherwhise your backup won't run automatically as the backup server won't be able to login. Also the user you use to connect to the remote machine should have full access to all files you want to backup of course. if you want to do a full system backup there is no way around using root for this.

addon: backup from or to a qnap
NOTE *the description below was written before Qnap supported Docker containers. Nowadays I recommend to simply setup a docker container and run your linux stuff in there rather than hacking it into the qnap OS as the docker method will remain functional evena after firmware upgrades. I leave this here just for reference.*

a little addon to install rubi on a qnap or to backup from a qnap (only on intel versions, as there seems to be no ipkg repo for the ARM based ones)

install ipkg on the web gui

```
ipgk install cpio
ipkg install nano
```

(if you prefer nano over vi)

```
ipkg install coreutils
```

(onyl necessary if you plan to backup to this qnap, it’s not netcessary on the source i think)

```
ipkg install perl
```

download and copy rubi to your harddisk (/share/HDA_DATA/.rubi) and edit the two .pl files in the lib/ subdirectory. change the path of perl in the first line from /usr/bin/perl to /opt/bin/perl

```
crontab -e
```

to enter the crontab entry

to keep the whole installation running EVEN after a reboot ;) be careful, as far as i understand the box (which i don’t own and don’t have access to the web interface to) it should automatically be mounted. however, on one of two boxes this did not work out, so here is a shell approach to hack it.. but maybe some playing around in the ui might help as well.. just make sure that the optware stuff is autostarted and it should theoretically work :) otherwhise: on the qnap do the follwing...

```
mount /dev/mtdblock5 -t ext2 /tmp/config cd /tmp/config vi autorun.sh
```

(it may as well be mtdblock4 or 6, depending on your qnap box, just try which one will mount)

add the follwing contents to the file:

```
rm -rf /opt ln -sf /share/HDA_DATA/.qpkg/Optware /opt echo "export
PATH=$PATH:/opt/bin:/opt/sbin" >> /etc/profile
```

make sure your harddisk is HDA_DATA, it may also be something else..

make it executable and unmount the partition.

```
chmod +x autorun.sh cd / umount /tmp/config
```

now reboot to see if it works :)