BackupPC

This guide is written when installing BackupPC V4.4.0 which is included in the repository of Debian 11 Bullseye.

Server Setup

Installation

1. Make a backup of `/etc/BackupPC` or `/etc/backuppc`, and `/etc/apache2` if you install it over a previous BackupPC installation
2. For a previous package install, uninstall with the packet manager first
3. If you previously installed BackupPC from tarball, uninstall:
    1. dpkg –remove BackupPC
    2. search for all folders named BackupPC and manually remove them and their contents
4. Install with `apt install backuppc`
5. Install Apache Server through Proxy with user backuppc and group backuppc and enable the VirtualHost.
6. Change the site access password

```
htpasswd /etc/backuppc/htpasswd <user>
```

7. Compare configuration entries of `config.pl` in `/etc/backuppc/` or `/etc/BackupPC/` and correct all file paths. The Debian packet manager install uses different folders to keep the installation files.

Setup encrypted LVM partition

- LUKS - Disk Encryption
- Encrypted partitions/folders with auto-mount

Setup boot configuration

- Remove service backuppc from auto-start at boot to make sure the encrypted volume is mounted before the backuppc service is started

```
sudo systemctl disable backuppc.service
```

- Add the service start command to Services - rc.local after mounting the encrypted data drive with one of the following commands

```
systemctl start backuppc.service
service backuppc start
```

Maintenance

- Delete a backup. If you delete several backups, delete non-filled backups which were taken after a filled backup first.

```
/usr/local/BackupPC/bin/BackupPC_backupDelete -h host -n num
```

  Other Command Line Utilities and BackupPC_backupDelete

Links

- BackupPC Documentation
- How to start, stop, and restart services on Debian

Host Setup

All hosts are setup with rsync through ssh. For Windows 10 hosts I use the Windows Subsystem for Linux which allows to setup a Debian layer to access the host. To backup the localhost we need a small tweak which is explained below.

Main Configuration on BackupPC server

- Modify the following lines in /etc/BackupPC/config.pl (after migration of all hosts from previous versions to V4) and leave all other options on their default settings (reference V4.3.2)

```
$Conf{PoolV3Enabled} = 0;
$Conf{BlackoutPeriods} = [{
  hourBegin => 10,
  hourEnd => 3,
  weekDays => [0, 1, 2, 3, 4, 5, 6],
}];
$Conf{XferMethod} = 'rsync';
```

- Add *target="blank"* to the additional navigation bar links to open in a new browser tab

```
$Conf{CgiNavBarLinks} = [
{
  link  => "?action=view&type=docs\" target=\"blank",
  lname => "Documentation",
}
{
  link  => "https://github.com/backuppc/backuppc/wiki\"
target=\"blank",
  lname => "Wiki",
}
{
```

```
  link  => "https://backuppc.github.io/backuppc\" target=\"blank",
  lname => "Homepage",
}
];
```

Check whether a host is accessible

- Make sure the host is added to the /etc/BackupPC/hosts file. All hosts should have the DHCP flag set to 0, even if they obtain their IP addresses per DHCP.
- First DNS is used to lookup the IP address given the host's name using perl's gethostbyname() function. This should succeed for machines that have dynamic or fixed IP addresses that are known via DNS. You can manually see whether a given host have a DNS entry according to perl's gethostbyname function with this command:

```
perl -e 'print(gethostbyname("myhost") ? "ok\n" : "not found\n");'
```

- If gethostbyname() fails, BackupPC then attempts a NetBios multicast to find the host. Provided your host is configured properly, it should respond to this NetBios multicast request. Specifically, BackupPC runs a command of this form:

  ```
  nmblookup myhost
  ```

  If this fails you will see output like:

  ```
  name_query failed to find name myhost
  ```

  If it is successful you will see output like:

  ```
  local.domain.subnet.ip myhost<00>
  ```

- Depending on your netmask you might need to specify the -B option to nmblookup. For example:

  ```
  nmblookup -B local.domain.subnet.255 myhost
  ```

  If necessary, experiment with the nmblookup command which will return the IP address of the host given its name. Then update $Conf{NmbLookupFindHostCmd} with any necessary options to nmblookup.

- You need to login to the host as backuppc user via SSH at least once manually to accept the host key:

  ```
  ssh <host>
  ```

Configuration on Server for Localhost

- Host access configuration:

```
$Conf{RsyncClientPath} = '/usr/bin/rsync';
$Conf{RsyncSshArgs} = ['-e', '/usr/bin/sudo -p'];
```

- Host share configuration:

```
$Conf{BackupFilesOnly} = ['/etc', '/root', '/home', '/usr', '/var',
'/boot', '/sys'];
```

- Install ssh and rsync:

```
sudo apt install ssh rsync
```

- localhost does not need ssh, this is achieved with a trick, see BackupPC v4 and rsync for localhost
- Allow user backuppc to run rsync as root, add the following line after %sudo:

```
sudo visudo
%sudo ALL=(ALL:ALL) ALL
backuppc ALL=(root:root) NOPASSWD: /usr/bin/rsync
```

Configuration on Server for local hosts

- Host access configuration:

```
$Conf{RsyncClientPath} = 'sudo /usr/bin/rsync';
$Conf{RsyncSshArgs} = ['-e', '$sshPath'];
```

- Host share configuration:

```
$Conf{RsyncShareName} = ['/'];
$Conf{BackupFilesExclude} = ['/proc', '*.vdi'];
```

Configuration on Server for remote hosts

- Host access configuration:

```
$Conf{RsyncClientPath} = 'sudo /usr/bin/rsync';
$Conf{RsyncSshArgs} = ['-e', '$sshPath -p <port>'];
$Conf{PingCmd} = '/usr/bin/sudo /usr/sbin/hping3 $host -p <port> -c 1 -
S';
$Conf{PingMaxMsec} = 300;
$Conf{ClientNameAlias} = 'subdomain.domain.tld';
```

- Host share configuration:

```
$Conf{RsyncShareName} = ['/'];
$Conf{BackupFilesExclude} = ['/proc', '*.vdi'];
```

- To make the host pingable from the backuppc sever you need to replace “ping” with “hping3”

```
apt update
apt install hping3
```

- Read more: How to ping BackupPC hosts behind a NAT router or firewall

Configuration on Server for remote hosts accessible through relais

- Modify ping command in host access configuration (note: $sshPath is not resolved for PingCmd):

```
$Conf{PingCmd} = '/bin/ssh -p <port> -o ConnectTimeout=2 $host echo "1
packets transmitted, 1 received, 0% packet loss, time 0ms"';
```

Configuration for Windows 10 running WSL1

1. Windows 10: Install Windows Subsystem for Linux on Windows 10 hosts, which installs a *Debian* command line layer on top of your Windows 10 installation.
   WSL1 installs a bridged network connection which receives an IP address in the same subnet as Windows.
2. Install ssh and rsync:

```
sudo apt install ssh rsync
```

3. Harden SSH and restart the service.
4. Create user backuppc with a restricted shell, not able to do local but only remote logins (do not expire the user):

```
sudo adduser --shell /bin/rbash --disabled-password backuppc
```

5. Copy the Server's backuppc id_rsa.pub file to the Host's backuppc authorized_keys file. The authorized_keys file should contain further restrictions to prevent e.g. port forwarding, preceed the ssh-rsa string with 'restrict,from=“local.domain.subnet.ip”':

```
restrict,from="local.domain.subnet.ip" ssh-rsa <BASE64-PUBKEY-
REPRESENTATION> backuppc
```

   It should further be owned by root and being read- but not writable by the backuppc user, to prevent removal of SSH restrictions. In older SSH versions you must enter the following instead of restrict:

```
no-pty,no-port-forwarding,no-X11-forwarding,no-agent-
forwarding,from="local.domain.subnet.ip" ssh-rsa <BASE64-PUBKEY-
REPRESENTATION> backuppc
```

6. Allow user backuppc to run rsync as root, add the following line after %sudo:

```
sudo visudo
%sudo ALL=(ALL:ALL) ALL
backuppc ALL=(root:root) NOPASSWD: /usr/bin/rsync
```

7. If it's a new machine you are connecting make sure the public key of the host is added to the server and you have connected manually from server to host to add to the `known_hosts` file

8. Test rsync access as root with the following command from user `backuppc` on the BackupPC Server:

```
ssh <host> -p <port> sudo rsync --version
```

9. For Windows: start the following program as a task after user login in `Task Scheduler`:

```
C:\Users\<user>\AppData\Local\Microsoft\WindowsApps\bash.exe -c "sudo
/etc/init.d/ssh start"
```

- Allowing automatic command execution as root on Linux using SSH
- authorized_keys

Configuration for Windows 11 running WSL2 or Hosts without fixed IP address

1. Windows 11: Enable Windows Subsystem for Linux on Windows 11 hosts as *Windows feature*, then install *Debian* from the Microsoft store.
   WSL2 installs a NAT'ed network connection and assignes a new IP address on every reboot.
2. Install ssh, autossh and rsync:

```
sudo apt install ssh autossh rsync
```

3. Harden SSH and restart the service.
4. Create user backuppc with a restricted shell, not able to do local but only remote logins (do not expire the user):

```
sudo adduser --shell /bin/rbash --disabled-password backuppc
```

5. Copy the Server's backuppc id_rsa.pub file to the Host's backuppc authorized_keys file
6. Allow user backuppc to run rsync as root, add the following line after %sudo:

```
sudo visudo
%sudo ALL=(ALL:ALL) ALL
backuppc ALL=(root:root) NOPASSWD: /usr/bin/rsync
```

7. Establish a ssh connection from host to server with reverse tunneling:

```
autossh -M 0 -f -N -R <reverse_port>:localhost:<ssh_port>
host.domain.tld -p <wan_port>
```

   Note: -M -f are autossh options, all others are ssh options; <reverse_port> is the port you will use to access the host from the server through the reverse ssh tunnel, <ssh_port> is the ssh port on the host, and <wan_port> is the ssh port on the server accessible from the wan.

8. Use `autossh` instead of `ssh` to reestablish the ssh connection if it drops - add the following options in the server's `sshd_config` to have ssh drop the connection if unresponsive

```
ClientAliveInterval 60
ClientAliveCountMax 10
```

9. If it's a new machine you are connecting make sure the public keys are exchanged on host and server and you have connected manually from each side to add both to the `known_hosts` file

10. Test rsync access as root with the following command from user `backuppc` on the BackupPC Server:

```
ssh localhost -p <reverse_port> sudo rsync --version
```

11. For Windows: start the following programs as a task after user login in `Task Scheduler` (sleep 1 second is required to give `autossh` time to drop into background before the shell is terminated):

```
C:\Users\<user>\AppData\Local\Microsoft\WindowsApps\bash.exe -c "sudo
/etc/init.d/ssh start"
C:\Users\<user>\AppData\Local\Microsoft\WindowsApps\bash.exe -c
"autossh -M 0 -f -N -R <reverse_port>:localhost:<ssh_port>
host.domain.tld -p <wan_port> && sleep 1"
```

12. Note that these commands work if WSL was installed as Windows feature (e.g. WSL2, Windows 11) and Debian through the Microsoft Store, otherwise you need to find the location of `bash.exe`. In addition you need to allow <user> to execute `/etc/init.d/ssh` as root without password.

13. The BackupPC host config file needs to backup `localhost` on port <reverse_port>.

Reverse Ports

- Hera: 60022
- Vulcan: 60122
- Cassandra: 60522

Links

- autossh in background
- Remotely access pc behind firewall from my dynamic-ip pc via a static-ip VPS
- Public explanation

Configuration on Synology DSM6 Hosts

Rsync over ssh: "ERROR: module is read only" suddenly appeared

- Synology DSM6 contains a modified rsync client, with which backup of a DSM host is not possible. The easiest way is to install an additional rsync client, which is compiled as static and therefore does not depend on any libraries. You can either follow the following steps, or download the rsync client from this website (rename rsync.gz to rsync). It was compiled on a Debian 10.3 Linux host (amd64) and should run on all Intel based Synology NAS. I confirmed it on DS218+, DS220+, DS716+ II, and DS718+. For other platforms you would need to cross-compile.
- Harden SSH and reboot.
- Create a new user in the DSM web interface. Add him to user groups *users* and *administrators*. Give him permissions for *homes*, but not any other shared folder.
- On another host running linux on amd64 hardware, download sources and compile rsync:

```
cd /home/<user>/temp
wget https://download.samba.org/pub/rsync/rsync-3.1.3.tar.gz
tar xzvf rsync-3.1.3.tar.gz
cd rsync-3.1.3
export CFLAGS=--static
./configure
make
```

- Copy the *rsync* binary to the host's folder /var/services/homes/backuppc/bin
- Copy the Server's backuppc id_rsa.pub file to the Host's backuppc authorized_keys file in folder ~/.ssh. The authorized_keys file should contain further restrictions to prevent e.g. port forwarding, preceed the ssh-rsa string with 'restrict,from=“local.domain.subnet.ip”':

```
restrict,from="local.domain.subnet.ip" ssh-rsa <BASE64-PUBKEY-
REPRESENTATION> backuppc
```

  It should further be owned by root and being read- but not writable by the backuppc user, to prevent removal of SSH restrictions.

- Log in to the DiskStation with SSH, sudo to root, and create the file *backuppc* in folder /etc/sudoers.d/. As there is no *visudo* on DSM you must be extra careful to copy the following lines exactly, otherwise you may render sudo unable to elevate to root:

```
cd /etc/sudoers.d
sudo vim
# Allow backuppc
backuppc ALL=(root:root) NOPASSWD:
/var/services/homes/backuppc/bin/rsync
```

- Rsync over ssh: “ERROR: module is read only” suddenly appeared
- Allowing automatic command execution as root on Linux using SSH
- authorized_keys

Special host configurations

- Linux host share configuration for hosts with more than one file system, for example /home residing on an encrypted share:

```
$Conf{RsyncShareName} = ['/', '/home'];
```

- Windows 10 host share configuration:

```
$Conf{RsyncShareName} = ['/', '/mnt/c', '/mnt/d'];
$Conf{BackupFilesExclude} = [
      '/proc',
      '*.vdi',
      'Windows10Upgrade',
      'System Volume Information',
      '$RECYCLE.BIN',
      '*.mkv',
];
```

- Synology DSM6 host share configuration. **IMPORTANT: you need to exclude any encrypted shared folders containing the encrypted physical files from the backup.**. If you have a shared folder named `documents` then add the line as shown below in the sample config file:

```
$Conf{RsyncShareName} = ['/', '/volume1'];
$Conf{BackupFilesExclude} = [
      '/proc',
      '@eaDir',
      'media',
      '*.mkv',
      '*.vdi',
      # exclude any shared folders here
      '@documents@',
];
```

- Synology DSM6 host access configuration:

```
$Conf{RsyncClientPath} = 'sudo /var/services/homes/backuppc/bin/rsync';
```

Ping commands

- Regular ping (built in)

```
$ /bin/ping <host> -c 1
```

- Host behind firewall

```
$ sudo /usr/sbin/hping3 <host> -p <port> -c 1 -S
```

- Host behind relais host and firewall

```
$ ssh -p <port> -o ConnectTimeout=2 <host> echo "1 packets transmitted,
1 received, 0% packet loss, time 0ms"
```

- No ping (not recommended)

```
$ /bin/echo
```

Alternative setups for Windows hosts

I'm not using any of the methods described in below links. I have collected those during my search for the best way to integrate Windows hosts into BackupPC. If you need to backup a host which does not run Windows 10, or does not provide the Microsoft Windows Subsystem for Linux, then those links might be helpful.

- Cygwin
- Backup Windows System with BackupPC Using Rsyncd
- cygwin-rsyncd: Rsyncd for Cygwin
- Setting Up BackupPC on Windows (rsyncd)
- Setting up a Windows Backuppc Client using Rsync/Cygwin
- Installing SFTP/SSH Server on Windows using OpenSSH
- How to Transfer Files with Rsync over SSH
- How to setup the rsync daemon on Linux
- Linux rsync command

Remove files from backup

- Removing files from backups is not recommended, but if you must here are some hints on how to do it.
- You can delete individual files from below the `pc/HOST` directories (e.g. “*.mp3”)
- The meta data (uid/gid/mtime/mode/type) for all files in a directory is stored in the “attrib” file in each directory. So technically you should delete both the files and the corresponding entries in the attrib file - which would require you writing some perl code using the BackupPC::Attrib module. However, BackupPC will ignore extra attrib entries if the corresponding file doesn't exist, so something like: `find /data/BackupPC/pc/HOST/5 -name “*.mp3” -exec rm -f {} ;` should be sufficient without updating the attrib files. At this point not much disk space will be freed.
- If you backed up the encrypted files on a Synology DSM6.2 before realising that those encrypted shared folders cannot be backed up, remove the entire directory. As an example, for a shared folder named `documents` the file system will contain the folder `documents` which serves the decrypted files real-time, DO NOT touch these. Next to the shared folder `documents` exists another folder `@documents@` which contains the encrypted physical copy of the file, so remove directory and entire content of `@documents@`.
- As for the `pool/cpool`, you cannot easily determine the name of the pool file without computing the MD5 digest - see BackupPC::Lib::File2MD5() and BackupPC::Lib::MD52Path().
- Just waiting overnight until BackupPC_nightly runs. It will remove any pool files that are no longer needed.