Administration Manual for UrBackup Server 1.4.x

Martin Raiber

August 22, 2015

Contents

1 Introduction 3

2 Installation 3
2.1 Server installation . . . 3
2.1.1 Server installation on Windows . . . 3
2.1.2 Server installation on Ubuntu . . . 4
2.1.3 Server installation on Debian . . . 4
2.1.4 Server installation on other GNU/Linux distributions or FreeBSD . . . 4
2.1.5 GNU/Linux server installation hints . . . 5
2.1.6 Operating system independent server installation steps . . . 5
2.2 Client installation . . . 5
2.2.1 Windows client installation . . . 5
2.2.2 Automatic rollout to multiple Windows computers . . . 6
2.2.3 Client installation on GNU/Linux . . . 6

3 Architecture 7
3.1 Server architecture . . . 7
3.2 Client architecture . . . 7

4 Security 8
4.1 Server webinterface rights management . . . 8
4.2 Make webinterface accessible via SSL . . . 9
4.2.1 Apache configuration . . . 9
4.2.2 Lighttp configuration . . . 9
4.3 Client security . . . 9
4.4 Transfer security . . . 10
4.5 Internet mode security . . . 10

5 Client discovery in local area networks 10

6 Backup process 11
6.1 File backup . . . 11
6.2 Image backup . . . 12
6.3 Collision probabilities . . . 12
6.3.1 File backup collision probability . . . 12
6.4 Image backup collision probability . . . 12

7 Internet clients **13**
7.1 Automatically push server configuration to clients . . . 13
7.2 Download a preconfigured client installer . . . 13
7.3 Manually add and configure clients . . . 13
7.4 File transfer over Internet . . . 14

8 Server settings **14**
8.1 Global Server Settings . . . 14
8.1.1 Backup storage path . . . 14
8.1.2 Do not do image backups . . . 14
8.1.3 Do not do file backups . . . 14
8.1.4 Automatically shut down server . . . 14
8.1.5 Download client from update server . . . 15
8.1.6 Autoupdate clients . . . 15
8.1.7 Max number of simultaneous backups . . . 15
8.1.8 Max number of recently active clients . . . 15
8.1.9 Cleanup time window . . . 15
8.1.10 Automatically backup UrBackup database . . . 15
8.1.11 Total max backup speed for local network . . . 15
8.1.12 Global soft file system quota . . . 15
8.2 Mail settings . . . 16
8.2.1 Mail server settings . . . 16
8.2.2 Configure reports . . . 16
8.3 Client specific settings . . . 16
8.3.1 Backup window . . . 18
8.3.2 Excluded files . . . 19
8.3.3 Default directories to backup . . . 19
8.4 Internet settings . . . 19
8.5 Advanced settings . . . 20
8.5.1 Enabling temporary file buffers . . . 20
8.5.2 Transfer modes . . . 20
8.5.3 File hash collection . . . 20
8.5.4 Database cache size . . . 21
8.5.5 File entry cache database . . . 21
8.6 Use symlinks during incremental file backups . . . 21
8.7 Trust client hashes during incremental file backups . . . 22
8.8 End-to-end verification of all file backups . . . 22

9 Miscellaneous **22**
9.1 Manually update UrBackup clients . . . 22
9.2 Logging . . . 22
9.3 Used network ports . . . 23
9.4 Mounting (compressed) VHD files on GNU/Linux . . . 23
9.5 Mounting VHDs as a volume on Windows . . . 23
9.6 Decompress VHD files . . . 23
9.7 Assemble multiple volume VHD images into one disk VHD image . . . 24

10 Storage **24**
10.1 Nightly backup deletion . . . 24
10.2 Emergency cleanup . . . 24
10.3 Cleanup for servers with file backups with lots of files . . . 25
10.4 Cleaning the storage folder of files not known by UrBackup . . . 25
10.5 Archiving . . . 25
10.5.1 Archival window . . . 25

10.6 Suitable file systems 26
10.6.1 Ext4/XFS 26
10.6.2 NTFS 26
10.6.3 btrfs 26
10.6.4 ZFS 26
10.7 Storage setup proposals 27
10.7.1 Mirrored storage with ZFS 27
10.7.2 Btrfs 28

1 Introduction

UrBackup is a client/server backup system. This means there is a server which backs up clients. Accordingly UrBackup is divided into a client and server software. The client software currently runs on Windows on GNU/Linux with only the Windows client being able to perform image backups. The server part of UrBackup runs on Windows, GNU/Linux and FreeBSD.
A lot of effort in UrBackup was made to make setup as easy as possible. If you are happy with the default settings (see section 8) the only thing you need to define on the server side is where backups should be stored. On the clients you only need to say which directories should be backed up. If server and clients are in the same subnet the server will automatically discover the clients and then start backing them up (for details see section 5). This also makes building a decentralized backup strategy very easy, as e.g. one backup server per subnet is responsible for backing up all clients in this subnet. If a computer is moved from one subnet to another this new client is discovered and the server in the new subnet automatically takes over backing it up. If you want to implement something like this, you should also read the section on security (see 4) for details on when a client accepts a server.
Installation instructions are in section 2. The interested administrator should also read up on the general UrBackup architecture (section 3), how the backups are stored and performed (section 6), proposals on which file systems are suited (section 10.6) and take a look at some sample a setup with the next generation file systems ZFS and btrfs at section 10.7.
Since version 1.0 UrBackup can also backup clients over the Internet without VPN (see section 7), and archive backups (see section 10.5).

2 Installation

This section will explain how to install the UrBackup server on various operating systems and how to distribute and install the UrBackup client.

2.1 Server installation

2.1.1 Server installation on Windows

- Download the NSIS (.exe) or MSI installer. You can only use the MSI installer, if you have a 64-bit operating system and at least Windows Vista/2008.
- Install the UrBackup Server.
- Go to the web interface (`http://localhost:55414`) and then go to the settings and configure the folder where UrBackup should store the backup. This folder should have following properties:
  - It should be on a NTFS formatted volume (not ReFS or FAT).
  - There should be enough free space to accommodate the backups
  - Preferably the volume should be dedicated to UrBackup backups

    - The volume should be persistently online while the UrBackup Server instance is running. UrBackup does not support different backup volumes/drives
    - While migration is possible it will be lengthy and difficult. So best plan ahead.
    - You can easily increase the size of the backup storage volume, if you use Windows dynamic volumes or a hardware raid. If you are using a plain volume change it to a dynamic volume before the first backup.
    - Turn on compression for the urbackup folder (in Explorer: Right click and properties). If you are not using a really old computer it should pay off without decreasing the backup speed. Possible exception: If you plan to backup files with more than 50GB or turn off the image compression and plan to backup volumes with more than 50GB you should not turn on compression. NTFS cannot compress files larger than about 50GB.
- Continue with the operating system independent installation steps in section 2.1.6.

2.1.2 Server installation on Ubuntu

Install the UrBackup Server via running following commands:

```
sudo add-apt-repository ppa:uroni/urbackup
sudo apt-get update
sudo apt-get install urbackup-server
```

See section 2.1.5 for further installation hints and 2.1.6 for operating system independent installation steps.

2.1.3 Server installation on Debian

Follow the download link for Debian on `http://urbackup.org/download.html`. Packages are available for Debian stable, testing and unstable for the CPU architectures i686 and AMD64. For Debian stable there is also an ARM package (for e.g. Raspberry PI or Cubieboards).

The Package can be installed via:

```
sudo apt-get update
sudo dpkg -i urbackup-server*.deb
sudo apt-get -f install
```

The Debian stable package may also work on Ubuntu.
See section 2.1.5 for further installation hints and 2.1.6 for operating system independent installation steps.

2.1.4 Server installation on other GNU/Linux distributions or FreeBSD

Baring details on `http://urbackup.org/download.html` you need to compile the server.

- Download the urbackup server source tarball and extract it.
- Install the dependencies. Those are gcc, g++, make, libcrypto++ and libcurl (as development versions).
- Compile and install the server via *./configure*, *make* and *make install*.
- Run the server with *start_urbackup_server*.
- Add */usr/sbin/start_urbackup_server* to your */etc/rc.local* to start the UrBackup server on server start-up.

See section 2.1.5 for further installation hints for GNU/Linux systems and 2.1.6 for operating system independent installation steps.

2.1.5 GNU/Linux server installation hints

Go to the webinterface (`http://localhost:55414`) and configure the backup storage path in the settings. A few hints for the backup storage:

- It should be easily extendable, which can be done by using a hardware raid, the volume manager LVM or the next generation file systems btrfs and ZFS.
- You should compress the file backups. This can be done by using ZFS (`http://zfsonlinux.org/`) or btrfs.
- Prefer btrfs, because UrBackup can put each file backup into a separate sub-volume and is able to do a cheap block based deduplication in incremental file backups. See section 10.7.2 on how to setup a btrfs backup storage. You should set a generously low soft file system quota (see section 8.1.12) if using btrfs, because btrfs currently still has issues in out-of-space situations and may require manual intervention.
- If your priority is stability the currently best option is to use ZFS in connection with the FreeBSD operating system, instead of GNU/Linux and ZFS or btrfs. Look at FreeNAS, for a graphical user interface.

2.1.6 Operating system independent server installation steps

After you have installed the UrBackup server you should perform following steps:

- Go to the user settings and add an admin account. If you do not do this everybody who can access the server will be able to see all backups!
- Setup the mail server by entering the appropriate mail server settings (See section 8.2.1).
- If you want the clients to be able to backup via Internet and not only via local network, configure the public server name or IP of the server in the Internet settings (See section 8.4).
- If you want to get logs of failed backups go the "Logs" screen and configure the reports for you email address.
- Change any other setting according to your usage scenario. See section 8 for descriptions of all available settings.
- Install the clients (see section 2.2).

2.2 Client installation

2.2.1 Windows client installation

If you plan on using the client in the same local network as the server, or the client is in your local network during setup time:

- Download the client from `http://www.urbackup.org`.
- Run the installer.
- Select the backup paths you want to backup on the client.
- The server will automatically find the client and start backups.

If the client is only reachable via Internet:

- Create a new Internet client on the status page.

- Download the client installer on the status page for the Internet client and send it to the new client. Alternatively, create a user for the new client (in the settings) and send the user the username/password. The user can then download the client installer from the server on the status page and install it.
- Select the backup paths you want to backup on the client or configure appropriate default directories to backup on the server (see section 8.3.3).
- The server will automatically start backups once the client is connected.

This is the easiest method to add new internet clients. Other methods to add internet clients are described in section 7.

2.2.2 Automatic rollout to multiple Windows computers

First configure the general default backup paths so that the correct folders get backed for each client (see section 8.3.3). Then install the clients using one of the following methods.

On local network:

Add the MSI client installer as group policy to the domain controller. Alternatively you can use the NSIS (.exe) installer with the switch "/S" to do a silent install and use something like "psexec". The server will automatically find and backup the new clients.

For internet clients:

Adapt the script at `https://urbackup.atlassian.net/wiki/display/US/Download+custom+client+installer+via+Python` to your server URL and create a python executable from the modified script via cx_Freeze (`http://cx-freeze.sourceforge.net/`). Executing the python executable on the client automatically creates a new internet client on the server, downloads a custom client and runs the installer. You could also add the silent install switch ("/S") when starting the downloaded installer such that it needs no user intervention.

2.2.3 Client installation on GNU/Linux

There are currently no client binaries for operating systems other than Windows. You have to compile the UrBackup client yourself:

- Download and install the urbackup client dependencies. Those are gcc, g++, make, libwxgtk (if you want the GUI) and libcrypto++ (or sometimes libcryptopp) as development packages.
- Download the client tarball and extract it.
- Run *./configure*, *make* and *make install*. If you do not need the GUI use *./configure −−enable-headless*.
- If you are configuring an Internet client and have no GUI you can configure the client via command line at this point:

  ```
  echo "internet_server=example.com
  internet_server_port=55415
  computername=hostname
  internet_authkey=foobar
  internet_mode_enabled=true" > /usr/local/var/urbackup/data/settings.cfg
  ```

  See section 7.3 for how to manually add an Internet client and how to get the authentication key.

- Start the client via *start_urbackup_client −−loglevel debug −−no_daemon* to see if everything works. If using an Internet server, connecting may take a while, as it waits for a local backup server first. If you want to avoid the waiting period start the client with *start_urbackup_client −−loglevel debug −−no_daemon −−internetonly*.
- If you use the GUI, start it via running *urbackup_client_gui* and configure the client via accessing the GUI via the tray icon. Note that there are issues with the tray icon not working correctly with the Ubuntu Unity window manager.
- Alternatively, and if you have no GUI, you can configure the client on the server. You can configure the paths by changing the default directories to backup.
- Preferably you would want to add scripts as */etc/urbackup/prefilebackup* and */etc/urbackup/postfilebackup* to respectively create and destroy LVM, ZFS or btrfs snapshots of the folders you are backing up, such that the backups are consistent.
- Add */usr/sbin/start_urbackup_client* to your */etc/rc.local* to start the client on system start-up.

3 Architecture

UrBackup is divided into a server and a client software part. The server is responsible for discovering clients, backing them up, deleting backups if the storage is depleted or too many backups are present, generating statistics and managing client settings. The client is relatively dumb. It listens to server commands which tell it e.g. that a file list should be build or which file the server wants to download. The server also starts a channel on which the clients can request the server to start a backup or to update the client specific settings.

3.1 Server architecture

The server is organized into a core part and an interface. Currently only a webinterface is available. The web interface is accessible via FastCGI (on port 55413) and HTTP (on port 55414). You can use the FastCGI port to make the webinterface accessible via SSL (using e.g. apache web server). For details on that see section 4.2. The server core part consists of several threads with different tasks. One thread discovers new clients, another checks if a client needs to be backed up, while others send pings to clients to see if they are still alive or send them the current backup status. One updates file statistics or deletes old backups. Because there are so many threads UrBackup server profits from modern multi core CPUs (the more cores the better!).

3.2 Client architecture

The client is divided into a core process and an interface process. The interface process displays the tray icon and the dialogues and sends settings and commands to the core client process. The core client process listens on port 35622 UDP for UDP broadcast messages from the server and on receiving one sends a message with its name back to the server. As name the Windows computer name is used. It listens on port 35623 TCP for commands from the client interface process and the server and on port 35621 TCP for file requests from the server. The server establishes a permanent connection to each client on its command port with which the clients can request backups or change their settings. The core client process is responsible for building a list of all files in the directories to be backed up. This list is created in the UrBackup client directory as 'urbackup/ data/ filelist.ub'. To speed up the directory list creation directories to be backed up are constantly watched via the Windows Change Journal. The Windows Change Journal can only be used for whole partitions. Thus the first time a directory on a volume is added the UrBackup core client process reads all the directory entries on the new volume into the client database file in 'urbackup/backup_client.db'. After a volume is successfully indexed the database is constantly updated to be in sync with the file system. Thus if large changes in the volume occur the database gets updated more often. This does not have a big performance penalty as only directories are saved in the database. The updating is done every 10 seconds or if a file list is requested. The server downloads the file list from the client and starts the backup by downloading changed or new files from the build in client file server. The image backup is done using only the command port.

4 Security

4.1 Server webinterface rights management

The server web interface is protected by a pretty standard user system. You can create, manage and delete accounts. Those accounts are only linked loosely to clients by rights management. Be aware that after first installing UrBackup there is no administrator password set and everybody can see all backed up files! If you want to limit access you should immediately go to the account management in the settings and create an administrator account and set its password.
An admin account can do everything including browsing file backups of all clients. The web interface allows one to create a 'limited' account that can only browse backups and view statistics from one client. The more sophisticated rights editor can be used to allow an account to access several clients or to limit some aspects. For example you could setup an account which can do everything except browse backups. Following domains, with which you can limit or expand an account's rights, are currently available:

Domain	Description
browse_backups	Browse and download files from file backups
lastacts	View the last actions (file or image backups) the server did (including backup size and duration)
progress	View the progress of currently running file or image backups
settings	Allows settings to be changed
client_settings	Allows client specific settings to be changed
status	Allows the current status to be viewed (last seen, last file backup and last image backup)
logs	View the logs which were creating during backups
manual_archive	Manually archive file backups
stop_backup	Stop backups for client on the server
piegraph*	View statistics
users*	Get client names
general_settings*	Change general settings (like backup storage path)
mail_settings	Change the mail server settings
usermod*	Create, change and delete users
remove_client*	Remove clients and delete all their backups
start_backup*	Start backups for a client on the server
download_image	Download images of volumes from the server via restore CD

You can set the domains not marked with stars(*) either to one or several client ids (separated by ',') or to 'all' - meaning the account can access all clients. The entries with stars(*) have to be set to 'all' or 'none' and don't allow client ids. In order to be able to view statistics you need to set both 'piegraph' and 'users' to 'all'. There is a special domain 'all' which is a wild card for all domains (this means if you set 'all' to 'all' the account has the right to do everything).

Currently a user needs the "status" right for at least one client, in order for the user to be able to log in.

4.2 Make webinterface accessible via SSL

The server web interface is accessible via FastCGI (on port 55413 TCP). With this you can connect UrBackup with pretty much every modern web server and thus make the web interface accessible via SSL. This section will describe how to do this with apache and lighttp.

4.2.1 Apache configuration

Either add a symlink to the 'www' UrBackup directory or define it as an alias. For the symlink method you need to go to your SSL webroot and then do e.g.:

```
ln -s /var/lib/urbackup/www urbackup
```

Be sure you have set 'Option +FollowSymLinks' in the web server configuration on the directory you link into. From now on it is assumed that urbackup should be accessible via https://hostname/urbackup. Download and install 'libapache2-mod-fastcgi' (this may have another name on other distributions). Add following line to the 'fastcgi.conf':

```
FastCgiExternalServer /var/www/urbackup/x -host 127.0.0.1:55413
```

The path depends of cause on where your web root is and where you want the web interface to be. UrBackup should now be accessible via apache.

4.2.2 Lighttp configuration

Link the urbackup/www directory into the webroot as described in the apache configuration. Add

```
include "conf.d/fastcgi.conf"
```

to your 'lighttp.conf' file. Then add

```
fastcgi.server = (
  "/urbackup/x" =>
  (( "host" => "127.0.0.1",
     "port" => 55413
  ))
)
```

to the 'fastcgi.conf' file.

4.3 Client security

UrBackup Client only answers commands if the server or the interface process supply it with credentials. The server credential is saved in '/var/ lib/ urbackup/ server_ident.key'. If it does not exist the server will randomly generate it the first time it runs. The server identity is also confirmed by private/public key authentication. If not present the server will generate a private and public ECDSA key in 'server_ident.priv' and 'server_ident.pub'.

The client interface credential is generated in the same way and resides in 'pw.txt' and 'pw_change.txt' in the UrBackup directory on the client. To give the client core process interface commands you need the contents of 'pw.txt' or 'pw_change.txt' depending on what the command is:

pw.txt:

- Getting the current status
- Get the incremental file backup interval
- Start backups

- Pause backups

pw_change.txt

- Get the paths which are backed up during file backups
- Change the paths which are backed up during file backups
- Get all settings
- Change all settings
- Get log entries/logs
- Accept a new server

Per default only privileged users can access 'pw_change.txt'. On Windows this leads to a elevation prompt on selecting a menu item which requires the contents of 'pw_change.txt'. If you want to allow the commands without elevation prompt, either disable UAC or change the permissions on 'pw_change.txt' to allow non-privileged users read access. The client core process saves the server credentials from which it accepts commands and which it allows to download files in 'server_idents.txt' - one credential per line. If the server is new enough to have private/public key authentication, the server's public key is also saved in 'server_idents.txt'.

If you want to manually add a server to 'server_idents.txt' you need to remove the preceding '#I' and '#' at the end of the contents of 'server_ident.key'. After installation the 'server_idents.txt' does not exist and the client core process accepts(and adds) the first server it sees (with the public key of the server). After that no other servers with different credentials are accepted and you need to add their credentials either manually, or via clicking on the popup box, once the client has detected the new server. This prevents others from accessing files you want to be backed up in public places.
If you want to have several servers to be able to do backups of a client you have two options. Either you manually supply the server credentials to the client (by copying them into 'server_idents.txt') or you give all servers the same credentials by copying the same 'server_ident.key', 'server_ident.priv' and 'server_ident.pub' to all servers.

4.4 Transfer security

The transfer of data from client to server is unencrypted on the local network allowing eavesdropping attacks to recover contents of the data that is backed up. With this in mind you should use UrBackup only in trusted local networks. Be aware that it is really easy to redirect traffic in local networks, if no action has been taken to prevent that.

4.5 Internet mode security

The Internet mode uses strong authentication and encryption. The three way handshake is done using a shared key and PBKDF2-HMAC using SHA512 with 20000 iterations. The data is encrypted using AES256 in CFB mode. Additionally the local network server authentication via server identity key and ECDSA private/public key authentication is done.

5 Client discovery in local area networks

UrBackup clients should be discovered automatically given that server and client reside in the same sub-network. The client discovery works as follows:
The UrBackup server broadcasts a UDP message every 50 seconds on all adapters into the local subnet of this adapter. (On Linux you can configure which network adapters UrBackup should broadcast on.) On receiving such a broadcast message the client answers back with its fully

qualified domain name. Thus it may take up to 50 seconds until a client is recognized as online. If the client you want to backup is not in the same subnet as the server and broadcast packages therefore do not reach the client you can add its IP or host name manually by clicking "show details" in the settings and then adding an "extra client". The server will then additionally send an UDP message directly to that entered IP or resolved host name allowing switches to forward the message across subnet boundaries. Be aware though that all connections are from server to client. If you have NAT between server and client, you should use “Internet clients" (see section 7). Using “Internet clients” all connections are from client to server.

6 Backup process

This section will show in detail how a backup is performed.

6.1 File backup

- The server detects that the time to the last incremental backup is larger than the interval for incremental backups or the last time to the last full backup is larger than the interval for full backups. Backups can be started on client requests as well.
- The server creates a new directory where it will save the backup. The schema for this directory is YYMMDD-HHMM with YY the year in a format with two decimals. MM the current month. DD the current day. And HHMM the current hour and minute. The directory is created in the backup storage location in a directory which name equals the client name.
- The server requests a file list construction from the client. The client constructs the file list and reports back that it is done.
- The server downloads 'urbackup/data/filelist.ub' from the client. If it is an incremental backup the server compares the new 'filelist.ub' with the last one from the client and calculates the differences.
- The server starts downloading files. If the backup is incremental only new and changed files are downloaded. If the backup is a full one all files are downloaded from the client.
- The server downloads the file into a temporary file. This temporary file is either in the urbackup_tmp_files folder in the backup storage dir, or, if you enabled it in the advanced settings, in the temporary folder. On successfully downloading a file the server calculates its hash and looks if there is another file with the same hash value. If such a file exists they are assumed to be the same and a hard link to the other file is saved and the temporary file deleted. If no such file exists the file is moved to the new backup location. File path and hash value are saved into the server database.
- If the backup is incremental and a file has not changed a hard link to the file in the previous backup is created.
- If the backup is incremental, “Use symlinks during incremental file backups” is enabled and a directory with more than 10 files or folders is unchanged, it is symbolically linked to the same folder in the last backup. Because the last backup will probably be deleted before the current backup, the folder is first moved to a pool directory (“.directory_pool” in the client folder) and then linked from both places. The reference count of the directory is increased/decreased every time another symbolic link is created/removed to that directory.
- If the client goes offline during the backup and the backup is incremental the server continues creating hard links to files in the previous backup but does not try to download files again. The files that could not be downloaded are then not saved into the server side file list. If

the backup is a full one and the client goes offline the backup process is interrupted and the partial file list is saved, which includes all files downloaded up to this point.

- If all files were transferred the server updates the 'current' symbolic link in the client backup storage location to point to the new backup. This only happens if the client did not go offline during the backup.

6.2 Image backup

The server detects that the time to the last full image backup is larger than the interval for full image backups, the time to the last incremental backup is larger than the interval for incremental image backups or the client requested an image backup. The server then opens up a connection to the client command service requesting the image of a volume. The client answers by sending an error code or by sending the image. The image is sent sector for sector with each sector preceded by its position on the hard disk. The client only sends sectors used by the file system. If the backup is incremental the client calculates a hash of 256 kbyte chunks and compares it to the previous image backup. If the hash of the chunk has not changed it does not transfer this chunk to the server, otherwise it does. Per default the server writes the image data directly into a VHD file. If enabled in the advanced configuration the server writes the image data to a temporary file first. The temporary files have a maximum size of 1GB. After this size is exceeded the server continues with a new temporary file. The image data is written to a VHD file in parallel and is located in the client directory in the backup storage location. The VHD file's name is 'Image_<Volume>_<YYMMDD_HHMM>.vhd'.<Volume>being the drive letter of the backed up partition and YY the current year, MM the current month, DD the current day in the month and HHMM the hour and minute the image backup was started.

6.3 Collision probabilities

In this section we will look at the probability that the UrBackup backup system considers data the same, even though it is different. This can be caused by a hash collision (data has the same hash, even though the data is different). If happening, a collision can lead to files being incorrectly linked or blocks in image backups not transferred.

6.3.1 File backup collision probability

UrBackup uses SHA512 to hash the files before file deduplication. In comparison ZFS uses SHA256 for block deduplication. The choice of SHA512 is safer. The Wikipedia page for "Birthday attack" has a probability table for SHA512. According to it one needs $1.6 * 10^{68}$ different files (of same size) to reach a probability of 10^{-18} of a collision. It also states that 10^{-18} is the best case uncorrectable bit error rate of a typical hard disk. To have $1.6 * 10^{68}$ different files of $1KB$ you need $1.4551915 * 10^{56}$ EB of hard disk space. So it is ridiculously more likely that the hard disk returns bad data or the data gets corrupted in RAM, rather than UrBackup linking the wrong files to each other.

6.4 Image backup collision probability

For the blocks in an image backup SHA256 is used. They are 256kbyte in size. The chance of a hash collision with SHA256 is 1:(2^{256}) ($p = \frac{1}{2^{256}}$) for two hashes. In the worst case you have $2TB/256kbyte = 8388608$ different blocks in an incremental image backup. The chance of having a collision in any of the 8388608 blocks (the worst case) is then $1-(1-\frac{1}{2^{256}})^{8388608} \approx 7 * 10^{-71}$. This is again ridiculously low compared to e.g. the probability of 10^{-18} of a typical hard disk having a uncorrectable bit error.

7 Internet clients

UrBackup is able to backup clients over the internet, enabling mixed LAN and Internet backups. This can be useful e.g. for mobile devices which are not used in the LAN all the time, but are connected to the Internet. As it causes additional strain on the backup file system this feature is disabled by default. You need to enable and configure it in the settings and restart your server to use it. The minimum you have to configure is the server name or IP on which the backup server will be available on the Internet. As you probably have a Firewall or Router in between backup server and Internet you also need to forward the configured port (default: 55415) to the backup server.
There are three ways to configure the clients illustrated in the three following sections.

7.1 Automatically push server configuration to clients

If the client is a mobile device it is easiest to let the server push its name and settings to the client. This will happen automatically. The server will also automatically generate a key for each client and push that one to the client as well. This assumes that the local area network is a secure channel. If a client has been compromised you can still change the key on the server and on the client.

7.2 Download a preconfigured client installer

If you enabled “Download client from update server” in the server settings, or manually downloaded a client update and put it in the appropriate folder (see section 9.1), users can download a preconfigured client installer from the server on the “Status” page. The installer includes the server IP/domain name, the port it is listening on, the clientname and the authkey. This allows the installed client to automatically connect to the server the installer was downloaded from.

7.3 Manually add and configure clients

UrBackup also allows manually adding clients and manually configuring the shared key. To add such a client following steps are necessary:

1. Go to the “Status” screen as administrator
2. Under “Internet clients” enter the name of the Laptop/PC you want to add. This must be the fully qualified computer name (i.e. the one you see in the advanced system settings) or the computer name configured on the client.
3. After pressing “add” there will be a new client in the “Status” screen. Go to settings and select that client there.
4. In the Internet settings enter an authentication key for that client. The key acts just like every normal password and should therefore be sufficiently complex. Having a different key for every client makes revoking compromised keys easier, but is not a requirement.
5. On the client go to the settings and enter the same key there in the internet settings. Also enter the public IP or name of your backup server and the port it is reachable at.
6. The server and client should now connect to each other. If it does not work the client shows what went wrong in the “Status” window.

7.4 File transfer over Internet

If a client is connected via Internet UrBackup automatically uses a bandwidth saving file transfer mode. This mode only transfers changed blocks of files and should therefore conserve bandwidth on files which are not changed completely, such as database files, virtual hard disks etc.. This comes at a cost: UrBackup has to save hashes of every file. Those hashes are saved the folder ".hashes". They are only saved if the Internet mode is enabled. If the hashes of a file are not present e.g. because Internet mode was just enabled, they are created from the files during the backup and may therefore slow down the backup process.

8 Server settings

The UrBackup Server allows the administrator to change several settings. There are some global settings which only affect the server and some settings which affect the client and server. For those settings the administrator can set defaults or override the client's settings.

8.1 Global Server Settings

The global server settings affect only the server and can be changed by any user with "general_settings" rights.

8.1.1 Backup storage path

The backup storage path is where all backup data is saved. To function properly all of this directories' content must lie on the same file system (otherwise hard links cannot be created). How much space is available on this file system for UrBackup determines partly how many backups can be made and when UrBackup starts deleting old backups. Default: "".

8.1.2 Do not do image backups

If checked the server will not do image backups at all. Default: Not checked.

8.1.3 Do not do file backups

If checked the server does no file backups. Default: Not checked.

8.1.4 Automatically shut down server

If you check this UrBackup will try to shut down the server if it has been idle for some time. This also causes too old backups to be deleted when UrBackup is started up instead of in a nightly job. In the Windows server version this works without additional work as the UrBackup server process runs as a SYSTEM user, which can shut down the machine. On Linux the UrBackup server runs as a limited user which normally does not have the right to shut down the machine. UrBackup instead creates the file '/var/lib/urbackup/shutdown_now', which you can check for existence in a cron script e.g.:

```
if test -e /var/lib/urbackup/shutdown_now
then
shutdown -h +10
fi
```

Default: Not checked.

8.1.5 Download client from update server

If this is checked the server will automatically look for new UrBackup client versions. If there is a new version it will download it from the Internet. The download is protected by a digital signature. Default: Checked.

8.1.6 Autoupdate clients

If this is checked the server will send new versions automatically to its clients. The UrBackup client interface will ask the user to install the new client version. If you check silent autoupdate (see Section 8.1.6) it will update in the background. The installer is protected by a digital signature so malfeasance is not possible. Default: Checked.

8.1.7 Max number of simultaneous backups

This option limits the number of file and image backups the server will start simultaneously. You can decrease or increase this number to balance server load. A large number of simultaneous backups may increase the time needed for backups. The number of possible simultaneous backups is virtually unlimited. Default: 10.

8.1.8 Max number of recently active clients

This option limits the number of clients the server accepts. An active client is a client the server has seen in the last two month. If you have multiple servers in a network you can use this option to balance their load and storage usage. Default: 100.

8.1.9 Cleanup time window

UrBackup will do its clean up during this time. This is when old backups and clients are deleted. You can specify the weekday and the hour as intervals. The syntax is the same as for the backup window. Thus please see section 8.3.1 for details on how to specify such time windows. The default value is 1-7/3-4 which means that the cleanup will be started on each day (1-Monday - 7-Sunday) between 3 am and 4 am.

8.1.10 Automatically backup UrBackup database

If checked UrBackup will save a backup of its internal database in a subdirectory called 'urbackup' in the backup storage path. This backup is done daily within the clean up time window.

8.1.11 Total max backup speed for local network

You can limit the total bandwidth usage of the server in the local network with this setting. All connections between server and client are then throttled to remain under the configured speed limit. This is useful if you do not want the backup server to saturate your local network.

8.1.12 Global soft file system quota

During cleanups UrBackup will look at the used space of the file system the backup folder is on. If the used space is higher than the global soft file system quota UrBackup will delete old backups, if possible, till the used space is below the quota. Be aware that not only UrBackup's files count against the quota, but other files as well. A quota that only takes into account UrBackup's files is planned. You can specify the quota via a percentage of total space, or by a size. For example let the size of the Backup device be 1 Tera-byte: If you set the global file system quota to "90%", UrBackup will delete old backups as soon as more than about 900 Giga-bytes of the available space is used. You could also directly set the quota to 900 Giga-bytes by setting it to "900G". Other units are possible, e.g. "900000M" or "1T".

8.2 Mail settings

8.2.1 Mail server settings

If you want the UrBackup server to send mail reports a mail server should be configured in the 'Mail' settings page. The specific settings and their description are:

Settings	Description	Example
Mail server name	Domain name or IP address of mail server	mail.example.com
Mail server port	Port of SMTP service. Most of the time 25 or 587	587
Mail server username	Username if SMTP server requires one	test@example.com
Mail server password	Password for user name if SMTP server requires credentials	password1
Sender E-Mail Address	E-Mail address UrBackup's mail reports will come from	urbackup@example.com
Send mails only with SSL/TLS	Only send mails if a secure connection to the mail server can be established (protects password)	
Check SSL/TLS certificate	Check if the server certificate is valid and only send mail if it is	
Server admin mail address	Address for fatal errors (such as if an emergency cleanup fails or other fatal errors occur)	

To test whether the entered settings work one can specify an email address to which UrBackup will then send a test mail.

8.2.2 Configure reports

To specify which activities with which errors should be sent via mail you have to go to the 'Logs' page. There on the bottom is a section called 'Reports'. There you can say to which email addresses reports should be sent(e.g. user1@example.com;user2@example.com) and if UrBackup should only send reports about backups that failed/succeeded and contained a log message of a certain level.
If you select the log level 'Info' and 'All' a report about every backup will be sent, because every backup causes at least one info level log message. If you select 'Warning' or 'Error' backups without incidents will not get sent to you.

Every web interface user can configure these values differently. UrBackup only sends reports of client backups to the user supplied address if the user has the 'logs' permission for the specific client. Thus if you want to send reports about one client to a specific email address you have to create a user for this client, login as that user and configure the reporting for that user. The user 'admin' can access logs of all clients and thus also gets reports about all clients.

8.3 Client specific settings

Settings	Description	Default value
Interval for incremental file backups	The server will start incremental file backups in such intervals.	5h

Interval for full file backups	The server will start full file backups in such intervals.	30 days
Interval for incremental image backups	The server will start incremental image backups in such intervals.	7 days
Interval for full image backups	The server will start full image backups in such intervals.	30 days
Maximal number of incremental file backups	Maximal number of incremental file backups for this client. If the number of incremental file backups exceeds this number the server will start deleting old incremental file backups.	100
Minimal number of incremental file backups	Minimal number of incremental file backups for this client. If the server ran out of backup storage space the server can delete incremental file backups until this minimal number is reached. If deleting a backup would cause the number of incremental file backups to be lower than this number it aborts with an error message.	40
Maximal number of full file backups	Maximal number of full file backups for this client. If the number of full file backups exceeds this number the server will start deleting old full file backups.	10
Minimal number of full file backups	Minimal number of full file backups for this client. If the server ran out of backup storage space the server can delete full file backups until this minimal number is reached. If deleting a backup would cause the number of full file backups to be lower than this number it aborts with an error message.	2
Maximal number of incremental image backups	Maximal number of incremental image backups for this client. If the number of incremental image backups exceeds this number the server will start deleting old incremental image backups.	30
Minimal number of incremental image backups	Minimal number of incremental image backups for this client. If the server ran out of backup storage space the server can delete incremental image backups until this minimal number is reached. If deleting a backup would cause the number of incremental image backups to be lower than this number it aborts with an error message.	4
Maximal number of full image backups	Maximal number of full image backups for this client. If the number of full image backups exceeds this number the server will start deleting old full image backups.	5
Minimal number of full image backups	Minimal number of full image backups for this client. If the server ran out of backup storage space the server can delete full image backups until this minimal number is reached. If deleting a backup would cause the number of full image backups to be lower than this number it aborts with an error message.	2
Delay after system start up	The server will wait for this number of minutes after discovering a new client before starting any backup	0 min
Backup window	The server will only start backing up clients within this window. See section 8.3.1 for details.	1-7/0-24
Max backup speed for local network	The server will throttle the connections to the client to remain within this speed window.	-

Perform auto-updates silently	If this is selected automatic updates will be performed on the client without asking the user	Unchecked
Soft client quota	During the nightly cleanup UrBackup will remove backups of this client if there are more backups than the minimal number of file/image backups until this quota is met. The quota can be in percent (e.g. 20%) or absolute (e.g. 1500G, 2000M).	""
Excluded files	Allows you to define which files should be excluded from backups. See section 8.3.2 for details	""
Default directories to backup	Default directories which are backed up. See section 8.3.3 for details	""
Volumes to backup	Specifies of which volumes an image backup is done. Separate different drive letters by a semicolon or comma. E.g. 'C;D'	C
Allow client-side changing of the directories to backup	Allow client(s) to change the directories of which a file backup is done	Checked
Allow client-side starting of incremental/full file backups	Allow the client(s) to start a file backup	Checked
Allow client-side starting of incremental/full image backups	Allow the client(s) to start an image backup	Checked
Allow client-side viewing of backup logs	Allow the client(s) to view the logs	Checked
Allow client-side pausing of backups	Allow the client(s) to pause backups	Checked
Allow client-side changing of settings	If this option is checked the clients can change their client specific settings via the client interface. If you do not check this the server settings always override the clients' settings.	Checked

8.3.1 Backup window

The server will only start backing up clients within the backup windows. The clients can always start backups on their own, even outside the backup windows. If a backup is started it runs till it is finished and does not stop if the backup process does not complete within the backup window.
A few examples for the backup window:
1-7/0-24: Allow backups on every day of the week on every hour.
Mon-Sun/0-24: An equivalent notation of the above
Mon-Fri/8:00-9:00, 19:30-20:30;Sat,Sun/0-24: On weekdays backup between 8 and 9 and between 19:30 and 20:30. On Saturday and Sunday the whole time.

As one can see a number can denote a day of the week (1-Monday, 2-Tuesday, 3-Wednesday, 4-Thursday, 5-Friday, 6-Saturday, 7-Sunday). You can also use the abbreviations of the days (Mon, Tues, Wed, Thurs, Fri, Sat, Sun). The times can either consist of only full hours or of hours with minutes. The hours are on the 24 hour clock. You can set multiple days and times per window definition, separated per ",". You can also set multiple window definitions. Separate them with ";".

8.3.2 Excluded files

You can exclude files with wild card matching. For example if you want to exclude all MP3s and movie files enter something like this:

```
*.mp3;*.avi;*.mkv;*.mp4;*.mpg;*.mpeg
```

If you want to exclude a directory e.g. Temp you can do it like this:

```
*/Temp/*
```

You can also give the full local name

```
C:\Users\User\AppData\Local\Temp\*
```

or the name you gave the location e.g.

```
C_\Users\User\AppData\Local\Temp
```

Rules are separated by a semicolon (";")

8.3.3 Default directories to backup

Enter the different locations separated by a semicolon (";") e.g.

```
C:\Users;C:\Program Files
```

If you want to give the backup locations a different name you can add one with the pipe symbol ("|") e.g.:

```
C:\Users|User files;C:\Program Files|Programs
```

gives the "Users" directory the name "User files" and the "Program files" directory the name "Programs".

Those locations are only the default locations. Even if you check "Separate settings for this client" and disable "Allow client to change settings", once the client modified the paths, changes in this field are not used by the client any more.

8.4 Internet settings

Settings	Description	Default value
Internet server name/IP	The IP or name the clients can reach the server at over the internet	""
Internet server port	The port the server will listen for new clients on	55415
Do image backups over internet	If checked the server will allow image backups for this client/the clients	Not checked
Do full file backups over internet	If checked the server will allow full file backups for this client/the clients	Not checked
Max backup speed for internet connection	The maximal backup speed for the Internet client. Setting this correctly can help avoid saturating the Internet connection of a client	-
Total max backup speed for internet connection	The total accumulative backup speed for all Internet clients. This can help avoid saturating the server's Internet connection	-
Encrypted transfer	If checked all data between server and clients is encrypted	Checked
Compressed transfer	If checked all data between server and clients is compressed	Checked
Calculate file-hashes on the client	If checked the client calculates hashes for each file before the backups (only hashes of changed files are calculated). The file then does not have to be transferred if another client already transferred the same file	Not checked

8.5 Advanced settings

In this section you will find global server settings which you only have to change for heavy or custom workloads. Most settings will need a server restart to come into effect.

8.5.1 Enabling temporary file buffers

Earlier versions of UrBackup always saved incoming data from clients first to temporary files and then copied it to the final destination (if the data is new) – the rationale being, that the final destination may be slow and you want to get the data from the client as fast as possible.
With UrBackup 1.1 this default behaviour was changed to directly copy the data to the final backup storage. The two settings allow you to re-enable the old behaviour, e.g., because your backup storage is slow because it is deduplicated. If you re-enable it make sure you have at least 1GB of space for each client, and at least as much space as the biggest file you are going to backup times the number of clients, on your temporary storage. You can change the temporary storage directory via the environment variable *TMPDIR* on GNU/Linux and in the server settings on Windows.

8.5.2 Transfer modes

UrBackup has different transfer modes for files and images. Those are:

- *raw.* Transfer the data as 'raw' as possible. This is the fastest transfer mode and uses the least amount of CPU cycles on server and client.
- *hashed.* Protects the transferred data from bit errors by hashing the data during the transfer. This uses CPU cycles on the client and the server.
  UrBackup uses TCP/IP to transfer the images and files. TCP/IP implements its own bit error detection mechanism (CRC32). If the network induces a lot of bit errors and if a lot of data is transferred (>2TB), however, the bit error detection mechanism of TCP/IP is not enough to detect all occurring errors. The 'hashed' transfer mode adds an additional layer of protection to make bit errors less probable.
- *Block differences - hashed.* Only available for file backups (as it is automatically done for images). Blocks of the transferred files are compared using CRC32 and MD5 hash functions. Only blocks which have changed are sent over the network. In cases where only some blocks of a file change, this reduces the amount of transferred data. It also causes more messages to be sent between server and client and uses CPU cycles, which is why it is only enabled for Internet clients per default.

8.5.3 File hash collection

During full file backups or for new files in incremental backup a database entry, which maps the files hash to its storage path is created. This entry allows succeeding same files to be linked to the file encountered first, without storing it twice. To speed up this process, updates to the database are batched, i.e., file entries are first stored in a temporary table, and later moved over to the real database. As the temporary table is only visible to the thread currently running for one client, the longer this copy process is delayed the more it becomes possible that another client does not link to an already existing file, because it cannot find it in the database. With the file hash collection parameters you can influence when the copying from the temporary table to the shared database happens:

- *File hash collection amount.* After x amount of file entries are in the temporary table the contents of the temporary table are copied to the shared (final) table.
- *File hash collection timeout.* After not having created any file entries in the temporary table for x milliseconds the contents of the temporary table are copied to the shared (final) table.

8.5.4 Database cache size

UrBackup uses a per thread database cache. With the database cache size parameters you can influence the size of the database caches of some of the threads.

- *File hash collection database cachesize.* The size of the database cache for the thread doing the file lookups and linking. Increase this cache size if you have slow file backups (stalled at 100% and slowly decreasing queue). Make sure you have enough RAM, as UrBackup will use this amount of Megabytes times the number of simultaneous file backups.
- *Update stats database cachesize.* Size of the database cache for the thread which updates the statistics (i.e., which client uses how much space). There will only ever be one such thread and it will not be run while other backups are running, so you can set this to a relatively high value.

8.5.5 File entry cache database

Optionally UrBackup can use a cache database for file entries storing the file hash to files mapping used to hard link to files, if they already exist on the backup storage. Enabling the file entry cache may be advantageous if there are a lot of files on some clients or if you do full backups often. The file entry cache is enabled by selecting the file entry cache type. The cache is created when the server is restarted. Cache creation may take some time.

- *Cache database type for file entries.* By selecting something other than "None" the file entry cache is enabled. SQLite probably gives better performance when the underlying storage is slow and if the file entries do not fit into memory. LMDB should only be used on 64bit systems. When in doubt select "SQLite".
- *Cache database size for file entries.* Only relevant if LMDB was selected. This is the maximum database size of the LMDB database. LMDB creates a memory mapped file of this size, so UrBackup uses this much virtual memory. On Windows LMDB also creates a sparse file of this size (which can confuse backup programs).
- *Suspend index limit.* When the file entry cache is enabled, file entries are put into a temporary table and copied to the main table during a statistics update. If there are more file entries in the temporary table then the suspend index limit, indexes on the main table are destroyed, the file entries moved over from the temporary table and then rebuild. This may be faster for a large number of file entries then moving the file entries with indexes enabled.

8.6 Use symlinks during incremental file backups

If enabled UrBackup will use symbolic links to link unchanged directories with more than 10 directory/files. This will greatly improve the incremental file backup speed, if only few directories are changed, as less hard links have to be created and hard linking operations are expensive on some file systems such as e.g. NTFS and spinning disks.

The disadvantages are:

- UrBackup needs to keep track of the number of times a directory is used. For hard links the operation system does this. UrBackup may be slower and less reliable doing this.
- If the backup storage folder is moved all symbolic links are invalidated. To restore the symbolic links please run on Windows

  ```
  C:\Program Files\UrBackupServer\remove_unkown.bat
  ```

and on Linux while the server is not running:

```
start_urbackup_server --remove_unknown
```

Per default symbolic links are used. If the btrfs mode is used this option has no effect.

8.7 Trust client hashes during incremental file backups

To not loose any file hashes for internet client, the UrBackup server adds file entries (i.e. a file path and the sha512 hash of a file) each "minimal number of incremental file backups" times. If the sha512 hashes are calculated on the client side UrBackup is able to use the client provided file hash. Should it be wrong, however, a entry will be in the database mapping a file hash to wrong content. The entry may then be used to link files for other clients, causing file backups to be corrupted.

If not checked the UrBackup server will not trust the client provided sha512 hash sums and calculate them itself.

Per default the UrBackup server trusts the client to provide correct sha512 hashes.

8.8 End-to-end verification of all file backups

This is a setting for debugging purposes or for the paranoid. If end-to-end verification is enabled UrBackup clients >=1.3 will create file hashes for every file for every file backup reading every file that is to be backed up. At the end of the backup process the hashes of the files stored on the server are compared to the hashes calculated on the client. If hashes differ the backup fails and an email is sent to the server admin.

9 Miscellaneous

9.1 Manually update UrBackup clients

You should test UrBackup clients before using them on the clients. This means UrBackup should not automatically download the newest client version from the Internet and install it. This means disabling the autoupdate described in Section 8.1.6. You can still centrally update the client from the server if you disabled autoupdate. Go to `https://sourceforge.net/projects/urbackup/files/Client/` and download all files in the *update* folder of the current client to */var/urbackup* on Linux and *C:\\Program Files\UrBackupServer\urbackup* per default on Windows. UrBackup will then push the new version to the clients once they reconnect. If you checked silent autoupdates, the new version will be installed silently on the clients, otherwise there will be a popup asking the user to install the new version.

9.2 Logging

UrBackup generally logs all backup related things into several log facilities. Each log message has a certain severity, namely *error*, *warning*, *info* or *debug*. Each log output can be filtered by this severity, such that e.g. only errors are shown. Both server and client have separate logs. During a backup process the UrBackup server tries to log everything which belongs to a certain backup in a client specific logs and at the end sends this log to the client. Those are the logs you see on the client interface. The same logs can also be viewed via the web interface in the "Logs" section. One can also send them per mail as described in subsection 8.2.2.
Everything which cannot be accredited to a certain client or which would cause too much log traffic is logged in a general log file. On Linux this is */var/log/urbackup.log* on Windows *C:\\Progam files\UrBackupServer\urbackup.log* for the server per default. The client has as defaults */var/log/urbackup_client.log* and *C:\\Progam files\UrBackup\debug.log*. Per default

those files only contain log messages with severity *warning* or higher. In Windows there is a *args.txt* in the same directory as the log file. Change *warn* here to *debug*, *info* or *error* to get a different set of log messages. You need to restart the server for this change to come into effect. On Linux this depends on the distribution. On Debian one changes the setting in */etc/default/urbackup_srv*.

9.3 Used network ports

The Server binds to following default ports:

Port	**Usage**	**Incoming/Outgoing**
55413	FastCGI for web interface	Incoming
55414	HTTP web interface	Incoming
55415	Internet clients	Incoming
35623	UDP broadcasts for discovery	Outgoing

The Client binds to following default ports (all incoming):

Port	**Usage**
35621	Sending files during file backups (file server)
35622	UDP broadcasts for discovery
35623	Commands and image backups

9.4 Mounting (compressed) VHD files on GNU/Linux

If you compiled UrBackup with fuse (file system in user space) support or installed the Debian/Ubuntu packages the UrBackup server can mount VHD(Z) files directly. You compile UrBackup with fuse support by configuring:

```
./configure --with-mountvhd
```

You will be able to mount a VHD(Z) file e.g. with

```
start_urbackup_server --mountvhd /media/backup/urbackup/testclient/\
Image_C_140420-1956.vhdz --mountpoint /media/testclient_C
```

All files in the backed up "C" volume will then be available read-only in */media/testclient_C*. Unmount the mounts created by UrBackup (see output of *mount*), to stop the background process.

9.5 Mounting VHDs as a volume on Windows

Starting from Vista Windows can mount VHD files directly. If the VHD files are compressed, they need to be decompressed first (see next section 9.6). Then go to system settings->Manage->Computer management->Disk management->Other Actions->Add Virtual Hard Disk. Mount the VHD file read only. The image will appear as another drive in the explorer. It may not work if the VHD file is on a network drive.

9.6 Decompress VHD files

If you want to mount the VHD files on Windows and they are compressed (file extension is VHDZ), you need to decompress them first. Use *C:\Program Files\UrBackupServer\uncompress_image.bat* for that. Calling the batch file without parameters will open a file selection screen where you can select the VHDZ file to be decompressed. A temporary inflated copy is created and renamed in-place once the decompression is done. If the image is incremental the parent-VHD is automatically decompressed as well. If you want to prevent this please use the method decribed in section 9.7 to build a separate uncompressed image. All the image files will still have the VHDZ extension, as otherwise it would have to change database entries, but the files will not be compressed anymore. On Linux the same thing can be done with *start_urbackup_server −−decompress [filename]*.

9.7 Assemble multiple volume VHD images into one disk VHD image

UrBackup stores each volume of an image backup separately. If you want to boot an image backup, without using the restore CD, as an virtual machine you have to re-assemble multiple volumes into one disk VHD image. On Windows this can be done by running *C:\Program Files\UrBackupServer\assemble_disk_image.bat*. In a first step it will ask for the VHD images to assemble. Select e.g. Image_C_XXXXX.vhd and Image_SYSVOL_XXXXX.vhd. The source images can also be incremental or compressed. Then it will ask where the output VHD disk image should be saved. After that it will write the master boot record from Image_C_XXXXX.vhd.mbr and the contents of the volumes into the output disk image at the appropriate offsets.
On Linux the same thing can be done with

```
start_urbackup_server \
--assemble "/full/path/Image_C_XXXXX.vhd;/full/path/Image_SYSVOL_XXXXX.vhd" \
--assemble_output "full_disk.vhd"
```

This tool can also be used to decompress images without decompressing their parents by selecting a single VHD file as input.

10 Storage

The UrBackup server storage system is designed in a way that it is able to save as much backups as possible and thus uses up as much space on the storage partition as possible. With that in mind it is best practice to use a separate file system for the backup storage or to set a quota for the 'urbackup' user. Some file systems behave badly if they are next to fully occupied (fragmentation and bad performance). With such file systems you should always limit the quota UrBackup can use up to say 95% of all the available space. You can also setup a soft quota within UrBackup (see section 8.1.12) which causes UrBackup to delete backups to stay within this quota, if possible.

10.1 Nightly backup deletion

UrBackup automatically deletes old file and image backups between 3am and 5am. Backups are deleted when a client has more incremental/full file/image backups then the configured maximum number of incremental/full file/image backups. Backups are deleted until the number of backups is within these limits again.
If the administrator has turned automatic shut-down on, this clean up process is started on server start up instead (as the server is most likely off during the night). Deleting backups and the succeeding updating of statistics can have a huge impact on system performance.

During nightly backup deletion UrBackup also tries to enforce the global and client specific soft quotas. It is only able to delete backups if a client has already more backups than the configured minimal number of incremental/full file/image backups.

10.2 Emergency cleanup

If the server runs out of storage space during a backup it deletes backups until enough space is available again. Images are favoured over file backups and the oldest backups are deleted first. Backups are only deleted if there are at least the configured minimal number of incremental/full file/image backups other file/image backups in storage for the client owning the backup. If no such backup is found UrBackup cancels the current backup with a fatal error. Administrators should monitor storage space and add storage or configure the minimal number of incremental/full file/image backups to be lower if such an error occurs.

10.3 Cleanup for servers with file backups with lots of files

UrBackup's database is in a mode which enables high concurrency. Since the cleanup procedure can sometimes be bottlenecked by the database it may be advisable to switch the database into a mode which allows less concurrency but is fast for some operations for the cleanup procedure. This is not possible while UrBackup is running, so you should tweak the backup window such that you can be sure there are no backups running at some point. Then you can stop the server run the cleanup separately by calling

```
start_urbackup_server --cleanup x
```

on GNU/Linux or on Windows:

```
cleanup.bat x
```

Where x is the percent of space to free on the backup storage or the number of Bytes/ Megabytes/ Gigabytes e.g. "20G" or "10%". If it should only delete old backups use "0%".

10.4 Cleaning the storage folder of files not known by UrBackup

Sometimes, e.g., by using a database backup, there are backups in the storage directory which UrBackup does not know about, i.e., there are no entries for those backups in the database. In those cases the command

```
start_urbackup_server --remove_unknown
```

on GNU/Linux or on Windows:

```
remove_unknown.bat
```

removes files and folders in the urbackup storage directory which are not present in the UrBackup database.

10.5 Archiving

UrBackup has the ability to automatically archive file backups. Archived file backups cannot be deleted by the nightly or emergency clean up – only when they are not archived any more. You can setup archival under Settings->Archival for all or specific clients. When an archival is due and the the server is currently in a archival window (See 10.5.1) the last file backup of the selected type will be archived for the selected amount of time. After that time it will be automatically not archived any more. You can see the archived backups in the "Backups" section. If a backup is archived for only a limited amount of time there will be a time symbol next to the check mark. Hovering over that time symbol will tell you how long that file backup will remain archived.

10.5.1 Archival window

The archival window allows you to archive backups at very specific times. The format is very similar to *crontab*. The fields are the same except that there are no minutes:

Field	Allowed values	Remark
Hour	0-23	
Day of month	1-31	
Month	1-12	No names allowed
Day of week	0-7	0 and 7 are Sunday

To archive a file backup on the first Friday of every month we would then set "Archive every" to something like 27 days. After entering the time we want the backups archived for we would then add

```
*;*;*;5
```

as window (hour;day of month;month;day of week). To archive a backup every Friday we would set “Archive every” to a value greater than one day but less than 7 days. This works because both conditions have to apply: The time since the last backup archival must be greater than “Archive every” and the server must be currently in the archive window.
Other examples are easier. To archive a backup on the first of every month the window would be

```
*;1;*;*
```

and “Archive every” something like 2-27 days.
One can add several values for every field by separating them via a comma such that

```
*;*;*;3,5
```

and “Archive every” one day would archive a backup on Wednesday and Friday. Other advanced features found in *crontab* are not present.

10.6 Suitable file systems

Because UrBackup has the option to save all incoming data to temporary files first (see Section 8.5.1) and then copies them to the final location in parallel backup performance will still be good even if the backup storage space is slow. This means you can use a fully featured file system with compression and de-duplication without that much performance penalty. At the worst the server writes away an image backup over the night (having already saved the image’s contents into temporary files during the day). This section will show which file systems are suited for UrBackup.

10.6.1 Ext4/XFS

Ext4 and XFS, are both available in Linux and can handle big files, which is needed for storing image backups. They do not have compression or de duplication though. Compression can be achieved by using a fuse file system on top of them such as fusecompress. There are some block-level de-duplication fuse layers as well, but I would advise against them as they do not seem very stable. You will have to use the kernel user/group level quota support to limit the UrBackup storage usage.

10.6.2 NTFS

NTFS is pretty much the only option you have if you run the UrBackup server under Windows. It supports large files and compression as well as hard links and as such is even more suited for UrBackup than the standard Linux file systems XFS and Ext4.

10.6.3 btrfs

Btrfs is a next generation Linux file system that is still under development and as such it is probably not suited for production use yet. It supports compression and offline block-level deduplication. UrBackup has a special snapshotting backup mode which makes incremental backups and deleting file backups much faster with btrfs. With btrfs UrBackup also does a cheap (in terms of CPU und memory requirements) block-level deduplication on incremental file backups. See 10.7.2 for details.

10.6.4 ZFS

ZFS is a file system originating from Solaris. It is available as a fuse module for Linux (zfs-fuse) and as a kernel module (ZFSOnLinux). ZFSOnLinux as of 0.6.2 still has memory issues with UrBackup/rsync style workloads, so you should carefully evaluate if it is stable enough. There were licensing issues which prevented prior porting of ZFS to Linux. If you want the most performance and stability an option would be using a BSD (e.g. FreeBSD, FreeNAS) or Debian/kFreeBSD. The ZFS in the BSD kernels is stable. The upstream Solaris ZFS has been available for some time and as such should be very stable as well. ZFS has some pretty neat features like compression, block-level de-duplication, snapshots and build in raid support that make it well suited for backup storage. How to build a UrBackup server with ZFS is described in detail in section 10.7.1.

10.7 Storage setup proposals

In this section a sample storage setup with ZFS is shown which allows off-site backups via Internet or via tape like manual off-site storage and a storage setup using the Linux file system btrfs using the btrfs snapshot mechanism to speed up file backup creation and destruction and to save the file backups more efficiently.

10.7.1 Mirrored storage with ZFS

Note: It is assumed that UrBackup runs on a UNIX like system such as Linux or BSD. An example would be Debian/Linux or Debian/kFreeBSD with the kFreeBSD kernel being preferred, because of its better ZFS performance. We will use all ZFS features such as compression, de-duplication and snapshots. It is assumed that the server has two hard drives (sdb,sdc) dedicated to backups and a hot swappable hard drive slot (sdd). It is assumed there is a caching device to speed up de-duplication as well in /dev/sde. Even a fast USB stick can speed up de-duplication because it has better random access performance than normal hard disks. Use SSDs for best performance.

First setup the server such that the temporary directory (/tmp) is on a sufficiently large performant file system. If you have a raid setup you could set /tmp to be on a striped device. We will now create a backup storage file system in /media/BACKUP.
Create a ZFS-pool 'backup' from the two hard drives. The two are mirrored. Put a hard drive of the same size into the hot swappable hard drive slot. We will mirror it as well:

```
zpool create backup /dev/sdb /dev/sdc /dev/sdd cache /dev/sde -m /media/BACKUP
```

Enable de-duplication and compression. You do not need to set a quota as de-duplication fragments everything anyway (that's why we need the caching device).

```
zfs set dedup=on backup
zfs set compression=on backup
```

Now we want to implement a grandfather, father, son or similar backup scheme where we can put hard disks in a fireproof safe. So each time we want to have an off-site backup we remove the hot swappable device and plug in a new one. Then we either run

```
zpool replace backup /dev/sdd /dev/sdd
```

or

```
zpool scrub
```

You can see the progress of the re-silvering/scrub with 'zpool status'. Once it is done you are ready to take another hard disk somewhere.

Now we want to save the backups on a server on another location. First we create the ZFS backup pool on this other location.
Then we transfer the full file system (otherserver is the host name of the other server):

```
zfs snapshot backup@last
zfs send backup@last | ssh -l root otherserver zfs recv backup@last
```

Once this is done we can sync the two file systems incrementally:

```
zfs snapshot backup@now
ssh -l root otherserver zfs rollback -r backup@last
zfs send -i backup@last backup@now | ssh -l root otherserver zfs recv backup@now
zfs destroy backup@last
zfs rename backup@last backup@now
ssh -l root otherserver zfs destory backup@last
ssh -l root otherserver zfs rename backup@last backup@now
```

You can also save these full and incremental zfs streams into files on the other server and not directly into a ZFS file system.

10.7.2 Btrfs

Btrfs is an advanced file system for Linux capable of creating copy on write snapshots of sub-volumes. Currently, as of Linux kernel 4.1.6, btrfs is still unstable. During testing users of UrBackup ran into performance problems, were unable to delete files or had kernel lock-ups. It is advised that you think twice before using btrfs as storage backend, even though it does have considerable advantages compared to other file systems. For UrBackup to be able to use the snapshotting mechanism the Linux kernel must be at least 3.6.

If UrBackup detects a btrfs file system it uses a special snaphotting file backup mode. It saves every file backup of every client in a separate btrfs sub-volume. When creating an incremental file backup UrBackup then creates a snapshot of the last file backup and removes, adds and changes only the files required to update the snapshot. This is much faster than the normal method, where UrBackup links (hard link) every file in the new incremental file backups to the file in the last one. It also uses less metadata (information about files, i.e., directory entries). If a new/changed file is detected as the same as a file of another client or the same as in another backup, UrBackup uses cross device reflinks to save the data in this file only once on the file system. Using btrfs also allows UrBackup to backup files changed between incremental backups in a way that only changed data in the file is stored. This greatly decreases the storage amount needed for backups, especially for large database files (such as e.g. the Outlook archive file). The ZFS deduplication in the previous section (10.7.1) saves even more storage, but comes at a much greater cost in form of a massive decrease of read and write performance and high CPU and memory requirements.

In order to create and remove btrfs snapshots UrBackup installs a setuid executable *urbackup_snapshot_helper*. UrBackup also uses this tool to test if cross-device reflinks are possible. Only if UrBackup can create cross-device reflinks and is able to create and destroy btrfs snapshots, is the btrfs mode enabled. *urbackup_snapshot_helper* needs to be told separately where the UrBackup backup folder is. This path is read from */etc/urbackup/backupfolder*. Thus, if */media/backup/urbackup* is the folder where UrBackup is saving the paths, following commands would properly create this file:

```
mkdir /etc/urbackup
echo "/media/backup/urbackup" > /etc/urbackup/backupfolder
```

You can then test if UrBackup will use the btrfs features via running

```
urbackup_snapshot_helper test
```

If the test fails, you need to check if the kernel is new enough and that the backup folder is on a btrfs volume.

You should then be able to enjoy much faster incremental file backups which use less storage space.